STIBO SYSTEMS
MASTER DATA MANAGEMENT

USER GUIDE

Data Exchange

Release 2023.4 – December 2023

Table of Contents

Table of Contents **2**

Data Exchange **29**

Inbound Tools 29

Data Format 29

Receiver Method 29

Outbound Tools 29

Data Format 30

Delivery Method 30

Security 30

Gateway Tool 30

Web Service Tools 30

Data Formats **30**

Advanced STEPXML Format **34**

Schema 34

Format Availability 34

Mapping 35

Inbound Data 35

Outbound Data 38

Alphabetical Index - XML Format **41**

Format Availability 41

Mapping 41

Outbound Data 41

Ariba CIF Format **43**

Format Availability 43

Mapping 43

Outbound Data 43

Ariba CIF 3.0 Format **45**

Format Availability 45

Mapping 45

Outbound Data 45

BMEcat Format **47**

Format Availability 47

Mapping 47

Inbound Data 49

Outbound Data 50

BMEcat 2005 Format **53**

Format Availability 53

Mapping 53

Inbound Data 55

Outbound Data 56

BMEcat 2005 Component Model Configuration **61**

BMEcat 2005 for ETIM Data Configuration **79**

Prerequisites 79

Configuring the Export 79

BMEcat 2005.1 Format **81**

CSV Format **82**

Format Availability 82

Mapping 82

Inbound Data 83

Outbound Data 85

Exporting Data via JDBC with CSV Format **89**

Install Required Drivers 89
Configure Properties 89
Select Format 90
Map Data 90
Select Delivery Method 92
cXML Format 94
Format Availability 94
Mapping 94
Outbound Data 94
ECLASS Format 96
Format Availability 97
Multiple Versions 97
Finding an Attribute 98
Inbound Data 99
ECLASS Advanced in the Web UI 100
ECLASS Classification Import 101
Import Report Messages 104
Multi and Single Value Attribute Options 104
Dimension Dependency for LOVs and Attributes 105
LOV Value Filters on ECLASS Attribute-Classification Links 106
LOVs' Usage of Value IDs 107
ETIM Format 109
Format Availability 109
Mapping 109
ETIM Configuration 110
ETIM Versions 110
ETIM Classification 111
ETIM Feature Types 113
Valid Values are Determined by ETIM Version 113
Valid Units are Determined by Classification to Attribute Link 114
Handling Import Errors 115
ETIM and ETIM v2 Format 116
Format Availability 116
Mapping 116
Inbound Data 116
Inbound Parameters 116
Import Manager 117
IIEP 118
ETIM IXF Format 119
Format Availability 119
Mapping 119
Inbound Data 119
Create Product to ETIM Classification Link 123
Set Validity on ETIM Attributes for Products 126
Manually Import ETIM Attribute Values 131
Excel Format 142
Format Availability 143
Mapping 143
Inbound Data 143
Outbound Data 146
Date and Number Formatting with Excel 149

Importing Calculated Attribute Value Templates with Excel **151**

Importing Multi-Valued References or Links with Excel **155**

Managing Attribute Parameters with Excel **164**

Excel Custom Template **167**

Format Availability 167

Mapping 167

Output 167

Excel Custom Template Configuration **169**

Configuration 169

Image conversion configurations with Excel Custom Template 177

Excel List of Values **179**

Mapping 179

Exporting List of Values 179

Excel Smartsheet Format **181**

Format Availability 181

Setup Requirements 181

Additional Information 182

Trusted Folder Setup for Smartsheets 182

Initiating Products Using a Smartsheet **186**

Prerequisites 186

Product Initiation 186

Exporting a Template with a Supplier Selector 191

Maintaining Products Using a Smartsheet **195**

Prerequisites 195

Product Maintenance 195

Classification Roots for Inherited Attributes **198**

Classification Roots for Inherited Attributes Use Cases **201**

Inheritance Scenario 201

Smartsheet and LOVs **206**

Limitations 206

Smartsheet Data and Template Configurations **207**

Select Objects 207

Select Format 208

Map Data 214

Special options for multi-object type Smartsheets 214

Special options for multi-level Smartsheets 214

Additional Information About Mapping 216

Advanced 216

Select Delivery Method 216

Saving and Modifying the Configuration 216

Smartsheet Import Configurations **217**

Creating an Import Configuration 217

Select Format Step 217

Advanced Settings Step 217

Saving and Modifying the Configuration 219

Using a Smartsheet **220**

Entering Values for Multi-Valued LOVs and References 220

Validating the Sheet 222

Copying Data in a Smartsheet Template ...224

Smartsheet Types ... 226

Multi-level Smartsheets ... 226

Example ... 227

Multiple Product Object Type Smartsheets 228

Open Format Smartsheet ... 230

Export Manager Configuration ... 230

Default Cell Formats for Open Format Smartsheets ... 231

Working with Formulas in Open Format Smartsheets ... 232

Validation of Formulas ... 232

Additional Considerations for Open Format Smartsheets ... 232

Handling of Cells with Number Formats ... 233

Read-Only Cells ... 234

Background Shading and Text Formatting ... 234

Excel Conditional Formatting ... 235

Copy and Paste Behavior for Rows and Columns ... 235

Additional Information About Smartsheets ... 236

System Configuration ... 236

Smartsheet Cover Tab ... 236

Smartsheet Conditional Attribute Handling 238

Smartsheet Tool Tips for Column Headers 238

Mandatory Attributes and References in Smartsheets ... 240

Visual Indication of Mandatory in Smartsheets ... 240

Configure Standard Mandatory Settings in Smartsheets ... 242

Configure Workflow State Mandatory in Smartsheets ... 244

Conditional Attribute Handling in Smartsheets ... 244

Mandatory Handling in Smartsheets ... 244

Conditionally Mandatory Handling in Smartsheets ... 245

Configure Metadata Attribute for Mandatory in Smartsheets ... 247

Mandatory Attributes in Smartsheets ... 247

Mandatory References in Smartsheets ... 248

Smartsheet Background Processes and Queues ... 250

Using Smartsheet Import Background Process ... 250

Using Smartsheet Export Background Process ... 251

Smartsheet Multi-Supplier View ... 253

Smartsheet System Properties ... 254

Limit Supplier Browsing ... 254

Service URL Override ... 254

Trusted Headers Authentication ... 255

Increasing Default of Preformatted Template Rows ... 255

Use Decimal Separators from Native Excel Client ... 255

Smartsheet Template Product Row ... 257

Prerequisites ... 257

Configuration ... 258

'Example' Product ... 258

Smartsheet Template Product Reference 258
Example Setup 259
FAB-DIS Format 263
Format Availability 263
Mapping 263
Inbound Data 263
Outbound Data 264
FAB-DIS Export Format 266
FAB-DIS ETIM Export Mapping 267
FAB-DIS Media Export Mapping 270
FAB-DIS Import Format 273
FAB-DIS Import Workbench Configuration 276
Prerequisites 276
Asset Importer Configuration 276
IIEP Configuration 278
FAB-DIS Import Mapping 282
IIEP Mapping 282
Automatic Mapping 285
Media Mapping via Asset Importer Configuration 286
FAB-DIS Import Web UI Configuration .. 287
FAB-DIS Import Background Processes 288
Troubleshooting 288
FAB-DIS Import Example 290
ETIM Tab 290
MediaTab 292
FixedWidth Format 296
Format Availability 296
Mapping 296
Inbound Data 296
FixedWidth Format Configuration Examples 299
Example Data 299
End Positions (Zero Based) Example 299
End Positions (One Based) Example 301
Width Example 303
Flatplan Excel Format 305
Format Availability 305
Mapping 305
Inbound Data 305
Outbound Data 306
Generic JSON Format 308
Format Availability 308
Mapping 308
Inbound Data 308
Outbound Data 310
Generic JSON Inbound Processing Instructions 314
Instruction Definition in Generic JSON .. 315
Record Processing Instruction in Generic JSON 317
Source Processing Instructions in Generic JSON 318
Preventing Duplicate Field Names 319
MultiSource Processing Instructions in Generic JSON 320

SourceID Processing Instructions in Generic JSON 322

Repeated Processing Instructions in Generic JSON 325

DimensionPointID Processing Instructions in Generic JSON 329

Generic JSON Import Simple Example ... 331

Input Document 331

Template 331

Conversion Preview 331

Generic JSON Import References and Links Example 333

Input Document 333

Template 333

Conversion Preview 333

Generic JSON Import Data Containers Example 337

Input Document 337

Template 338

Conversion Preview 339

Generic JSON Import Advanced Example 343

Input Document 343

Creating a Template 344

Remove Redundant Elements 344

Add Record and Source Instructions 345

Add Repeated Instruction 347

Add Repeated Instruction With a Filter .. 348

Add MultiSource Instruction 350

Conversion Preview 351

Generic JSON Outbound Processing Instructions 353

Template, Mapping, and Output 353

Generic JSON Configuration Examples 356

Mapping Attributes Example 356

Mapping Data Container Example 361

Mapping Data Path Example 364

Generic XML Format 368

Limitations 368

Format Availability 368

Mapping 368

Inbound Data 368

Outbound Data 370

Generic XML Inbound Processing Instructions 373

DimensionPointID Processing Instructions in Generic XML 374

MultiSource Processing Instructions in Generic XML 376

Record Processing Instruction in Generic XML 378

Repeated Processing Instructions in Generic XML 379

Source Processing Instructions in Generic XML 382

Preventing Duplicate Field Names 382

SourceID Processing Instructions in Generic XML 383

Generic XML Import Simple Example 386

Input Document 386

Template 386

Conversion Preview 386

Generic XML Import Data Containers Example 387

Input Document 387

Template 387

Conversion Preview 388

Generic XML Import Advanced Example 392

Input Document 392

Creating a Template 393

Remove Redundant Elements 393

Add Record and Source Instructions 394

Add Repeated Instruction 395

Add Repeated Instruction With a Filter 397

Add MultiSource Instruction 398

Conversion Preview 399

Generic XML Outbound Processing Instructions 401

Allow Empty Tags Parameter in Generic XML 402

Attributes Processing Instructions in Generic XML 406

Attribute Aspects in Generic XML 411

Attribute Links and LOVs Processing Instructions in Generic XML 414

Mapping Attribute Links, LOVs, and LOV Values in Generic XML 419

Attribute Links and Units Processing Instructions in Generic XML 424

Mapping Attribute Links, Units, and Unit Base in Generic XML 428

DoNotEmit Processing Instructions in Generic XML 432

ExportDate Processing Instructions in Generic XML 436

List of Values Object Type in Generic XML 438

Multi-Valued Objects in Generic XML 442

Multi-Cardinality Tags in Generic XML 445

Repeated Targets 446

MultiTarget 447

Composite Mapping Targets 448

Nestable Data Paths in Generic XML 451

Data Path Concept 451

Nesting Concept 452

Valid Path Step Types 452

Data Sources Available for Data Paths 452

Nestable Data Paths in Generic XML Example 454

Data Path Configuration 455

Mapping for Nestable Data Paths in Generic XML Example 458

Nestable Data Paths in Generic XML and Performance 466

RemoveIfEmpty Processing Instructions in Generic XML 467

Reporting Deleted Assets with a Generic XML OIEP 471

Object Type and Event Type Selection 471

Template 472

Mapping 473

Results 473

Reporting Deleted Classifications with a Generic XML OIEP 475
Object Type and Event Type Selection ..475
Template 476
Mapping 477
Result 477
Reporting Deleted Entities with a Generic XML OIEP 478
Object Type and Event Type Selection ..478
Template 479
Mapping 480
Results 480
Reporting Deleted Product References with a Generic XML OIEP 481
Product 481
Object Type and Event Type Selection ..481
Template 482
Mapping 483
Results 483
Mapping to Report Deleted Product References with a Generic XML OIEP 485
Single-Cardinality Tags in Generic XML .493
Record 495
Target Default 496
Target Override Default 496
Units with a Generic XML OIEP 497
Object Type and Event Type Selection ..497
Template 498
Mapping 498
Results 499
Mapping Unit Aspects with a Generic XML OIEP 500
Generic XML Sample Export 502
IDoc MATMAS 05 Format 504
Format Availability 504
Mapping 504
Inbound Data 504
Outbound Data 506
Publication Excel Format 509
Format Availability 509
Mapping 509
Inbound Data 509
Outbound Data 510
SmartLabel Format 512
Format Availability 512
Mapping 512
Outbound Data 512
Configuring the SmartLabel Template ... 515
SmartLabel Attributes 515
Modifying the SmartLabel Template 516
STEPXML Format 519
Limitations 519
Format Availability 519
Mapping 520
Inbound Data 520
Outbound Data 522
STEPXML Import and Export Options 524

STEPXML Outbound Parameters **532**
Global Settings 532
Data Objects 534
Configuration 537
Publishing 547
STEPXML Default Data Outbound **549**
Product 549
Entity 551
Classification 552
Asset 552
STEPXML Tags and Examples **554**
XSD (XML Schema Definition) 554
Tags for Common Data 554
Examples 555
Assets Tag in STEPXML **556**
Name 556
Values 556
AssetContent Tag in STEPXML **559**
Asset Content Templates 559
Asset Content Output 560
AssetPushConfiguration Tag in STEPXML **563**
AttributeLink Tag in STEPXML **564**
IncludeInherited 564
AutoApprove in STEPXML **566**
Backward Compatibility in STEPXML **567**
Classifications Tag in STEPXML **569**
IncludeParent 569
Additional STEPXML Tags 569
Name 570
Values 570
ClassificationReference 570
MetaData 570
Product to Classification Links Owned by the Classification 570
Context Data (Qualifiers) in STEPXML **571**
Create Objects in STEPXML **573**
Cross Context Inheritance in STEPXML **575**
Data Containers in STEPXML **580**
Setting Up Data Containers 580
Adding and Maintaining Data Containers 581
<KeyDefinition> Tag for Data Container Keys 583
Delete Objects in STEPXML **586**
Event-Based OIEP Triggered by Deleting Products, Classifications, and Assets 586
Delete Product Reference in STEPXML **588**
Delete Values in STEPXML **592**
Export Inheritance in STEPXML Example **593**
Filter Data Containers in STEPXML **595**
Known Limitation 596
Filter Products in STEPXML **597**
Filter References in STEPXML **601**
Asset References 601
Classification References 601
Product References 601

Results 602

Filter Values in STEPXML 604

Inherit References in STEPXML 605

Inherit Values in STEPXML 608

IncludeInherited 608

IncludeInheritedAttributes 608

Considerations 608

Examples 610

Inherit Values: Selected Only Example 611

Inherit Values: Selected and Children Example 613

Inherit Values: Exclude Selected, All Local on Children, Inherited on Children by Object Type Example 615

Inherit Values: Local on Selected, All on Children by Object Type Example 617

Inherit Values: Exclude Selected, All on Children Example 619

Inherit Values: Local on Selected, Inherited on Children Example 621

ImageConversionConfiguration Tag in STEPXML 623

IncludeKey Tag in STEPXML 624

Minimum, Referenced, and Selected in STEPXML 626

Hierarchy Export Including Objects Referenced Example 627

Product Export Including Definitions, LOVs, and Units for Attributes Example 629

Products Tag in STEPXML 631

ExportSize 631

IncludeParent 631

Additional STEPXML Tags 631

Name 631

Values 632

AssetCrossReference 632

ProductCrossReference 632

ClassificationReference 632

Product Nesting Example 632

Nested structure 632

Non-nested structure 633

Product with Units, References, and Multi-Values Attributes Example 633

Product to Classification Links Owned by the Classification 634

ReplacementRules Tag in STEPXML 635

Recommended Process 635

ReplacementRules Tag Example 636

ReplacementRules Tag Structure 637

ReplacementRules in STEPXML for Assets Examples 646

Remove Asset Values 646

ReplacementRules in STEPXML for Attributes Examples 648

Remove Attribute Group Links 648

Replace Unit Links 649

ReplacementRules in STEPXML for Classifications Examples 651

Remove Values On Classification Folders 651

Replace Product References 653

ReplacementRules in STEPXML for Entities Examples 655

Replace Cross References 655
Replace Data Containers 656
ReplacementRules in STEPXML for Products Examples 658
Remove References 658
Remove Unnecessary Attribute Links 659
Replace Attribute Values by Context 660
Update Override Sub Products 662
Replace Classification References 664
ReplacementRules in STEPXML for User Groups Examples 666
Update Privilege Rules for a User Group 666
ReplacementRules in STEPXML for User Type Links Examples 668
Replace Links on Attribute Object Types 668
Replace Reference Source and Target for Link Types 669
Replace Links on Object Types 671
Referenced and Embedded XML Attributes in STEPXML 673
SequenceProduct Tag in STEPXML 678
SingleUpdateMode in STEPXML 682
STEP-ProductInformation Tag in STEPXML 683
DefinitionsAsComments 683
EventsAsComments 683
ExportDeletedData 684
ExportDerivedAttrs 684
ExportValidAndAbsent 684
FollowOverrideSubProducts 684
ResolveInlineRefs 685
STEPXML Configuration Export Format 686
Format Availability 686
Mapping 686
Outbound Data 686
Output Template 689
UNSPSC Format 692
Format Availability 692
Mapping 692
Inbound Data 692
xCBL Format 696
Format Availability 696
Mapping 696
Outbound Data 696
Data Mapping 699
Inbound Data Mapping 699
Outbound Data Mapping 699
Inbound Map Data Options 700
Formats 700
Map To Object Type 701
Mapping Data 701
Modifying Data 701
Profiling Data 702
Tips for Mapping Data 702
Inbound Map Data - Auto Map 703
Predefined Header Text 703
Auto Map Example 705

Inbound Map Data - Constant **707**
Mapping a Constant 707
Inbound Map Data - Generate Profile **711**
Profiling Setup 711
Profiling Data 711
On Demand Data Validation 715
Inbound Map Data - Map **717**
Mapping the Same Column Multiple Times 717
Mapping Options 718
Add Child to Override - Map Inbound **720**
Asset Reference - Map Inbound **723**
Attribute - Map Inbound **730**
Attribute Info - Map Inbound **735**
Attribute Info File Structure Requirements 737
Attribute Validation Info - Map Inbound **740**
Attribute Validation Info File Structure Requirements 742
Classification Reference - Map Inbound **745**
Data Container - Map Inbound **748**
Data Container ID - Map Inbound **752**
Data Container Type ID - Map Inbound **753**
Data Type - Map Inbound **762**
Delete Values - Map Inbound **763**
Entity Reference - Map Inbound **768**
Entity Reference via Source Record ID - Map Inbound **773**
Prerequisites 773
Map Entity References 773
ID or Key - Map Inbound **777**
Mapping ID 777
Mapping Key 780
Multivalued Variable - Map Inbound **782**
Name - Map Inbound **786**
Object Type - Map Inbound **788**
Map to Create New Items with Object Type 789
Map to Change the Existing Object Type 790
Overrides Product - Map Inbound **793**
Parent - Map Inbound **796**
Product Classification Links - Map Inbound **798**
Product Reference - Map Inbound **801**
Reference Meta-Data - Map Inbound **804**
Reference Target - Map Inbound **809**
Reference Type ID - Map Inbound **810**
Variable - Map Inbound **817**
Inbound Map Data - Map To **821**
Mapping Based on Object Type 821
Asset Object Type - Map To Inbound **822**
Attribute Object Type - Map To Inbound **823**
Classification Object Type - Map To Inbound **824**
Entity Object Type - Map To Inbound **826**
Product Object Type - Map To Inbound **828**
Inbound Map Data - Remove **830**
Remove a Mapped Column 830
Inbound Map Data - Transform **832**

Transforming Data 832
Dimension Dependent Data 833
Outbound Map Data Options 834
Formats 834
Mapping in Outbound Tools 834
Mapping 836
Transformations 836
Outbound Map Data - Data Source 837
Available Data Source Options 837
Asset References - Data Source Outbound 840
Attribute Information - Data Source Outbound 844
Attributes (and Data Containers) - Data Source Outbound 847
Attributes Inherited from Classification - Data Source Outbound 851
Business Functions - Data Source Outbound 855
Classification References – Data Source Outbound 864
Classifications - Data Source Outbound 866
Constant Value - Data Source Outbound 868
Custom Attributes - Data Source Outbound 872
Data Container Type ID - Data Source Outbound 873
Data Owner Node - Data Source Outbound 874
Data Path - Data Source Outbound 875
Data Type - Data Source Outbound 876
Entity References – Data Source Outbound 877
ID or Key - Data Source Outbound 879
Index Words – Data Source Outbound 882
Mapping Index Words Linked to Products Using Word 883
Mapping Index Words Linked to Products Using Sort Word 884
Mapping Index Words from Classification 885
Inherit Data and References - Data Source Outbound 887
Insert Data Containers - Data Source Outbound 890
Insert Referenced Objects - Data Source Outbound 895
Insert References - Data Source Outbound 898
Is Deleted - Data Source Outbound 904
Multi Level Parent Attributes - Data Source Outbound 905
Multi Level References - Data Source Outbound 909
Mapping Multi Level References 910
Multi Level References - Examples with Multi-Value Reference Types 914
Multi Level References - Examples with Single Value Reference Types 918
Multi Level References - Examples with Entities 920
Multi Level References - Examples with Products 923
Name - Data Source Outbound 927
Object Type Name - Data Source 929

Outbound .....

Page Number – Data Source Outbound . 931

Parent ID - Data Source Outbound 932

Product Classification Links – Data Source Outbound 934

Product-Override Child ID – Data Source Outbound 936

Product References - Data Source Outbound 939

Reference Type ID - Data Source Outbound 942

STEP Workflow Task Info - Data Source Outbound 943

System Setup - Data Source Outbound .. 950

Outbound Map Data - Mapping Targets .. 951

Tabular Format Targets 951

XML Format Targets 952

JSON Format Targets 953

Outbound Map Data - Transform 954

Transforming Data 954

Data Container Separator - Transform Outbound 956

Multi-Value Separator - Transform Outbound 959

Aspect - Transform Outbound 962

Attribute Meta Data Aspect 968

Attribute Type Aspect 970

Base Unit ID, Name, and Value Aspects . 972

Closest Attribute Link Meta Data Aspect 976

Closest Attribute Link Meta Data Aspect and Display Sequence 978

Dimension Dependencies Aspects 981

Is Deleted Aspect 984

Key Aspect 988

LOV ID Aspect 991

LOV Value-ID Aspect 994

LOV Value-ID or Value Aspect 997

Unit Aspect 1002

Unit ID Aspect 1005

Value and Unit Aspect 1008

Value and Unit Meta Data Aspect 1010

Value Aspect 1013

Transformation Examples 1015

Concatenate Column with Unit Data Example 1016

Concatenate Data Using Variables Example 1020

Concatenate into Single-Valued Data Example 1025

Dimension Point for Single Dimension Data Example 1028

LOV Value-ID or Value Aspect 1032

Extract Characters at Position Example 1037

Extract Characters Between and Extract Characters After Example 1040

Format Number Example 1045

Header and Value Aspects Example 1048

If Equals Source Example 1054

Insert Text Example 1056

Math Operation Example 1060

Merge Attributes and Constant Example **1064**

Merge Data for Multi-Valued Data Example **1067**

Replace Substrings of the Value Example **1081**

Split and Extract Data Example **1085**

Replace the Whole Value Example **1092**

Transform Date by Locale or by Pattern **1098**

Export Transformations 1099

Import Transformations 1102

Transform Using Attribute Transformation Example **1106**

Transformation Lookup Table Example **1108**

Export Manager **1112**

Setup Requirements 1112

Additional Information 1112

Creating a Data Export **1113**

Considerations for Exporting Products ... 1116

Export Manager - Select Configuration **1118**

Export Manager - Select Objects **1120**

Export Manager - Select Format **1123**

Export Manager - Map Data **1125**

Export Manager - Advanced **1126**

Export Manager - Select Delivery Method **1129**

Delivery Methods 1129

Azure Blob Storage Delivery Method ... **1131**

Cloud Blob Storage Delivery Method ... **1134**

Configuring the Export Manager Delivery 1134

Email Delivery Method **1137**

File Delivery Method **1139**

FTP Delivery Method **1141**

JDBC Delivery Method **1143**

Server Side Delivery Method **1146**

SFTP Delivery Method **1148**

WebSphere Commerce Import Delivery Method **1151**

Running a Data Export **1152**

Maintaining a Saved Export Configuration **1154**

View a Saved Export Configuration 1154

Edit a Saved Export Configuration 1154

Monitoring a Data Export **1157**

Monitoring the Export Process 1157

Execution Report Flipper 1158

Result Flipper 1160

Scheduling a Data Export **1161**

Scheduled Data Export - Select Export Configuration **1163**

Scheduled Data Export - Schedule Process **1164**

Scheduled Data Export - Name Process **1165**

Gateway Integration Endpoints **1166**

Setup Requirements 1166

Additional Information 1166

Initial Setup for a Gateway Integration Endpoint **1167**

Create the Gateway Setup Group 1167

Link the Gateway Object Type to Setup Group 1168
Create an Instance of the Gateway Endpoint Object 1169
Creating a Gateway Integration Endpoint 1171
Configuring a Gateway Integration Endpoint 1172
Configuring a Gateway Integration Endpoint - Amazon S3 Blob Storage 1174
Prerequisites 1174
Configuring the Gateway Integration Endpoint 1175
Using the Gateway Integration Endpoint 1176
Configuring a Gateway Integration Endpoint - Google Cloud Storage 1178
Prerequisites 1178
Configuring the Gateway Integration Endpoint 1179
Using the Gateway Integration Endpoint 1180
Configuring a Gateway Integration Endpoint - Microsoft Azure Blob Storage 1182
Prerequisites 1182
Configuring the Gateway Integration Endpoint 1183
Using the Gateway Integration Endpoint 1185
Configuring a Gateway Integration Endpoint - REST 1186
Authentication 1188
Basic Authentication 1188
Token-Based Authentication 1189
Proxy Configuration Setup 1190
Running a Gateway Integration Endpoint 1192
Maintaining a Gateway Integration Endpoint 1194
Monitoring a Gateway Integration Endpoint 1195
Monitor from STEP 1195
Monitor Outside of STEP 1195
View all endpoints 1195
View a specific gateway endpoint 1195
Gateway Integration Endpoint Statistics 1197
Import Manager 1198
Setup Requirements 1198
Additional Information 1198
Creating a Data Import 1199
Import Manager - Select Configuration 1201
Import Manager - Select Data Source 1203
Using FTP 1203
Using File 1204
Import Manager - Select Format 1205
Import Manager - Map Data 1207
Import Manager - Identify Objects 1208
Importing With Object IDs 1208
Importing with Unique Key 1209
Importing Without Object IDs 1209
Import Manager - Identify Destination 1212
Import Manager - Select Business Rules 1214

Business Rule Limitations 1216

Import Manager - Advanced Settings ... 1218

General Advanced Settings 1218

Remove Options 1220

Adding and Removing Un-Mapped References 1226

Running a Data Import 1228

Maintaining a Saved Import Configuration 1230

View a Saved Import Configuration 1230

Edit a Saved Import Configuration 1230

Monitoring a Data Import 1232

Monitoring the Import Process 1232

Properties Flipper 1233

Execution Report 1234

Reimporting a Corrected File 1235

Import Error Messages 1236

Import Error Message Examples 1240

Illegal Unit for Attribute 1240

Unknown Parent Product 1240

Maximum Length, Illegal Value 1241

Out of Range, Illegal Value 1242

Not a Number, Illegal Value 1242

Object Existed with Another Object Type .1243

Unknown Classification 1244

Not Privileged to Create Object 1244

Unknown Asset ID 1245

Array Operation Failed 1245

Optimistic Locking Errors 1245

The Transaction has been Rolled Back: Unique Constraint 1245

Inbound Integration Endpoints 1251

Setup Requirements 1251

Additional Information 1251

Initial Setup for an Inbound Integration Endpoint 1253

Create the Inbound Integration Setup Group 1254

Link the IIEP Object Type to Setup Group 1255

Create an Instance of the IIEP Object 1256

Creating an Inbound Integration Endpoint 1258

IIEP - Identify Endpoint 1260

IIEP - Choose Receiver 1261

Receiver Methods 1261

Amazon SQS Receiver 1264

Dynamic JMS Receiver 1266

Dynamic JMS Configuration Examples ...1267

Azure Message Bus 1267

TibcoJMS 1269

Apache Active MQ 5.15.8 1269

Apache RabbitMQ 1.11.1 1270

WebLogic JMS Connection 1272

Hotfolder Receiver 1273

Hotfolder Receiver Using File Sequence 1275

Hotfolder Receiver Using Meta Files 1277

IBM MQ SSL Receiver 1279

JMS Receiver **1283**
Configuration Examples 1285
FileInitialContextFactory JMS Receiver 1285
JMS WebSphere Delivery Using SSL ..1285
Apache ActiveMQ JMS Receiver 1286
Kafka Receiver **1287**
Execution Report 1290
Considerations for Setting Up Kafka Receiver or Delivery **1292**
Compatibility 1292
Message Size Limitation 1292
Option 1 1292
Option 2 1294
Option 3 1295
Topic Partitions 1295
Event Message Serialization 1296
Reference Handling for Dangling References 1296
Kafka Topic Offset 1299
Static Membership 1299
SASL Authentication 1299
Oracle AQ Receiver **1301**
Product Data Exchange Receiver **1303**
Token-based Authentication Function Example 1304
REST Receiver **1308**
Web UI File Loading Receiver **1310**
IIEP - Configure Endpoint **1311**
IIEP - Configure Pre-processor **1314**
IIEP - Configure Transformation by XSLT Pre-processor **1315**
Initial Setup for XSLT Stylesheets 1315
Create an XSLT Stylesheet Object 1319
Configure the Transformation by XSLT Pre-processor 1321
IIEP - Configure XML Normalizer Pre-processor **1323**
IIEP - Configure Processing Engine **1329**
IIEP - Configure Asset Importer Processing Engine **1330**
Metadata File Format 1335
IIEP - Configure Business Rule Based Message Processor Processing Engine **1336**
Splitter Business Action Details and Examples 1338
JSON Message Splitter with JavaScript Business Action Example 1338
CSV Message Splitter with JavaScript Business Action Example 1339
Importer Business Action Details and Examples 1340
JSON Message Importer with JavaScript Business Action Example ...1340
IIEP - Configure FAB-DIS Importer Processing Engine **1343**
IIEP - Configure STEP Importer Processing Engine **1344**
IIEP - Configure Match and Merge Importer Processing Engine **1347**
Match and Merge Importing Process and Logic 1347
Match and Merge IIEP Configuration 1347

Source Record ID in STEPXML 1350

IIEP - Configure Post-processor 1352

IIEP - Schedule Endpoint 1353

IIEP - Error Handling & Reporting 1356

Running an Inbound Integration Endpoint 1359

Enable and Invoke an Inbound Integration Endpoint 1359

Disable an Inbound Integration Endpoint 1362

Maintaining an Inbound Integration Endpoint 1363

Inbound Integration Endpoint Tab 1363

Description Flipper 1363

Configuration Flipper 1364

Receiver Configuration Flipper 1365

Integration Endpoint Log Flipper 1366

Background Processes Tab 1366

Statistics Tab 1367

Error Log Excerpts Tab 1367

Log Tab 1368

Status Tab 1368

Monitoring an IIEP via Background Process 1369

Background Process Priority Settings 1369

Background Legacy Process Queue Settings 1370

Legacy Background Process Queue 1370

Legacy Background Process Queue Size 1370

Monitoring an IIEP via External System 1371

Performance Data 1371

Monitoring Sensors 1371

REST API Monitoring 1372

Import Error Messages 1373

Import Error Message Examples 1377

Illegal Unit for Attribute 1377

Unknown Parent Product 1377

Maximum Length, Illegal Value 1378

Out of Range, Illegal Value 1379

Not a Number, Illegal Value 1379

Object Existed with Another Object Type 1380

Unknown Classification 1381

Not Privileged to Create Object 1381

Unknown Asset ID 1382

Array Operation Failed 1382

Optimistic Locking Errors 1382

The Transaction has been Rolled Back: Unique Constraint 1382

Additional Information About Inbound Integration Endpoints 1388

Handling Failed IIEP Background Processes 1389

Disable and Enable a Background Process 1389

Resume a Background Process 1389

Resolving a Failed Background Process 1390

Inbound Integration Endpoint Structure 1393

Receiver 1393

Pre-processor 1394

Processing Engine / Converter 1394
Error Reporter 1394
Post-processor 1394
Integration Endpoint Transactional Settings 1395
Transactional Settings Options 1397
Reference Target Lock Policy on Object Types 1400
Strict or Relaxed 1400
Outbound Integration Endpoints 1402
Event-Based Outbound Integration Endpoint 1402
Setup Requirements for Event-Based OIEPs 1402
Additional Information for Event-Based OIEPs 1403
Select Objects Outbound Integration Endpoint 1403
Setup Requirements for Select Objects OIEPs 1403
Additional Information for Select Objects OIEPs 1404
Initial Setup for an Outbound Integration Endpoint 1405
Create the Outbound Integration Setup Group 1405
Link the OIEP Object Type to Setup Group 1407
Create an Instance of the OIEP Object 1408
Creating an Outbound Integration Endpoint 1410
Creating an Event-Based Outbound Integration Endpoint 1411
OIEP - Event Based - Identify Endpoint 1413
OIEP - Event-Based - Choose Data Source 1414
OIEP - Event Based - Configure Endpoint 1415
OIEP - Event-Based - Manual Configuration 1417
OIEP - Configuration Flipper 1424
Process Engine 1424
Error Handling & Reporting 1424
Schedule 1427
Priority 1430
Legacy Queue for Endpoint 1430
Legacy Queue for Endpoint Processes 1430
Transactional Settings 1431
Maximum Number of Threads 1431
Maximum Number of Waiting Processes 1431
Maximum Number of Old Processes 1431
Maximum Age of Old Processes 1431
Context Mode and Contexts 1431
Workspace 1433
OIEP - Configuration Flipper for Business Rule Based Message Processor 1434
Node Handler Business Action Details and Examples 1436
JSON Message Node Handler JavaScript Business Action Example 1437
Joiner Business Action Details and Examples 1438
JSON Message Joiner JavaScript 1439

Business Action Example

OIEP - Event-Based - Output Templates Flipper 1440

Configure the Object Types and Event Types 1440

Using Multiple OIEPs or Multiple Output Templates 1442

Multiple OIEPs 1442

Multiple Output Templates 1442

Limitations of Multiple Output Templates 1443

Configure the Format 1443

Configure the Pre-processor and Post-processor 1447

Pre-processor 1447

Post-processor 1448

OIEP - Pre-processor - Business Action 1450

Business Action Configuration 1450

OIEP Configuration 1451

OIEP - Post-processor - Context Splitter 1452

OIEP - Post-processor - Copy Context Dependent Values and References 1454

OIEP - Post-Processor - Chained Post-processor 1457

Configure the Chained Post-processor 1457

OIEP - Post-processor - Transformation by XSLT 1458

Configure the Transformation by XSLT Post-processor 1459

OIEP - Event-Based - Pre- and Post-processing Flipper 1461

OIEP - Delivery Method Flipper 1463

Delivery Methods 1463

Amazon SQS Delivery Method 1467

Azure Blob Storage Delivery Method 1473

Change Package Git Delivery Method 1476

Cloud Blob Storage Delivery Method 1477

Configuring the OIEP Delivery 1478

Copy to Directory Delivery Method 1481

Deploy Delivery Method 1484

Dynamic JMS Delivery Method 1487

Dynamic JMS Configuration Examples 1489

Azure Message Bus 1489

TibcoJMS 1491

Apache Active MQ 5.15.8 1491

RabbitMQ 1492

WebLogic JMS Connection 1494

Email Delivery Method 1495

FTP Delivery Method 1498

Git Delivery Method 1501

IBM MQ SSL Delivery Method 1502

JDBC Delivery Method 1507

JMS Delivery Method 1510

Kafka Delivery Method 1514

Considerations for Setting Up Kafka Receiver or Delivery 1518

Compatibility 1518

Message Size Limitation 1518

Option 1 1518
Option 2 1520
Option 3 1521
Topic Partitions 1521
Event Message Serialization 1522
Reference Handling for Dangling References 1522
Kafka Topic Offset 1525
Static Membership 1525
SASL Authentication 1525
Mongo Delivery Method 1526
Mongo Delivery Method Elements 1532
Default Databases and Collections 1534
STEP JSON and Mongo JSON 1535
JavaScript Triggers 1535
Bound Variables 1536
Example JavaScript to Maintain a Collection in a Classification 1536
Prerequisites for Configuring the MongoDB Adapter 1538
Configure the OIEP 1538
Configuring the JavaScript Triggers 1540
MongoDB Adapter Setup Quick Guides 1541
STEP JSON Schema 1541
Configuring Mongo Authentication Quick Guide 1541
SSL Configuration Quick Guide 1542
Mongo Delivery Method Conversion Example 1544
Values 1544
References and Links 1546
Attribute Types 1548
Data Containers 1548
Reference Types 1550
Asset Push Locations 1550
Tables 1550
Table Formatting 1551
Attribute Links 1552
Data Container Type Links 1552
Data Containers Types 1553
Unit 1553
List Of Values 1554
Oracle AQ Delivery Method 1555
Product Data Exchange 2 Delivery Method 1557
REST Delivery Method 1560
REST Direct Delivery Method 1562
Authentication 1565
Basic Authentication 1566
Token-Based Authentication 1566
Additional Request Information 1567
SFTP Delivery Method 1570
Wiki Delivery Method 1573
OIEP - Event-Based - Event Triggering Definitions Tab 1575
Triggering Object Types Flipper 1576
Triggering Attributes Flipper 1579
Triggering Table Types Flipper 1580

Reference Type Triggers Flipper 1581
Triggering Data Container Types 1582
Miscellaneous Triggers Flipper 1583
Triggering Workspace Flipper 1584
OIEP - Event-Based - Event Queue Configuration Flipper 1586
Creating a Select Objects Outbound Integration Endpoint 1588
OIEP - Select Objects - Identify Endpoint 1590
OIEP - Select Objects - Choose Data Source 1591
OIEP - Select Objects - Configure Endpoint 1593
OIEP - Select Objects - Manual Configuration 1595
OIEP - Configuration Flipper 1599
Process Engine 1599
Error Handling & Reporting 1599
Schedule 1602
Priority 1605
Legacy Queue for Endpoint 1605
Legacy Queue for Endpoint Processes 1605
Transactional Settings 1606
Maximum Number of Threads 1606
Maximum Number of Waiting Processes 1606
Maximum Number of Old Processes 1606
Maximum Age of Old Processes 1606
Context Mode and Contexts 1606
Workspace 1608
OIEP - Configuration Flipper for Business Rule Based Message Processor 1609
Node Handler Business Action Details and Examples 1611
JSON Message Node Handler JavaScript Business Action Example 1612
Joiner Business Action Details and Examples 1613
JSON Message Joiner JavaScript Business Action Example 1614
OIEP - Select Objects - Object Selection Configuration Flipper 1615
OIEP - Select Objects - Output Templates Flipper 1618
Configure the Object Types 1618
Using Multiple OIEPs or Multiple Output Templates 1620
Limitations of Multiple Output Templates 1620
Configure the Format 1620
Configure the Pre-processor and Post-processor 1624
Pre-processor 1624
Post-processor 1625
OIEP - Pre-processor - Business Action 1627
Business Action Configuration 1627
OIEP Configuration 1628
OIEP - Post-processor - Context Splitter 1629
OIEP - Post-processor - Copy Context Dependent Values and References 1631

OIEP - Post-Processor - Chained Post-processor 1634
Configure the Chained Post-processor ...1634
OIEP - Post-processor - Transformation by XSLT 1635
Configure the Transformation by XSLT Post-processor 1636
OIEP - Event-Based - Pre- and Post-processing Flipper 1638
OIEP - Delivery Method Flipper 1640
Delivery Methods 1640
Amazon SQS Delivery Method 1644
Azure Blob Storage Delivery Method ...1650
Change Package Git Delivery Method ..1653
Cloud Blob Storage Delivery Method ...1654
Configuring the OIEP Delivery 1655
Copy to Directory Delivery Method 1658
Deploy Delivery Method 1661
Dynamic JMS Delivery Method 1664
Dynamic JMS Configuration Examples ...1666
Azure Message Bus 1666
TibcoJMS 1668
Apache Active MQ 5.15.8 1668
RabbitMQ 1669
WebLogic JMS Connection 1671
Email Delivery Method 1672
FTP Delivery Method 1675
Git Delivery Method 1678
IBM MQ SSL Delivery Method 1679
JDBC Delivery Method 1684
JMS Delivery Method 1687
Kafka Delivery Method 1691
Considerations for Setting Up Kafka Receiver or Delivery 1695
Compatibility 1695
Message Size Limitation 1695
Option 1 1695
Option 2 1697
Option 3 1698
Topic Partitions 1698
Event Message Serialization 1699
Reference Handling for Dangling References 1699
Kafka Topic Offset 1702
Static Membership 1702
SASL Authentication 1702
Mongo Delivery Method 1703
Mongo Delivery Method Elements 1709
Default Databases and Collections 1711
STEP JSON and Mongo JSON 1712
JavaScript Triggers 1712
Bound Variables 1713
Example JavaScript to Maintain a Collection in a Classification 1713
Prerequisites for Configuring the MongoDB Adapter 1715
Configure the OIEP 1715
Configuring the JavaScript Triggers 1717

MongoDB Adapter Setup Quick Guides 1718
STEP JSON Schema 1718
Configuring Mongo Authentication Quick Guide 1718
SSL Configuration Quick Guide 1719
Mongo Delivery Method Conversion Example 1721
Values 1721
References and Links 1723
Attribute Types 1725
Data Containers 1725
Reference Types 1727
Asset Push Locations 1727
Tables 1727
Table Formatting 1728
Attribute Links 1729
Data Container Type Links 1729
Data Containers Types 1730
Unit 1730
List Of Values 1731
Oracle AQ Delivery Method 1732
Product Data Exchange 2 Delivery Method 1734
REST Delivery Method 1737
REST Direct Delivery Method 1739
Authentication 1742
Basic Authentication 1743
Token-Based Authentication 1743
Additional Request Information 1744
SFTP Delivery Method 1747
Wiki Delivery Method 1750
Running an Outbound Integration Endpoint 1752
Enable and Invoke an Outbound Integration Endpoint 1752
Disable an Outbound Integration Endpoint 1755
Maintaining an Outbound Integration Endpoint 1757
Outbound Integration Endpoint Tab 1757
Description Flipper 1757
Integration Endpoint Log Flipper 1758
Configuration Tab 1759
Configuration Flipper 1760
Event Queue Configuration Flipper 1761
Object Selection Configuration Flipper 1762
Output Templates Flipper 1762
Delivery Method Flipper 1763
Background Processes Tab 1763
Statistics Tab 1764
Error Log Excerpts Tab 1765
Log Tab 1765
Status Tab 1765
Monitoring an OIEP via Background Process 1767
Background Process Priority Settings 1767
Legacy Background Process Queue Settings 1768
Legacy Background Process Queue 1768

Legacy Background Process Queue Size 1768

Monitoring an OIEP via External Systems 1769

Performance Data 1769

Monitoring Sensors 1769

REST API Monitoring 1770

Additional Information About All Outbound Integration Endpoints 1771

Handling Failed OIEP Background Processes 1772

Disable and Enable a Background Process 1772

Resume a Background Process 1772

Resolving a Failed Background Process 1773

Integration Endpoint Transactional Settings 1775

Transactional Settings Options 1777

Outbound Integration Endpoint Structure 1780

Business Rules Event Processing 1780

Processing Engine / Converter 1780

Error Handling & Reporting 1781

Post-Processor 1781

Delivery 1781

Additional Information About Event-Based OIEPs 1782

Event-Based OIEP Event Batching 1783

Event-Based OIEP Examples 1784

Event-Based Example Business Rules for Derived Events 1785

Event-Based Example Leaf Products with Inherited Data 1791

Event-Based Example Products Linked to Classification 1794

Event-Based Example Upward Inheritance 1798

Event-Based OIEP Event Actions 1801

Forward Events 1802

Rewind Events 1803

Purge Events 1803

Republish Events 1804

Skip All Events 1806

Event-Based OIEP Multithreading Support 1807

Multithreading Setup 1807

Additional Considerations 1808

Increasing the Batch Size 1808

Serialized and Parallelized 1808

Event-Based OIEP Order of Delete Events 1809

Unrelated Object Event Order Processing 1809

Event-Based OIEP Overview and Process Flow 1810

Event-Based OIEP Queued Events 1811

Estimate Unread Events 1811

View First Event Batch 1811

Event-Based OIEP Status and Queue Status 1814

Set the Queue Status 1814

HTTP Proxy Configurations 1816

Defining HTTP Proxy Configurations 1816
Mutual Transport Layer Security 1817
Outbound mTLS 1817
mTLS Configuration Properties 1818
Using Multiple Certificates 1820
Inbound mTLS for Stibo Systems SaaS .. 1821
SSL Client Certificates 1823
Trusted Certificates 1823
Prerequisites 1823
Adding a trusted certificate 1825
Trusted Certificates editor 1826
Tested URLs 1827
Tested URLs editor 1828
mTLS Client Certificate 1828
Prerequisites 1828
Download a client public key 1829
Web Service Endpoints 1829
Create a Web Service Endpoint 1830
Web Service Endpoint - Find Similar 1832
Configure a STEP Find Similar Web Service Endpoint 1832
Web Service Endpoint - Match and Merge 1835
Configure a STEP Match and Merge Web Service Endpoint 1835
Business Conditions and Contexts 1839
Match Advanced STEPXML Output Template 1839
Error handling 1841

Data Exchange

Data stored in STEP can be exchanged and integrated with external systems. This allows STEP, for example, to receive data from an upstream system such as an ERP, and then send data downstream to a web server.

Although this section deals with configuration of imports and exports in workbench, the formats, receiver methods, and delivery methods are the same regardless of the interface used. For information about imports and exports in Web UI, refer to **Data Exchange in Web UI**.

In addition to exchanging data, images and documents can also be exchanged. Assets and data can be sent downstream together using STEPXML. For details, refer to **Assets and Content with STEPXML**. For exchanging assets independent of data, refer to the **Digital Assets** documentation.

Inbound Tools

There are two primary methods for importing data to STEP:

1. **Import Manager** uses a selected format to import objects, classifications, and entities on demand.
2. **Inbound Integration Endpoints** (IIEPs) monitor a data source for receipt of new messages of a specified data format to import at scheduled intervals.

Data Format

The data format defines the inbound file type that will be processed. The same formats are available for both inbound tools (refer to **Data Formats**).

Receiver Method

For an IIEP, the selected receiver method determines how STEP accesses the inbound message (refer to the **Receiver Methods** section of the **IIEP - Choose Receiver** topic). The Import Manager does not use a receiver method.

Outbound Tools

The two primary methods for exporting data from STEP are:

1. **Export Manager** is used to export fixed data sets of objects, classifications, and entities on demand or at scheduled intervals using a selected format and delivery method.
2. **Outbound Integration Endpoints** (OIEPs) are primarily used to monitor STEP and automatically export relevant data via a selected data format and delivery method. OIEPs can also be used to export fixed data sets, and can be invoked on demand or at scheduled intervals.

Data Format

The data format determines the output file type. The same formats are available for both outbound tools (refer to **Data Formats**).

Delivery Method

The delivery method determines how the output file arrives at its destination. The available delivery methods are determined by the outbound tool used (refer to **OIEP - Delivery Method Flipper** and the **Export Manager - Select Delivery Method**).

Security

STEP supports Transport Layer Security (TLS) with mutual authentication (mTLS) for outbound HTTP traffic. This functionality is more secure than TLS, as it requires both the server and client to verify each other before data is transferred. mTLS functionality has been tested with the gateway integration endpoint REST plugin, the REST and REST Direct outbound integration endpoint (OIEP) delivery plugins, and URL Connections directly from business rule JavaScript. To learn more about Mutual Transport Layer Security and how to configure it, refer to the **Mutual Transport Layer Security** documentation in the Data Exchange section.

Gateway Tool

Gateway Integration Endpoints enable STEP to access external systems through business rules that make calls to fetch data or update status. The business rule uses synchronous REST calls from JavaScript.

For example, to create objects in STEP using an ID from an external ERP system, STEP can use a gateway endpoint to access and retrieve the external system ID.

Web Service Tools

Web Service Endpoints allow users to configure a web service call to receive requested data and transmit this response as needed. The web services supported by STEP are defined in the **Web Service Endpoints** topic.

Data Formats

For inbound data, the same formats are generally available using Import Manager or an IIEP.

For outbound data, the same formats are generally available using Export Manager or an OIEP.

Successfully importing 'less than' and 'greater than' tags requires special formatting based on the format. Refer to the **Character Tags** topic in the **System Setup** documentation for details.

Exceptions are noted by the asterisks (*) in the table below.

Format	Description	Inbound	Outbound
Advanced STEPXML	XML that uses a template based on the STEPXML XSD. Refer to **Advanced STEPXML Format** documentation. *Although Advanced STEPXML is not available for selection in inbound tools, Advanced STEPXML files can be imported successfully using the **STEPXML Format** option.	Yes*	Yes
Alphabetical Index - XML	XML that used with the InDesign solution to create indexes. Refer to **Alphabetical Index - XML Format** documentation.	No	Yes
Ariba CIF	CIF is a simple, comma-delimited list of catalog items and their attributes. Refer to **Ariba CIF Format** documentation.	No	Yes
Ariba CIF 3.0	CIF is a simple, comma-delimited list of catalog items and their attributes. Refer to **Ariba CIF 3.0 Format** documentation.	No	Yes
BMEcat	XML-based standard for electronic data transfer by electronic catalogs. Refer to **BMEcat Format** documentation.	Yes	Yes
BMEcat 2005	XML-based standard for electronic data transfer by electronic catalogs. Refer to **BMEcat 2005 Format** documentation.	Yes	Yes
BMEcat 2005.1	XML-based standard for ECLASS Advanced data. Refer to **BMEcat 2005.1 Format** documentation.	No	Yes
CSV	Comma-separated values (CSV) file. Refer to **CSV Format** documentation.	Yes	Yes
cXML	XML 'commerce eXtensible Markup Language' (cXML). Refer to **cXML Format** documentation.	No	Yes
ECLASS	ECLASS (formerly eCl@ss) is a hierarchical classification and product description system for grouping materials, products, and services. Refer to **ECLASS Format** documentation. *This import is only available via the eCl@ss Classification Import wizard.	Yes*	No
ETIM	CSV Electro-Technical Information Model (ETIM) classification structure where each file contains a single language. Refer to	Yes	No

Format	Description	Inbound	Outbound
	ETIM and ETIM v2 Format documentation.		
ETIM IXF	XML Electro-Technical Information Model (ETIM) classification structure that includes multiple languages. Refer to **ETIM IXF Format** documentation.	Yes	No
ETIM v2	CSV Electro-Technical Information Model (ETIM) classification structure where each file contains a single language. Refer to **ETIM and ETIM v2 Format** documentation.	Yes	No
Excel	Microsoft Excel spreadsheet file. Refer to **Excel Format** documentation.	Yes	Yes
Excel Custom Template	Microsoft Excel spreadsheet file. Refer to **Excel Custom Template** documentation.	No	Yes
Excel Smartsheet	Smartsheet spreadsheet file. Refer to **Excel Smartsheet Format** documentation.	Yes	Yes
FAB-DIS Format	Excel file used as an exchange format, primarily by companies doing business in France, to share quality product information between manufacturers and distributors. Refer to **FAB-DIS Format** documentation.	Yes	Yes
FixedWidth	Text file with rows and columns of data, where each column has a fixed width. Refer to **FixedWidth Format** documentation.	Yes	No
Flatplan Excel	Excel file for publications. Refer to **Flatplan Excel Format**. *This export is only available via Export Manager.	Yes	Yes*
Generic JSON	JavaScript Object Notation (JSON) structure of attribute / value pairs that can be adjusted by changing the template. Refer to **Generic JSON Format** documentation.	Yes	Yes
Generic XML	XML with a variety of accessible formats. Refer to **Generic XML Format** documentation.	Yes	Yes
IDoc	XML format used when exchanging information with SAP. Refer to	Yes	Yes

Format	Description	Inbound	Outbound
MATMAS 05	**IDoc MATMAS 05 Format** documentation.		
Publication Excel	Excel file for publications. Refer to **Publication Excel Format**. *This export is only available via Export Manager.	Yes	Yes*
SmartLabel	Generic XML-based including a pre-defined template for integrators to provide a head start in configuring a SmartLabel™ export solution for both food and non-food product data. Refer to **SmartLabel Format** documentation.	No	Yes
STEPXML	XML generated based on parameters displayed in a dialog. Unlike Advanced STEPXML format, no template is required. Refer to **STEPXML Format** documentation.	Yes	Yes
STEPXML Configuration Export	Cross-context XML file that contains the full STEP configuration (minus user objects) and the data nodes that are required for being able to import the configuration on to an empty STEP system; template required. Refer to **STEPXML Configuration Export Format** documentation.	No	Yes
UNSPSC	United Nations Standard Products and Services Code® (UNSPSC) multi-level classification folders. Refer to **UNSPSC Format** documentation.	Yes	No
xCBL	XML Common Business Library (xCBL) for business-to-business e-commerce. Refer to **xCBL Format** documentation.	No	Yes

Advanced STEPXML Format

The Advanced STEPXML format and the STEPXML format provide access to the same functionality but each has its own interface. The STEPXML format is easier for beginners to use since its user interface includes parameters with valid options. Using the Advanced STEPXML format requires knowledge of the STEPXML tags, XML attributes, and valid settings for the schema that must be created manually in the Export Manager or OIEP template parameter (refer to the **Outbound Data** section below).

As defined in the tags and examples documentation, not all data that can be exported using Advanced STEPXML can be imported again. Available tags and examples using STEPXML are available in the **STEPXML Tags and Examples** documentation.

Note: For assistance in creating a valid Advanced STEPXML template, use the STEPXML user interface to output a template and then modify as needed for inbound or outbound STEPXML data. For more information, refer to **STEPXML Format**.

Schema

The schema is not publicly available, which means it is not possible to validate STEPXML without access to the XSD file.

The XSD file for STEPXML is available at:

```
http://[application server]/files
```

A web view of the schema is available at:

```
http://[application server]/xsd/index.html
```

The STEPXML **XSD** file can also be found in the **STEP API Documentation**, which is accessible from the Start Page of your STEP instance.

The STEPXML schema is extended continuously but should always be backward compatible. For more information on legacy naming conventions, refer to the **Backward Compatibility in STEPXML** documentation.

Format Availability

Advanced STEPXML is available for selection in:

- IIEP - Although Advanced STEPXML is not available for selection in inbound tools, Advanced STEPXML files can be imported successfully using the **STEPXML Format** option.
- Import Manager - Although Advanced STEPXML is not available for selection in inbound tools, Advanced STEPXML files can be imported successfully using the **STEPXML Format** option.
- Export Manager - refer to **Creating a Data Export**.
- OIEP - refer to **Creating an Outbound Integration Endpoint**.
- WSE - refer to **Web Service Endpoints**.

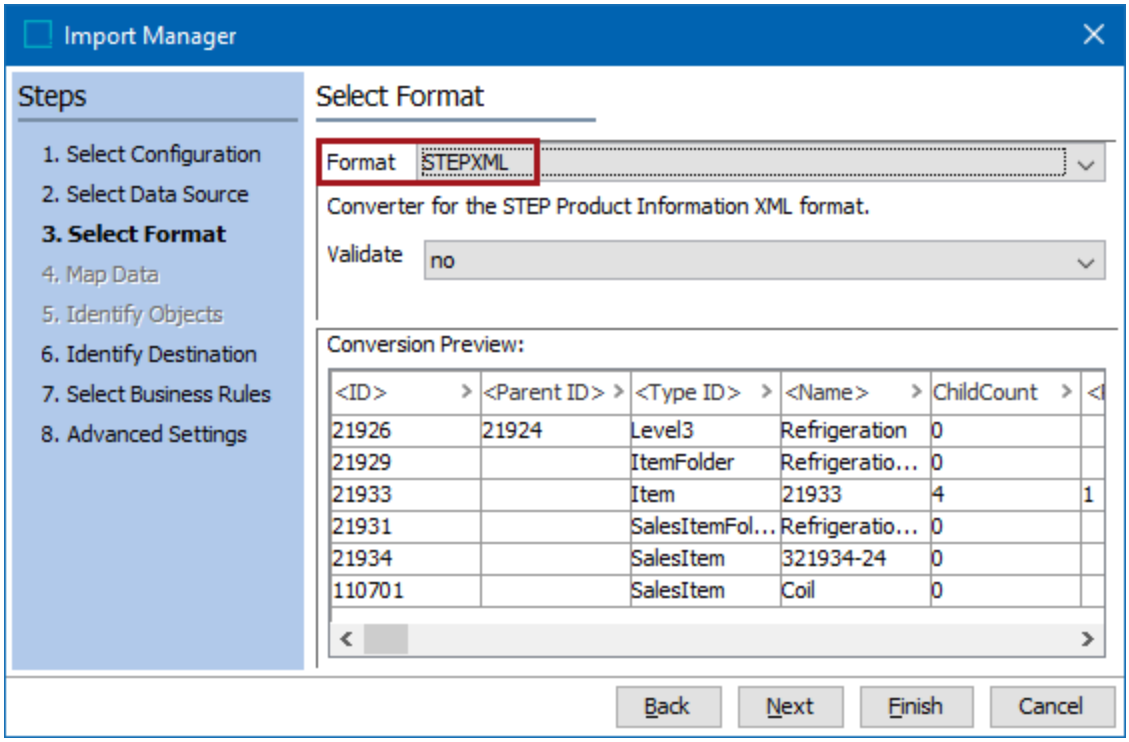
Mapping

Since STEPXML is the native format for STEP, mapping data is not required and the Map Data step is disabled.

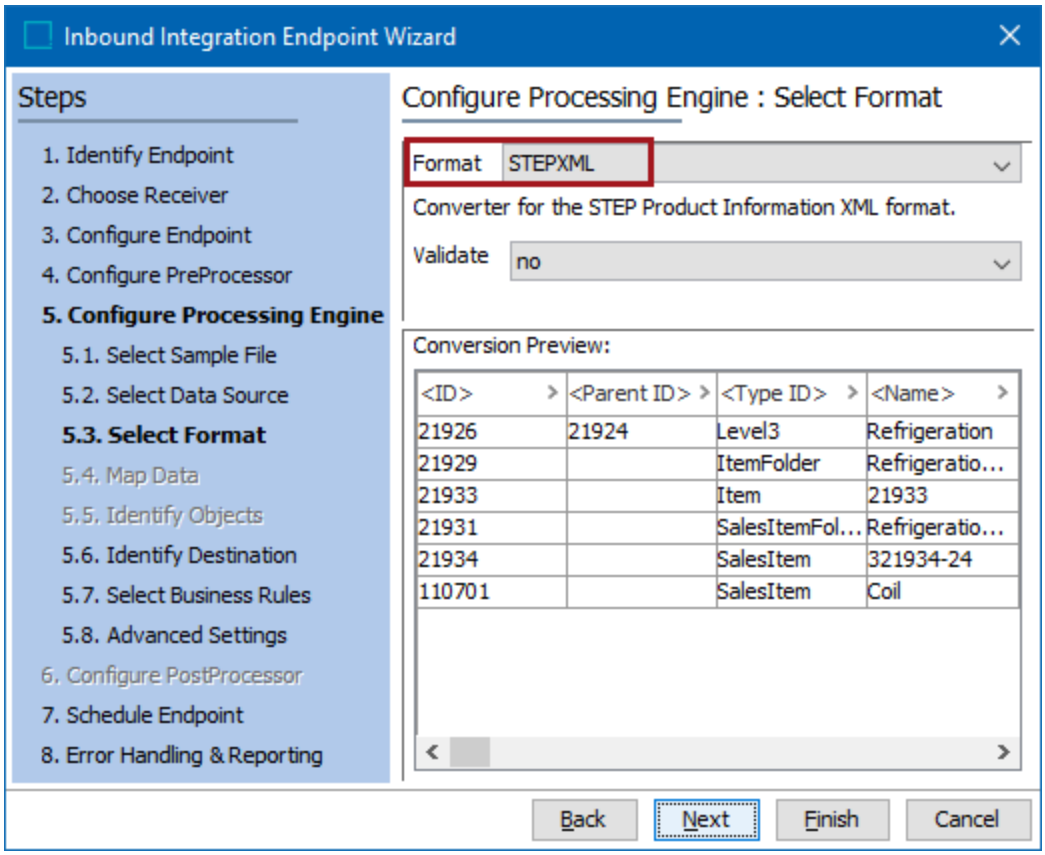
Inbound Data

Although Advanced STEPXML is not available for selection in inbound tools, Advanced STEPXML files can be imported successfully using the **STEPXML Format** option.

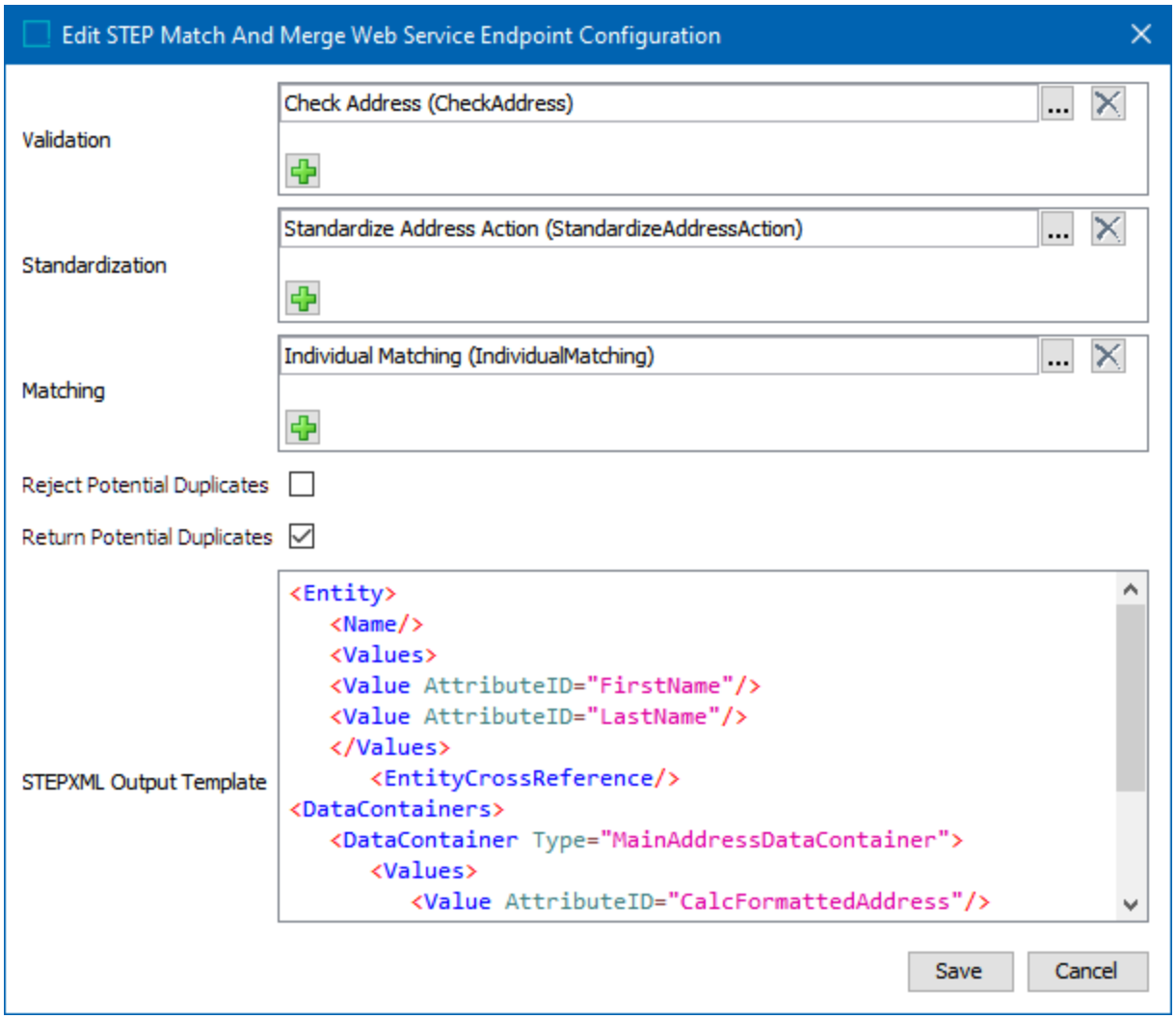
Import Manager

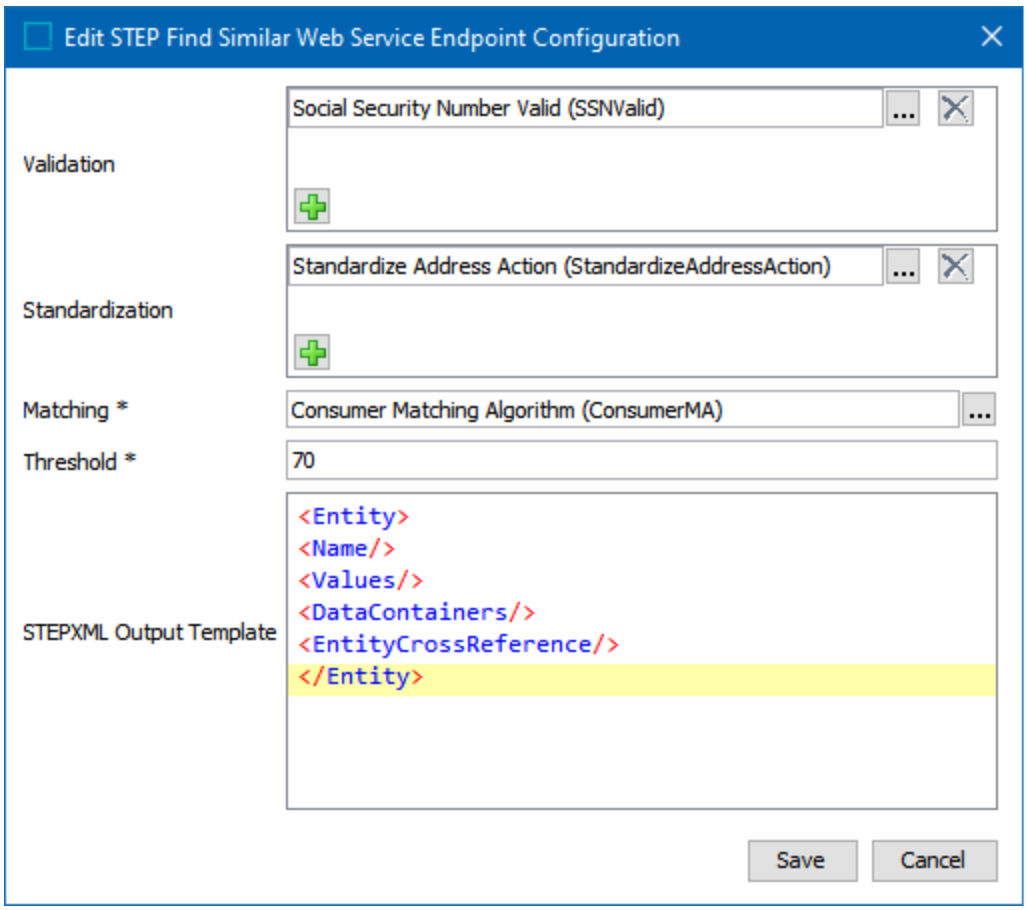


IIEP


Inbound Integration Endpoint Wizard
Steps
1. Identify Endpoint
2. Choose Receiver
3. Configure Endpoint
4. Configure PreProcessor
5. Configure Processing Engine
5.1. Select Sample File
5.2. Select Data Source
5.3. Select Format
5.4. Map Data
5.5. Identify Objects
5.6. Identify Destination
5.7. Select Business Rules
5.8. Advanced Settings
6. Configure PostProcessor
7. Schedule Endpoint
8. Error Handling & Reporting
Configure Processing Engine : Select Format
Format STEPXML
Converter for the STEP Product Information XML format.
Validate no
Conversion Preview:
<ID> <Parent ID> <Type ID> <Name>
21926 21924 Level3 Refrigeration
21929 ItemFolder Refrigeratio...
21933 Item 21933
21931 SalesItemFol... Refrigeratio...
21934 SalesItem 321934-24
110701 SalesItem Coil
Back
Next
Finish
Cancel

WSE


Edit STEP Match And Merge Web Service Endpoint Configuration
Validation
Check Address (CheckAddress)
Standardization
Standardize Address Action (StandardizeAddressAction)
Matching
Individual Matching (IndividualMatching)
Reject Potential Duplicates
Return Potential Duplicates
STEPXML Output Template
<Entity>
<Name/>
<Values>
<Value AttributeID="FirstName"/>
<Value AttributeID="LastName"/>
</Values>
<EntityCrossReference/>
<DataContainers>
<DataContainer Type="MainAddressDataContainer">
<Values>
<Value AttributeID="CalcFormattedAddress"/>
Save
Cancel



Outbound Data

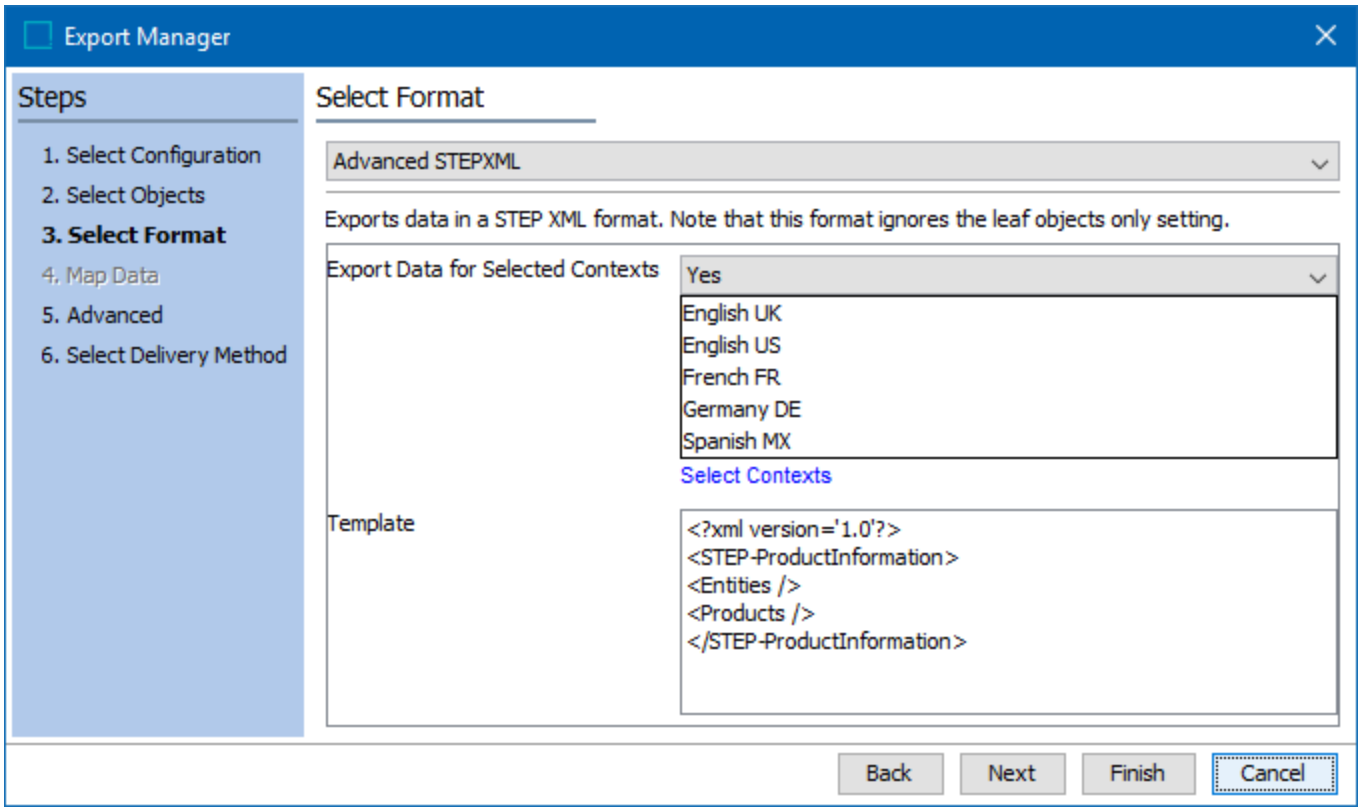
The following parameters are available for outbound data using the Export Manager and/or OIEPs, as defined below:

1. **Export Data for Selected Contexts** - For Export Manager only, use this parameter to select multiple contexts for exported data. For OIEPs, multiple contexts are selected in the Contexts parameter on the Configuration flipper, as defined in the **OIEP - Configuration Flipper** topic.
2. **Template** - For both Export Manager and OIEPs, the default template loaded for export will output all entities and products. Update it with your own template to configure the data, control the attribute groups and reference types, and define the layout of the export.

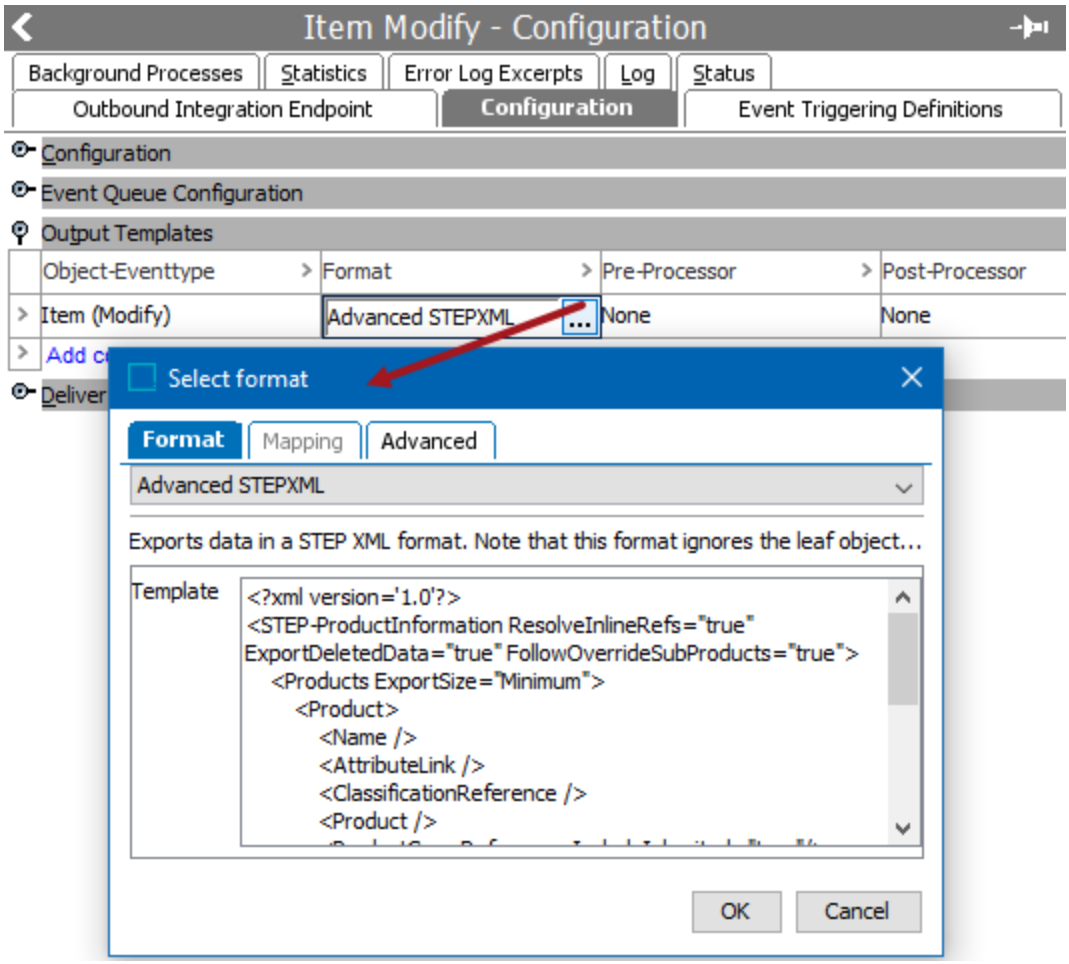
   Since no undo functionality is available in the Template field, it is good practice to use an external XML editor program for creating and editing a template. To test a template's validity for output, paste the XML into STEP.

   Available tags and example templates using STEPXML are available in the **STEPXML Tags and Examples** documentation. Ultimately, the template must be based on the STEPXML XSD. Click the **STEP API Documentation** button on the Start Page for a link to the XSD file.

Export Manager



OIEP


Item Modify - Configuration
Background Processes
Statistics
Error Log Excerpts
Log
Status
Outbound Integration Endpoint
Configuration
Event Triggering Definitions
Configuration
Event Queue Configuration
Output Templates
Object-Eventtype
Format
Pre-Processor
Post-Processor
Item (Modify)
Advanced STEPXML
None
None
Select format
Format
Mapping
Advanced
Advanced STEPXML
Exports data in a STEP XML format. Note that this format ignores the leaf object...
Template
<?xml version='1.0'?>
<STEP-ProductInformation ResolveInlineRefs="true"
ExportDeletedData="true" FollowOverrideSubProducts="true">
<Products ExportSize="Minimum">
<Product>
<Name />
<AttributeLink />
<ClassificationReference />
<Product />
OK
Cancel

Alphabetical Index - XML Format

The Alphabetical Index - XML format exports data from an index word hierarchy for a STEP publication. 'Stock number' or 'article number' can also be exported for creating indexes from a STEP publication. For more information, refer to the **Creating Document Indexes** topic or the **Exporting Index Data from STEP** topic in the **Publisher (Adobe InDesign Integration)** documentation.

Format Availability

Alphabetical Index - XML is available for selection in:

- Export Manager - refer to **Creating a Data Export**
- OIEP - refer to **Creating an Outbound Integration Endpoint**

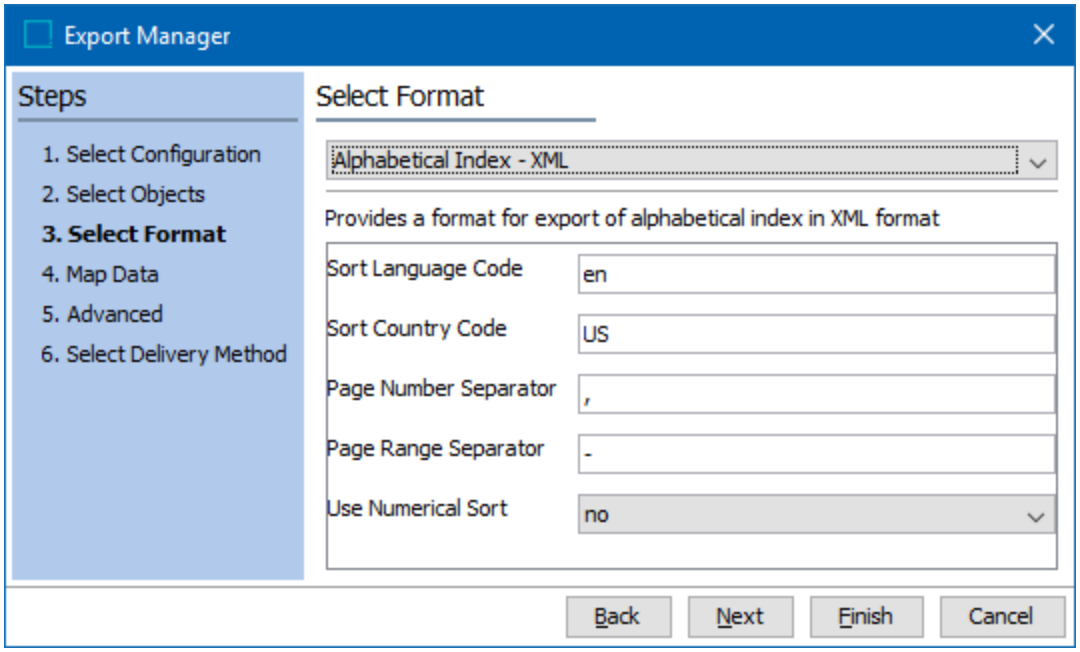
Mapping

This format requires creating a data map between STEP and the data being processed, and may also include data transformations. For details, refer to **Data Mapping**.

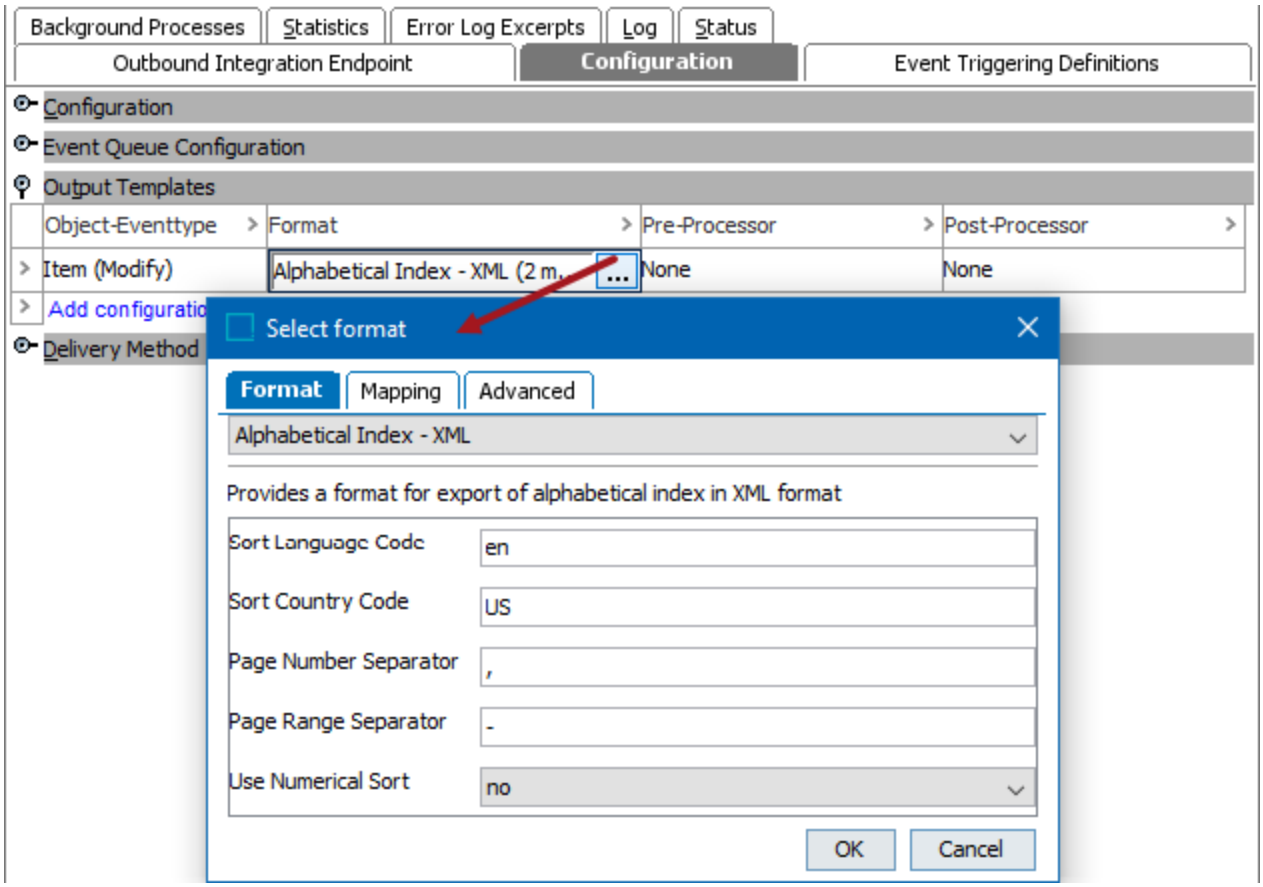
Outbound Data

The same parameters are available in both Export Manager and OIEP.

Export Manager



OIEP


Background Processes
Statistics
Error Log Excerpts
Log
Status
Outbound Integration Endpoint
Configuration
Event Triggering Definitions
Configuration
Event Queue Configuration
Output Templates
Object-Eventtype
Format
Pre-Processor
Post-Processor
Item (Modify)
Alphabetical Index - XML (2 m...
None
None
Add configuratio
Delivery Method
Select format
Format
Mapping
Advanced
Alphabetical Index - XML
Provides a format for export of alphabetical index in XML format
Sort Language Code
en
Sort Country Code
US
Page Number Separator
,
Page Range Separator
-
Use Numerical Sort
no
OK
Cancel

Ariba CIF Format

The Ariba CIF format is a simple comma-delimited list of catalog items and their attributes. For more information, search the web.

Format Availability

Ariba CIF is available for selection in:

- Export Manager - refer to **Creating a Data Export**
- OIEP - refer to **Creating an Outbound Integration Endpoint**

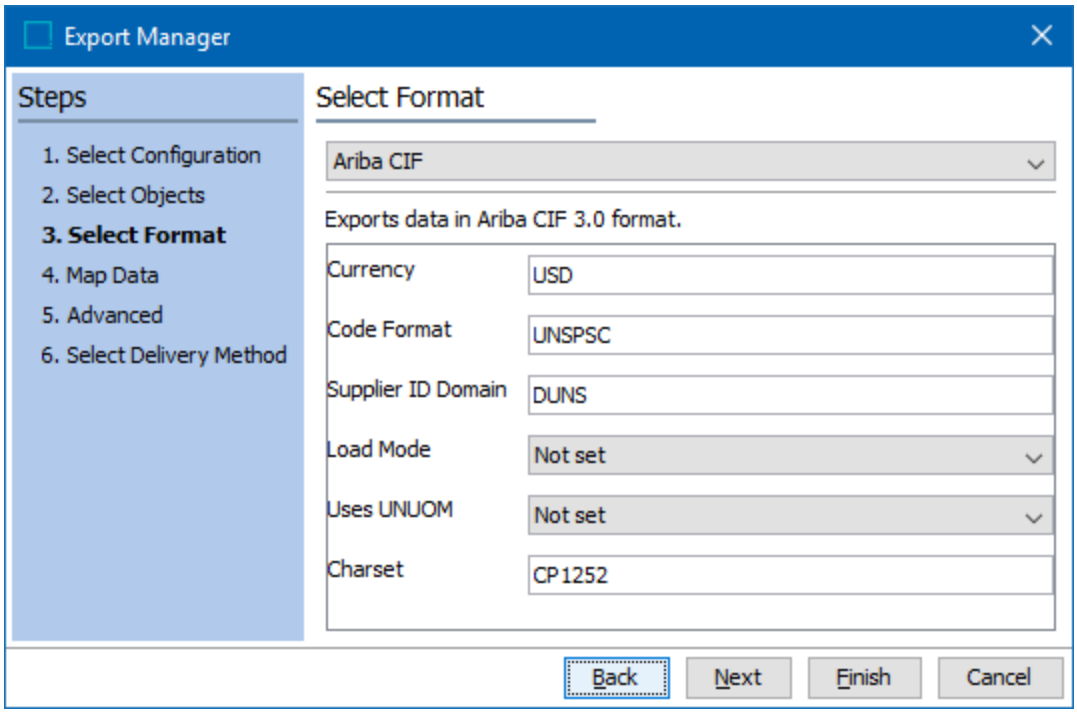
Mapping

This format requires creating a data map between STEP and the data being processed, and may also include data transformations. For details, refer to **Data Mapping**.

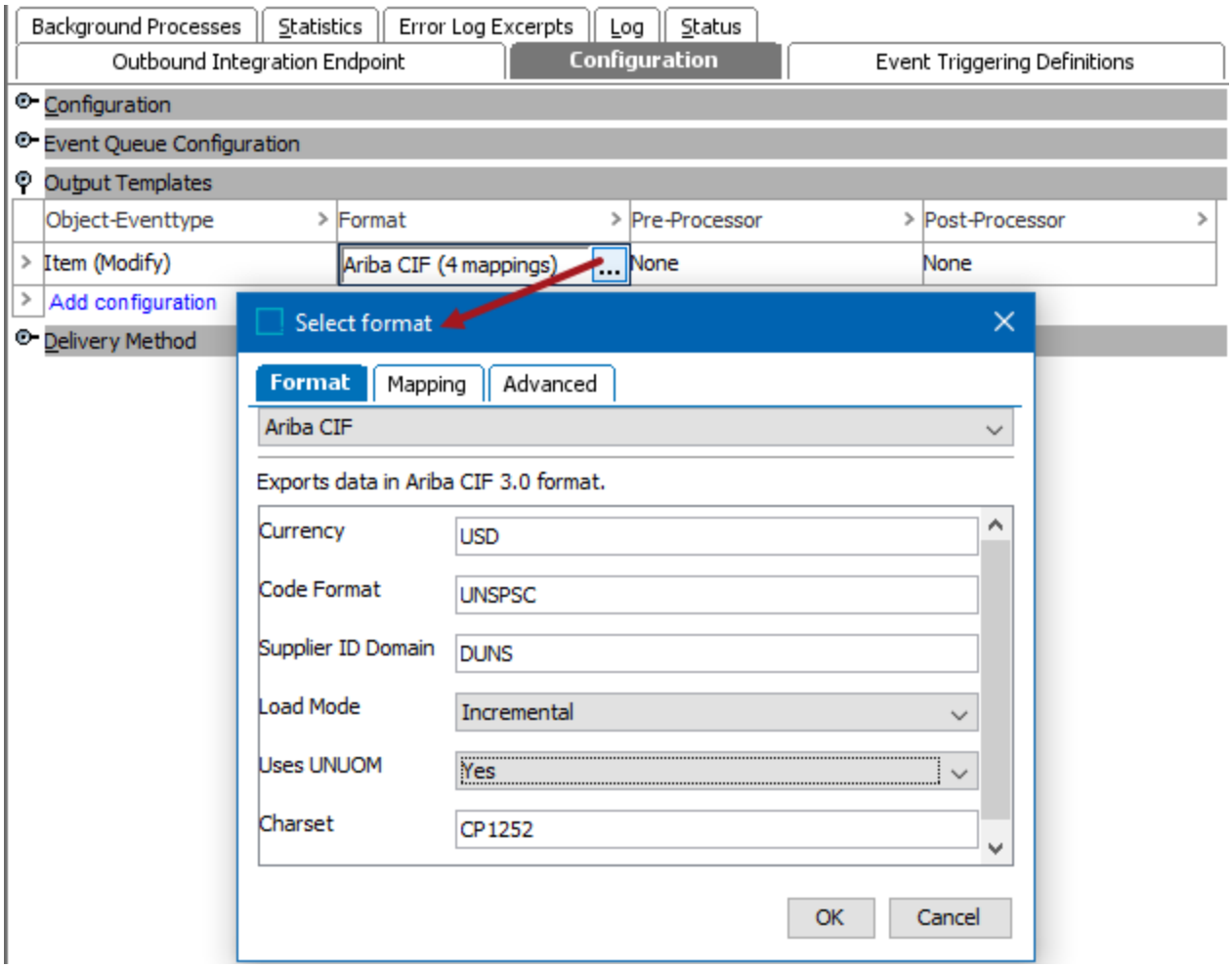
Outbound Data

The same parameters are available in both Export Manager and OIEP.

Export Manager



OIEP


Background Processes
Statistics
Error Log Excerpts
Log
Status
Outbound Integration Endpoint
Configuration
Event Triggering Definitions
Configuration
Event Queue Configuration
Output Templates
Object-Eventtype
Format
Pre-Processor
Post-Processor
Item (Modify)
Ariba CIF (4 mappings)
None
None
Add configuration
Delivery Method
Select format
Format
Mapping
Advanced
Ariba CIF
Exports data in Ariba CIF 3.0 format.
Currency
USD
Code Format
UNSPSC
Supplier ID Domain
DUNS
Load Mode
Incremental
Uses UNUOM
Yes
Charset
CP1252
OK
Cancel

Ariba CIF 3.0 Format

The Ariba CIF 3.0 format is a simple comma-delimited list of catalog items and their attributes. For more information, search the web.

Format Availability

Ariba CIF 3.0 is available for selection in:

- Export Manager - refer to **Creating a Data Export**
- OIEP - refer to **Creating an Outbound Integration Endpoint**

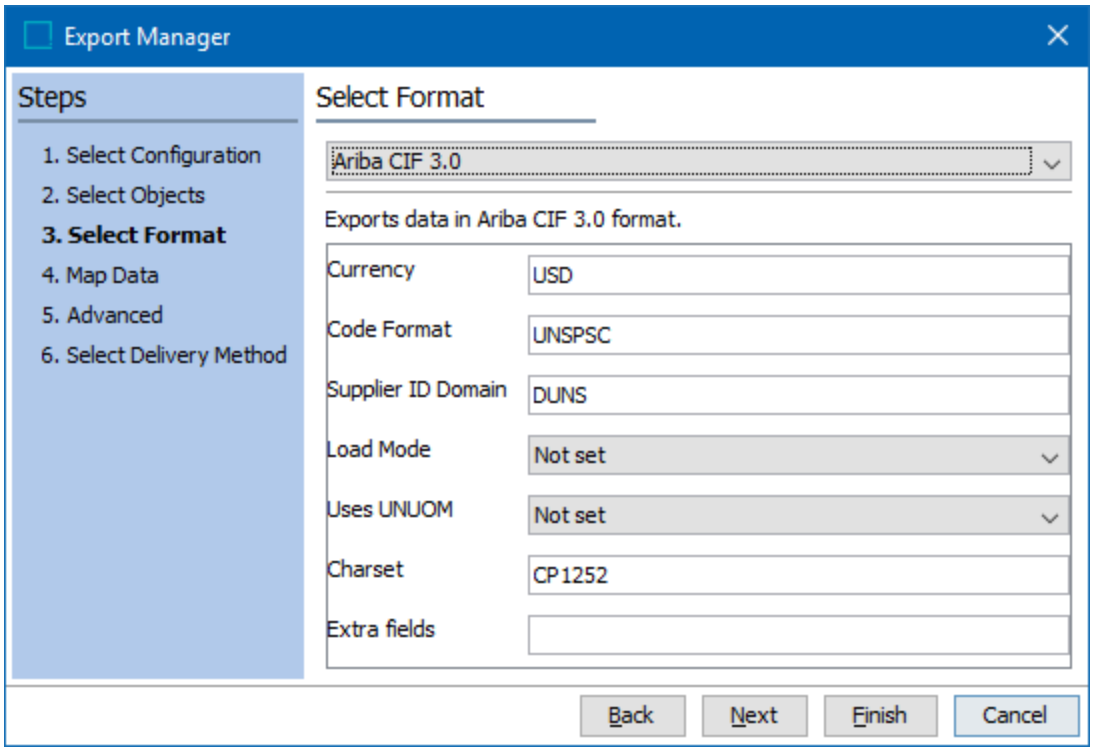
Mapping

This format requires creating a data map between STEP and the data being processed, and may also include data transformations. For details, refer to **Data Mapping**.

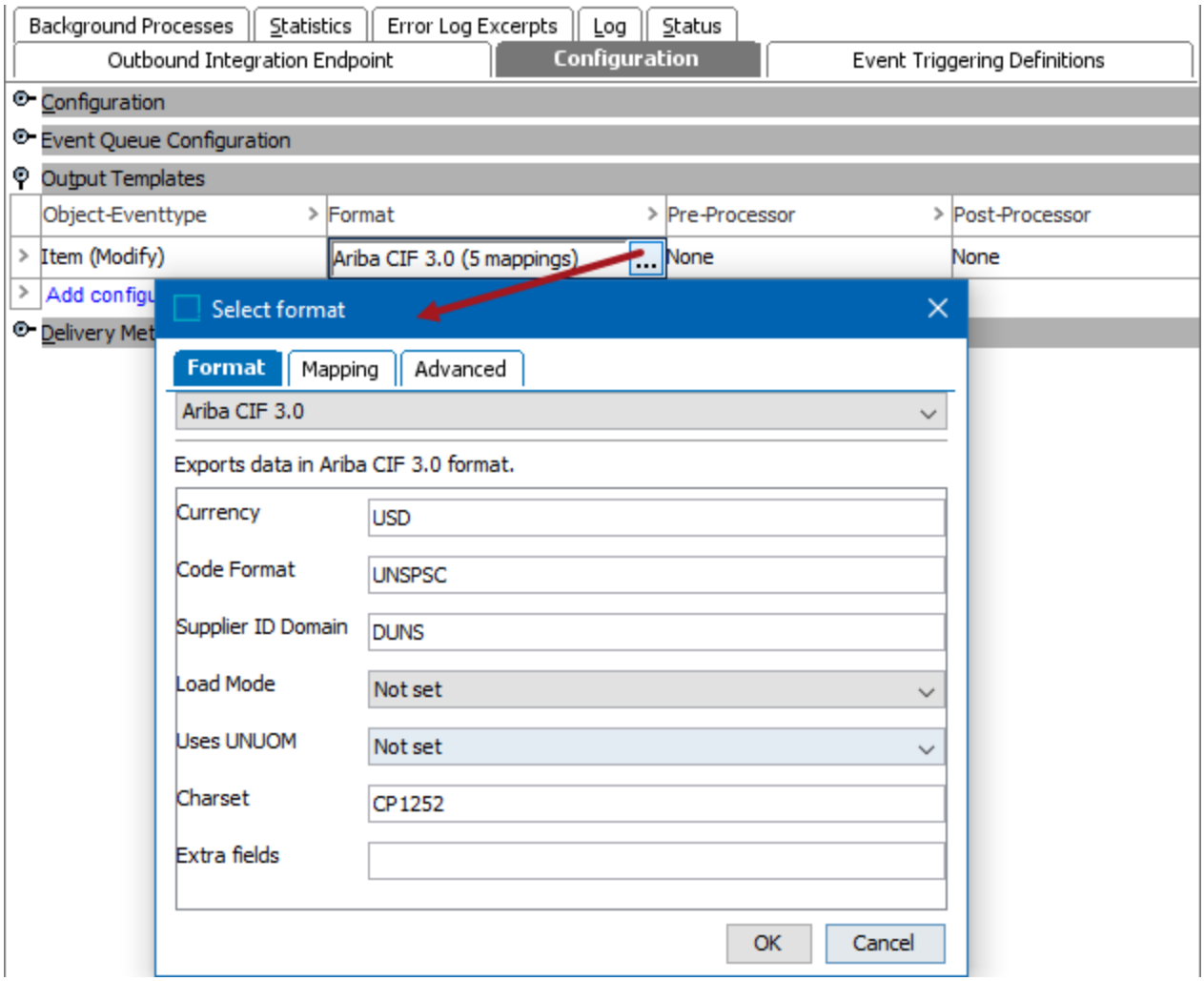
Outbound Data

The same parameters are available in both Export Manager and OIEP.

Export Manager



OIEP


Background Processes
Statistics
Error Log Excerpts
Log
Status
Outbound Integration Endpoint
Configuration
Event Triggering Definitions
Configuration
Event Queue Configuration
Output Templates
Object-Eventtype
Format
Pre-Processor
Post-Processor
Item (Modify)
Ariba CIF 3.0 (5 mappings)
None
None
Add configu
Delivery Met
Select format
Format
Mapping
Advanced
Ariba CIF 3.0
Exports data in Ariba CIF 3.0 format.
Currency
USD
Code Format
UNSPSC
Supplier ID Domain
DUNS
Load Mode
Not set
Uses UNUOM
Not set
Charset
CP1252
Extra fields
OK
Cancel

BMEcat Format

Both BMEcat (version 1.2) and BMEcat 2005 are supported in STEP and use an XML-based standard for electronic data transfer by electronic catalogs. For more information about the standard, search the web.

BMEcat requires the 'X.Import.BMEcat' license and / or the 'X.Export.BMEcat' license. No add-on components must be installed.

For more information on BMEcat 2005, refer to the **BMEcat 2005 Format** documentation.

Format Availability

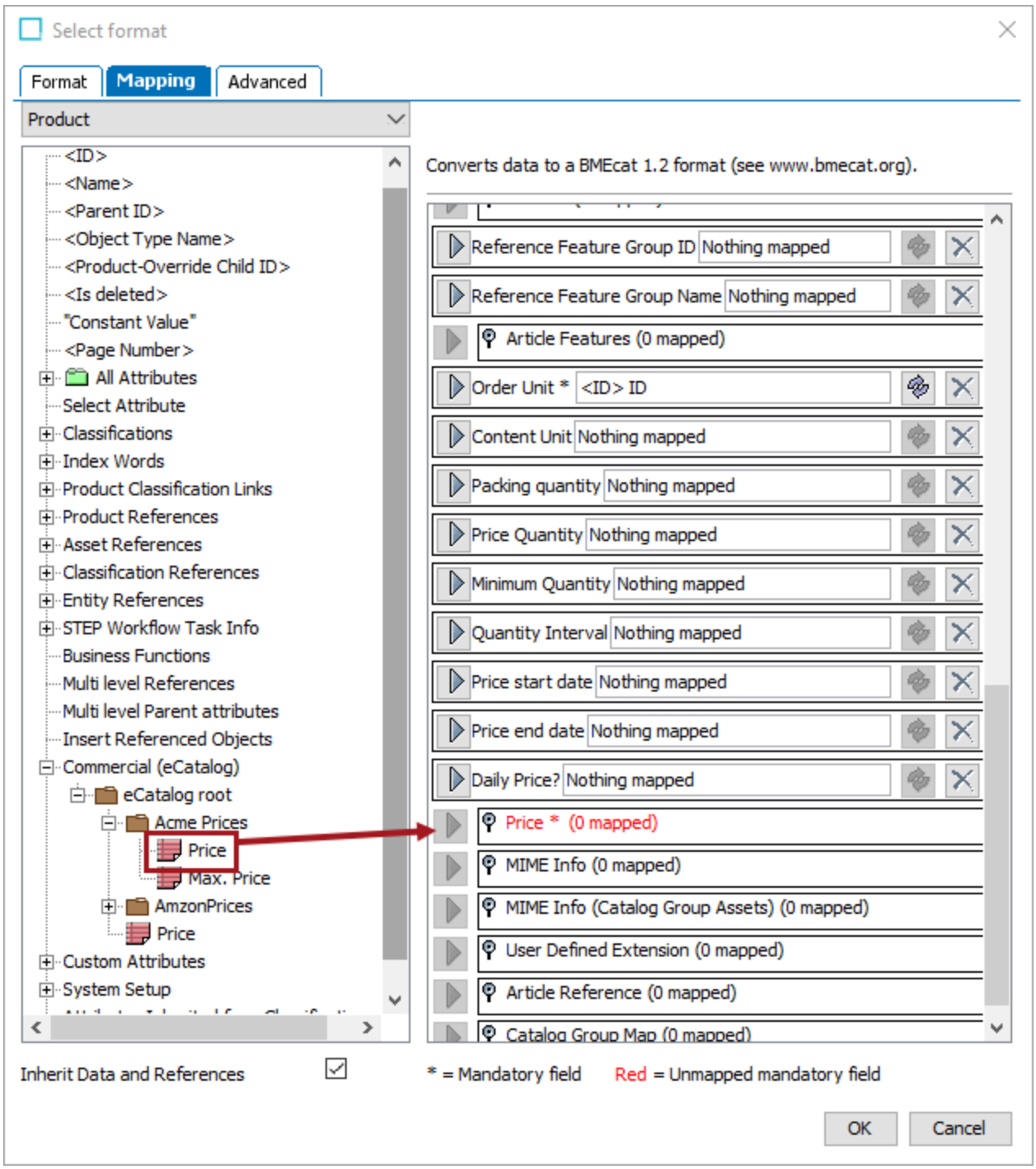
BMEcat is available for selection in:

- IIEP - refer to **Creating an Inbound Integration Endpoint**
- Import Manager - refer to **Creating a Data Import**
- Export Manager - refer to **Creating a Data Export**
- OIEP - refer to **Creating an Outbound Integration Endpoint**

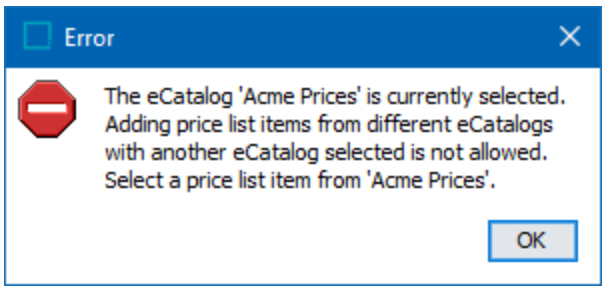
Mapping

This format requires creating a data map between STEP and the data being processed, and may also include data transformations. For details, refer to **Data Mapping**.

To map price data for eCatalogs in outbound integration endpoints (OIEPs) or in the Export Manager, first expand **Commercial (eCatalog)** as shown in the screenshot below. Next, expand the relevant eCatalog (in this example, 'Acme Prices'), then select the relevant price list. In this example, the price list is named 'Price.' Then click the arrow icon next to **Price** in the conversion column to export the data associated with the commercial term into the BMEcat export.



A user can map multiple price lists from the same eCatalog, but cannot map price lists from multiple eCatalogs. If this is attempted, the system displays a warning message that looks like this:



For more information on configuring eCatalogs for export via BMEcat, refer to the 'Configure an eCatalog Configuration for BMEcat' section of the **Creating an eCatalog Configuration** topic in the **eCatalogs** documentation.

BMEcat 1.2 and BMEcat 2005 are the only formats that allow access to Product Attribute Link metadata attributes, via the transformation aspect 'Closest Attribute Link Meta Data.' For details, refer to **Aspect - Transform Outbound**.

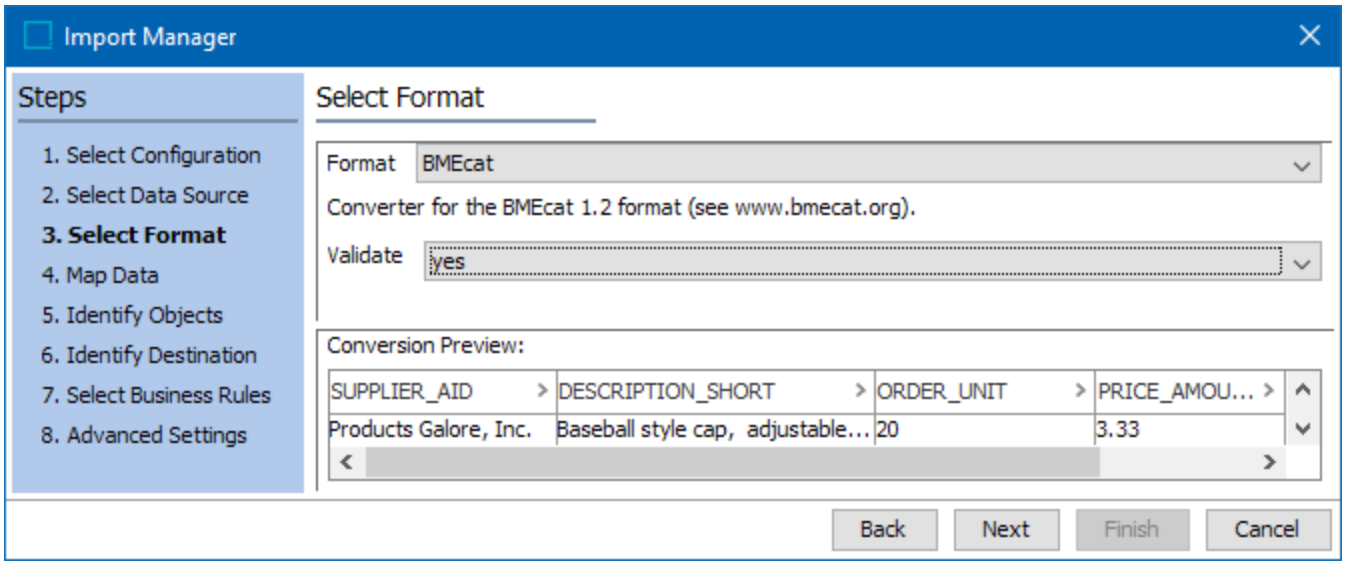
If mapping an ETIM file to the BMEcat2005 format, refer to the **LOV Value-ID or Value Aspect** topic.

Inbound Data

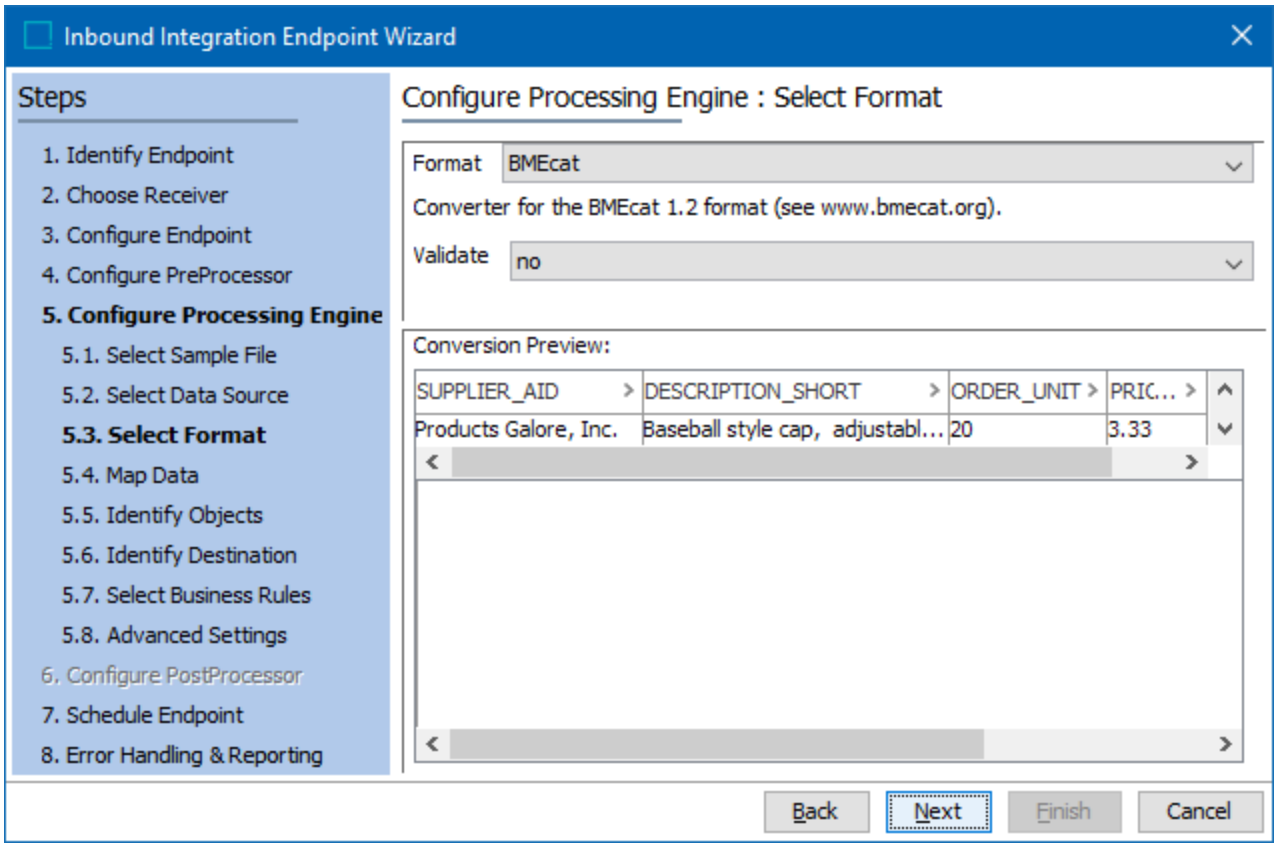
Inbound Parameters

- **Validate** determines if STEP checks the validity of the BMEcat file against a BMEcat DTD file before loading any data.
- **Conversion Preview** displays data being imported when the BMEcat DTD file is in the system., otherwise the field is empty.

Import Manager



IIEP



Outbound Data

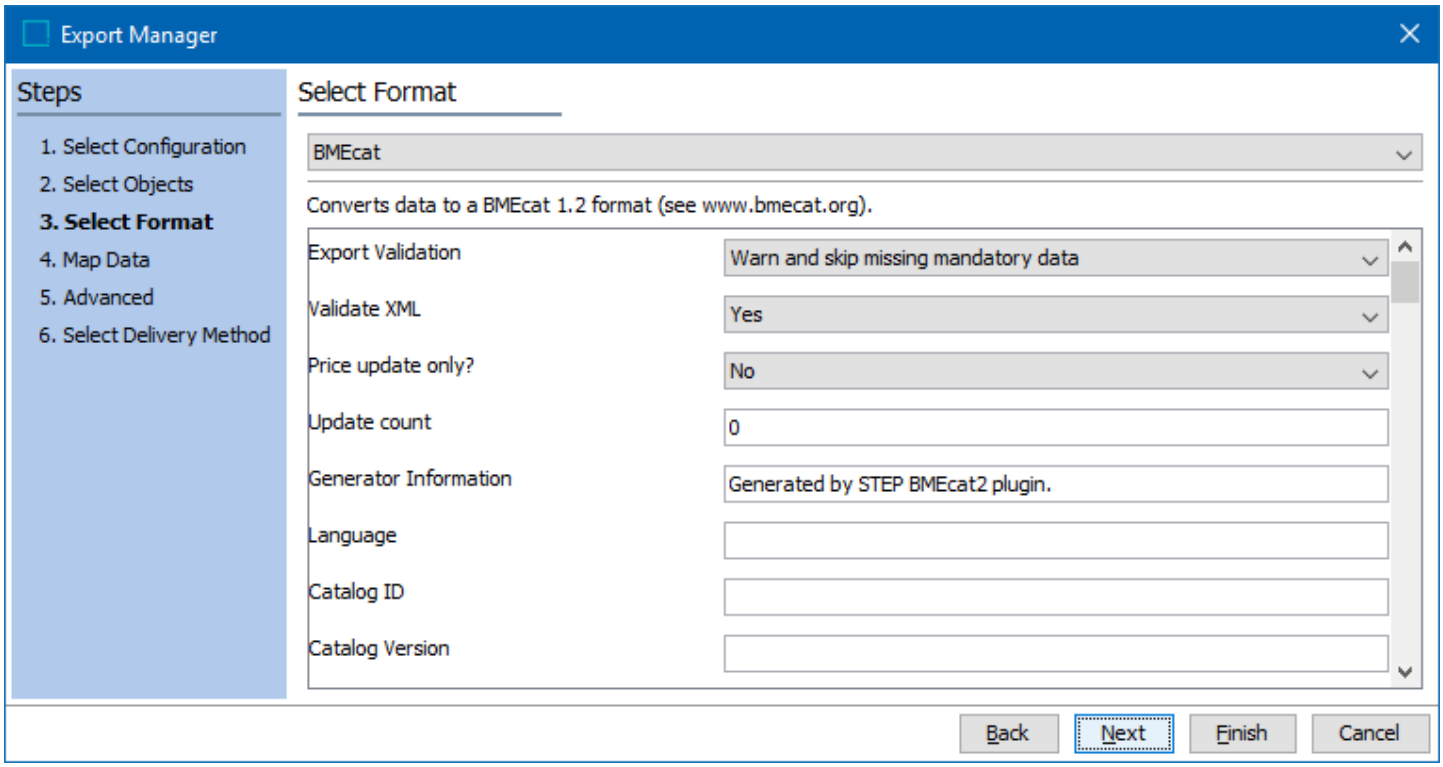
Exporting the BMEcat format involves supplying header data and catalog / product data via the parameters displayed on the Select Format step and during the Map Data step. Some information is required, as is indicated on the Map Data step. Although the export process can be run while mandatory fields are missing, the background process (BGP) will fail.

Outbound Parameters

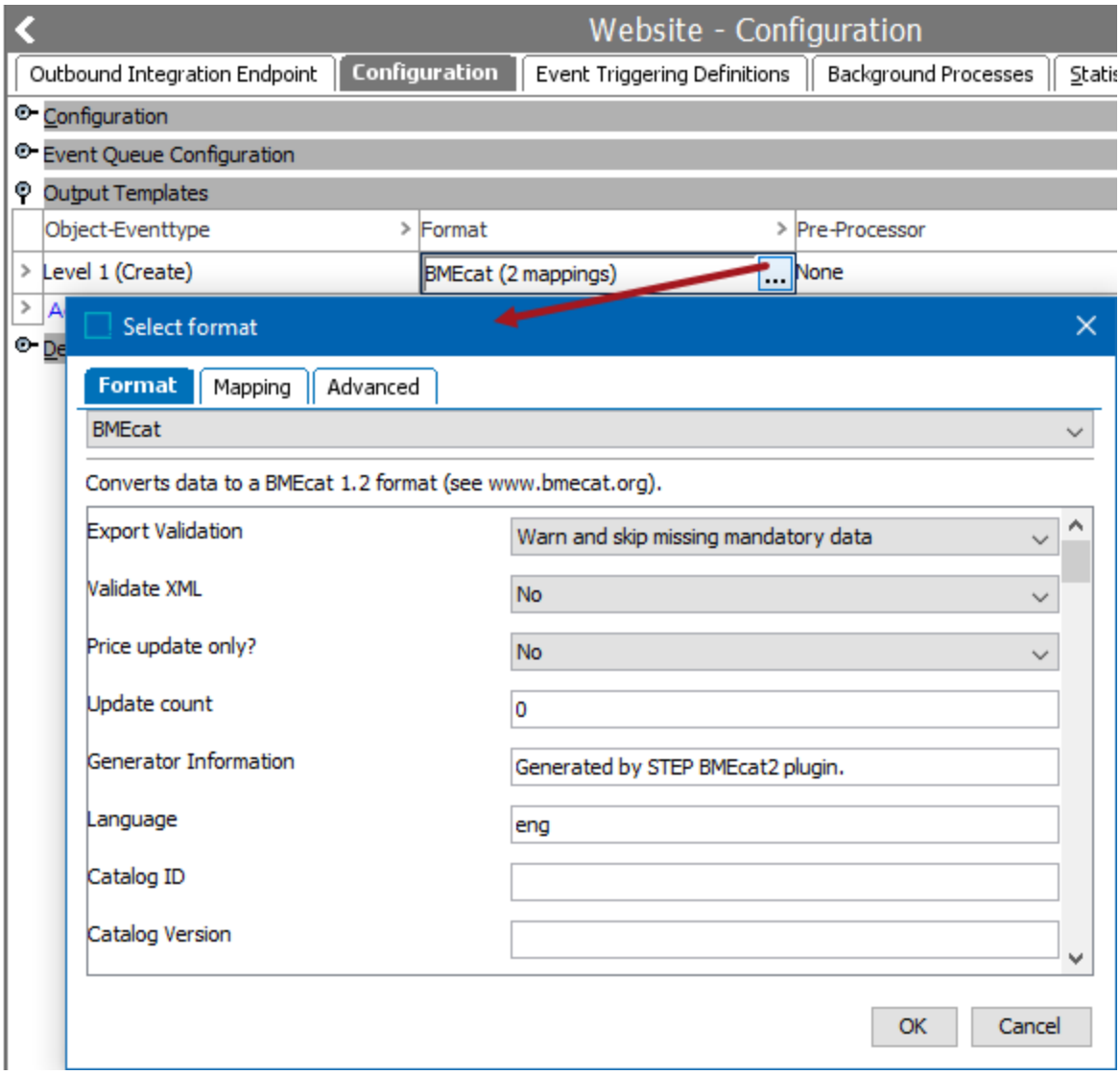
1. For the **Export Validate** parameter, select an option from the dropdown to determine how mandatory fields (marked in red on the Map Data step) are handled during export:
   - **Warn and skip missing mandatory data** - When a value is missing, the product is not exported and the execution report includes the warning 'Skipping product ([ID] / [Name]) missing mandatory data: xxxx.'
   - **Warn and allow missing mandatory data** - When a value is missing, the product is exported minus the missing mandatory data, and the execution report includes the warning 'Found product ([ID] / [Name]) missing mandatory data: xxxx.'
   - **Abort on missing mandatory data** - When a value is missing, the export fails and the execution report includes 'Aborting export due to product ([ID] / [Name]) missing mandatory data: xxxx' and the time of the failure.

2. For the **Validate XML** parameter, **Yes** indicates that the BMEcat 1.2 export file is validated against the BMEcat DTD included in STEP. When validation fails, the BGP also fails, and the problem is reported in the BGP execution report. If set to **No**, validation is skipped, and the BGP does not fail due to differences found compared to the DTD.
3. For the **Price update only?** parameter, **No** indicates that the catalog section of the file is included. If set to No, the catalog section is excluded and only fields related to prices are exported.
4. For the **Update count** parameter, allows you to track the version number displayed in the tag with the 'prev_ version' XML attribute within the file. This parameter must be manually incremented as needed.
5. For the **Allow Empty Values** parameter, **Yes** indicates that data sources that are mapped but do not have a value are exported. If set to **No**, mapped data sources without values are skipped during export.
6. For an explanation of the remaining parameters, search the web to find and refer to the 'Specification BMEcat® Version 1.2' document. No validation is performed on the text entered or the selections made, but if Validate XML = Yes, errors will be reported in the execution report, as defined above.

Export Manager



OIEP


Website - Configuration
Outbound Integration Endpoint
Configuration
Event Triggering Definitions
Background Processes
Configuration
Event Queue Configuration
Output Templates
Object-Eventtype
Format
Pre-Processor
Level 1 (Create)
BMEcat (2 mappings)
None
Select format
Format
Mapping
Advanced
BMEcat
Converts data to a BMEcat 1.2 format (see www.bmecat.org).
Export Validation
Warn and skip missing mandatory data
Validate XML
No
Price update only?
No
Update count
0
Generator Information
Generated by STEP BMEcat2 plugin.
Language
eng
Catalog ID
Catalog Version
OK
Cancel

BMEcat 2005 Format

Both BMEcat® (version 1.2) and BMEcat® 2005 are supported in STEP and use an XML-based standard for electronic data transfer by electronic catalogs. For more information about the standard, search the web.

To use the BMEcat functionality, the BMEcat commercial license needs to be enabled. Contact your account manager or partner manager to begin the process of enabling a license or licenses for your system.

For more information on BMEcat (version 1.2), refer to the **BMEcat Format** documentation.

Format Availability

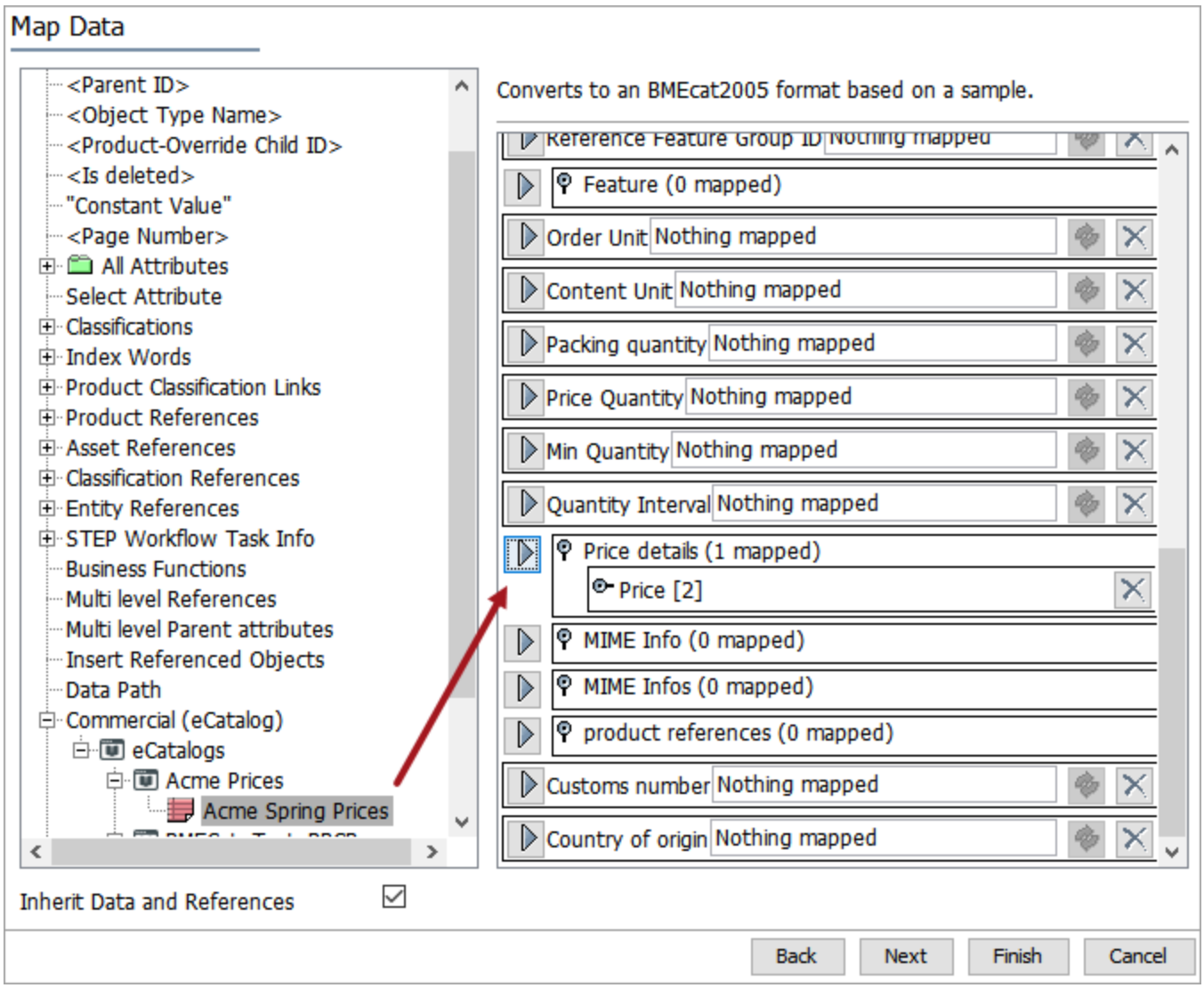
BMEcat 2005 is available for selection in:

- IIEP - refer to **Creating an Inbound Integration Endpoint**
- Import Manager - refer to **Creating a Data Import**
- Export Manager - refer to **Creating a Data Export**
- OIEP - refer to **Creating an Outbound Integration Endpoint**

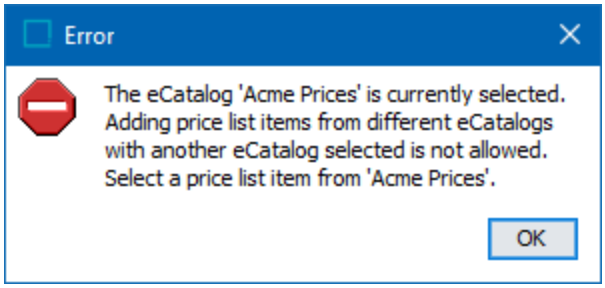
Mapping

This format requires creating a data map between STEP and the data being processed, and may also include data transformations. For details, refer to **Data Mapping**.

To map price data for eCatalogs in outbound integration endpoints (OIEPs) or in the Export Manager, first expand **Commercial (eCatalog)** as shown in the screenshot below. Next, expand the relevant eCatalog (in this example, 'Acme Prices'), then select the relevant price list. In this example, the price list is named 'Price details.' Then click the arrow icon next to **Price** in the conversion column to export the data associated with the commercial term into the BMEcat 2005 export.



A user can map multiple price lists from the same eCatalog, but cannot map price lists from multiple eCatalogs. If this is attempted, the system displays a warning message that looks like this:



For more information on configuring eCatalogs for export via BMEcat, refer to the 'Configure an eCatalog Configuration for BMEcat' section of the **Creating an eCatalog Configuration** topic in the **eCatalogs** documentation.

BMEcat 2005 and BMEcat 1.2 are the only formats that allow access to Product Attribute Link metadata attributes, via the transformation aspect 'Closest Attribute Link Meta Data.' For details, refer to **Aspect - Transform Outbound**.

Note: When using the auto map option in the BMEcat 2005 format to input eCl@ss or ETIM data, the auto map will initially try to match the attribute IDs in the import file with the IDs of attributes in the system. If no match is found, the auto map then adds the prefix 'eClass_ or 'ETIM_' to the input attribute's IDs that failed to match to attribute IDs in the system, and attempts to match the attribute IDs again. If no match is found, the user must manually map the attribute.

For more information on auto mapping, refer to the **Inbound Map Data - Auto Map** topic in the **Data Mapping** documentation.

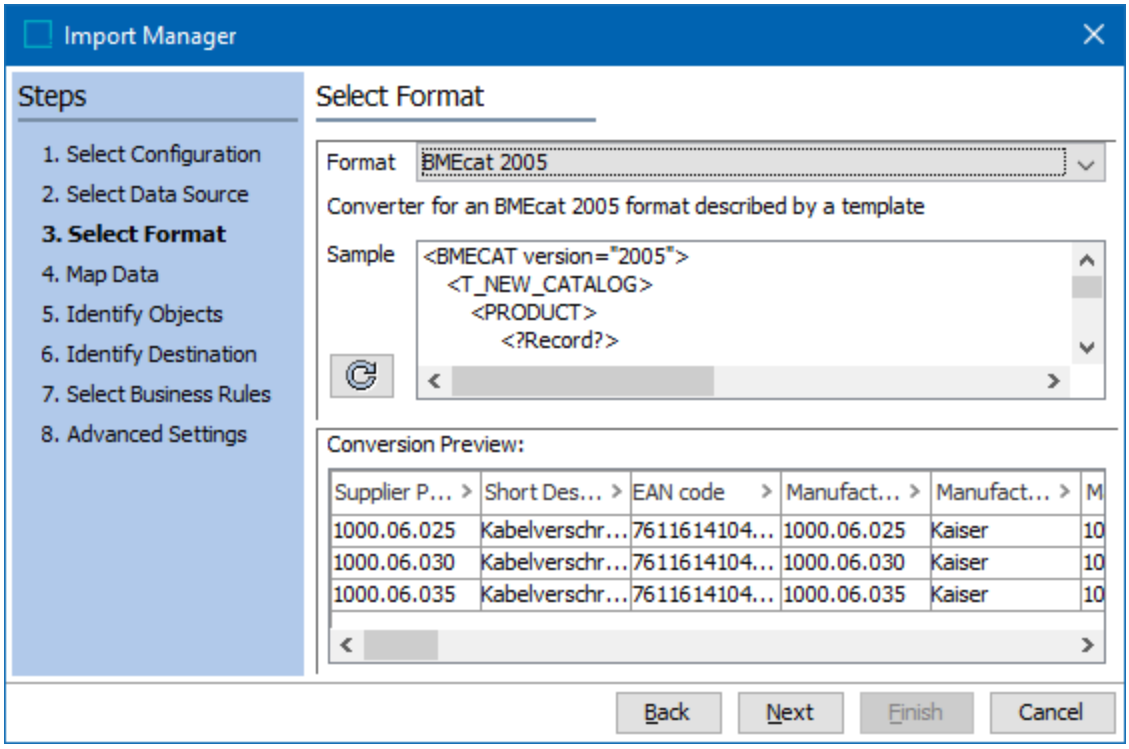
Inbound Data

Inbound Parameters

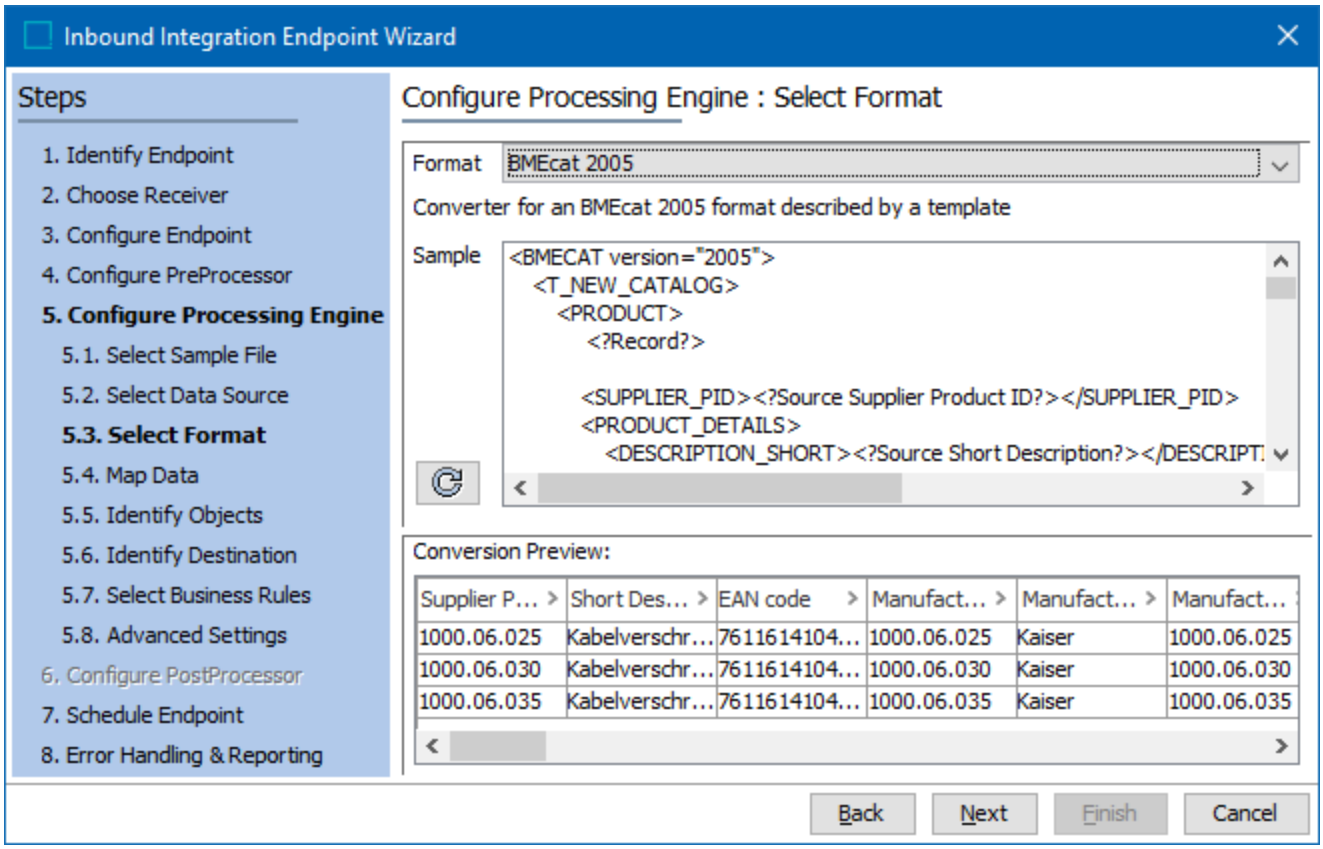
The following parameters are displayed on the inbound wizards:

- **Sample** is updated automatically when the format is selected.
- **Refresh** button updates the Conversion Preview pane.
- **Conversion Preview** displays data being imported.

Import Manager



IIEP



Outbound Data

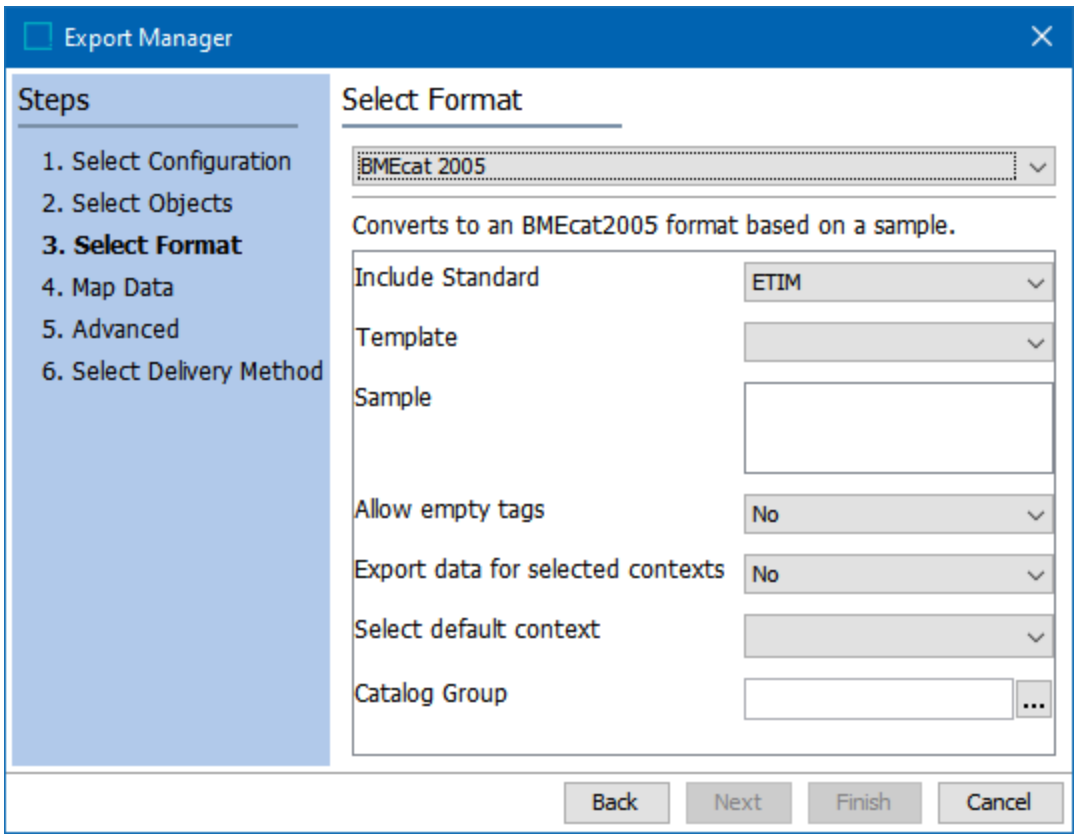
Exporting the BMEcat format involves supplying header data and catalog / product data via the parameters displayed on the Select Format step and during the Map Data step of the outbound wizards. Some information is required, as is indicated on the Map Data step. Although the export process can be run while mandatory fields are missing, the background process (BGP) will fail.

Important: In order to export using BMEcat 2005, the BMEcat component model must be configured first. Refer to the **BMEcat 2005 Component Model Configuration** topic in the **Data Exchange** documentation for more information.

For details on configuration of BMEcat 2005 for ETIM data, refer to the **BMEcat 2005 for ETIM Data Configuration** topic.

Outbound Parameters

The parameters that are available to fill out are controlled by the XML template that is uploaded. The fields listed below are ones that all users will see.



- In the **Include Standard** field, make a selection from the dropdown menu: ETIM, ECLASS, or General. If other standards have been configured, they will also display in the dropdown.
- **Template**: Depending on what was selected in the **Include Standard** parameter will dictate what templates are available to select from in the dropdown for template options. For example, if ETIM was selected as the Standard, only templates associated with the ETIM BMEcat template standard will display as option in the dropdown. Once selected, it will populate the 'Sample' field.
- The **Sample** field populates automatically depending on the **Template** selected.

  Note: For more information on Generic XML instructions, refer to the **Generic XML Outbound Processing Instructions** topic in the **Data Exchange** documentation

- For the **Allow empty tags** parameter, **Yes** indicates that export tags with empty values are included in the output. If set to **No**, tags containing empty values are not included in the export.

  For example, setting this parameter to Yes enables all ETIM attributes, even those without a value in STEP, to be exported. Additionally, when ETIM values exported via BMEcat are empty, they should be represented by a dash (-). For information on transforming an empty attribute value to a text string, refer to the **Replace the whole value** section in the **Transformations** topic of the **Resource Materials** documentation.

- The required **Export data for selected contexts** parameter allows you to output values for language dependent attributes, for example, the ETIM Short Description, Long Description, Keyword, and Remark values, and others. Select **Yes** in the dropdown to display the **Select Contexts** link. Click the link, select the required contexts for the export from the Select Contexts dialog, and click the **Select** button.

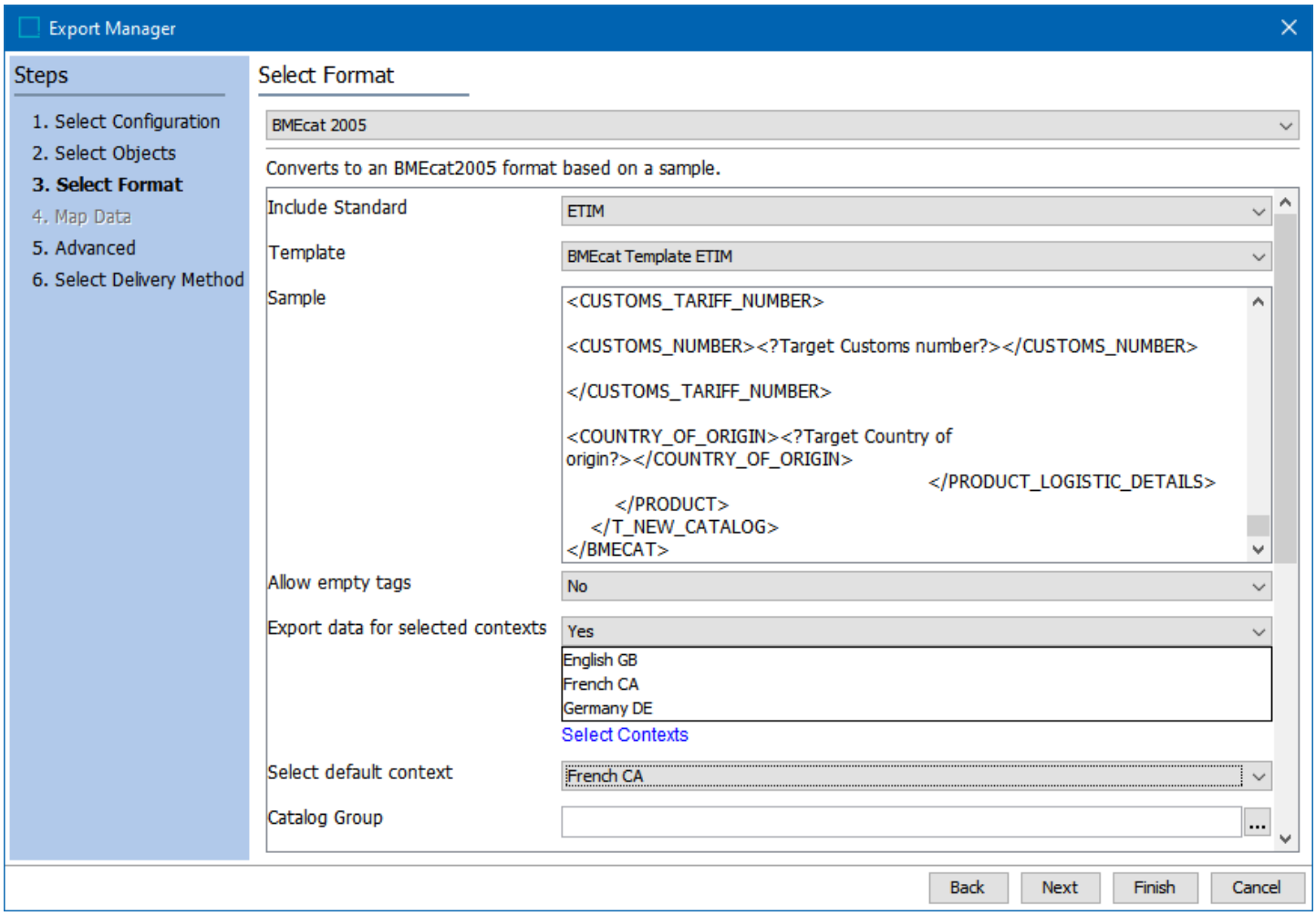
  The chosen contexts will be presented as <LANGUAGE> elements within the <HEADER> section of the exported XML file. If the system contains the specified context-dependent attribute values, they will be incorporated into the <PRODUCT_DETAILS> segment.

  Important: When using an OIEP for BMECat format export, ensure consistency between the contexts specified in the 'Export data for selected contexts' parameter and OIEP > Configuration tab > Configuration flipper > Contexts parameter. A mismatch may lead to contexts being presented differently in the exported file: those added in OIEP > Configuration tab > Configuration flipper > Contexts parameter will appear in the <PRODUCT_DETAILS> element, and those in 'Export data for selected contexts' parameter will be in the <HEADER> element.

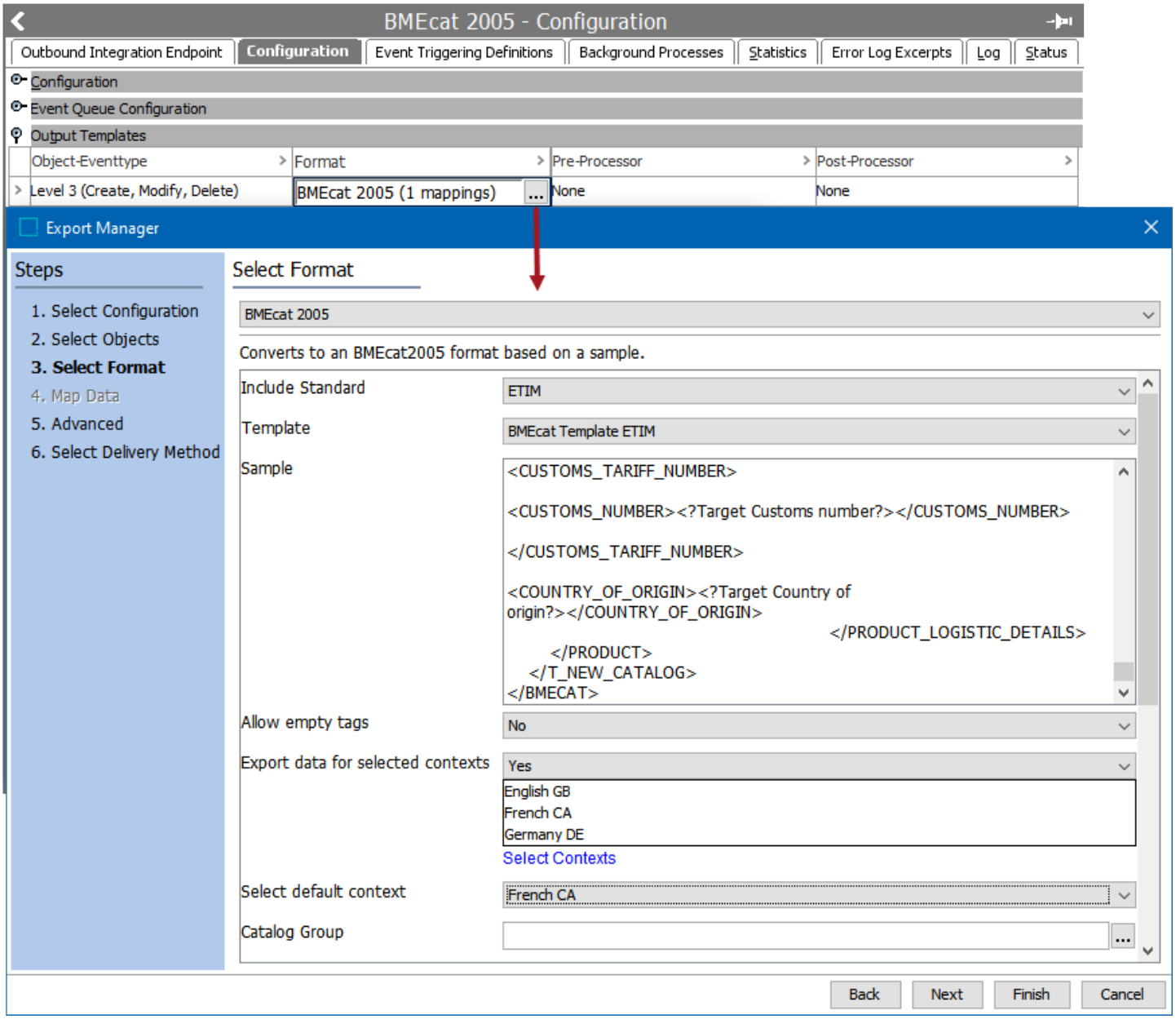
- The **Select default context** parameter shows the default context that is selected for the output. The parameter only displays contexts that were selected in the **Export data for selected contexts** parameter, and automatically selects the first context from the list. However, this can be changed by clicking the dropdown menu and selecting a different option.
- The **Catalog Group** parameter can be defined if a catalog group structure is needed.

For an explanation of additional parameters, search the web to find and refer to the 'Specification BMEcat® 2005' document. No validation is performed on the templates entered or the selections made, but if the Validate XML parameter displays and is = Yes, any errors will be reported in the execution report.

Export Manager



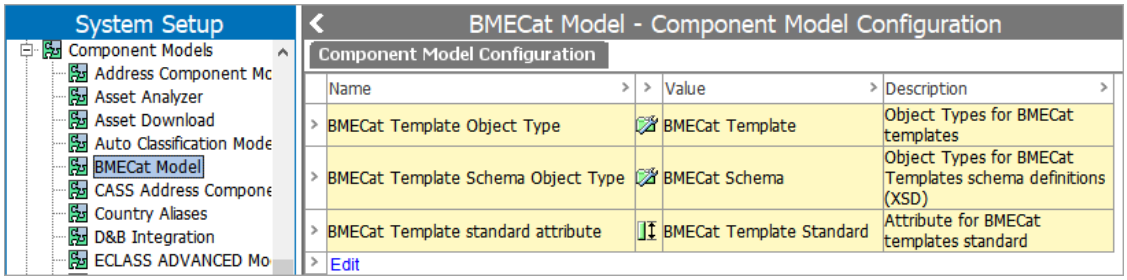
OIEP



Note: When using an OIEP for BMECat format export, there is a subtle distinction in how contexts are filled in the <HEADER> and <PRODUCT_DETAILS> elements. Specifically, for the context-dependent attribute values populated in the <PRODUCT_DETAILS> element, the values set in OIEP > Configuration tab > Configuration flipper > Contexts parameter take precedence and are populated accordingly. However, the contexts presented as <LANGUAGE> elements within the <HEADER> section are determined by the settings made in the above mentioned 'Export data for selected contexts' parameter itself.

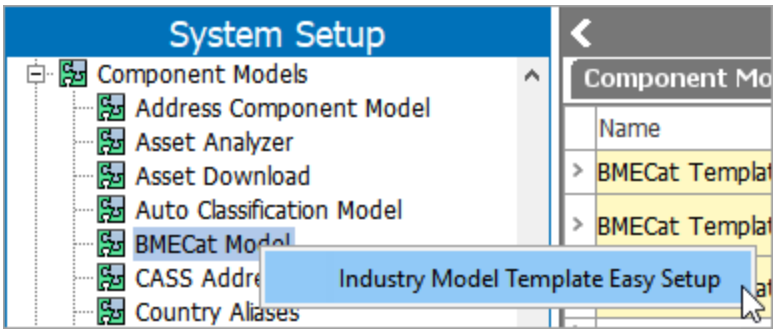
BMEcat 2005 Component Model Configuration

The BMEcat component model must be configured to use the BMEcat 2005 exporter. This allows users to control how they export configured data, and enables users to select the desired standard and template to support industry standards for ECLASS and ETIM.

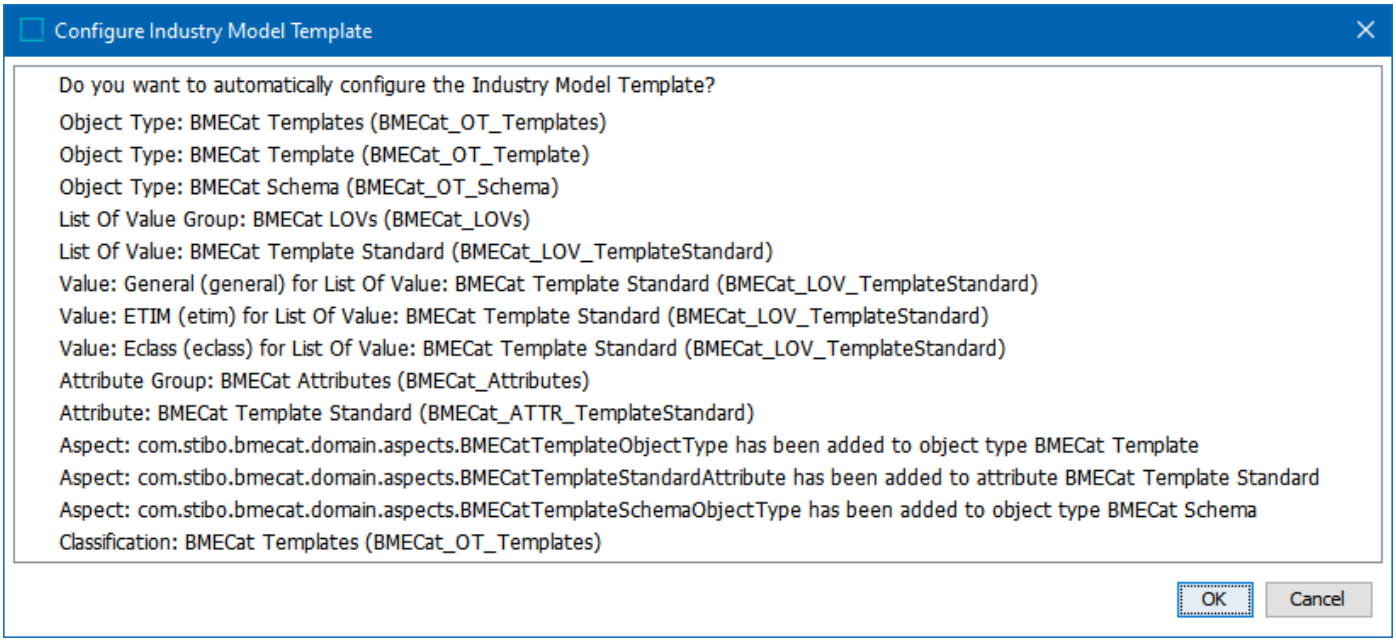


To configure the component model:

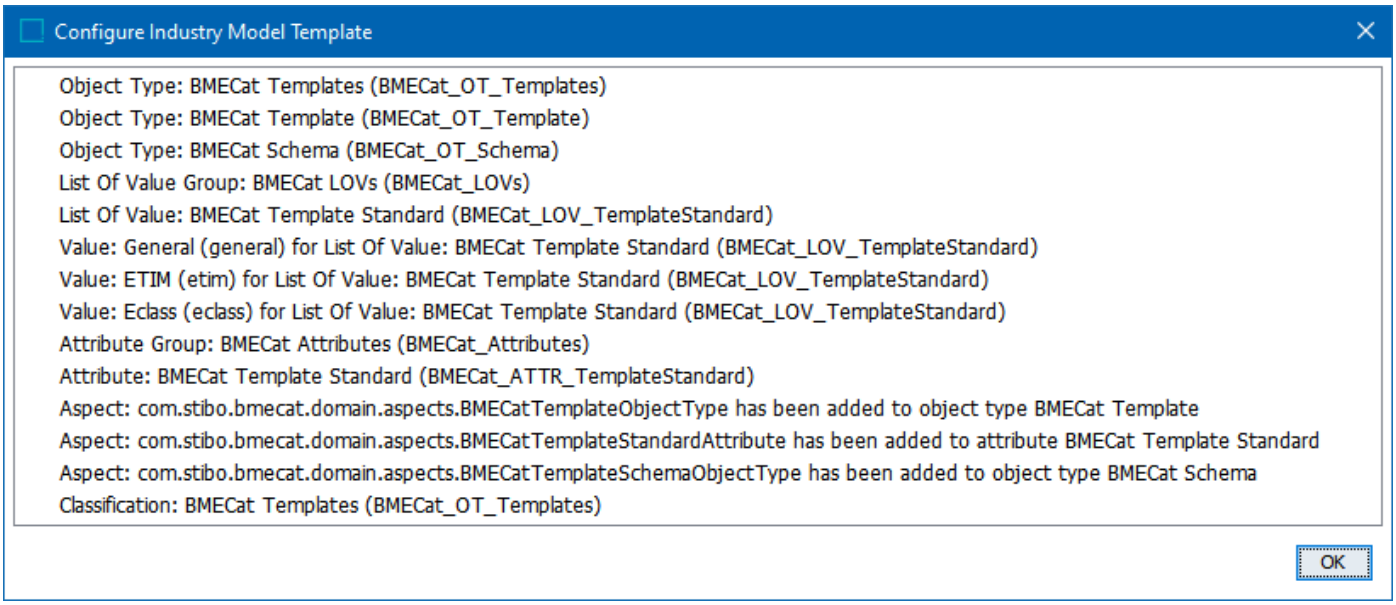
1. In the System Setup tab, under the 'Component Models' node, right-click on the **BMEcat Model** component model and select **Industry Model Template Easy Setup**.



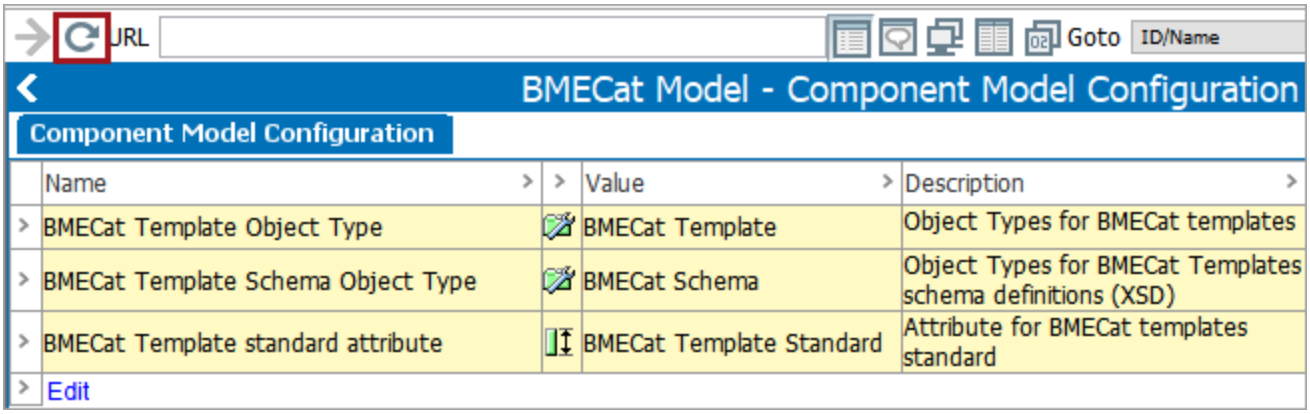
   Selecting **Industry Model Template Easy Setup** creates default values that are pre-defined. Once clicked, the **Configure Industry Model Template** dialog displays to explain what will be created. Press, **OK** to accept the default values.



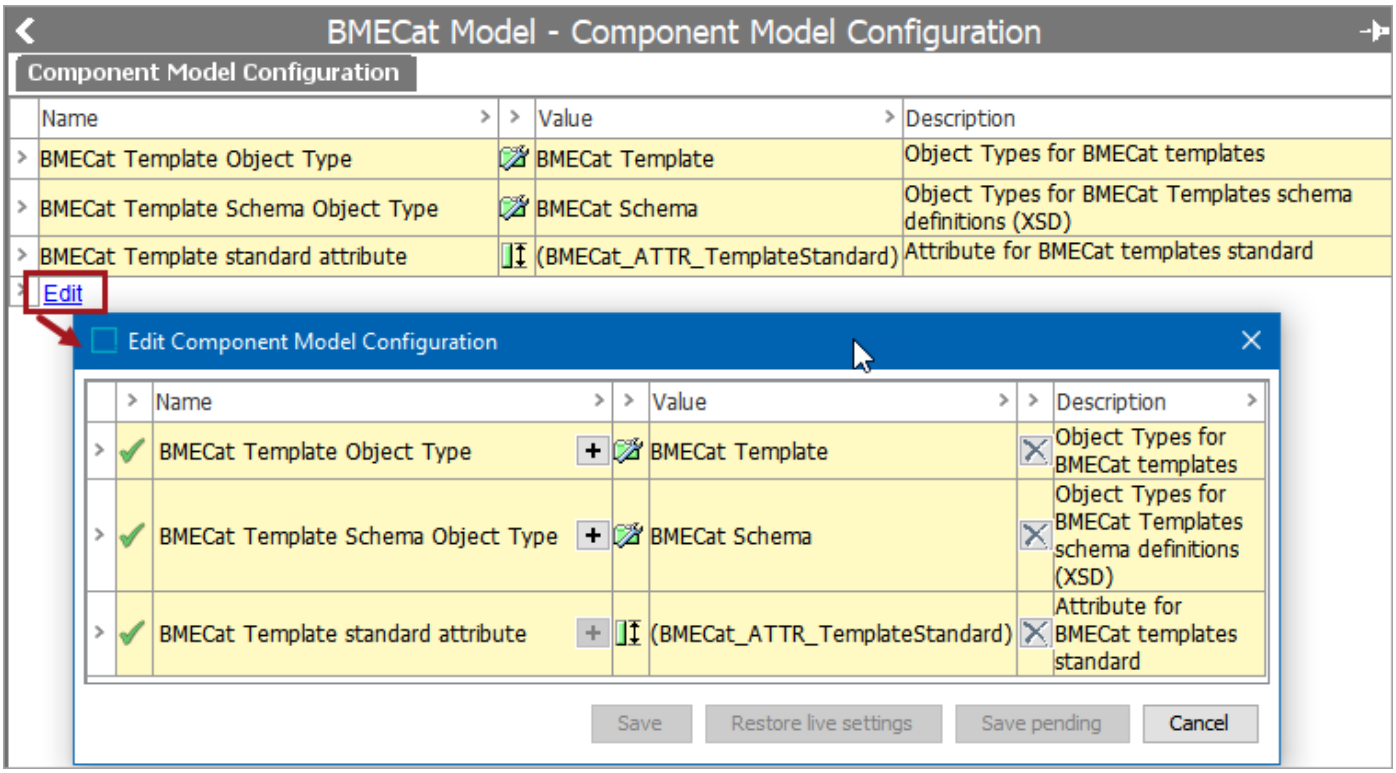
When completed successfully, a confirmation dialog displays. Click **OK** to close.



2. Click the **Refresh** button in workbench to see the default BMEcat values if they do not automatically display upon closing the **Configure Industry Model Template** confirmation dialog.



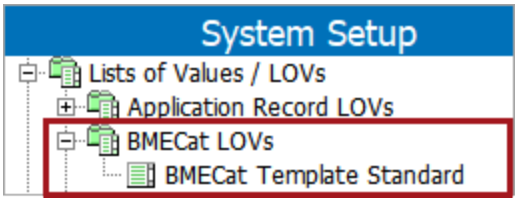
BMEcat Template Object Types are created as assets. If a user would like to add more object types, they can add them by clicking on the **Edit** button and adding the additional desired asset object type.



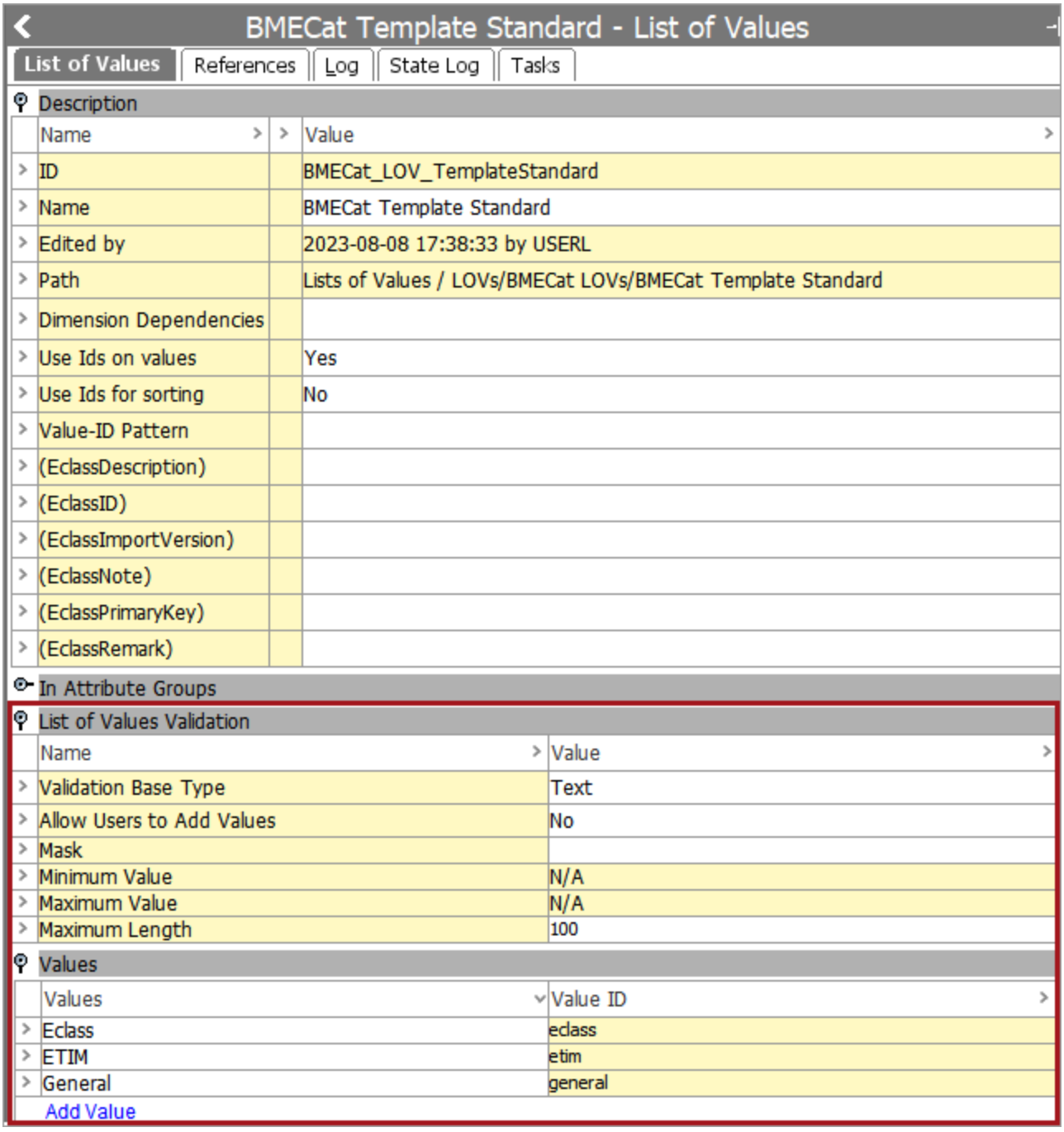
For more on creating assets, refer to the Assets topic in the Getting Started documentation.

Note: Although the BMEcat Template Schema Object Type displays in the component model, it is not configurable. Users should continue to contact their Professional Services or Stibo Systems representative for BMEcat Schema validation solution needs.

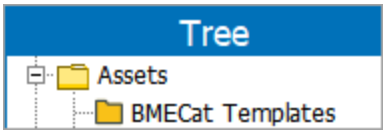
3. In the System Setup tab, under the LOV node, a **BMEcat LOVs** folder displays, which was created when selecting the **Industry Model Template Easy Setup** action. Expanding the folder displays the **BMEcat Template Standard** List Of Values.



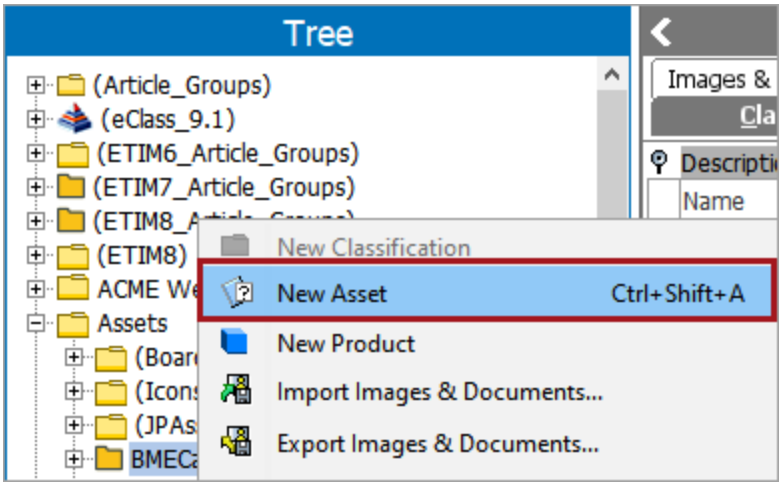
4. Click **BMEcat Template Standard** to display the List Of Values (LOV) information, which comes pre-configured with the needed information for the **List of Values Validation** and **Values** sections. This LOV contains the template standard types (ECLASS, ETIM, and General) that will be associated with the **BMEcat Template Object Type** assets created. If other template standard types are needed for selection outside of those offered, they can be added by clicking **Add Value** under the Values section.



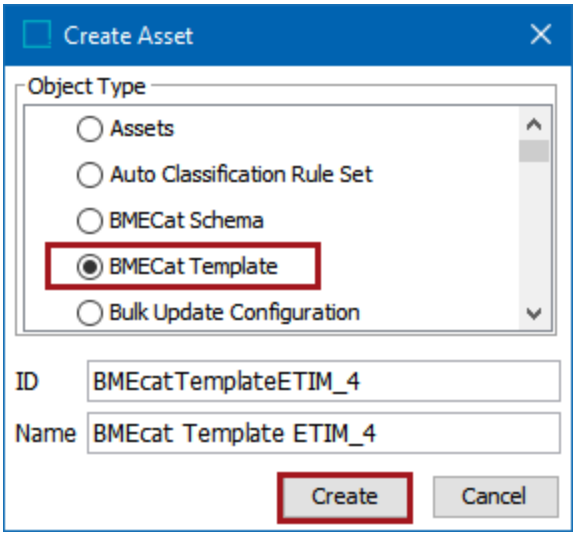
5. Next, in the Tree tab, under the Assets node, add any needed templates to the **BMEcat Templates** folder.



6. To upload templates, right-click on the **BMEcat Templates** folder, and select **New Asset**.

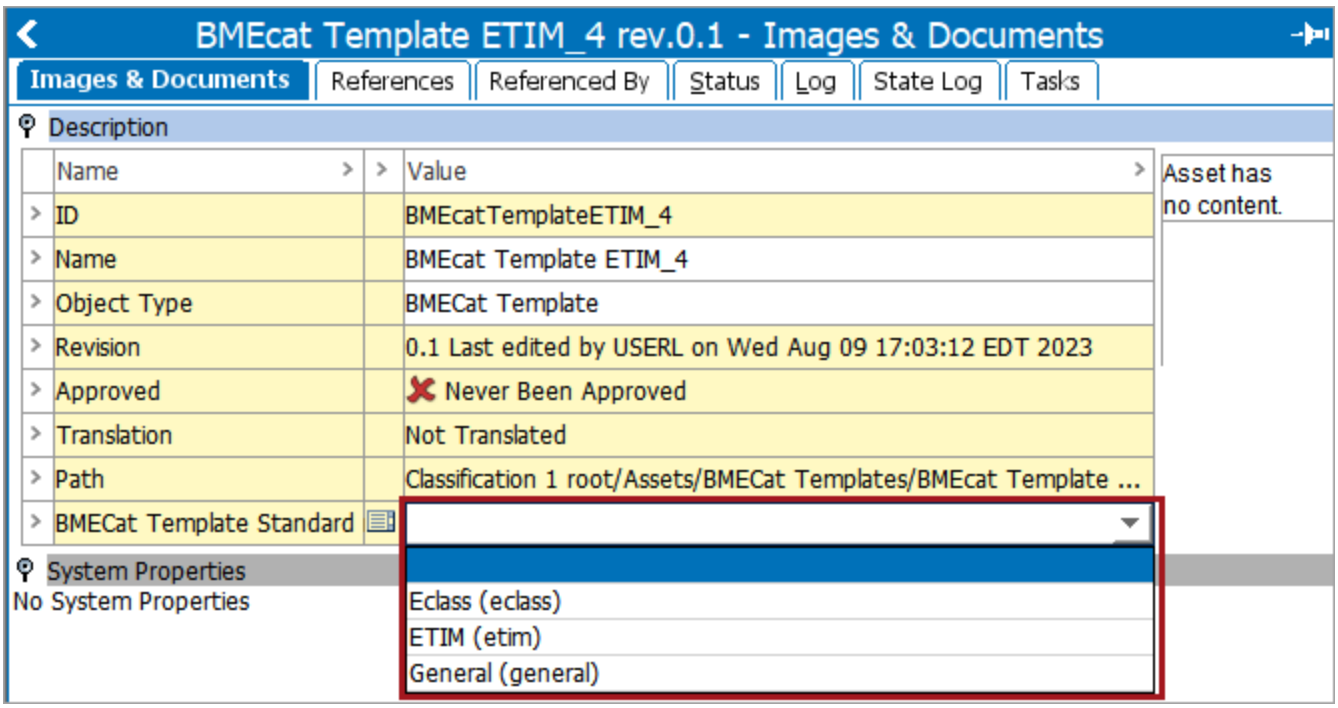


7. Select the **BMEcat Template** for the Object Type, and give an ID and Name to the template being created. Click **Create**.

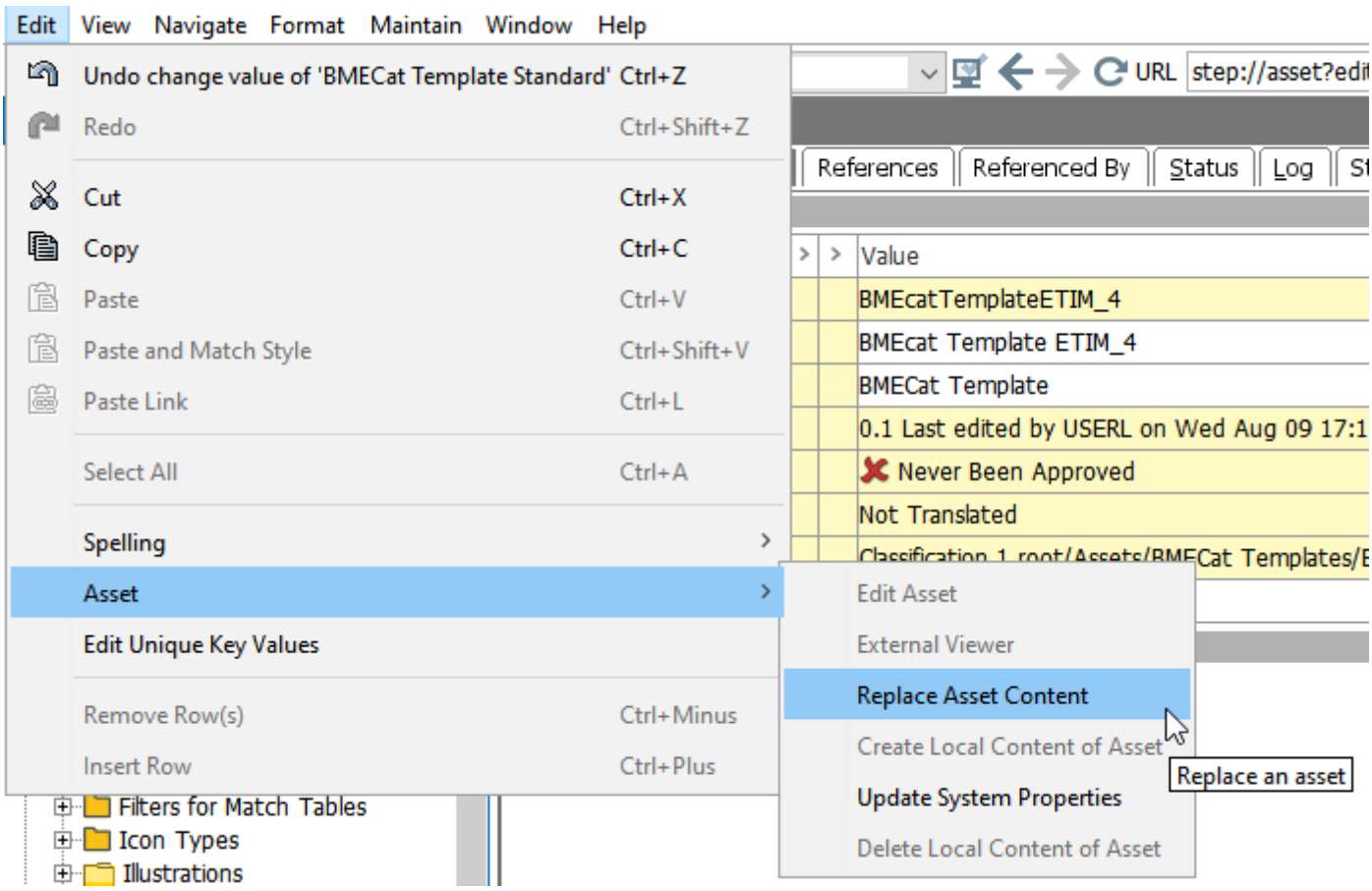


Note: Although the BMEcat Schema displays as a selectable option in the 'Create Asset' dialog, it is not configurable. Users should continue to contact their Professional Services or Stibo Systems representative for BMEcat Schema validation solution needs.

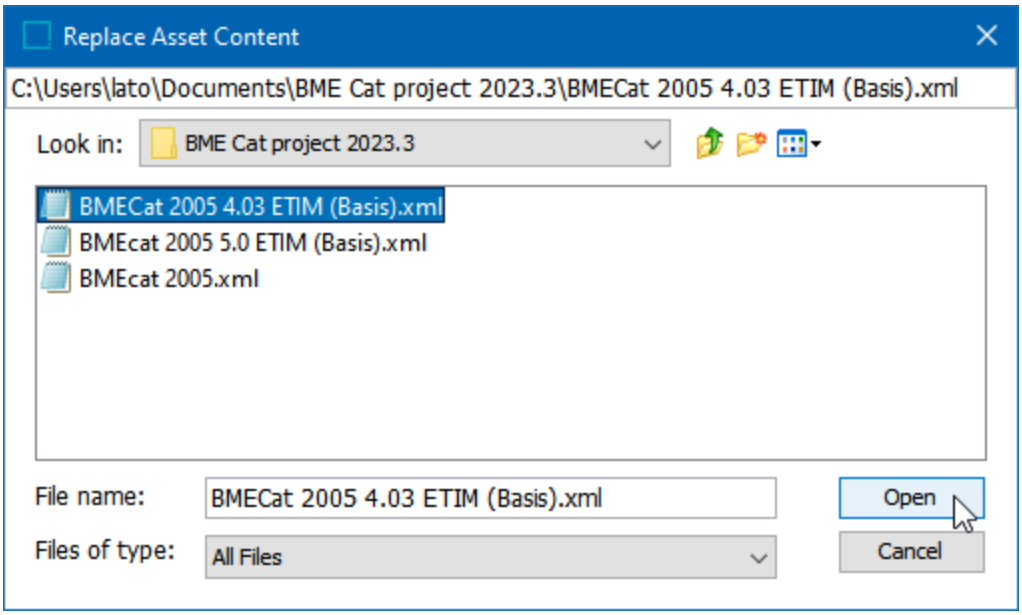
8. In the **BMEcat Template Standard** field, select from the dropdown the type of standard the newly created **BMEcat Template** should be associated with.



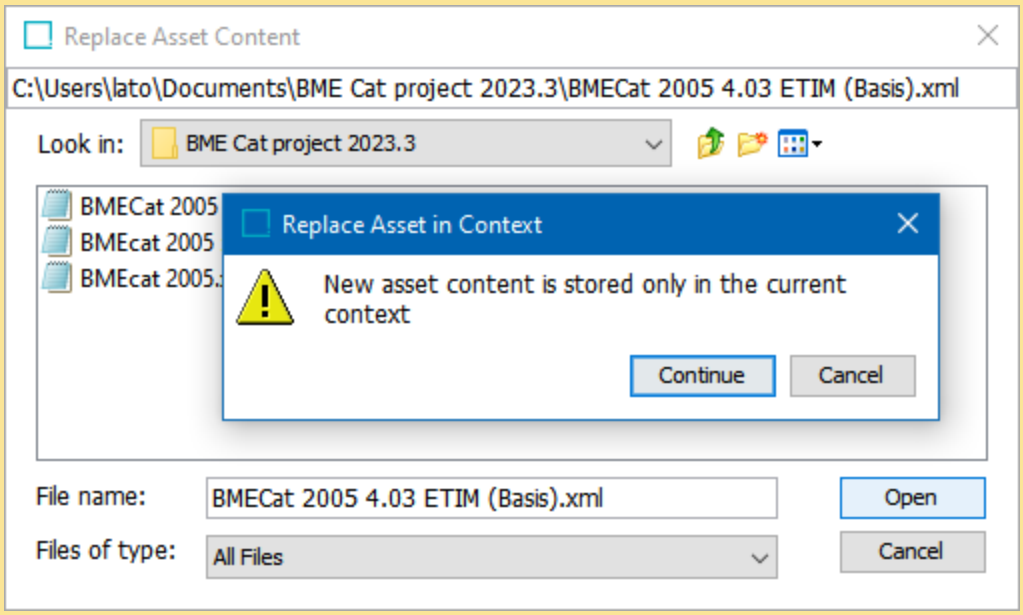
9. To add content to the BMEcat Template, click Edit > Asset > **Replace Asset Content**.



10. Select the desired template, and click **Open**.



Important: BMEcat Templates are only stored in the current context that they are being uploaded in.



Displayed below is an example of a General template for BMEcat 2005:

```
<BMECAT version="2005" xmlns="http://www.bmecat.org/bmecat/2005fd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.bmecat.org/bmecat/2005fd bmecat_2005.xsd">

    <HEADER>
        <CATALOG>
            <LANGUAGE><?Parameter Language?></LANGUAGE>
            <CATALOG_ID>Industry_EGH_<?ExportDate YYMM?></CATALOG_ID>
            <CATALOG_VERSION><?Parameter Catalog version?></CATALOG_VERSION>
            <CATALOG_NAME>Acme Industry EGH_<?ExportDate MM-YYYY?></CATALOG_NAME>
            <DATETIME type="generation_date">
                <DATE><?ExportDate?></DATE>
            </DATETIME>
            <TERRITORY><?Parameter Territory?></TERRITORY>
            <CURRENCY><?Parameter Currency?></CURRENCY>
```

```
<MIME_ROOT><?Parameter MIME root directory?></MIME_ROOT>
        </CATALOG>
        <BUYER>
            <BUYER_ID type="buyer_specific"><?Parameter Buyer ID?></BUYER_ID>
            <BUYER_NAME><?Parameter Buyer Name?></BUYER_NAME>
        </BUYER>
        <SUPPLIER>
            <SUPPLIER_ID type="duns">331716014</SUPPLIER_ID>
            <SUPPLIER_NAME><?Parameter Supplier name?></SUPPLIER_NAME>
            <ADDRESS type="supplier">
<!-- NEW TAG --><CONTACT><?Parameter Contact name?></CONTACT>
<!-- NEW TAG --><STREET><?Parameter Street?></STREET>
<!-- NEW TAG --><ZIP><?Parameter Zip code?></ZIP>
<!-- NEW TAG --><CITY><?Parameter Town or city?></CITY>
<!-- NEW TAG --><COUNTRY><?Parameter Country?></COUNTRY>
<!-- NEW TAG --><VAT_ID><?Parameter VAT-ID?></VAT_ID>
                <EMAIL><?Parameter E-mail?></EMAIL>
<!-- NEW TAG --><URL><?Parameter Internet address?></URL>
                <ADDRESS_REMARKS><?Parameter Address remarks?></ADDRESS_REMARKS>
```

```
            </ADDRESS>
        </SUPPLIER>
    <USER_DEFINED_EXTENSIONS>
            <!--<UDX.EDXF.VERSION><?Parameter UDX version?></UDX.EDXF.VERSION>-->
        </USER_DEFINED_EXTENSIONS>
    </HEADER>
    <T_NEW_CATALOG>
        <PRODUCT mode="new">
            <?Product?>

            <SUPPLIER_PID><?Target Supplier Product ID?></SUPPLIER_PID>
            <PRODUCT_DETAILS>
                <DESCRIPTION_SHORT lang="[?Target lang?]"><?MultiTarget Short
Description?><?Target Value?></DESCRIPTION_SHORT>
                <DESCRIPTION_LONG lang="[?Target lang?]"><?MultiTarget Long
Description?><?Target Value?></DESCRIPTION_LONG>
<INTERNATIONAL_PID type="[?Target type?]"><?MultiTarget International product
number?><?Target Value?></INTERNATIONAL_PID>
                <SUPPLIER_ALT_PID><?Target Supplier Alternative Product ID?></SUPPLIER_
ALT_PID>
             <!--   <BUYER_PID type="buyer_specific"><?Target Buyer Product ID?></BUYER_
PID> -->  <!--It is actually Multy, but for different types!-->
                <MANUFACTURER_PID><?Target Manufacturer Product ID?></MANUFACTURER_PID>

<MANUFACTURER_NAME><?Target Manufacturer Name?></MANUFACTURER_NAME>
<MANUFACTURER_TYPE_DESCR lang="[?Target lang?]"><?MultiTarget Manufacturer Type
```

```
    Description?><?Target Value?></MANUFACTURER_TYPE_DESCR>

                    <ERP_GROUP_BUYER><?Target Buyer ERP Product Group?></ERP_GROUP_BUYER>

                    <ERP_GROUP_SUPPLIER><?Target Supplier ERP Product Group?></ERP_GROUP_
    SUPPLIER>

                    <DELIVERY_TIME><?Target Delivery Time?></DELIVERY_TIME>



                    <SPECIAL_TREATMENT_CLASS type="[?Target type?]"><?MultiTarget Special
    Treatment Class?><?Target Value?></SPECIAL_TREATMENT_CLASS>
                    <KEYWORD lang="[?Target lang?]"><?MultiTarget KEYWORD?><?Target
    Value?></KEYWORD>

    <!--<KEYWORD><?MultiTarget Keywords?></KEYWORD>-->

                    <REMARKS lang="[?Target lang?]"><?MultiTarget Remarks?><?Target
    Value?></REMARKS>
                    <SEGMENT lang="[?Target lang?]"><?MultiTarget Segment?><?Target
    Value?></SEGMENT>

                    <PRODUCT_ORDER><?Target Product Order?></PRODUCT_ORDER>

                    <PRODUCT_STATUS lang="[?Target lang?]" type="bargain"><?MultiTarget
    Status?><?Target Value?></PRODUCT_STATUS>



                </PRODUCT_DETAILS>

                <PRODUCT_FEATURES>

    <REFERENCE_FEATURE_SYSTEM_NAME><?Target Reference Feature System Name?></REFERENCE_
    FEATURE_SYSTEM_NAME>
                    <REFERENCE_FEATURE_GROUP_NAME><?Target Reference Feature Group
    Name?></REFERENCE_FEATURE_GROUP_NAME>
                    <REFERENCE_FEATURE_GROUP_ID><?Target Reference Feature Group
    ID?></REFERENCE_FEATURE_GROUP_ID>



                    <FEATURE>

                        <?MultiTarget Feature?>
```

```
                    <FNAME><?Target Name?></FNAME>
                    <FVALUE lang="[?Target lang?]"><?MultiTargetSplitIfRange
FVALUE?><?Target Value?></FVALUE>
                    <FUNIT><?Target Unit?></FUNIT>
                    <FORDER><?Target Order?></FORDER>
                    <FDESCR><?Target Description?></FDESCR>
                    <FVALUE_DETAILS><?Target Value Details?></FVALUE_DETAILS>
                </FEATURE>
            </PRODUCT_FEATURES>
            <PRODUCT_ORDER_DETAILS>
                <ORDER_UNIT><?Target Order Unit?></ORDER_UNIT>
                <CONTENT_UNIT><?Target Content Unit?></CONTENT_UNIT>
                <NO_CU_PER_OU><?Target Packing quantity?></NO_CU_PER_OU>
                <PRICE_QUANTITY><?Target Price Quantity?></PRICE_QUANTITY>
                <QUANTITY_MIN><?Target Min Quantity?></QUANTITY_MIN>
                <QUANTITY_INTERVAL><?Target Quantity Interval?></QUANTITY_INTERVAL>
            </PRODUCT_ORDER_DETAILS>

                    <PRODUCT_PRICE_DETAILS>
                <?MultiTarget Price details?>
<!-- NEW TAG --><DATETIME type="valid_start_date"><DATE><?Target Datetime (Valid start
```

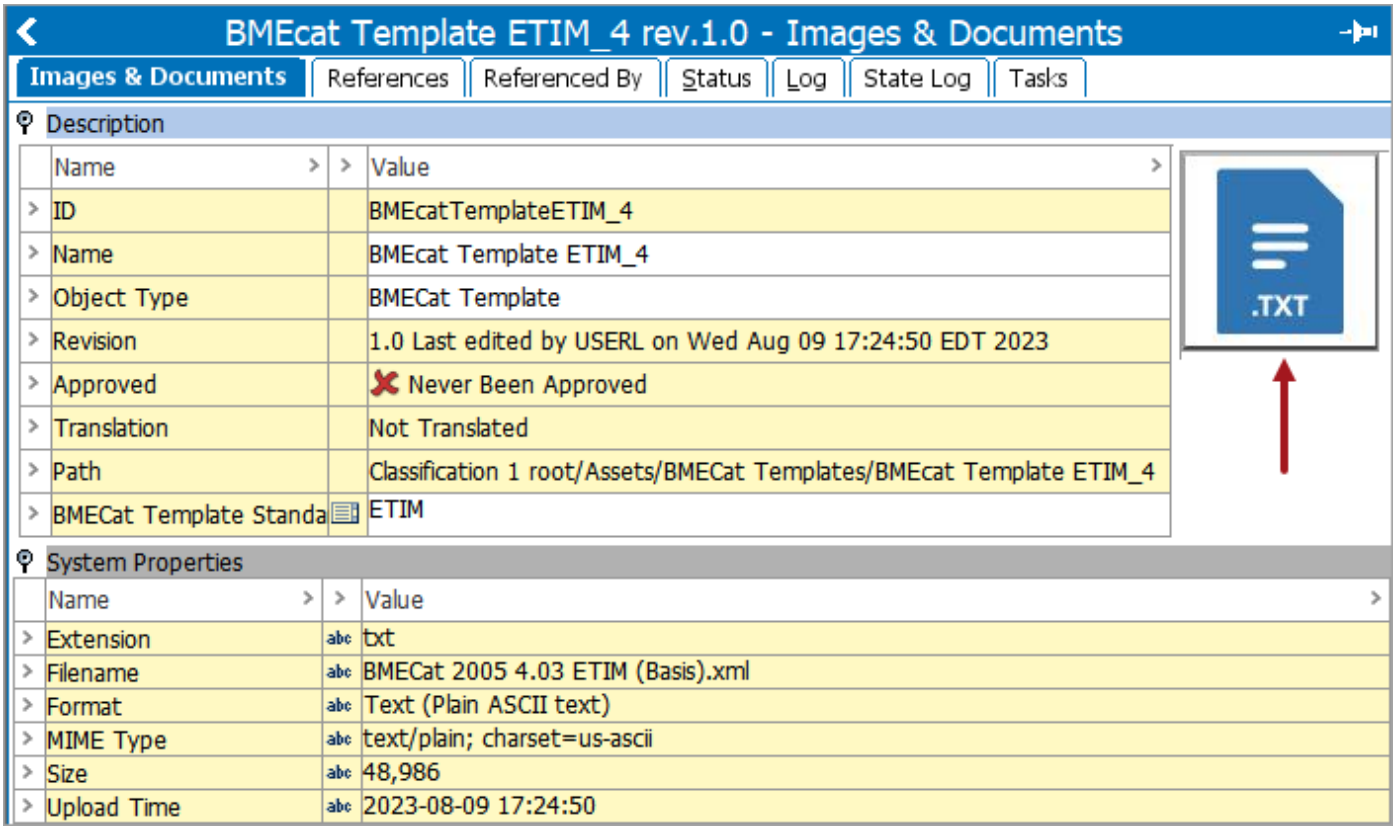
```
    date)?></DATE></DATETIME>
    <!-- NEW TAG <DATETIME type="valid_end_date"><DATE><?Target Datetime (Valid end
    date)?></DATE></DATETIME>-->

                <VALID_START_DATE><?Target Price start date?></VALID_START_DATE>

                <VALID_END_DATE><?Target Price end date?></VALID_END_DATE>

                <DAILY_PRICE><?Target Daily price?></DAILY_PRICE>

                <PRODUCT_PRICE price_type="net_list">

                    <PRICE_AMOUNT><?Target Price Amount?></PRICE_AMOUNT>

                    <PRICE_CURRENCY><?Target Currency?></PRICE_CURRENCY>

                    <TAX><?Target Tax rate?></TAX>

                    <PRICE_FACTOR><?Target Price Factor?></PRICE_FACTOR>

                    <LOWER_BOUND><?Target Lower quantity limit?></LOWER_BOUND>

                    <TERRITORY><?Target Territory?></TERRITORY>

                </PRODUCT_PRICE>

            </PRODUCT_PRICE_DETAILS>



            <MIME_INFO>
```

```
                    <MIME>

                        <?MultiTarget MIME Info?>

                        <MIME_TYPE><?Target MIME Type?></MIME_TYPE>

                        <MIME_SOURCE><?Target Source?></MIME_SOURCE>

                        <MIME_DESCR><?Target Description?></MIME_DESCR>

                        <MIME_PURPOSE><?Target Purpose?></MIME_PURPOSE>

                    </MIME>

                </MIME_INFO>




                    <UDX.EDXF.MIME_INFO>

    <UDX.EDXF.MIME>

    <?MultiTarget MIME Infos?>

    <UDX.EDXF.MIME_SOURCE lang="[?Target lang?]"><?MultiTarget Source?><?Target
    Source?></UDX.EDXF.MIME_SOURCE>

    <UDX.EDXF.MIME_CODE lang="[?Target lang?]"><?MultiTarget Code?><?Target
    Code?></UDX.EDXF.MIME_CODE>
```

```
<UDX.EDXF.MIME_FILENAME lang="[?Target lang?]"><?MultiTarget Filename?><?Target
Filename?></UDX.EDXF.MIME_FILENAME>

<UDX.EDXF.MIME_DESIGNATION lang="[?Target lang?]"><?MultiTarget Description?><?Target
Description?></UDX.EDXF.MIME_DESIGNATION>

<UDX.EDXF.MIME_ALT lang="[?Target lang?]"><?MultiTarget Alternative text?><?Target
Alternative text?></UDX.EDXF.MIME_ALT>

<UDX.EDXF.MIME_ISSUE_DATE><?MultiTarget Issue date?></UDX.EDXF.MIME_ISSUE_DATE>

<UDX.EDXF.MIME_EXPIRY_DATE><?MultiTarget Expiry date?></UDX.EDXF.MIME_EXPIRY_DATE>

</UDX.EDXF.MIME>

</UDX.EDXF.MIME_INFO>



            <PRODUCT_REFERENCE type="[?Target type?]">

<?MultiTarget product references?>

                <PROD_ID_TO><?Target Reference product?></PROD_ID_TO>

                <CATALOG_ID><?Target Catalog ID?></CATALOG_ID>

                <CATALOG_VERSION><?Target Catalog Version?></CATALOG_VERSION>

<REFERENCE_DESCR><?Target Reference description?></REFERENCE_DESCR>
```

```
                </PRODUCT_REFERENCE>

    <PRODUCT_LOGISTIC_DETAILS>

    <CUSTOMS_TARIFF_NUMBER>

    <CUSTOMS_NUMBER><?Target Customs number?></CUSTOMS_NUMBER>

    </CUSTOMS_TARIFF_NUMBER>

    <COUNTRY_OF_ORIGIN><?Target Country of origin?></COUNTRY_OF_ORIGIN>

    </PRODUCT_LOGISTIC_DETAILS>

          </PRODUCT>

      </T_NEW_CATALOG>

    </BMECAT>
```

11. When properly uploaded, the template will display.



Should changes need to be made to the template after upload, double-clicking on the icon will open the template for edits.

For examples of BMEcat 2005 being used for ETIM, refer to the BMEcat 2005 for ETIM Data Configuration topic in the Data Exchange documentation.

BMEcat 2005 for ETIM Data Configuration

When using the BMEcat 2005 format to export ETIM data, the following configuration options and mapping data source are available:

Prerequisites

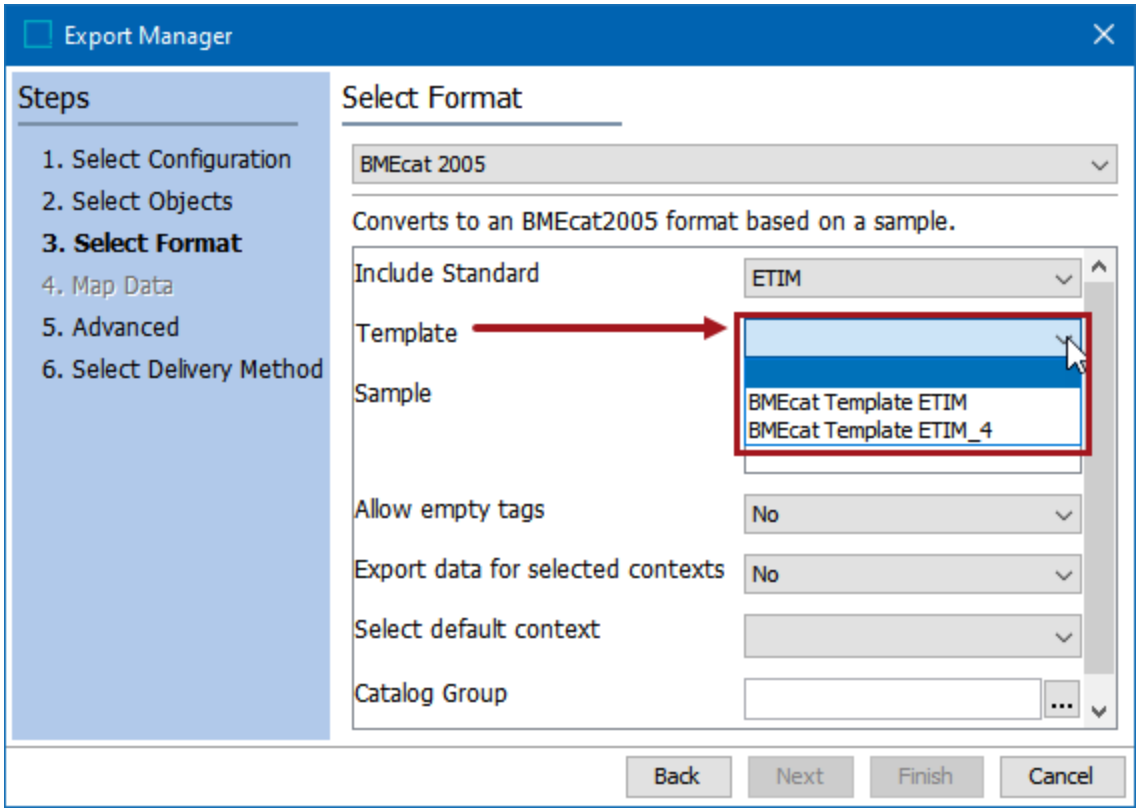
The BMEcat component model must first be configured. For more information, refer to the **BMEcat 2005 Component Model Configuration** topic in the **Data Exchange** documentation.

Configuring the Export

To export ETIM in BMEcat 2005:

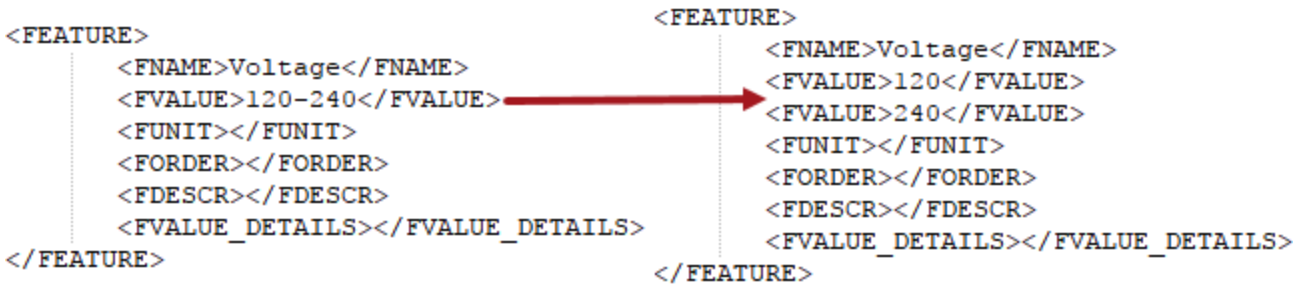
1. For the **Include Standard** parameter, select **ETIM**.
2. For the **Template** parameter, select the desired ETIM template.

> **Note:** The needed templates must already be uploaded for selection in the dropdown menu. For directions on how to do this, refer to the **BMEcat 2005 Component Model Configuration** topic in the **Data Exchange** documentation.



Using BMEcat 2005 to export ETIM IXF Format classification data, as defined in the **ETIM IXF Format** topic, includes the following instructions:

- **MultiTargetSplitIfRange** - This standard General XML instruction allows two FVALUES to be output when an attribute with the Validation Base Type of 'Number Range' is mapped. Instead of exporting 10-15 as a single value, for example, the individual values of '10' and then '15' are exported in separate FVALUE tags.



- **Lang** - The 'lang' target appears under the FEATURE > FVALUE target and when used with the 'Language Dimension' aspect, context information is output. The 'Replace X with Y' transformation can be used to provide ETIM-accepted three-character language codes. Use of the 'Validate XML' option on the Select Format step also provides validation of language codes.

> **Note:** If the data being exported does not include context data, remove the 'lang' target from the Generic XML format sample, which removes it from the list of available targets.

Refer to the online help version of this topic for two example templates.

Attributes Inherited from Classification

The export data source 'Attributes Inherited from Classification' allows mapping attributes inherited from a classification to a product being exported using the format is BMEcat 2005 or FAB-DIS. For details, refer to the **Attributes Inherited from Classification - Data Source Outbound** topic in the **Data Mapping** documentation.

BMEcat 2005.1 Format

BMEcat® 2005.1 is supported in STEP, is only available as XML based format, and uses ECLASS Advanced data. ECLASS Advanced describes the individual products in the catalog, and provides a more intricate classification structure suited for advanced applications like engineering, IoT, digital twin, and particularly for handling complex products. For more information about the standard, search the web.

The ECLASS Advanced commercial license is required to use BMEcat 2005.1 functionality. Contact your account manager or partner manager to begin the process of enabling a license or licenses for your system.

For more information on importing and exporting using BMEcat 2005.1, refer to the following topics in the Solution Enablement: ECLASS Advanced documentation:

- ECLASS Advanced Data Importer
- Exporting in BMEcat 2005.1 Format

For more information on ECLASS Advanced, refer to the ECLASS Quick Start Guide Introduction topic in the Solution Enablement: ECLASS Advanced documentation.

CSV Format

The comma-separated values (CSV) file format is available for inbound and outbound data exchanges. Using CSV requires creating a data map between STEP and the data being processed, and may also include data transformations.

The following sample CSV data import file shows that the first row is a header, and the remaining data is delimited by a comma. Exported CSV files always include a header row, though this is optional for imports.

```
<Name>,<Parent ID>,Primary Color,Secondary Color
Mens T PBO,18209,Black,Orange
Mens T PBG,18209,Blue,Green
Mens T PGS,18209,Green,Silver
Mens T PGW,18209,Gray,White
Mens T POY,18209,Orange,Yellow
```

Keep the following points in mind when working with STEP data using CSV format:

- The following node types / super types can be imported and exported via CSV format: products, classifications, entities, assets (objects, not content), and attributes.
- A CSV import or export file will include data in the same arrangement as relational database tables. This means that each object is displayed as a single row in the file and each object property item is displayed as a single column.
- References and/or data containers can be exported in this format where multiple values are separated by a delimiter. References are separated by semicolons, while data containers use a pound sign (#) and the common prefix of the data container for each related attribute column.
- Imports and exports are context and workspace specific. By default, data is imported to or extracted from the context and workspace in use when the process is started.
- When planning to import data back into STEP, include STEP ID in the export.

Format Availability

CSV is available for selection in:

- IIEP - refer to **Creating an Inbound Integration Endpoint**
- Import Manager - refer to **Creating a Data Import**
- Export Manager - refer to **Creating a Data Export**
- OIEP - refer to **Creating an Outbound Integration Endpoint**

Mapping

This format requires creating a data map between STEP and the data being processed, and may also include data transformations. For details, refer to **Data Mapping**.

Inbound Data

CSV import allows creation of and updates to products, classifications, entities, attributes, and references. However, system setup objects (for example: LOVs, users, reference types, and so on), cannot be created or modified via import.

Because the Map Data process allows selection of only a single node type, only one node type / super type (products, entities, etc.) can be imported at a time. When multiple super types exist in the same import file, a separate import is required to successfully import each type of object, starting with classification data, then product data, and finally, entity data. When the inbound file includes data for node types other than the one selected, two things may happen: 1) assuming none of the data prevents the import, new objects are created using the supplied information and the selected super type, 2) the execution report details the skipped records when included data, like parent ID, is not found in the selected super type hierarchy. Alternately, split the inbound data file by super type and process accordingly.

For information on parallel imports involving multiple references on object types, refer to **Reference Target Lock Policy on Object Types**.

Deleting Values During Import

When importing comma and tab delimited files, the values are imported exactly as provided in the file per the mapped data. This means that when a blank value is imported, an attribute that previously had a value is overwritten as blank. If the value being deleted was inherited, the result is not a blank field, but the inherited value is restored.

This functionality is the same as when importing STEPXML files but differs from imports of Excel.

Inbound Parameters

The following parameters are available in both Import Manager and IIEP.

1. **Delimiter** - determines the character used to delimit the exported data. Options are semi-colon (;), comma (,), the tab character, the pipe symbol (|), or colon (:).
   - **Tab** is recommended to avoid splitting the data on printable characters that may occur within the values.
   - Only the delimiters in the dropdown are allowed. If the file uses a different delimiter, before importing, first update the file outside of STEP to use a delimiter allowed by STEP.
2. **Character Set** - determines the characters that can be successfully imported. Options include Windows-1252, ISO-8859-1 (also known as the Latin-1 character set), UTF-8, or UTF-16.
   - **UTF-8** is recommended unless you have a reason to do otherwise.
   - **UTF-8** or **UTF-16** (a Unicode character set) is required to correctly import special symbols, like a trademark symbol.
   - **Shift-JIS** character set used for the Japanese language.
3. **Has Header** - indicates if the file has a header row.
   - **Yes** indicates that the first line (row) of the CSV file has header information, such as attribute names that match STEP, the Auto Map feature is available to map the columns of data to the appropriate object in

STEP.

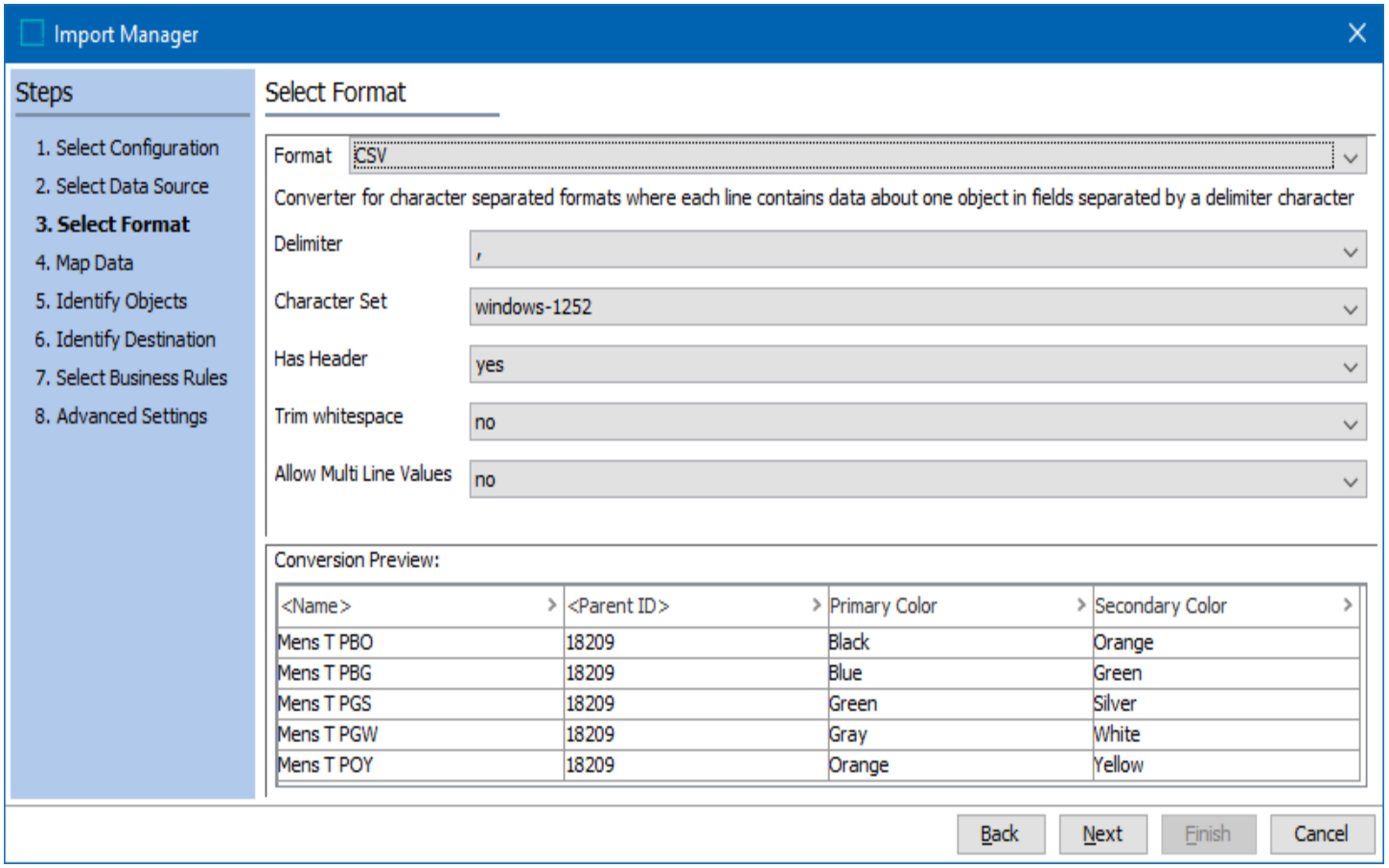
- **No** indicates that the first line (row) of the CSV file has actual data and no labels are included to identify data. Although a header row is not required, without it the user must be able to identify the data for manual mapping.

4. **Trim whitespace** - determines how blank spaces before or after the value are handled.
   - **Yes** removes leading and trailing spaces in imported values.
   - **No** leaves leading and trailing spaces in imported values.

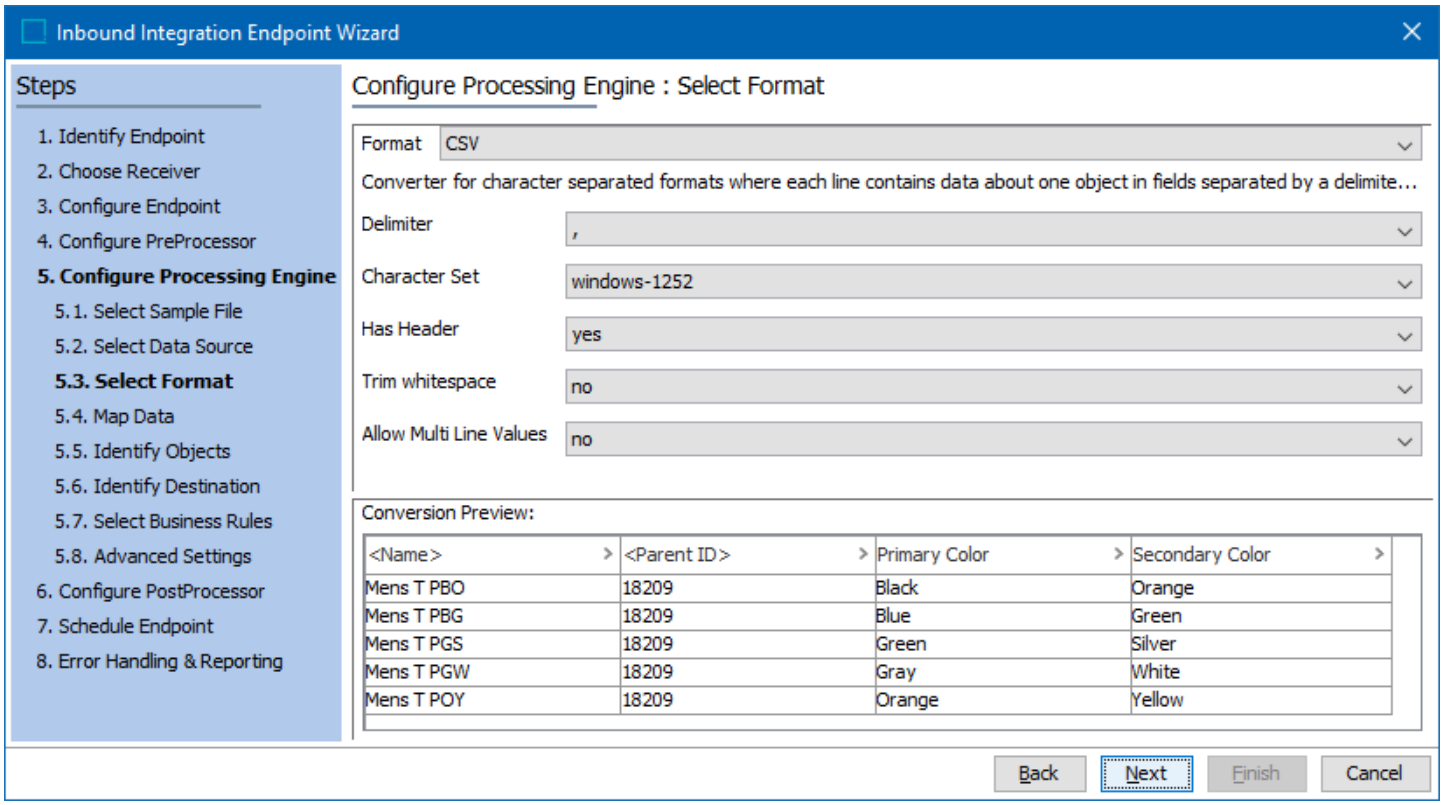
5. **Allow Multi Line Values** - determines how new lines (return codes) are handled in the file, indicated by data with double quoted values that split over several lines. Typically, the newline character is interpreted as the end of data, delimiting a data record. However, sometimes data spans more than one line–that is, includes newline characters. In such cases, values must be quoted with the double quote character (") to be imported correctly.
   - **Yes** values with new lines are quoted with the double quote character (").
   - **No** new line characters are interpreted as the end of data, delimiting a data record.

6. **Conversion Preview** - displays a sample of the first few lines of the file to allow verification that the selected options are correct.

Import Manager



IIEP



Outbound Data

When data leaves STEP via CSV format, several options are available to customize the output.

Note: When using the JDBC delivery method users must select 'CSV' as the format. For more information on how to configure the CSV format for the JDBC delivery method, refer to the **Exporting Data via JDBC with CSV Format** topic in this guide.

Outbound Parameters

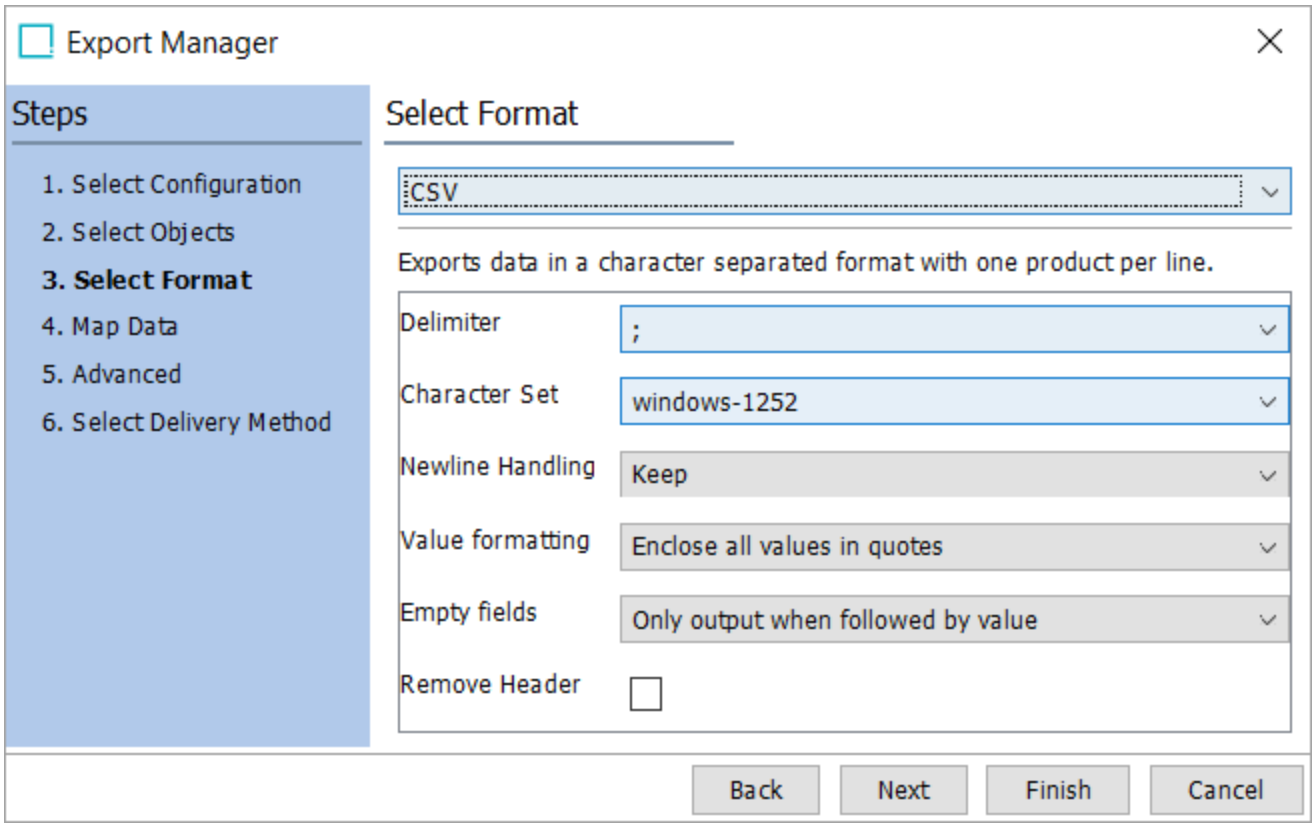
The following parameters are available in both Export Manager and OIEP.

1. **Delimiter** - determines the character used to delimit the exported data. Options are semi-colon (;), comma (,), the tab character, the pipe symbol (|), or colon (:).
   - **Tab** is recommended to avoid splitting the data on printable characters that may occur within the values.
   - Only the delimiters in the dropdown are allowed. If a different delimiter is required, after exporting, update the file outside of STEP.
2. **Character Set** - determines the characters that can be exported. Options include Windows-1252, ISO-8859-1 (also known as the Latin-1 character set), UTF-8, or UTF-16.

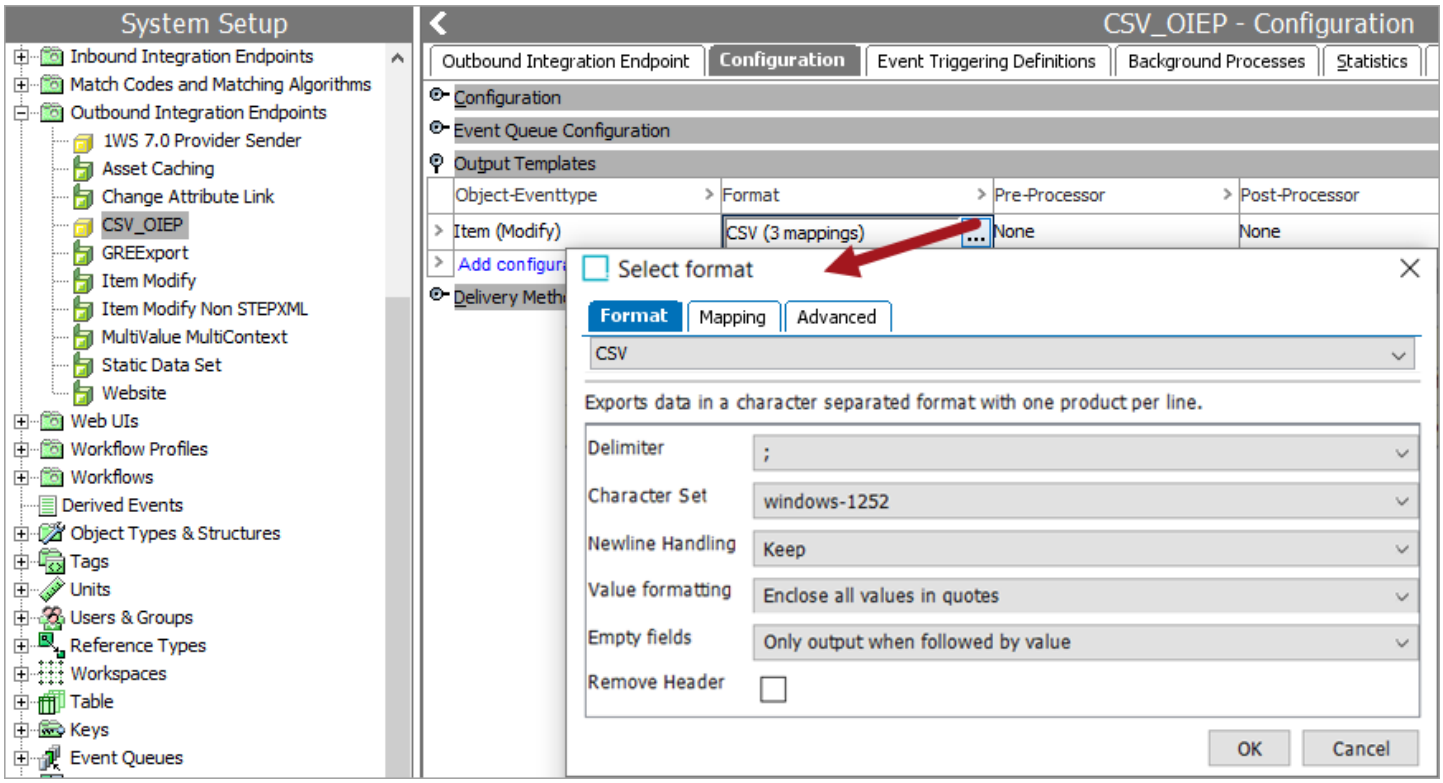
- **UTF-8** is recommended unless you have a reason to do otherwise.
- **UTF-8** or **UTF-16** (a Unicode character set) is required to correctly export special symbols, like a trademark symbol.
- **Shift-JIS** character set used for the Japanese language.

3. **Newline Handling** - determines how new lines (return codes) are handled when they occur in the data being exported.
   - **Convert to Space** (recommended) when new lines exist in the data being exported, to prevent data integrity issues caused by character-separated files having new lines within a record.
   - **Keep** allows new lines to be included and can be used when the file recipient can handle them within a record.
4. **Value formatting** - determines how values are displayed in the exported file.
   - **Enclose all values in quotes** takes STEP values and adds quotes during export.
   - **Export without quotes unless value contains delimiter character(s)** exports STEP values without modification unless a delimiter is found.
5. **Empty fields** - determines how fields without values are exported.
   - **Only output when followed by value** exports only fields that include a value.
   - **Always output** exports all fields, regardless of the presence of a value, and is required when splitting multivalued fields.
6. **Remove Header** - determines whether the headers can be exported.

Export Manager



OIEP


System Setup
Inbound Integration Endpoints
Match Codes and Matching Algorithms
Outbound Integration Endpoints
1WS 7.0 Provider Sender
Asset Caching
Change Attribute Link
CSV_OIEP
GREExport
Item Modify
Item Modify Non STEPXML
MultiValue MultiContext
Static Data Set
Website
Web UIs
Workflow Profiles
Workflows
Derived Events
Object Types & Structures
Tags
Units
Users & Groups
Reference Types
Workspaces
Table
Keys
Event Queues
CSV_OIEP - Configuration
Outbound Integration Endpoint
Configuration
Event Triggering Definitions
Background Processes
Statistics
Configuration
Event Queue Configuration
Output Templates
Object-Eventtype
Format
Pre-Processor
Post-Processor
Item (Modify)
CSV (3 mappings)
None
None
Add configura
Delivery Meth
Select format
Format
Mapping
Advanced
CSV
Exports data in a character separated format with one product per line.
Delimiter
;
Character Set
windows-1252
Newline Handling
Keep
Value formatting
Enclose all values in quotes
Empty fields
Only output when followed by value
Remove Header
OK
Cancel

Exporting Data via JDBC with CSV Format

Exporting data with the JDBC delivery method requires the use of the CSV format. Setup for this involves the following steps in the Export Manager or in an OIEP:

- **Add required drivers** for JDBC
- **Configure properties** in the sharedconfig.properties file
- **Select format** of CSV, using the required settings
- **Map Data**, making sure to include the required action field
- **Select Delivery Method** of JDBC, using the necessary settings

Each of these steps is described below.

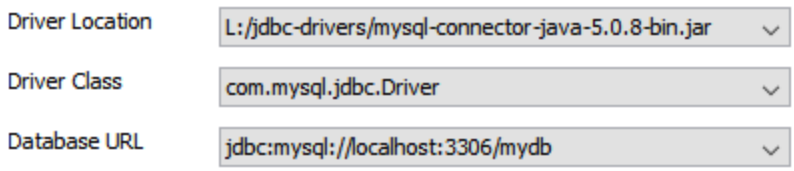
For details on a use case in which JDBC is used to update analytics data, refer to the **Analytics using JDBC Example** topic in the **Analytics** guide.

Install Required Drivers

JDBC specification 4.1-compliant drivers should be placed in a directory accessible from all application servers. These drivers can then be made available for the delivery plugin via the dynamic properties JDBCDeliveryPlugin.DriverPath.[n] and JDBCDeliveryPlugin.DriverClass.[n]. For more information regarding applicable Java drivers, review the RDBMS vendor's homepage on the web.

Configure Properties

The options available in the following dropdown parameters on the JDBC delivery method are supplied by the sharedconfig.properties file on the STEP application server. Multiple entries can be added using the dynamic properties. Each configuration entry must have a unique integer (JDBCDeliveryPlugin.DriverPath.1, JDBCDeliveryPlugin.DriverPath.2, JDBCDeliveryPlugin.DriverPath.3, etc.). When duplicates exist, only the last value is displayed in the dialog.



Use the following steps to supply data for the parameters displayed above:

1. Prior to configuration, clicking the **Driver Location** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **JDBCDeliveryPlugin.DriverPath.[n]** property. As an example:

   ```
   JDBCDeliveryPlugin.DriverPath.1 = L:/shared/mysql-connector-java-5.1.42-bin.jar
   ```

   In this example, the drivers are stored on the application server's L:/shared drive.

2. Prior to configuration, clicking the **Driver Class** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **JDBCDeliveryPlugin.DriverClass.[n]** property. As an example:

   ```
   JDBCDeliveryPlugin.DriverClass.1 = com.mysql.jdbc.Driver
   ```

   In this example, the drivers class used is 'com.mysql.jdbc.Driver.'

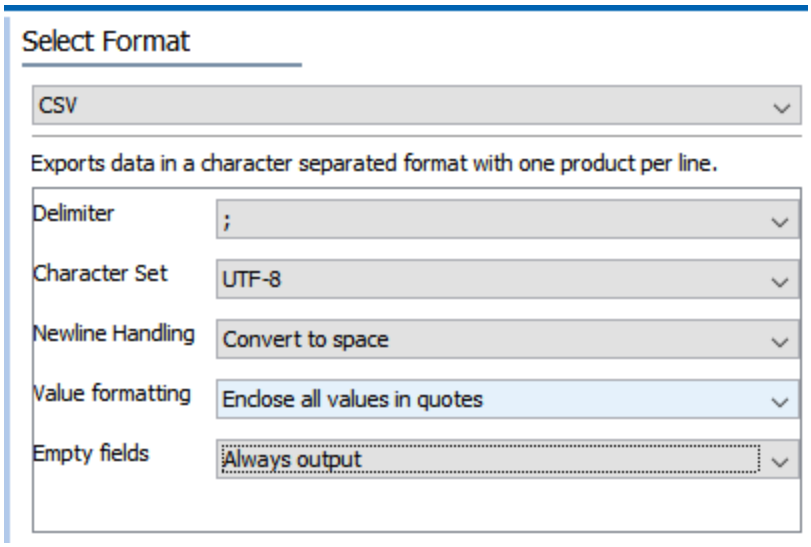
3. Prior to configuration, clicking the **Database URL** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **JDBCDeliveryPlugin.URL.[n]** property.

   ```
   JDBCDeliveryPlugin.URL.1 = jdbc:mysql://localhost:3306/mydb
   ```

   In this example, the database URL used is 'jdbc:mysql://localhost:3306/mydb.'

Select Format

When exporting data using the JDBC delivery option it is important that, in the 'Select Format' step, users select 'CSV.' JDBC does not support any other formats in STEP. In order to enable proper functioning of the JDBC delivery option, the configuration of the 'CSV' format option should look like this:

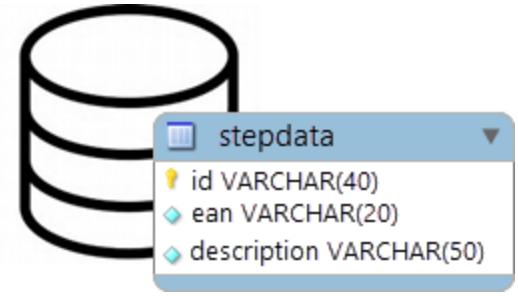


Map Data

The JDBC delivery option uses export mapping to allow STEP data to be inserted or updated ("upsert") into, or deleted from ("delete") a destination database table. When a configured object in the exported CSV file finds a match in the destination database, the interaction can be configured to either / both:

- 'upsert' the object, which means the object, if found based on the JDBC Key Columns, will be updated with the information in the exported CSV file, or if not found based on the JDBC Key Columns, the information in the exported CSV file will be inserted into the table in the destination database
- 'delete' the object, which means that if an object indicated in the exported CSV file is found in the destination database based on the JDBC Delete Key Columns, it will be deleted

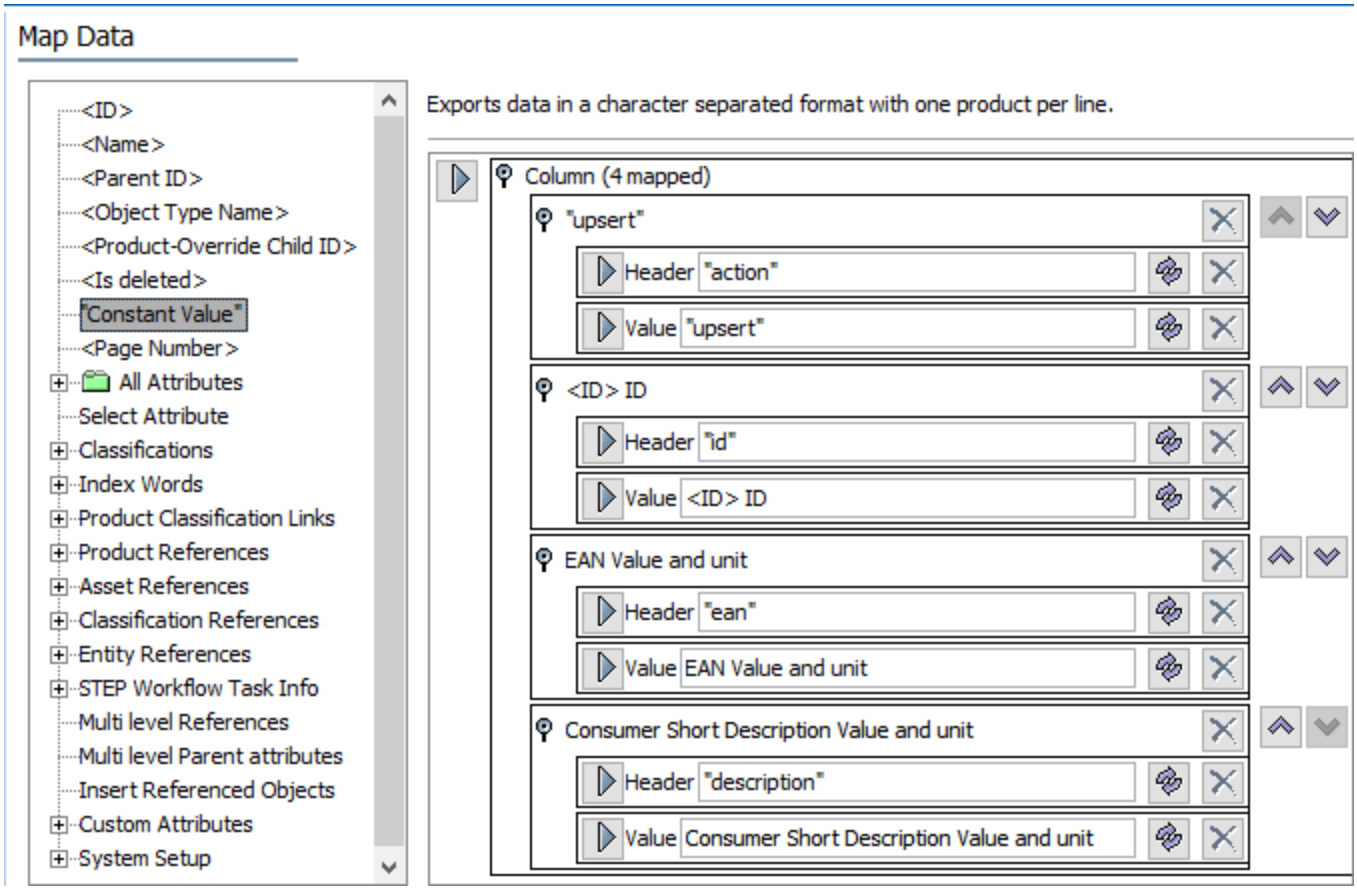
For example, consider the following external database sample table:



In the 'Map Data' step, map the required objects considering the following points to ensure the CSV file can be successfully processed:

- The column headers in the exported CSV file must match the table column headers in the destination database. Map a constant value on the Header row for each mapped object to supply the external database table column header. For example, notice that the external table has a row named 'description' but is mapped to the STEP attribute 'Consumer Short Description.' The header parameter is used to make the exported data match the external table. For details about using the constant value data source, refer to the **Constant Value - Data Source Outbound** topic.
- Create an additional mapped column with the header of 'action' and the appropriate value of either delete or upsert. Use the transformation button to change the text displayed for both the Header and the Value.

Below is an example of an 'upsert' action in Map Data for the external database table shown above. This setup will direct the process to look for three configured attributes (ID, EAN, and Consumer Short Description) in the table in the destination database, and either insert the value (if not found), or update the value (if found):



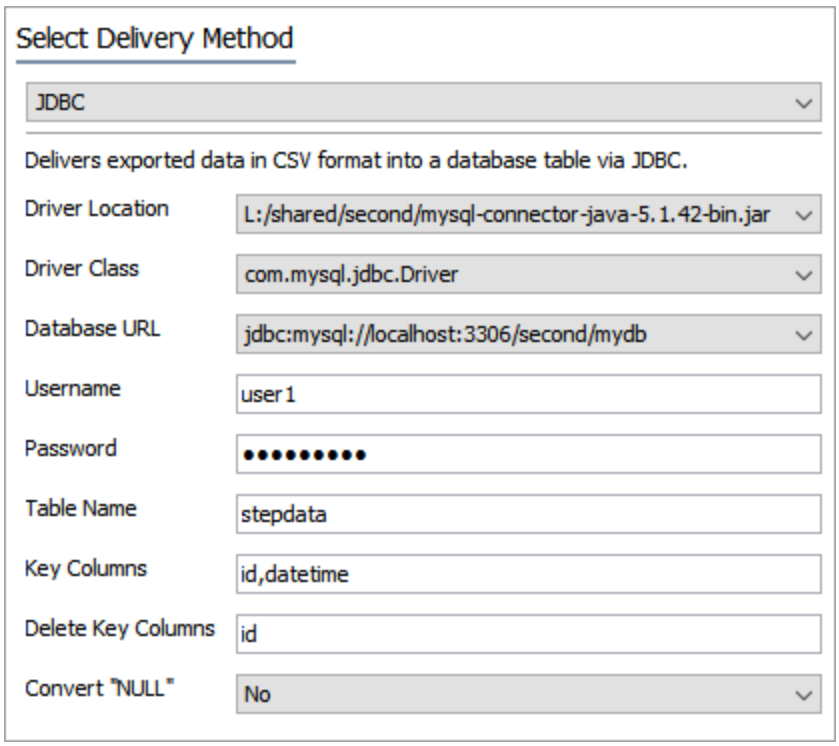
Important: For the external destination database, the JDBC delivery method assumes that the destination data table has a column used as a key, and a number of columns for values (for example, `key value1 value2 [value (n)]`.) This means a destination data table having only a single column (e.g., ID) will cause the JDBC export to fail, however, a table having at least two (2) columns (e.g., ID and Name) will work properly.

To create mapping for the delete action, the action column would have a value of "delete."

If using an OIEP, one output template would be created to handle Create and Modify events, and a second output template would handle Delete events. In this way, separate mapping is available for each, allowing for one to include the upsert action and the other to include the delete action.

Select Delivery Method

When setting the delivery method for the export, use the following parameters to configure it.



1. In **Select Delivery Method**, choose **JDBC** from the dropdown.
2. In **Driver Location**, select from the dropdown one of the paths to the relevant .jar file.
3. In **Driver Class**, select from the dropdown one of the pre-configured driver class.
4. In **Database URL**, select from the dropdown one of the pre-configured URLs to the destination database.
5. In **Username**, enter the username required to access the destination database.
6. In **Password**, enter the password required to access the destination database.
7. In **Table Name**, enter the name of the table in the destination database to which STEP will be publishing data
8. In **Key Columns**, list the names of the columns appearing on both the exported CSV file and the table in the destination database, separated by commas (and no spaces), into which STEP will publish data
9. In **Delete Key Columns**, list the names of the columns appearing on both the exported CSV file and the table in the destination database, separated by commas (and no spaces), from which STEP will delete data. The headers contained in this field can differ from the headers in the 'Key Columns' field, but they must also be part of the upsert key definition.
10. In **Convert "NULL"**, choose Yes if the string “NULL” should be converted to the value null. This may, for instance, be used for clearing a value in a column in the target database. This parameter defaults to No.

cXML Format

The cXML format exports data using Ariba cXML 1.2 and is an XML-based language designed specifically for communication of e-commerce business documents. For more information, search the web.

Format Availability

cXML is available for selection in:

- Export Manager - refer to **Creating a Data Export**
- OIEP - refer to **Creating an Outbound Integration Endpoint**

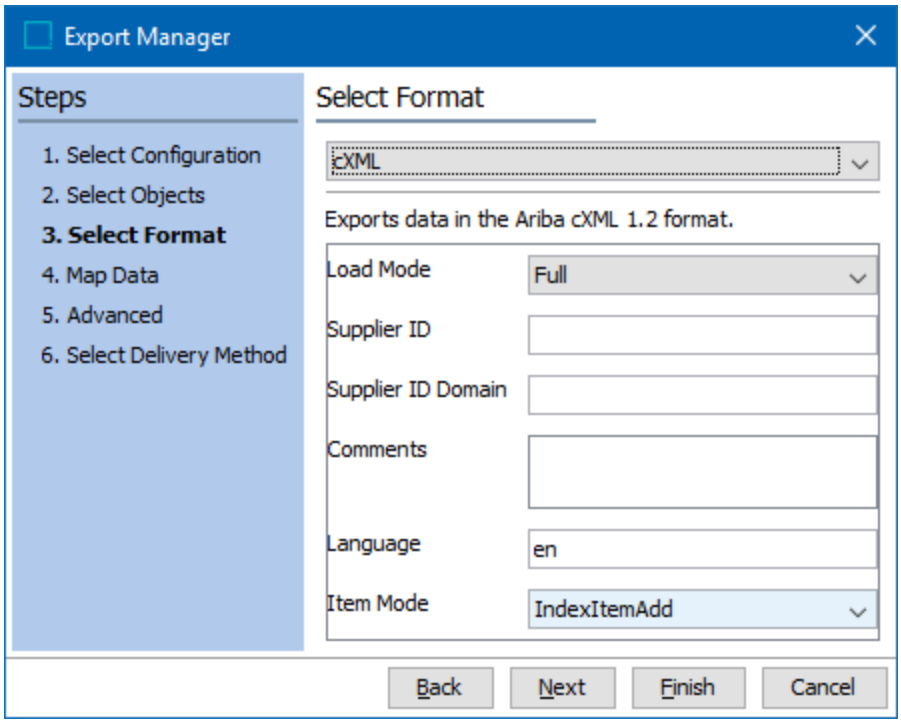
Mapping

This format requires creating a data map between STEP and the data being processed, and may also include data transformations. For details, refer to **Data Mapping**.

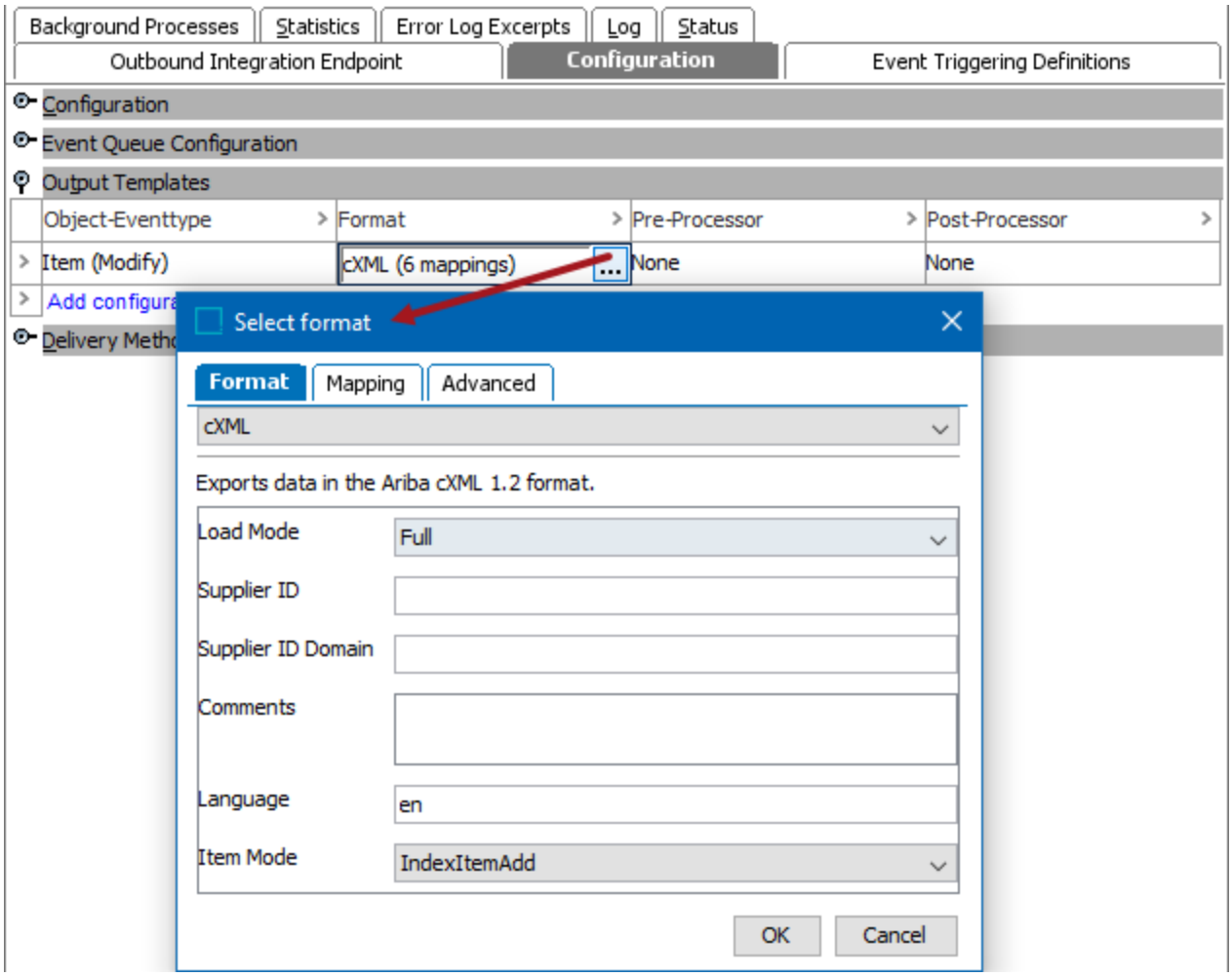
Outbound Data

The same parameters are available in both Export Manager and OIEP.

Export Manager



OIEP


Background Processes
Statistics
Error Log Excerpts
Log
Status
Outbound Integration Endpoint
Configuration
Event Triggering Definitions
Configuration
Event Queue Configuration
Output Templates
Object-Eventtype
Format
Pre-Processor
Post-Processor
Item (Modify)
cXML (6 mappings)
None
None
Add configura
Delivery Metho
Select format
Format
Mapping
Advanced
cXML
Exports data in the Ariba cXML 1.2 format.
Load Mode
Full
Supplier ID
Supplier ID Domain
Comments
Language
en
Item Mode
IndexItemAdd
OK
Cancel

ECLASS Format

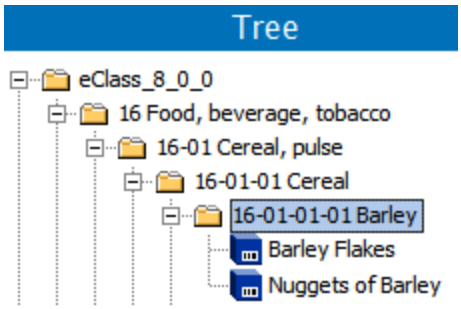
ECLASS is a hierarchical classification and product description system for grouping materials, products, and services. This includes a logical structure with a level of detail that corresponds to product-specific attributes. These classifications and standardized attributes are an international standard and can be used for reliable information exchange between suppliers and their customers. For detailed information about ECLASS, search the web.

Note: ECLASS (formerly eCl@ss) has updated their branding. These branding changes will be reflected in Stibo Systems UIs in a future release. STEP functionality supports ECLASS Basic and the viewing and editing of products, via the Web UI, for ECLASS Advanced.

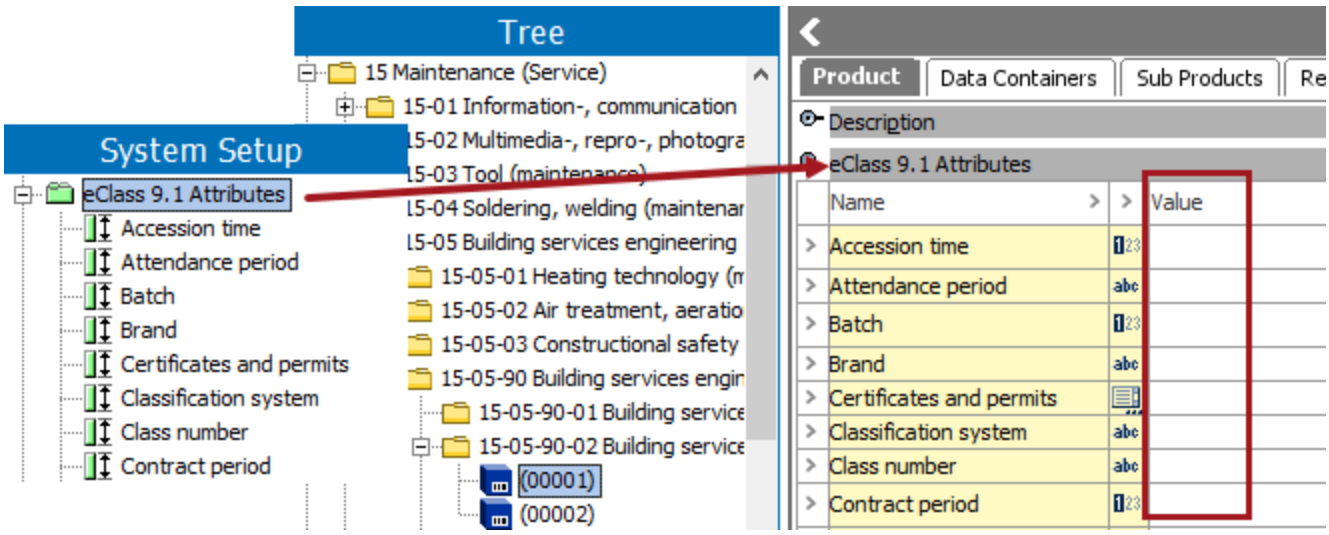
ECLASS is an add-on component. Contact your Stibo Systems representative to activate ‘eclassimporter’ on your STEP system.

With ECLASS you can import XML formats for any language using versions 7.0, 7.1, 8.0, 8.1, 9.0, 9.1, 10.0, 11.0, 11.1, 12.0, 13.0 and 14.0. For more information on the ECLASS license, contact your Stibo Systems account manager.

In STEP, ECLASS classifications are linked to objects (materials, products, and/or services) via a 'product-to-classification link type' reference type. This displays objects within the ECLASS folder as shown in the image below.



When a product is linked to an ECLASS classification, via a product-to-classification link type with 'Inheritance of Specification Attributes / Data Container Types' set to 'Yes,' the ECLASS attribute links are inherited from the classification to the linked products, and the attributes are displayed and maintained on the product tab under a flipper that indicates the version. For more information on link inheritance, refer to the Inheritance Example for a Product to Classification Link Type topic in the **System Setup** documentation.

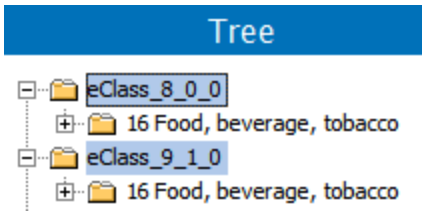


Format Availability

ECLASS import format is available for selection only in the eCl@ss Classification Import wizard. For details, refer to ECLASS Classification Import.

Multiple Versions

The ECLASS classification data is delivered in a .ZIP file, where the file name indicates the version. The import wizard uses a background process to create attributes (if necessary), and classification folders that include the version number. This allows you to access multiple versions of ECLASS within the same system.

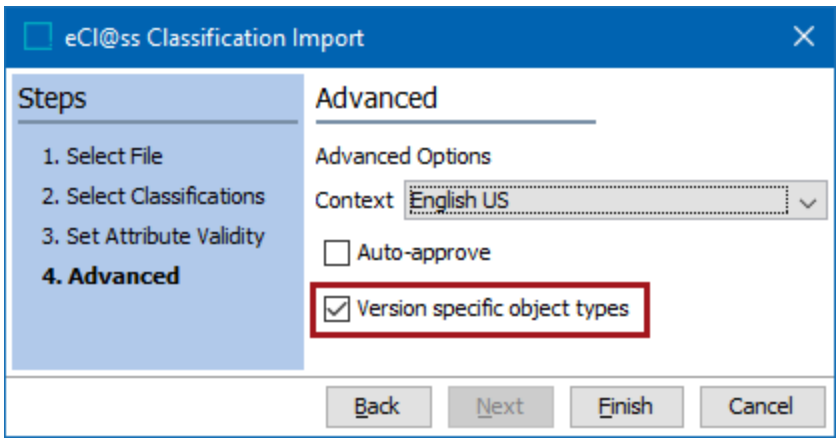


Update objects to a newer classification version

If you want to update existing objects to a newer ECLASS classification version, you can re-classify them using the Bulk Update operation 'Industry Classification Synchronization.' For more information, refer to the References and Links: Industry Classification Synchronization Operation topic of the **Bulk Update** documentation.

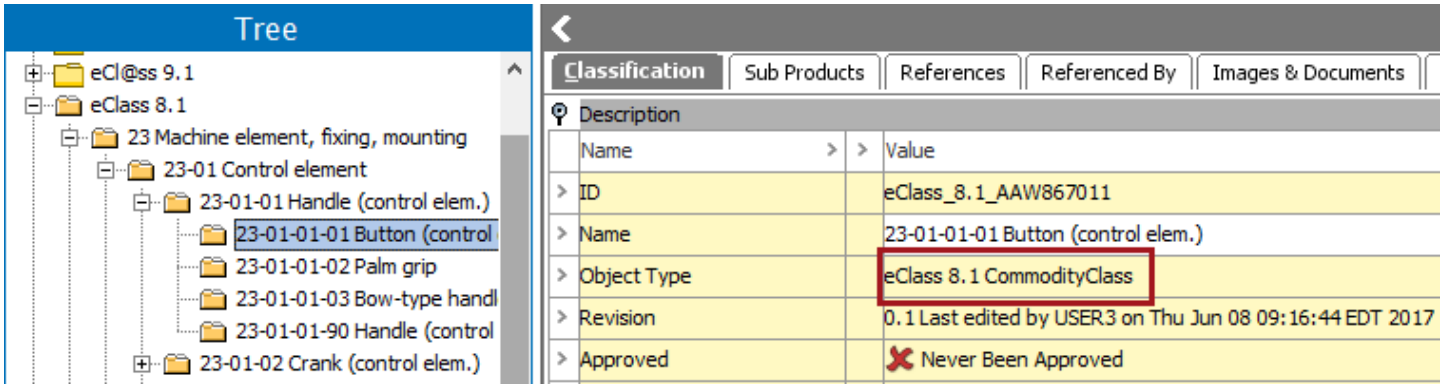
Version-specific classification object types

To create a version-specific object type structure, select 'Version specific object types' on the **Advanced** step of the ECLASS Classification Importer.



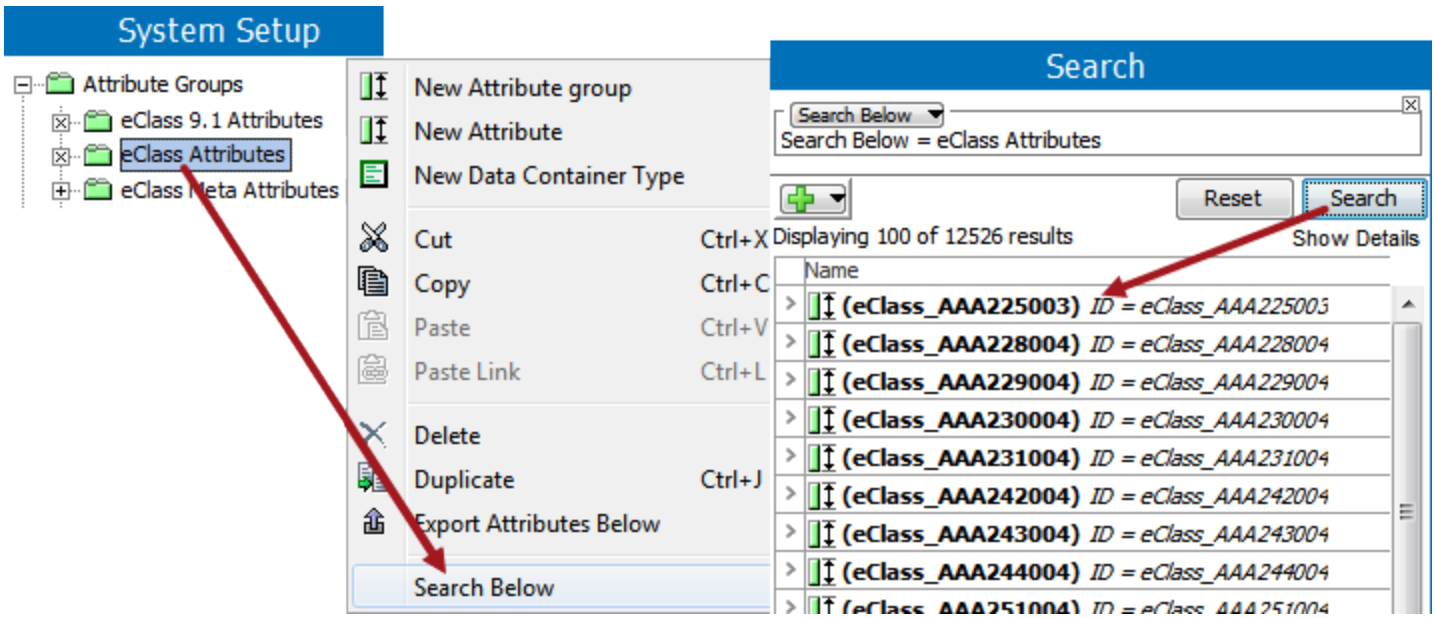
Selecting this option during import will automatically create a new version-specific object type structure for ECLASS classifications if one does not already exist for that version.

Note: For optimal functionality of the Industry Standard Mapper solution, it is crucial that the system consistently maintains a version-specific object type structure for ECLASS classifications. Ensure the 'Version specific object types' option is selected during every import to utilize the Industry Mapper Solution.



Finding an Attribute

Depending on the number of classifications imported, System Setup may not allow browsing through the attributes created. However, you can view a portion of them, or find a particular one, using the right-click 'Search Below' functionality.



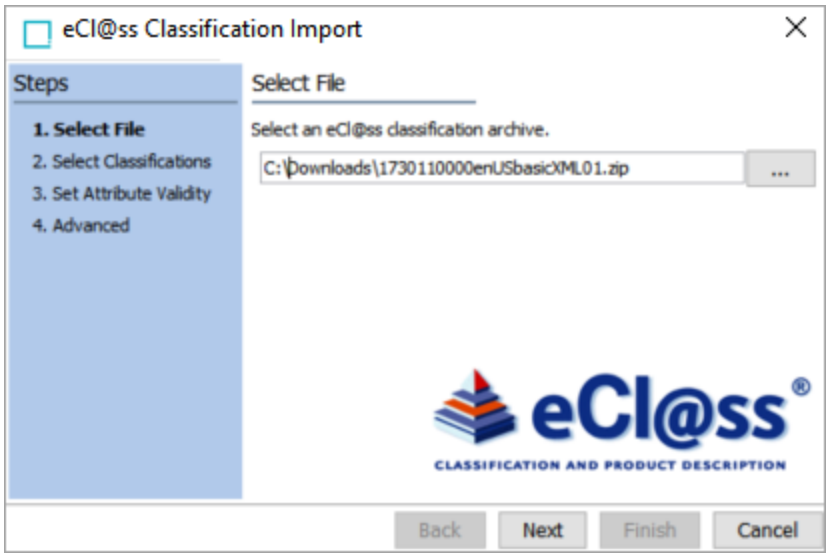
Any units or LOVs created upon import of an ECLASS XML file will also be stored in their own unit / LOV groups labeled 'eCl@ss', with the version specified as a metadata attribute on the object.

Inbound Data

Running the eCl@ss Classification Import wizard creates three attribute groups (if they do not already exist):

- **eClass [version number] Attributes** - specification attributes applicable to the named version. When multiple ECLASS versions are imported, multiple eClass [version number] Attributes groups are created.
- **eClass Attributes** - specification attributes used by all imported versions; by default, not displayed on products or classifications since the 'Show in Workbench' checkbox is not checked. For more information, refer to the Attribute Groups topic in the **System Setup** documentation.
- **eClass Meta Attributes** - description attributes for metadata on attributes, units, LOVs, and classifications.

eCl@ss Classification Import



ECLASS Advanced in the Web UI

ECLASS Advanced users can configure an ECLASS Advanced Editor Screen in the Web UI. You can link to versions, add and edit blocks, modify attributes, as well as modify attribute values.

For more information on the ECLASS Advanced Editor, refer to the ECLASS Advanced Editor Screen topic in the **Web User Interfaces** documentation.

For more information on importing ECLASS data into STEP, refer to the ECLASS Classification Import section of the documentation.

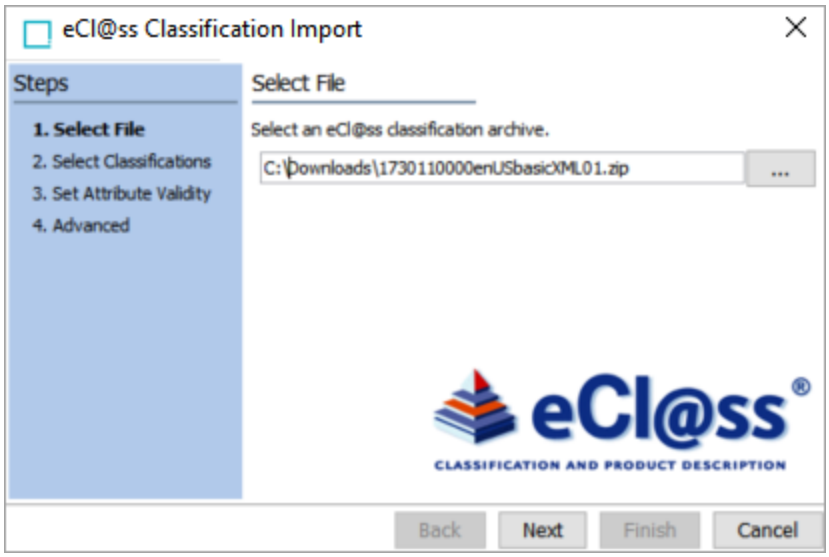
ECLASS Classification Import

Note: ECLASS (formerly eCl@ss) has updated their branding. These branding changes will be reflected in Stibo Systems UIs in a future release. STEP functionality supports ECLASS Basic and the viewing of products, via the Web UI, in the context of an ECLASS Advanced taxonomy.

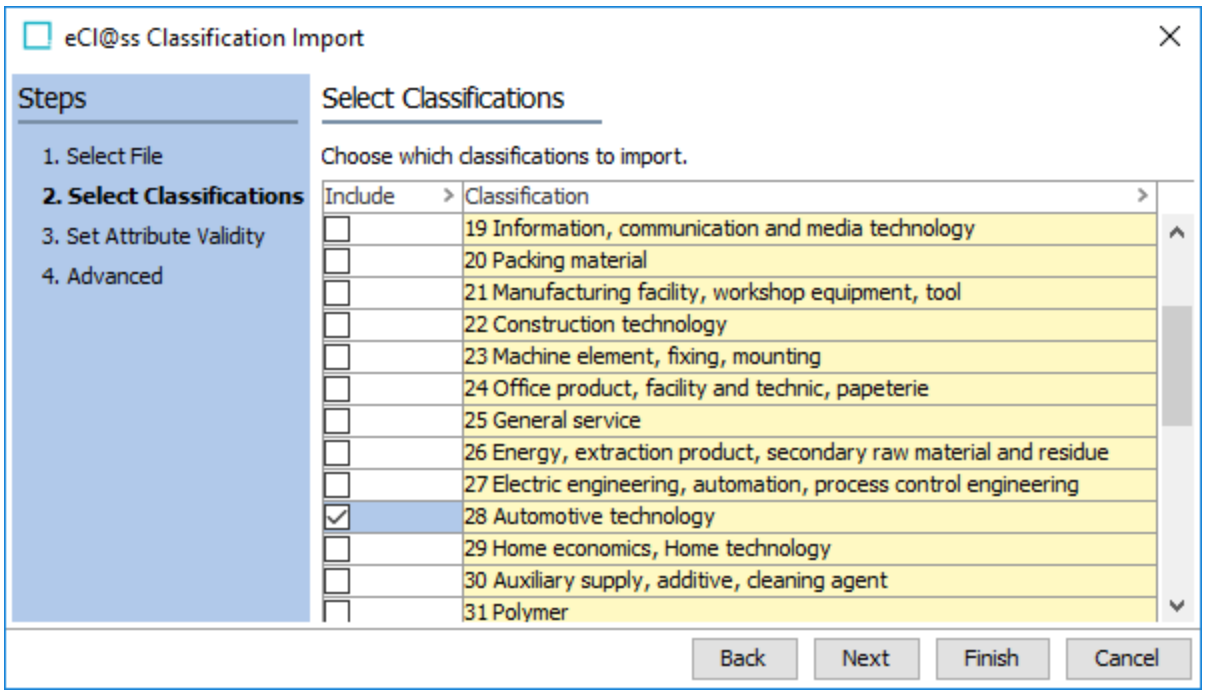
Use the following steps to start the ECLASS import.

1. On File menu > Import > click eCl@ss... to open the **eCl@ss Classification Import** wizard.
2. For the **Select File** step, click the ellipsis button (**...**) and search or browse for the relevant ECLASS file from your local machine. Once selected, click the Select File button, and then **Next**.

   Note: If an invalid file is selected for import, an error message prompting the user to select a valid ECLASS XML file will appear when 'Next' is selected.

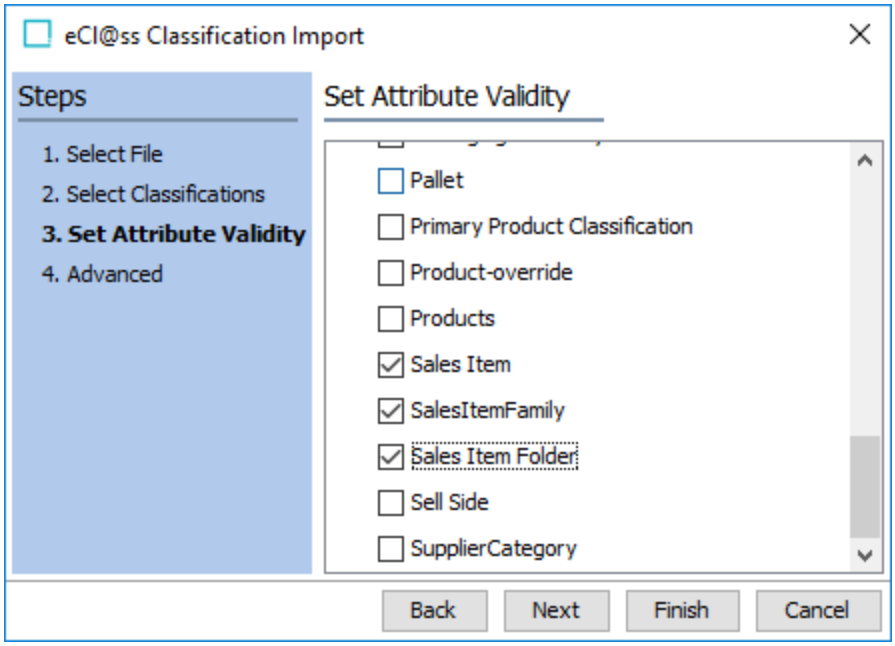


3. For **Select Classifications**, select all classifications to import, and click **Next**.



4. For **Set Attribute Validity**, select the valid product object types for the imported ECLASS attributes. This step is optional, and only available for ECLASS Basic XML imports.

Note: If an imported attribute already exists in the system, any selection made on this step will be added to the list of valid object types for the existing attribute.

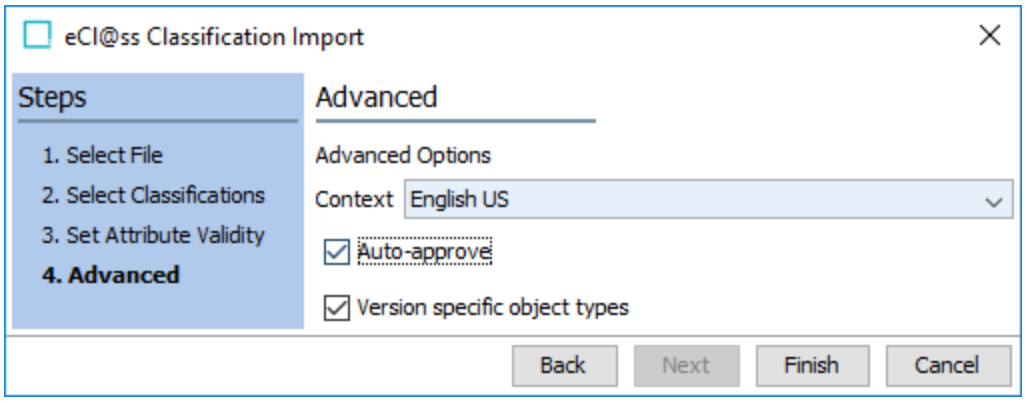


Important: If attribute validity is not set during the import it will have to be set manually later, otherwise you will not be able to maintain these attributes on linked ECLASS products.

5. For **Advanced**, select the context for the language of the ECLASS file you are importing. When checked, the 'Auto-approve' option will attempt to approve the ECLASS Basic classifications and any products that are linked to the classifications.

   Click 'Version specific object types' if the importer should specify the ECLASS version in the object types of the imported classifications. This will automatically create a new version specific object type structure for ECLASS classifications if one does not already exist for that version.

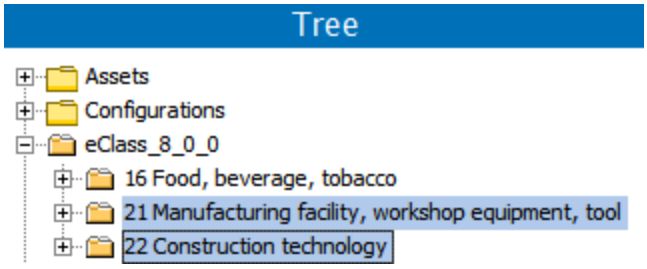
   Click **Finish** to begin the import.



Important: It is not possible to import a classification structure with the 'Version specific object types' option selected if classification object types have previously been created for classifications of the same version (but without this option selected). Attempting to do so will display an error in the background process and the import will fail.

6. Start the import. If required, refer to the information in **Running a Data Import** section.

   When the Import Manager Pipeline background process is complete, the newly created classification folders are displayed in the Tree.



7. Only complete this last step if you will import multiple ECLASS languages for different contexts:
   - **Configure the language displayed in the Tree for the ECLASS classifications:** In System Setup > Object Types & Structures > Alternate Classifications > eCl@ss Folder, for the top-level folder, set the

Dimension Dependencies parameter to Language.

- **Configure the language displayed for the eCl@ss attributes on the attribute editor:** In System Setup > Attribute Groups > eClass Meta Attributes, for every attribute in this group, set the Dimension Dependencies parameter to Language.
- **Import additional ECLASS data:** Run the import wizard again for each ECLASS language required.

Import Report Messages

If you have previously used the ECLASS Basic data import functionality on your system, there are special circumstances that will prompt the ECLASS Basic importer to make updates to the existing ECLASS data model. These updates help ensure that the ECLASS Basic data model is aligned with the ECLASS dictionary. When these data model updates occur, they will be output in the import execution report. These execution report updates are described below.

Multi and Single Value Attribute Options

If you have already imported data using ECLASS Basic before, there are two types of messages STEP can generate based on changes to the ECLASS Basic data model. If ECLASS Basic attributes are changed, one or more of the following messages will display in the execution log:

- 'Single valued attributes changed to multi valued attributes: Attribute ID, Attribute ID 2...'

  This message displays when an existing attribute has been changed from single-valued to multi-valued.

  When this message displays, it means the existing data will not be impacted. Be aware of outbound integrations and export configurations that rely on a specific valency of the attribute.

- 'Multi valued attributes changed to single valued attributes: Attribute ID 1, Attribute ID 2...'

  This message displays when an existing attribute has been changed from multi-valued to single-valued.

  When this message appears, existing data will not be impacted. Be aware of outbound integrations and export configurations that rely on a specific valency of the attribute.

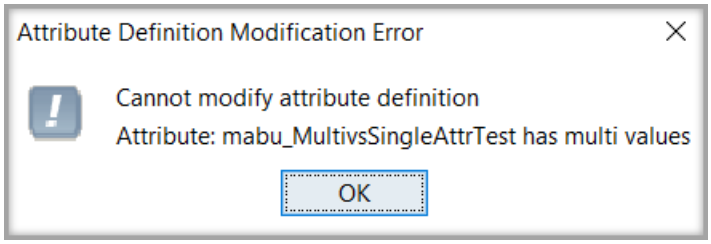
When an existing attribute cannot be changed from single-valued to multi-valued or multi-valued to single-valued, one of the following messages will display:

- 'Cannot update single-valued attributes to multi-valued attributes: Attribute ID, Attribute ID 2...'
- 'Cannot update multi-valued attributes to single-valued attributes: Attribute ID, Attribute ID 2...'

If the 'Cannot update multi-valued attributes...' message displays, the user must locate the nodes containing the attributes that have been identified, correct the attribute value, and delete the incorrect values. The user must set the attribute to single-valued.

Changing an attribute from multi-valued to single-valued manually may cause additional user prompts to appear:

The current message displays when the data revision has a multi-value attribute for one or more products. The user will receive a pop-up message when attempting to update the data that states, 'Cannot modify attribute definition. Attribute: attribute_ID has multi values.'



When this pop-up displays, the user must ensure that only one value is assigned and applied to all affected products in the current revision to make it a single. After the revised data shows a single-value, STEP allows the update.

The second message displays when there are multiple values in one or more older data revisions. A dialog will display that says 'Attribute has historic Values.' If this occurs, the pop-up message displays, 'The historic values have to be deleted from the database to complete the change from multi-valued to single-valued,' with the options to 'Delete' and 'Cancel.'

Important: When a user selects 'Delete,' all historical data stored in the older revisions is removed for the particular attribute and relevant products.

For more information on updating attribute values, refer to the **Editing Attribute Values** topic in the System Setup documentation.

Dimension Dependency for LOVs and Attributes

When importing ECLASS with the List of Values (LOV) data type, a 'value_meaning' tag may be included in the ECLASS dictionary file. If so, the LOV should be created as language dimension-dependent.

ECLASS data types without a value_meaning tag should be imported as a dimension-independent list of values, as the value of the list of value are not translatable.

For existing lists of values, the dimension dependency should be updated based on the presence of a value_meaning tag in the ECLASS import file:

1. No value_meaning tag: The LOV should be updated to be dimension independent, if dimension dependent. If already dimension independent the list of values should not be changed.

2. No value_meaning tag: If the LOV contains LOV values in more dimensions, such as values in more languages, the dimension dependency should not be changed, but reported in the execution log.
3. Value_meaning tag: The LOV should be updated to be dimension-dependent, if dimension-independent. If already dimension-dependent the LOV should not be changed.

For more information on changing dimensions, refer to **Dimension Dependent Attributes** topic in the **Attributes** section of the **System Setup** documentation.

When making changes to the list of values, there are several messages that can display in the execution log:

- Cannot make list of values dimension-dependent as the user is not privileged to modify list of values dimension dependencies. Failed list of values: lov ID1, lov ID2…
- Cannot remove list of values dimension dependencies as the user is not privileged to modify list of values dimension dependencies. Failed list of values: lov ID1, lov ID2

If the execution log delivers one of these messages, the attempt to revise dimension dependencies has failed. To successfully make the desired changes, a user that is privileged to modify dimension dependencies must run the import again.

If the dimension dependencies change, the execution log message will state:

- List of values dimension dependencies update for: lov ID 1, lov ID 2…

  When this message displays, it means the language dimension dependency has been added for the LOV and existing data will not be affected.

If the LOV is language-dependent and contains LOV values in more dimensions (languages), the execution log message will state:

- List of values cannot be made dimension-independent because dimension dependent values exists: lov ID 1, lov ID 2…

When this message displays, the user must remove the dimension dependency manually in the workbench for each of the listed LOVs. When doing so, the user can choose which dimension is most important when merging the dimension-dependent values.

LOV Value Filters on ECLASS Attribute-Classification Links

When importing an LOV with or without constraints, the following execution log messages can display:

- Added value filter for list of values eClass_LOV to classifications: eClass_classification…

  Example execution log message:

  Added value filter for list of value eClass_AAB637006 to classifications: eClass_10_AFR604003, eClass_10_ACN874011

When this message displays, it means that value filters have been added. If you had values that were filtered away, those values will be displayed as 'orphan values'. For more information on Orphaned values, refer to the **Linked Orphan Documentation**.

This message displays when a value filter is added to an existing LOV.

- Failed to add value filters to list of values eClass_LOV to classifications: eClass_classification…

  Example execution log message:

  Failed to add value filter for list of value eClass_AAB637006 to classifications: eClass_10_AFR604003, eClass_10_ACN874011

This message displays when the ECLASS import fails to add a value filter to an existing LOV. This can only occur if a value filter is created based on a value that is not a legal LOV value. This indicates that there is an error in the ECLASS XML import file. When this happens, ECLASS should supply a new file without error.

This is an alternate option if you cannot wait to get a new corrected import file from ECLASS: the user can add the value that caused the import to fail to the LOV and the import file should be re-imported.

LOVs' Usage of Value IDs

The following execution log messages will display following an update of the LOVs during an ECLASS Basic data import:

If the existing LOVs were changed to use value IDs, the message will read:

- List of values changed to use value IDs: attribute ID 1, attribute ID 2...

  When this message displays, the LOV values that are already in use by these products will not be affected by this change.

If the existing LOVs were changed to *not* use value IDs, the message will read:

- List of values changed to not use value IDs: attribute ID 1, attribute ID 2...

  When this message displays, existing data will not be affected by this change. However, removal of LOV value IDs may affect export configurations in the case where the export configuration is set to export LOV value IDs.

If the existing LOVs cannot be changed to *not* use value IDs, the message will read:

- Cannot change list of values to not use value IDs: attribute ID 1, attribute ID 2...

  This message is unlikely to occur, however if it does, the LOV affected is used by a data container key. To fix that you must:

  - Deactivate the data container key.
  - Remove the attribute that uses the LOV from the data container key.

- Change the LOV to be single-valued.
- Optionally find another attribute to be part of the data container key.
- Active that data container key again.

ETIM Format

Electro-Technical Information Model (ETIM) is a classification structure used to standardize the electronic exchange of product data for electrical and electronic products and to enable the electronic trading of these products. The standard is designed to meet the requirements of the electrical industry and allows for a uniform, technical description of electrical goods and their assignment to a class of products. In order to support customers who need to work with data using the ETIM standard, the inbound data exchange functionality includes ETIM formats to allow for easy processing and setup of ETIM classifications and their related data elements. For more information on ETIM, search the web.

ETIM is an add-on component. Contact your Stibo Systems representative to activate 'etim-importer' on your STEP system.

Format Availability

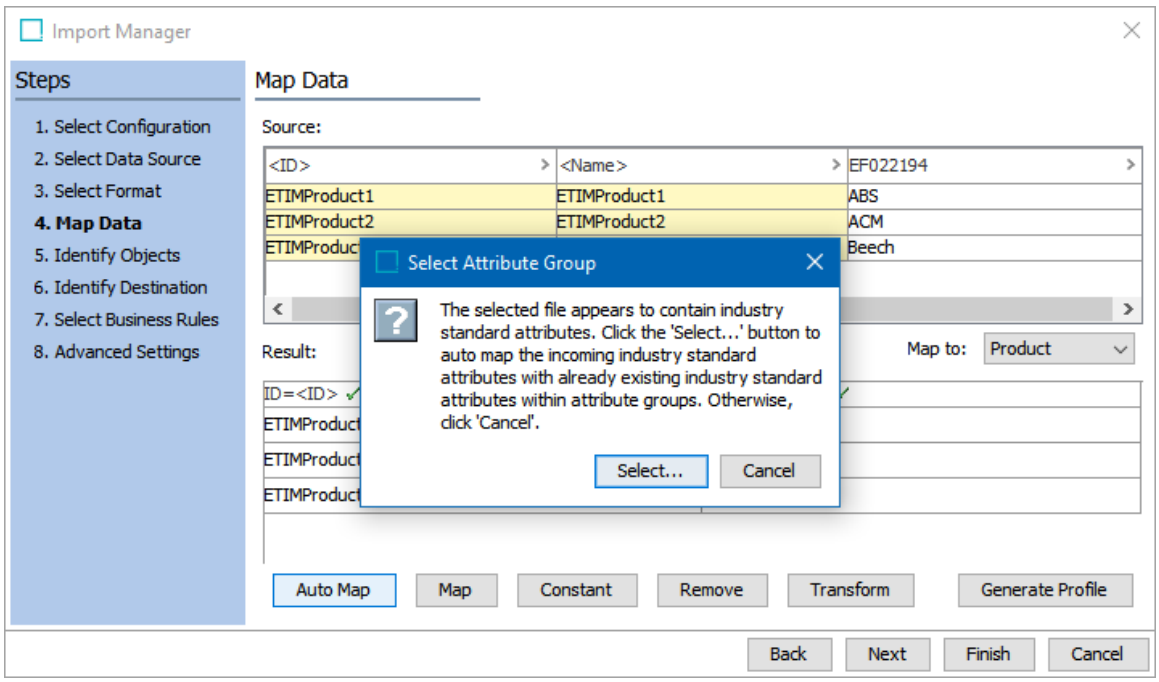
ETIM format is available for selection in:

- IIEP - refer to **Creating an Inbound Integration Endpoint**
- Import Manager - refer to **Creating a Data Import**

Mapping

Importing the ETIM data model does not require any data mapping. However, when importing product data described in the ETIM standard, the Auto Map functionality becomes enabled when the system identifies industry standard attributes when importing various import formats. Auto Map allows you to map these attributes with attributes that are part of an already-installed ETIM version.

For example, if you have the same attribute in ETIM7 and ETIM8 attribute groups, the message in the image below displays. After clicking 'Select...,' you choose an attribute group that corresponds to the ETIM version you want to map to. And, then you continue with the import process as described in **Import Manager - Map Data** topic in the **Import Manager** documentation.



ETIM Configuration

Although the import tool does most of the configuration for ETIM classifications, some tasks must be performed manually.

1. Use Import Manager or IIEP and select the appropriate ETIM file version (as defined below) to perform the initial ETIM import. For more information, refer to **Creating a Data Import** or refer to **Creating an Inbound Integration Endpoint**.
2. Manually create the classification link from product to ETIM classification as defined in the **Create Product to ETIM Classification Link** documentation.
3. Either set the object type validity upon import, or manually make the ETIM attributes valid for the product object types as defined in the **Set Validity on ETIM Attributes for Products** documentation.

ETIM Versions

ETIM 5 and ETIM IXF classifications can be imported into STEP. The ETIM 5 import tool cannot delete ETIM data or mark ETIM data as obsolete. However, ETIM IXF formats will not import data marked as deleted into the system.

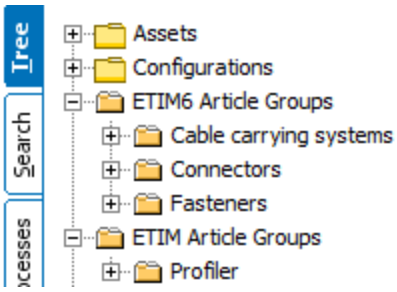
Multiple versions can exist in STEP simultaneously using multiple individual hierarchy structures.

- ETIM 5 uses CSV and each file contains a single language. For more information about the ETIM 5 formats, refer to **ETIM and ETIM v2Format**.
- ETIM 6, ETIM 7, ETIM 8, and ETIM 9 use the ETIM IXF format. ETIM IXF is the ETIM XML standard and includes multiple languages (English, German, Dutch, and a number of other European languages). For more information about the ETIM IXF format, refer to **ETIM IXF Format**.

Important: To avoid a corrupt import, do not manipulate the ETIM files prior to import. Filtering of classifications should be handled by the import tools.

ETIM Classification

ETIM classifications are imported into STEP as an alternate classification, below the ETIM Article Groups node in the Tree.

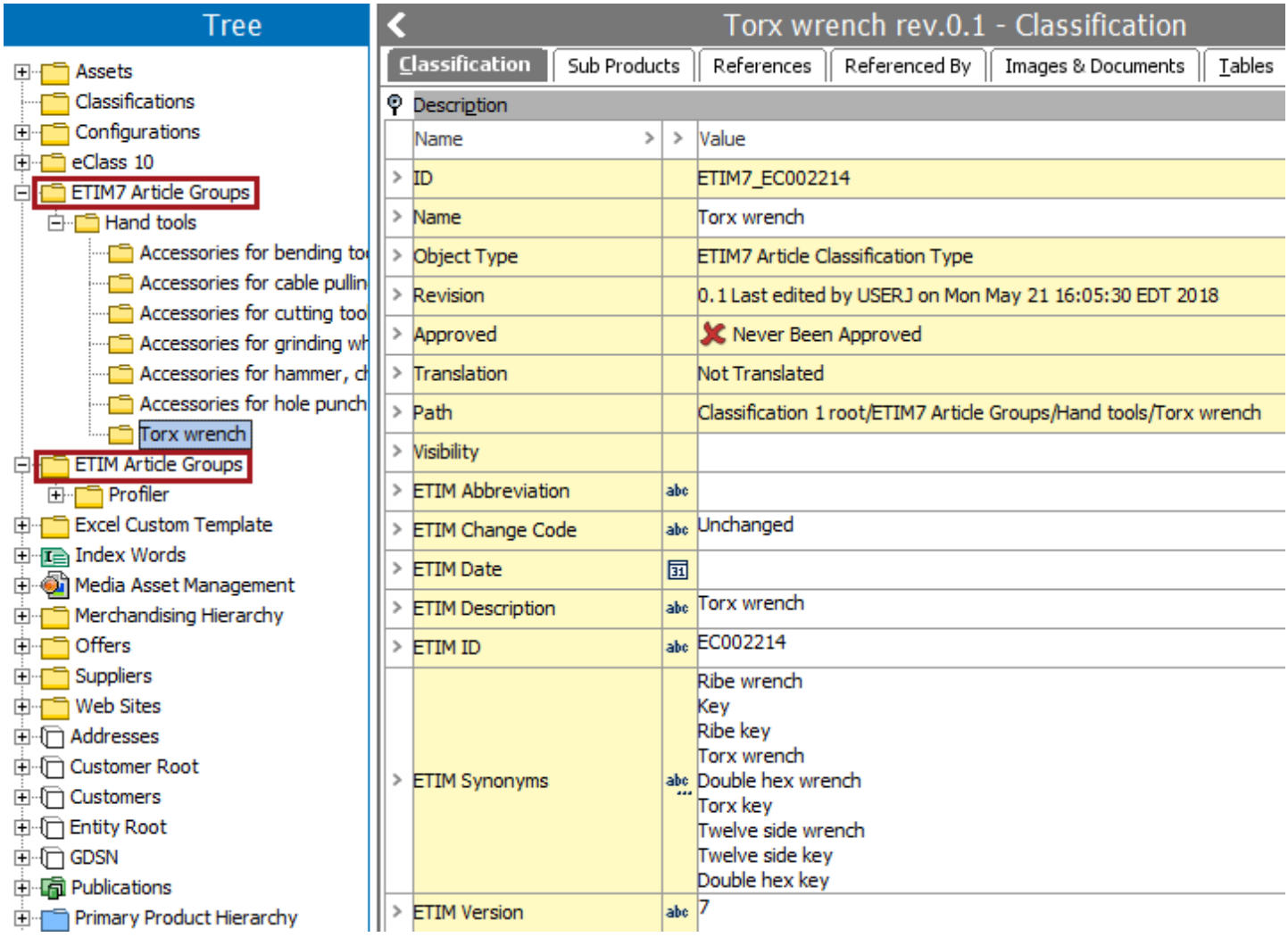


Each imported ETIM article group classification will include the following metadata attributes on the Classification tab within the Description flipper (as shown in the image below).

Attribute Name	Description
ID	**ETIM 5**: Value displays **ETIM** preceding the ETIM ID. **ETIM IXF**: Value displays **ETIM*_** preceding the ETIM ID; where * indicates the ETIM version number.
Name	The ETIM Description of the article group.
ETIM Abbreviation	**ETIM IXF** only: Abbreviation from ETIM.
ETIM Change Code	**ETIM IXF** only: Change Code values from the ETIM file.
ETIM Date	The date of the last change to the classification in ETIM.
ETIM Description	Description from ETIM, matches Name attribute.
ETIM ID	ID from ETIM, matches ID attribute.

Attribute Name	Description
ETIM Synonyms	Multivalued and contains 0 or more ETIM synonyms for the article classification.
ETIM Version	The Article Classification version from the ETIM system.

This image shows the metadata attributes for an ETIM version 7 article group. In this system, a second article group also exists for ETIM version 5, and is named 'ETIM Article Groups.'



Attributes are linked to the relevant ETIM classification and inherited only in that classification. For example, when an item is classified in ETIM classification A and is then reclassified to ETIM classification B, the attributes from ETIM classification A become orphaned, although the values remain on the item. At the same time, the item will inherit the valid ETIM attributes from classification B.

ETIM Feature Types

ETIM operates with 'features' that roughly translate to attributes and LOVs. An ETIM feature can be of one of the following types.

Type	Type Description	Result
Logic	The logical data type holds two values: true / false or yes / no.	An LOV with the ID 'ETIM Logical' is created in STEP. It contains the values true and false. Features are imported as attributes using the LOV.
Alphanumeric	The alphanumeric data type contains a set of predefined values.	An LOV contains the predefined set of values. Each value is provided with a value ID. Features are imported as attributes using the LOV.
Numeric	The numeric data type is a number that allows a unit.	Features are imported as Number validation type.
Range	The numeric range that allows a valid unit.	Features are imported as Number Range validation type.

Valid Values are Determined by ETIM Version

In ETIM, the set of valid values for a feature depends on the classification of the linked article and the ETIM version. In STEP, this is achieved by creating a new attribute for each classification the feature is linked into. Each of the attributes (for the same ETIM feature) use the same LOV but with an individual filter of the valid values.

Note: The core STEP solution does not offer any tools to validate that the user does not use an invalid unit when adding a value for an ETIM Classification.

ETIM 5

Typically, only one of the following ETIM 5 formats will be available in your import tools format list:

- ETIM - this import converter creates a new attribute for each classification the feature is linked into. Each of the attributes (for the same ETIM feature) uses the same LOV but with an individual LOV filter of the valid values.
- ETIM v2 - this import converter avoids having multiple attributes for the same feature by moving the LOV filter specifying the valid values from the attribute to the attribute link. This should make it far easier to import, export, and maintain the attributes.

For more information about the ETIM 5 formats, refer to **ETIM and ETIM v2 Format**.

ETIM IXF

- ETIM IXF, versions 6, 7, 8, and 9 - this import converter creates a new attribute for each classification the feature is linked into. Each of the attributes (for the same ETIM feature) uses the same LOV but with an individual LOV filter of the valid values.

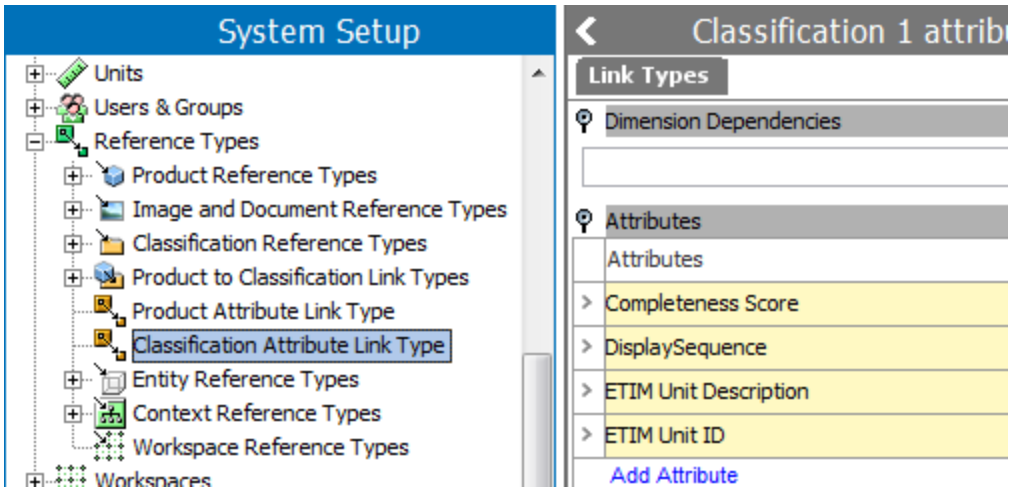
For more information about the ETIM IXF format, refer to **ETIM IXF Format**.

Valid Units are Determined by Classification to Attribute Link

An ETIM feature does not have a valid unit in ETIM 5 or ETIM IXF. However, when it is linked into an article classification, one (and only one) valid unit can be specified. This is different from STEP where an attribute has the same valid units no matter where the attribute is used.

Note: STEP does not include functionality to ensure only valid units are used when adding a value for an ETIM classification.

On the **Classification to Attribute Link**, the **ETIM Unit ID** and **Description** are stored as metadata to indicate that when used in this particular classification, only the unit specified is valid. Additional metadata includes 'ETIM Sort Number', 'ETIM Unit Description', and 'ETIM Unit ID.' The 'ETIM Sort Number' specifies the order in which ETIM determines the attribute is most relevant. If System Settings > 'Default Attribute to use as Display Sequence Attribute' is specified, the sort number will be put as metadata in this attribute, instead of using the 'ETIM Sort Number.' Notice however, that STEP does not support display sequence settings on classifications.



Note: It is possible to load various localizations of the ETIM classification system. For example, you can load an English version into the English context and a German version into the German context. While the IDs are the same across the various localizations, the descriptions of article groups, classifications, attributes, units, and lists of values are localized.

For more information about valid units in Web UI, refer to the **Classification-Specific Attribute Value Components** topic in the **Web User Interfaces** documentation.

Handling Import Errors

Depending on the locale configuration and the format of the import files, the import tool may report one of the following errors. Although the report can be long, the error is likely caused by the format of two attributes:

- **ETIM Version** - In the ETIM configuration file the values of this attribute contain 4 characters (5,00) while STEP only expects 3 characters.
- **ETIM Date** - The date format in the ETIM configuration file is dd.mm.yyyy hh:mm:ss while the STEP attribute is defined as 'ISO Date and Time' (yyyy-mm-dd hh:mm:ss).

These errors can be resolved by manually importing the problem attributes as defined in the **Manually Import ETIM Attribute Values** documentation.

ETIM and ETIM v2 Format

ETIM 5 uses CSV and each file contains a single language. Typically, only one of the following ETIM 5 formats will be available in your Import Manager:

- ETIM - this import converter creates a new attribute for each classification the feature is linked into. Each of the attributes (for the same ETIM feature) uses the same LOV but with an individual LOV filter of the valid values. This results in duplicate attributes and multiple LOVs when more than one classification has multiple sets of valid values.
- ETIM v2 - this import converter avoids having multiple attributes for the same feature by moving the LOV filter specifying the valid values from the attribute to the attribute link. This should make it far easier to import, export, and maintain the attributes.

For details about the ETIM versions available in STEP, refer to the **ETIM Format** documentation.

Format Availability

ETIM format is available for selection in:

- IIEP - refer to **Creating an Inbound Integration Endpoint**
- Import Manager - refer to **Creating a Data Import**

Mapping

This format does not require a data map.

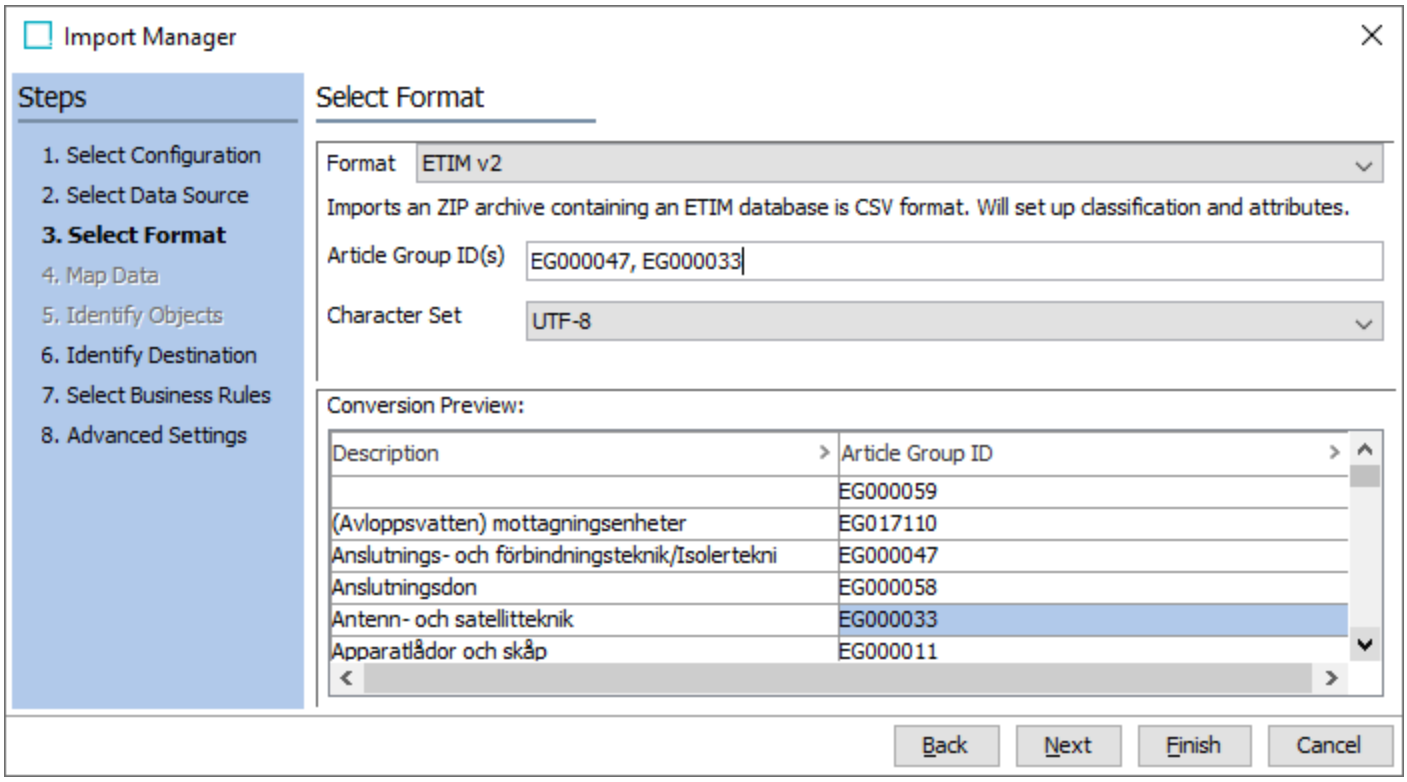
Inbound Data

Inbound Parameters

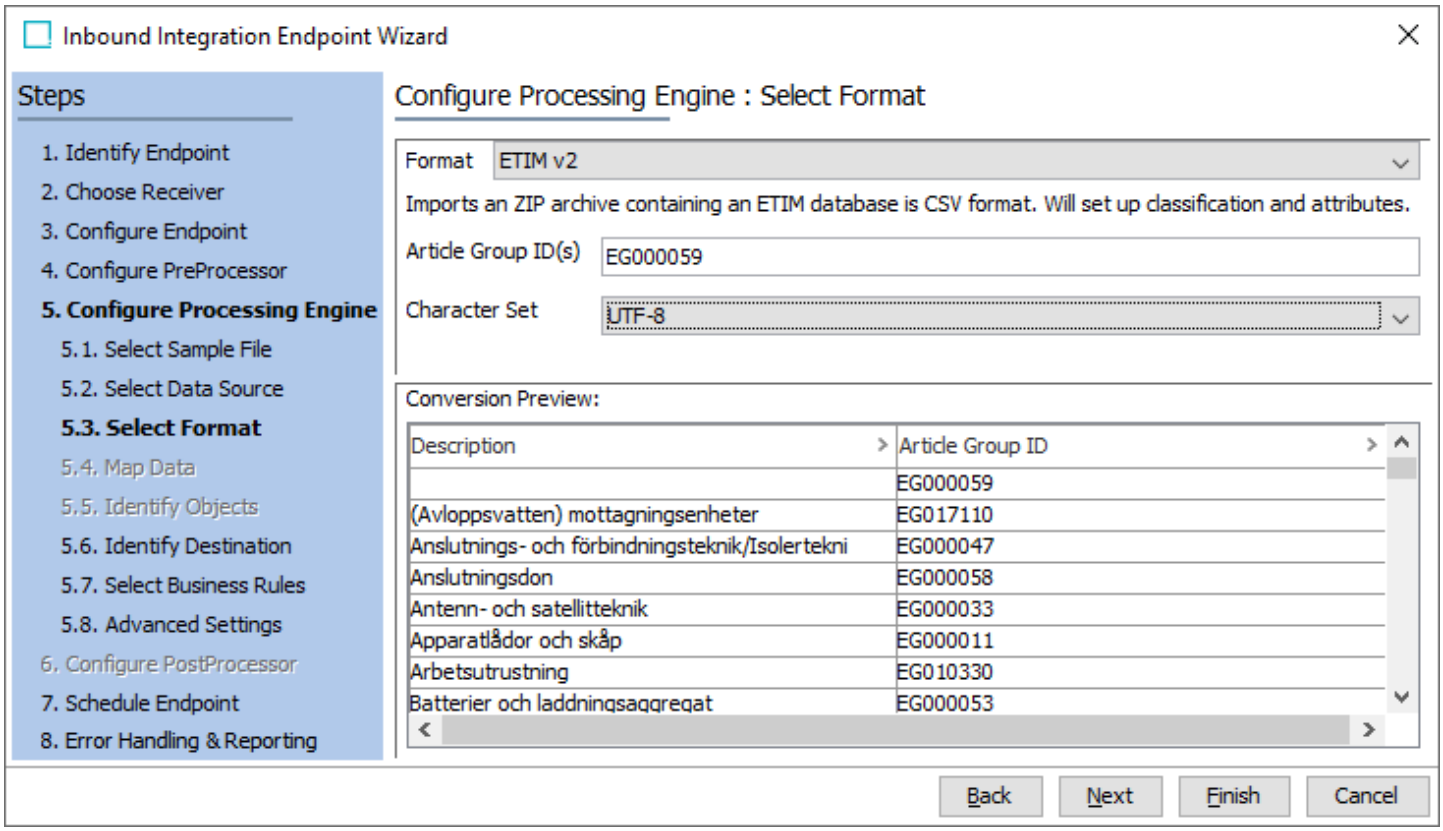
- **Article Group ID(s)** allows you to list one or more classifications as follows:
  - A single Article Group ID to import one classification.
  - A comma-separated list of Article Group IDs to import several classifications.
  - Blank to import the full ETIM classification system.
- **Character Set** allows you to select the option that includes any special characters required by the data being imported.
- **Conversion Preview** displays the data selected in the Article Group ID(s) parameter in the selected language.

Note: Once the ETIM import completes successfully, additional manual configuration is required. For details, refer to the **ETIM Configuration** section of the **ETIM Format** documentation.

Import Manager



IIEP


Inbound Integration Endpoint Wizard
Steps
1. Identify Endpoint
2. Choose Receiver
3. Configure Endpoint
4. Configure PreProcessor
5. Configure Processing Engine
5.1. Select Sample File
5.2. Select Data Source
5.3. Select Format
5.4. Map Data
5.5. Identify Objects
5.6. Identify Destination
5.7. Select Business Rules
5.8. Advanced Settings
6. Configure PostProcessor
7. Schedule Endpoint
8. Error Handling & Reporting
Configure Processing Engine : Select Format
Format
ETIM v2
Imports an ZIP archive containing an ETIM database is CSV format. Will set up classification and attributes.
Article Group ID(s)
EG000059
Character Set
UTF-8
Conversion Preview:
Description
Article Group ID
EG000059
(Avloppsvatten) mottagningsenheter
EG017110
Anslutnings- och förbindningsteknik/Isolertekni
EG000047
Anslutningsdon
EG000058
Antenn- och satellitteknik
EG000033
Apparatlådor och skåp
EG000011
Arbetsutrustning
EG010330
Batterier och laddningsaggregat
EG000053
Back
Next
Finish
Cancel

ETIM IXF Format

ETIM IXF is the ETIM XML format and includes multiple languages (English, German, Dutch, and a number of other European languages). The import converter creates a new attribute for each classification the feature is linked into. Each of the attributes (for the same ETIM feature) uses the same LOV but with an individual LOV filter for the valid values. STEP supports ETIM version 6, 7, 8, and 9; STEP also supports the dynamic, or pre-release versions in between major releases. For more information about ETIM functionality in STEP, refer to **ETIM Format**.

For ETIM IXF imports, deprecation is supported for classifications, features (i.e., attributes), classification-feature links, and LOV filtering. This means that when a new ETIM version is installed (in the ETIM IXF format):

- Any deprecated classifications will not be imported.
- Any deprecated links between classifications and features (i.e., attributes) will not be imported.
- Any deprecated values within an LOV will not be included in the LOV, via LOV filtering.

Existing ETIM taxonomy data on the system will never be removed via ETIM imports. If the ETIM Change Code in an ETIM import file is updated in any way for an existing ETIM data element on the system, the ETIM importer will update the ETIM Change Code metadata for those data elements.

Format Availability

ETIM IXF format is available for selection in:

- IIEP - refer to **Creating an Inbound Integration Endpoint**
- Import Manager - refer to **Creating a Data Import**

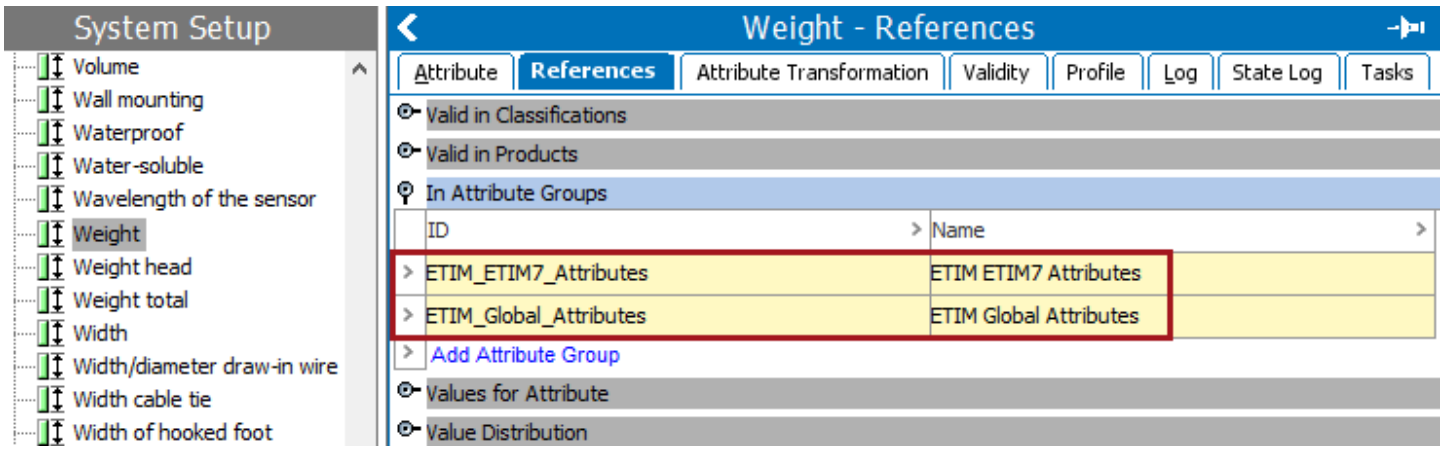
Mapping

This format does not require a data map.

Inbound Data

Importing ETIM data results in the creation of version-specific attribute groups, but the ETIM specification attributes created are not version specific. When the same specification attribute is required for multiple ETIM versions, it is linked to each applicable version-specific attribute group.

In the example below, the 'Weight' specification attribute is needed for both ETIM5 and ETIM7 data. The single attribute with the name 'Weight' is added to two attribute groups and is displayed on the References tab of the attribute under the 'In Attribute Groups' flipper.



Inbound Parameters

- **Language** allows selection of a single ISO Language Code to import. Multiple languages must be imported separately. In the Advanced Settings step you will select a context for the import.
- **Article Group ID(s)** allows the user to list one or more classifications as follows:
  - A single Article Group ID to import one classification.
  - A comma-separated list of Article Group IDs to import several classifications.
  - Blank to import the full ETIM classification system.
- **Prefix** allows the user to manage the taxonomies of the ETIM structures within STEP by prepending text for the prefix parameter value. By default, this will be read from the file being imported into STEP, but it can be changed depending on how it should be displayed in STEP, e.g., if it is an ETIM version 6 file it will read ETIM6, and if it is ETIM version 7 it will read ETIM7. However, if the file is an ETIM pre-release file, sometimes known as a dynamic file, it will load with the ETIM version number followed by a '-D.' So an ETIM version 6 dynamic file will automatically be populated as ETIM6-D.

  Note: This ETIM-D version number should only be changed if a user wants to display multiple taxonomies for each dynamic version of the ETIM 6 imports. Otherwise each dynamic ETIM file will continue to load into the ETIM6-D file structure.

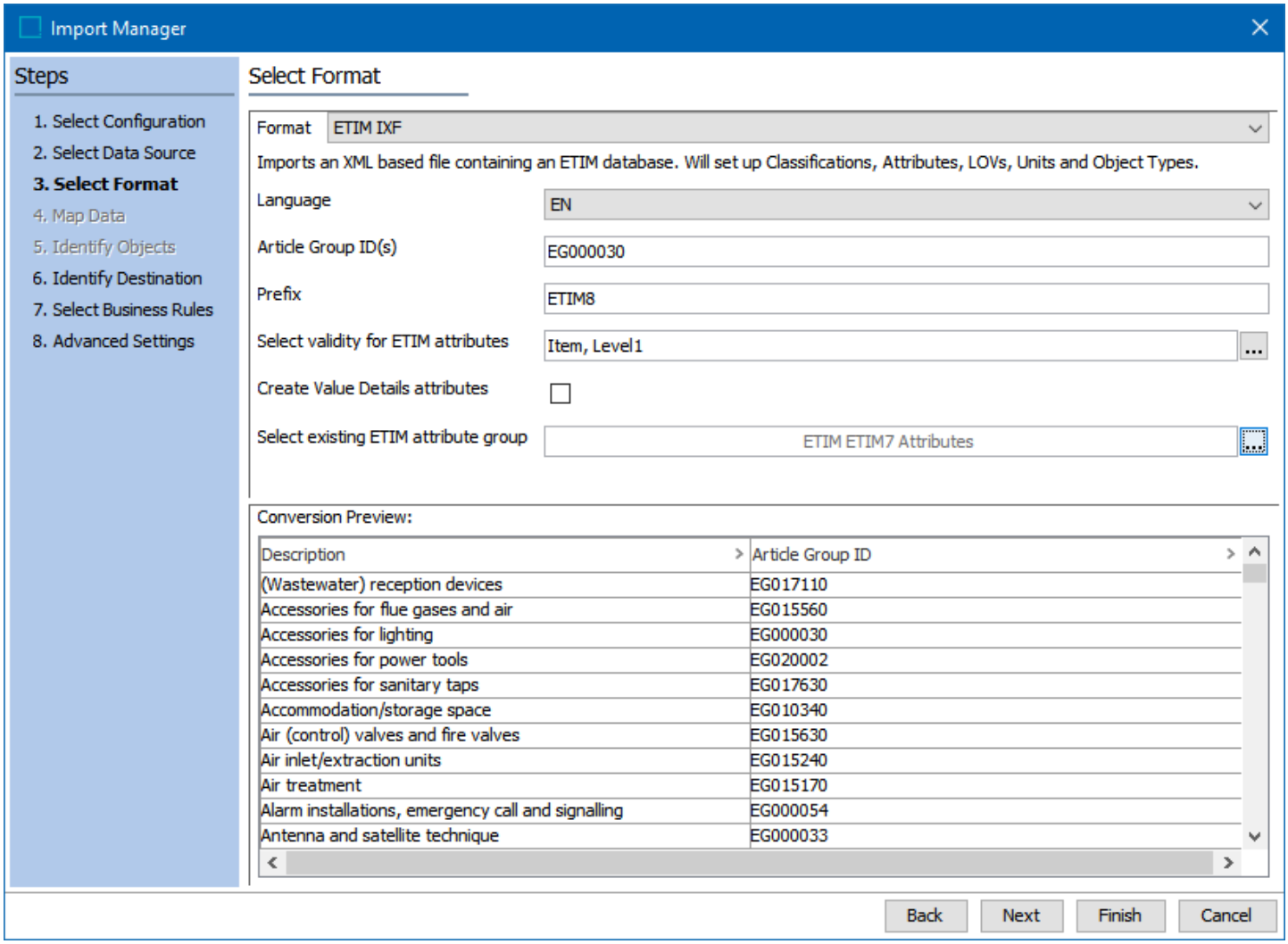
- **Select validity for ETIM attributes** allows the user to select the object types that are valid for the ETIM attributes. Click the ellipsis button (...) to display a list of object types that can be checked for validity in both the Import Manager and IIEP.
- **Create Value Details attributes**, when checked, a new attribute group with the name 'ETIM Value Details Attributes' is created. The attributes created in this group are written with the ID composed of the prefix 'ETIM_', the ETIM attribute ID, and the text '_FVD' (which indicates FValue Details), 'ETIM _EF007220_ FVD.' These attributes (or the group) will be made valid on the object types selected in the previous parameter **Set validity for ETIM attributes**, and then mapped to the 'Value Details' target. When exporting

this data, using the 'ETIM FVALUE Details' aspect, allows the value from these attributes to be output in the FVALUE_DETAILS tag.

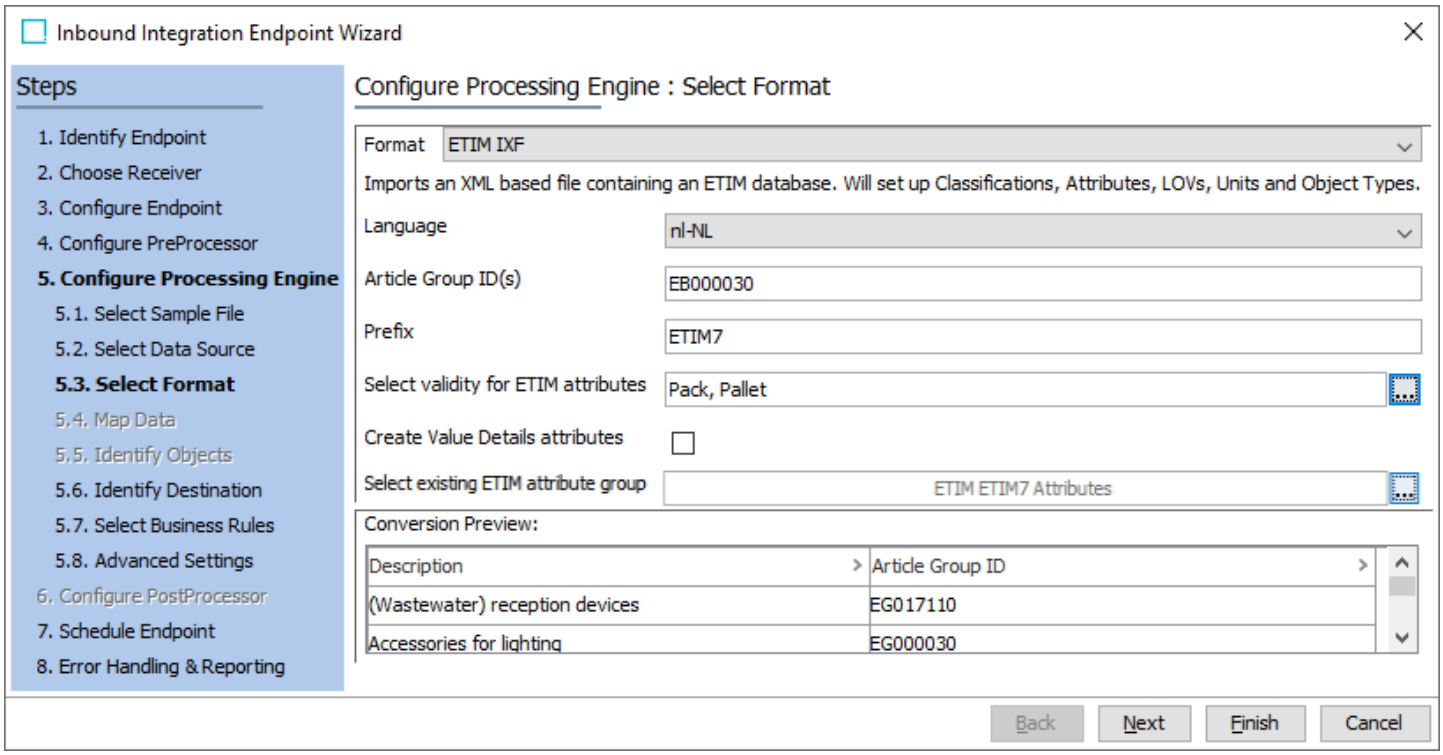
- **Select existing ETIM attribute group** allows users with an existing ETIM version already installed in their system to verify if there are changes between existing ETIM attributes on the system and incoming ETIM attributes (e.g., a change in the validation base type, unit, and/or LOV). If changes are found within a duplicate ETIM attribute, a new ETIM attribute will be created with a different ID, and this ETIM attribute will reflect the changes based on the newly imported ETIM version. **This parameter must be filled in for the import to work.**
- **Conversion Preview** displays article groups that were read from the ETIM import file in the selected language.

This should complete the ETIM IXF configuration. If changes need to be made to the configuration, refer to the **ETIM Configuration** section of the **ETIM Format** documentation.

Import Manager



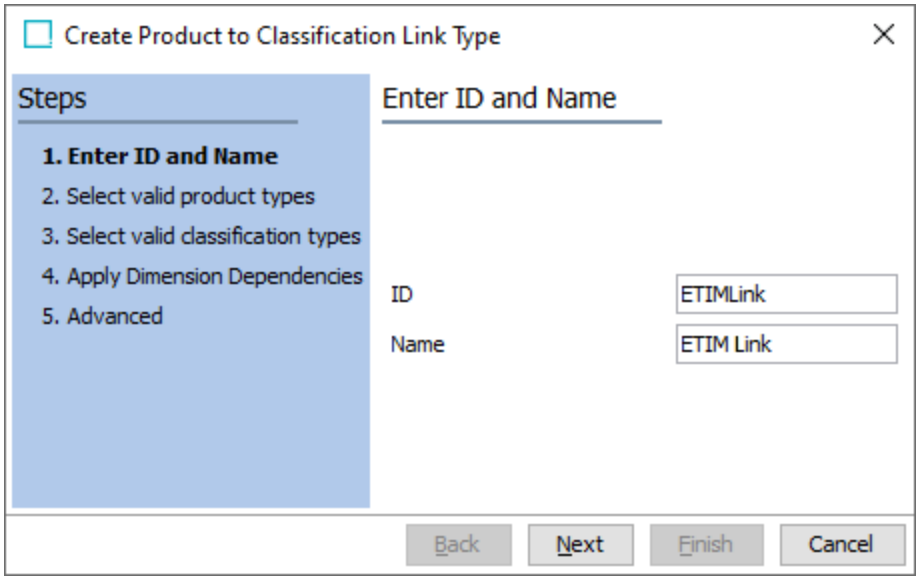
IIEP


Inbound Integration Endpoint Wizard
Steps
1. Identify Endpoint
2. Choose Receiver
3. Configure Endpoint
4. Configure PreProcessor
5. Configure Processing Engine
5.1. Select Sample File
5.2. Select Data Source
5.3. Select Format
5.4. Map Data
5.5. Identify Objects
5.6. Identify Destination
5.7. Select Business Rules
5.8. Advanced Settings
6. Configure PostProcessor
7. Schedule Endpoint
8. Error Handling & Reporting
Configure Processing Engine : Select Format
Format
ETIM IXF
Imports an XML based file containing an ETIM database. Will set up Classifications, Attributes, LOVs, Units and Object Types.
Language
nl-NL
Article Group ID(s)
EB000030
Prefix
ETIM7
Select validity for ETIM attributes
Pack, Pallet
Create Value Details attributes
Select existing ETIM attribute group
ETIM ETIM7 Attributes
Conversion Preview:
Description
Article Group ID
(Wastewater) reception devices
EG017110
Accessories for lighting
EG000030
Back
Next
Finish
Cancel

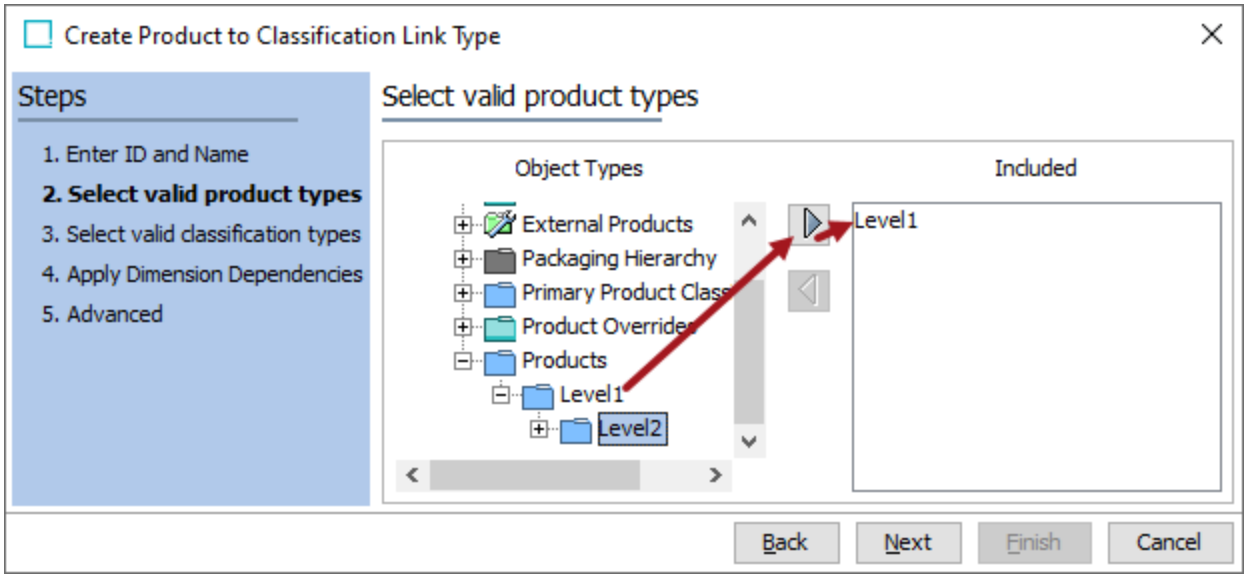
Create Product to ETIM Classification Link

For details about other setup required for using ETIM classifications, refer to the **ETIM Configuration** section of the **ETIM Format** documentation.

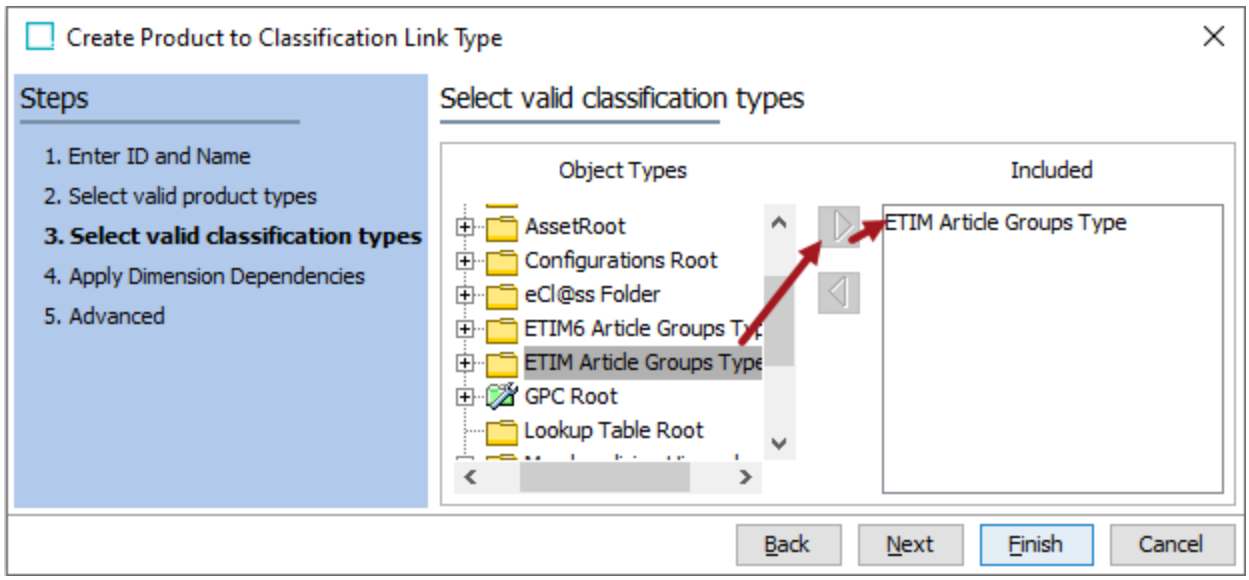
1. System Setup > Reference Types > **Product to Classification Link Types**, right-click and create a new Product to Classification Link Type.
2. Type an ID and Name for the new link type and click the **Next** button.



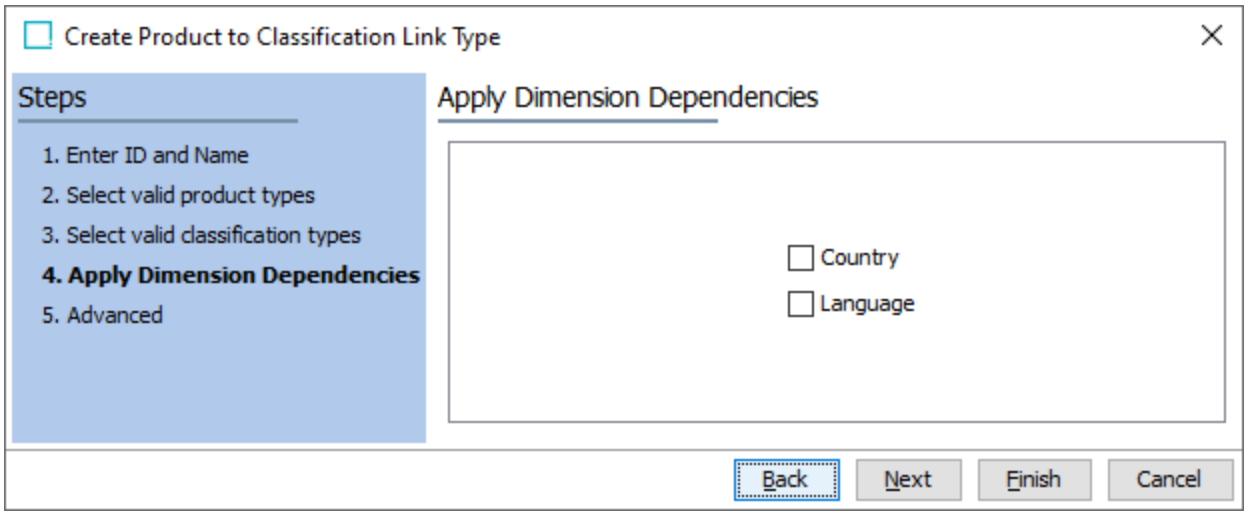
3. Select the valid product type(s), click the arrow button to move to the 'Included' list, and click the **Next** button.



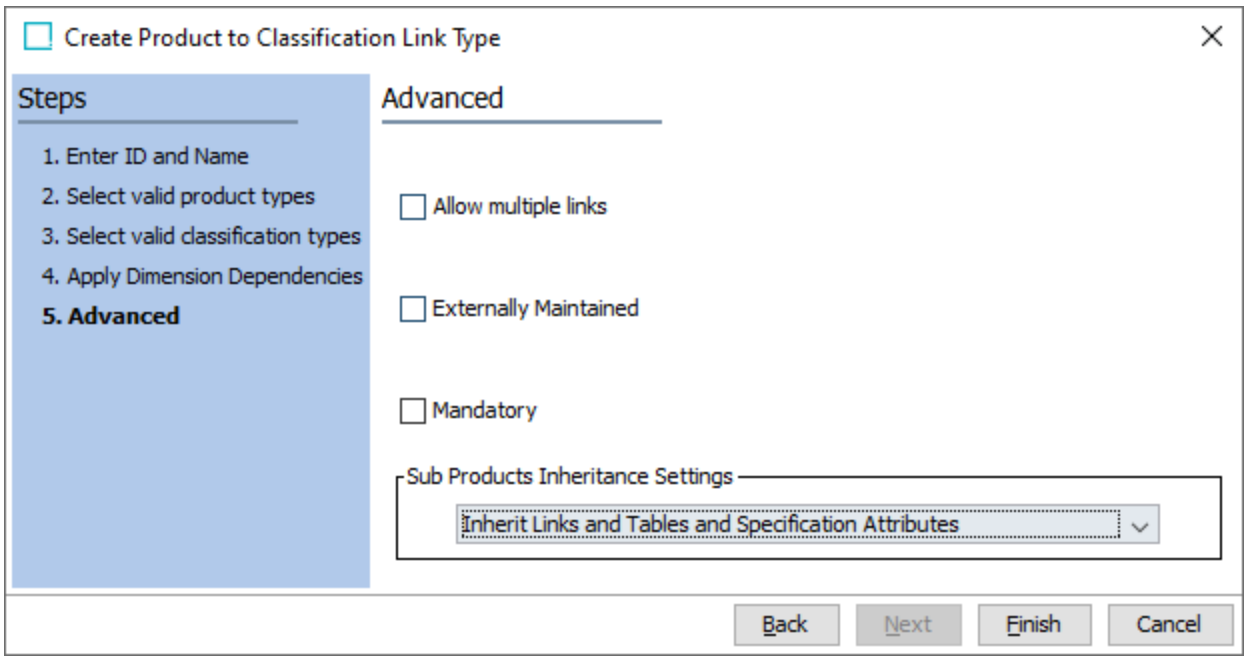
4. Select the valid classification type(s), click the arrow button to move to the 'Included' list, and click the **Next** button.



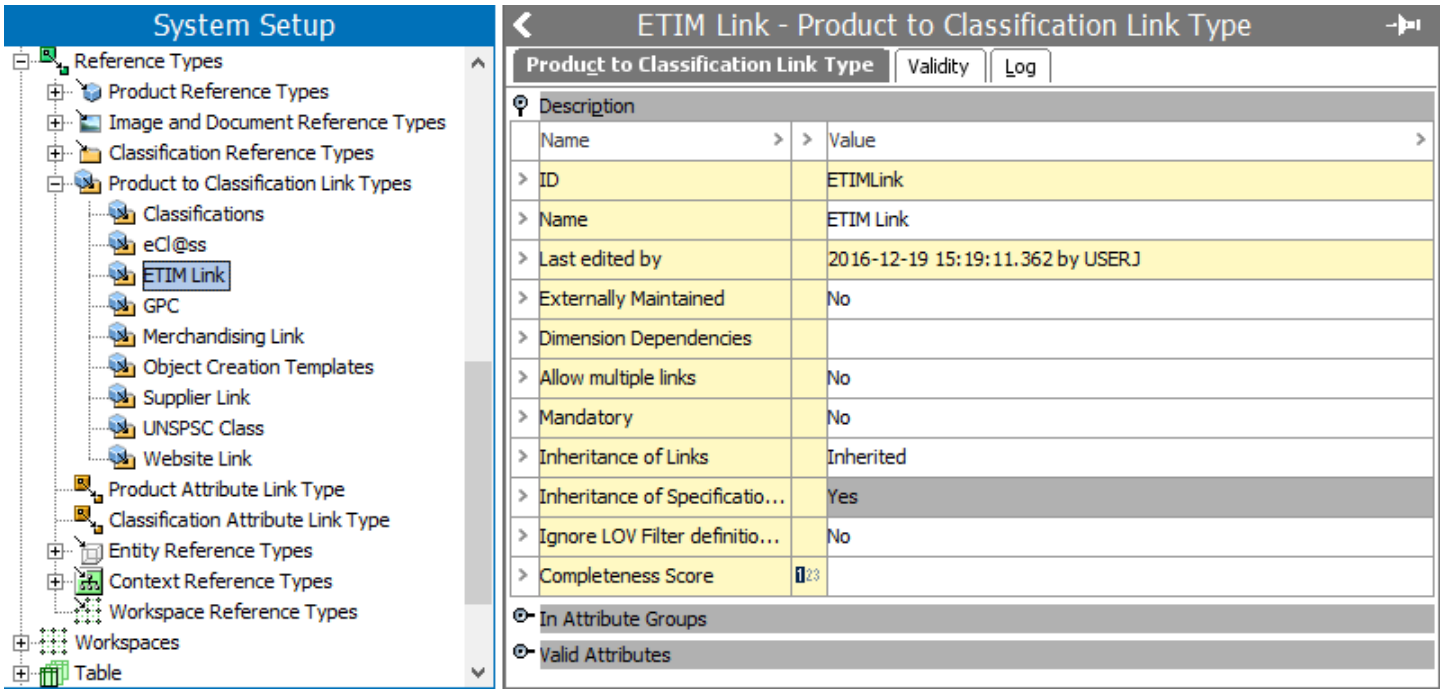
5. Select any necessary dimension dependencies and click the **Next** button.



6. For the Advanced options, uncheck all check boxes and for Sub Products Inheritance Settings, select 'Inherit Links and Tables and Specification Attributes.'



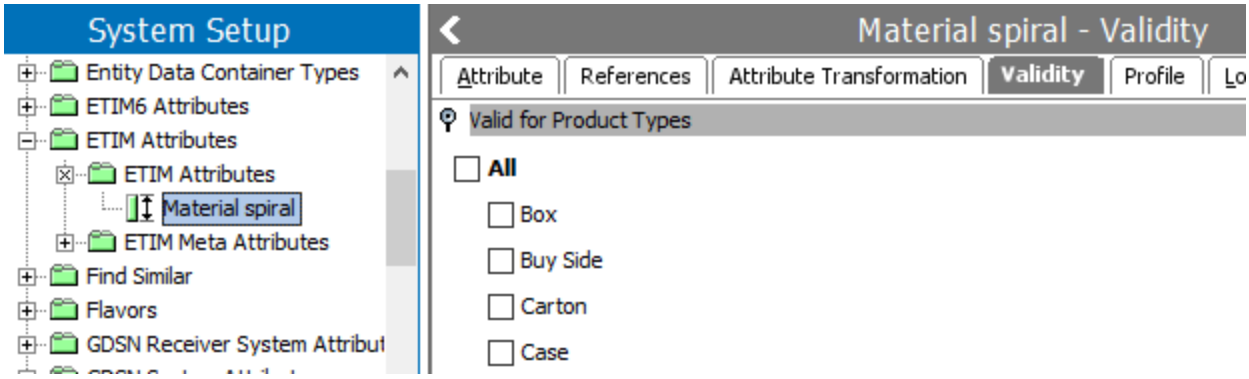
7. Click the **Finish** button and verify the correct settings are displayed in the editor.



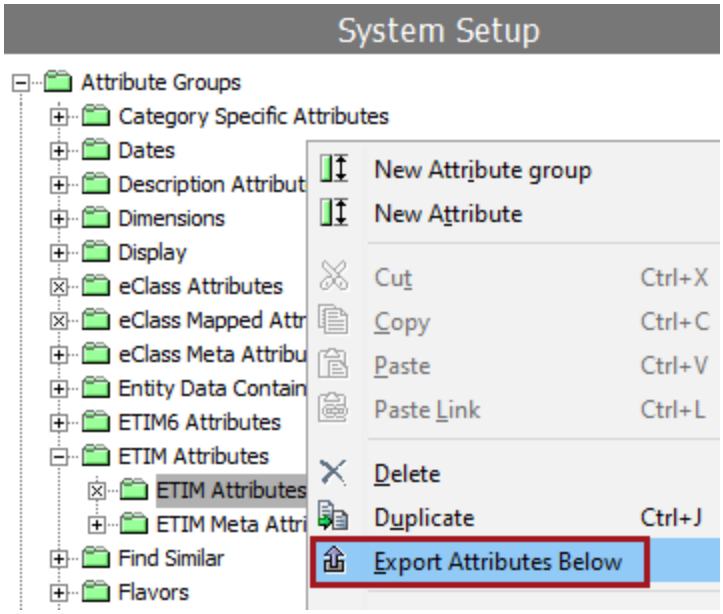
Set Validity on ETIM Attributes for Products

For details about other setup required for using ETIM classifications, refer to the **ETIM Configuration** section of the **ETIM Format** documentation.

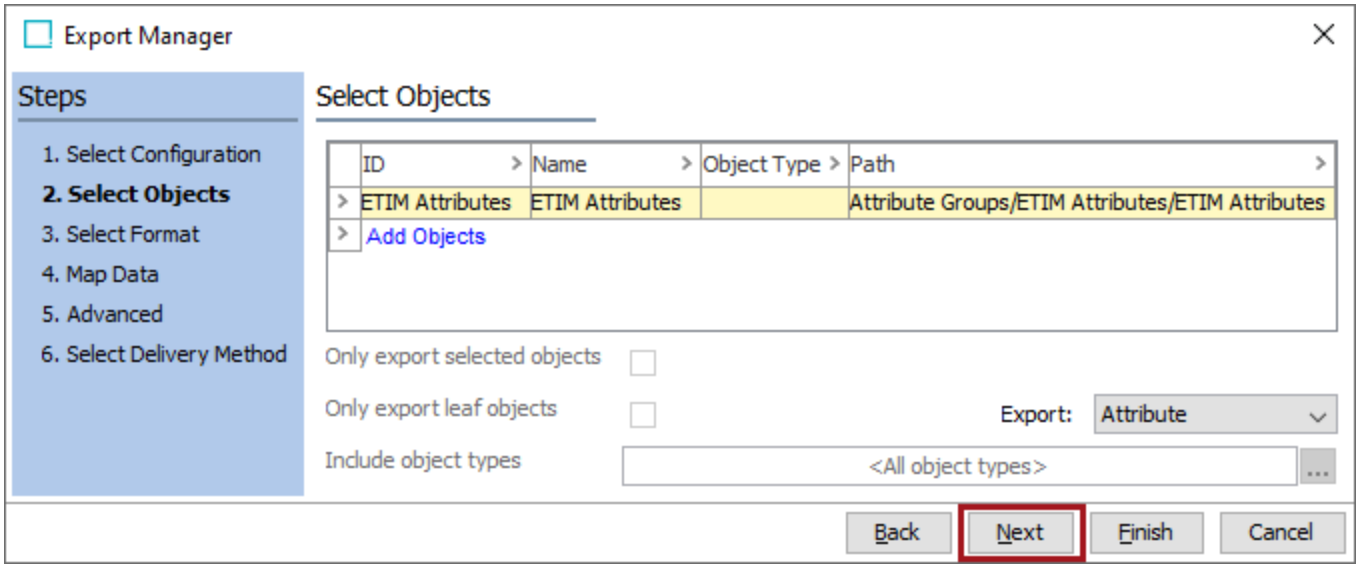
Use the following steps to set the valid product type on the Validity tab for the exported ETIM attributes.



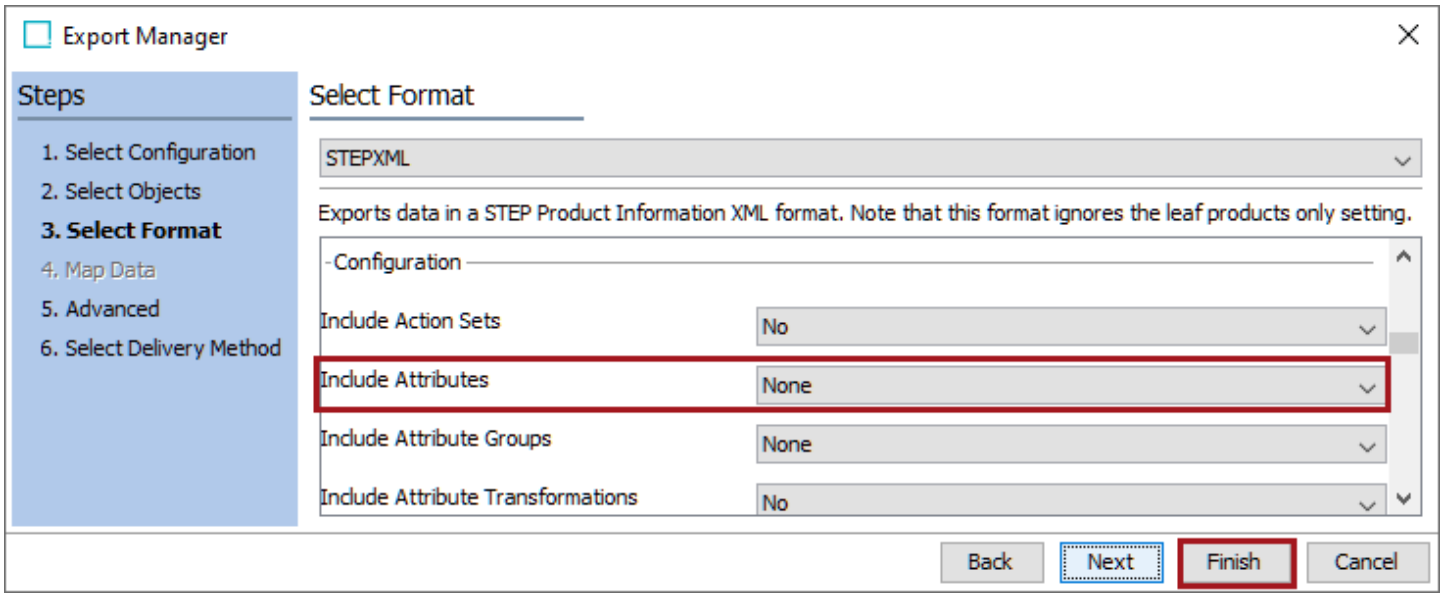
1. Determine the product object type that should be selected on the ETIM Attributes Validity tab.
2. System Setup > Attribute Groups > ETIM Attributes > ETIM Attributes, right-click and select **Export Attributes Below**. This assumes that you want to set validity on all ETIM attributes. If needed, select only the subset that you want to modify and export those.



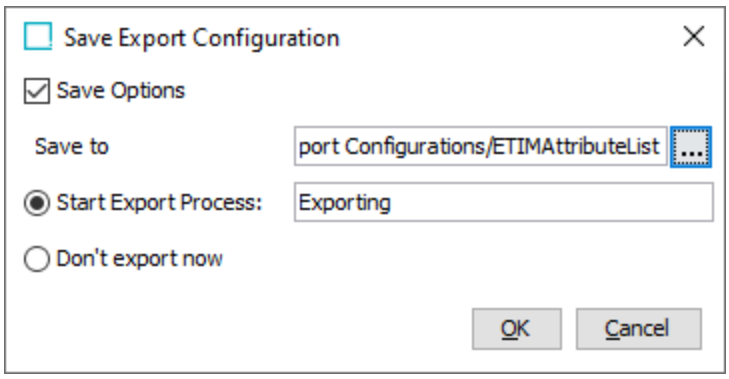
3. The Export Manager displays with the selected ETIM Attributes, click the **Next** button.



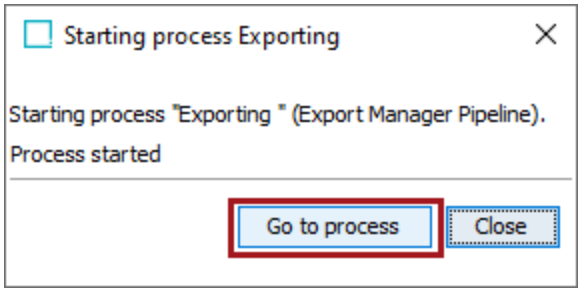
4. To ensure the smallest XML export file, on Select Format step:
   - Set **STEPXML** as the format.
   - Set the **Include Attributes** parameter to **Selected**.
   - Set all other parameters to No / None.
   - Click **Finish**.



5. On the Save Export Configuration dialog, set the options as desired, and click **OK** to start the export.



6. Click the **Go to process** button.

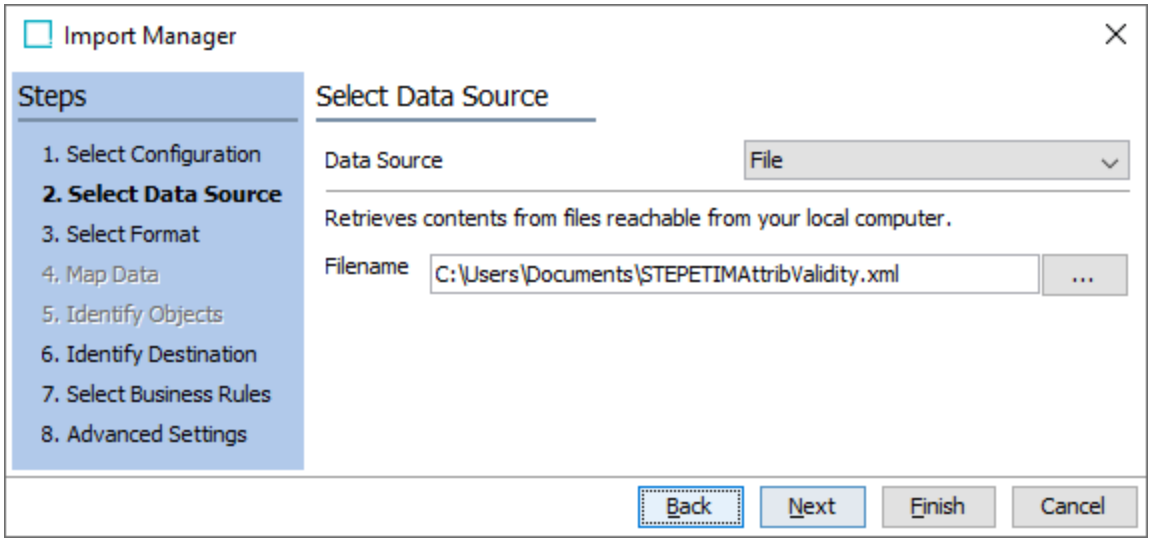


7. Open the export file for editing and make the following updates as shown in the image below. In the example, 'Item' is the product object type that will be set as valid.

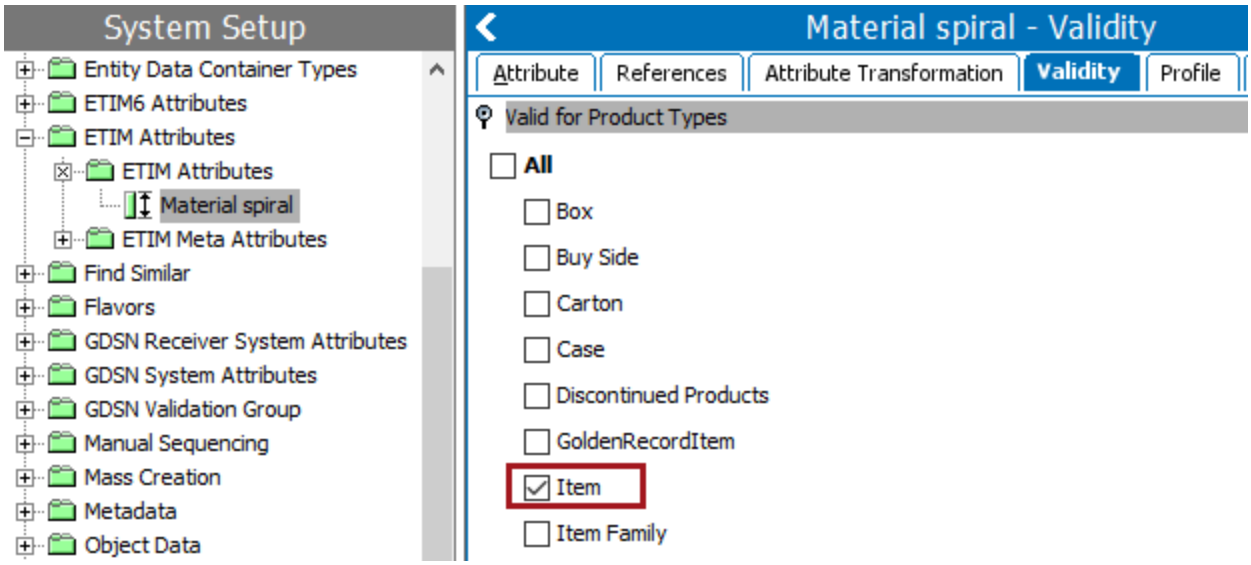
   Below each instance of the <AttributeGroupLink AttributeGroupID="ETIM Attributes"/> tag, add the text **<UserTypeLink UserTypeID="Item"/>**, substituting the text 'Item' with the name of your own desired product object type determined in the first step. The double quote marks are required.

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Configuration:
<STEP-ProductInformation ResolveInlineRefs="true">
<AttributeList ExportSize="Selected"/>
</STEP-ProductInformation>

Export from Primary Product Hierarchy
Classifications All
Products All
Assets All
-->
<STEP-ProductInformation ExportTime="2017-01-05 09:35:33" ExportContext="Context1"
ContextID="Context1" WorkspaceID="Main" UseContextLocale="false">

  <AttributeList>

    <Attribute ID="ETIM EF021685 A" MultiValued="false" ProductMode="Normal" FullTextIndexed="false"
    ExternallyMaintained="false" Derived="false" HierarchicalFiltering="true"
    ClassificationHierarchicalFiltering="true" Selected="true" Referenced="true">
      <Name>Material spiral</Name>
      <ListOfValueLink ListOfValueID="ETIM EF021685"/>
      <AttributeGroupLink AttributeGroupID="ETIM Attributes"/>
      <UserTypeLink UserTypeID="Item"/>
    </Attribute>

    <Attribute ID="ETIM EF008218 A" MultiValued="false" ProductMode="Normal" FullTextIndexed="false"
    ExternallyMaintained="false" Derived="false" HierarchicalFiltering="true"
    ClassificationHierarchicalFiltering="true" Selected="true" Referenced="true">
      <Name>Position cable feed</Name>
      <ListOfValueLink ListOfValueID="ETIM EF008218"/>
      <AttributeGroupLink AttributeGroupID="ETIM Attributes"/>
      <UserTypeLink UserTypeID="Item"/>
    </Attribute>
```

8. Save the modified file with a new name.
9. Open Import Manager (File > Import > Data) and import the modified file. For details on the import process, refer to **Creating a Data Import**.



10. Verify that imported attributes now display the valid product type.



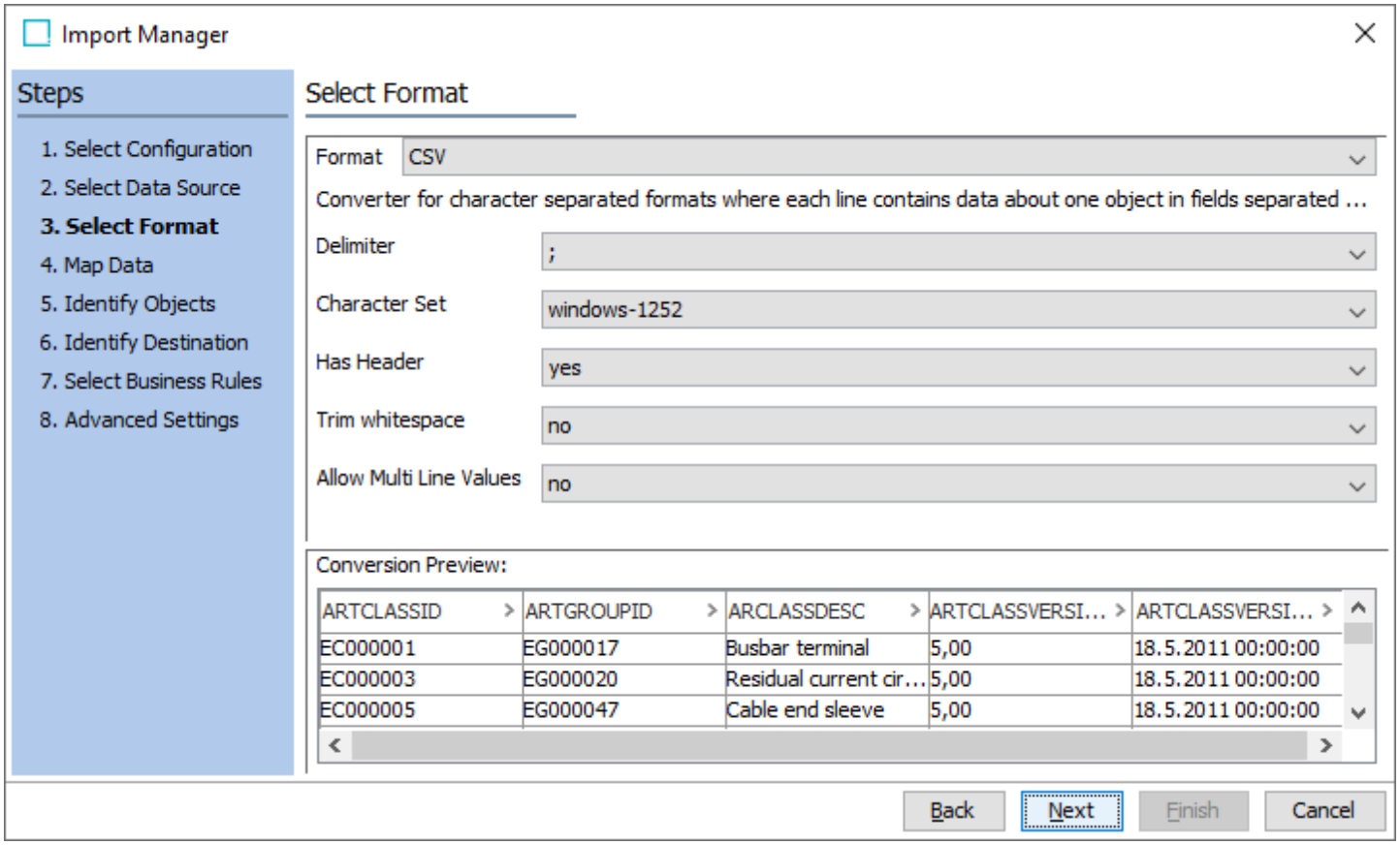
Manually Import ETIM Attribute Values

Occasionally, while using Import Manager to import a set of ETIM classifications, ETIM attribute values are not imported because the number of allowed characters / or the validation base type in STEP does not match the values included in the ETIM file. When this happens, use the following steps to manually import the ETIM attributes.

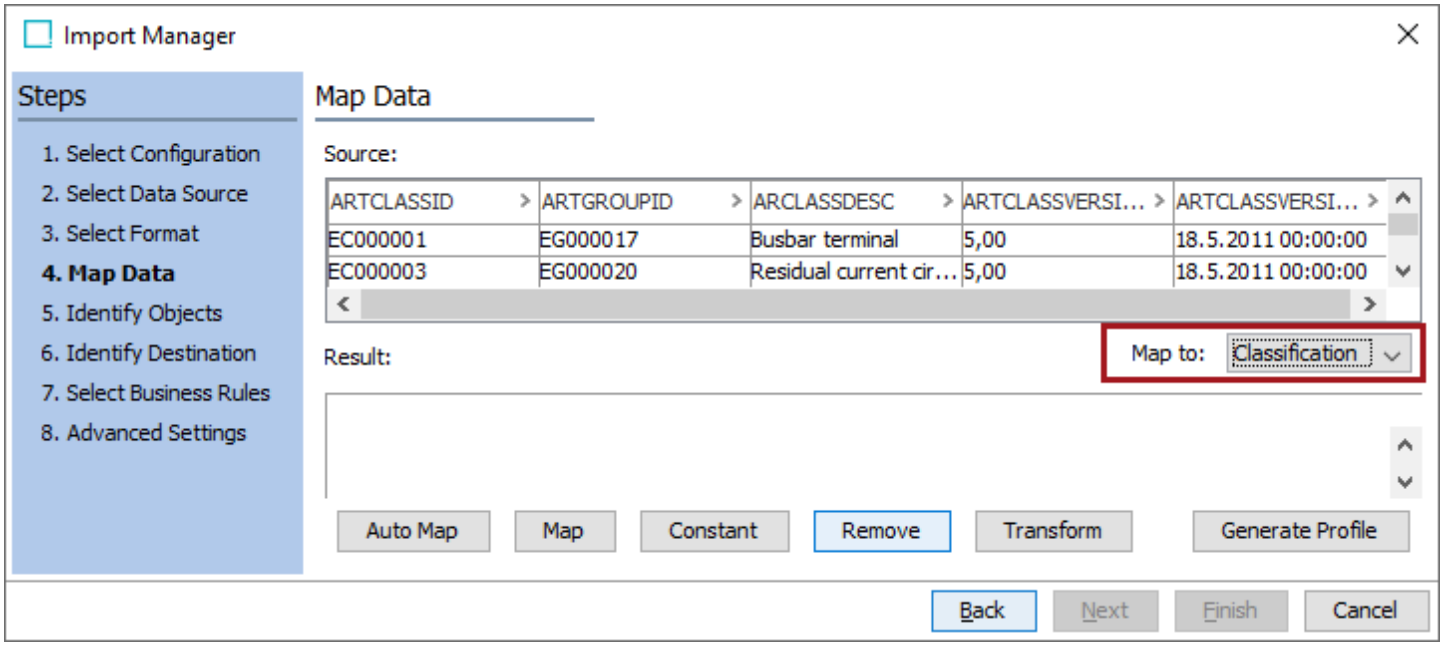
1. Start the Import Manager and complete the following steps:
   - **Import Manager - Select Configuration**
   - **Import Manager - Select Data Source**

2. For the Import Manager Select Format step, select the appropriate ETIM version.
   - ETIM version 5 using **ETIM and ETIM v2 Format**
   - ETIM versions 6, 7, and 8 using **ETIM IXF Format**

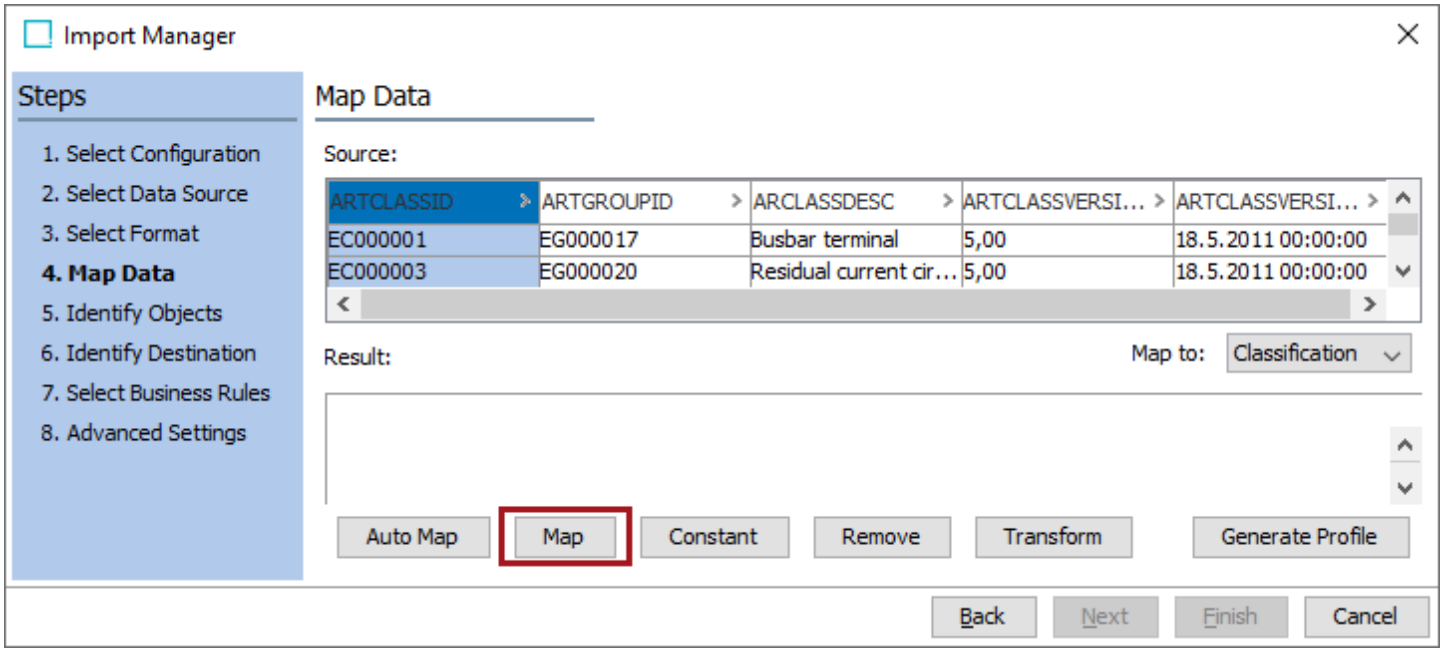
3. On Import Manager Select Format, choose **CSV** and leave the other settings as default. The Conversion Preview is displayed.



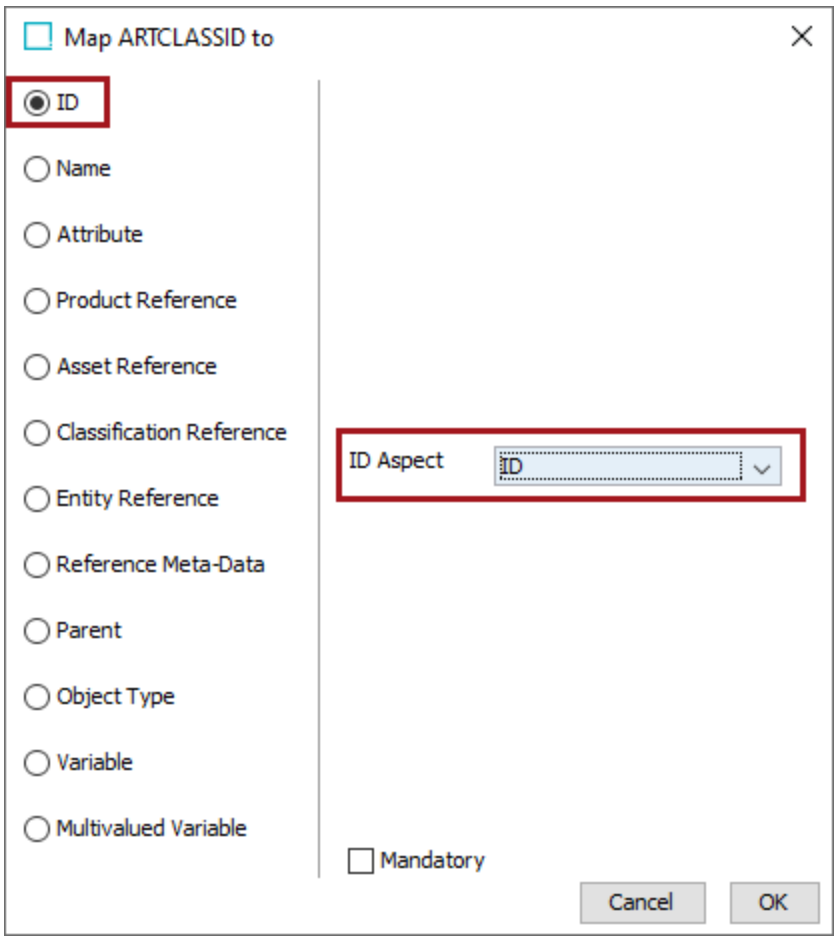
4. On Import Manager Map Data, set **Map to** as **Classification**.



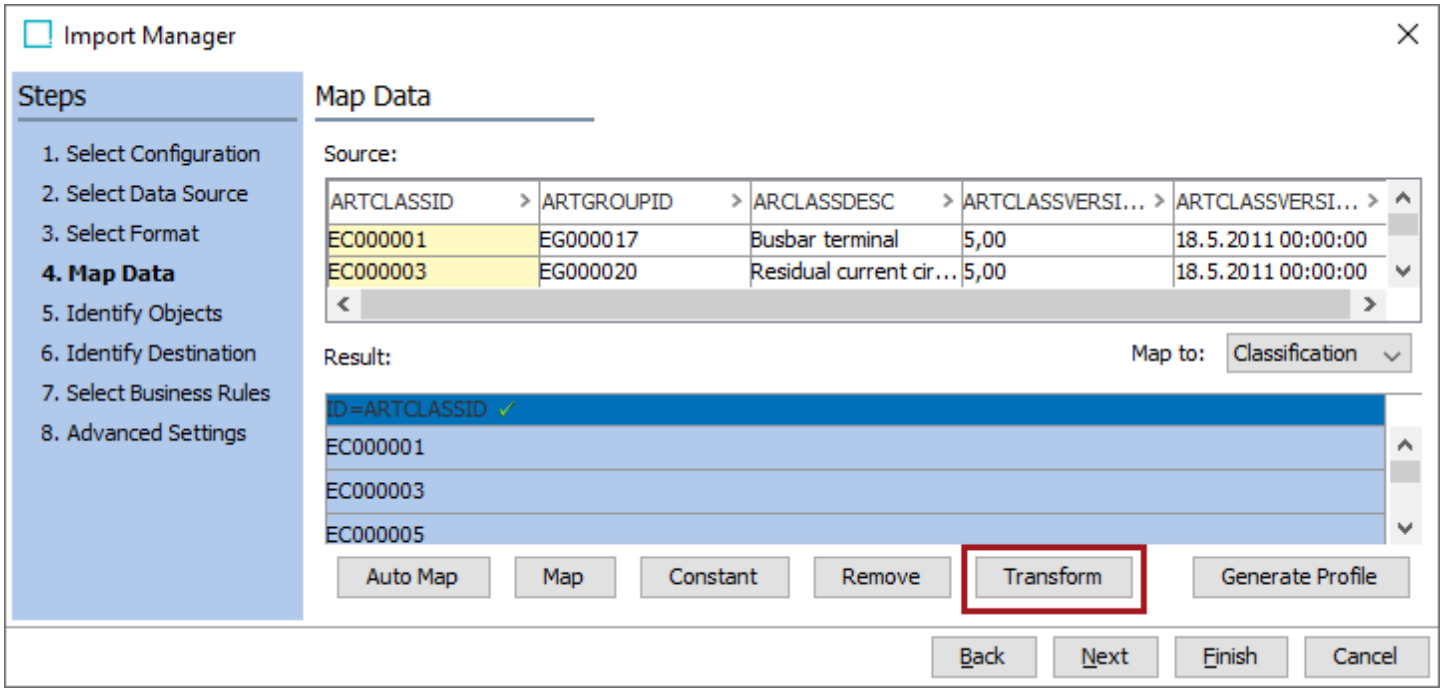
5. In the Source table, select the **ARTCLASSID** column and click the **Map** button below the Result table.



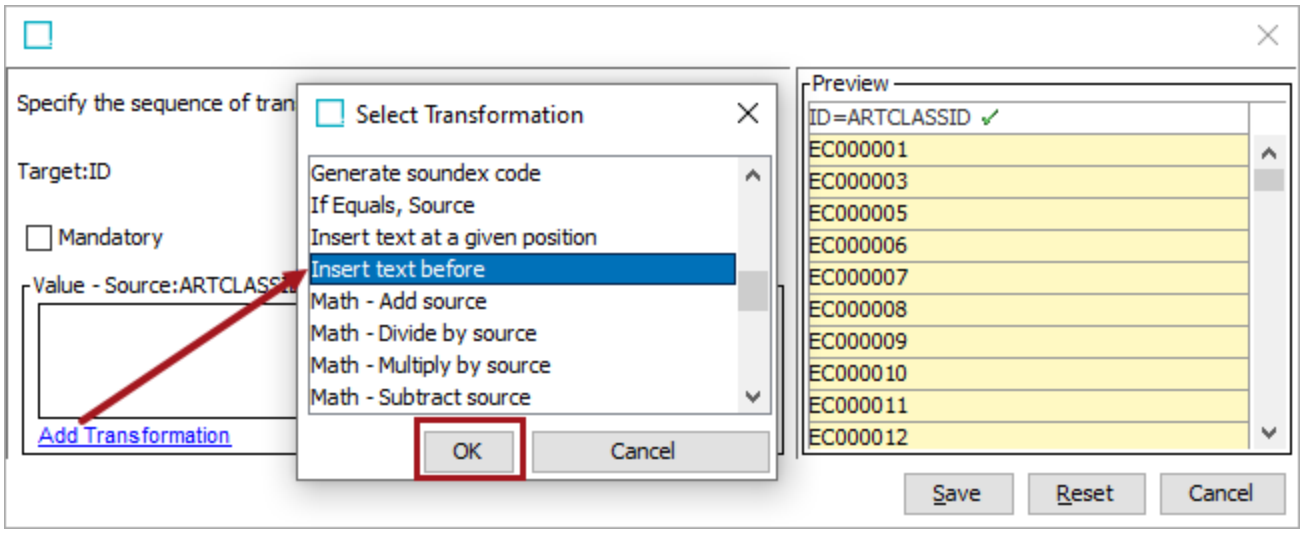
6. In the Map To window, select the **ID** radio button, set the **ID Aspect** to **ID**, and click **OK**.



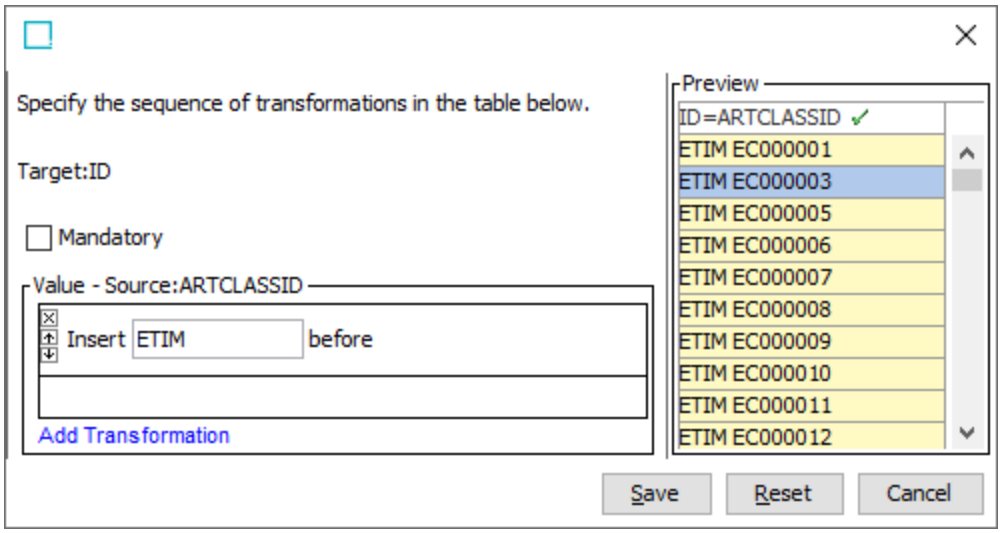
7. In the Result table, select the **ID=ARTCLASSID** column and click the **Transform** button.



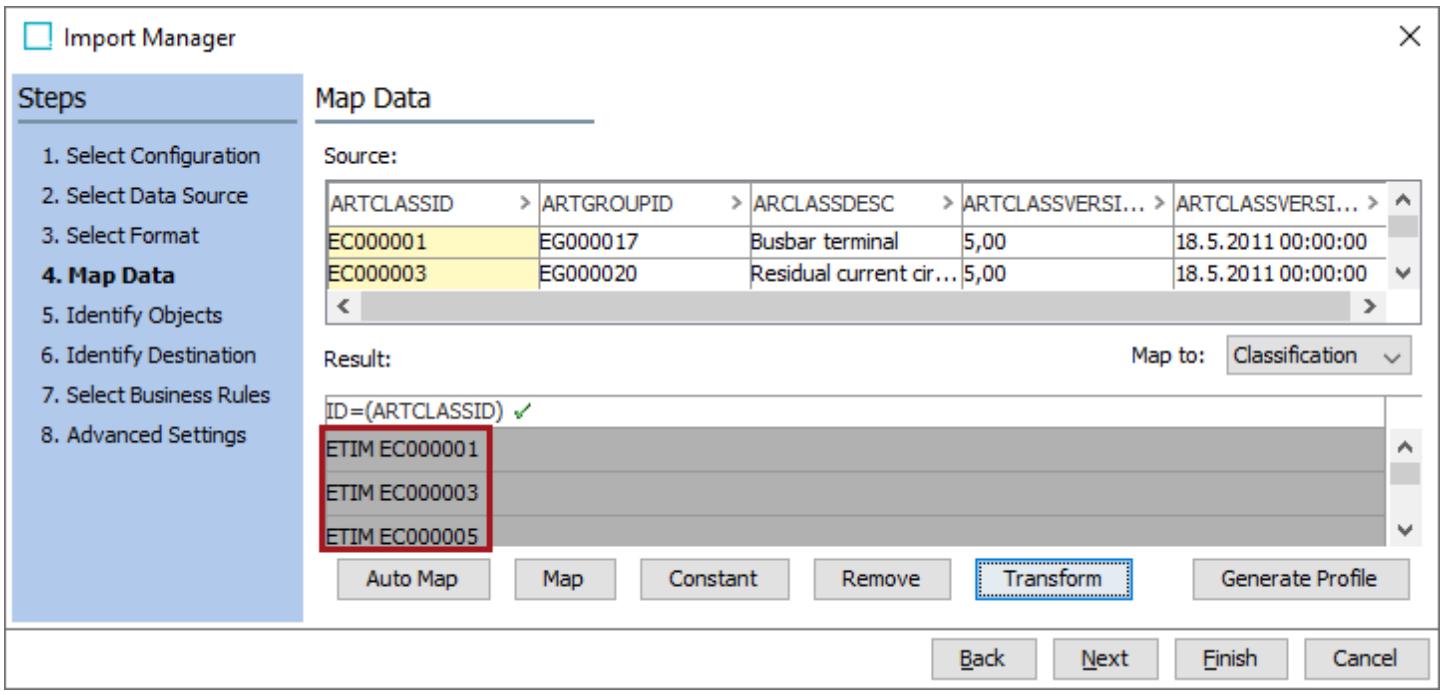
8. In the Transformation dialog, click the **Add Transformation** link, select the **Insert text before** option, and click **OK**.



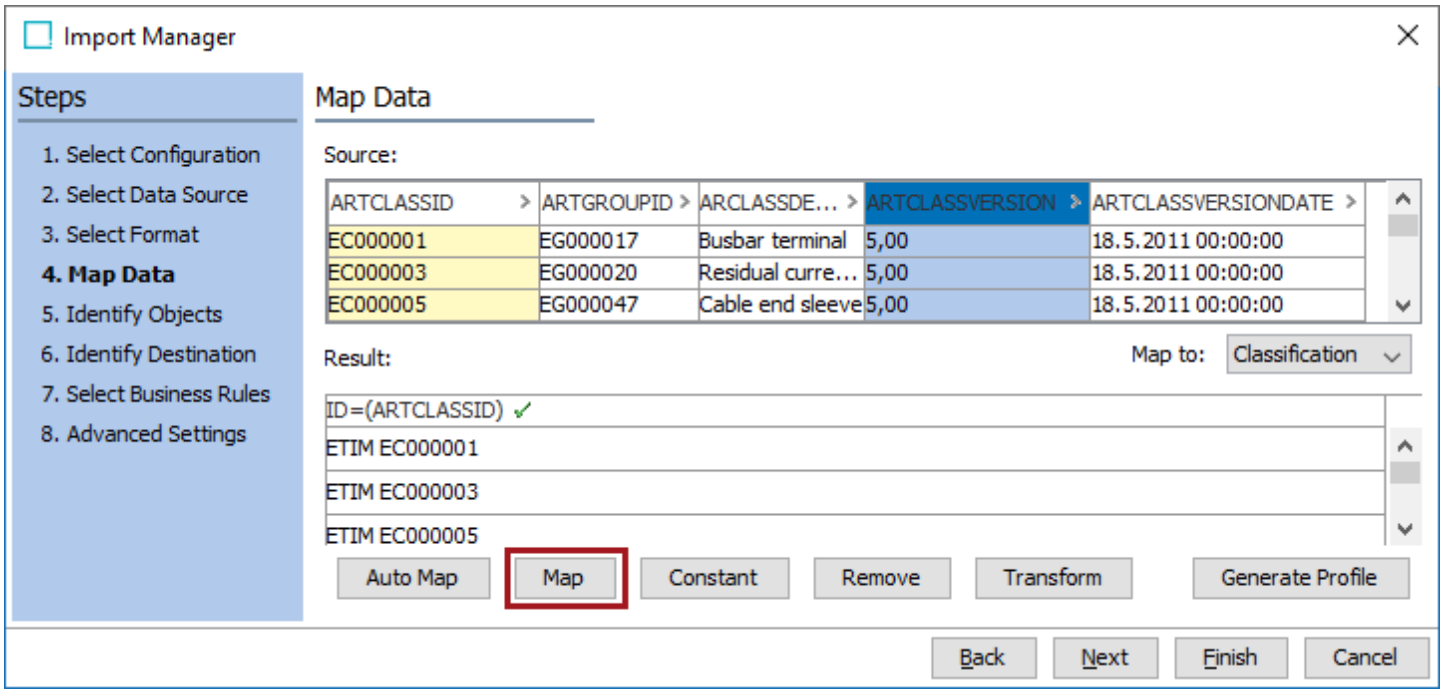
9. Enter the ETIM version your want to import (including a trailing space) as the value to insert in order to match the ID in STEP and click the **Save** button. This text entered is based on the suffix added when the ETIM Article Groups were imported. The results are displayed in the Preview pane.



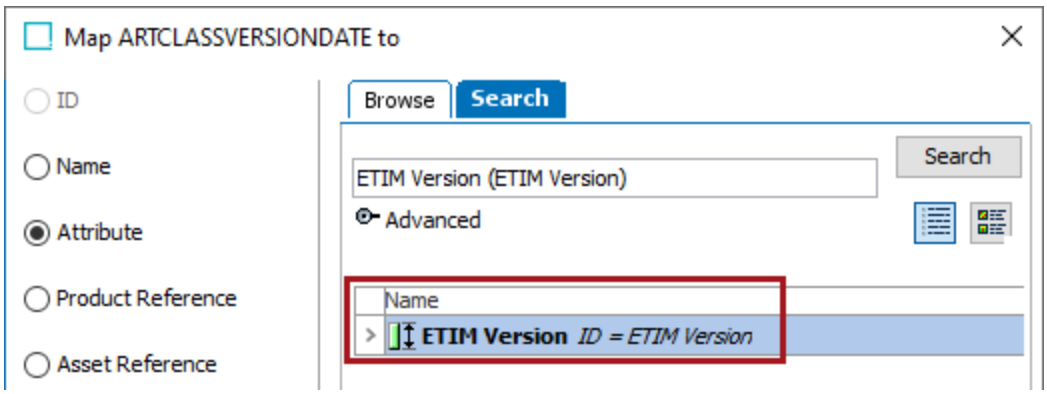
10. The Result table ID=ARTCLASSID column now displays the selected text preceding the ID originally displayed in the Source table.



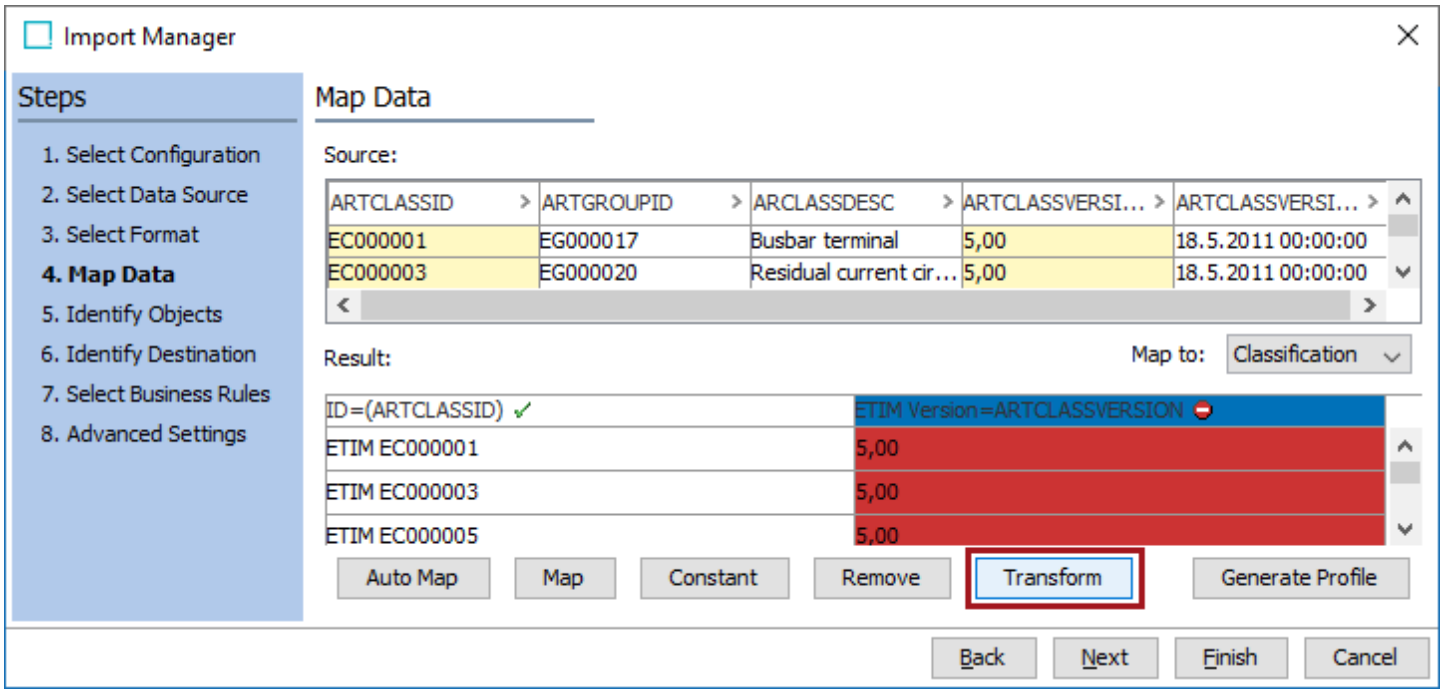
11. In the Source table, select the **ARTCLASSVERSION** column and click the **Map** button below the Result table.



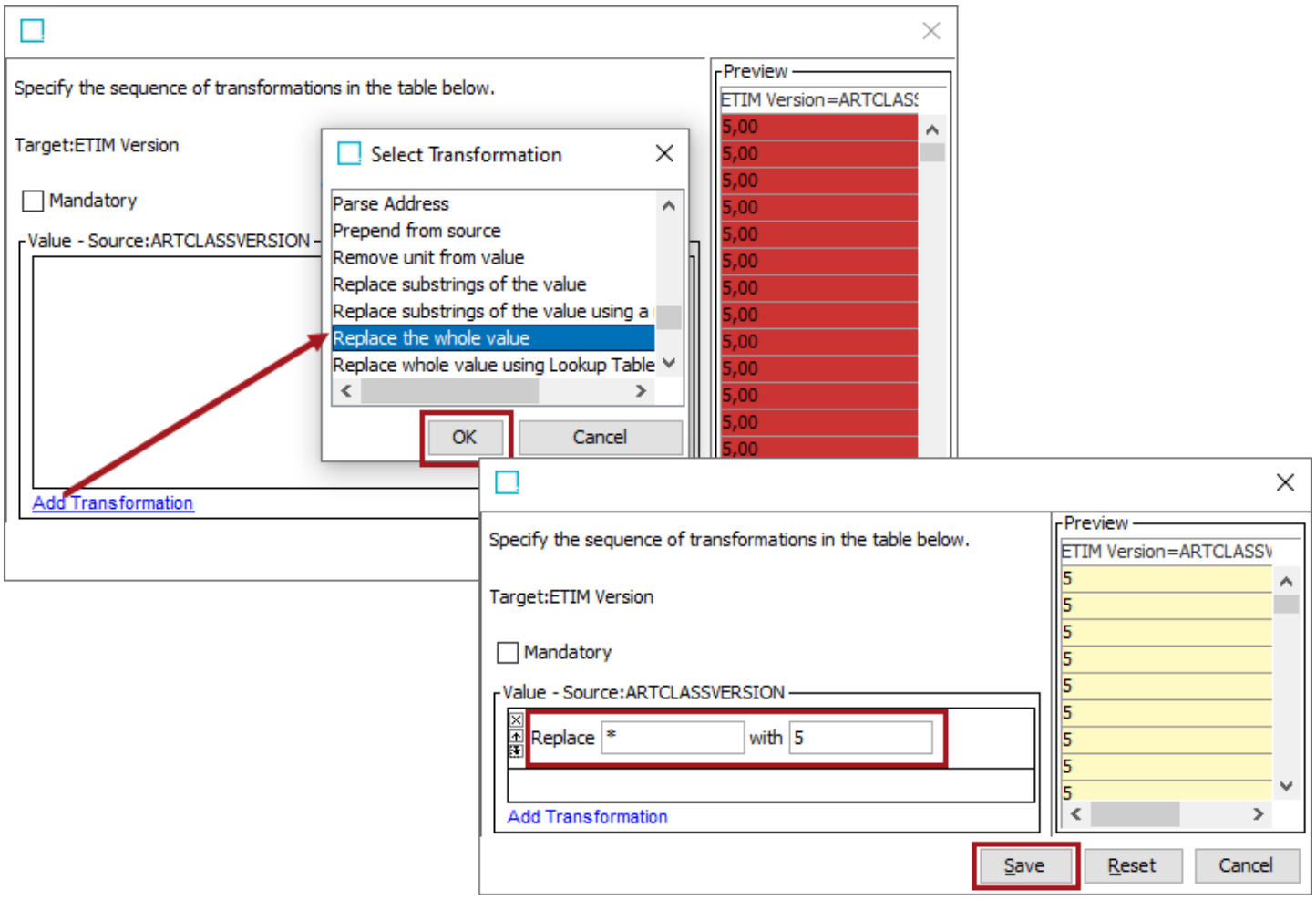
12. In the Map To window, select the **Attribute** radio button, search to find and select the **ETIM Version** attribute, and click **OK**.



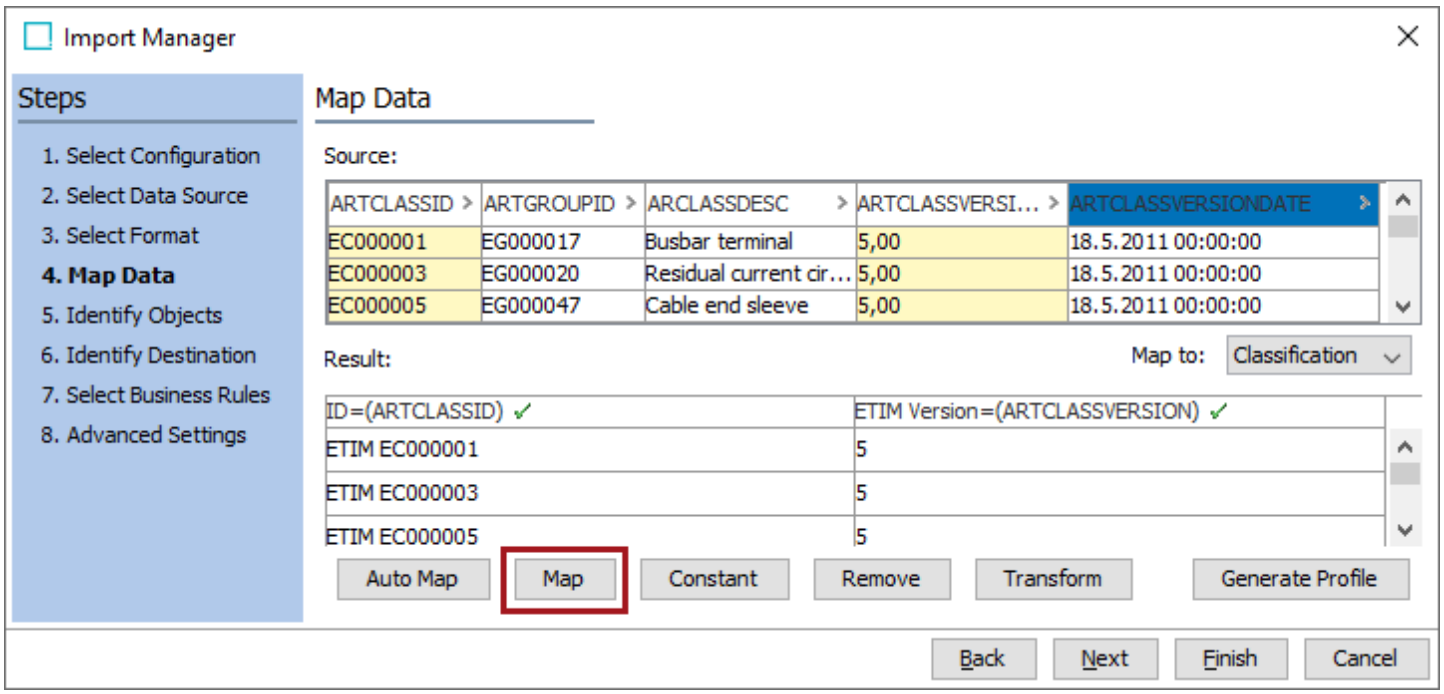
13. The ETIM Version attribute has a maximum length of three (3) characters, so it displays as an error in the Result table. Select the **ETIM Version=ARTCLASSVERSION** column and click the **Transform** button.



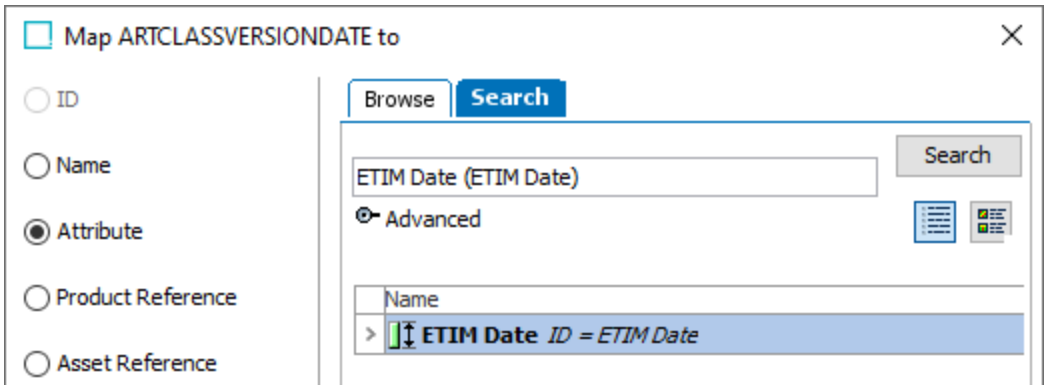
14. In the Transformation dialog, click the **Add Transformation** link, select the **Replace the whole value** option. For the values, add **replace * with** and enter a single digit based on the version being imported, and click the **Save** button.



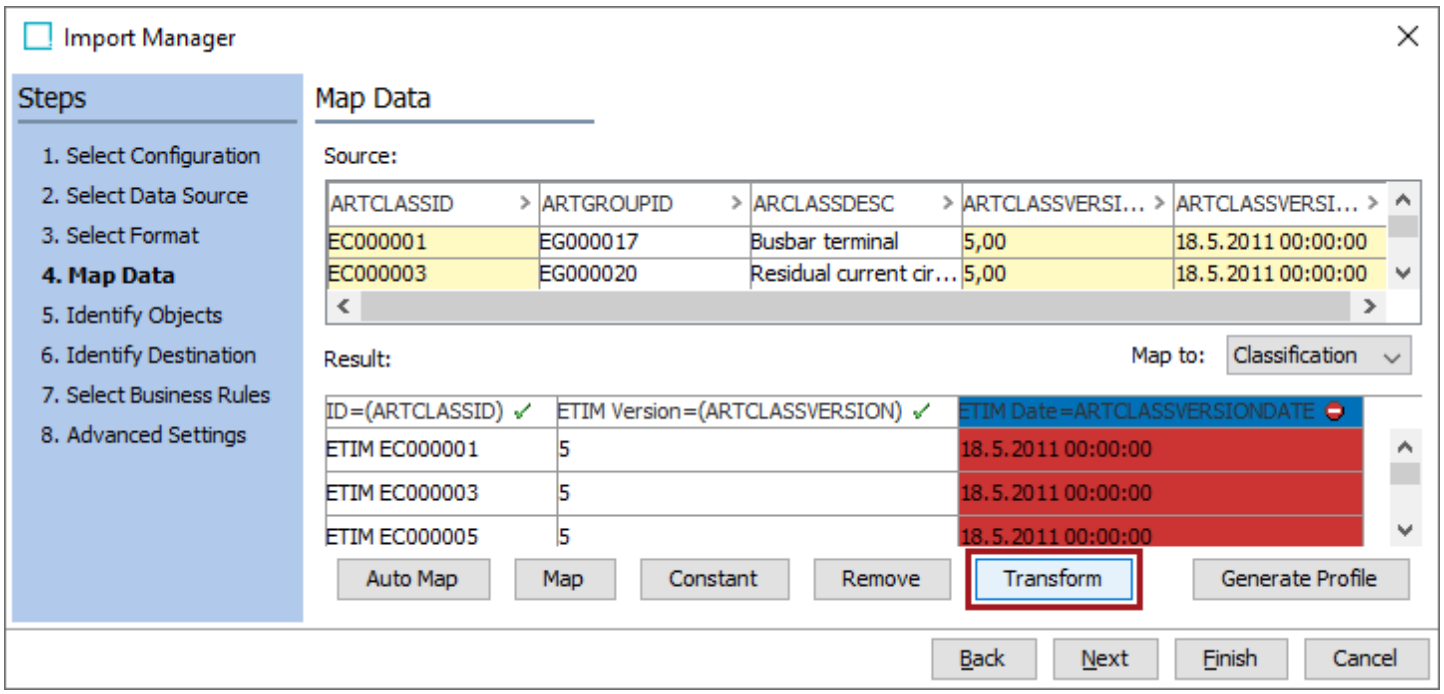
15. The Result table ID=ARTCLASSVERSION column now displays the selected single digit for all rows. In the Source table, select the **ARTCLASSVERSIONDATE** column and click the **Map** button below the Result table.



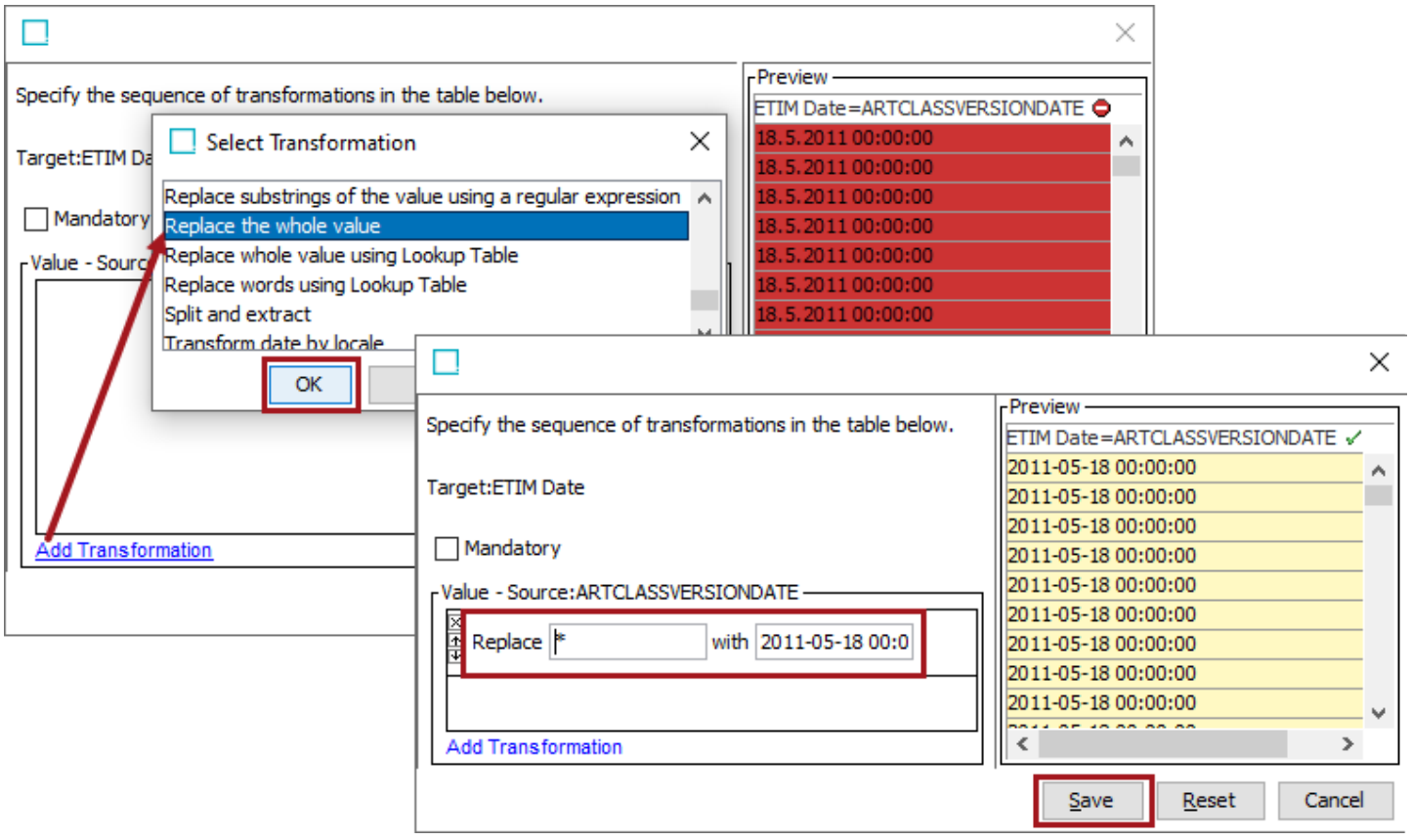
16. In the Map To window, select the **Attribute** radio button, search for and select the **ETIM Date** attribute, and click **OK**.



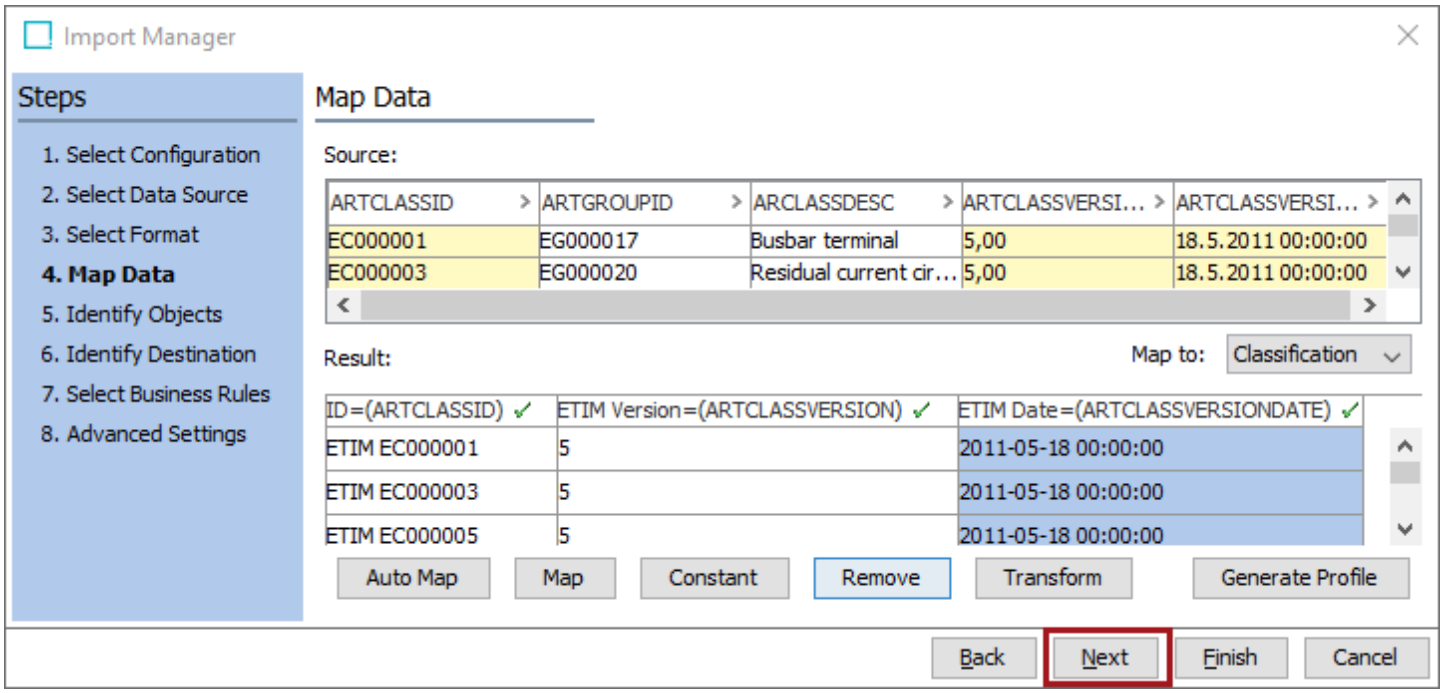
17. The ETIM Date attribute has a validation type of 'ISO Date and Time' format, it does not match the format in the ETIM file, so it displays as an error in the Result table. Select the **ETIM Date=ARTCLASSVERSIONDATE** column and click the **Transform** button.



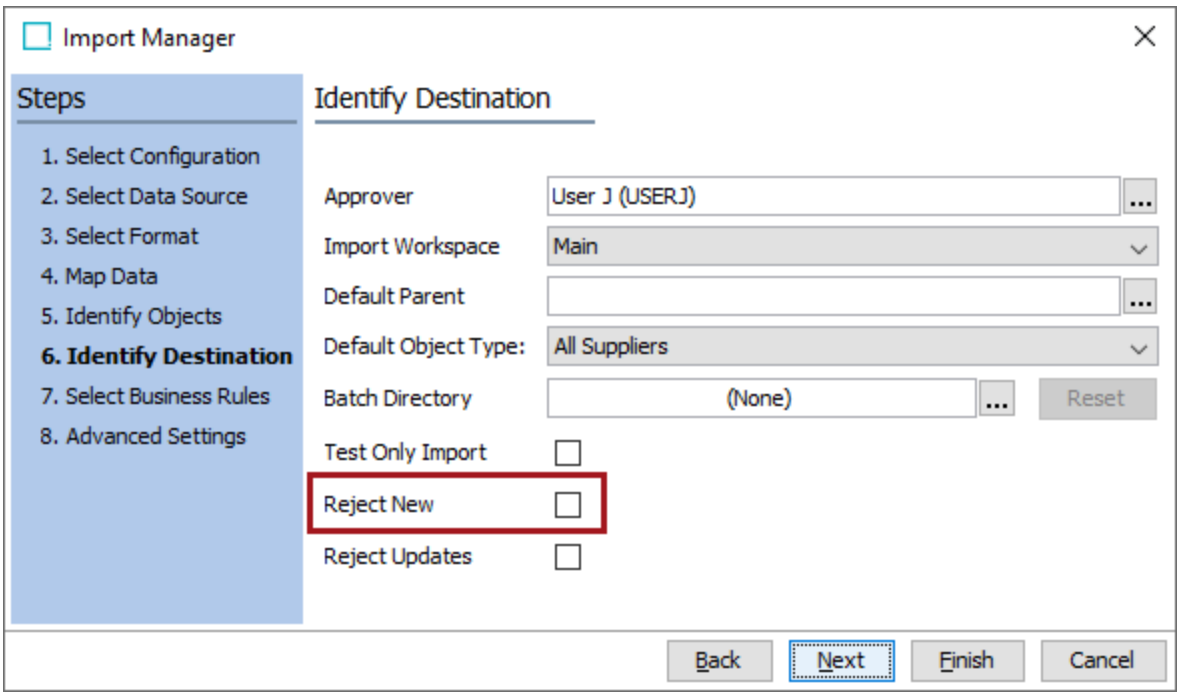
18. In the Transformation dialog, click the **Add Transformation** link, select the **Replace the whole value** option, **replace * with 2011-05-18 00:00:00**, and click the **Save** button. This date is used since the date is the same for all ETIM Classifications.



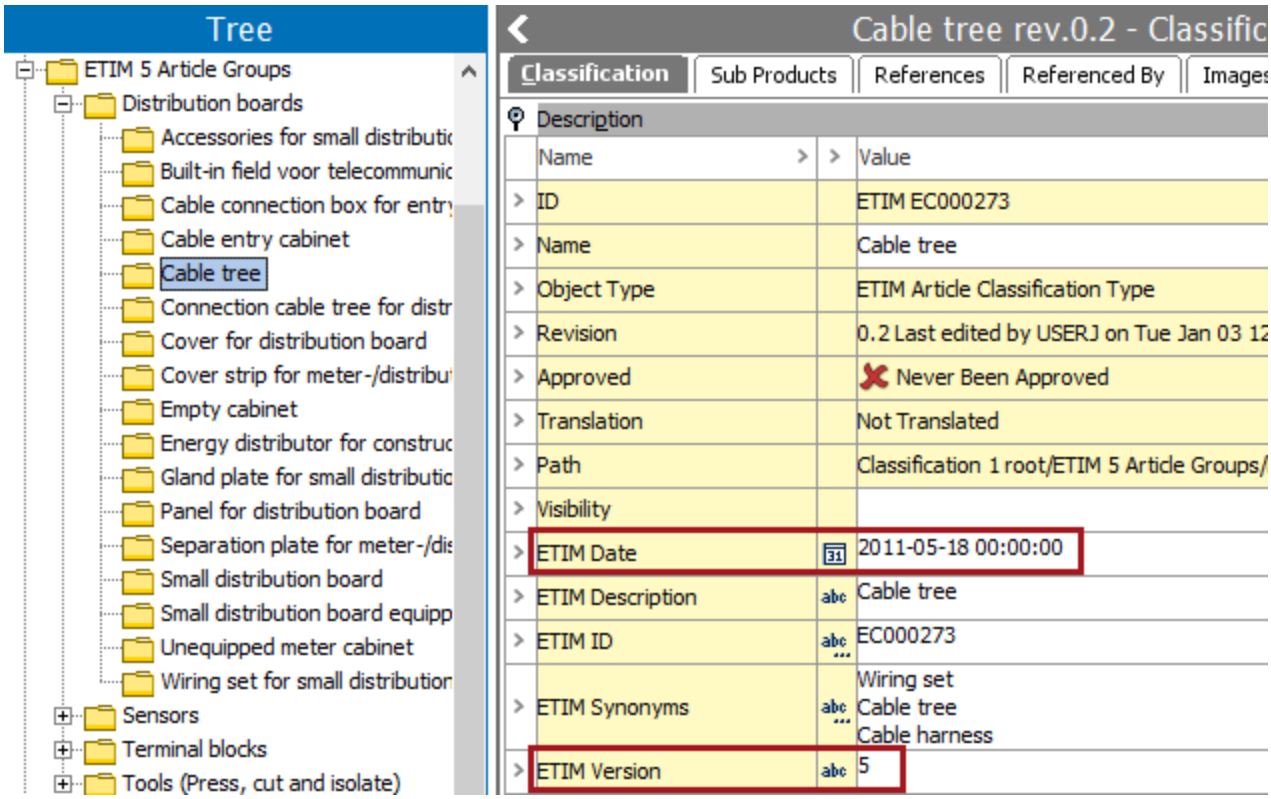
19. All three (3) mapped columns now show valid values in the Result table. Click **Next** (selecting defaults) until the Import Manager Identify Destination step displays.



20. Determine how to set the **Reject New** checkbox and click the **Finish** button:
    - **Check** the checkbox if you imported only some ETIM classifications. This prevents the import of classifications which do not already exist in STEP. The execution report shows all objects rejected as a result of this option
    - **Uncheck** the checkbox if you imported all ETIM classifications.



21. Follow the steps to start the import as described in **Running a Data Import**.
22. When the import completes successfully, the transformed values are displayed for all ETIM Article Classification object types for the attributes ETIM Date and ETIM Version.



Excel Format

The following sample Excel data import file shows that the first row is a header, and the remaining rows are import data. Exported Excel files always include a header row, though this is optional for imports.

	A	B	C	D	E
1	<Name>	<Parent ID>	Primary Color	ClassificationID	DescriptionL
2	Mens T PBO w Class	18209	Black	22586	Comfortable fitting 100% cotton tee.
3	Mens T PBG w Class	18209	Blue	22586	Comfortable fitting 100% cotton tee.
4	Mens T PGS w Class	18209	Green	22586	Comfortable fitting tee of cotton/poly blend.
5	Womens T PGW w Class	18209	Gray	22584	Comfortable fitting tee of cotton/poly blend.
6	Womens T POY w Class	18209	Orange	22584	Comfortable fitting 100% cotton tee.

Versions of Excel newer than 2007 can be used for imports. However, when exporting into Excel, the Excel version is set to Excel 2007, which allows the exported data to be read by later versions.

It is recommended that users use the newest supportable version of Excel. File errors may occur during different STEP processes if using a template set to an older Excel version. For more information on supported versions of Excel, refer to the current **Platform and Software Support** documentation in the **System Release and Patch Notes**.

Keep the following points in mind when working with STEP data using the Excel format:

- The following node types / super types can be imported and exported via Excel format: products, classifications, entities, assets (objects, not content), and attributes (objects and definitions).
- A Excel import or export file will include data in the same arrangement as relational database tables. This means that each object is displayed as a single row in the file and each object property item is displayed as a single column.
- References and/or data containers can be exported in this format where multiple values are separated by a delimiter. References are separated by semicolons, while data containers use a common prefix (#) for attribute columns. For a more easily readable file layout, refer to the options for Insert References / Data Containers on the **Outbound Map Data - Data Source** topic and the options for Inserted References / Data Containers on the **Inbound Map Data - Map** topic.
- Imports and exports are context and workspace specific. By default, data is imported to or extracted from the context and workspace in use when the process is started.
- When planning to import data back into STEP, include STEP ID in the export.
- There is a significant risk of importing orphan data when using empty value Excel or CSV exports across categories and/or object types and re-importing them when individual attributes are mapped versus using attribute groups (where invalid and non-linked attributes in the group(s) are excluded).

Important: Excel binary files, those with an XLSB extension, are not supported during import.

Format Availability

Excel is available for selection in:

- IIEP - refer to **Creating an Inbound Integration Endpoint**
- Import Manager - refer to **Creating a Data Import**
- Export Manager - refer to **Creating a Data Export**
- OIEP - refer to **Creating an Outbound Integration Endpoint**

Mapping

This format requires creating a data map between STEP and the data being processed, and may also include data transformations. For details, refer to **Data Mapping**.

Inbound Data

Excel import allows creation of and updates to products, classifications, entities, attribute values, data container values, and references. However, system setup objects (for example, LOVs, users, reference types, and so on), cannot be created via import.

Because the Map Data process allows selection of only a single node type, only one node type / super type (products, entities, etc.) can be imported at a time. When multiple super types exist in the same import file, a separate import is required to successfully import each type of object, starting with classification data, then product data, and finally, entity data. When the inbound file includes data for node types other than the one selected, two things may happen: 1) assuming none of the data prevents the import, new objects are created using the supplied information and the selected super type, 2) the execution report details the skipped records when included data, like parent ID, is not found in the selected super type hierarchy. Alternately, split the inbound data file by super type and process accordingly.

Note the following specifications about importing Excel data:

- Unicode is used as the character set for importing Excel data files.
- Formulas are not allowed in the XLS or XLSX file. Data columns that include calculations or formulas must be loaded as fixed values.
- Excel binary files, those with an XLSB extension, are not supported during import.
- Transformations in the import mapping can also be used to concatenate using the 'Append from' transformation.
- When loading data that includes special characters (such as trademark ™, registered trademark ®, or mathematical symbols), ensure that the load file conforms to the Unicode character set.
- Ensure that data originally entered as a fraction (e.g., 3/16) was not converted to a date by Excel.
- It is strongly recommended that all Excel cells are set to 'Text' format.

- It is recommended to remove any formatting applied to numeric values in Excel. This allows the Excel value to be entered into STEP, not the value that Excel transforms it into for display purposes. However, setting the inbound parameter 'Use date and number formatting from sheet' to 'yes' will cause the values to be imported as-is from the sheet. Refer to the 'Inbound Parameters' subsection below for more information.
- Enter attribute values and units **separated by a space** in the same cell, with the units following the values. STEP separates the values from the units and validates them individually. If your attribute values and units are in separate columns, you can either use the concatenation method in Excel or use a database program to combine them.

Important: Only the first sheet in the workbook is available to import, so be sure to move the sheet into the first position if importing a workbook with multiple sheets.

When using Excel to import a reference type that allows multiple references, several elements affect the data added to STEP, as specified in the **Importing Multi-Valued References or Links with Excel** topic.

To use Excel for importing calculated attribute value templates, refer to the **Importing Calculated Attribute Value Templates with Excel** topic.

Excel can be used to modify or create multiple attributes at the same time within System Setup. For details, refer to the **Managing Attribute Parameters with Excel** topic.

Deleting Values During Import

Removing a value requires the use of the entry **[delete]** (including the brackets) in your load file and that the object is mapped. Unlike other import formats, empty Excel cells are ignored and do not cause a value to be deleted. If the value being deleted was inherited, the result is not a blank field, but the inherited value is restored.

This functionality differs from imports using STEPXML or CSV files.

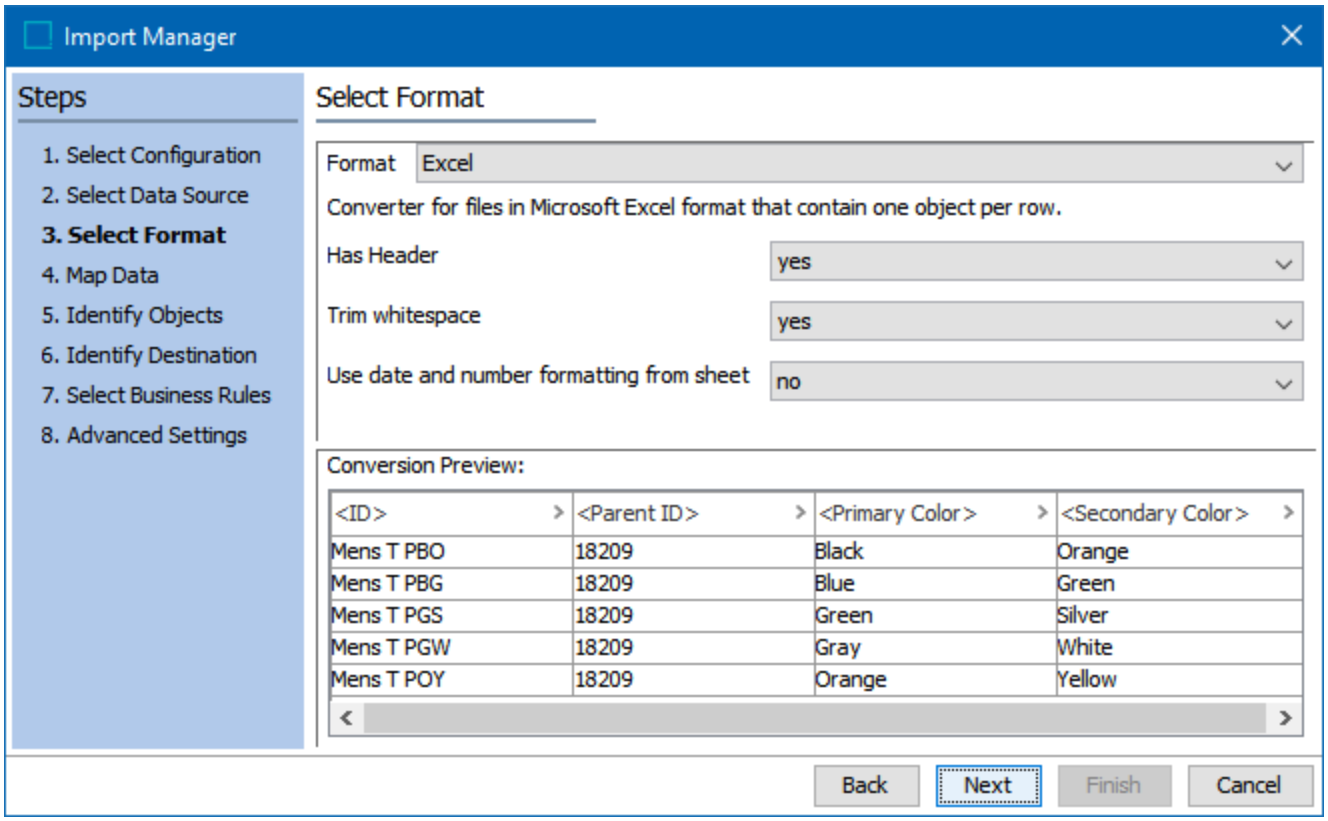
Inbound Parameters

The following parameters are available in both Import Manager and IIEP:

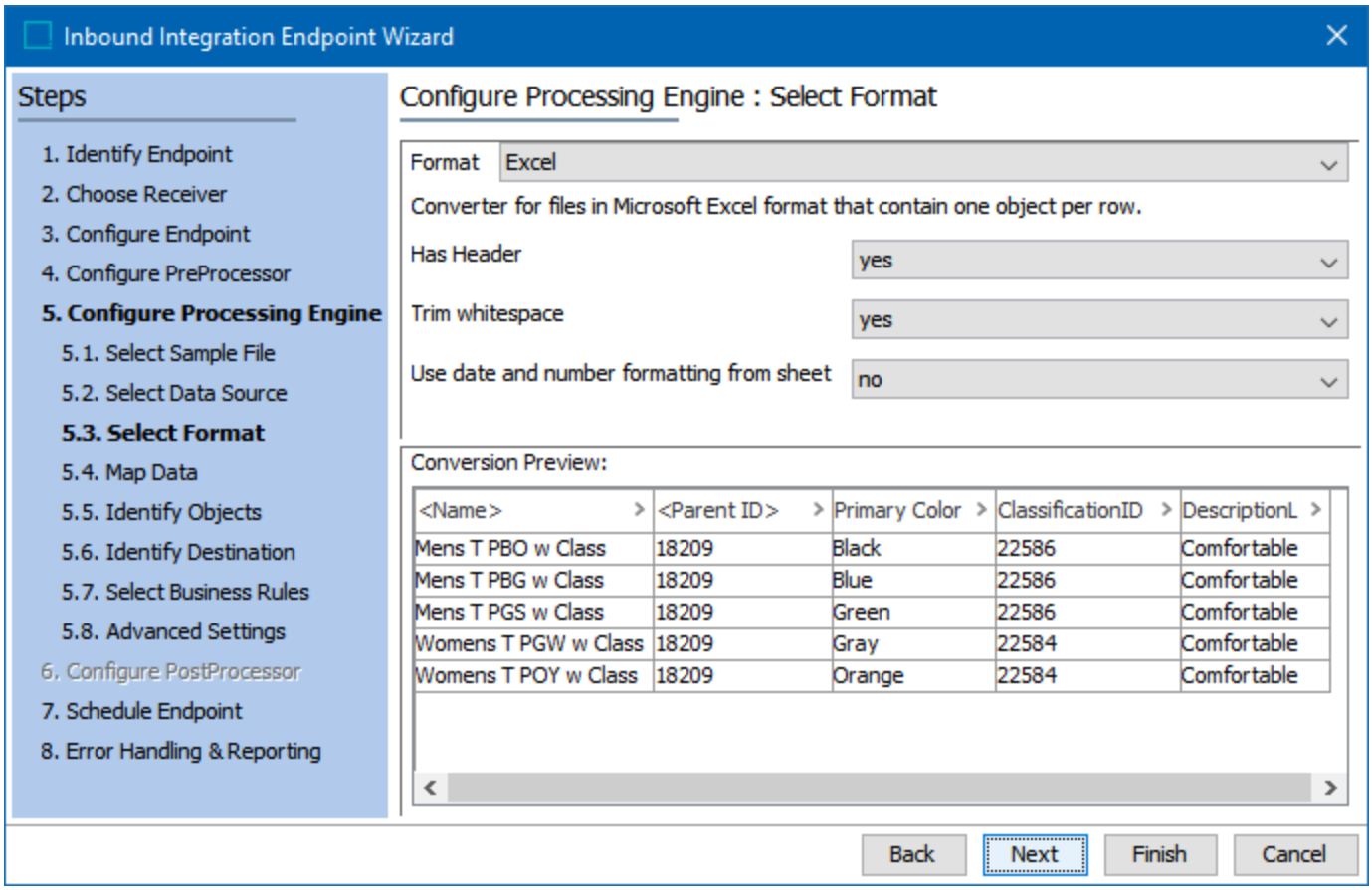
- **Conversion Preview** - a sample of the first few lines of the file is displayed to allow verification that the selected options are correct.
- **Has Header** - select if the file has a header line. If the first line (row) of the Excel file has header information, such as attribute names that match STEP, the Auto Map feature is available to map the columns of data to the appropriate object in STEP. Although a header row is not required, without it the user must be able to identify the data for manual mapping.
- **Trim whitespace** - select 'yes' to remove leading and trailing spaces in values, or select 'no' to leave them.
- **Use date and number formatting from sheet** - by default, this is set to 'no.' The 'no' setting allows for numeric values to be automatically imported in a standardized, US-English based format, and for date values to be automatically imported in ISO format, instead of the regional formats that may appear in source

spreadsheets. For details and examples using this parameter, refer to the **Date and Number Formatting with Excel** topic.

Import Manager



IIEP

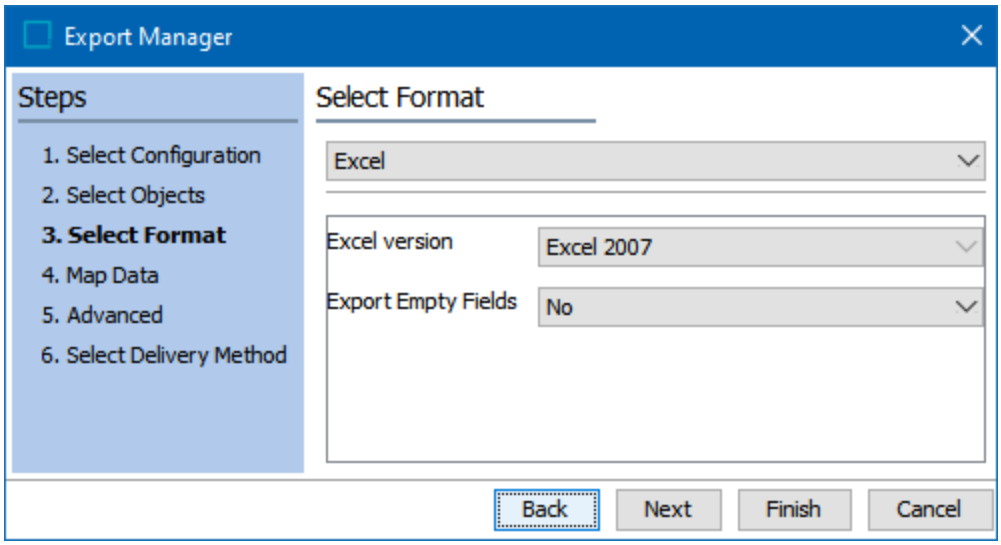


Outbound Data

When data leaves STEP via Excel format, the UTF-8 character set is used to export the data.

Note: When exporting object type, the default selection on the mapping step is 'Object Type Name.' However, when importing the same file, the auto-mapping feature expects the 'Object Type ID.' To generate an export file that is prepared for reimport, use the ID Aspect to map object type. For more information, refer to the **Aspect - Transform Outbound** section of the **Data Mapping** documentation.

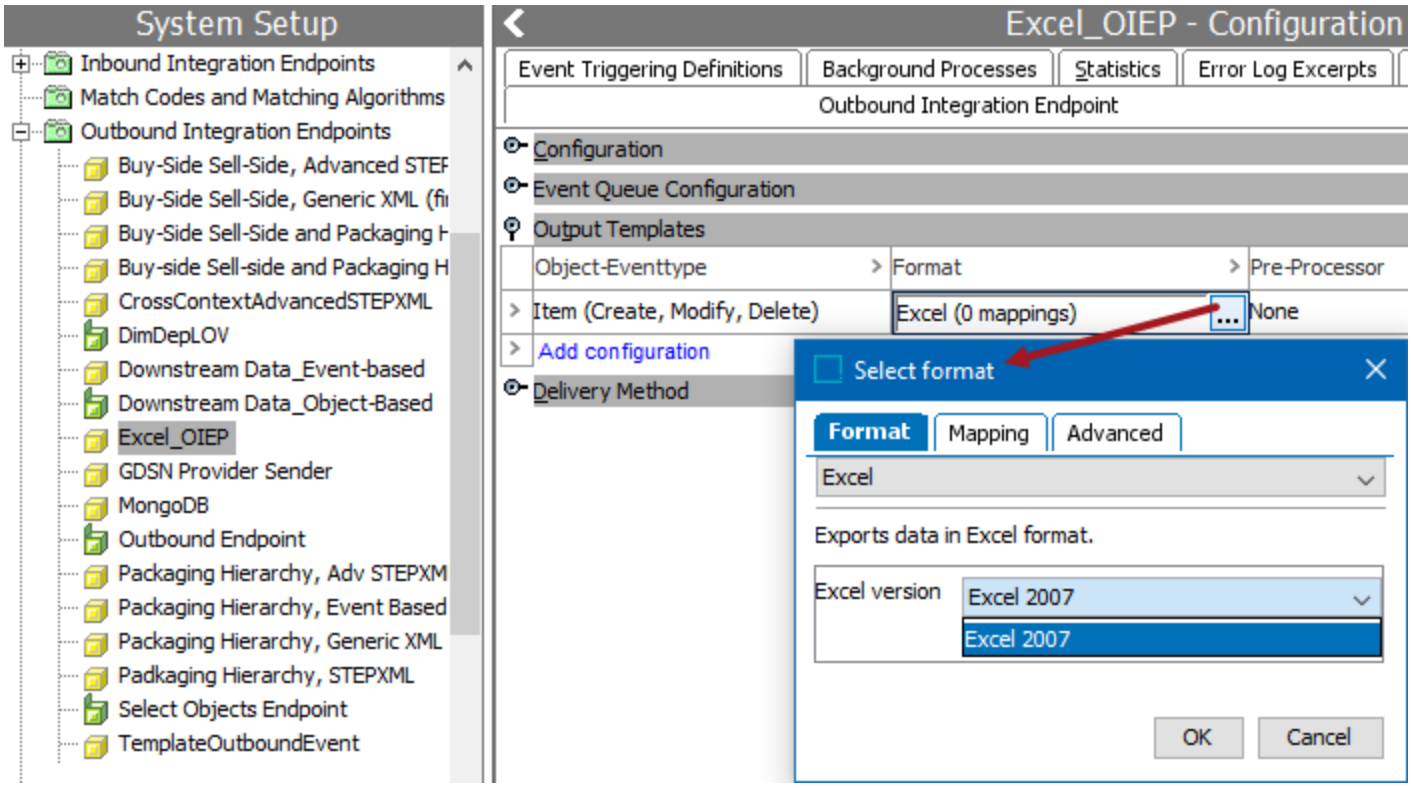
Export Manager



The following parameters are available when choosing the format 'Excel' in the Export Manager:

- **Excel Version** - Excel 2007, which has a limit 1,048,576 rows.
- **Export Empty Fields** - determines how fields without values are exported.
  - **No** - exports only fields that include a value.
  - **Yes** - values (attributes, references, metadata, data containers, etc.,) that are valid and linked will be included in the export even when no objects are populated. Attributes that are not valid / linked are extended when an attribute group is used in the mapping.

OIEP

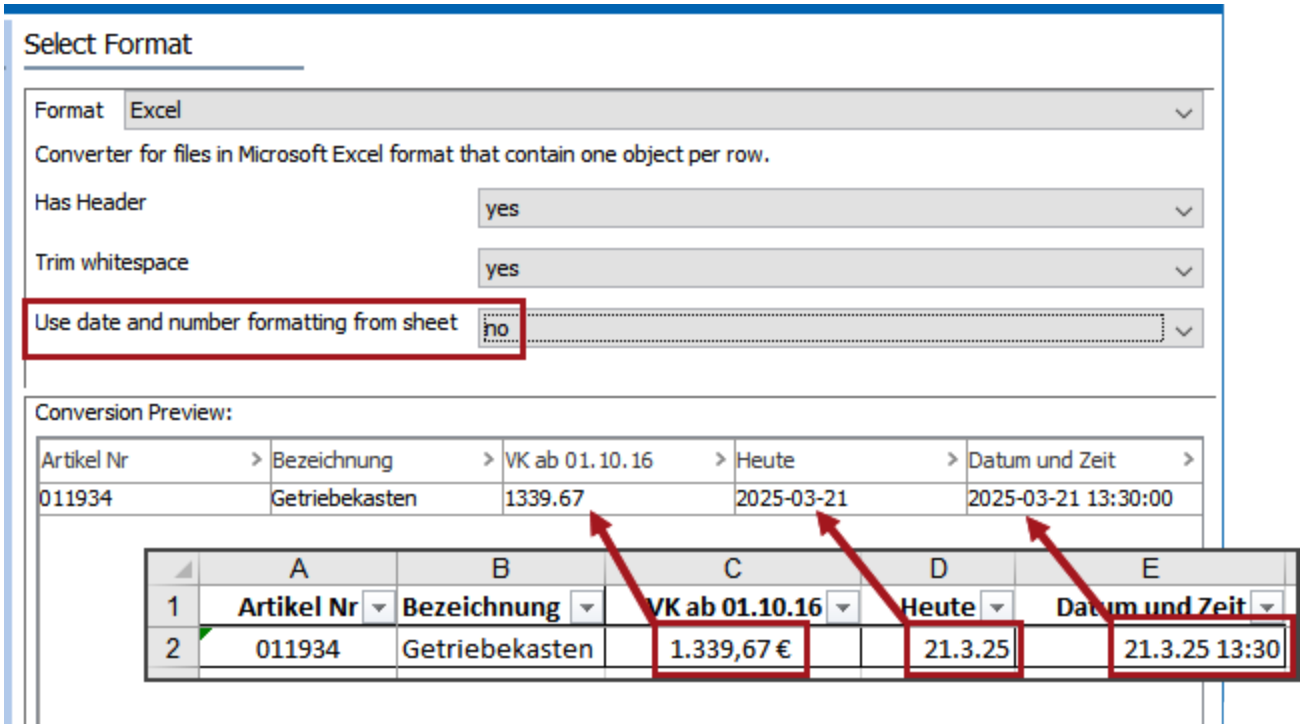

System Setup
Inbound Integration Endpoints
Match Codes and Matching Algorithms
Outbound Integration Endpoints
Buy-Side Sell-Side, Advanced STEP
Buy-Side Sell-Side, Generic XML (fi
Buy-Side Sell-Side and Packaging H
Buy-side Sell-side and Packaging H
CrossContextAdvancedSTEPXML
DimDepLOV
Downstream Data_Event-based
Downstream Data_Object-Based
Excel_OIEP
GDSN Provider Sender
MongoDB
Outbound Endpoint
Packaging Hierarchy, Adv STEPXM
Packaging Hierarchy, Event Based
Packaging Hierarchy, Generic XML
Padkaging Hierarchy, STEPXML
Select Objects Endpoint
TemplateOutboundEvent
Excel_OIEP - Configuration
Event Triggering Definitions
Background Processes
Statistics
Error Log Excerpts
Outbound Integration Endpoint
Configuration
Event Queue Configuration
Output Templates
Object-Eventtype
Format
Pre-Processor
Item (Create, Modify, Delete)
Excel (0 mappings)
None
Add configuration
Delivery Method
Select format
Format
Mapping
Advanced
Excel
Exports data in Excel format.
Excel version
Excel 2007
Excel 2007
OK
Cancel

Date and Number Formatting with Excel

By default, the **Use date and number formatting from sheet** parameter is set to 'no.' The 'no' setting allows for numeric values to be automatically imported in a standardized, US-English based format, and for date values to be automatically imported in ISO format, instead of the regional formats that may appear in source spreadsheets.

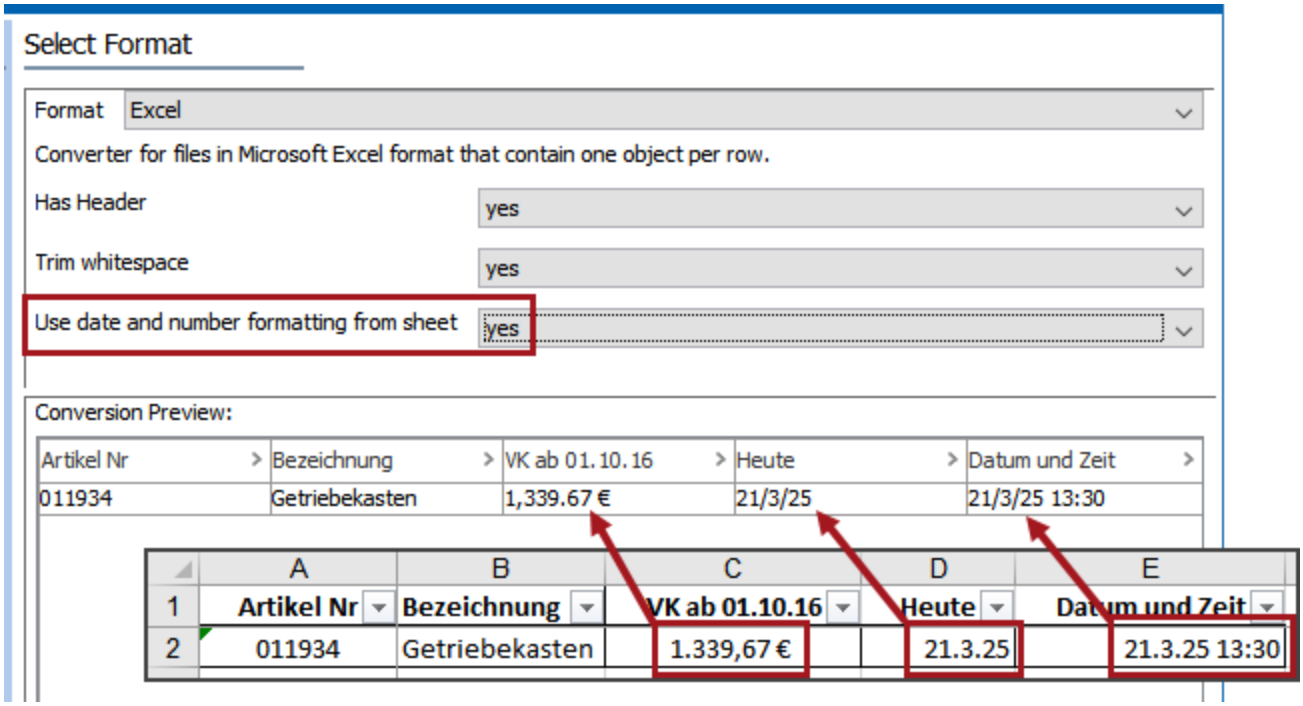
As an example of the 'no' setting, the below screenshot shows the Conversion Preview for a spreadsheet created on a computer with its region set to Germany, displaying currency in a Euro format and dates in a German format. When **no** is selected for 'Use date and number formatting from sheet,' the currency and dates are automatically transformed into the following standard formats: date values as yyyy-MM-dd, date-time values as yyyy-MM-dd HH:mm:ss, and numeric values as #.##########.

This setting is helpful for users in regions other than the United States, who can avoid the need to apply numerous and sometimes non-intuitive transformations on their source data, which allows for more predictable and consistent data imports.



Note: When importing Excel data using the **no** setting for 'Use date and number formatting from sheet,' if date values will be mapped to attributes with the 'ISO Date' or 'ISO Date and Time' base validation, then these attributes must have Strict Validation set to 'Yes.' This ensures that imported date information will remain standardized throughout the system and not cause search issues later due to non-standardized date formats. For more information, refer to the **Validation Rules** section of the **Attributes** documentation.

As an example of the 'yes' setting, the below screenshot also shows the same example German spreadsheet with **yes** is selected for 'Use date and number formatting from sheet.' When 'yes' is selected, the currency and date values display mostly 'as-is' from the source spreadsheet.



For details on the other parameters available when importing Excel, refer to the **Excel Format** topic.

Importing Calculated Attribute Value Templates with Excel

Adding a calculated attribute value template to an Excel file for successful import into STEP requires the following specific setup:

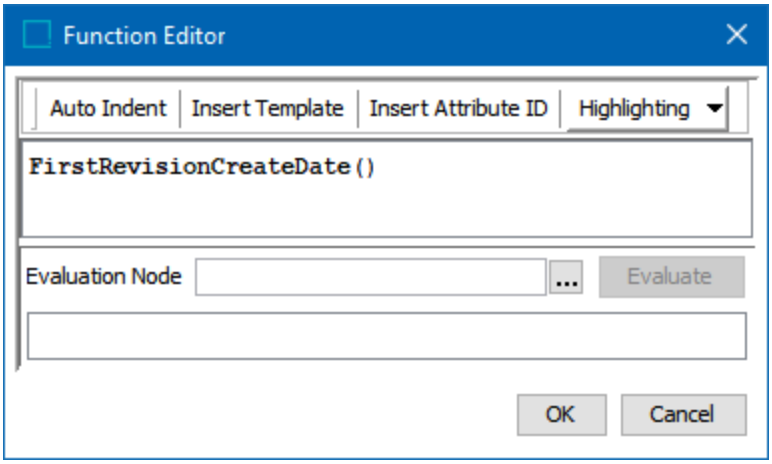
- The entire expression must be in a single Excel cell.
- The expression must begin with **<derive expr="**
- The expression must end with **"/>**
- Within the expression, double quote (“) characters must be represented by **"**
- Within the expression, less than (<) characters must be represented by **<**
- Within the expression, greater than (>) characters must be represented by **>**
- Within the expression, single quote (') characters are allowed as-is, no alternate representation is required.

Simple Example

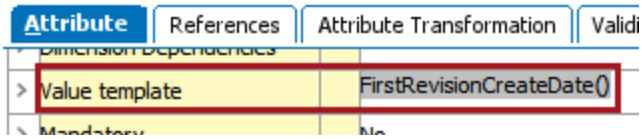
To add a simple value template to an Excel file for import:

1. If necessary, create a calculated attribute; otherwise, continue to the next step. For information on creating a calculated attribute, refer to the **Calculated Attributes** topic in System Setup documentation.

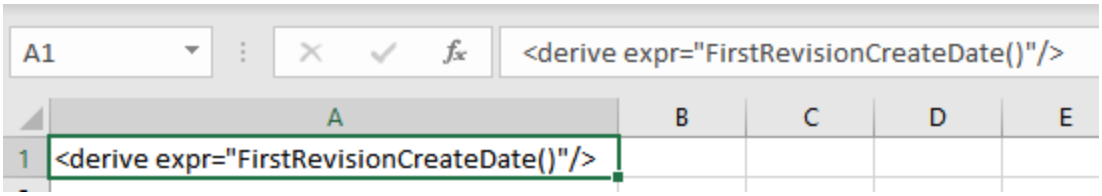
   This example uses the 'FirstRevisionCreateDate' which is available on the Insert Template tab under the 'Other' function section.



2. View the calculated function displayed in the attribute editor.



3. While in the attribute editor, on the 'Value template' parameter, select and copy the Value column text. In this example, select **FirstRevisionCreateDate()**.
4. In the Excel file being prepared for import into STEP, paste the copied text into a single cell. For details on importing via Excel, refer to the **Excel Format** topic.



Important: If the pasted text includes an extra set of double quotes surrounding the entire value template, they must be removed for successful import. For example, with this pasted text, the first and last characters must be removed: **"**<derive expr="FirstRevisionCreateDate()"/>**"**

5. Save the Excel file and import.

Complex Examples

The steps to add a complex value template are the same as the simple example, and the same requirements apply for successful import.

The calculated function in the Function Editor displays as follows:

Function Editor

Auto Indent | Insert Template | Insert Attribute ID | Highlighting

```
{
x := iterate(iterate(references('product', 'Alternative'), 'referencetarge
'iterate(iterate(references("product", "Spare Part"), "referencetarget()")
"stepname()")')
}
list(x, " > ")
```

Evaluation Node: XTG-75632 (XTG-75632)

Competitor Products > Cutting Tools > Spark Plugs Products > Taps and Dies > Hand Tools

To create the above template correctly via an Excel import, verify that the appropriate cell in the XLS file is as follows:

	A	B
1	<ID>	Template

	GetFollowOnRefO	<derive expr="{
		x := iterate(iterate(references('Product', 'Alternative'), 'referencetarget()'),
		'iterate(iterate(references("product", "Spare Part"),
		"referencetarget()"), "stepname()")')
		}

2		list(x, ' > ')"/>

Another complex example is this calculated function in the Function Editor:

Function Editor

Auto Indent | Insert Template | Insert Attribute ID | Highlighting

```
{
targid := iterate(iterate(references("product",
"Alternative"),'referencetarget()'), 'stepid()'),
targname := iterate(iterate(references("product",
"Alternative"),'referencetarget()'), 'stepname()'),
equiv := iterate(references("product",
"Alternative"),'value("EquivalenceRating")'),
getexc := iterate(references("product", "Alternative"),'value("GetException")'),
record := iterate(targid, 'concatenate(item,", [", listitem(targname,index), "]
Rating = ", listitem(equiv, index), ": Exception Value = ",
listitem(getexc,index))')
}
substitute(list2multivalue(record), "<multisep/>", "\n")
```

To create the above template correctly via an Excel import, verify that the appropriate cell in the XLS file is as follows:

	A	B
1	<ID>	Template

	GetMetaData	<derive expr="{
		targid := iterate(iterate(references("product", "Alternative"),'referencetarget()'), 'stepid()'),
		targname := iterate(iterate(references("product", "Alternative"),'referencetarget()'), 'stepname()'),
		equiv := iterate(references("product", "Alternative"),'value("EquivalenceRating")'),
		getexc := iterate(references("product", "Alternative"),'value("GetException")'),
		record := iterate(targid, 'concatenate(item,", [", listitem(targname,index), "] Rating = ", listitem(equiv, index),
		": Exception Value = ", listitem(getexc,index))')
		}

2		substitute(list2multivalue(record), "&ltmultisep/>", "\n")"/>

Importing Multi-Valued References or Links with Excel

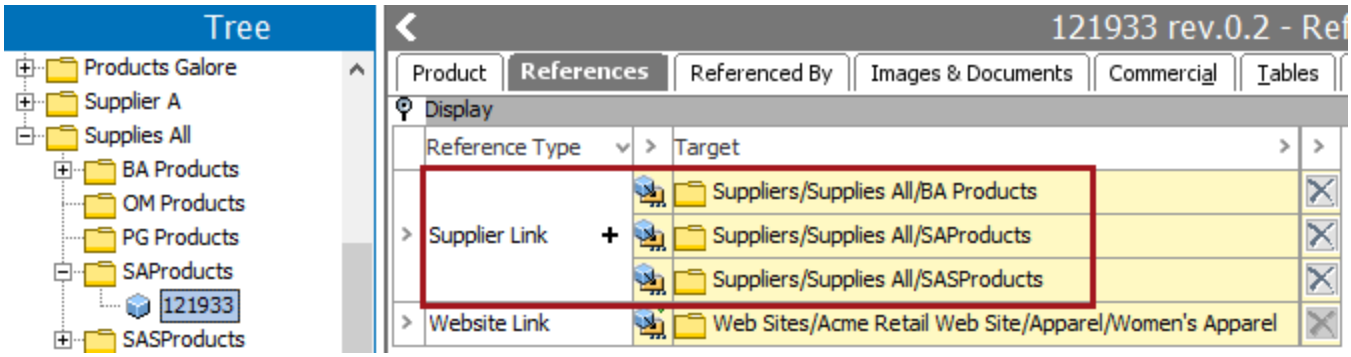
When using Excel to import a reference type that allows multiple references, it is important to keep in mind the following points to ensure the intended references are applied in STEP.

- As an Excel file is processed by the importer, each row is read with no awareness of the previous or following rows. This means that if the same object is listed in the import file multiple times, updates made by the first occurrence of the object may be overwritten by subsequent occurrences.
- The Import Manager 'Advanced Settings' Remove Options functionality allows you to control when data in STEP is replaced by the inbound data, or instead, is appended by the inbound data.
- The combination of the inbound file structure, the import data mapping, and the Remove Unmapped options determine the data that will be available in STEP after a successful import.

Possible options for the Excel file layout are: 1) listing the object to be imported multiple times within the file, 2) listing each object once, with multiple references in the same cell, or 3) listing each object once, with multiple references in individual cells on the same row. Each setup is described below.

Object listed multiple times, each with a single link

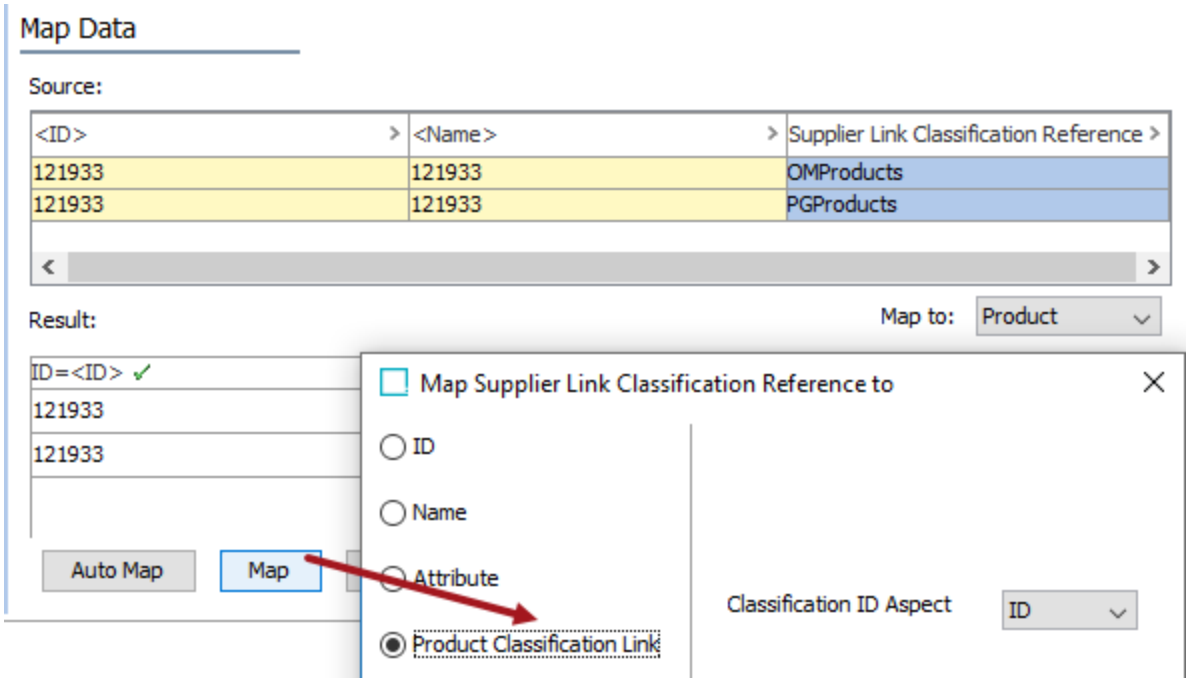
Consider the scenario shown below where in STEP the product with ID 121933 is already linked to three classifications (ID BAProducts, ID SAProducts, and ID SASProducts) using the product to classification link type Supplier Link. This link type has the 'Allow multiple links' option enabled.



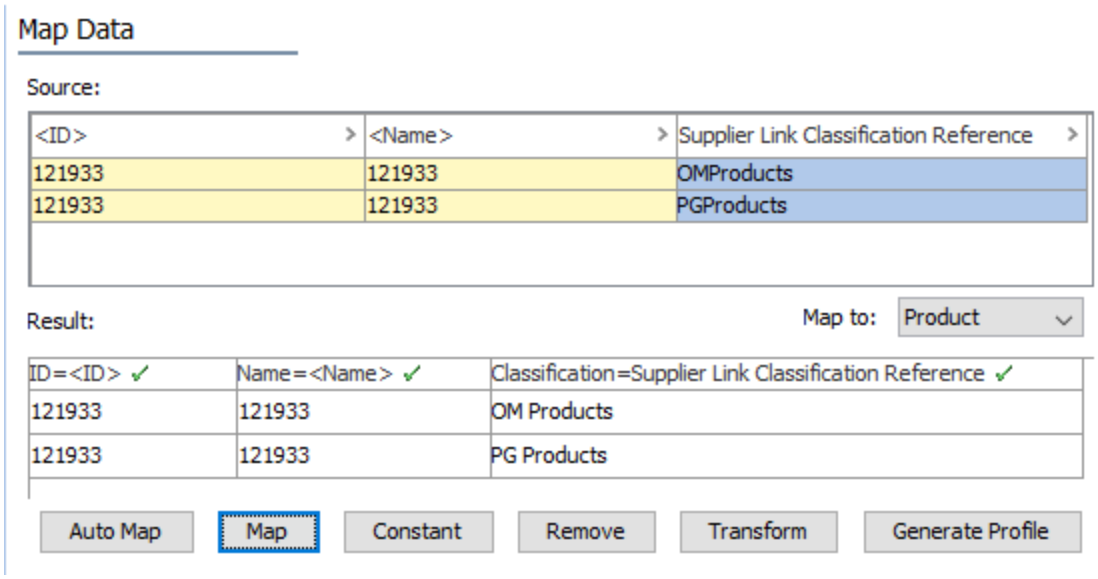
An Excel import file has been prepared and includes two different valid classifications (ID OMProducts and ID PGProducts) for the product to classification link type with the ID Supplier Link.

	A	B	C
1	<ID>	<Name>	Supplier Link Classification Reference
2	121933	121933	OMProducts
3	121933	121933	PGProducts

Using the Import Manager's Map Data functionality, automatically map all possible columns, and then manually map the Product Classification Link type by ID.

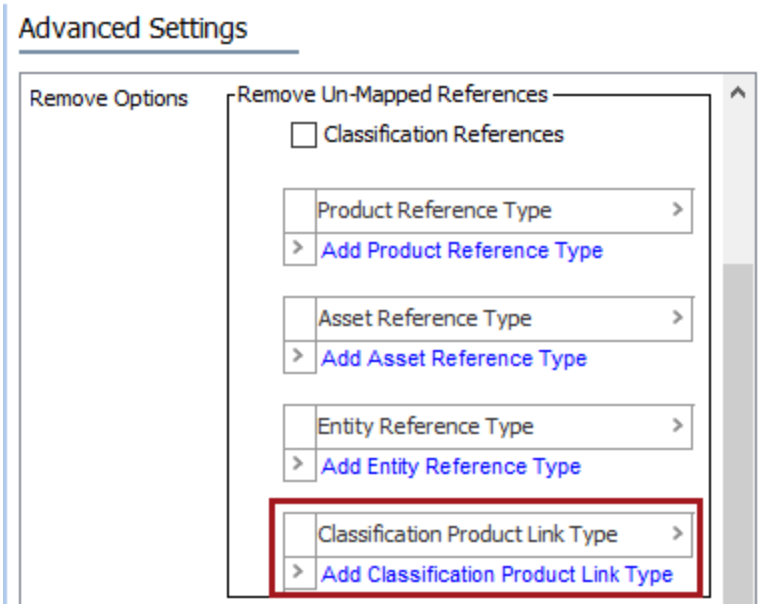


The multi-value classification references are displayed and ready to import.

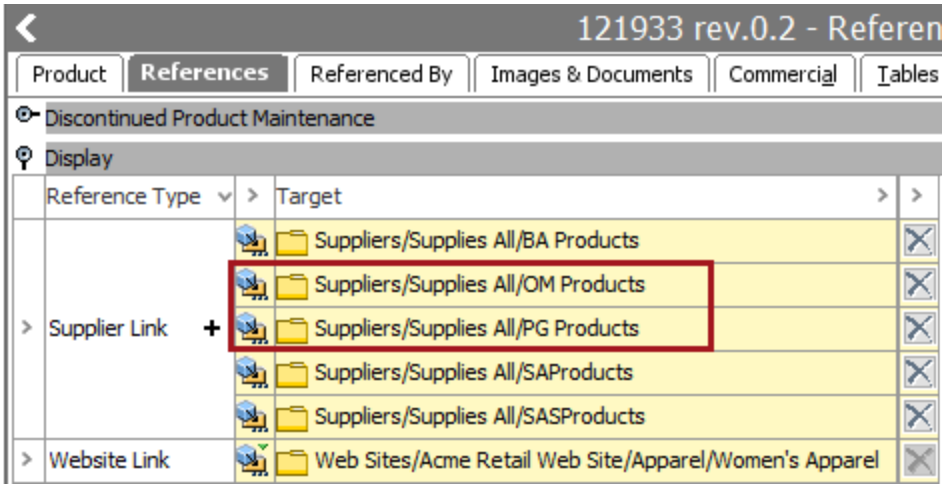


Complete the import wizard steps, noting that the selection on the Advanced tab determines how existing data in STEP is affected.

Remove Un-Mapped Classification References Not Set (Default)

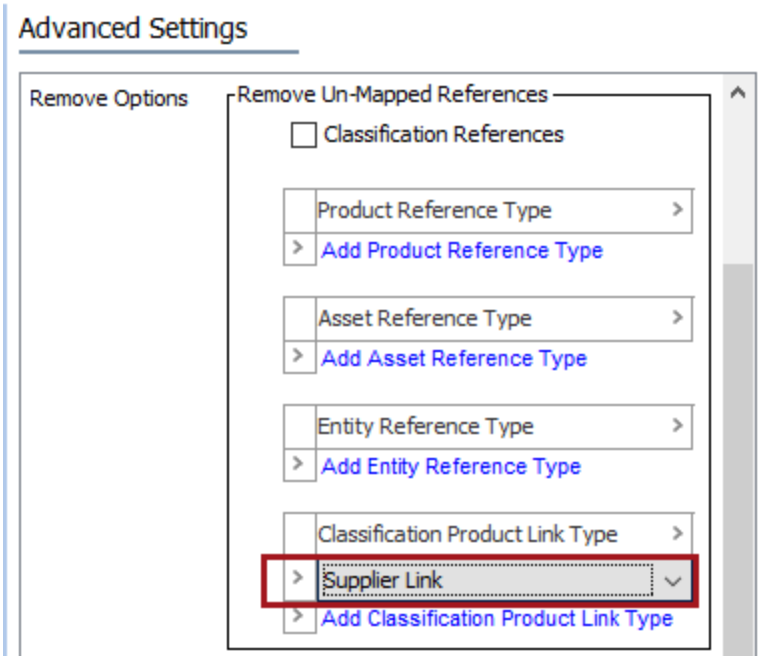


The imported classifications are added to the existing ones for the mapped product to classification link type.

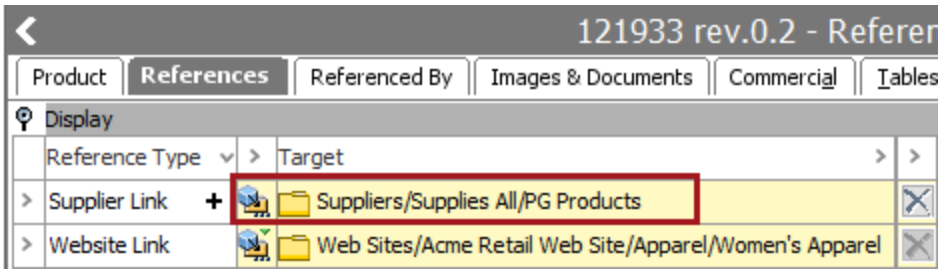


Setting the Remove Un-Mapped Classification References option indicates that after the import, STEP will include only the product to classification link types (based on ID) contained in the import file.

Remove Un-Mapped Classification References Set



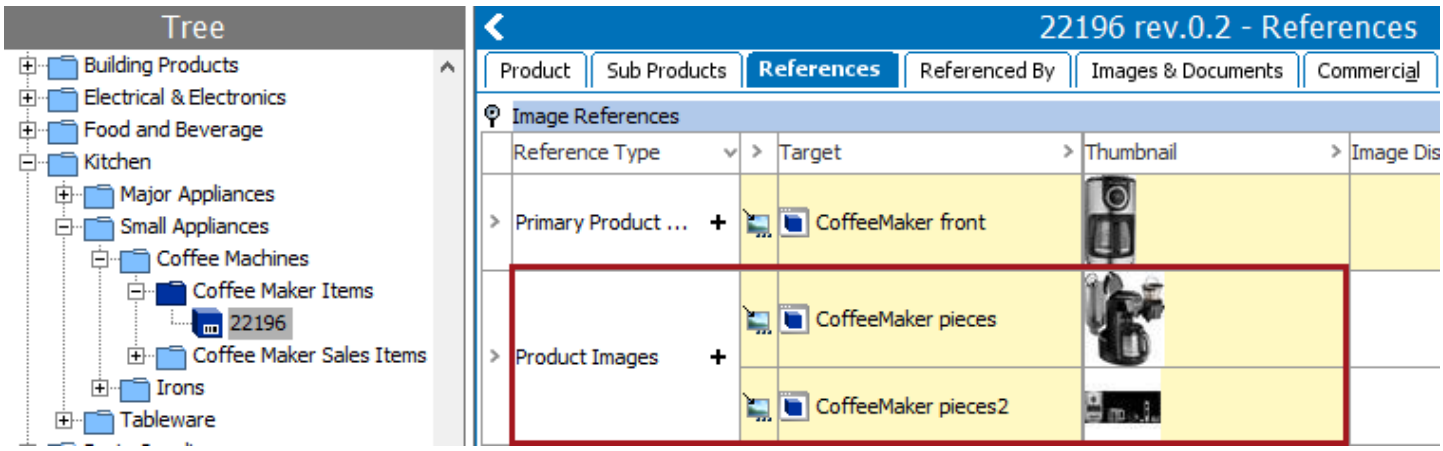
The imported classifications replace the existing ones for the mapped product to classification link type. Note that since each row in the Excel file is processed individually, only the final link for the object remains after the import is completed.



This result can be avoided by listing each object only once in the file, and either including all links of the same type within the same cell (as defined below) or by including each link in its own cell on the same row.

Object listed once, multiple references in the same cell

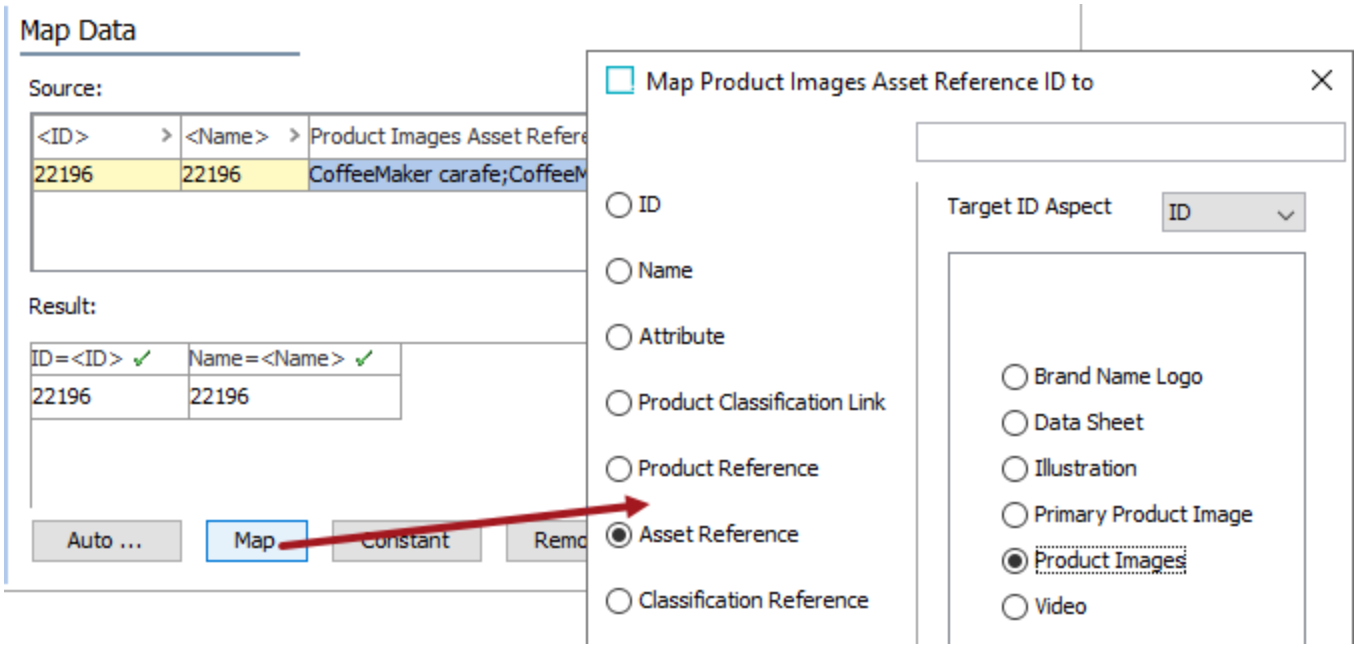
Consider the scenario shown below where in STEP a product already references two images (IDs CoffeeMaker pieces and CoffeeMaker pieces2) using the Product Images reference type. This reference type has the 'Allow multiple references' option enabled.



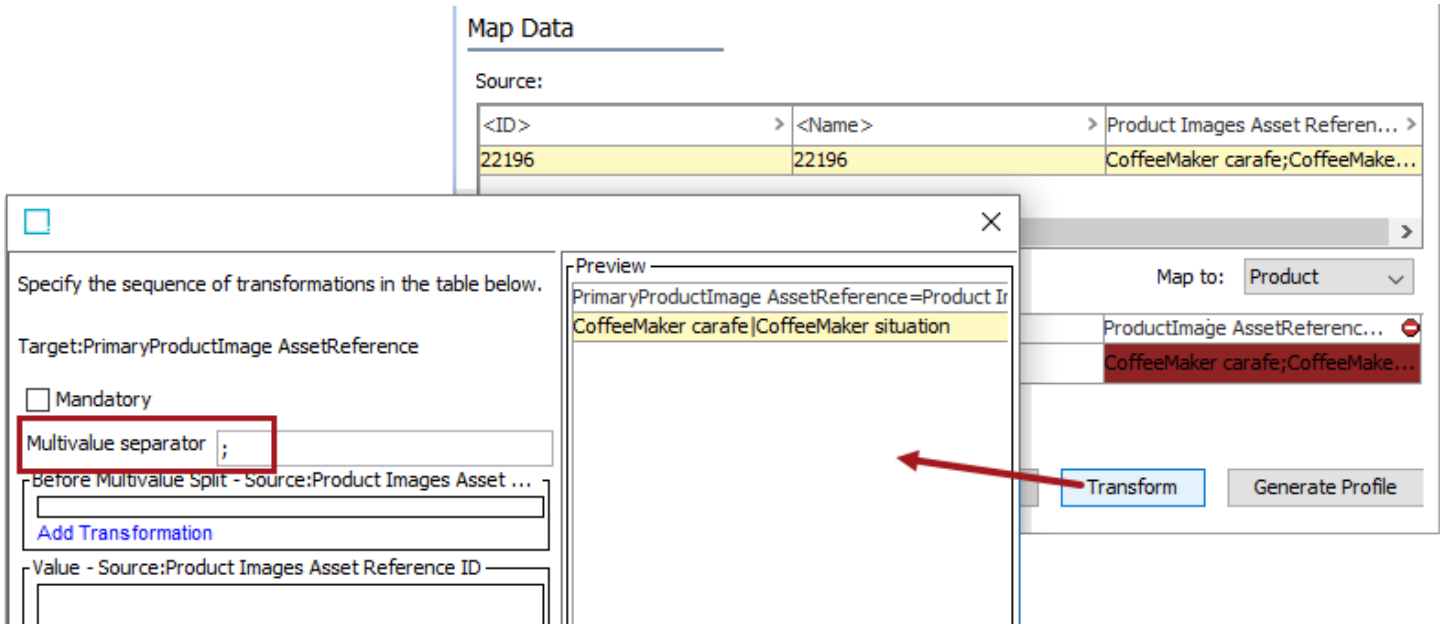
An Excel import file has been prepared and includes two different images (IDs CoffeeMaker carafe and CoffeeMaker situation) for the Product Images reference type.

	A	B	C
1	<ID>	<Name>	Product Images Asset Reference ID
2	22196	22196	CoffeeMaker carafe;CoffeeMaker situation

Using the Import Manager's Map Data functionality, automatically map all possible columns, and then manually map the new images to the Product Images reference type.

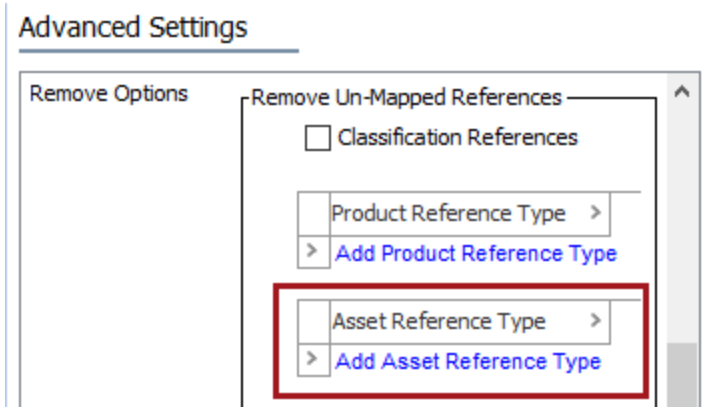


Finally, transform the multi-value references to an input format that STEP can read [replace each semicolon (;) with a pipe (|)], and verify that the Preview pane no longer shows a red background (which indicates an error).



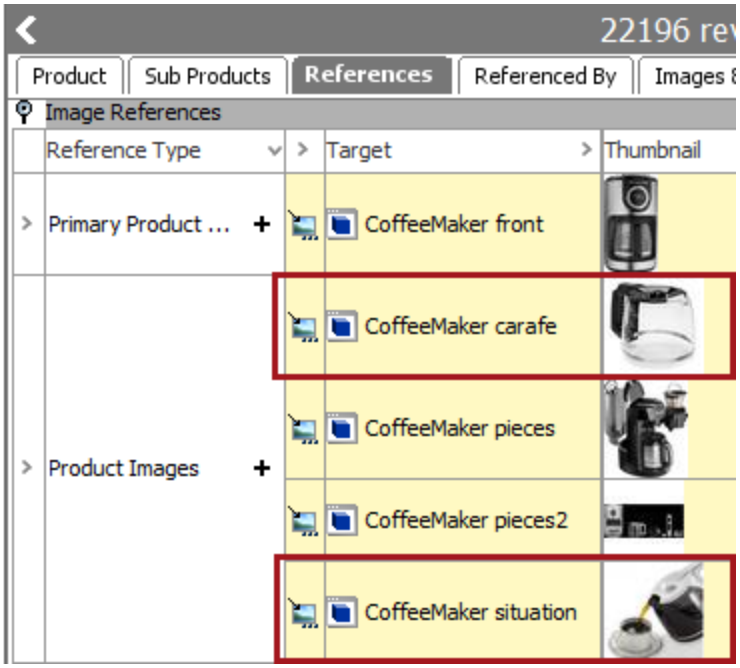
Complete the import wizard steps, noting that the selection on the Advanced tab determines how existing data in STEP is affected.

Remove Un-Mapped Asset References Not Set (Default)



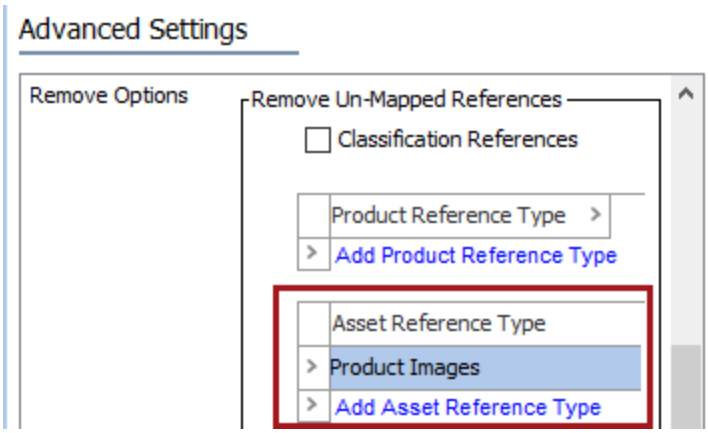
The imported images are added to the existing images for the selected reference type.

Remove Un-Mapped Asset References Not Set (Default)



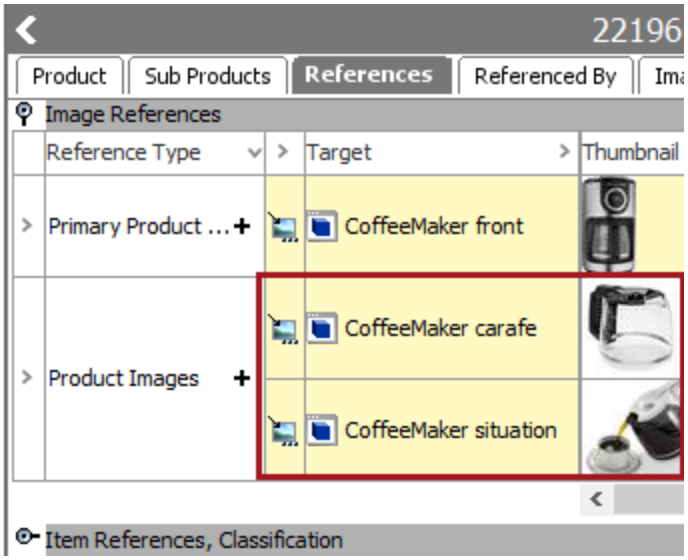
Setting the Remove Un-Mapped References options indicates that after the import, STEP will include only the references contained in the import file.

Remove Un-Mapped Asset References Set to Asset Reference Type



The imported images replace the existing images for the selected reference type.

Remove Un-Mapped Asset References Set to Asset Reference Type

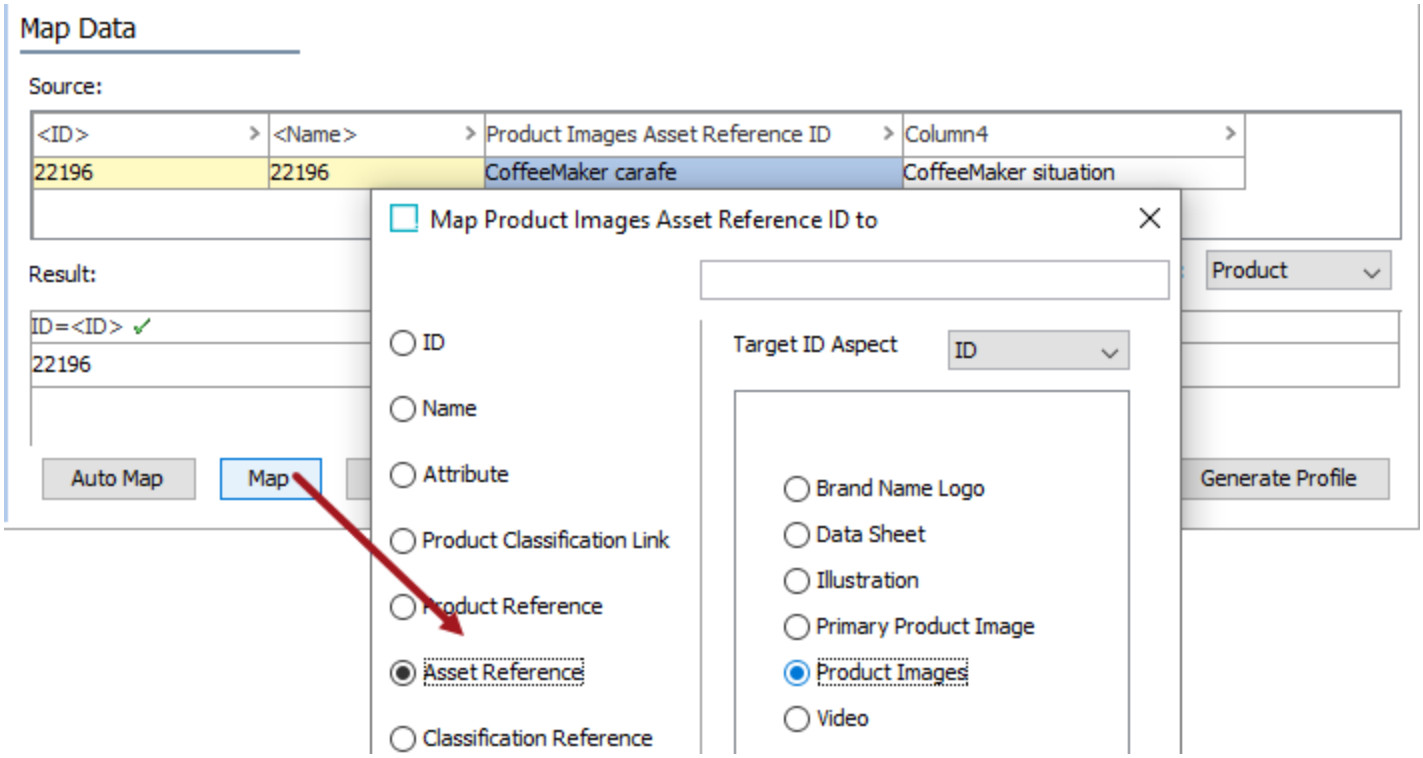


Object listed once, multiple references in individual cells

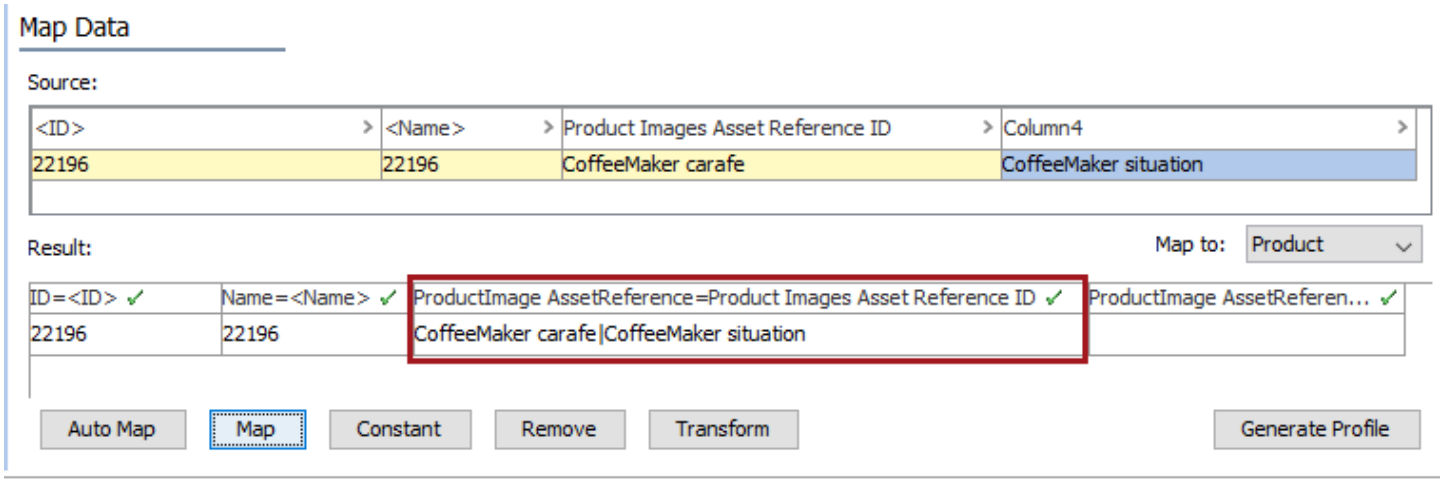
Consider the same scenario shown above where in STEP a product already references two images (IDs CoffeeMaker pieces and CoffeeMaker pieces2) using the Product Images reference type. This reference type has the 'Allow multiple references' option enabled. An Excel import file has been prepared and includes two different images (IDs CoffeeMaker carafe and CoffeeMaker situation) for the Product Images reference type.

	A	B	C	D
1	<ID>	<Name>	Product Images Asset Reference ID	
2	22196	22196	CoffeeMaker carafe	CoffeeMaker situation

Using the Import Manager's Map Data functionality, automatically map all possible columns, and then manually map each column of the new images to the Product Images reference type.



Note that as each individual cell is mapped to the same asset reference type, the values are added to the first cell in an input format that STEP can read [multi-values separated by a pipe (|)] in the same manner as the data was handled by the transformation in the previous example.



Complete the import wizard steps, noting that the selection on the Advanced tab determines how existing data in STEP is affected. The results of this import are identical as those in the previous example.

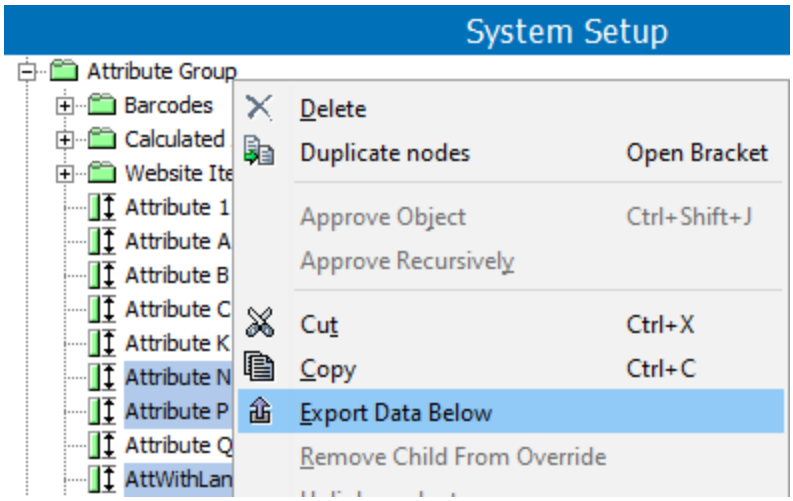
Managing Attribute Parameters with Excel

STEP attributes can be created and updated using an Excel import file. This allows a user to update multiple attribute parameters at the same time, or create multiple new attributes with the desired parameter settings.

The easiest way to manage STEP attributes using Excel is to first export one or more attributes, then modify the Excel output, and finally import the modified file back into STEP. Use the configuration defined below to perform all necessary steps.

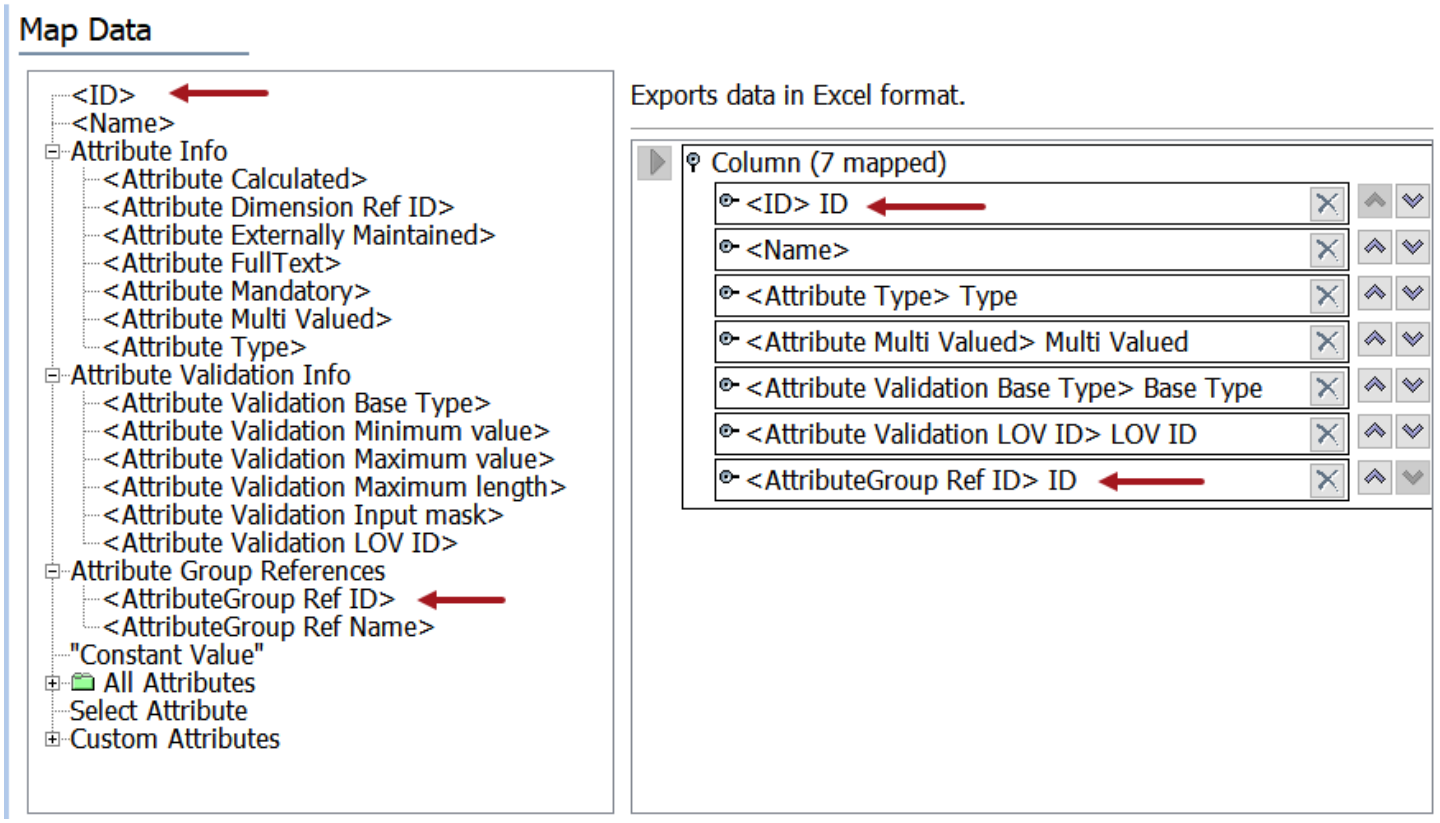
Configuration

1. In System Setup, select one or more attributes to export as follows:
   - To update existing attributes, select all attributes to be modified.
   - To create new attributes, select one attribute of each Validation Base Type to represent the attributes that will be created. These attributes will create a template to be modified for the new attributes. Validation Base Types are defined in the Validation Base Type table within the **Validation Rules** topic in the **System Setup**.
2. Right-click the selected attributes and click Export Data Below.



3. In the Export Manager, click **Next** to display the Map Data step, and map all required data sources as defined in the **Export Manager - Map Data** topic.

Important: At a minimum, you must map the ID and AttributeGroup Ref ID (to indicate the parent) for the new attribute.



4. Complete the Export Manager wizard as required and click the **Finish** button to display the Save Export Configuration window and generate the Excel file as defined in **Running a Data Export**.
5. Click **OK** to complete the wizard and start the export as defined in **Monitoring a Data Export**.
6. Open the Excel file and modify to create or update the attributes as desired, as shown in the image below:
   - Update the values for all columns that require changes. Based on the data sources you mapped, your export file may include different or additional columns than those shown.
   - Update the **Attribute Type** column to include "Property" for Description attributes or "Normal" for Specification attributes.
   - Update any LOV attributes so the **Attribute Validation Base Type** column is blank, and the **Attribute Validation LOV ID** column must include the ID of the LOV.

   The following image shows the original export file first, followed by the modified file that will create three new attributes. Notice that the ID, Name, and Attribute Type columns have been updated for all three attributes. The other columns have been modified as needed based on the previous notes.

	A	B	C	D	E	F
1	<ID>	<Name>	<Attribute Type>	<Attribute Multi Valued>	<Attribute Validation Base Type>	<Attribute Validation LOV ID>
2	AttributeP	Attribute P	Property	false	text	
3	Attribute N	Attribute N	Normal	false	number	
4	AttWithLangDepLOV2	AttWithLangDepLOV	Normal	false	LOV	DimensionDependentLOV2

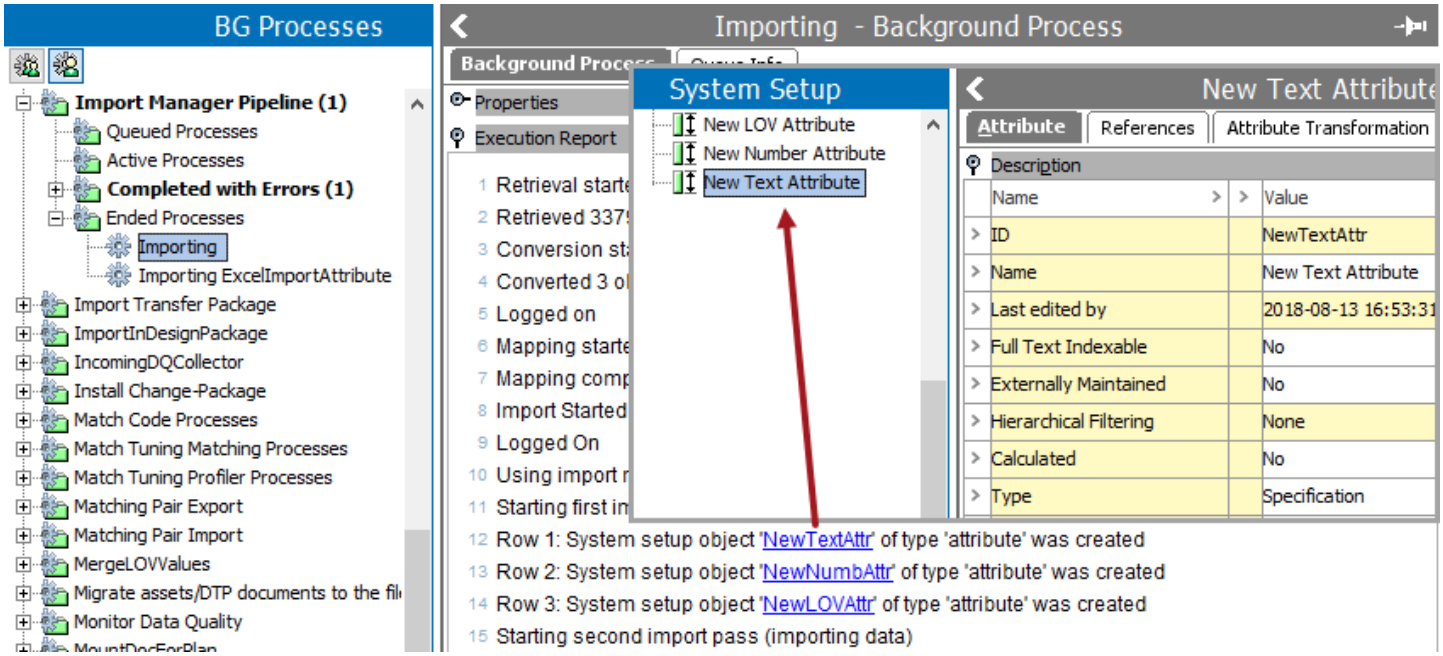
	A	B	C	D	E	F
1	<ID>	<Name>	<Attribute Type>	<Attribute Multi Valued>	<Attribute Validation Base Type>	<Attribute Validation LOV ID>
2	NewTextAttr	New Text Attribute	Normal	true	text	
3	NewNumbAttr	New Number Attribute	Normal	false	number	
4	NewLOVAttr	New LOV Attribute	Normal	false		DimensionDependentLOV

7. Save the modified Excel file and import using Import Manager, as defined in the **Creating a Data Import** topic. After completing the required parameters on a step, click **Next** to move to the next step in the wizard.
   - On the Select Data Source step, choose your modified Excel file, as defined in the **Import Manager - Select Data Source** section.
   - On the Map Data step, in the **Map to** parameter, select **Attribute** as the super type, as defined in the **Inbound Map Data - Map To** section.
   - On the Map Data step, click **Auto Map**, as defined in the **Inbound Map Data - Auto Map** section.

     Although the **AttributeGroup Ref Name** column is not auto mapped, it is not needed to create an attribute when the **AttributeGroup Ref ID** column exists in the import file.

   - If the **Attribute Dimension dependencies** column was included in the import file, manually map this column using the **Map** button, the Attribute Info radio button, and the **Dimension Links** option, as defined in the **Attribute Info - Map Inbound** section.
   - On the Identify Objects step, verify the data that will be created or modified, as defined in the **Import Manager - Identify Objects** section.
   - On the remaining steps, complete parameters as required and click Finish to display the Save Import Configuration dialog, as described in **Running a Data Import**.
8. View the result of the background process as defined in the **Monitoring a Data Import** topic.

   Click a link included in the Execution Report and change from the BG Processes tab to the System Setup tab. The attribute for the created or modified by the import is highlighted in System Setup.



Excel Custom Template

The Excel Custom Template is an export component that allows users to quickly export one or multiple objects into a polished Excel output. This is done by adding placeholder tags into an Excel template for the attribute values that should be exported. The placeholder tags are extracted from the template and STEP data is mapped to those placeholder tags in the template. In this way, presentation-ready Excel files can be exported. The example below depicts a product fact sheet, one possible output of the Excel custom template.

The versions of Excel that are available in STEP are:

- Excel 2007

It is recommended that the newest supportable version of Excel is used. File errors may occur during different STEP processes if using a template set to an older Excel version. For more information on supported versions of Excel, refer to the current **Platform and Software Support** documentation in the **System Release and Patch Notes**.

To access the Excel Custom Template, the following add-on component must first be installed on your system in addition to the normal update procedures for the release version being installed:

- formatted-excel

For on-premises systems, instructions for installing components can be found in the **SPOT Program** topic in the **System Administration Guide** found in **Downloadable Documentation**. For SaaS systems, contact Stibo Systems Support.

Format Availability

Excel Custom Template is available for selection in:

- Export Manager - refer to **Creating a Data Export**

Mapping

This format requires creating a data map between STEP and the data being exported, and may also include data transformations. For details, refer to **Data Mapping**.

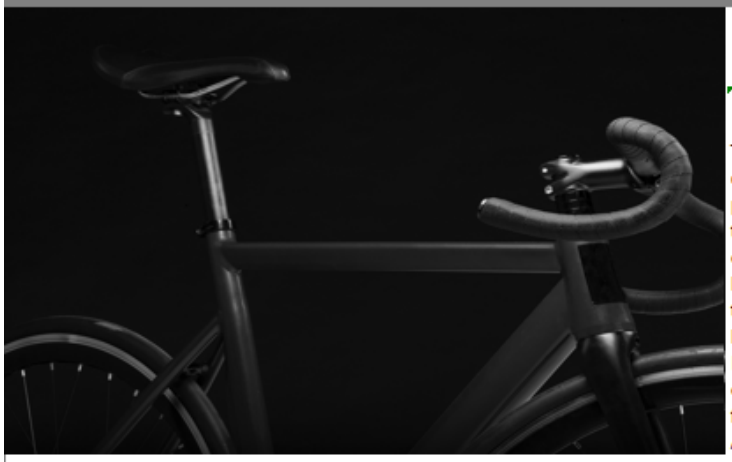
Output

The Excel Custom Template component is intended to produce Excel documents that are ready to present. Below is one example of how an Excel output might look.

Stibo Systems Bike Shop

Product Sheet

Men Road Bike



Flusdale FLU12 105 2017 Road Bike

1216 €

This is for the core, the local heroes, the crit assassins, the working-man racers with chiseled legs and dreams of glory, because they know that when it comes to pure performance for the buck, nothing can touch FLU12. Lighter, stiiffer and smoother than most carbon frames, it's not just the finest alloy racing bike ever made, it's the only choice for those who know.
Lorem ipsum dolor sit amet, proin potenti in a pellentesque at eu, eget elit amet ut turpis excepturi risus, porta sollicitudin felis phasellus wisi purus. Ut wisi tellus bibendum quis in, urna praesent ipsum eu placerat, pulvinar minima vestibulum leo augue elementum, nisl odio ad at condimentum aspernatur arcu, ut fermentum eget. Integer adipiscing fusce, suscipit in, nam pellentesque risus, nulla dictum fermentum auctor, nulla nulla est in. Nulla a morbi, urna purus etiam mauris dolor. Adipiscing nunc labore eros elit donec, lorem varius amet nisl magna rutrum odio,

Product Details

Frame:	FLU12, SmartForm C16069 Alloy, SPEED SAVE, BB30a, Di2 Ready	**Brakeset:**	Shimano 105 5800 caliper
Fork:	FLU12, SPEED SAVE, BallisTec full carbon, 1-1/8" to 1-1/4" steerer, integrated crown race	**Handlebar:**	Flusdale C3, butted 6061 Alloy Butted 6061 Alloy, Compact
Front Derailleur:	Shimano 105, braze-on	**Stem:**	Flusdale C3, 6061 Alloy, 31.8, 6 deg
Rear Derailleur:	Shimano 105	**Bar Tape:**	Flusdale Bar Tape w/Gel 2.5mm
Number of Gears	22	**Wheelset:**	Mavic Aksium WTS

Excel Custom Template Configuration

The Excel Custom Template is an export component that allows users to quickly export one or multiple objects into a polished Excel output.

Configuration

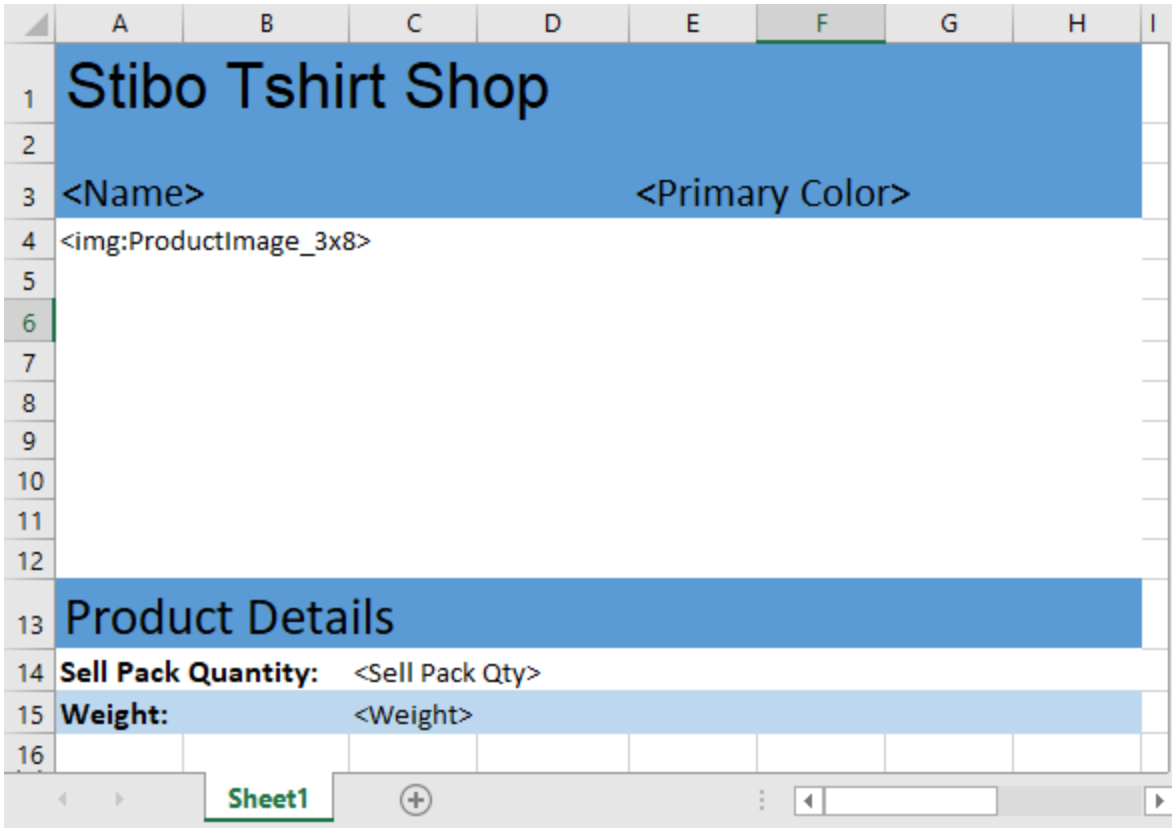
To configure an export of objects using the Excel Custom Template, follow these steps:

1. **Create Excel template with placeholder tags** – Users may set up a custom Excel template to fit a wide array of requirements. A template can include static elements, like headers or non-STEP images, that will display unchanged in the final export. To export attribute values from STEP into the export file, users must add placeholder tags to the template. A placeholder tag consists of any text appearing between a greater than (<) and less than (>) symbol. As long as the attribute is properly mapped in the Export Manager (addressed later in this same topic) and is both valid and instantiated for that object, the attribute value will be pulled into the exported Excel file.

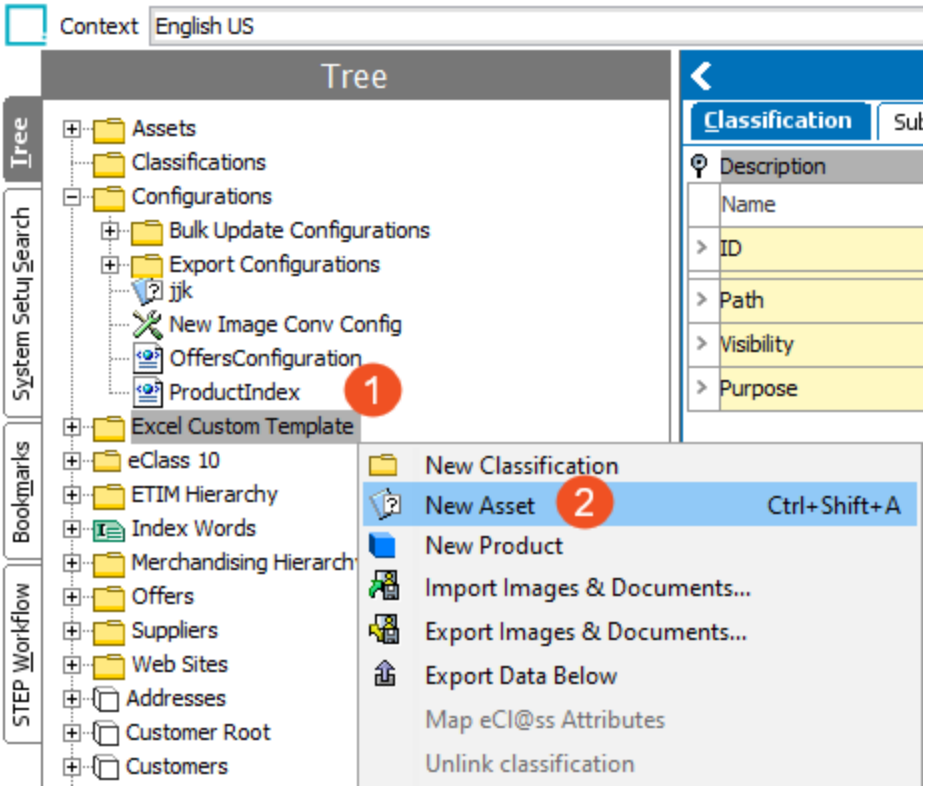
   STEP images may also be brought in, but the placeholder tag for images must be formatted differently.

   What follows is an example of a correctly formatted image tag: **<img:ProductImage_10x16>**.

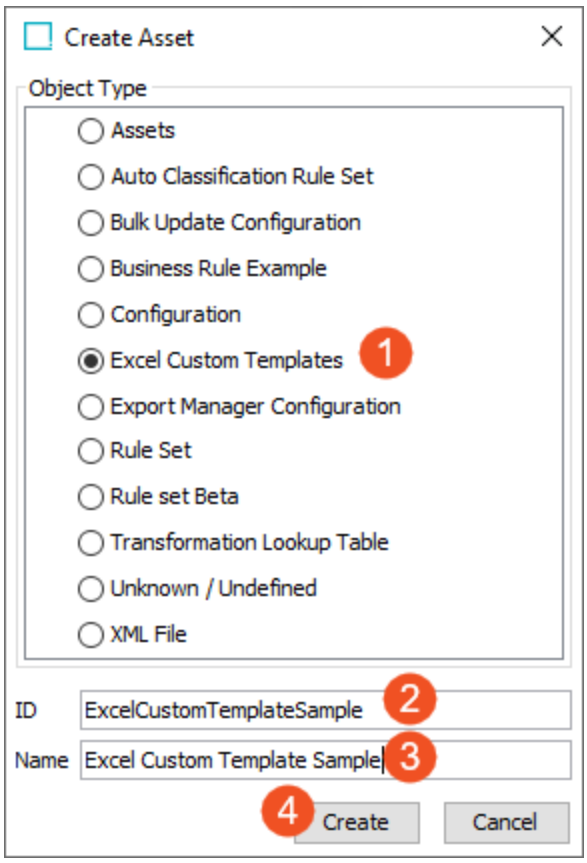
   Just as with attribute placeholder tags, image placeholder tags must begin and end with the greater than (<) and less than (>) symbols. Inside of these tags, the image placeholder tags must begin with 'img:.' Any text can be used following the colon, but recommended practice is to insert text relevant to the kind of image that should be exported. If the image will always be, for example, the primary product image, then the image tag text might be 'PrimaryProductImage.' It is important to note that only PNG, JPG, and JPEG image file types are allowed using the Excel Custom Template export format. Following the text is an underscore (_) followed by two numbers on either side of an 'x.' These numbers are dimensions, and dictate how many Excel cells the exported image will take up, the first determining the width (columns) of the image, and the second the height (rows). Below is an example of a simple Excel Custom template that will produce a simple Excel export file:



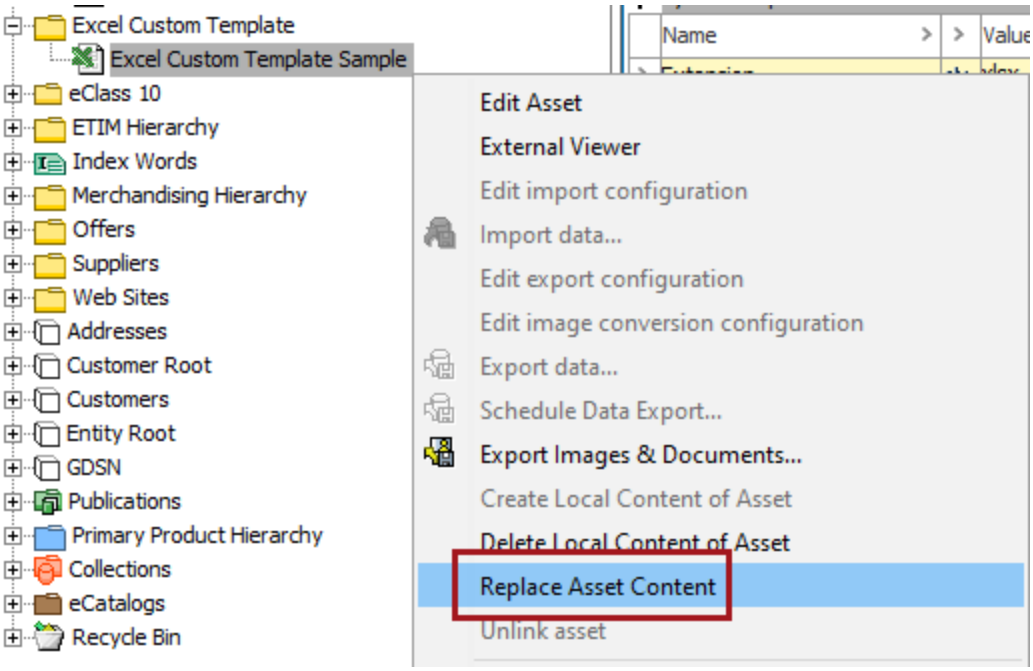
2. **Create a Custom Excel Template asset type** – In the System Setup tab in workbench, expand the Object Types & Structures node and select the 'Assets' node. Right-click 'Assets' and select 'New Object Type.' Assign an ID and name to a new object type ('Custom Excel Templates' would be appropriate), and click 'Create.'
3. **Save the template as an asset in STEP** – The Excel custom templates are stored in STEP as assets. During export of objects using the Excel Custom Template export format, users will navigate to the appropriate template and select it for the export. To save the template as an asset, users may elect to create an asset root node and title it 'Excel Custom Templates,' as shown in the screenshot below. To save the template as an asset, select the root asset in which the template will be saved, right-click, and select 'New Asset.'



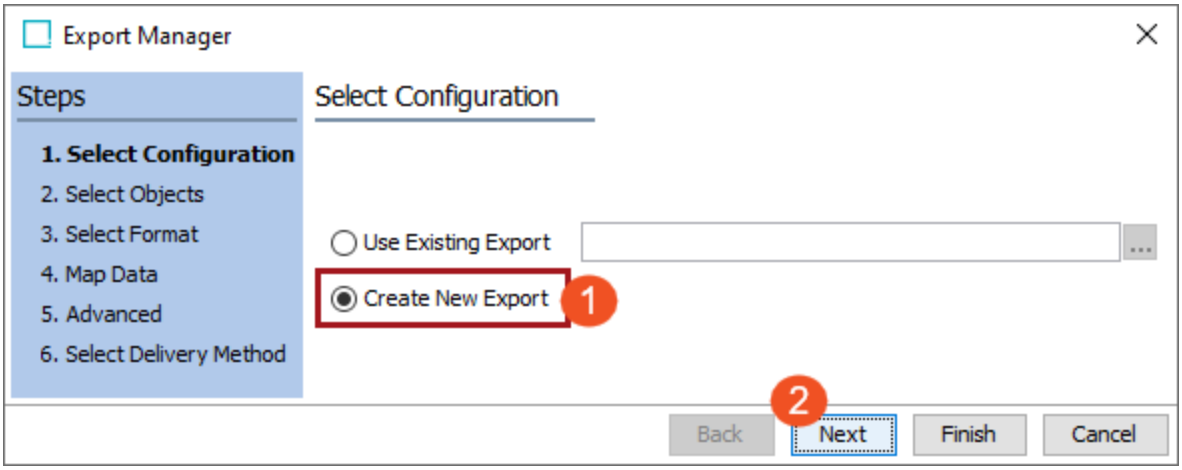
In the 'Create Asset' dialog, select the asset object type created in step two, in this case it is 'Excel Custom Templates,' and assign an ID and name to the template being saved.



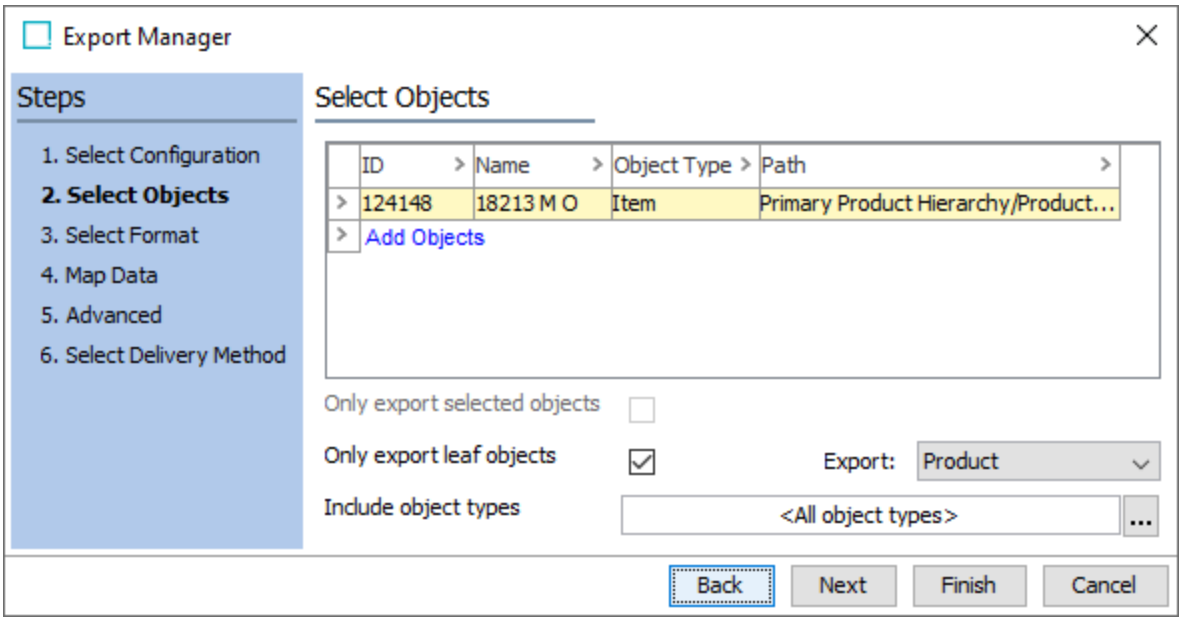
Be sure to also upload the template to the asset by right-clicking the asset and selecting 'Replace Asset Content.' STEP will prompt the user to select a file from the user's system to upload to the asset object.



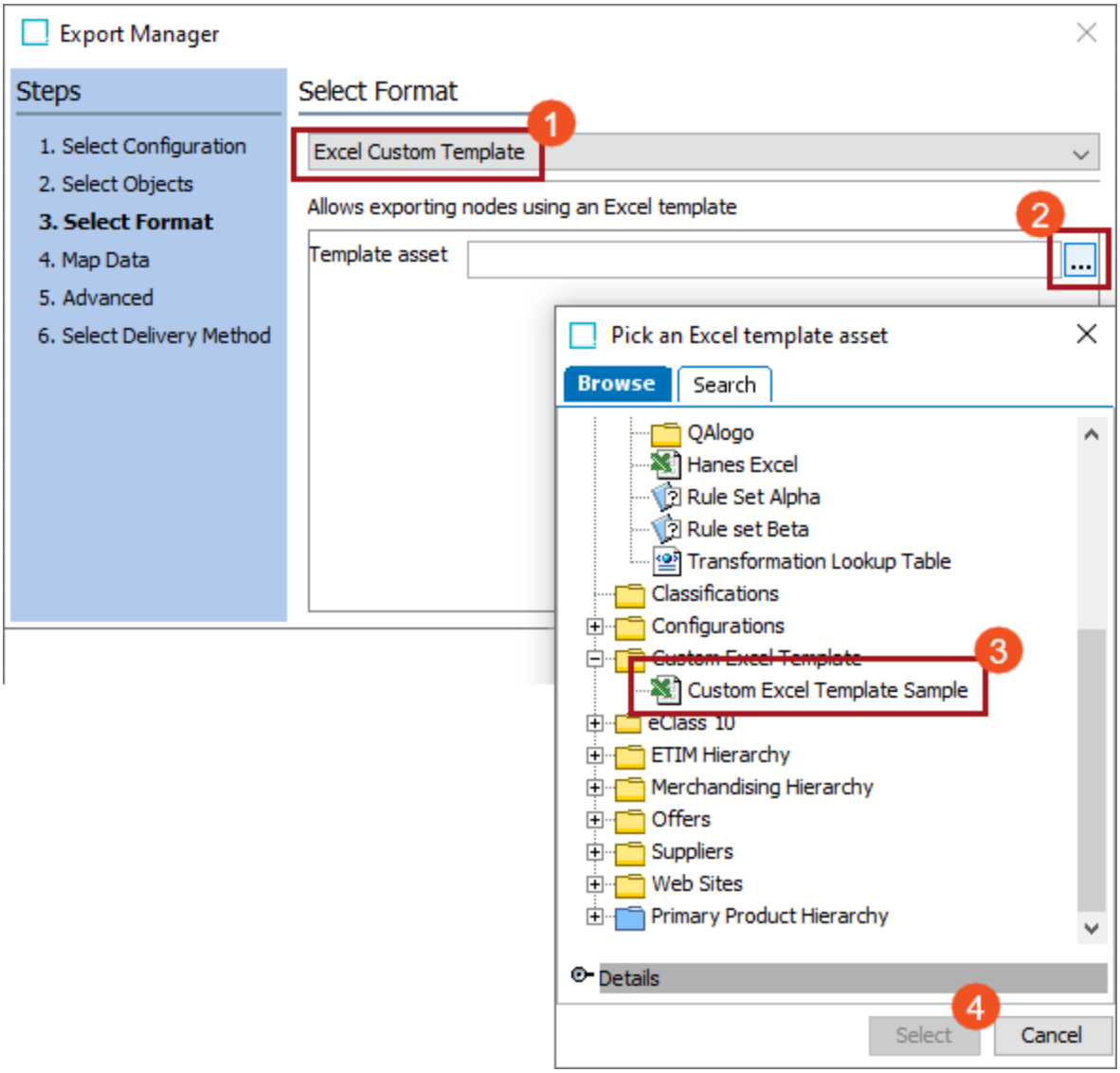
4. **Create an export configuration asset** – In order to retain the mappings that users establish in the Export Manager during configuration of their custom Excel export, it is useful to create and save an export configuration asset into which the export configuration can be saved. To create an export configuration asset, navigate to the 'Asset' root node, select it, right-click and select 'Add Asset' from the menu. Select the 'Export Manager Configuration' and then assign an ID and name to the asset.
5. **Export object(s)** – To access the Export Manager and generate an Excel export file from the custom template, select 'File' from the toolbar, then 'Export,' and then 'Data....' The user will map the placeholder tags to STEP data, and then execute the export. For more information on the Export Manager functionality, refer to the **Export Manager** section of the **Data Exchange** documentation.
    - **Select Configuration** – Select 'Create New Export,' then 'Next.'



    - **Select Objects** – Select one or more object(s). You may select multiple objects to export using a custom Excel template, but each object and its data will go into its own Excel file and then gathered into a .ZIP file, rather than all objects and their data going into a single Excel file.

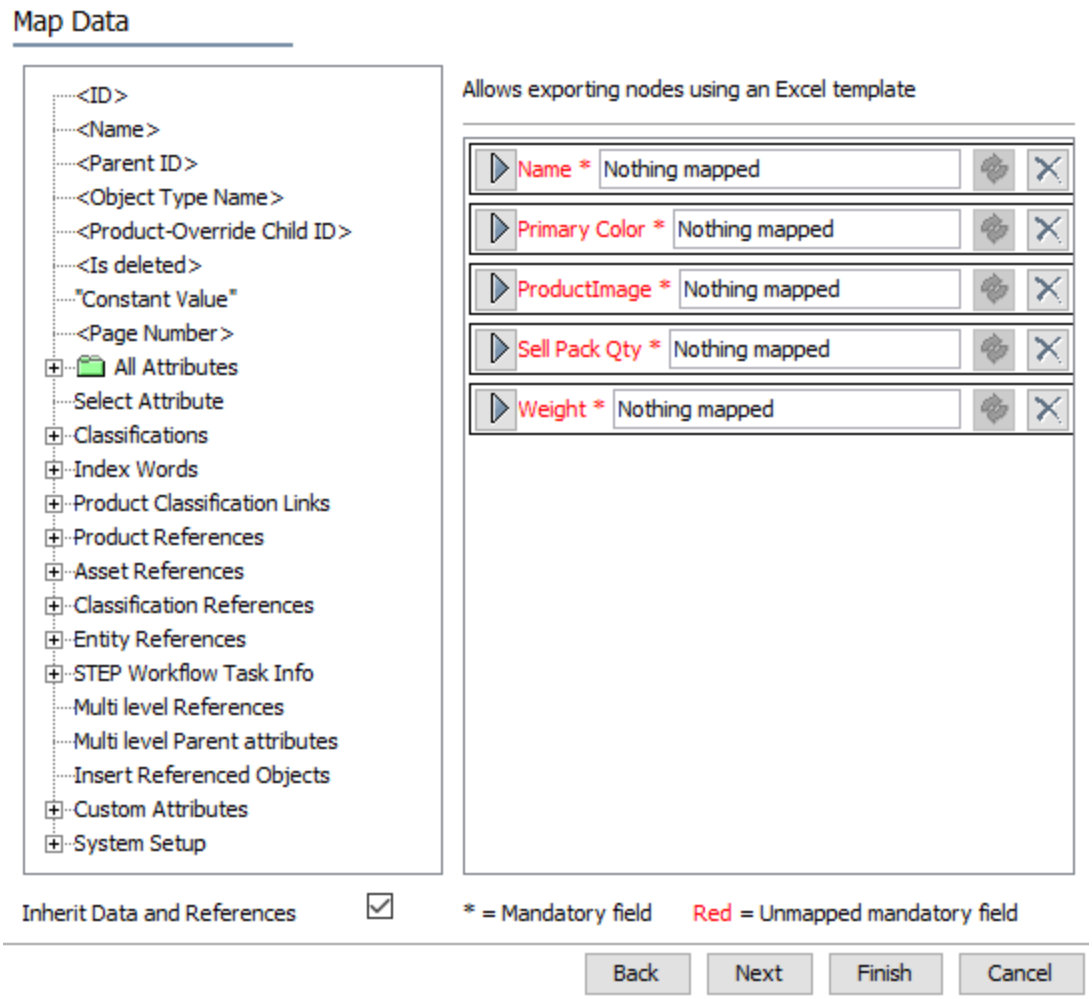


- **Select Format** – Select the 'Excel Custom Template' export format from the dropdown, and then, in the 'Pick an Excel template asset' dialog, select the Excel Custom template asset that was saved in step three.

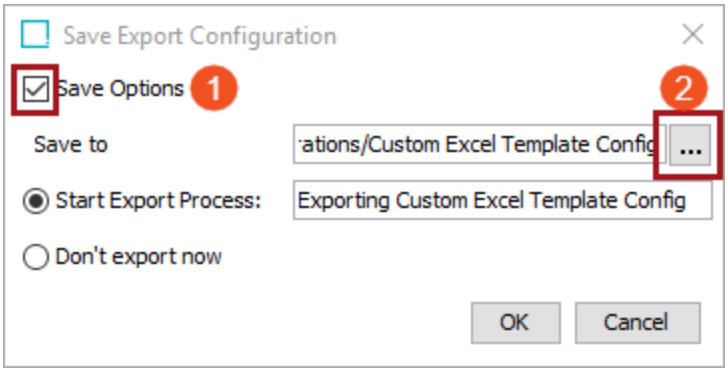


- **Map Data** – In the 'Map Data' screen, users will map the placeholder tags inserted into the custom Excel template to STEP attribute values. Listed in the right panel (mapping targets) are the placeholder tags that STEP read from the Excel template. In the left panel are the data sources. The placeholder tags must be mapped to a data source from the left panel for the data to export correctly into an Excel file. Map all targets in the right panel. When that is complete, click 'Next.'

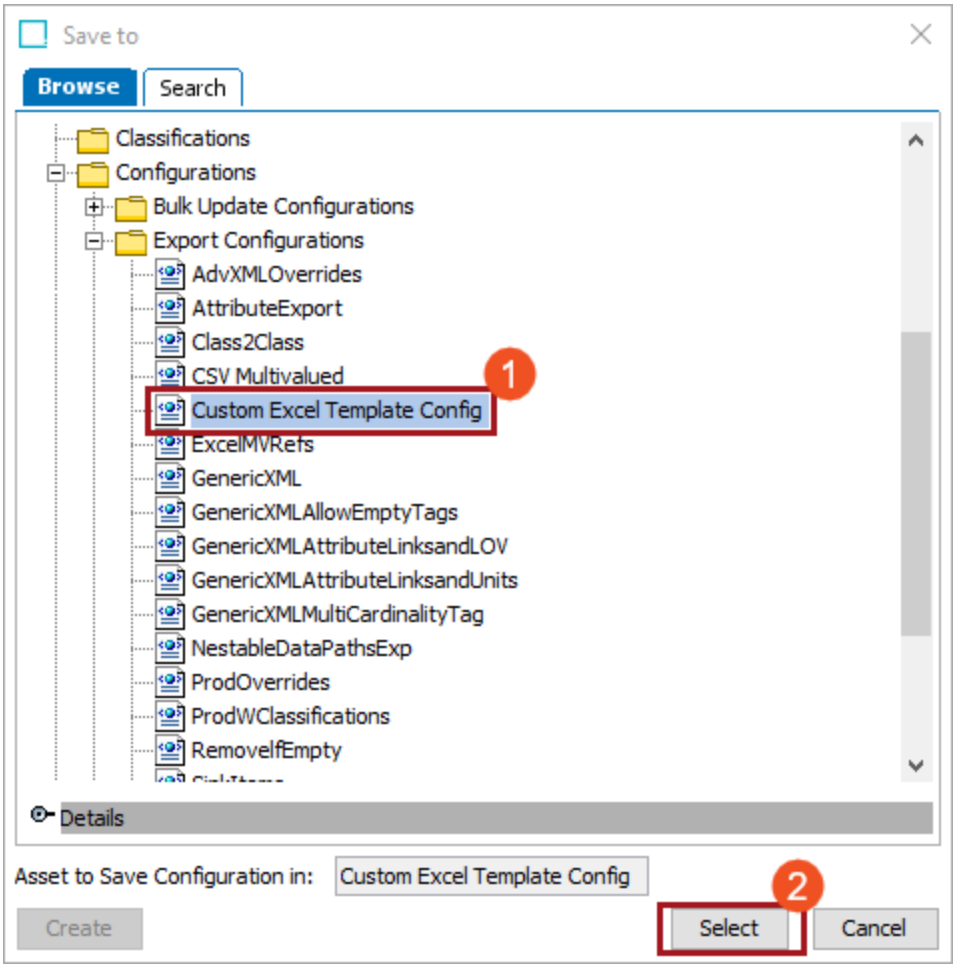
  For more information on mapping data, refer to the **Outbound Map Data - Data Source** topic.



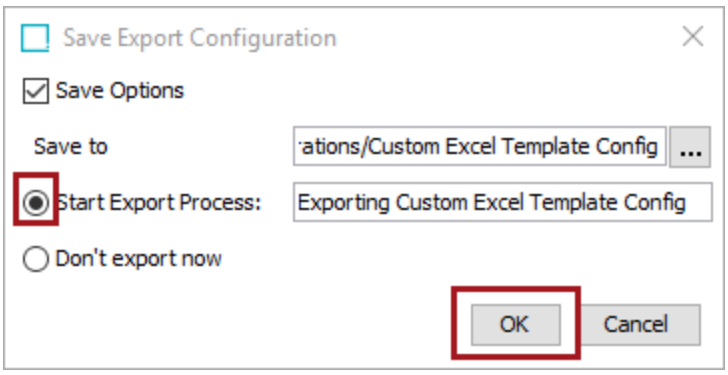
- **Advanced** – Make default selections, if applicable.
- **Select delivery method** – Select delivery method for the Excel document. For this example, 'File' has been chosen. Then click 'Finish.' Single-object exports will export to a single Excel file; multiple-object exports will export to a .ZIP file containing an Excel file for each exported object.
- **Generate and save the export configuration** – Before the export is executed, a 'Save Export Configuration' window displays. Users may leave the 'Save Options' box blank and execute the search as it has been configured. Doing this means the export can only be run once as configured. If an export of the object or objects using the custom template must be run again, the mappings established between the placeholder tags and the attributes and references must be re-established. This is why, if exports are to be run on a custom Excel template more than once, it is recommended practice to save the configuration as an export configuration asset. To do this, check the box for 'Save Options,' and then click the ellipsis button (...) for the 'Save to' parameter.



In the 'Save to' window that displays, navigate to the export configuration asset created in step four. Select the asset and click 'Select.'



Then, back in the 'Save Export Configuration' window, check 'Start Export Process' and then click 'OK.'



When the exported Excel file is opened (accessible in the completed background process), the result should look like the sample output shown below. This output is derived from the template image accompanying step one in this topic.

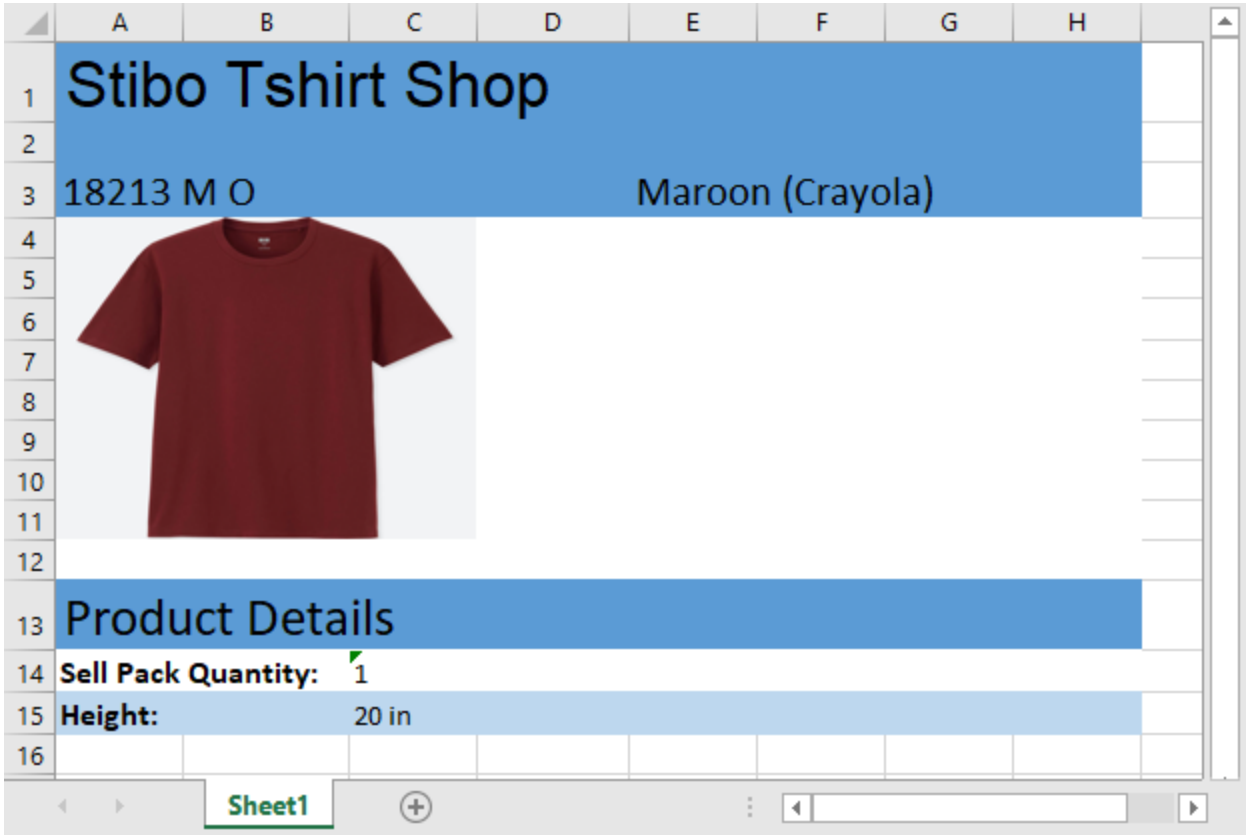
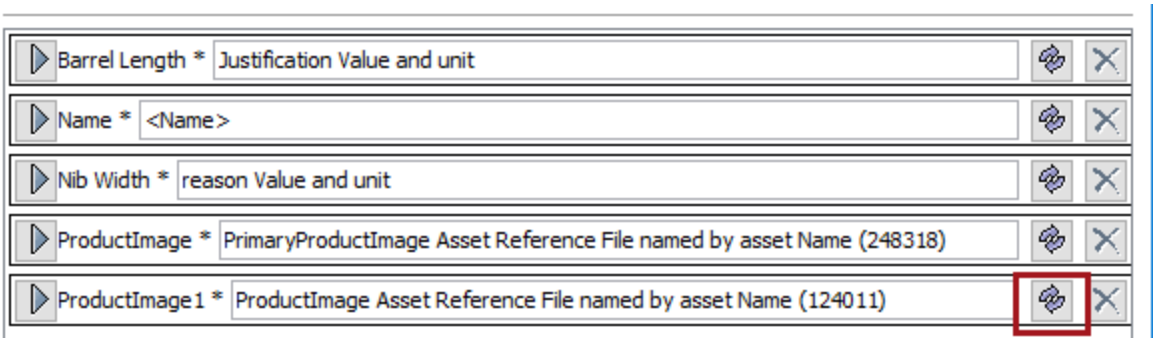
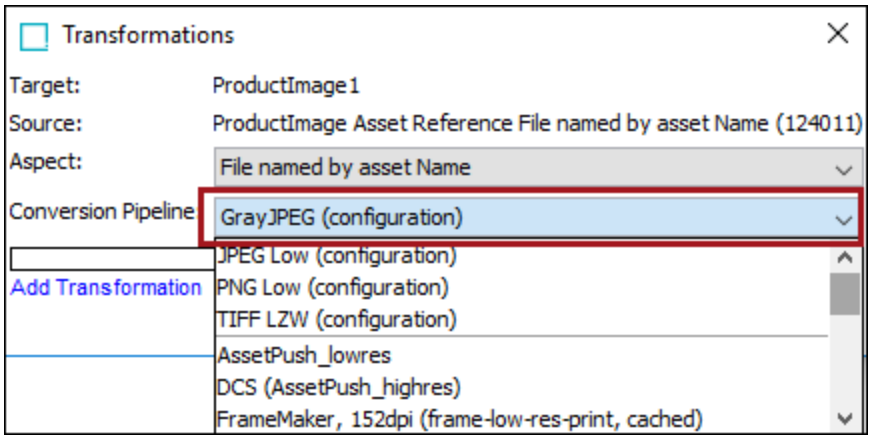


Image conversion configurations with Excel Custom Template

Users can also deploy an Image Conversion Configuration when exporting data using the Excel Custom Template. These are useful if users want to, for example, remove image distortion from exported images, or ensure that very large images are properly downscaled on export. To apply an image conversion configuration, users can click the 'Add Transformation' button during the 'Mapping' step once the images have been mapped to the appropriate image reference type.



In the 'Transformations' dialog that displays, users must select values for the 'Aspect' and 'Conversion Pipeline' parameters. For 'Aspect', select either 'File named by asset Name' or 'File named by asset ID.' Only these two aspects will enable application of an image conversion pipeline to the export. With either selected, the 'Conversion Pipeline' aspect's dropdown menu becomes active. Select the appropriate pipeline from the dropdown and save.



For more information on creating and configuring image conversion configurations, refer to the **Image Conversion Configuration** topic of the **Digital Assets** documentation.

It may be useful to keep the following points in mind when working with the Excel Custom Template export format:

- The user can also access the Excel Custom Template export from a Task List or Node List screen in Web UI using the 'Export Action' toolbar action button.
- There is no import aspect to the Excel custom template. Custom templates can only be used to export STEP data.
- All values will be handled as text in Excel. (**Not** applicable to images.)
- Data can only be exported to standard Excel cells.

Excel List of Values

The following sample Excel List of Values(LOVs) file shows that the first row is a header, and the remaining rows are data. Both Import and Export files always include a header row.

	A	B	C	D
1	<LOV ID>	<LOV Name>	<LOV Value ID>	<LOV Value>
2	Colours	Colours	279888	Green
3	Colours	Colours	279890	Red
4	Colours	Colours	279889	Blue

Note: The <LOV Value ID> column is available even for LOVs that do not use IDs on values. It is blank in that case.

Versions of Excel newer than 2007 can be used for imports. However, when exporting into Excel, the Excel version is set to Excel 2007, which allows the exported data to be read by later versions.

It is recommended that users use the newest supportable version of Excel. File errors may occur during different STEP processes if using a template set to an older Excel version. For more information on supported versions of Excel, refer to the current **Platform and Software Support** documentation in the **System Release and Patch Notes**.

Mapping

This format requires creating a data map between STEP and the data being processed, and may also include data transformations. For details, refer to **Data Mapping**.

Exporting List of Values

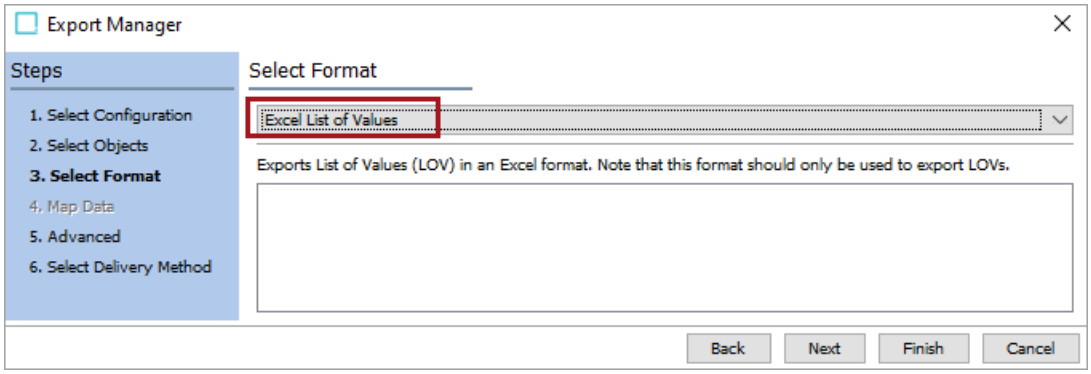
The export and import Lists of Values (LOVs) via the Excel format is done when configuring an outbound integration endpoint (OIEP) or an export in the Export Manager.

The successful imports of LOV data into existing LOVs in STEP via Excel files requires Excel files that are formatted with the same column headers. Excel List of Values is selected in the 'Format' field in the 'Select Format' step of the Export Manager when exporting a Excel file.

While import the field ‘Replace List of Values’ is set to ‘No’ by default and can be changed to ‘Yes’ if there is a requirement to replace the existing list of values with the imported list of values.

When the LOV value is dimension dependent, the export and import is based on the context selected during the export and import process. If the LOV is dimension dependent and there is an empty value in the selected context but it is non-empty in the other contexts, it exports the empty value from the selected context.

The ‘Excel List of Values’ has a limit of 100 million characters for the <LOV Value> column. If the import file exceeds the 100 million characters limit with respect to the <LOV Value> column, the import process gets aborted.


Export Manager
Steps
1. Select Configuration
2. Select Objects
3. Select Format
4. Map Data
5. Advanced
6. Select Delivery Method
Select Format
Excel List of Values
Exports List of Values (LOV) in an Excel format. Note that this format should only be used to export LOVs.
Back
Next
Finish
Cancel

Excel Smartsheet Format

Smartsheets are a specialized Excel format with the ability to validate products before being imported into STEP. Typically, this format is used when maintaining product data or when initiating new products into STEP via a supplier item Web UI. If used to maintain product data, users can download a Smartsheet from the Supplier Web UI, and once the desired changes have been made, the data can be automatically validated in Excel. The Excel file can then be uploaded back into the Supplier Web UI. New products can also be initiated into STEP via predefined Smartsheet templates.

The data export and template configurations are created in the Export Manager, then added to the relevant template widgets and the export action in a supplier item Web UI. An import configuration should also be created and defined on the 'SmartSheetImportAction' in a supplier item Web UI (in case an import configuration is not specified in the export configuration).

The Excel Smartsheet format option in STEP is not designed to support large amounts of data, for example, 10,000 products and validation business rules. If you experience performance issues working with large data sets / business rules / mappings, test the same export or import (with the Test Only Import checkbox) using the standard Excel format and compare the performance results. If the standard Excel format performance is acceptable, use that format instead of the Excel Smartsheet format.

Using Smartsheet format and a supplier item Web UI may require special licenses. Ask your Stibo Systems account manager for more information.

Format Availability

While Smartsheets are designed to be used from the Supplier Web UI, they can also be used by non-supplier users. However, future Smartsheet enhancements will most likely depend on the Supplier Web UI.

Note: Be sure to select the correct export template, 'SheetExporter,' when exporting Smartsheets otherwise an error will occur.

Important: Objects should **not** be exported to Excel Smartsheets from the workbench. Smartsheet templates exported from the workbench will not be able to correct supplier linking of products created during import and will also provide different browsing capabilities. A maintenance Smartsheet exported from STEP Workbench will not contain any information about workflow and state, so it will not be able to supply information about workflow mandatory columns. In addition, any business rules regarding the current workflow will be unavailable during import. It will also supply a different set of possible targets when maintaining product cross-references.

Setup Requirements

Before Smartsheets can be downloaded from a supplier item Web UI and/or imported into STEP, some initial configuration steps are required.

1. In Export Manager, create a Smartsheet data export and/or template configuration as described in **Smartsheet Data and Template Configurations**.
2. In Import Manager, create an import configuration as described in **Smartsheet Import Configurations**.

3. In the Supplier Web UI, configure the Smartsheet Import Action as described in the **Smartsheet Import Action** section of the **Web User Interfaces** documentation.
4. In the Supplier Web UI, configure the Smartsheet Export Action as described in the **Smartsheet Export Action** section of the **Web User Interfaces** documentation.
5. In the Supplier Web UI, configure the Smartsheet Export Widget as described in the **Smartsheet Export Widget** section of the **Web User Interfaces** documentation.

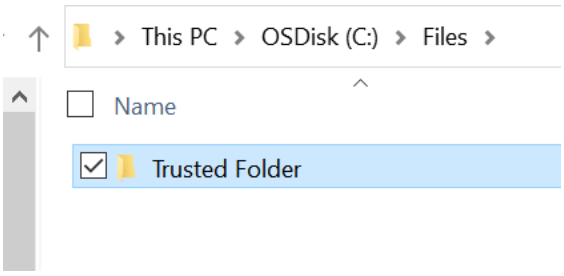
Additional Information

1. Review the available system configuration properties, the Smartsheet cover sheet tab, and Smartsheet special attribute handling, as described in the **Additional Information About Smartsheets** topic in this guide.
2. Review the available Web UI components for use with Smartsheets as described in the **Smartsheets in Web UI** topic of the **Web User Interfaces** documentation.
3. Understand the background processes and queues used with Smartsheets, refer to the **Smartsheet Background Processes and Queues** topic in this guide.
4. Alter the behavior of Smartsheets using configuration properties as described in the **Smartsheet System Properties** topic of this guide.
5. Review the available types of Smartsheet and the required handling as defined in **Using a Smartsheet** topic of this guide.
6. Understand how to initiate a product with a Smartsheet as defined in the **Initiating Products Using a Smartsheet** topic of this guide.
7. Understand how to maintain products with a Smartsheet as defined in the **Maintaining Products Using a Smartsheet** topic of this guide.
8. Understand the limitations of Smartsheet format and LOVs, as detailed in the **Smartsheet and LOVs** topic of this guide.

Trusted Folder Setup for Smartsheets

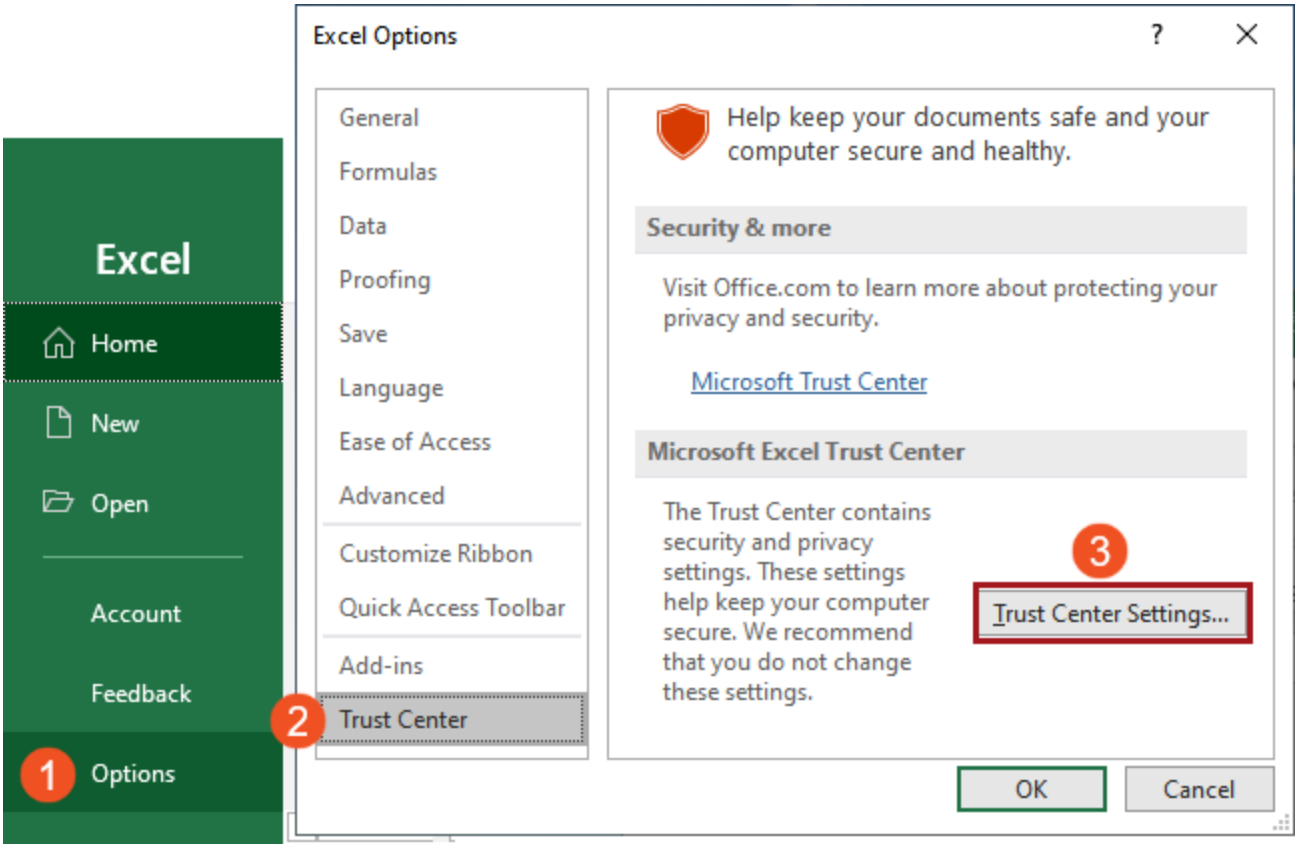
In order to open and use Smartsheets, which contain digitally signed macros in Excel 2007, some users may need to create a folder on their computer that has been deemed a 'trusted location.' Once placed inside the 'trusted location,' Smartsheets containing privately signed macros with Excel 2007 can then be opened and used. To set up a folder that has the 'trusted location' status, follow the instructions below:

1. Create a folder that will become the 'trusted location' on the computer where needed, and name it accordingly. In this example, it is called, 'Trusted Folder.'

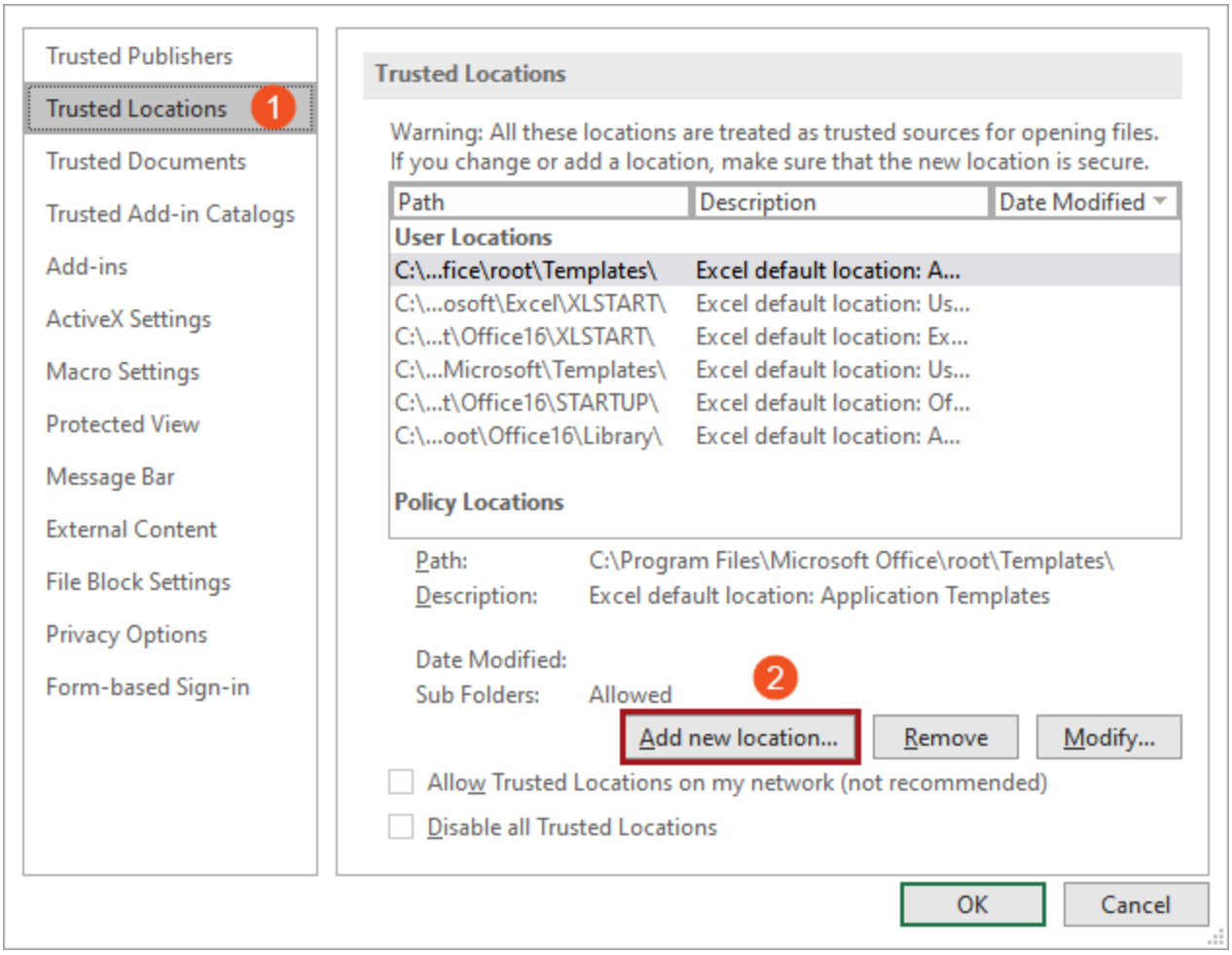


2. Open Excel, but do not open any spreadsheets or documents.

3. Click on Options > Trust Center > and click on Trust Center Settings.

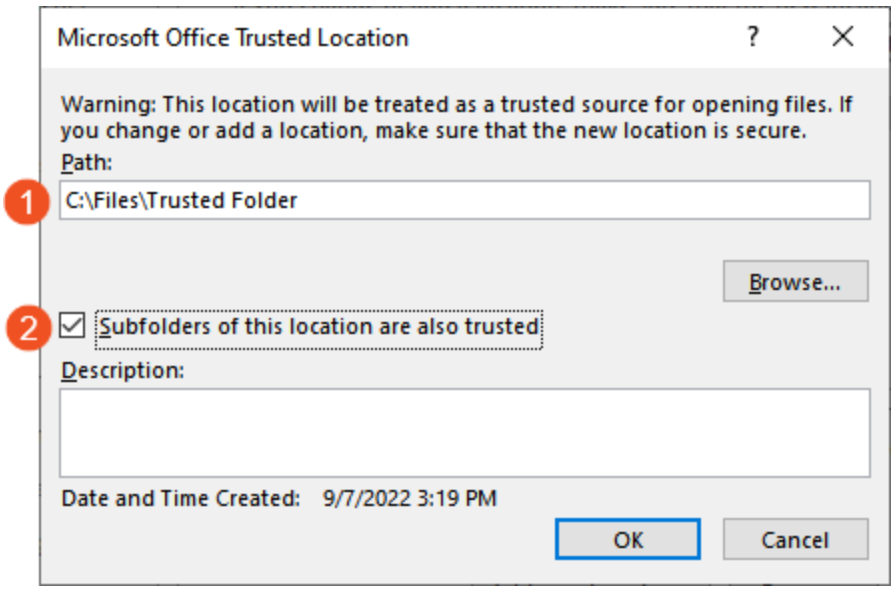


4. Click on Trusted Locations > Add new location...



5. Enter the path for the folder you created that will receive 'trusted location' status. The 'Trusted Folder' location path was entered.

   Additionally, click the box to enable 'Subfolders of this location are also trusted.'



6. Click 'OK' until all dialogs are closed.
7. Place all Smartsheets into the folder that received the 'trusted location' status before opening and using them.

Initiating Products Using a Smartsheet

Smartsheets offer suppliers a simple and efficient means for initiating new products into STEP. The guidelines laid out below cover the general process.

Prerequisites

Several setup steps are required before Smartsheets can be used to initiate new products into STEP:

1. In Export Manager, create a Smartsheet data template configuration as described in **Smartsheet Data and Template Configurations**. This export configuration should be configured on the Smartsheet Export Widget so that the end user can select it when exporting a Smartsheet template from the Supplier Web UI.

   Important: When exporting Smartsheets, it is important to remember to select the 'SheetExporter' that was created, otherwise an error will occur.

2. In Import Manager, create an import configuration as described in **Smartsheet Import Configurations**. This import configuration should be specified in the accompanying template configuration and/or a supplier item Web UI Smartsheet Import Action. Without a configuration, a Smartsheet cannot be imported into the Supplier Web UI.
3. In the Supplier Web UI, configure the Smartsheet Import Action as described in the **Smartsheet Import Action** section of the **Web User Interfaces** documentation. This action is used to import the new product data.
4. In the Supplier Web UI, configure the Smartsheet Export Widget as described in the **Smartsheet Export Widget** section of the **Web User Interfaces** documentation. This widget is used to export Smartsheet templates from the Supplier Web UI.

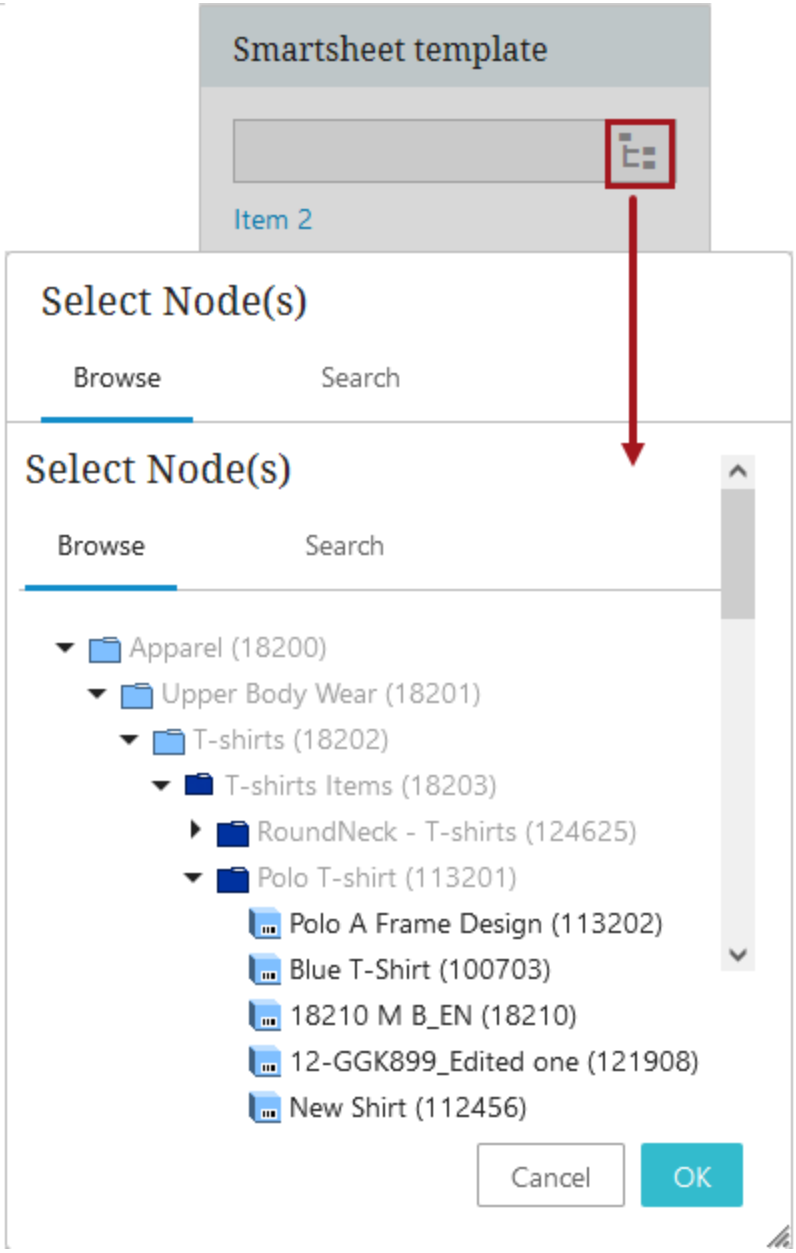
Product Initiation

New products can be initiated into STEP by: exporting a Smartsheet template from the Supplier Web UI, adding product data to the exported Excel file, and ultimately re-importing the edited Smartsheet back into the Supplier Web UI.

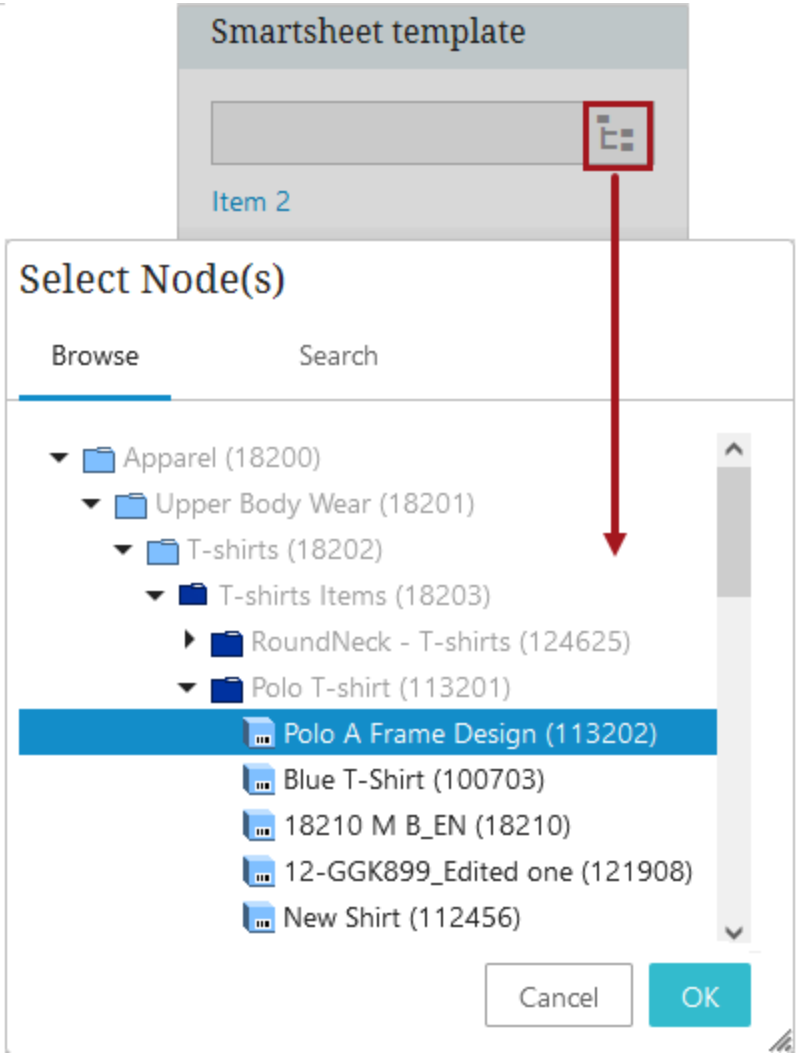
The Smartsheet Export widget allows export of a Smartsheet template, i.e., an empty Smartsheet that can be used to on-board new products. It allows you to start the download of the Multi-Level Smartsheet template, using an export configuration. Access is based on the user's supplier privileges, meaning that the component is only available if the user is a member of a supplier group.

Detailed steps for export are:

1. In the **Smartsheet Export** widget, click the browse hierarchy button to display the Select Node(s) dialog.

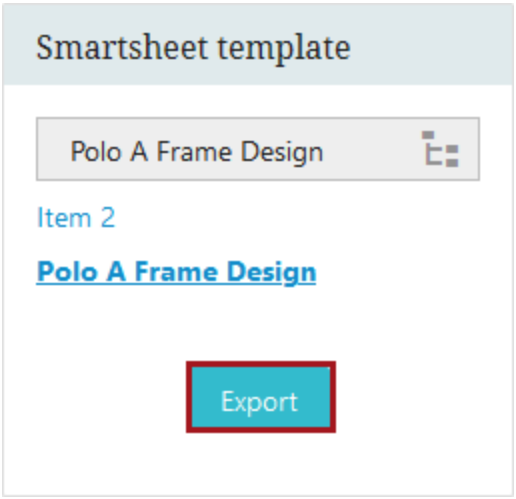


2. Select a valid node based on the object type(s) specified in the Smartsheet template and enable the OK button. An 'Invalid Selection' error is displayed and the OK button is not enabled when an invalid node is selected.



The category specific attributes that appear in the exported Smartsheet are dependent on the selected node and all applicable child nodes. No actual product data is exported.

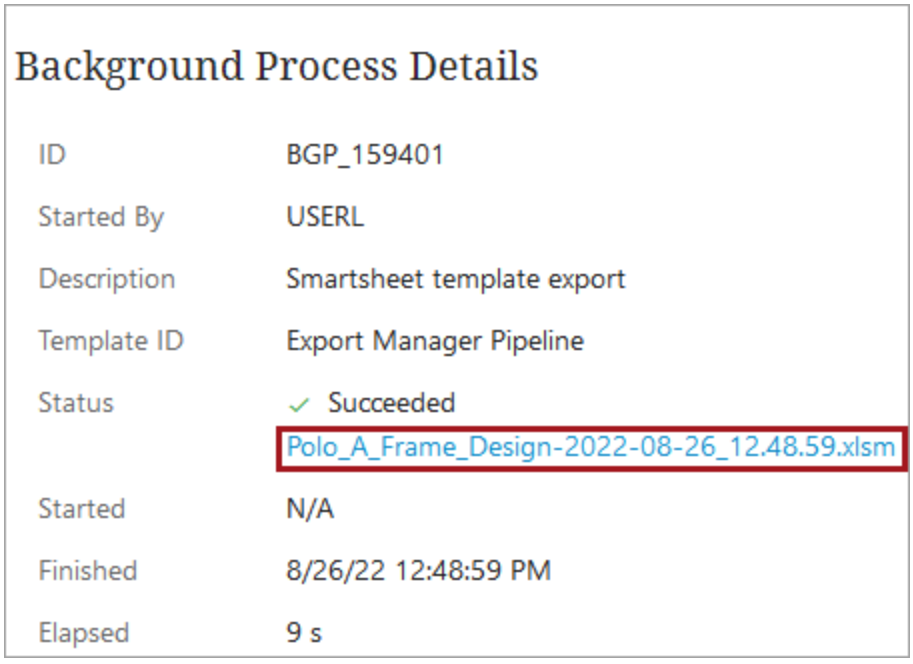
3. Click the **Export** button to start the background process and export the template.



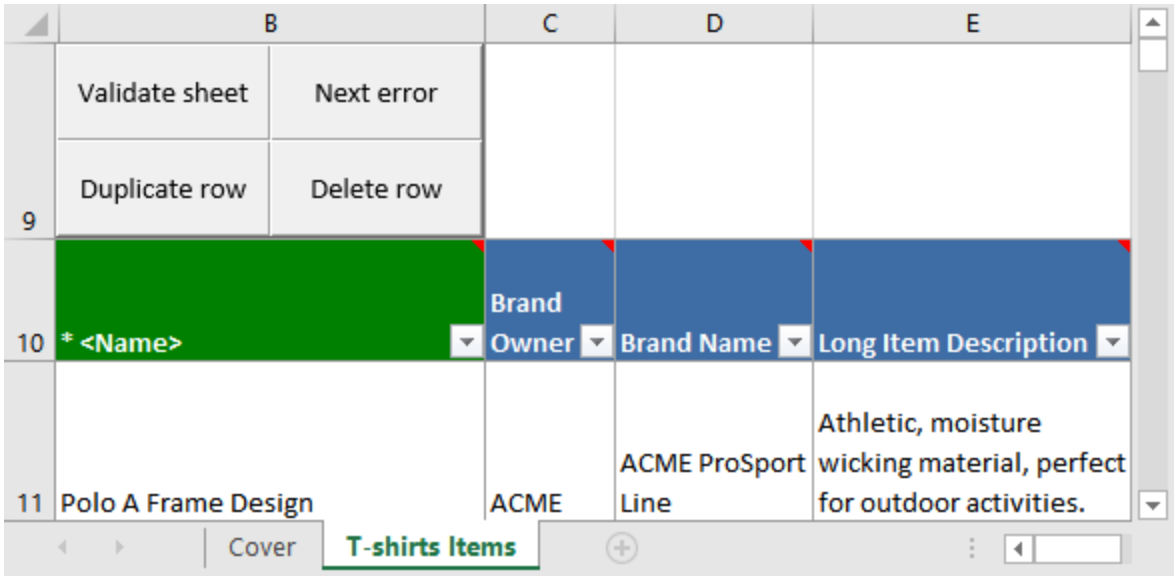
4. On the status dialog, click the link to the background process.

The Background Process Details page displays.

5. Upon completion of the background process, click the link (for the file extension .xlsm) and choose to open the template.



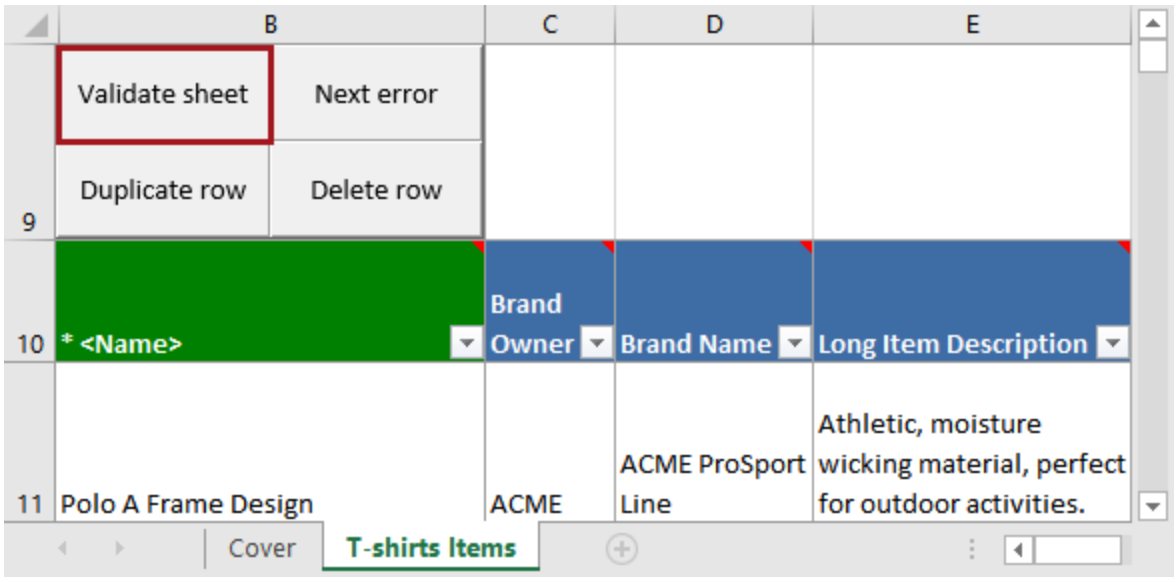
6. Enter product data into the exported Smartsheet.



For more information on navigating and editing a Smartsheet, refer to the **Using a Smartsheet** section of the documentation.

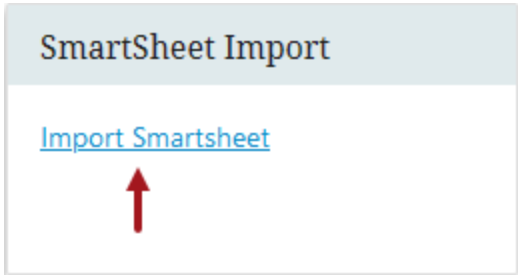
For information about the color coding of the cells, refer to the Cover tab, which is described in the **Additional Information About Smartsheets** topic.

7. If required, validate the sheet by clicking the **Validate sheet** button and following the sheet validation process.



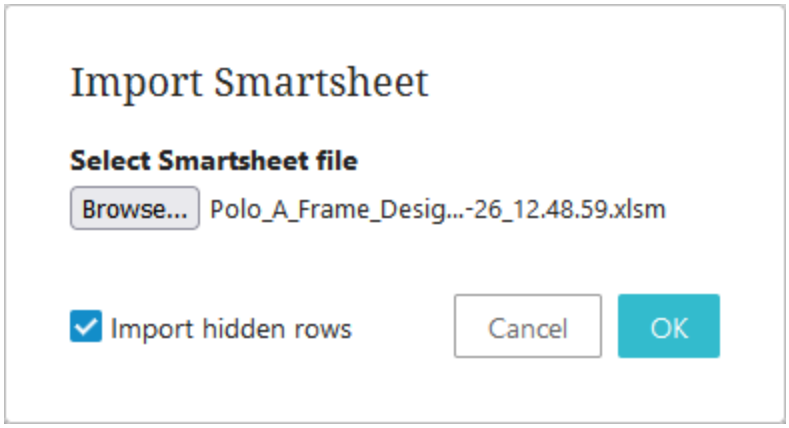
For more information on validating a Smartsheet, refer to the **Using a Smartsheet** section of the documentation.

8. Once the sheet has been validated, save the document.
9. Navigate back to a supplier item Web UI and click on the Smartsheet Import Action link.



10. In the Import Smartsheet dialog, click the **Browse** button to display a File Upload dialog. Navigate to the Smartsheet, select it, and then click **Open**.

    Checking the 'Import hidden rows' checkbox will ensure that any hidden rows will be included in the import. If you opt not to import hidden rows, the background process will indicate how many hidden rows were skipped.



11. Click the newly enabled **OK** button to import the data via a background process.

Note: If errors occur during import, a Smartsheet file with the products that failed can be downloaded from the import process page. The error messages are displayed as screen tips in the Smartsheet file when hovering over an error cell. **Next Error** can be used to troubleshoot a set of errors. When the errors have been corrected, the file can be imported again. However, there is limited support for this type of re-import. Some cells will be locked and consequently skipped during import.

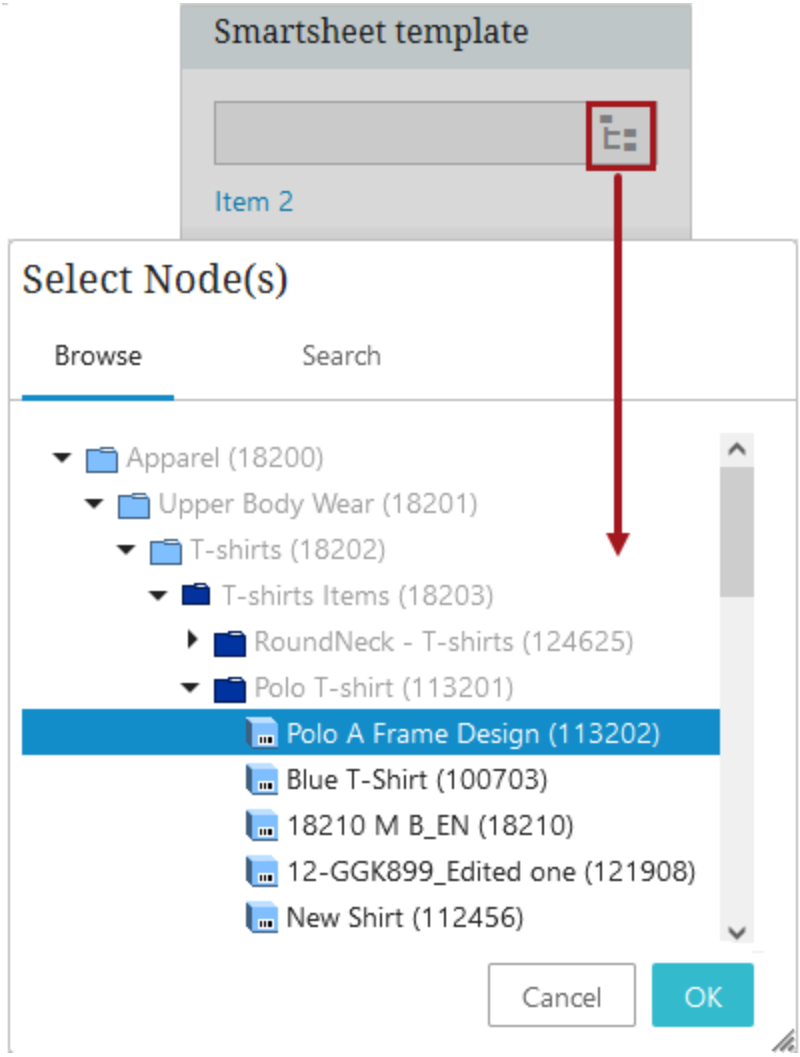
Exporting a Template with a Supplier Selector

Users that belong to multiple supplier groups can, within a single Smartsheet, load data for all suppliers to which they have access. Once configured, a supplier selector column will appear on the Smartsheet template. Users can select suppliers on a row-by-row basis, with each object being linked to the selected supplier upon import.

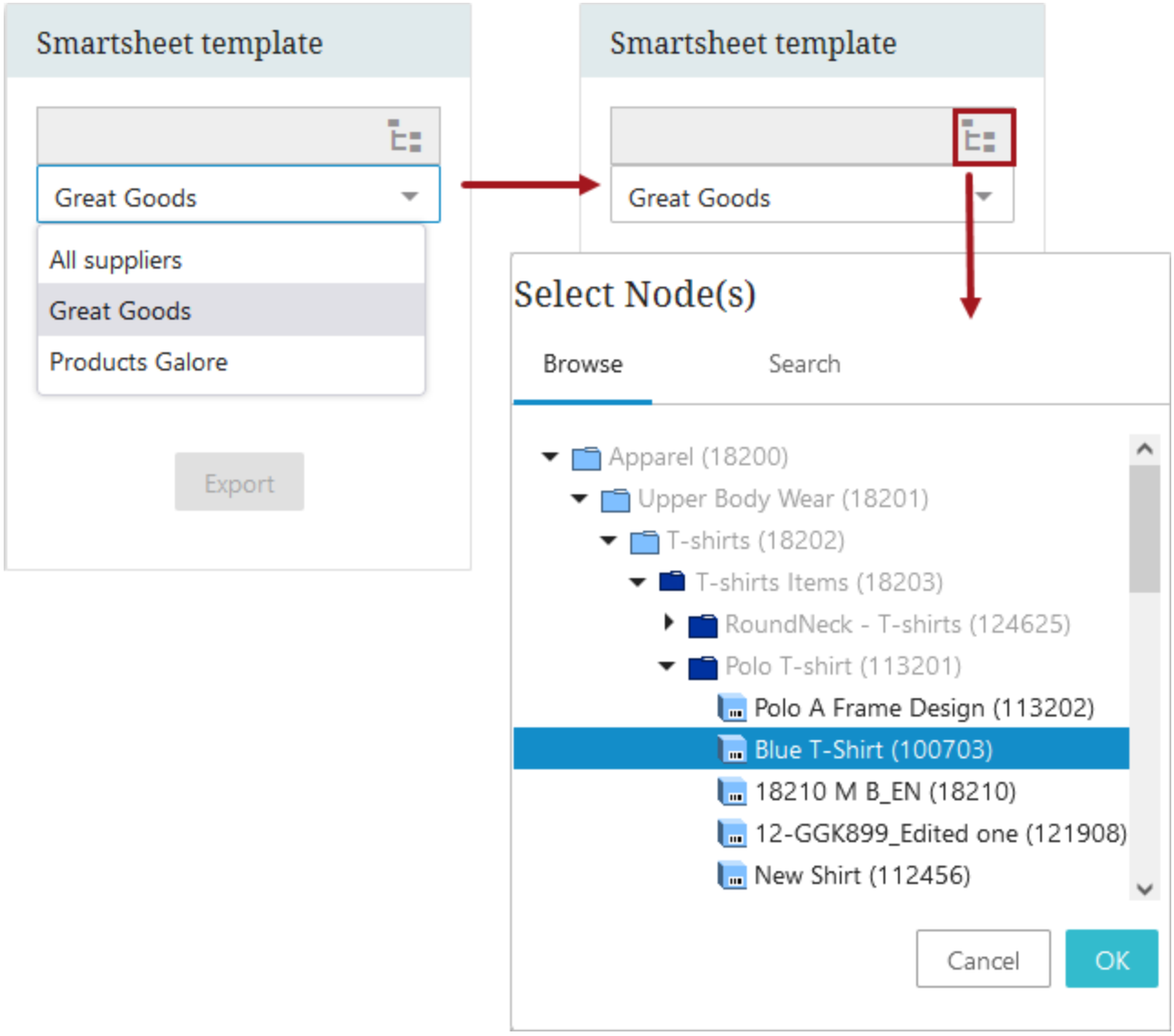
For information on how to configure a Smartsheet template with a supplier selector column, refer to the **Smartsheet Data and Template Configurations** section of the documentation.

Important: Before a Smartsheet can be exported with a supplier selector column, ensure that the 'Enable all-view for users that are a member of multiple suppliers' field is set to 'Y' in system properties.

1. In the **Smartsheet template** widget, click the browse hierarchy icon and select a valid node.



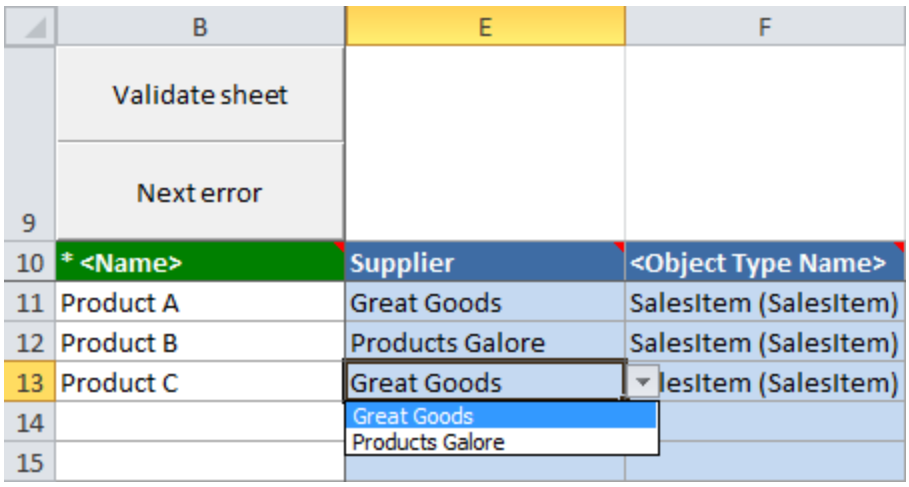
2. If the user is a member of multiple supplier groups the supplier dropdown will appear underneath the node picker. If the user is only a member of a single supplier user group, the supplier dropdown will not be present.



Users can only make a selection from the supplier groups they belong to. Additionally, when selecting 'All suppliers', the supplier selector column will only include the supplier groups that the user belonged to at the time of the export.

If the user is not a member of multiple supplier groups or the user specifies a supplier using the dropdown, the supplier selector column will not appear in the exported sheet. If 'All suppliers' is selected, the column will appear in the export.

3. Click **Export** and download the Smartsheet.



Users who do not belong to the same supplier group as a product they are working with will be unable to validate the product. Additionally, if the user does not belong to the applicable supplier groups when importing the data, the newly created products will fail to link to their designated supplier groups. To avoid this, set up a business condition to run on import, checking for the supplier link. If the condition fails, then the creation of the product in question is rolled back.

Important: When using a Multi-Level Smartsheet, the supplier selector column can only be included on a top level.

Maintaining Products Using a Smartsheet

Smartsheets offer suppliers an efficient means for maintaining product data in STEP. The guidelines laid out below cover the general process.

Prerequisites

Several setup steps are required before Smartsheets can be used to maintain product data in STEP:

1. In Export Manager, create a Smartsheet data template configuration as described in **Smartsheet Data and Template Configurations**. This export configuration should be configured on the 'Smartsheet Export Action' so that the end user can select it when exporting product data from a workflow state in the Supplier Web UI.

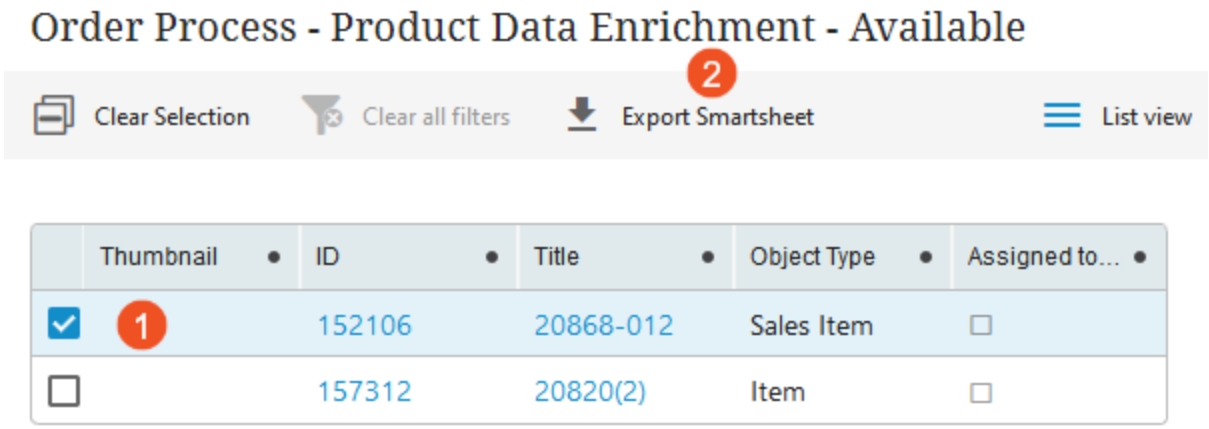
   Important: When exporting Smartsheets, it is important to remember to select the 'SheetExporter' that was created, otherwise an error will occur.

2. In Import Manager, create an import configuration as described in **Smartsheet Import Configurations**. This import configuration should be specified in the accompanying data export configuration and/or a supplier item Web UI Smartsheet Import Action. Without a configuration, a Smartsheet cannot be imported into the Supplier Web UI.
3. In the Supplier Web UI, configure the Smartsheet Import Action as described in the **Smartsheet Import Action** section of the **Web User Interfaces** documentation. This action is used to import the modified product data.
4. In the Supplier Web UI, configure the Smartsheet Export Action as described in the **Smartsheet Export Action** section of the **Web User Interfaces** documentation. This widget is used to export Smartsheet maintenance sheets from the Supplier Web UI.

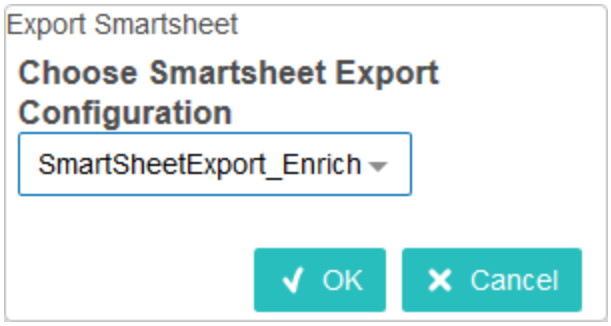
Product Maintenance

Product data can be maintained via a Smartsheet by: downloading product data from the Supplier Web UI, editing the product data in the exported Excel document, and ultimately re-importing the edited Smartsheet back into the Supplier Web UI. Detailed steps are listed below:

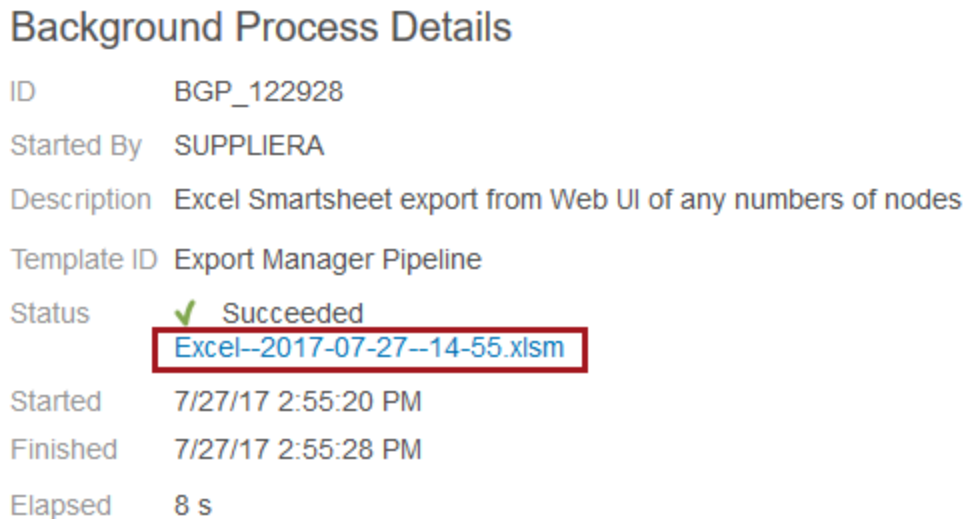
1. In the Supplier Web UI, navigate to the workflow widget and click the relevant workflow state. Then, on the Tasklist screen, individually select the products to export, or use multi-select, or the Select All button.
2. Click the Smartsheet Export Action button. In the sample image, this button is labeled 'Export Smartsheet.'



3. In the export dialog, select an export configuration from the dropdown and click **OK**.



4. Review the results of the successful background process, and click the file name link to open the Smartsheet for editing.



5. Modify the product data in the exported Smartsheet.

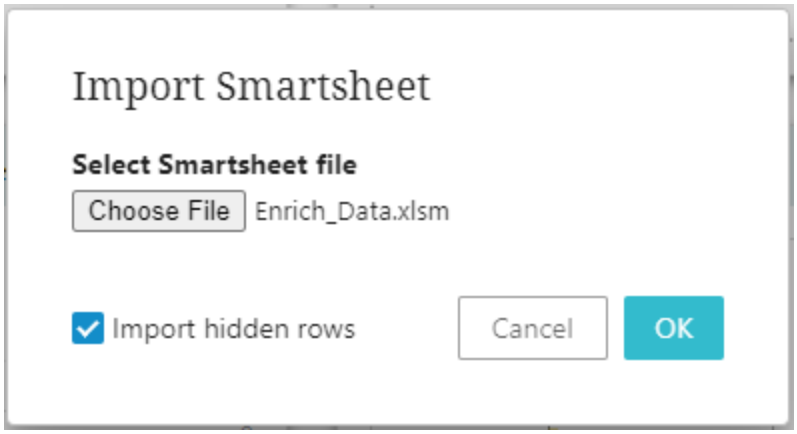
For more information on navigating and editing a Smartsheet, refer to the **Using a Smartsheet** section of the documentation.

6. If required, validate the sheet by clicking the **Validate sheet** button and following the sheet validation process.

   For more information on validating a Smartsheet, refer to the **Using a Smartsheet** section of the documentation.

7. Once the sheet has been validated, save the document.
8. Navigate back to a supplier item Web UI and click on the Smartsheet Import Action link.
9. In the Import Smartsheet dialog, click **Choose File** and select the Smartsheet, then click **OK**.

   Checking the 'Import hidden rows' checkbox will ensure that any hidden rows will be included in the import. If you opt not to import hidden rows, the background process will indicate how many hidden rows were skipped.



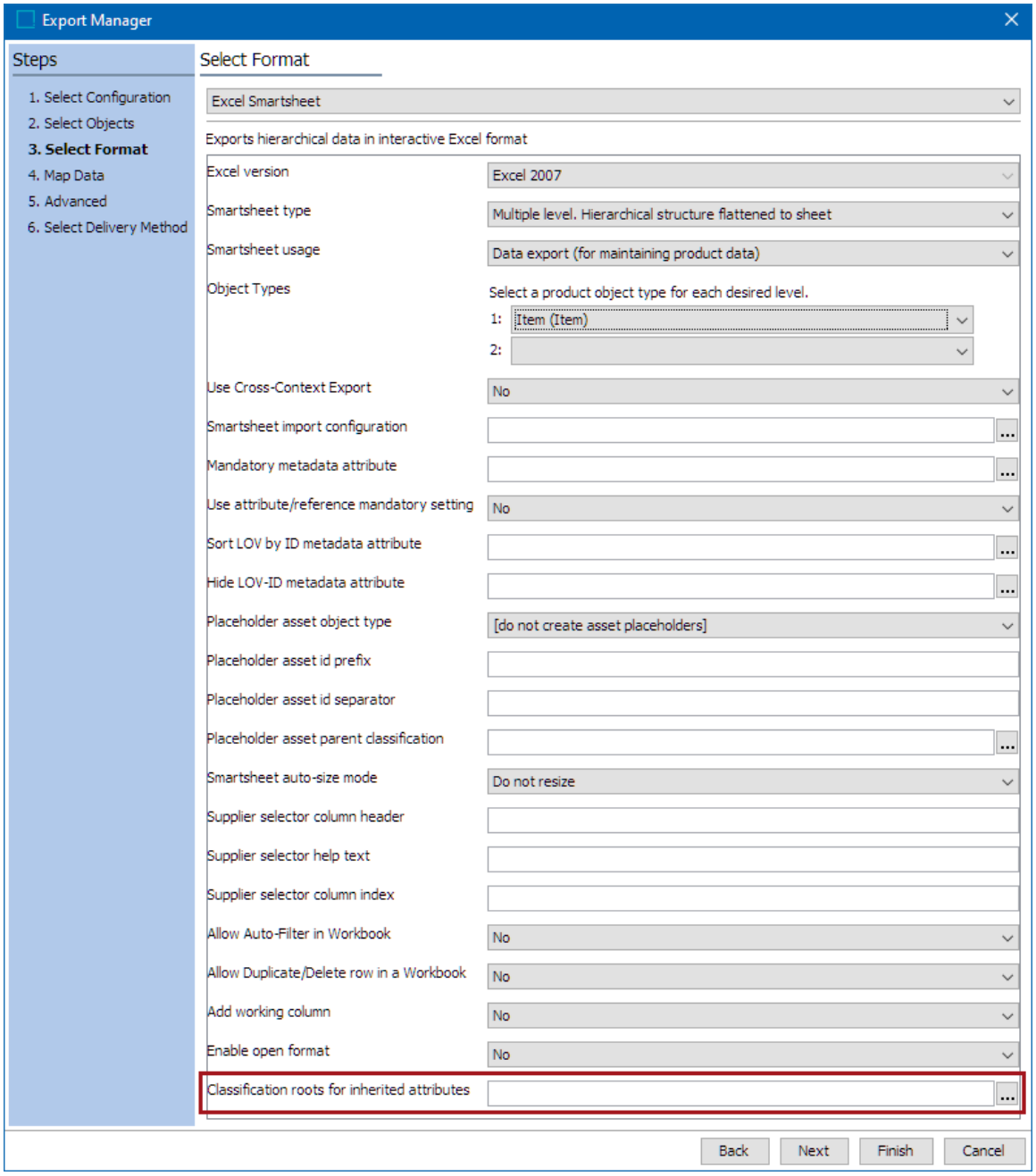
Note: If errors occur during import, a Smartsheet file with the products that failed can be downloaded from the import process page. The error messages are displayed as screen tips in the Smartsheet file when hovering over an error cell. **Next Error** can be used to troubleshoot a set of errors. When the errors have been corrected, the file can be imported again. However, there is limited support for this type of re-import. Some cells will be locked and consequently skipped during import.

Classification Roots for Inherited Attributes

Smartsheets can include attributes linked from classifications for onboarding and maintenance of data in STEP.

Important: Currently, in order to get the classification linked attributes exported properly when using this parameter, the user must select the exact classification node that the attribute(s) are linked to. Additionally, not all LOV-based attributes are handled correctly during export, and do not always display as read-only in the Smartsheet.

The parameter 'Classification roots for inherited attributes' found on the 'Select Format' step of the Export Manager wizard, allows system administrators to configure one or more Product Classification Links Types when configuring a Smartsheet. When setting up a Smartsheet export, it is essential to ensure that the attributes linked to the classification are valid for exported object types. In case if the attributes with valid object types are not linked, the attributes will not export in the Smartsheet.



When the 'Classification roots for inherited attributes' parameter is configured, meaning classification root nodes have been selected and displayed in the field, the Export Manager determines the valid attributes for new products in the product category. If a Product Classification Link type allows for inheritance of attribute validity from the linked classification then also the Export Manager determines the valid attributes for new products in the product category. These attributes are included in the Smartsheet output.

If nothing has been configured for 'Classification roots for inherited attributes,' there are no changes to the current way of determining what should be exported.

If one or more Classification roots have been configured, the system checks for any attributes that are linked below the Classification roots. It checks whether Smartsheet usage is configured as a Template export (for onboarding new products) or as a Data export (for maintaining product data).

When the Smartsheet format is selected, Classification roots for inherited attributes parameter checks for:

- Each product category selected for export has a Product Classification Link type to one or more nodes below the classification root. If nothing is selected there are no changes to the current way of determining what should be exported.
- If the link type 'Inheritance of Specification Attributes / Data Container Types' is configured as 'No,' there are no changes to the current way of determining what should be exported.
- If the link type 'Inheritance of Specification Attributes / Data Container Types' is configured as 'Yes,' it finds all attributes below the root valid for the exported products via classification links. If these attributes are mapped in the export configuration, then the STEP XML generated also includes valid attributes from the product hierarchy.
- If the link type 'Ignore LOV Filter definitions on inherited Attributes for links' is configured as 'No,' there are no changes to determining how LOV values should be exported.
- If the link type 'Ignore LOV Filter definitions on inherited Attributes for links' is configured as 'Yes,' it determines which LOV attributes have a filter applied and collects valid values for the selected product category.

For onboarding new products, the attribute validity will apply via Classification inheritance for each product category, and the filtered LOV values will be the same for all new products created. The Smartsheet file will have attributes that are valid via classification links.

For maintaining product data, the Smartsheet includes attribute columns that are valid via classification links. If the attribute is not valid for a given product row, the intersecting cell will be read-only. When the LOV attributes have a filter applied, each product row only allows valid values.

Important: Attributes that are inherited through a product-classification link cannot be exported to a Smartsheet.

For more information on configuring these functions, refer to the **Smartsheet Data and Template Configurations** topic of the documentation.

For more information on maintaining Classification roots, refer to the **Classification Roots for Inherited Attributes Use Cases** topic of the documentation.

Classification Roots for Inherited Attributes Use Cases

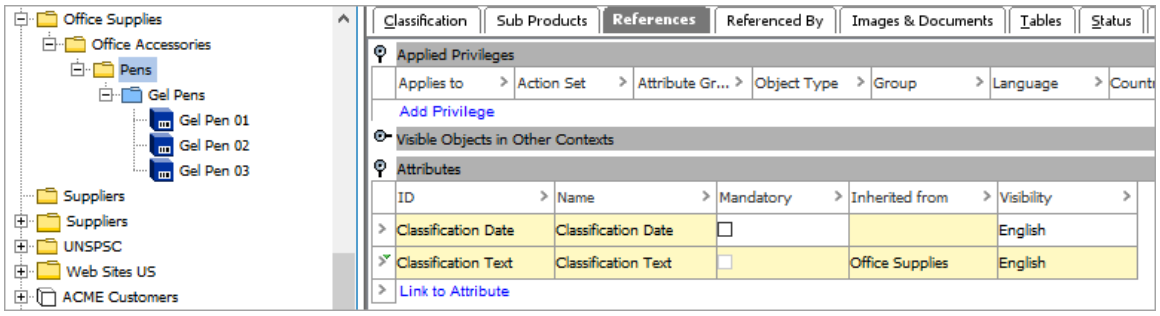
Inheritance Scenario

Important: Currently, in order to get the classification linked attributes exported properly when using this new parameter, the user must select the exact classification node that the attribute(s) are linked to. Additionally, not all LOV-based attributes are handled correctly during export, and do not always display as read-only in the Smartsheet.

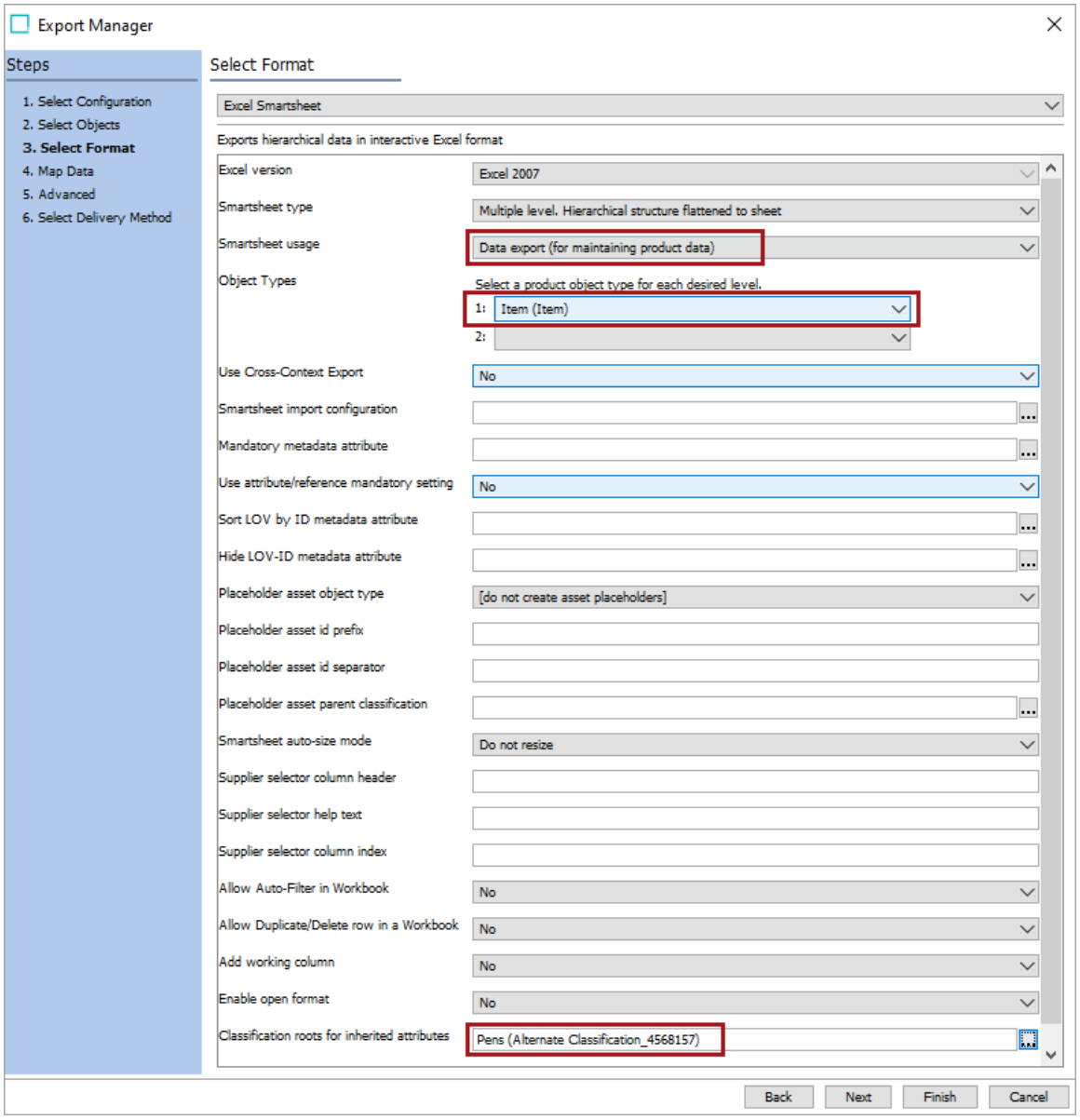
The attributes linked to the Classification when exported using Excel Smartsheet will include only the linked attributes. To get the inherited attributes the user should select the parent hierarchy in the parameter 'Classification roots for inherited attributes' found on the 'Select Format' step of the Export Manager wizard.

When the user selects the Classification Folder **Pens** in the parameter 'Classification roots for inherited attributes', the attribute **Classification Date** will be exported and it will not include the attribute **Classification Text** as it is inherited from **Office Supplies** even if it is mapped in the Export Manager wizard.

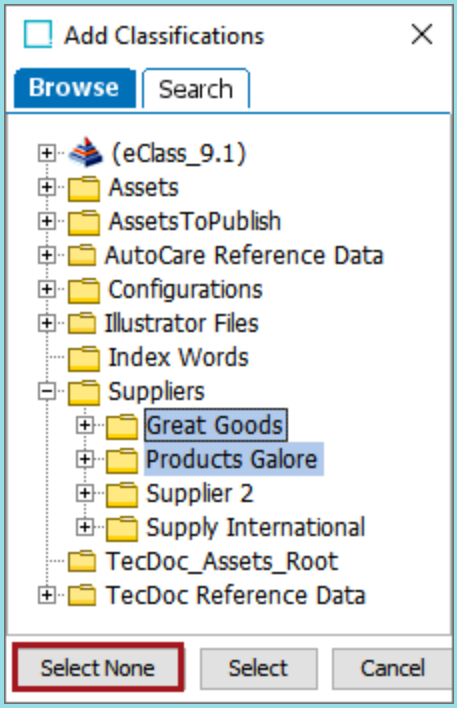
1. Select the products under the classification and export it using Export Data Below option. Select **Only export selected objects** to export only the root nodes, and not the objects that live below the selected node.



2. In Select Format window, for Data export (for maintaining product data) Select the **Object Types** and **Classification roots for inherited attributes**.

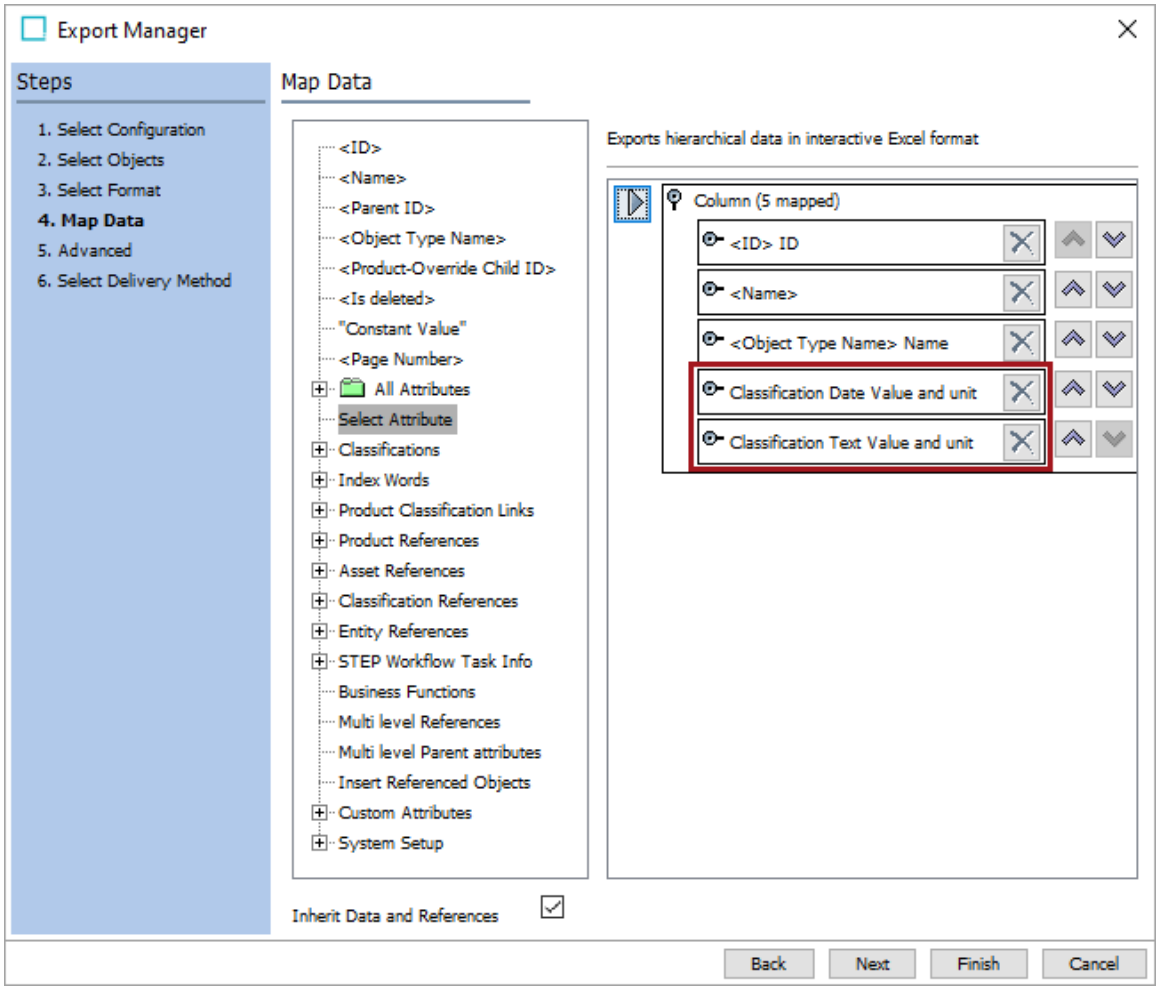


Note: If changes need to be made to the 'Classification roots for inherited attributes' parameter, this can be done by clicking the 'Select None' button.



- If there are classification(s) that have been selected, clicking on the 'Select None' button will clear the selection AND close the 'Add Classifications' dialog.
- If there are classifications that have been selected and you want to deselect the classifications but still remain in the 'Add Classifications' dialog to select other classifications, then you need to press 'Ctrl' + the previous selected classifications to deselect it.

3. Map the relevant attributes in Map Data wizard and complete the export process.



4. In the Smartsheet file the attribute **Classification Date** is exported and the attribute **Classification Text** is not included even if it is mapped in the Export Manager wizard.

Validate sheet			
Next error			
* <ID>	<Name>	<Object Type Name>	Classification Date
4568160	Gel Pen 01	Item	
4568161	Gel Pen 02	Item	
4568162	Gel Pen 03	Item	

5. To get the inherited attributes in the exported file the user have to select the higher hierarchy i.e., Classification Folder **Office Supplies**.

Validate sheet				
Next error				
* <ID>	<Name>	<Object Type Name>	Classification Text	Classification Date
4568160	Gel Pen 01	Item		
4568161	Gel Pen 02	Item		
4568162	Gel Pen 03	Item		

Note: When the products are linked to the classification without any parent folder, the inherited attributes will not be included in the export file if Smartsheet usage is selected as Template export (for onboarding new products).

For more information on configuring these functions, refer to the **Smartsheet Data and Template Configurations** topic of the documentation.

For more information on Classification roots for inherited attributes, refer to the **Classification Roots for Inherited Attributes** topic of the documentation.

Smartsheet and LOVs

Smartsheet incorporates LOVs, which allow the user to select values from a dropdown, thereby minimizing the number of mistakes. While there are a lot of benefits, there are a few limitations that need to be considered when performing data exchange with the Smartsheet format.

Limitations

The following is a list of limitations to consider when using LOVs in a Smartsheet:

- The error message 'Lookup values were not applied, please reopen workbook.' is caused by a client-side runtime error upon opening the workbook. In very rare cases, reopening the workbook resolves the issue. However, a re-export is usually required to successfully open the workbook.
- Smartsheet always includes LOVs and applies filters from attributes and attribute links but never filters classification references. Since Smartsheet requires the same (sub)set of attributes and LOV values for all products in a worksheet, it is not possible to support classification reference filtering.
- If the exported list of values is empty after applying filters, the following messages will be displayed in the background process log:
  - For hard LOV: 'Column [attribute id] has empty look up list. Column will read-only.'
  - For medium LOV: 'Column [attribute id] has empty look up list. Column will not facilitate lookup.'

  For more information on LOVs, refer to the **List of Values (LOVs)** topic in the **System Setup** documentation.

- It is recommended that the newest supportable version of Excel is used. For Excel 2007 and later, the maximum number of exported LOVs is 16,380. Applying a filter can reduce the number of LOVs, allowing for a successful export. After applying filters, if the number of LOVs exceeds the numbers referenced above, the following message will display in the background process report: 'Column [attribute id] exported without look-up because it has a LOV with [n] values. This exceeds the allowed max of [n-max] for this version of Excel.' The column will be editable, but will not facilitate any lookup (dropdown or popup selection).
- In cross-context Smartsheets, the following messages can appear due to attribute / LOV configurations:
  - 'Column [attribute id] marked read-only because its LOV ('[lov id]') is dimension dependent and does not meet the requirements for use in Smartsheet.'
  - 'Column [attribute id] marked read-only because both attribute and its LOV are dimension dependent.'

  For more information on cross-context exports for Smartsheet, refer to the **Smartsheet Data and Template Configurations** topic.

- The property Portal.Excel.UseLOVFilters in the sharedconfig.properties file no longer has an impact in Smartsheets. It was only for Quicksheets, which is deprecated.
- LOVs in Smartsheet exports may be empty due to a mismatch between the user's value privileges and LOV privileges.

Smartsheet Data and Template Configurations

Before a Smartsheet can be downloaded and edited, a Smartsheet data export or template configuration must be created in the Export Manager. Once created, the configuration(s) can be assigned to the relevant template widgets and export actions in the Supplier Web UI, or it can be selected when exporting a Smartsheet.

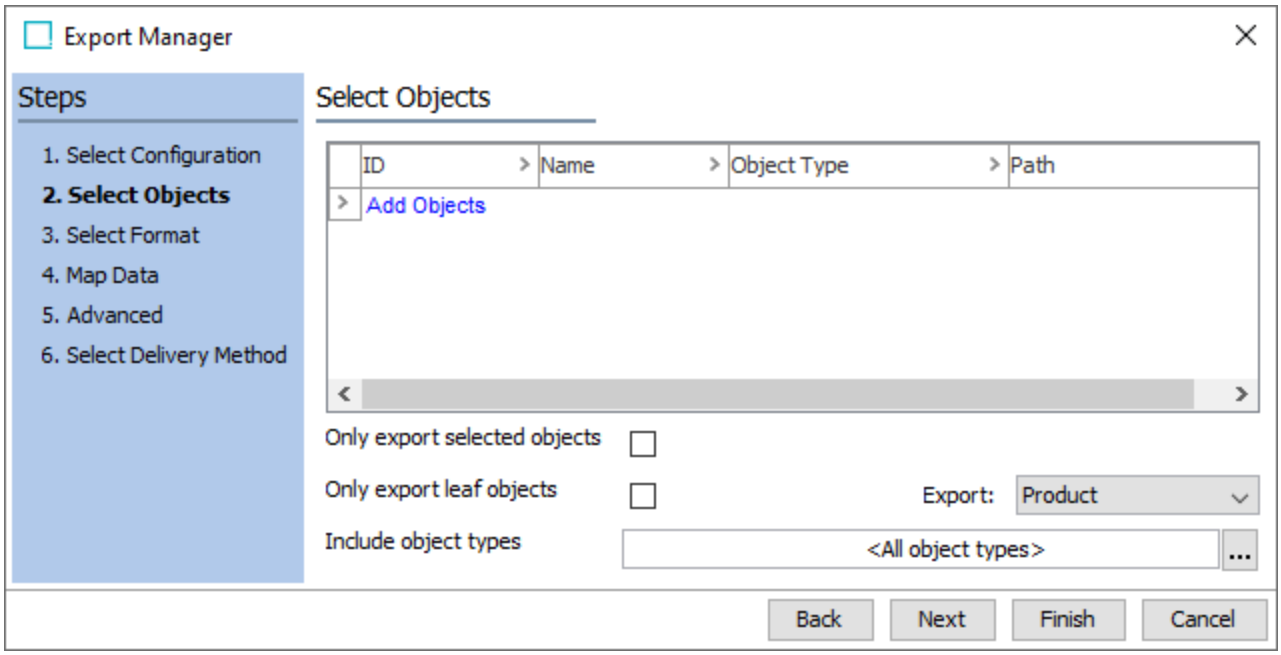
Important: Objects should **not** be exported to Excel Smartsheets from the workbench. Smartsheet **templates** exported from the workbench will not be able to correct supplier linking of products created during import and will also provide different browsing capabilities. A **maintenance** Smartsheet exported from the workbench will not contain any information about workflow and state, so it will not be able to supply information about workflow mandatory columns. In addition, any **business rules** regarding the current workflow will be unavailable during import. It will also supply a different set of possible targets when maintaining product cross-references.

To create a configuration, open the Export Manager and use the following sections to create an Excel Smartsheet data file or an Excel Smartsheet template. The following sections only describe the settings that are specific to Smartsheets. For detailed information about each of the steps in the Export Manager, Refer to the **Export Manager** documentation.

Important: While Smartsheets are designed to be used from the Supplier Web UI, they can also be used by non-supplier users. However, future Smartsheet enhancements will most likely depend on the Supplier Web UI.

Select Objects

Within the Export Manager, on the Select Objects step:

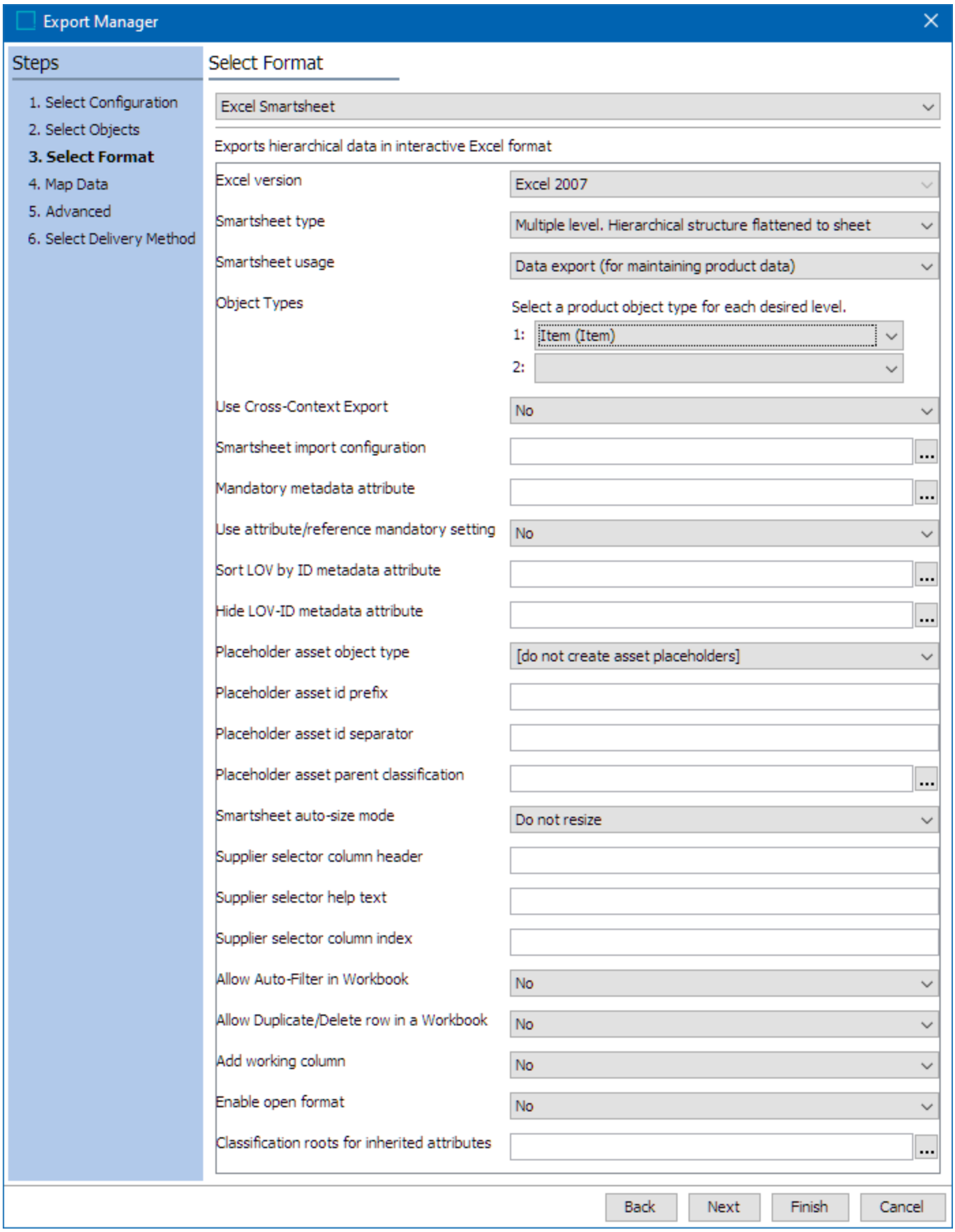


For both data export and template configurations, apply the following settings:

- If the export configuration is saved and used from the Supplier Web UI, the actual products selected in this step are irrelevant. The products will be replaced with the products / categories selected in the Web UI.
- Do not change the **Export** and **Include object types** default settings.
- For Smartsheet multilevel data exports, check **Only export selected products** and leave **Only export leaf objects** unchecked.
- For Smartsheet template export specifically, note the following:
  - It is recommended to enable **Only export selected products** so that the Export Manager does not have to examine the child products of the selected template product.
  - The selected objects are not exported in Smartsheet templates. However, the first product in the list or selection will be the parent of the top-level products created when the edited template is imported.
  - Enabling **Only export leaf objects** is not recommend since the import may fail, or new products may be created in the wrong place when the resulting template is later used via the saved configuration.
- Click the Next button to display the next step.

Select Format

On this step, select **Excel Smartsheet** as the file export format. Complete the parameters displayed as follows:


Export Manager
Steps
1. Select Configuration
2. Select Objects
3. Select Format
4. Map Data
5. Advanced
6. Select Delivery Method
Select Format
Excel Smartsheet
Exports hierarchical data in interactive Excel format
Excel version
Excel 2007
Smartsheet type
Multiple level. Hierarchical structure flattened to sheet
Smartsheet usage
Data export (for maintaining product data)
Object Types
Select a product object type for each desired level.
1:
Item (Item)
2:
Use Cross-Context Export
No
Smartsheet import configuration
Mandatory metadata attribute
Use attribute/reference mandatory setting
No
Sort LOV by ID metadata attribute
Hide LOV-ID metadata attribute
Placeholder asset object type
[do not create asset placeholders]
Placeholder asset id prefix
Placeholder asset id separator
Placeholder asset parent classification
Smartsheet auto-size mode
Do not resize
Supplier selector column header
Supplier selector help text
Supplier selector column index
Allow Auto-Filter in Workbook
No
Allow Duplicate/Delete row in a Workbook
No
Add working column
No
Enable open format
No
Classification roots for inherited attributes
Back
Next
Finish
Cancel

1. In the mandatory **Excel version** parameter, use the dropdown to select Excel 2007. Using an older version of Excel could cause problems with the files, including errors upon import. It is best to always choose the newest, supported version of Excel.
2. In the mandatory **Smartsheet type** parameter, select one of the following options from the dropdown:
   - Select **Multiple level. Hierarchical structure flattened to sheet** if you want to work with multi-level Smartsheets. This is used to initiate and maintain products of different object types and at different hierarchy locations in STEP.
   - Select **Multiple object types. Different product types intermixed** if you want to work with multi-object type Smartsheets. This handles multiple object types simultaneously, without them needing to be hierarchically related. This functionality works best when there is a large overlap between the set of attributes that are valid for each of the object types.

   For more information on Smartsheet types, refer to the **Using a Smartsheet** section of the documentation.
3. In the mandatory **Smartsheet usage** parameter, select one of the following options from the dropdown:
   - Select **Data export (for maintaining product data)** to export data for products already in the STEP database.
   - Select **Template export (for onboarding new products)** to export an empty Smartsheet with formatted columns that can be used to initiate new products.

   Note: Template exports cannot export reference values but reference types can be exported. It will not contain any data for the object type.
4. In the mandatory **Object Types** parameter, select one or more object types to be included in the export from the dropdown.
   - For multi-level Smartsheets, specify the object type of the products for each level in the multi-level export. The object type must be valid for each level. The top level must be allowed under the root product selected in the Select Object step of the wizard. A maximum of five (5) entries are allowed.
   - For multi-object type Smartsheets, specify the object type(s) to export. There is no limit on how many entries are allowed.
5. In the mandatory **Use Cross-Context Export** parameter, use the dropdown to determine if language dependent attributes should be exported in the language of the selected context.

   Set to **Yes** to specify which contexts should be included in the export. A column is created in the Smartsheet for each selected context on all language-dependent attributes. Remember to select the export context specified in the Advanced step of the wizard in the cross-context list. If the export context is not selected, there is a risk of exporting data in an incorrect context.

   To add a context, click the **Select Context** link under the dropdown and choose the context(s) from the window that appears.

   Important: References and links cannot be exported to a Smartsheet using this feature.
6. In the potentially mandatory **Smartsheet import configuration**, select an existing import configuration to apply to the Smartsheet. The import configuration may run business rules and other actions on the products

upon import. This field is only mandatory if an import configuration has not been set on the 'SmartSheetImportAction' in the Supplier Web UI.

For more information of setting up the import configuration, refer to the **Smartsheet Import Configurations** section of the documentation.

7. In the optional **Mandatory metadata attribute** parameter, click the ellipsis button (...) and select an attribute to mark columns as mandatory in the Smartsheet. This attribute must be a Description attribute and have the validation base type Integer or Numeric. Once created, the attribute must be added as metadata to all relevant attributes or attribute links. For any attribute that *should* be made mandatory in the Smartsheet, a non-zero integer must be populated in the metadata attribute.

   This parameter can be used in conjunction with the following parameter that respects standard mandatory settings.

   For more information, refer to the **Mandatory Attributes and References in Smartsheets** topic.

8. In the optional **Use attribute/reference mandatory setting** parameter, select Yes to mark columns as mandatory in the Smartsheet based on standard mandatory settings. Leave the option set to No if mandatory attributes and references should be identified via other options, like via mandatory definition in a workflow (most common way).

   This parameter can be used in conjunction with the previous parameter that uses a metadata attribute for setting mandatory status or derived from workflows or the 'map data' step of the export wizard.

   For more information, refer to the **Mandatory Attributes and References in Smartsheets** topic.

9. In the optional **Sort LOV by ID metadata attribute** parameter, click the ellipsis button (...) and select a metadata attribute to sort attribute columns that use LOVs by value ID. This attribute must be a Description attribute and have the validation base type Integer or Numeric. Once created, this attribute must be added as metadata to all relevant attributes. For any LOV that *should* have its values sorted by ID, a non-zero integer must be populated in the metadata attribute.

   This configuration changes the order in dropdown lists and pop-up LOV selectors in the Smartsheet column (s).

   The metadata value is ignored if the attribute is not LOV based or if the LOV does not use value ID. This makes it possible to have the same values for two different columns sorted both by value and value ID.

   For more information, refer to the **Adding IDs to Values in LOV** section of the **System Setup** documentation.

10. In the optional **Hide LOV-ID metadata attribute** parameter, click the ellipsis button (...) and select a metadata attribute to exclude external IDs from individual LOV-based Smartsheet columns. This is useful if you only want to display the value of the LOVs rather than their value IDs. This attribute must be a Description attribute and have the validation base type Integer or Numeric. Once created, this attribute must be added as metadata to all relevant attributes. For any LOV that *should not* display value IDs, a non-zero integer must be populated in the metadata attribute.
11. In the mandatory **Placeholder asset object type** parameter, use the dropdown and select the object type of the assets that are created and referenced during import. This must be specified to enable asset placeholders. The available options are system dependent.

Note: Placeholder assets make it possible for the created assets to have a determinable ID so other processes are able to import and maintain actual asset content.

If **[do not create asset placeholders]** is selected, the asset placeholder system is disabled and normal asset references are maintained in the Smartsheet.

12. In the optional **Placeholder asset ID prefix** parameter, enter a prefix for the ID placeholder assets created upon import. This ensures that asset IDs are unique across all suppliers during export from Workbench. The prefix will be changed during Web UI export.
13. In the optional **Placeholder asset ID separator** parameter, enter a separator to be inserted in the final ID between the prefix and the file name.

    For example, if a member of the user group 'Vendor ltd.' enters the asset file name 'Blue Running Shoe.png', the ID of the final placeholder asset is 'Vendor ltd.:Blue Running Shoe.png', with ':' being the separator. This ID is, with high probability, unique between vendors. If the ID contains invalid characters or exceeds the 40 characters limit, the name will fail validation.

14. In the optional **Placeholder asset parent classification** parameter, click the ellipsis button (...) and specify where placeholder assets are created when the Smartsheet is re-imported. This parameter may be useful if the imported Smartsheet was exported from STEP Workbench (and is therefore not associated with a Supplier group). If this parameter is filled in and the Smartsheet is exported / imported from the Web UI, then this parameter is overwritten and the asset placeholder will go below the correct Supplier Asset folder in the Supplier Hierarchy.
15. In the mandatory **Smartsheet auto-size mode** parameter, use the dropdown to determine whether to resize the exported Smartsheets by column-header width or by cell-content width using word wrap, or not to resize at all.
16. In the optional **Supplier selector column header** parameter, enter a name for the supplier selector column.

    Note: This parameter, as well as the 'Supplier select help text' and 'Supplier select column index' below, are only available when creating a Template export and when 'Enable all-view for users that are a member of multiple suppliers' is set to 'Y' in system properties (Users & Groups node).

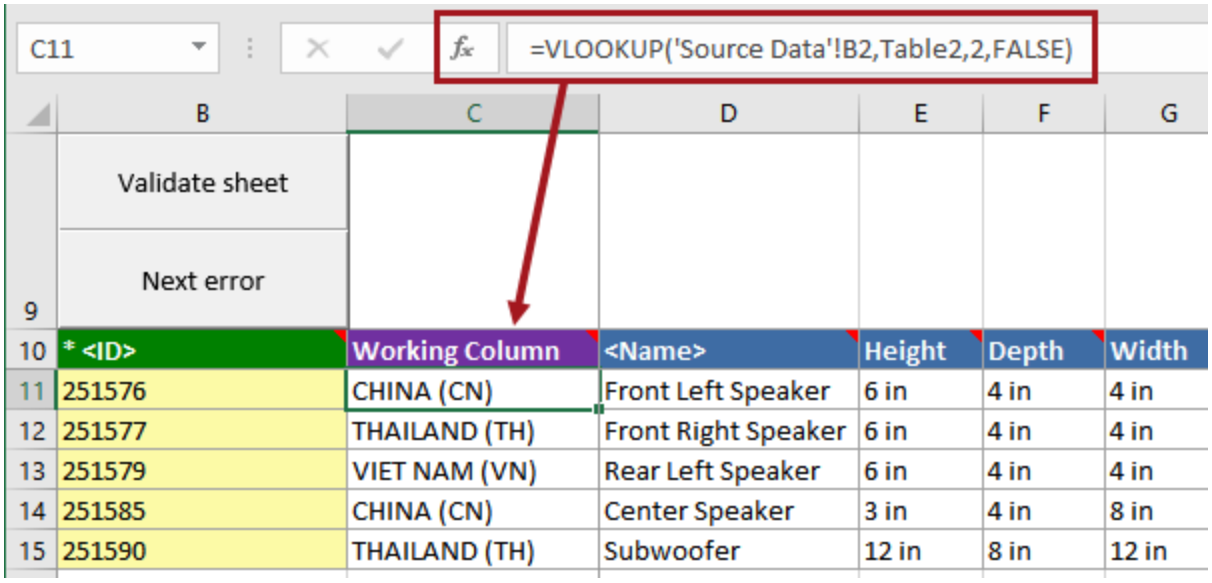
    Unless this parameter is set, the supplier selector column is called 'Supplier' by default.

17. In the optional **Supplier selector help text** parameter, enter what the tooltip says when the cursor is placed over the supplier selector column header in the exported sheet.
18. In the optional **Supplier selector column index** parameter, determine which column the supplier selector occupies. If the parameter is not set, then the column will be placed as far to the left as possible. If this parameter is set, all columns mapped in the Map Data step will be bumped to the right of whichever column is specified. Valid values are between 2 and 999.
19. For the mandatory **Allow Auto-Filter in Workbook** parameter, determine whether or not rows can be filtered in the Smartsheet via the dropdown.
20. For the mandatory **Allow Duplicate / Delete row in a Workbook** parameter, determine whether or not entire rows can be duplicated and deleted in the Smartsheet via the dropdown.

21. For the **Add working column** parameter, select 'Yes' to add a working column to the Smartsheet. The working column allows users to enter Excel formulas, which are restricted from other data cells in the Smartsheet. A common Excel function used within the working column is VLOOKUP.

    Data that is generated in cells within the working column can be copied and pasted into other cells in the Smartsheet by using the **Values** option within the Excel 'Paste Special' feature.

> **Note:** If a standard paste operation is performed to paste values generated by the formulas in the working column, or if an option other than 'Values' is selected when using 'Paste Special,' the cells will be skipped on import and no values will be imported.



	B	C	D	E	F	G
9						
10	* <ID>	Working Column	<Name>	Height	Depth	Width
11	251576	CHINA (CN)	Front Left Speaker	6 in	4 in	4 in
12	251577	THAILAND (TH)	Front Right Speaker	6 in	4 in	4 in
13	251579	VIET NAM (VN)	Rear Left Speaker	6 in	4 in	4 in
14	251585	CHINA (CN)	Center Speaker	3 in	4 in	8 in
15	251590	THAILAND (TH)	Subwoofer	12 in	8 in	12 in

    The working column will appear in the second column of the exported Smartsheet (column C). It will be frozen, and any values it contains will be ignored during both product validation and data imports.

> **Note:** If the **Supplier selector column index** parameter is set to **2**, the working column will still appear in the second column of the Smartsheet. The Supplier column will instead be bumped to the third column (column D).

22. For the **Enable open format** parameter, select 'Yes' to export an open format Smartsheet.

    In open format Smartsheets, almost all cells are unlocked ('open'), providing users with the option to employ Excel formulas, number formatting (e.g., General, Currency, Date), text formatting (e.g., bold, italic), text wrapping, and cell color (background shading). For full details on this functionality, refer to the **Open Format Spreadsheet** topic in this guide.

23. For the **Classification roots for inherited attributes** parameter, the Export Manager determines the valid attributes for new products in the product category if a Product Classification Link type allows for inheritance of attribute validity from the linked classification. These attributes will be included in the Smartsheet output. For full details on this functionality, refer to the **Classification roots for inherited attributes** topic in this guide.

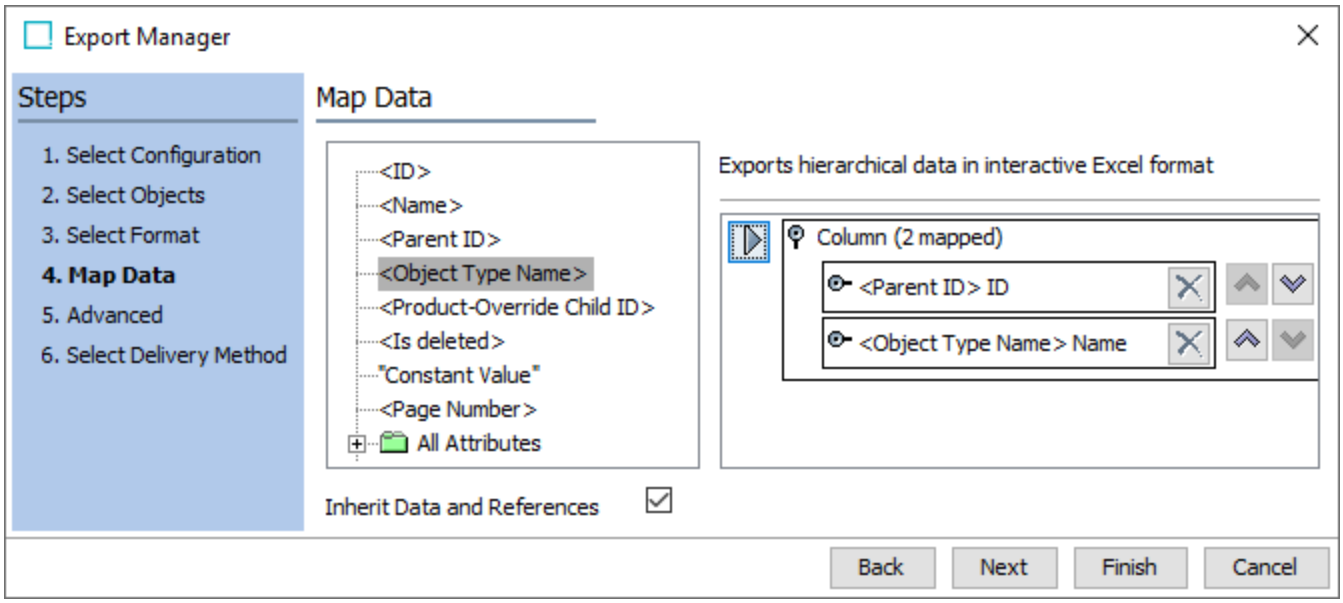
Map Data

On this step, mapping options are available depending on whether you are making a multi-object type or multi-level Smartsheet.

Special options for multi-object type Smartsheets

- **Parent ID** is used as a Parent ID selector in the Excel sheet so users can indicate the correct parent IDs for the products they are working with.
- **Object Type Name** Smartsheet column is used as an Object Type selector in the Excel sheet so users can indicate the object types for the products they are working with.

These two columns can be mapped in multi-level Smartsheet as well, but cannot be manually edited.



Special options for multi-level Smartsheets

- **Start level** specifies that the selected column starts a new object level. The top level can be left empty because it is implicit, however a value must be provided for all other Start levels below the first. The actual value is not important. To add a value, click the transformation button ( ) and enter a value in the window that appears.

  The Start level columns are also used for grouping products at each level. For example, all rows with 'T-Shirt' in the **Name** column (which is designated as a Start level column) become one product. All rows that have both 'Pink' in the **Color** column and 'T-Shirt' in the **Name** column become one SKU Level product under the new T-Shirt product.

  It is not recommended to group by multivalued attributes and reference columns.

- **Mandatory** makes the column appear mandatory in the downloaded Smartsheet. If the Smartsheet is validated in Excel, cells that are empty but marked as mandatory will appear as invalid. However, no errors

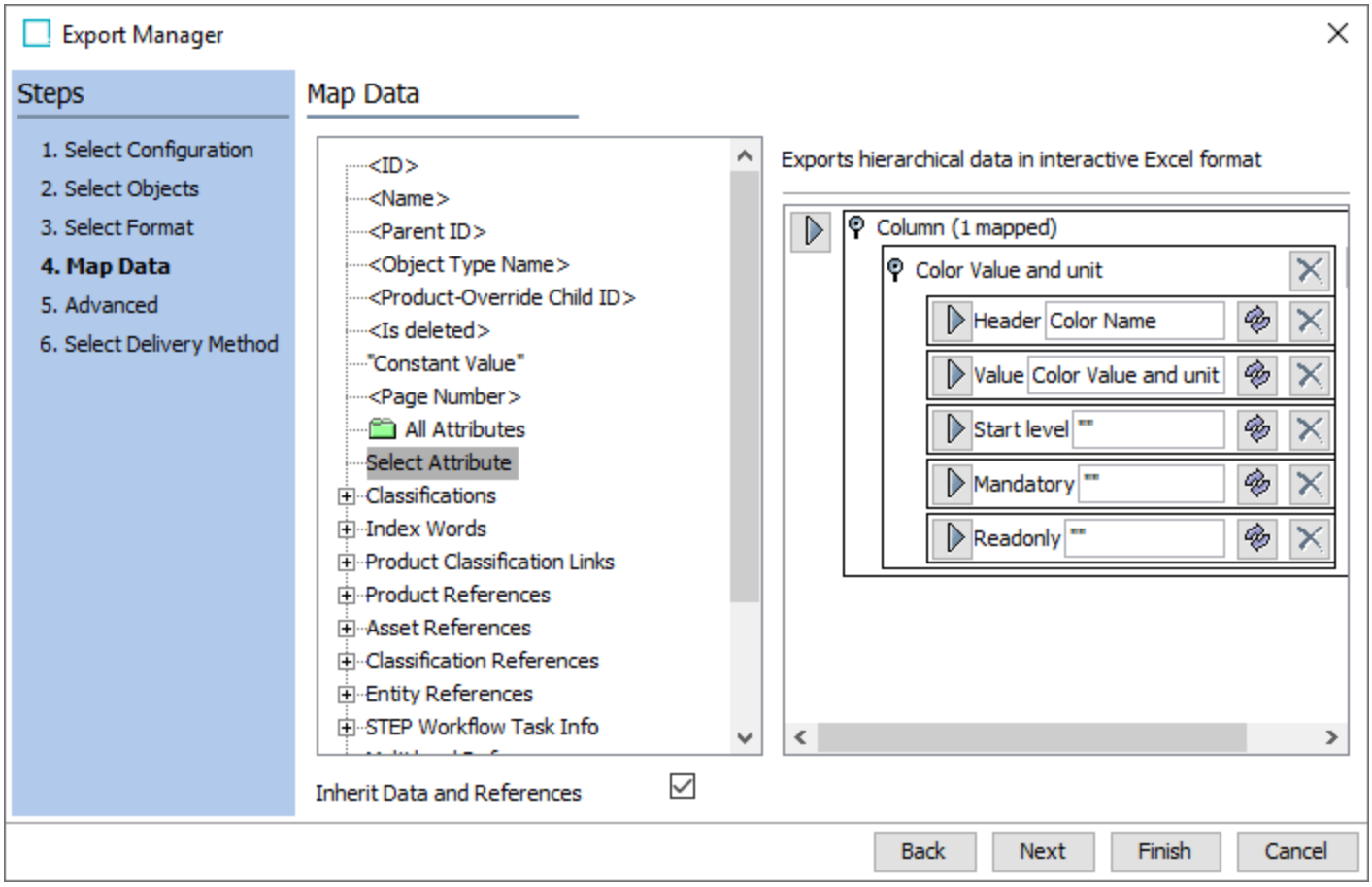
are generated when the Smartsheet is imported. To mark a column as mandatory, click the transformation button ( ) and enter any value in the window that appears.

For more information, refer to the **Mandatory Attributes** section of the **System Setup** documentation.

- **Read only** makes the column read-only and locks all cells in the column in Excel. This setting is only relevant for data export. In template export, mappings marked as read-only are omitted.

  To mark a column as read only, click the transformation button ( ) and enter any value in the Transformation dialog. Click the **Save** button on the Transformation dialog and the 'Readonly' mapped value is updated.

The following image shows that none of the special options have a transformation.



Important: Do not add any kind of transformation to the value of a mapping unless it is marked as read-only. Transformations are applied during Smartsheet data export. However, during Smartsheet import, the column will be handled as if it contains Value and Unit. This can lead to import errors or unexpected imported values.

Important: Do not apply aspects to any transformed values. Smartsheets do not contain this information and will treat the value in the cell according to the step type for the column.

Additional Information About Mapping

If attribute groups are mapped at this step, the individual attributes are ordered using the **Display Sequence** metadata for the attributes. If an attribute defines or starts a level in the Smartsheet, the first column in the group is used for sorting at this level.

Note: Attribute groups are not mapped recursively.

Sub-level description attributes cannot be exported to a Smartsheet unless a product of the sub-level object type is included in the export. This is usually not the case in template exports where it is recommended to select the **Export only selected products** option.

When Smartsheets are exported, LOV values are embedded in the Excel workbook so that a list of valid values is available when values are specified in an LOV-based column. The exported values are filtered according to the filters specified on the attribute and on the attribute link in the Product hierarchy.

Note: Smartsheets only support LOV filters on attributes and attribute links.

Advanced

On this step, do not select **Locale conversion from context**. It can break the automatic conversion to the client's native decimal separator and can cause the import of the Smartsheet to fail.

Select Delivery Method

On this step, if configured, a Smartsheet background process can be specified on this step. For more information, refer to the **Using Smartsheet Export Background Process** section of the **Smartsheet Background Processes and Queues** topic of this documentation.

Saving and Modifying the Configuration

For this dialog, saving an export configuration is described in the **Running a Data Export** topic of the **Export Manager** documentation.

Once saved, an export configuration can be edited, as described in the **Maintaining a Saved Export Configuration** topic of the **Export Manager** documentation.

Smartsheet Import Configurations

In order to import Smartsheets into STEP an import configuration must be available. This import configuration can be specified upon export of the product data / template (via the Smartsheet export configuration) or upon import (via the 'SmartsheetImportAction' configuration in Web UI).

For more information on configuring the Web UI import action, refer to the **Smartsheet Import Action** section of the **Web User Interfaces** documentation.

Smartsheets can be imported via STEP Workbench Import Manager, however it is recommended to use the Smartsheet Import Action in the Supplier Web UI.

Note: While an import configuration is required, the vast majority of a Smartsheet's functionality is determined by the product data / template export configuration. For more information, refer to the **Smartsheet Data and Template Configurations** section of the documentation.

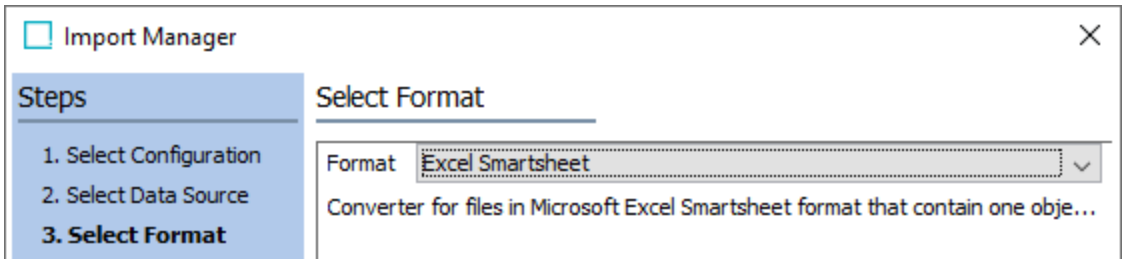
Creating an Import Configuration

To create a configuration, go through all the steps of the Import Manager wizard. The following section only describes the settings that are specific to Smartsheets.

For more information on the Import Manager, refer to the **Import Manager** section of the documentation.

Select Format Step

In the **Format** parameter, ensure 'Excel Smartsheet' is selected.



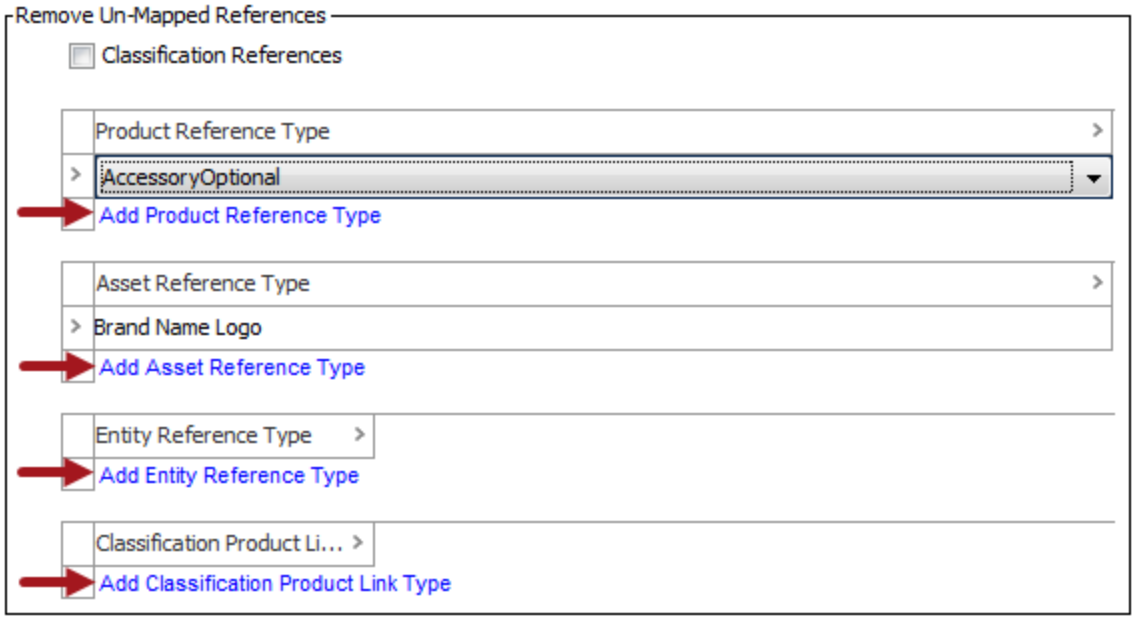
Important: If users encounter an error during the import process, it could be due to the template being used is set to an older version of Excel. It is recommended that users use the newest, supportable version of Excel. For more information on supported versions of Excel, refer to the current **Platform and Software Support** documentation in the **System Release and Patch Notes**.

Advanced Settings Step

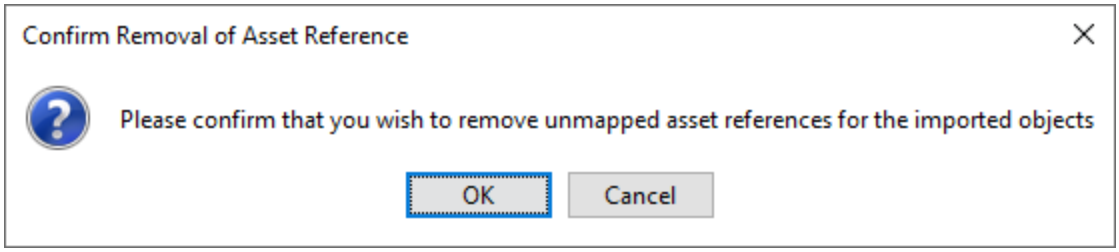
If desired, it is possible to delete / replace references when importing a Smartsheet.

References can be cleared or replaced by selecting them during the final step in Import Manager:

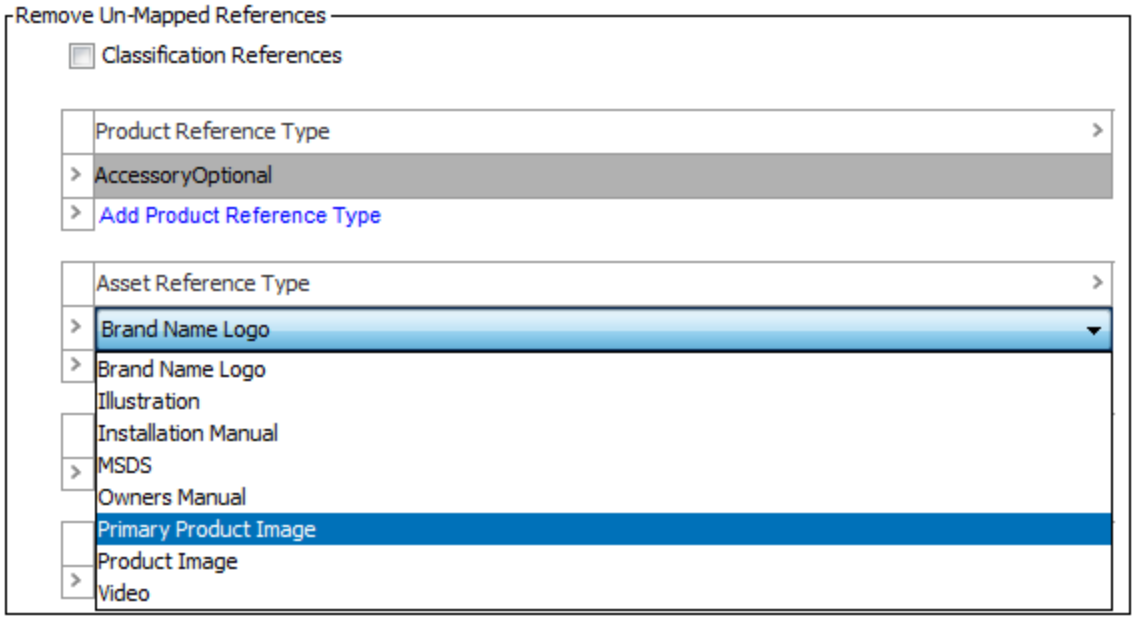
1. In the **Remove Un-Mapped References** parameter, click the relevant 'Add Reference Type' option.



2. A warning message will appear. Click **OK**.



3. Click to open the dropdown list and select the appropriate reference type.



In the above example, the asset reference 'Primary Product Image' will be replaced by any new references included in the Smartsheet import. If the reference field is blank, the import will clear the reference from the corresponding product. Only Primary Product Image references that are included in the input file will be present on the products included in the file.

If configured, a Smartsheet background process can also be specified on this step. For more information, refer to the **Smartsheet Background Processes and Queues** section of the documentation.

Saving and Modifying the Configuration

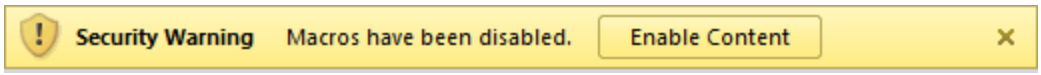
Saving an import configuration is described in the **Running a Data Import** topic of the **Import Manager** documentation.

Once saved, an import configuration can be edited, as described in the **Maintaining a Saved Import Configuration** topic of the **Import Manager** documentation.

Using a Smartsheet

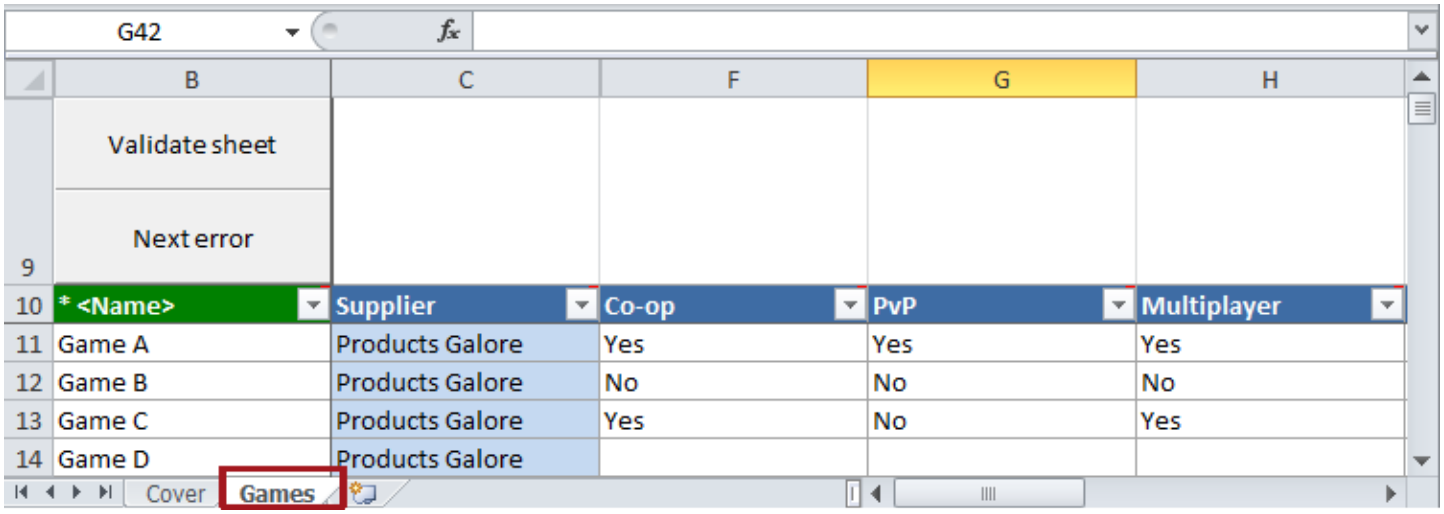
Unlike typical Excel sheets, Smartsheets are very restrictive and must be handled in a very specific way.

1. In **Excel**, open the applicable Smartsheet. If the Smartsheet has never been opened before it will open on the **Cover** tab, which contains information about the download date and the color codes used in the Smartsheet. Additionally, a security warning will appear at the top of the cover page.



   Click **Enable Content** to enable macros, otherwise, the Smartsheet will not work correctly.

2. In the workbook, one or more additional tabs are available, each corresponding to the categories selected during the export process. Click the desired tab to open the sheet.



Important: Objects are broken into tabs based on the lowest level of the selected hierarchy in which common attributes are linked. Typically, this means that they broken up according to each individual object hierarchy selected in the export configuration. However, multiple additional tabs will be created for the same hierarchy if the exported attributes do not all link to the lowest level of that hierarchy.

3. Enter the data for the objects to be imported.

Note: In template sheets, all rows are editable, but any unmapped columns are locked to the user. In maintenance sheets, all unmapped rows and columns are locked.

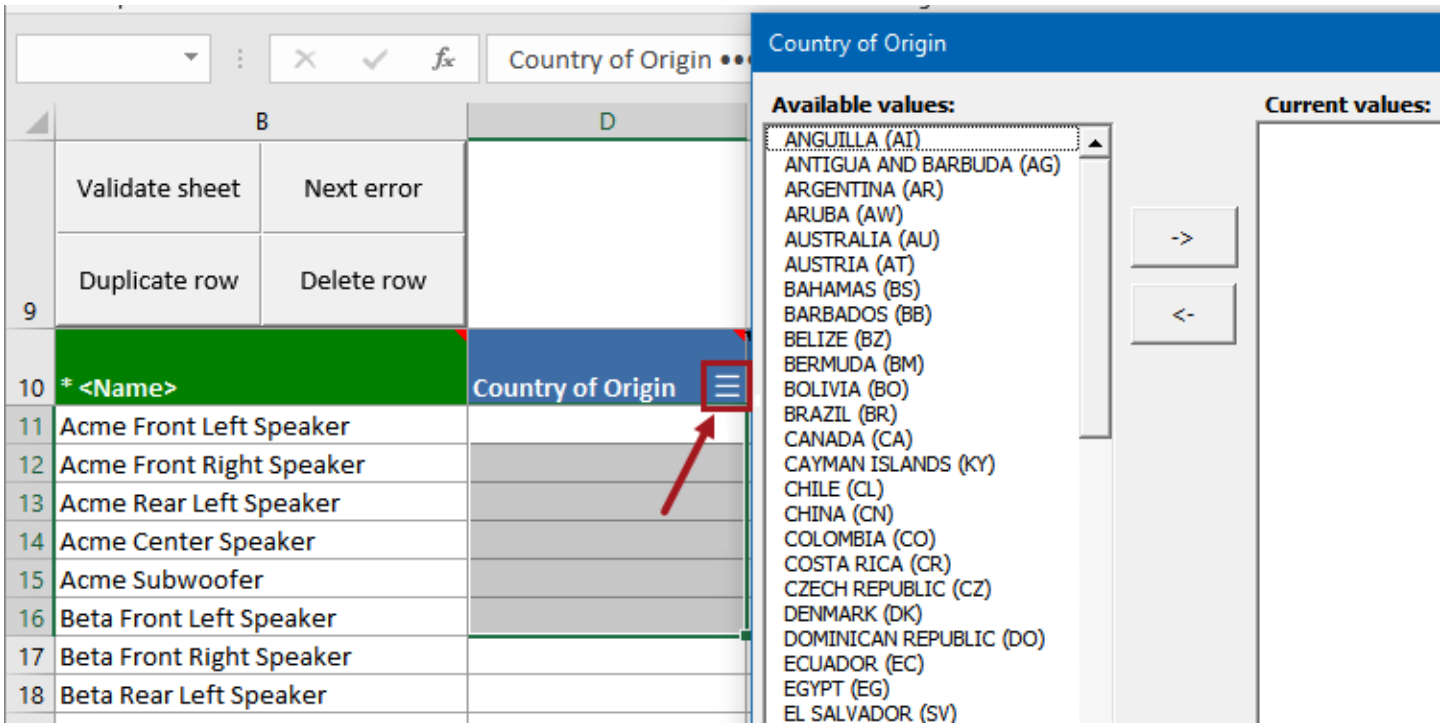
Entering Values for Multi-Valued LOVs and References

To enter values from multi-valued LOVs, multi-product reference types, multi-asset reference types, and parent IDs:

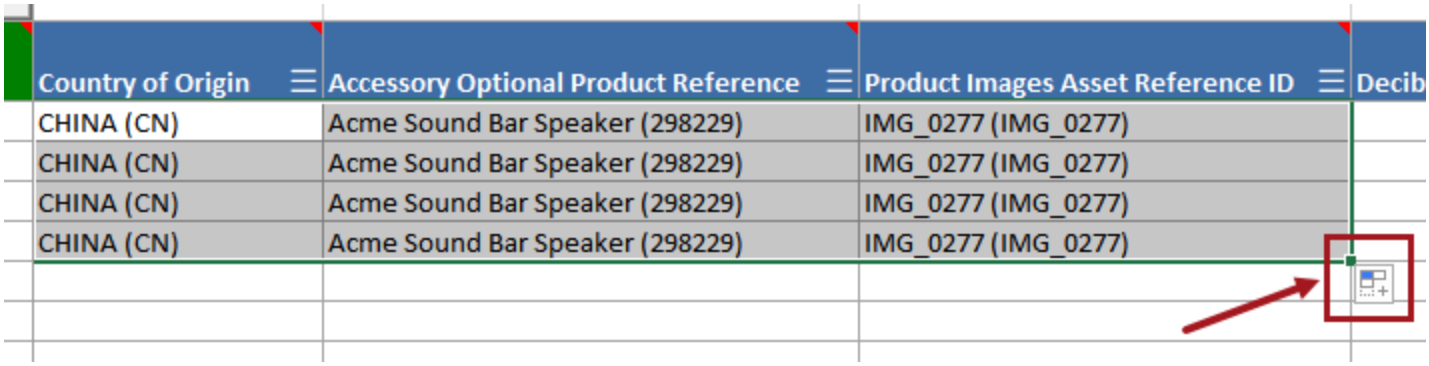
1. Select one or more cells in the column in which you want to add data. Both adjacent and non-adjacent cells within the same column can be selected. Hold the Ctrl key while clicking the relevant cells to select non-adjacent cells.

   Note: If entering data in multiple cells at once in a Smartsheet with a Supplier column, you must ensure that you have the same supplier selected in each row where you are adding data. Otherwise, the edit will be rejected.

2. Click on the column header, which contains a collapsed menu icon ('hamburger button')
3. The dialog from which multiple values or parent IDs are chosen is displayed.
4. Make the required selections, then close the multi-value dialog. All selected cells will be populated at once.



Like other values in Smartsheets, values populated from multi-valued LOVs, multi-product reference types, multi-asset reference types, and parent IDs can also be dragged down in adjacent columns using the Excel fill handle, as shown in the below screenshot. For more information on LOVs, refer to the **Smartsheet and LOVs** topic in the **Excel Smartsheet Format**.



Country of Origin	Accessory Optional Product Reference	Product Images Asset Reference ID	Decib
CHINA (CN)	Acme Sound Bar Speaker (298229)	IMG_0277 (IMG_0277)	
CHINA (CN)	Acme Sound Bar Speaker (298229)	IMG_0277 (IMG_0277)	
CHINA (CN)	Acme Sound Bar Speaker (298229)	IMG_0277 (IMG_0277)	
CHINA (CN)	Acme Sound Bar Speaker (298229)	IMG_0277 (IMG_0277)	

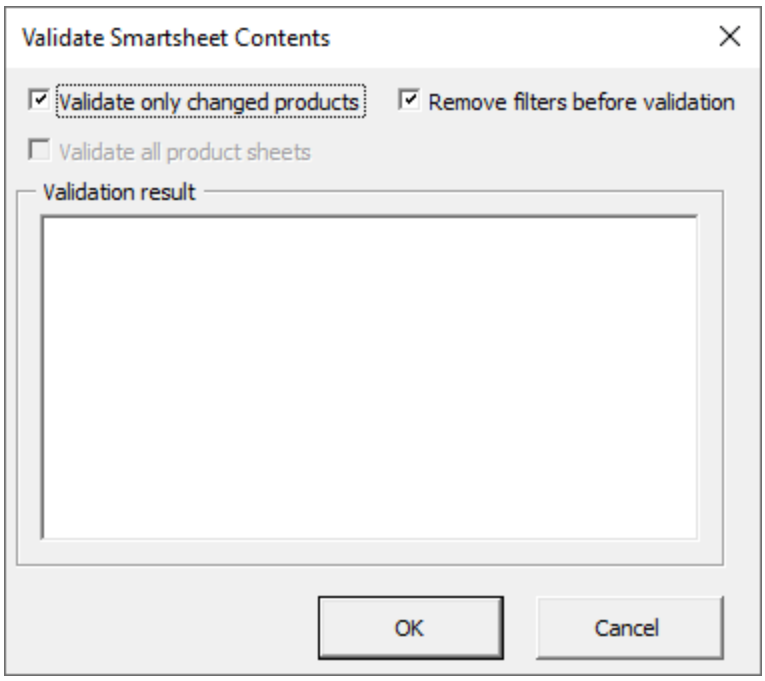
Validating the Sheet

To ensure that values have entered properly, the Smartsheet must be validated. Running a validation operation on a Smartsheet checks that the data on the sheet has been entered completely and correctly. If the data contains errors, the validation check will return error message(s) and/or shade cells red to indicate where data needs to be corrected.

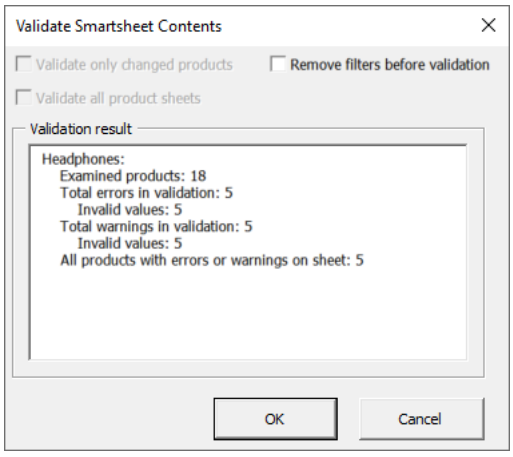
Also, the Data Issues Report bind allows users to write JavaScript-based business conditions that, when run, will insert custom messages related to attribute and reference issues into Smartsheets. Upon validation, the messages are displayed when clicking on / hovering over the cells. Warnings are identified via orange-highlighted cells, and errors via red-highlighted cells. If both an error and a warning are found in the same row, the error will supersede the warning and the first cell in the row will display with a red highlight.

To validate a Smartsheet:

1. Click **Validate sheet** in the top left corner. In the 'Validate Smartsheet Contents' dialog, three check boxes are available:
   - **Validate only changed products** - The Smartsheet will only validate products that have been changed. Leaving this box unchecked means all products are validated.
   - **Validate all product sheets** - The Smartsheet will validate all sheets in the workbook. Leaving this box unchecked means only products that appear on the currently selected sheet will be validated. This checkbox is only available if multiple sheets exist in the workbook.
   - **Remove filters before validation** - The Smartsheet will remove all filters from the sheet(s) during validation. Leaving this box unchecked means that any rows currently being filtered will not be validated. Additionally, if the user hides any rows they will not be considered when troubleshooting errors via the **Next Error** button.



2. Click **OK**.
3. In the **Login** dialog, enter the relevant STEP user name and password and click **Login**.
4. After the validation is run the **Validate Smartsheet Contents** window will display if the validation succeeded or if there are errors or warnings that need to be corrected.



Errors and warnings can be located and manually corrected via the **Next Error** button.

- Errors for mandatory fields are shown in the cell that is missing data.
- Errors and warnings from the business condition are shown are shown at the cell-level. Messages (configured or otherwise) are displayed when clicking on / hovering over the cells. Warnings are identified via orange-highlighted cells, and errors via red-highlighted cells. If both an error and a warning are found in the same row, the error will supersede the warning and the first cell in the row will display with a red

highlight. (For more info about configured messages, so the Data Issues Reports Bind topic in the Resource Materials documentation.)

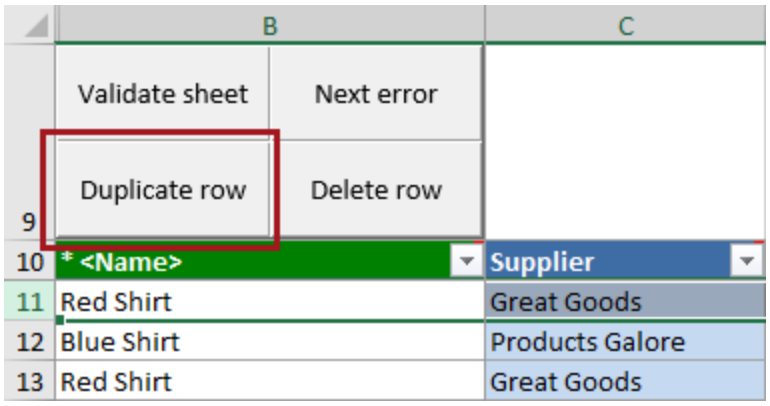
5. After resolving the errors, save the file.

Note: Some fields, such as node pickers, will prompt the user for a STEP user name and password. The system will only prompt for these credentials once.

Copying Data in a Smartsheet Template

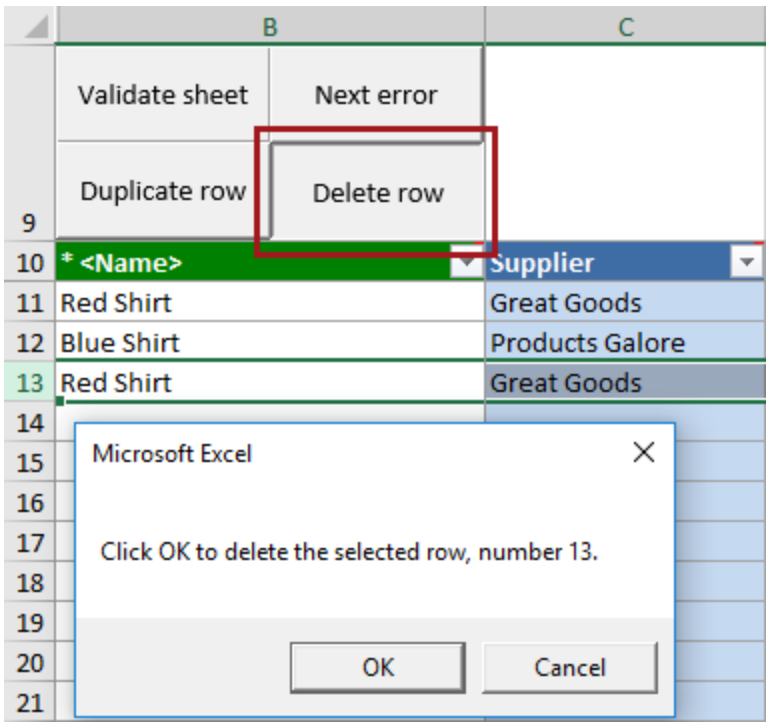
When initiating products using a Smartsheet template, users can duplicate data on a row-by-row basis, potentially saving them a lot of time. Note, however, that rows can only be copied in Smartsheet templates, and may need to be configured to do so.

1. To copy a row, select it and click the **Duplicate row** button.



2. A copy of the row will appear in the next open row.

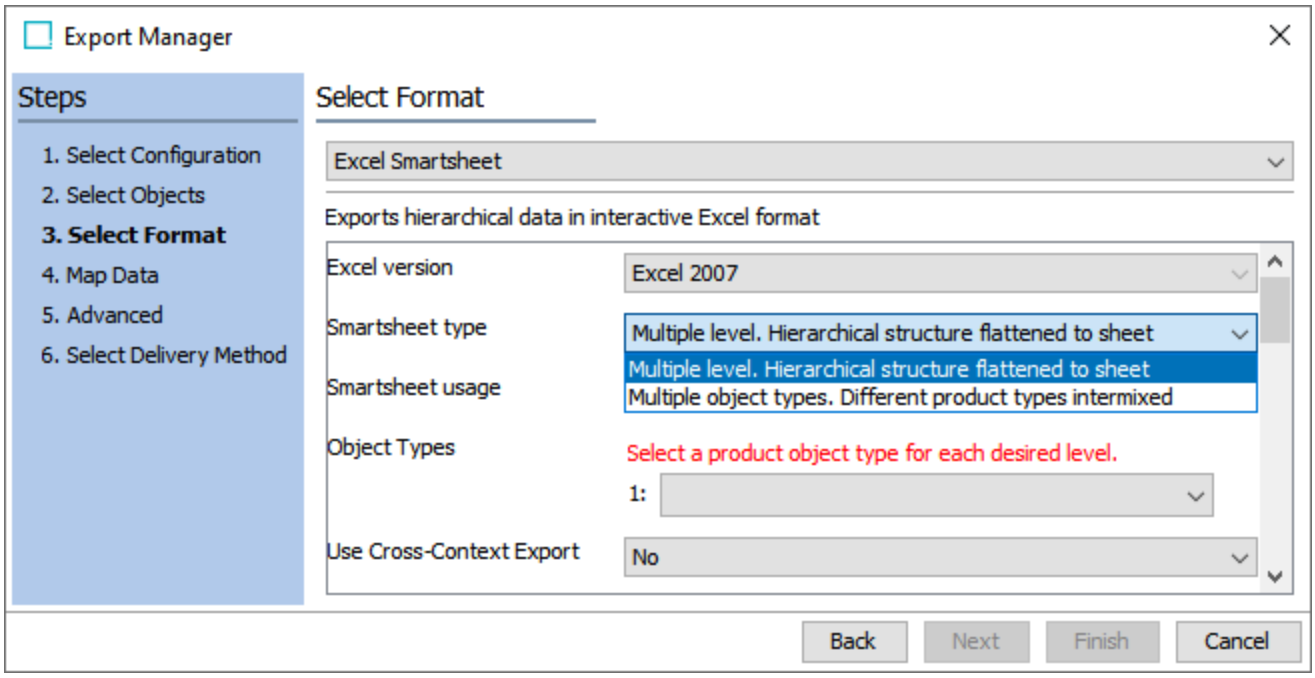
   This action cannot be reversed using the 'Undo' function built into Excel. Instead, to remove an unneeded row, select the row and click **Delete** row. When asked if you want to delete the selected row, click **OK** to delete it.



For more information on configuring these functions, refer to the **Smartsheet Data and Template Configurations** section of the documentation.

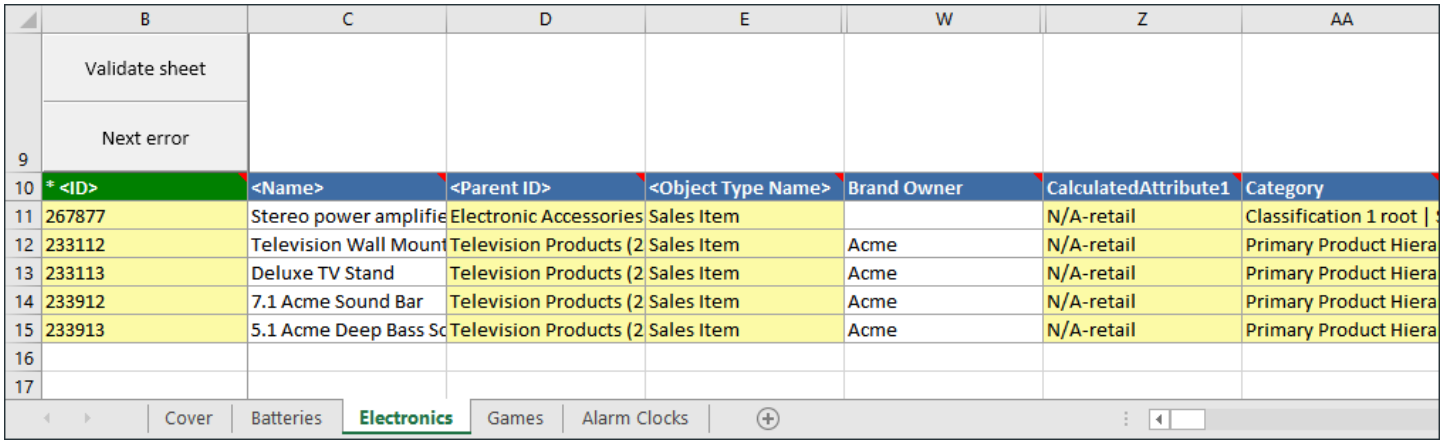
Smartsheet Types

When creating a Smartsheet export, users can choose between two Smartsheet types: Multiple level and Multiple object types.



Multi-level Smartsheets

In some cases, it is necessary to use Excel Smartsheets to initiate and maintain products of different object types and at different hierarchy locations in STEP. This is useful for maintaining / initiating product hierarchies, and displays the hierarchy in a flattened structure for easy editing.

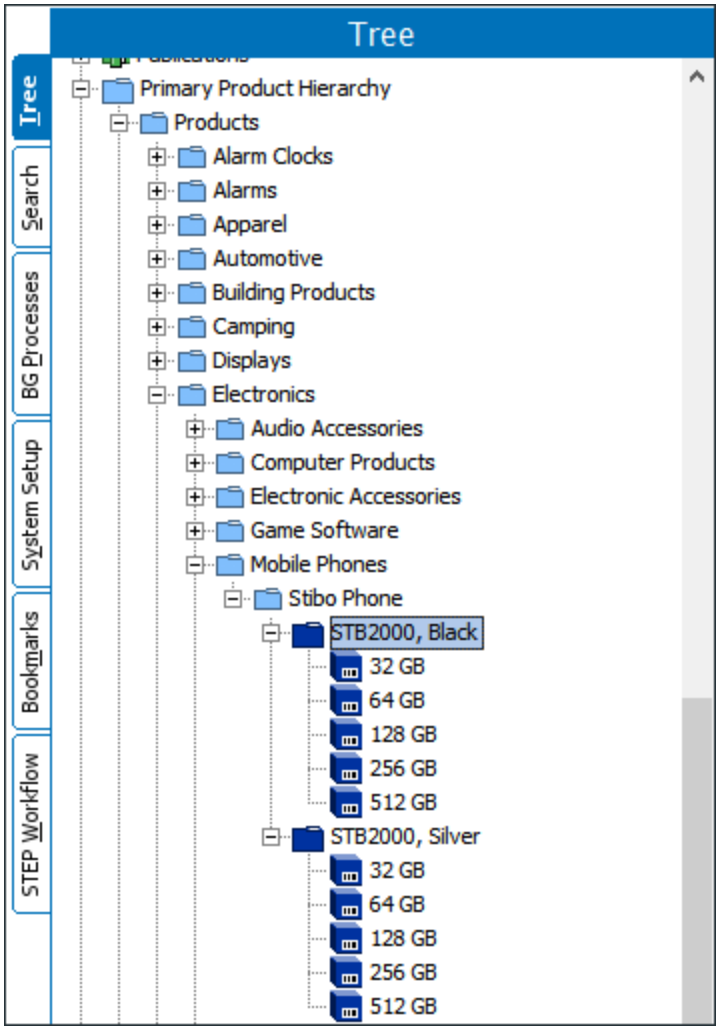


Multi-level sheets enable product data onboarding and maintenance by supporting multi-level product variant data and multiple languages. Template sheets offer easy bulk onboarding of multi-level product variants. These, along with maintenance sheets, enable suppliers to maintain data about the suppliers' own products or to onboard new products.

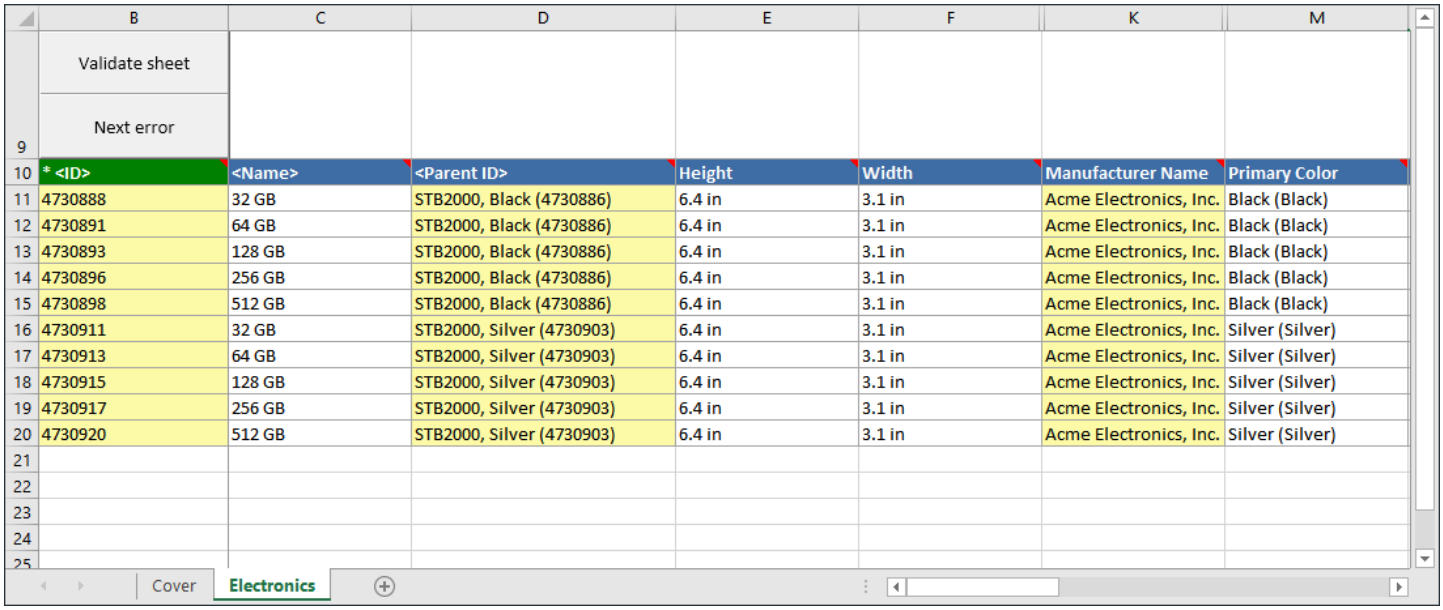
For each product category selected for download, a sheet will be added to the Excel workbook. The sheets are based on where attributes are linked within the product hierarchy. For more information on attribute links, refer to the **Attribute Links** section of the **Attributes** documentation in **System Setup** documentation.

Example

In the data model below the phone 'STB2000' has a black and silver model as well as a variant for each size.



When exported to a multi-level Smartsheet the hierarchy is flattened for easy editing.

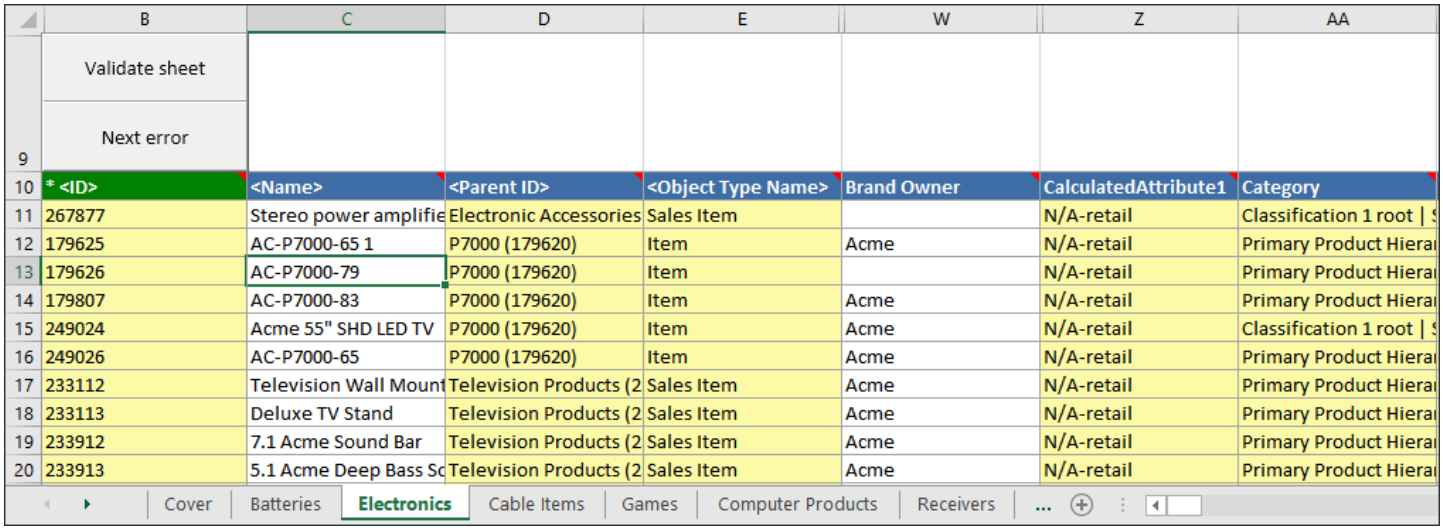


	B	C	D	E	F	K	M
9	Validate sheet Next error						
10	* <ID>	<Name>	<Parent ID>	Height	Width	Manufacturer Name	Primary Color
11	4730888	32 GB	STB2000, Black (4730886)	6.4 in	3.1 in	Acme Electronics, Inc.	Black (Black)
12	4730891	64 GB	STB2000, Black (4730886)	6.4 in	3.1 in	Acme Electronics, Inc.	Black (Black)
13	4730893	128 GB	STB2000, Black (4730886)	6.4 in	3.1 in	Acme Electronics, Inc.	Black (Black)
14	4730896	256 GB	STB2000, Black (4730886)	6.4 in	3.1 in	Acme Electronics, Inc.	Black (Black)
15	4730898	512 GB	STB2000, Black (4730886)	6.4 in	3.1 in	Acme Electronics, Inc.	Black (Black)
16	4730911	32 GB	STB2000, Silver (4730903)	6.4 in	3.1 in	Acme Electronics, Inc.	Silver (Silver)
17	4730913	64 GB	STB2000, Silver (4730903)	6.4 in	3.1 in	Acme Electronics, Inc.	Silver (Silver)
18	4730915	128 GB	STB2000, Silver (4730903)	6.4 in	3.1 in	Acme Electronics, Inc.	Silver (Silver)
19	4730917	256 GB	STB2000, Silver (4730903)	6.4 in	3.1 in	Acme Electronics, Inc.	Silver (Silver)
20	4730920	512 GB	STB2000, Silver (4730903)	6.4 in	3.1 in	Acme Electronics, Inc.	Silver (Silver)
21							
22							
23							
24							
25							

Cover | Electronics

Multiple Product Object Type Smartsheets

In some cases it is necessary to use Excel Smartsheets to initiate / maintain products of different object types intermixed in one sheet. The multi-object Smartsheet allows users to easily upload multiple object types simultaneously, and without them needing to be hierarchically related.



	B	C	D	E	W	Z	AA
9	Validate sheet Next error						
10	* <ID>	<Name>	<Parent ID>	<Object Type Name>	Brand Owner	CalculatedAttribute1	Category
11	267877	Stereo power amplifie	Electronic Accessories	Sales Item		N/A-retail	Classification 1 root | S
12	179625	AC-P7000-65 1	P7000 (179620)	Item	Acme	N/A-retail	Primary Product Hierar
13	179626	AC-P7000-79	P7000 (179620)	Item		N/A-retail	Primary Product Hierar
14	179807	AC-P7000-83	P7000 (179620)	Item	Acme	N/A-retail	Primary Product Hierar
15	249024	Acme 55" SHD LED TV	P7000 (179620)	Item	Acme	N/A-retail	Classification 1 root | S
16	249026	AC-P7000-65	P7000 (179620)	Item	Acme	N/A-retail	Primary Product Hierar
17	233112	Television Wall Mount	Television Products (2	Sales Item	Acme	N/A-retail	Primary Product Hierar
18	233113	Deluxe TV Stand	Television Products (2	Sales Item	Acme	N/A-retail	Primary Product Hierar
19	233912	7.1 Acme Sound Bar	Television Products (2	Sales Item	Acme	N/A-retail	Primary Product Hierar
20	233913	5.1 Acme Deep Bass Sc	Television Products (2	Sales Item	Acme	N/A-retail	Primary Product Hierar

Cover | Batteries | Electronics | Cable Items | Games | Computer Products | Receivers | ...

Multi-object Smartsheets handle multiple mutually independent object types in the same Excel workbook. For each product category selected for download, a sheet will be added to the Excel workbook. Each sheet can contain all configured object types alongside each other.

This Smartsheet type is especially useful when working in a workflow that supports multiple object types in the same state.

Note: In order to support this, the object type must be mapped in the export configuration. If importing new products, this column must be populated with the desired object type for all products. Additionally, the configuration can contain a mapping for Parent ID which displays in the downloaded sheet as a column with a selector dialog so that end users select from valid parent objects.

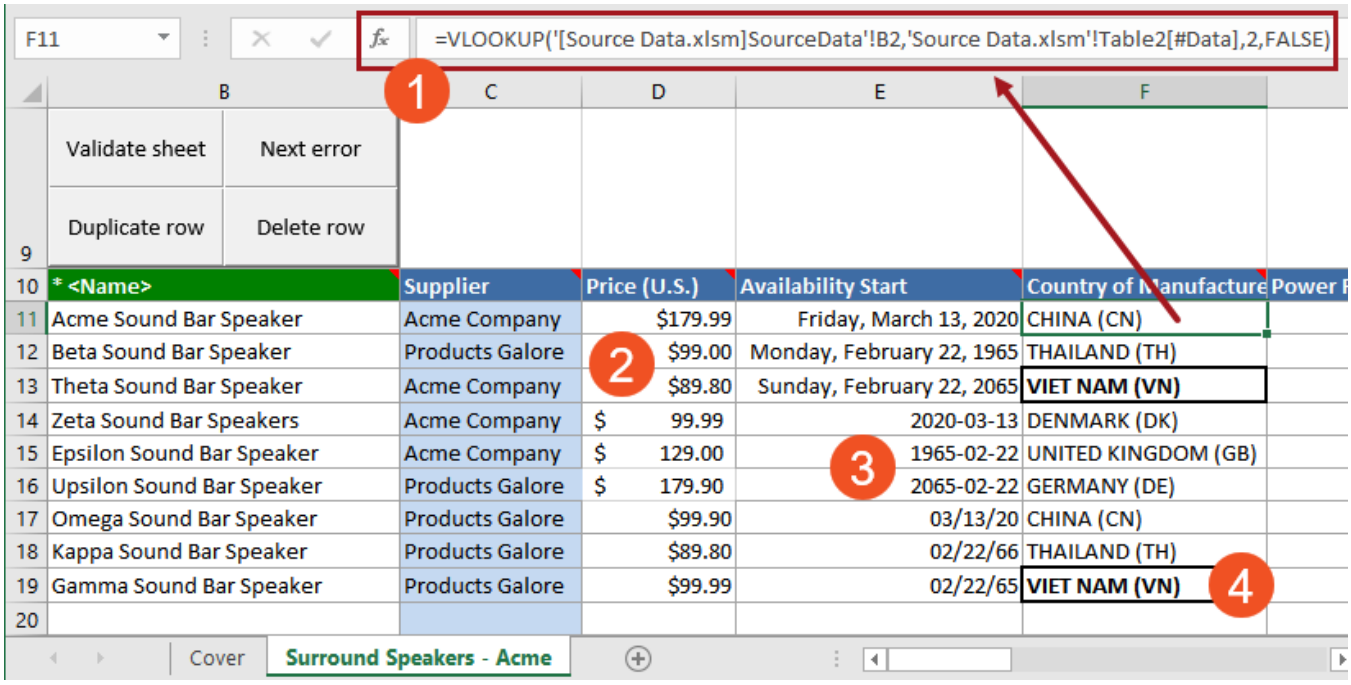
Open Format Smartsheet

In open format Smartsheets, almost all cells are unlocked ('open'), providing users with the option to employ Excel formulas, number formatting (e.g., General, Currency, Date), text formatting (e.g., bold, italic), and cell color (background shading). Enabling these features in Smartsheets improves the product onboarding and maintenance process for suppliers by allowing more use of standard Excel functionality, as opposed to using non-open-format Smartsheets, which do not allow for formulas or visual enhancements like text formatting. Wrap Text functionality is also supported in open format.

The open format option is available for both data export (product maintenance) and template export (product onboarding) Smartsheets.

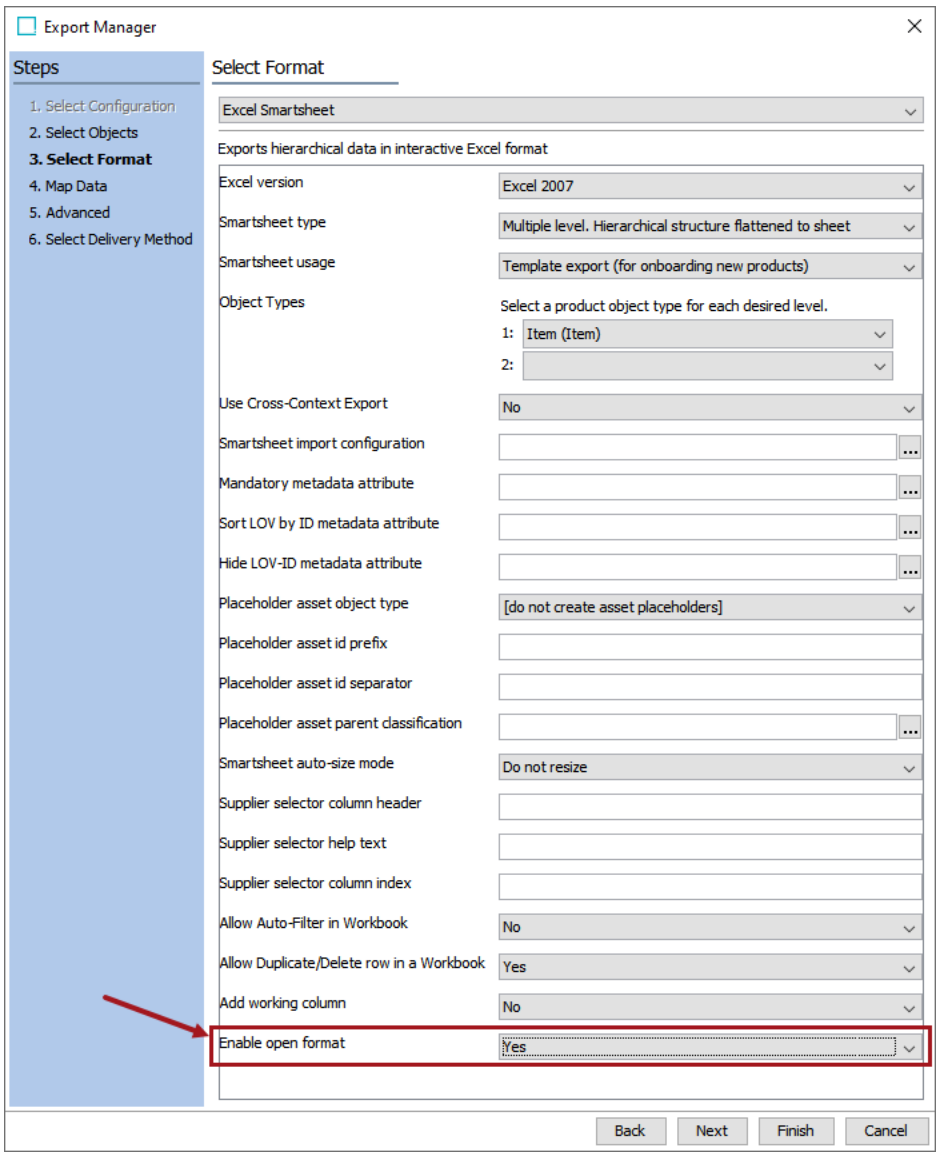
The following screenshot shows an open format Smartsheet that employs several features that are unavailable in non-open-format Smartsheets:

1. Excel formulas. In this example, the formula uses a VLOOKUP function to pull data from an external spreadsheet.
2. Currency and Accounting number formats.
3. Various date formats.
4. Text and cell formatting. This example uses bold text and a thick cell border.



Export Manager Configuration

To export an open format Smartsheet, choose 'Yes' for the **Enable open format** parameter on the 'Select Format' screen of the Export Manager wizard. The default value is No. For full details on how to configure Smartsheet export configurations, refer to the **Smartsheet Data and Template Configurations** topic in this guide.



Default Cell Formats for Open Format Smartsheets

Since data export (product maintenance) sheets already contain data, and template export (product onboarding) sheets do not, the default cell format is different for each sheet, as follows:

- **Open format data export (maintenance) Smartsheets** are exported with all cells set to the **Text** format, which prevents the automatic conversion of exported number values that happens in the General format. For example, if a value with leading 0's (e.g., 0006) is exported with the General cell format, when the user opens the Excel file, Excel automatically removes the leading 0's, changing 0006 to 6.
- **Open format template (onboarding) Smartsheets** are exported with all cells set to the **General** format. The General format makes it easier for users to start entering formulas in the new sheet, as opposed to the Text format, which accepts formulas exactly as entered.

Regardless of the default cell formats, users can change formats on an as-needed basis in either type of open format sheet. For example, to enter a formula in Open Format data export sheets, change the cell format from Text to General.

Working with Formulas in Open Format Smartsheets

Open format Smartsheets allows formulas to be used in any cell where data can be entered. This includes functions such as VLOOKUP, which can populate values from another tab on the Smartsheet or from an external data source.

Though formulas are allowed on a limited basis in Smartsheets that use the 'working column' (more information on which can be found in the **Smartsheet Data and Template Configurations** topic in this guide), formulas used in open format Smartsheets do not require that a 'paste special' operation be used to paste 'values only' into other cells. In open format Smartsheets, values derived from formulas can be left 'as-is.' STEP will recognize the cell contents as actual values and import the evaluations of the formulas and not the formulas themselves.

Note: When executing validation, product rows that contain formulas are always validated, even if the values have not changed. This might impact performance for large Smartsheets that use a significant number of formulas.

Validation of Formulas

When Smartsheets are validated, the following behavior is applied to formulas:

- The 'Validate Sheet' Smartsheet button sends the *result* of the formula evaluation for validation and not the formula itself.
- Cells that contain formulas are always validated, since their values can change along with the source data. Changing a value in an external cell in source data is treated as a change in the sheet.

Additional Considerations for Open Format Smartsheets

Because of the lack of restrictions in open format Smartsheets, additional considerations apply to their use, including:

- What is displayed in the sheet may not always be what you get on import, especially when using number and date formats.
- Unlocking cells for editing allows read-only cells to be edited (though any values in read-only cells are ignored upon import and validation).
- Background cell shading can be used, but in most cases, the shading is removed when the sheet is validated.

Conversely, some cells are still not editable (e.g., header rows and cells that fall outside of the range of 'active' Smartsheet columns and rows), and other standard Excel functionality is still unavailable, e.g., the ability to cut and paste entire columns and rows.

The following subsections address the most important considerations to keep in mind when working with open format Smartsheets.

Handling of Cells with Number Formats

Since cell number format options are available in open format Smartsheets, cells may contain values that are displayed in a particular locale. Numbers may also be displayed in various formats when currency, accounting, and percentage formats are applied.

Numeric data in cells that use either the **General** or **Text** format will be sent to STEP using the displayed value. For all other number formats, the underlying cell value is being used for validation and import, without any formatting. The validate and import implementations are otherwise identical to non-open-format Smartsheets.

Note: Number formatting is not supported in cells with multi-values.

The following table provides more specific details on how STEP validates and imports example text in open format Smartsheets:

Excel Number Format	Excel Display Value Example	Value Imported / Validated
General	$1234,586.80	Exactly as displayed in Excel. If the number is generated by a formula, then the imported / validated value is the result of the formula evaluation and not the formula
Number	89,324.34	The underlying number in the cell. E.g., the cell could be formatted to two decimal places, but the underlying number to import will actually be 89324.345
Currency	£12,350.00	The underlying number in the cell without the formatting. E.g., 12350
Accounting	$ 12,351.00	The underlying number in the cell without the formatting. E.g., 12351
Date	18. oktober 2023	If the base validation type is Date then dates will be imported / validated as 'dd-MMM-yyyy' (E.g., 18-OCT-2023). If the base validation type is ISODateAndTime then dates will be imported / validated as 'yyyy-mm-dd HH:mm:ss' (E.g., 2023-10-18 08:17:40). If the base validation type is ISODate then dates will be imported / validated as 'yyyy-mm-dd' (E.g., 2023-10-18). Otherwise the underlying number without the formatting. E.g., 1234.5667
Time	02:57	The underlying number in the cell without the formatting. E.g., 0.12291667

Percentage	10020.00%	The underlying number in the cell without the formatting. E.g., 100.2
Fraction	1234 69/200	The underlying number in the cell without the formatting. E.g., 1234.345
Scientific	1.23E+03	The underlying number in the cell without the formatting. E.g., 1234.5667
Text	12345.6545	Exactly as displayed in Excel
Special	ISBN --1234-6	The underlying number in the cell without the formatting. E.g., 12345.66789
Custom	£(12,345.14)	If the cell contains a date or a time, then the value imported will be as for date / time cells. Otherwise, the underlying cell value will be imported / validated. E.g., -12345.1433

Read-Only Cells

In open format Smartsheets, since all cells are unlocked in the 'working area,' both the textual content and background shading of read-only cells can be edited, including special fields, like the ID field in data export / maintenance Smartsheets. This enables easier copying and pasting of values across cells.

The 'working area' of a Smartsheet is any cell where data is expected to be entered (attribute or reference columns).

Note: Though read-only cells are editable, any values in read-only cells are ignored upon import and validation of Smartsheets, even if the values have been changed. Additionally, if background shading is applied to a read-only cell, its color will return to pale yellow when the row is validated. This keeps the user informed of the cell's special meaning.

Background Shading and Text Formatting

Though cell shading and text coloring is available in open format Smartsheets, it is not recommended to rely on this formatting since it is almost always removed when Smartsheets are validated. Some considerations are as follows:

- *All* background shading and text coloration will be removed from an open format Smartsheet the first time it is validated. After the first validation, shading and text coloring can be re-added, but it will be removed on subsequent validations from cells in rows where data has been changed.
- Since cells that contain formulas are always validated, background shading will always be removed from cells and rows that contain formulas.
- The pale yellow background shading of read-only and conditionally read-only cells can be overwritten, which is not recommended since read-only cells can also be edited. If the shading is removed, this could be

confusing for users who think that the changed content can be imported into STEP. When the sheet is validated, the pale yellow color is restored.

> **Note:** Changed values in read-only cells will not be restored to their original value(s) after validation.

- The red shading of error-indicating cells can also be overwritten, which only affects the color, and not the error. When the sheet is validated, the red color is restored.
- Any text coloration, e.g., red, is removed upon validation. However, non-color-based formatting, such as bold, italic, and strike-through, will remain after validation. Cell borders also remain after validation.
- Text styles (e.g., bold and italic) are for local use only, and are not imported into STEP. Text in STEP must still be formatted using features such as the Rich Text Editor in the Web UI.

Excel Conditional Formatting

Validation does **not** remove cell formatting that is applied using Excel's conditional formatting feature.

Due to this, it is advised to **not** use conditional formatting, since this formatting can override the red color of cells where validation has failed. I.e., if Excel conditional formatting is applied to a cell and that cell fails validation, the cell remains the conditional formatting color instead of red.

Copy and Paste Behavior for Rows and Columns

As with non-open-format Smartsheets, entire rows and columns cannot be cut and pasted / copied and pasted due to hidden, protected columns and rows within the Smartsheet that contain STEP-related information. The closest available functionality to 'copy and paste' row is the **Duplicate Row** functionality, which is available in template (onboarding) sheets. No duplication functionality is available for columns, since they are mapped to individual STEP attributes. For more information on using the Duplicate Row button, refer to the **Using a Smartsheet** topic in this guide.

Additional Information About Smartsheets

This topic provides configuration and general information, as well as links for Excel Smartsheets.

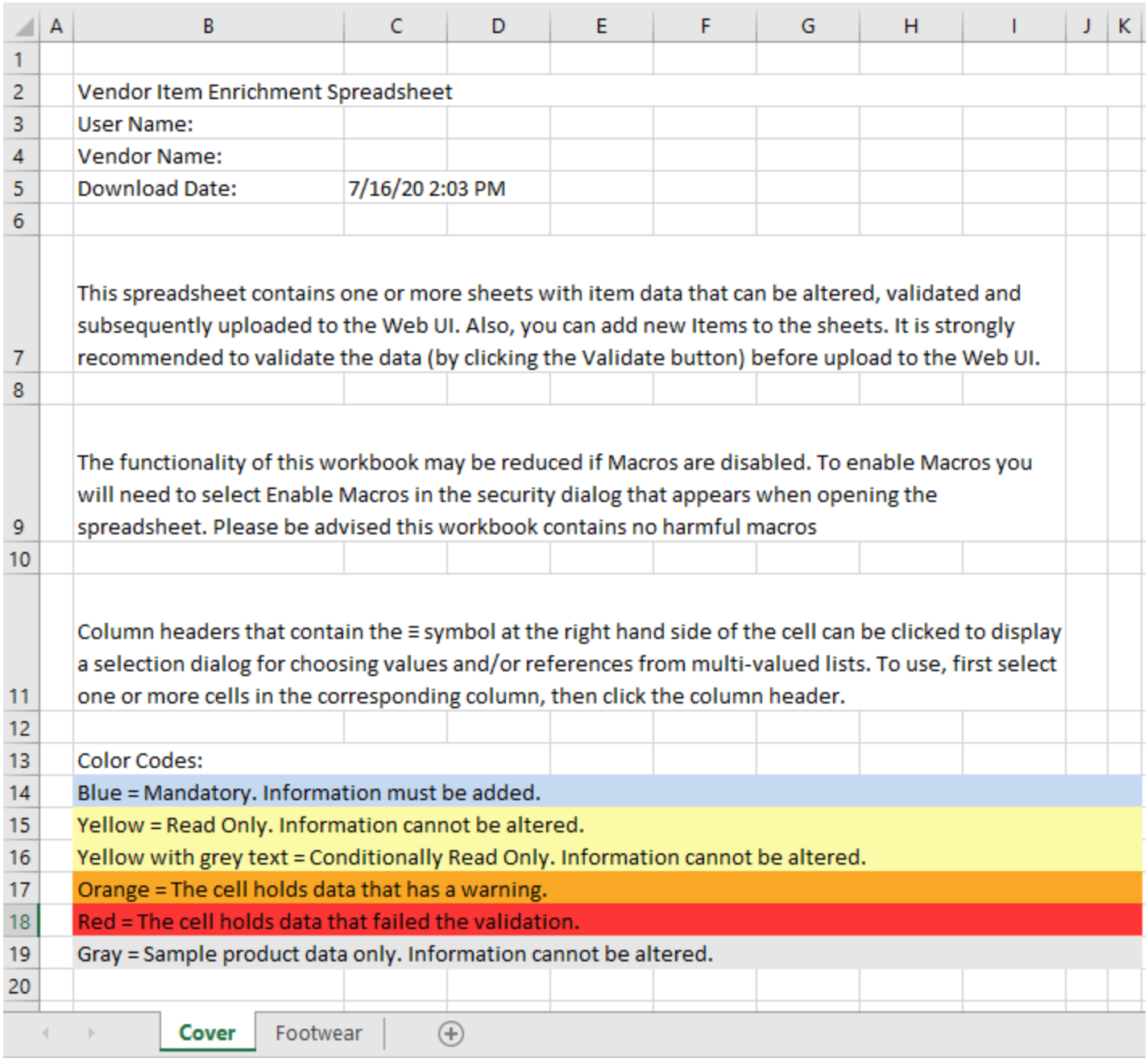
System Configuration

Smartsheets have some additional configuration options that can impact the entire functionality and the rest of the system. These options include:

- Using special background processes and queues for Smartsheets specifically. For more information, refer to the **Smartsheet Background Processes and Queues** topic in this guide.
- Allow a supplier user to view all tasks across all suppliers for which the user has privileges, in both the workbench and Web UI. For more information, refer to the **Smartsheet Multi-Supplier View** topic in this guide.
- Limit supplier browsing based on user privileges. For more information, refer to the **Smartsheet System Properties** topic in this guide.
- Override the URL of the Web UI server when exporting a Smartsheet. For more information, refer to the **Smartsheet System Properties** topic in this guide.
- Add Smartsheet validation to Excel's Trusted Header authentication. For more information, refer to the **Smartsheet System Properties** section in this guide.
- The addition of a special 'Template Product' row that contains sample product data, which provides users with a visual example of how to correctly fill out attributes. For more information, refer to the **Smartsheet Template Product Row** topic in this guide.

Smartsheet Cover Tab

In addition to the tab containing data, all Smartsheets exported contain an cover sheet tab. This tab includes details about the file, information about macros, and an explanation of color coding of various fields.



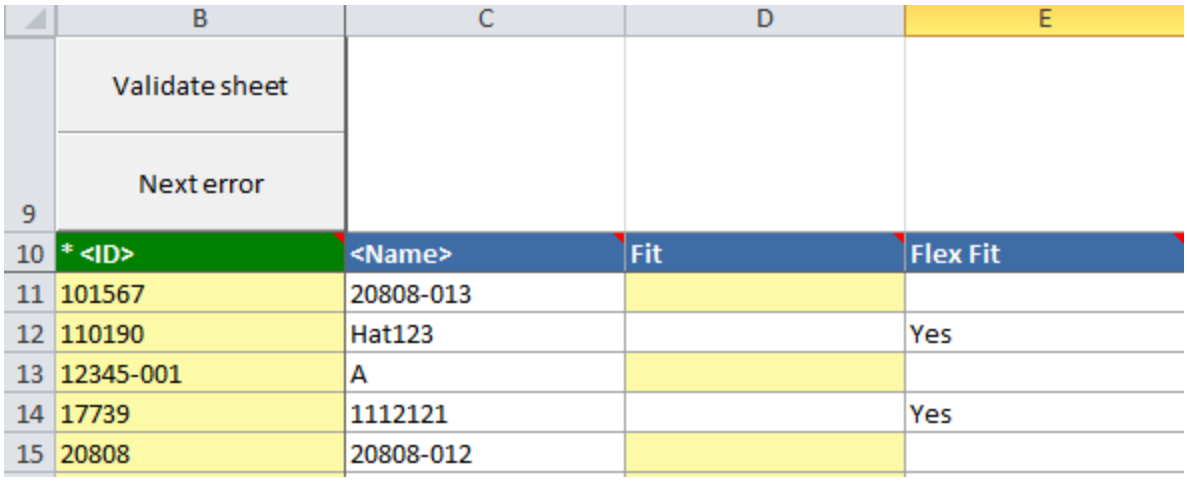
Because values in read-only cells can be edited in **open format** Smartsheets, the yellow rows on the cover sheet of open format Smartsheets read as follows:

- Yellow = Read Only. Information will be ignored upon import and validation.
- Yellow with gray text = Conditionally Read Only. Information will be ignored upon import and validation.

For more information on open format Smartsheets, refer to the **Open Format Smartsheet** topic in this guide.

Smartsheet Conditional Attribute Handling

When exporting products with conditional attributes, any cells containing invalid conditionally displayed values will be automatically locked for editing. To unlock the cell and make changes, the user must first fulfill the proper conditions for the attribute and then validate the sheet.



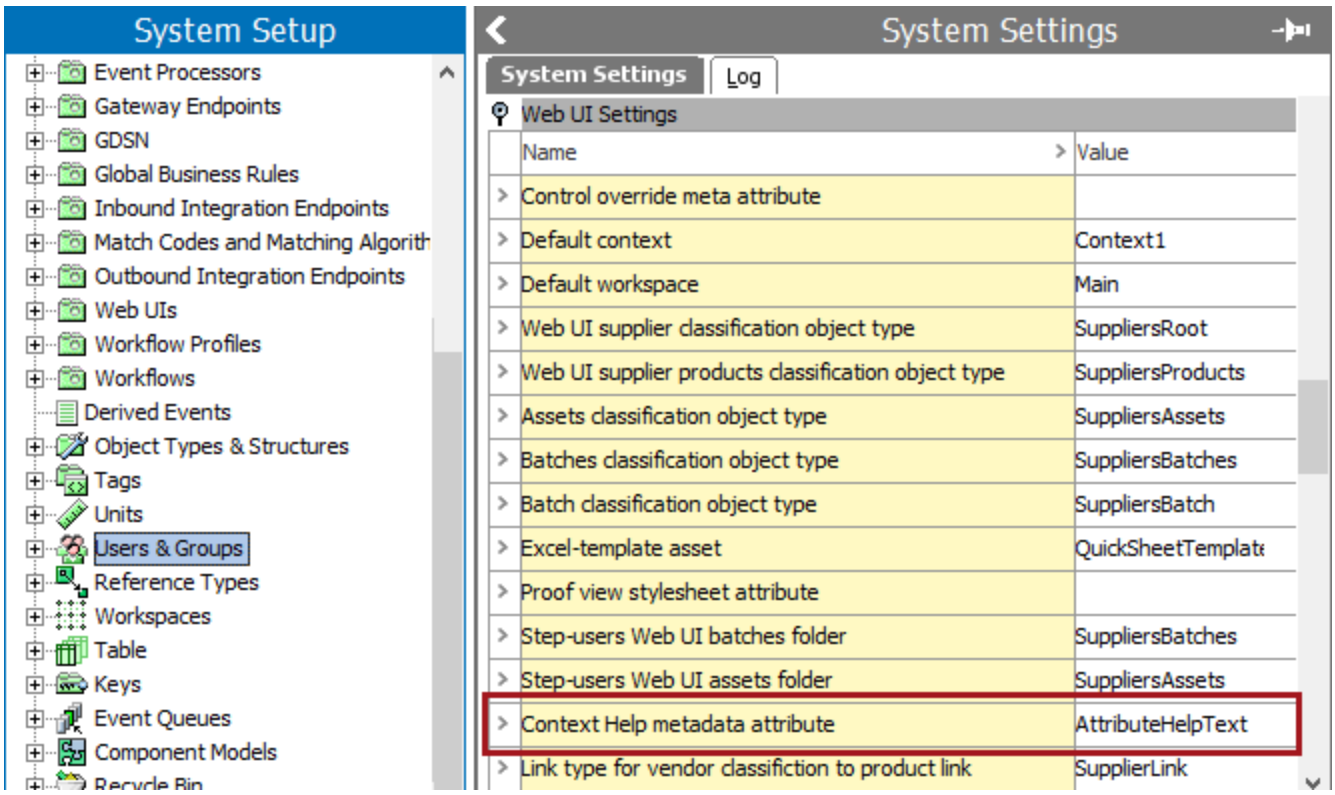
	B	C	D	E
9	Validate sheet Next error			
10	* <ID>	<Name>	Fit	Flex Fit
11	101567	20808-013		
12	110190	Hat123		Yes
13	12345-001	A		
14	17739	1112121		Yes
15	20808	20808-012		

For more information on how conditional attributes behave in Smartsheet exports, refer to the **Conditional Attribute Display** section of the **System Setup** documentation.

Smartsheet Tool Tips for Column Headers

Smartsheet column headers are capable of displaying tool tips when mousing over them. The text provided in the tool tips is controlled by a metadata attribute that must be configured in System Setup.

To configure the metadata attribute, navigate to System Setup > User & Groups > Web UI Settings > **Context Help Metadata attribute** and enter the name of the metadata attribute.



Note: Specification attributes valid by object type will only be included for the products exported in a Smartsheet if the attribute is linked to a direct ancestor of the exported product. This means that attributes linked to the product or classification is included in a Smartsheet export.

Additionally:

- attributes must be linked to product or classification categories
- attributes must be valid for the object types being exported

Mandatory Attributes and References in Smartsheets

It is often the case that certain attributes and references are considered critical to all processes, or to a particular process, and therefore there is a desire to make them mandatory (i.e., to enforce that they are populated). Smartsheets support mandatory attributes and references by providing visual indicators and error reporting on missing values.

Both **attributes** and **references** can be made mandatory within Smartsheets, providing both a visual indicator and error reporting on missing values. This is considered a process-based standard mandatory setting, preferred over general business rules or approval settings, as it allows for specific attributes and/or references to be indicated as required at the time that they should be provided (rather than at a later approval stage).

Mandatory attributes and references can be configured for support in Smartsheets based on one or more of the following settings:

- **Standard mandatory** - by indicating that attributes and references set to mandatory directly on the attribute, attribute link, or reference type should be respected. Refer to the **Configure Standard Mandatory Settings in Smartsheets** topic in this guide.
- **Workflow state** - using the workflow State Editor, where *conditionally* mandatory attributes and references are also supported. Refer to the **Configure Workflow State Mandatory in Smartsheets** topic in this guide.
- **Map Data** - using the 'Mandatory' mapping column on the Map Data step of Export Manager. Refer to the **Map Data** section of the **Smartsheet Data and Template Configurations** topic in this guide.
- **Metadata attribute** - by creating a description attribute that determines an attribute or reference is mandatory. This is legacy functionality and the least desirable method. Refer to the **Configure Metadata Attribute for Mandatory in Smartsheets** topic in this guide.

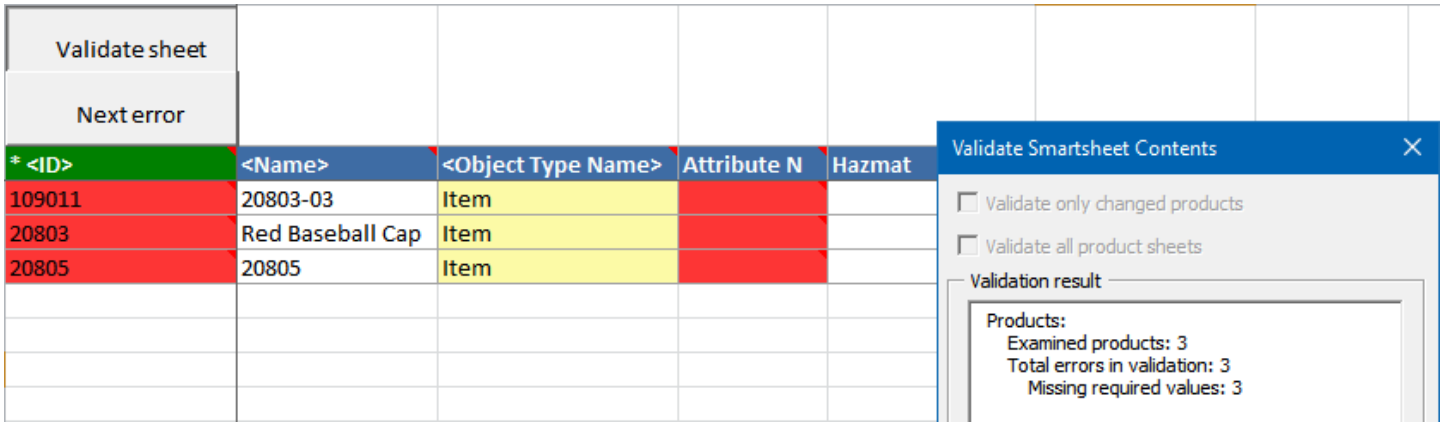
Note: When any one of these settings is used, the object is mandatory in a Smartsheet. There is no adverse effect if multiple mandatory methods are set.

Visual Indication of Mandatory in Smartsheets

Mandatory attributes configured for Smartsheets using any of the above methods have their value cells highlighted blue to prompt the user to populate the field, as shown in the screenshot below.

* <ID>	<Name>	<Object Type Name>	Attribute N	Hazmat
109011	20803-03	Item		
20803	Red Baseball Cap	Item		
20805	20805	Item		

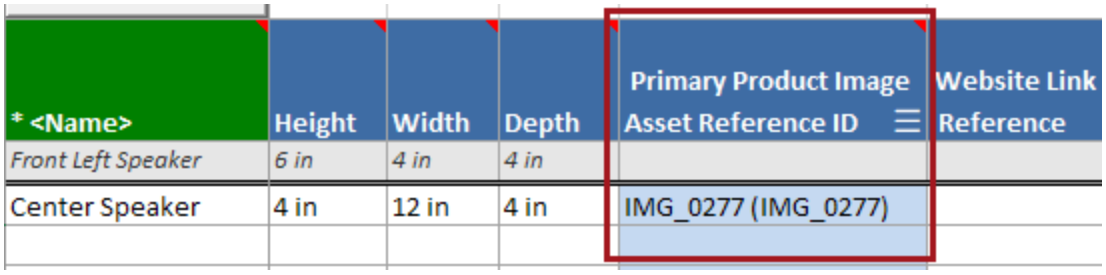
If the user validates the sheet and has not populated the mandatory values, an error is reported. In this case, the row indicator, as well as the missing field, are highlighted red.



Although users may still choose to import the sheet without populating the mandatory values, this method highlights data that should be populated and users can be instructed not to import until validation of the Smartsheet completes without errors.

Mandatory references are configured in the workbench similarly to how mandatory attributes are configured.

The handling is the same, with both a visual indicator and error reporting on missing values. References marked as mandatory have their value cells highlighted blue to prompt the user to populate the field, as shown in the screenshot below. If the user validates the sheet and has not populated the mandatory values, an error is reported.



Note: Supported reference types are product reference types, image and document reference types, product to classification link types, and entity reference types. Product attribute link types, classification reference types, and classification attribute link types are not supported.

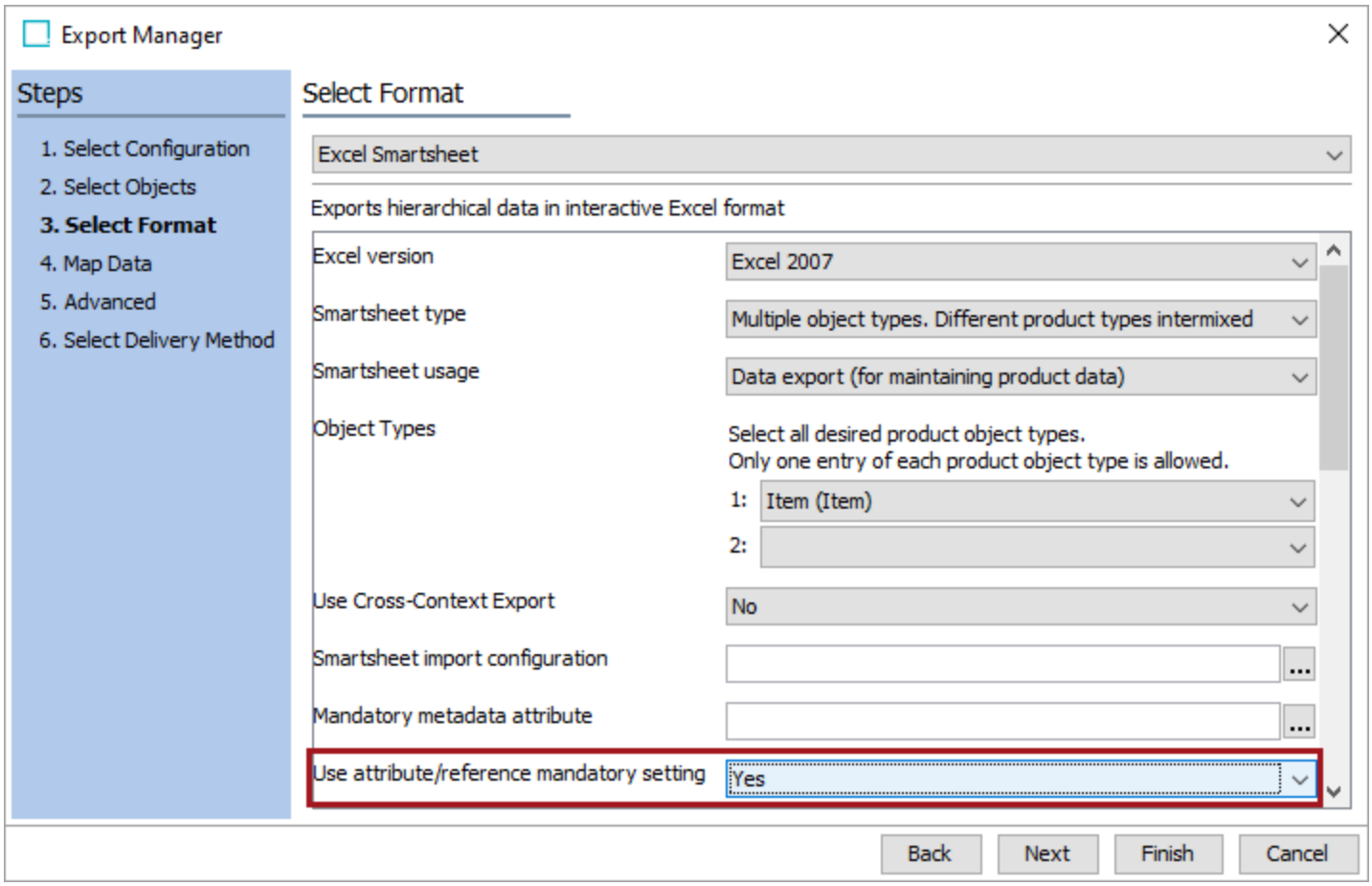
Configure Standard Mandatory Settings in Smartsheets

For general information on making **attributes** and **references** mandatory within Smartsheets, providing both a visual indicator and error reporting on missing values, refer to the **Mandatory Attributes and References in Smartsheets** topic.

Smartsheets can be configured to honor the standard methods for setting mandatory status as defined below.

Regardless of the standard method used to set an attribute or reference as mandatory, the Smartsheet configuration must also be set to honor the standard mandatory settings.

1. Verify or set the mandatory status on an attribute or reference using one of the following methods:
   - On the attribute itself, as defined in the **Mandatory Attributes** topic in the **System Setup** documentation.
   - On an attribute linked to a hierarchy, as defined in the **Mandatory Linked Attribute Values** section of the **Maintaining Attribute Link Metadata Values** topic in the **System Setup** documentation.
   - On a reference type itself, as defined in the **Reference Type - Advanced** topic in the **System Setup** documentation.
2. In Export Manager, set the 'Use attribute/reference mandatory setting' parameter on the Excel Smartsheet format to **Yes**.



3. Complete the additional Export Manager steps and click the **Finish** button to save the configuration, as defined in the **Smartsheet Data and Template Configurations** topic.

Configure Workflow State Mandatory in Smartsheets

For general information on making **attributes** and **references** mandatory within Smartsheets, providing both a visual indicator and error reporting on missing values, refer to the **Mandatory Attributes and References in Smartsheets** topic.

In Smartsheets, you can enforce the mandatory status of attributes, attribute groups, and/or references on workflow states. Conditionally mandatory attributes and references are also supported. Attributes and references mandatory for *transitions* are not enforced in Smartsheets, since the selected transition is not known.

Prerequisite

Apply mandatory and/or conditionally mandatory attributes and references on workflow states as defined in the **Mandatory Attributes and References in Workflows** topic in the **Workflows** documentation.

Conditional Attribute Handling in Smartsheets

When exporting products with conditional attributes, any cells containing invalid conditionally displayed values will be automatically locked for editing. To unlock the cell and make changes, the user must first fulfill the proper conditions for the attribute and then validate the sheet.

	B	C	D	E
9	Validate sheet Next error			
10	* <ID>	<Name>	Fit	Flex Fit
11	101567	20808-013		
12	110190	Hat123		Yes
13	12345-001	A		
14	17739	1112121		Yes
15	20808	20808-012		

For more information on how conditional attributes behave in Smartsheet exports, refer to the **Conditional Attribute Display** section of the **System Setup** documentation.

Mandatory Handling in Smartsheets

When maintaining objects in a Smartsheet that have been exported for a workflow state (i.e., from the Web UI Task List screen), any cells containing attributes or references that are mandatory for that particular state are shaded blue.

Important: Attributes and references mandatory for transitions are not enforced in Smartsheets, since the selected transition is not known.

Unless the condition driving the mandatory status of the attributes or references returns 'False', the user must enter values for all applicable attributes / references before the sheet can be validated. If the user validates the sheet and has not populated the mandatory values, an error is reported.

	B	C	F
9	Validate sheet Next error		
10	* <Name>	Supplier	Co-op
11	Game A	Products Galore	Yes
12	Game B	Products Galore	No
13	Game C	Products Galore	Yes

Conditionally Mandatory Handling in Smartsheets

Smartsheets indicate the mandatory status of conditionally mandatory attributes / references for workflow states in the same manner as non-conditional mandatory attributes / references, by shading the cells blue. If the user validates the sheet and has not populated the mandatory values, an error is reported.

* <Name>	Height	Width	Depth	Primary Product Image Asset Reference ID	Website Link Reference
Front Left Speaker	*6 in*	*4 in*	*4 in*		
Center Speaker	4 in	12 in	4 in	IMG_0277 (IMG_0277)	

Unless the condition driving the conditionally mandatory status of the attributes returns 'False,' the user must enter values for all applicable attributes before the sheet can be validated. Additionally, if changes are made to the attribute values or other factors driving the condition, the sheet needs to be validated to reflect those changes.

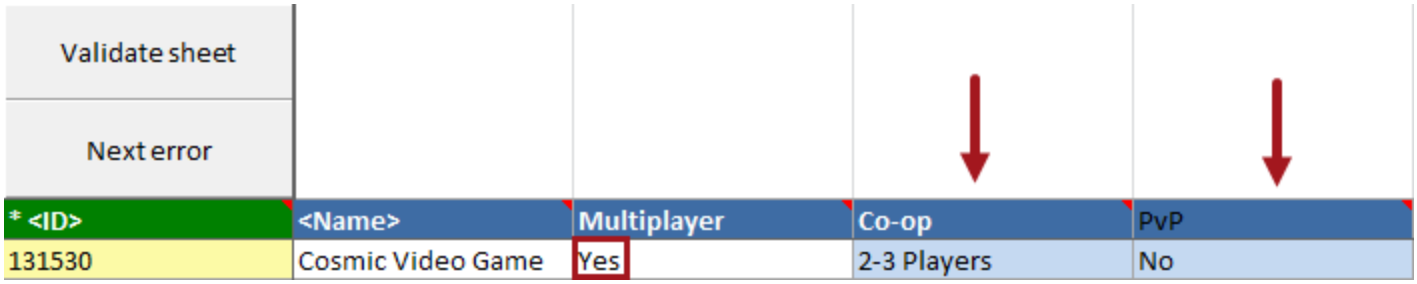
For more information about conditionally mandatory attributes and references in workflows, refer to the **Conditionally Mandatory Attributes and References in Workflows** section of the **Workflows** documentation.

Example

The following example uses a conditionally mandatory attribute, but the same functionality applies for conditionally mandatory references.

In this example workflow state, a business condition has been applied that states that if the value of the attribute 'Multiplayer' is 'Yes,' the attributes 'Co-op' or 'PvP' are mandatory. Otherwise, 'Co-op' and 'PvP' are not mandatory.

In the image below, the condition is returning true, so 'Co-op' or 'PvP' are mandatory (and are highlighted blue). The sheet can be validated without errors.



In the image below, the condition is returning false, so 'Co-op' and 'PvP' are not mandatory (and are not highlighted). The sheet can be validated without errors.

In the image below, a validation was run and reported the object and its missing attributes by highlighting their cells red. The sheet failed because condition was true while 'Co-op' and 'PvP' are missing a value.

Configure Metadata Attribute for Mandatory in Smartsheets

For general information on making **attributes** and **references** mandatory within Smartsheets, providing both a visual indicator and error reporting on missing values, refer to the **Mandatory Attributes and References in Smartsheets** topic.

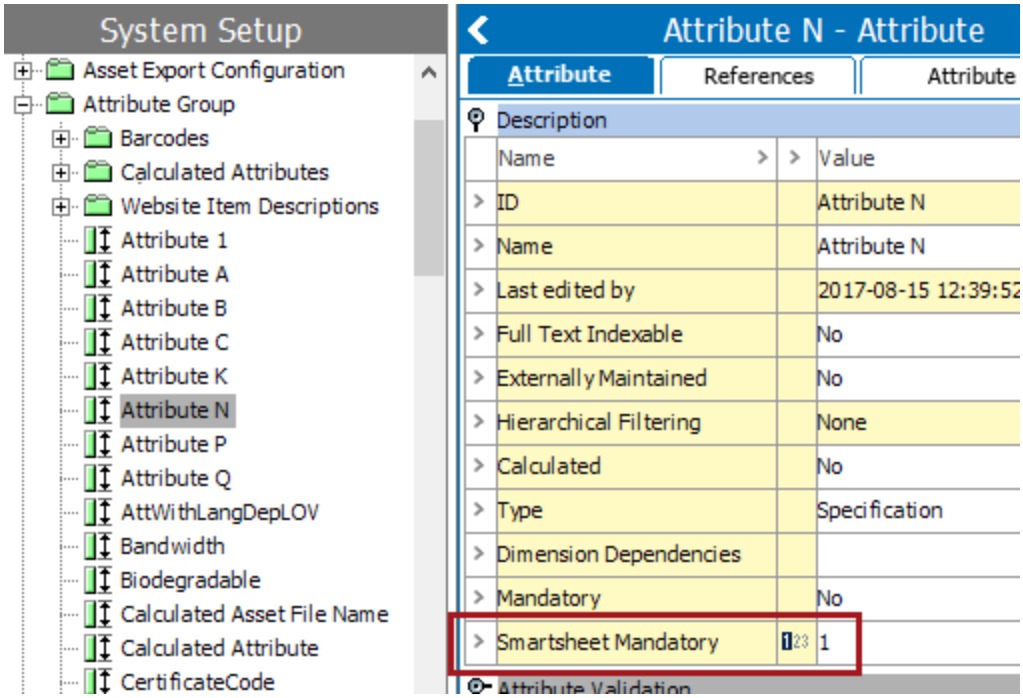
A metadata attribute can be used to indicate that an attribute or reference is mandatory in a Smartsheet. The following sections demonstrate how to configure an attribute for this purpose.

Mandatory Attributes in Smartsheets

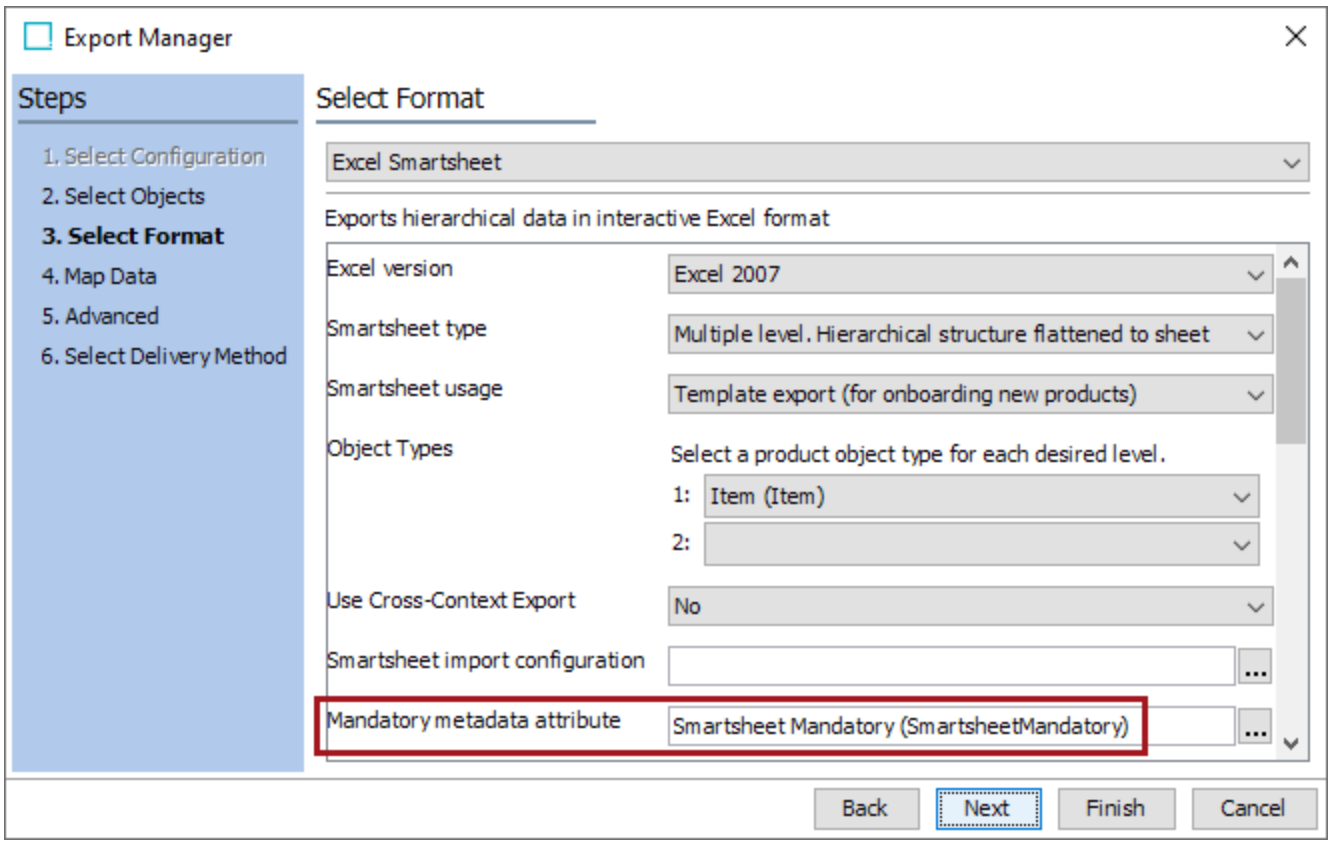
Configure a metadata attribute to indicate mandatory values in Smartsheets:

1. Create a description (metadata) attribute with a validation base type of Integer and make it valid on the attribute basic object type.

   Details on how to make an attribute valid as metadata on an attribute are available in the **Attribute Metadata on Attributes** topic in the **Description Attributes** section of the **System Setup** documentation. The end result should resemble the image below, which includes a metadata attribute named 'Smartsheet Mandatory' that is available on all valid attributes. This example shows an attribute named 'Attribute N.' (The metadata attribute can be named anything; 'Smartsheet Mandatory' is simply an example attribute.)



2. Select the metadata attribute in the 'Mandatory metadata attribute' field in the Select Format step of the Smartsheet export configuration.



More information on configuring Smartsheet exports is available in the **Smartsheet Data and Template Configurations** section of this guide.

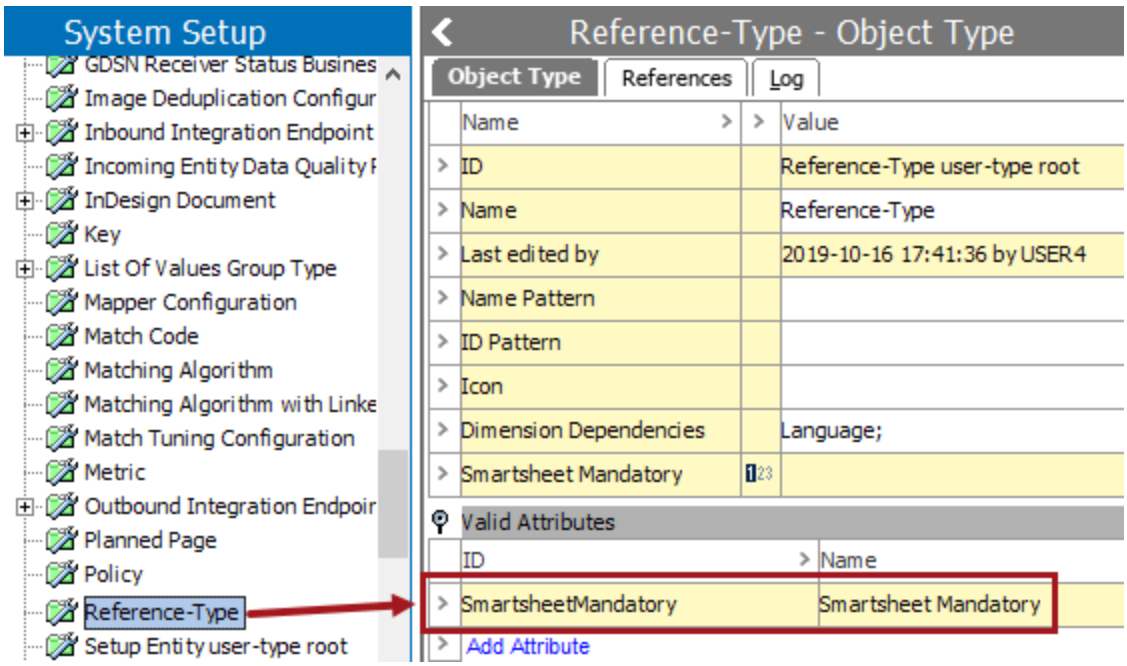
3. For any attribute that should be made mandatory in the Smartsheet, populate a non-zero integer in the metadata attribute.

   For example, as shown in step 1, a metadata value of '1' will cause the attribute to be made mandatory in the Smartsheet. Note that the value itself is not used for anything, so entries of 1, 10, and 100 are equivalent, as the system is concerned only with the presence / absence of a non-zero value.

Mandatory References in Smartsheets

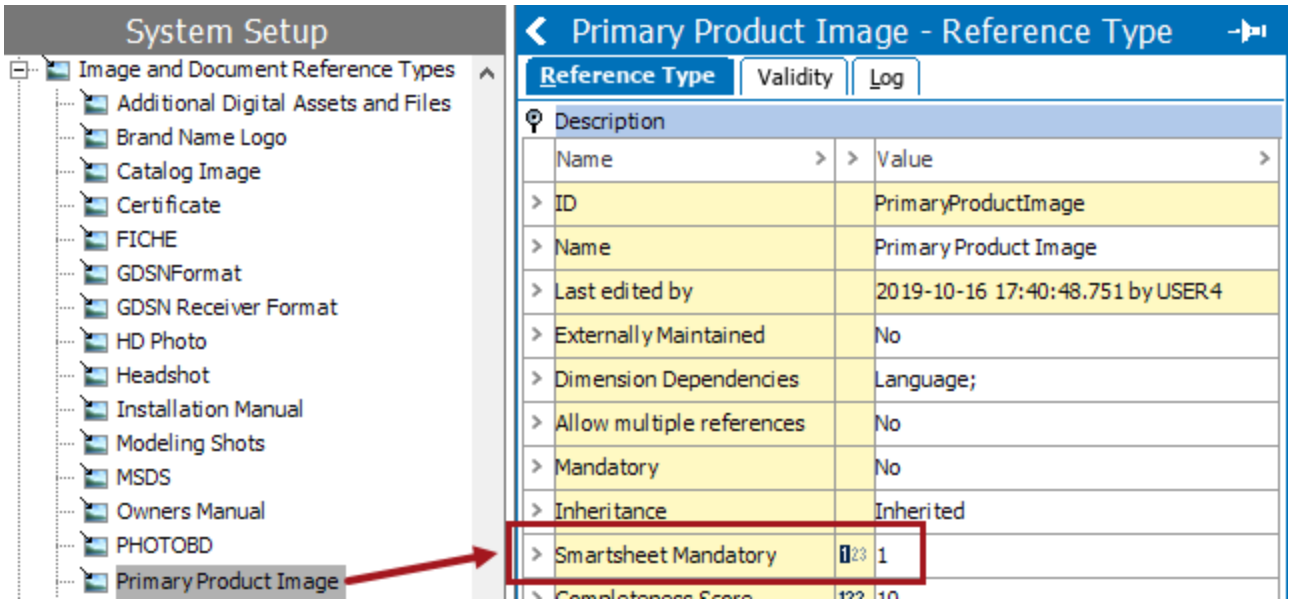
Configure a metadata attribute to indicate mandatory values in Smartsheets:

1. Create a Description (metadata) attribute with a validation base type of Integer and make it valid on the 'Reference-Type' object type. The process for making an attribute valid as metadata on reference types is the same as that for attributes, which is detailed in the **Attribute Metadata on Attributes** topic in the **Description Attributes** section of the **System Setup** documentation.



2. For the relevant reference type(s) that should be made mandatory in the Smartsheet, populate a non-zero integer in the metadata attribute.

   The end result should resemble the image below, where there is a metadata attribute named 'Smartsheet Mandatory' that is available on all reference types. In this example, the Primary Product Image reference type is shown. (The metadata attribute name can be anything; 'Smartsheet Mandatory' is simply an example.)

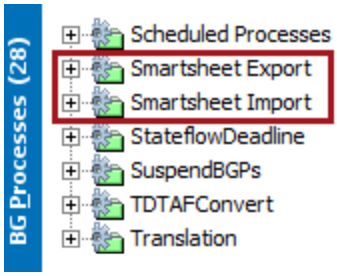


   As with mandatory attributes, the metadata attribute will be selected in the 'Mandatory metadata attribute' field in the Smartsheet export configuration.

Smartsheet Background Processes and Queues

When the default configurations are used, Smartsheet imports and exports are placed in the queue along with other import and export processes. However, Smartsheet imports and exports can be set up to run on unique queues so that they are not competing for resources with other import and export processes.

To accomplish this, the user must populate the appropriate process template when executing an import or export (or have it set in the configuration). Once executed, the import or export is displayed in a separate area in the **Background Processes** tab in workbench. Smartsheet export processes will appear in the **Smartsheet Export** folder while import processes will appear in the **Smartsheet Import** folder.



To view the background processes initiated by another user, click the 'All Users' button at the top of the BG Processes tab.

For details on the states of a background process, refer to the **BG Processes States** topic in the **System Setup** documentation.

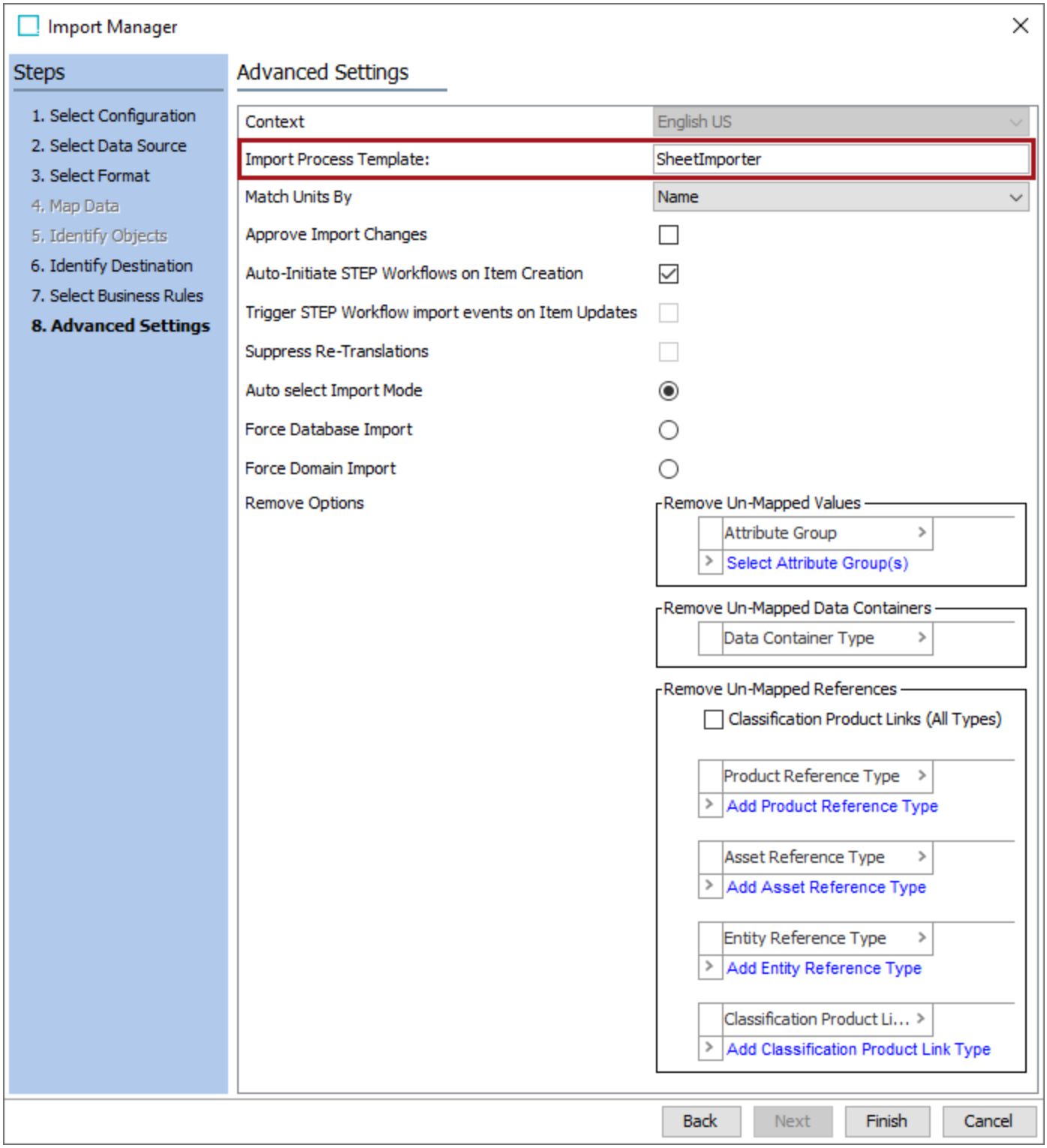
Important: 'BackgroundProcess.ProcessType.SheetExporter.Queue' and 'BackgroundProcess.ProcessType.SheetImporter.Queue' default to the same queue as Export Manager and Import Manager, and must be changed in sharedconfig.properties before they can run in their own queues. This step must be completed before the corresponding process templates can be used.

Using Smartsheet Import Background Process

On the Advanced Setting step in the Import Manager, navigate to the **Import Process Template** field and type 'SheetImporter.'

This will tell the system to display the import in the **Smartsheet Import** folder on the **Background Processes** tab.

Refer to the **Import Manager** section of the documentation for more information.

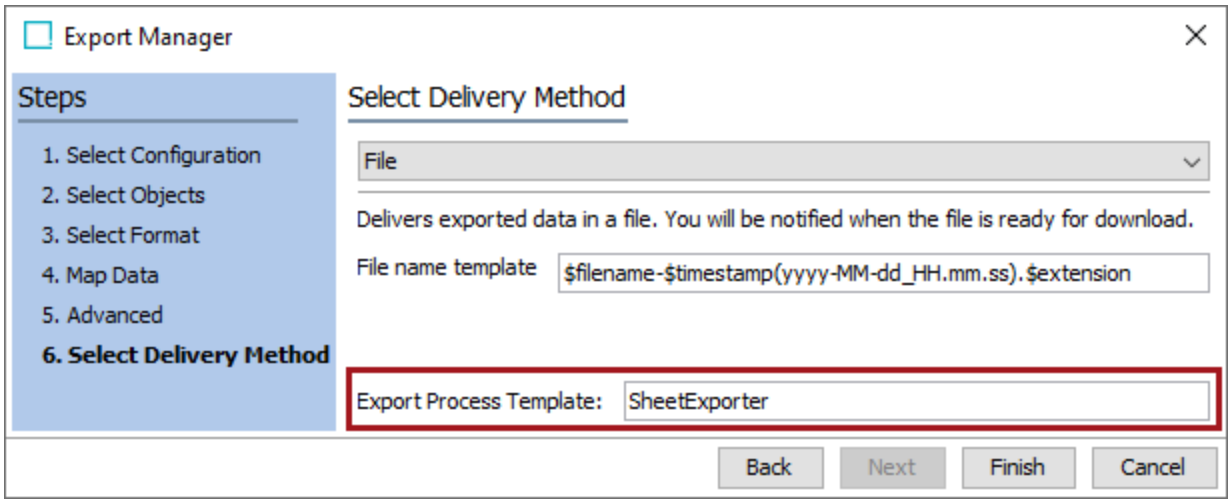


Using Smartsheet Export Background Process

On the Select Delivery step in Export Manager, navigate to the **Export Process Template** field and type 'SheetExporter.'

This will tell the system to display the export in the **Smartsheet Export** folder on the **Background Processes** tab.

Refer to the **Export Manager** section of the documentation for more information.



Smartsheet Multi-Supplier View

A supplier user can view all tasks across all suppliers for which the user has privileges, in both the workbench and Web UI. In other words, users in multiple supplier user groups will be able to view all objects (e.g., products, entities, and assets) linked to those groups without having to toggle between suppliers in the Web UI. This makes it easier to get an overview of tasks for users working across multiple suppliers or multiple Global Location Numbers (GLNs) modeled as STEP suppliers. In addition, users can create references between objects linked to the varying suppliers for which they have access.

This functionality requires that multi-supplier users who initiate products or entities must select the specific supplier for which the products or entities are applicable as they will not necessarily be working within a single supplier. However, as assets may at times be re-used among multiple GLNs / suppliers, the user will have the option to select multiple suppliers for which assets may be uploaded and assigned.

Within the workbench, users can view any objects belonging to the suppliers for which they are a member, regardless of whether the object is linked to one or all suppliers that the user has access to.

To enable this functionality, set the 'Enable all-view for users that are a member of multiple suppliers' parameter to 'Y' as defined in the **Web UI Settings** topic in the **System Setup** documentation.

Smartsheet System Properties

Configuration properties, set in the sharedconfig.properties file on the application server, are available for Smartsheets and can alter the system behavior.

Limit Supplier Browsing

It is possible to allow the Smartsheet user to browse and search the full data set in STEP in accordance with the user's privileges. This is controlled using the case-sensitive `SmartSheet.LimitSupplierBrowsing` property, for example.

```
SmartSheet.LimitSupplierBrowsing=true
```

If this property is true, users of a Smartsheet that has been exported for a specific supplier will only be able to browse / search for products and assets linked into appropriate supplier classifications. This means that while product references and asset references remain intact, the Smartsheet displays only the objects the user has privileges to access.

If this property is false, users will be able to browse / search for all products and assets using the same privileges as in workbench.

Important: Regardless as to how this property is set, the browse / query service disregards whether or not the current user is a supplier user. If the property is true, even super users are limited to supplier classifications. If false, supplier users will be able to access the full hierarchy using normal privileges.

The setting defaults to true, as these limitations have always been applied in Smartsheets.

Service URL Override

In order for the online data validation of Smartsheet data to work, the address (URL) of the service must be embedded in the exported sheet. When exported through Web UI, this setting is assigned the address of the Web UI server. It is possible to override this with a configuration setting (for STEP Workbench export or load balancing). The STEP attributes and references used in the Smartsheet can only be validated if this URL ultimately (through whichever Web UI server) points back to the STEP server from which the Smartsheet is exported. For example, it is possible to export a Smartsheet from WebUIServerA and have it validated at WebUIServerB as long as they share the same STEP backbone.

This setting is controlled via the case-sensitive `SmartSheet.ValidationService.URL` property.

Note: Changing this setting may corrupt existing Smartsheet configurations. After a change, test the existing configurations and modify or recreate as necessary.

Trusted Headers Authentication

Make use of Trusted Headers authentication in conjunction with Excel Smartsheets to enable end users to, for example, use their intranet login and password when validating Excel Smartsheets. Use the following instructions to configure Trusted Headers in Excel Smartsheets:

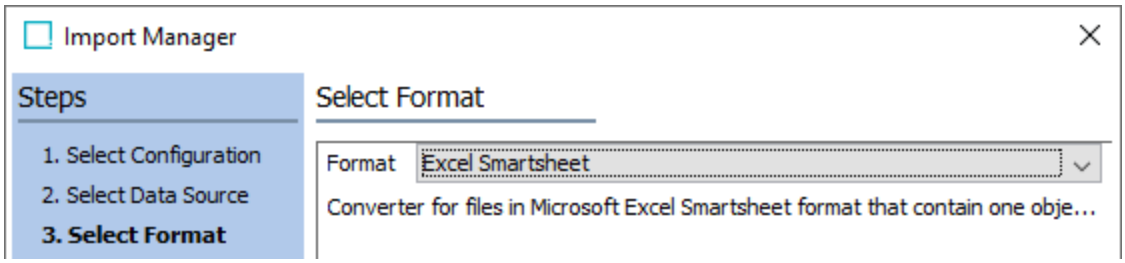
1. An authentication proxy needs to be in place that can:
   - Make basic authentication from the calls from Excel
   - Extract the User ID and password from the basic authentication token
2. Forward all requests, including all payloads, from Excel to STEP if the request is authenticated
3. Add a header to all requests to STEP containing the User ID
4. Set the following properties in the sharedconfig.properties file:
   - `Excel.UseTrustedHeaders=true`
   - `Excel.TrustedHeaders.Username=[Name_of_the_header_that_contains_the_User_ID]`

Increasing Default of Preformatted Template Rows

When creating a new item onboarding Smartsheet template, the default max is 1,000 rows. Using the case-sensitive `SmartSheet.PreformattedTemplateRows` property, you can increase the generated default up to 5,000 rows. For example:

```
SmartSheet.PreformattedTemplateRows=5000
```

The maximum number will be displayed as a pop-up warning, when you edit cells beyond the preformatted limit. If you continue and exceed more than the maximum allowed number of rows, an error message will be displayed upon import that states the exceeded row limit. Also, the additional row(s) will be cleared so that invalid new item onboarding rows are not created.



Note: There is no row limit for imported data export (maintenance) Smartsheets.

Use Decimal Separators from Native Excel Client

Users have the option to export price data with a decimal separator format that conforms to Excel's local settings. This setting can be found in Excel under File > Options > Advanced > Editing options > Use system separators.

To conform to Excel's decimal separator format, set the following property:

```
SmartSheet.UseClientSeparator=true
```

To ignore Excel's decimal separator format and export the data as is, set the following property:

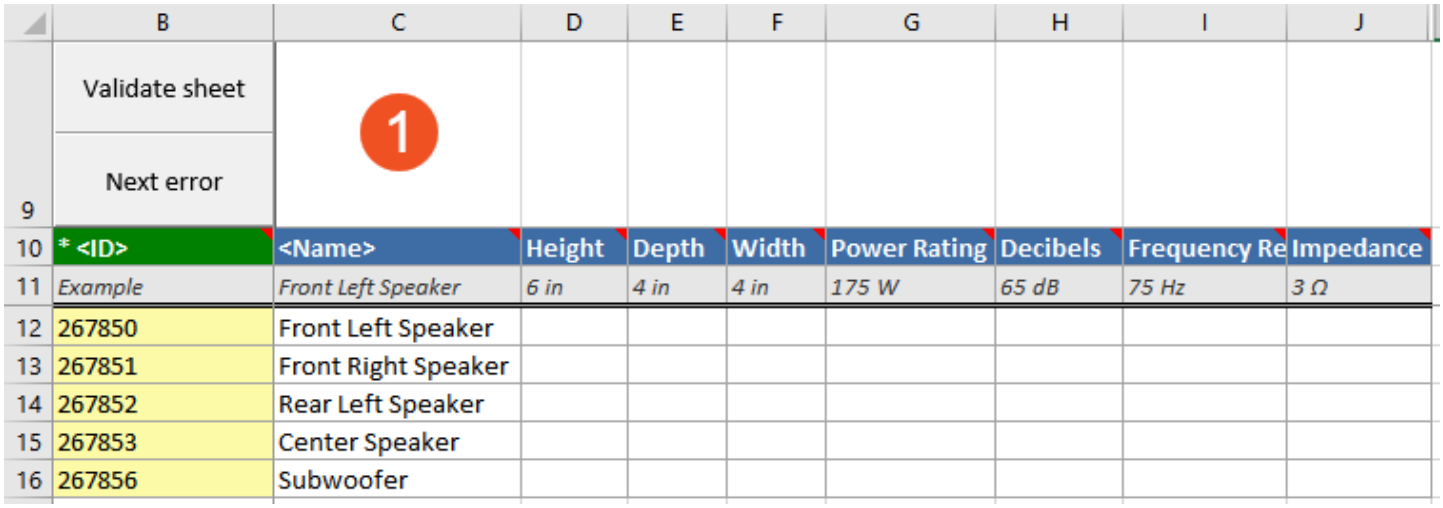
```
SmartSheet.UseClientSeparator=false
```

For example, if the Excel setting uses ',' as a decimal separator and the Smartsheet export uses a '.', if this property is set to 'false' the Smartsheet will ignore the Excel setting and display '.' for decimal separators instead.

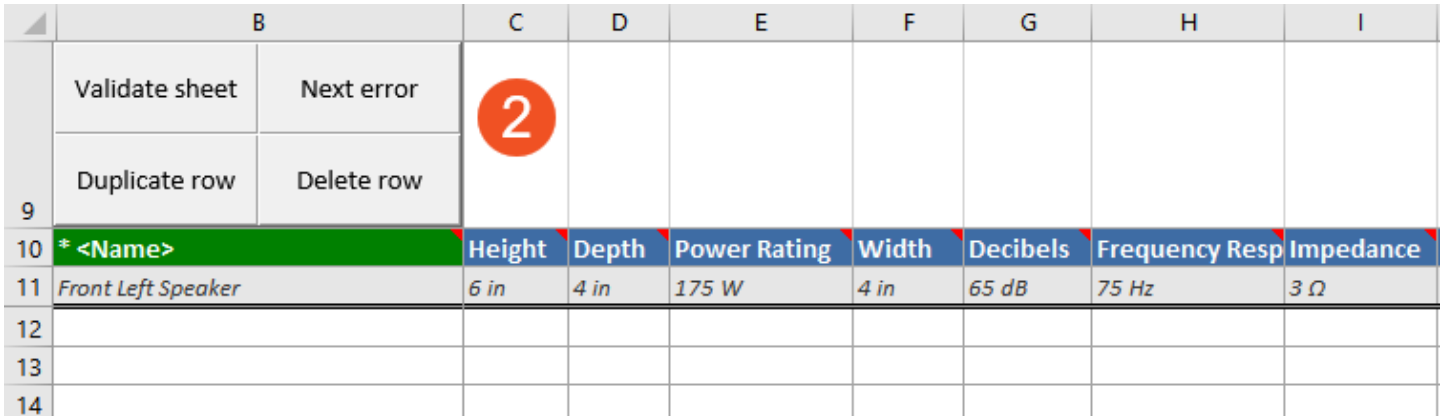
Smartsheet Template Product Row

A 'template product' row can be added to Smartsheet data and template exports, providing an additional level of guidance to suppliers and vendors who initiate and/or maintain product data in STEP. This row contains sample product data, which provides users with a visual example of how to correctly fill out attributes. It is read-only, frozen, and will be ignored for product validations and imports.

The following screenshots show how a template product row displays in a Smartsheet data export (1), which would be used to maintain existing products, and in a Smartsheet template export (2), which would be used to onboard new products.



	B	C	D	E	F	G	H	I	J
9	Validate sheet / Next error	1							
10	* <ID>	<Name>	Height	Depth	Width	Power Rating	Decibels	Frequency Re	Impedance
11	*Example*	*Front Left Speaker*	*6 in*	*4 in*	*4 in*	*175 W*	*65 dB*	*75 Hz*	*3 Ω*
12	267850	Front Left Speaker							
13	267851	Front Right Speaker							
14	267852	Rear Left Speaker							
15	267853	Center Speaker							
16	267856	Subwoofer							



	B		C	D	E	F	G	H	I
9	Validate sheet	Next error	2						
	Duplicate row	Delete row							
10	* <Name>		Height	Depth	Power Rating	Width	Decibels	Frequency Resp	Impedance
11	*Front Left Speaker*		*6 in*	*4 in*	*175 W*	*4 in*	*65 dB*	*75 Hz*	*3 Ω*
12									
13									
14									

Prerequisites

This topic presumes that users have knowledge of how to configure a Smartsheet Export Configuration and how to export Smartsheet data and Smartsheet templates. For more information, refer to the following topics:

- **Smartsheet Data and Template Configurations** in the 'Excel Smartsheet Format' section of the **Data Exchange** documentation

- **Smartsheet Export Widget** in the **Web User Interfaces** documentation. This widget is used to download Smartsheet template files.
- **Smartsheet Export Action** in the **Web User Interfaces** documentation. This widget is used to download Smartsheet data maintenance files.

Configuration

To add the template row, no configuration is needed within the Smartsheet Export Configuration or within the Web UI components that are used to export Smartsheets (Smartsheet Export Widget and Smartsheet Export Action). Instead, the row is included by referencing an 'example' product to the top level of the product category from which you want to export your Smartsheet. The reference type used to link the example product is the **Smartsheet Template Product** (SmartsheetTemplateProduct) product reference type.

When the Smartsheet is exported, the template product row is included if the exporter finds the Smartsheet Template Product reference. If not found, the row is not included.

The next two subsections explain the configuration of the 'example' product and the configuration of the Smartsheet Template Product reference type.

Note: The sample row will be unavailable if the user does not have the permissions to view the referenced products.

'Example' Product

The 'example' product used to hold the sample data for the template product row does not have to be a specially created product, but it must meet the following criteria:

- Contain valid example data, using the same attributes mapped in the Smartsheet Export Configuration.
- Have the same data structure (object types and levels) as the actual products that will be onboarded or enriched using the Smartsheet.

The product does not need to be located within the same product hierarchy / product category as the product(s) that will be used to export the Smartsheet data or template.

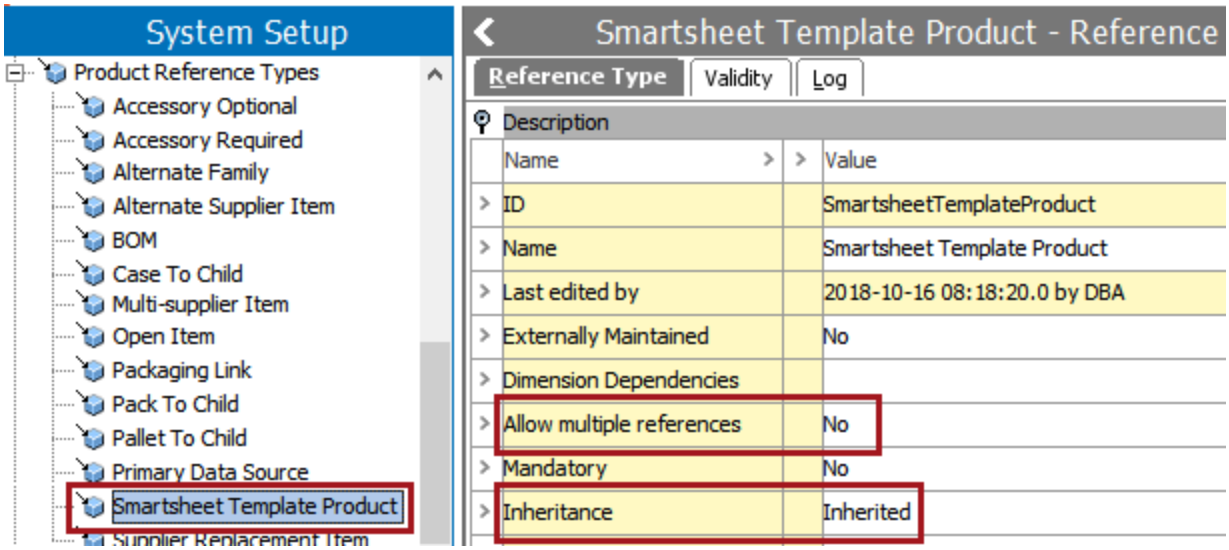
Smartsheet Template Product Reference

The Smartsheet Template Product (SmartsheetTemplateProduct) reference type is automatically included in STEP versions 9.1 and above.

By default, all product object types in the system are added as Valid Target Types to this reference type. Users are responsible for adding a product object type as the Valid Source Type.

The Smartsheet Template Product reference type is automatically configured as follows:

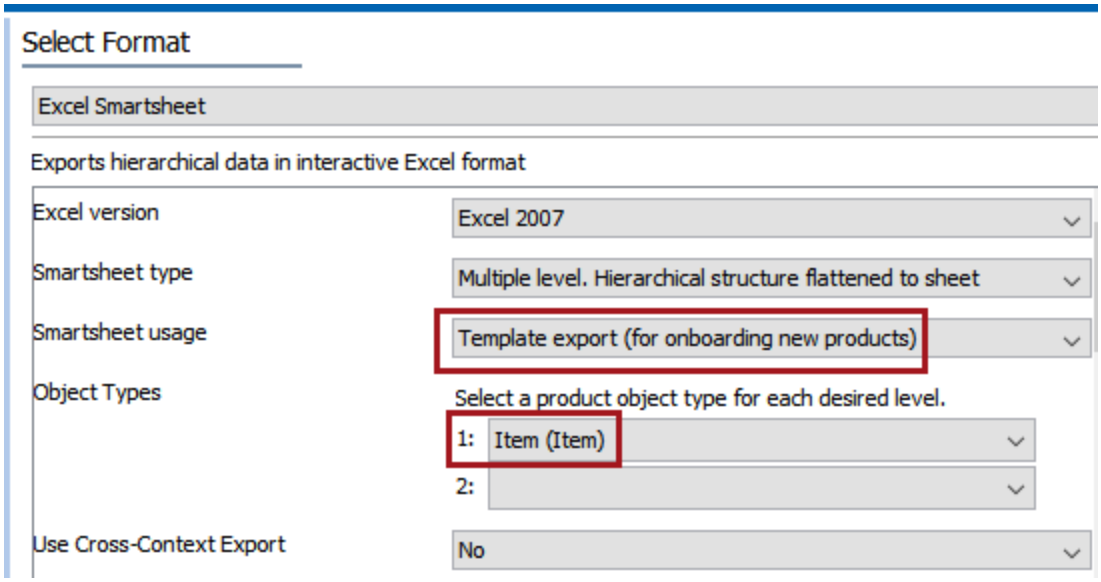
- **Allow multiple references** is set to **No** because only one source product should contain the example attribute values.
- **Inheritance** is set to **Inherited** because the example product values should be applicable to child products within the category.



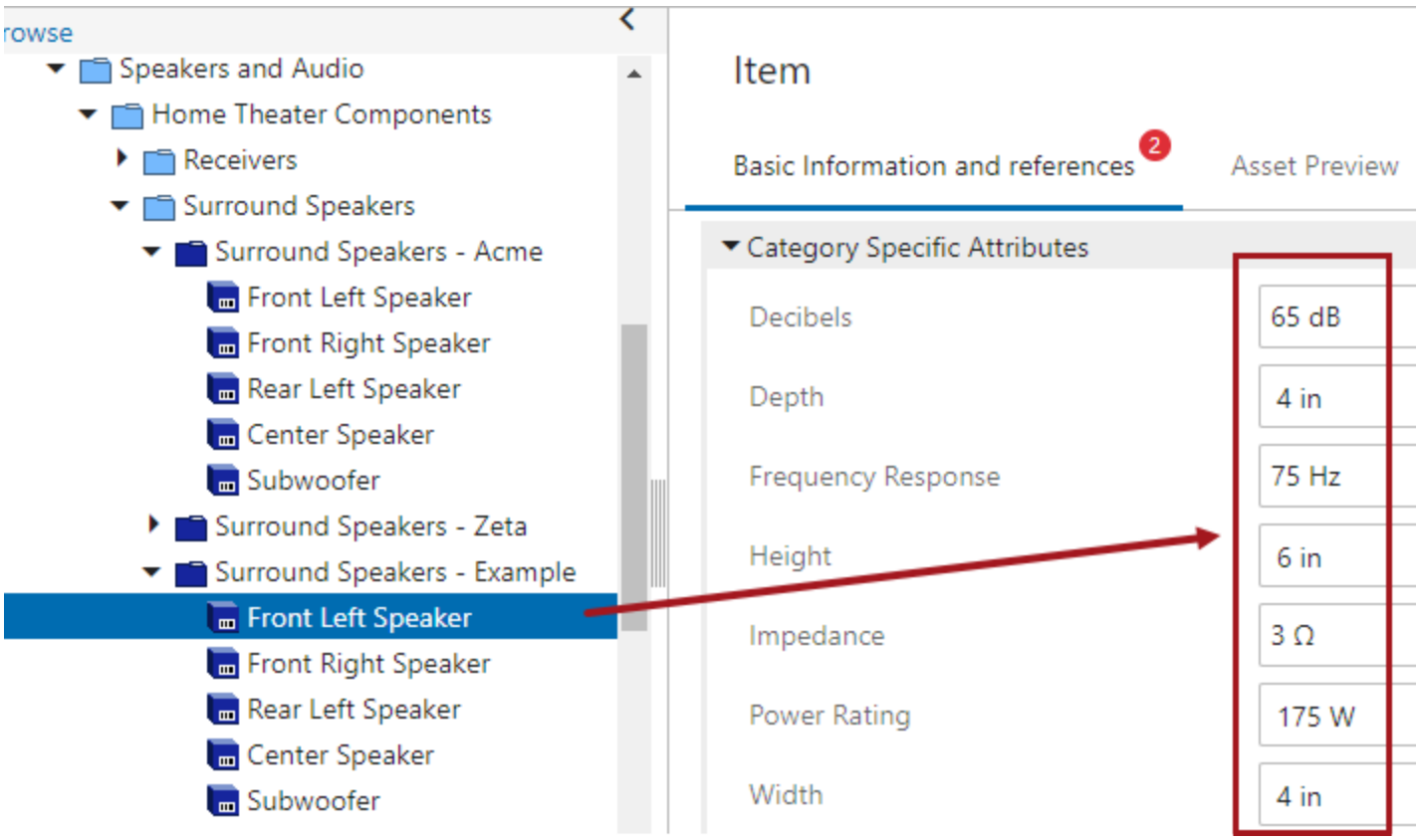
Example Setup

The following is a very basic setup that will generate a template product row on a Smartsheet template by using example data for one object type.

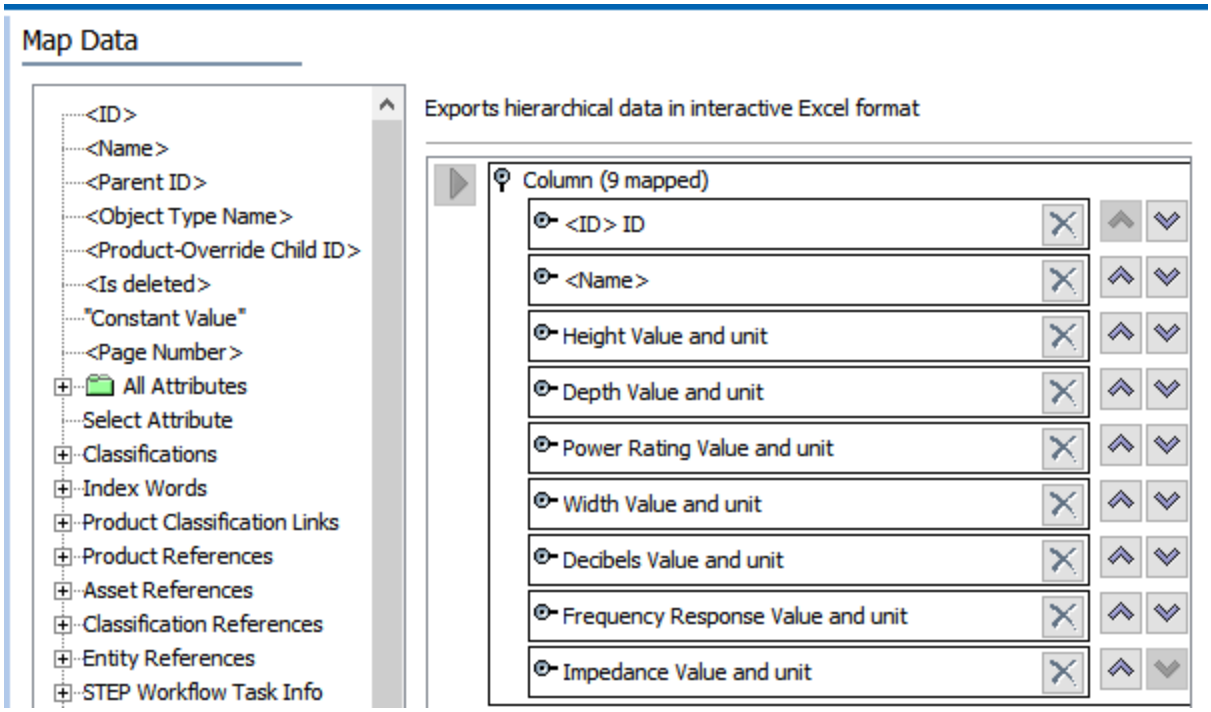
1. In this example, the Smartsheet Export Configuration is set to 'Template export' using the object type of 'Item.'



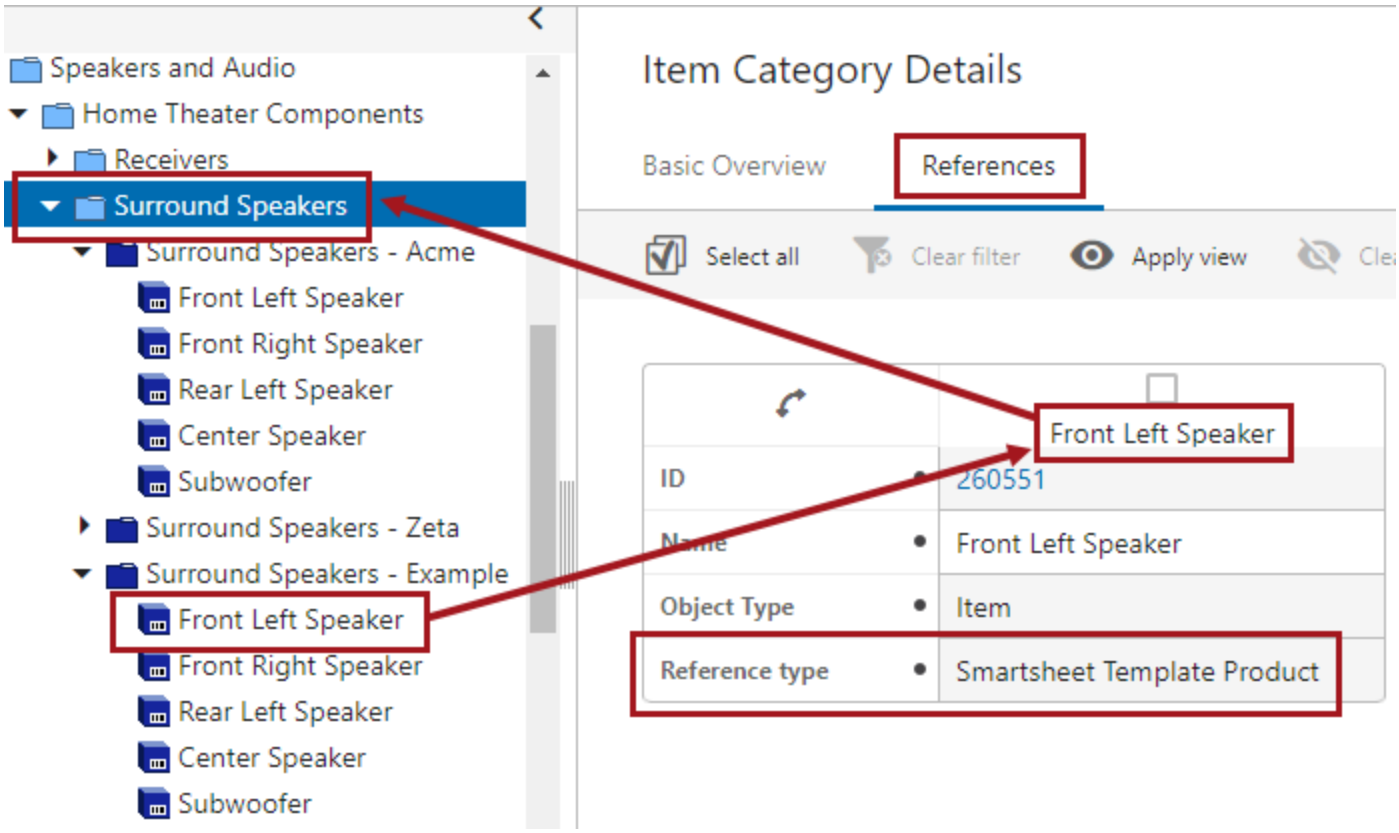
2. The example product is an Item object named 'Front Left Speaker.' It contains values for category-specific attributes such as Decibels, Frequency Response, Height, Width, and so forth.



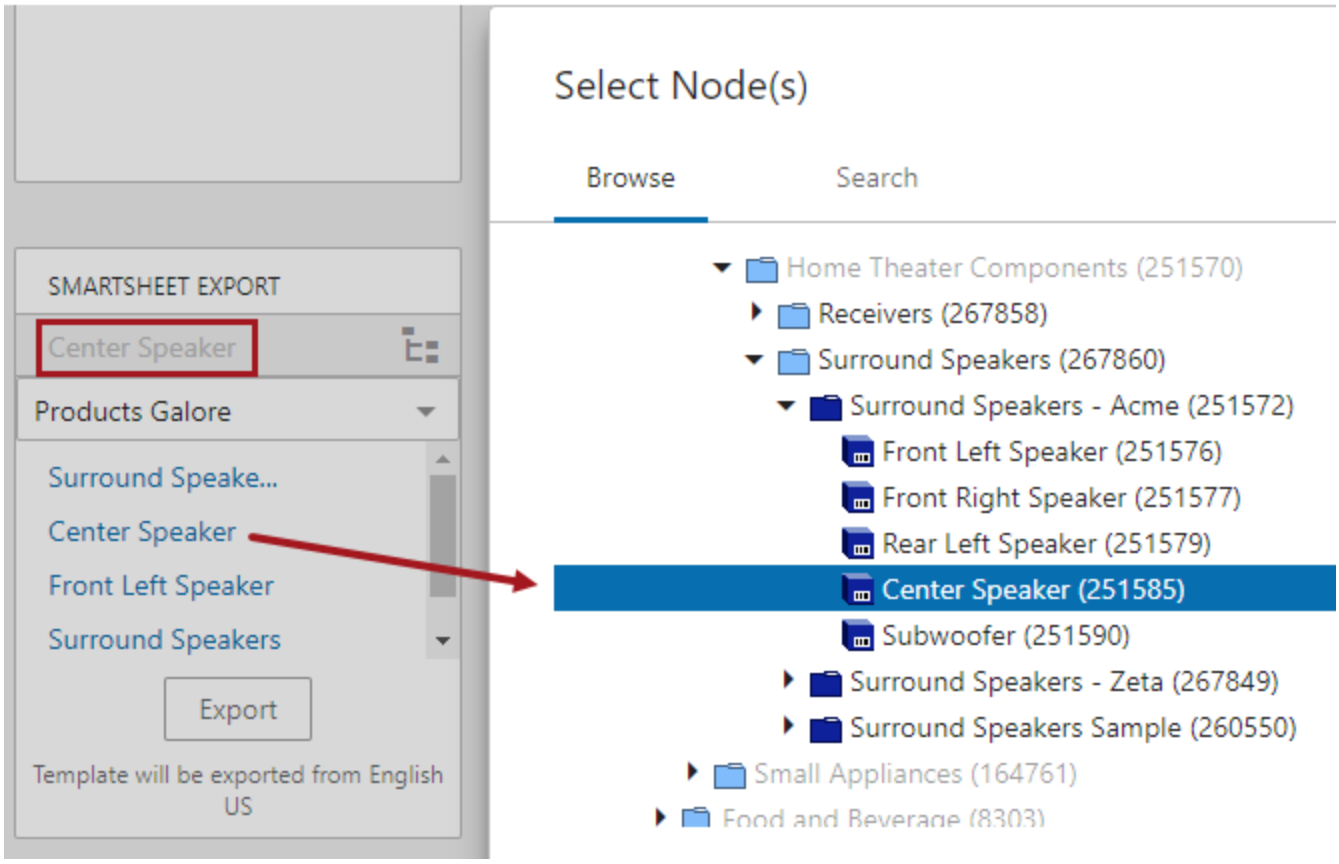
3. The same attributes have been mapped in the Smartsheet export configuration.



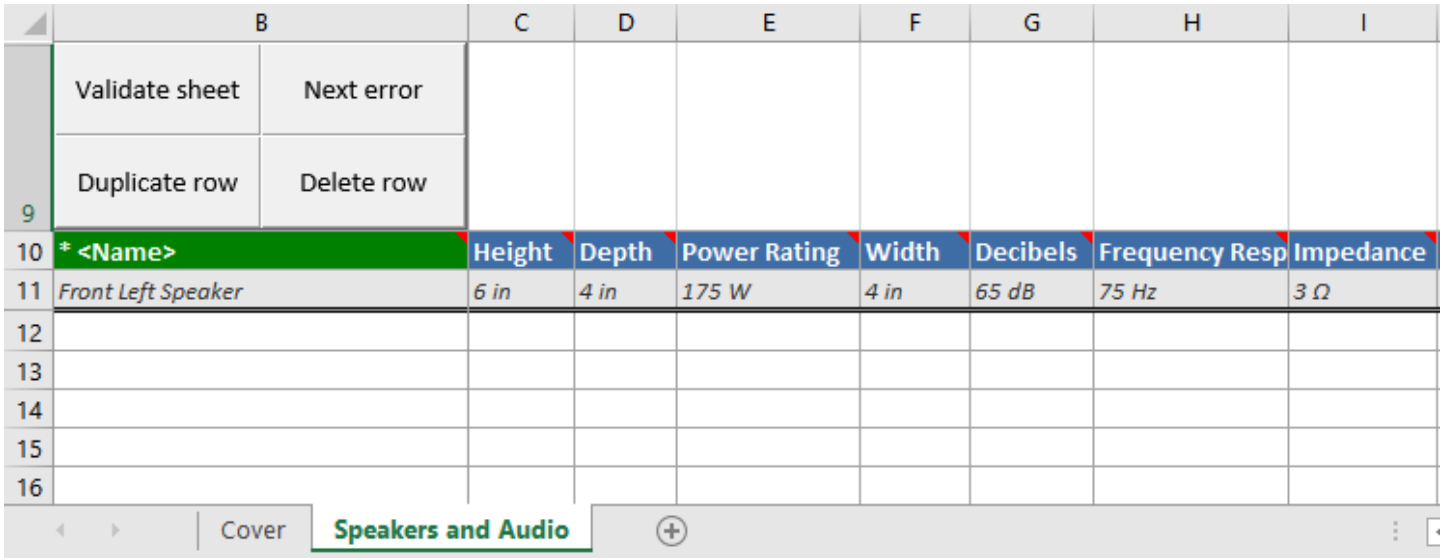
4. The 'Front Left Speaker' example product is referenced by the top-level of the product category ('Surround Speakers') using the Smartsheet Template Product reference.



5. In the Web UI, this Smartsheet template is exported from the Smartsheet Export Widget using the 'Center Speaker' object. 'Center Speaker' is the same object type (Item) as the referenced example product ('Front Left Speaker').



6. The resulting Smartsheet template contains a template product row with the sample information from the referenced 'Front Left Speaker' product.



FAB-DIS Format

FAB-DIS is an exchange format primarily used by companies doing business in France and facilitates the sharing of quality product information between manufacturers and distributors. A FAB-DIS file is delivered as an Excel file containing seven tabs. For more information on FAB-DIS, search the web.

In STEP, two tabs are supported: ETIM and MEDIA. Once exported, these tabs can be combined with the manually added additional five tabs, obtained from other sources, into one FAB-DIS file. For imports, data from the ETIM and MEDIA tabs is saved to STEP.

To access the 'FAB-DIS' option, the 'fabdis' component must be activated on your system. Contact your Stibo Systems representative for details.

Format Availability

FAB-DIS format is available for selection in:

- IIEP - refer to **Creating an Inbound Integration Endpoint**
- Export Manager - refer to **Creating a Data Export**
- OIEP - refer to **Creating an Outbound Integration Endpoint**

Mapping

This format requires creating a data map between STEP and the data being processed and may also include data transformations. For details, refer to **Data Mapping**.

For inbound mapping examples, refer to the **FAB-DIS Import Format** topic.

For outbound mapping examples, refer to the **FAB-DIS Export Format** topic.

Inbound Data

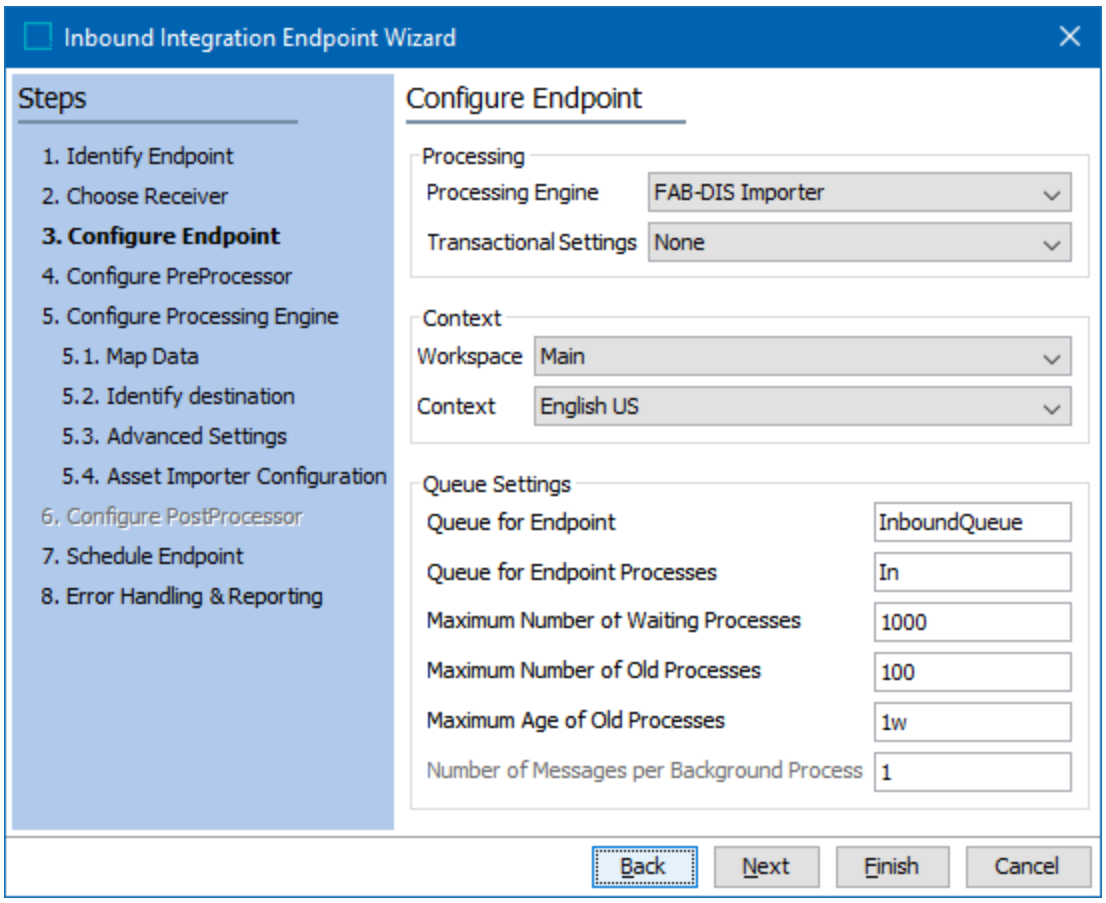
The FAB-DIS format is not available from Import Manager, but you can achieve the same result by manually running an IIEP using the FAB-DIS Importer processing engine.

Ultimately, Web UI provides the easiest FAB-DIS data import via the **File Loading Widget**. Web UI monitoring is available via the **Recent Background Processes Panel**. Information about both of these is included in the **Web User Interfaces** documentation.

IIEP

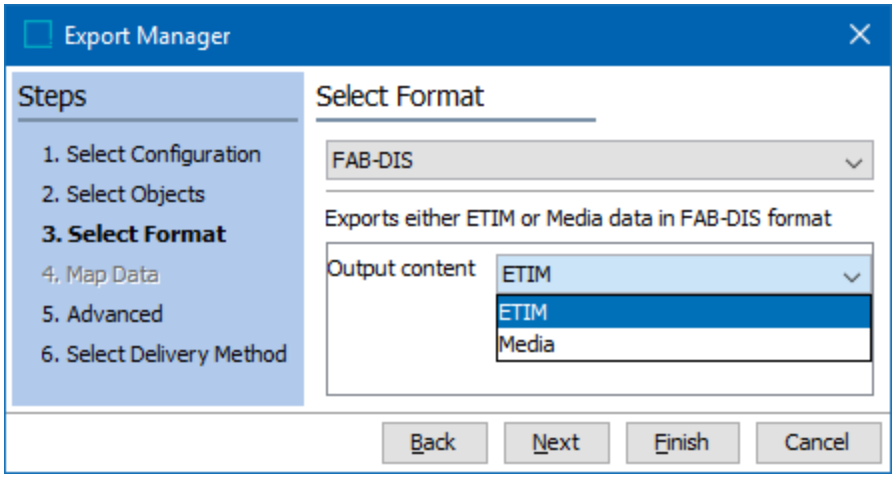
Minimum manual mapping is required when configuring the IIEP since the FAB-DIS Importer processing engine automatically maps the ETIM attributes included in the FAB-DIS import file.

Important: To import successfully, a FAB-DIS file must have only the expected columns, and no additional columns. For details, refer to the **Prerequisites** section of the **FAB-DIS Import Format** topic.

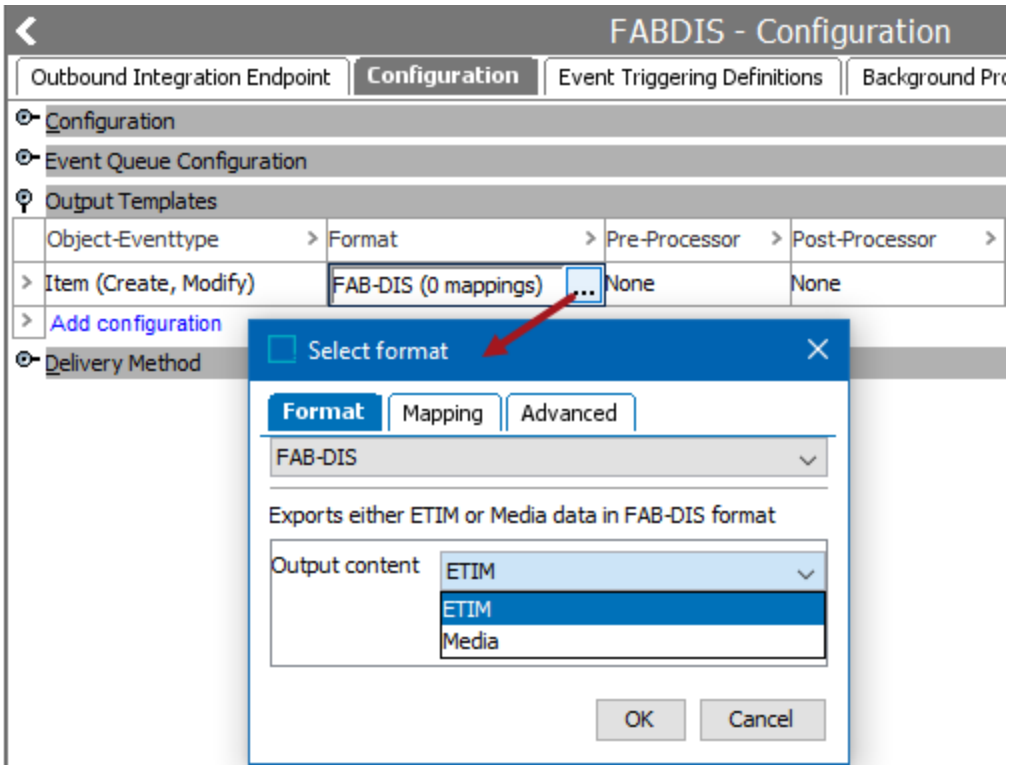


Outbound Data

Export Manager



OIEP


FABDIS - Configuration
Outbound Integration Endpoint
Configuration
Event Triggering Definitions
Background Pro
Configuration
Event Queue Configuration
Output Templates
Object-Eventtype
Format
Pre-Processor
Post-Processor
Item (Create, Modify)
FAB-DIS (0 mappings)
None
None
Add configuration
Delivery Method
Select format
Format
Mapping
Advanced
FAB-DIS
Exports either ETIM or Media data in FAB-DIS format
Output content
ETIM
ETIM
Media
OK
Cancel

FAB-DIS Export Format

The FAB-DIS export format allows users to assign STEP attribute values for products and assets to the fixed columns required by the format.

After a STEP user selects products to export, mapping is required to identify STEP data that should be output. Mapping is the same on the Select Format step of Export Manager, and on the Format column on the Output Templates flipper of an OIEP, as defined in **Outbound Map Data Options** topic of the **Data Mapping** section.

Mapping for the FAB-DIS ETIM and Media tabs is defined in these topics:

- FAB-DIS ETIM Export Mapping
- FAB-DIS Media Export Mapping

FAB-DIS ETIM Export Mapping

The ETIM tab of a FAB-DIS file has six (6) columns of data which are represented as mapping targets on the Map Data step of the Export Manager and OIEP.

In the output, a single product ID can be listed multiple times when:

- a single product includes multiple ETIM feature IDs. A row is output for each feature ID and STEP ID combination.
- an attribute has a validation base type of 'Number Range.' Two rows are exported, one for the minimum and one for the maximum value.
- an attribute has a validation base type of 'Number Range' and allows multiple values. A row is exported for each range, with the minimum and maximum values in the same cell.

Important: The examples and images below define one way to map FAB-DIS data. The actual mapping steps required for your data are determined by your data model. For assistance, contact your Stibo Systems account representative or technical support.

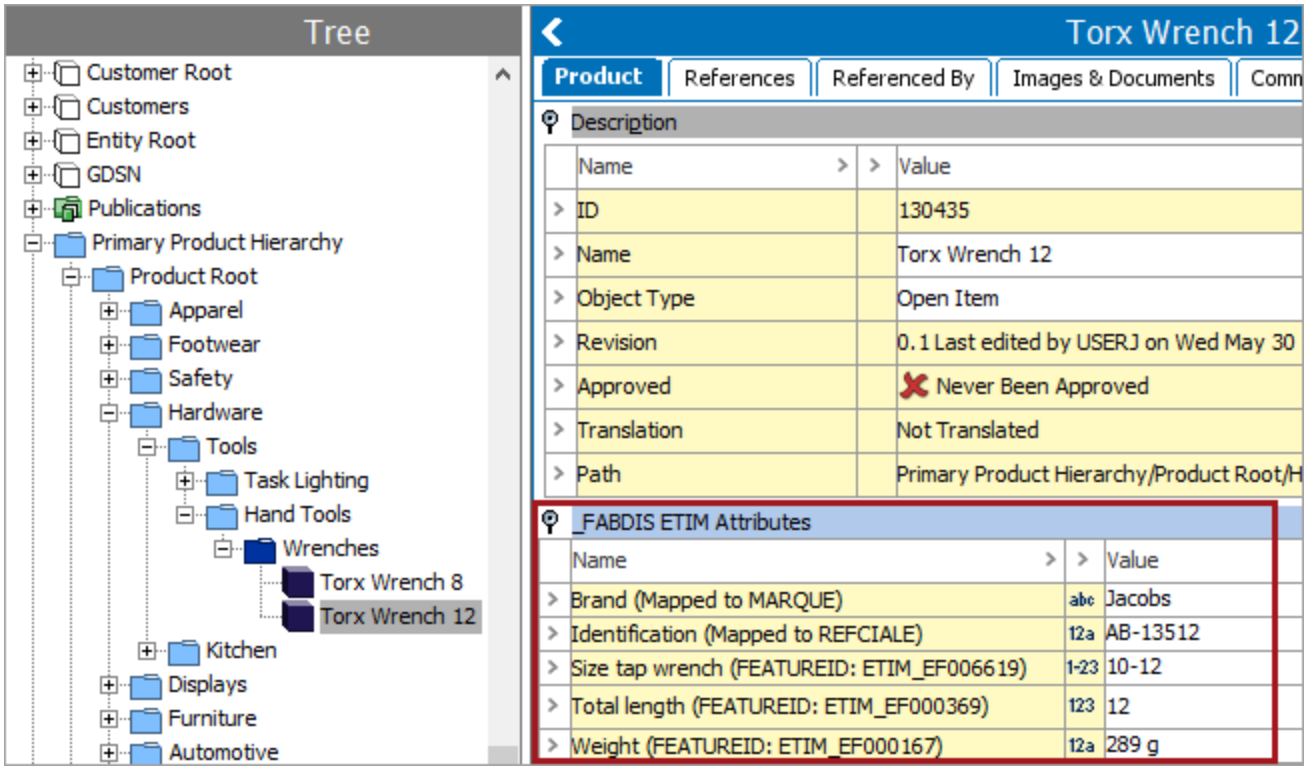
STEP Data for ETIM Mapping Targets

STEP data should be mapped to the mapping targets as follows:

1. MARQUE (text) is an attribute on the product that holds the brand name.
2. REFCIALE (text), based on your data model, can be the product STEP ID, or an attribute on the product that holds the ID. The REFCIALE is repeated in the output as many times as there are different ETIM features.
3. ARTCLASSID (text) is the ETIM class ID, which can be an attribute on the product or the classification link. For the classification link option, use the Multi Level References data source (and the Referenced node attributes option) to access an attribute on the classification that holds the ETIM ID, or optionally use a transformation to extract the ETIM ID directly from the STEP ID. For more information, refer to the **Multi Level References - Data Source Outbound** topic of the **Data Mapping** documentation and the **Transformations** topic within the **Resource Materials** online help.
4. FEATUREID (text) is the ETIM feature ID which can be mapped to the ETIM Attributes group.
5. FVALUE (text) is the ETIM feature value which can be mapped to the ETIM Attributes group.
6. ARTCLASSVERSION (text) is the ETIM class version, for example, 6.0. Use the Constant data source to map this text.

Sample Data

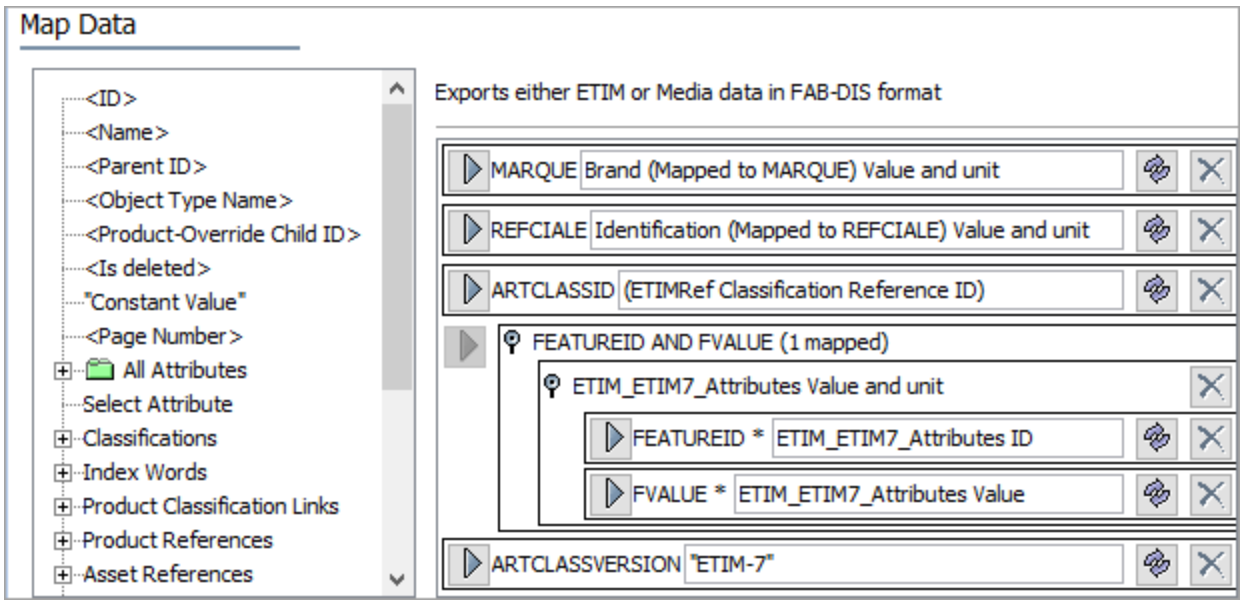
The following attributes are assigned to the '_FABDIS ETIM Attributes' group, and include values for the selected product. To clarify the mapping and output displayed within the images below, mapping information has been added to each attribute name in parentheses.



Mapping

The following mappings are displayed in this example:

- The MARQUE and REFCIALE targets are mapped to attributes on the product that contain the necessary data.
- The ARTCLASSID target is mapped to the classification reference, and a transformation is used to remove the 'ETIM_' prefix on the ID.
- The 'FEATUREID AND FVALUE' target is mapped to the 'ETIM_ETIM7_Attributes' group so that all attributes within the group with a value will be output. The individual FEATUREID target is manually updated to use the 'ID' aspect via the transformation button.
- The ARTCLASSVERSION target is mapped to the 'Constant Value' data source and the version is manually added.



Result

The following output is generated using the data and the map shown above and includes four rows for each product, where the FEATUREID indicates the ETIM attribute ID. The range attribute is displayed on two rows, indicated by the red box in the image below.

	A	B	C	D	E	F
1	MARQUE	REFCIALE	ARTCLASSID	FEATUREID	FVALUE	ARTCLASSVERSION
2	John+Williams	AB-13508	EC002214	ETIM_EF000369	78	ETIM-7
3	John+Williams	AB-13508	EC002214	ETIM_EF000167	226	ETIM-7
4	John+Williams	AB-13508	EC002214	ETIM_EF006619	8	ETIM-7
5	John+Williams	AB-13508	EC002214	ETIM_EF006619	10	ETIM-7
6	Jacobs	AB-13512	EC002214	ETIM_EF000369	12	ETIM-7
7	Jacobs	AB-13512	EC002214	ETIM_EF000167	289	ETIM-7
8	Jacobs	AB-13512	EC002214	ETIM_EF006619	10	ETIM-7
9	Jacobs	AB-13512	EC002214	ETIM_EF006619	12	ETIM-7

FAB-DIS Media Export Mapping

The MEDIA tab of a FAB-DIS file has eight (8) columns of data which are represented as mapping targets on the Map Data step of the Export Manager and OIEP.

For assets, attributes can be stored on:

- an 'Images and Document References' type by adding Valid Attributes on the reference type itself, as defined in the **Metadata Attributes on Reference and Link Types** topic of the **System Setup** documentation.
- the asset itself by making description attributes valid for the asset object type, as defined in the **Linking Description Attributes to an Object Type** section of the **Description Attributes** topic of the **System Setup** documentation.

In the output, a single product ID is listed multiple times when multiple product-to-asset references exist.

Important: The examples and images below define one way to map FAB-DIS data. The actual mapping steps required for your data are determined by your data model. For assistance, contact your Stibo Systems account representative or technical support.

Media Mapping Targets

STEP data should be mapped to the mapping targets as follows:

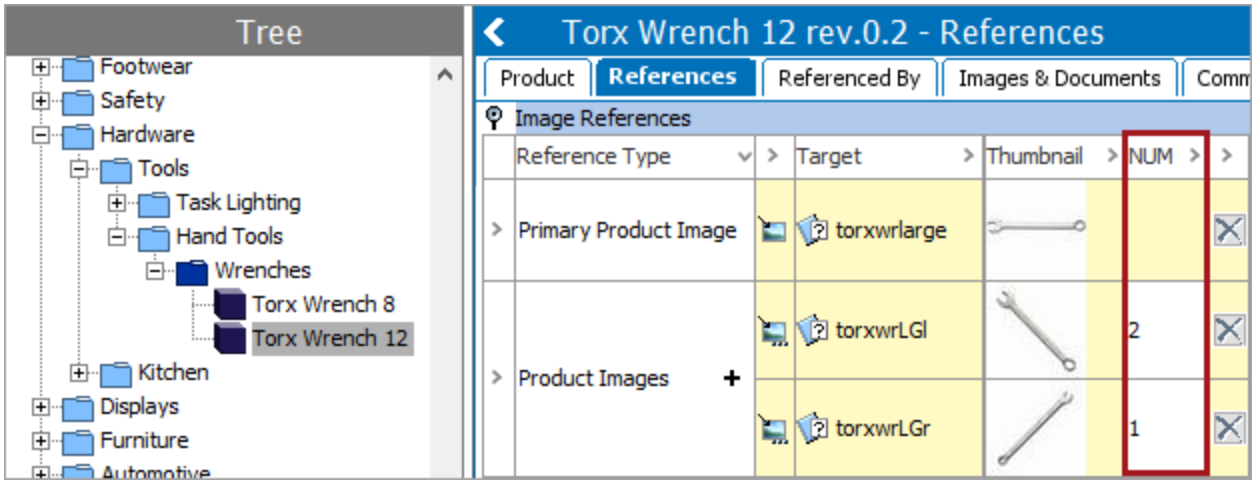
1. MARQUE (text) is an attribute that holds the brand name.
2. REFCIALE (text), based on your data model, can be the asset STEP ID, or an attribute on the asset that holds the ID. The REFCIALE is repeated in the output as many times as there are different pieces of media.
3. TYPM (text) is the media type of the asset as defined in an attribute, as an object type, or the reference type. One option to access this data is to use the Multi Level References data source (and the Referenced node attributes option), as defined in the **Multi Level References - Data Source Outbound** topic of the **Data Mapping** documentation.

   Note: The output for the TYPM column must match one of the 12 valid descriptions allowed by FAB-DIS. For example, low-definition photo = PHOTOBD, high-definition photo = PHOTOHD, and video = VIDEO. Search 'FAB-DIS' on the web for requirements details.

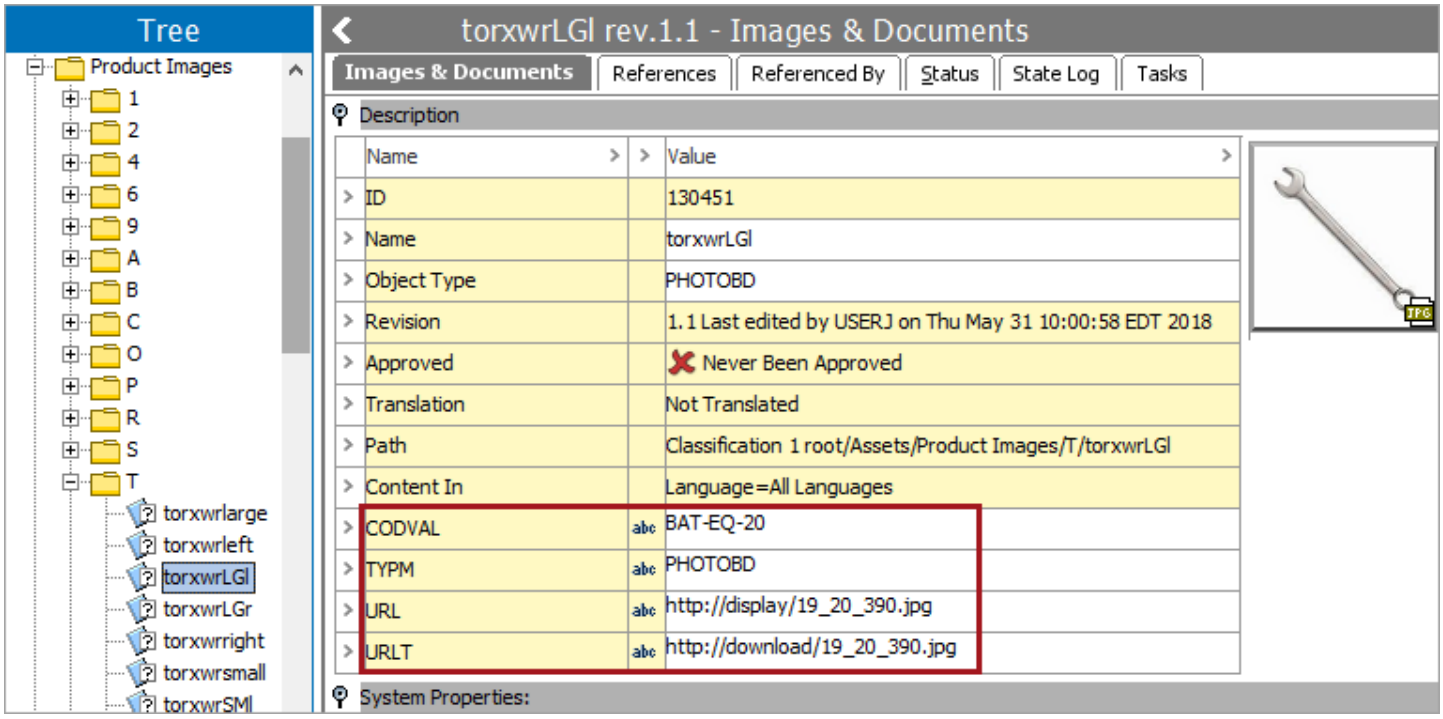
4. NUM (numeric) is the order number of a feature that is used for sorting. This field is required and must be a whole number. For example, if there are two photos with the same TYPM for the same product, the first will have NUM = 1 and the second will have NUM = 2. Use metadata on the asset reference to supply this information.
5. CODVAL (text) is the certificate code, keyword value, or sales pitch. This is an attribute on the asset itself.
6. NOM (text) is the name of the document or photo. Use either the name of the asset or an attribute on the asset itself.
7. URL (text) is the URL to display the document or photo on a merchant site. This is an attribute, possibly a calculated attribute, on the asset itself.
8. URLT (text) is the URL to download the document or photo. This is an attribute on the asset itself.

Sample Data

The NUM value is stored as metadata on the product References tab.



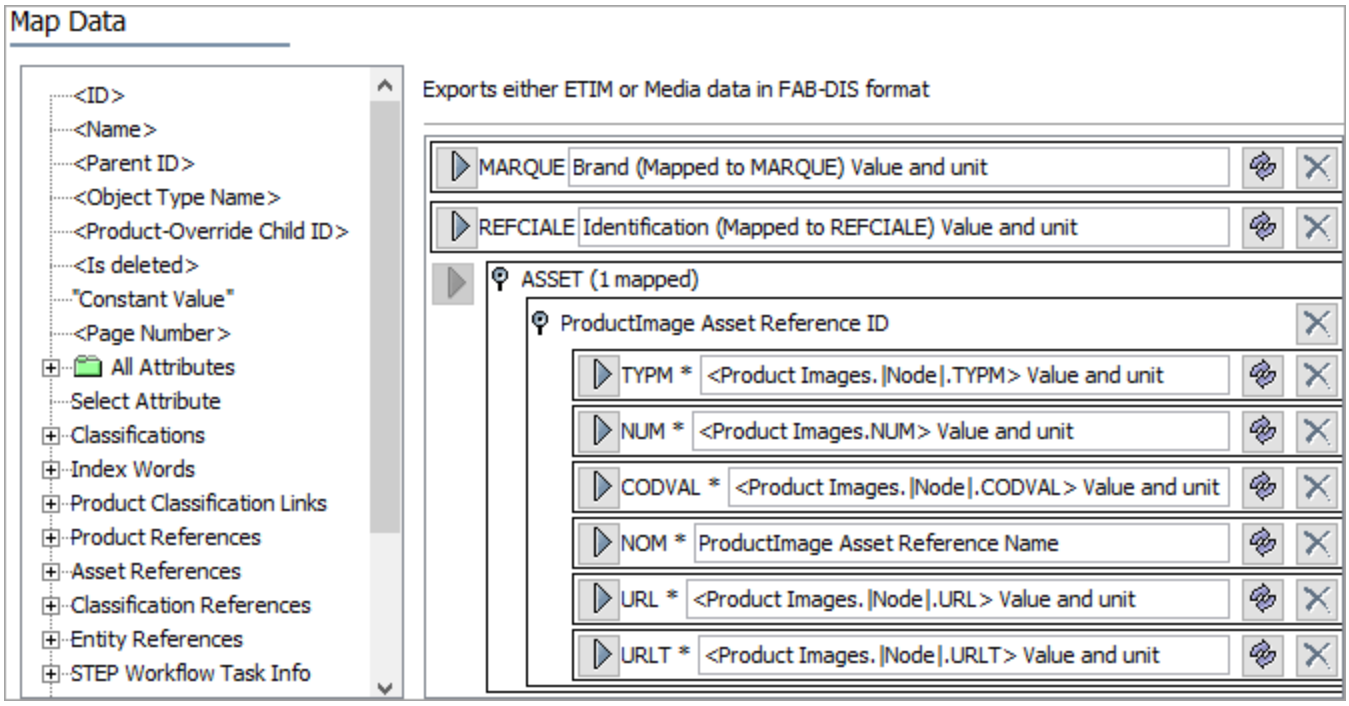
The remaining values are stored as metadata on the asset, as shown below for the Product Images reference with NUM = 2.



Mapping

The following mapping are displayed in this example:

- The MARQUE and REFCIALE targets are mapped to attributes on the product that contain the necessary data.
- The NUM target is mapped to a metadata attribute on the asset reference.
- The CODVAL, NOM, TYPM, URL and URLT targets are mapped to attributes on the asset itself.



Result

The following output is generated using the data and the map shown above. The FAB-DIS Media file includes two rows for each product because each product has two 'Product Images' references (as shown below).

	A	B	C	D	E	F	G	H
1	MARQUE	REFCIALE	TYPM	NUM	CODVAL	NOM	URL	URLT
2	John+Williams	AB-13508	PHOTOHD	1	BAT-EQ-72	torxwrSMl	http://display/19_72_902.jpg	http://download19_72_902.jpg
3	John+Williams	AB-13508	PHOTOHD	2	BAT-EQ-89	torxwrSMr	http://display/19_89_928.jpg	http://download19_89_928.jpg
4	Jacobs	AB-13512	PHOTOBD	2	BAT-EQ-20	torxwrLGl	http://display/19_20_390.jpg	http://download/19_20_390.jpg
5	Jacobs	AB-13512	PHOTOBD	1	BAT-EQ-35	torxwrLGr	http://display/19_35_392.jpg	http://download19_35_392.jpg

FAB-DIS Import Format

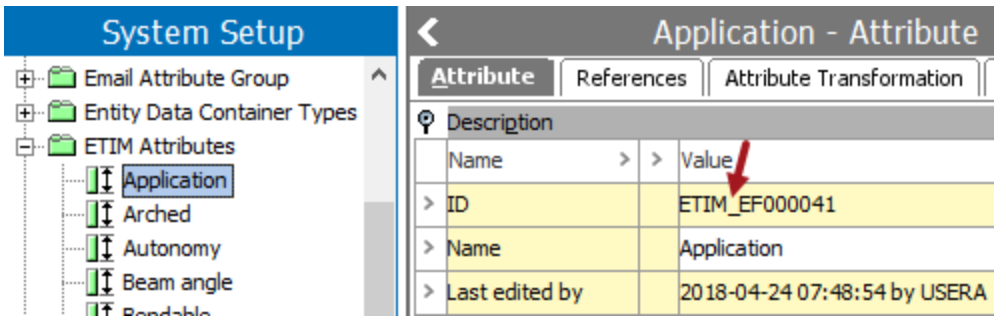
Importing FAB-DIS data from the ETIM and Media tabs of an Excel file is available in an IIEP via the FAB-DIS Importer processing engine. Other tabs in the FAB-DIS file are ignored by the importer.

Prerequisites

The following items are necessary to successfully import FAB-DIS from workbench and from Web UI:

- The IIEP must use a hotfolder receiver.
- The file must be Excel version 2007 (or later), with an .XLSX file extension. All other file formats return an error.
- On the application server, the case-sensitive property **AssetImporter.ObjectTypeChange** must be set to 'true' in the sharedconfig.properties file. This property allows the asset importer to change the object type of the assets (unless it has been set manually), such as when an asset placeholder is updated with content. If not set, FAB-DIS imports will not work.
- The data can be processed using any of these configurations, all of which can also be zipped:
  - Excel file without asset files
  - Excel file and separate asset files, in a single folder
  - Excel file that includes URLs to assets
- For the ETIM and Media tabs, the file must contain only the columns required by the format, and use only the headers required by the format. While the order of the columns is not important, extra data columns, blank columns, and missing columns will result in an import error.
- The file must contain values for the MARQUE and REFCIALE columns for every row of data. Missing values result in an error.
- The file should not contain any special formatting, specifically on the ETIM tab when LOV values are being used, since they are case-sensitive in STEP.
- The file should not contain data for multi-valued LOVs as they are not supported.
- For the ETIM tab import, the FAB-DIS importer processing engine uses automatic mapping for the ETIM attribute values included in the import file. This requires that ETIM data has previously been imported using STEP version 9.0 or later.

  To verify your STEP system has the correct ETIM installed, in System Setup, open an ETIM attribute group and select any attribute. The attribute ID must begin with the 'ETIM_' prefix, specifically, the 'ETIM' text must be followed by an underscore (_). The attribute group and classification IDs may include an ETIM version number.



For example, ETIM_EF000041 is the ID of an ETIM attribute in STEP 9.0. The same attribute in an earlier version might have had the ID 'ETIMEF000041' where the underscore is missing.

- For the Media tab import, asset files names are case-sensitive. When using the NOM column to indicate asset file names that are included in the .ZIP file, an error is returned if the asset file name does not match the case of the file name in the Excel file.

FAB-DIS Data File

STEP can process data from the ETIM tab or the Media tab of a FAB-DIS file.

On the **FAB-DIS ETIM** tab, a single product ID can be listed multiple times when:

- a single product includes multiple ETIM feature IDs. A row is included for each feature ID and STEP ID combination.
- an attribute has a validation base type of 'Number Range.' Two rows are included, one for the minimum and one for the maximum value.

On the **FAB-DIS Media** tab, a single product ID can be listed multiple times when a single REFCIALE includes references to multiple assets or media-related entries.

To import new or updated assets, a value should be supplied for one or more of the following columns: URLT, URL, or NOM. Processing is successful based on access to the asset, using the following priority:

- URLT - via freely accessible URL (no password required)
- URL - via freely accessible URL (no password required)
- NOM - via file name of asset within the download folder

If the assets cannot be accessed, and the 'Allow assets without content' option is active, asset placeholders are created or updated.

Note: Retrieving assets from URLs is likely to increase the time required for the import.

Configuration

Review the following topics for information on required configurations to import FAB-DIS format data and monitor the progress of an import.

- FAB-DIS Import Workbench Configuration
- FAB-DIS Import Mapping
- FAB-DIS Import Web UI Configuration
- FAB-DIS Import Background Processes

For an example of FAB-DIS data processed by an IIEP, refer to the **FAB-DIS Import Example** topic.

FAB-DIS Import Workbench Configuration

Importing FAB-DIS data from the ETIM and Media tabs of an Excel file is available in an IIEP via the FAB-DIS Importer processing engine. For more information, refer to the **FAB-DIS Import Format** topic.

Important: The workbench configuration covered in this topic is required for FAB-DIS data imports from the workbench or Web UI. After completing this configuration, to import FAB-DIS using the Web UI, refer to the **FAB-DIS Import Web UI Configuration** topic.

In order to import data from both the ETIM and Media tabs of a FAB-DIS file, both an Asset Importer configuration and an IIEP configuration are required. The Asset Importer must be configured first as it must subsequently be selected within the IIEP configuration.

Below, details are provided for how to configure both importers for use with the FAB-DIS format. General configuration information for Asset Importer and IIEPs is covered within the dedicated documentation sections for those importers and only information specific to the FAB-DIS configuration is covered in this topic.

Prerequisites

On the application server, set the case-sensitive `AssetImporter.ObjectTypeChange` property to 'true' in the sharedconfig.properties file. This property allows the asset importer to change the object type of the assets, such as when asset placeholder is updated with content. If set to 'false' or set at all, FAB-DIS imports will not work. For example:

```
AssetImporter.ObjectTypeChange=true
```

Note: If an asset's object type has been set manually, the object type is not updated during import, regardless of this setting.

Asset Importer Configuration

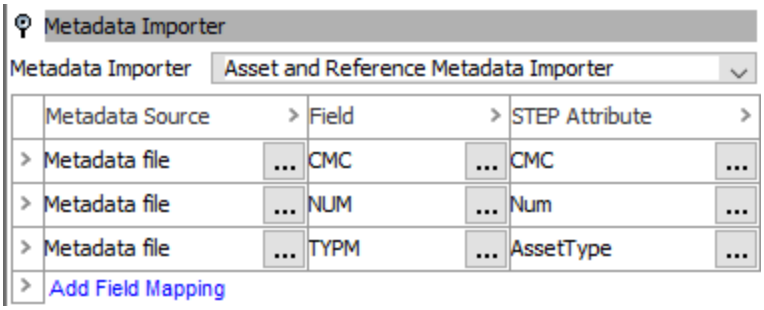
Asset Importer functionality is used to handle assets from the FAB-DIS Media tab. Media tab data triggers an asset importer background process and therefore, an asset importer configuration is required.

Note: This configuration is not necessary to import data from the ETIM tab only.

Follow the steps below to configure the asset importer, using the additional **Asset Importer** topics in the **Digital Assets** documentation.

1. Create an Asset Importer Configuration as defined in the **Asset Importer Configuration** topic.
2. For the **Importer Validator**, this option is not required and works best when the 'Allow asset with no content' option in the IIEP is left unchecked. Setting validations can adversely affect the use of the 'Allow asset with no content' option since assets without content will fail any specified validation. For more information, refer to the **IIEP Configuration** section below.

3. For the **Hierarchy Builder**, set the parameters based on your data model, as defined in the **Hierarchy Builder** topic.
4. For the **Asset Matcher**, set the parameters based on your data mode, as defined in the **Asset Matcher** topic:
   - Specifically, for the 'Match on' parameter, from the dropdown select 'Asset Name' to ensure that assets are updated, and not duplicated.
5. For the **Content Importer**, set the parameters based on your data model, as defined in the **Content Importer** topic.
6. For the **Metadata Importer**, if metadata is required, set the following parameters as defined in the **Metadata Importer** topic. For more information on metadata mapping and handling CODVAL entries, refer to the **FAB-DIS Import Mapping** topic.
   - For the 'Metadata Importer' dropdown, select **Asset and Reference Metadata Importer** to map data to the asset or to the reference between the asset and the product. When the mapped attribute is valid on the reference, the metadata is written on the reference. If the attribute is only valid on the asset, the metadata is written to the asset. If the attribute is valid on both the reference and the asset, the metadata is only written to the reference.



   When multiple products with different metadata reference the same asset, the import writes the same metadata to each reference.

   - Click the 'Add Field Mapping' link to display the Mappings dialog. Add the necessary mappings between the Field parameter (indicating the name of the column in the FAB-DIS file) and the STEP Attribute parameter. In the table, click the ellipsis button (...) to display the dialog for editing the mapped selections.
7. For the **Product Linker**, set the following parameters as defined in the **Product Linker** topic:
   - For 'Match Metadata Field' parameter, type 'ID'. While 'ID' is not a column in the import file, it is used to determine the correct product for the asset by carrying forward the ID mapping (including any transformations) specified for the ETIM tab in the IIEP configuration.
   - For 'Match On' parameter, select ID from the dropdown, unless the Excel file includes a name or key.
   - For 'Match Below Product Root' parameter, click the ellipsis button (...) to select a location in Tree to search for existing products.
   - For the 'Reference Type Field' and 'Default Reference Type' parameters, determine the necessary option based on your data model:
     - To match the value provided in the TYPM column of the import file to the ID of a reference type in STEP, add TYPM to the 'Reference Type Field' parameter. Additionally, supply a reference ID in the

'Default Reference Type' parameter to indicate the required default reference type when the value in the TYPM column does not match a STEP reference ID.

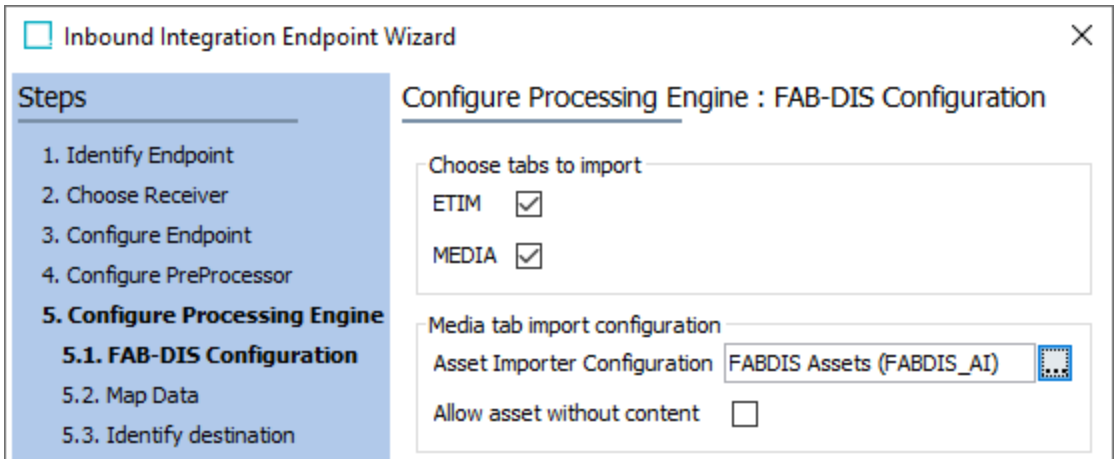
- To set the same reference type for all assets in the import file, leave the 'Reference Type Field' parameter blank, and set the 'Default Reference Type' parameter as the required reference type.

8. Set other parameters on the **Asset Importer Configuration Type** tab as required and as defined in the **Asset Importer Configuration** section of the **Digital Assets** documentation.

IIEP Configuration

The following specific inbound integration endpoint (IIEP) settings are required to import a FAB-DIS file. Additional topics can be found in the **Inbound Integration Endpoints** documentation.

1. Create a FAB-DIS IIEP. For details, refer to the **Creating an Inbound Integration Endpoint** topic.
2. On the **Choose Receiver** step, the Receiver parameter must be set to Hotfolder Receiver for importing assets or importing via the Web UI File Loading widget. Otherwise, select any receiver. If imports will be performed in Web UI and will also be performed using different receiver methods, create an IIEP for each receiver. For more information, refer to the **IIEP - Choose Receiver** topic.
3. On the **Configure Endpoint** step, select FAB-DIS Importer as the Processing Engine. This gives access to the FAB-DIS Configuration step below.
4. On the **FAB-DIS Configuration** step, select the data that will be imported:



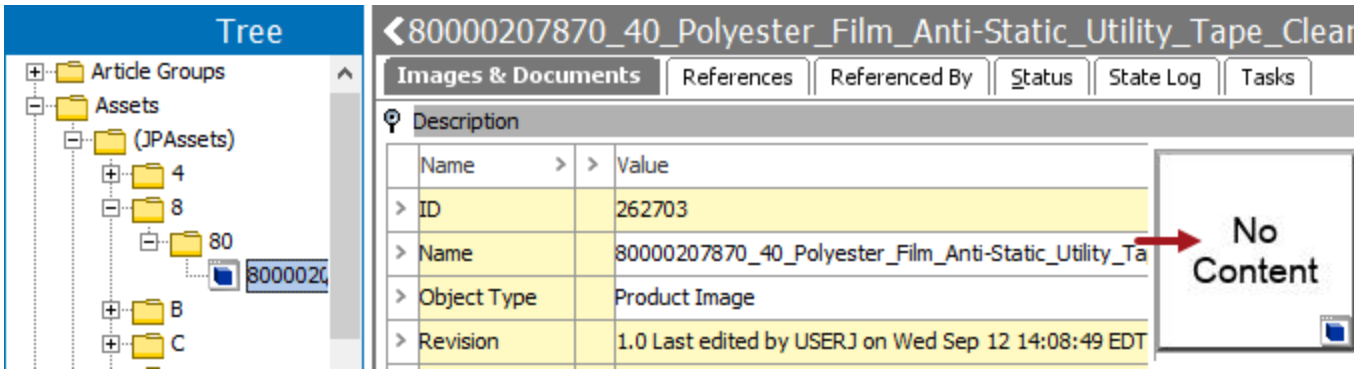
Note: To enable the **Finish** button, at least one checkbox must be checked on the FAB-DIS Configuration step, and one Source field must be mapped on the Map Data step.

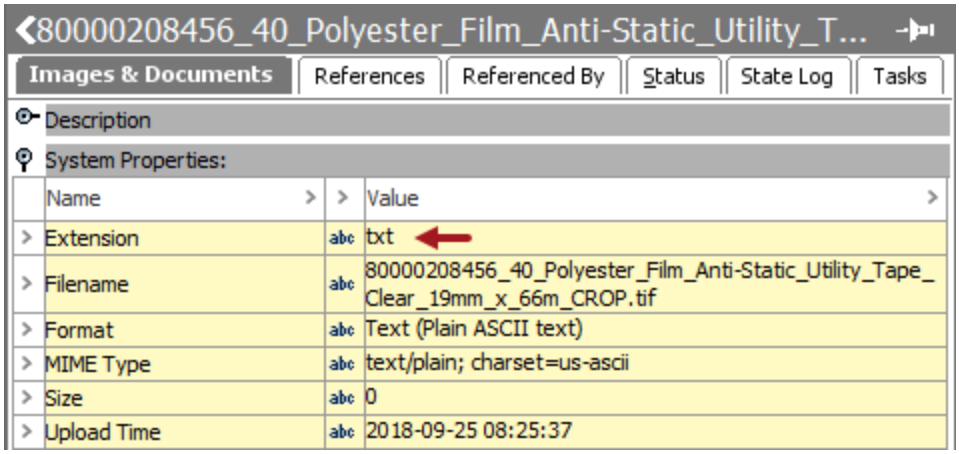
- Check the **ETIM** checkbox if data from the FAB-DIS ETIM tab should be processed.
- Check the **MEDIA** checkbox if data from the FAB-DIS Media tab should be processed. New products identified on the Media tab only, based on the REFCIALE column, are created in order to connect them to the imported assets, but they will only include the data mapped to the MARQUE and REFCIALE columns.

This selection displays the 'Media tab import configuration' group. For the Asset Importer Configuration parameter, click the ellipsis button (...) to display the Select Asset Importer Configuration dialog. Choose a configuration and click the 'Select' button. The configuration is displayed in the IIEP wizard.

Note: When the MEDIA checkbox is checked, an 'Asset Importer Configuration' must be selected to enable the **Next** button.

Check the **Allow asset without content** checkbox to create an *asset placeholder* when asset content is not available for import. These placeholder assets are created with the file name found in the Excel file as the STEP asset name, and include a link to the product identified in the Excel file, as well as any mapped metadata. As shown in the images below, a placeholder Object Type is set, and the asset thumbnail shows 'no content' in the 'Description' flipper. The System Properties flipper 'Extension' parameter is set with the default '.txt' extension. If the asset importer is configured to match on Asset Name, when an asset with content and the same name is imported, the object type and extension parameters are updated to reflect the actual asset information.



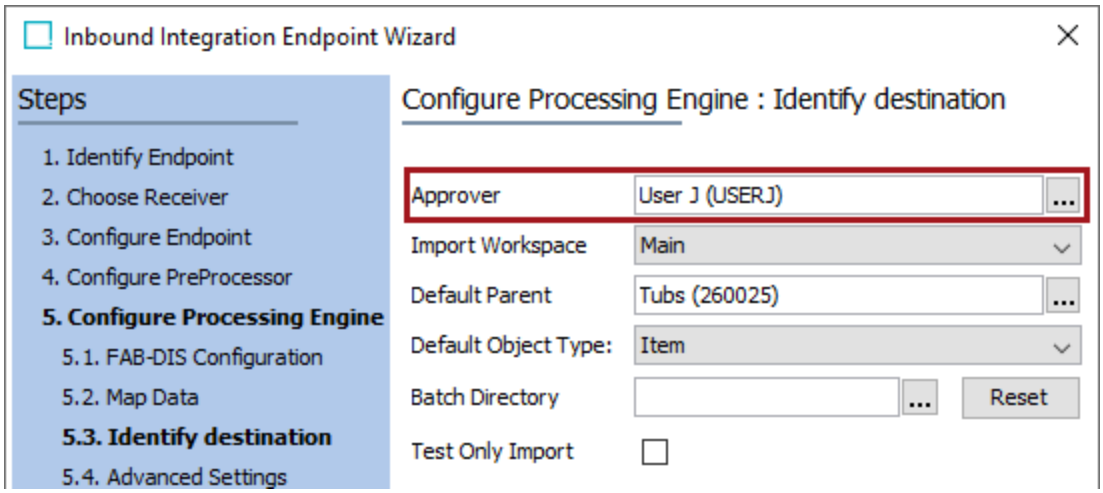


Considerations

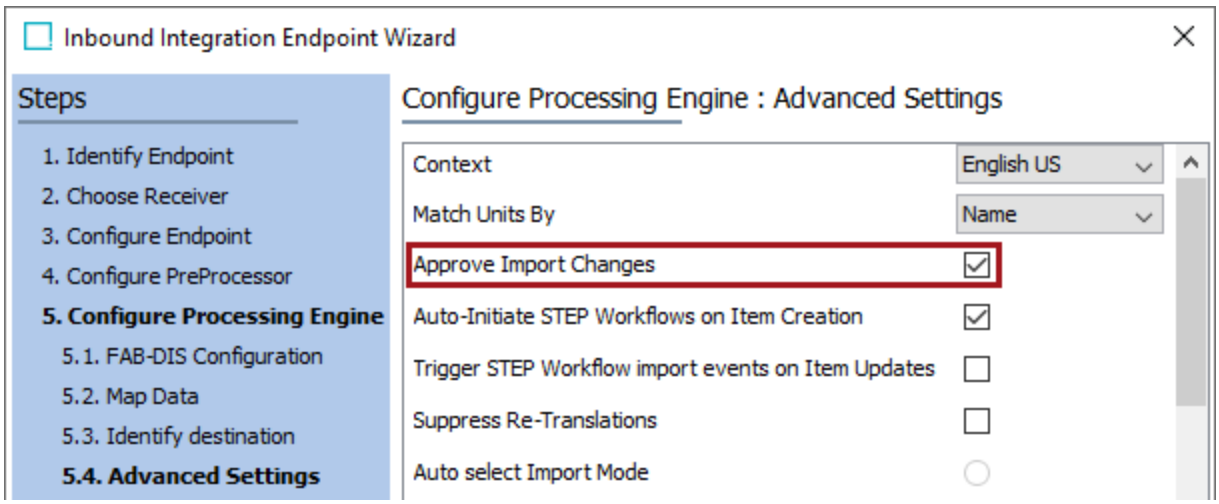
For the following scenarios, it is advised to use only one of these options, not both:

- Checking the 'Allow asset without content' checkbox while the asset importer Asset Matcher is configured to match on Asset Name can result in the following behavior: if the an asset *with* content exists in STEP, and then an asset with the same name but *without* content is imported, the existing asset content is removed.
- Checking the 'Allow asset without content' checkbox while the asset importer Import Validator has restrictions will case the import to return an execution error when no content is found.

5. On the **Map Data** step, complete the manual mapping instructions as needed based on the **FAB-DIS Import Mapping** topic. Complete mapping of FAB-DIS includes manual IIEP mapping, automatic ETIM attribute value mapping, and Media mapping via the asset importer configuration.
6. On the **Identify Destination** step, set any of the parameters needed. For more information, refer to the **IIEP - Identify Endpoint** topic.
   - On the **Default Parent** parameter, set the Tree location where imported products belong. This information is not available in the FAB-DIS file.
   - On the **Default Object Type** parameter, set the object type for new products. This information is not available in the FAB-DIS file.
   - On the **Approver** parameter, select a user if inbound objects should be approved upon successful import. The selected user must have privileges to approve new and modified objects.



7. On the **Advanced Settings** step, check the 'Approve Import Changes' checkbox. If this checkbox is not checked, objects are not approved upon import, even if an approver is selected. These are the same options that are available in Import Manager. For more information, refer to the **Import Manager - Advanced Settings** topic.



8. Change parameters on other steps as required, as defined in the **Creating an Inbound Integration Endpoint** topic.
9. Once a FAB-DIS import file is available in the IIEP receiver location, and the IIEP is invoked (on schedule or manually), FAB-DIS products and assets are created or updated in STEP. For more information, refer to the **Running an Inbound Integration Endpoint** topic.

FAB-DIS Import Mapping

When configuring an IIEP to import FAB-DIS data from the ETIM or Media tab of a FAB-DIS Excel file, review the information and prerequisites on the **FAB-DIS Import Format** topic.

Mapping for FAB-DIS allows data in the FAB-DIS Excel file to be associated with attributes in STEP. It is an intermediate step in the overall required FAB-DIS configuration as defined in the **FAB-DIS Import Workbench Configuration** topic.

The ETIM tab of a FAB-DIS file has six columns of data. Four columns are represented as mapping sources on the Map Data step of the IIEP. The remaining two columns are automatically mapped by the processing engine.

The Media tab of a FAB-DIS file has eight columns of data. Two columns are always represented as mapping sources on the Map Data step of the IIEP, and additional mapping sources are included based on the data, as described in the **ETIM and Media Manual Mapping** section below.

Mapping Process

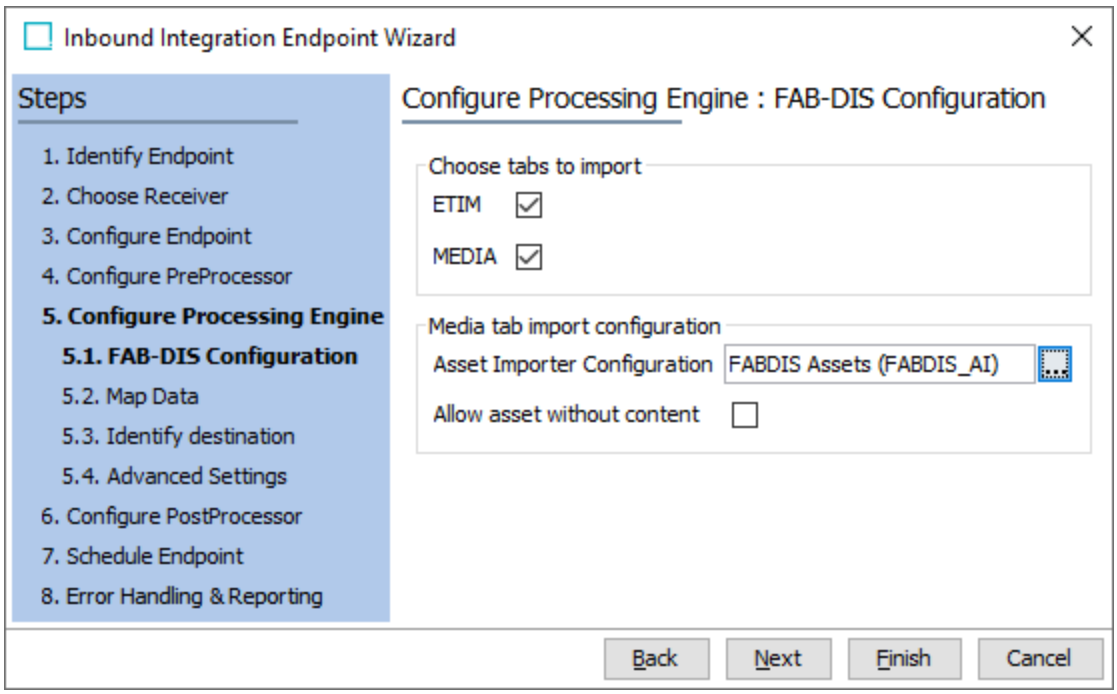
The full mapping process is addressed in the following sections of this topic:

1. IIEP Mapping
2. Automatic Mapping
3. Media Mapping via Asset Importer Configuration

Important: The examples and images below define one way to map FAB-DIS data. The actual mapping steps required for your data are determined by your data model. For assistance, contact your Stibo Systems account representative or technical support.

IIEP Mapping

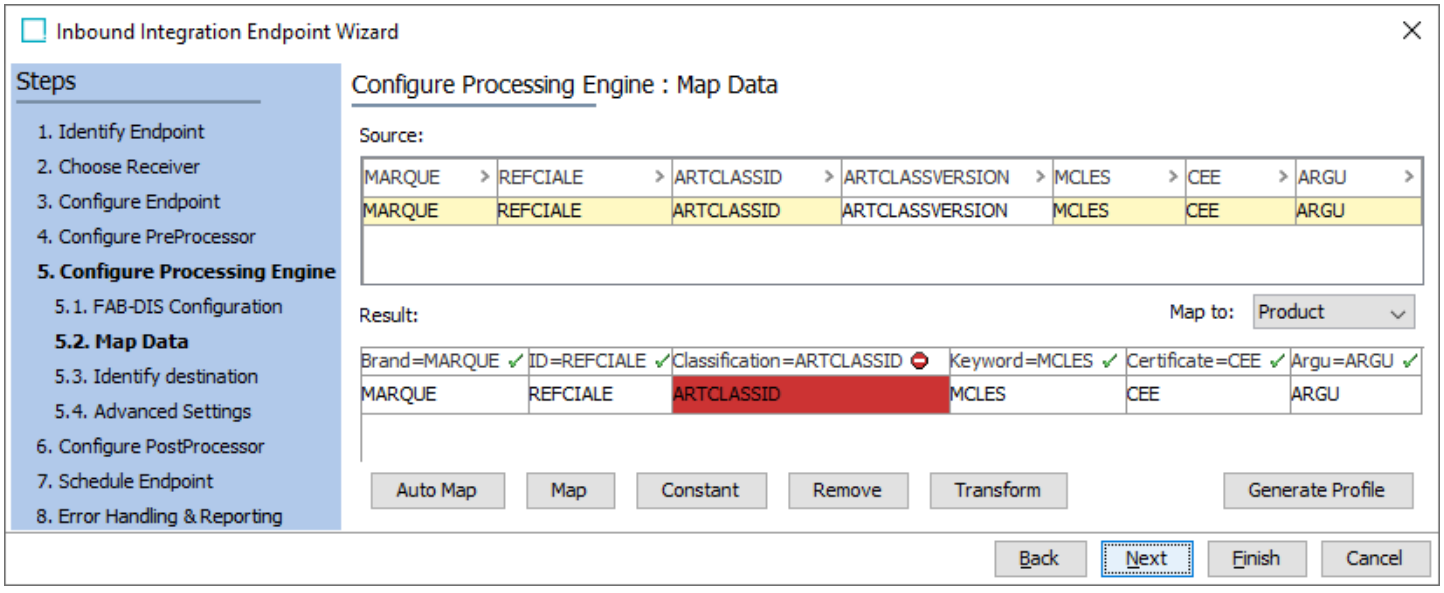
For an IIEP, the FAB-DIS Configuration step of the FAB-DIS Importer processing engine includes checkboxes for selecting the data to be imported. These checkboxes determine the mapping options available in the following Map Data step.



Note: To enable the **Next** or **Finish** buttons, at least one checkbox must be checked on the FAB-DIS Configuration step.

ETIM and Media Manual Mapping

Manual mapping on the Map Data step of the IIEP allows users to assign inbound values to STEP attributes and references for the products and assets included in the import file. For an existing IIEP, Map Data is accessed from the Inbound Integration Endpoint tab > Configuration flipper > via the Edit Configuration link, as defined in the **Inbound Map Data Options** topic of the **Data Mapping** section of this guide.



The Product super type is selected automatically in the 'Map to:' parameter. This setting is also used when only assets are imported because products can be created from the Media tab, and because assets are handled by the asset importer configuration.

Note: To enable the **Next** or **Finish** buttons, at least one Source field must be mapped.

The image above shows an IIEP configured for both ETIM and Media, so all available manual mapping sources are displayed. The following FAB-DIS import data can be manually mapped:

- MARQUE (text), *for both ETIM and Media*, is an attribute on the product that holds the brand name.
- REFCIALE (text), *for both ETIM and Media*, can be the product STEP ID, or an attribute on the product that holds the ID. The REFCIALE is repeated in the input as many times as there are different ETIM features.
- ARTCLASSID (text), *only for ETIM*, is the ETIM classification ID. Using a Product to Classification Link Type allows the products to display as children (sub products) of the classification. For more information, refer to the **Product to Classification Link Types** topic in **System Setup** documentation.

  If the ID in the FAB-DIS file does not match the STEP ID of the desired ETIM classification, during mapping, use the **Insert text before** transformation to include the prefix applied to your STEP ETIM classification IDs. For more information, refer to the **Transformations** topic in the **Resource Materials** online help.

  Note: Sample data is not displayed while mapping a FAB-DIS file. This causes an error to display on the ARTCLASSID column in the Results area of Map Data (shown above as the red cell). This error can be ignored without any negative impact provided the mapping and transformation are created correctly. For details, refer to the **FAB-DIS Import Example** topic.

- ARTCLASSVERSION (text), *only for ETIM*, is the ETIM class version. Although included in the inbound file, mapping this data is not required for import, but is available if necessary for your data model.
- TYPM of MCLES (text) and CEE (text), *only for Media*, are applicable to the product as no NOM, URL, or URLT values will be present with these types of entries and only a CODVAL value is provided for the REFCIALE.

  In all cases of TYPM mapping, the entry in the CODVAL field is being written to the attribute.

  A TYPM value of MCLES is a type of media identified by Keywords in the CODVAL column for an attribute on the product. Multiple entries for CODVAL are handled in a pipe (|) delimited list. Additionally, a NUM column value is required, and determines the order of the delimited data.

  A TYPM value of CEE is the type of media identified by Certificate Codes in the CODVAL column for an attribute on the product. Multiple entries for CODVAL are handled in a pipe (|) delimited list. Additionally, a NUM column value is required, and determines the order of the delimited data.

  A TYPM value of ARGU can be applicable to either a product or an asset, so ARGU can be mapped in both the IIEP and the asset importer, with CODVAL values being applied to the product whenever an asset is not available. Multiple CODVAL entries are not allowed for ARGU on assets. When no asset is provided, the CODVAL is written to the product. Multiple CODVAL entries for the same product are handled in a pipe (|) delimited list. Additionally, a NUM column value is required, and determines the order of the delimited data.

A TYPM value of all other types is a type of media that can be associated with an asset, e.g., a NOM, URL, or URLT value is provided with the entry in the input file. Therefore, within the Metadata Importer configuration of the asset importer, each TYPM can be individually mapped so that, for example, CMC can be entered as the Field and mapped to a Certificate Code attribute on the asset (or reference between the asset and the product).

> **Note:** For additional TYPM value mapping, refer to the **Media Mapping via Asset Importer Configuration** section below.

For details on all mapping options, refer to the **Inbound Map Data - Map** topic in the **Data Mapping** documentation of this guide.

Automatic Mapping

No user action is required to perform automatic mapping.

ETIM Automatic Mapping

When importing ETIM data, the following columns are automatically mapped by the processing engine:

- FEATUREID (text) is the ETIM feature ID and is mapped to the defined ETIM attribute.
- FVALUE (text) is the ETIM feature value and is mapped to the defined ETIM attribute.

> **Important:** The LOV 'Value ID' (not the LOV 'Values') is required for the FVALUE on an LOV attribute. For more information, refer to the **List of Values (LOVs)** topic in **System Setup** documentation.

Media Automatic Mapping

When importing Media data, the following columns are used to identify the asset, and are automatically mapped by the processing engine using the following priority:

- URLT (text) is the URL to download the document or photo, and when included in the FAB-DIS file, it is used for identifying the asset. If the file is not available from this URL, an error is reported in the execution log of the background process. If the asset cannot be found and the 'Allow asset without content' box is checked, an *asset placeholder* is created.
- URL (text) is the URL to display the document or photo on a merchant site, and is used to identify the asset when no URLT is available. If the file is not available from this URL, an error is reported in the execution log of the background process. If the asset cannot be found and the 'Allow asset without content' box is checked, an *asset placeholder* is created.
- NOM (text) is the name of the asset included in the .ZIP file, and is used to identify the asset only when no URLT or URL is available. If the .ZIP file includes subfolders, the path to the asset must be in the Excel file.

If a path is provided in the NOM column, use the Asset Importer Configuration > Asset Matcher parameters to generate a name for the asset as defined in the **Asset Matcher** topic of the **Digital Assets** documentation.

The URLT, URL, and NOM columns can also be mapped to attributes using the Metadata Importer section of the Asset Importer Configuration. For more information, refer to the **Metadata Importer** topic of the **Digital Assets** documentation.

When STEP cannot access the asset (i.e., due to an incorrect file name or blocked access to the URL), an asset placeholder can be created as defined in the **IIEP Configuration** section of the **FAB-DIS Import Workbench Configuration** topic.

Media Mapping via Asset Importer Configuration

When importing Media data, the following columns can be mapped within the asset importer configuration.

- TYPM (text) is the media type of the asset as defined in an attribute, as an object type, or the reference type.

  The TYPM field is mapped in the asset importer configuration and can be mapped as the reference type of the assets or as metadata on the asset or reference, depending on the configuration and data model. This field also plays a special role in CODVAL mapping, described in the **ETIM and Media Manual Mapping** section above.

  In all cases of TYPM mapping, the entry in the CODVAL field is being written to the attribute.

- NUM (numeric, required) is the order number of a feature that is used for sorting.

  The NUM field is mapped in the asset importer configuration and can be mapped as metadata on the asset or reference, and is also automatically used (without any mapping) to sequence multiple CODVAL entries for the same asset or product.

- CODVAL (text) is the certificate code, keyword value, or sales pitch.

  The CODVAL field can be mapped directly in the asset importer as metadata on the asset or reference. However, since the meaning of CODVAL entries varies based on the TYPM for the row, special handling is applied to the CODVAL field to enable more complex mapping. Specifically, both the IIEP and asset importer allow for users to map TYPM entries to individual attributes.

  A TYPM value of ARGU can be applicable to either a product or an asset, so ARGU can be mapped in both the IIEP and the asset importer, with CODVAL values being applied to the product whenever an asset is not available. Multiple CODVAL entries are not allowed for ARGU on assets. When no asset is provided, the CODVAL is written to the product. Multiple CODVAL entries for the same product are handled in a pipe (|) delimited list. Additionally, a NUM column value is required, and determines the order of the delimited data.

For details on the asset importer, refer to the **Asset Importer Configuration** topic in the **Digital Assets** documentation.

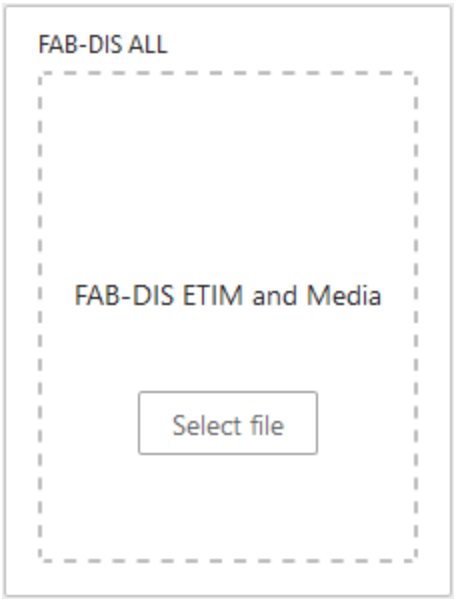
FAB-DIS Import Web UI Configuration

Importing FAB-DIS data from the ETIM and Media tabs of an Excel file is available in an IIEP via the FAB-DIS Importer processing engine. For more information and prerequisites, refer to the **FAB-DIS Import Format** topic.

Important: Before an import of FAB-DIS data can be successful from the workbench or Web UI, the workbench configuration must be completed as defined in the **FAB-DIS Import Workbench Configuration** topic.

After completing the workbench configuration, FAB-DIS files can be processed using the Web UI File Loading Widget. This allows users to import FAB-DIS data without using workbench.

1. In workbench, note the FAB-DIS IIEP created in the previous configuration section. This IIEP will be selected and used by the File Loading widget.
2. In the Web UI, add a File Loading Widget to your homepage, as defined in the **File Loading Widget** topic of **Web User Interfaces**.



3. Choose a method to invoke the IIEP:
   - Drag-and-drop one or more FAB-DIS files onto the widget.
   - Click the 'Select File' button and choose the file(s).

   The file(s) are immediately processed by the configured IIEP.
4. Monitor the background processes generated by the import using a Background Processes List Screen. For more information, refer to the **Background Process List Screen** topic of **Web User Interfaces**.

FAB-DIS Import Background Processes

Importing FAB-DIS data from the ETIM and Media tabs of an Excel file is available in an IIEP via the FAB-DIS Importer processing engine. When a FAB-DIS Importer IIEP is invoked, the following multiple background processes (BGPs) can run.

1. The initial BGP displays the following information:
   - The user, date and time that the process was invoked.
   - A link to view the secondary BGP.
   - When necessary, the problem that caused the secondary BGP to fail.
2. The second BGP takes the following actions:
   - For all imports, determines if the data meets the prerequisites defined in the **Prerequisites** section of the **FAB-DIS Import Format** topic.
   - If the Excel file has an ETIM tab, data is processed and products are created and/or updated. All ETIM information is reported in this BGP execution report. Unlike typical multi-BGP scenarios, when this BGP completes with errors, if Media data is being processed, the next BGP is started.
   - If the Excel file has a Media tab, products are created or updated with only the data mapped to the MARQUE and REFCIALE columns. These products are referenced to the imported assets.
   - If the Excel file has a Media tab, a subsequent BGP is started to import assets. A link is displayed to view the asset importer BGP.
3. The last BGP runs the asset importer when required.

Troubleshooting

Most of the errors reported by the FAB-DIS Importer processing engine give details and direction for resolving the problem. For details on import requirements, refer to the **Prerequisites** section of the **FAB-DIS Import Format** topic.

The following reasons for failure involve multiple elements, and therefore the resolution is not presented clearly in the execution report.

Error	Reason	Resolution
Classification 'ETIM7_' not found is reported on the initial execution report	For the ETIM import, a transformation is defined in the Map Data step of the IIEP for the ARTCLASSID column. The transformation inserts 'ETIM7_' before the classification ID. An row in the Media tab contains a REFCIALE that is not found on the ETIM tab, which indicates a new product should be created. The new product is created but cannot be added to a	Manually select a classification for the new product, or reimport with the new product's REFCIALE on the ETIM tab.

Error	Reason	Resolution
	classification since the classification is not available in the FAB-DIS file. When this error is reported, the XML also includes a malformed tag, displayed as: <ClassificationReference ClassificationID="ETIM7_"/>	
Asset file '<file name>' not found. asset with no content will be imported. is reported on the asset importer execution report	The asset file cannot be found in the .ZIP file or via external download. This can happen when the URL or URLT cannot be accessed by the importer, the value in the NOM field does not match an asset in the .ZIP file, a .ZIP file was not supplied, or no URL or URLT value is present.	Make the asset freely accessible.

FAB-DIS Import Example

Before configuring an IIEP to import FAB-DIS data from the ETIM or Media tab of an FAB-DIS Excel file, review the information and prerequisites on the **FAB-DIS Import Format** topic.

The following sections show ETIM and Media data processed by a fully-configured IIEP running the FAB-DIS Importer processing engine.

ETIM Tab

This section demonstrates data from the FAB-DIS ETIM tab that is mapped in the IIEP, and the results in Tree once the IIEP is invoked.

ETIM Sample Data

Notice that data from rows with the same 'REFCIALE' value are written to the same product.

	A	B	C	D	E	F
1	MARQUE	REFCIALE	ARTCLASSID	FEATUREID	FVALUE	ARTCLASSVERSION
2	ARNOULD	00001	EC011609	EF005474	EV009848	ETIM-6.0
3	ARNOULD	00001	EC011609	EFI08530	79	ETIM-6.0
4	ARNOULD	00001	EC011609	EFI08530	92	ETIM-6.0
5	ARNOULD	00001	EC011609	EF012591	TRUE	ETIM-6.0
6	ARNOULD	00002	EC011609	EF001438	2550	ETIM-6.0
7	ARNOULD	00002	EC011609	EFI08530	45	ETIM-6.0
8	ARNOULD	00002	EC011609	EFI08530	75	ETIM-6.0
9	ARNOULD	00002	EC011609	EF012591	FALSE	ETIM-6.0

ETIM Mapping

The sample data above is mapped in the IIEP, as shown in the image below.

- The MARQUE source is mapped to the 'Brand Name' attribute on the product.
- The REFCIALE source is mapped to the STEP ID attribute on the product.
- The ARTCLASSID source is mapped to the Product Classification Link using the ID aspect, and the **Insert Text Before** transformation is used to add the 'ETIM7_' prefix to the ID.
- The ARTCLASSVERSION source is not mapped since the ETIM version is not needed for the data model.
- The 'FEATUREID' and 'FVALUE' sources are automatically mapped by the processing engine. The following FEATUREID data is included in this import file:

  'EF005474' is the 'Degree of protection (IP)' attribute, and uses the LOV ValueID identified in the FValue column

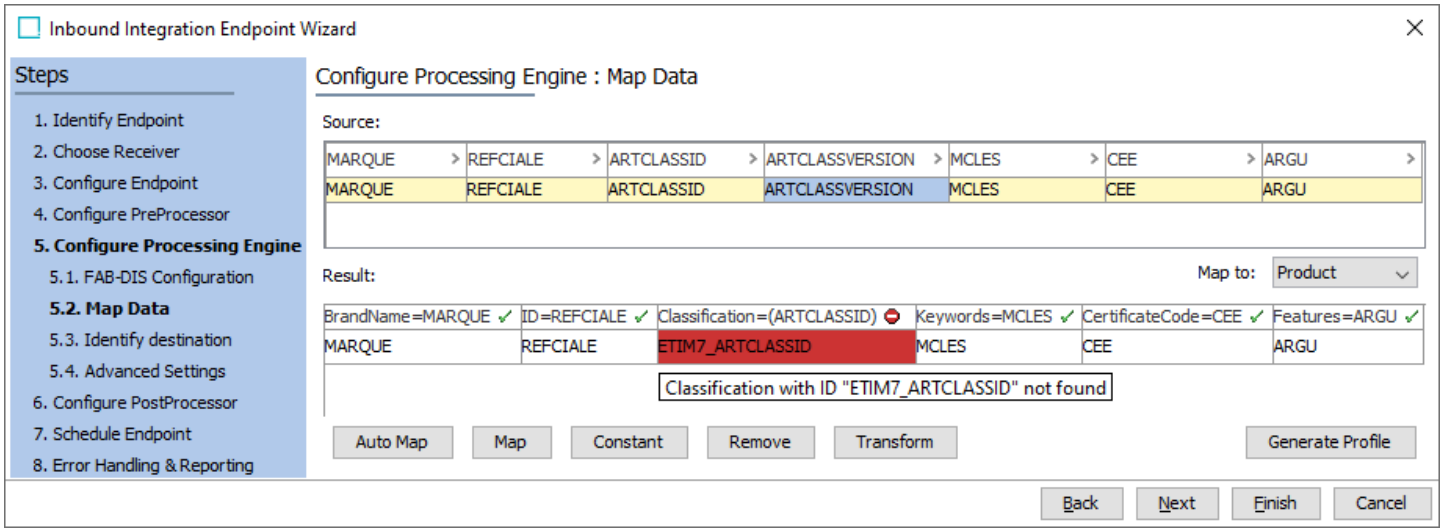
  'EFI08530' is the 'Hoisting load' attribute, and is of validation base type 'Number Range,' so two rows are used to supply the range values

'EF012591' is the 'Arched' attribute, and uses the 'Logical' LOV with only TRUE or FALSE values

'EF001438' is the 'Length' attribute and is of validation base type 'Number'

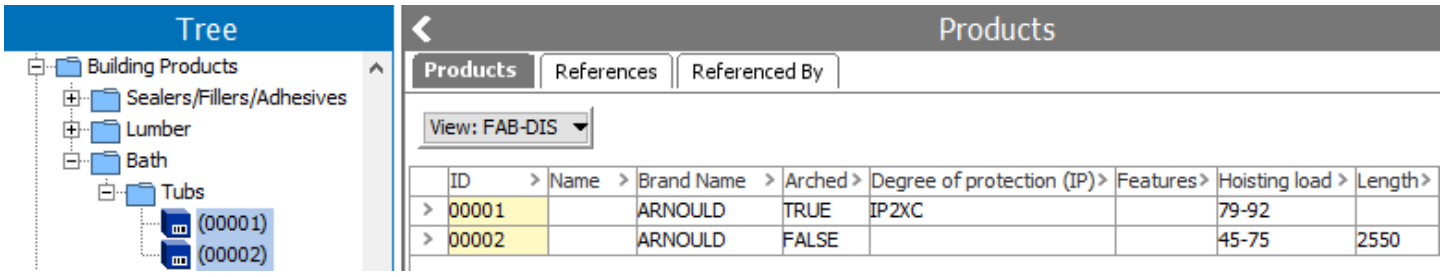
The TYPM values of MCLES, CEE, and ARGU indicate product data sources on the Media tab. For details, refer to the **Media Mapping** section below.

Note: The Source and Result areas of the Map Data step do not show sample data since it is not available to the IIEP during configuration. As a result, the ARTCLASSID mapped column displays an error. Hovering over the red error field displays the reason (as shown below) and indicates that there is no product classification link in the system with the ID of 'ETIM7_ARTCLASSID.' This error can be ignored and will not be reported during import when actual classification IDs are processed.

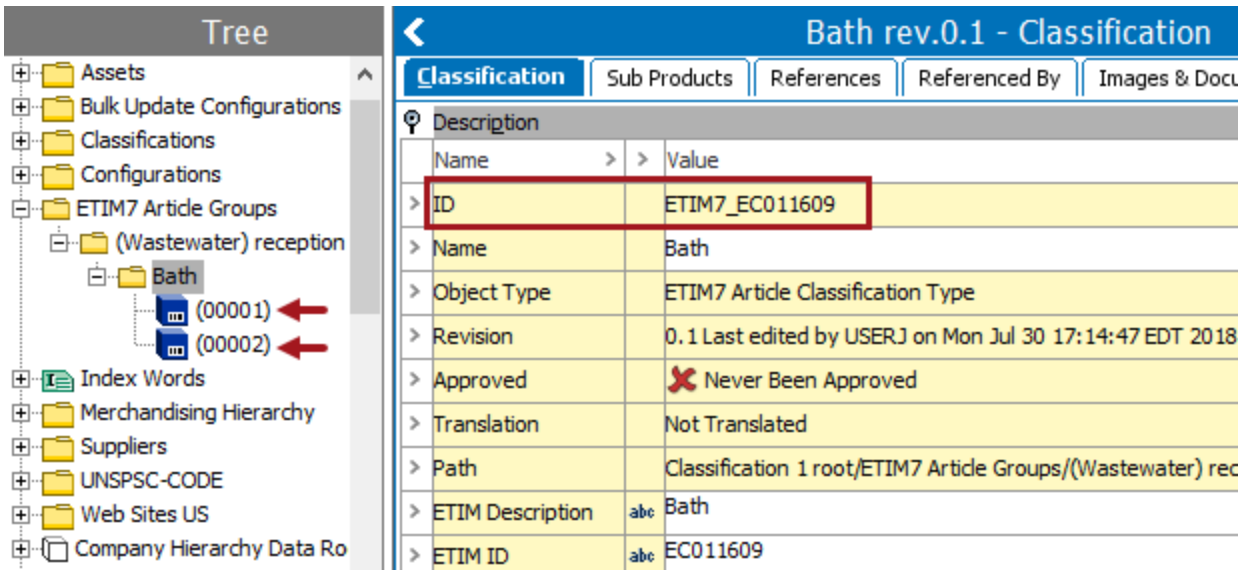


ETIM Result

When the IIEP is invoked, the following products are created in STEP from the ETIM tab of the input file.



The new products are also displayed as sub products to the specified classification (ETIM7_EC011609) via the Product to Classification Link Type ID.



MediaTab

This section demonstrates data from the FAB-DIS Media tab that is mapped in the IIEP and in the Asset Importer configuration, and the results in Tree once the IIEP is invoked.

Media Sample Data

Notice that on the Media tab, data from rows with the same 'REFCIALE' value are written to the same product.

	A	B	C	D	E	F	G	H
1	MARQUE	REFCIALE	TYPM	NUM	CODVAL	NOM	URL	URLT
2	ARNOULD	00001	ARGU	1	Freestanding	49028.tif	http://images.com/mws/media/49028.jpg	http://images.com/mws/media/49028.tif
3	ARNOULD	00005	MCLES	1	KW1 | KW2			
4	ARNOULD	00006	CEE	1	Cert7846			
5	ARNOULD	00007	FICHE	1		BlackTub.tif	http://images.com/mws/media/BlackTub.jpg	http://images.com/mws/media/BlackTub.tif
6	ARNOULD	00007	PHOTOBD	2		COPPER.tif	http://images.com/mws/media/COPPER.jpg	http://images.com/mws/media/COPPER.tif
7	ARNOULD	00007	CMC	3	Cert9221	whitetub.tif	http://images.com/mws/media/whitetub.jpg	http://images.com/mws/media/whitetub.tif
8	ARNOULD	00008	ARGU	1	Jet			

Media Mapping

Since the Media tab can contain data for both products and assets, the sample data is mapped in both the IIEP and Asset Importer configuration, as shown in the images below.

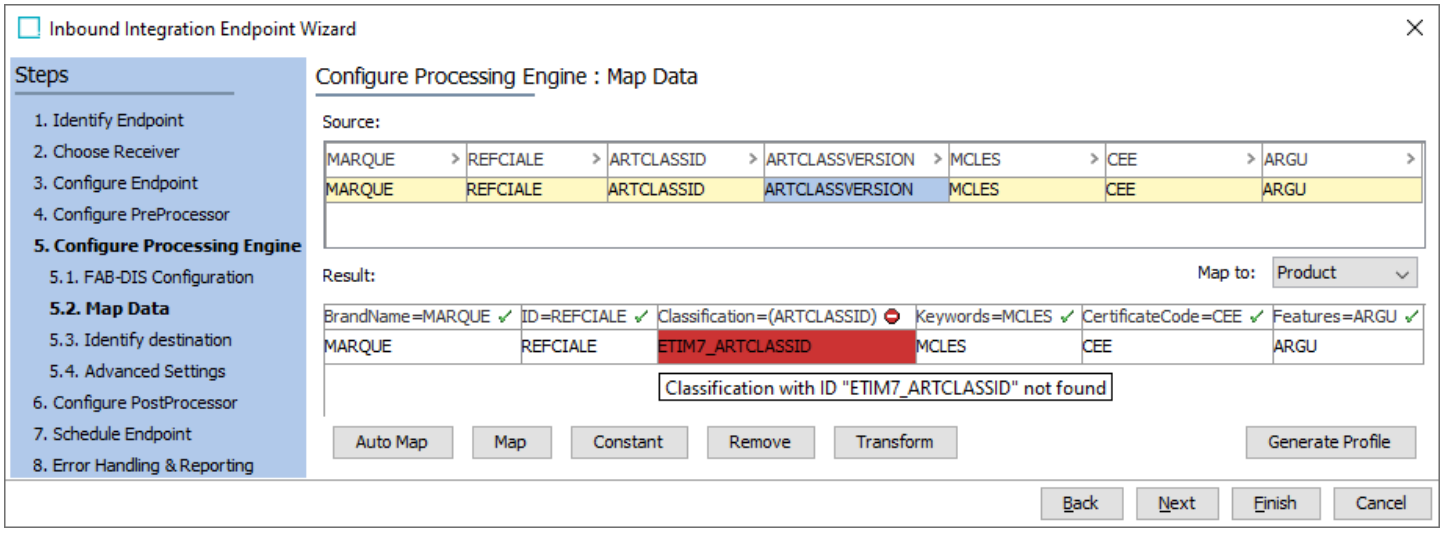
Important: The values in the TYPM column represent a variety of data that belongs to the product or the asset, such as ARGU or CMC. The CODVAL source is not mapped, but data from this column is used when the TYPM column values are MCLES, CEE, or ARGU.

The **IIEP Map Data** step image shows the following mappings for data on the Media tab:

- The MARQUE source is mapped to an attribute that holds the brand name, which is on the product.
- The REFCIALE source is mapped to the STEP ID attribute, which is on the product.
- The TYPM value MCLES source is mapped to the 'Keywords' attribute, which is on the product. This indicates that the CODVAL value on the MCLES row will be written to the mapped attribute.
- The TYPM value CEE source is mapped to 'CertificateCode' attribute, which is on the product. This indicates that the CODVAL value on the CEE row will be written to the mapped attribute.
- The TYPM value ARGU source is mapped to 'Features' attribute, which is on the product. This indicates that the CODVAL value on the ARGU row will be written to the mapped attribute *but only when an asset does not exist for the row*.

  The TYPM value ARGU source can also belong to an asset. To write the CODVAL value on the ARGU row to an asset, ARGU must be mapped in the Asset Import Configuration.

The URLT, URL, and NOM sources are not manually mapped in the IIEP, but are used by the importer to find the asset being imported. They can also be mapped as metadata in the asset importer configuration, as shown in the next section.



The **Asset Importer Configuration** image shows the following mappings for data on the Media tab:

- The TYPM source is mapped two times. In the Product Linker flipper it holds the reference type used between the asset and the product. A less common use would be as metadata on the asset via the Metadata Importer flipper where it is mapped to an attribute. Both are demonstrated below.
- The TYPM value ARGU is mapped to the 'Features' attribute on the asset. The CODVAL value on the ARGU row will be written as metadata.
- The TYPM value CMC is mapped to the 'CertificateCode' attribute on the asset. The CODVAL value on the CMC row will be written as metadata.
- The NUM source is mapped to the 'ImagePriority' attribute as metadata on the asset reference.
- The URLT, URL, and NOM values are mapped to metadata attributes on the asset.

FABDIS Assets rev.0.4 - Asset Importer Configuration Type

Asset Importer Configuration Type | Log | Status

Metadata Importer

Metadata Importer: Asset and Reference Metadata Importer

Metadata Source	Field	STEP Attribute
Metadata file	TYPM	AssetType
Metadata file	ARGU	Features
Metadata file	CMC	CertificateCode
Metadata file	NUM	ImagePriority
Metadata file	NOM	FileName
Metadata file	URL	URL
Metadata file	URLT	URLT

Add Field Mapping

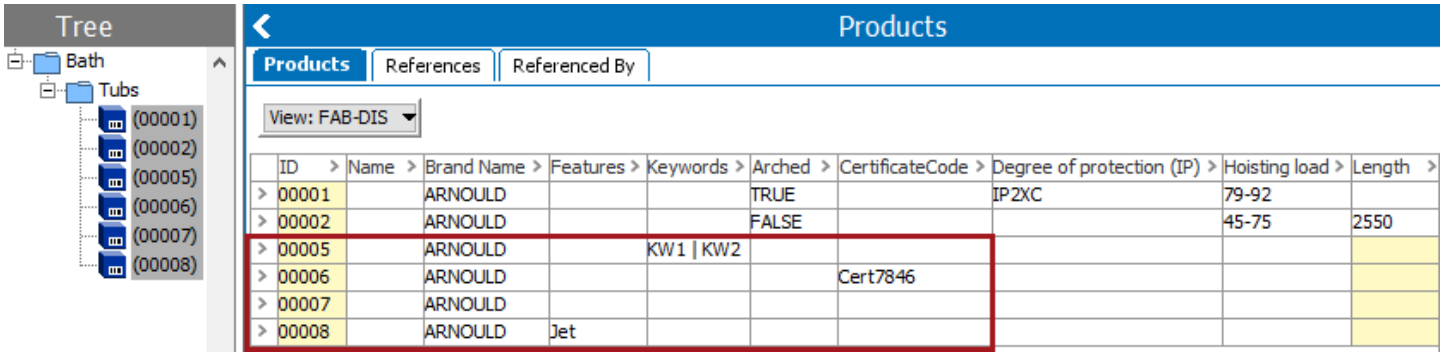
Product Linker

Product Linker: Metadata Product Linker

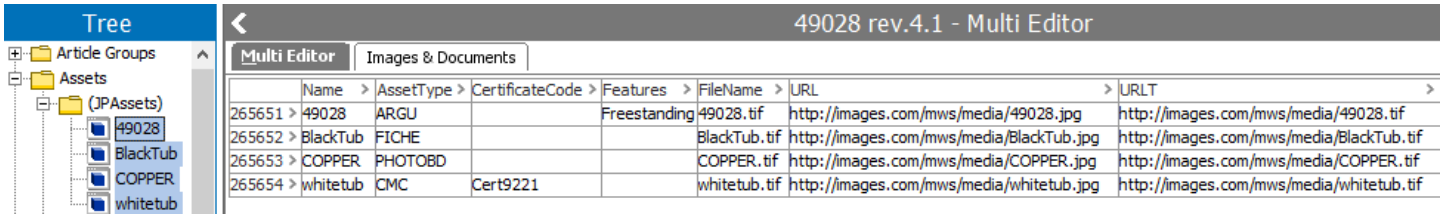
Match Metadata Field	ID
Match On	ID
Key	
Match Below Product Root	Tubs (260025)
Reference Type Field	TYPM
Default Reference Type	HD Photo
Allow Multiple Products	☑

Media Result

When the IIEP is invoked, the products and assets from the Media tab can be created or updated in STEP. Product data is imported for the columns MARQUE (to BrandName), REFCIALE (to ID), the CODVAL value for a TYPM row of MCLES (to Keywords), the CODVAL value for a TYPM row of CEE (to CertificateCode), and the CODVAL value for a TYPM row of ARGU (to Features) when no asset is provided (as is the case for ID 00008).

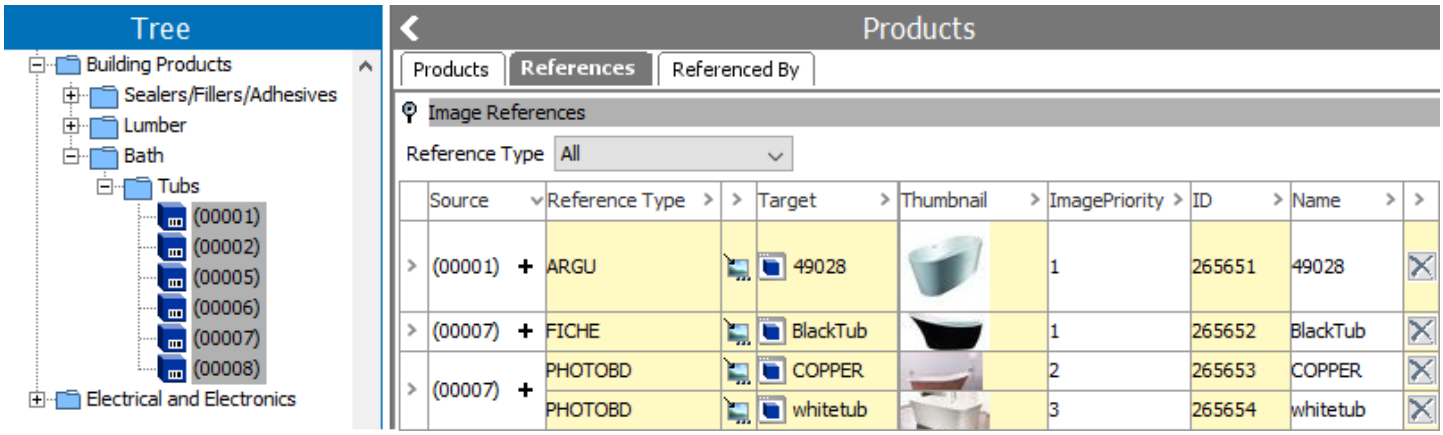


New assets are created or existing assets are updated based on the Asset Importer Configuration. Based on the settings in the Metadata Importer flipper, asset data is imported for the columns TYPM (to AssetType), the CODVAL value for a TYPM row of CMC (to CertificateCode), the CODVAL value for a TYPM row of ARGU (to Features), NOM (to FileName), URL (to URL), and URLT (to URLT).



	Name	AssetType	CertificateCode	Features	FileName	URL	URLT
265651	49028	ARGU		Freestanding	49028.tif	http://images.com/mws/media/49028.jpg	http://images.com/mws/media/49028.tif
265652	BlackTub	FICHE			BlackTub.tif	http://images.com/mws/media/BlackTub.jpg	http://images.com/mws/media/BlackTub.tif
265653	COPPER	PHOTOBD			COPPER.tif	http://images.com/mws/media/COPPER.jpg	http://images.com/mws/media/COPPER.tif
265654	whitetub	CMC	Cert9221		whitetub.tif	http://images.com/mws/media/whitetub.jpg	http://images.com/mws/media/whitetub.tif

The products have references to the new assets via the Product to Classification Link Type ID, where the link type is provided by the TYPM value based on the Product Linker 'Reference Type Field' setting. The 'ImagePriority' data is provided by the NUM value based on the Metadata Importer mapping.



	Source	Reference Type		Target	Thumbnail	ImagePriority	ID	Name
	(00001)	ARGU		49028		1	265651	49028
	(00007)	FICHE		BlackTub		1	265652	BlackTub
	(00007)	PHOTOBD		COPPER		2	265653	COPPER
		PHOTOBD		whitetub		3	265654	whitetub

FixedWidth Format

A FixedWidth format is a text file with rows and columns, and each column has a fixed width.

Format Availability

FixedWidth is available for selection in:

- IIEP - refer to **Creating an Inbound Integration Endpoint**
- Import Manager - refer to **Creating a Data Import**

Mapping

This format requires creating a data map between STEP and the data being processed, and may also include data transformations. For details, refer to **Data Mapping**.

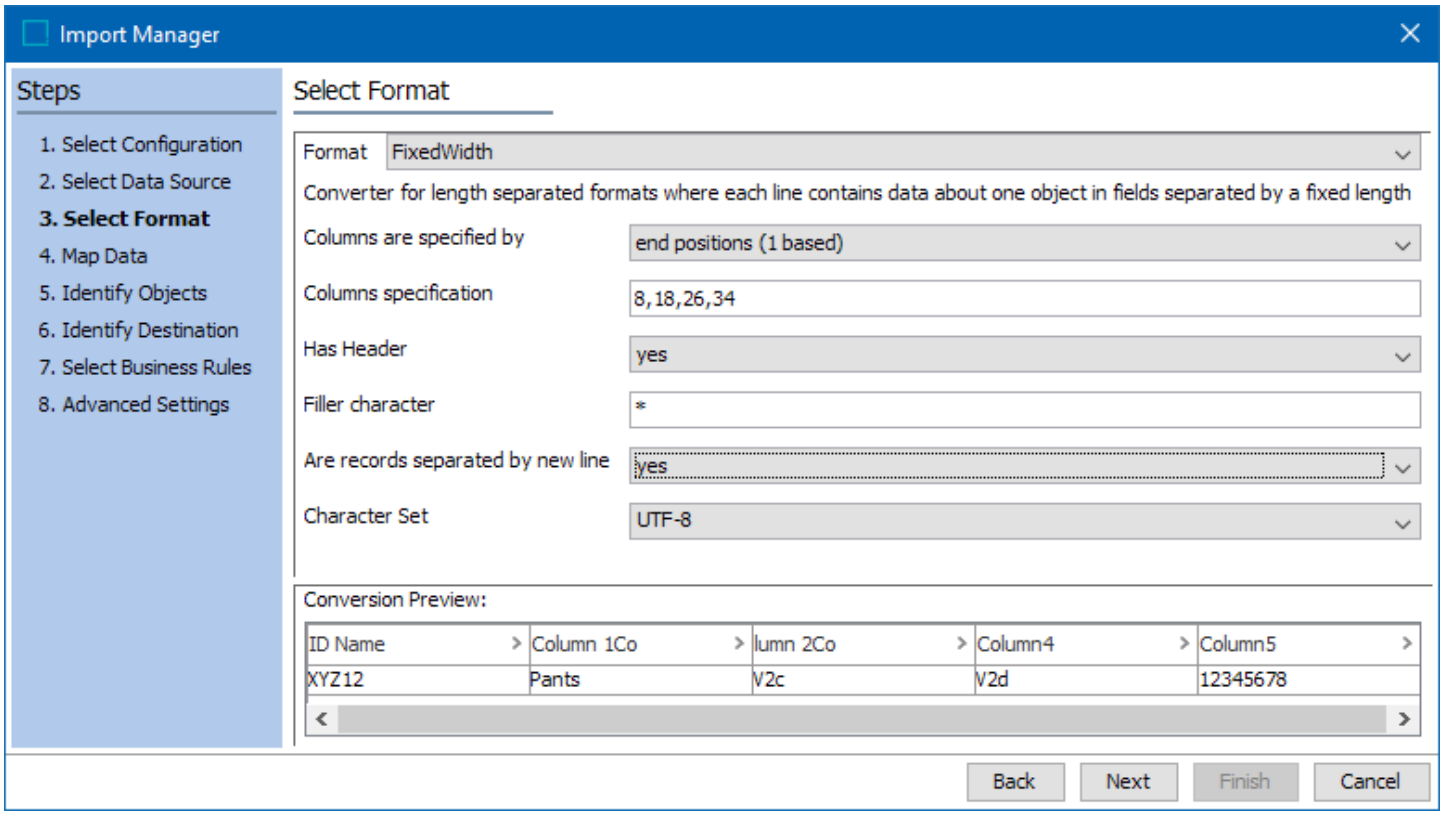
Inbound Data

For examples of ways to modify fixed width data for import and further explanations on the inbound parameters, refer to **FixedWidth Format Configuration Examples**.

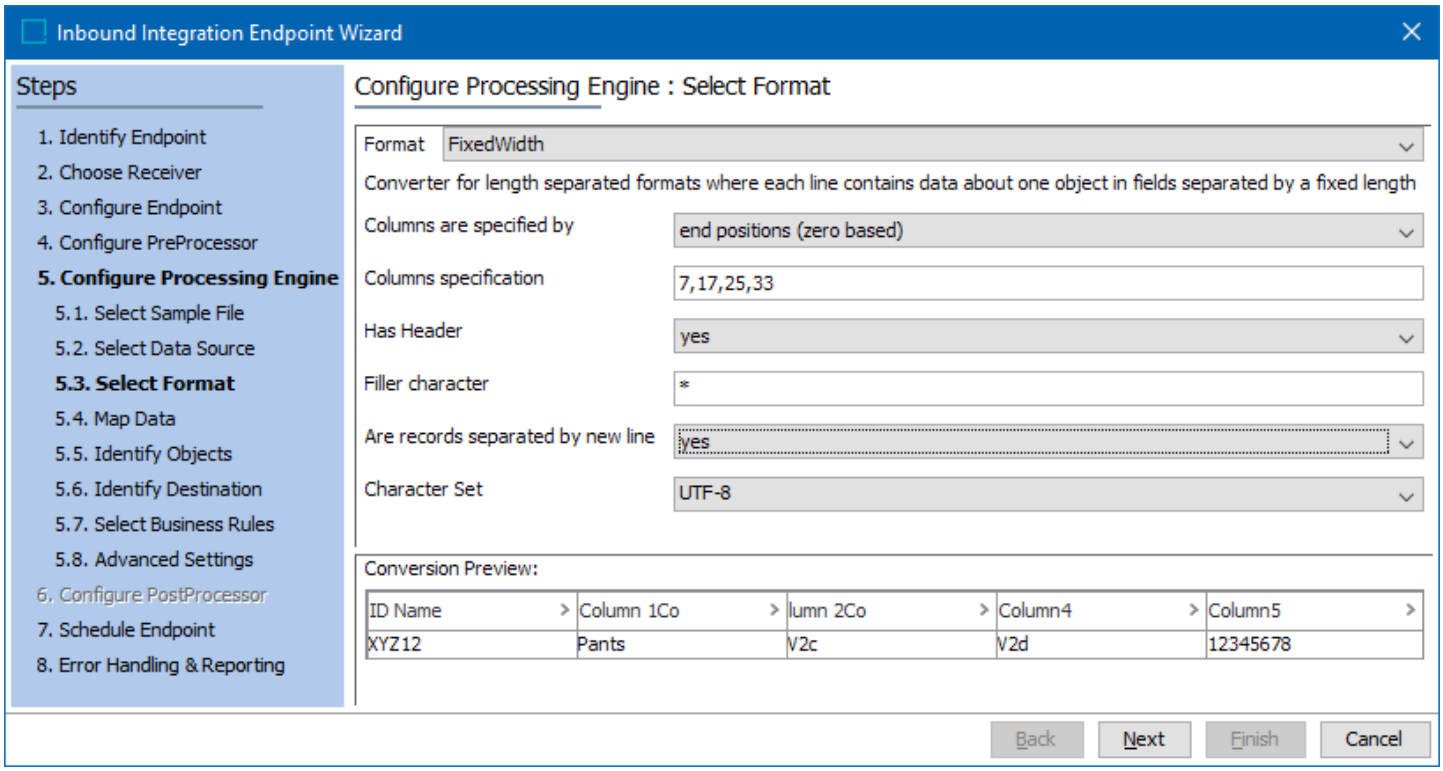
Inbound Parameters

- **Columns are specified by** allows you to determine how the initial element is identified via the following options:
  - **end positions (zero-based)** - the last character counted is counted with a 'zero' value.
  - **end positions (1 based)** - every character is counted and indicates the last character in a value set.
  - **width**- the numeric values of how long each section should be from the last character grouping is counted, and each new batch starts with 'one' as the value of the character.
- **Columns specification** allows you to further specify columns via the following options:
  - For **end positions (zero-based)**, specify the number (position) of the before last (n-1) character of each column.
  - For **end positions (1 based)**, type the number (position) of the last character of each column.
  - For **width**, specify the width / length in characters of each column.
- **Has Header** allows you to indicate if the file has a header line. If the first line (row) of the FixedWidth file has header information such as attribute names, you can use the Auto Map feature mapping the columns of data to STEP objects.
- **Filler Character**, if necessary, specify the filler character that you want to remove from the fields.
- **Are records separated by new line**, when set to 'yes', records in the inbound file are separated by new lines.
- **Character Set** select the desired character set to determine the characters that can be successfully imported. Options include Windows-1252, ISO-8859-1 (also known as the Latin-1 character set), UTF-8, or UTF-16. The common setting is **UTF-8**, unless you have a reason to do otherwise.

Import Manager



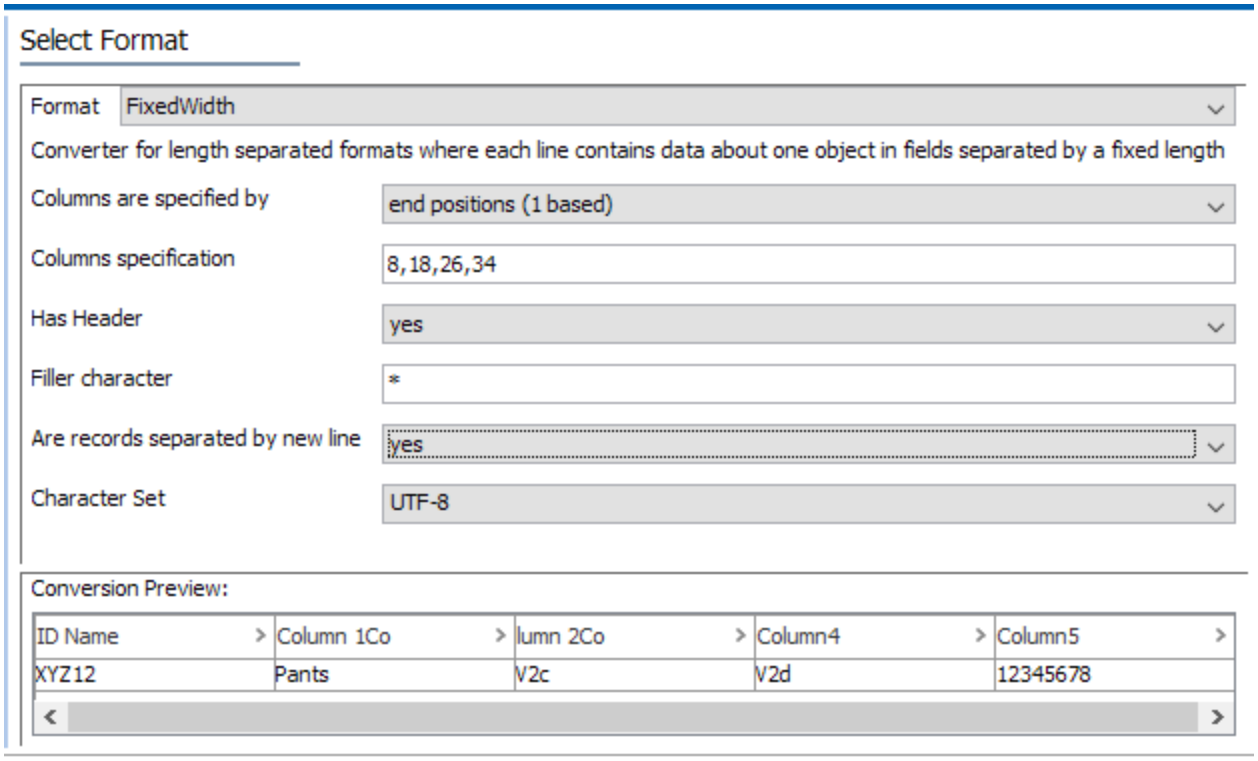
IIEP


Inbound Integration Endpoint Wizard
Steps
1. Identify Endpoint
2. Choose Receiver
3. Configure Endpoint
4. Configure PreProcessor
5. Configure Processing Engine
5.1. Select Sample File
5.2. Select Data Source
5.3. Select Format
5.4. Map Data
5.5. Identify Objects
5.6. Identify Destination
5.7. Select Business Rules
5.8. Advanced Settings
6. Configure PostProcessor
7. Schedule Endpoint
8. Error Handling & Reporting
Configure Processing Engine : Select Format
Format
FixedWidth
Converter for length separated formats where each line contains data about one object in fields separated by a fixed length
Columns are specified by
end positions (zero based)
Columns specification
7,17,25,33
Has Header
yes
Filler character
*
Are records separated by new line
yes
Character Set
UTF-8
Conversion Preview:
ID Name
Column 1Co
lumn 2Co
Column4
Column5
XYZ12
Pants
V2c
V2d
12345678
Back
Next
Finish
Cancel

FixedWidth Format Configuration Examples

The 'Columns are specified by' parameter determines how the initial element in the import file is identified and therefore affects how the full data file is imported. Each option enables you to modify the separation of the data fields differently, as demonstrated in the following examples. Choose the option that causes the Conversion Preview panel to display the data attribute values as they should exist in STEP after the import.

For more details, refer to the **FixedWidth Format** documentation.



Example Data

Each of the examples below use the following input file.

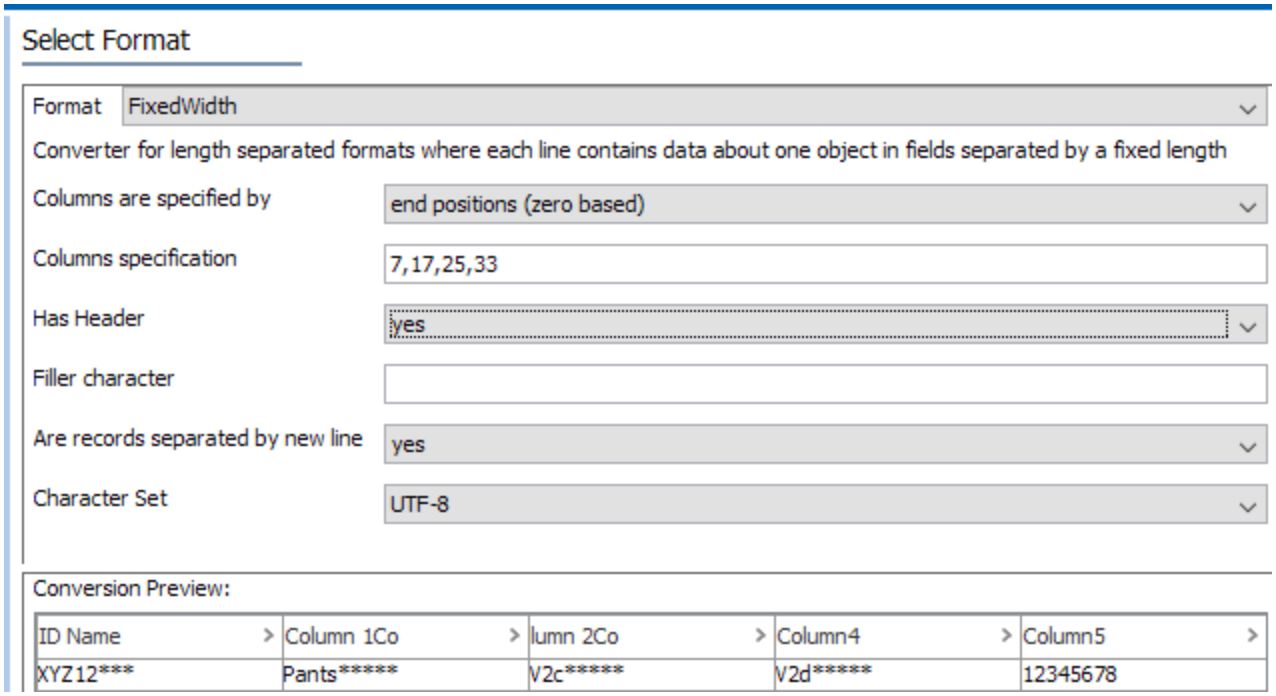
```
ID      Name    Column  1Column 2Column 3
XYZ12***Pants*****V2c*****V2d*****12345678
```

End Positions (Zero Based) Example

The 'end positions (zero-based)' option means the last character counted is counted with a 'zero' value. In other words if the input values was 'trv2**tgh4****j5r' then the numeric value to put 'trv2**' in its own column and 'tgh4***' in its own column would be 5,13. This is because in the first batch of characters, 'trv2**', the last asterisk (*) counts as 'zero.' The last asterisk (*) in the 'tgh4****' is technically number 14, but because it counts as 'zero', you put the number 13 next to signal that the character after it is considered the end of the batch.

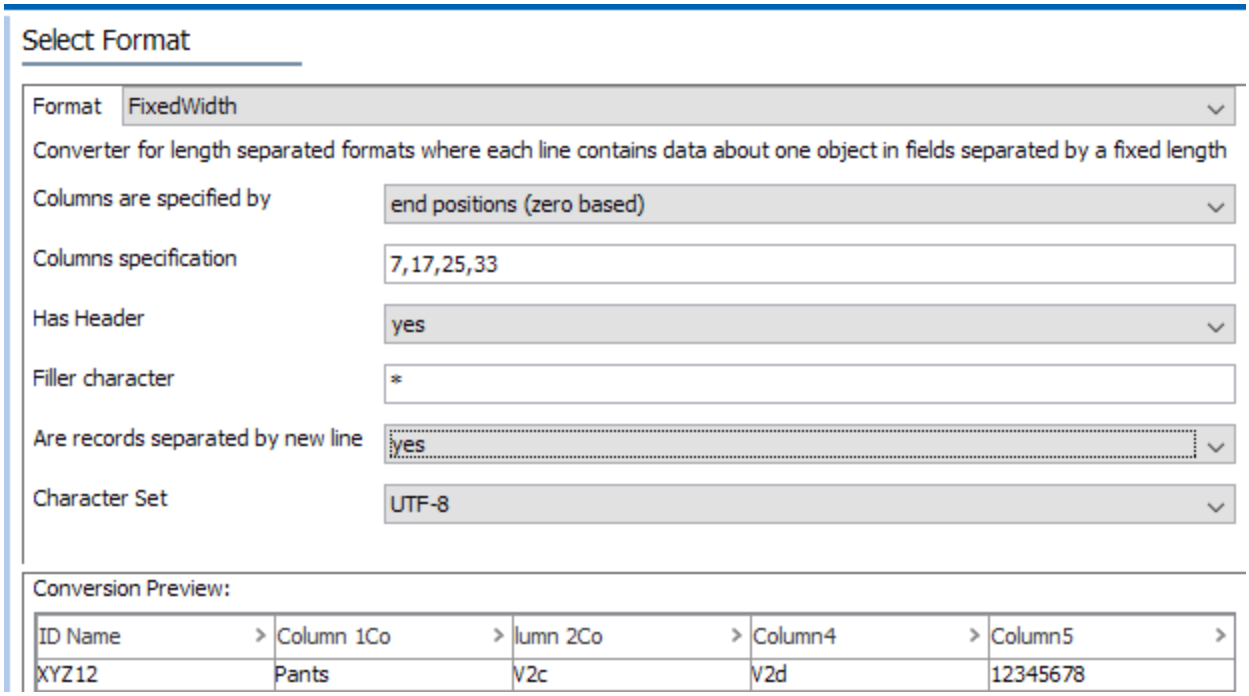
1. **Columns are Specified by** is set to **end positions (zero based)**.
2. **Columns Specification** is set to **7,17,25,33**.
3. **Has Header** is set to **Yes**.
4. **Are records separated by new line** is set to **Yes**.
5. **Character set** is set to **UTF-8**.

   The Conversion Preview displays the filler characters.



6. **Filler character** is set to * (which is being used to take up unneeded space between values)

   The Conversion Preview no longer displays the filler characters.

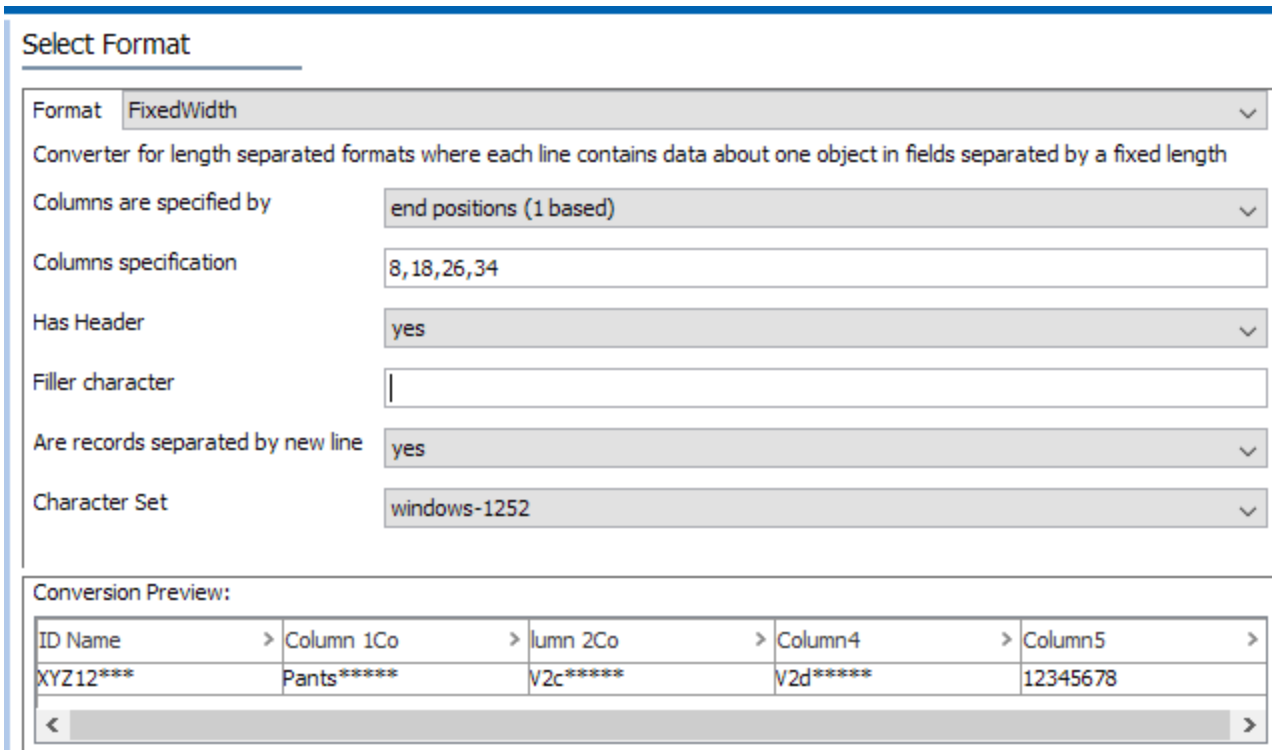


End Positions (One Based) Example

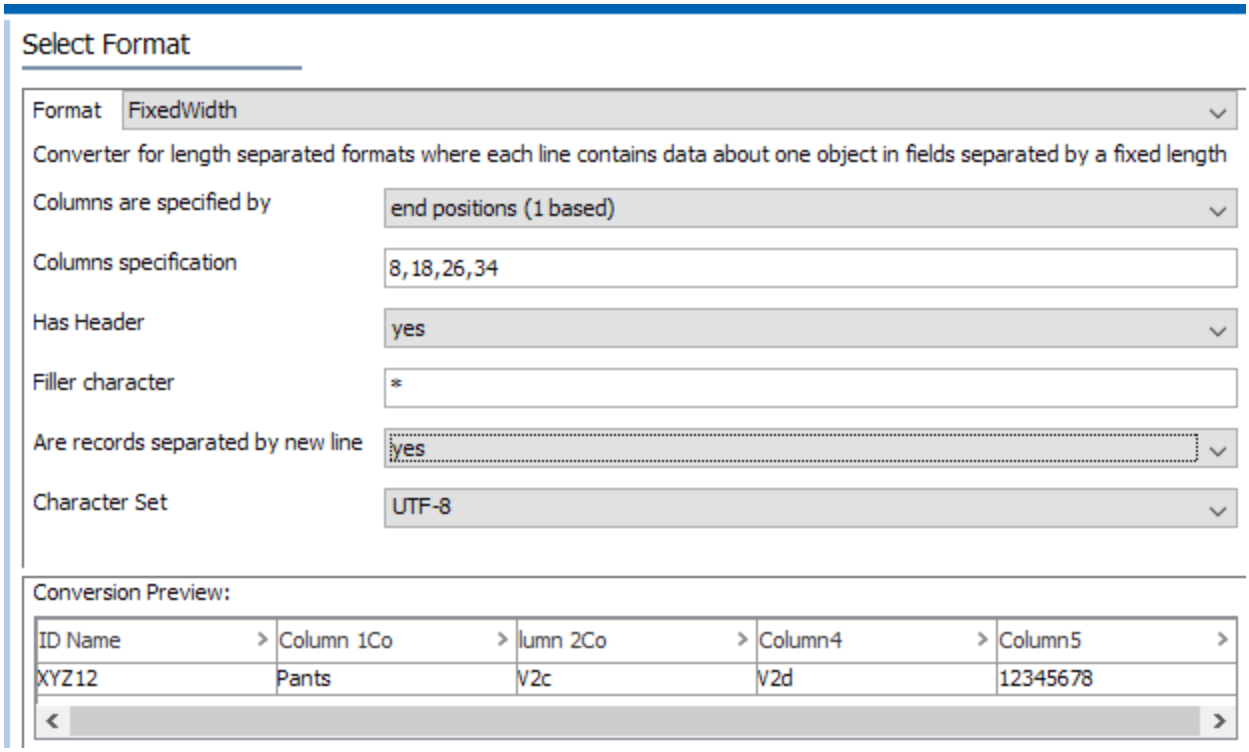
The 'end positions (1 based)' option indicates then the user needs to put in the numeric values of the last digit that should be considered part of each 'value.' This includes any symbols that are used as spaces. For example, if the value was 'trv2**tgh4****j5r' then the numeric value to put 'trv2**' in its own column would be 6 because each character counts individually.

1. **Columns are Specified by** is set to **end positions (1 based)**.
2. **Column specification** is set to **8,18,26,34**.
3. **Has Header** is set to **Yes**.
4. **Are records separated by new lines** is set to **Yes**.
5. **Character set** is set to **UTF-8**.

   The Conversion Preview displays the filler characters.



6. **Filler character** is set to * (which is being used to take up unneeded space between values)
   The Conversion Preview no longer displays the filler characters.



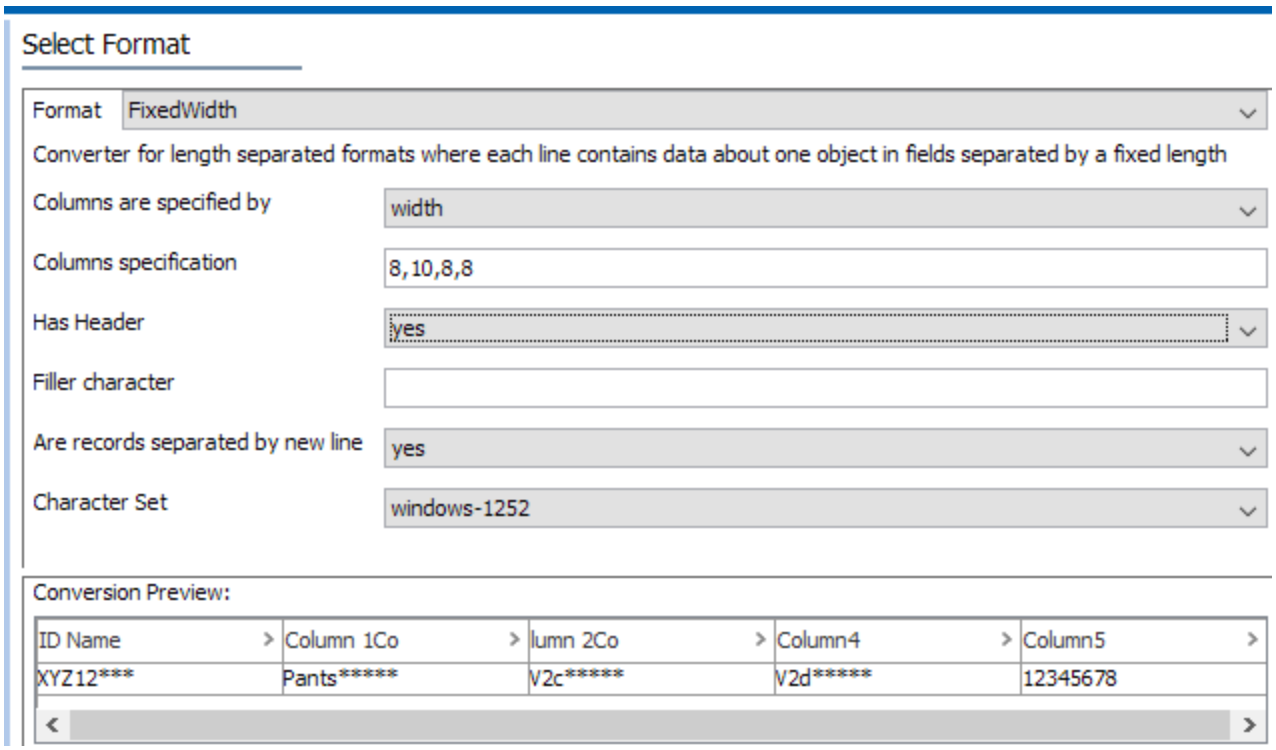
Width Example

The 'width' option indicates the number of characters in each section, excluding the last section. Each new section starts with 'one' as the value of the character.

For example, for 'trv2**tgh4****j5r' because there are 6 characters in the first section, '6' is put first. There are 8 in the next section, and the remaining numbers are in the third section, which is not counted. Thus the numbers entered would be 6,8.

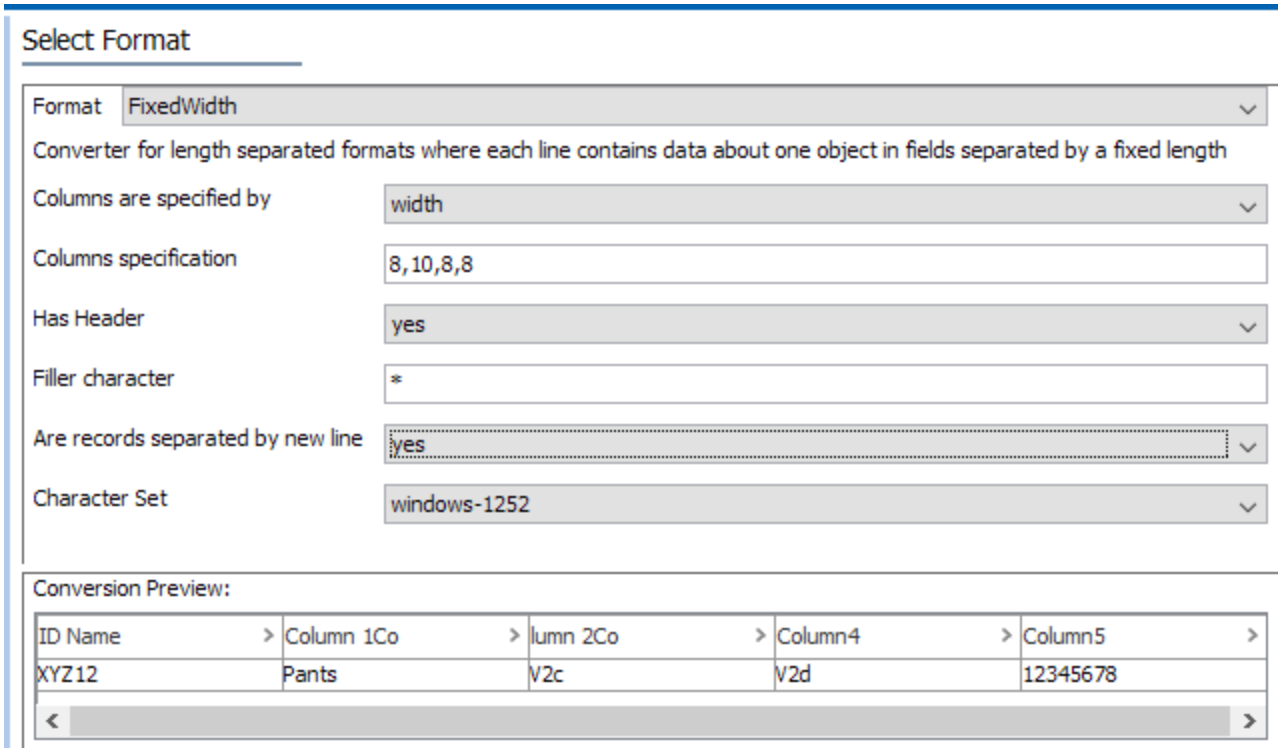
1. **Columns are Specified by** is set to **width**.
2. **Column specification** is set to **8,10,8,8**.
3. **Has Header** is set to **Yes**.
4. **Are records separated by new lines** is set to **Yes**.
5. **Character set** is set to **UTF-8**.

   The Conversion Preview displays the filler characters.



6. **Filler character** is set to * (which is being used to take up unneeded space between values)

   The Conversion Preview no longer displays the filler characters.


Select Format
Format
FixedWidth
Converter for length separated formats where each line contains data about one object in fields separated by a fixed length
Columns are specified by
width
Columns specification
8,10,8,8
Has Header
yes
Filler character
*
Are records separated by new line
yes
Character Set
windows-1252
Conversion Preview:
ID Name
Column 1Co
lumn 2Co
Column4
Column5
XYZ12
Pants
V2c
V2d
12345678

Flatplan Excel Format

Two Excel formats are available for the export and import of Flatplans and Flatplanner publications: **Flatplan Excel** and **Publication Excel**.

Format Availability

Flatplan Excel is available for selection in:

- IIEP - refer to **Creating an Inbound Integration Endpoint**
- Import Manager - refer to **Creating a Data Import**
- Export Manager - refer to **Creating a Data Export**

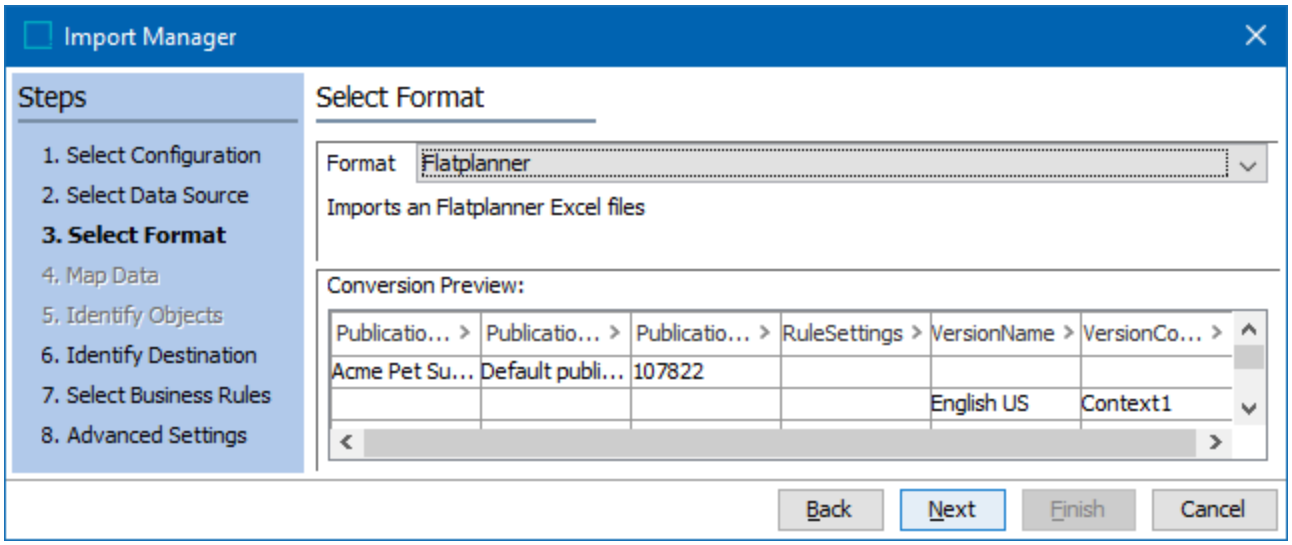
Mapping

Mapping data is not required and the Map Data step is disabled.

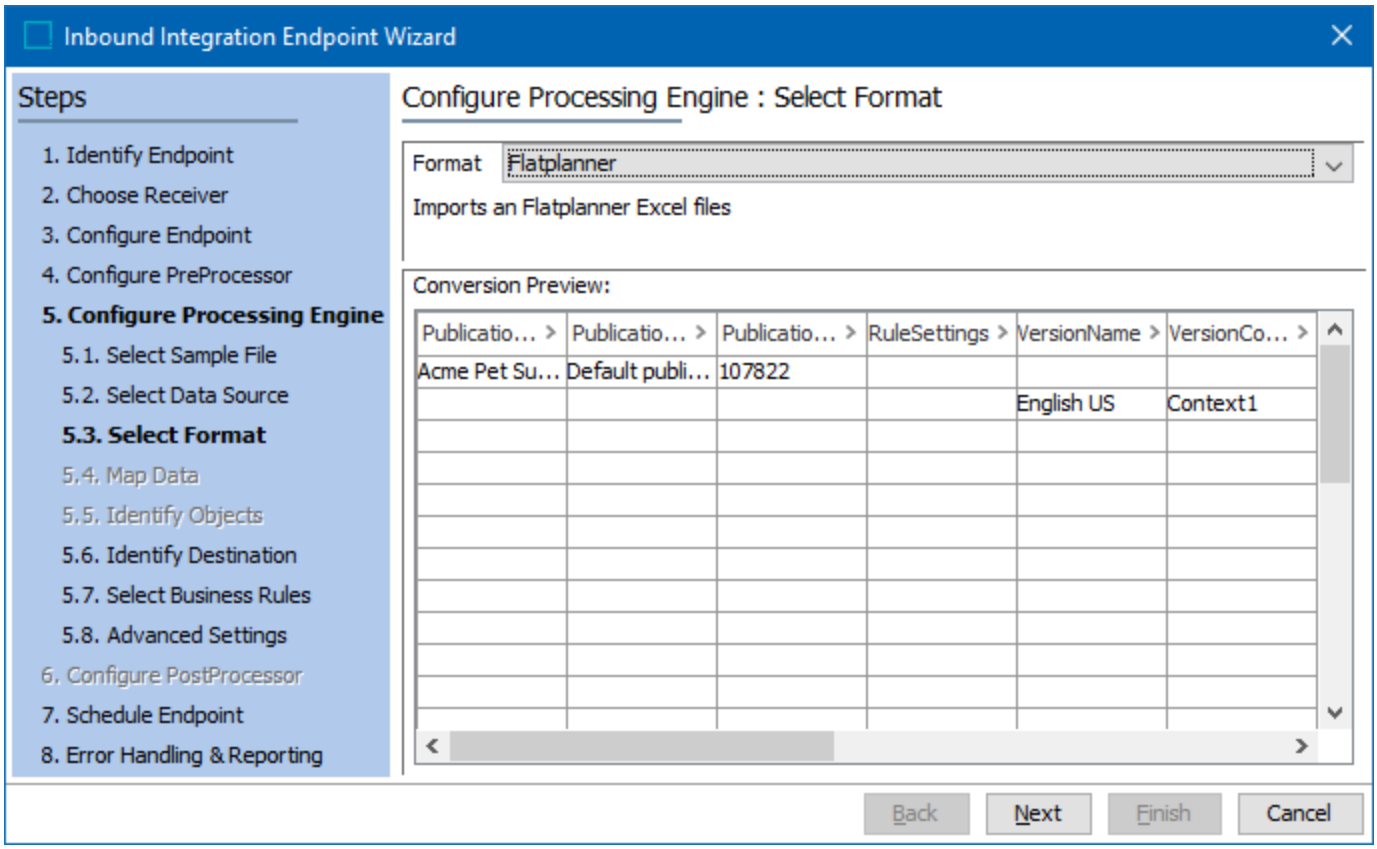
Inbound Data

The Flatplan Excel format is displayed as 'Flatplanner' format for inbound data.

Import Manager



IIEP

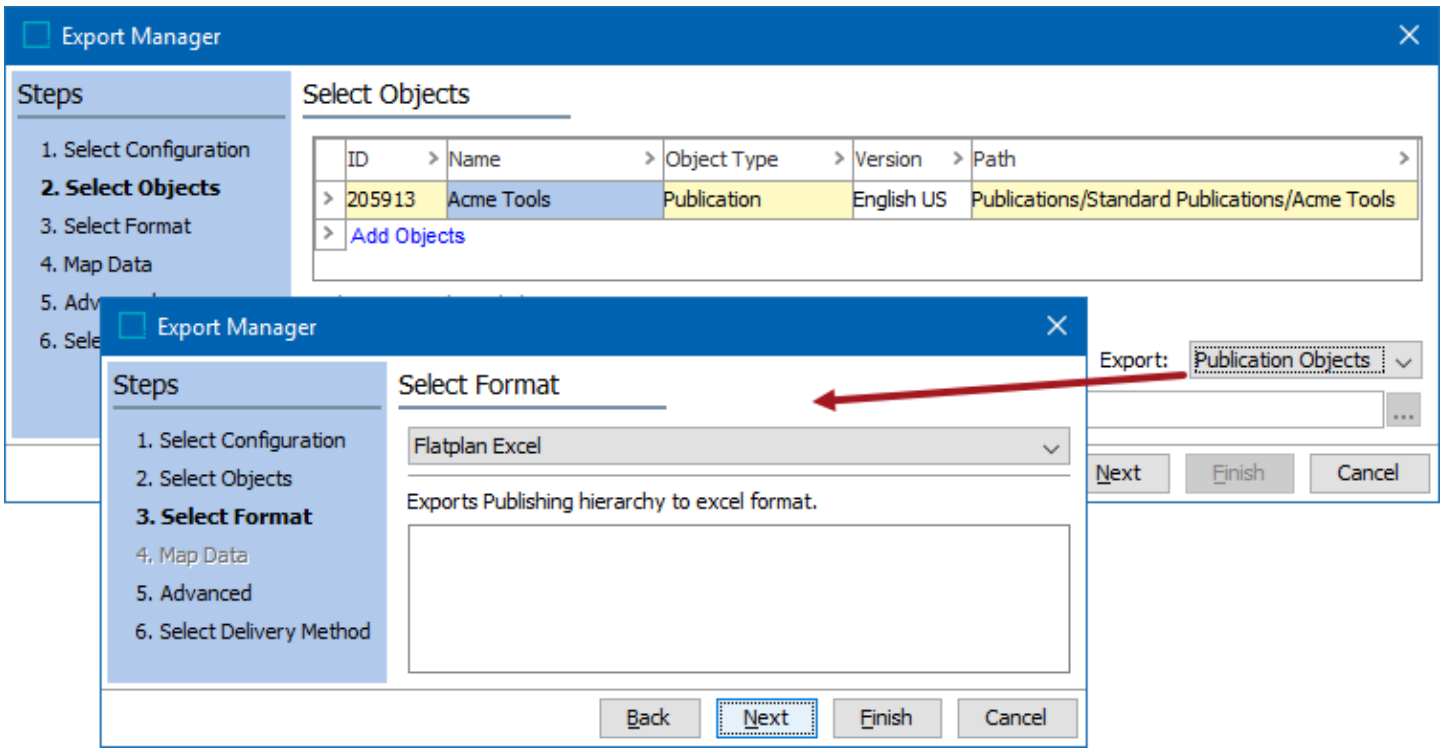


Outbound Data

For step-by-step instructions on how to export a Publication Excel or Flatplan Excel spreadsheet from a Flatplanner publication in STEP, refer to **Exporting a Publication Excel Sheet** in the **Publisher (Adobe InDesign Integration)** documentation.

Export Manager

After selecting a publication on Tree, right-click and choose **Export Data Below** to refer to the Flatplan Excel format option in Export Manager.


Export Manager
Steps
1. Select Configuration
2. Select Objects
3. Select Format
4. Map Data
Select Objects
ID
Name
Object Type
Version
Path
205913
Acme Tools
Publication
English US
Publications/Standard Publications/Acme Tools
Add Objects
Export:
Publication Objects
Next
Finish
Cancel
Export Manager
Steps
1. Select Configuration
2. Select Objects
3. Select Format
4. Map Data
5. Advanced
6. Select Delivery Method
Select Format
Flatplan Excel
Exports Publishing hierarchy to excel format.
Back
Next
Finish
Cancel

Generic JSON Format

The Generic JSON format option significantly simplifies the work for STEP administrators and enterprise architects required to have STEP produce JSON (JavaScript Object Notation) messages.

Although the following alternatives provide a large degree of freedom in composing the JSON messages, they also require either programming skills and code management or custom development:

- The OIEP **Business Rules Based Message Processor** option can generate JSON messages constructed via business rules and the public Scripting API.
- Extensions developed either via the **Extension API** or by Stibo Systems. Refer to the **STEP API Documentation**, at [system]/sdk or accessible from the Start Page.

While the Generic JSON option is not as flexible as the others, it requires no programming skills and can be configured by users with basic knowledge of STEP and JSON. Refer to the **Template, Mapping, and Output** section and the **Generic JSON Configuration Examples** section below for illustrations.

Format Availability

Generic JSON is available for selection in:

- IIEP- refer to **Creating an Inbound Integration Endpoint**
- Import Manager- refer to **Creating a Data Import**
- Export Manager - refer to **Creating a Data Export**
- OIEP - refer to **Creating an Outbound Integration Endpoint**

Mapping

This format requires creating a data map between STEP and the data being processed, and may also include data transformations. For details, refer to **Data Mapping**.

Inbound Data

The JSON file is parsed via a template that extracts data and represents it in a tabular format. This requires that you have a representative source file, namely, one where all the different elements that can occur, and that you want to handle, are represented.

The following is an example of a Generic JSON template:

```
1    {
2      "Products": [
```

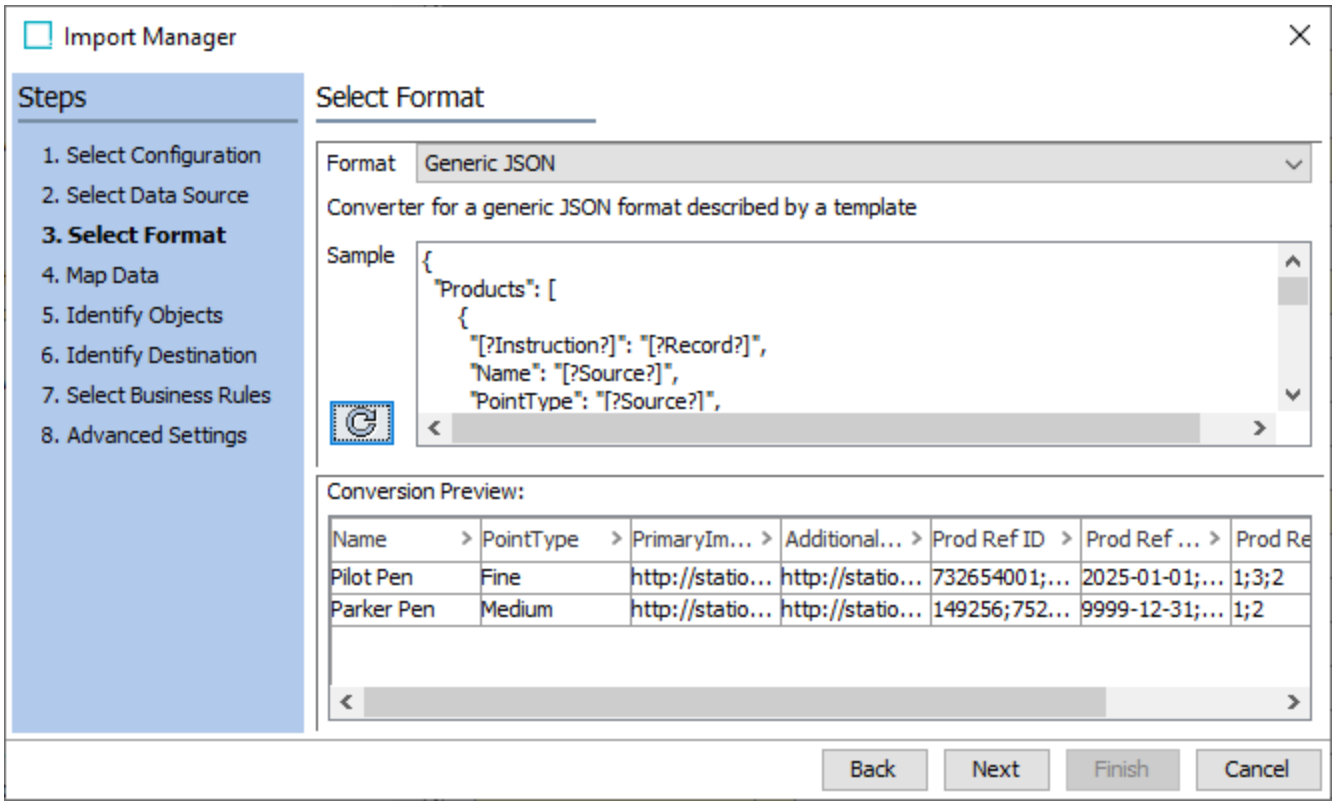
```
3           {
4             "Name": ""
5           }
6         ]
7     }
```

In the example, nodes are matched withing the Products array. Below each product, a Name is specified.

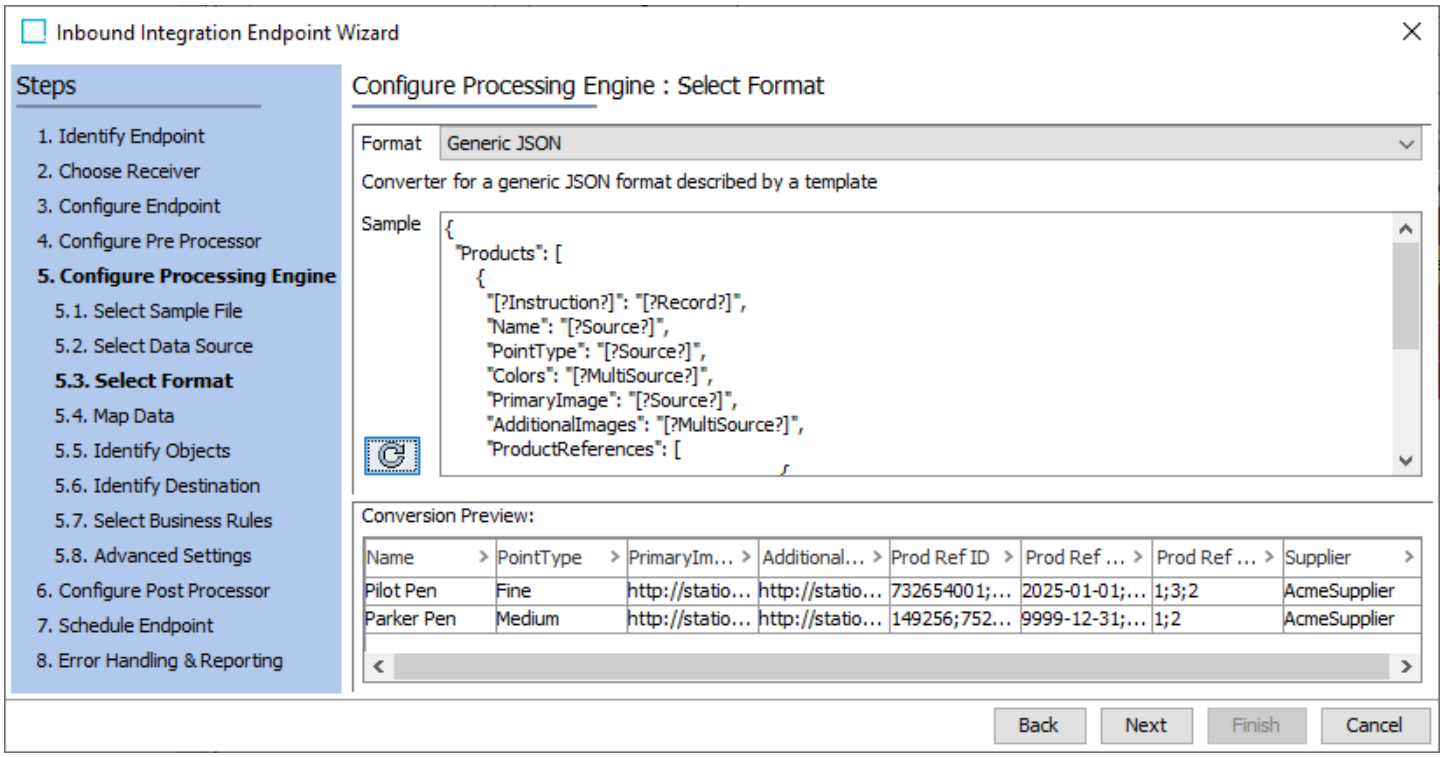
For details about the available processing instructions, refer to the **Generic JSON Inbound Processing Instructions** topic in the **Data Exchange** documentation.

For details about mapping the processing instructions to STEP data, refer to **Inbound Map Data Options** topic in the **Data Mapping** section of the **Data Exchange** documentation.

Import Manager



Inbound Integration Endpoint (IIEP)



Outbound Data

Generic JSON export supports the following object types: products, entities, classifications, and assets, and further allows exporting attribute groups, attributes, data container types, reference and link types, object types, LOVs, and units.

The same parameters are available in both Export Manager and OIEP:

- **Template** - For a Generic JSON export, a template including instructions and placeholders determines how data should be handled, then STEP data is mapped into the JSON template. For details, refer to **Outbound Map Data Options** documentation.

  Since no undo functionality is available in the Template field, it is good practice to use an external source-code editor for creating and editing a template. To test a template's validity for output, paste the code into STEP.

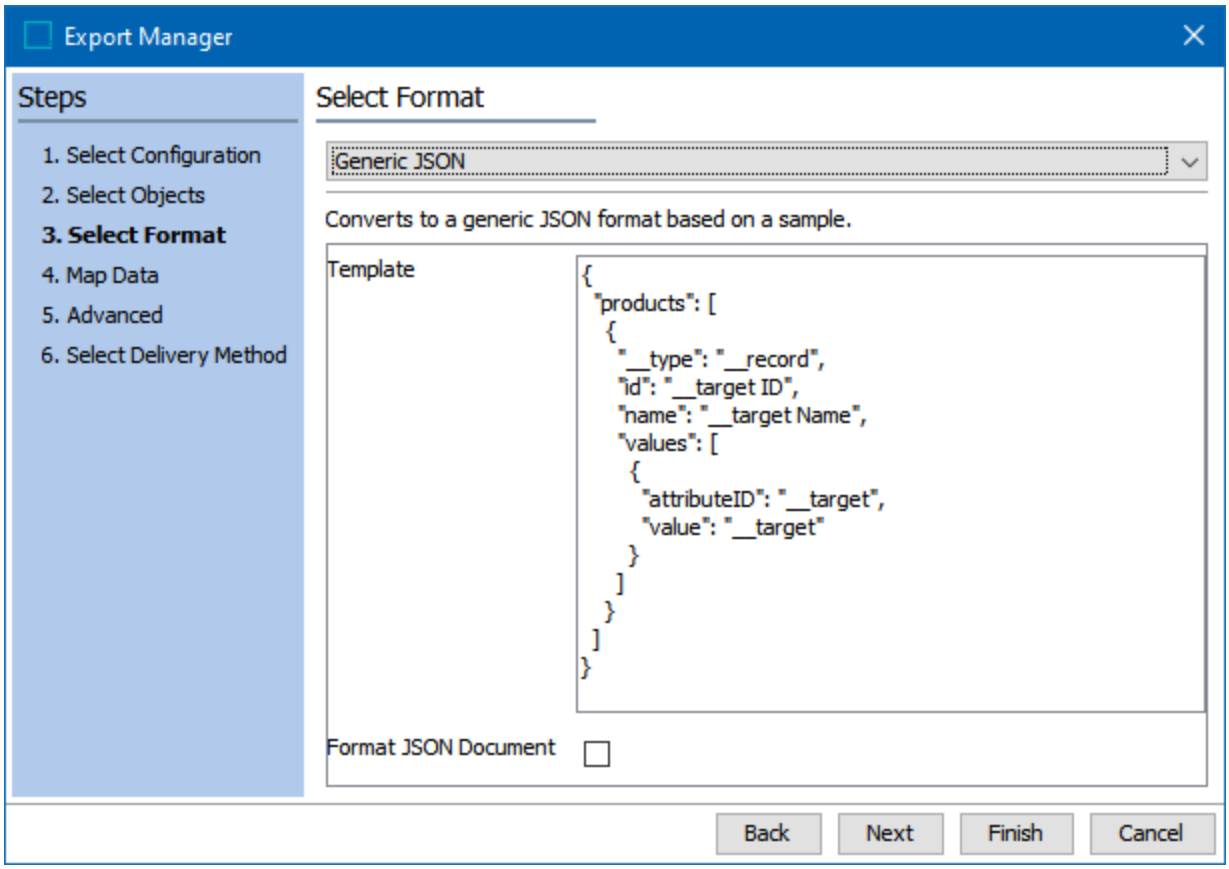
- **Format JSON Document** - This option is disabled by default. This means that the data is output into a single line of comma-separated attribute / value pairs as shown below.

```
{"products":[{"id":"Print_Article_103130","name":"CSHSOVE200WH 1","values":[{"attributeID":"Acme","va
Anrufe oder Kommunikation über Sprachchat;Über-Ohr-Design für optimale Passform;Verkabelte und drahtl
wichtig - perfekt für unterwegs;Kompatibel mit Apple und Android Mobilgeräten","value":"Ohrpolster au
enjoy audio. Integrated microphone to make calls or communicate via voice chat. Over-ear design for c
hones","value":"Headphones"},{"attributeID":"Over-Ear","value":"Over-Ear"},{"attributeID":"CSHSOVE200
ributeID":"Wireless","value":"Wireless"},{"attributeID":"Over-Ear","value":"Over-Ear"}]},{"id":"Print
form","value":"Integriertes Mikrofon für Anrufe oder Kommunikation über Sprachchat;In-Ohr-Design für
,{"attributeID":"CSHSIE200WH 5","value":"CSHSIE200WH 5"},{"attributeID":"Up to 333 feet","value":"Up
0WH 2","value":"CSHSOVE200WH 2"},{"attributeID":"Wireless","value":"Wireless"},{"attributeID":"Over-E
n-Ear"}]},{"id":"Print_Article_104430","name":"CSHSIE200BLK 6","values":[{"attributeID":"Acme","value
rystal clear sound and an optimal fit. An integrated microphone enables phone calls or communication
RN 2"},{"attributeID":"Wireless","value":"Wireless"},{"attributeID":"In-Ear","value":"In-Ear"}]},{"id
lue":"16 mm"}]},{"id":"3","name":"ThickPad X1 Carbide 3","values":[{"attributeID":"16 mm","value":"16
ues":[{"attributeID":"16 mm","value":"16 mm"},{"attributeID":"12780","value":"12780"}]},{"id":"231911
```

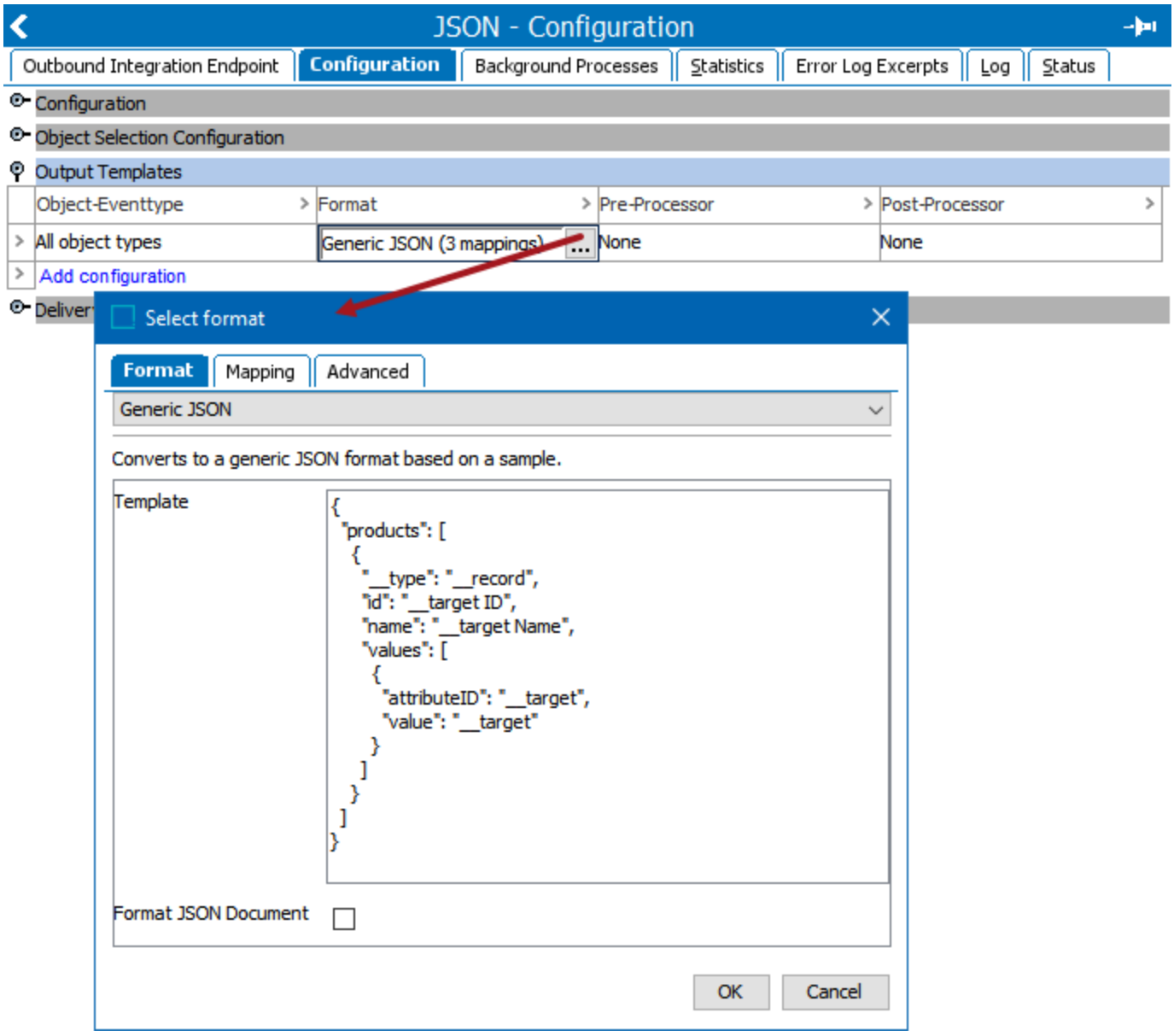

Check the 'Format JSON Document' parameter to export the data in a formatted output for development and debugging purposes.


```
{
  "products" : [
    {
      "id" : "Print_Article_103130",
      "name" : "CSHSOVE200WH 1",
      "values" : [
        {
          "attributeID" : "Acme",
          "value" : "Acme"
        },
        {
          "attributeID" : "Kristallklarer Klang für den Audiogenuss;Integriertes Mikrofon für Anrufe
          "value" : "Kristallklarer Klang für den Audiogenuss;Integriertes Mikrofon für Anrufe oder K
        },
        {
          "attributeID" : "Headset to enjoy crystal clear sound and an optimal fit. An integrated mic
          "value" : "Headset to enjoy crystal clear sound and an optimal fit. An integrated microphon
        },
        {
          "attributeID" : "Headset Über-Ohr weiß",
          "value" : "Headset Über-Ohr weiß"
        },
        {
          "attributeID" : "Headphones",
          "value" : "Headphones"
        },
        {
          "attributeID" : "Over-Ear",
          "value" : "Over-Ear"
        },
        {
          "attributeID" : "CSHSOVE200WH",
          "value" : "CSHSOVE200WH"
        },
        {
          "attributeID" : "3.5 mm",
          "value" : "3.5 mm"
        },
        {
          "attributeID" : "Stereo",
          "value" : "Stereo"
        },
        {
```

Export Manager



Outbound Integration Endpoint (OIEP)



Generic JSON Inbound Processing Instructions

Descriptions and examples of the available processing instructions used by STEP within Generic JSON are covered in the following sections:

- Instruction Definition in Generic JSON
- Record Processing Instruction in Generic JSON
- Source Processing Instructions in Generic JSON
- MultiSource Processing Instructions in Generic JSON
- SourceID Processing Instructions in Generic JSON
- Repeated Processing Instructions in Generic JSON
- DimensionPointID Processing Instructions

For examples of an input document, template, and the conversion preview, refer to the following sections:

- Generic JSON Import Simple Example
- Generic JSON Import References and Links Example
- Generic JSON Import Data Containers Example
- Generic JSON Import Advanced Example

Note: Only instructions included in both the document and the generic template are processed. Instructions not defined in the template are ignored.

Instruction Definition in Generic JSON

Generic JSON uses a similar character pattern to identify information to transfer as is used in Generic XML, with starting characters of “[?” and ending characters of “?]”, like this: "[?Instruction?]".

The template must be valid JSON; therefore the processing instruction values “[?Record?]” and “[?Repeated?]” must be identified with a name. A template can contain multiple instructions, but there must be exactly one “[?Record?]” processing instruction.

In the example below:

- The “[?Record?]” instruction is used to identify each item in the "Items" array.
- There are two “[?Repeated?]” instructions defined in: “Variants” and “Manuals”.
- The “[?Source?]”, “[?SourceID?]”, and “[?DimensionPointID?]” processing instructions identify values and use the Name from the JSON document.

  The JSON Name can be overridden by a different name within the processing instruction when the optional identifier is used. In this example, “[?Source Description, Short?]” overrides “ConsumerShortDescription” with “Description, Short” when viewed in the Format and Mapping screens of the Import Data wizard.

  Ideally, optional identifiers are defined to match the ID of the attribute in STEP to facilitate mapping using the Auto Map button in the Import Manager.
- STEP extracts data for each “Items” object inside the array, regardless of the number that exists. "Items" is the record object of the template. The record must not be declared as a repeated element.

```
{
    "Items": [
      {
        "[?Instruction?]": "[?Record?]",
        "EAN": "[?Source?]",
        "PrimarySpecs": {
          "ProductInformation": {
            "General": {
              "ProductName": "[?Source?]",
              "ConsumerShortDescription": "[?Source Description, Short?]",
              "AvailableFrom": "[?Source?]"
            }
          },
            "Variants": [
              {
```

```
                    "[?Instruction?]": "[?Repeated?]",
                    "Name": "[?SourceID?]",
                    "Value": "[?Source?]"
                  }
                ],
                "Manuals": [
                    {
                    "[?Instruction?]": "[?Repeated?]",
                    "type": "[?DimensionPointID?]",
                    "value": "[?Source?]"
                    }
        ]
            },
            "SyncProtocols": "[?MultiSource?]"
          }
        ]
    }
```

Record Processing Instruction in Generic JSON

The template must declare which element corresponds to a single record (product, entity, etc.). This is done using the "[?Record?]" processing instruction, which applies to the objects in the array. A template must contain exactly one "[?Record?]" processing instruction.

```
{
  "Products": [
    {
      "[?Instruction?]": "[?Record?]",
      "ID": "42"
    }
  ]
}

Also acceptable:
{
  "Products": [
    {
      "__type": "__record",
      "ID": "42"
    }
  ]
}
```

In this example, STEP extracts data for each product object inside the "Products" array, regardless of the number that exist. "Products" is the record object of the template. The record must not be declared as a repeated element. Although this is a valid template, no data is being supplied by this template, only the record is identified.

Source Processing Instructions in Generic JSON

To extract the value of a name-value pair as a field, place a "[?Source [OptionalIdentifier]?]" processing instruction within the "[?Record?]" scope of the template in place of a value. By default, the heading (column) name of a field is the JSON name of the name-value pair.

```
{
  "Products": [
    {
      "[?Instruction?]": "[?Record?]",
      "ID": "[?Source?]",
      "Name": "[?Source?]",
      "Description": "[?Source?]"

    }
  ]
}
```

In the above example, each record encountered in the "Products" array is transformed into a row with data for the columns ID, Name, and Description in the Import Manager wizard.

To make the mapping step easier using the Auto Map button, use your own identifiers that match STEP attribute IDs by defining an optional identifier. As shown in the "Text" name-value pair in the example below, the Text field will display with a heading of 'Description' since the optional identifier was used within the "[?Source?]" processing instructions. The optional identifier displays in the Map Data step.

```
{
  "Products": [
    {
      "[?Instruction?]": "[?Record?]",
      "ID": "[?Source?]",
      "Name": "[?Source?]",
      "Text": "[?Source Description?]"

    }
  ]
}
```

Preventing Duplicate Field Names

Fields extracted using the "[?Source?]" and "[?MultiSource?]" instructions can sometimes have the same names, either because of identical JSON names or because of duplication of names explicitly chosen by the template creator. In this situation, the Import Manager ensures that all field names are unique by adding a post-fix of the form (n), where n is the first integer >= 2 for which the combined field name is not already in use.

If a repeated scope contains multiple "[?Source?]" and "[?MultiSource?]" instructions, and one of them results in a name duplication, only that field will be renamed.

Note: Resolve name collisions by providing explicit, non-duplicating names.

MultiSource Processing Instructions in Generic JSON

To extract the values of a JSON array as a multi-value field that can be mapped to a multi-valued attribute / reference / set of data container values in STEP, place a "[?MultiSource [OptionalIdentifier]?]" instruction in place of the array for the attribute of the "[?Record?]" scope:

```
{
  "Products" : [
    {
      "[?Instruction?]": "[?Record?]",
      "ID" : "[?Source?]",
      "Keywords" : "[?MultiSource?]"
    }
  ]
}
```

Within the same scope, multiple values in an array using a "[?MultiSource [OptionalIdentifier]?]" declaration contribute to the same multi-value field. The input document below uses the template above.

```
{
  "Products" : [
    {
      "ID" : "42",
      "Keywords" : [
        "Red",
        "Green",
        "Blue"
        ]
    }
  ]
}
```

It produces the fields 'ID' and 'Keywords' with values ID=42 and Keywords=Red;Green;Blue. Note that if a "[?Source?]" instruction is used for mapping to an array, only the first value in the array is extracted from the document. A document with the 'Keywords' array in the 'Products' array could be processed by using the following template to only extract the first value:

```
{
  "Products" : [
    {
      "[?Instruction?]": "[?Record?]",
      "ID" : "[?Source?]",
      "Keywords" : "[?Source First Keyword?]"
    }
  ]
}
```

Like the "[?Source?]" instruction, the "[?MultiSource?]" instruction also supports explicitly specifying a field name using an optional identifier as shown above with the "First Keyword" optional identified instead of "Keywords". For more information, refer to the **Preventing Duplicate Field Names** section of the **Source Processing Instructions in Generic JSON** documentation.

SourceID Processing Instructions in Generic JSON

"[?SourceID?]" instructions must be placed inside repeated scope and are used in conjunction with a "[?Repeated?]" processing instruction.

- Only one "[?SourceID?]" is allowed for each repeated scope.
- "[?SourceID?]" is used to extract the identifier for a repeated element when the identifier is a value in a JSON object.

The following is an example of an input document that requires the repeated instruction and ID source declarations to happen on different objects:

```
{
  "Products": [
    {
      "ID": "EXA-5002-1004",
      "Attributes": [
        {
          "AttributeID": "52",
          "Value": "Some description text"
        },
        {
          "AttributeID": "77",
          "Value": "53 kg"
        }
      ]
    }
  ]
}
```

The following is a Generic JSON template for processing the input document:

```
{
  "Products": [
    {
```

```
          "[?Instruction?]": "[?Record?]",
          "ID": "[?Source ID?]",
          "Attributes": [
            {
              "[?Instruction?]": "[?Repeated?]",
              "AttributeID": "[?SourceID?]",
              "Value": "[?Source?]"
            }
          ]
        }
      ]
    }
```

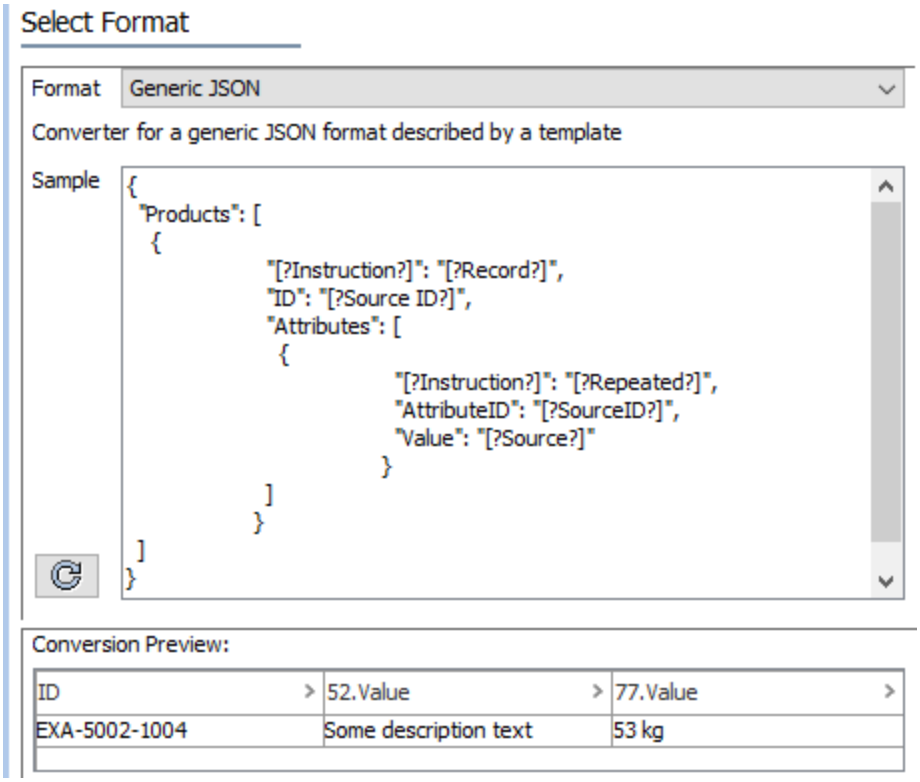
When used on the input document, this template outputs the fields 'ID', '52.Value', '77.Value'. In this case, the subtree from 'Value' and down is said to be in repeated scope.

Follow the steps and use the following template to extract the input document:

1. Copy the above Generic JSON input to a new file and save as a JSON file.
2. Open Import Manager, select the file as the Data Source Filename, and click 'Next.'
3. Enter or paste the template below in the Import Manager, and generate a conversion preview by clicking the refresh button, .

```
    {
      "Products": [
        {
          "[?Instruction?]": "[?Record?]",
          "ID": "[?Source ID?]",
          "Attributes": [
            {
              "[?Instruction?]": "[?Repeated?]",
              "AttributeID": "[?SourceID?]",
              "Value": "[?Source?]"
            }
          ]
        }
      ]
    }
```

For the input document, this template generates the fields **ID**, **52.Value** and **77.Value**.



Repeated Processing Instructions in Generic JSON

Additional declarations are required when extracting data from repeated structures below the record node. Declaration of a repeated node happens with the "[?Repeated [OptionalSpaceSeparatedFilters]?]" processing instruction. The "[?Repeated?]" instruction must be placed inside "[?Record?]" scope. For example:

```
{
  "Products": [
    {
      "ID": "EXA-5002-1004",
      "Values": [
        {
          "AttributeID": "52",
          "Value": "Some description text"
        },
        {
          "AttributeID": "77",
          "Value": "53 kg"
        }
      ]
    }
  ]
}
```

Note that the "Values" array contains multiple objects. To extract data from both occurrences, the template must declare that the array of objects is repeated, which means that a set of fields has to be generated for each node encountered. The template also has to declare an ID source in each object.

The following template could be used to extract the data from the above Generic JSON:

```
{
  "Products": [
    {
      "[?Instruction?]": "[?Record?]",
      "ID": "[?Source ID?]",
```

```
          "Values": [
            {
              "[?Instruction?]": "[?Repeated?]",
              "AttributeID": "[?SourceID?]",
              "Value": "[?Source?]"
            }
          ]
        }
      ]
    }
```

The ID source is used to map each occurrence of a repeated object to a specific set of fields of the form [Identifier].[SourceTagName]. This is required to ensure that related repeated nodes map to the same fields across different records.

By default, a repeated scope produces a set of fields for each ID Source value encountered in the input document. Using the 'optional space separated filters' allows you to select values from specific ID keys or to guard against additional columns being generated as input documents evolve in the future.

```
    {
      "ARTICLE": {
        "ARTICLE_PRICE_DETAILS": {
          "DATETIME": {
            "type": "valid_start_date",
            "DATE": "2010-01-01"
          },
          "PRODUCT_PRICE": [
            {
              "price_type": "net_list",
              "PRICE_AMOUNT": 11.08,
              "PRICE_CURRENCY": "EUR",
              "TAX": 0.19,
              "LOWER_BOUND": 1
            },
            {
              "price_type": "nrp",
              "PRICE_AMOUNT": 23.05,
```

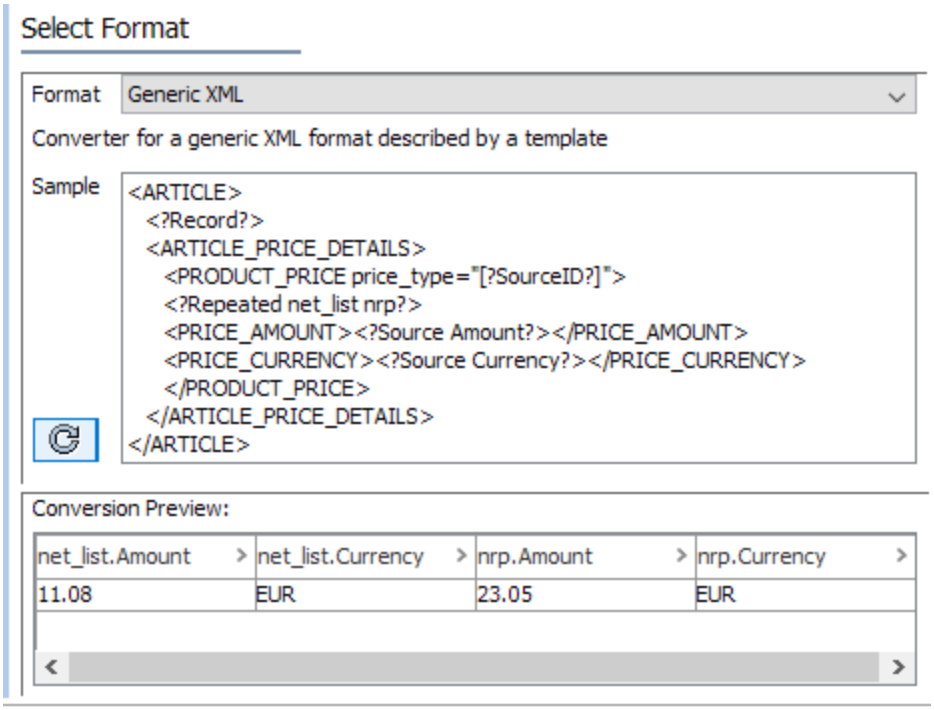
```
              "PRICE_CURRENCY": "EUR",
              "TAX": 0.19,
              "LOWER_BOUND": 1
            },
            {
              "price_type": "employee",
              "PRICE_AMOUNT": 8.52,
              "PRICE_CURRENCY": "EUR",
              "TAX": 0.19,
              "LOWER_BOUND": 1
            }
          ]
        }
      }
    }
```

The price and currency information can be extracted for the net_list and nrp price types only by following the proceeding steps and using the following template:

1. Copy the above Generic JSON to a new file, and save as a JSON file
2. Open Import Manager, select the file as the Data Source Filename, and click 'Next'
3. The user can enter / paste the template below in the Import Manager, and a conversion preview can be generated by clicking the refresh button, .

```
    {
      "ARTICLE": {
       "[?Instruction?]": "[?Record?]",
        "ARTICLE_PRICE_DETAILS": {
          "PRODUCT_PRICE": [
            {
              "[?Instruction?]": "[?Repeated net_list nrp?]",
              "price_type": "[?SourceID?]",
              "PRICE_AMOUNT": "[?Source Amount?]",
              "PRICE_CURRENCY": "[?Source Currency?]"
            }
          ]
        }
```

```
14          }
15      }
```



This produces the fields:

- net_list.Amount = 11.08
- net_list.Currency = EUR
- nrp.Amount = 23.05
- nrp.Currency = EUR

The "[?Repeated?]" instruction can take an arbitrary number of filter terms separated by white-space characters. When no terms are defined, the filtering functionality is disabled.

DimensionPointID Processing Instructions in Generic JSON

Values in Generic JSON for the same attribute in different languages could be represented using the same ID with an additional attribute stating the language. This normally causes a name collision in Generic JSON. However, Generic JSON can import such dimension-dependent data using the "[?DimensionPointID?]" instruction. This instruction can be used in a "[?Repeated?]" block, in combination with "[?SourceID?]", or alone. When the "[?DimensionPointID?]" instruction is used, the name of the field extracted is appended with the value of the "[?DimensionPointID?]" in square brackets, thus avoiding the name collision.

The values appended to the column names using the "{?DimensionPointID?]" do not have to match the Dimension Point IDs, as this can be handled in the Map Data step of the Import Manger. It is important to note that the use of the "[?DimensionPointID?]" instruction does not actually import into the specified dimension points; this has to be set up using transformations in the Map Data step on the Import Manager. For more on this functionality, refer to the **Inbound Map Data - Map** topic.

The following example shows how language-dependent data can be extracted using Generic JSON. Note that the ID for both the English and German attribute is the same value, 52.

```
{
  "Products": [
    {
      "ID": "EXA-5002-1004",
      "Attributes": [
            {
                "AttributeID": 52,
                "Lang": "Eng",
                "Value": "Some English descriptive text"
            },
            {
                "AttributeID": 52,
                "Lang": "De",
                "Value": " Einige beschreibenden Text "
            }
        ]
    }
  ]
}
```

The following Generic JSON template extracts the ID source from a value and appends the value inside the "Lang" attribute in square brackets, e.g., '52[Eng]' and '52[De]'. The template applies this format so the attributes can be mapped to the respective languages with a Dimension Point Transformation and selection of the corresponding context.

```
    {
      "Products": [
        {
          "[?Instruction?]": "[?Record?]",
          "ID": "[?Source?]",
          "Attributes": [
                {
                    "[?Instruction?]": "[?Repeated?]",
                    "AttributeID": "[?SourceID?]",
                    "Lang": "[?DimensionPointID?]",
                    "Value": "[?Source?]"
                }
            ]
        }
      ]
    }
```

Generic JSON Import Simple Example

The following is an example of an import document and template that uses most of the features present in a Generic JSON template. This example uses the Import Manager, but an IIEP can also be configured with the same input and template.

Input Document

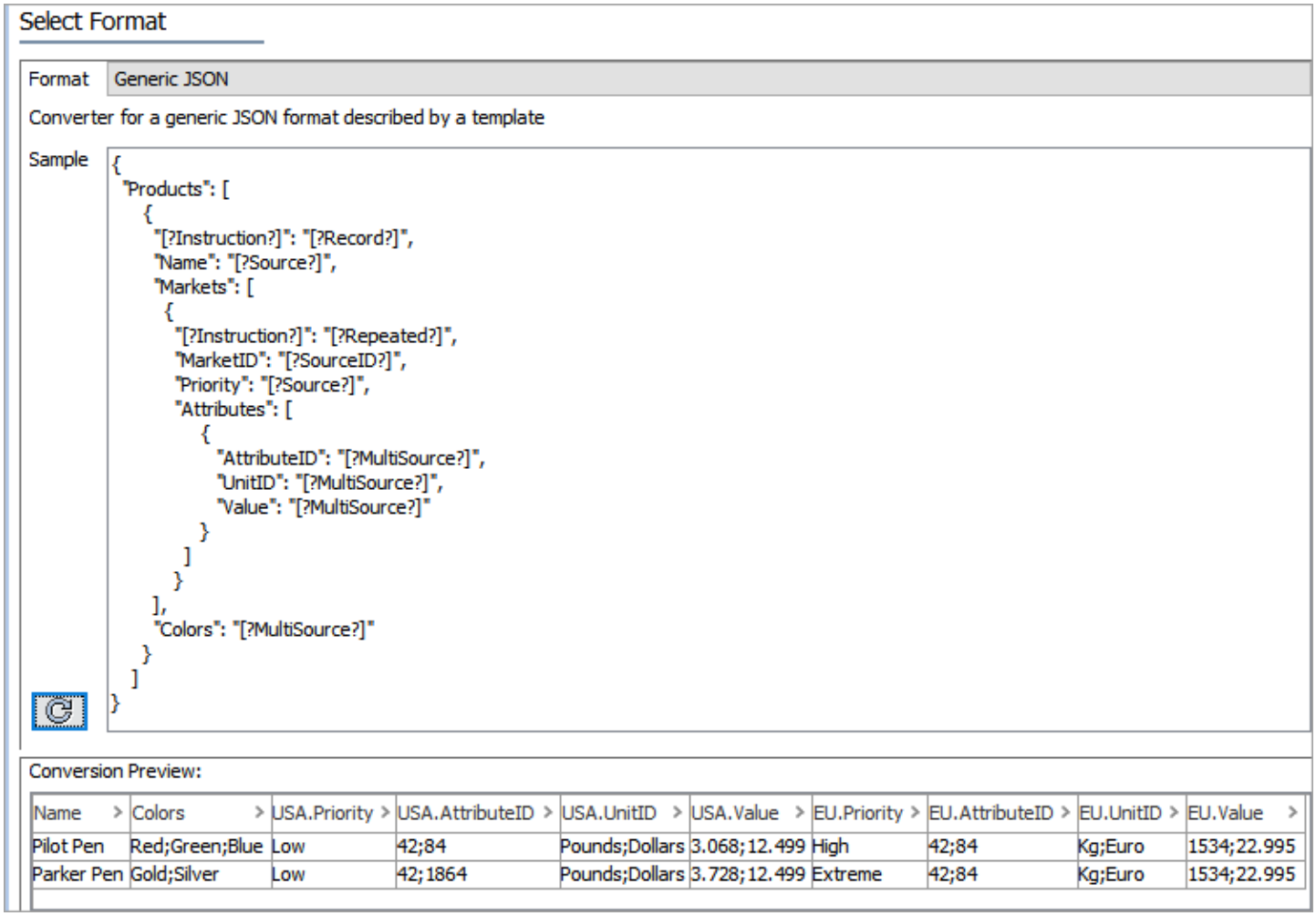
Copy the text from this topic in online help to a new file and save it as a JSON file. Open the Import Manager and select the file as the Data Source Filename and click 'Next.'

Template

Copy the text from this topic in online help and paste it in the Import Manager Select Format step (refer to the image in the Conversion Preview section below), Sample field.

Conversion Preview

Click the Sample reload button . The Conversion Preview area displays the results of the input document against the provided template.



The Map Data step (defined in the **Inbound Map Data - Map** topic) presents the Generic JSON data for further mapping, if necessary, as well as transformations. To learn more about inbound transformations, refer to the **Inbound Map Data - Transform** documentation.

Generic JSON Import References and Links Example

Using the Import Manager, this example demonstrates how to take a JSON input document to create a template, then modify the template to resolve errors and accurately read the input, and prepare the data for mapping.

The following is an example of a JSON Import for product records that contain a Name, PointType, PrimaryImage, AdditionalImages, ProductReferences, and Supplier attributes. The Name, PointType, PrimaryImage and Supplier attributes are single values and use the [?Source?] instruction in the JSON template below. The AdditionalImages array and ProductReferences object array included in the import example are mapped to references in STEP and use the "[?MultiSource?]" instruction in the JSON template below to handle multiple reference targets. There are some limitations with what can be transformed by the JSON template and what can be mapped using the Import Manager, therefore references of different types in an array cannot be mapped using the [?Repeated?] and [?SourceID?] instructions and when multiple references of the same type are included in an object array, the metadata cannot be mapped. However, a single reference target can be mapped to accept metadata on the reference using the Reference Meta-Data mapping option.

Input Document

For this example import, copy the text from the online help version of this topic to a new file and save it as a JSON file. Open the Import Manager and select the file as the Data Source Filename and click 'Next.'

Template

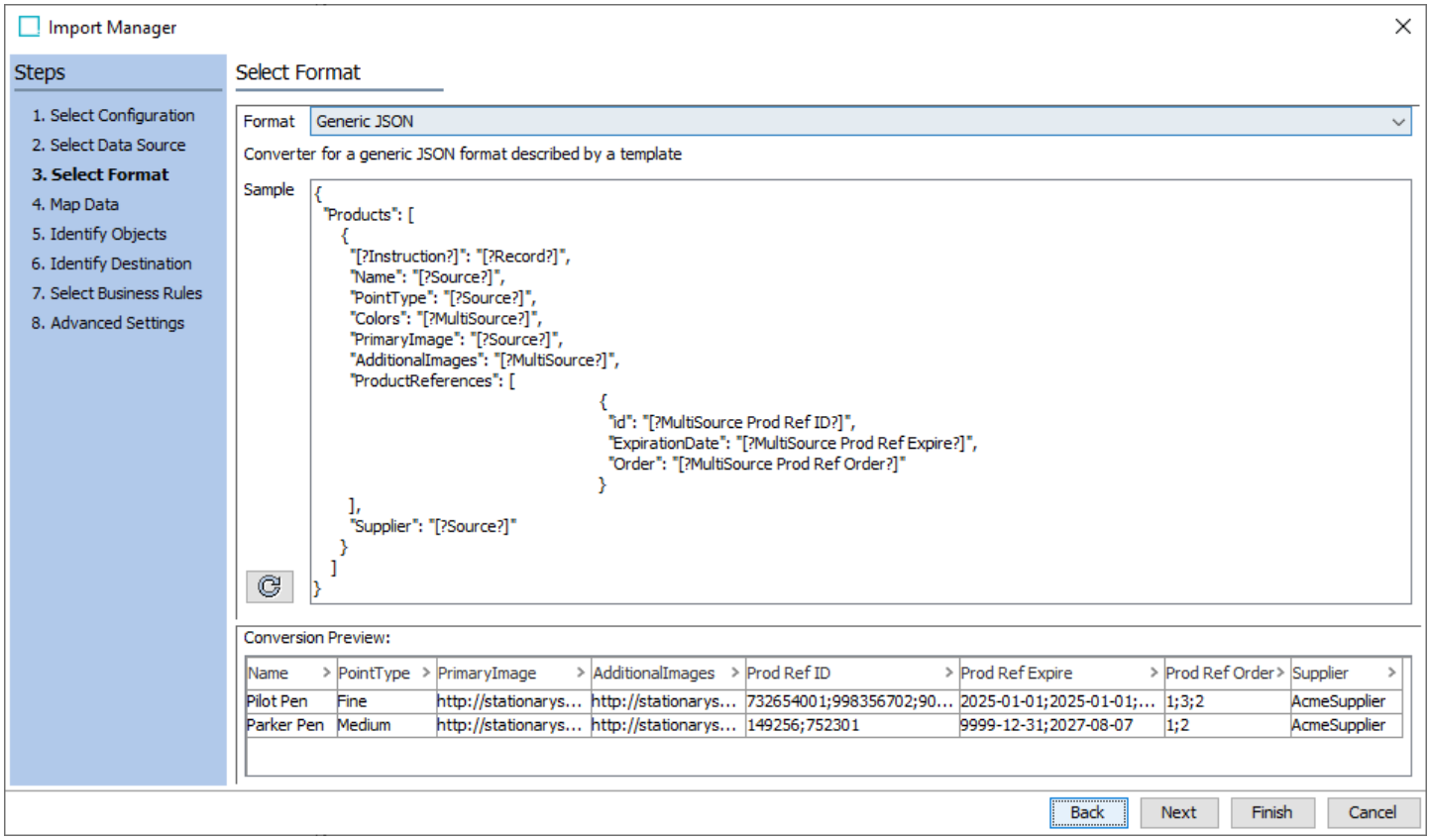
In the template in the online help version of this topic:

- Redundant objects in the "Products" and "ProductReferences" arrays have been removed.
- The "[?Instruction?]": "[?Record?]" definition is added inside the first object in the Products array. "
- [?Source?]" replaces values for "Name", "PointType", “PrimaryImage”, and “Supplier” attributes without using the optional identifiers.
- In the "ProductReferences" array, the "[?MultiSource?]" instruction is added in place of the values for each of the attributes within the respective objects with optional identifiers.

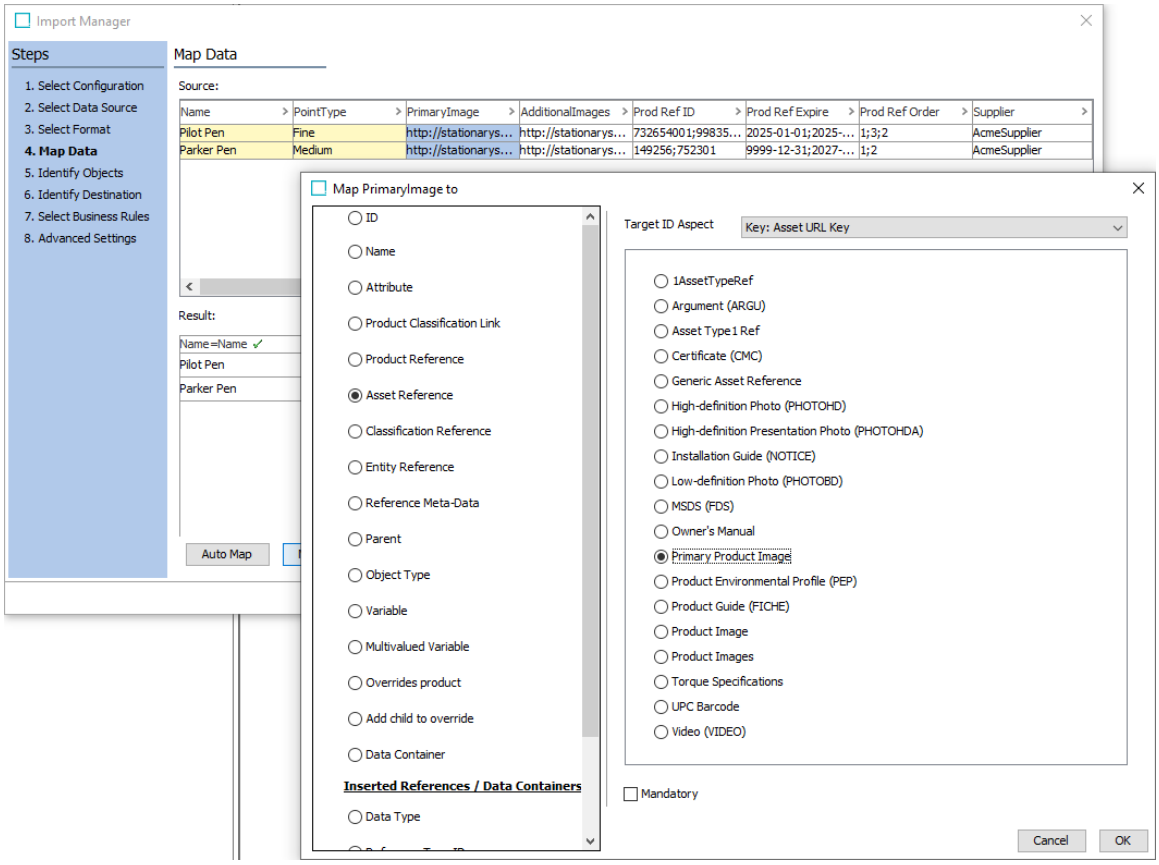
Conversion Preview

In the Conversion Preview section of the Import Manager's Select Format step, the template results in the values from the "ProductReferences" objects in arrays being concatenated with a semicolon separator. There are three product references in the JSON, based on the semicolon-separated values in each column, along with metadata in the adjacent columns.

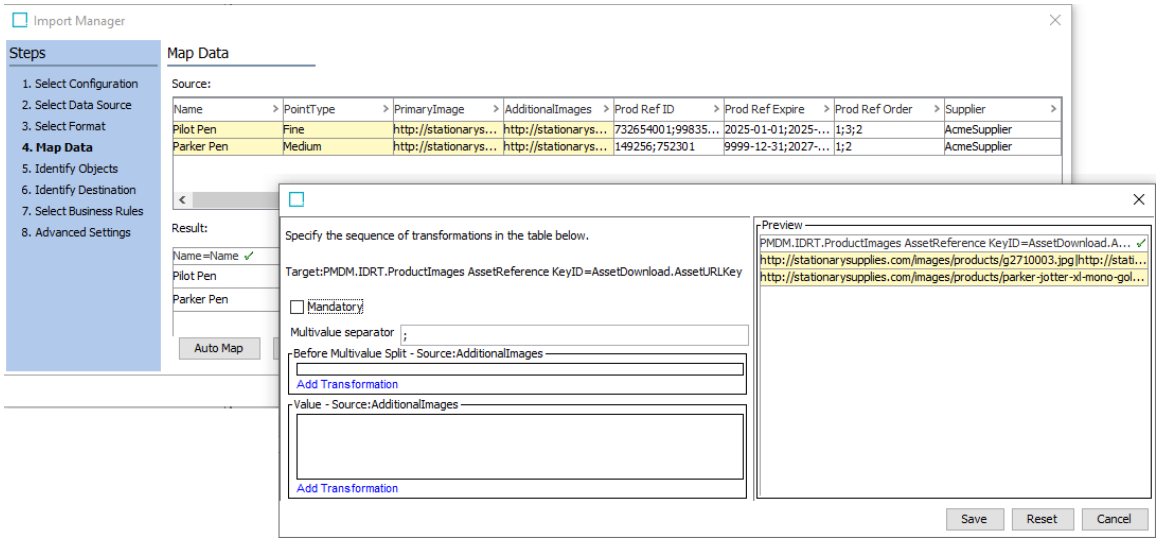
Click the Sample reload button, . The Conversion Preview area displays the results of the input document against the provided template.



After clicking the Next button, the mapping step is presented where the Name and PointType attributes have been mapped. To map the next three columns and the last column, the Reference mapping uses the relevant reference type or classification link types and Target ID Aspect using Keys configured on the system to match the value(s).



Since the Reference Meta-Data mapping does not support multiple references, the example prioritizes the creation of the target references, by mapping to the Prod Ref ID column, while ignoring the columns that include Prod Ref Expire and Prod Ref Expire attributes. For [?MultiSource?] mappings, like “AdditionalImages,” adding a multi-value transformation is required to recognize the individual values, which are matched on the AssetDownload.AssetURLKey.



When the import completes successfully, the new Products are created with references and classification links populated as defined in the JSON document.

Generic JSON Import Data Containers Example

Using the Import Manager, this example demonstrates how to take a JSON input document to create a template, then modify the template to resolve errors and accurately read the input, and prepare the data for mapping.

The following is an example of a JSON Import for a customer record that contains an ID, Name, and a Status attribute which uses "[?Source?]" instructions in the template. The Email Address and Phone Number object arrays included in the import example are mapped to data containers in STEP using the "[?MultiSource?]" instruction in the JSON template below.

Input Document

For this example import, copy the following text to a new file and save it as a JSON file. Open the Import Manager and select the file as the Data Source Filename and click 'Next.'

Note that any number of additional items can be included by repeating the data within the "Customers" array, but in this example, there is only one customer.

```
{
  "Customers": [
    {
      "ID": "EXT-4500-01",
      "Name": "John Smith",
      "Status": "Active",
      "Emails": [
          {
          "EmailAddress": "john.smith@stibosystems.com",
          "Type": "Work",
          "Unsubscribe": "false"
          },
          {
          "EmailAddress": "john.smith@gmail.com",
          "Type": "Personal",
          "Unsubscribe": "false"
          },
          {
          "EmailAddress": "jsmith@yahoo.com",
          "Type": "Other",
          "Unsubscribe": "true"
          }
```

```
            ],
          "PhoneNumbers": [
              {
              "AreaCode": "555",
              "PhoneNumber": "4567890",
              "Type": "Work"
              },
              {
              "AreaCode": "555",
              "PhoneNumber": "3239897",
              "Type": "Mobile"
              }
            ]
        }
      ]
    }
```

Template

In this template:

- Redundant objects in the "Emails" and "PhoneNumbers" arrays have been removed.
- The "[?Instruction?]": "[?Record?]" definition is added inside the first object in the array.
- "[?Source?]" replaces values for "ID", "Name", and "Status", using the optional identifiers in "Name" and "Status".
- In the "Emails" and "PhoneNumbers" arrays, the "[?MultiSource?]" instruction is added in place of the values for each of the attributes within the respective objects.

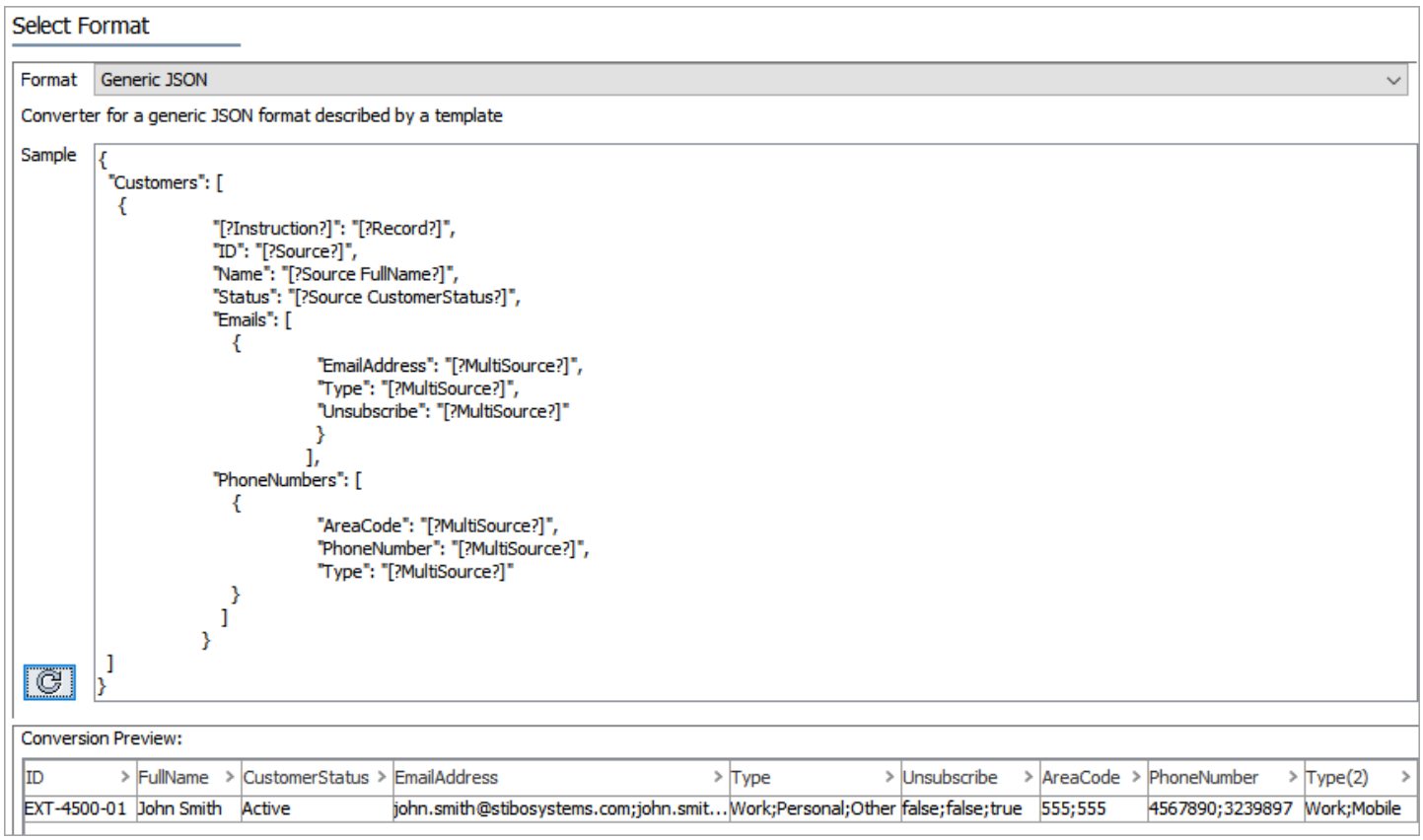
```
    {
      "Customers": [
        {
          "[?Instruction?]": "[?Record?]",
          "ID": "[?Source?]",
          "Name": "[?Source FullName?]",
          "Status": "[?Source CustomerStatus?]",
          "Emails": [
```

```
            {
            "EmailAddress": "[?MultiSource?]",
            "Type": "[?MultiSource?]",
            "Unsubscribe": "[?MultiSource?]"
            }
          ],
        "PhoneNumbers": [
            {
            "AreaCode": "[?MultiSource?]",
            "PhoneNumber": "[?MultiSource?]",
            "Type": "[?MultiSource?]"
            }
          ]
      }
    ]
  }
```

Conversion Preview

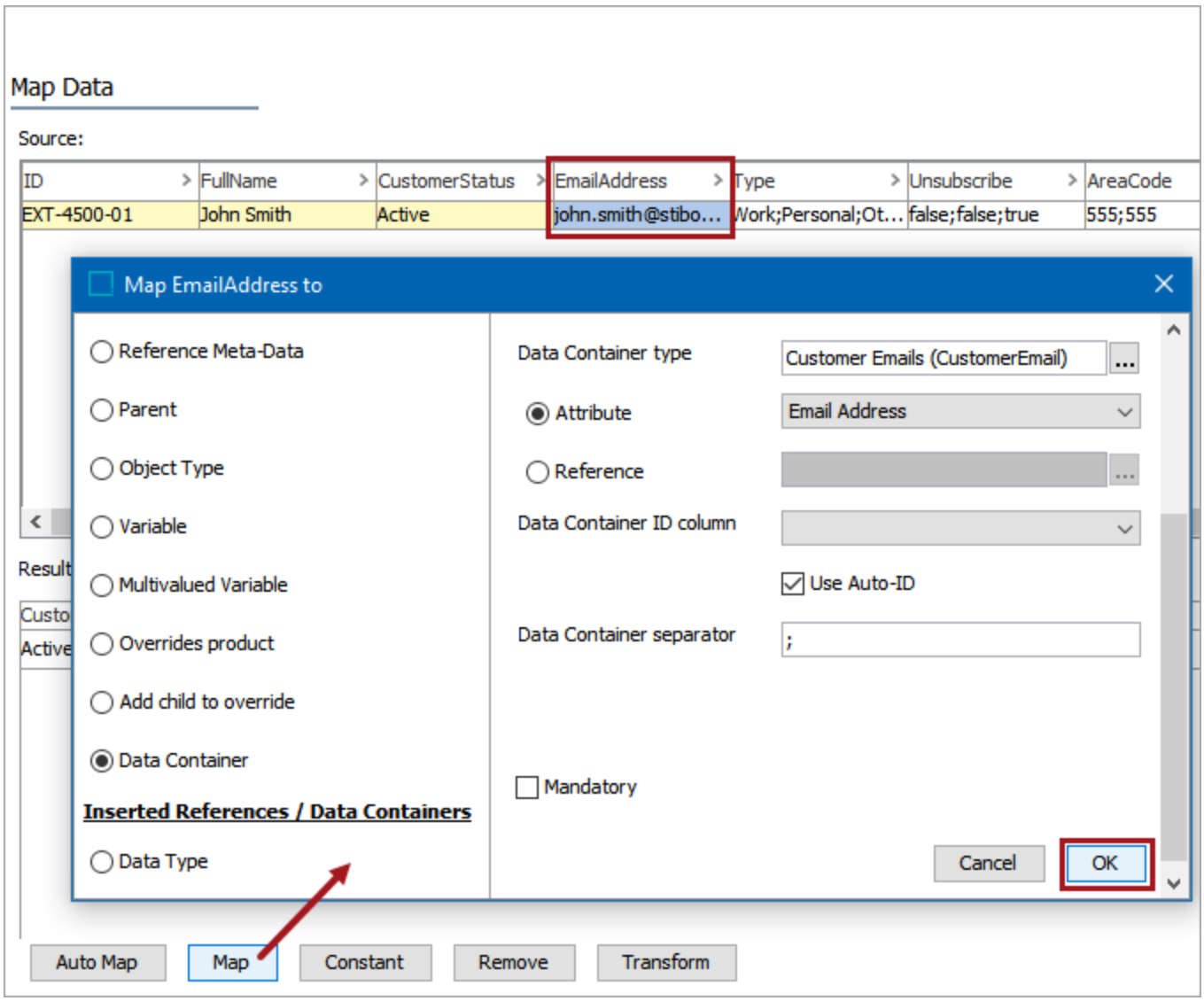
In the Conversion Preview section of the Import Manager's Select Format step, the template results in the values from the "Emails" and "PhoneNumbers" objects in arrays being concatenated with a semicolon separator. There are three email addresses and two phone numbers in the JSON, based on the semicolon-separated values in each column.

Click the Sample reload button . The Conversion Preview area displays the results of the input document against the provided template.

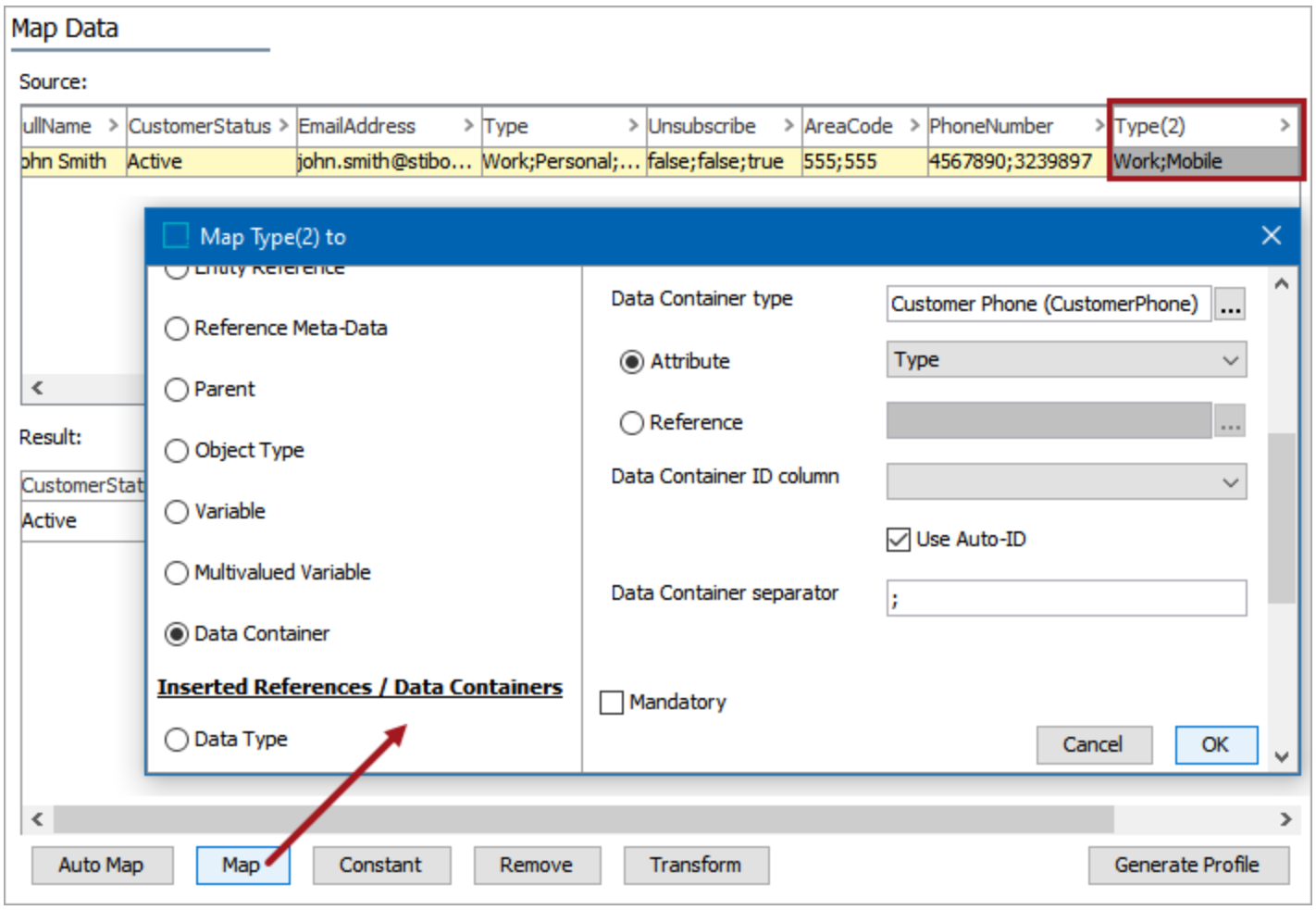


After clicking the Next button, the mapping step is presented where the ID, Name, and Status attributes have been mapped. To map the next three columns, the Data Container mapping uses the same Data Container type that is relevant for Customer Email addresses.

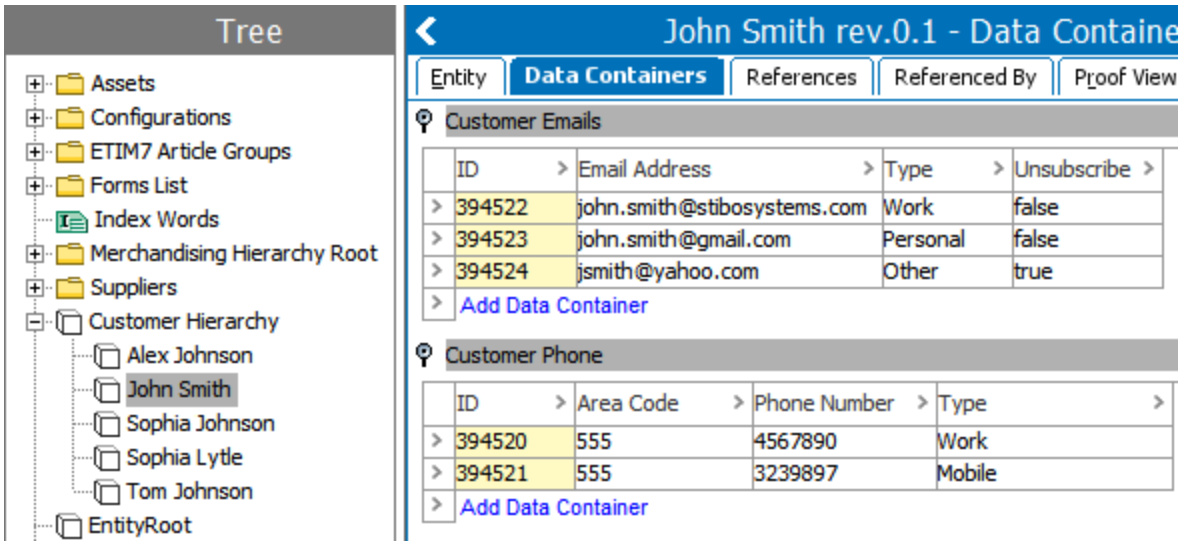
On the 'Map Email Address to' dialog, select the relevant attribute, check the 'Use Auto-ID' checkbox, and change the Data Container separator from the default of '#' to ';'. Click OK and the values are recognized as multiple rows in the data container. The same process is used for the next two columns, with relevant attributes selected.



In the next three columns, make mappings related to the Customer Phone Numbers using a different Data Container type. When switching types, the separator resets to the default.



When the import completes successfully, the new Customer is created with data containers populated as defined in the JSON document:



Generic JSON Import Advanced Example

The following is an example of a complex import document and template that uses most of the features present in a Generic JSON template.

Using the Import Manager, this example demonstrates how to take a JSON input document to create a template, then modify the template to resolve errors and accurately read the input, and prepare the data for mapping.

Input Document

Copy the following text to a new file and save as a JSON file. Open Import Manager and select the file as the Data Source Filename and click 'Next.' Note that any number of additional items can be included by repeating the data within the Item tag.

```
{
    "Items": [
        {
        "EAN": 2700524977488,
        "PrimarySpecs": {
          "ProductInformation": {
            "General": {
              "ProductName": "AC-UZ444",
              "ConsumerShortDescription": "Active 3D Glasses",
              "AvailableFrom": "2015-01-01 00:00:00"
           }
        },
            "Variants": [
              {
                "Name": "Color",
                "Value": "Gray"
              },
              {
                "Name": "Battery",
                "Value": "C2"
              }
            ],
            "Manuals": [
              {
                "type": "ManualDE",
```

```
                    "value": "Manual-9"
                  },
                  {
                    "type": "ManualEN",
                    "value": "Manual-10"
                  }
                  ]
                },
            "SyncProtocols": [
                "Acme",
                "Sony",
                "Sharp"
            ]
          }
        ]
    }
```

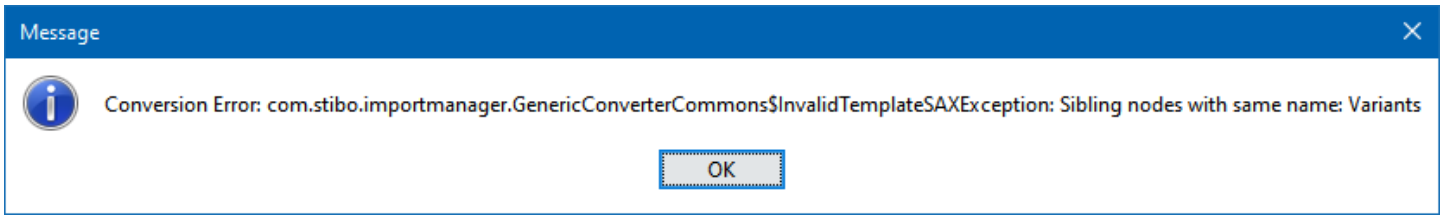
Creating a Template

To create a template from an import file, copy the source file and paste into the Sample field on the Import Manager's Select Format step. Then, remove any duplicate elements. Since there is only a single Item element in the example above, it can be pasted as-is.

Note: When creating a template from a source file, ensure all the included elements that must be handled are represented. Only instructions in both the document and the generic template are processed. Instructions that are not defined in the template are ignored.

Remove Redundant Elements

The source file includes redundant elements that should be removed to create a valid template. When duplicates exist, clicking the Sample reload button displays a message about sibling nodes with the same name:

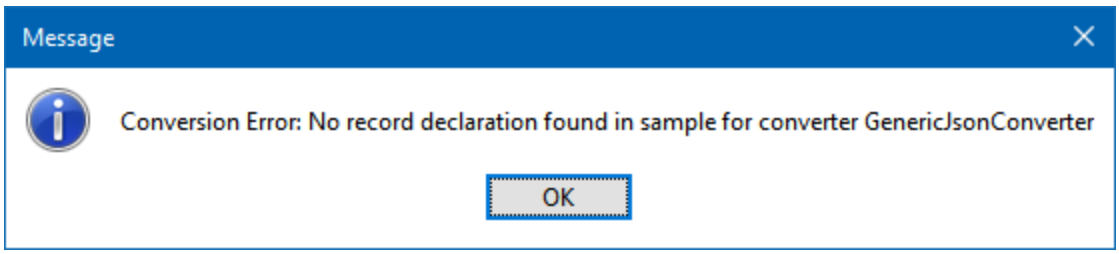


Remove repetitions of the 'Variants', 'Manual', objects, and values in the SynchProtocols array until only one of each remains. The Sample field now displays the following:

```
{
    "Items": [
        {
        "EAN": 2700524977488,
        "PrimarySpecs": {
          "ProductInformation": {
            "General": {
              "ProductName": "AC-UZ444",
              "ConsumerShortDescription": "Active 3D Glasses",
              "AvailableFrom": "2015-01-01 00:00:00"
           }
        },
            "Variants": [
              {
                "Name": "Color",
                "Value": "Gray"
              }
            ],
            "Manuals": [
              {
                "type": "ManualDE",
                "value": "Manual-9"
              }
              ]
            },
        "SyncProtocols": [
            "Acme"
        ]
      }
    ]
}
```

Add Record and Source Instructions

The source file includes actual data, but must include instructions to represent the data instead. To identify missing instructions, click the Sample reload button and a message is displayed about a missing record declaration:



Use the "[?Instruction?]: "[?Record?]" and "[?Source?]" instructions to handle all the cases where you need to extract one value.

```
{
    "Items": [
        {
        "[?Instruction?]": "[?Record?]",
        "EAN": "[?Source?]",
        "PrimarySpecs": {
          "ProductInformation": {
            "General": {
              "ProductName": "[?Source?]",
              "ConsumerShortDescription": "[?Source?]",
              "AvailableFrom": "[?Source?]"
           }
        },
            "Variants": [
              {
                "Name": "Color",
                "Value": "Gray"
              }
            ],
            "Manuals": [
              {
                "type": "ManualDE",
                "value": "Manual-9"
              }
              ]
            },
        "SyncProtocols": [
```

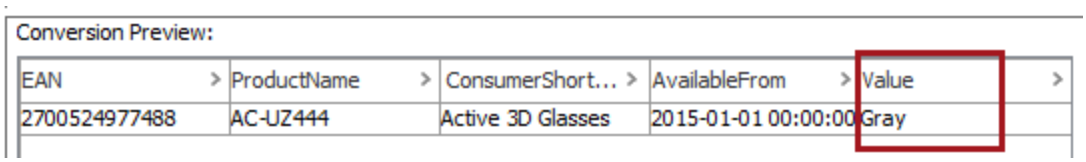
```
                "Acme"
            ]
          }
        ]
    }
```

Add Repeated Instruction

To review how each option affects the outcome, make the following updates in the Sample template and then click the Sample reload button to update the Conversion Preview.

For the "Variants" object, if we insert the "[?Source?]" instruction in the "Value" element as shown below, the result is the value for the first repetition, 'Gray.'

```
    "Variants": [
                {
                   "Name": "Color",
                   "Value": "[?Source?]"
                }
               ]
```


Conversion Preview:

EAN	ProductName	ConsumerShort...	AvailableFrom	Value
2700524977488	AC-UZ444	Active 3D Glasses	2015-01-01 00:00:00	Gray

To get values from both "Variants" elements, there are two options: MultiSource and Source / Repeated / SourceID.

To get the Value from each repetition and be able to map them all to a multi-valued attribute / multiple reference / links targets / data container rows for a column, we can use the "[?MultiSource?]" instruction.

```
    "Variants": [
                  {
                    "Name": "Color",
```

```
                    "Value": "[?MultiSource?]"
                  }
                ]

```

Conversion Preview:

EAN	ProductName	ConsumerShort...	AvailableFrom	Value
2700524977488	AC-UZ444	Active 3D Glasses	2015-01-01 00:00:00	Gray;C2

To map the data to different attributes, as is the case in our example, where the Name value of each repetition identifies the attribute, use these instructions:

- "[?Instruction?]": "[?Source?]"
- one "[?Repeated?]" to indicate that "Variants" is a repeated element
- one "[?SourceID?]" to indicate that the "Name" value is used as an identifier for each repetition

```
    "Variants": [
                  {
                    "[?Instruction?]": "[?Repeated?]",
                    "Name": "[?SourceID?]",
                    "Value": "[?Source?]"
                  }
                ]
```

Conversion Preview:

EAN	ProductName	ConsumerSh...	AvailableFrom	Color.Value	Battery.Value
2700524977488	AC-UZ444	Active 3D Glasses	2015-01-01 00...	Gray	C2

The Import Manager displays the column header for each element using the pattern [Identifier].[SourceTagName].

Add Repeated Instruction With a Filter

To review how each option affects the outcome, make the following updates in the Sample template, and then click the Sample reload button to update the Conversion Preview.

The repeated Manual element is handled differently since the type of identifier is an attribute.

Start by placing the "[?Source?]" instruction to the value:

```
    "Manuals": [
                {
                  "type": "ManualDE",
                  "value": "[?Source?]"
                }
              ]
```

Conversion Preview:

EAN	ProductName	ConsumerShortDes...	AvailableFrom	ManualDE	Color.Value	Battery.Value
2700524977488	AC-UZ444	Active 3D Glasses	2015-01-01 00:00:00	Manual-9	Gray	C2

To get values for all repetitions, use the "[?Instruction?]": "[?Repeated?]" instruction and also add an identifier instruction "[?SourceID?]".

```
    "Manuals": [
                {
                  "[?Instruction?]": "[?Repeated?]",
                  "type": "[?SourceID?]",
                  "value": "[?Source?]"
                }
              ]
```

Conversion Preview:

EAN	ProductName	ConsumerShort...	AvailableFrom	Color.Value	Battery.Value	ManualDE.Manual	ManualEN.Manual
2700524977488	AC-UZ444	Active 3D Glasses	2015-01-01 00:00:00	Gray	C2	Manual-9	Manual-10

To filter repeated elements so that only elements with specific identifiers are considered, add the identifier to the "[?Repeated?]" instruction. To get only 'ManualEN' data, update the template as follows.

```
    "Manuals": [
                {
```

```
                    "[?Instruction?]": "[?Repeated ManualEN?]",
                    "type": "[?SourceID?]",
                    "value": "[?Source?]"
                  }
                ]
```

Conversion Preview:

EAN	ProductName	ConsumerShortDescription	AvailableFrom	Color.Value	Battery.Value	ManualEN.Manual
2700524977488	AC-UZ444	Active 3D Glasses	2015-01-01 00:00:00	Gray	C2	Manual-10

Add MultiSource Instruction

For the repeated "SyncProtocols" element there is no identifier, so use the "[?MultiSource?]" processing instruction. When this instruction is used, the "[?Repeated?]" instruction is not required.

```
    "SyncProtocols": "[?MultiSource?]"
```

Conversion Preview:

EAN	ProductName	ConsumerShortDescription	AvailableFrom	Protocol	Color.Value	Battery.Value	ManualEN.Manual
2700524977488	AC-UZ444	Active 3D Glasses	2015-01-01 00:00:00	Acme;Sony;Sharp	Gray	C2	Manual-10

Now that all instructions have been added, the template is below.

```
{
    "Items": [
        {
        "[?Instruction?]": "[?Record?]",
        "EAN": "[?Source?]",
        "PrimarySpecs": {
          "ProductInformation": {
            "General": {
              "ProductName": "[?Source?]",
              "ConsumerShortDescription": "[?Source?]",
```

```
                "AvailableFrom": "[?Source?]"
            }
        },
            "Variants": [
              {
                "[?Instruction?]": "[?Repeated?]",
                "Name": "[?SourceID?]",
                "Value": "[?Source?]"
              }
            ],
            "Manuals": [
              {
                "[?Instruction?]": "[?Repeated?]",
                "type": "[?SourceID?]",
                "value": "[?Source?]"
              }
              ]
            },
          "SyncProtocols": "[?MultiSource?]"
        }
      ]
  }
```

Conversion Preview

Test the state of the Sample template to ensure that all data has been replaced with valid instructions.

Click the Sample reload button . The Conversion Preview area displays the results of the input document against the provided template.

Select Format

Format: Generic JSON

Converter for a generic JSON format described by a template

Sample

```
{
  "Items": [
    {
    "[?Instruction?]": "[?Record?]",
    "EAN": "[?Source?]",
    "PrimarySpecs": {
     "ProductInformation": {
      "General": {
       "ProductName": "[?Source?]",
       "ConsumerShortDescription": "[?Source?]",
       "AvailableFrom": "[?Source?]"
      }
    },
       "Variants": [
        {
         "[?Instruction?]": "[?Repeated?]",
         "Name": "[?SourceID?]",
         "Value": "[?Source?]"
        }
       ],
       "Manuals": [
        {
         "[?Instruction?]": "[?Repeated?]",
         "type": "[?SourceID?]",
         "value": "[?Source?]"
        }
        ]
       },
    "SyncProtocols": "[?MultiSource?]"
   }
  ]
}
```

Conversion Preview:

EAN	ProductName	ConsumerShortDescr...	AvailableFrom	SyncProtocols	Color.Value	Battery.Value	ManualDE.value	ManualEN.value
2700524977488	AC-UZ444	Active 3D Glasses	2015-01-01 00:00:00	Acme;Sony;Sharp	Gray	C2	Manual-9	Manual-10

Generic JSON Outbound Processing Instructions

Using Generic JSON for outbound data is defined in the following sections.

Generic JSON export supports the following object types: products, entities, classifications, and assets, and further allows exporting attribute groups, attributes, data container types, reference and link types, object types, LOVs, and units.

Template, Mapping, and Output

Selecting the Generic JSON format option displays the default template which can be used to export basic property ID and Name together with attribute ID and value pairs for product objects.

```
{
 "products": [
   {
     "__type": "__record",
     "id": "__target ID",
     "name": "__target Name",
     "values": [
       {
         "attributeID": "__target",
         "value": "__target"
       }
     ]
   }
 ]
}
```

The following instructions are the only ones available for the Generic JSON format option:

"__type": "__record"

This key value pair must be included in a template and indicates that the enclosing JSON object should be resolved / populated once for every object to be handled by the format option.

"__target <OPTIONAL IDENTIFIER>"

This instruction defines a mapping target and functions as a placeholder for a JSON string value (on the right side of the colon (:)). When provided, the optional identifier is shown in the Map Data screen.

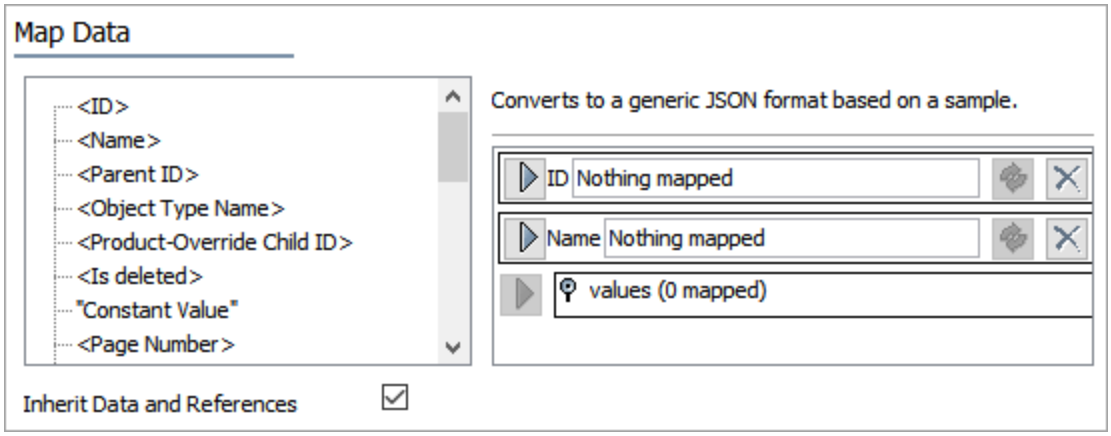
Refer to the **Generic JSON Configuration Examples** section below for how the instructions work with mapping to produce messages that include data beyond basic properties and attribute values.

Considerations

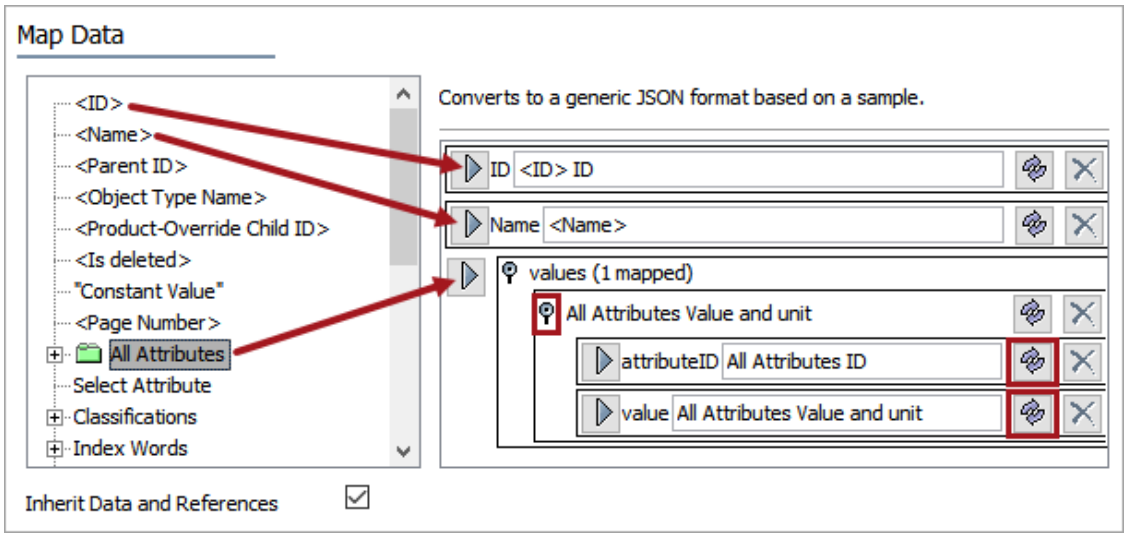
- You cannot map data as a Boolean or a number. Data output using the **"__target <OPTIONAL IDENTIFIER>"** instruction is either a string or null (when explicitly mapped but no data is available).
- When data is mapped to an array, as in the 'values' example below, an empty array is output if no data is available (for example, if there were no values for the products).
- As with the Excel or CSV format plugins, root objects for the export (i.e., the objects **"__type": "__record"** resolves against) must be of the same super type (products, classifications, assets, entities, etc.)

Using the default template, the Map Data step shows the mapping targets (right side) extracted from the template. The targets are mapped using the data sources (left side) and use standard mapping functionality, as defined in the **Outbound Map Data - Mapping Targets** topic.

Before mapping, the Map Data steps is as shown below:



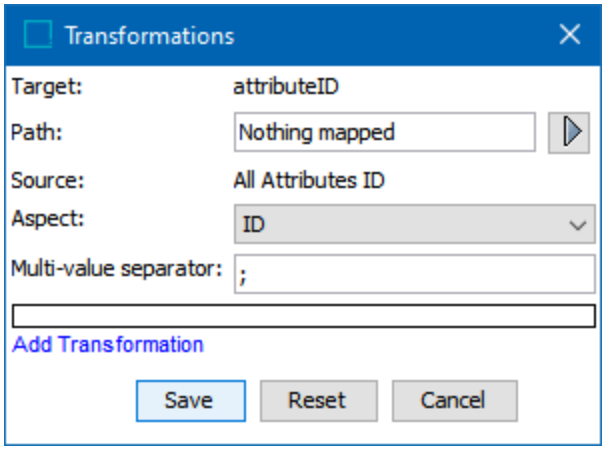
After mapping, the targets are supplied and transformed as required:



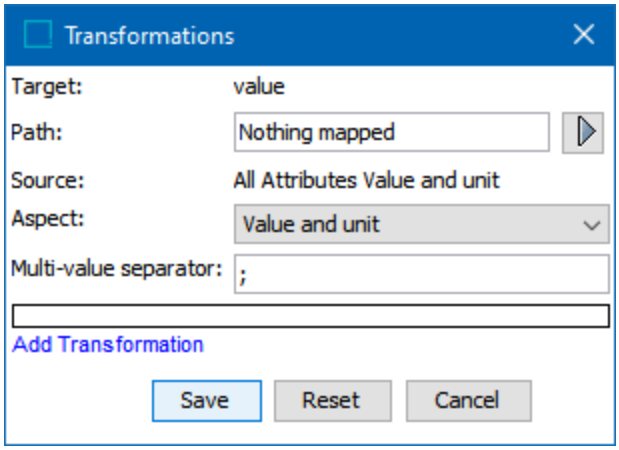
- <ID> is mapped to the 'ID' target
- <Name> is mapped to the 'Name' target
- All Attributes is mapped to the 'values' section to recursively populate all attributes that have a value for the products to be exported.

  Expand the 'values' mapped section and click the transform button ( ) to select which aspect of the mapping to use for the respective targets shown below:

  - For the 'attributeID' target, select the 'ID' aspect with default semicolon separator.



  - For the 'value' target, select the 'Value and unit' aspect with default semicolon separator.



Exporting with the 'Format JSON Document' option checked produces the following output:

```
{
  "products" : [
    {
      "id" : "179626",
      "name" : "AC-P7000-79",
      "values" : [
        {
          "attributeID" : "SecondaryColor",
          "value" : "Black"
        },
        {
          "attributeID" : "ProductHeight",
          "value" : "12 in"
        },
        {
          "attributeID" : "ManufacturerDual",
          "value" : "Acme"
        },
        {
          "attributeID" : "DisplayLanguage",
          "value" : "English;German;Spanish"
        },
        {
          "attributeID" : "ProductWeight",
          "value" : "8 lb"
        },
        {
          "attributeID" : "ProductDepth",
```

Generic JSON Configuration Examples

Attributes, data containers, and data path data sources can be used when exporting as illustrated in the examples below.

Mapping Attributes Example

This example shows how to use the 'Select Attribute' or 'All Attributes' sources to export the following data for each product using the Generic JSON format:

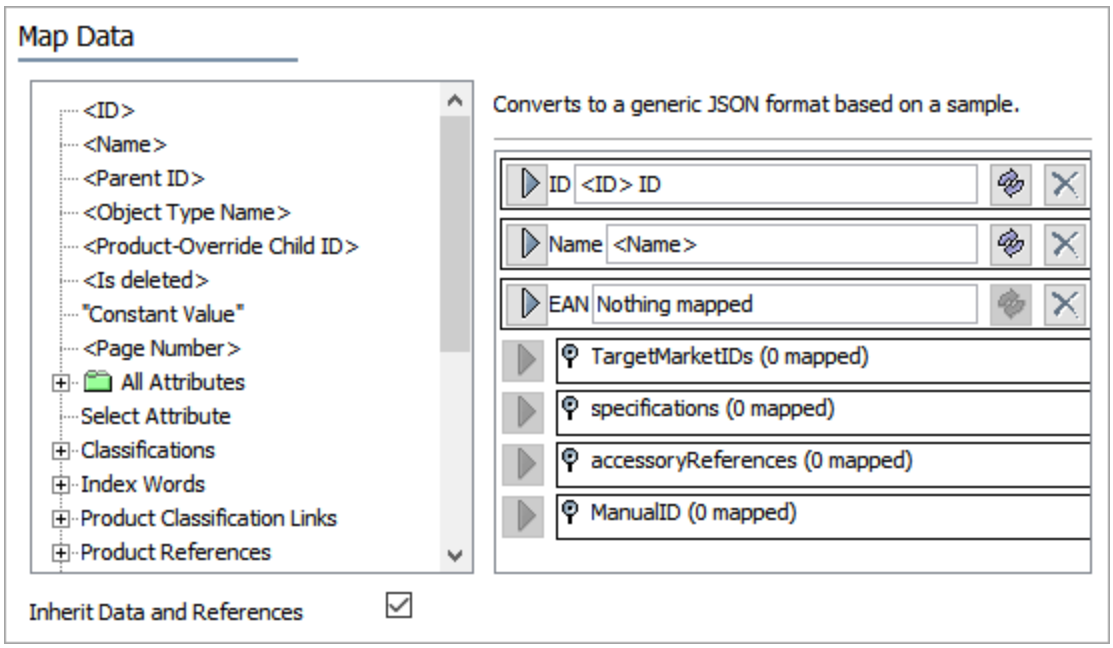
- Product ID
- Product Name
- Product EAN (single valued attribute on product)

- Product Target Market LOV value IDs as string array (multivalued LOV based attribute)
- Product attribute values for specific attribute group with values output as string array to account for multivalued attributes
- ID and EAN (attribute value) for Accessory objects (product objects referenced with a specific reference type)
- IDs of Manuals (assets referenced using a specific reference type)

The template is:

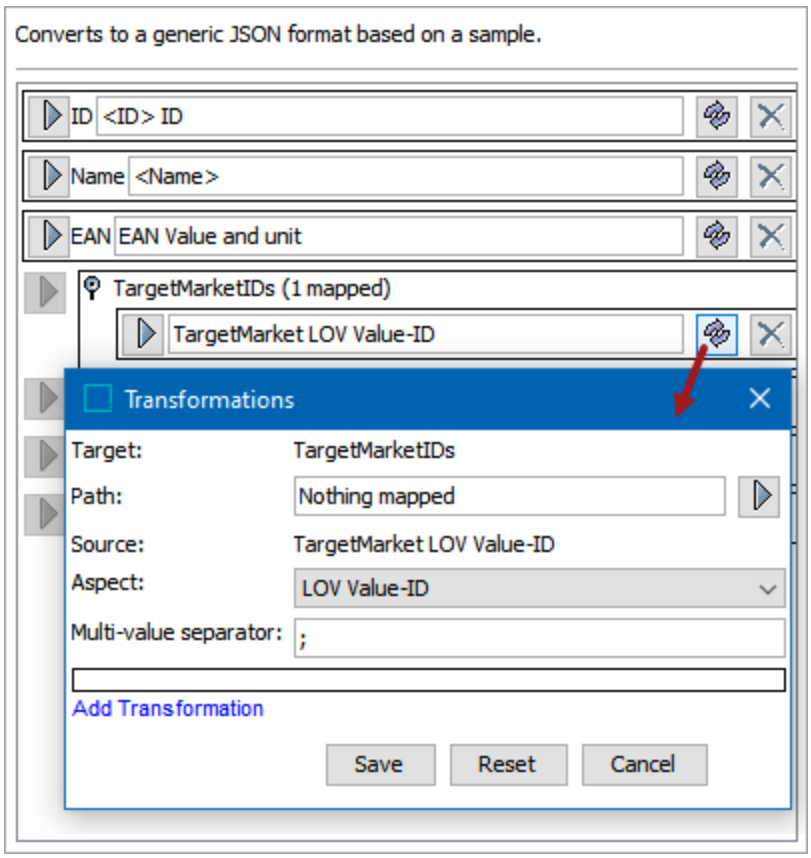
```
{
    "products": [
        {
            "__type": "__record",
            "id": "__target ID",
            "name": "__target Name",
            "ean": "__target EAN",
            "targetMarketIDs": [ "__target TargetMarketIDs" ],
            "specifications": [
                {
                    "attributeID": "__target SpecAttributeID",
                    "values": [ "__target SpecValues" ]
                }
            ],
            "accessoryReferences":[
                {
                    "targetId": "__target AccessoryID",
                    "targetEAN": "__target AccessoryEAN"
                }
            ],
            "pdfManuals": [ "__target ManualID" ]
        }
    ]
}
```

The Map Data screen prior to mapping includes the following targets:

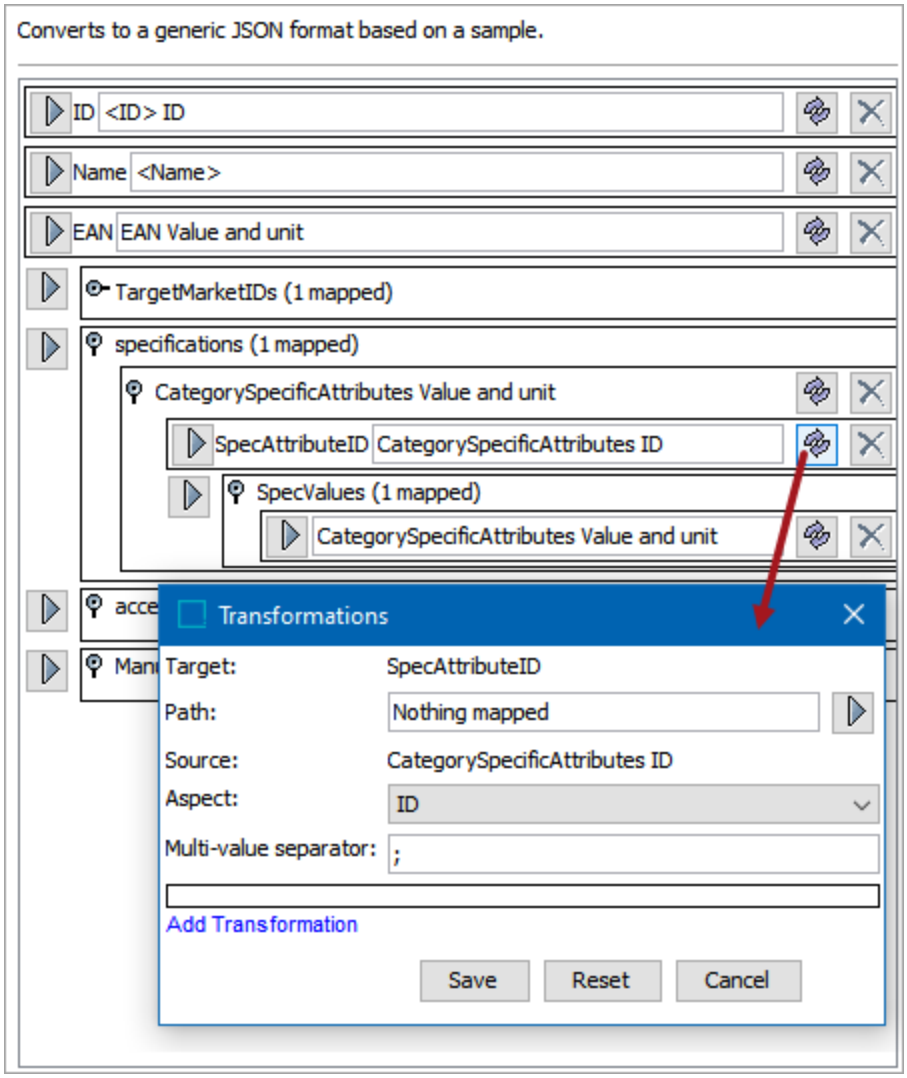


Mapping of ID and Name are supplied by the template. EAN is mapped via the Select Attribute data source.

For Target Market IDs, use the Select Attribute data source, and then select the 'LOV Value-ID' aspect to export the value IDs as shown below.

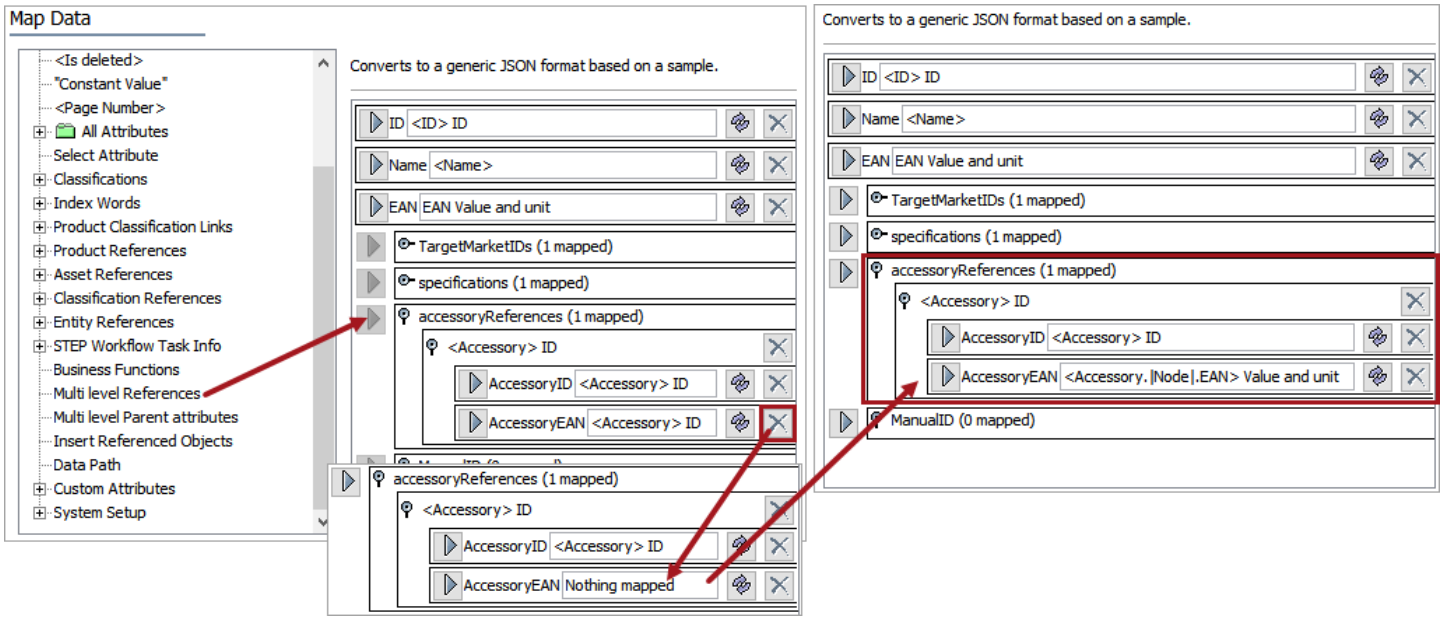


For the 'specifications' section, use All Attributes to map an attribute group (CategorySpecificAttributes) and then use the aspect parameter to output attribute IDs.

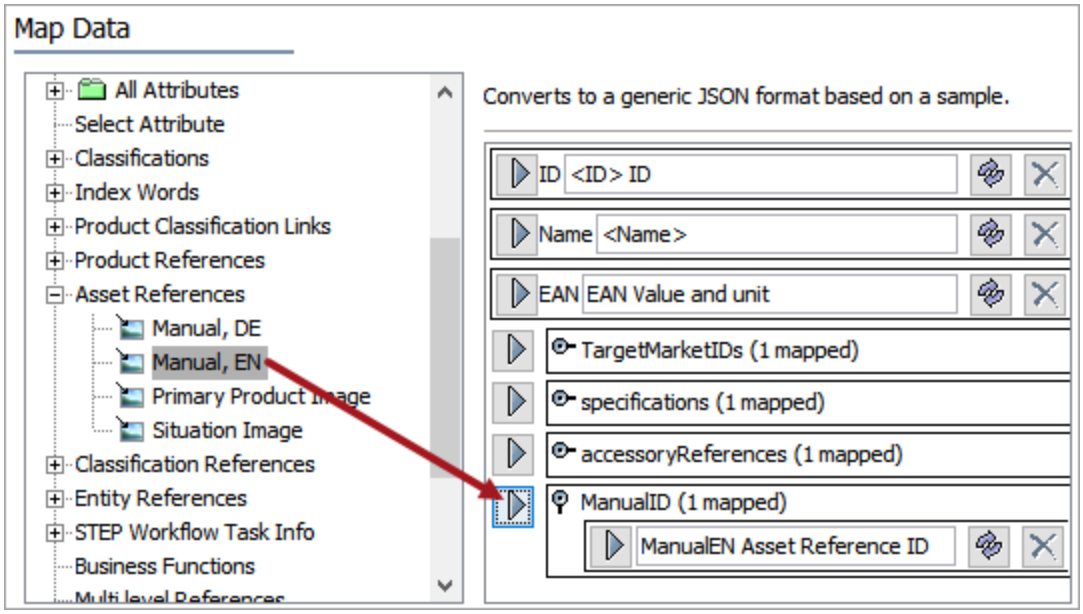


For the 'accessoryReferences' section, where both the target object ID and the EAN value from the target objects must be exported, use the 'Multi level References' option.

In this example, in the 'Multi level References' data source uses the 'Accessory' reference type and the 'Referenced node' source are selected to map the 'Accessory ID' target. The <Accessory> ID is initially mapped to both the AccessoryID and the AccessoryEAN targets. The mapping on the AccessoryEAN row is manually deleted. For the AccessoryEAN row, the 'Multi level References' data source uses the 'Accessory' reference type again, and the 'Referenced node attributes' source allows selection of the EAN attribute.



For the 'pdfManuals' string array to contain the IDs of assets referenced with a specific reference type, the type can be mapped directly from 'Asset References':

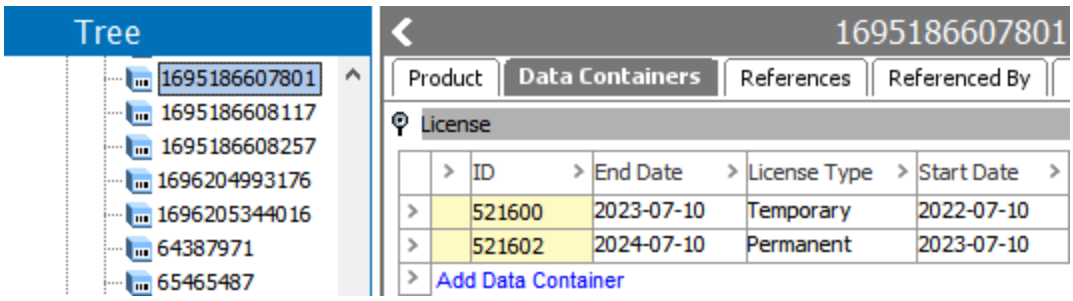


The image below shows a sample of the exported message.

```
{
  "products" : [
    {
      "id" : "I-SalesItem-13113",
      "name" : "AC-P7000-79",
      "ean" : "4905524975376",
      "targetMarketIDs" : [
        "FR",
        "DE",
        "NL"
      ],
      "specifications" : [
        {
          "attributeID" : "DisplayLanguage",
          "values" : [
            "English",
            "German",
            "Spanish"
          ]
        },
        {
          "attributeID" : "LEDBacklight",
          "values" : [
            "Yes"
          ]
        },
        {
          "attributeID" : "Series",
          "values" : [
            "P7000"
          ]
        },
        {
          "attributeID" : "AudioPowerOutput",
          "values" : [
            "12.5W+12.5W+20W+20W"
          ]
        },
        {
          "attributeID" : "SoundMode",
          "values" : [
            "Cinema",
            "Compressed Audio",
            "Game",
            "Music",
          ]
        },
        {
          "attributeID" : "DepthWithTableTopStand",
          "values" : [
            "44.2 cm"
          ]
        },
        {
          "attributeID" : "WidthWithTableTopStand",
          "values" : [
            "203.1 cm"
          ]
        }
      ],
      "accessoryReferences" : [
        {
          "targetId" : "I-SalesItem-1111",
          "targetEAN" : "4905524977623"
        },
        {
          "targetId" : "I-SalesItem-1121",
          "targetEAN" : "4905524975872"
        }
      ],
      "pdfManuals" : [
        "Manual-8"
      ]
    },
    {
      "id" : "I-SalesItem-13112",
      "name" : "AC-P7000-65",
      "ean" : "4905524975345",
      "targetMarketIDs" : [
        "NL"
      ],
      "specifications" : [
        {
          "attributeID" : "DisplayLanguage",
          "values" : [
            "English",
            "German",
            "Spanish"
          ]
        },
```


Mapping Data Container Example

This example shows exporting a data container by mapping the data container and the individual attributes from the 'All Attributes' data source. The data container being exported has three (3) attributes and two (2) instances.



On the Export Manager 'Select Format' step, the default template is displayed and these changes are made to target the data container and its attributes:

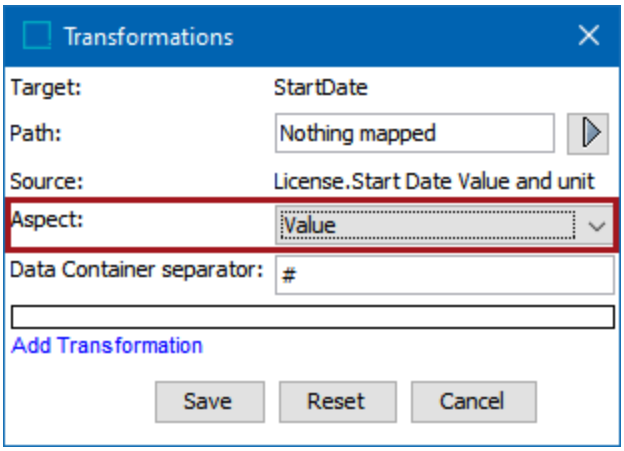
- Line 7 - update default "values" to "Licenses"
- Line 9 - update default "attributeID" to "StartDate"
- Line 10 - update default "value" to "LicenseType"; add a comma and hard return to allow for a new line
- new inserted Line 11 - "EndDate": "__target"

The updated template is:

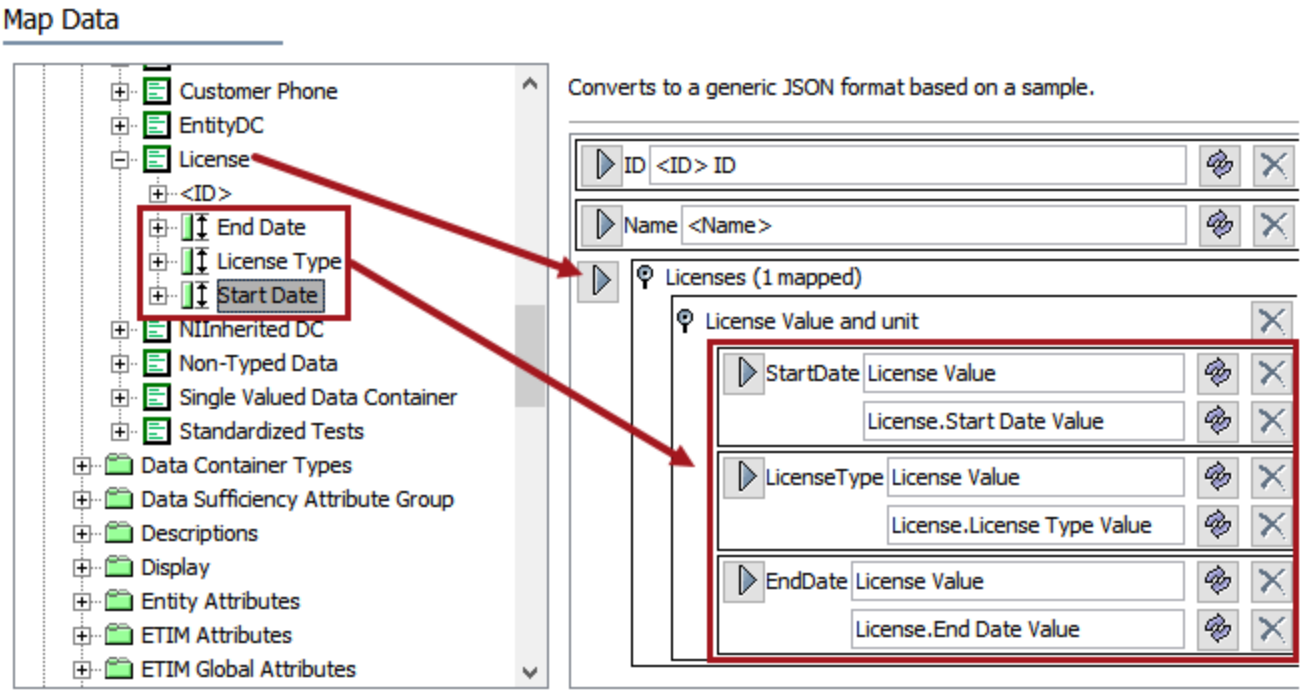
```
1   {
2     "products": [
3       {
4         "__type": "__record",
5         "id": "__target ID",
6         "name": "__target Name",
7         "Licenses": [
8           {
9             "StartDate": "__target",
10            "LicenseType": "__target",
11            "EndDate": "__target"
12          }
13        ]
14      }
15    ]
16  }
```

On the Export Manager 'Map Data' step,

- <ID> is mapped to the 'ID' target
- <Name> is mapped to the 'Name' target
- Under the All Attributes data source, the 'License' data container is mapped to the 'Licenses' target.
- Under the All Attributes data source, the 'License' data container is expanded to display the 'End Date', 'License Type', and 'Start Date' attributes. The individual attributes are each mapped to the appropriate row under the 'License Value and unit' flipper.
- For each of the data container attributes, click the Transformation button ( ) and update the Aspect to be consistent with the data being exported. In this example, Value is selected for all three columns of the data container.



After mapping and transformations, the targets are displayed as follows:



Exporting produces the following output where the data container is represented in an array:

```
{
  "products": [
    {
      "id": "11223344",
      "name": "1695186607801",
      "Licenses": [
        {
          "StartDate": "2022-07-10",
          "LicenseType": "Temporary",
          "EndDate": "2023-07-10"
        },
        {
          "StartDate": "2023-07-10",
          "LicenseType": "Permanent",
          "EndDate": "2024-07-10"
        }
      ]
    }
  ]
}
```

Mapping Data Path Example

This example shows how to use the 'Data Path' data source option to nest data for children objects inside the JSON structure for the parent.

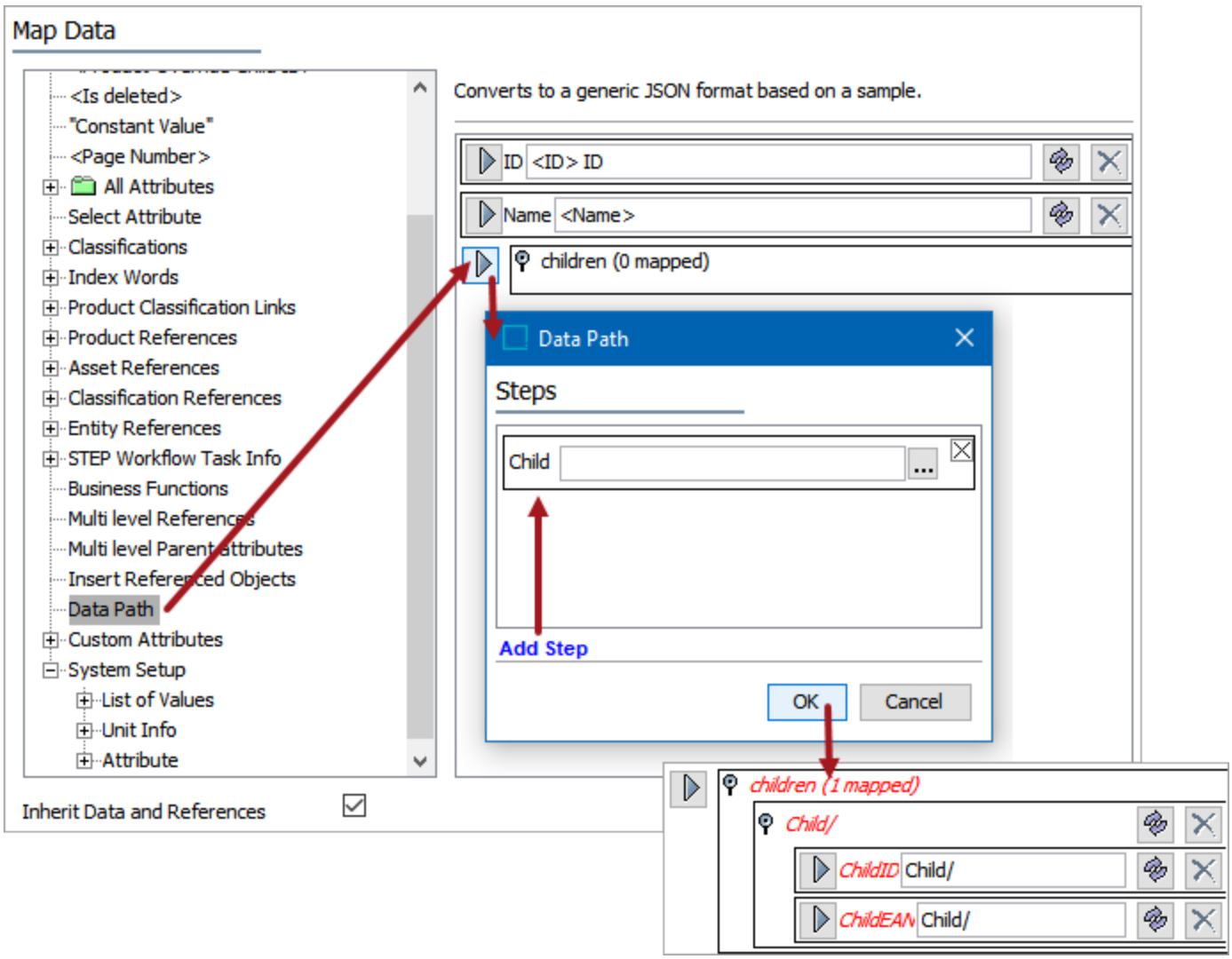
Note: This mapping is used when only the parent level is selected on the Select Objects step for the Generic JSON format. The 'Only export selected objects' checkbox must be checked on the Select Objects step.

The template is:

```
    {
        "products": [
            {
                "__type": "__record",
                "id": "__target ID",
                "name": "__target Name",
                "children":[
                    {
                        "id": "__target ChildID",
                        "ean": "__target ChildEAN"
                    }
                ]
            }
        ]
    }
```

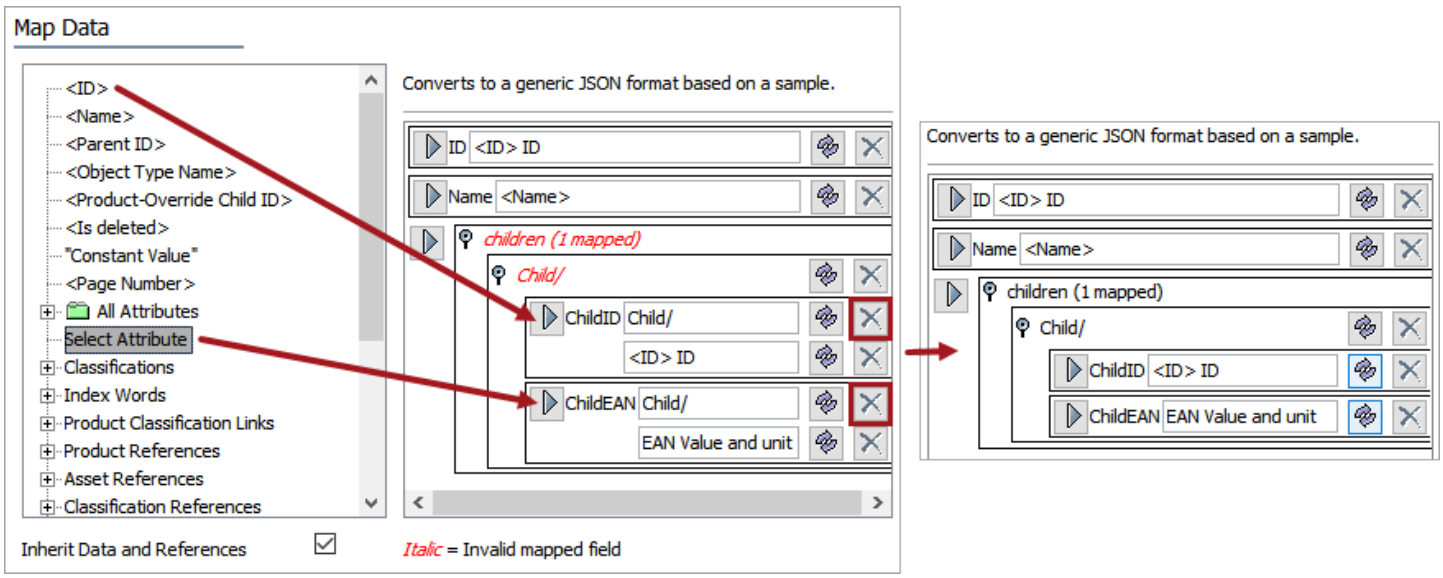
Mapping of ID and Name are supplied by the template.

For the 'children' section, use the 'Data Path' data source and add the 'Child' step.



Expand the 'children' section, map ID to the 'ChildID' placeholder, map the desired attribute to the 'ChildEAN' placeholder, and remove the extraneous Child/ mappings.

Note: The red text (Invalid mapped field) is displayed when the mapping can result in exporting child objects.



The image below shows the output message where 'I-SalesItemFamily-1311' is selected for export:

```
{
  "products" : [
    {
      "id" : "I-SalesItemFamily-1311",
      "name" : "P7000",
      "children" : [
        {
          "id" : "I-SalesItem-13113",
          "ean" : "4905524975376"
        },
        {
          "id" : "I-SalesItem-13112",
          "ean" : "4905524975345"
        },
        {
          "id" : "I-SalesItem-13111",
          "ean" : "4905524975314"
        }
      ]
    }
  ]
}
```

Generic XML Format

Generic XML is an XML-based language used to specify how data is extracted from or written to an XML document during import or export. In STEP, the Generic XML format allows you to import data from and export data to a variety of different XML formats without the need for extensions or customizations. STEP uses a template and different processing instructions for inbound and outbound Generic XML data, as defined below.

Limitations

When using the Generic XML functionality, be mindful of these limitations:

- Only products, entities, classifications, assets, attribute definitions, and LOV definitions can be imported and exported with Generic XML.
- The Generic XML import functionality cannot handle nested records.
- Neither data nor objects can be deleted when importing with Generic XML.

Format Availability

Generic XML is available for selection in:

- IIEP - refer to **Creating an Inbound Integration Endpoint**
- Import Manager - refer to **Creating a Data Import**
- Export Manager - refer to **Creating a Data Export**
- OIEP - refer to **Creating an Outbound Integration Endpoint**

Mapping

This format requires creating a data map between STEP and the data being processed, and may also include data transformations. For details, refer to **Data Mapping**.

Inbound Data

The XML file is parsed via a template that extracts data and represents it in a tabular format. This requires that you have a representative source file, namely, one where all the different elements that can occur, and that you want to handle, are represented.

The following is an example of a Generic XML template:

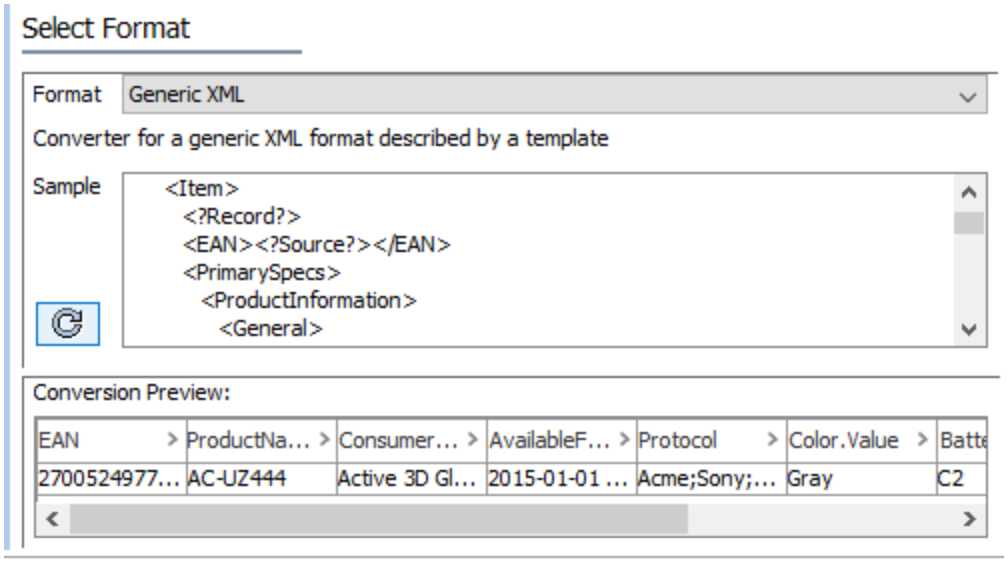
```
<Products>
   <Product>
      <Name/>
   </Product>
</Products>
```

In the example, nodes are matched with the tag <Products> on the first level of the document. Below <Products>, the template matches nodes with the tag name <Product>.

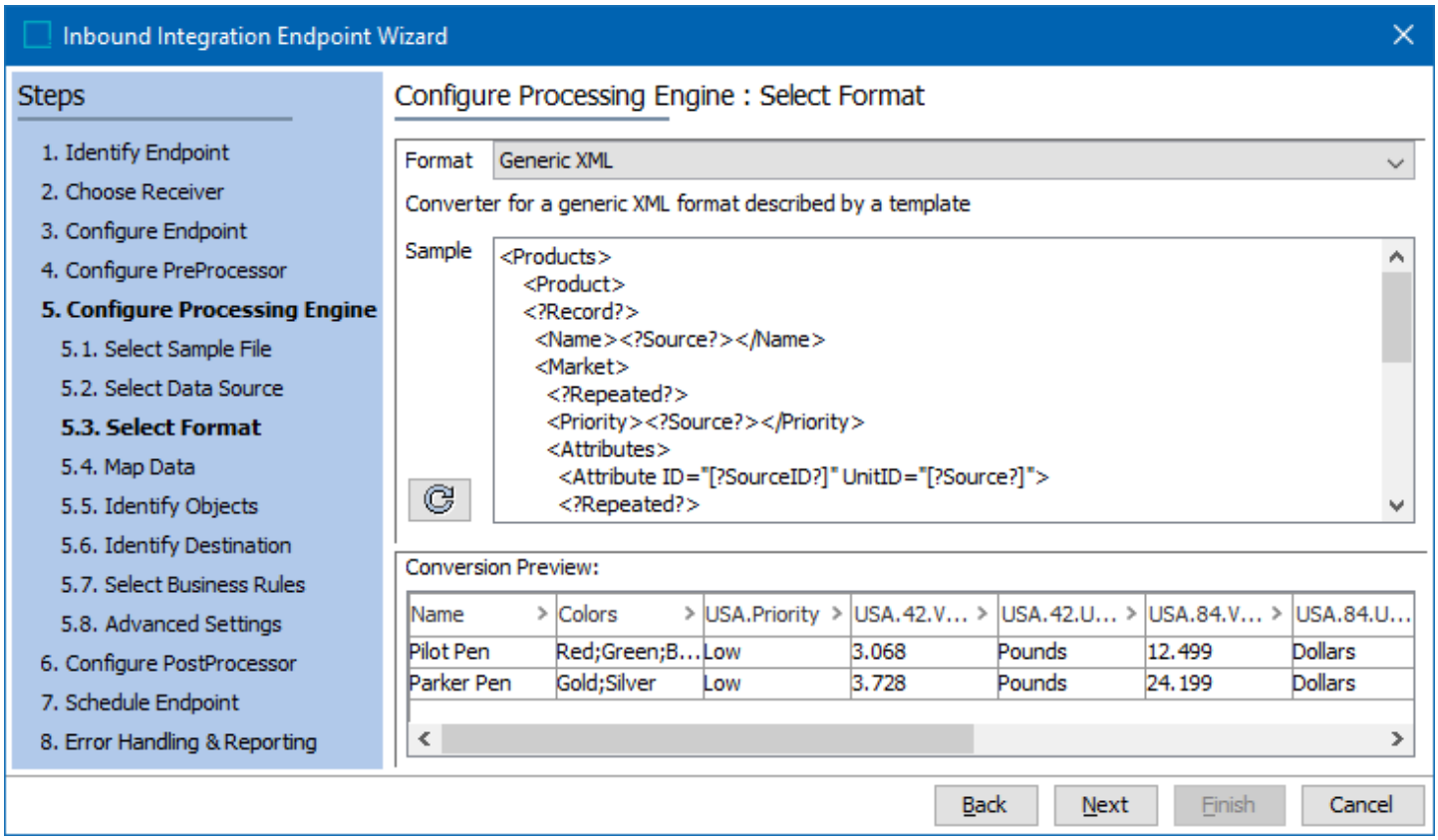
For details about the available processing instructions, refer to the **Generic XML Inbound Processing Instructions** documentation.

For details about mapping the processing instructions to STEP data, refer to **Inbound Map Data Options** documentation.

Import Manager



IIEP



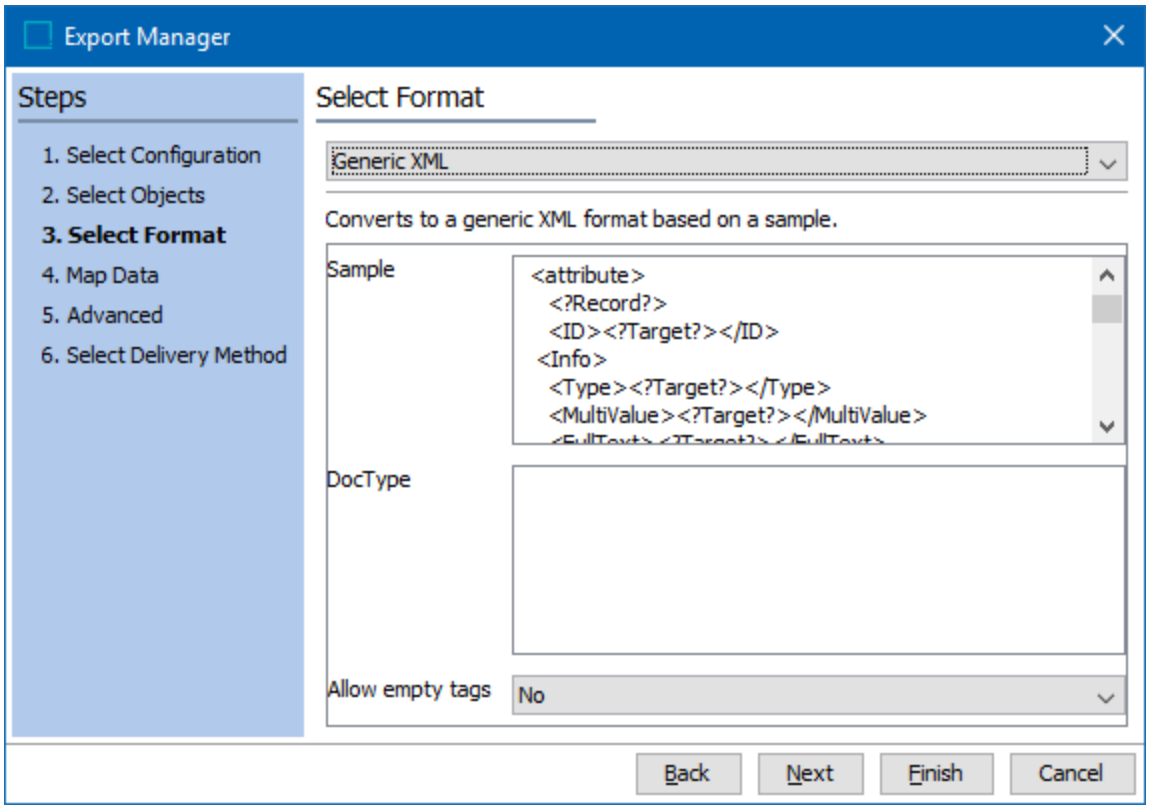
Outbound Data

For a Generic XML export, a template including instructions and placeholders determines how data should be handled, then STEP data is mapped into the XML template. For details, refer to **Outbound Map Data Options** documentation.

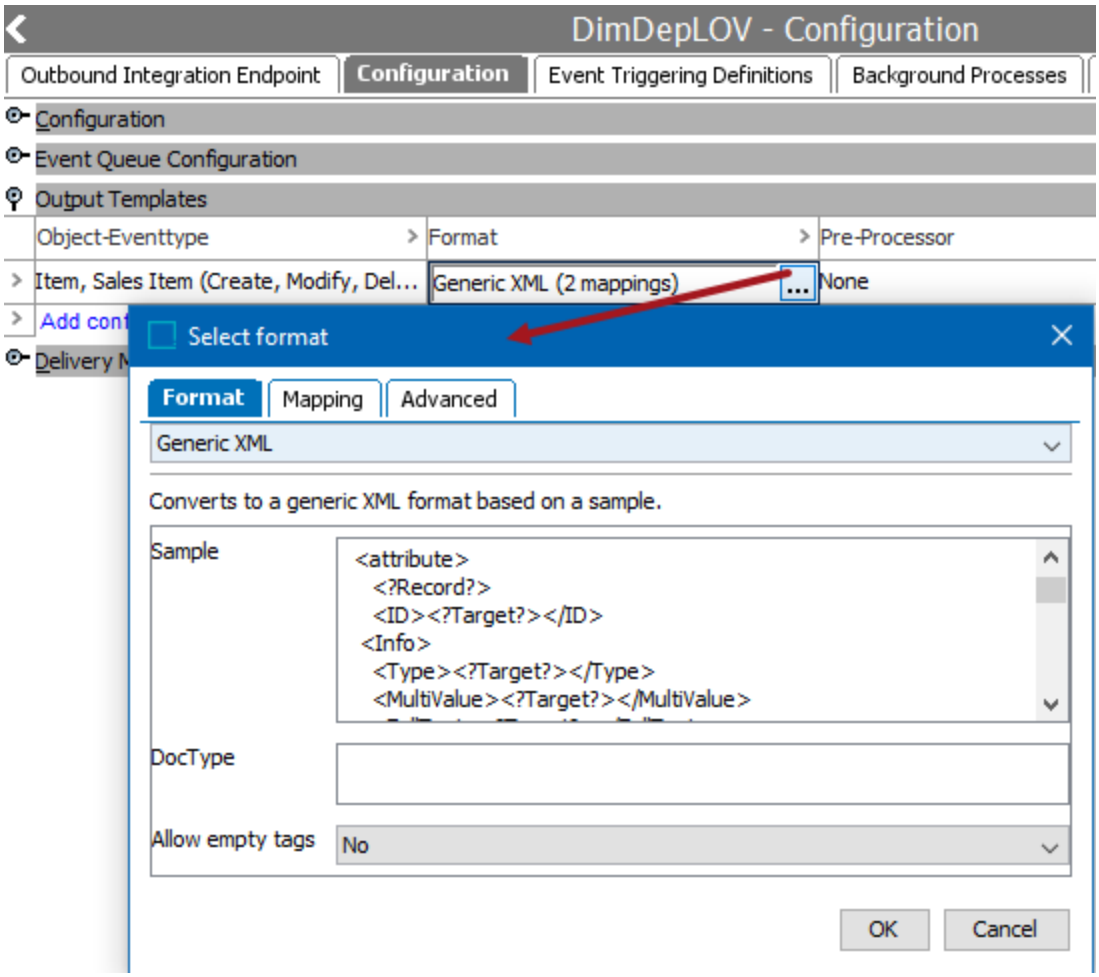
Since no undo functionality is available in the Template field, it is good practice to use an external XML editor program for creating and editing a template. To test a template's validity for output, paste the XML into STEP.

For details about the available processing instructions, some of which are only available in an OIEP, refer to the **Generic XML Outbound Processing Instructions** documentation.

Export Manager



OIEP


DimDepLOV - Configuration
Outbound Integration Endpoint
Configuration
Event Triggering Definitions
Background Processes
Configuration
Event Queue Configuration
Output Templates
Object-Eventtype
Format
Pre-Processor
Item, Sales Item (Create, Modify, Del...
Generic XML (2 mappings)
None
Add conf
Delivery M
Select format
Format
Mapping
Advanced
Generic XML
Converts to a generic XML format based on a sample.
Sample
<attribute>
<?Record?>
<ID><?Target?></ID>
<Info>
<Type><?Target?></Type>
<MultiValue><?Target?></MultiValue>
DocType
Allow empty tags
No
OK
Cancel

Generic XML Inbound Processing Instructions

Descriptions and examples of the available processing instructions used by STEP within Generic XML are covered in the following sections:

- Record Processing Instruction in Generic XML
- Source Processing Instructions in Generic XML
- MultiSource Processing Instructions in Generic XML
- SourceID Processing Instructions in Generic XML
- Repeated Processing Instructions in Generic XML
- DimensionPointID Processing Instructions

For examples of an input document, template, and the conversion preview, refer to the following sections:

- Generic XML Import Simple Example
- Generic XML Import Data Containers Example
- Generic XML Import Advanced Example

Note: Only instructions included in both the document and the generic template are processed. Instructions not defined in the template are ignored.

DimensionPointID Processing Instructions in Generic XML

Values in generic XML for the same attribute in different languages could be represented using the same ID with an additional attribute stating the language. This normally causes a name collision in generic XML. However, generic XML can import such dimension dependent data using the <?DimensionPointID?> instruction. This instruction can be used in a ?Repeated? block in combination with ?SourceID? or alone. When the ?DimensionPointID? instruction is used, the name of the field extracted will be appended with the value of the ?DimensionPointID? in square brackets, thus avoiding the name collision.

It is important to note that the use of the ?DimensionPointID? instruction, does not actually import into the specified dimension points - this has to be set up using transformations in the Map Data step. Refer to the **Inbound Map Data - Map** topic in the **Data Format** documentation. For this reason, the values appended to the column names using the ?DimensionPointID? do not have to match the Dimension Point IDs, as this can be handled in the Map Data step of the import.

The following example shows how language dependent data can be extracted using generic XML. Note that the ID of the attribute is the same value, 52.

```
<Products>
   <Product ID="EXA-5002-1004">
      <Attributes>
        <Attribute>
          <ID>52</ID>
          <Lang>Eng</Lang>
          <Value>Some description text</Value>
        </Attribute>
        <Attribute>
          <ID>52</ID>
          <Lang>De</Lang>
          <Value>Einige beschreibenden Text</Value>
        </Attribute>
      </Attributes>
   </Product>
</Products>
```

The following Generic XML template extracts the sourceID from a node value, and appends the value inside the <Lang> tags in square brackets, e.g., '52[Eng]' and '52[De]' so they can be mapped to the respective languages.

```
<Products>
   <Product ID="[?Source?]">
   <?Record?>
   <Attributes>
     <Attribute>
       <?Repeated?>
       <ID><?SourceID?></ID>
       <Lang><?DimensionPointID?></Lang>
       <Value><?Source?></Value>
      </Attribute>
   </Attributes>
   </Product>
</Products>
```

MultiSource Processing Instructions in Generic XML

To extract the value of a repeating XML element as a multi-value field that can be mapped to a multi-valued attribute / reference in STEP, place a <?MultiSource [OptionalIdentifier]?> instruction in either the node or the attribute of the <?Record?> scope:

```
<Products>
  <Product ID="[?Source?]">
    <?Record?>
    <KeyWords>
        <KeyWord><?MultiSource?></KeyWord>
    </KeyWords>
  </Product>
</Products>
```

Within the same scope, multiple nodes / attributes matching a <?MultiSource [OptionalIdentifier]?> declaration will contribute to the same multi-value field. The input document below, uses the template above.

```
<Products>
  <Product ID="42">
    <KeyWords>
      <KeyWord>Red</KeyWord>
      <KeyWord>Green</KeyWord>
      <KeyWord>Blue</KeyWord>
    </KeyWords>
  </Product>
</Products>
```

It produces the fields 'ID' and 'KeyWord' with values ID=42 and KeyWord=Red|Green|Blue. Note that container-node, 'KeyWords' in this case, is not a requirement to extract a multi-value field. A document with the 'KeyWord' nodes directly under the 'Product' node could be processed by using the following template:

```
<Products>
```

```
      <Product ID="[?Source?]">
        <?Record?>
        <KeyWord><?MultiSource KeyWords?></KeyWord>
      </Product>
    </Products>
```

Like the <?Source?> instruction, the <?MultiSource?> instruction also supports explicitly specifying a field name using an optional identifier. For more information, refer to the **Preventing Duplicate Field Names** section of the **Source Processing Instructions in Generic XML** documentation.

Record Processing Instruction in Generic XML

The template must declare which element corresponds to a single record (product, entity, etc.). This is done using the <?Record?> processing instruction, which applies to the XML node that surrounds it. A template must contain exactly one <?Record?> processing instruction.

```
<Products>
   <Product>
      <?Record?>
      <Name/>
   </Product>
</Products>
```

In this example, STEP will extract data for each product object inside the <Product> and <Products> elements, regardless of the number that exist. The product node is the record node of the template. The record node must not be declared as a repeated node. Although this is a valid template, no data is being supplied by this template.

Repeated Processing Instructions in Generic XML

Additional declarations are required when extracting data from repeated structures below the record node. Declaration of a repeated node happens with the <?Repeated [OptionalSpaceSeparatedFilters]?> processing instruction. The <?Repeated?> instruction must be placed inside <?Record?> scope. For example:

```
<Products>
   <Product ID="EXA-5002-1004">
      <Values>
         <Value AttributeID="52">Some description text</Value>
         <Value AttributeID="77">53 kg</Value>
      </Values>
    </Product>
</Products>
```

Note that the **Value** node is repeated. To extract data from both occurrences, the template must declare that a node is a repeated node, which means that a set of fields has to be generated for each node encountered. The template also has to declare an ID source on or below the repeated node.

The following template could be used to extract the data from the above Generic XML:

```
<Products>
  <Product ID="[?Source ID?]">
  <?Record?>
    <Values>
      <Value AttributeID="[?SourceID?]"><?Repeated?><?Source?></Value>
    </Values>
  </Product>
</Products>
```

The ID source is used to map each occurrence of a repeated node to a specific set of fields of the form [Identifier].[SourceTagName]. This is required to ensure that related repeated nodes map to the same fields across different records.

By default, a repeated scope produces a set of fields for each ID Source value encountered in the input document. Using the 'optional space separated filters' allows you to select values from specific ID keys or to guard against additional columns being generated as input documents evolve in the future.

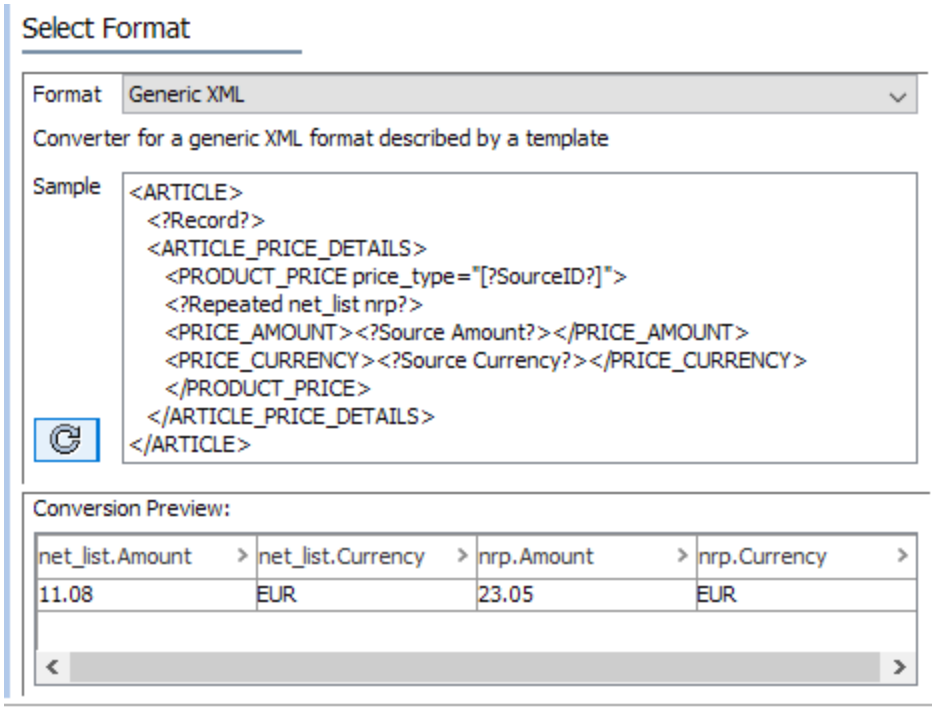
```
<ARTICLE>
   <ARTICLE_PRICE_DETAILS>
      <DATETIME type="valid_start_date">
      <DATE>2010-01-01</DATE>
      </DATETIME>
   <PRODUCT_PRICE price_type="net_list">
      <PRICE_AMOUNT>11.08</PRICE_AMOUNT>
      <PRICE_CURRENCY>EUR</PRICE_CURRENCY>
      <TAX>0.19</TAX>
      <LOWER_BOUND>1</LOWER_BOUND>
   </PRODUCT_PRICE>
   <PRODUCT_PRICE price_type="nrp">
      <PRICE_AMOUNT>23.05</PRICE_AMOUNT>
      <PRICE_CURRENCY>EUR</PRICE_CURRENCY>
      <TAX>0.19</TAX>
      <LOWER_BOUND>1</LOWER_BOUND>
   </PRODUCT_PRICE>
   <PRODUCT_PRICE price_type="employee">
      <PRICE_AMOUNT>8.52</PRICE_AMOUNT>
      <PRICE_CURRENCY>EUR</PRICE_CURRENCY>
      <TAX>0.19</TAX>
      <LOWER_BOUND>1</LOWER_BOUND>
   </PRODUCT_PRICE>
   </ARTICLE_PRICE_DETAILS>
</ARTICLE>
```

The price and currency information can be extracted for the net_list and nrp price types only by following the proceeding steps and using the following template:

1. Copy the above Generic XML to a new file, and save as an XML file
2. Open Import Manager, select the file as the Data Source Filename, and click 'Next'
3. The user can enter/ paste the template below in the Import Manager, and a conversion preview can be generated by clicking the refresh button, .

```
<ARTICLE>
   <?Record?>
   <ARTICLE_PRICE_DETAILS>
```

```
        <PRODUCT_PRICE price_type="[?SourceID?]">
        <?Repeated net_list nrp?>
        <PRICE_AMOUNT><?Source Amount?></PRICE_AMOUNT>
        <PRICE_CURRENCY><?Source Currency?></PRICE_CURRENCY>
        </PRODUCT_PRICE>
      </ARTICLE_PRICE_DETAILS>
    </ARTICLE>
```



This produces the fields:

- net_list.Amount = 11.08
- net_list.Currency = EUR
- nrp.Amount = 23.05
- nrp.Currency = EUR

The <?Repeated?> instruction can take an arbitrary number of filter terms separated by white-space characters. When no terms are defined, the filtering functionality is disabled.

Source Processing Instructions in Generic XML

To extract the content of an XML element as a field, place a <?Source [OptionalIdentifier]?> processing instruction within the <?Record?> scope of the template. By default, the heading (column) name of a field is the name of the surrounding parent node.

```
<Products>
   <Product>
     <?Record?>
     <Name><?Source?></Name>
     <Text><?Source Description?></Text>
    </Product>
</Products>
```

In the above example, each product hierarchy encountered exports a row with data for the columns Name and Description.

To make the mapping step easier, use your own identifiers by defining an optional identifier. As shown in the <Text> instruction in the above example, the Text field will display with a heading of 'Description' since the optional identifier was used within the source tag. The optional identifier displays in the Map Data step.

You can also extract the value of an attribute by using the [?Source [Optional Identifier]?] instruction as follows:

```
<Products>
   <Product Name="[?Source?]" Text="[?Source Description?]">
   <?Record?>
   </Product>
</Products>
```

Preventing Duplicate Field Names

Fields extracted using the <?Source?> and <?MultiSource?> instructions can sometimes have the same names, either because of identical tag or attribute names, or because of duplication of names explicitly chosen by the template creator. In this situation, the Import Manager ensures that all field names are unique by adding a post-fix of the form (n), where n is the first integer >= 2 for which the combined field name is not already in use.

If a repeated scope contains multiple <?Source?> or <?MultiSource?> instructions, and one of them results in a name duplication, only that field will be renamed.

Note: Resolve name collisions by providing explicit, non-duplicating names.

SourceID Processing Instructions in Generic XML

?SourceID? instructions must be placed inside repeated scope. Only one ?SourceID? is allowed for each scope.

- <?SourceID?> is used to extract the identifier for a repeated element when the identifier is a nested element.
- [?SourceID?] is used to extract the identifier for a repeated element when the identifier is an attribute.

The following is an example of an input document that requires the repeated node and ID source declarations to happen on different nodes:

```
<Products>
   <Product ID="EXA-5002-1004">
      <Attributes>
        <Attribute>
          <ID>52</ID>
          <Value>Some description text</Value>
         </Attribute>
        <Attribute>
          <ID>77</ID>
          <Value>53 kg</Value>
        </Attribute>
      </Attributes>
   </Product>
</Products>
```

The following is a Generic XML template for processing the input document:

```
<Products>
   <Product ID="[?Source?]">
   <?Record?>
      <Values>
         <Value AttributeID="[?SourceID?]">
         <?Repeated?>
         <?Source?>
         </Value>
      </Values>
    </Product>
```

```
    </Products>
```

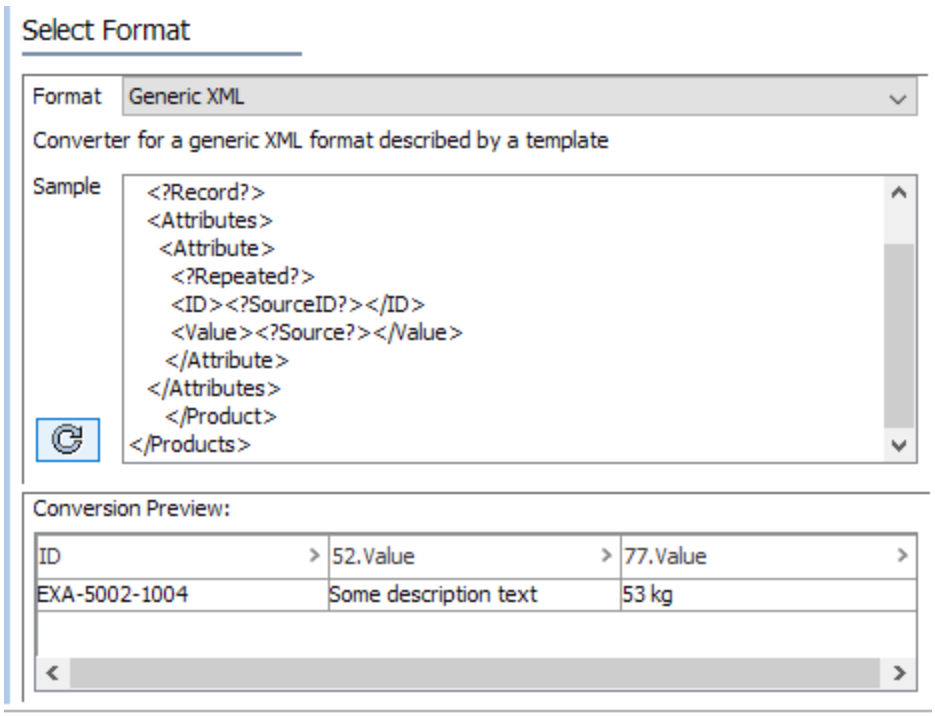
When used on the input document, this template outputs the fields 'ID', 'Value-52', 'Value-77.' In this case, the subtree from 'Value' and down is said to be in repeated scope.

The Generic XML template this time extracts the sourceID from a node value. Follow the proceeding steps and use the following template to extract the input document:

1. Copy the above Generic XML to a new file, and save as an XML file.
2. Open Import Manager, select the file as the Data Source Filename, and click 'Next.'
3. Enter or paste the template below in the Import Manager, and generate a conversion preview by clicking the refresh button, .

```
<Products>
   <Product ID="[?Source?]">
   <?Record?>
   <Attributes>
     <Attribute>
       <?Repeated?>
       <ID><?SourceID?></ID>
       <Value><?Source?></Value>
      </Attribute>
   </Attributes>
      </Product>
</Products>
```

For the input document, this template generates the fields **ID**, **52.Value** and **77.Value**.


Select Format
Format
Generic XML
Converter for a generic XML format described by a template
Sample
<?Record?>
<Attributes>
<Attribute>
<?Repeated?>
<ID><?SourceID?></ID>
<Value><?Source?></Value>
</Attribute>
</Attributes>
</Product>
</Products>
Conversion Preview:
ID
52.Value
77.Value
EXA-5002-1004
Some description text
53 kg

Generic XML Import Simple Example

The following is an example of an import document and template that uses most of the features present in a Generic XML template.

Input Document

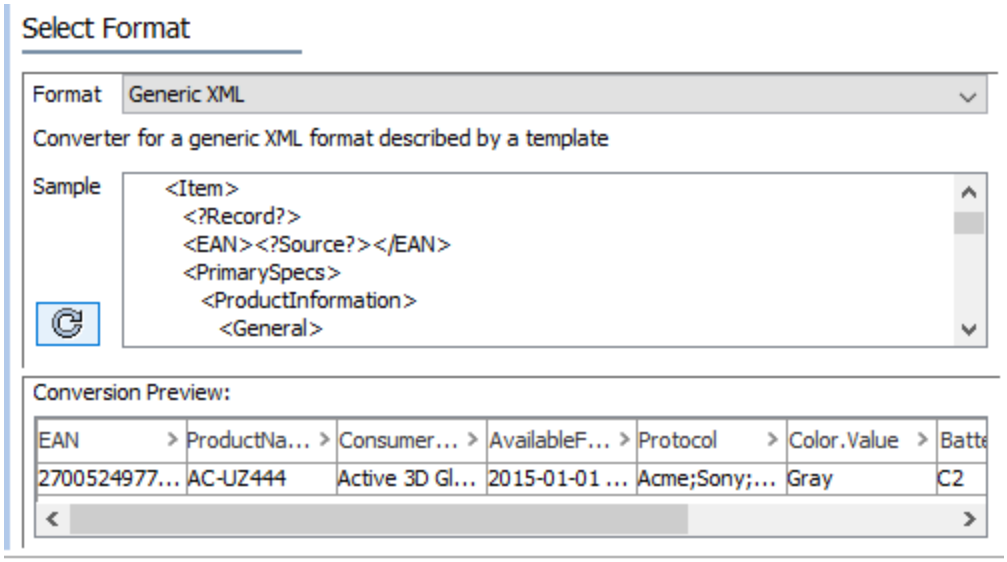
Copy the text from the online help version of this topic to a new file and save as an XML file. Open Import Manager and select the file as the Data Source Filename and click 'Next.'

Template

Copy the text in the online help version of this topic and paste it in the Import Manager Select Format step, Sample field.

Conversion Preview

Click the Sample reload button . The Conversion Preview area displays the results of the input document against the provided template. The following Map Data step presents data in the same format that is used for and CSV / Excel import. Additionally, Generic XML data must be mapped and can be transformed. To learn more about inbound transformations, refer to the **Inbound Map Data - Transform** documentation.



Generic XML Import Data Containers Example

The following is an example of an XML Import for a customer record that contains an ID, Name, Status attribute, which uses [?Source?] instructions, while the Email Addresses and Phone Numbers nodes are mapped to data containers in STEP using the [?MultiSource?] instruction in the XML template below.

Input Document

Copy the following text to a new file and save it as an XML file. Open the Import Manager and select the file as the Data Source Filename and click 'Next.'

Note that any number of additional items can be included by repeating the data within the Customers node, but in this example, there is only one customer.

```
<?xml version="1.0" encoding="UTF-8" ?>
<Customers>
 <Customer>
<ID>EXT-4500-01</ID>
<Name>John Smith</Name>
<Status>Active</Status>
<Emails>
     <Email Type="Work" Unsubscribe="false">john.smith@stibosystems.com</Email>
     <Email Type="Personal" Unsubscribe="false">john.smith@gmail.com</Email>
     <Email Type="Other" Unsubscribe="true">jsmith@yahoo.com</Email>
</Emails>
<PhoneNumbers>
     <PhoneNumber Type="Work" AreaCode="555">4567890</PhoneNumber>
     <PhoneNumber Type="Mobile" AreaCode="555">3239897</PhoneNumber>
</PhoneNumbers>
 </Customer>
</Customers>
```

Template

In this template:

- The XML declaration (the first line that indicates the XML version) was removed
- Redundant Email and PhoneNumber nodes in the Emails and PhoneNumbers nodes have been removed.
- The <?Record?> definition is added inside the first Customer node.

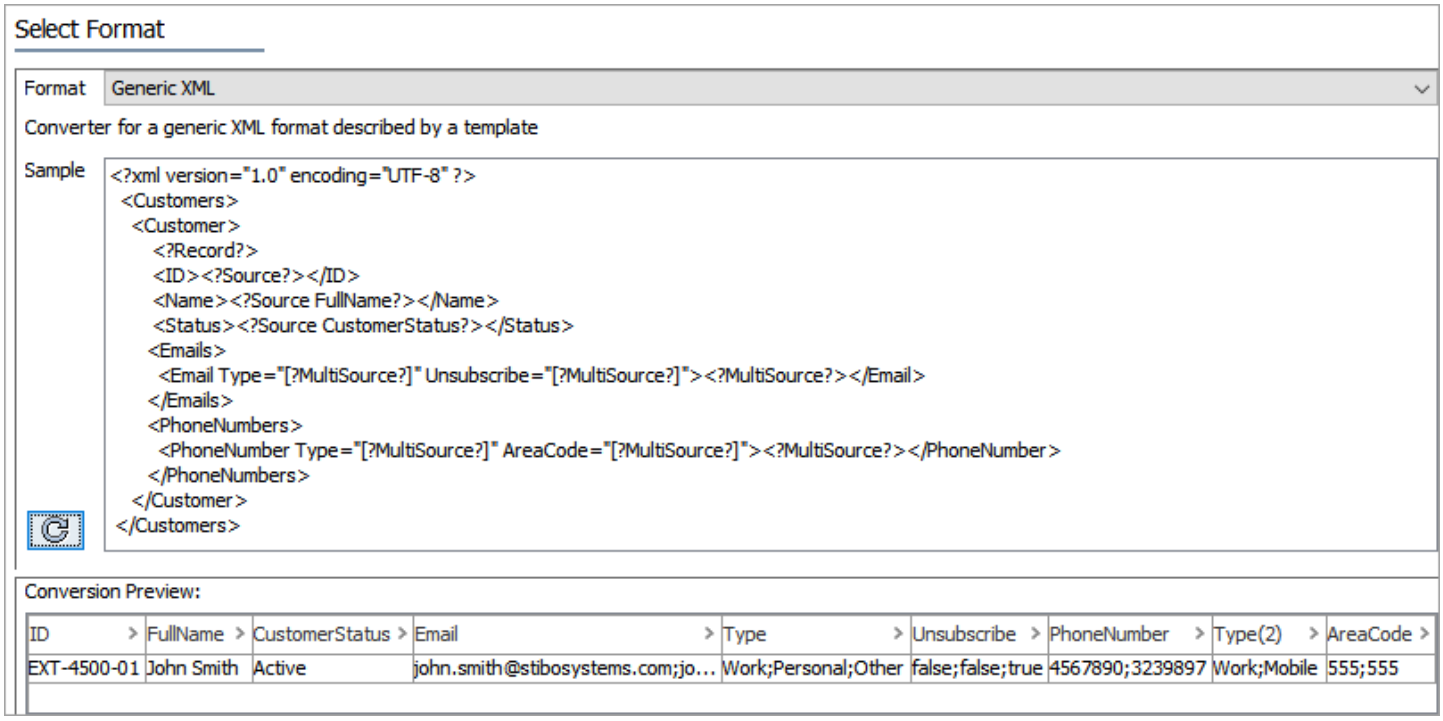
- The instruction <?Source?> replaces values for 'ID', 'Name', and 'Status', using the optional identifiers of 'FullName' and 'CustomerStatus' in 'Name' and 'Status', respectively.
- In the 'Emails' and 'PhoneNumbers' nodes, the [?MultiSource?] and <?MultiSource?> instructions are added in place of the values for each of the element and attribute values within the respective nodes.

```
<?xml version="1.0" encoding="UTF-8" ?>
<Customers>
 <Customer>
<?Record?>
<ID><?Source?></ID>
<Name><?Source?></Name>
<Status><?Source?></Status>
<Emails>
     <Email Type="[?MultiSource?]" Unsubscribe="[?MultiSource?]"><?MultiSource?></Email>
</Emails>
<PhoneNumbers>
     <PhoneNumber Type="[?MultiSource?]" AreaCode="
[?MultiSource?]"><?MultiSource?></PhoneNumber>
</PhoneNumbers>
 </Customer>
</Customers>
```

This results in the following preview within the Select Format step of the Import Manager wizard, where values from the 'Emails' and 'PhoneNumbers' nodes are concatenated with a semicolon separator. There are three email addresses and two phone numbers in the XML, each represented by the semicolon separated values in each column. While these could also be imported as references with metadata, in this case, two data containers configured for entity objects are mapped.

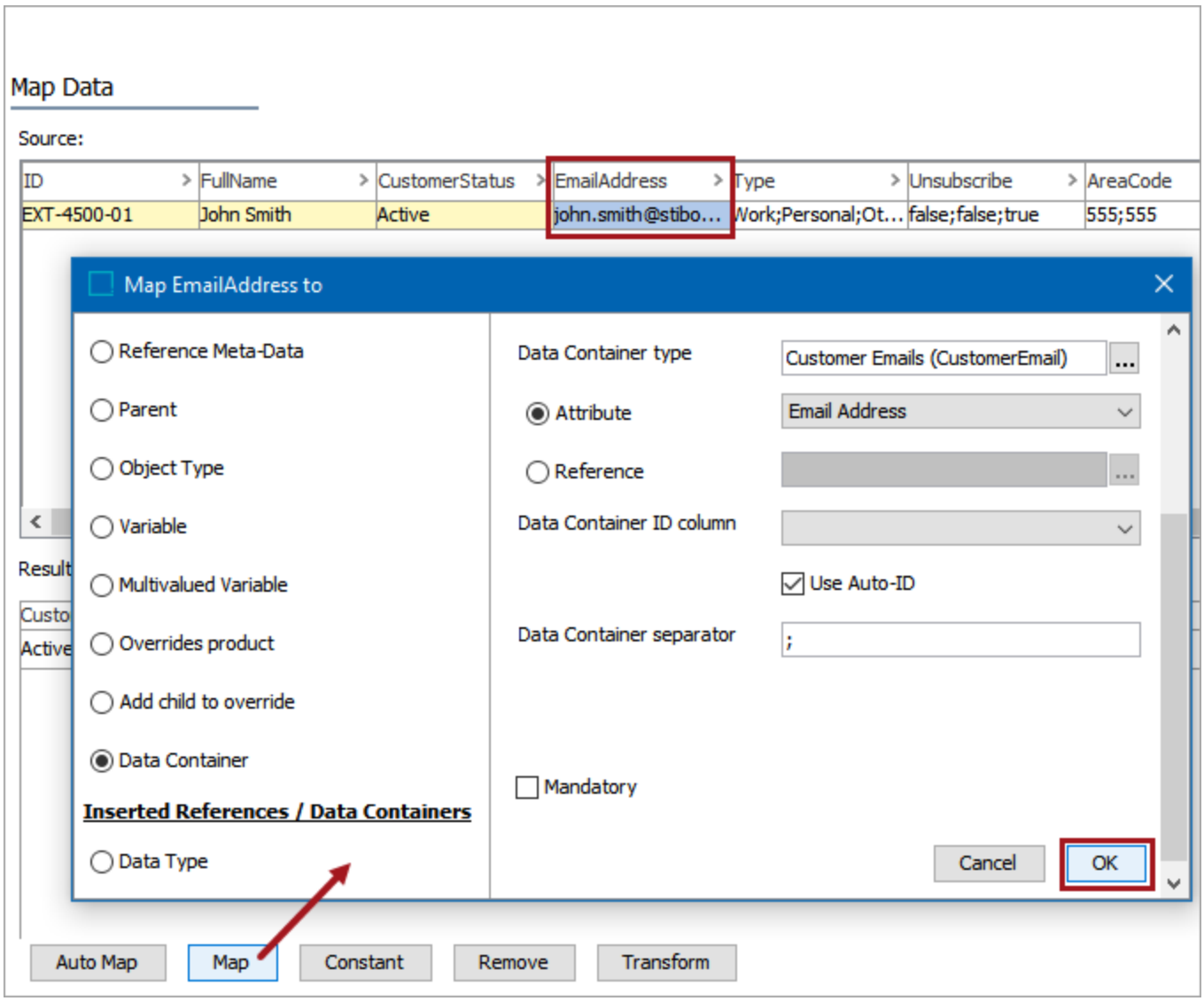
Conversion Preview

Click the Sample reload button . The Conversion Preview area displays the results of the input document against the provided template.

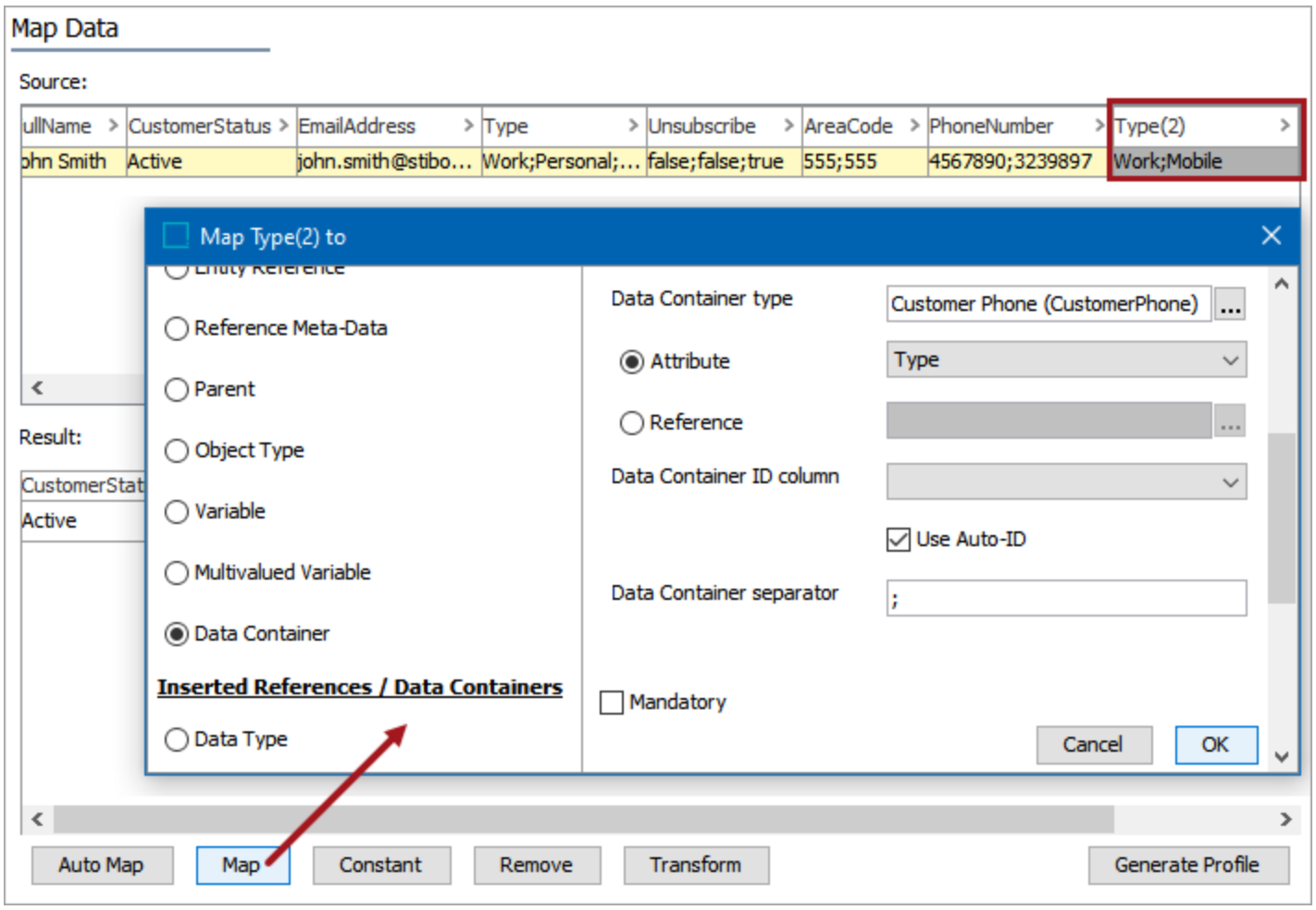


After clicking the Next button, the mapping step is presented where the ID, Name, and Status attributes have been mapped. To map the next three columns, the Data Container mapping uses the same Data Container type that is relevant for Customer Email addresses.

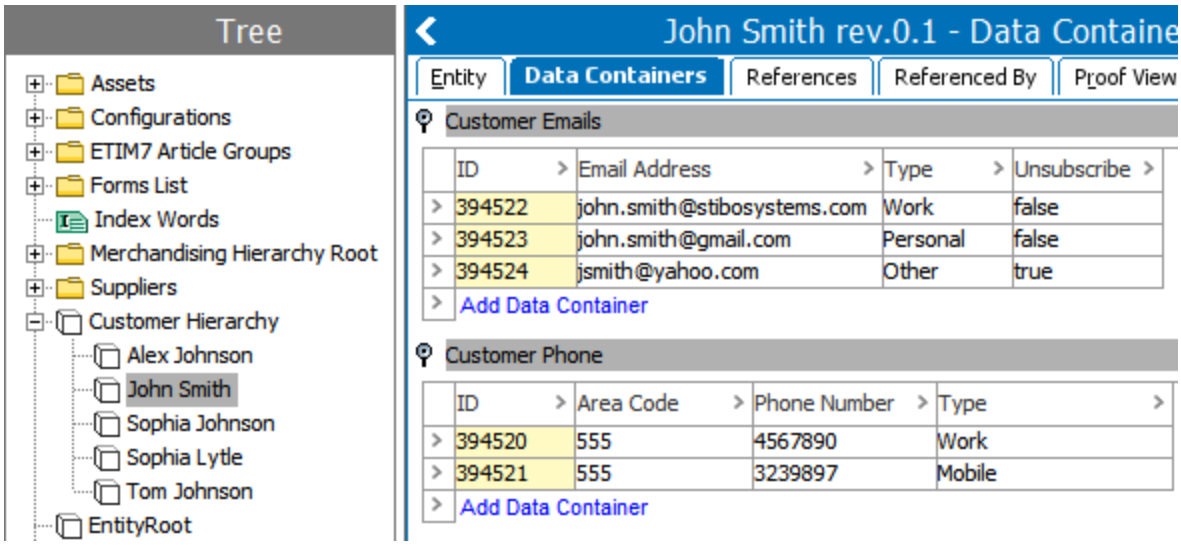
On the 'Map Email Address to' dialog, select the relevant attribute, check the 'Use Auto-ID' checkbox, and change the Data Container separator from the default of # to ;. Click OK and the values are recognized as multiple rows in the data container. The same process is used for the next two columns, with relevant attributes selected.



In the next three columns, make mappings related to the Customer Phone Numbers using a different Data Container type. When switching types, the separator resets to the default.



When the import completes successfully, the new Customer is created with data containers populated as defined in the XML document:



Generic XML Import Advanced Example

The following is an example of a more complex import document and template that uses most of the features present in a Generic XML template.

Input Document

Copy the following text to a new file and save as an XML file. Open Import Manager and select the file as the Data Source Filename and click 'Next.' Note that any number of additional items could be included by repeating the data within the Item tag.

```
<?xml version="1.0" encoding="utf-8"?>
<ItemLoad>
   <Items>
      <Item>
         <EAN>2700524977488</EAN>
         <PrimarySpecs>
            <ProductInformation>
               <General>
                  <ProductName>AC-UZ444</ProductName>
                  <ConsumerShortDescription>Active 3D Glasses</ConsumerShortDescription>
                  <AvailableFrom>2015-01-01 00:00:00</AvailableFrom>
               </General>
               <Variants>
                  <Name>Color</Name>
                  <Value>Gray</Value>
               </Variants>
               <Variants>
                  <Name>Battery</Name>
                  <Value>C2</Value>
               </Variants>
               <Manuals>
                  <Manual type="ManualDE">Manual-9</Manual>
                  <Manual type="ManualEN">Manual-10</Manual>
               </Manuals>
            </ProductInformation>
         </PrimarySpecs>
         <SyncProtocols>
            <Protocol>Acme</Protocol>
```

```
                <Protocol>Sony</Protocol>
                <Protocol>Sharp</Protocol>
            </SyncProtocols>
        </Item>
    </Items>
</ItemLoad>
```

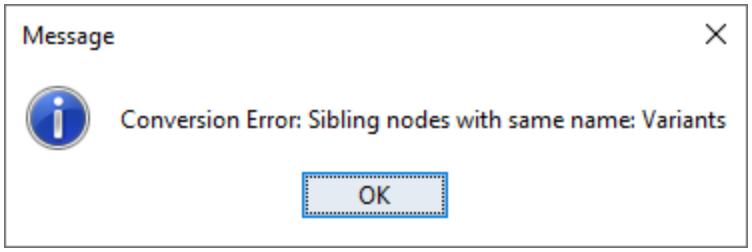
Creating a Template

To create a template from an import file, copy the file without the XML declaration (the first line that indicates the XML version), and paste into the Sample field on the Import Manager Select Format step. Then, remove any duplicate elements. Since there is only a single Item element in the example above, it can be pasted as is.

Note: When creating a template from a source file, ensure all of the elements that can occur, and that must be handled, are represented. Only instructions included in both the document and the generic template are processed. Instructions not defined in the template are ignored.

Remove Redundant Elements

Click the Sample reload button and a message is displayed about sibling nodes with the same name:



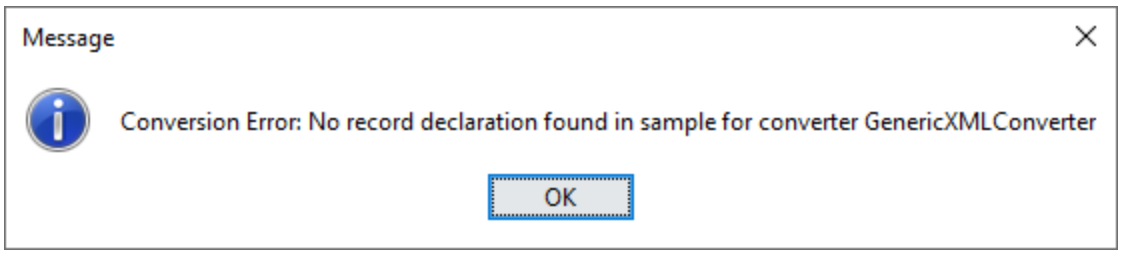
Remove of repetitions of the 'Variants', 'Manual', and 'Protocol' elements, until only one of each remains. The Sample field now displays the following:

```
<ItemLoad>
    <Items>
        <Item>
            <EAN>2700524977488</EAN>
            <PrimarySpecs>
                <ProductInformation>
```

```
                <General>
                   <ProductName>AC-UZ444</ProductName>
                   <ConsumerShortDescription>Active 3D Glasses</ConsumerShortDescription>
                   <AvailableFrom>2015-01-01 00:00:00</AvailableFrom>
                </General>
                <Variants>
                   <Name>Color</Name>
                   <Value>Gray</Value>
                </Variants>
                <Manuals>
                   <Manual type="ManualDE">Manual-9</Manual>
                </Manuals>
             </ProductInformation>
          </PrimarySpecs>
          <SyncProtocols>
             <Protocol>Acme</Protocol>
          </SyncProtocols>
       </Item>
    </Items>
 </ItemLoad>
```

Add Record and Source Instructions

Click the Sample reload button and a message is displayed about a missing record declaration:



Use the <?Record?> and <?Source?> instructions to handle all the cases where you need to extract one value between a start and end tag.

```
<ItemLoad>
```

```
      <Items>
         <Item>
           <?Record?>
            <EAN><?Source?></EAN>
            <PrimarySpecs>
               <ProductInformation>
                  <General>
                     <ProductName><?Source?></ProductName>
                     <ConsumerShortDescription><?Source?></ConsumerShortDescription>
                     <AvailableFrom><?Source?></AvailableFrom>
                  </General>
                  <Variants>
                     <Name>Color</Name>
                     <Value>Gray</Value>
                  </Variants>
                  <Manuals>
                     <Manual type="ManualDE">Manual-9</Manual>
                  </Manuals>
               </ProductInformation>
            </PrimarySpecs>
            <SyncProtocols>
               <Protocol>Acme</Protocol>
            </SyncProtocols>
         </Item>
      </Items>
   </ItemLoad>
```

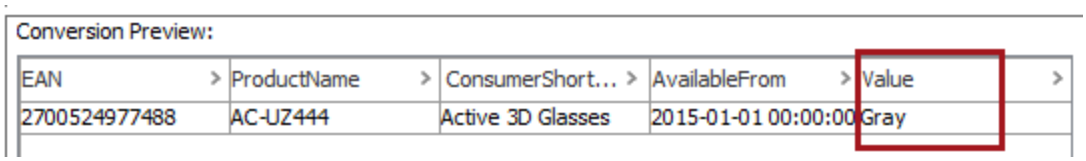
Add Repeated Instruction

To review how each option affects the outcome, make the following updates in the Sample template and then click the Sample reload button to update the Conversion Preview.

For the 'Variants' element, if we insert the <?Source?> instruction in the 'Value' element as shown below, the result will be the value for the first repetition, 'Gray.'

```
   <Variants>
     <Name>Color</Name>
```

```
3       <Value><?Source?></Value>
4     </Variants>
```

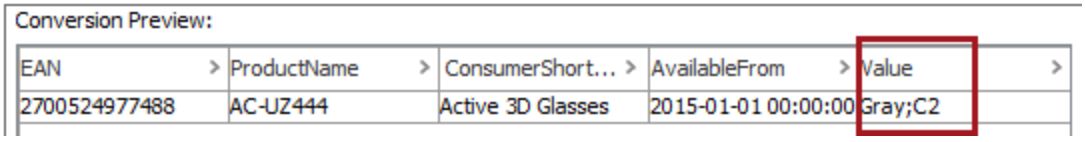

Conversion Preview:

EAN	ProductName	ConsumerShort...	AvailableFrom	Value
2700524977488	AC-UZ444	Active 3D Glasses	2015-01-01 00:00:00	Gray

To get values from both 'Variants' elements, we have two options: MultiSource and Source / Repeated / SourceID.

To get the Value from each repetition and be able to map them all to a multi valued attribute / multiple reference / links targets, we can use the <?MultiSource?> instruction.

```
1     <Variants>
2         <Name>Color</Name>
3         <Value><?MultiSource?></Value>
4     </Variants>
```


Conversion Preview:

EAN	ProductName	ConsumerShort...	AvailableFrom	Value
2700524977488	AC-UZ444	Active 3D Glasses	2015-01-01 00:00:00	Gray;C2

If the data should be mapped to different attributes, as is the case in our example, where the Name value of each repetition identifies the attribute, use these three instructions:

- <?Source?>
- one <?Repeated?> to indicate that Variants is a repeated element
- one <?SourceID?> to indicate that the Name value should be used as an identifier for each repetition

```
1     <Variants>
2         <?Repeated?>
3         <Name><?SourceID?></Name>
4         <Value><?Source?></Value>
5     </Variants>
```

Conversion Preview:

EAN	ProductName	ConsumerSh...	AvailableFrom	Color.Value	Battery.Value
2700524977488	AC-UZ444	Active 3D Glasses	2015-01-01 00...	Gray	C2

The Import Manager displays the column header for each element using the pattern [Identifier].[SourceTagName].

Add Repeated Instruction With a Filter

To review how each option affects the outcome, make the following updates in the Sample template and then click the Sample reload button to update the Conversion Preview.

The repeated Manual element is handled differently since the type of identifier is an attribute.

Again, start by placing the <?Source?> instruction to the value:

```
<Manuals>
   <Manual type="ManualDE"><?Source?></Manual>
</Manuals>
```

Conversion Preview:

EAN	ProductName	ConsumerShortDes...	AvailableFrom	ManualDE	Color.Value	Battery.Value
2700524977488	AC-UZ444	Active 3D Glasses	2015-01-01 00:00:00	Manual-9	Gray	C2

Again, to get values for all repetitions, use the <?Repeated?> instruction and also add an identifier instruction. This time, since the identifier is an attribute value, use the use the square bracket version [?SourceID?].

```
<Manuals>
    <Manual type="[?SourceID?]"><?Repeated?><?Source?></Manual>
</Manuals>
```

Conversion Preview:

EAN	ProductName	ConsumerShort...	AvailableFrom	Color.Value	Battery.Value	ManualDE.Manual	ManualEN.Manual
2700524977488	AC-UZ444	Active 3D Glasses	2015-01-01 00:00:00	Gray	C2	Manual-9	Manual-10

To filter repeated elements so that only elements with specific identifiers are considered, add the identifier to the <?Repeated?> instruction. To get only ManualEN data, update the template as follows.

```
<Manuals>
    <Manual type="[?SourceID?]"><?Repeated ManualEN?><?Source?></Manual>
</Manuals>
```

Conversion Preview:

EAN	ProductName	ConsumerShortDescription	AvailableFrom	Color.Value	Battery.Value	ManualEN.Manual
2700524977488	AC-UZ444	Active 3D Glasses	2015-01-01 00:00:00	Gray	C2	Manual-10

Add MultiSource Instruction

For the repeated Protocol element there is no identifier, so it makes sense to use the <?MultiSource?> processing instruction. When this instruction is used, the <?Repeated?> instruction is not required.

```
<SyncProtocols>
    <Protocol><?MultiSource?></Protocol>
</SyncProtocols>
```

Conversion Preview:

EAN	ProductName	ConsumerShortDescription	AvailableFrom	Protocol	Color.Value	Battery.Value	ManualEN.Manual
2700524977488	AC-UZ444	Active 3D Glasses	2015-01-01 00:00:00	Acme;Sony;Sharp	Gray	C2	Manual-10

Now that all instructions have been added, the template is below.

```
<ItemLoad>
   <Items>
      <Item>
         <?Record?>
         <EAN><?Source?></EAN>
         <PrimarySpecs>
            <ProductInformation>
               <General>
                  <ProductName><?Source?></ProductName>
                  <ConsumerShortDescription><?Source?></ConsumerShortDescription>
                  <AvailableFrom><?Source?></AvailableFrom>
               </General>
```

```
                    <Variants>
                        <?Repeated?>
                        <Name><?SourceID?></Name>
                        <Value><?Source?></Value>
                    </Variants>
                    <Manuals>
                        <Manual type="[?SourceID?]"><?Repeated?><?Source?></Manual>
                    </Manuals>
                </ProductInformation>
            </PrimarySpecs>
            <SyncProtocols>
                <Protocol><?MultiSource?></Protocol>
            </SyncProtocols>
        </Item>
    </Items>
</ItemLoad>
```

Conversion Preview

Click the Sample reload button . The Conversion Preview area displays the results of the input document against the provided template.

Select Format

Format: Generic XML

Converter for a generic XML format described by a template

Sample

```
<ItemLoad>
<Items>
<Item>
<?Record?>
<EAN><?Source?></EAN>
<PrimarySpecs>
<ProductInformation>
<General>
<ProductName><?Source?></ProductName>
<ConsumerShortDescription><?Source?></ConsumerShortDescription>
<AvailableFrom><?Source?></AvailableFrom>
</General>
<Variants>
<?Repeated?>
<Name><?SourceID?></Name>
<Value><?Source?></Value>
</Variants>
<Manuals>
<Manual type="[?SourceID?]"><?Repeated?><?Source?></Manual>
</Manuals>
</ProductInformation>
</PrimarySpecs>
<SyncProtocols>
<Protocol><?MultiSource?></Protocol>
</SyncProtocols>
</Item>
</Items>
</ItemLoad>
```

Conversion Preview:

EAN	ProductName	ConsumerShortDescription	AvailableFrom	Protocol	Color.Value	Battery.Value	ManualDE.Manual	ManualEN.Manual
2700524977488	AC-UZ444	Active 3D Glasses	2015-01-01 00:00:00	Acme;Sony;Sharp	Gray	C2	Manual-9	Manual-10

Generic XML Outbound Processing Instructions

Descriptions and examples of the available processing instructions used by STEP within Generic XML are covered in the following sections:

- Allow Empty Tags Parameter in Generic XML
- Attributes Processing Instructions in Generic XML
- Attribute Links and LOVs Processing Instructions in Generic XML
- Attribute Links and Units Processing Instructions in Generic XML
- DoNotEmit Processing Instructions in Generic XML
- ExportDate Processing Instructions in Generic XML
- List of Values Object Type in Generic XML
- Multi-Valued Objects in Generic XML
- Multi-Cardinality Tags in Generic XML
- Nestable Data Paths in Generic XML
- RemoveIfEmpty Processing Instructions in Generic XML
- Reporting Deleted Assets with a Generic XML OIEP
- Reporting Deleted Classifications with a Generic XML OIEP
- Reporting Deleted Entities with a Generic XML OIEP
- Reporting Deleted Product References with a Generic XML OIEP
- Single-Cardinality Tags in Generic XML
- Units with a Generic XML OIEP

For an example of selected objects, template, mapping, and the result, refer to the **Generic XML Sample Export** section.

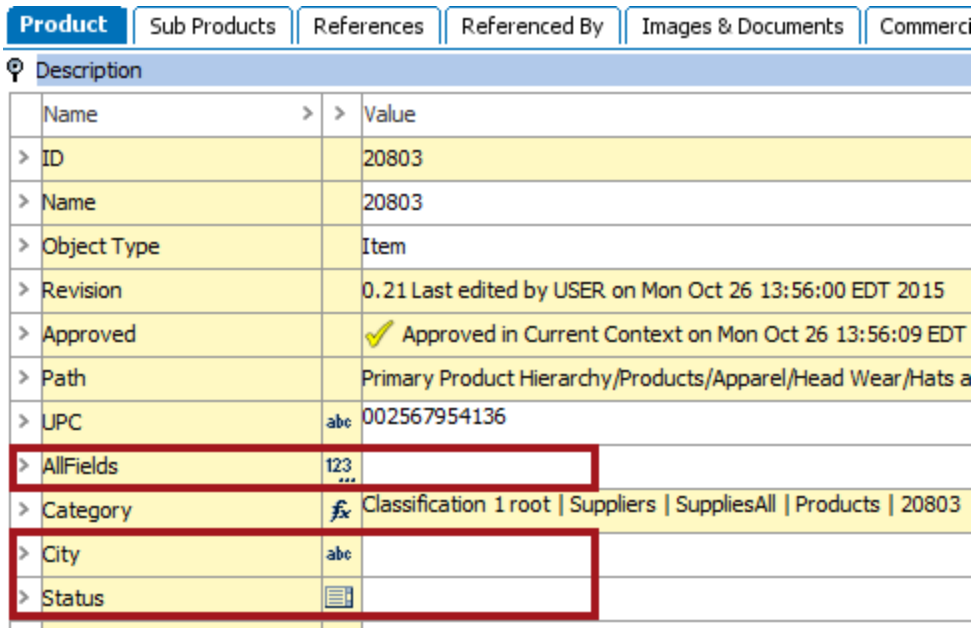
Allow Empty Tags Parameter in Generic XML

The 'Allow empty tags' parameter determines if a mapped attribute tag is exported when no value is present. For example, consider an object with an attribute value that is then removed. Using this parameter in an event-based OIEP, the value deletion would cause an update event, and the output would include this attribute with an empty value.

If tags for parent attributes with children that do not contain values should be removed completely, refer to the **RemoveIfEmpty Processing Instructions in Generic XML** section.

Objects

The object selected for export includes attributes without values.



Template

Choose **Generic XML** format and provide a template in the Sample field.

```
    <Products>
       <Product>
          <?Record?>
          <ProductID><?Target?></ProductID>
          <ProductUPC><?MultiTarget?></ProductUPC>
```

```
            <ProductAllFields><?MultiTarget?></ProductAllFields>
            <ProductCategory><?MultiTarget?></ProductCategory>
            <ProductCity><?MultiTarget?></ProductCity>
            <ProductStatus><?MultiTarget?></ProductStatus>
        </Product>
    </Products>
```

An empty tag is any that would be output as <Tag></Tag>. Non-empty tags are all other combinations, for example, <Tag>value</Tag> or <Tag attribute="attr"></Tag>.

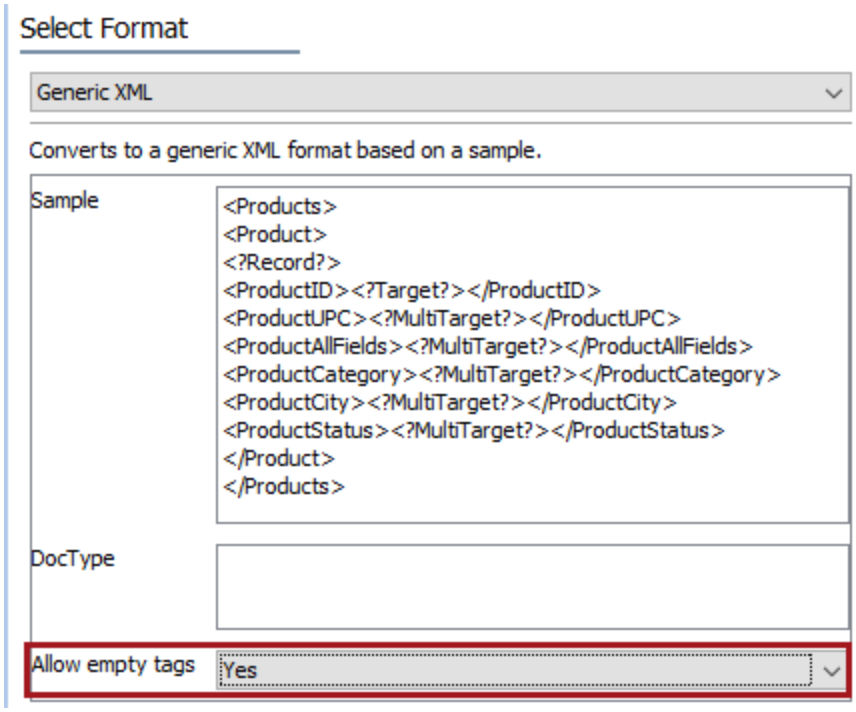
When no data is found for a tag, the **Allow empty tags** parameter works differently for <?Target?> and <?MultiTarget?> tags, as defined below.

<?Target?> Tag

- When **Allow empty tags** is set to **No**, and no data is found for the tag, but an attribute link is specified, the tag is exported. In this sample tag: <ProductCity **lang="en"**><?Target?></ProductCity>, the attribute link portion is the 'lang="en"' text. This data is exported since the values can be exported by the specified attribute link instead of the tag value.
- When **Allow empty tags** is set to **No**, and no data is found for the tag, and no attribute link is specified, the tag is not exported. In this sample tag: <ProductCity><?Target?></ProductCity>, no attribute link is identified. This tag would be eliminated from the output.

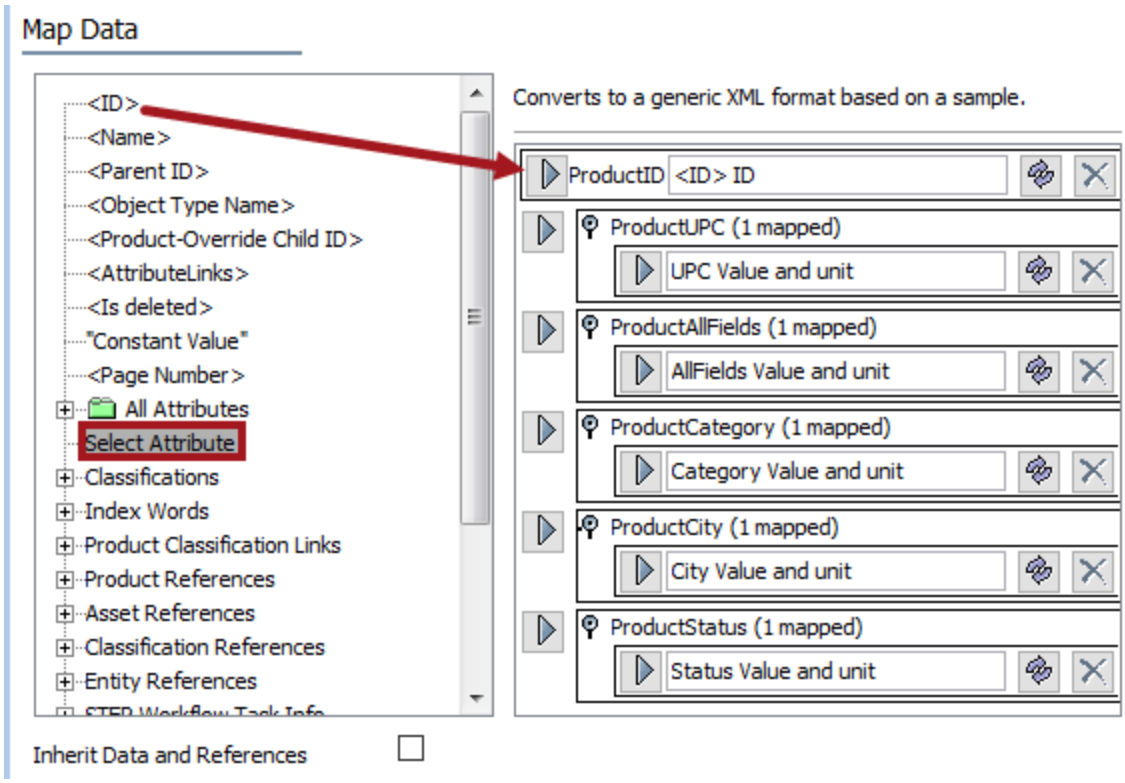
<?MultiTarget?> Tag

- When no data is found for the tag, regardless of the **Allow empty tags** parameter setting (**Yes** or **No**), and regardless of the presence of an attribute, the tag is always removed.



Mapping

Use the **Select Attribute** aspect for the attribute data targets after ID.



Results

With 'Allow empty tags' parameter = Yes, tags are exported for attributes with empty values.

```
<Products>
  <Product>
    <ProductID>20803</ProductID>
    <ProductUPC>002567954136</ProductUPC>
    <ProductAllFields></ProductAllFields>
    <ProductCategory>Classification 1 root | Suppliers | SuppliesAll | Products | 20803</ProductCategory>
    <ProductCity></ProductCity>
    <ProductStatus></ProductStatus>
  </Product>
</Products>
```

With 'Allow empty tags' parameter = No, only tags with values are exported.

```
<Products>
  <Product>
    <ProductID>20803</ProductID>
    <ProductUPC>002567954136</ProductUPC>
    <ProductCategory>Classification 1 root | Suppliers | SuppliesAll | Products | 20803</ProductCategory>
  </Product>
</Products>
```

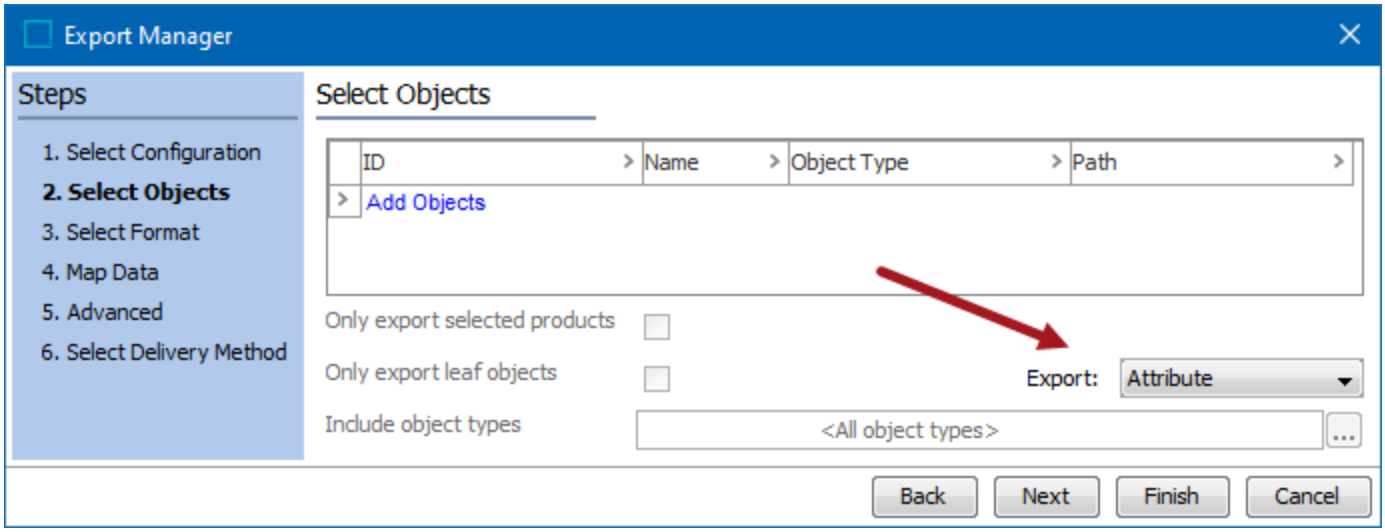
Attributes Processing Instructions in Generic XML

The Attribute template and mapping make it possible to specify which attribute setup information is included when an attribute or attribute group is exported. This type of mapping and export is used when the intent is to export data for attributes themselves (e.g., ID, Name, Type, Validation, etc.), not attribute values.

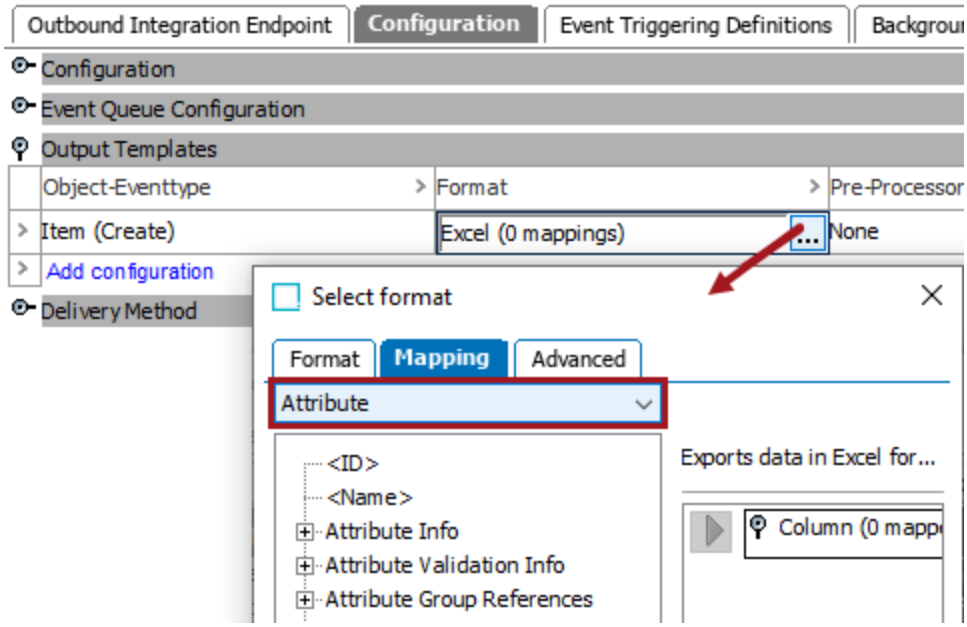
Follow the steps below to extract all available attribute data. If some data points on attributes are not required, they can be removed from the template and the mapping.

Objects

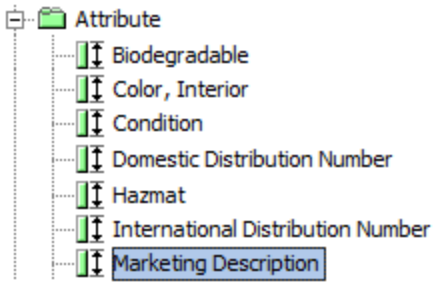
Choose 'Attribute' for the export in Export Manager:



or in an OIEP:



Use the **Add Objects** link to select a single attribute, multiple attributes, or an attribute group to export.



Note: If exporting a group, data for each field in the template is evaluated for each child in the group.

Template

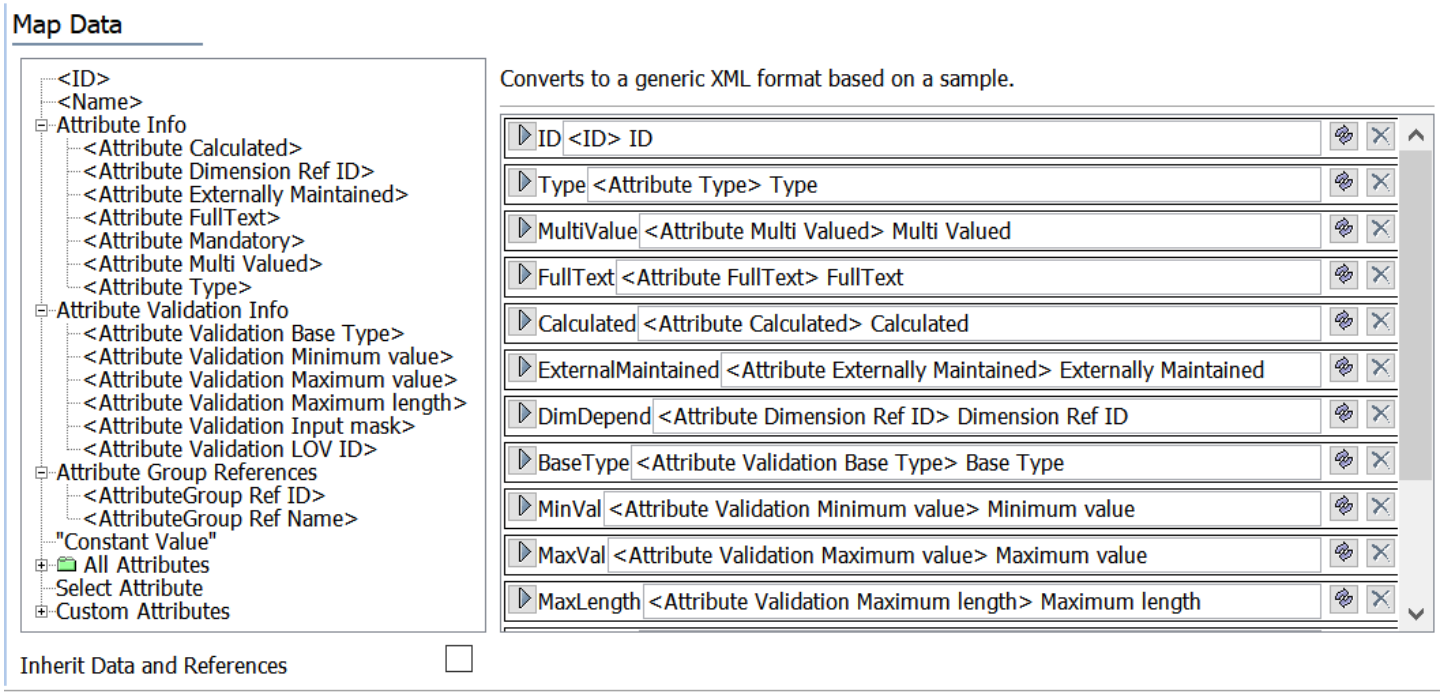
Choose the **Generic XML** format option. The following is an example of what could be provided in the Sample field. This template corresponds to the mapping options below. However, any element can be removed and/or renamed as needed to meet the export requirements.

```
<root>
   <attribute>
      <?Record?>
      <ID><?Target?></ID>
```

```
        <Info>
           <Type><?Target?></Type>
           <MultiValue><?Target?></MultiValue>
           <FullText><?Target?></FullText>
           <Calculated><?Target?></Calculated>
           <ExternalMaintained><?Target?></ExternalMaintained>
           <DimDepend><?Target?></DimDepend>
        </Info>
        <ValidationInfo>
           <BaseType><?Target?></BaseType>
           <MinVal><?Target?></MinVal>
           <MaxVal><?Target?></MaxVal>
           <MaxLength><?Target?></MaxLength>
           <InputMask><?Target?></InputMask>
           <LOVID><?Target?></LOVID>
        </ValidationInfo>
        <AttrGroupReference>
           <RefID><?Target?></RefID>
           <RefName><?Target?></RefName>
        </AttrGroupReference>
        </attribute>
     </root>
```

Mapping

The mapping screen auto-populates based on the provided template, but will display each field in the right panel as 'Nothing mapped.' Select the appropriate aspect in the left panel and use the arrow buttons to map it to a template field on the right. If all available fields are mapped, the end result will look like this:



Note: The Inherit Data and References checkbox option does not affect the Attributes instruction since it handles attribute definitions.

Refer to the **Attribute Aspects in Generic XML** documentation for a description of each available data point.

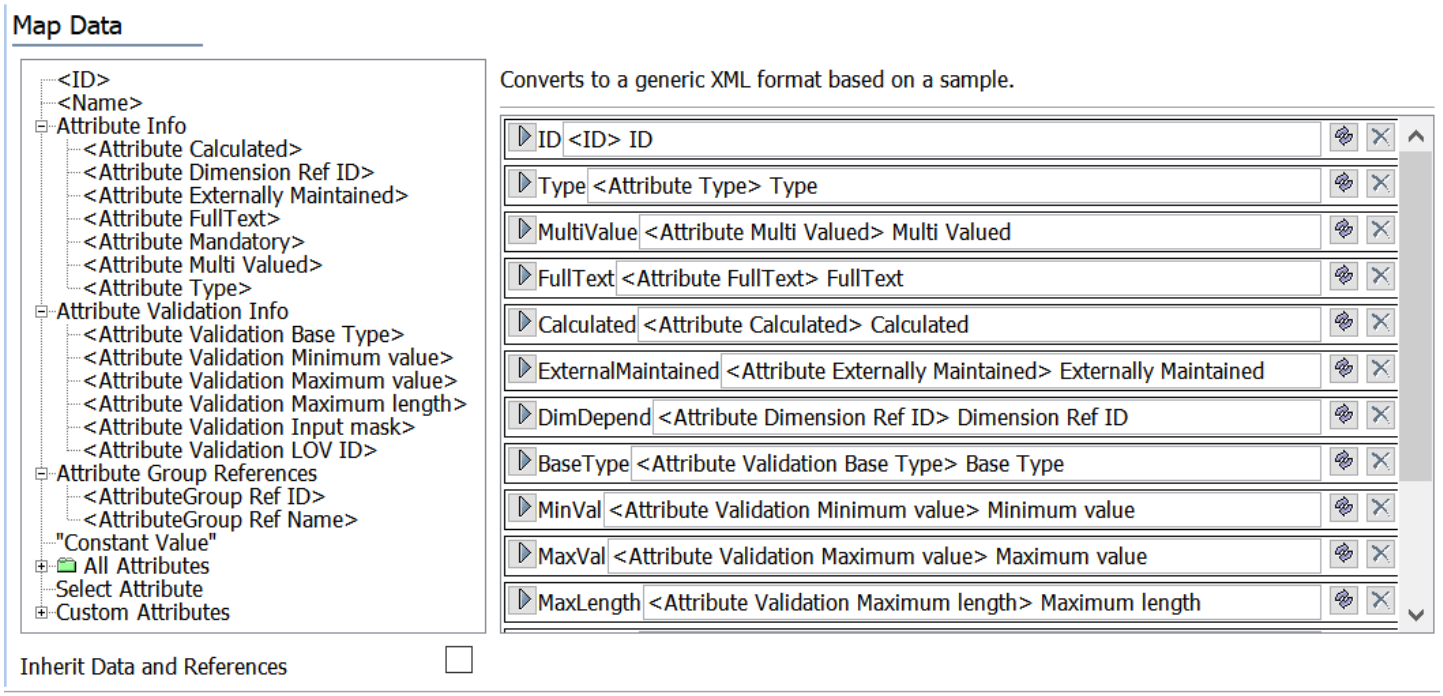
Results

A sample output of the above mapping is shown below.

```
<root>
  <attribute>
    <ID>Marketing Description</ID>
    <Info>
      <Type>Property</Type>
      <MultiValue>false</MultiValue>
      <FullText>false</FullText>
      <Calculated>true</Calculated>
      <ExternalMaintained>false</ExternalMaintained>
      <DimDepend>Language; Country</DimDepend>
    </Info>
    <ValidationInfo>
      <BaseType>text</BaseType>
      <MinVal/>
      <MaxVal/>
      <MaxLength>100</MaxLength>
      <InputMask/>
      <LOVID/>
    </ValidationInfo>
    <AttrGroupReference>
      <RefID>AttributeGroup</RefID>
      <RefName>Attribute Group</RefName>
    </AttrGroupReference>
  </attribute>
</root>
```

Attribute Aspects in Generic XML

Aspects, also called data sources, are displayed when mapping Attributes for export. The description of each aspect is included below.



Aspect	Description
ID	Extracts only the ID of the attribute not the attribute value. For example, if an attribute has ID 1234 and name 'Weight,' then '1234' is extracted.
Name	Extracts only the name of the attribute and not the attribute value. For example, if an attribute has ID 1234 and name 'Weight,' then the word 'Weight' is extracted.
Attribute Type	Extracts the type of the attribute, i.e., either Description or Specification. For example, for a description attribute the word 'Description' is extracted.
Attribute Multi Valued	Extracts if the attribute is multi-valued. If the attribute is a multi-valued attribute the word 'true' is extracted, otherwise the word 'false' is extracted.

Aspect	Description
Attribute FullText	Extracts if the attribute is full text indexable. If the attribute is a full text indexable attribute the word 'true' is extracted, otherwise the word 'false' is extracted.
Attribute Calculated	Extracts if the attribute is a calculated attribute. If the attribute is a calculated attribute the word 'true' is extracted, otherwise the word 'false' is extracted.
Attribute Externally Maintained	Extracts if the attribute is externally maintained. If the attribute is externally maintained the word 'true' is extracted, otherwise the word 'false' is extracted.
Attribute Dimension Ref ID	Extracts if the attribute has dimension dependencies. If the attribute has dimension dependencies they are listed and separated by a semicolon (;).
Attribute Validation Base Type	Extracts the validation base type of the attribute. For example, if the attribute has Text as validation base type, the word 'text' is extracted.
Attribute Validation Minimum value	Extracts the minimum value for the attribute, if available. For example, for a number attribute with a minimum value of 100, the number '100' is extracted. For a number attribute without a minimum value, a single target tag is extracted with a trailing slash (i.e., <MinVal/>).
Attribute Validation Maximum value	Extracts the maximum value for the attribute, if available. For example, for a number attribute with a maximum value of 500, the number '500' is extracted. For a number attribute without a maximum value, a single target tag is extracted with a trailing slash (i.e., <MaxVal/>).
Attribute Validation Maximum length	Extracts the maximum length of the attribute, if available. For example, for an attribute with a maximum length of 20, the number '20' is extracted. For a number attribute without a maximum length, a single target tag is extracted with a trailing slash (i.e., <MaxLen/>).

Aspect	Description
Attribute Validation Input mask	Extracts the attribute validation input mask, if available. For example, if an attribute has an attribute validation mask of 'F0,' then 'F0' is extracted. For an attribute without an input mask, a single target tag is extracted with a trailing slash (i.e., <InputMask/>).
Attribute Validation LOV ID	Extracts the ID of the List Of Values, if available. For example, for an attribute configured to use the List Of Value with ID Colors, the word 'Colors' is extracted. For an attribute without an LOV, a single target tag is extracted, with a trailing slash (i.e., <LOVID/>).
AttributeGroup Ref ID	Extracts a list of IDs of the attribute groups that include the attribute. Each group ID is separated by a semicolon (;). For example, if the attribute is in attribute groups with ID Group1 and ID Group2 then 'Group1;Group2' is extracted.
AttributeGroup Ref Name	Extracts a list of names of the attribute groups that include the attribute. Each group name is separated by a semicolon (;). For example, if the attribute is in attribute groups named GroupName1 and GroupName2, then 'GroupName1;GroupName2' is extracted.

For a description of the Constant Value, All Attributes, Select Attributes, and Custom Attributes aspects, refer to **Outbound Map Data - Data Source** section.

Attribute Links and LOVs Processing Instructions in Generic XML

Using the following Generic XML template and mapping allows the export of System Setup information along with Product information. In this scenario, the following is exported:

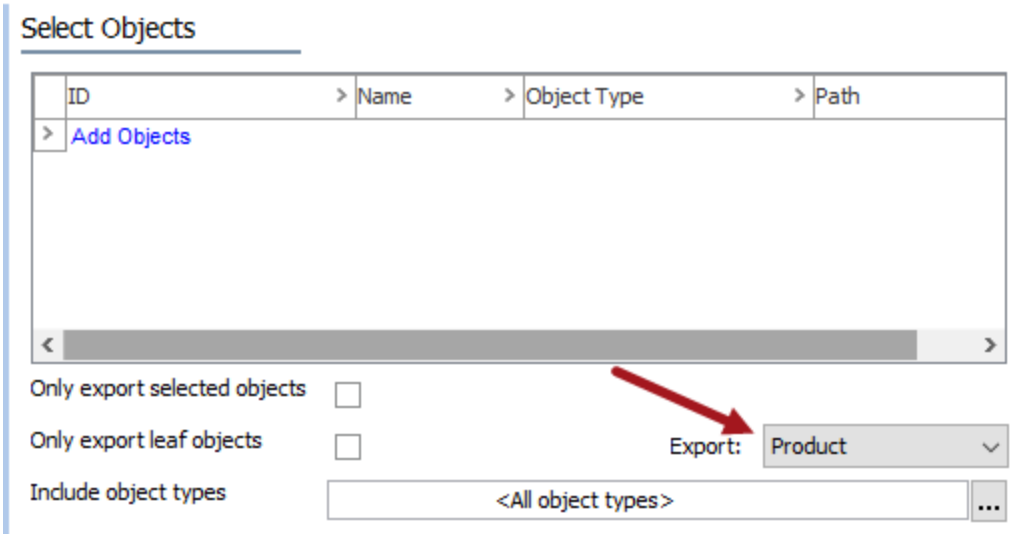
- the ID and Name of the Product
  - the ID and Name of the linked attributes on the Product
    - the ID, Name, and Values of the LOV on any of the linked attributes

Units for linked attributes can also be mapped and exported as discussed in **Attribute Links and Units Processing Instructions in Generic XML**.

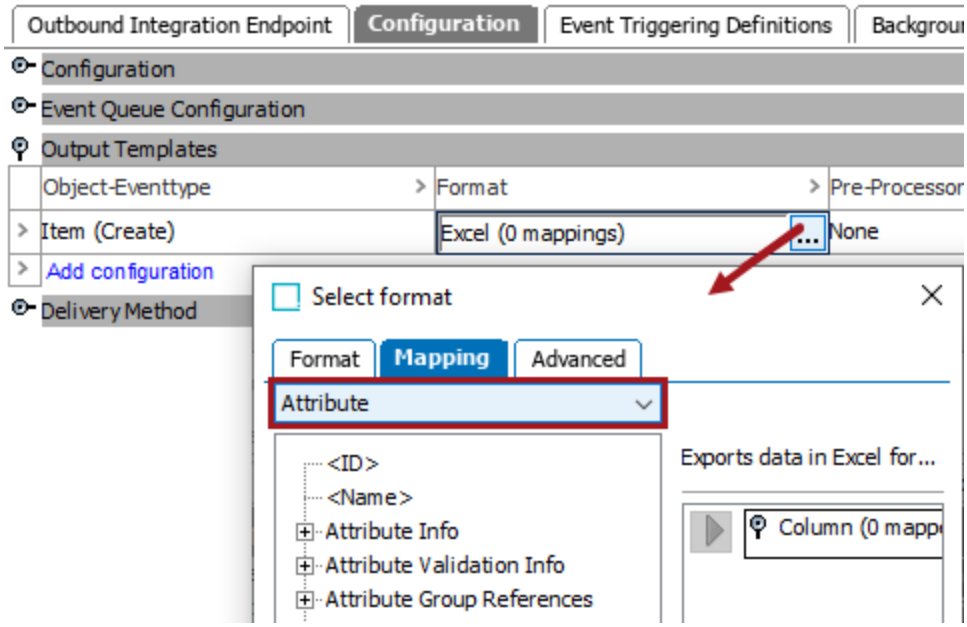
The data path data source is also available using the IDoc MATMAS 05 format.

Objects

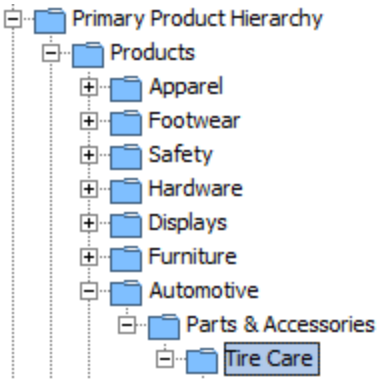
Choose **Product** as object type to export in Export Manager:



or in an OIEP:



Use the **Add Objects** link to select the data to export.



Template

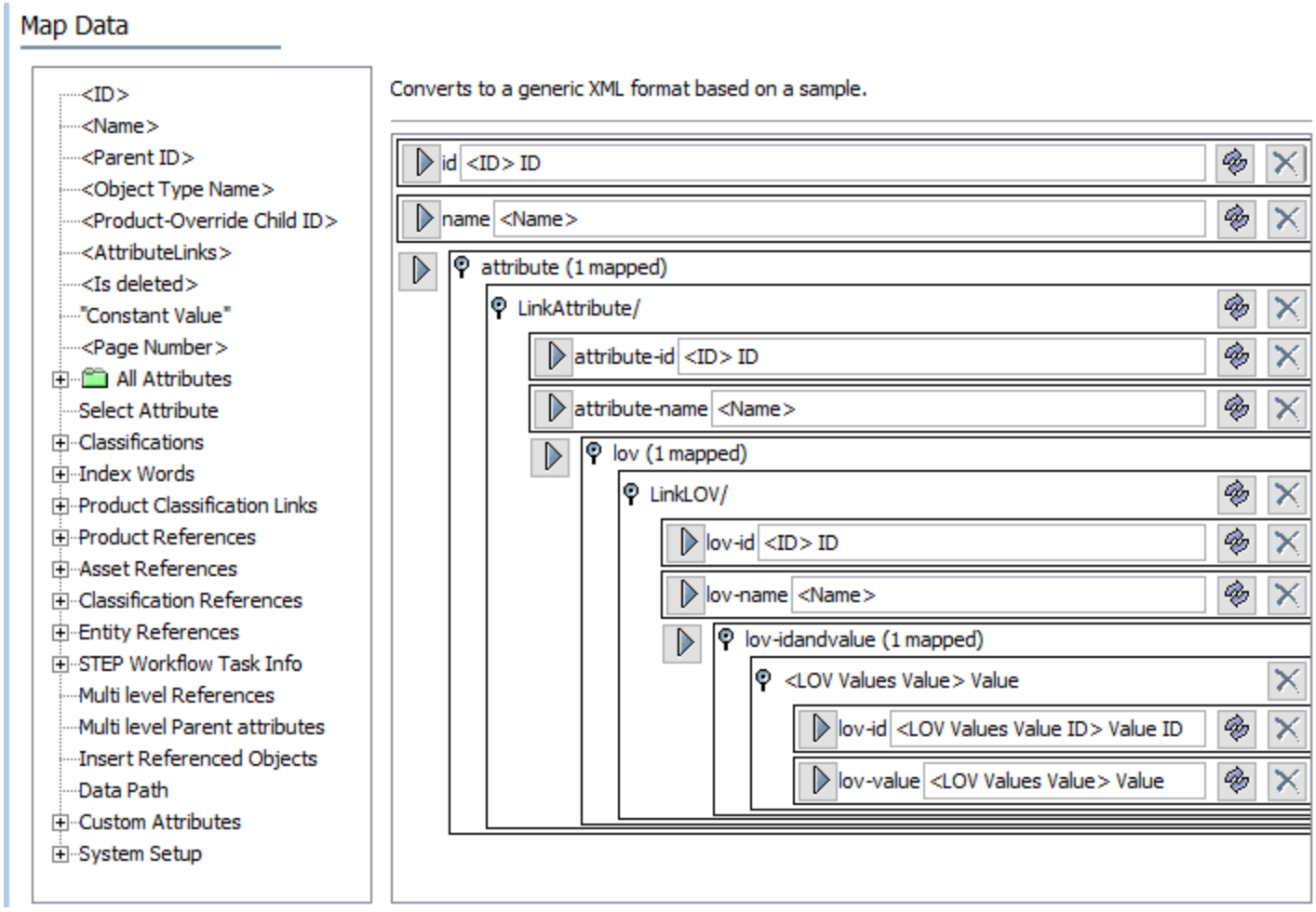
Choose **Generic XML** for format and provide the following template text in the Sample field.

```
    <products>
       <product>
          <?Product?>
          <id><?Target?></id>
          <name><?Target?></name>
       <attribute>
          <?MultiTarget?>
```

```
            <id><?Target attribute-id?></id>
            <name><?Target attribute-name?></name>
         <lov>
            <?MultiTarget?>
            <id><?Target lov-id?></id>
            <name><?Target lov-name?></name>
         <lov-idandvalue>
            <?MultiTarget?>
            <value><?Target lov-value?></value>
            <id><?Target lov-id?></id>
         </lov-idandvalue>
         </lov>
         </attribute>
         </product>
      </products>
```

Mapping

The Data Path aspect enables the complete set of mapping rules displayed below:



For details on mapping, refer to **Mapping Attribute Links, LOVs, and LOV Values in Generic XML**.

Results

The Product, its linked Attributes, and any LOVs (and the values) used by the linked attributes are exported.

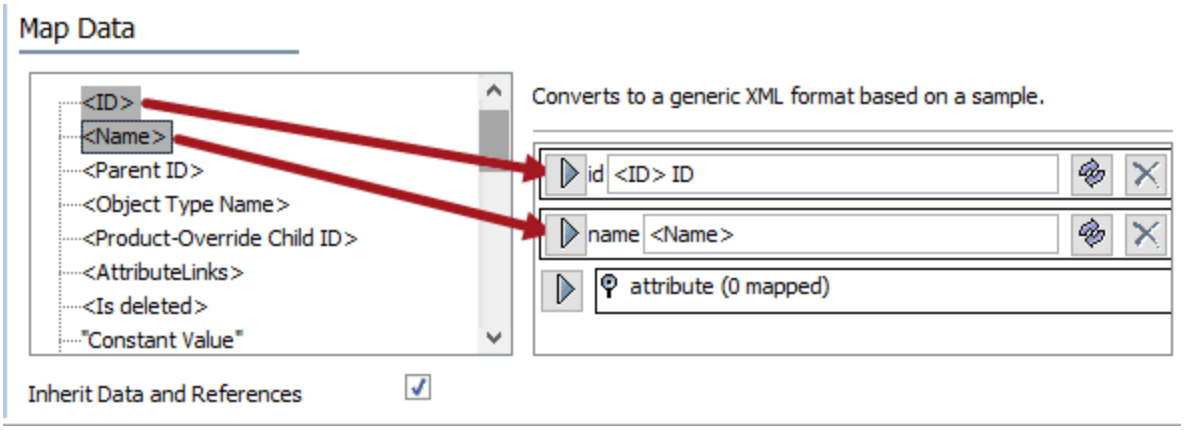
Note: Only local attribute links are exported.

```
<product>
  <id>8312</id>
  <name>Tire Care</name>
  <attribute>
    <id>AirGaugeIncluded</id>
    <name>Air gauge included</name>
    <lov>
      <id>YesNo</id>
      <name>Yes/No (translatable)</name>
      <lov-idandvalue>
        <id/>
        <value>No</value>
      </lov-idandvalue>
      <lov-idandvalue>
        <id/>
        <value>Yes</value>
      </lov-idandvalue>
    </lov>
  </attribute>
  <attribute>
    <id>PressureReliefValve</id>
    <name>Pressure Relief Valve</name>
    <lov>
      <id>YesNo</id>
      <name>Yes/No (translatable)</name>
      <lov-idandvalue>
        <id/>
        <value>No</value>
      </lov-idandvalue>
      <lov-idandvalue>
        <id/>
        <value>Yes</value>
      </lov-idandvalue>
    </lov>
  </attribute>
  <attribute>
    <id>HoseLength</id>
    <name>Hose length</name>
  </attribute>
</product>
```

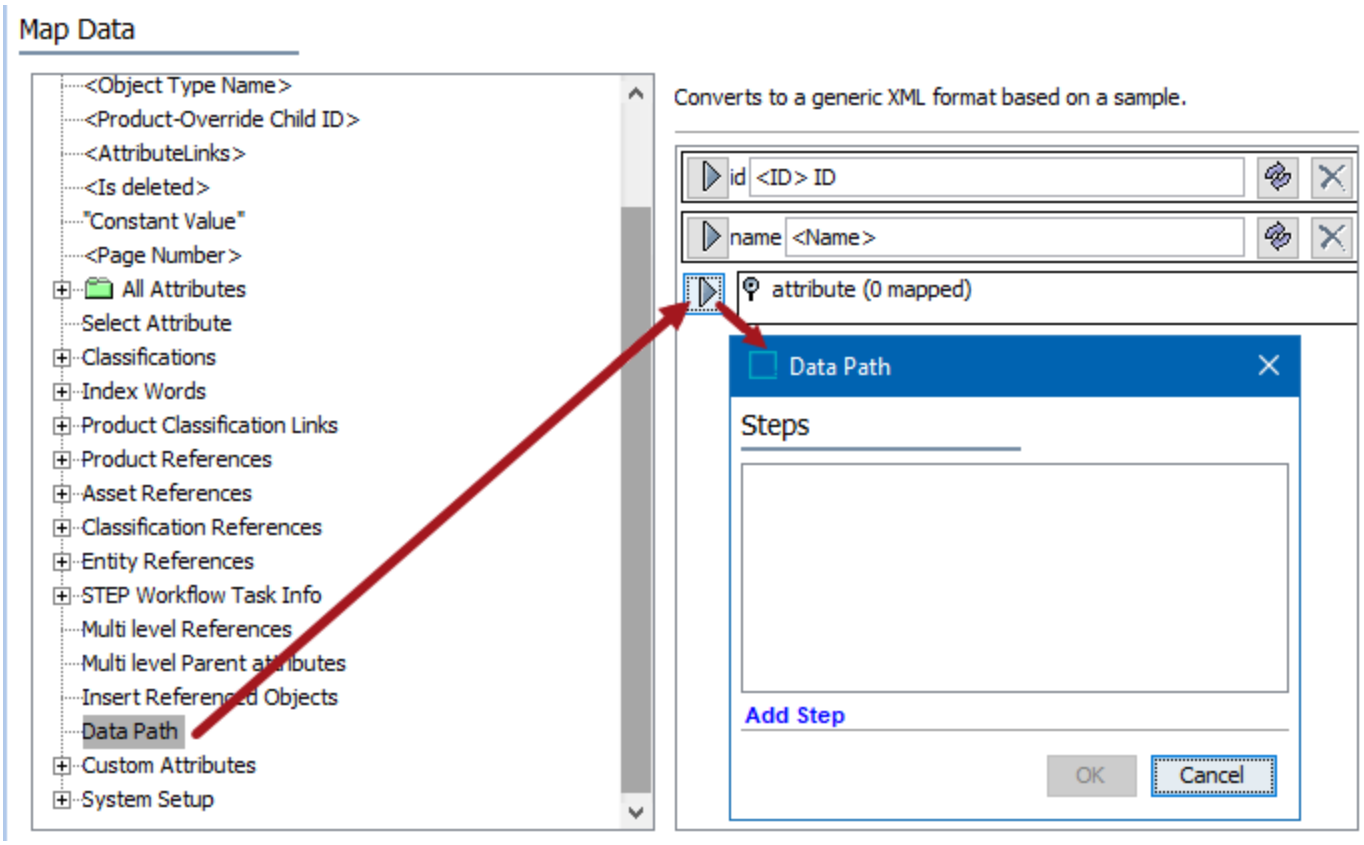
Mapping Attribute Links, LOVs, and LOV Values in Generic XML

The **Attribute Links and LOVs Processing Instructions in Generic XML** section discusses additional setup for a Generic XML or IDoc MATMAS 05 export. Use the following steps to map the Product, linked attributes, and their LOV information.

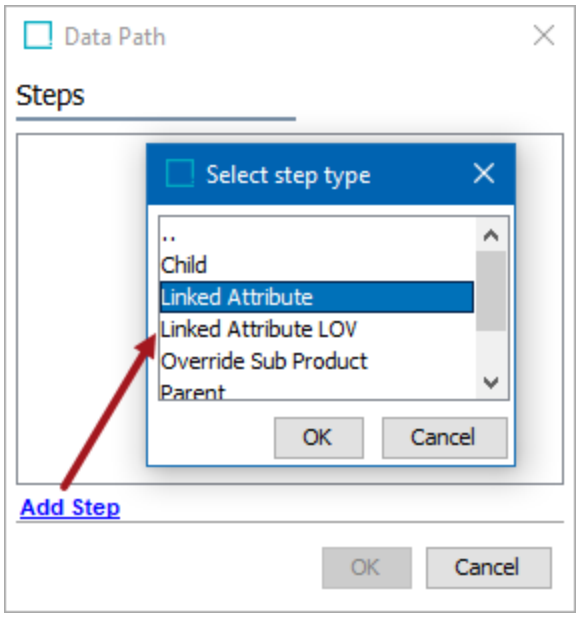
1. Map **ID** and **Name** for the Product.



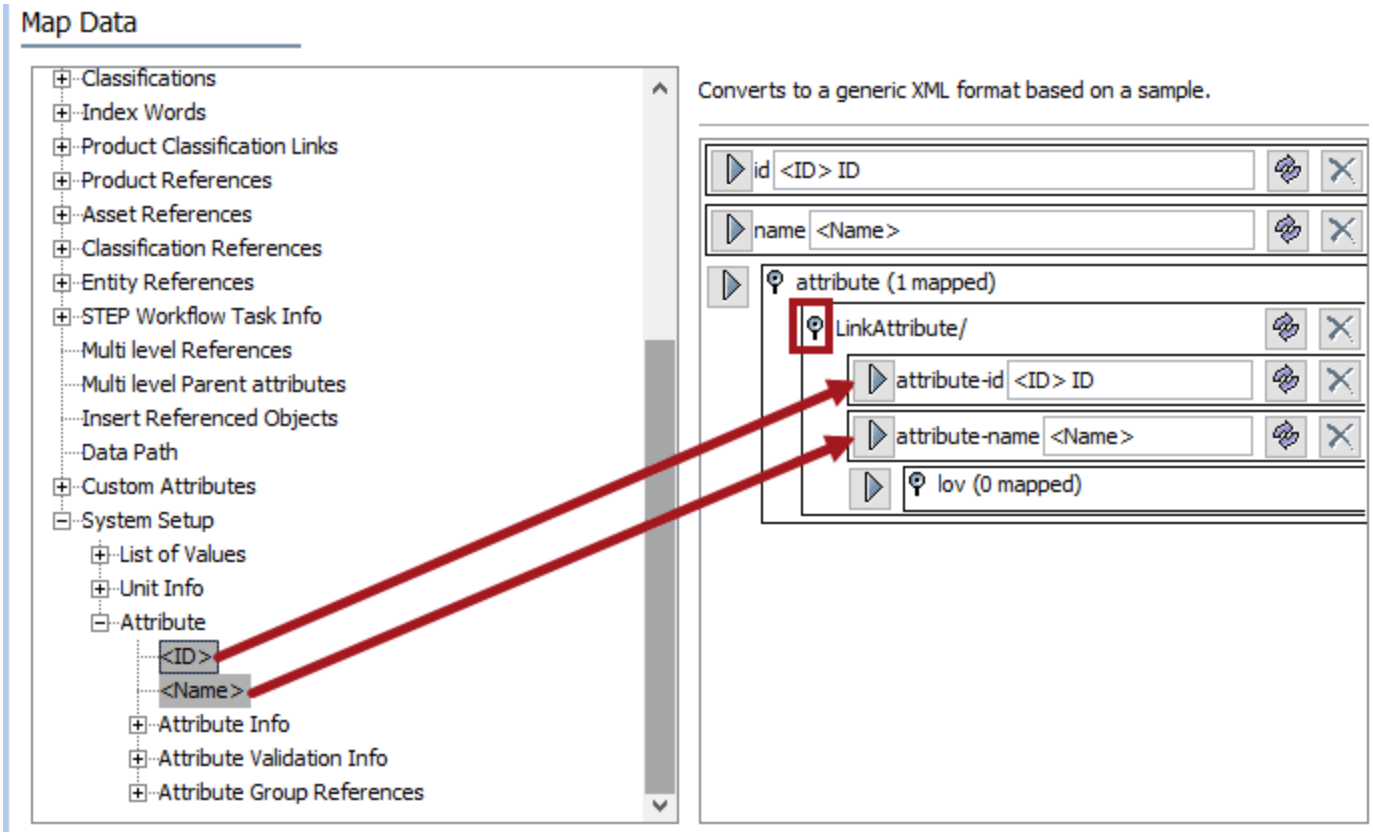
2. Map **Data Path** to attribute.



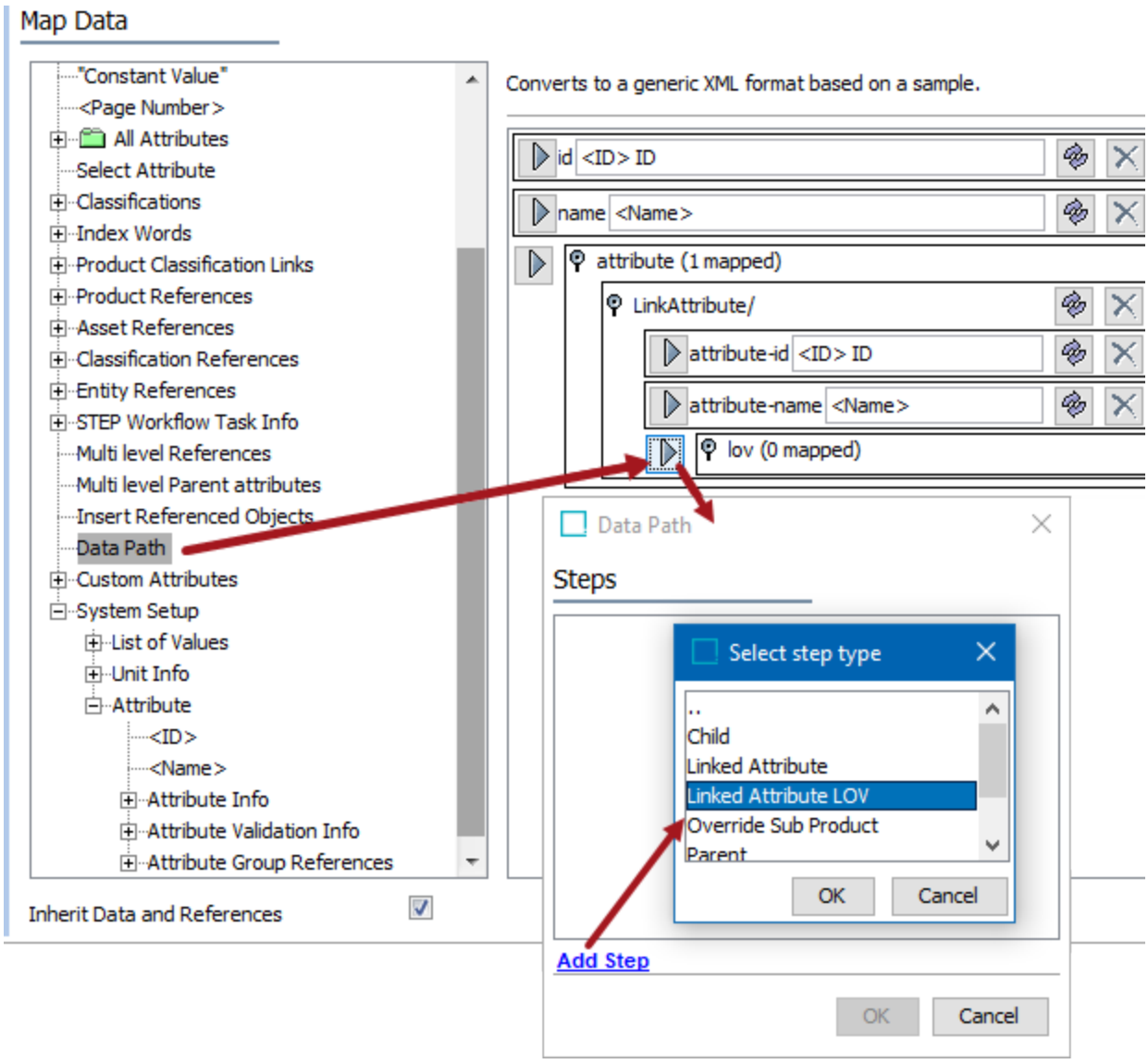
3. Click the **Add Step** link and select **Linked Attribute**.



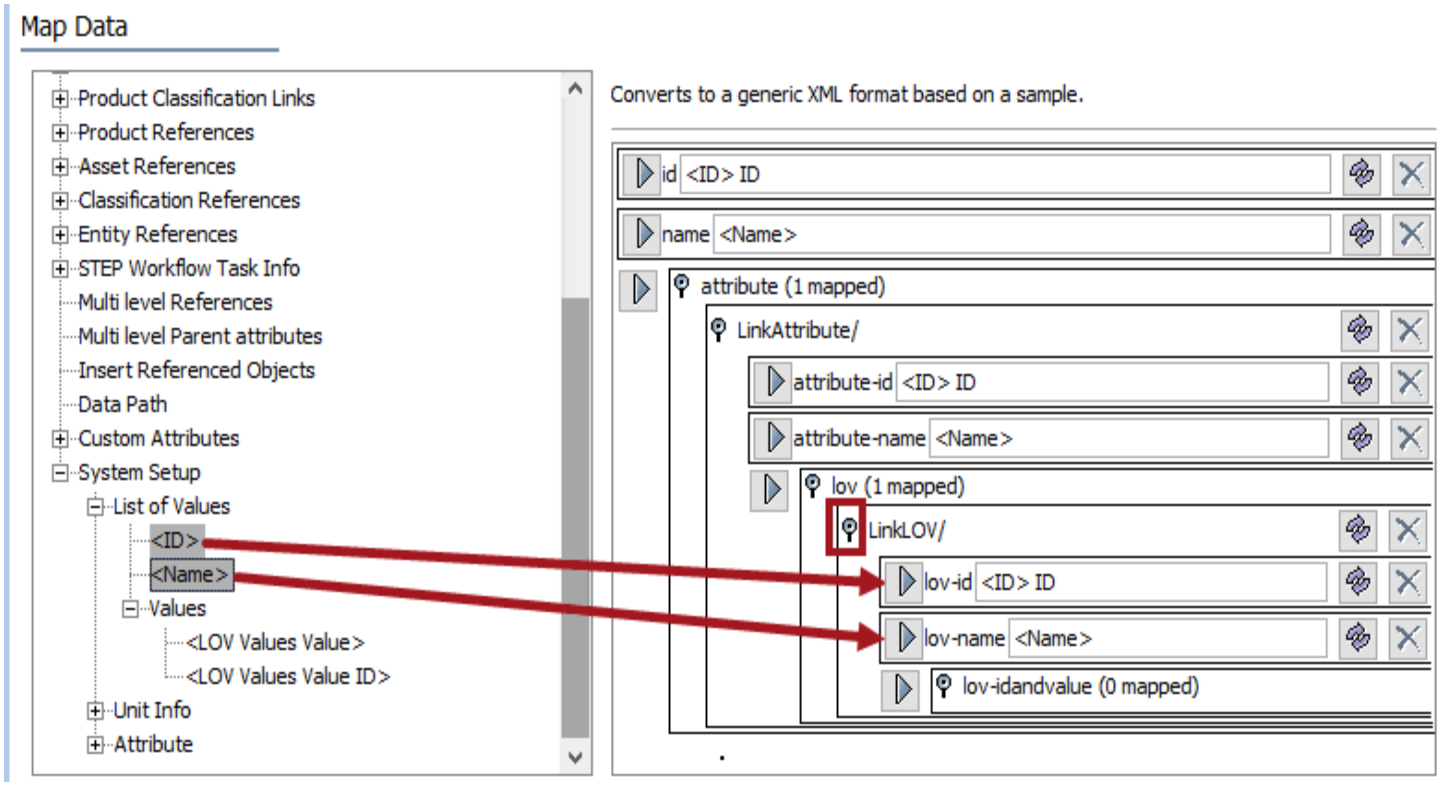
4. Click OK to close the **Select step type** and **Data Path** dialogs.
5. Open the LinkAttribute/ flipper and map Attribute **ID** and **Name**.



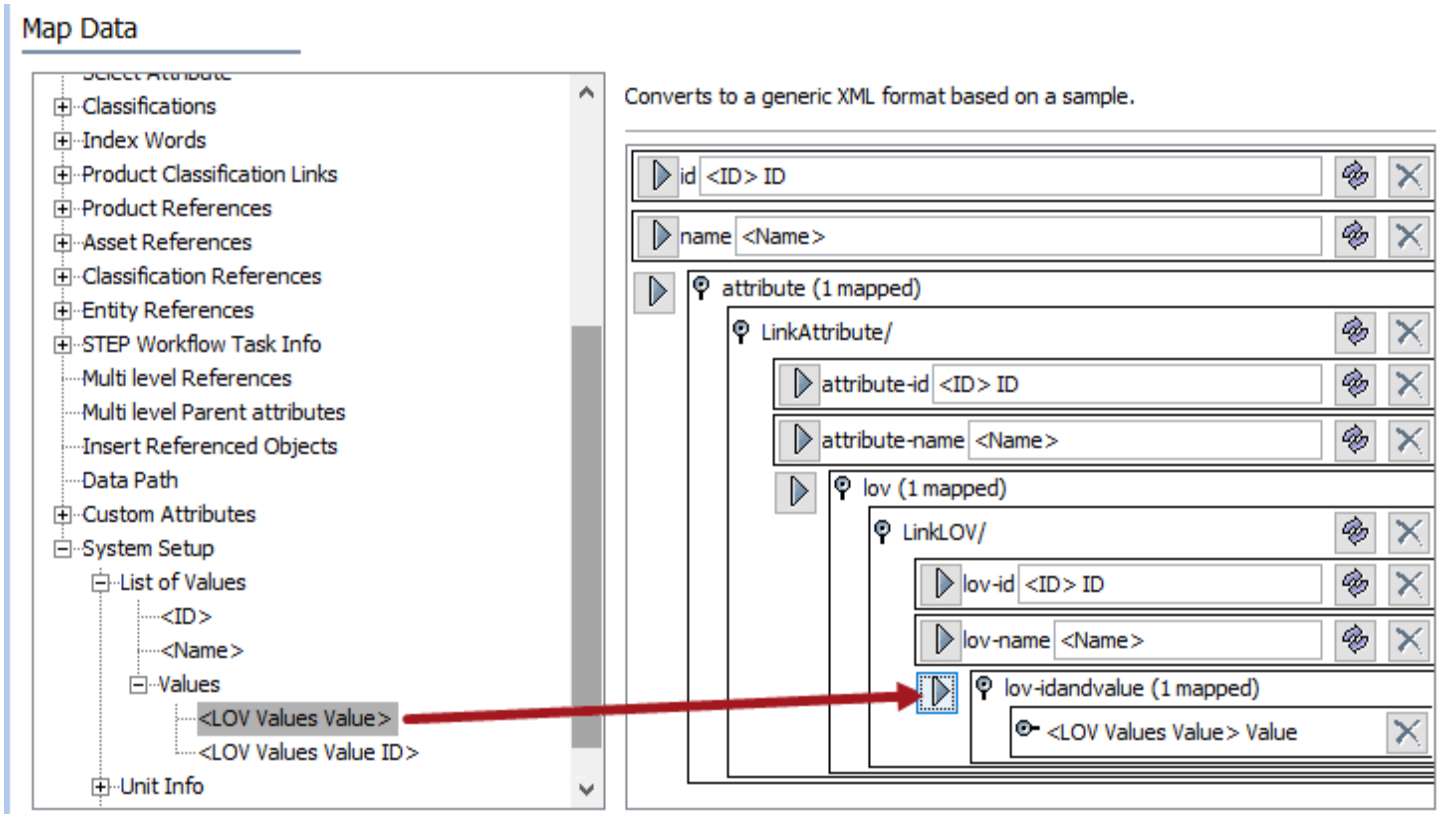
6. Map **Data Path** to lov, click **Add Step** and select **Linked Attribute LOV**.



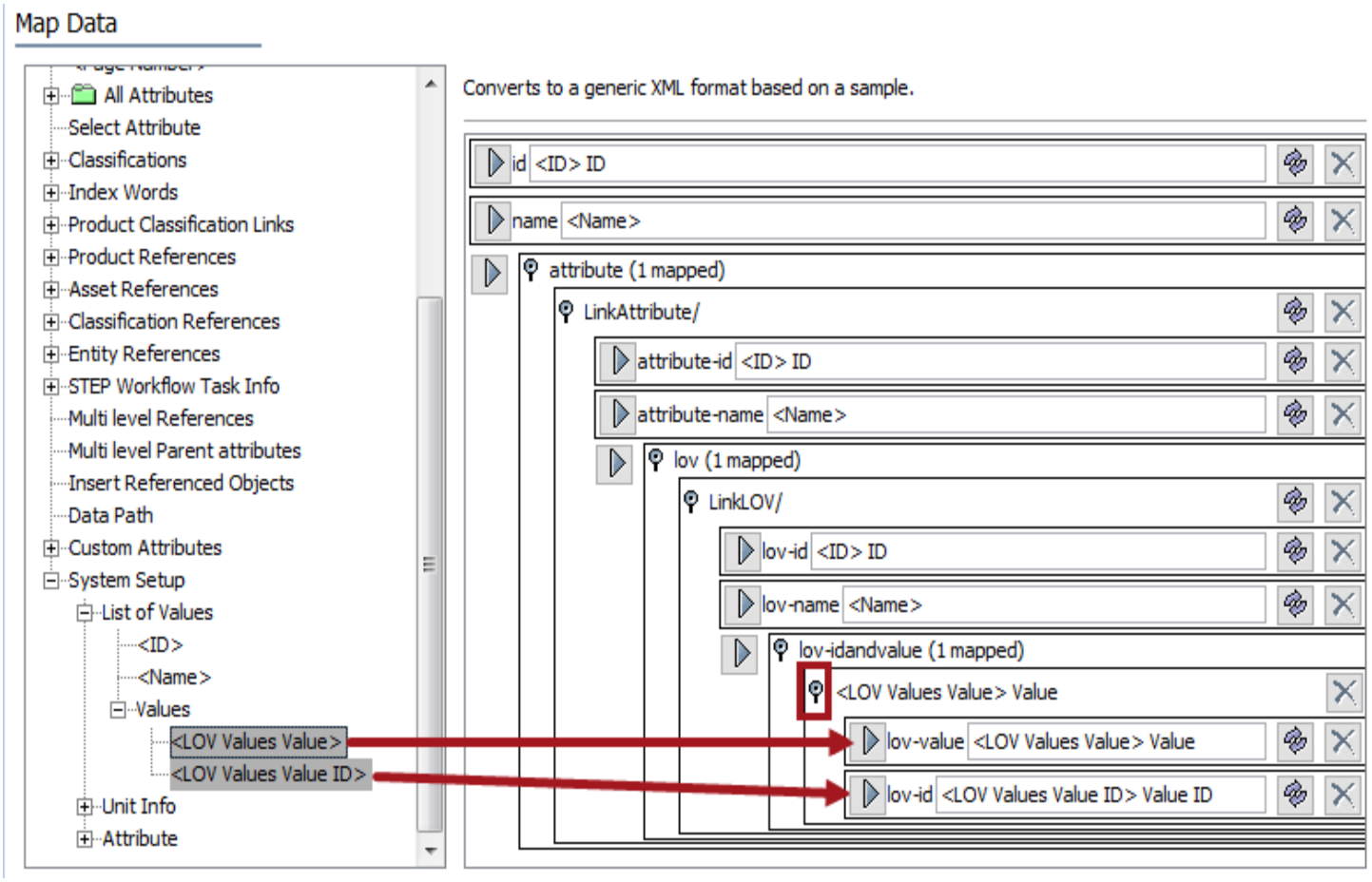
7. Click OK to close the **Select step type** and **Data Path** dialogs.
8. Open the LinkLOV/ flipper and map Attribute **ID** and **Name**.



9. Map **LOV Values Value** to lov-idandvalue.



10. Open the <LOV Values Value> Value flipper and map **LOV Values Value** and **LOV Values ID**.



Attribute Links and Units Processing Instructions in Generic XML

Using the following Generic XML template and mapping allows the export of System Setup information along with Product information. In this scenario, the following is exported:

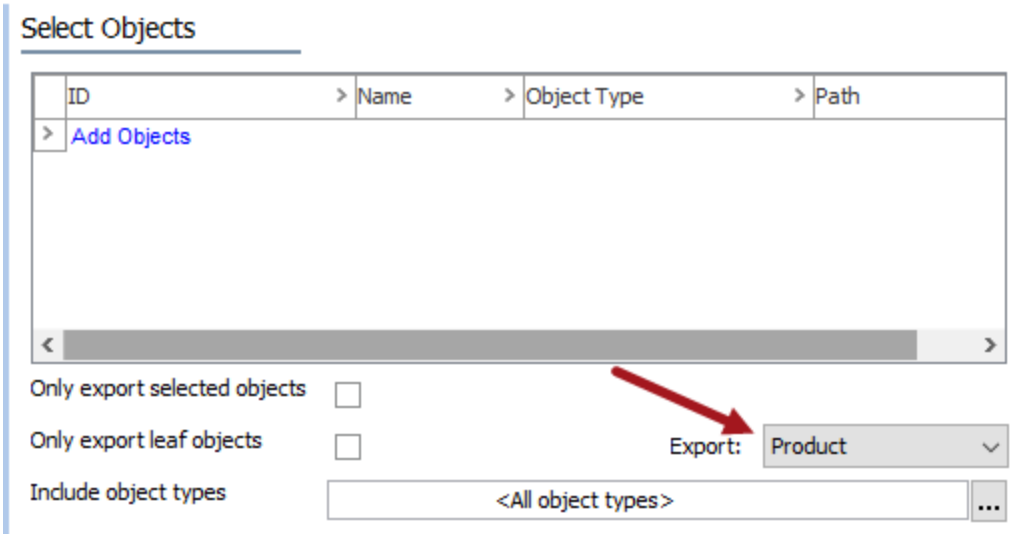
- the ID and Name of the Product
  - the ID and Name of the linked attributes on the Product
    - the ID, Name, and Base of the Unit on any of the linked attributes

LOVs for linked attributes can also be mapped and exported as described in **Attribute Links and LOVs Processing Instructions in Generic XML**.

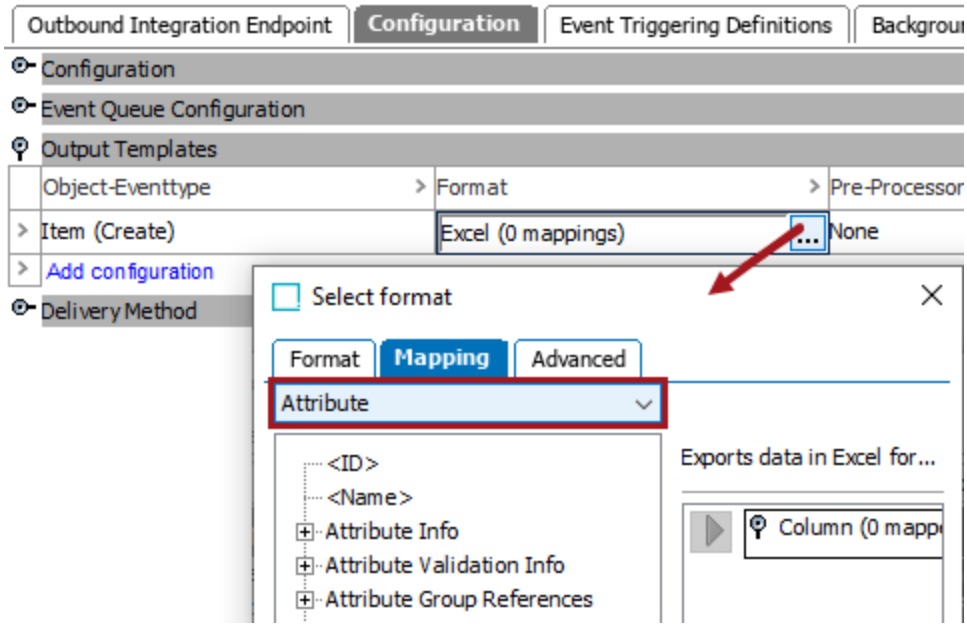
The data path data source is also available using the IDoc MATMAS 05 format.

Objects

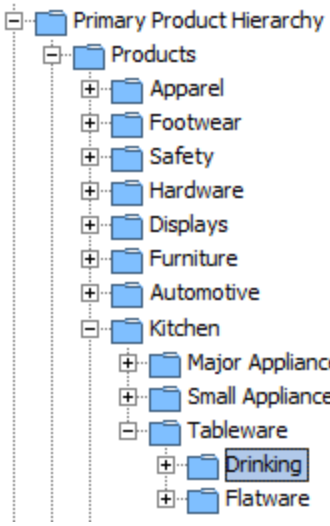
Choose **Product** as object type to export in Export Manager:



or in an OIEP:



Use the **Add Objects** link to select the data to export.



Template

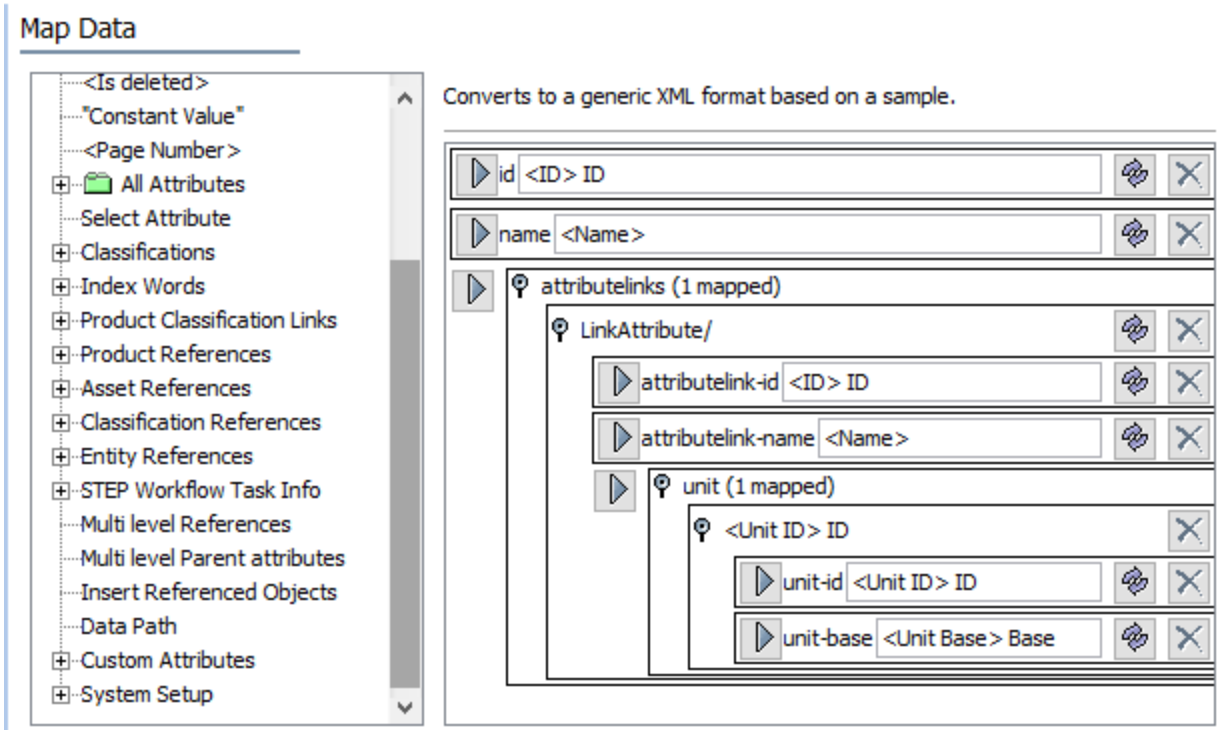
Choose **Generic XML** for format and provide the following template text in the Sample field.

```
<products>
   <product>
      <?Product?>
```

```
        <id><?Target?></id>
        <name><?Target?></name>
      <attributelinks>
        <?MultiTarget?>
        <id><?Target attributelink-id?></id>
        <name><?Target attributelink-name?></name>
      <unit>
        <?MultiTarget?>
        <attributelink-unit-id><?Target unit-id?></attributelink-unit-id>
        <attributelink-unit-base><?Target unit-base?></attributelink-unit-base>
      </unit>
      </attributelinks>
      </product>
    </products>
```

Mapping

The Data Path aspect enables the complete set of mapping rules displayed below:



For details on mapping, refer to the **Mapping Attribute Links, Units, and Unit Base in Generic XML** section.

Results

The Product, its linked Attributes, and any Units (and the Base) used by the linked attributes are exported.

Note: Only local attribute links are exported.

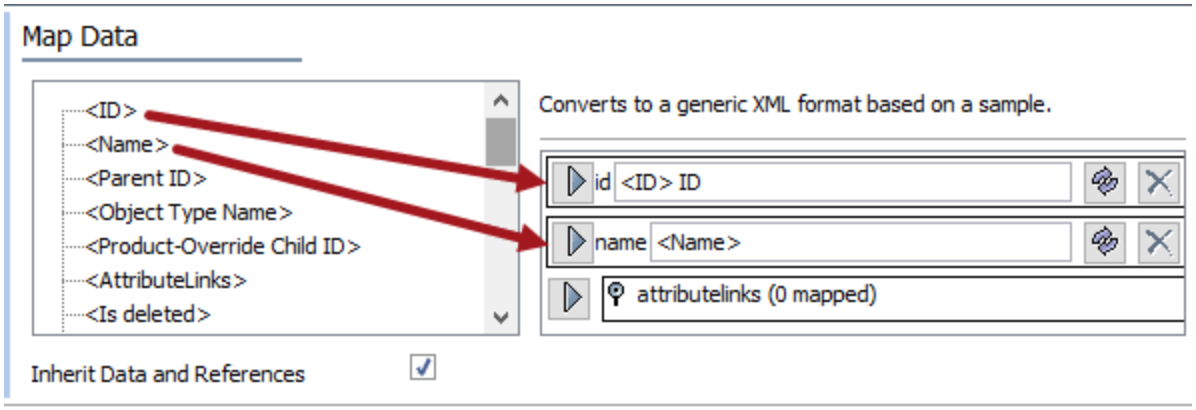
```
<products>
  <product>
    <id>8318</id>
    <name>Drinking</name>
    <attributelinks>
      <id>Capacity</id>
      <name>Capacity</name>
      <unit>
        <attributelink-unit-id>unece.unit.FTK</attributelink-unit-id>
        <attributelink-unit-base>unece.unit.MTK</attributelink-unit-base>
      </unit>
      <unit>
        <attributelink-unit-id>unece.unit.MTK</attributelink-unit-id>
        <attributelink-unit-base/>
      </unit>
      <unit>
        <attributelink-unit-id>unece.unit.OZA</attributelink-unit-id>
        <attributelink-unit-base>unece.unit.MTQ</attributelink-unit-base>
      </unit>
    </attributelinks>
  </product>
</products>
```

Mapping Attribute Links, Units, and Unit Base in Generic XML

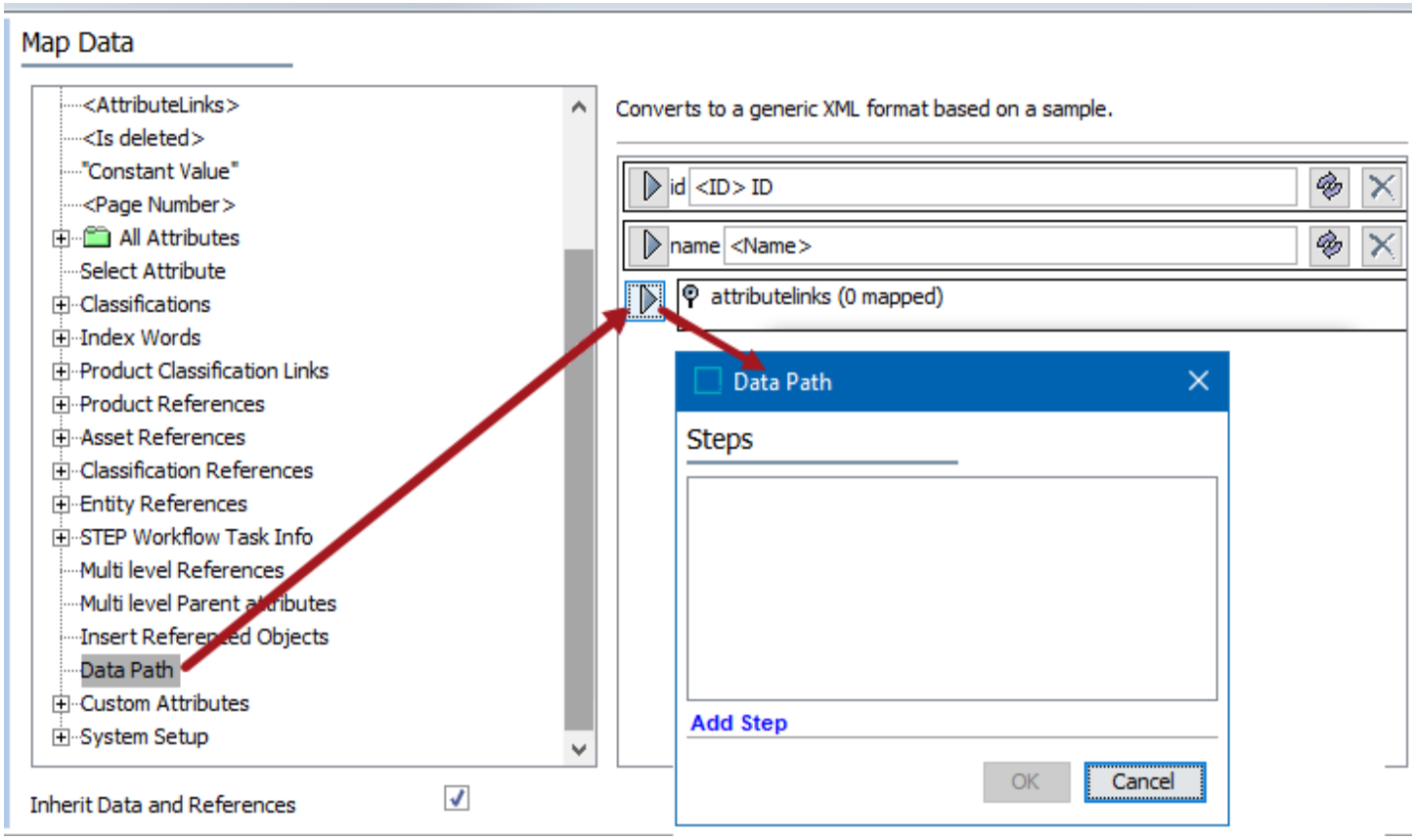
The **Attribute Links and Units Processing Instructions in Generic XML** section discusses additional setup for a Generic XML export. Use the following steps to map the Product, linked attributes, and their Unit information.

The data path data source is also available using the IDoc MATMAS 05 format.

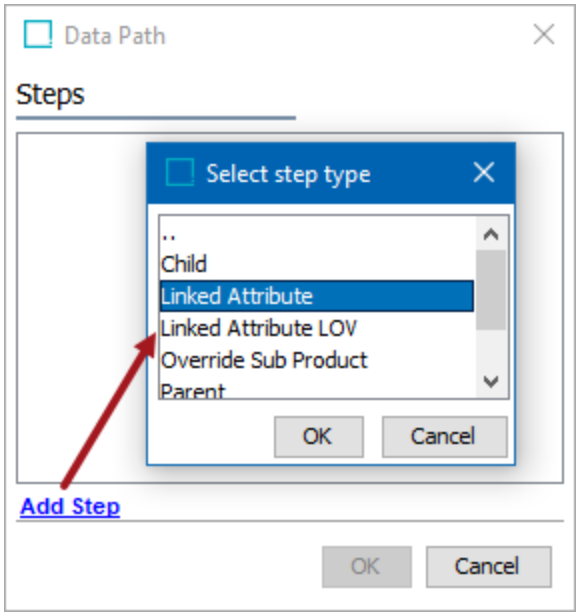
1. Map **ID** and **Name** for the Product.



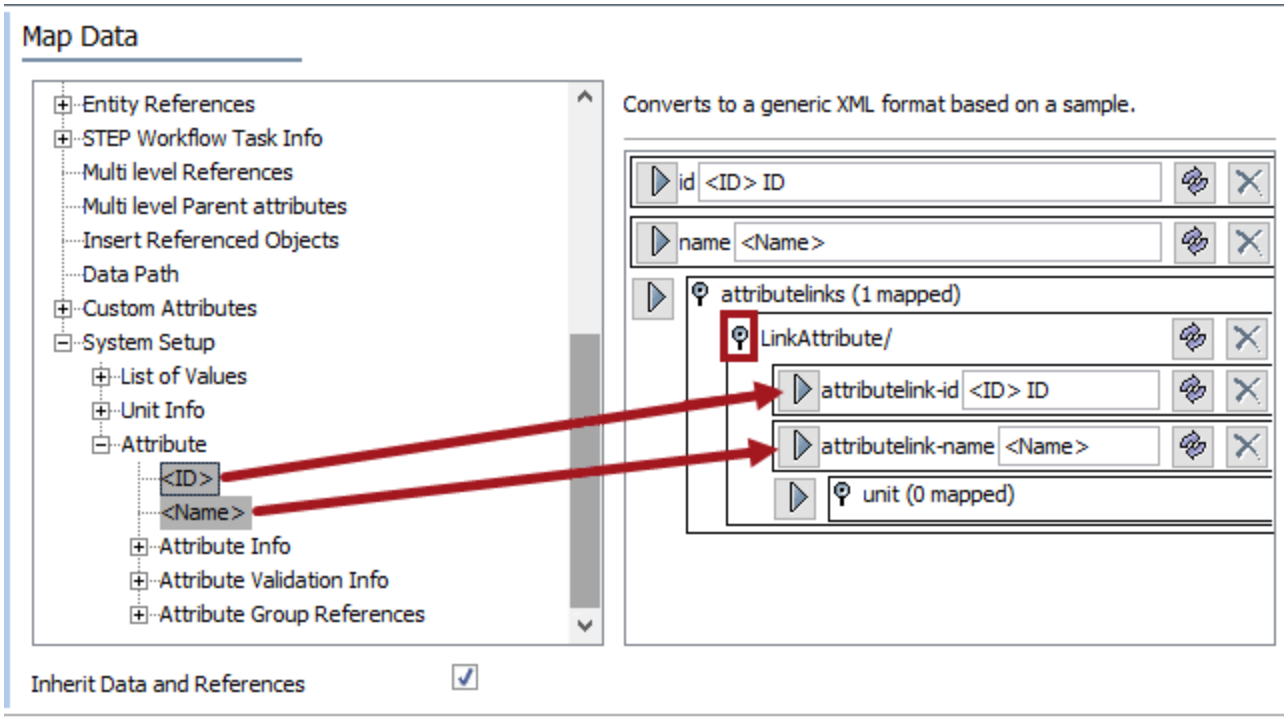
2. Map **Data Path** to attributelinks.



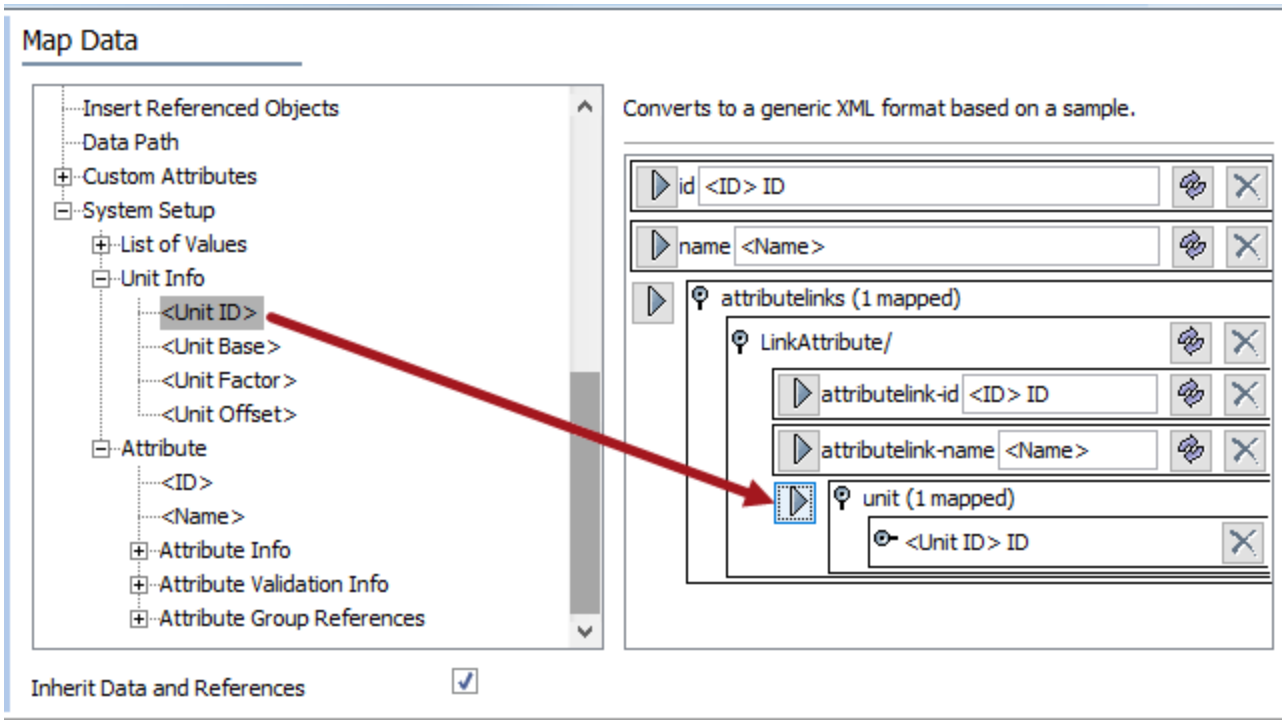
3. Click the **Add Step** link and select **Linked Attribute**.



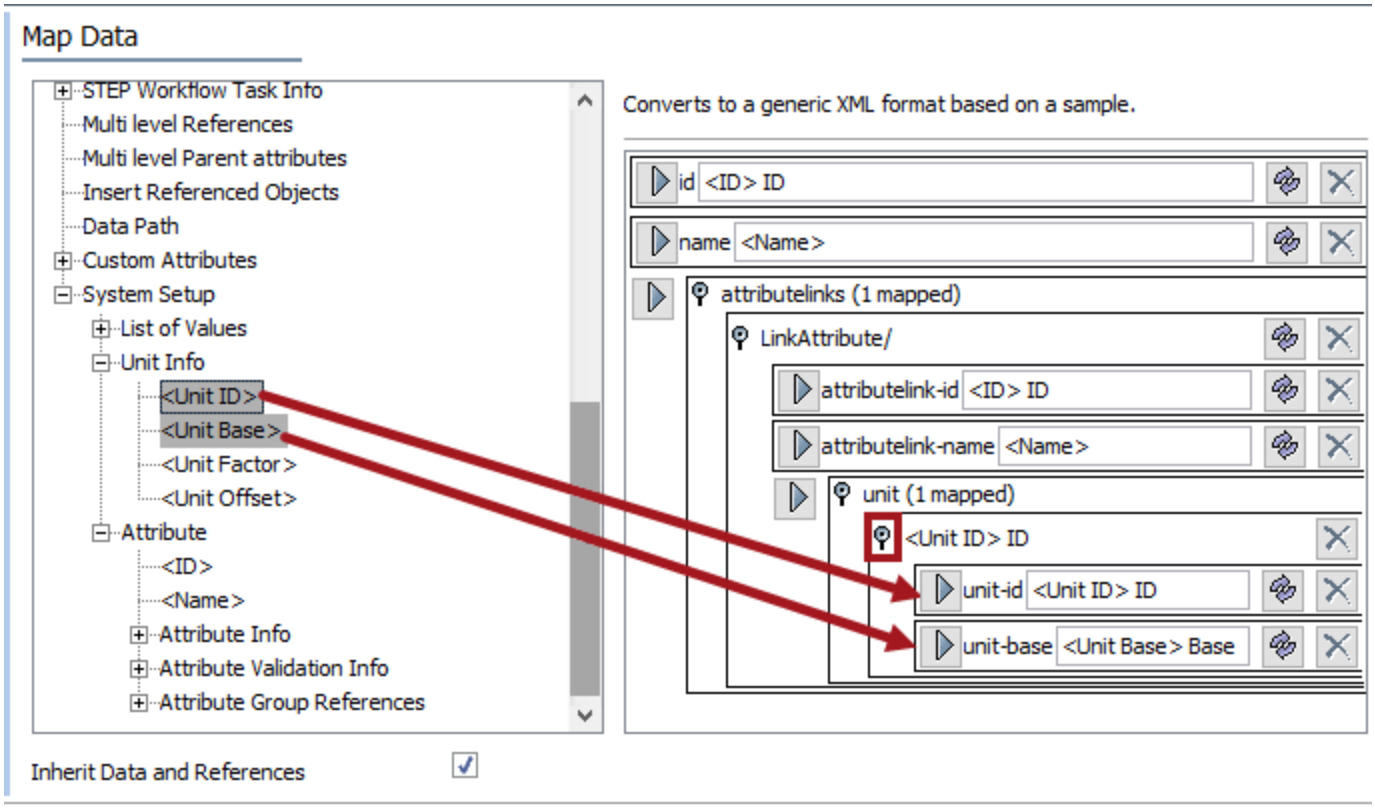
4. Click OK to close the **Select step type** and **Data Path** dialogs.
5. Open the LinkAttribute/ flipper and map Attribute **ID** and **Name**.



6. Map **Unit ID** to unit.



7. Open the <Unit ID> flipper and map Unit **ID** and **Base**.

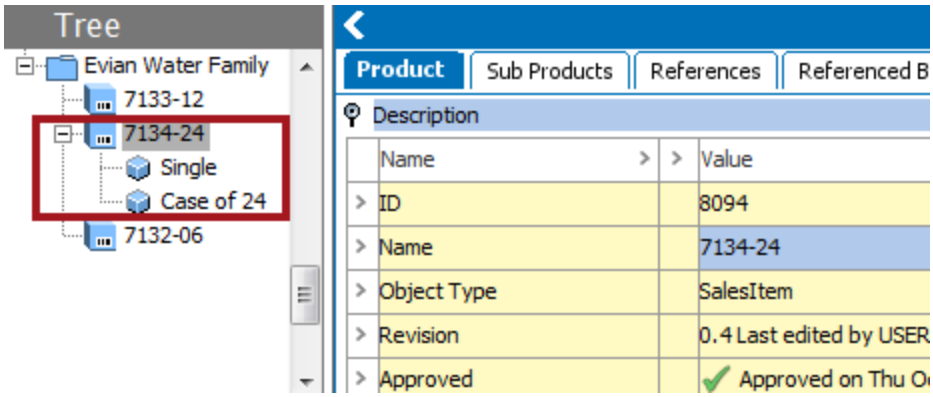

Map Data
STEP Workflow Task Info
Multi level References
Multi level Parent attributes
Insert Referenced Objects
Data Path
Custom Attributes
System Setup
List of Values
Unit Info
<Unit ID>
<Unit Base>
<Unit Factor>
<Unit Offset>
Attribute
<ID>
<Name>
Attribute Info
Attribute Validation Info
Attribute Group References
Converts to a generic XML format based on a sample.
id <ID> ID
name <Name>
attributelinks (1 mapped)
LinkAttribute/
attributelink-id <ID> ID
attributelink-name <Name>
unit (1 mapped)
<Unit ID> ID
unit-id <Unit ID> ID
unit-base <Unit Base> Base
Inherit Data and References

DoNotEmit Processing Instructions in Generic XML

The Generic XML <?DoNotEmit?> tag allows you to hide levels that exist for the exported values or references from different products, by displaying data on the same level. The tag surrounding the DoNotEmit instruction is excluded from the output.

Objects

The following SalesItem and children are selected for export.



Template

These templates allow you to compare the difference that using DoNotEmit has on the output.

Without DoNotEmit

```
<Families>
   <Family ID="[?Target ID?]">
      <?Record?>
      <Name><?Target Name?></Name>
   <Children>
   <Child>

      <?MultiTarget?>
      <ID><?Target ID?></ID>
      <Name><?Target Name?></Name>
   </Child>
```

```
        </Children>
        </Family>
    </Families>
```

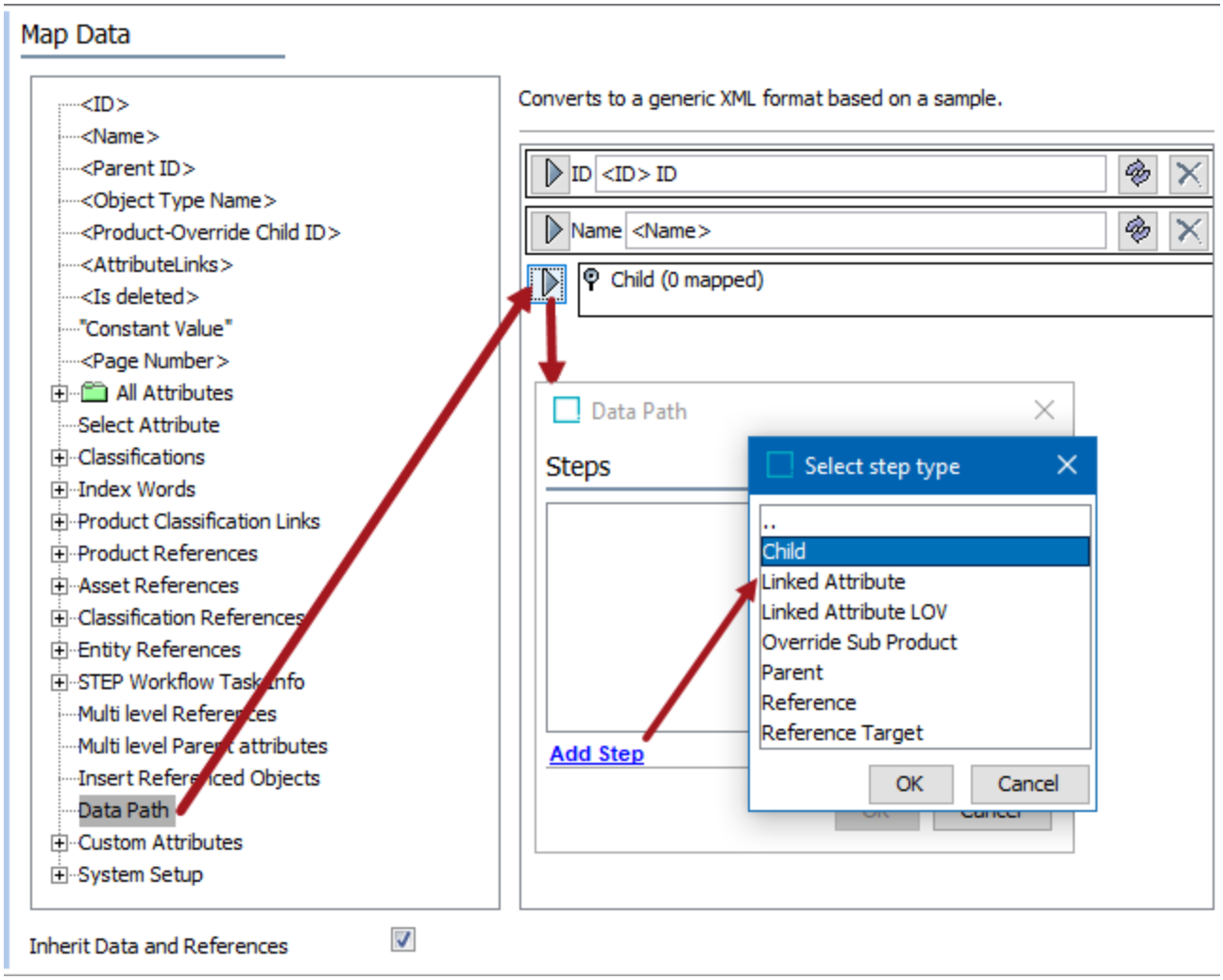
With DoNotEmit

```
    <Families>
        <Family ID="[?Target ID?]">
            <?Record?>
            <Name><?Target Name?></Name>
        <Children>
        <Child>
            <?DoNotEmit?>
            <?MultiTarget?>
            <ID><?Target ID?></ID>
            <Name><?Target Name?></Name>
        </Child>
        </Children>
        </Family>
    </Families>
```

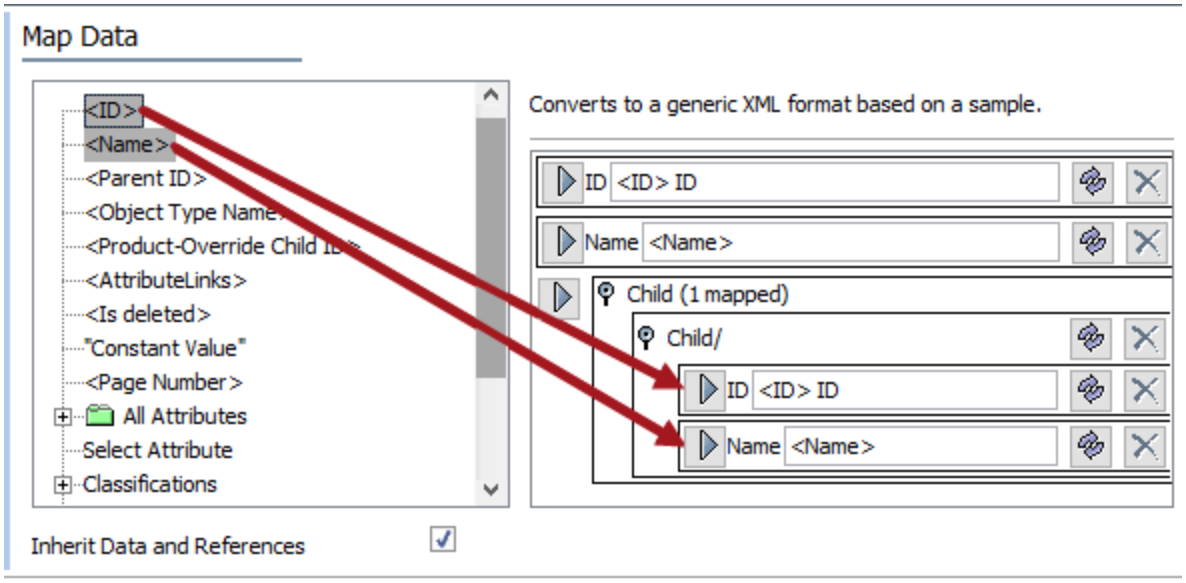
Mapping

Mapping for export is the same for both templates.

1. After mapping the SalesItem ID and Name, use the Data Path - Child option to map the Child target.



2. Map the Child's ID and Name.



Results

The following results are achieved when exporting using the sample templates and mapping above.

Without DoNotEmit

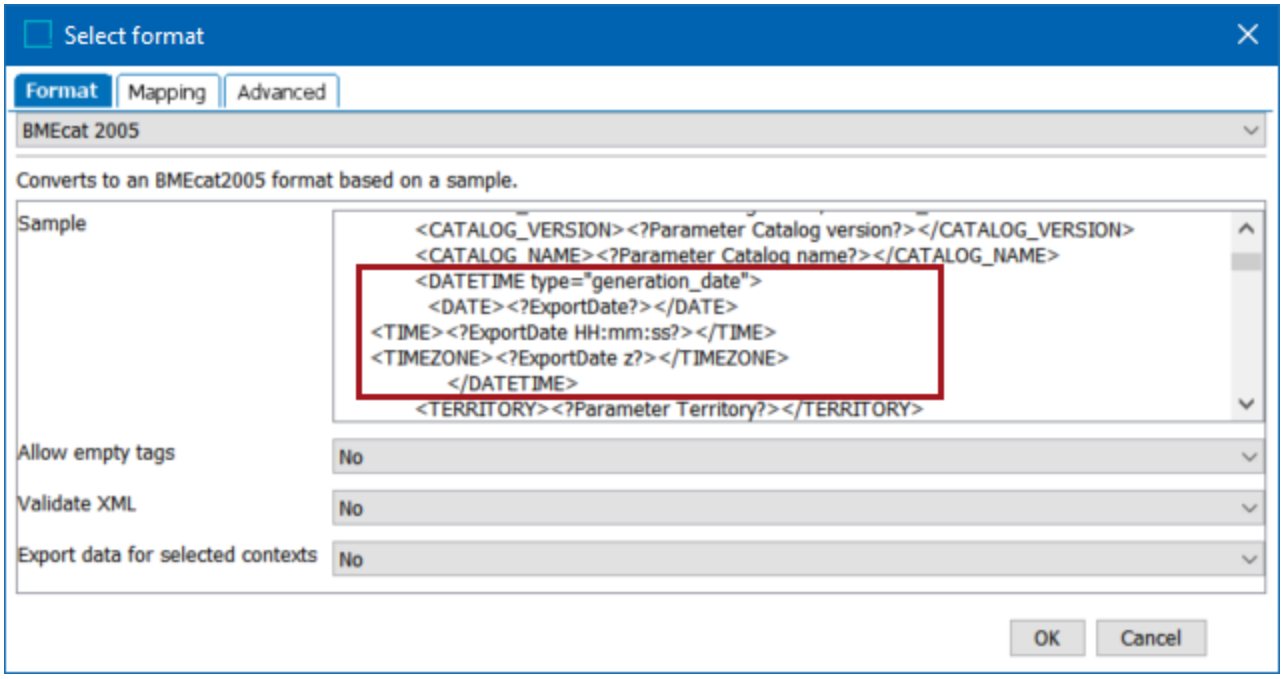
```
<Families>
  <Family ID="8094">
    <Name>7134-24</Name>
    <Children>
      <Child>
        <ID>8082</ID>
        <Name>Single</Name>
      </Child>
      <Child>
        <ID>107807</ID>
        <Name>Case of 24</Name>
      </Child>
    </Children>
  </Family>
</Families>
```

With DoNotEmit

```
<Families>
  <Family ID="8094">
    <Name>7134-24</Name>
    <Children>
      <ID>8082</ID>
      <Name>Single</Name>
      <ID>107807</ID>
      <Name>Case of 24</Name>
    </Children>
  </Family>
</Families>
```

ExportDate Processing Instructions in Generic XML

ExportDate instruction in Generic XML (GXML) automatically generates the current time and date of when an export occurs. The default format for this processing instruction is YYYY-MM-DD, but can be changed based on the specified processing instructions.



In the example above, the highlighted tags would produce the following date and time information:

```
<?xml version="1.0" encoding="utf-8"?>
<BMECAT xmlns="http://www.bmecat.org/bmecat/2005fd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.bmecat.org/bmecat/2005fd bmecat_2005.xsd"
version="2005">
  <HEADER>
    <CATALOG>
      <DATETIME type="generation_date">
        <DATE>2019-09-17</DATE>
        <TIME>11:06:06</TIME>
        <TIMEZONE>EDT</TIMEZONE>
      </DATETIME>
    </CATALOG>
    <SUPPLIER>
      <SUPPLIER_ID type="duns"/>
      <ADDRESS type="supplier"/>
    </SUPPLIER>
  </HEADER>
  <T_NEW_CATALOG/>
</BMECAT>
```

- The <?ExportDate?> processing instruction, located between the <DATE> tags, produces the date in the default format of YYYY-MM-DD.
- The <?ExportDate HH:mm:ss?> processing instruction, located between the <TIME> tags, produces the time of the export, in the format determined by the specific parameter (HH:mm:ss).
- The <?ExportDate z?> processing instruction, located between the <TIMEZONE> tags, produces the time zone of where the export took place, in the format determined by the specific parameter (z).

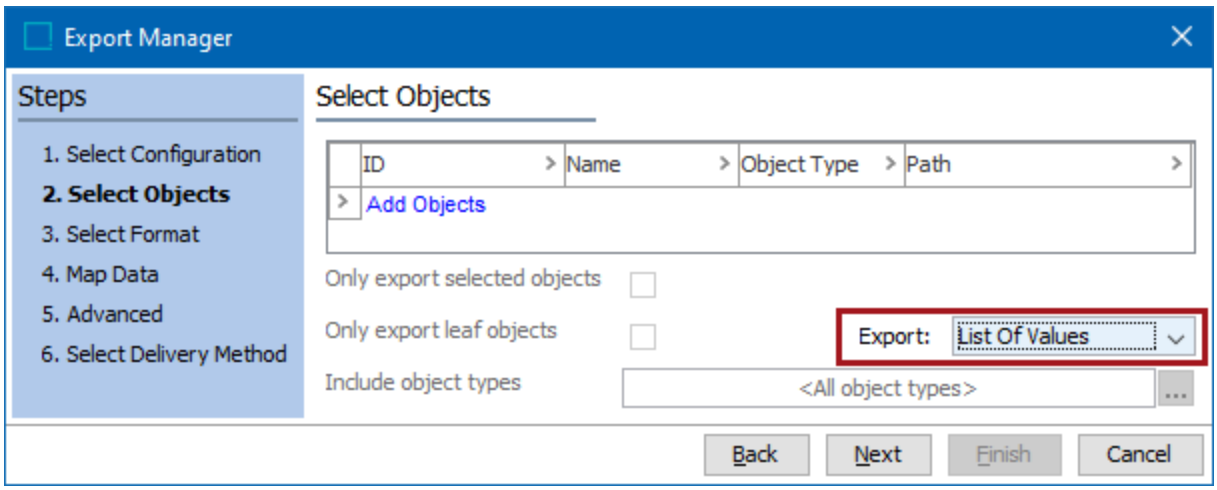
For demonstrative purposes, this example displays three different tag sets (<DATE>, <TIME>, and <TIMEZONE>; however, users can produce the same date / time information with a single tag set like this: <DATE><?ExportDate YYYY-MM-DD HH:mm:ss z?></DATE>.

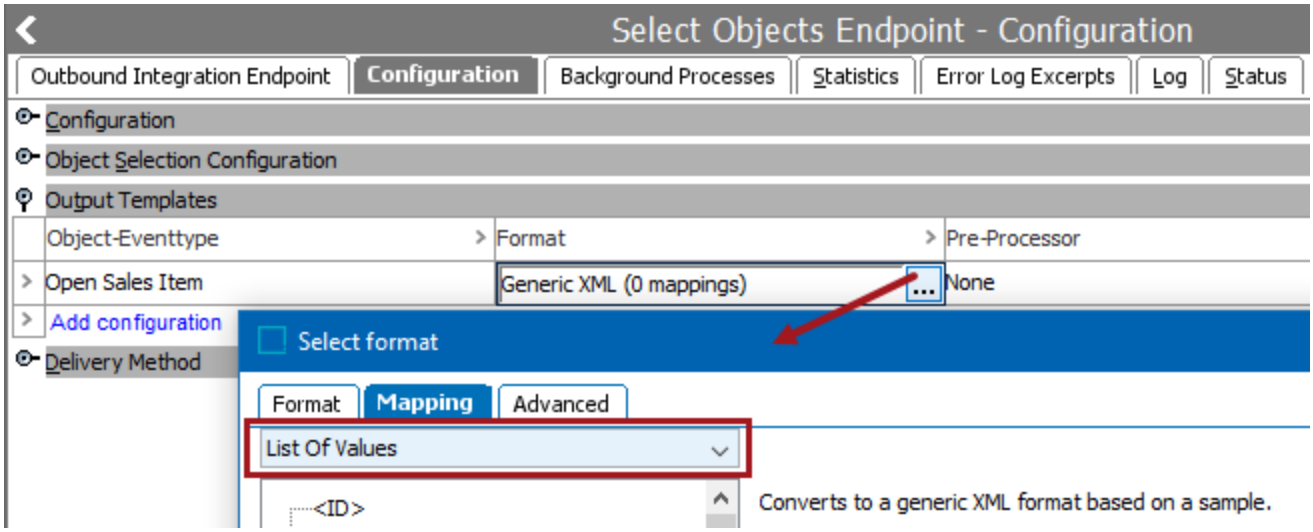
List of Values Object Type in Generic XML

Export tools require that you select objects and provide both a sample template and mapping when exporting Generic XML.

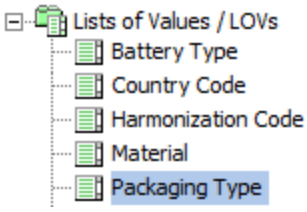
Objects

1. Select **List of Values** as object type to export, based on the export tool being used.





2. Select the LOV to export, based on your export tool.



For Export Manager, refer to the **Export Manager - Select Objects** topic.

For OIEP, refer to the **Object Types** section of the **OIEP - Event-Based - Event Triggering Definitions Tab** topic or the **OIEP - Select Objects - Object Selection Configuration Flipper** topic.

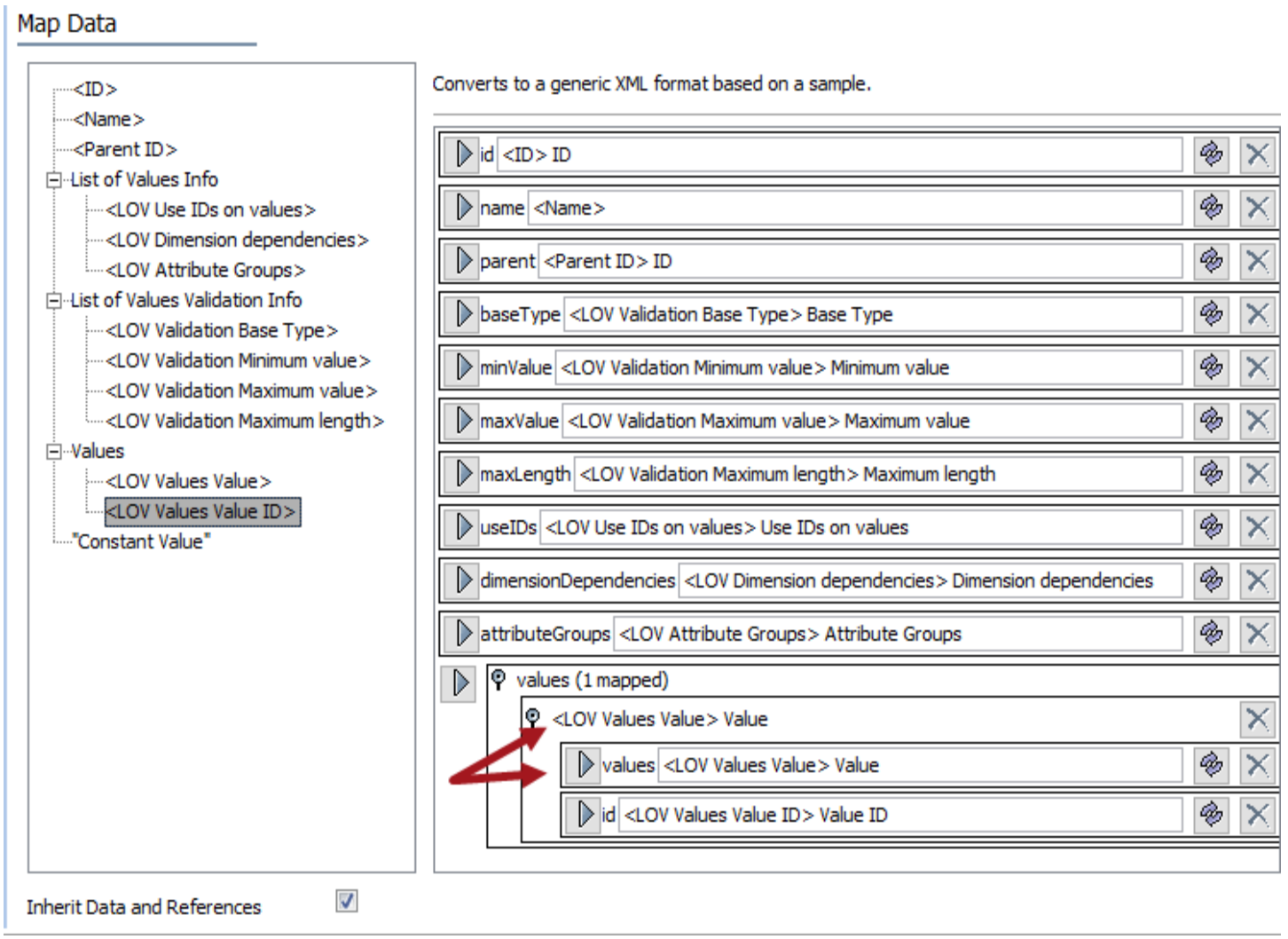
Template

Choose **Generic XML** as the export format and provide the following template text in the Sample field. The template must define all data / fields of the LOV that should be output. All options are included in the following sample template.

```
<root>
   <lov>
      <?Record?>
      <id><?Target?></id>
      <name><?Target?></name>
      <parent><?Target?></parent>
      <baseType><?Target?></baseType>
      <minValue><?Target?></minValue>
      <maxValue><?Target?></maxValue>
      <maxLength><?Target?></maxLength>
      <useIDs><?Target?></useIDs>
      <dimensionDependencies><?Target?></dimensionDependencies>
      <attributeGroups><?Target?></attributeGroups>
      <values id="[?Target id?]"><?MultiTarget?><?Target?></values>
   </lov>
</root>
```

Mapping

Below, the <LOV Values Value> node is mapped twice to allow the Value ID and content to be displayed on the same line.



Results

The output includes information for the LOV selected for export. If the LOV parameter 'Use IDs on Values' is set to Yes, the export will include both the Value and Value ID.

```
<root>
  <lov>
    <id>PackagingTypeLOV</id>
    <name>Packaging Type</name>
    <parent>List Of Values group root</parent>
    <baseType>text</baseType>
    <minValue></minValue>
    <maxValue></maxValue>
    <maxLength>100</maxLength>
    <useIDs>true</useIDs>
    <dimensionDependencies></dimensionDependencies>
    <attributeGroups></attributeGroups>
    <values id="PX">Pallet</values>
    <values id="SW">Shrinkwrapped</values>
    <values id="SPL">Spool</values>
    <values id="BP">Banded Package</values>
    <values id="TU">Tube</values>
    <values id="CG">Card</values>
    <values id="BX">Box</values>
    <values id="LSE">Loose</values>
    <values id="CT">Carton</values>
    <values id="CY">Cylinder</values>
    <values id="BG">Bag</values>
    <values id="CS">Case</values>
  </lov>
</root>
```

Multi-Valued Objects in Generic XML

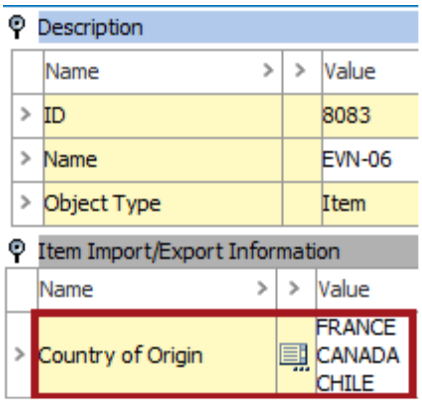
There are two Generic XML options for exporting fields that allow multiple values.

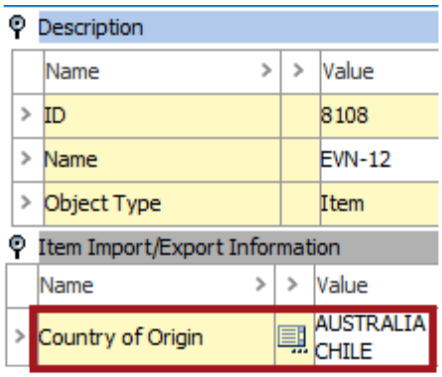
- Repeated Target exports a delimited list of the values in a single tag
- MultiTarget exports each value individually, in separate tags

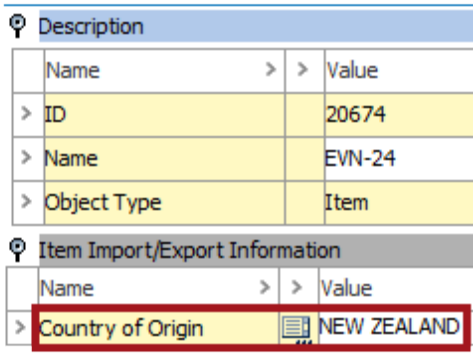
MultiTarget is usually the preferred option.

Objects

The Country of Origin multi valued attribute is exported for each of these selected products:







Template

Repeated Target

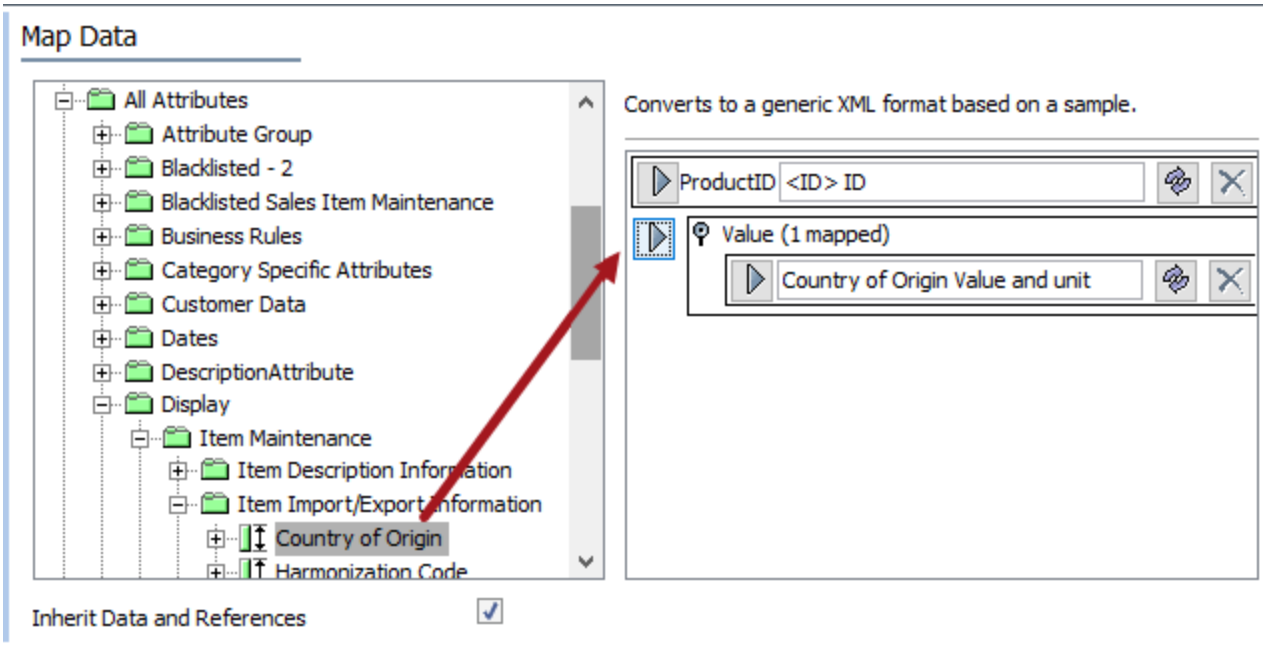
```
    <Products>
       <Product>
          <?Record?>
          <ProductID><?Target?></ProductID>
          <Value><?Target?></Value>
          <Value><?Target?></Value>
       </Product>
    </Products>
```

MultiTarget

```
<Products>
   <Product>
      <?Record?>
      <ProductID><?Target?></ProductID>
      <Value><?MultiTarget?></Value>

   </Product>
</Products>
```

Mapping

The mapping steps are the same regardless of the Multi Valued option used.



Results

Repeated Target

```
<Products>
  <Product>
    <ProductID>8083</ProductID>
    <Value>FRANCE;CANADA;CHILE</Value>
  </Product>
  <Product>
    <ProductID>8108</ProductID>
    <Value>AUSTRALIA;CHILE</Value>
  </Product>
  <Product>
    <ProductID>20674</ProductID>
    <Value>NEW ZEALAND</Value>
  </Product>
</Products>
```

MultiTarget

```
<Products>
  <Product>
    <ProductID>8083</ProductID>
    <Value>FRANCE</Value>
    <Value>CANADA</Value>
    <Value>CHILE</Value>
  </Product>
  <Product>
    <ProductID>8108</ProductID>
    <Value>AUSTRALIA</Value>
    <Value>CHILE</Value>
  </Product>
  <Product>
    <ProductID>20674</ProductID>
    <Value>NEW ZEALAND</Value>
  </Product>
</Products>
```

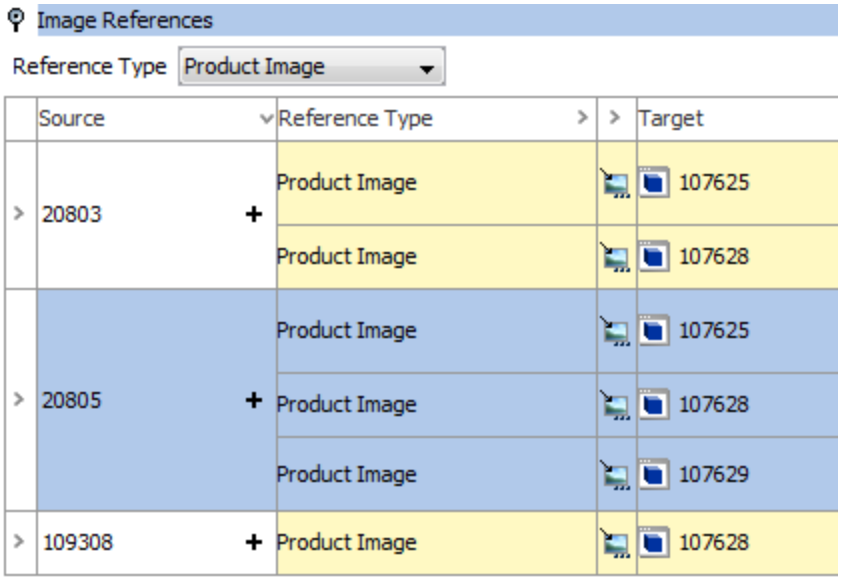
Multi-Cardinality Tags in Generic XML

Single cardinality tags are used when the data produced by the associated data source is considered a 'single piece of information' and will be output inside a single occurrence of the enclosing XML element. When more than one piece of data exists for the same data source (for example, multiple images on the same product), the output requires multiple-cardinality tags.

The following multi-cardinality mapping options are discussed below: Repeated Target, MultiTarget, and Composite Mapping Targets.

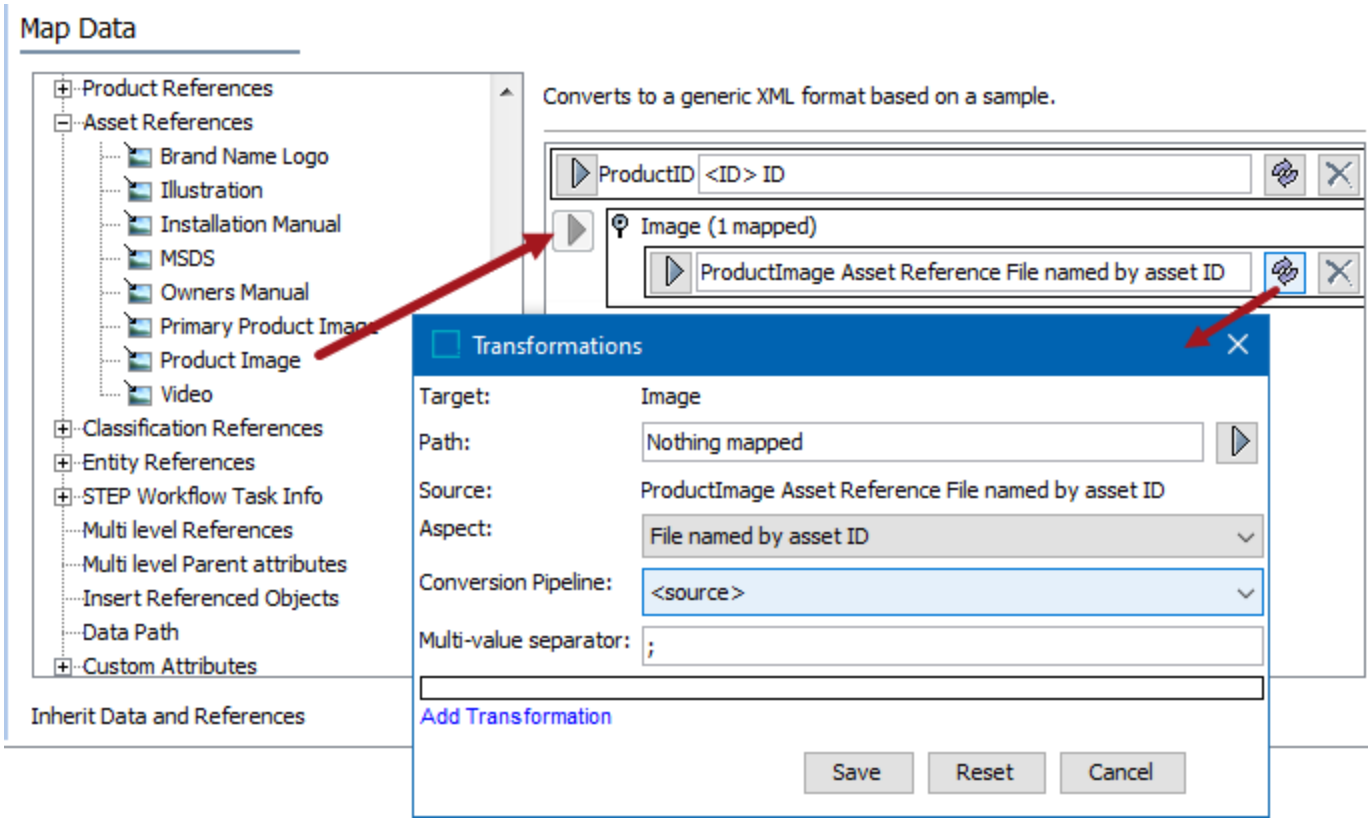
Objects

The following objects are used for each of the examples in this section.



Mapping

The following mapping is used for each of the examples in this section.



Repeated Targets

Repeating the same mapping target allows multiple instances of the same element to be exported. In the following sample template, note that the image target tag appears two times as <Image><?Target?></Image>.

When using Repeated Targets, mapping target declarations must be repeated immediately and the tag name and mapping target ID must be identical. This technique **cannot** be used for mapping targets declared below composite mapping targets.

Template

```
<Products>
   <Product>
      <?Record?>
      <ProductID><?Target?></ProductID>
      <Image><?Target?></Image>
      <Image><?Target?></Image>
  </Product>
</Products>
```

Results

The results include all instances of the images exported, even when more than two are referenced by the product. This is the same result as with MultiTarget if the mapping targets are not composite.

```
<Products>
  <Product>
    <ProductID>20803</ProductID>
    <Image>107628.png</Image>
    <Image>107625.tif</Image>
  </Product>
  <Product>
    <ProductID>20805</ProductID>
    <Image>107628.png</Image>
    <Image>107629.jpg</Image>
    <Image>107625.tif</Image>
  </Product>
  <Product>
    <ProductID>109308</ProductID>
    <Image>107628.png</Image>
  </Product>
</Products>
```

MultiTarget

The <?MultiTarget [Optional Identifier]?> processing instruction allows for multiple objects, multiple attributes, or multiple values for a single attribute to be represented in the same way.

In the following sample template, the image multitarget tag appears only one time as <Image><?MultiTarget?></Image>. However, when more than one image exists in the data, all of them are exported, as shown in the results section.

This technique **can** be used for mapping targets declared below composite mapping targets.

Template

```
1  <Products>
2     <Product>
3        <?Record?>
4        <ProductID><?Target?></ProductID>
5        <Image><?MultiTarget?></Image>
6     </Product>
7  </Products>
```

Results

The results include all instances of the images exported, even when more than two are referenced by the product. This is the same result as with the Repeated Target if the mapping targets are not composite.

```
<Products>
  <Product>
    <ProductID>20803</ProductID>
    <Image>107628.png</Image>
    <Image>107625.tif</Image>
  </Product>
  <Product>
    <ProductID>20805</ProductID>
    <Image>107628.png</Image>
    <Image>107629.jpg</Image>
    <Image>107625.tif</Image>
  </Product>
  <Product>
    <ProductID>109308</ProductID>
    <Image>107628.png</Image>
  </Product>
</Products>
```

Composite Mapping Targets

The previous multi-cardinality examples show exporting simple values, also called flat values. Often there is a need for repeating a more complex structure, which is possible in Generic XML with composite mapping targets.

For example, when exporting product data where each product has a number of asset references and the output needs to contain a small XML structure (substructure) for each asset. Each substructure will contain the required values of the asset, which in some cases, will include the same value as other substructures.

Mapping a data source to a composite mapping target will cause the composite substructure to be evaluated and written to the output once for each value produced by the data source.

Template

```
1  <products>
2     <product>
3        <?Record?>
4        <ProductID><?Target?></ProductID>
5        <image>
6        <?MultiTarget?>
7           <id><?Target?></id>
```

```
            <filename><?Target?></filename>
            <description><?Target?></description>
        </image>
    </product>
</products>
```

Results

- The image tag is a multi-cardinality mapping target
- The additional mapping targets **inside** the image tag make it a **composite** mapping target
- The remaining mapping targets (id, filename and description) become ordinary mapping targets placed **below** the composite mapping target

```
<products>
  <product>
    <ProductID>20803</ProductID>
    <image>
      <id>107628</id>
      <filename>107628.png</filename>
      <description>20801</description>
    </image>
    <image>
      <id>107625</id>
      <filename>107625.tif</filename>
      <description>20805</description>
    </image>
  </product>
  <product>
    <ProductID>20805</ProductID>
    <image>
      <id>107628</id>
      <filename>107628.png</filename>
      <description>20801</description>
    </image>
    <image>
      <id>107629</id>
      <filename>107629.jpg</filename>
      <description>20803</description>
    </image>
    <image>
      <id>107625</id>
      <filename>107625.tif</filename>
      <description>20805</description>
    </image>
  </product>
  <product>
    <ProductID>109308</ProductID>
    <image>
      <id>107628</id>
      <filename>107628.png</filename>
      <description>20801</description>
    </image>
  </product>
</products>
```

Nestable Data Paths in Generic XML

The Nestable Data Path feature is only available during export of nested repeated structures using Generic XML.

- The **Nestable Data Paths in Generic XML Example** topic illustrates the concepts and capabilities provided by this feature.
- This **Nestable Data Paths in Generic XML and Performance** topic provides a step-by-step example to demonstrate how data paths can affect Generic XML performance.

Data Path Concept

The mapping step of the export wizard is used to bind Data Sources to Mapping Targets. Data sources evaluate and produce values for the export result relative to the object selected for export (for example product, entity, classification). When bound correctly, an ID data source will produce the ID of the object currently being processed for export, while an Attribute data source will produce the value of some specific attribute on the object being processed.

The Data Path data source makes it possible to change the object that other data sources, nested below the Data Path source, use when they are asked to evaluate. Instead of evaluating against the object being exported, they will be evaluated relative to some other object, or reference, reached by following a path from the exported object. For example, the following mapping target structure is produced by the Nestable Data Path Example:

- Product-ID
- Product-Name
- Accessory (composite, multiple-cardinality)
  - Accessory-ID
  - Accessory-Description
- Image (composite, multiple-cardinality)
  - Image-ID
  - Image-Name

Mapping the existing ID data source to the Accessory-ID mapping target would result in exporting the ID value from the product.

Instead, mapping a Data Path data source to the Accessory composite target, configured with the path (Reference Accessory, Reference Target), two things will happen:

- The Accessory sub-tree will be evaluated and output once for each reference of type Accessory found from the object being processed for export.
- Data sources mapped in the sub-tree will be evaluated relative to the referenced products(s) found by following the data path.

The ID source mapped to Accessory-ID mapping target will result in the IDs of the accessory products, and an Attribute source mapped to Accessory-Description will produce an attribute value from the accessory object.

Nesting Concept

A Data Path can be inserted below another Data Path, creating a nesting effect.

By mapping a Data Path data source to the composite Image target, configured with the path (Reference Image, Reference Target), two things will happen:

- The Image sub-tree will be evaluated and output once for each reference of type Image found from the accessory product found by the parent Data Path source.
- Data sources mapped in the sub-tree will be evaluated relative to the referenced asset(s) found by following the Data Path.

An ID source mapped to Image-ID will produce the IDs of the asset associated with the accessory product, and a Name source mapped to Image-Name will produce the name of the asset.

Valid Path Step Types

Configuring the path of a Data Path source is made up of a number of path steps. Each step describes how to reach a new set of references or objects by moving from the references / objects reached by the previous step.

The first step in a top-level Data Path source is taken relative to the object being processed for export. Following steps, either in the same path or the path of a nested Data Source, are taken relative to the set of object or references produced by the preceding step.

The Data Path data source supports different types of path steps. Starting from the objects selected for export, it is valid to configure Data Path sources with initial steps of type Parent, Reference and Override Sub Product. Following steps, including steps configured for nested Data Path sources, are only valid according to the following rules:

Step	Reach	Valid followed by
Parent	Parent node from node	Parent
Reference	Reference of a specific type from node	Reference Target
Reference Target	Target of a reference from a reference	Reference
Override Sub Product	Adopted children of a product-override node	Reference

Note: Although the mapping step of the Data Export wizard does not prevent the user from configuring invalid paths, paths that are not valid will result in an undefined exported.

Data Sources Available for Data Paths

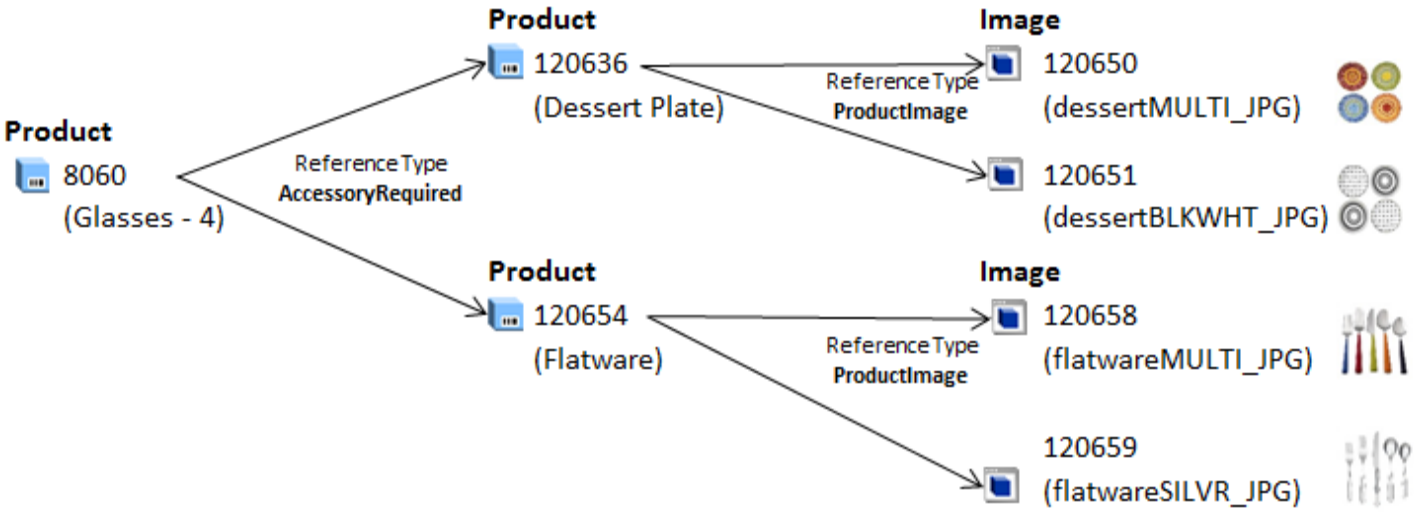
The data sources that are to be mapped below a Data Path source require special support for evaluating relative to the objects / references identified by the paths. The following standard data sources support such nesting:

- ID
- Name
- Parent ID
- Object-Type
- Constant Value
- Attribute
- Attribute Group

Note: Data sources without special support evaluate relative to the object being processed for export even when placed below a Data Path source.

Nestable Data Paths in Generic XML Example

In the following example, the data set contains a product which has multiple references of type AccessoryRequired, to accessory products. Each accessory product has multiple references of type ProductImage, to assets.



Using Nested Data Paths when mapping will produce an export that includes all of the IDs shown above, in the following levels:

- Level 1 Product tag including the ID and Name of the product selected for export
- Level 2 Accessory tag including the ID, Name and Description for each product with an AccessoryRequired reference type
- Level 3 Image tag including the ID and Name for each image with a ProductImage reference type

Template

Choose **Generic XML** for **Format** and provide the following template text in the Sample field.

```
<Products>
   <Product>
   <?Product?>
   <ID><?Target?></ID>
   <Name><?Target?></Name>
      <Accessory>
      <?MultiTarget?>
```

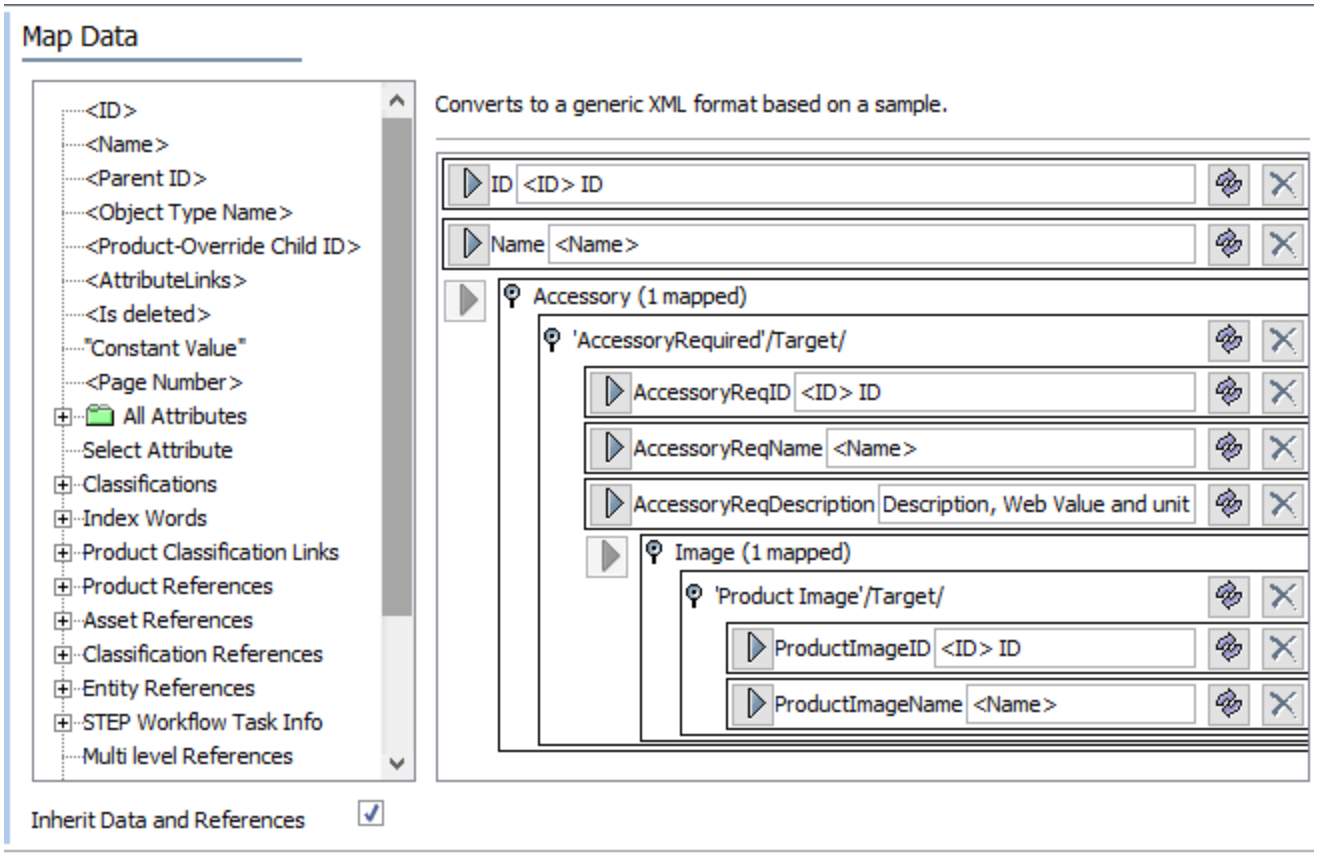
```
            <ID><?Target AccessoryReqID?></ID>
            <Name><?Target AccessoryReqName?></Name>
            <Description><?Target AccessoryReqDescription?></Description>
               <Image>
               <?MultiTarget?>
               <ID><?Target ProductImageID?></ID>
               <Name><?Target ProductImageName?></Name>
               </Image>
            </Accessory>
         </Product>
      </Products>
```

Data Path Configuration

The mapping target structure resulting from the above template is:

Target/Structure	Data Source	Configuration
ID	ID	
Name	Name	
Accessory	Data Path	Data Path=Reference 'AccessoryRequired', Reference Target (composite, multiple-cardinality)
AccessoryReqID	ID	
AccessoryReqName	Name	
AccessoryReqDescription	Attribute	Select Attribute='Description, Web' Value and unit
Image	Data Path	Data Path=Reference 'ProductImage', Reference Target (composite, multiple-cardinality)
ProductImageID	ID	
ProductImageName	Name	

Mapping



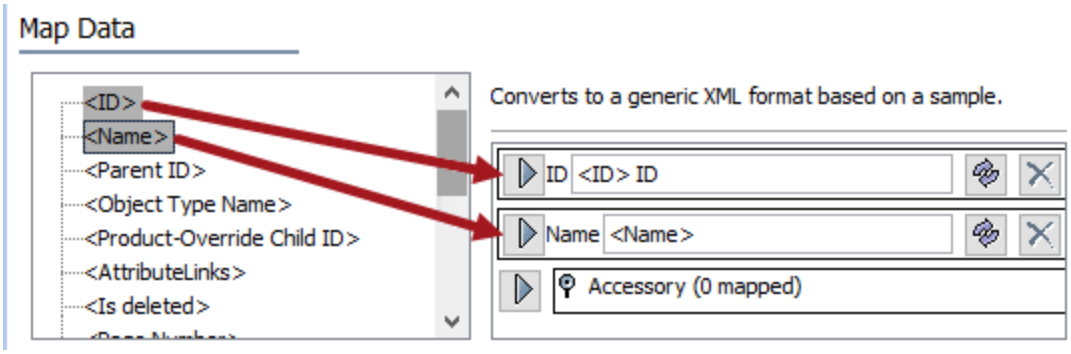
For step-by-step mapping instructions, refer to the **Mapping for Nestable Data Paths in Generic XML Example** documentation.

Results

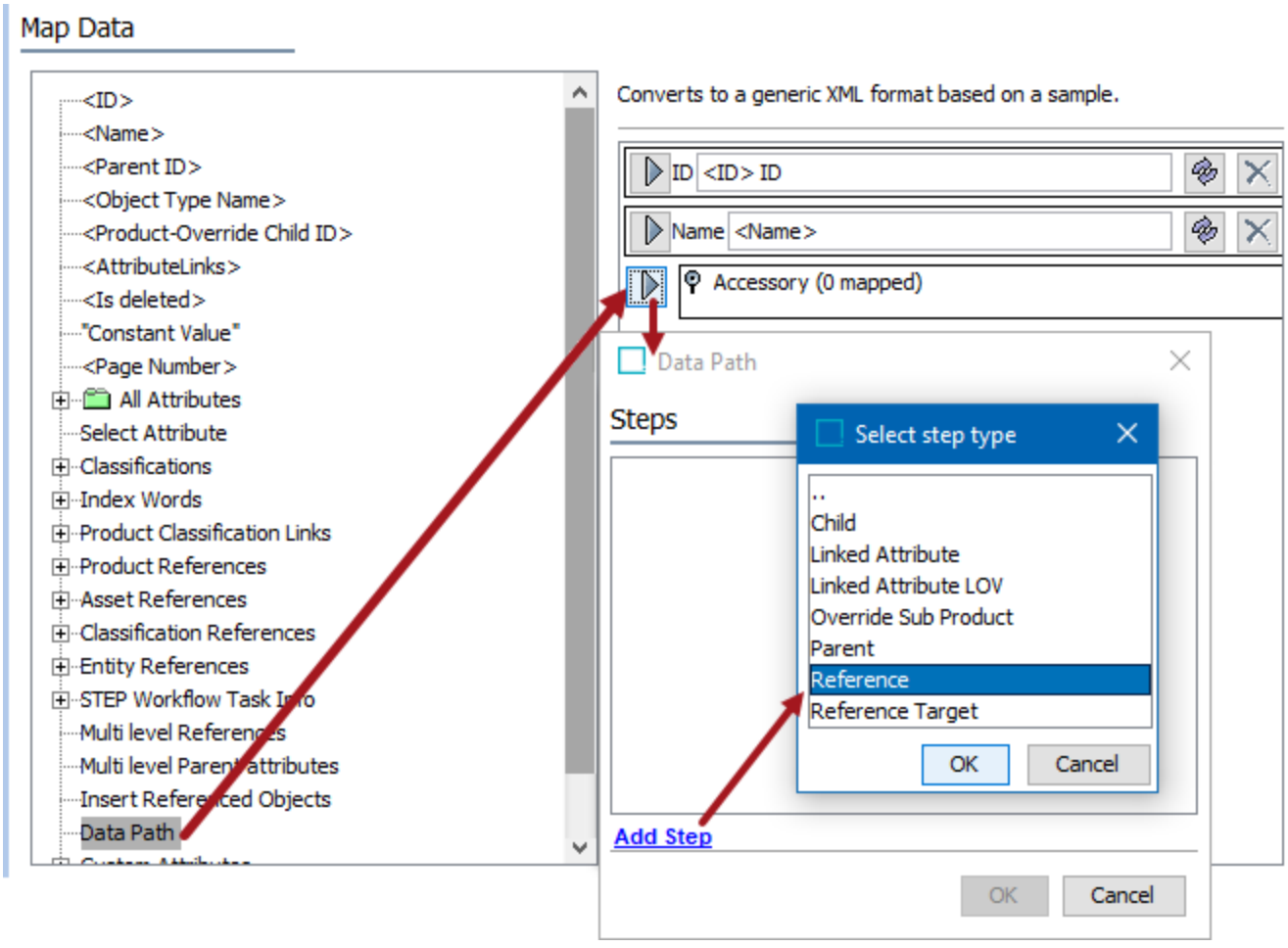
```
<Products>
  <Product>
    <ID>8060</ID>
    <Name>Glasses - 4</Name>
    <Accessory>
      <ID>120636</ID>
      <Name>Dessert Plate</Name>
      <Description>Set of 4 festive dessert plates.</Description>
      <Image>
        <ID>120650</ID>
        <Name>dessertMULTI_JPG</Name>
      </Image>
      <Image>
        <ID>120651</ID>
        <Name>dessertBLKWHT_JPG</Name>
      </Image>
    </Accessory>
    <Accessory>
      <ID>120654</ID>
      <Name>Flatware</Name>
      <Description>Five-piece, single place setting.</Description>
      <Image>
        <ID>120658</ID>
        <Name>flatwareMULTI_JPG</Name>
      </Image>
      <Image>
        <ID>120659</ID>
        <Name>flatwareSILVR_JPG</Name>
      </Image>
    </Accessory>
  </Product>
</Products>
```

Mapping for Nestable Data Paths in Generic XML Example

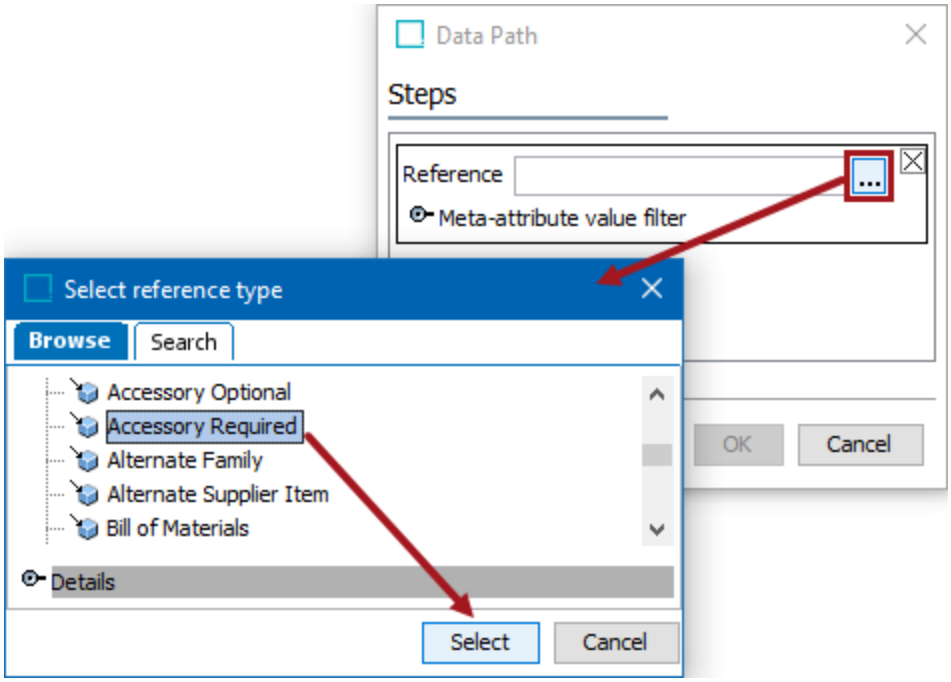
1. Map **ID** and **Name** for Level 1.



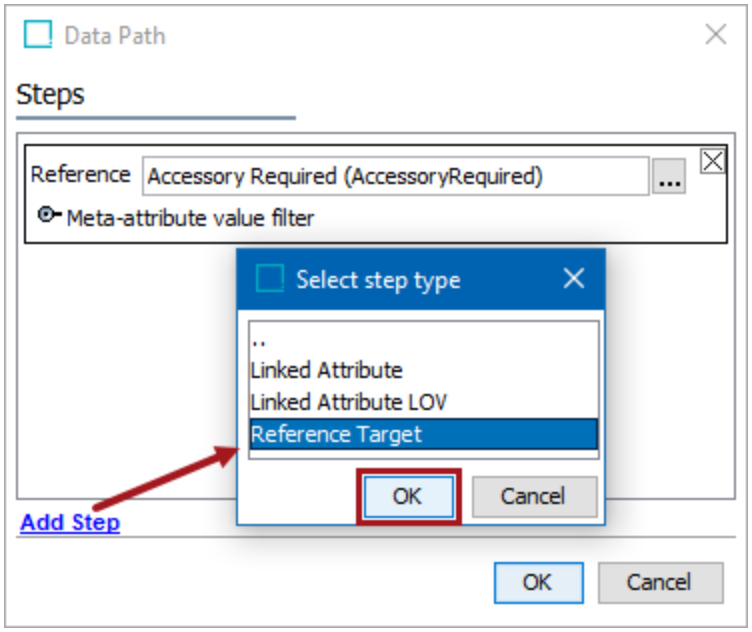
2. Map **Data Path** to Accessory, click the **Add Step** link, select the **Reference** step type, and click **OK**.



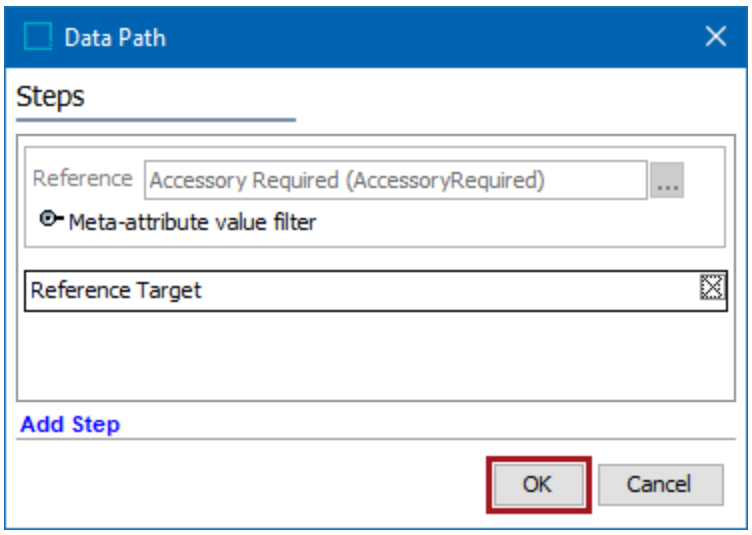
3. On the Data Path dialog, click the ellipsis button (...), select AccessoryRequired reference type, and click the **Select** button.



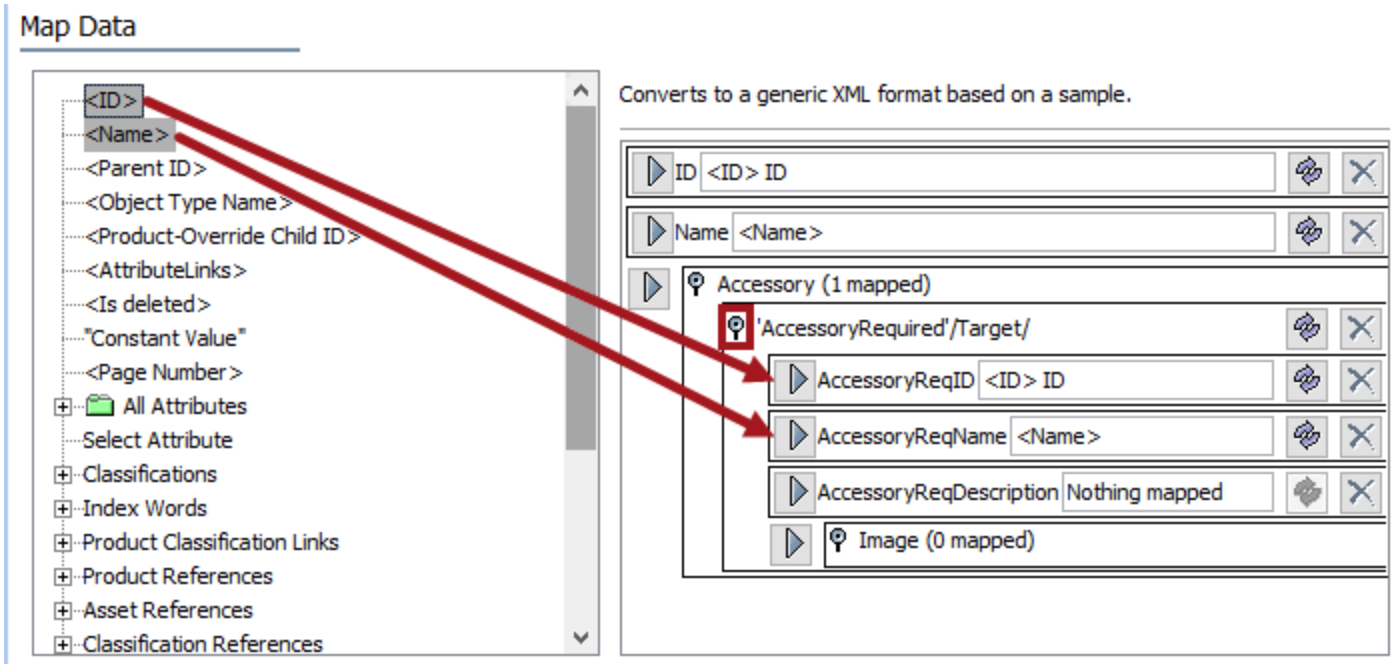
4. Click the **Add Step** link, select **Reference Target** for Accessory and click **OK**.



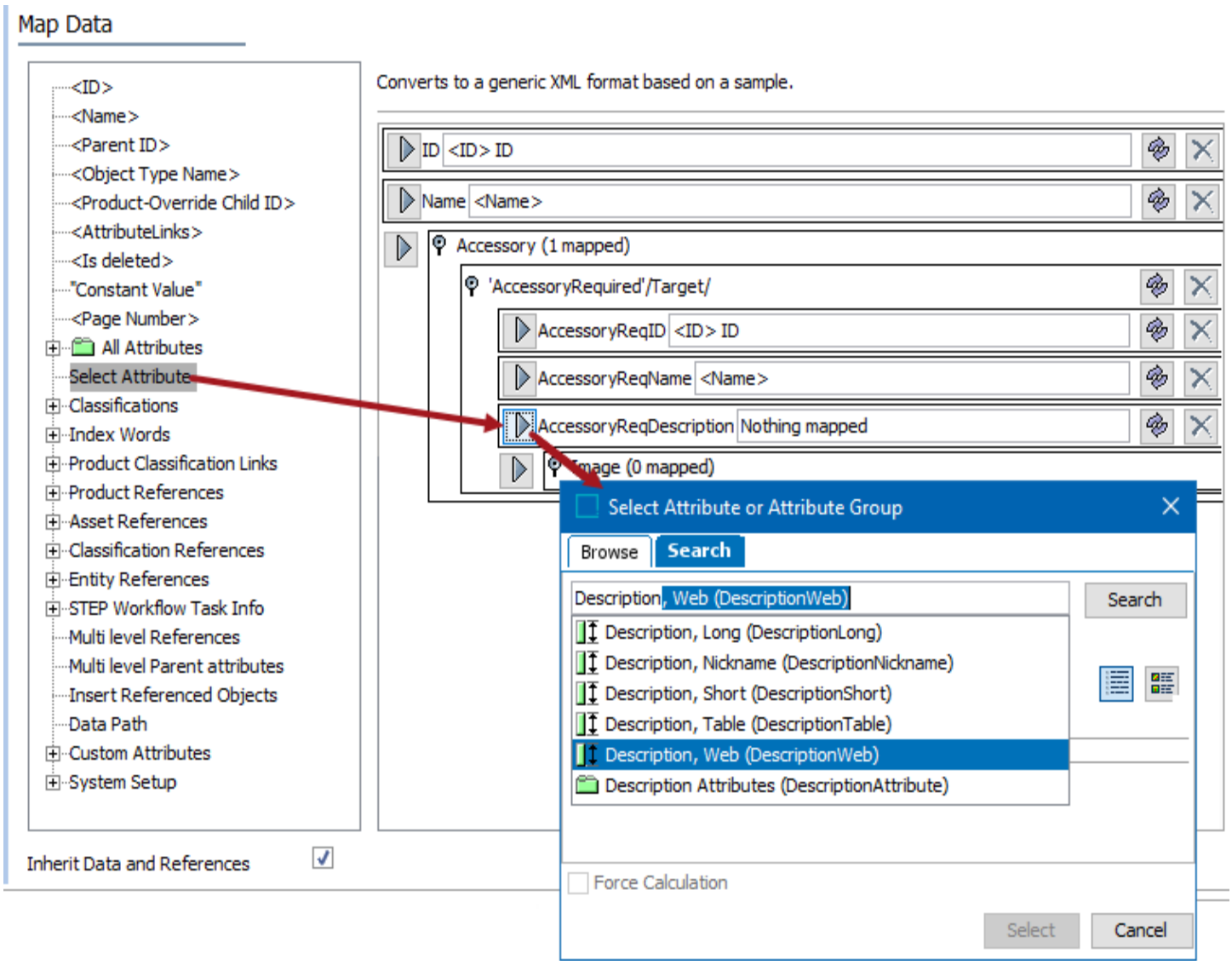
5. Click **OK** to confirm the Reference and Reference Target Data Path fields for the Accessory.



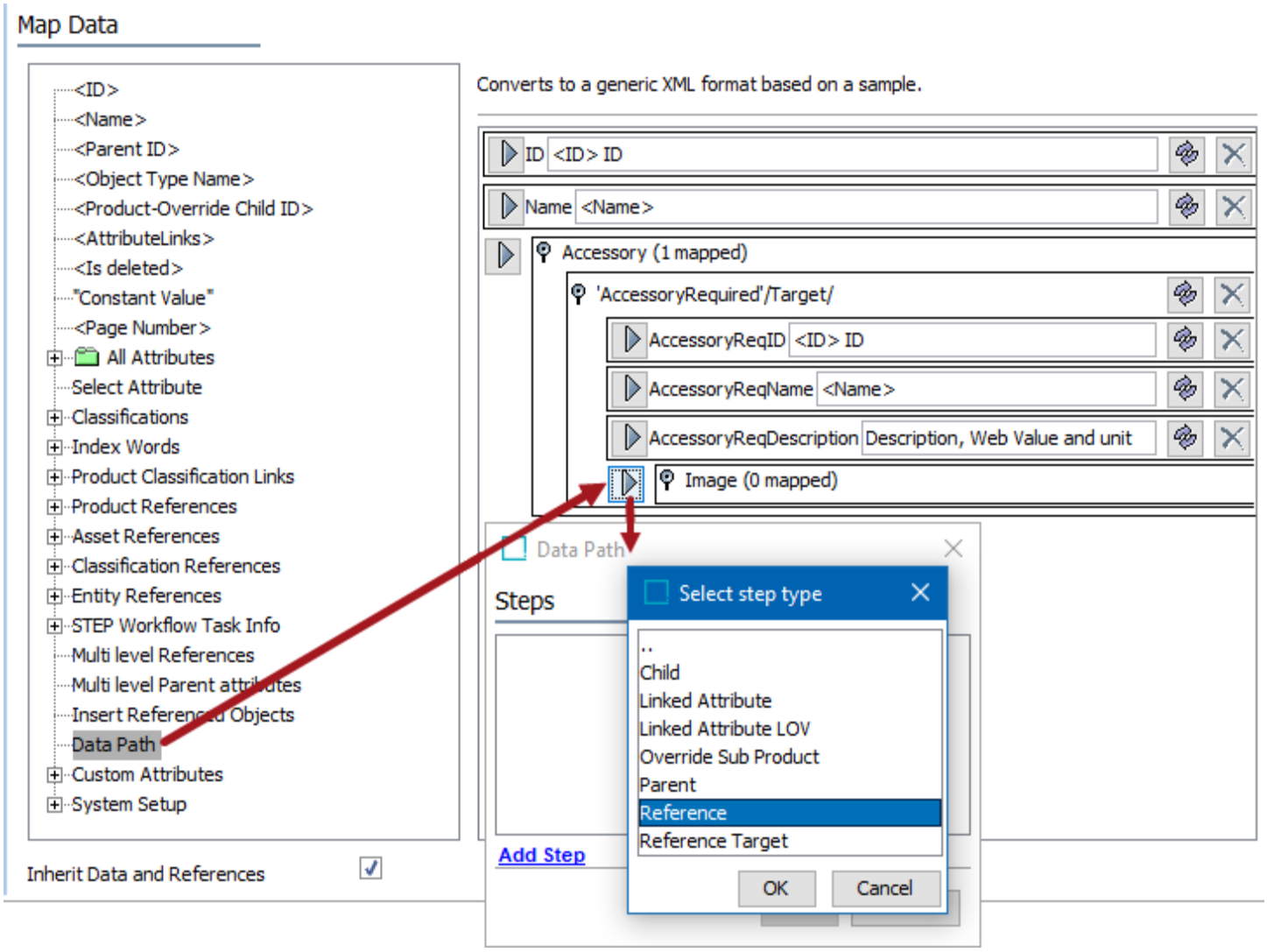
6. Open the 'AccessoryRequired'/Target/ flipper to display the nested mapping options. Map **ID** and **Name** for the Level 2 products being exported.



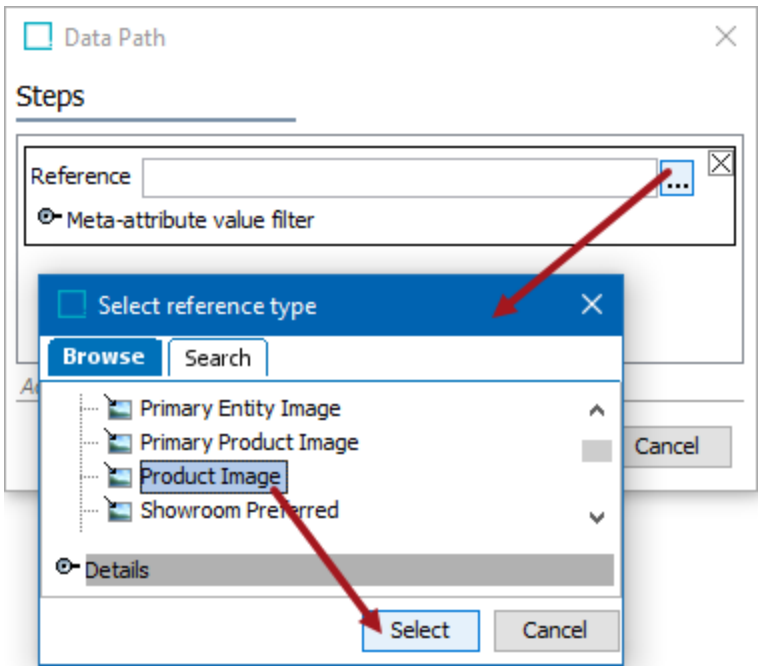
7. Use **Select Attribute** to map 'Description, Web' for Level 2 (products) and click the **Select** button.



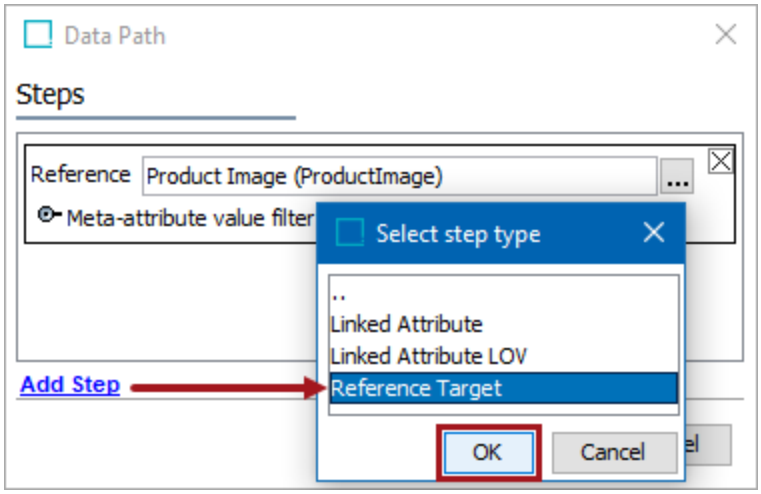
8. Map **Data Path** to Image, click the **Add Step** link, select the **Reference** step type, and click **OK**.



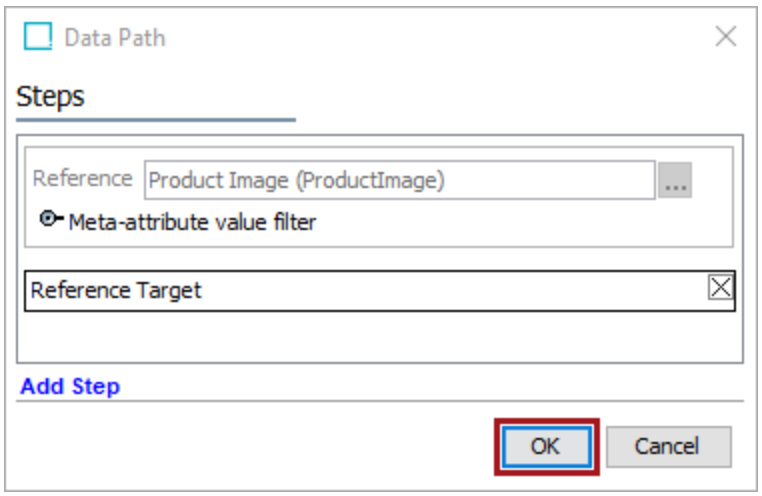
9. On the Data Path dialog, click the ellipsis button (...), select ProductImage reference type, and click the **Select** button.



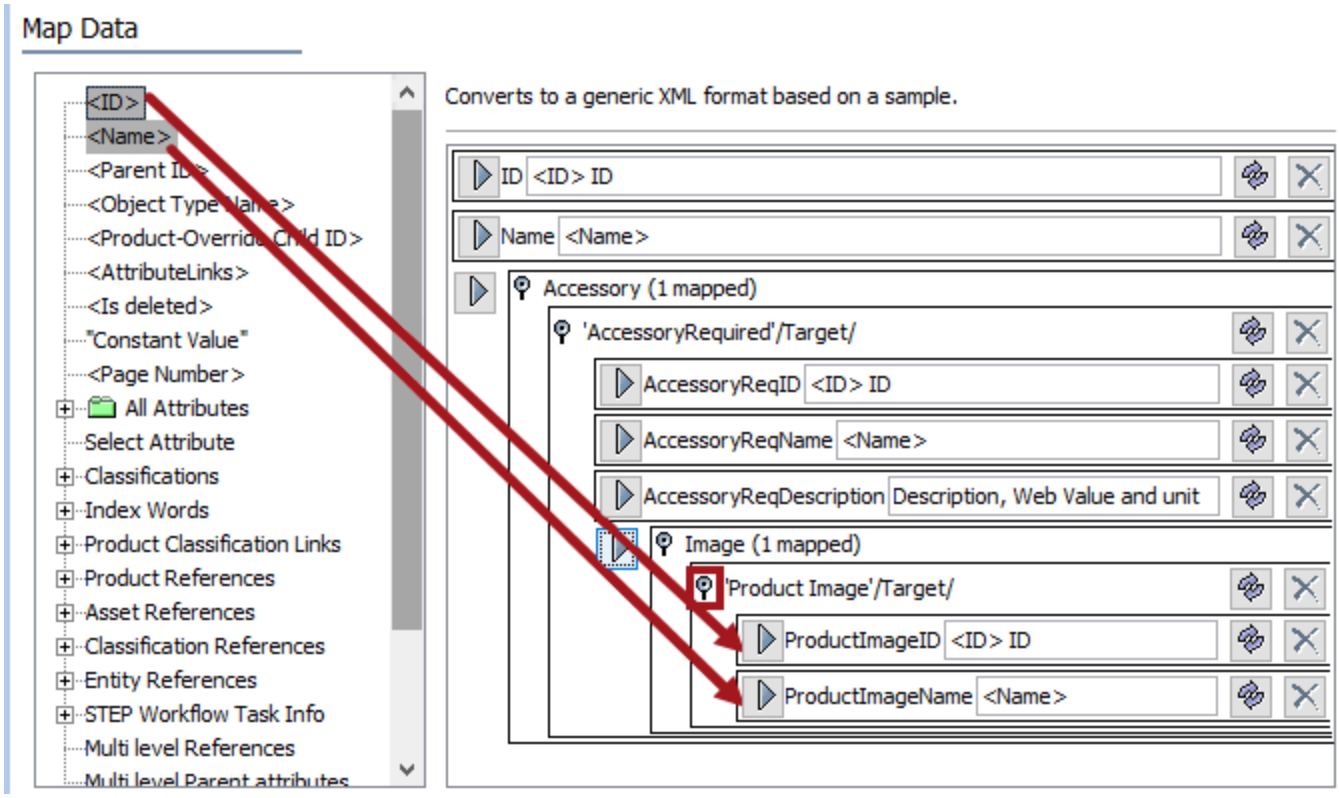
10. Click the **Add Step** link, select **Reference Target** for Image and click **OK**.



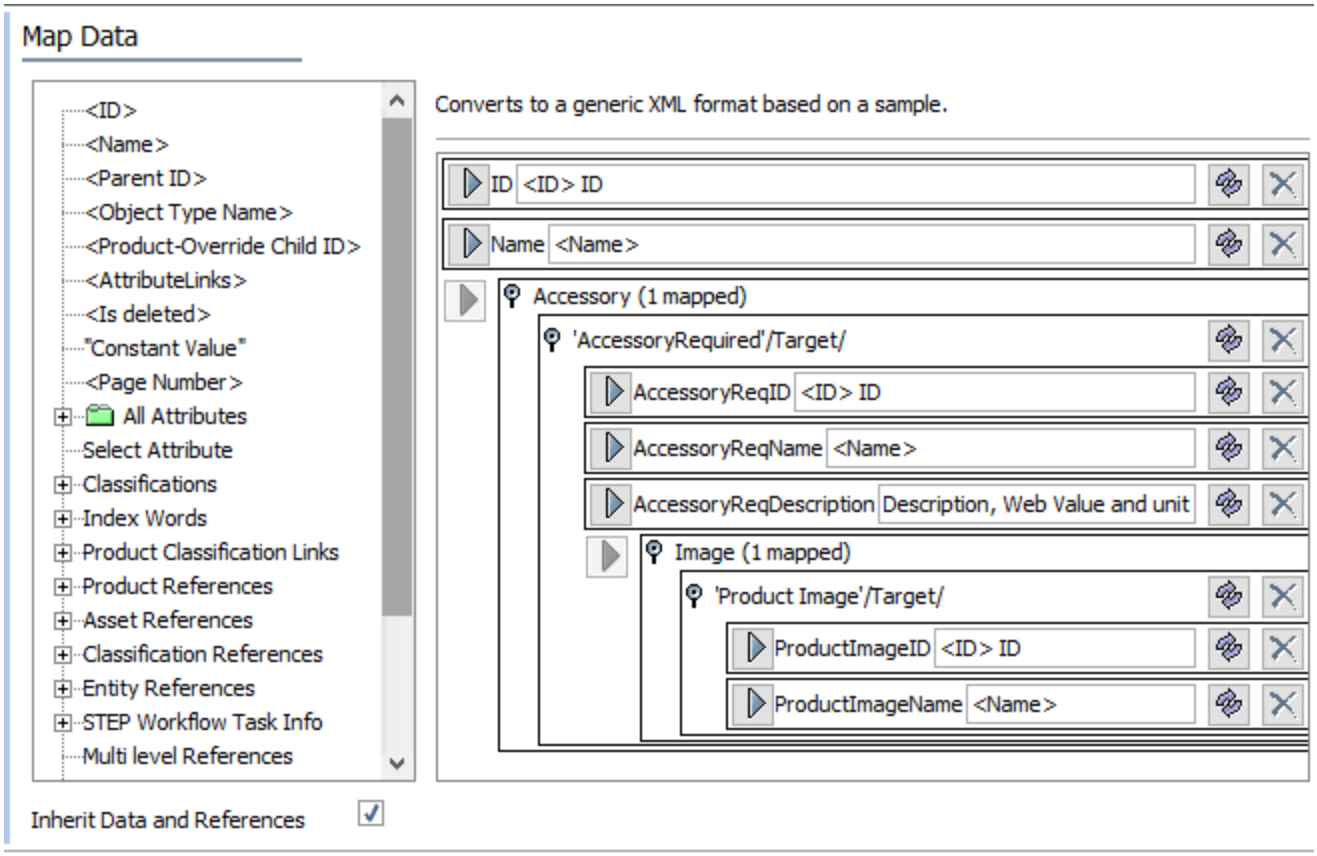
11. Click **OK** to confirm the Reference and Reference Target for the Image.



12. Open the 'ProductImage'/Target/ flipper to display the nested mapping options. Map **ID** and **Name** for Level 3 (images).



All mapping is complete.


Map Data
<ID>
<Name>
<Parent ID>
<Object Type Name>
<Product-Override Child ID>
<AttributeLinks>
<Is deleted>
"Constant Value"
<Page Number>
All Attributes
Select Attribute
Classifications
Index Words
Product Classification Links
Product References
Asset References
Classification References
Entity References
STEP Workflow Task Info
Multi level References
Converts to a generic XML format based on a sample.
ID <ID> ID
Name <Name>
Accessory (1 mapped)
'AccessoryRequired'/Target/
AccessoryReqID <ID> ID
AccessoryReqName <Name>
AccessoryReqDescription Description, Web Value and unit
Image (1 mapped)
'Product Image'/Target/
ProductImageID <ID> ID
ProductImageName <Name>
Inherit Data and References

Nestable Data Paths in Generic XML and Performance

This online help step-by-step example demonstrates how data paths can affect Generic XML export performance. Data exported as Generic XML first uses the STEPXML format (in an intermediate process) to handle the data before converting it to the final Generic XML format.

Using the data paths option adds additional STEPXML intermediate processing. This additional intermediate processing can result in reduced performance based on the data being processed. The biggest performance impacts come when processing overridden products, large numbers of children under overridden products, and multiple data path mappings in the same export.

Intermediate STEPXML processing and output is not visible to the user in STEP but is included in this online help topic to illustrate the reduced performance when using the data path mapping option.

Refer to the online version of this topic to view the details and full XML code.

RemoveIfEmpty Processing Instructions in Generic XML

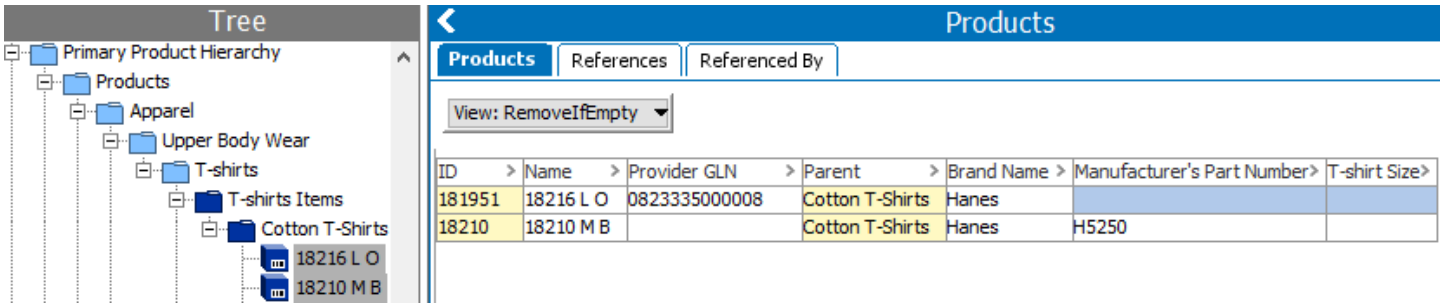
In an outbound Generic XML template, the Generic XML instruction **RemoveIfEmpty** makes it possible to remove the open and close tags surrounding a value if there is no value for any **one** of the attributes in the section.

The **RemoveIfEmpty recursive** instruction allows you to remove an entire section of the output. This means that if a section has no values for any of the attributes, the section is eliminated from the file. The 'RemoveIfEmpty recursive' tag is useful when a whole section should be excluded when no values are found. It should not be used if the section includes only mandatory attributes since the output would be incomplete.

If the open and close tags of an individual line should remain even when no value is supplied, use the 'Allow empty tags' option, described in the **Allow Empty Tags Parameter in Generic XML** section.

Objects

The objects selected for export includes two products, one without either of the optional values (Manufacturer's part Number and T-shirt Size) and the other with only one of the optional attributes.



Template

Choose **Generic XML** format and provide the following template text in the Sample field.

With RemoveIfEmpty

```
<Products>
   <Product>
   <?Record?>
   <ID><?Target?></ID>
   <NAME><?Target?></NAME>
      <mandatoryAttributes>
```

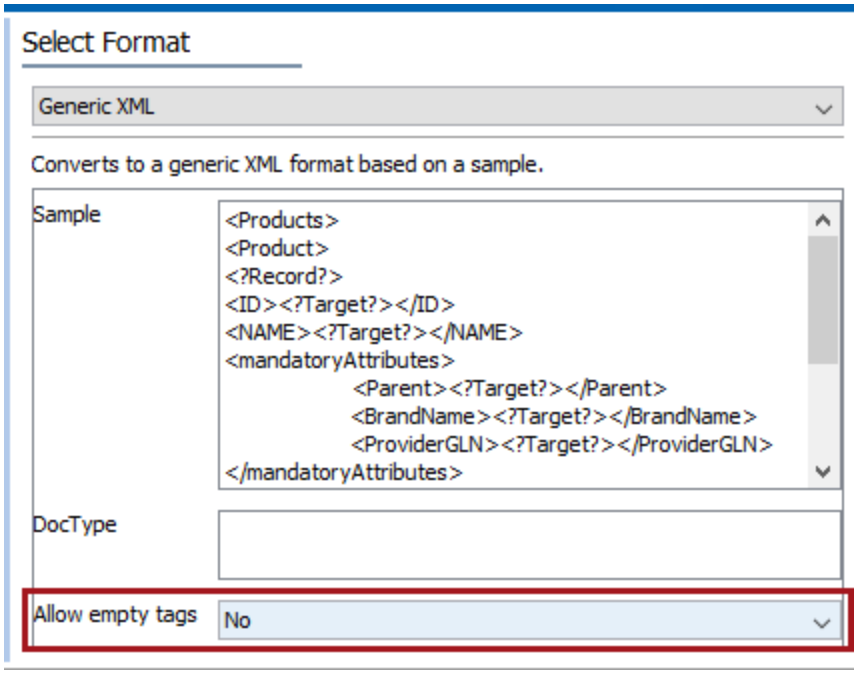
```
            <Parent><?Target?></Parent>
            <BrandName><?Target?></BrandName>
            <ProviderGLN><?Target?></ProviderGLN>
         </mandatoryAttributes>
         <optionalAttributes>
            <?RemoveIfEmpty?>
            <PartNo><?Target?></PartNo>
            <Size><?Target?></Size>
         </optionalAttributes>
      </Product>
   </Products>
```

With 'RemoveIfEmpty recursive'

```
   <Products>
      <Product>
      <?Record?>
      <ID><?Target?></ID>
      <NAME><?Target?></NAME>
         <mandatoryAttributes>
            <Parent><?Target?></Parent>
            <BrandName><?Target?></BrandName>
            <ProviderGLN><?Target?></ProviderGLN>
         </mandatoryAttributes>
         <optionalAttributes>
            <?RemoveIfEmpty recursive?>
            <PartNo><?Target?></PartNo>
            <Size><?Target?></Size>
         </optionalAttributes>
      </Product>
   </Products>
```

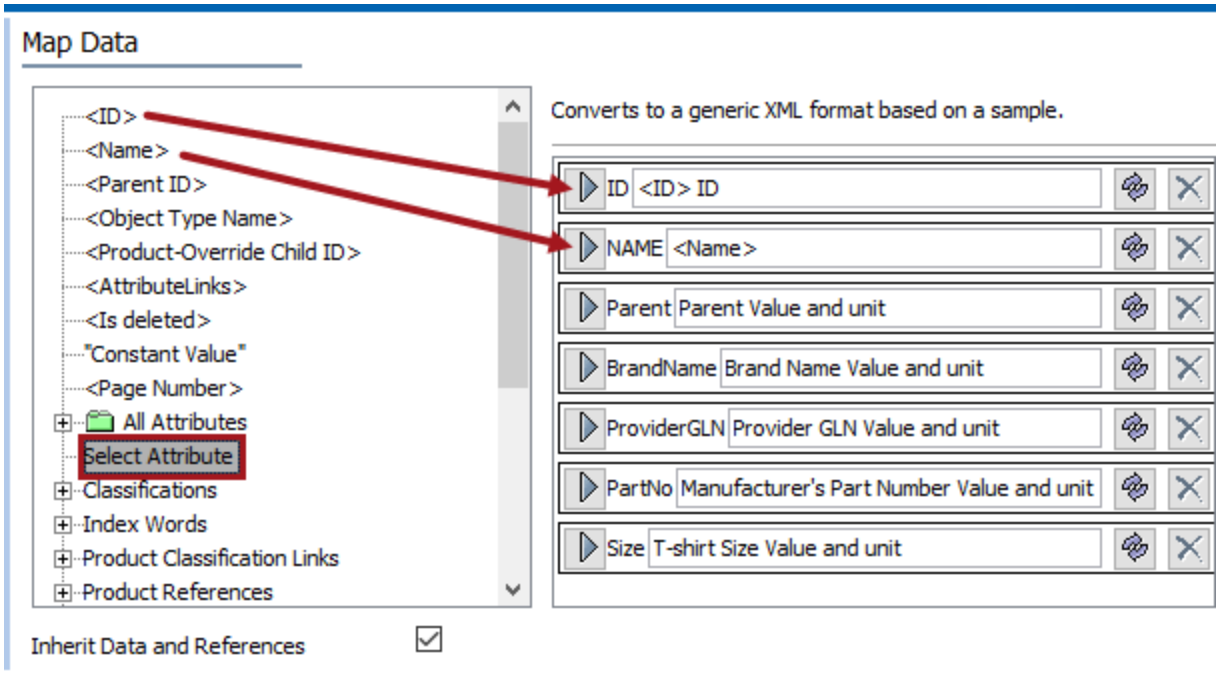
- The RemoveIfEmpty and/or 'RemoveIfEmpty recursive' tags are not used below the 'mandatoryAttributes' section since all attributes should be output, even if they are empty.
- The RemoveIfEmpty and/or 'RemoveIfEmpty recursive' tags are included below the 'optionalAttributes' section since that information is only required if one or more of the targets have values.

Set **Allow empty tags** to **No**. If set to Yes, the 'optionalAttributes' section that is using 'RemoveIfEmpty recursive' instruction will output even when all values are blank.



Mapping

Use the **Select Attribute** aspect for the attribute data targets after ID and Name.



Results

With RemoveIfEmpty

```
<Products>
  <Product>
    <ID>181951</ID>
    <NAME>18216 L O</NAME>
    <mandatoryAttributes>
      <Parent>Cotton T-Shirts</Parent>
      <BrandName>Hanes</BrandName>
      <ProviderGLN>0823335000008</ProviderGLN>
    </mandatoryAttributes>
  </Product>
  <Product>
    <ID>18210</ID>
    <NAME>18210 M B</NAME>
    <mandatoryAttributes>
      <Parent>Cotton T-Shirts</Parent>
      <BrandName>Hanes</BrandName>
      <ProviderGLN/>
    </mandatoryAttributes>
  </Product>
</Products>
```

With 'RemoveIfEmpty recursive'

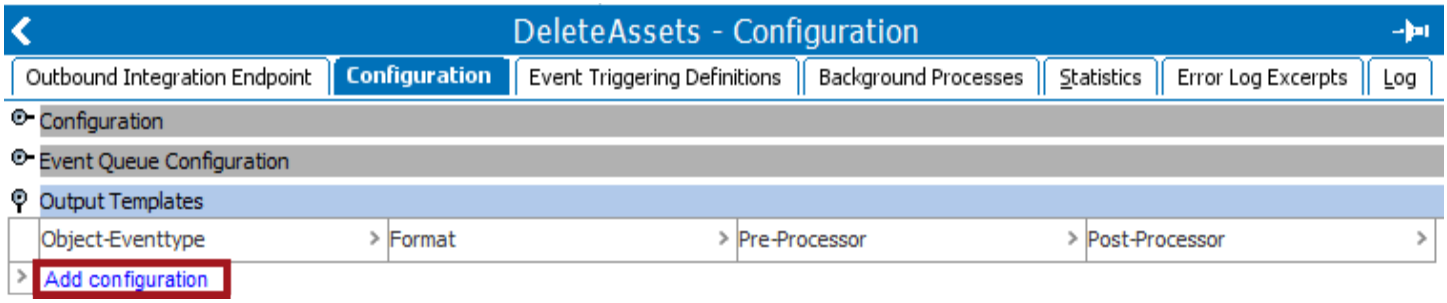
```
<Products>
  <Product>
    <ID>181951</ID>
    <NAME>18216 L O</NAME>
    <mandatoryAttributes>
      <Parent>Cotton T-Shirts</Parent>
      <BrandName>Hanes</BrandName>
      <ProviderGLN>0823335000008</ProviderGLN>
    </mandatoryAttributes>
  </Product>
  <Product>
    <ID>18210</ID>
    <NAME>18210 M B</NAME>
    <mandatoryAttributes>
      <Parent>Cotton T-Shirts</Parent>
      <BrandName>Hanes</BrandName>
      <ProviderGLN/>
    </mandatoryAttributes>
    <optionalAttributes>
      <PartNo>H5250</PartNo>
    </optionalAttributes>
  </Product>
</Products>
```

Reporting Deleted Assets with a Generic XML OIEP

When assets are deleted and approved in STEP, outbound integration endpoints (OIEPs) can pass deletion events when the OIEP is configured to listen for Delete events on the relevant objects. As a result, the downstream system is alerted to the deleted asset.

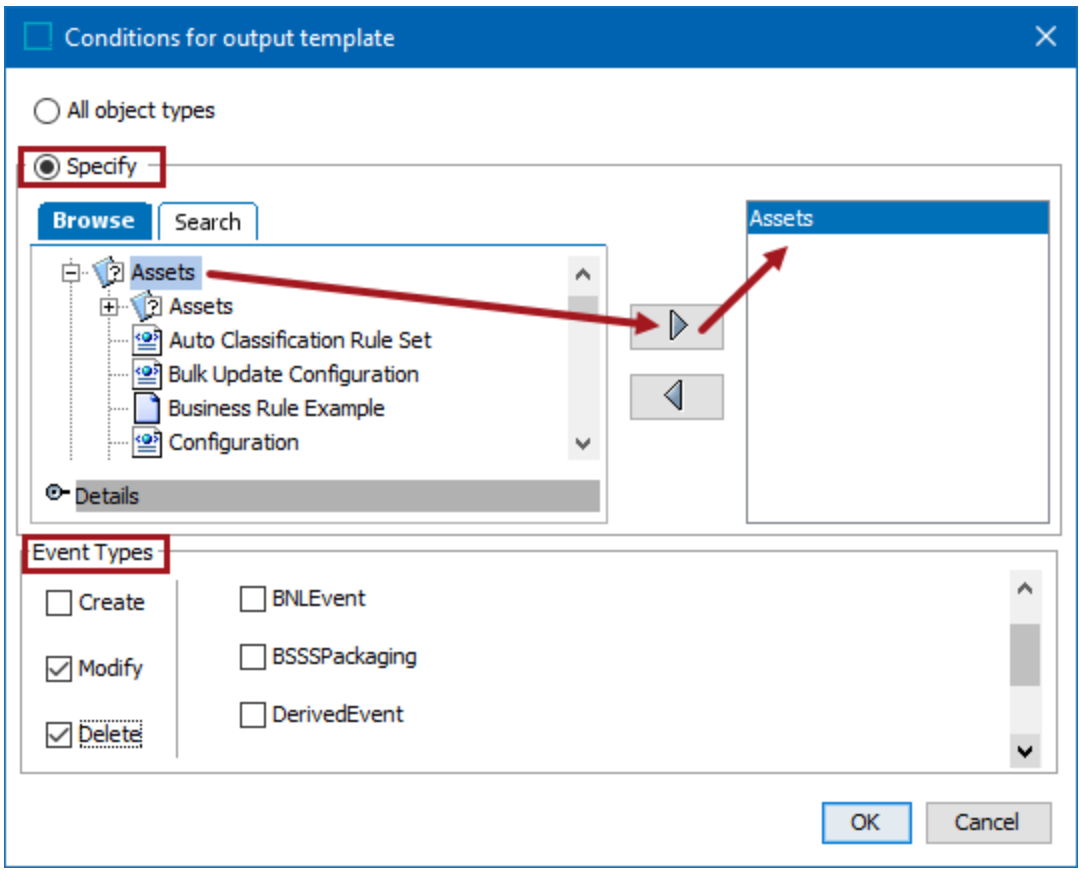
Object Type and Event Type Selection

On the OIEP, open the Configuration tab. Under the Output Templates flipper, click the **Add configuration** link.



Set the Object Types and Event Types.

For this example, Modify events are also included to demonstrate the available functionality.



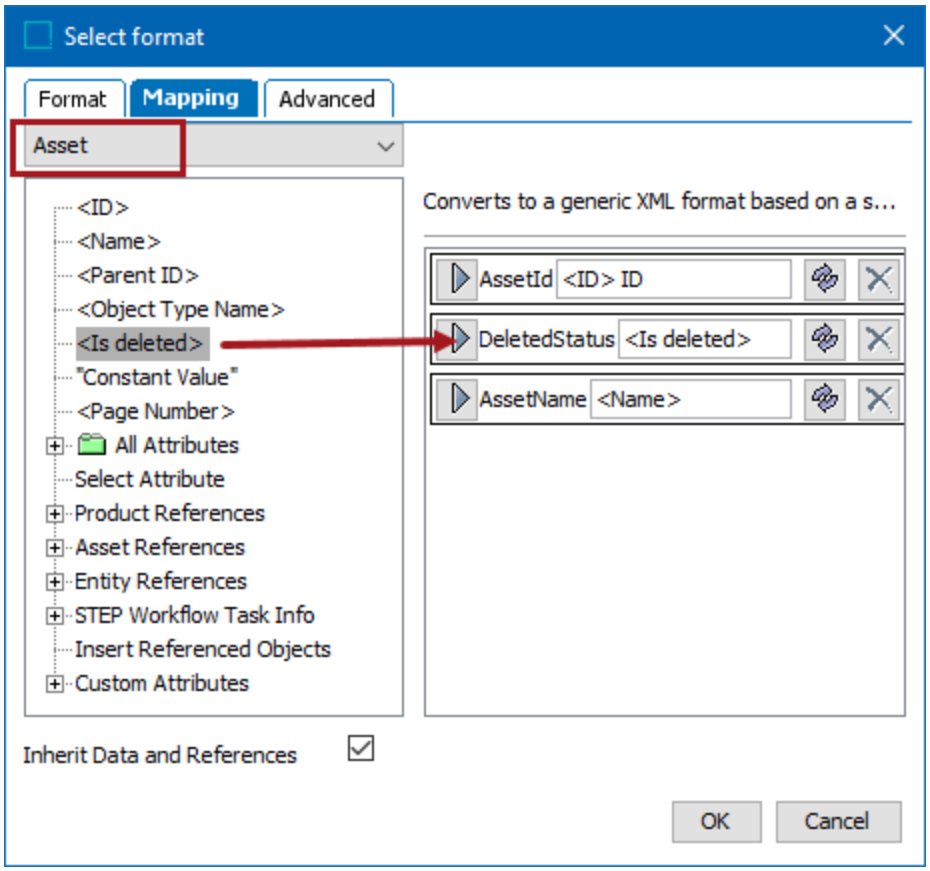
Template

In the Output Template, click the ellipsis button (...), and set the format to **Generic XML**.

Provide the following text in the Sample field.

```
<root>
   <Assets>
      <?Record?>
      <Asset AssetID="[?Target AssetId?]" DeletedStatus="[?Target DeletedStatus?]">
      <AssetName> <?Target?> </AssetName>
      </Asset>
   </Assets>
</root>
```

Mapping



Results

Selecting **Approve Deletion** for an Asset in the Recycle Bin and/or **Approve Object** for modification of an Asset on the Tree triggers the OIEP.

Invoking the OIEP generates the following output:

```
<root>
  <Assets>
    <Asset AssetID="MSDS_00020" DeletedStatus="IsDeleted">
      <AssetName/>
    </Asset>
  </Assets>
  <Assets>
    <Asset AssetID="MSDS_00019" DeletedStatus="">
      <AssetName>MSDS_00019 modified</AssetName>
    </Asset>
  </Assets>
</root>
```

For deleted Assets, the Deleted Status displays **"IsDeleted"**. Only ID is retained for export on Delete events.

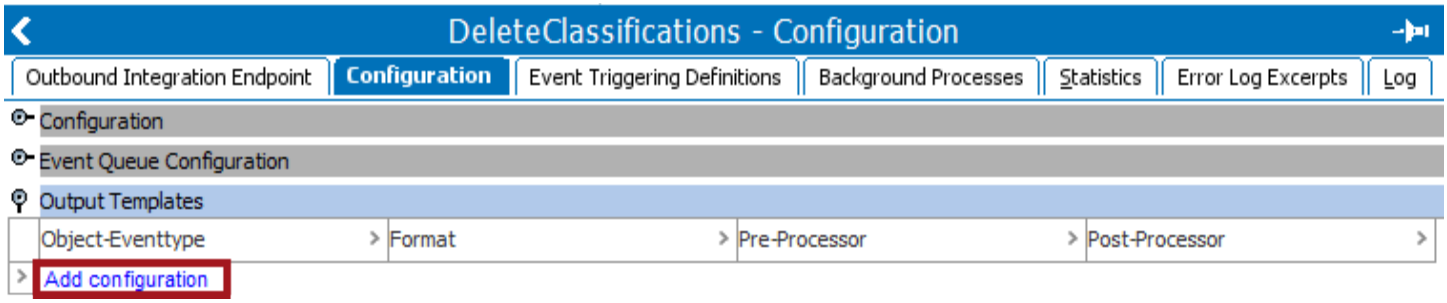
For modified Assets, the Deleted Status displays **" "**, which indicates the asset is not deleted. All mapped fields are exported for Create and Modify events.

Reporting Deleted Classifications with a Generic XML OIEP

When classifications are deleted and approved in STEP, outbound integration endpoints (OIEPs) can pass deletion events when the OIEP is configured to listen for Delete events on the relevant objects. As a result, the downstream system is alerted to the deleted classification.

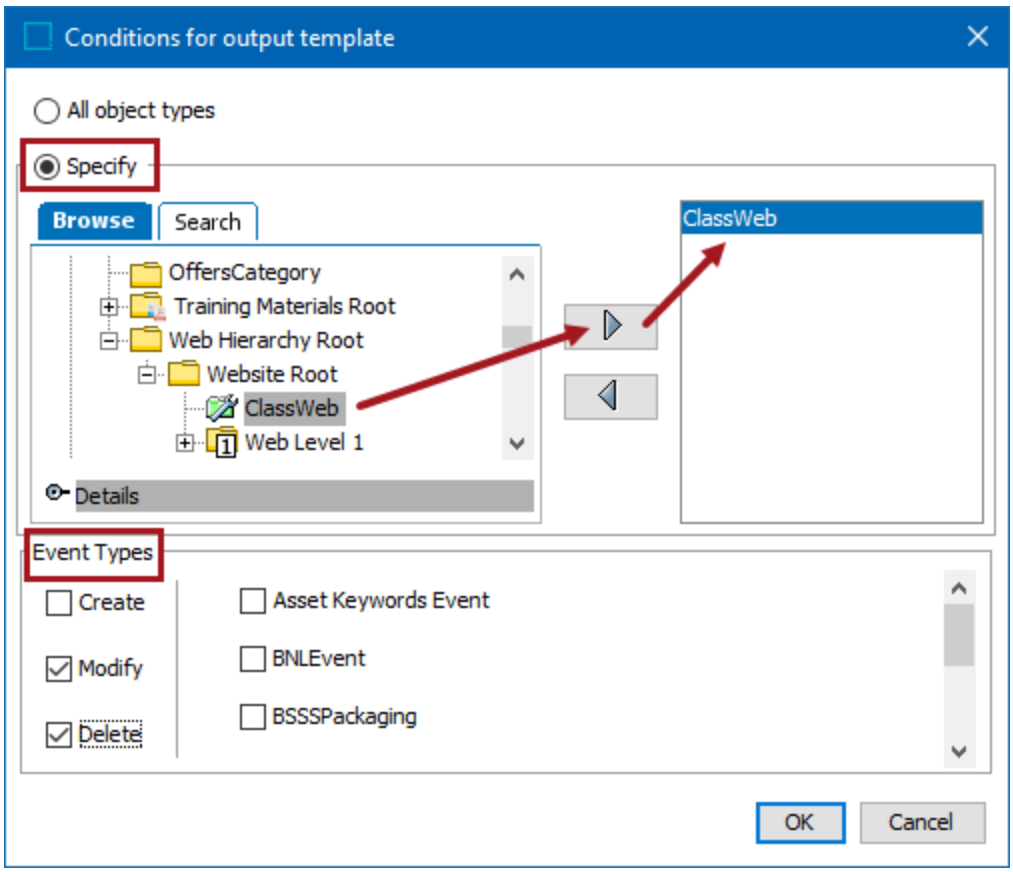
Object Type and Event Type Selection

On the OIEP, open the Configuration tab. Under the Output Templates flipper, click the **Add configuration** link.



Set the Object Types and Event Types.

For this example, Modify events are also included to demonstrate the available functionality.



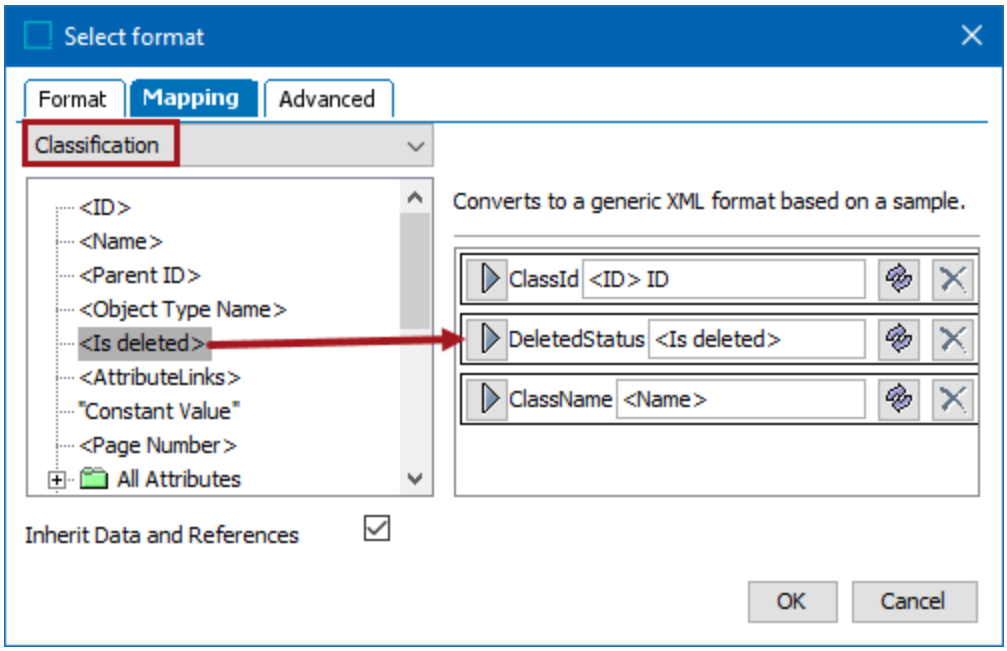
Template

In the Format field, click the ellipsis button (...), and set the format to **Generic XML**.

Provide the following text in the Sample field.

```
<root>
   <Classifications>
      <?Record?>
      <Class ClassID="[?Target ClassId?]" DeletedStatus="[?Target DeletedStatus?]">
      <ClassName> <?Target?> </ClassName>
      </Class>
   </Classifications>
</root>
```

Mapping



Result

Approve Deletion for a Classification in the Recycle Bin and/or **Approve Object** for modification of a Classification on the Tree triggers the OIEP.

Invoking the OIEP generates the following output:

```
<root>
  <Classifications>
    <Class ClassID="SofasAndChairs" DeletedStatus="IsDeleted">
      <ClassName/>
    </Class>
  </Classifications>
  <Classifications>
    <Class ClassID="TablesAndChairs" DeletedStatus="">
      <ClassName>Tables And Chairs modified</ClassName>
    </Class>
  </Classifications>
</root>
```

For deleted Classifications, the Deleted Status displays **"IsDeleted"**. Only ID is retained for export on Delete events.

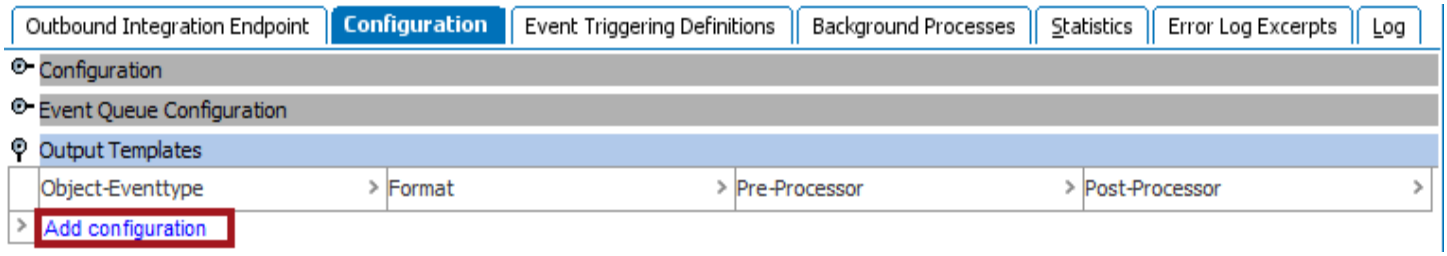
For modified Classifications, the Deleted Status displays **" "**, which indicates the classification is not deleted. All mapped fields are exported for Create and Modify events.

Reporting Deleted Entities with a Generic XML OIEP

When Entities are deleted and approved in STEP, outbound integration endpoints (OIEPs) can pass deletion events when the OIEP is configured to listen for Delete events on the relevant objects. As a result, the downstream system is alerted to the deleted asset. Entities must have the Revisability field set to Workspace Revisable, not Global Revisable.

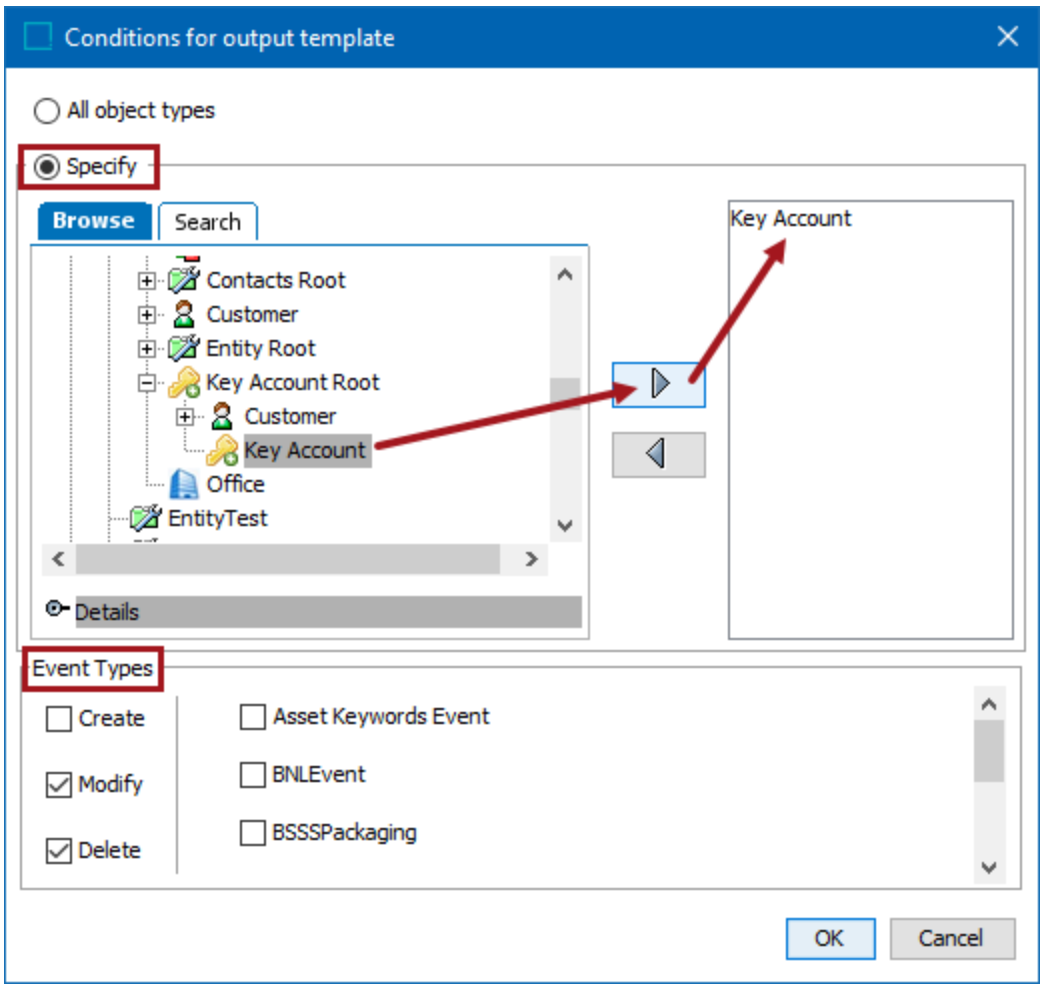
Object Type and Event Type Selection

On the OIEP, open the Configuration tab. Under the Output Templates flipper, click the **Add configuration** link.



Set the Object Types and Event Types.

For this example, Modify events are also included to demonstrate the available functionality.



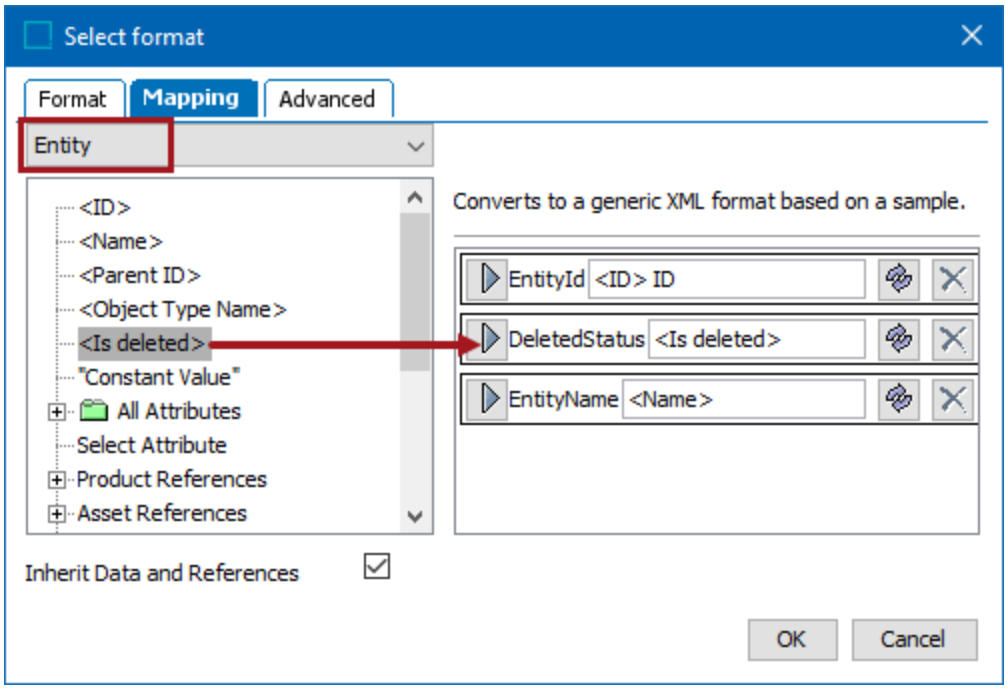
Template

In the Output Template, click the ellipsis button (...), and set the format to **Generic XML**.

Provide the following text in the Sample field.

```
<root>
   <Entities>
      <?Record?>
      <Entity EntityID="[?Target EntityId?]" DeletedStatus="[?Target DeletedStatus?]">
      <EntityName> <?Target?> </EntityName>
      </Entity>
   </Entities>
</root>
```

Mapping



Results

Approve Deletion for an Entity in the Recycle Bin and/or **Approve Object** for modification of a Entity that is Workspace Revisable on the Tree triggers the OIEP.

Invoking the OIEP generates the following output:

```
<root>
  <Entities>
    <Entity EntityID="KeyB" DeletedStatus="IsDeleted">
      <EntityName/>
    </Entity>
  </Entities>
  <Entities>
    <Entity EntityID="KeyA" DeletedStatus="">
      <EntityName>Key A modified</EntityName>
    </Entity>
  </Entities>
</root>
```

For deleted Entities, the Deleted Status displays **"IsDeleted"**. Only ID is retained for export on Delete events.

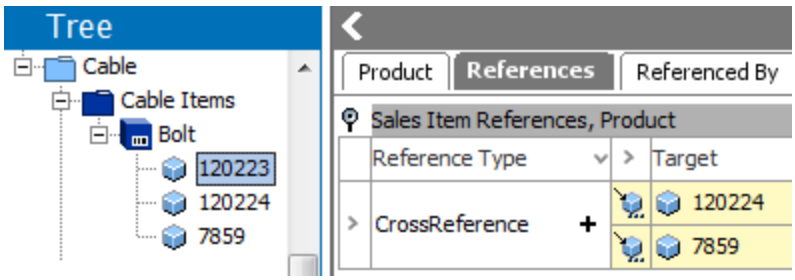
For modified Entities, the Deleted Status displays **" "**, which indicates the entity is not deleted. All mapped fields are exported for Create and Modify events.

Reporting Deleted Product References with a Generic XML OIEP

Outbound integration endpoints (OIEPs) can pass deletion events for product references when the OIEP is configured to listen for modify events on the relevant objects. The Modify Event Type is necessary since the object itself is not being deleted, but rather a reference on the object which continues to exist.

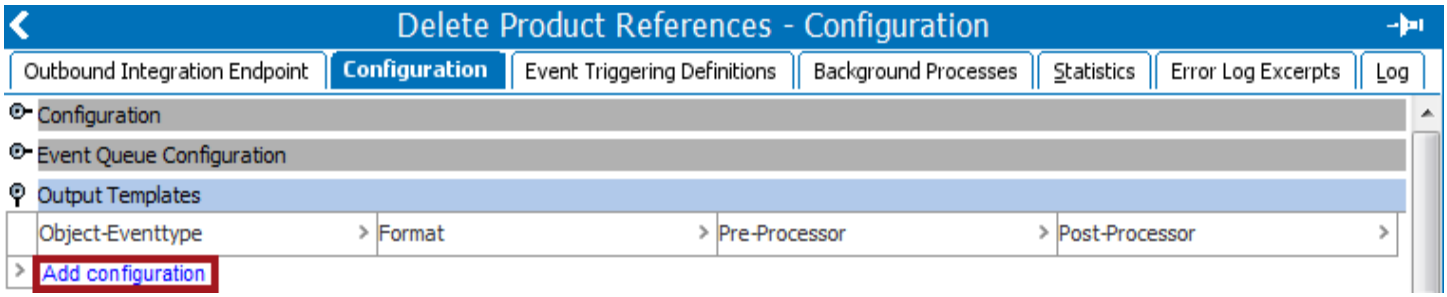
Product

On product 120223, the Reference Type of CrossReference to Target 7859 must be deleted in order to trigger the OIEP.

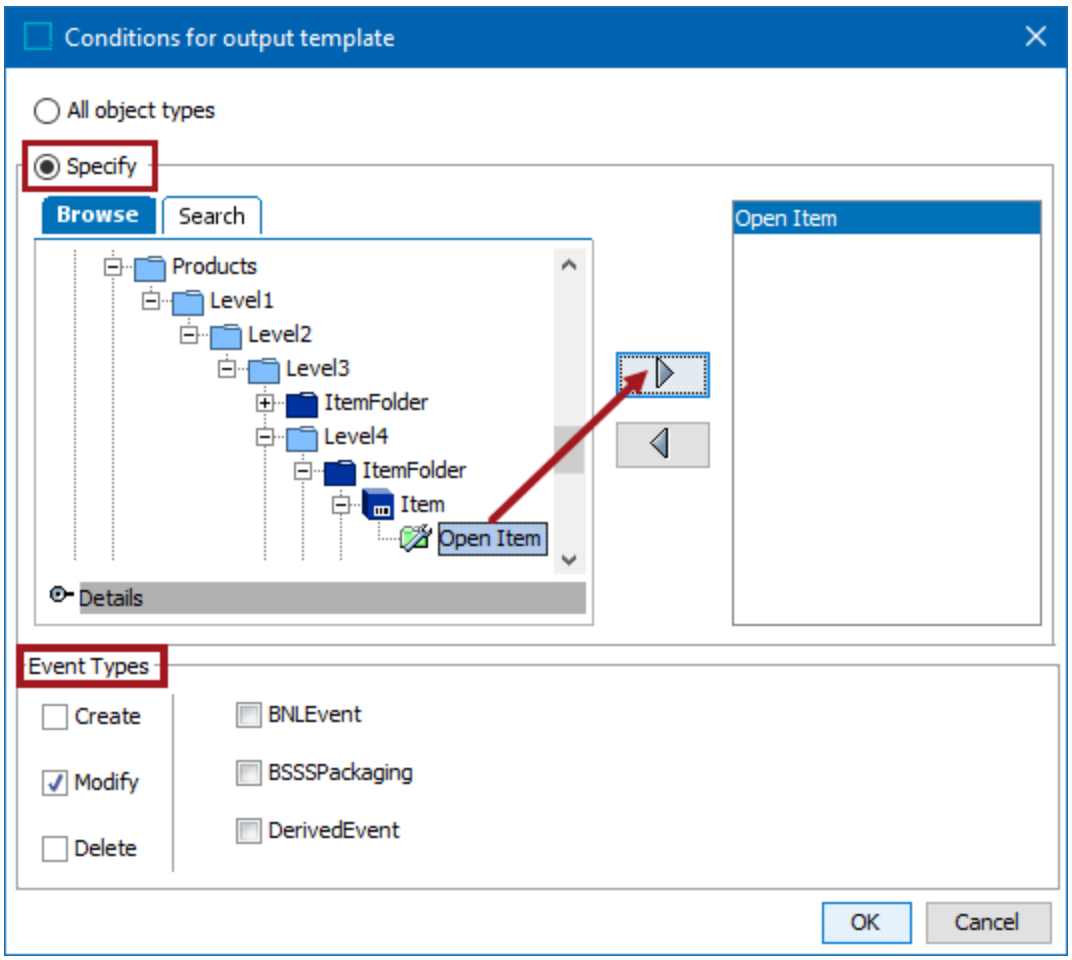


Object Type and Event Type Selection

On the OIEP, open the Configuration tab. Under the Output Templates flipper, click the **Add configuration** link.



Set the Object Types and Event Types.



Template

In the Output Template, click the ellipsis button (...), and set the format to **Generic XML**.

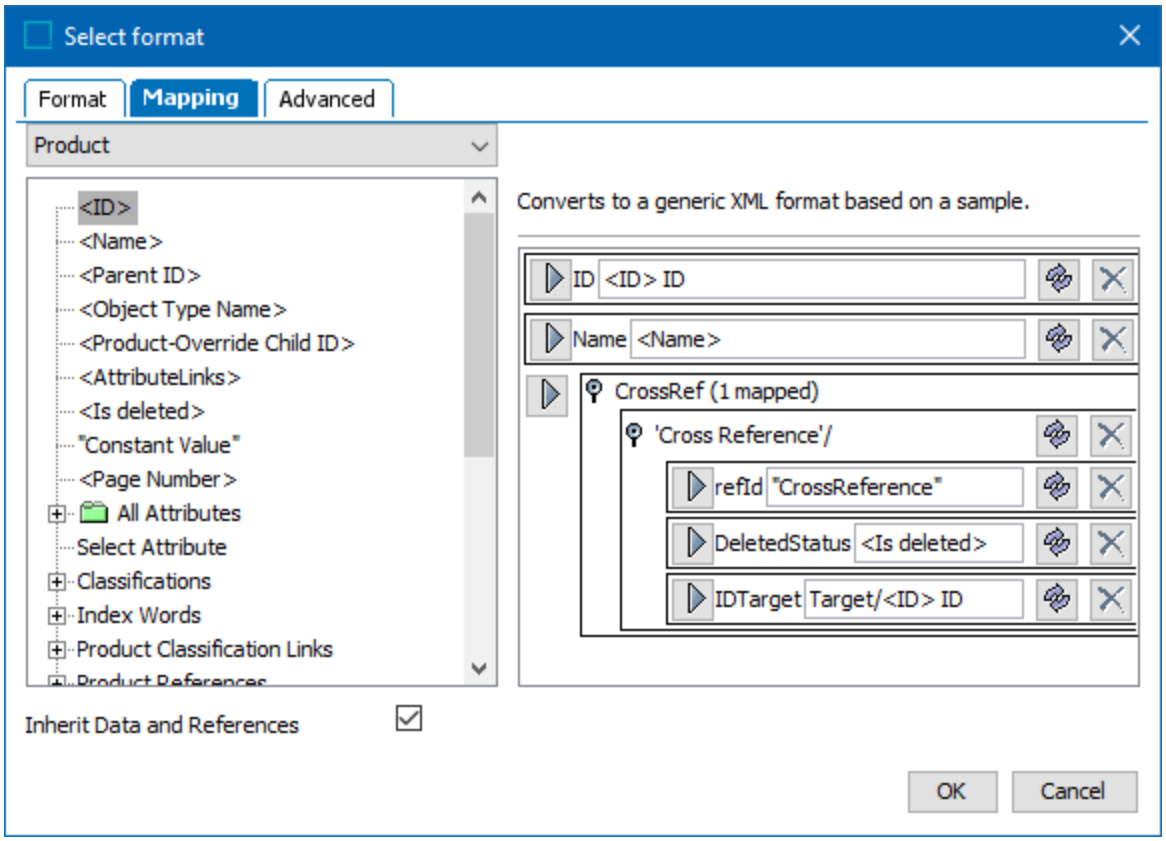
Provide the following text in the Sample field.

```
    <Products>
       <Product ID="[?Target ID?]">
       <?Record?>
       <Name><?Target?></Name>
       <CrossRefs>
          <CrossRef refTypeID="[?Target refId?]" DeletedStatus="[?Target DeletedStatus?]">
          <?MultiTarget?>
          <IDTarget><?Target?></IDTarget>
          </CrossRef>
```

```
        </CrossRefs>
        </Product>
    </Products>
```

Mapping

The Constant Value 'CrossReference' is used to identify the type of reference being monitored. The <Is deleted> aspect monitors for deletions. Data Path mapping to the Target reference provides the deleted ID.



For detailed mapping steps, refer to the **Mapping to Report Deleted Product References with Generic XML OIEP** documentation.

Results

When the Product Reference for item 7859 is deleted and the product 120223 is approved, the OIEP is triggered. Invoking the OIEP generates the following file:

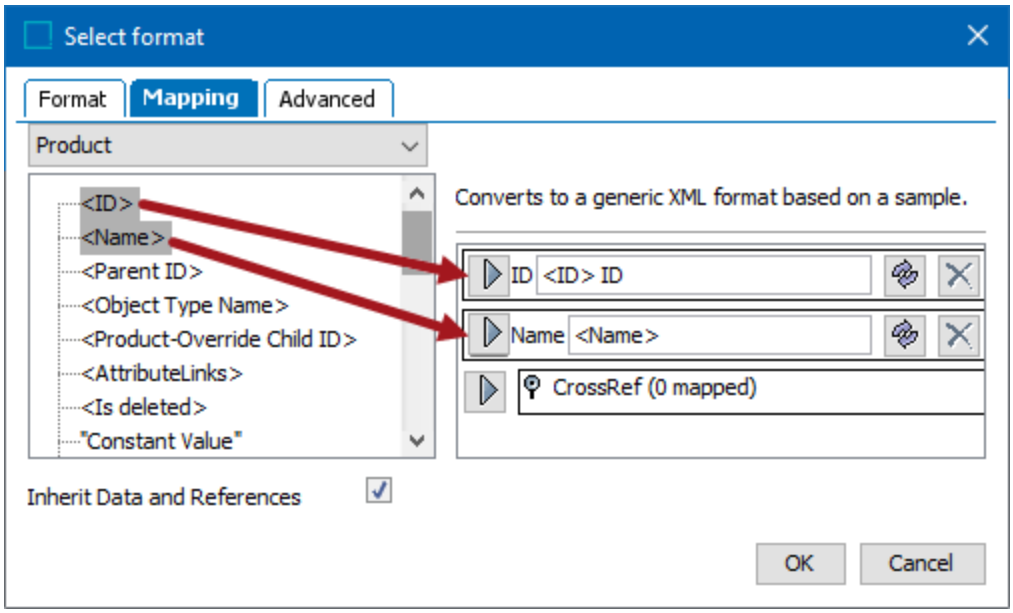
```
<Products>
  <Product ID="120223">
    <Name>120223</Name>
    <CrossRefs>
      <CrossRef refTypeID="CrossReference" DeletedStatus="">
        <IDTarget>120224</IDTarget>
      </CrossRef>
      <CrossRef refTypeID="CrossReference" DeletedStatus="IsDeleted">
        <IDTarget>7859</IDTarget>
      </CrossRef>
    </CrossRefs>
  </Product>
</Products>
```

Note: Both of the product references originally on the product are exported. The DeletedStatus target shows blank for the reference that still exists, and shows "IsDeleted" for the removed reference.

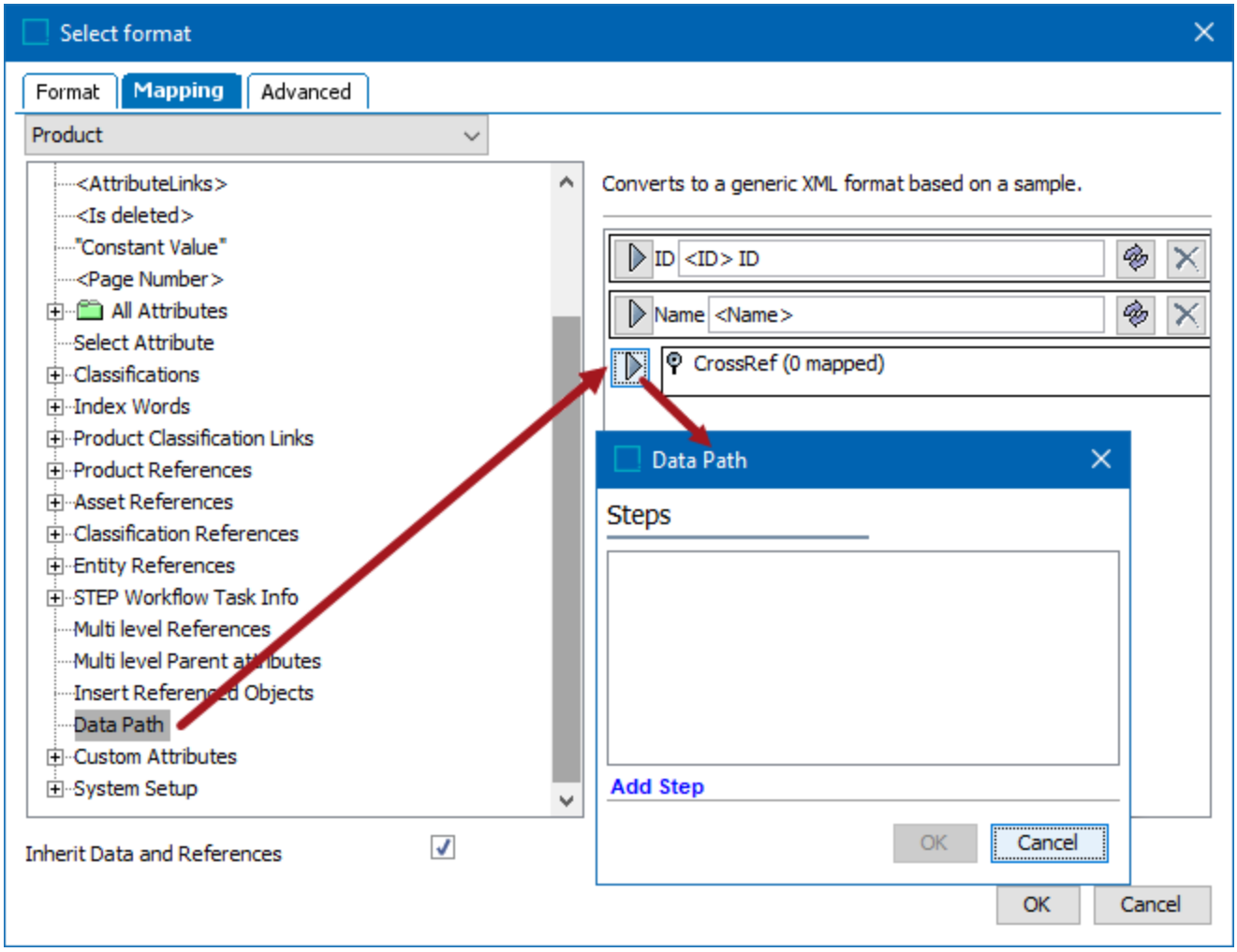
Mapping to Report Deleted Product References with a Generic XML OIEP

The **Reporting Deleted Product References with Generic XML OIEP** section discusses additional setup for a Generic XML export. Use the following steps to map the Product and monitor for deletion of product references.

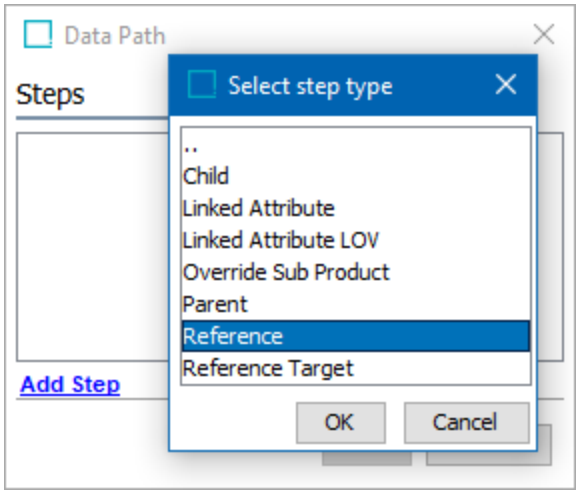
1. Map **ID** and **Name** for the Product.



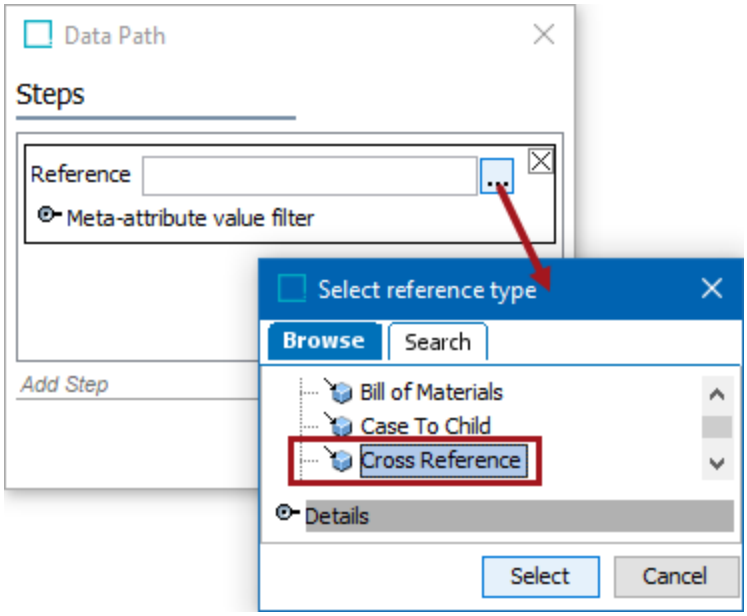
2. Map **Data Path** to CrossRef.



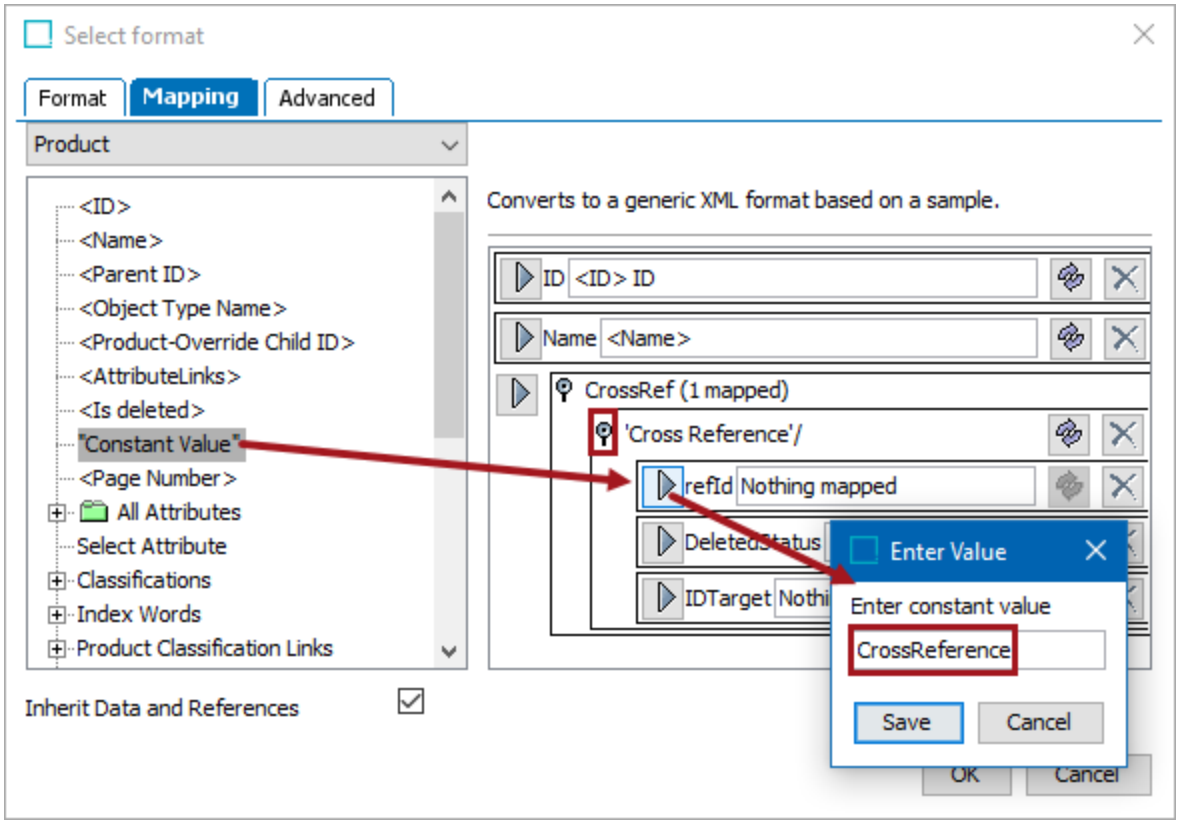
3. Click the **Add Step** link and select **Reference**.



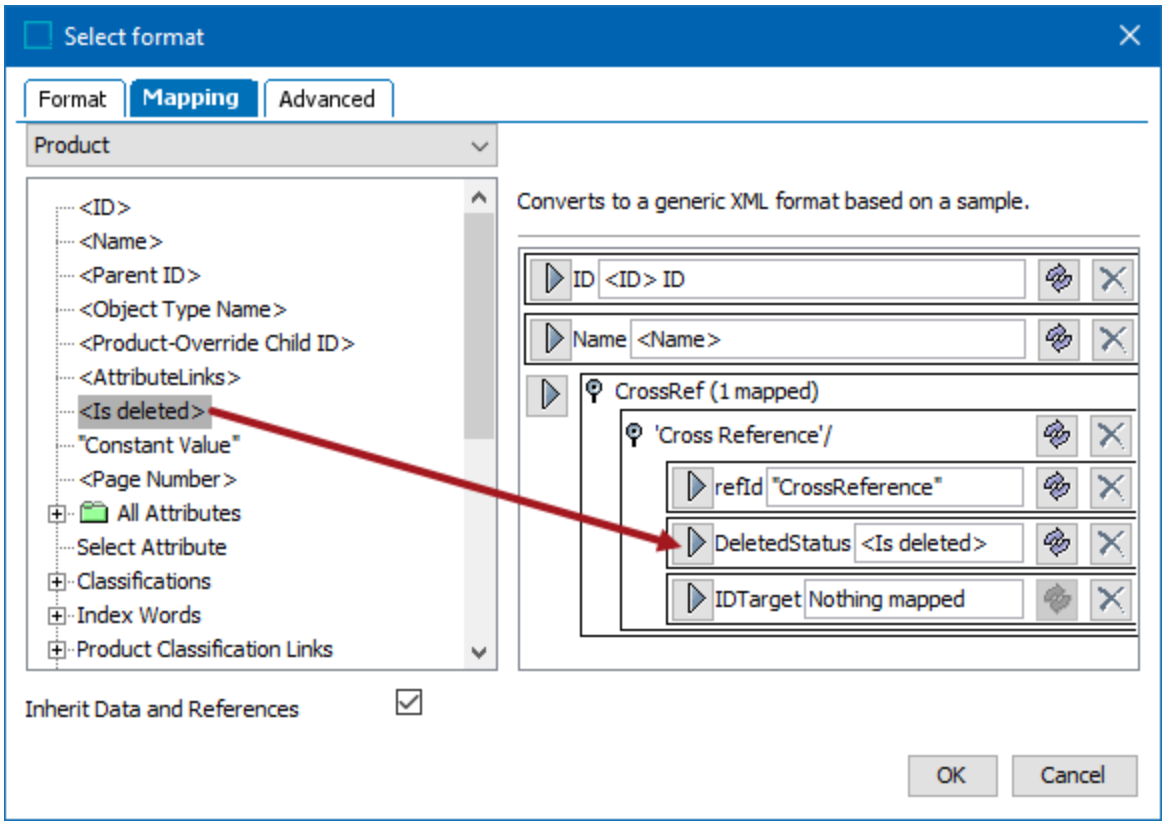
4. Click **OK** to close the **Select step type** dialog.
5. On the Data Path dialog, click the ellipsis button (...) and select the **Reference Type** to monitor, then click the **Select** button.



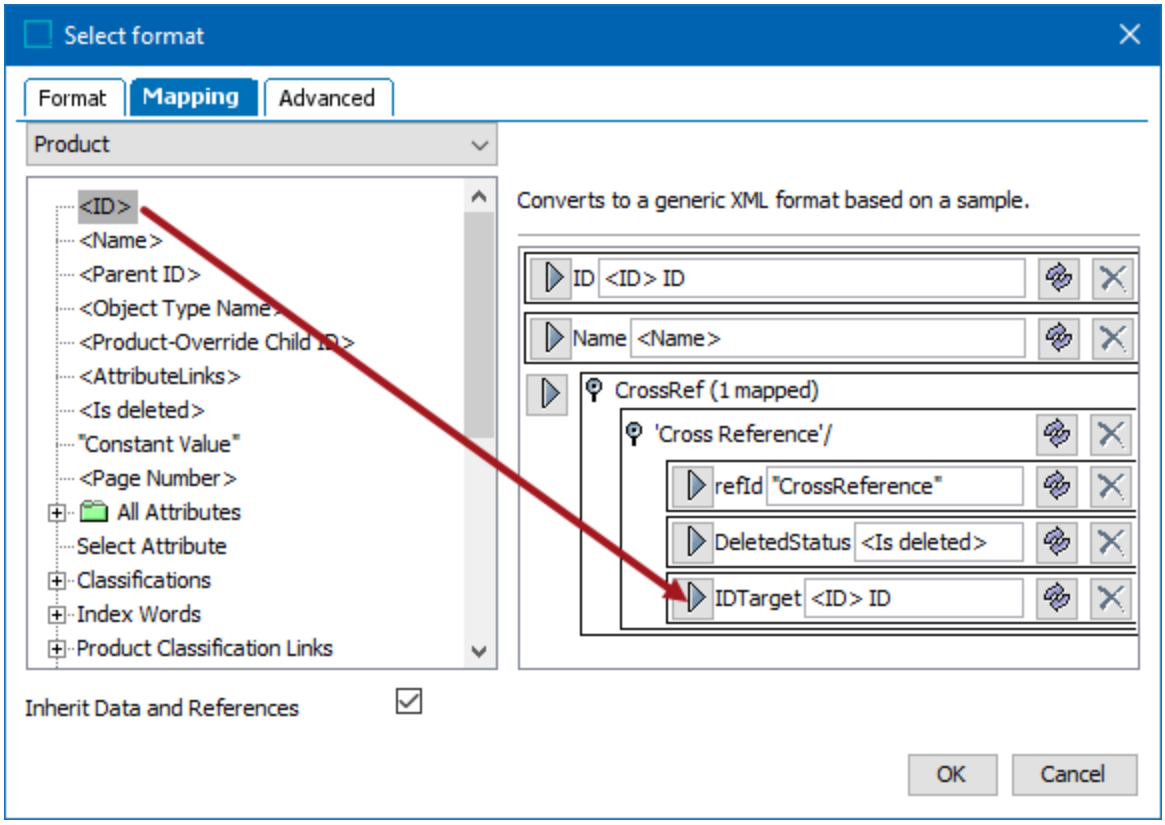
6. Click **OK** to close the **Data Path** dialog.
7. Open the 'CrossReference'/ flipper, map **Constant Value** to refID, add the text, and click the **Save** button.



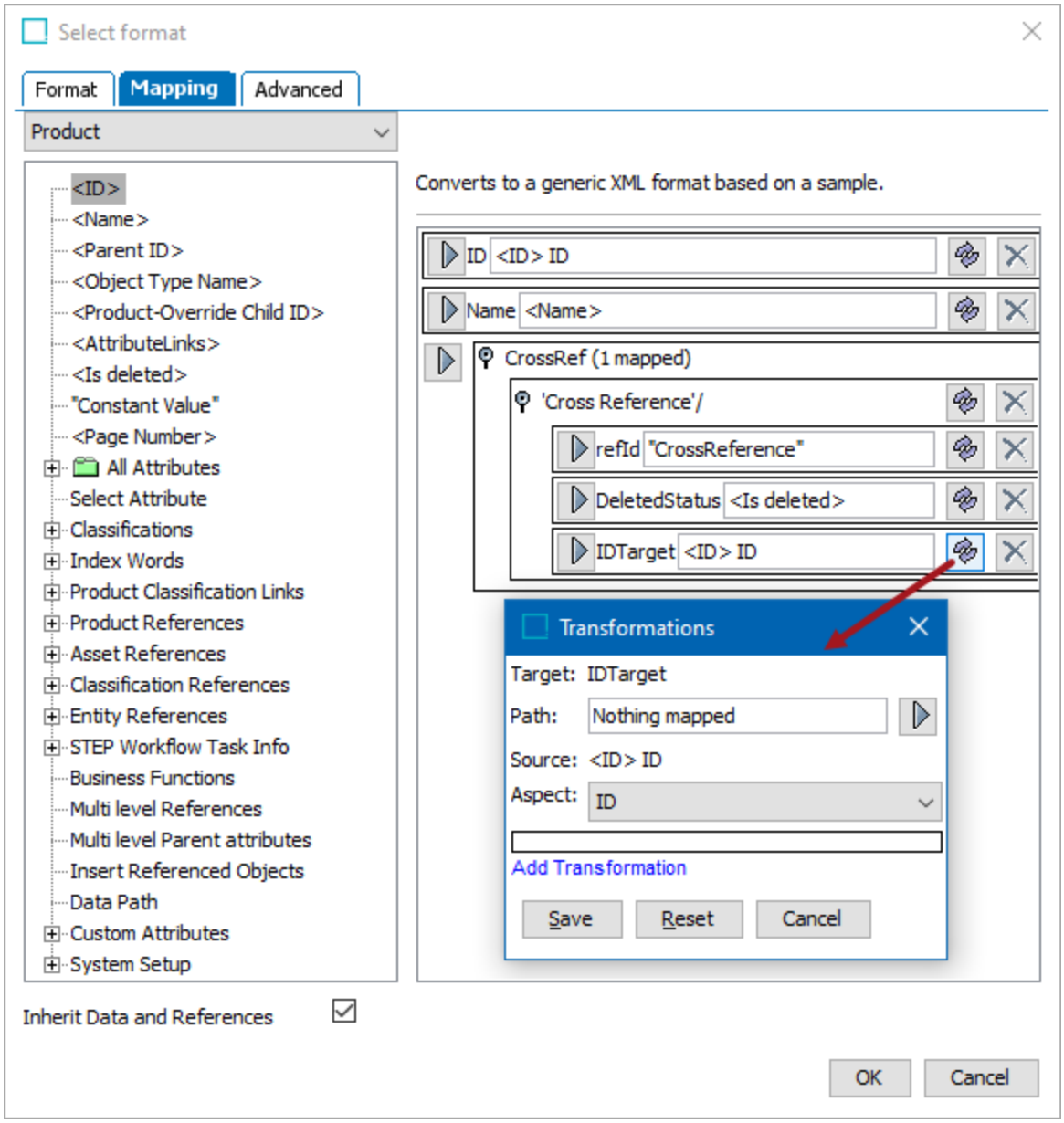
8. Map **<Is deleted>** to DeletedStatus.



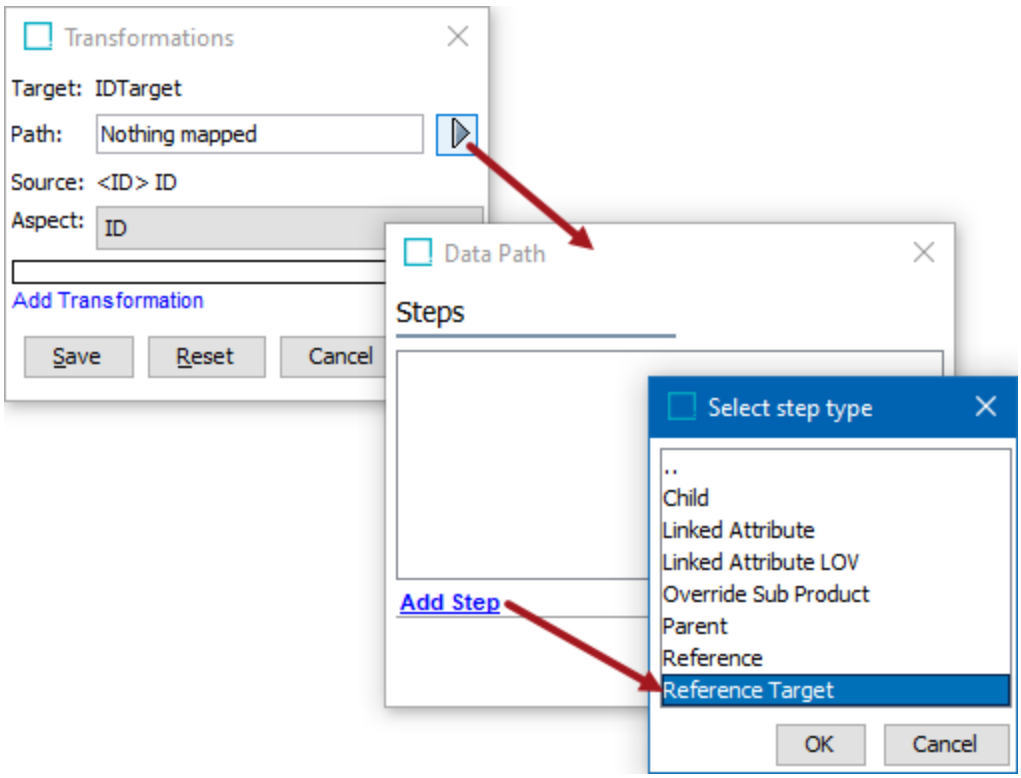
9. Map **ID** to IDTarget.



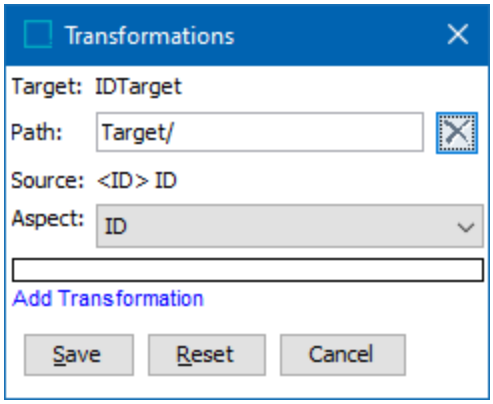
10. Click the **Transformations** button.



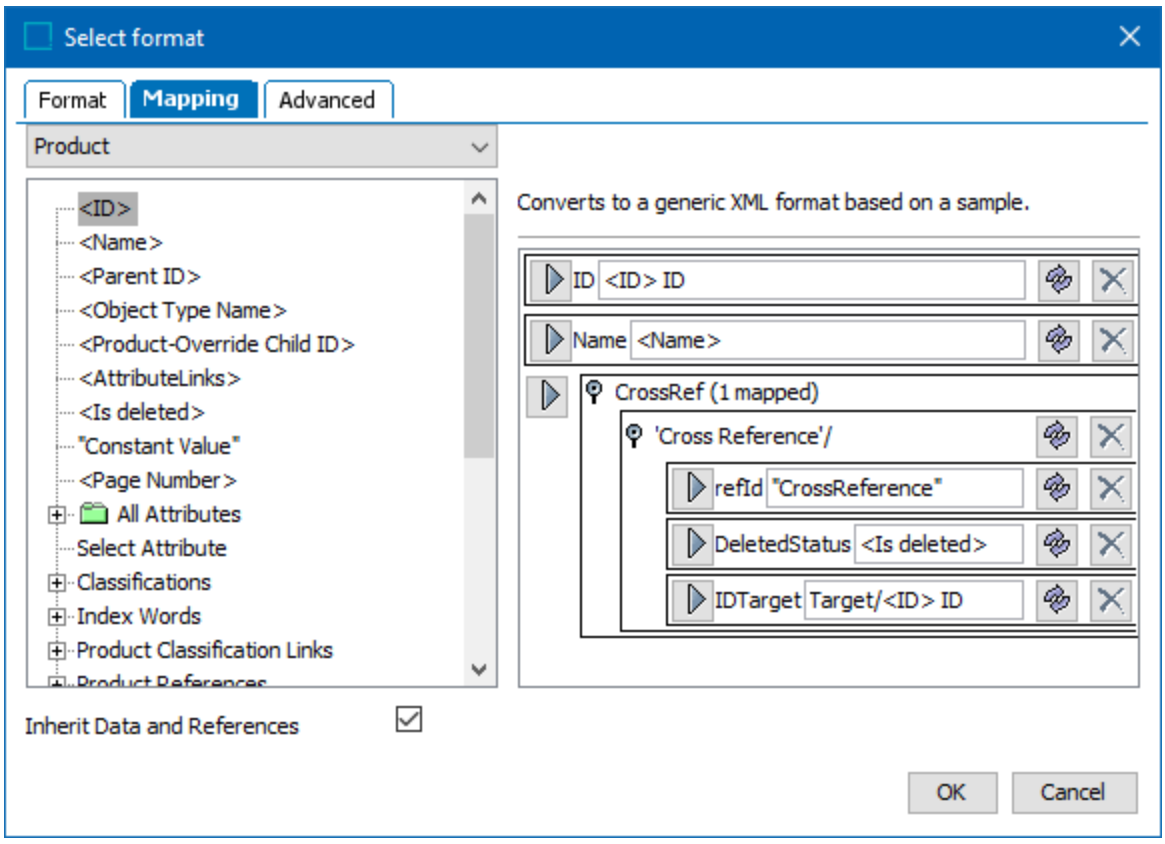
11. Click the **Path** button, click the **Add Step** link, and select **Reference Target**.



12. Click **OK** to close the **Select step type** and **Data Path** dialogs.
13. Click **Save** to close the **Transformations** dialog.



All mapping is complete.


Select format
Format
Mapping
Advanced
Product
<ID>
<Name>
<Parent ID>
<Object Type Name>
<Product-Override Child ID>
<AttributeLinks>
<Is deleted>
"Constant Value"
<Page Number>
All Attributes
Select Attribute
Classifications
Index Words
Product Classification Links
Converts to a generic XML format based on a sample.
ID <ID> ID
Name <Name>
CrossRef (1 mapped)
'Cross Reference'/
refId "CrossReference"
DeletedStatus <Is deleted>
IDTarget Target/<ID> ID
Inherit Data and References
OK
Cancel

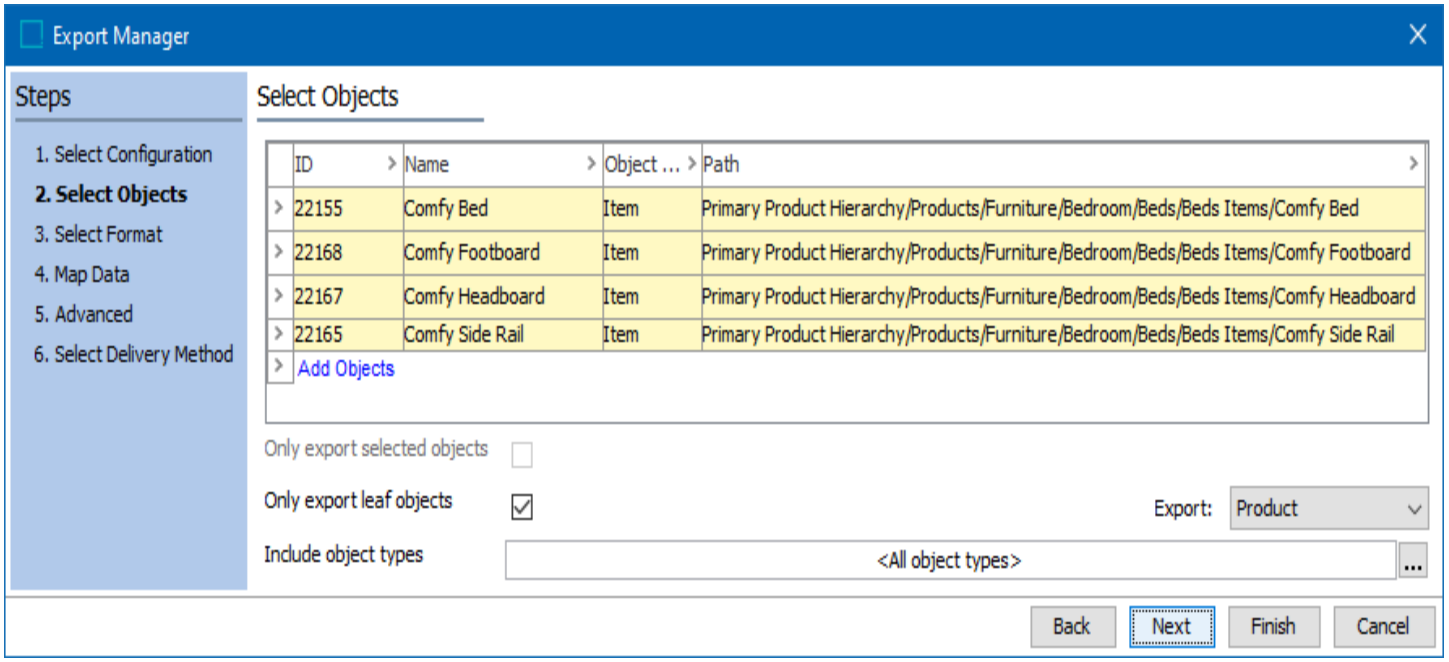
Single-Cardinality Tags in Generic XML

A Generic XML template can include a variety of tags to affect the generated XML document. The following tags are single-cardinality, which means that the values exported by the associated data source is considered a 'single piece of information' and it will be exported inside a single occurrence of the enclosing XML element.

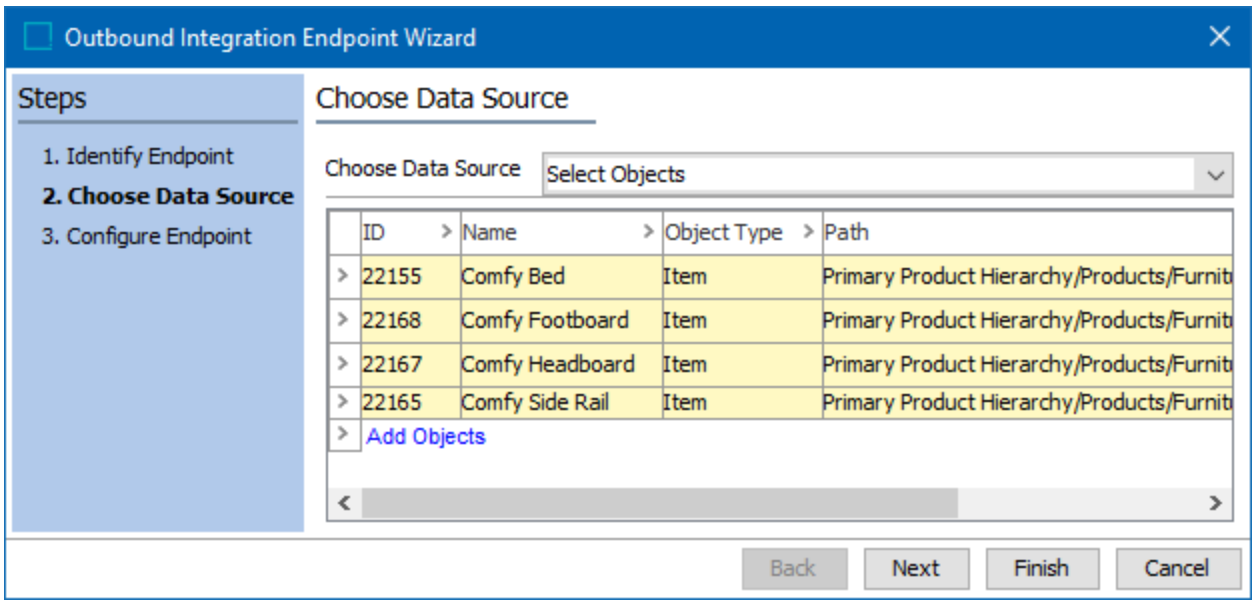
The following single cardinality mapping target tags are discussed below: Record, Target Default, and Target Override Default.

Objects

Choose the objects to export in Export Manager:



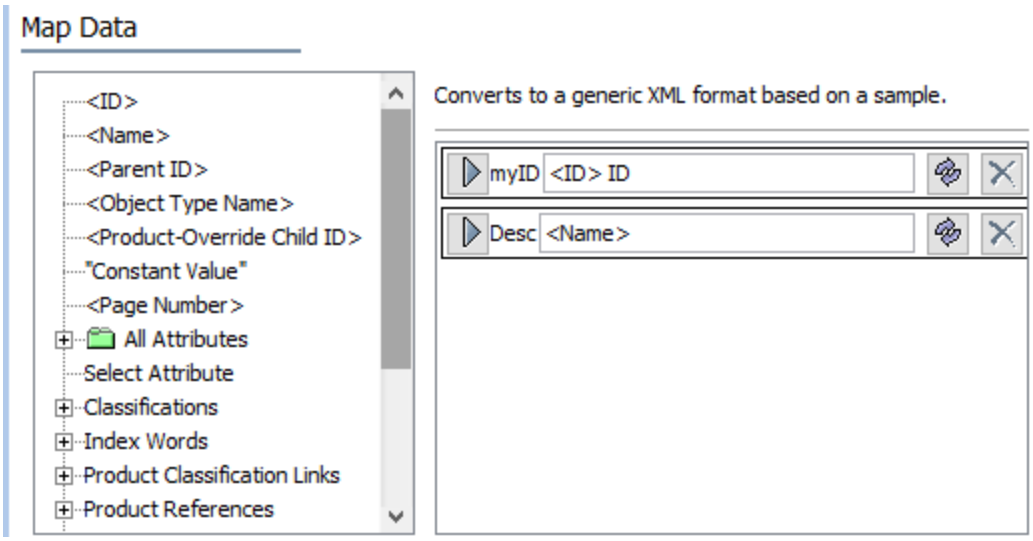
or in an OIEP:



Template

```
<Products>
   <Product>
      <?Record?>
      <ID><?Target myID?></ID>
      <Desc><?Target?></Desc>
   </Product>
</Products>
```

Mapping



Results

```
<Products>
  <Product>
    <ID>22155</ID>
    <Desc>Comfy Bed</Desc>
  </Product>
  <Product>
    <ID>22165</ID>
    <Desc>Comfy Side Rail</Desc>
  </Product>
  <Product>
    <ID>22167</ID>
    <Desc>Comfy Headboard</Desc>
  </Product>
  <Product>
    <ID>22168</ID>
    <Desc>Comfy Footboard</Desc>
  </Product>
</Products>
```

Record

The <?Record?> tag is a processing instruction that tells Export Manager which part of the template to repeat for each selected object. There must be exactly one <?Record?> in the template.

In the example, the Product tag is the Record node of the template. Therefore everything between the <Product> and </Product> tags will be repeated for each object exported from STEP and will be included in the Generic XML output document.

Target Default

The <?Target?> tag defines a mapping target to which a data source can be bound. Each mapping target must have a unique identifier so it can be correctly used in the mapping step. By default, the name of the surrounding tag is used. Our example template includes two mapping targets: myID and Desc.

Target Override Default

When the same default identifier is used by more than one mapping target, you must override the default. For example, if you are exporting the ID of a product and the ID of an asset, <ID> would default for both objects. To create a unique identifier, specify a custom identifier within the <?Target [Optional Identifier]?> tag as indicated by 'myID' in the example above. Use any unique text in place of 'myID' to accurately label the required data.

Note: To avoid duplicate ID labels, use custom IDs for attribute-based mapping targets.

Rules

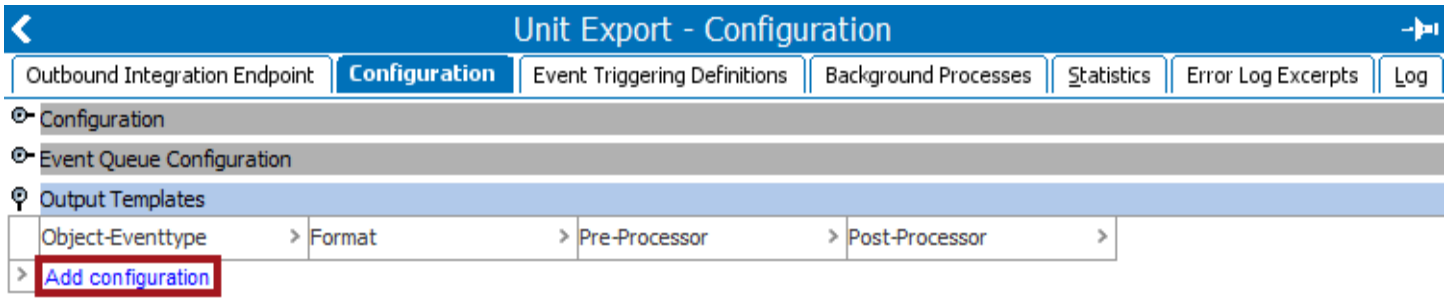
- There must be exactly one <?Record?> in the template.
- Mapping targets may only be used inside <?Record?> scope.
- Each mapping target must have a unique label.

Units with a Generic XML OIEP

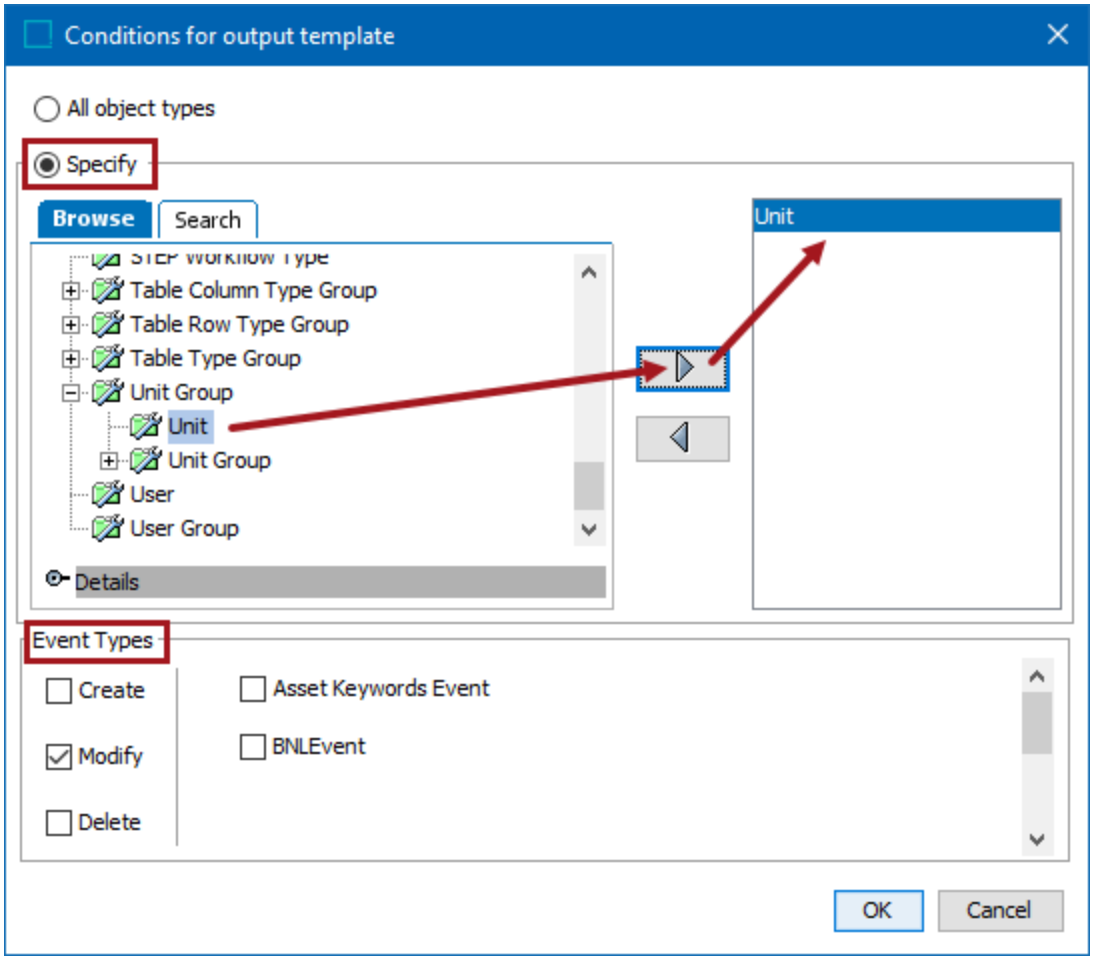
Outbound integration endpoints (OIEPs) allow the export of unit information.

Object Type and Event Type Selection

On the OIEP, open the Configuration tab. Under the Output Templates flipper, click the **Add configuration** link.



Set the Object Types and Event Types.



Template

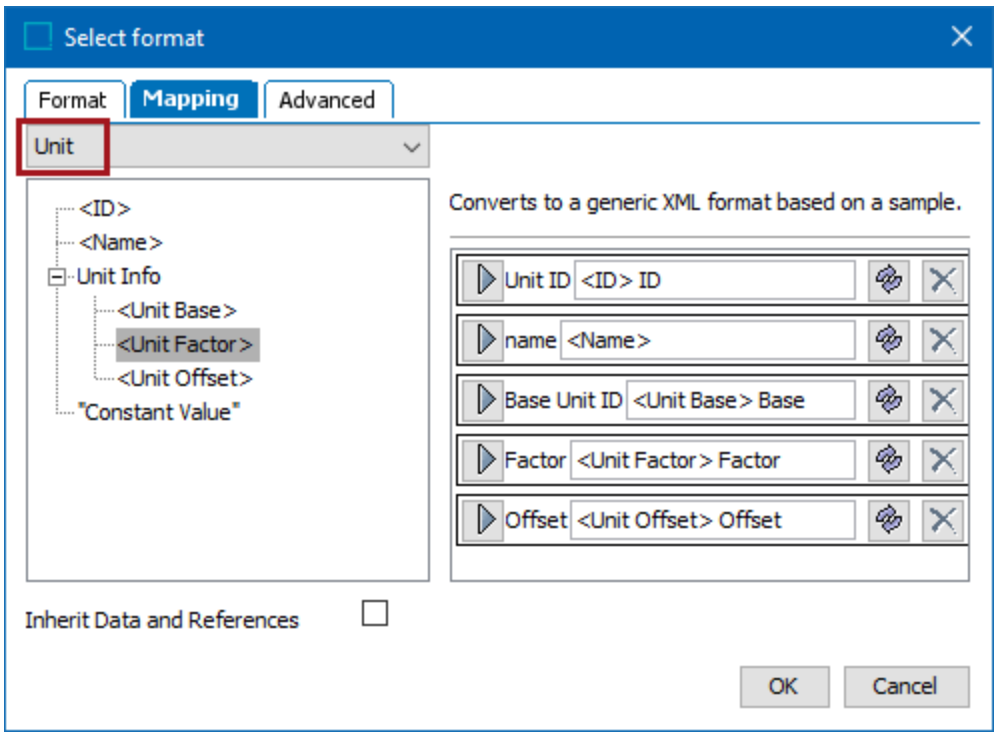
In the Output Template, click the ellipsis button (...), and set the format to **Generic XML**.

Provide the following text in the Sample field.

```
<ExportUnits>
   <Units>
      <Unit ID="[?Target Unit ID?]">
         <?Record?>
         <Name>
             <?Target name?>
         </Name>
         <UnitConversion BaseUnitID="[?Target Base Unit ID?]" Factor="[?Target Factor?]" Offset="[?Target Offset?]"/>
      </Unit>

   </Units>

</ExportUnits>
```

Mapping

Select **Unit** from the dropdown and then map the data sources to the targets provided by the template.



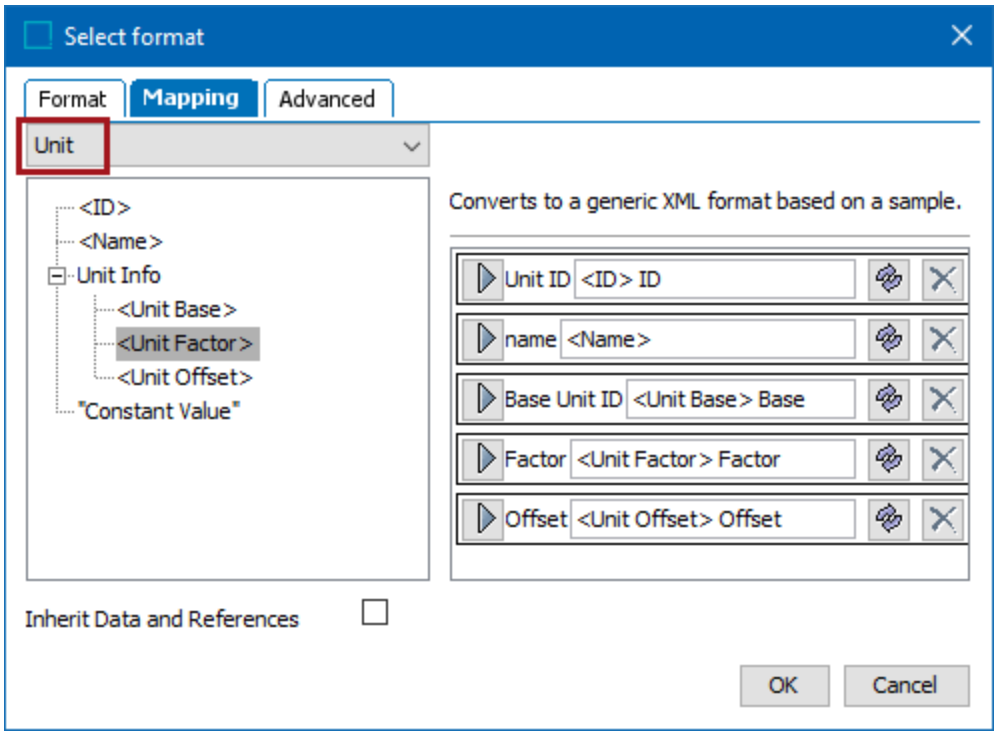
Refer to the **Mapping Units Aspects with Generic XML OIEP** section for more information.

Results

```
<ExportUnits>
  <Units>
    <Unit ID="unece.unit.INK">
      <Name>in²</Name>
      <UnitConversion BaseUnitID="unece.unit.MTK" Factor="6.4515E-4" Offset="0.0"/>
    </Unit>
    <Unit ID="unece.unit.ACR">
      <Name>acre</Name>
      <UnitConversion BaseUnitID="unece.unit.MIK" Factor="3.0" Offset="1.0"/>
    </Unit>
    <Unit ID="unece.unit.FTK">
      <Name>ft²</Name>
      <UnitConversion BaseUnitID="unece.unit.MTK" Factor="0.09290305" Offset="0.0"/>
    </Unit>
    <Unit ID="unece.unit.CMK">
      <Name>cm²</Name>
      <UnitConversion BaseUnitID="unece.unit.MTK" Factor="2.0E-4" Offset="0.0"/>
    </Unit>
  </Units>
</ExportUnits>
```

Mapping Unit Aspects with a Generic XML OIEP

Aspects, also called data sources, are displayed when mapping Units for export. The description of each Aspect is included below.



Aspect	Description
ID	Extracts the ID of the unit. For example, if a unit has the ID 1234 and the name 'Pound,' then '1234' is extracted.
Name	Extracts the name of the unit. For example, if a unit has the ID 1234 and the name 'Pound,' then the word 'Pound' is extracted.
Unit Base	Extracts the ID of the base unit for the unit. For example, if a unit has base unit 'm,' then the 'm' is extracted.
Unit Factor	Extracts the conversion factor that should be multiplied to a value with the unit to get the value in the base unit. For example, if a unit has a base unit conversion of 'value(g) = .001 * value(kg),' then '.001' is extracted.

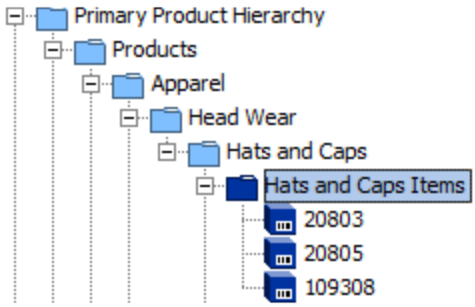
Aspect	Description
Unit Offset	Extracts the conversion factor that should be added to a value with the unit to get the value in the base unit. For example, if a unit has a base unit conversion of 'Value(K) = .556 * Value(°F) + 255.3722,' then '255.3722' is extracted.

Generic XML Sample Export

Export Manager requires that you select objects and provide both a sample template and mapping when exporting Generic XML.

Objects

Select the Objects to export.

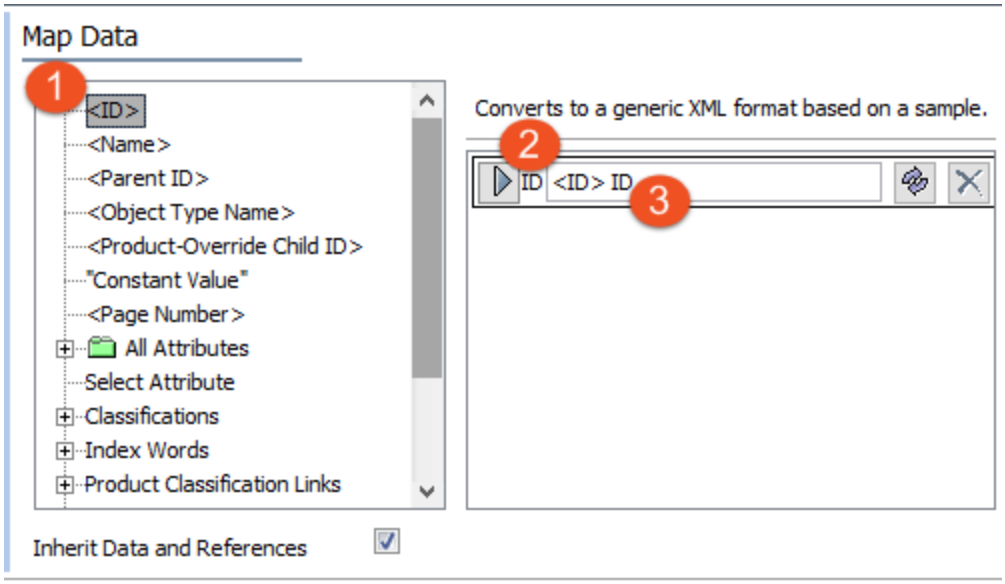


Template

Choose **Generic XML** for **Format** and provide the following template text in the Sample field.

```
    <Products>
       <Product>
          <?Record?>
          <ID><?Target?></ID>
       </Product>
    </Products>
```

Mapping



1. Available data source options are listed based on the object(s) selected.
2. Mapping targets are listed based on the template you provided.
3. Select a data source and click the mapping target arrow button to complete the mapping rule.

Results

After completing additional Export Manager fields, the previous settings will generate the following XML output document:

```
<Products>
  <Product>
    <ID>20803</ID>
  </Product>
  <Product>
    <ID>20805</ID>
  </Product>
  <Product>
    <ID>109308</ID>
  </Product>
</Products>
```

- the <Product>...</Product> tag section is repeated for each product exported from STEP
- the content of the <ID>...</ID> tag is based on the mapping rule

IDoc MATMAS 05 Format

The IDoc MATMAS 05 format is XML-based and is used when exchanging information with SAP. For more information, search the web.

Format Availability

IDoc MATMAS 05 is available for selection in:

- IIEP - refer to **Creating an Inbound Integration Endpoint**
- Import Manager - refer to **Creating a Data Import**
- Export Manager - refer to **Creating a Data Export**
- OIEP - refer to **Creating an Outbound Integration Endpoint**

Mapping

This format requires creating a data map between STEP and the data being processed, and may also include data transformations. For details, refer to **Data Mapping**.

Inbound Data

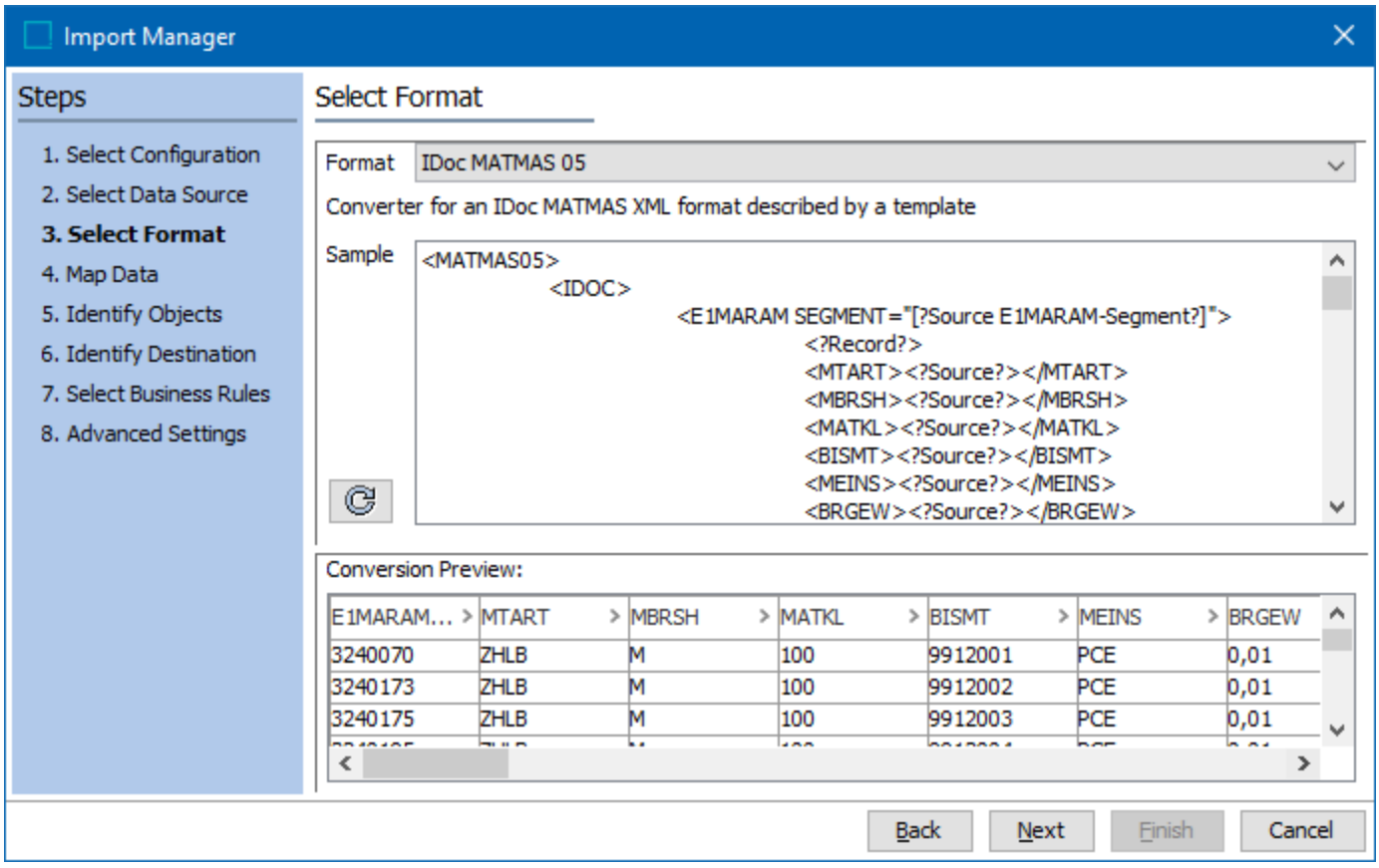
After importing the required SAP attributes into STEP, then use a source sample file from the SAP system to import values for the products. Mapping is required to identify where the values will live in STEP, and may also include data transformations.

Inbound Parameters

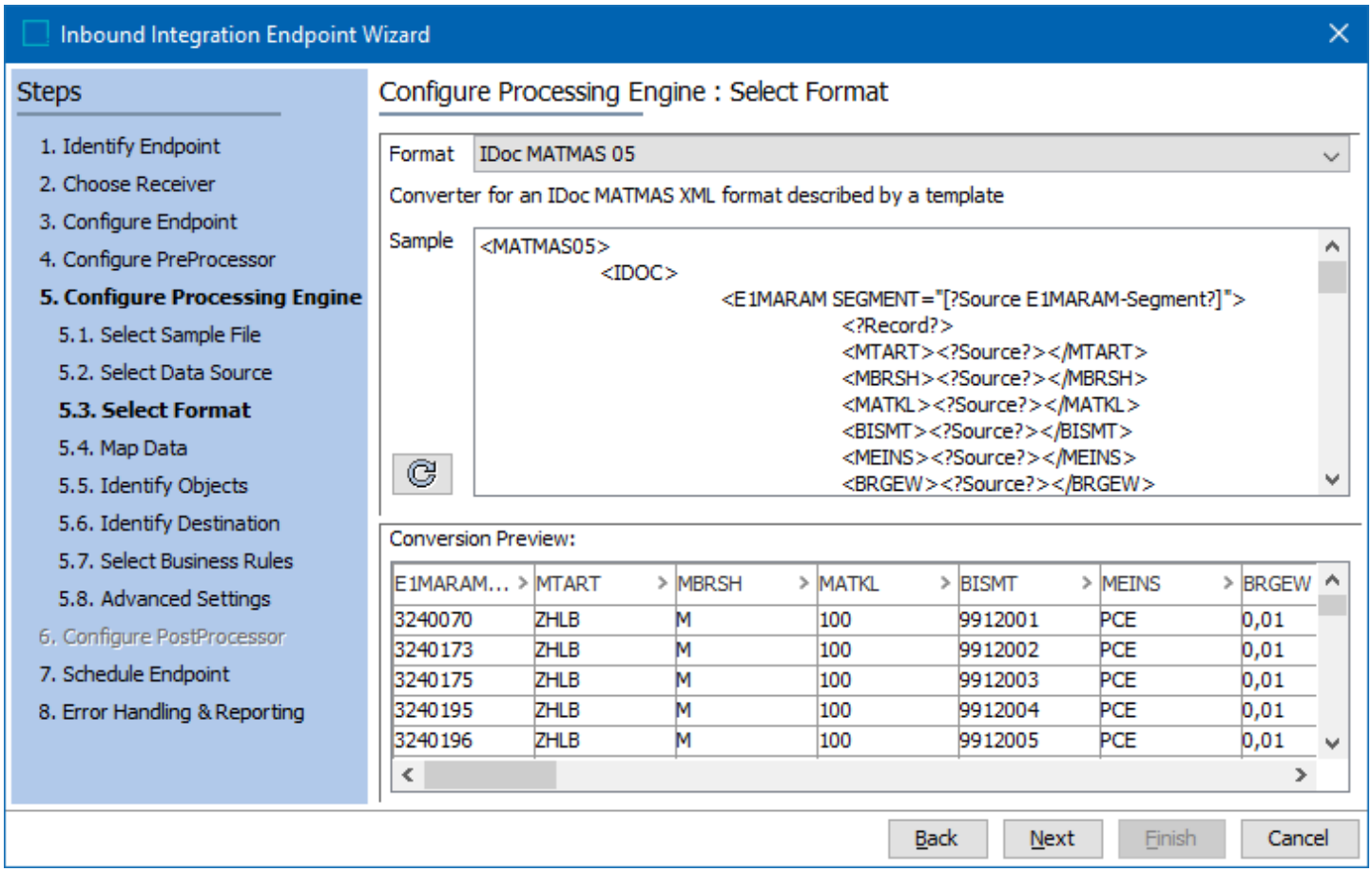
The following parameters are available in both Import Manager and IIEP.

- **Sample** - displays an import template based on the file selected on the Data Source step of the import wizard.
- **Conversion Preview** - displays values from the sample file in columns.

Import Manager



IIEP



Outbound Data

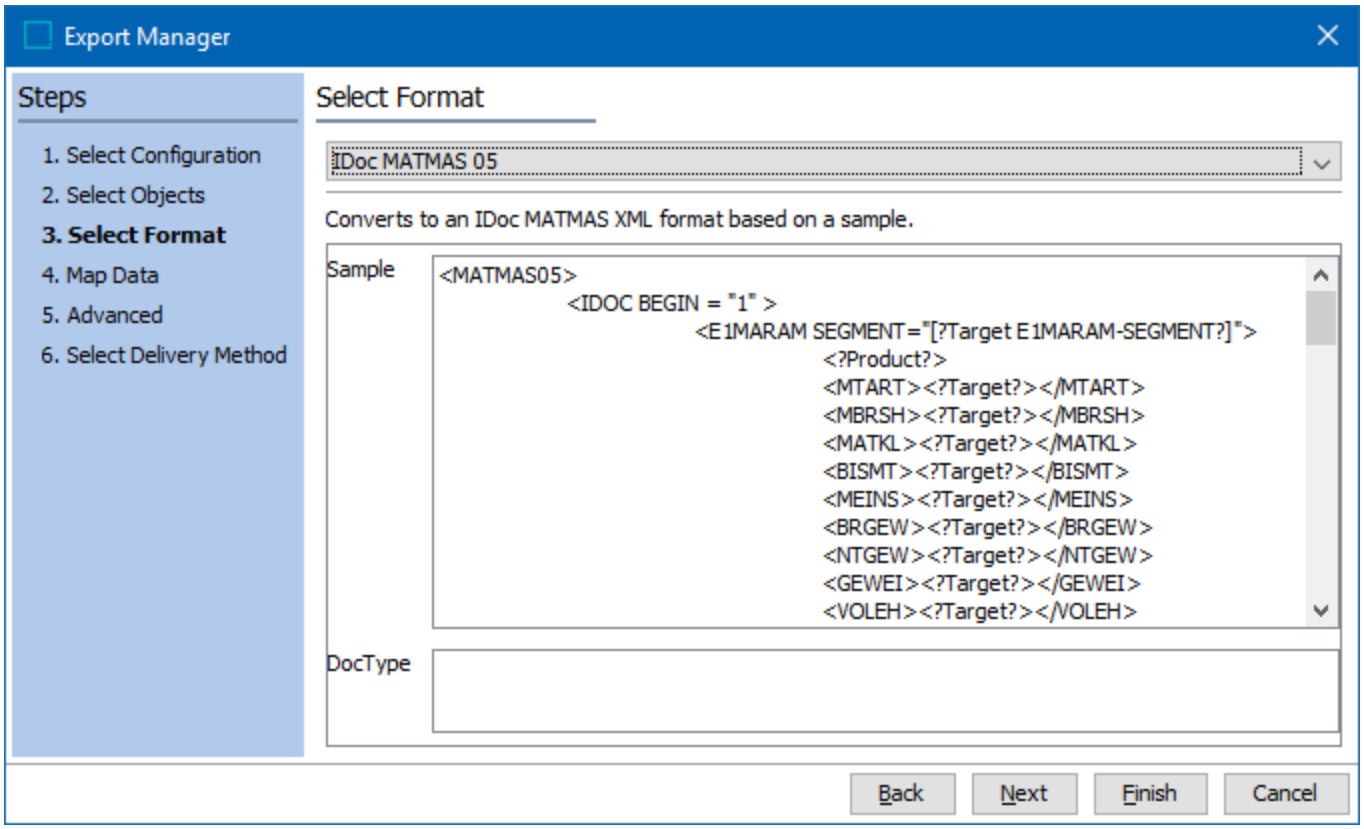
Mapping is required to identify where the values live in STEP, and may also include data transformations.

Outbound Parameters

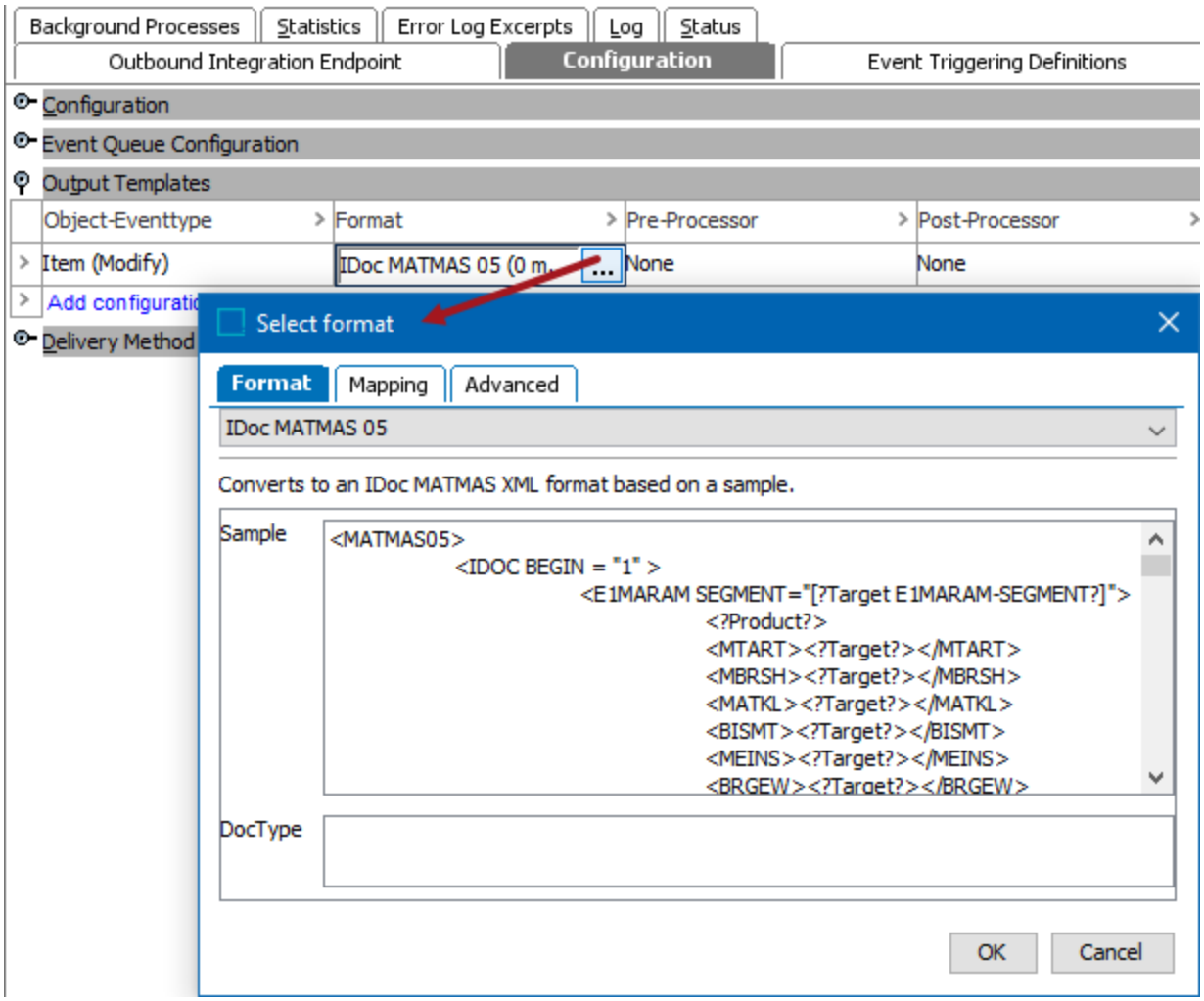
The following parameters are available in both Export Manager and OIEP.

- **Sample** - displays the standard template for export. This template may be modified to exclude attributes not required by your system. Since no undo functionality is available in the Sample field, it is good practice to use an external XML editor program for creating and editing a template. To test a template's validity for output, paste the XML into STEP.
- **DocType** - not valid for IDoc MATMAS 05 format

Export Manager



OIEP


Background Processes
Statistics
Error Log Excerpts
Log
Status
Outbound Integration Endpoint
Configuration
Event Triggering Definitions
Configuration
Event Queue Configuration
Output Templates
Object-Eventtype
Format
Pre-Processor
Post-Processor
Item (Modify)
IDoc MATMAS 05 (0 m...
None
None
Add configuratio
Delivery Method
Select format
Format
Mapping
Advanced
IDoc MATMAS 05
Converts to an IDoc MATMAS XML format based on a sample.
Sample
<MATMAS05>
<IDOC BEGIN = "1" >
<E1MARAM SEGMENT="[?Target E1MARAM-SEGMENT?]">
<?Product?>
<MTART><?Target?></MTART>
<MBRSH><?Target?></MBRSH>
<MATKL><?Target?></MATKL>
<BISMT><?Target?></BISMT>
<MEINS><?Target?></MEINS>
<BRGEW><?Target?></BRGEW>
DocType
OK
Cancel

Publication Excel Format

Publications can be exported and imported in an Excel format called Publication Excel. The Publication Excel format supports all publication types: standard Publisher (Adobe InDesign Integration) ('drag and drop'), Print Flatplanner, and Print AutoPage.

Format Availability

Publication Excel is available for selection in:

- IIEP - refer to **Creating an Inbound Integration Endpoint**
- Import Manager - refer to **Creating a Data Import**
- Export Manager - refer to **Creating a Data Export**

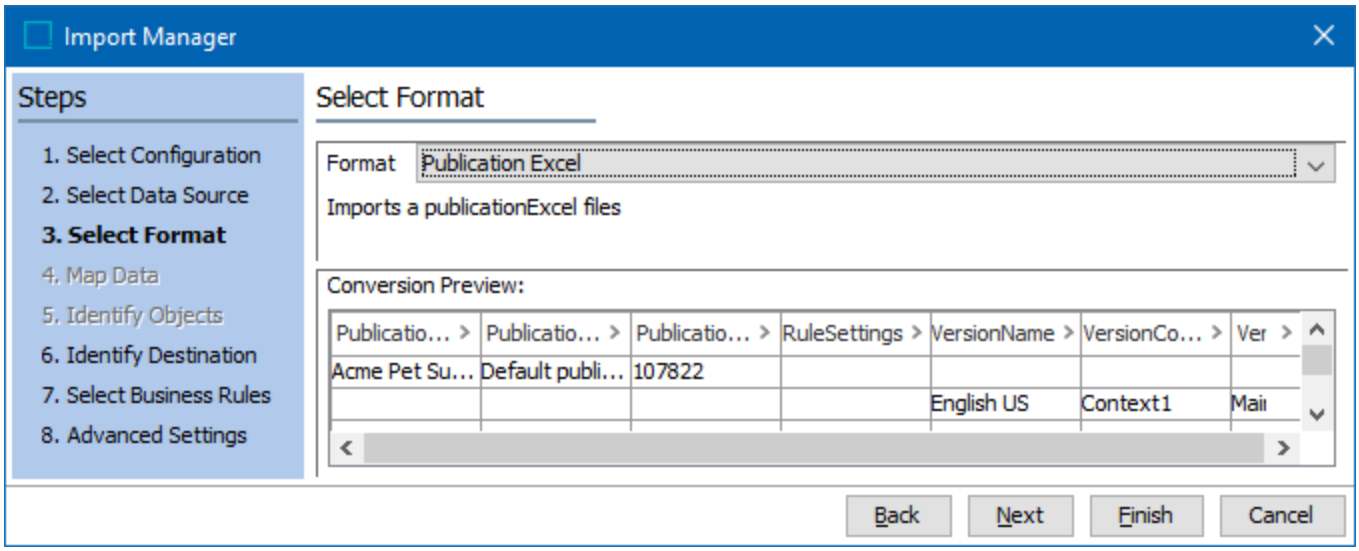
Mapping

Mapping data is not required and the Map Data step is disabled.

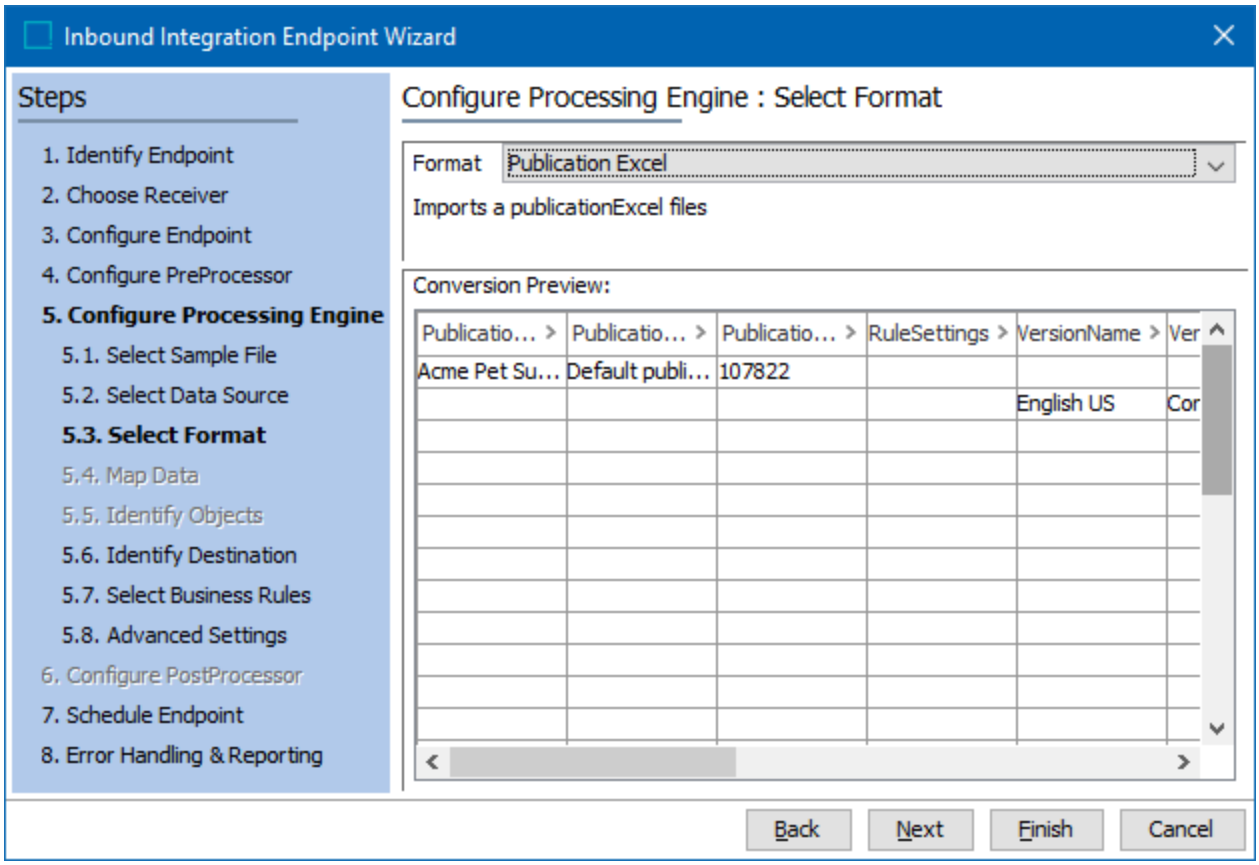
Inbound Data

For more information on the Publication Excel import process, refer to **Importing Publications in Excel**.

Import Manager



IIEP



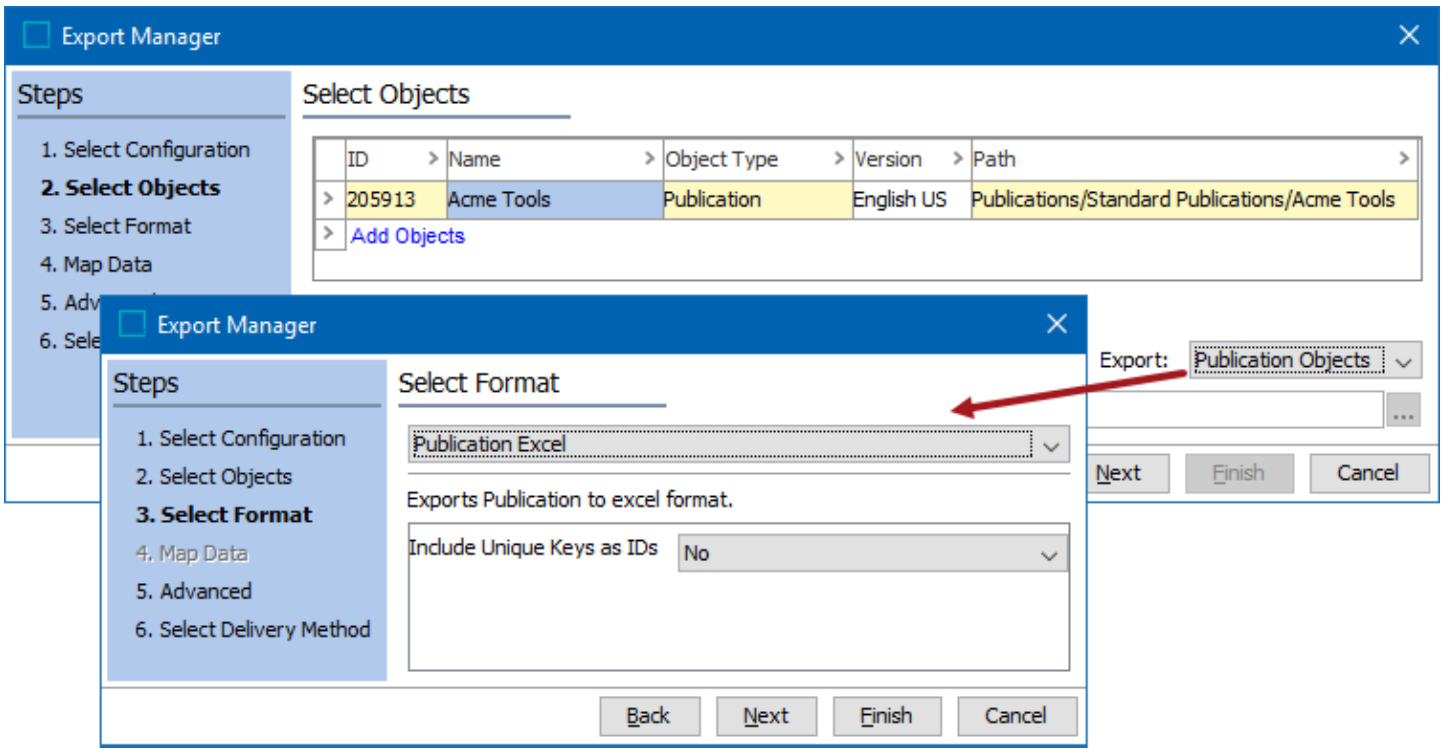
Outbound Data

For more information on the Publication Excel export process, refer to **Exporting Publications in Excel**.

For step-by-step instructions on how to export a Publication Excel or Flatplan Excel spreadsheet from a Flatplanner publication in STEP, refer to **Exporting a Publication Excel Sheet** in the **Publisher (Adobe InDesign Integration)** documentation.

Export Manager

After selecting a publication on Tree, right-click and choose **Export Data Below** to refer to the Publication Excel format option in Export Manager.


Export Manager
Steps
1. Select Configuration
2. Select Objects
3. Select Format
4. Map Data
Select Objects
ID
Name
Object Type
Version
Path
205913
Acme Tools
Publication
English US
Publications/Standard Publications/Acme Tools
Add Objects
Export:
Publication Objects
Next
Finish
Cancel
Export Manager
Steps
1. Select Configuration
2. Select Objects
3. Select Format
4. Map Data
5. Advanced
6. Select Delivery Method
Select Format
Publication Excel
Exports Publication to excel format.
Include Unique Keys as IDs
No
Back
Next
Finish
Cancel

SmartLabel Format

The SmartLabel format is based on Generic XML and includes a pre-defined template for integrators to provide a head start in configuring a SmartLabel™ export solution for both food and non-food product data. For more information, search the web.

The default template includes required SmartLabel attributes, plus category labels and comments to identify a repeating value group (RVG) section and a flattened style section. The default template should be modified to include only the data you require for a SmartLabel. The mapping process provides assistance in identifying and linking existing STEP attributes to the SmartLabel fields. Ultimately, the SmartLabel output is an XML document of the selected sections and mapped attributes.

Format Availability

SmartLabel is available for selection in:

- Export Manager - refer to **Creating a Data Export**
- OIEP - refer to **Creating an Outbound Integration Endpoint**

Mapping

This format requires creating a data map between STEP and the data being processed, and may also include data transformations. For details, refer to **Data Mapping**.

Outbound Data

Mapping is required to identify where the values live in STEP, and may also include data transformations.

Outbound Parameters

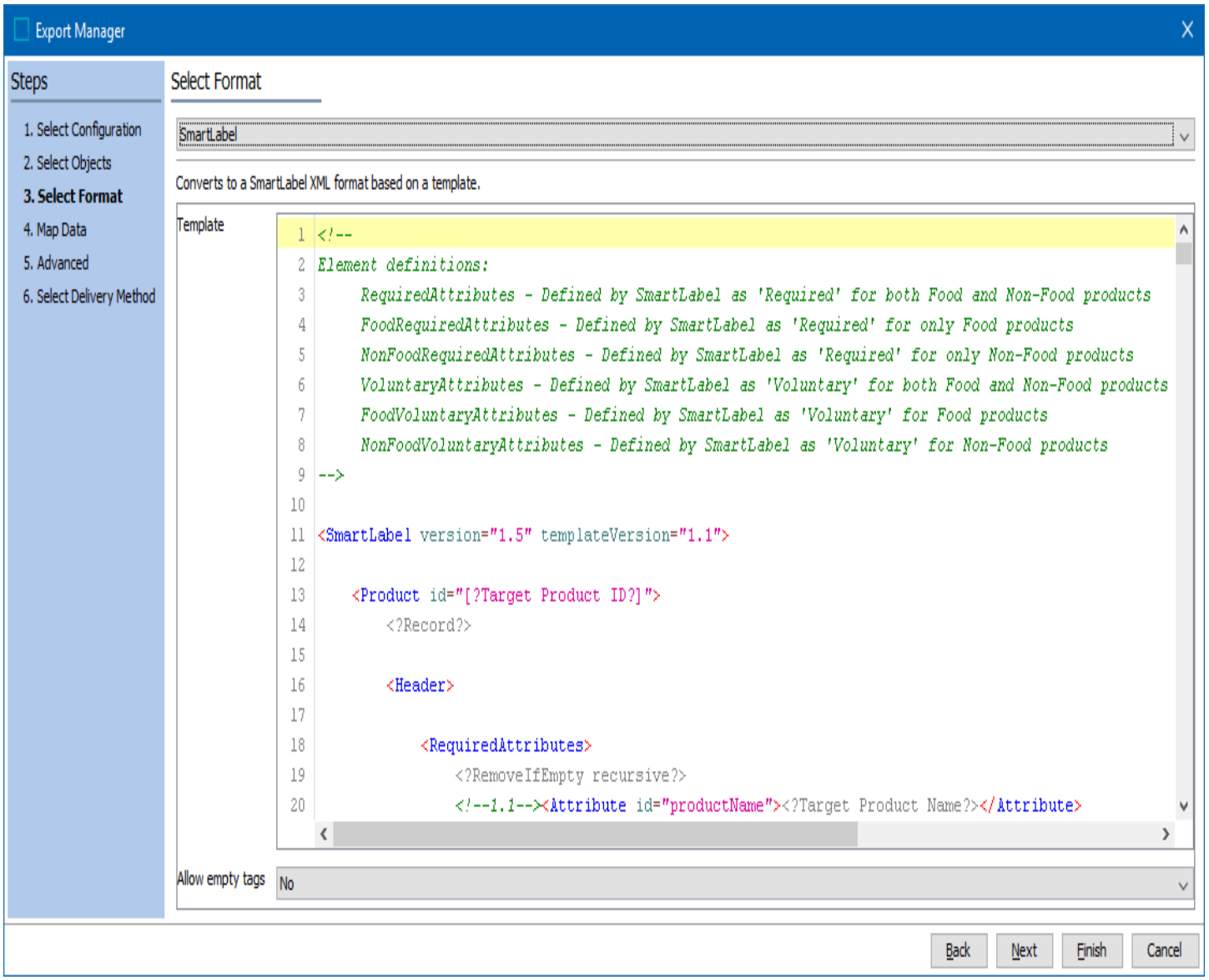
The following parameters are available in both Export Manager and OIEP.

- **Template** - displays the standard template for export each time a new export is created. The default template includes both food and non-food attributes, and should be modified to exclude attributes not required in your export. The modified template is saved with the export configuration or OIEP output template. For details on modifying the template to suit your data, refer to the **Configuring the SmartLabel Template** documentation.

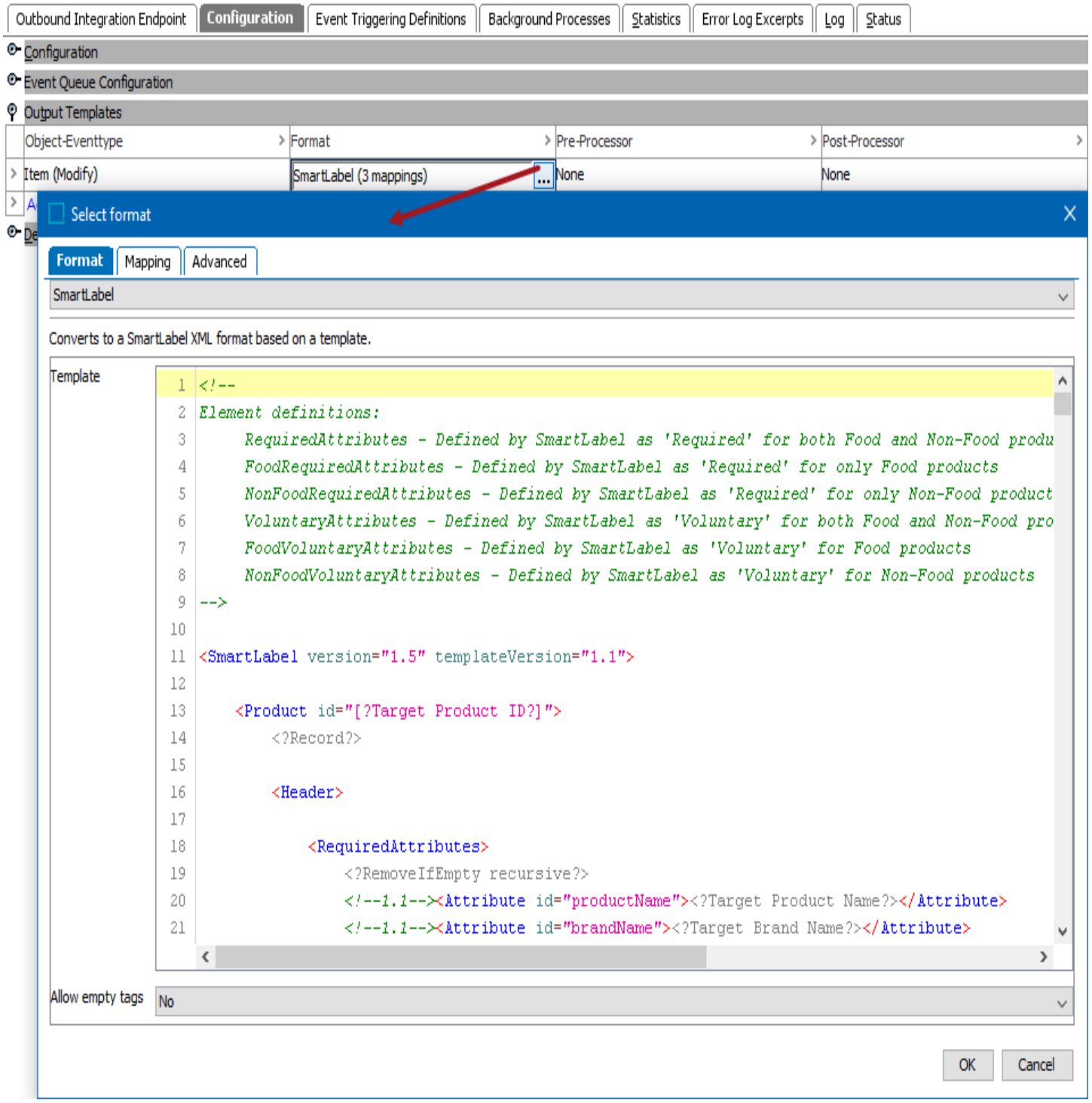
  'RemoveIfEmpty recursive' is applied to all sections to prevent unnecessary tags from being exported. For more information, refer to **RemoveIfEmpty Processing Instructions in Generic XML** in the **Generic XML Format** documentation.

- **Allow empty tags** - for details about this parameter, refer to the **Allow Empty Tags Parameter in Generic XML** documentation.

Export Manager



OIEP


Outbound Integration Endpoint
Configuration
Event Triggering Definitions
Background Processes
Statistics
Error Log Excerpts
Log
Status
Configuration
Event Queue Configuration
Output Templates
Object-Eventtype
Format
Pre-Processor
Post-Processor
Item (Modify)
SmartLabel (3 mappings)
None
None
Select format
Format
Mapping
Advanced
SmartLabel
Converts to a SmartLabel XML format based on a template.
Template
<!--
Element definitions:
RequiredAttributes - Defined by SmartLabel as 'Required' for both Food and Non-Food produ
FoodRequiredAttributes - Defined by SmartLabel as 'Required' for only Food products
NonFoodRequiredAttributes - Defined by SmartLabel as 'Required' for only Non-Food product
VoluntaryAttributes - Defined by SmartLabel as 'Voluntary' for both Food and Non-Food pro
FoodVoluntaryAttributes - Defined by SmartLabel as 'Voluntary' for Food products
NonFoodVoluntaryAttributes - Defined by SmartLabel as 'Voluntary' for Non-Food products
-->
<SmartLabel version="1.5" templateVersion="1.1">
<Product id="[?Target Product ID?]">
<?Record?>
<Header>
<RequiredAttributes>
<?RemoveIfEmpty recursive?>
<!--1.1--><Attribute id="productName"><?Target Product Name?></Attribute>
<!--1.1--><Attribute id="brandName"><?Target Brand Name?></Attribute>
Allow empty tags
No
OK
Cancel

Configuring the SmartLabel Template

Selecting the SmartLabel format in Export Manager or on an OIEP displays the default template, which includes required and voluntary attributes for both food and non-food products. This template is intended to be modified to include only the attributes you require and any additional comments to best meet your needs.

Additional comments can be added to your modified template as needed. The following text identifies comments:

```
<!-- shows where a comment begins
--> indicates the end of comment
```

The modified template is saved with the export configuration or in an OIEP output template. For information about saving a configuration for Export Manager, refer to **Running a Data Export**. For information about an OIEP output template, refer to **OIEP - Event-Based - Output Templates Flipper** or **OIEP - Select Objects - Output Templates Flipper**.

Note: To restore the default template once it has been modified, create a new export using Export Manager or a new output template using OIEP, and select the SmartLabel format.

SmartLabel Attributes

SmartLabel landing pages split the product attribution into the following sections: header, footer, attribute tabs, and sub-sections within attribute tabs. The export template uses these same groupings.

To make analysis and mapping easier, all attributes have been grouped into six possible categories. These indicate whether an attribute is required or voluntary for food and non-food products. Attribute groupings are identified within the XML by the following six elements:

- RequiredAttributes - Defined by SmartLabel as 'Required' for both Food and Non-Food products
- FoodRequiredAttributes - Defined by SmartLabel as 'Required' for only Food products
- NonFoodRequiredAttributes - Defined by SmartLabel as 'Required' for only Non-Food products
- VoluntaryAttributes - Defined by SmartLabel as 'Voluntary' for both Food and Non-Food products
- FoodVoluntaryAttributes - Defined by SmartLabel as 'Voluntary' for Food products
- NonFoodVoluntaryAttributes - Defined by SmartLabel as 'Voluntary' for Non-Food products

The default template values are meant to assist during the mapping process in STEP and the client SmartLabel solution. These values should be changed based on business requirements. It is not required that the values used exist as attribute names or IDs within STEP.

Required Attributes

For all required attributes, the default template includes the following predefined details:

1. SmartLabel Attribute Number - Defined in FOOD_ATTRIBUTES-Smartlabel_V_1-5 and NON-FOOD_ATTRIBUTES-Smartlabel_V_1-5.

2. Attribute id - Value that will be included in the export.
3. Attribute description - Value will be displayed during mapping process.

For example, the following image shows the required attribute 3.1 is included in the default template as a comment, along with the appropriate attribute ID and attribute description. The numbers correspond to the list above.

```
<FoodRequiredAttributes>
    <?RemoveIfEmpty recursive?>
    <!--3.1--><Attribute id="servingSize"><?Target Serving Size?></Attribute>
```

Voluntary Attributes

Although voluntary attributes have not been defined in the default template, a placeholder is included as well as the SmartLabel attribute numbers.

For example, a placeholder for voluntary attribute number 1.7 is included as a comment in the default template, but the ID and description are not provided.

```
<VoluntaryAttributes>
    <?RemoveIfEmpty recursive?>
    <!--1.7-->
</VoluntaryAttributes>
```

Modifying the SmartLabel Template

Modification to the template should be expected so that it will fit each unique data model. Attributes may need to be added, removed, or renamed. In addition, template changes may be required for attributes that have multiple values or exist on a referenced object.

Note: To quickly find the examples shown in this topic, it is helpful to display the unmodified default template in STEP, use Ctrl+A to select all text, and then paste it into an XML editor tool that includes a search feature.

MultiValues

The default template has been predefined with MultiTarget values based on GS1 attribute standards.

If an attribute has multiple values that will get sent to SmartLabel, MultiTarget must be used:

```
<Attribute id="ingredientPurpose"><?MultiTarget Ingredient Purpose?></Attribute>
```

If an attribute has a single value, Target would be used instead:

```
<Attribute id="ingredientPurpose"><?Target Ingredient Purpose?></Attribute>
```

Attributes on Repeating Referenced Objects

If attributes are to be captured from repeating referenced objects, a MultiTarget to the reference must be included in the template.

For example, the template includes this syntax on the ingredients sections, which can be found for other attribution as well.

```
<Ingredients>
    <?MultiTarget Ingredient Reference?>
    <!--Alternate 4.1--><Attribute id="ingredientSequence"><?Target Ingredient Sequence?></Attribute>
    <!--Alternate 4.1--><Attribute id="ingredientName"><?Target Ingredient Name?></Attribute>
    <!--4.8/4.9--><Attribute id="ingredientPurpose"><?MultiTarget FDA categories in 31 CFR - Ingredient Purpose?></Attribute>
</Ingredients>
```

In this example, if ingredient information was not stored on a referenced object then the <?MultiTarget Ingredient Reference?> section (indicated by the arrow above) would be removed and additional attribution added within the <Ingredients> tag as necessary.

Attributes on Nested Repeating Referenced Objects

If attributes are to be captured from nested repeating referenced objects, a MultiTarget to each reference must be added to the template.

An example of this is included in the template and can be used for other attribution.

```
<!-- Format used if multiple sets of serving/preparation information and/or nutrient details are stored on repeating referenced objects
<Nutrient>
    <?RemoveIfEmpty recursive?>
    <?MultiTarget Nutrient Info Reference?>
    <!-3.1-><Attribute id="servingSize"><?Target Serving Size?></Attribute>
    <!-3.2-><Attribute id="numberOfServingsPerPackage"><?Target Number of Servings Per Package?></Attribute>
    <!-3.9-><Attribute id="nutritionalReferalOrDisclosure"><?Target Nutrition Referal or Disclosure Statement?></Attribute>
    <!-3.10-><Attribute id="preparationType"><?Target Preparation Type?></Attribute>

    <NutrientDetail>
        <?RemoveIfEmpty recursive?>
        <?MultiTarget Nutrient Detail Reference?>
        <!-3.3/3.4-><Attribute id="measurementPrecision"><?Target Measurement Precision?></Attribute>
        <!-3.3/3.5-><Attribute id="nutrientType"><?Target Nutrient Type?></Attribute>
        <!-3.3/3.6-><Attribute id="quantityContained"><?Target Quantity Contained?></Attribute>
        <!-3.3/3.7-><Attribute id="dailyValuePercent"><?Target Daily Value Percent?></Attribute>
    </NutrientDetail>
</Nutrient>
-->
```

In this example, data is stored on a repeating ‘Nutrient Info’ reference and additional data nested below this on a repeating ‘Nutrient Detail’ reference. To use this example section, the closing comment tag around the nutrient section (indicated by the arrow on the last row) should be moved to the end of first line of the section.

Additionally, the comment tags surrounding the SmartLabel attribute numbers are incomplete (two dashes are required) and must be modified as required.

More nested levels can be added following the same formatting.

Empty Tags

The default template includes <?RemoveIfEmpty recursive?> defined above all attribute groups. This will ensure no tags for empty attributes or attribute groups are included in the export. If empty tags are required, then this should be placed within specific attribute groups or removed all together from the template.

STEPXML Format

STEPXML is the native XML format for STEP. Where other supported import / export formats are limited to specific super types and node types, STEPXML makes it possible to import / export the majority of the STEP configuration and data objects. STEPXML is especially suited for transferring data and configurations between systems in a DTAP (Development, Testing, Acceptance, and Production) environment and for inbound integrations with systems / middleware capable of producing STEPXML.

As illustrated below, all data that enters and exits STEP via any inbound / outbound functionality uses the STEPXML format. For example, when a CSV file is selected for import, STEP converts the data to STEPXML and then imports the data.

For users familiar with XML file structure and creation, the Advanced STEPXML format provides a template parameter where an XML can be supplied manually. For more information, refer to the **Advanced STEPXML Format** topic.

Limitations

The following limitations exist with STEPXML:

- Historical data (data in old revisions) cannot be exported or imported.
- Asset content (the binary files associated with Asset objects) can be exported but not imported.
- Table data can be exported but not imported.
- Individual values for existing multivalued attributes cannot be exported or imported.
- Legal source object types and legal target object types for existing reference / link type definitions cannot be removed or added.
- Legal units for existing attributes cannot be removed.

Note: When using the empty values export (which is intended for use with external systems) included data containers, references, and links that remain empty / undefined cannot be imported to STEP without generating errors.

Format Availability

STEPXML is available for selection in:

- IIEP - refer to **Creating an Inbound Integration Endpoint**
- Import Manager - refer to **Creating a Data Import**
- Export Manager - refer to **Creating a Data Export**
- OIEP - refer to **Creating an Outbound Integration Endpoint**

Mapping

Since STEPXML is the native format for STEP, mapping data is not required and the Map Data step is disabled.

Inbound Data

STEPXML can contain a number of different processing instructions for import. Some of these instructions are similar to the options available in the import wizard for the tabular formats (non-XML formats like Excel, CSV, and FixedWidth).

When updating most objects via import, the STEP ID is required. However, for products, entities, classifications, and assets, a key can also be used for identification. For more information about keys, refer to the **Unique Keys** topic of the **System Setup** documentation.

It is only necessary to include the data needing to be updated. By default, existing data that is not included in the import file will not be deleted or modified.

Deleting Values During Import

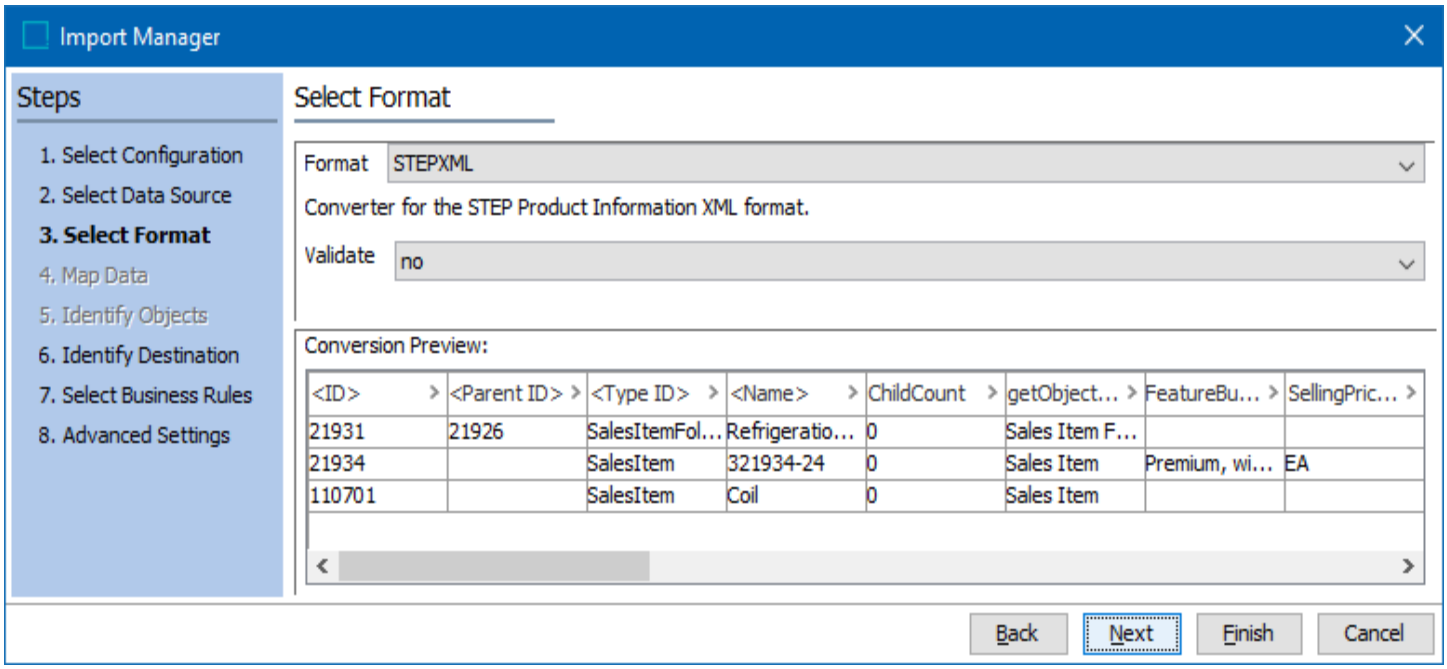
When importing STEPXML files, the values are imported exactly as provided in the file for the specified attributes. This means that when a blank value is imported, an attribute that previously had a value is overwritten as blank. If the value being deleted was inherited, the result is not a blank field, but the inherited value is restored.

This functionality is the same as when importing CSV files, but differs from imports of Excel.

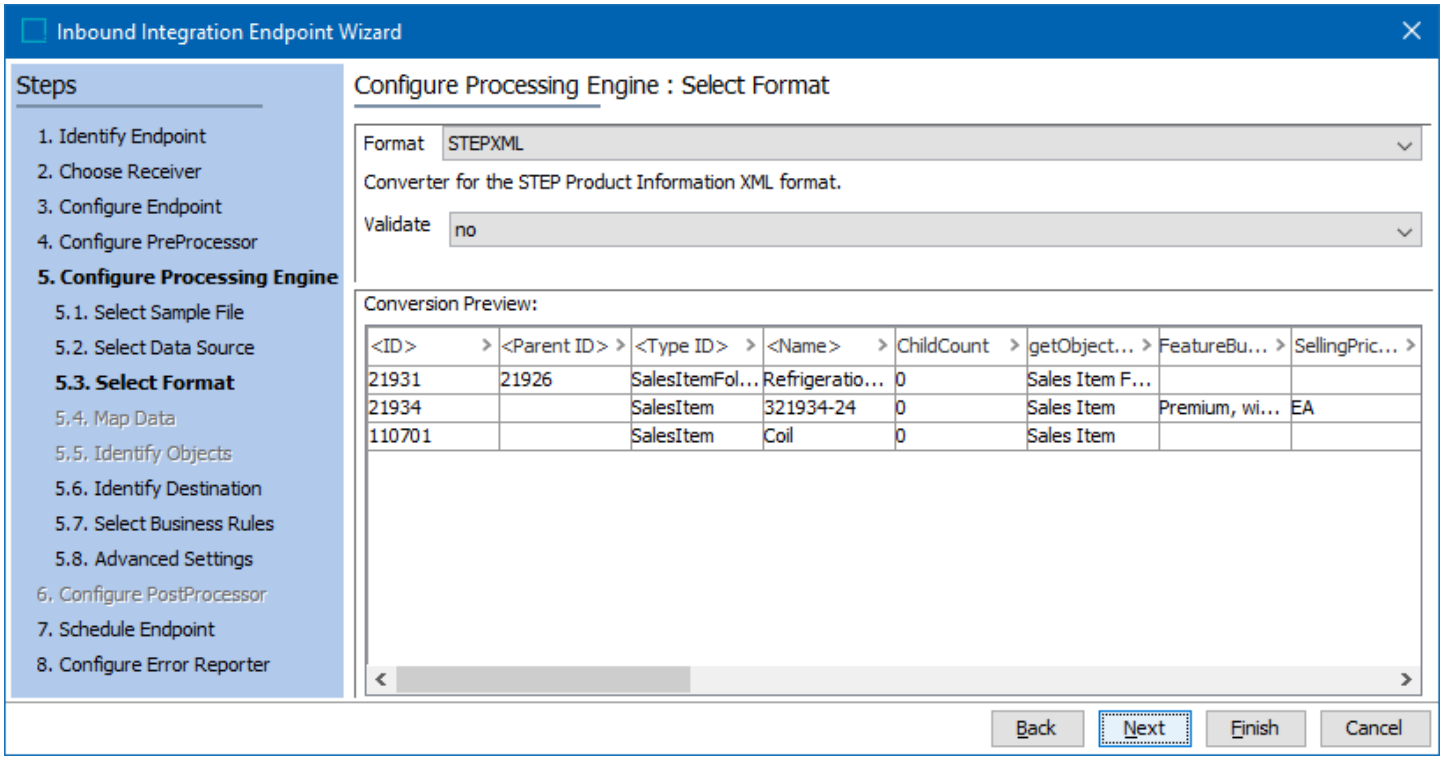
Inbound Parameters

An inbound file can create new objects, update existing objects, and delete existing values and links, among other options. For more information, refer to the **STEPXML Tags and Examples** topic.

Import Manager



IIEP



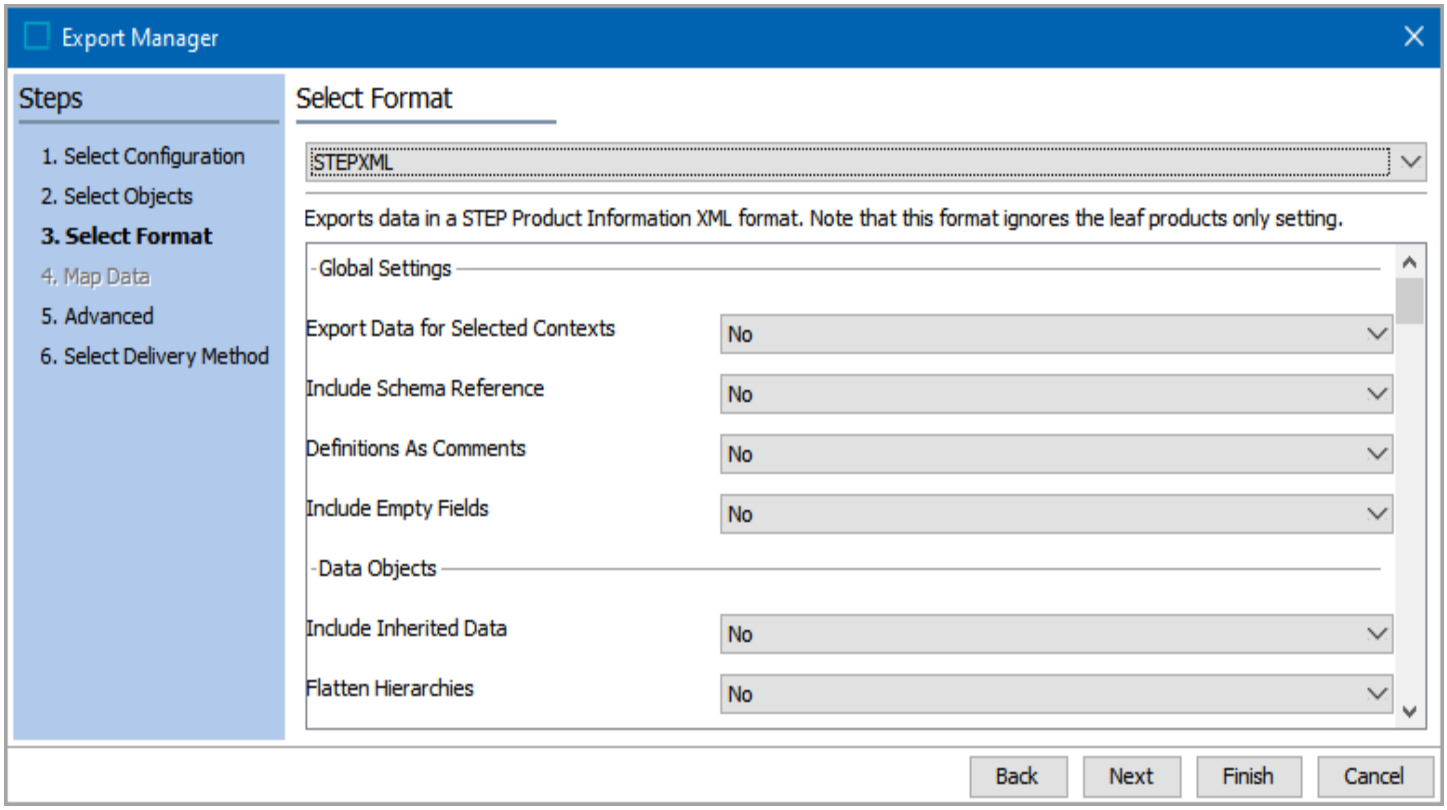
Outbound Data

The STEPXML format uses a number of parameters to determine the data included or excluded for the export file. While the output format is the same as using Advanced STEPXML, these parameters eliminate the need for creating an XML template.

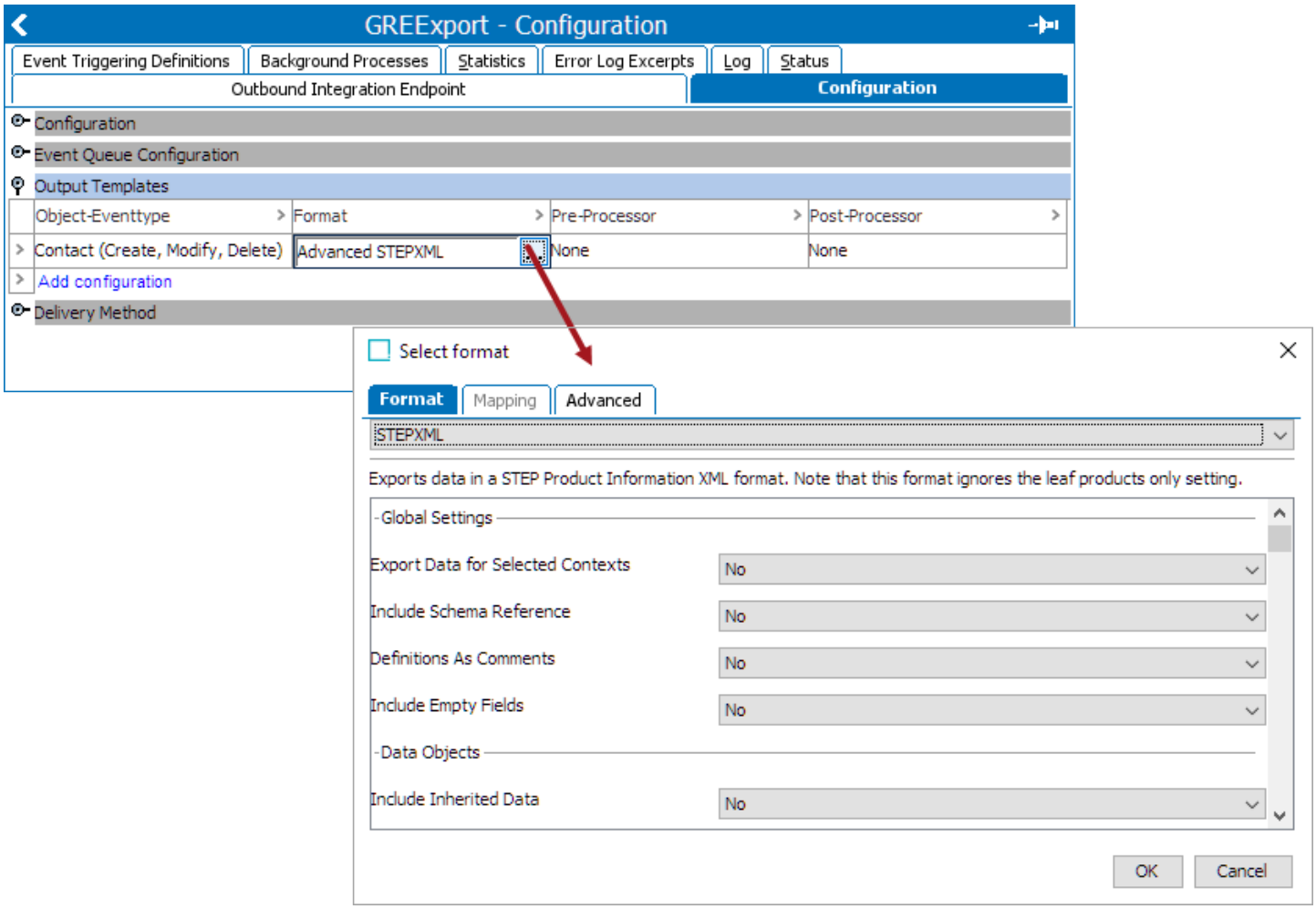
Outbound Parameters

All parameters available via STEPXML are also available in Advanced STEPXML, which includes additional options. An export using the STEPXML format generates a file that can be used as a template for additional imports or exports using Advanced STEPXML. For details about each STEPXML parameter, refer to the **STEPXML Outbound Parameters** topic.

Export Manager



OIEP


GREExport - Configuration
Event Triggering Definitions
Background Processes
Statistics
Error Log Excerpts
Log
Status
Outbound Integration Endpoint
Configuration
Configuration
Event Queue Configuration
Output Templates
Object-Eventtype
Format
Pre-Processor
Post-Processor
Contact (Create, Modify, Delete)
Advanced STEPXML
None
None
Add configuration
Delivery Method
Select format
Format
Mapping
Advanced
STEPXML
Exports data in a STEP Product Information XML format. Note that this format ignores the leaf products only setting.
Global Settings
Export Data for Selected Contexts
No
Include Schema Reference
No
Definitions As Comments
No
Include Empty Fields
No
Data Objects
Include Inherited Data
No
OK
Cancel

STEPXML Import and Export Options

The native STEPXML format can be used to import and export data and data types. The STEPXML format uses parameters to generate STEPXML template. This allows users with limited XML knowledge to create a STEPXML template, which can be further modified with tags not included as a parameter option.

All STEPXML parameters are represented by Advanced STEPXML tags. As indicated in the table below, some tags are explained in the linked online help topics. All Advanced STEPXML tags are fully documented in an XSD (XML Schema Definition) file linked from the **STEP API Documentation** within the **STEPXML** section, available at [system]/sdk or accessible from Start Page.

General information about using the tools, formats, and the parameters or tags are included in these topics:

- **STEPXML Outbound Parameters** - many STEPXML options are available as parameters in the Export Manager and an OIEP
- **STEPXML Tags and Examples** - Advanced STEPXML tags can be included in an XML template in the Export Manager and an OIEP
- **Creating a Data Import** - files created with STEPXML and Advanced STEPXML formats can be processed by Import Manager
- **Creating an Inbound Integration Endpoint** - files created with STEPXML and Advanced STEPXML formats can be processed by an IIEP

Description	STEPXML format Export Manager Parameter	Advanced STEPXML format Tag for Import and/or Export
Actions applied to each action set	Include Action Sets	
Actual page data details	Include Actual Page Data	
Asset contents along with the asset text and references	Include Asset Content	AssetContent
Asset description attributes and meta attributes applied	Include Assets	Assets AssetCrossReference ReplacementRules

Description	STEPXML format Export Manager Parameter	Advanced STEPXML format Tag for Import and/or Export
Asset push configurations	Include Asset Push Configurations	AssetPushConfiguration
Asset push event queues	Include Asset Push Event Queues	
Assets saved as bulk update	Include Bulk Update Configurations	
Attribute groups, the view definitions and settings applied to attribute groups	Include Attribute Groups	ReplacementRules
Attribute transformations and configurations	Include Attribute Transformations	
Attributes, the validation base types, dimension dependencies applied units, applied LOV hierarchy filters, calculated templates, etc.	Include Attributes	ReplacementRules
AutoPage references	Include Autopage References	
Classifications, the meta attributes applied, attribute links and references from classifications	Include Classifications	Classifications AttributeLink ClassificationReference ReplacementRules
Collections groups, collections, and search URLs; excludes products, classifications, or assets included in a collection	Include Collection Definitions	

Description	STEPXML format Export Manager Parameter	Advanced STEPXML format Tag for Import and/or Export
Comments (object definitions)	Definitions as Comments	
Component model ID, name, object types, and links	Include Component Models	
Context	Export Data for Selected Contexts	Qualifiers
Context qualifiers (for product, classification, or asset data), qualifier IDs, and dimension point IDs	Include Context Qualifiers	
Data container types	Include Data Container Definitions	DataContainerTypes
Data containers when exporting entities	Include Data Containers	DataContainer and
Derived Event Types (created in System Setup)	Include Derived Event Types	
Dimensions, dimension points, contexts, applied dimension points, and locale settings	Include Contexts	
eCatalog configurations and price lists; excludes product selection lists	Include eCatalogs	
Entity hierarchy, meta attributes applied, and entity references	Include Entities	ReplacementRules
Entity values	Include Entity Attribute Values	

Description	STEPXML format Export Manager Parameter	Advanced STEPXML format Tag for Import and/or Export
Event processor ID, name, links, and configuration	Include Event Processors	
Event queues, applied message templates, output formatting, and triggering definitions	Include Event Queues	
Export configuration assets	Include Export Configurations	
Flatplanner basket details	Include Baskets	
Global business rules and libraries, the ID, name, links, on approve setting, dependencies, valid object types, and templates	Include Business Rules (Global) and Libraries	
Hierarchies	Flatten Hierarchies	
Image conversion configurations	Include Image Conversion Configurations	ImageConversionConfigurations
Import configuration assets	Include Import Configurations	
Inherited values and references	Include Inherited Data	IncludeInherited
Integration endpoint ID, name, and configurations	Include Integration Endpoints	
LOV definitions and values in LOVs, applied meta attributes, validation base types, and applied units	Include List of Values	

Description	STEPXML format Export Manager Parameter	Advanced STEPXML format Tag for Import and/or Export
Match code ID, name, links, setup group, valid entity hierarchies, and valid object types	Include Match Codes	
Matching algorithm ID, name, links, and configuration	Include Matching Algorithms	
Object types: • User-created object types (below the 'Setup Group type root' node) • System object types • System Link types • System reference types	Include Link, Reference and Object Types	
Pagination rule details	Include Pagination Rules	
Planned pages	Include Planned Pages	
Product hierarchy, meta attributes, attribute links, attribute values, and references from products	Include Products	Products AttributeLink (for products and product overrides) FilterUserType ProductCrossReference ReplacementRules SequenceProduct and IncludeParent
Product overrides	Include	

Description	STEPXML format Export Manager Parameter	Advanced STEPXML format Tag for Import and/or Export
	Overridden Products	
Product values	Include Product Attribute Values	Values
Publication groups	Include Publication Groups	
Publication sections	Include Publication Sections	
Publications	Include Publications	
Setup entity ID, name, links, and configuration	Include Setup Entities	
Setup groups (defined in system setup) that hold integration endpoints, Web UI configurations, STEP workflows, and business rules	Include Setup Groups	
Style tags, character tags, special characters, footnotes, and hyperlinks, rendering, short cuts, and output formatting	Include Tags	
Table row type definitions, column type definitions, and table type definitions, transformations	Include Table Types	
Table row type definitions, column type definitions, and table type definitions, transformations	Include Table Definitions	
Tables (as they appear in the preview tab in STEP); excludes table transformations and commercial data	Include Tables	

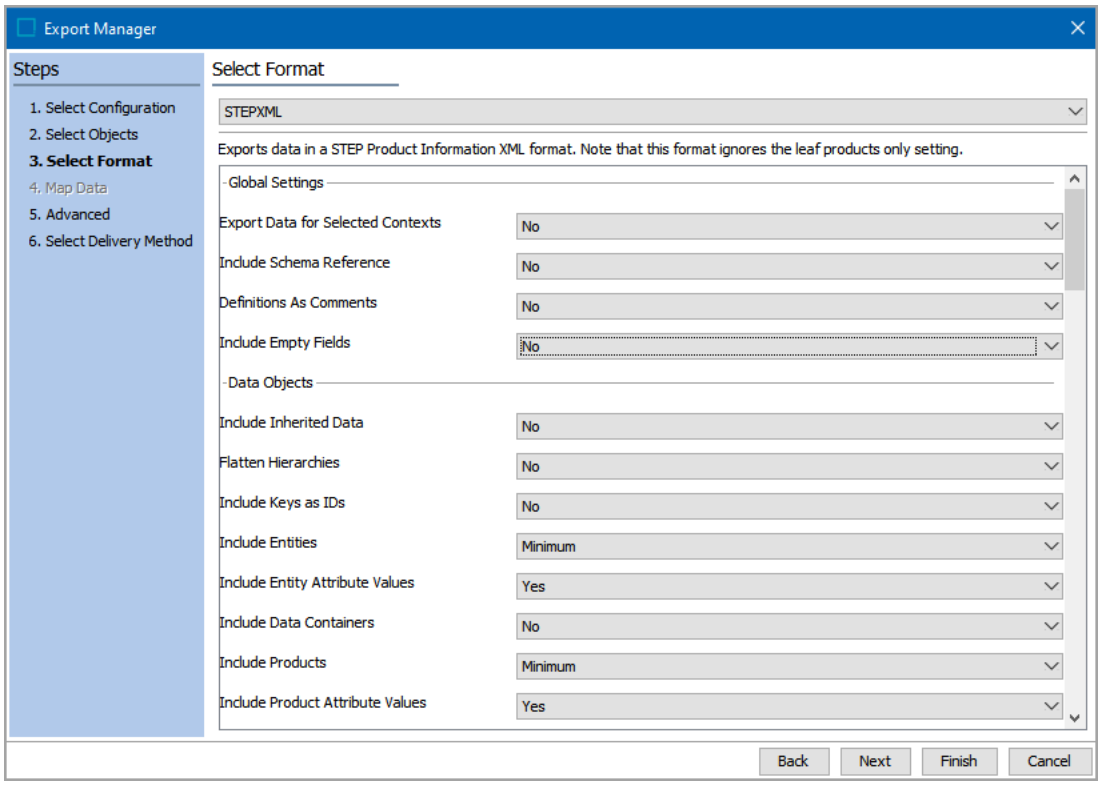
Description	STEPXML format Export Manager Parameter	Advanced STEPXML format Tag for Import and/or Export
Transformation lookup table configuration assets	Include Transformation Lookup Tables	
Translation configurations	Include Translation Configurations	
Unique Key definitions	Include Key Definitions	
Unique Keys	Include Keys as IDs	IncludeKey
Unit groups and units, meta attributes, values, and base unit conversions	Include Units	
User groups and users, applied privileges, meta attributes, restricted GUI setup, etc.; excludes user passwords	Include Users and User Groups	ReplacementRules
Users and Groups root node default settings: ○ Image & Document Settings flipper > Dimension Dependencies ○ Calculated Attribute Settings flipper > Dimension Dependencies ○ Product Information Manager Default Settings flipper ○ Flatplanner Default Settings flipper ○ DTP Default Settings flipper ○ GDSN Default Settings flipper ○ Terms List Settings flipper ○ WebServices Default Settings flipper ○ Web UI Default Settings flipper ○ Default Reference Type of Primary Image	Include System Settings	ReplacementRules

Description	STEPXML format Export Manager Parameter	Advanced STEPXML format Tag for Import and/or Export
flipper ○ Table Default width and height flipper		
Web UI configurations	Include Web UI Configurations	
Workflow ID, name, links, valid object types, and configuration	Include Workflows	
Workflow profiles, number of exceeded deadlines, how long the tasks have been assigned to the assignees, workflow throughput for the month versus the last six months, etc.	Include Workflow Profiles	
Workflow status flags	Include Status Flags	
Workflow task information for objects with active tasks	Include Workflow Tasks	

STEPXML Outbound Parameters

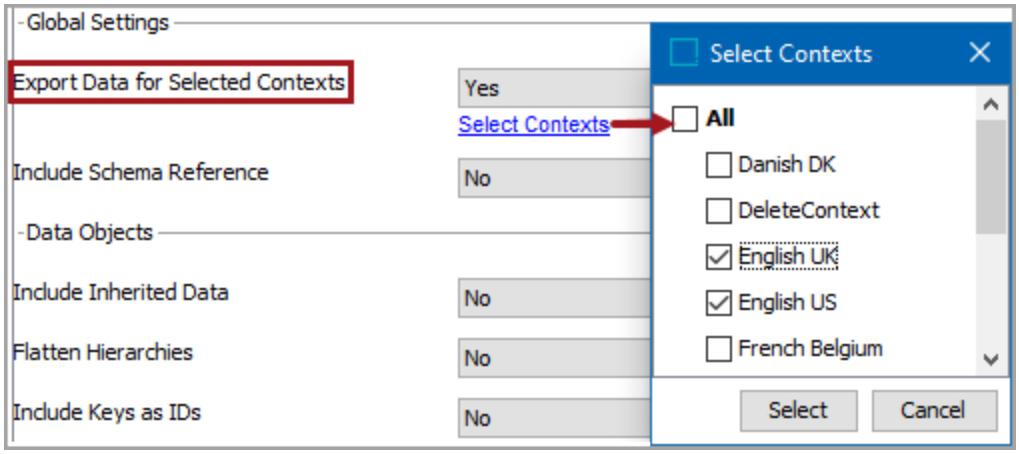
The following parameters are available when exporting using the STEPXML format and are separated into the following categories:

- **Global Settings** are applied to both data objects and configuration objects.
- **Data Objects** are applied to objects that hold data, with the most commonly used options at the top, and the others listed in groups of similar options. For example, the classification-related options are listed together.
- **Configuration** options are listed in alphabetical order, and are applied to the item described in the label.
- **Publishing** options are typically used for a print application.

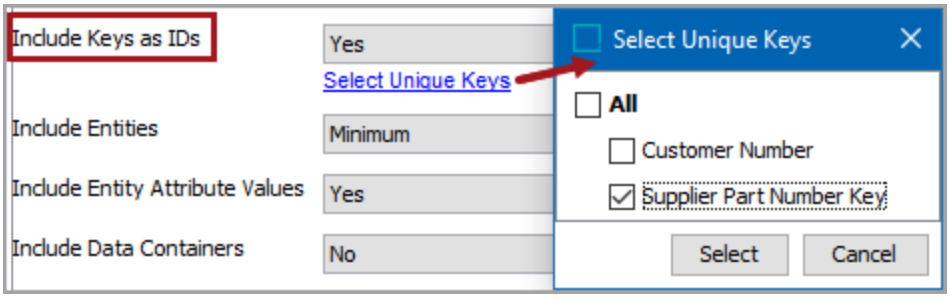


Global Settings

Global Settings Parameter	Description
Export Data for Selected Contexts	Select **No** to export only for the current context. Select **Yes** to export multiple contexts and display the **Select Contexts** link. Click the link to display the Select Contexts dialog. Select the necessary contexts and click the **Select** button.

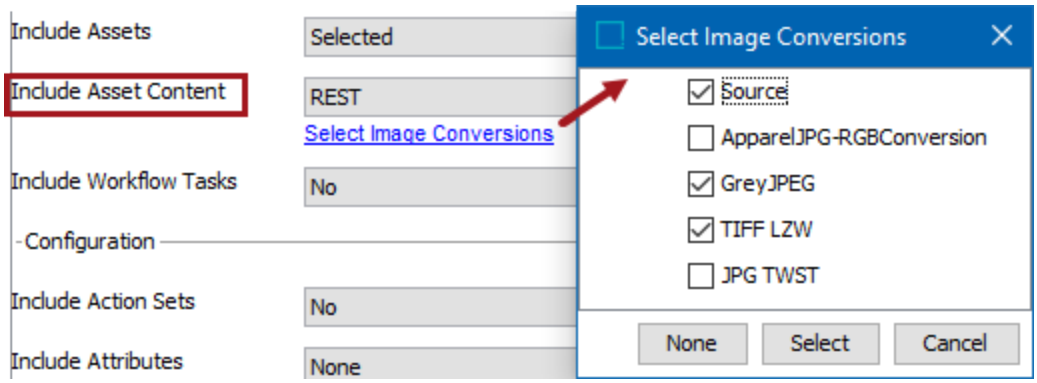
Global Settings Parameter	Description
	 For more information, refer to the Contexts topic of the System Setup documentation or the Context Data (Qualifiers) in STEPXML topic.
Include Schema Reference	Select **No** or **Yes** to output a reference to the schema file. For more information about schema, access the **STEP API Documentation** at [system]/sdk or from the Start Page.
Definitions as Comments	Select **No** or **Yes** to output a object definitions as comments. For more information, refer to the Configuration Management topic.
Include Empty Fields	Select **No** or **Yes** to include empty fields in the export. This is set to 'No' by default. If 'Yes' is selected, and attribute values and/or data containers and/or references or links (including metadata) are included in the STEPXML configuration, then the export will include all valid and linked data, whether the exported object has a value or not. **Note:** STEPXML is intended for use with external systems for integration purposes. Exported empty values are not intended to be imported into STEP.

Data Objects

Data Objects Parameter	Description
Include Inherited Data	Select **No** or **Yes** to control the output of inherited values and references. For more information, refer to the Export Inheritance in STEPXML Example topic in the Data Exchange documentation.
Flatten Hierarchies	Select **No** to output STEPXML in a nested layout. Select **Yes** to output a non-nested layout, where values and references localized on parent objects are included on each object below the parent. This option does not analyze the validity of the attributes. Values localized on parent objects will be exported and included on each exported leaf object. For more information, refer to the Object Type Hierarchy topic and related topics in the Getting Started documentation.
Include Keys as IDs	Select **No** to ignore keys. Select **Yes** to use keys instead of STEP IDs and display the **Select Unique Keys** link. Click the link to display the 'Select Unique Keys' dialog. Select the necessary keys, and click the **Select** button.  For more on keys, refer to the Unique Keys topic in the System Setup documentation. For an Advanced STEPXML template, this option is included in the IncludeKey Tag. For more information, refer to the IncludeKey Tag in STEPXML topic in the Data Exchange documentation.
Include Entities	For an entity hierarchy, meta attributes applied, and entity references are exported as follows: • Select **All** to output all entities and entity references.

Data Objects Parameter	Description
	• Select **Minimum** or **Selected** to output entities based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **Referenced** to output all entities in the hierarchy, and entities referenced from entities in the hierarchy. • Select **None** to output no entities or entity references. For more information, refer to the Entities topic in the Getting Started documentation.
Include Entity Attribute Values	Select **No** or **Yes** to control output of entity values. For more information, refer to the Entities topic in the Getting Started documentation.
Include Data Containers	Select **No** or **Yes** to control output of data containers when exporting entities. For more information, refer to the Data Containers topic in the System Setup documentation.
Include Products	For a product hierarchy, the meta attributes applied, attribute links, attribute values, and references from products are exported as follows: • Select **All** to output all products and product references. • Select **Minimum** or **Selected** to output entities based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **Referenced** to output all products in the hierarchy, and products referenced from products in the hierarchy. • Select **None** to output no products or product references. For more information, refer to the Products topic in the Getting Started documentation.
Include Product Attribute Values	Select **No** or **Yes** to control output of product values. For more information, refer to the Products topic in the Getting Started documentation.
Include Overridden Products	Select **No** or **Yes** to control output of product overrides. For more information, refer to the Product Overrides topic in the Getting Started documentation.

Data Objects Parameter	Description
Include Classifications	For classifications, the meta attributes applied, attribute links and references from classifications are exported as follows: • Select **All** to output all classifications and classification references. • Select **Minimum** or **Selected** to output entities based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **Referenced** to output all classifications in the hierarchy. • Select **None** to output no classifications or classification references. For more information, refer to the Classifications topic in the Getting Started documentation.
Include Tables	Select **No** or **Yes** to control output of STEP tables as they appear in the preview tab in STEP. Table transformations and commercial data are not included in the exported file. Tables cannot be imported. For more information, refer to the Tables topic.
Include Table Definitions	Select **No** or **Yes** to control output of tables, row type definitions, column type definitions, and table type definitions, including applied transformations are exported as follows: For more information, refer to the Tables topic.
Include Assets	For assets, the description attributes and meta attributes applied are exported as follows: • Select **All** to output all assets and activate the 'Include Asset Content' parameter. • Select **Minimum** or **Selected** to output entities based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no asset text, references, nor digital contents. For more information, refer to the Assets topic in the Getting Started documentation. To export asset content, use the 'Include Asset Content' parameter below. For an Advanced STEPXML template, this option is included in the Assets Tag. For more information, refer to the Assets Tag in STEPXML topic in the Data Exchange documentation.
Include Asset Content	Requires that the 'Include Assets' parameter is set to All, Selected, or Minimum. Select **None** to exclude asset contents.

Data Objects Parameter	Description
	Select **Binary** or **REST** to export digital asset contents along with the asset text and references, and display the **Select Image Conversions** link. Both options require a selection of at least one Image Conversion. Click the link to display the Select Image Conversions dialog. Select the necessary conversions and click the **Select** button.  For details about setting up export of asset content, refer to the Assets and Content with STEPXML topic in the Digital Assets documentation. For more information, refer to the Assets topic in the Getting Started documentation.
Include Workflow Tasks	Select **No** or **Yes** to control exporting STEP workflow task information for objects with active tasks. For more information, refer to the Workflows topic.

Configuration

Configuration Parameter	Description
Include Action Sets	Select **No** or **Yes** to control output of a list of all actions applied to each action set. For more information, refer to the Action Sets topic in the System Setup documentation.
Include Asset Push Event Queues	Asset push event queues are exported as follows: • Select **All** to output all asset push event queues. • Select **Selected** to output all asset push event queues based on the explanation in the

Configuration Parameter	Description
	Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no asset push event queues. For more information, refer to the Creating and Maintaining Asset Push Event Queues topic in the Digital Assets documentation.
Include Asset Push Configurations	Asset push configurations are exported as follows: • Select **All** to output all asset push configurations. • Select **Selected** to output all asset push configurations based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no asset push configurations. For more information, refer to the Asset Push topic in the Digital Assets documentation.
Include Attributes	For attributes, the validation base types, dimension dependencies applied units, applied LOV hierarchy filters, calculated templates, and so on, are exported as follows: • Select **All** to output all attributes, including fundamental system specific attributes (having an ID that starts with **stibo.** or **asset.**). • Select **Selected** or **Minimum** to output attributes based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no attributes. For more information, refer to the Attributes topic in the System Setup documentation.
Include Attribute Groups	For attribute groups, the view definitions and settings applied to attribute groups are exported as follows: • Select **All** to output all attribute groups. • Select **Selected** or **Minimum** to output attribute groups based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in Data Exchange documentation. • Select **None** to output no attribute groups. For more information, refer to the Attribute Groups topic in the System Setup documentation.

Configuration Parameter	Description
Include Attribute Transformations	Select **No** or **Yes** to control output of attribute transformations and their configurations. For more information, refer to the Attribute Transformations topic in the System Setup documentation.
Include Bulk Update Configurations	Assets saved as bulk update configurations are exported as follows: • Select **Yes** to output all bulk update configurations. • Select **Minimum** to output bulk update configurations based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **No** to output no bulk update configurations. For more information, refer to the Bulk Updates topic.
Include Business Rules (Global) and Libraries	For global business rules and libraries, the ID, name, links, on approve setting, dependencies, valid object types, and templates are exported as follows: • Select **All** to output all global business rules and libraries. • Select **Selected** or **Referenced** to output attributes based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no global business rules or libraries. Templates are exported in Base64 and can only be imported without modifications into another STEP system. For more information, refer to the Business Rules topic.
Include Collection Definitions	Select **No** or **Yes** to control output of collections groups and collections, including search URLs. The exported file will not contain products, classifications, or assets included in a collection. For more information, refer to the Collections topic in the Getting Started documentation.
Include Component Models	Select **No** or **Yes** to control output of details of component models, including ID, name, object types and links. The Component Model node in System Setup displays all available component models on your system. For more information, refer to the Component Models topic in the System Setup documentation.

Configuration Parameter	Description
Include Contexts	Select **No** or **Yes** to control output of dimensions, dimension points, contexts, applied dimension points, and locale settings. For more information, refer to the Contexts topic in the System Setup documentation.
Include Context Qualifiers	Select **No** or **Yes** to control output of a list of qualifiers used in exported data. When used in combination with export of product, classification, or asset data, controls qualifier IDs and dimension point IDs. For more information, refer to the Contexts topic in the System Setup documentation.
Include Data Container Definitions	Data container types are exported as follows: • Select **All** to output all data container types. • Select **Selected** or **Minimum** to output data container types based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no data container types. For more information, refer to the Data Containers topic in the System Setup documentation.
Include Derived Event Types	Select **No** or **Yes** to control output of objects created in System Setup as Derived Event Type. For more information, refer to the Derived Events topic in the System Setup documentation.
Include eCatalogs	Select **No** or **Yes** to control output of eCatalog configurations and price lists. Does not include product selection lists. For more information, refer to the eCatalogs topic.
Include Event Processors	Control output of event processor information, including ID, name, links, and configuration. • Select **All** to output all event processors. • Select **Minimum** to output event processors based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no event processors.

Configuration Parameter	Description
	The configurations are exported in Base64 and can only be imported without modifications into another STEP system. For more information, refer to the Event Processors topic in the System Setup documentation.
Include Event Queues	Select **No** or **Yes** to control output of event queues and applied message templates, output formatting, and triggering definitions. For more information, refer to the Event Queues topic in the System Setup documentation.
Include Export Configurations	Assets saved as export configurations are exported as follows: • Select **All** to output all export configurations. • Select **Minimum** to output export configurations based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no export configurations. For more information, refer to the Maintaining a Saved Export Configuration topic in the Data Exchange documentation.
Include Image Conversion Configurations	Image conversion configurations are exported as follows: • Select **All** to output all image conversion configurations. • Select **Selected** or **Minimum** to output image conversion configurations based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in Data Exchange documentation. • Select **None** to output no image conversion configurations. For more information, refer to the Image Conversion Configuration topic in the Digital Assets documentation.
Include Import Configurations	Assets saved as import configurations are exported as follows: • Select **All** to output all import configurations. • Select **Minimum** to output import configurations based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no import configurations.

Configuration Parameter	Description
	For more information, refer to the Maintaining a Saved Export Configuration topic in the Data Exchange documentation.
Include Integration Endpoints	Control output of integration endpoint ID, name, and configurations. The configurations are exported in Base64 and can only be imported without modifications into another STEP system. • Select **All** to output all integration endpoints. • Select **Selected** to output only integration endpoints in the selected hierarchy based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no integration endpoints. For more information, refer to the inbound and outbound integration endpoint topics in the Data Exchange documentation.
Include Key Definitions	Select **No** or **Yes** to control output of definitions of objects created as keys. For more information, refer to the Unique Keys topic in the System Setup documentation.
Include Link, Reference and Object Types	**Important:** Use caution when handling reference types and object types with an ID that start with **stibo.** since they are fundamental objects. User-created object types created below the 'Setup Group type root' node and system-specific object types, link types, and system specific reference types are exported as follows: • Select **All** to output all edge, reference, and object types. • Select **Selected** or **Minimum** to output only link, reference, and object types in the selected hierarchy based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no link, reference, and object types. **Note:** Edge types are identified with either 'PA' (product to attribute link type) or 'CA' (classification to attribute link type). For more information, refer to the Reference and Link Types topic in the System Setup documentation.
Include List of	LOV definitions and values in LOVs, including applied meta attributes, validation base

Configuration Parameter	Description
Values	types, and applied units are exported as follows: • Select **All** to output all LOV definitions and values. • Select **Selected** or **Minimum** to output LOV definitions and values based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no LOV definitions or values. For more information, refer to the List of Values (LOVs) topic in the System Setup documentation
Include Match Codes	Select **No** or **Yes** to control output of match code information, including ID, name, links, setup group, valid entity hierarchies, and valid object types. For more information, refer to the Match Codes topic.
Include Matching Algorithms	Control output of matching algorithms, including ID, name, links, and configuration. • Select **All** to output all matching algorithms. • Select **Selected** to output matching algorithms based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no matching algorithms. The configurations are exported in Base64 and can only be imported without modifications into another STEP system. For more information, refer to Configuring Matching Algorithms topic.
Include Setup Entities	Control output of setup entities (configurations with a URL that starts with 'step://SetupEntity') including ID, name, links, and configuration. For example, Asset Importer, Elasticsearch Configurations, Metrics, Sufficiencies, and Value Generators. For more information, refer to the STEP Terminology topic in the Getting Started documentation. The configurations are exported in Base64 and can only be imported without modifications into another STEP system. • Select **All** to output all setup entities. • Select **Selected** to output setup entities based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no setup entities.

Configuration Parameter	Description
Include Setup Groups	Control output of setup groups that are defined in system setup to hold integration endpoints, Web UI configurations, STEP workflows, and business rules. • Select **All** to output all setup groups. • Select **Selected** to output setup groups based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no setup groups. To export user-created object types created below the 'Setup Group type root' node, refer to the **Include Link, Reference and Object Types** configuration parameter above. For more information, refer to the Setup Groups topic in the System Setup documentation.
Include Status Flags	Workflow status flags are exported as follows: • Select **All** to output all workflow status flags. • Select **Selected** to output workflow status flags based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no workflow status flags. For information, refer to the Status Flags topic in the Workflows documentation.
Include System Settings	Select **No** or **Yes** to control output of the following default settings defined on the Users and Groups root node: • Image & Document Settings flipper > Dimension Dependencies • Calculated Attribute Settings flipper > Dimension Dependencies • Product Information Manager Default Settings flipper • Flatplanner Default Settings flipper • DTP Default Settings flipper • GDSN Default Settings flipper • Terms List Settings flipper • WebServices Default Settings flipper • Web UI Default Settings flipper • Default Reference Type of Primary Image flipper • Table Default width and height flipper

Configuration Parameter	Description
	For more information, refer to the System Settings topic in the System Setup documentation.
Include Table Types	Row type definitions, column type definitions, and table type definitions, including applied transformations are exported as follows: • Select **All** to output all table types. • Select **Minimum** to output table types based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no table types. For more information, refer to the Tables documentation.
Include Tags	Select **No** or **Yes** to control output of style tags, character tags, special characters, footnotes, and hyperlinks, including information about rendering, short cuts, and applied output formatting. For more information, refer to the Tags topic in the System Setup documentation.
Include Transformation Lookup Tables	Assets saved as transformation lookup table configurations are exported as follows: • Select **All** to output all transformation lookup table configurations. • Select **Minimum** to output transformation lookup table configurations based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no transformation lookup table configurations. For information, refer to the Transformation Lookup Tables topic in the Resource Materials online help.
Include Translation Configurations	Saved translation configurations are exported as follows: • Select **All** to export all saved translation configurations. • Select **Selected** to export selected saved translation configurations based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to export no saved translation configurations. For information on saving translation configurations for data objects, refer to the Starting a Structured Translation topic; for information on saving translation configurations for setup objects, refer to the Structured Translation for Setup Objects topic, both in the

Configuration Parameter	Description
	Translations documentation.
Include Units	Unit groups and units, including applied meta attributes, values, and base unit conversions are exported as follows: • Select **All** to output all unit groups and units. • Select **Selected** or **Minimum** to output unit groups and units based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no unit groups or units. For more information, refer to the Units topic in the System Setup documentation.
Include Users and User Groups	Control output of all user groups and users, including information about applied privileges, meta attributes, restricted GUI setup, etc. User passwords are not included in the output. Since creating a new user requires a password, new users cannot be created via STEPXML import. However, changes to existing users can be imported. **Note:** When changing a user from one assigned group to another, the user is added to the new group, but must be manually removed from the original group. • Select **All** to output all users or user groups. The Select Objects step allows you to limit the users and/or groups after setting the Export parameter to All, use the Add Objects link to define the objects to export. • Select **Selected** to output users or user groups based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no users or user groups. For more information, refer to the Users and Groups topic in the System Setup documentation.
Include Web UI Configurations	Control output of Web UI configurations in the exported file. • Select **All** to output all Web UI configurations. • Select **Selected** to output Web UI configurations based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no Web UI configurations.

Configuration Parameter	Description
	For more information, refer to the Managing Web UI Configurations topic in the Web User Interfaces documentation.
Include Workflows	Workflow information, including ID, name, links, valid object types, and configuration are exported as outlined below. The configurations are exported in Base64 and can only be imported without modifications into another STEP system. • Select **All** to output all workflows. • Select **Selected** to output workflows based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no workflows. For more information, refer to the Workflows documentation.
Include Workflow Profiles	Workflow profiles (including information such as number of exceeded deadlines, how long the tasks have been assigned to the assignees, throughput for the month versus the last six months, etc.) are exported as follows: • Select **All** to output all workflow profiles. • Select **Selected** to output workflow profiles based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no workflow profiles. For more information, refer to the Monitoring Workflows topic in the Workflows documentation.

Publishing

Publishing Parameter	Description
Include Publication Groups	Select **No** or **Yes** to control output of publication groups. For more information, refer to the Publication Hierarchy Elements topic in the Publisher (Adobe InDesign Integration) documentation.
Include Publications	Select **No** or **Yes** to control output of publications. For more information, refer to the Publication Hierarchy Elements topic in the Publisher (Adobe InDesign Integration) documentation.

Publishing Parameter	Description
Include Publication Sections	Select **No** or **Yes** to control output of publication sections. For more information, refer to the Publication Hierarchy Elements topic in the Publisher (Adobe InDesign Integration) documentation.
Include Planned Pages	Select **No** or **Yes** to control output of planned pages. For more information, refer to the Publication Hierarchy Elements topic in the Publisher (Adobe InDesign Integration) documentation.
Include Autopage References	Select **No** or **Yes** to control output of AutoPage references. For more information, refer to the Exporting and Importing AutoPage Publications in STEPXML topic in the Publisher (Adobe InDesign Integration) documentation.
Include Baskets	Select **No** or **Yes** to control output of details of Flatplanner baskets. For more information, refer to the Flatplanner Baskets topic in the Publisher (Adobe InDesign Integration) documentation.
Include Pagination Rules	Select **No** or **Yes** to control output of details of pagination rules. For more information, refer to the Exporting and Importing AutoPage Publications in STEPXML topic in thePublisher (Adobe InDesign Integration) documentation.
Include Actual Page Data	Select **No** or **Yes** to control output of details of actual page data. For more information, refer to the Saving Mounted Planned Pages to STEP topic in the Publisher (Adobe InDesign Integration) documentation.

STEPXML Default Data Outbound

The STEPXML format allows you to limit and/or include data based on the parameter settings. (For details, refer to the **STEPXML Outbound Parameters** topic.) However, for products, entities, classifications, and assets, when specific data is available on the exported objects, it is output by default. No outbound parameters allow this specific data to be excluded from the output.

Data output for the following export types does not include any additional default data: Publication Objects, Attribute, STEP Workflow, List of Values, and Change Packages.

Note: Selecting an option from the Export dropdown specifically impacts formats that require mapping. For STEPXML, which does not require mapping, the Export dropdown modifies the default settings on the STEPXML Format step. For other formats, when the Map Data step is disabled, this selection has no effect.

The sections below demonstrate the data exported automatically, in addition to the default outbound parameter settings.

Product

On a product export, default outbound parameters in the Data Objects section are 'Include Products = Minimum' and 'Include Product Attribute Values = Yes', while all other parameters are set to 'No' or 'None'. As shown below, more than just products and their values are exported in this scenario.

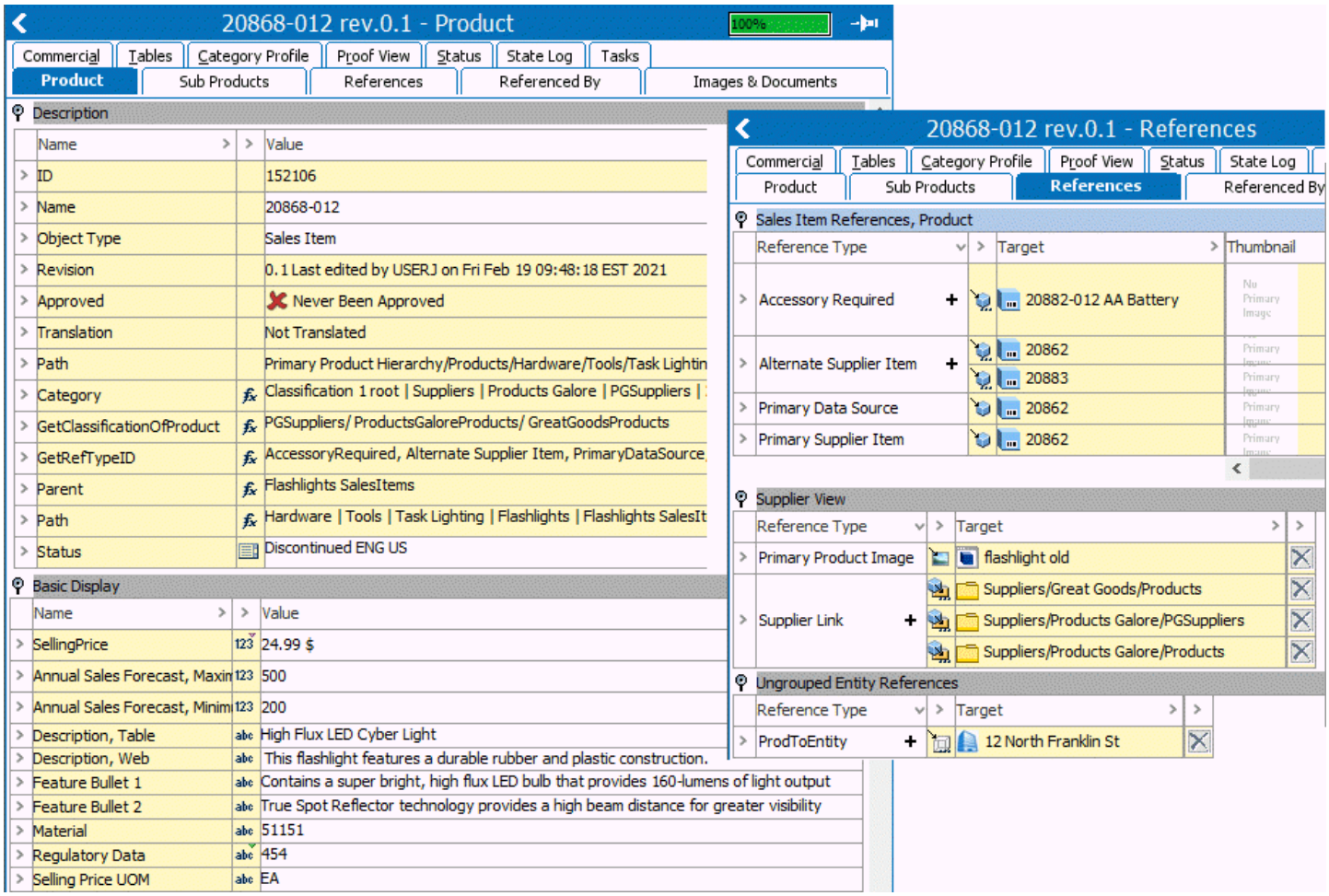
The data included in addition to the specified parameter settings is defined in the comment section of the export file. The comment section is the text between the **<!--** and the **-->** tags (shown in blue text below) at the top of the exported file.

```
<!--
    Configuration:
   <STEP-ProductInformation ResolveInlineRefs="true" FollowOverrideSubProducts="true">
   <Products ExportSize="Minimum">
   <Product>
   <Name/><AttributeLink/><DataContainerTypeLink/><ClassificationReference/><Product/>
   <ProductCrossReference/><AssetCrossReference/><EntityCrossReference/>
   <ClassificationCrossReference/><Values/><OverrideSubProduct/></Product></Products>
   </STEP-ProductInformation>

   Export from 20868-012
   Classifications All
   Products "152106"
   Assets All

   Exported by database exporter.
-->
```

When available on the exported objects, the following data is also exported for the objects:

- Attribute links and values - data from the References tab under the 'Linked Attributes from Product Hierarchy' and 'Linked Attributes from Classification Hierarchy' flippers
- References values - data from the References tab for referenced classification, entity, product, and asset objects
- Data Container Type links
- Override sub products

For example, the outbound parameter settings used for the following export are 'Include Products = Minimum' and 'Include Product Attribute Values = Yes' while all other parameters are set to 'No' or 'None' for the product shown in workbench below:



The output file is shown below. While the references information cannot be specifically selected on the outbound parameters step, it is included in the output.

```
- <STEP-ProductInformation ExportTime="2021-02-19 09:49:45" ExportContext="Context1" ContextID="Context1" WorkspaceID="Main" UseContextLocale="false">
  - <Products>
    - <Product ID="152106" UserTypeID="SalesItem" ParentID="20861">
        <Name>20868-012</Name>
        <AttributeLink AttributeID="AttrSize" />
      - <AttributeLink AttributeID="AirTransportationRestrictions">
        - <MetaData>
            <Value AttributeID="Caption1">Ground Transport Only</Value>
          </MetaData>
        </AttributeLink>
        <ClassificationReference ClassificationID="ProductsGaloreProducts" Type="SupplierLink" />
        <ClassificationReference ClassificationID="GreatGoodsProducts" Type="SupplierLink" />
        <ClassificationReference ClassificationID="PGSuppliers" Type="SupplierLink" />
      - <ProductCrossReference ProductID="20882" Type="AccessoryRequired">
        - <MetaData>
            <Value AttributeID="GetTargetProducName" Derived="true">20882-012 AA Battery</Value>
            <Value AttributeID="GetSourceProductID" Derived="true">152106</Value>
          </MetaData>
        </ProductCrossReference>
      - <ProductCrossReference ProductID="20883" Type="Alternate Supplier Item">
        - <MetaData>
            <Value AttributeID="Category" Derived="true">N/A | N/A</Value>
          </MetaData>
        </ProductCrossReference>
      - <ProductCrossReference ProductID="20862" Type="Alternate Supplier Item">
        - <MetaData>
            <Value AttributeID="Category" Derived="true">N/A | N/A</Value>
          </MetaData>
        </ProductCrossReference>
        <ProductCrossReference ProductID="20862" Type="PrimarySupplierItem" />
        <ProductCrossReference ProductID="20862" Type="PrimaryDataSource" />
        <EntityCrossReference EntityID="121605" Type="ProdToEntity" />
        <AssetCrossReference AssetID="20869" Type="PrimaryProductImage" />
      - <Values>
          <Value AttributeID="DescriptionTable">High Flux LED Cyber Light</Value>
          <Value AttributeID="SalesItemShortDescription">Flashlight, LED, Dorcy 41-4750, 160-lumen</Value>
          <Value AttributeID="FeatureBullet2">True Spot Reflector technology provides a high beam distance for greater visibility</Value>
          <Value AttributeID="SellingPriceUOM">EA</Value>
          <Value AttributeID="AnnualSalesForecastMaximum">500</Value>
          <Value AttributeID="SellingPrice" UnitID="iso4217.unit.USD">24.99</Value>
          <Value AttributeID="FeatureBullet1">Contains a super bright, high flux LED bulb that provides 160-lumens of light output</Value>
          <Value AttributeID="AnnualSalesForecast,Minimum">200</Value>
          <Value AttributeID="DescriptionWeb">This flashlight features a durable rubber and plastic construction.</Value>
          <Value AttributeID="Material">51151</Value>
          <Value AttributeID="Status" ID="111401">Discontinued ENG US</Value>
          <Value AttributeID="GetClassificationOfProdcut" Derived="true">PGSuppliers/ ProductsGaloreProducts/ GreatGoodsProducts</Value>
          <Value AttributeID="Path" Derived="true">Hardware | Tools | Task Lighting | Flashlights | Flashlights SalesItems</Value>
          <Value AttributeID="Category" Derived="true">Classification 1 root | Suppliers | Products Galore | PGSuppliers | 20868-012</Value>
          <Value AttributeID="GetRefTypeID" Derived="true">AccessoryRequired, Alternate Supplier Item, PrimaryDataSource, PrimarySupplierItem</Value>
          <Value AttributeID="Parent" Derived="true">Flashlights SalesItems</Value>
        </Values>
      </Product>
    </Products>
  </STEP-ProductInformation>
```

Entity

On an entity export, default outbound parameters in the Data Objects section are 'Include Entities = Minimum', 'Include Entity Attribute Values = Yes', 'Include Products = Minimum', and 'Include Product Attribute Values = Yes' while all other parameters are set to 'No' or 'None'. As shown below, more than just entities and their values are exported in this scenario. The output also includes the cross references.

```
- <STEP-ProductInformation ExportTime="2022-07-28 11:58:36" ExportContext="Context1" ContextID="Context1" WorkspaceID="Main" UseContextLocale="false">
  - <Entities>
    - <Entity ID="CUS_131002" UserTypeID="CD_Customer" ParentID="CustomerList">
        <Name>Jane Doe</Name>
        <EntityCrossReference EntityID="CUS_156406" Type="ConfirmedDuplicateContact" />
        <EntityCrossReference EntityID="ADD_107837" Type="Address" />
        <AssetCrossReference AssetID="businesscard" Type="PrimaryProductImage" />
        <EntityCrossReference EntityID="121608" Type="SendTo" />
        <EntityCrossReference EntityID="CUS_131005" Type="AffiliateOf" />
      - <Values>
          <Value AttributeID="PhoneNumber">7708889868</Value>
          <Value AttributeID="City">Ga</Value>
          <Value AttributeID="Customer_Number">1531356</Value>
          <Value AttributeID="CustomerEMail">jdoe@gmail.com</Value>
          <Value AttributeID="CountryCode" ID="128342">UNITED STATES</Value>
        </Values>
      </Entity>
    </Entities>
  </STEP-ProductInformation>
```

Classification

On a classification export, default outbound parameters in the Data Objects section are 'Include Entity Attribute Values = Yes', 'Include Product Attribute Values = Yes', and 'Include Classifications = Minimum', while all other parameters are set to 'No' or 'None'. As shown below, more than just entities and their values are exported in this scenario. The output also includes attribute links and cross references.

```
- <STEP-ProductInformation ExportTime="2022-07-28 12:07:21" ExportContext="Context1" ContextID="Context1" WorkspaceID="Main" UseContextLocale="false">
  - <Classifications>
    - <Classification ID="Classification 1 root" UserTypeID="Classification 1 user-type root" Selected="false">
        <Name>Classification 1 root</Name>
      - <Classification ID="WebHierarchyRoot" UserTypeID="WebHierarchyRoot" Selected="false">
          <Name>Web Sites</Name>
          <AttributeLink AttributeID="Part_Number" />
        - <MetaData>
            <Value AttributeID="Purpose">Organize websites and collections of products to sell on the website</Value>
          </MetaData>
        - <Classification ID="SalesPromotions" UserTypeID="WebsiteRoot" Selected="false">
            <Name>Sales Promotions</Name>
            <ClassificationCrossReference ClassificationID="Hats" Type="Primary Classification" />
          - <Classification ID="Apparel" UserTypeID="WebLevel1" Selected="false">
              <Name>Apparel</Name>
            - <Classification ID="Hats" UserTypeID="WebLevel2">
                <Name>Hats</Name>
                <ProductReference ProductID="101567" Type="WebsiteLink" />
                <ProductReference ProductID="109013" Type="WebsiteLink" />
                <ProductReference ProductID="23456-002" Type="WebsiteLink" />
                <ClassificationCrossReference ClassificationID="Shoes" Type="ClassificationReference" />
                <ClassificationCrossReference ClassificationID="AcmeRetailWebsiteRoot" Type="Class2SecondaryClass" />
                <ClassificationCrossReference ClassificationID="SalesPromotions" Type="Primary Classification" />
              </Classification>
            </Classification>
          </Classification>
        </Classification>
      </Classifications>
    </STEP-ProductInformation>
```

Asset

On an asset export, default outbound parameters in the Data Objects section are 'Include Entity Attribute Values = Yes', 'Include Product Attribute Values = Yes', and 'Include Assets = Minimum', while 'Include Asset Content' and all other parameters are set to 'No' or 'None'. As shown below, more than just assets and their attribute values are exported in this scenario. The output also includes classification references.

```
- <STEP-ProductInformation ExportTime="2022-07-28 12:16:30" ExportContext="Context1" ContextID="Context1" WorkspaceID="Main" UseContextLocale="false">
  - <Assets>
    - <Asset ID="157915" UserTypeID="ProductImage">
        <Name>mouse</Name>
        <ClassificationReference ClassificationID="AssetsRoot" />
      - <Values>
          <Value AttributeID="asset.pixel-height" UnitID="pixels">760</Value>
          <Value AttributeID="asset.size">157459</Value>
          <Value AttributeID="asset.compression">Zip</Value>
          <Value AttributeID="asset.ydpi" UnitID="dpi">96.012</Value>
          <Value AttributeID="asset.format">PNG (Portable Network Graphics image)</Value>
          <Value AttributeID="asset.xdpi" UnitID="dpi">96.012</Value>
          <Value AttributeID="asset.width" UnitID="mm">216.40211640211638</Value>
          <Value AttributeID="asset.filename">mouse.png</Value>
          <Value AttributeID="asset.uploaded">2022-06-17 17:05:37</Value>
          <Value AttributeID="asset.class">True color</Value>
          <Value AttributeID="asset.extension">png</Value>
          <Value AttributeID="asset.pixel-width" UnitID="pixels">818</Value>
          <Value AttributeID="asset.height" UnitID="mm">201.05820105820106</Value>
          <Value AttributeID="asset.colorspace">RGB</Value>
          <Value AttributeID="asset.depth" UnitID="bits/sample">8</Value>
          <Value AttributeID="asset.mime-type">image/png</Value>
          <Value AttributeID="asset.samples" UnitID="samples/pixel">3</Value>
          <Value AttributeID="AssetName" Derived="true">mouse</Value>
          <Value AttributeID="GetClassificationsOfAnAsset" Derived="true">Assets</Value>
        </Values>
      </Asset>
    </Assets>
  </STEP-ProductInformation>
```

STEPXML Tags and Examples

Advanced STEPXML tags determine how data is imported and exported from STEP. Update or replace the default template with Advanced STEPXML tags to generate the desired output.

XSD (XML Schema Definition)

An XSD (XML Schema Definition) file includes information on the required, optional, and deprecated elements for tags. XSD files are linked from the **STEP API Documentation** within the **STEPXML** section, at [system]/sdk or accessible from the Start Page.

- For successful importing or exporting, Advanced STEPXML format and STEPXML format files must conform to the PIM.XSD file. Click the **XSD** link to download the file.
- When the option to include empty values is selected for STEPXML or Advanced STEPXML exports, all linked and valid attributes, references, metadata, and data containers are included in the export.
- To successfully specify what should be exported when using the Advanced STEPXML format, the export file must conform to the PIMOutputTemplate.XSD file. Click the Output Template (Recorder File) **XSD** link to download the recorder XML file.

Note: STEPXML is intended for use with external systems for integration purposes. Exported empty values are not intended to be imported into STEP.

Important: The true and false settings for all XML attributes are case-sensitive and must be lower case.

Tags for Common Data

Although the list below is not all-inclusive, the most commonly required data can be handled using the tags defined in the following topics:

- AssetContent Tag in STEPXML
- AssetPushConfiguration Tag in STEPXML
- Assets Tag in STEPXML
- AttributeLink Tag in STEPXML
- Classifications Tag in STEPXML
- ImageConversionConfiguration Tag in STEPXML
- IncludeKey Tag in STEPXML
- Products Tag in STEPXML
- ReplacementRules Tag in STEPXML

- SequenceProduct Tag in STEPXML
- STEP-ProductInformation Tag in STEPXML

For details on tags, elements, and their XML attributes, refer to the available XSD (XML Schema Definition) files in the **STEPXML Tags and Examples** topic.

Examples

Although the list below is not all-inclusive, the most commonly required data can be handled using the tags defined in the following topics:

- Auto Approve in STEPXML
- Backward Compatibility in STEPXML
- Context Data in STEPXML
- Cross Context Inheritance in STEPXML
- Create Objects in STEPXML
- Data Containers in STEPXML
- Delete Objects in STEPXML
- Delete Product Reference in STEPXML
- Delete Values in STEPXML
- Export Inheritance in STEPXML Example
- Filter Data Containers in STEPXML
- Filter Products in STEPXML
- Filter References in STEPXML
- Filter Values in STEPXML
- Inherit References in STEPXML
- Inherit Values in STEPXML
- Minimum, Referenced, and Selected in STEPXML
- Referenced and Embedded XML Attributes in STEPXML
- SingleUpdateMode in STEPXML

Assets Tag in STEPXML

The assets tag is used to export the metadata on the asset and can include multiple additional tags, each is described below.

A single <Assets/> tag implies several other default tags. Consider this structure:

```
<?xml version='1.0'?>
<STEP-ProductInformation>
  <Assets>
    <Asset>
      <Name/>
      <Values/>
    </Asset>
  </Assets>
</STEP-ProductInformation>
```

For example, to only export the asset name, use the following structure:

```
<Assets>
  <Asset>
    <Name/>
  </Asset>
</Assets>
```

For details on tags, elements, and their XML attributes, refer to the available XSD (XML Schema Definition) files in the **STEPXML Tags and Examples** topic.

Name

The <Name/> tag returns the object name as output.

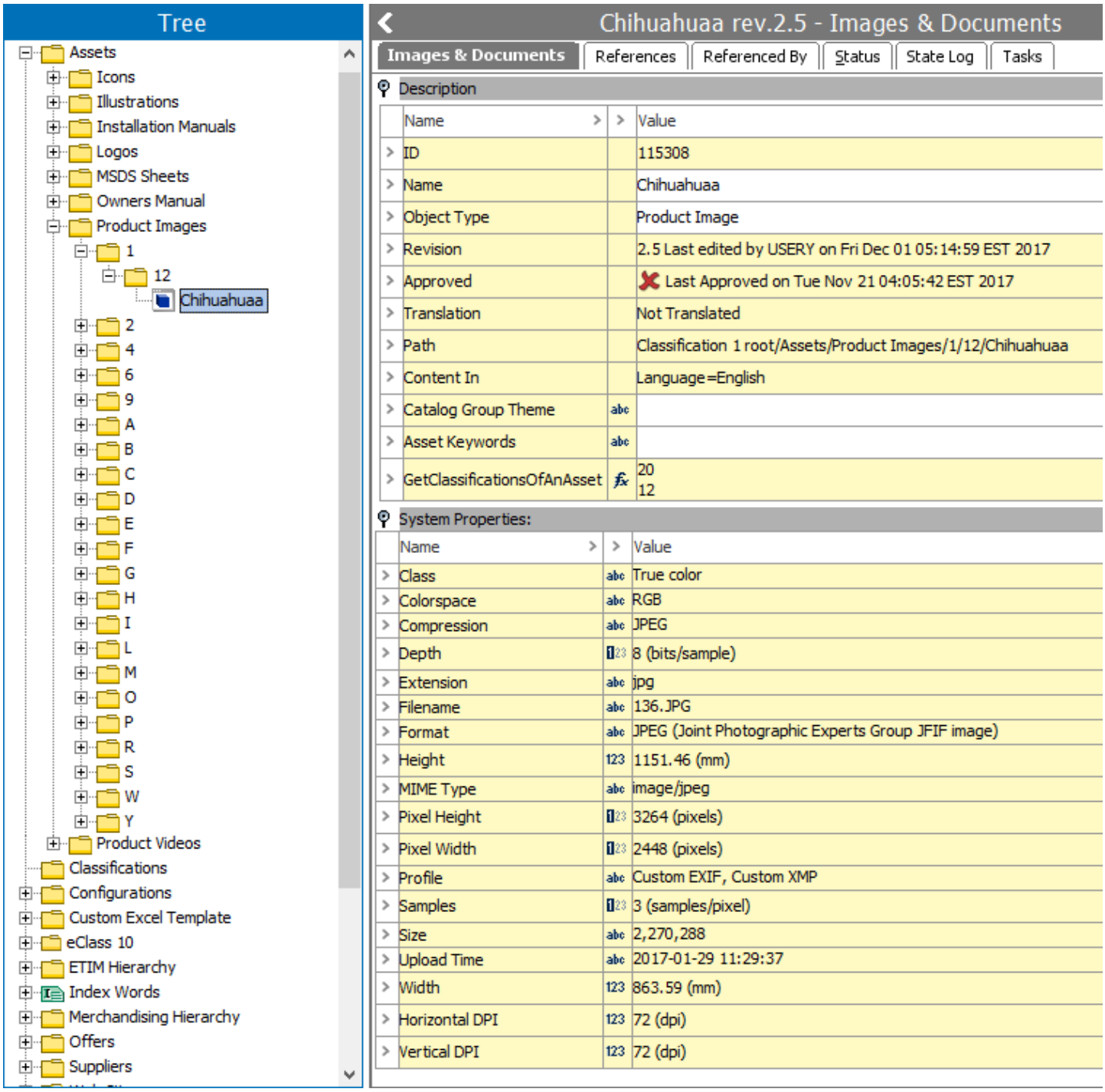
Values

The <Values/> tag returns the attribute values as output.

> **Note:** Since only description attributes can be added to an asset, value inheritance is not applicable.

For example, the following asset object is exported using Advanced STEPXML using the first template displayed previously in this topic. The results are shown below.

Object



Results

The following excerpt of the output shows the values from the Description flipper and the System Properties flipper of the asset.

```
<STEP-ProductInformation ExportTime="2018-03-02 16:11:18" ExportContext="Context1" ContextID="Contex

  <Assets>
    <Asset ID="115308" UserTypeID="ProductImage">
      <Name>Chihuahuaa</Name>
      <Values>
        <Value AttributeID="asset.pixel-height" UnitID="pixels">3264</Value>
        <Value AttributeID="asset.size">2270288</Value>
        <Value AttributeID="asset.compression">JPEG</Value>
        <Value AttributeID="asset.ydpi" UnitID="dpi">72</Value>
        <Value AttributeID="asset.format">JPEG (Joint Photographic Experts Group JFIF image)</Value>
        <Value AttributeID="asset.xdpi" UnitID="dpi">72</Value>
        <Value AttributeID="asset.width" UnitID="mm">863.5999999999999</Value>
        <Value AttributeID="asset.filename">136.JPG</Value>
        <Value AttributeID="asset.uploaded">2017-01-29 11:29:37</Value>
        <Value AttributeID="asset.class">True color</Value>
        <Value AttributeID="asset.extension">jpg</Value>
        <Value AttributeID="asset.pixel-width" UnitID="pixels">2448</Value>
        <Value AttributeID="asset.height" UnitID="mm">1151.4666666666665</Value>
        <Value AttributeID="asset.colorspace">RGB</Value>
        <Value AttributeID="asset.depth" UnitID="bits/sample">8</Value>
        <Value AttributeID="asset.mime-type">image/jpeg</Value>
        <Value AttributeID="asset.profile">Custom EXIF, Custom XMP</Value>
        <Value AttributeID="asset.samples" UnitID="samples/pixel">3</Value>
        <Value AttributeID="GetClassificationsOfAnAsset" Derived="true">20
12</Value>
      </Values>
    </Asset>
  </Assets>
</STEP-ProductInformation>
```

Including parent classifications

For each asset in the exported file, the classifications that the asset is linked to will be added to the export if the 'IncludeParentClassifications' attribute is used, set to true, and the template is configured to export classifications.

This attribute will only impact the domain exporter used per default on systems running In-Memory and will be ignored on systems using the legacy database exporter.

AssetContent Tag in STEPXML

The AssetContent tag is used to export digital asset content for both image assets and non-image assets via STEPXML. Asset content can be exported using either Base64 encoded binary, which encodes the asset content into a series of characters, or a relative REST resource URL, which exports a URL that points to where that specific asset content is stored. Refer to the 'Export Asset Content Template' and 'Export Asset Content Results' sections below.

This tag can be used for import only when used as described in the **Asset Content Templates** section below.

The AssetContent tag is valid in an outbound Advanced STEPXML format template, which requires an entry-level knowledge of XML structure. Alternatively, the STEPXML format provides the same functionality via a set of parameters, and automatically writes the tag and XML attributes in a template in the background, based on the parameter settings.

For more information on the export process, refer to **Assets and Content with STEPXML** in the **Digital Assets** documentation.

For details on tags, elements, and their XML attributes, refer to the available XSD (XML Schema Definition) files in the **STEPXML Tags and Examples** topic.

Asset Content Templates

When exporting asset content via Advanced STEPXML, a template is required and must include the **AssetContent** tag with the XML attribute **ExportType**. The valid XML attribute values for 'ExportType' are REST and Binary. These values indicate the method used to output the asset content.

As an example, when the value for the 'ExportType' XML attribute in the AssetContent tag is 'REST', the image content output is a URL. When the 'ExportType' value is 'Binary', the output is the image content encoded using Base64.

The AssetContent tag is further defined using the required **ImageConversionConfiguration** tag with the XML attribute **ID** to indicate the desired image configuration(s). In addition to exporting image asset content using one or many saved image conversions, the source image asset is also available for export, meaning no conversion is performed.

> **Note:** At least one image conversion configuration must be included in order to export image asset content, otherwise the AssetContent tag is not output.

When exporting non-image assets, only the ImageConversionConfiguration ID="Source" is valid. Documents, for example, are not affected by image conversion configurations.

For information on creating image conversion configurations, refer to **Image Conversion Configuration** topic in the **Digital Assets** documentation.

Unconverted 'Source' data can be imported which allows you to modify asset content. The import process creates new asset content or updates existing content using the AssetBinaryContent element with an empty string for the 'ImageConversionConfigurationID' tag. For details on this process, click the **STEP API Documentation** button on the Start Page for a link to the XSD file.

Relative REST URL Template Example

The AssetContent tag appearing in the following sample STEPXML template outputs a relative REST URL for both the 'source' asset and a conversion with the ID '184968.'

```
<?xml version="1.0" encoding="UTF-8"?>
<STEP-ProductInformation ResolveInlineRefs="true" FollowOverrideSubProducts="true">
  <Assets ExportSize="Minimum">
    <Asset>
     <Name/>
     <AssetPushLocation/>
     <ClassificationReference/>
     <Values/>
     <AssetContent ExportType="REST">
      <ImageConversionConfiguration ID="Source"/>
      <ImageConversionConfiguration ID="184968"/>
     </AssetContent>
    </Asset>
  </Assets>
</STEP-ProductInformation>
```

Binary Content Template Example

The AssetContent tag appearing in the following sample STEPXML template outputs binary content for both the 'source' asset and a conversion with the ID '124011.'

```
<?xml version="1.0" encoding="UTF-8"?>
<STEP-ProductInformation ResolveInlineRefs="true" FollowOverrideSubProducts="true">
  <Assets ExportSize="Minimum">
   <Asset>
    <Name/>
    <AssetPushLocation/>
    <ClassificationReference/>
    <Values/>
    <AssetContent ExportType="Binary">
      <ImageConversionConfiguration ID="Source"/>
      <ImageConversionConfiguration ID="124011"/>
     </AssetContent>
    </Asset>
  </Assets>
</STEP-ProductInformation>
```

Asset Content Output

Based on the output method selected, several tags and XML attributes are used to identify the asset content.

- **AssetContent** is used for both REST and Binary, holds the required AssetContentSpecification tags, and includes no XML attributes.

- **AssetContentSpecification** is used for both REST and Binary, and holds the image configuration IDs exported using the following XML attributes:
    - **ImageConversionConfigurationID** indicates the image conversion used for the exported asset content. As shown in the examples below, an empty (blank) ID indicates the original, unconverted image is exported. This results from selecting 'Source' when using the STEPXML export option for asset content or when including 'Source' in the Advanced STEPXML template.
    - **RelativeURL** is used only for the REST option and includes the relative URL that can be used to access the asset content, based on the STEP Core REST API.
    - **IncludesBinaryContent** is used only for the Binary option and indicates whether or not the BinaryContent tag is included in the output.
- **AssetBinaryContent** is used only for the Binary option. One of these tags is output for each of the AssetContentSpecification tags in the file and includes the following XML attributes:
    - **ImageConversionConfigurationID** indicates which of the converted images is represented by the binary content that follows. As shown in the examples below, an empty (blank) ID indicates the original, unconverted image is exported. This results from selecting 'Source' when using the STEPXML export option for asset content or when including 'Source' in the Advanced STEPXML template.
    - **MimeType** is used only for the Binary option, and allows the receiving system to decode the binary data and convert it back into an asset file.
    - **Checksum** is used only for the Binary option, allows the receiving system to verify that the binary data has been correctly received and decoded, and enables that system to determine if the data has changed since it was last received.
- **BinaryContent** is used only for the Binary option, holds the BASE64 encoded binary data, and includes no XML attributes.

Note: If no content is available for the selected image(s), the execution report generated by the export contains an entry for the image but the output will not include the XML tag.

Relative REST URL Output Example

When REST is the selected method for export, each configuration ID is accompanied by a relative URL that can be used to access the asset content.

```
<AssetContent>
  <AssetContentSpecification ImageConversionConfigurationID="" RelativeURL="/restapi/assets/21869/content"/>
  <AssetContentSpecification ImageConversionConfigurationID="185016" RelativeURL="/restapi/assets/21869/conversionconfigurations/185016"/>
</AssetContent>
```

The path that displays as the relative URL varies based on the conversion selected:

- For a **source** asset, meaning no conversion is applied, the path is: /restapi/assets/{assetID}/content
- For a **converted** asset, meaning a pre-configured conversion is applied to the selected asset content, the path is: /restapi/assets/{assetID}/conversionconfigurations/{conversionID}

For example, the system receiving the export file would prepend 'http://STEPserverURL.com' to the supplied relative URL, in addition to appending the desired context using '?context=Context1.'

The resulting URL could be: http://STEPserverURL.com/restapi/assets/21142/content?context=Context1

Binary Content Output Example

The sample export file below shows that an AssetBinaryContent tag is used for each AssetContentSpecification tag found within the AssetContent tag. Additionally, for each AssetBinaryContent tag, the binary data is included in a BinaryContent tag.

```
<AssetContent>
  <AssetContentSpecification ImageConversionConfigurationID="" IncludesBinaryContent="true"/>
  <AssetContentSpecification ImageConversionConfigurationID="185016" IncludesBinaryContent="true"/>
</AssetContent>
<AssetBinaryContent ImageConversionConfigurationID="" MIMEType="image/png" Checksum="591f854e6b3a43ad71017c7d182eb2c2154d8be2">
  <BinaryContent>iVBORw0KGgoAAAANSUhEUgAAAScAAAE6CAIAAADbcANTAAAAAXNSR0IArs4c6QAAAARnQU1BAACxjwv8YQUAAAAJcEhZcwAADsMAAA7DAcdvqGQAAMkmSU
  etqnqetVadc8O3/q2d+P9/TSiHX2yU3S02yu4KuP1/81BWLXXEqjrKx/NH2V7eFMraTixJdKpuEVF2V8BtKAtl1VJHrKqjfDx/lO06VbcU0am6RUTZ3TLD2tKIGsrH80fZrlM
  NH2W7TtUtRXSqbhERjoJy2fVSQTlqDbGqjvK1+aNs16m6pYh2qy57U4dt+1//63+Vjf7t3/7v//2///qv/5prahS0Sfn//J//46q9F1NIzf/8n/+zIEADjXYFlKtOliiaBzV3
  FmgWwFLA/MaKeSQ3Z84cBYalUnsGuyovZe4arhrRSjYvJsQwlS1n1NyyE0sZHZZhUlQa/Pa3v02Brux3ZDZ79uzbbrtt1qxZUwpMnz596NChZ5555jvvvKP9P//zP4sYGocfe
  PHHP/7x3/3d36277rr9+/cfOXLk5ptv3rt377/+67+eNm3am2++iRMffPP83g305l1o7nYpINMpb+bFNPjwww+PPPLIvn37/uQnP+natevZZ59dORp8TWEpoFrqZiOV2whGA
  ...
  ...
  ...
  jyFJstskmpKdsvIbMKjw7z0LCHhhsLsy5hvAffMadBkbm4uIhXvb29lJc0yGDiIXaR39jt6+vDMdZ41JmdnZ35fH5vb+/h4YHhn2vEYJUoBOc2IsKsSx5PT0/yaLtcLheLRb
  CcM/WWODJq4zIc3l5ubu7u7S0lMlkenp6/gq+51ooFNbX1yuVCnLKk+PD4ylQGWsWRcjZl3yqM9atHFJVHx8fDw7Ozs8PNzZ2Tk6Orq7u5NjgMPq3zZGPLMuRsy6BCO2YBQ
  iYeGQ6PDPDRs/CRmXcJQOTw0HBod5qFh4ycx6xKGyuGh4dDoMA8NGj+JWZcmukzgAAAAABJRU5ErkJggg==</BinaryContent>
</AssetBinaryContent>
<AssetBinaryContent ImageConversionConfigurationID="185016" MIMEType="image/jpeg" Checksum="6510942e4e5f4a82c4d72a3ce896fab45cde8caf">
  <BinaryContent>/9j/4AAQSkZJRgABAQEAXwBfAAD/4gGoSUNDX1BST0ZJTEUAAQEAAAGYQURCRQIQAABtbnRyR1JBWVhZWiAHzwAGAAMAAAAAAABhY3NwQVBQTAAAAABub
  JkZXNjAAAA9AAAAGl3dHB0AAABYAAAABRia3B0AAABdAAAABRrVFJDAAABiAAAAA50ZXh0AAAAAENvcHlyaWdodCAxOTk5IEFkb2JlIFN5c3RlbXMgSW5jb3Jwb3JhdGVkAA
  AAAAAAAAAAAAAAAAAAAAAAWFlaIAAAAAAAAPNUAAEAAAABFs9YWVogAAAAAAAAAAAAAAAAAAAAAGN1cnYAAAAAAAAAAQIzAAD/2wBDABsSFBcUERsXFhceHBsgKEIrKCUlKFE6
  BgMHAwQCAwAAAABAgMEEQUSISIxQQYTFFFhcTKBwSNCUpGhsfBicuEzgtHxJKJDU7L/2gAIAQEAAD8A0wIAAJAIABIAIBJBIAAAAAAABB5XNxTtaMqtWWIr9TP19euZzbp7
  NUD4nGNSDhJZjJYa8ygvezjWZWdT/ZP6M8bPs5XqPN1NUo/hi8tl7Z6fbWa+wpJPrJ72/mdYAAAAAAAABBmO0dr3Vz38fhqb/mVEuj/ABELy8yw0K8drfKMn9nV4ZfRmvJIM1
  obns3UU821WMovpPc0e1l2cpwe1dz7xr7kd0S7pU4UoKFOKjFcklg+wAAAAAAAAAACAZLXbXw185QXDPiRWzWG2uTIjvfJtvkl5m10q28JYUqTXHjMvdnYCASQSAAAAAAAA
  Z</BinaryContent>
</AssetBinaryContent>
```

AssetPushConfiguration Tag in STEPXML

Asset Push configurations can be imported / exported via the STEPXML and Advanced STEPXML formats using the following tags:

- **<AssetPushConfigurations>**

  The <AssetPushConfigurations> tag is used to contain all the Asset Push configurations included within the STEPXML / Advanced STEPXML file.

- **<AssetPushConfiguration>**

  The <AssetPushConfiguration> tag is used to contain a single Asset Push configuration included within the STEPXML / Advanced STEPXML file.

Note: While Asset Push configurations can be exported using the STEPXML and Advanced STEPXML formats, users would use the STEPXML format if importing an Asset Push configuration that was originally created using the Advanced STEPXML format. For more information, refer to the **Advanced STEPXML Format** topic in the **Data Exchange** documentation.

For an example of the <AssetPushConfiguration> tag used within STEPXML, access the online version of this topic.

Note: If a user attempts to import an asset push configuration that includes an ID of an Event Queue that is not included in the STEP system where the asset push configuration is being imported into, an error message will be displayed within the execution report of the import background process.

For details on tags, elements, and their XML attributes, refer to the available XSD (XML Schema Definition) files in the **STEPXML Tags and Examples** topic.

AttributeLink Tag in STEPXML

The **AttributeLink** tag outputs local attribute links for the products, product-overrides, or classifications in the export.

```
<AttributeLink/>
```

For details on tags, elements, and their XML attributes, refer to the available XSD (XML Schema Definition) files in the **STEPXML Tags and Examples** topic.

IncludeInherited

When modified with the **IncludeInherited** option, the **AttributeLink** tag also exports inherited attribute links.

```
<AttributeLink IncludeInherited="true">
```

The **AttributeLink** tag is allowed within the following tags:

- Product - Attribute links are inherited from parent products to top-level products.
- Product-Overrides - Attribute links are inherited from overridden product to product-override. Attribute links inherited from Classifications are not included.
- Classification - Attribute links are inherited from parent classifications to top-level classifications.

Template

No Inheritance	IncludeInherited
<?xml version="1.0" encoding="UTF-8"?> <STEP-ProductInformation> <Products> <Product> <Name/> <AttributeLink> <MetaData/> </AttributeLink> </Product> </Products> </STEP-ProductInformation>	<?xml version="1.0" encoding="UTF-8"?> <STEP-ProductInformation> <Products> <Product> <Name/> <AttributeLink IncludeInherited="true"> <MetaData/> </AttributeLink> </Product> </Products> </STEP-ProductInformation>

Only local attribute links for the exported product are output.

No Inheritance

```
<STEP-ProductInformation ExportTime="2015-12-23 14:58:34" ExportC

  <Products>
    <Product ID="20878" UserTypeID="Level4" ParentID="20876">
      <Name Changed="true">Batteries  Level4</Name>
      <AttributeLink AttributeID="Voltage"/>

    </Product>
  </Products>
</STEP-ProductInformation>
```

Local and inherited attribute links for the exported product are output.

IncludeInherited

```
<STEP-ProductInformation ExportTime="2015-12-23 15:02:53" ExportCon

  <Products>
    <Product ID="20878" UserTypeID="Level4" ParentID="20876">
      <Name Changed="true">Batteries  Level4</Name>
      <AttributeLink AttributeID="Voltage"/>
      <AttributeLink AttributeID="MaximumVoltage" Inherited="3"/>
      <AttributeLink AttributeID="ETLListed" Inherited="3"/>
      <AttributeLink AttributeID="CSAListed" Inherited="3"/>
      <AttributeLink AttributeID="ULListed" Inherited="3"/>

        <MetaData>
          <Value AttributeID="DisplaySequence">2</Value>
        </MetaData>

    </Product>
  </Products>
</STEP-ProductInformation>
```

AutoApprove in STEPXML

The data structure between the Approved workspace and the Main workspace must be consistent, and is automatically enforced within STEP. This required consistency causes AutoApprove to fail under the following conditions:

1. An object cannot be approved until its parent is in the approved workspace.
2. A reference or link cannot be approved until the target is in the approved workspace.
3. A mandatory attribute or a mandatory reference / link type cannot be approved until the required data exists.
4. An object that causes any business rule that is configured to run “on Approval” or run during import cannot be approved until the condition is satisfied.

In each of these cases, an error message is logged in the execution report when the approval fails.

Configuration

Insert **AutoApprove="Y"** in the **STEP-ProductInformation** tag to approve objects during import, as illustrated in the following example:

```
<STEP-ProductInformation ContextID="Context1" WorkspaceID="Main" AutoApprove="Y">
...
</STEP-ProductInformation>
```

For more information, refer to **STEP-ProductInformation Tag in STEPXML**.

For details on tags, elements, and their XML attributes, refer to the available XSD (XML Schema Definition) files in the **STEPXML Tags and Examples** topic.

Backward Compatibility in STEPXML

The STEPXML schema is extended continuously but should always be backward compatible. Due to this fact, you may encounter legacy naming in the STEPXML format.

The legacy names that most frequently cause problems are:

- **UserType** – generally this means object type. Nested UserType elements inside the UserTypes element represent object type definitions. UserTypeLink elements inside the Attributes element indicates that the attribute is valid for specific object type. UserTypeID attributes (for example in the Product element) specifies that the product is of a specific object type.

```
<Product ID="I-SalesItem-1111" UserTypeID="SalesItem" ParentID="I-Level3-111">
```

- **ProductMode** – attribute elements have a ProductMode attribute to indicate a description or specification attribute.

  This example shows the attribute is a description attribute (value: Property).

```
<Attribute ID="EAN" Multivalued="false" ProductMode="property" FullTextIndexed=false"
  ExternallyMaintained="true" Deriver=false" Selected="true" Referenced="true"> <Name>EAN</Name>
  </Attribute>
```

  This example shows the attribute is a specification attribute (value: Normal).

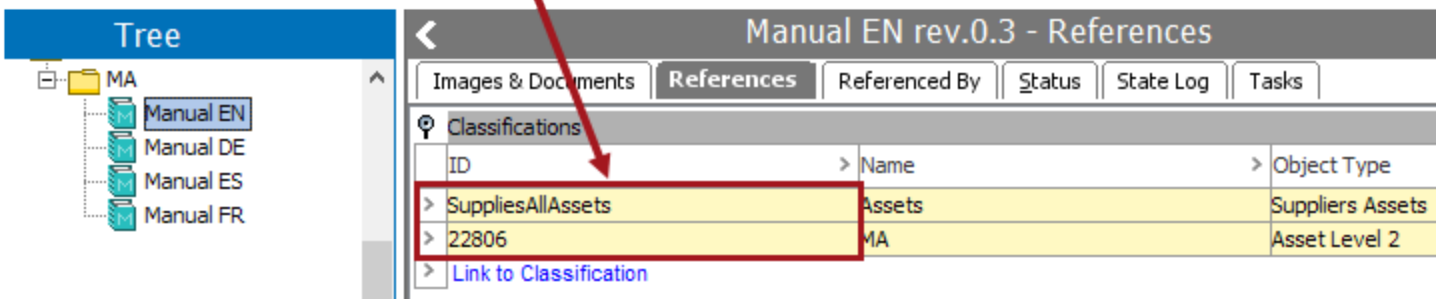
```
<Attribute ID= "Depth" Multivalued="false' DefaultUnitID="unece.unit.CMT" ProductMode="Normal" FullTEXTIndext="false"
  ExternallyMaintained="false" Derived="false" Selected="true" Referenced="true"></Attribute>
  </Attribute>
```

- **Derived** – attribute element that specifies if the attribute is calculated.

```
<Value AttributeID="CountryOfOrigin" Derived="true">GB</Value>
<Value AttributeID="ManufacturerEAN" Derived="true">9513864492123</Value>
```

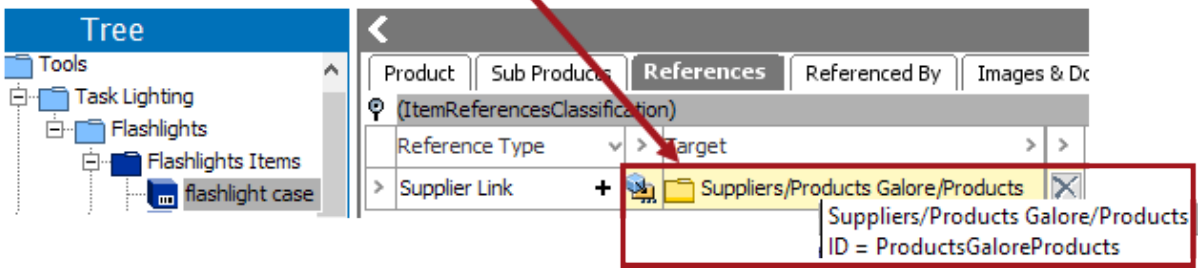
- **ClassificationReference** – used inside both the Product and Asset elements. For assets, this is the parent link (assets can have multiple parents).

```
<ClassificationReference ClassificationID="SuppliesAllAssets" />
<ClassificationReference ClassificationID="22806" />
```



For products, the element represents product-to-classification links.

```
<Product ID="flashlight case" UserTypeID="Item" ParentID="Flashlights Items">
  <Name>flashlight case</Name>
  <ClassificationReference ClassificationID="ProductsGaloreProducts" Type="SupplierLink"
```



For details on tags, elements, and their XML attributes, refer to the available XSD (XML Schema Definition) files in the **STEPXML Tags and Examples** topic.

Classifications Tag in STEPXML

The Classification tag can include an XML attribute.

```
<?xml version="1.0" encoding="UTF-8"?>
<STEP-ProductInformation >
  <Classifications ExportSize="Minimum">
    <Classification IncludeParent="true"/>
  </Classifications>
</STEP-ProductInformation>
```

Note: To enable the functionality to replace ExportSize="Minimum" with ExportSize="Referenced" for Classifications tag, when exporting products in Excel / CSV format with mapped Classification Product links, contact Stibo Systems Support. This will decrease the amount of classifications included in the export to only referenced classifications.

When an XML attribute is absent, the default setting is used. For example:

```
<Classifications>
    <Classification IncludeParent="true"/>
</Classifications>
```

For details on tags, elements, and their XML attributes, refer to the available XSD (XML Schema Definition) files in the **STEPXML Tags and Examples** topic.

The following XML attribute is available:

IncludeParent

- False - default setting
- True - when a classification is exported from STEP, the file includes all parent levels.

Additional STEPXML Tags

The Classification tag can include multiple additional internal tags, each is described below the following sample STEPXML. If no elements are explicitly specified, a default configuration will be used. For an example of the XML, refer to the online version of this topic.

Name

The <Name/> tag returns the object name as output.

Values

The <Values/> tag returns the attribute values as output. For more information about values output, refer to the **Inherit Values in STEPXML** section.

ClassificationReference

The <ClassificationReference/> tag returns all classification cross references as output.

MetaData

The <MetaData/> tag returns metadata as output.

Product to Classification Links Owned by the Classification

By default the product to classification links owned by the classification are exported in two places:

1. Under the Product as a reference to the classification.
2. Under the Classification as a reference to the product. For an XML example of the classification links, refer to the online version of this topic.

The reason the links are exported under the products is historical, and it has not been changed due to backwards compatibility. However, logically they should be exported under the classification, as they are owned by the classification. And when the links are modified, they affect the revision of the classification - not the product. It is possible to stop the classification owned links from being exported under the products by using the recorder option 'IncludeClassificationOwned' on the 'Products' element. By setting this to "false" the classification links will not be exported under the products:

```
<STEP-ProductInformation>
    <Products IncludeClassificationOwned="false">
        <Product>
        </Product>
      </Products>
</STEP-ProductInformation>
```

Context Data (Qualifiers) in STEPXML

A single STEPXML file can include data from multiple contexts, for example, data tied to specific dimensions points or combinations of dimension points from different dimensions. This 'cross-context' STEPXML will include a 'Qualifiers' section that maps the 'QualifierID' XML attribute value for dimension dependent elements to dimension points or combinations of dimension points. Cross-context STEPXML can be imported and exported.

For details on tags, elements, and their XML attributes, refer to the available XSD (XML Schema Definition) files in the **STEPXML Tags and Examples** topic.

Data Example

The following example shows data for a single product being exported and includes:

- Two contexts: 'US-eng' (Context1) and 'DE-de' (Context3)
- Two country dimension points: 'US' and 'DE'
- Two language dimension points: 'eng' and 'de'
- Three language-dependent attributes: 'LongAdvertisingCopy', 'ConsumerShortDescription', and 'Color'
- One country-dependent attribute: 'Warranty'
- Two attributes that are not dimension dependent: 'Weight' and 'EAN'

Within the output tool, set the **Export Data for Selected Contexts** option as discussed in the **STEPXML Outbound Parameters** topic.

Result

For the STEPXML output, refer to the online version of this topic.

Considerations for Qualifier IDs

Notice that the context element nested in the qualifier element is not required for imports. It is present to let downstream systems map values to specific contexts. The logic for this mapping is that a context uses the value for the qualifier where it has the lowest 'priority' number.

The qualifiers might not always be as straight-forward as in the example above, where you have one qualifier per dimension point. Although it is not recommended, dimension dependent data can depend on more than one dimension at a time. In that case, a 'pseudo qualifier' is displayed and references one dimension point per dependent dimension.

Several factors may complicate the reading of qualifier IDs:

- Dimensional values are inherited from more general dimension points.
- When adding multiple dimension dependencies, you get values for specific combinations of contexts, and the multi-context export result gets complicated to read.

- XML elements are repeated in their nested elements, and new XML elements are introduced. This means that while still adhering to the STEPXML XSD, the result is a superset of the typical single-context export format.
- A single qualifier will map to multiple contexts with different priorities, and as discussed above you may have several possible values with different priorities.

Since it can be very difficult to interpret multi-context export files, the recommended method for exporting multiple contexts is to use the Context Splitter post-processor available in an OIEP. For more information, refer to the **OIEP - Post-processor - Context Splitter** topic in the **Outbound Integration Endpoints** documentation.

Create Objects in STEPXML

New objects can be created in the Tree and in System Setup using the STEPXML import.

New users, however, cannot be created via STEPXML import because passwords are required and passwords are not included in the STEPXML format. When changing a user from one assigned group to another, the user is added to the new group, but must be manually removed from the original group.

Note: To create an STEPXML import template, first export relevant data, and use that as base for the import file.

For details on tags, elements, and their XML attributes, refer to the available XSD (XML Schema Definition) files in the **STEPXML Tags and Examples** topic.

Minimum Data Required

Minimum requirements for information vary based on the object type. For example, to create a new product, the file must include ID, parent ID, and object type ID, as shown in the following sample STEPXML:

```
<?xml version="1.0" encoding="UTF-8"?>
<STEP-ProductInformation ContextID="Context1" WorkspaceID="Main">
  <Products>
    <Product ID="181952LB" UserTypeID="Item" ParentID="18209">
      <Name>181952 LB</Name>
    </Product>
  </Products>
</STEP-ProductInformation>
```

Additionally, a default parent and object type can be specified using the products tag attributes. For more information, refer to the **Examples** section below and the **Products Tag in STEPXML** topic.

Using Autogenerated IDs

If an ID pattern is specified for the object type being created, the ID can be excluded, as shown in the following sample STEPXML:

```
<?xml version="1.0" encoding="UTF-8"?>
<STEP-ProductInformation ContextID="Context1" WorkspaceID="Main">
  <Products>
    <Product UserTypeID="Item" ParentID="18209">
      <Name>181952 LB</Name>
    </Product>
  </Products>
</STEP-ProductInformation>
```

For more information, refer to **Autogenerate Using Name Pattern and ID Pattern**.

Using Key Instead of ID

If a STEP Key (unique key) is specified for the object being created, the ID can be excluded. For examples and more information, refer to the **Creating Objects with Active Keys** topic of the **Keys** documentation within the **System Setup**.

Examples

As an example of minimum required data, the following STEPXML would create a new SalesItem object under SalesItemFamily-12456 with a default ID and default name, but nothing else:

```
<?xml version="1.0" encoding="utf-8"?>
<STEP-ProductInformation ContextID="Context1" WorkspaceID="Main" >
  <Products DefaultParentID="SalesItemFamily-12456" DefaultUserTypeID="SalesItem">
    <Product/>
  </Products>
</STEP-ProductInformation>
```

However, using the following STEPXML, the new object would include a specified ID and name, as well as several attribute values:

```
<?xml version="1.0" encoding="UTF-8"?>
<STEP-ProductInformation ContextID="Context1" WorkspaceID="Main">
  <Products>
    <Product ID="181951LB" UserTypeID="Item" ParentID="18209">
      <Name>18216 L B</Name>
      <Values>
        <Value AttributeID="TshirtSize">L</Value>
        <MultiValue AttributeID="CountryOfOrigin">
          <Value ID="CN">CHINA</Value>
        </MultiValue>
        <Value AttributeID="Material">100% Cotton</Value>
        <Value AttributeID="ShortItemDescription">T-shirt, short sleeve, Beefy-T,
Mens L, Blue</Value>
        <Value AttributeID="ProductWeight" UnitID="unece.unit.ONZ">2</Value>
      </Values>
    </Product>
  </Products>
</STEP-ProductInformation>
```

Cross Context Inheritance in STEPXML

When exporting across contexts, STEP can either:

- **Export all local and inherited data in every selected context.** Since most data is repeated for objects within the same hierarchy, with differences on relatively few attribute values or references, this method produces extremely large file sizes. Additionally, the downstream system must be configured to retrieve the necessary data while ignoring the insignificant data.

  This type of export is not recommended because while it does produce data by context, it also wastes system resources, negatively impacts delivery speed, and does not scale as the number of contexts increase.

- **Export all potentially relevant data in every selected context.** This method highlights the differences in the data among the contexts and produces manageable files faster. This option uses a combination of qualifier and inherited information in the file to indicate the location of the data based on the distance from the exported object.

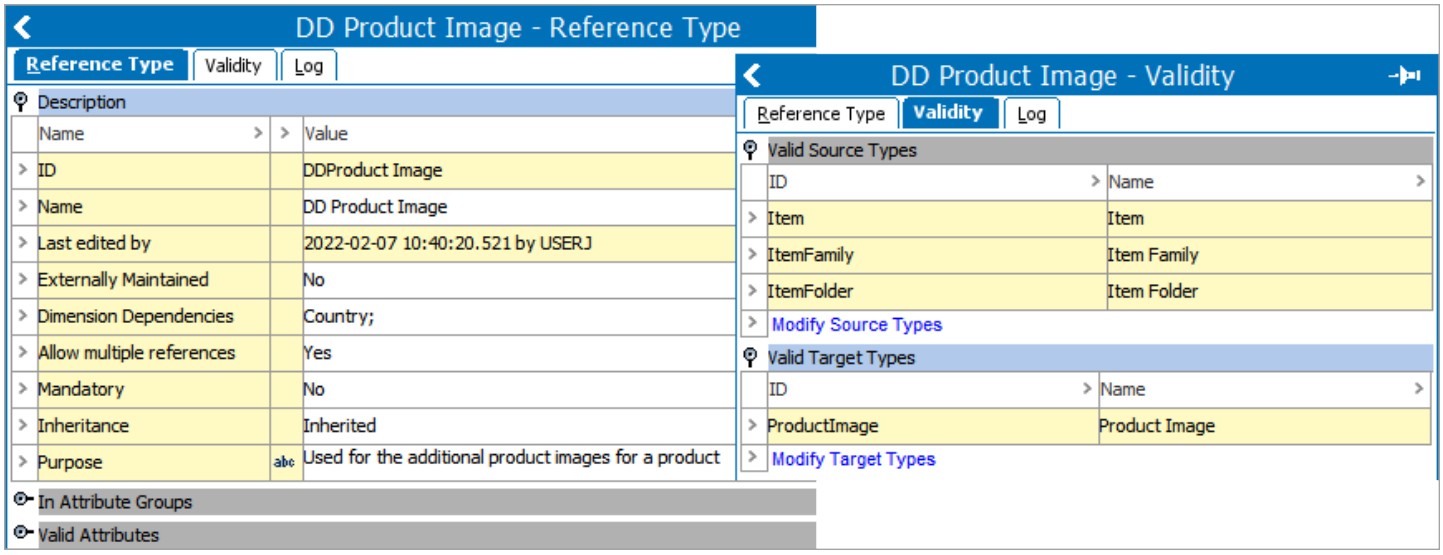
  This type of export is recommended and it uses the IncludeInherited XML attribute on the relevant tag, as defined in the **Inherit References in STEPXML** topic and the **Inherit Values in STEPXML** topic.

For details on tags, elements, and their XML attributes, refer to the available XSD (XML Schema Definition) files in the **STEPXML Tags and Examples** topic.

Data Example

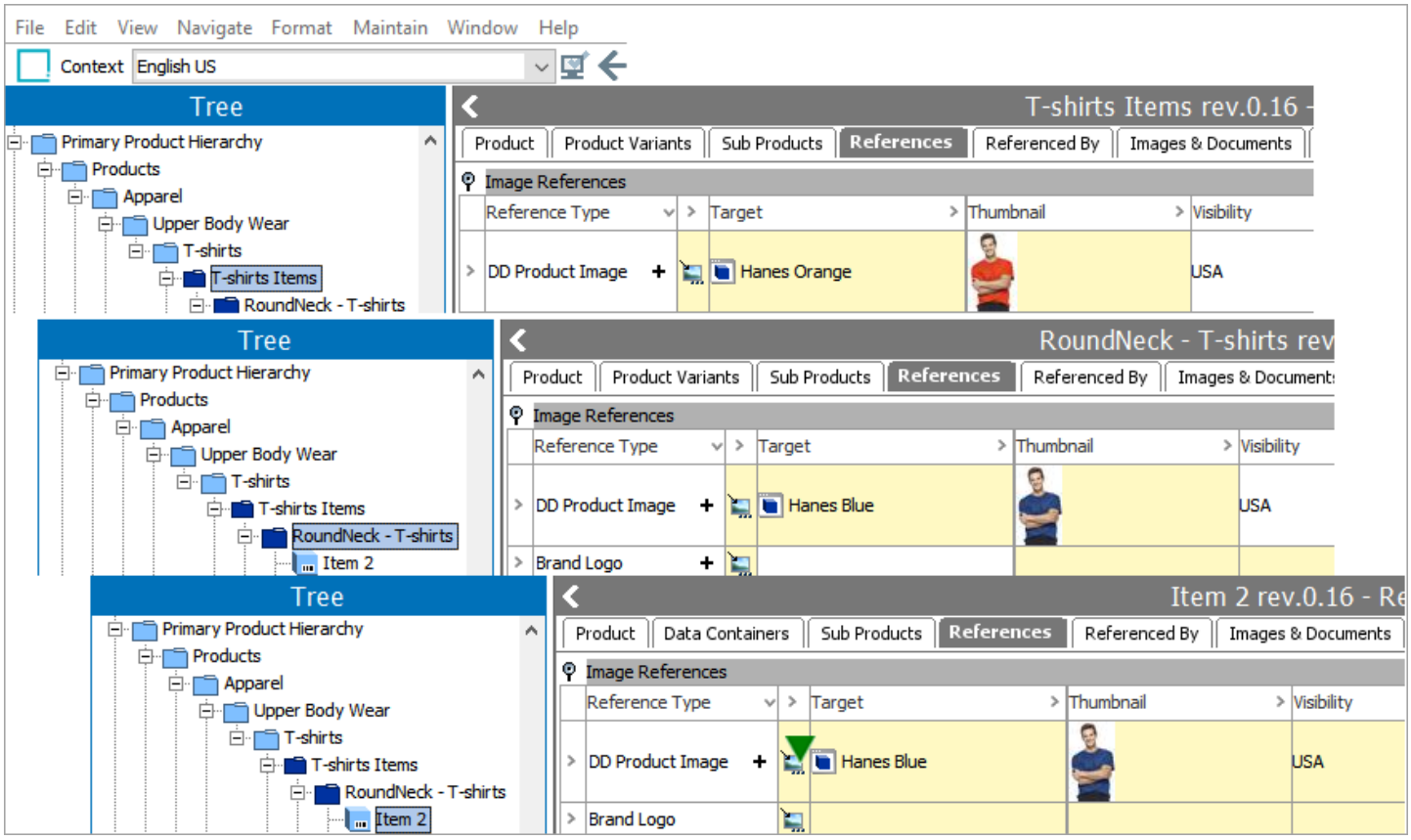
As an example of exporting all potentially relevant data, consider the following scenario.

The dimension dependent DDProductImage asset reference type is valid for three levels in the PPH.

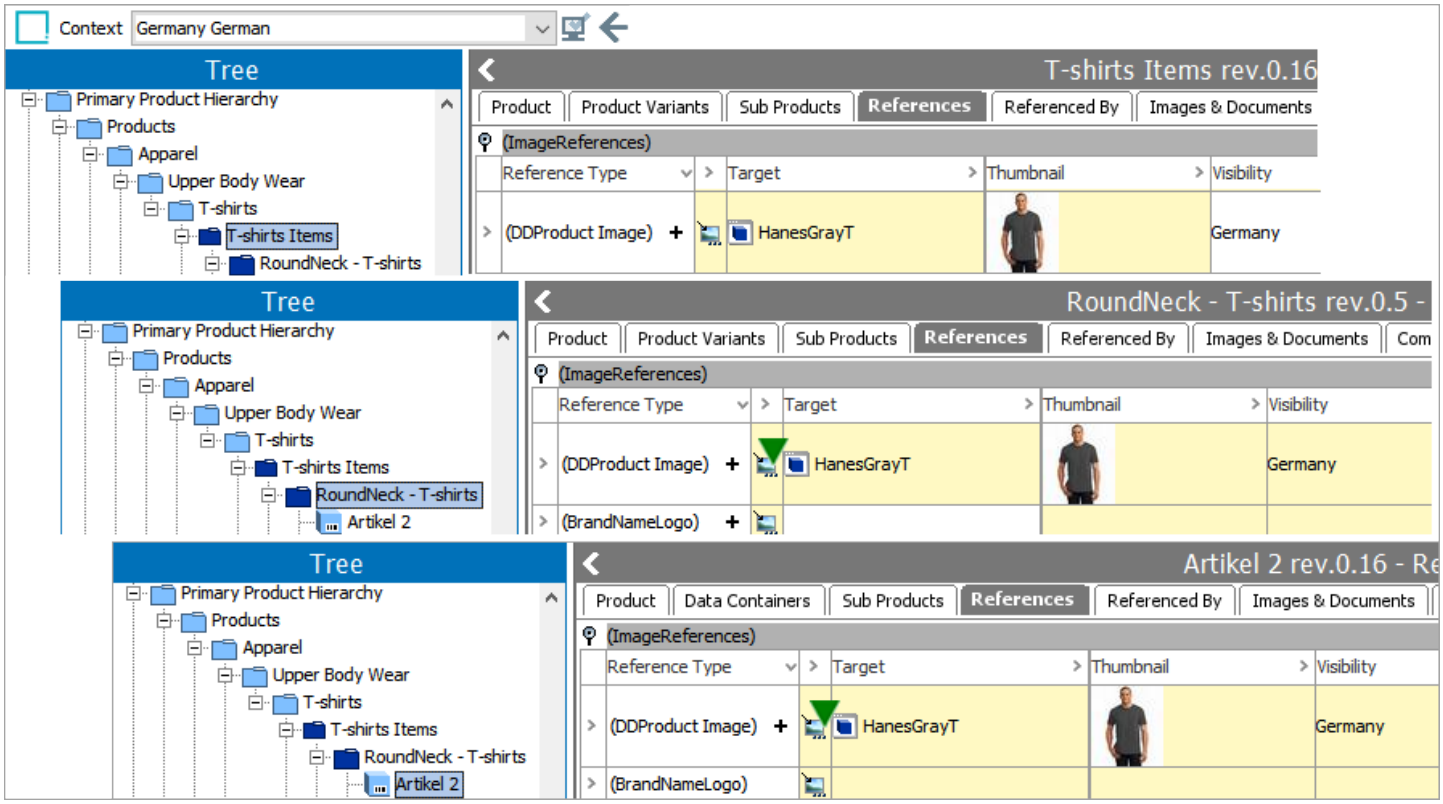


An asset has been referenced at the top level (Item Folder), and that selection has been overwritten at two lower levels (Item Family and Item) on two different contexts. Although displayed much smaller in the workbench product view user interface, a large green triangle (▼) has been manually added to these images to clearly identify the inherited values.

In the English context, for the DD Product Image (DDProduct Image) the top-level product has a referenced asset, the middle-level product has a different asset, and the bottom-level product inherits the reference to the asset.

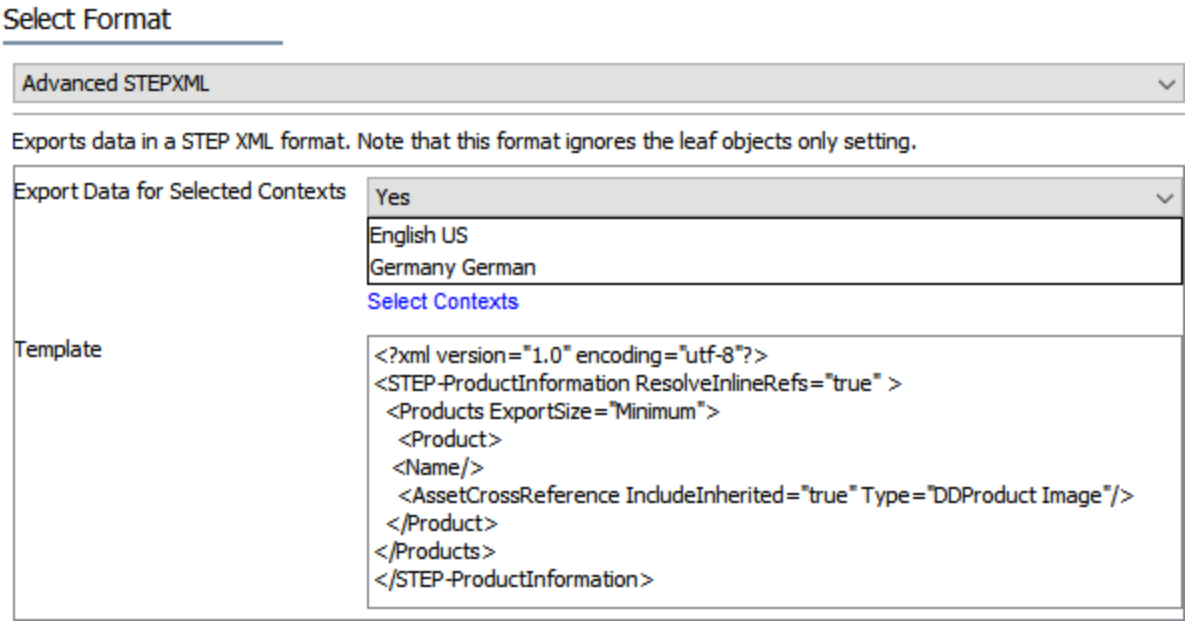


In the German context, for the DD Product Image (DDProduct Image) the top-level product has a referenced asset, the middle-level and the bottom-level products both inherit the reference to the asset.



Template

When the bottom-level product (Item 2 / Artikel 2) object is exported using the Advanced STEPXML format, both the English and the German contexts are selected in the dialog. For clarity in this example, only the name and the asset reverence are included in the template. For the AssetCrossReference tag, the IncludeInherited XML attribute is added and so is the asset reference type of 'DDProduct Image.'




```
<?xml version="1.0" encoding="utf-8"?>
<STEP-ProductInformation ResolveInlineRefs="true" >
  <Products ExportSize="Minimum">
    <Product>
   <Name/>
    <AssetCrossReference IncludeInherited="true" Type="DDProduct Image"/>
  </Product>
</Products>
</STEP-ProductInformation>
```

Result

The image below shows the significant parts of the output STEPXML file.

- The <Qualifiers> tags identify specific dimensions points or combinations of dimension points from different dimensions (as defined in the **Context Data (Qualifiers) in STEPXML** topic).
- The <Products> tag includes the data for the exported object. In this example it is a specific 'Item' object type. Information within this tag are local values and inherited values based on the template.

```
  <Qualifier ID="US">
    <DimensionPointLink DimensionPointID="US" />
    <Context ID="Context1" Priority="8" />
  </Qualifier>
  <Qualifier ID="Germany">
    <DimensionPointLink DimensionPointID="Germany" />
    <Context ID="Context5" Priority="7" />
  </Qualifier>
...removed for brevity...
  <Products>
    <Product ID="124626" UserTypeID="Item" ParentID="124625">
      <Name QualifierID="en-US">Item 2</Name>
      <Name QualifierID="German">Artikel 2</Name>
      <AssetCrossReference AssetID="20584" QualifierID="US" Type="DDProduct Image" Inherited="1" />
      <AssetCrossReference AssetID="112805" QualifierID="Germany" Type="DDProduct Image" Inherited="2" />
      <AssetCrossReference AssetID="20586" QualifierID="US" Type="DDProduct Image" Inherited="2" />
    </Product>
  </Products>
</STEP-ProductInformation>
```

As shown in the images of the Data Example section above, depending on the context, the 'DDProduct Image' asset reference value is different:

- For the German context, the asset reference is set at the top level and that same reference to the asset is inherited down to the other two levels. On the exported item (which is the bottom level), the Germany qualifier ID shows that the reference to the asset was inherited from two levels above. The 'Inherited="2"' XML attribute value indicates that the distance from the exported object to the value is 2 because the reference is a local setting for the top level, which is two objects above the one exported.
- For the US context, the asset reference is set at the first and a different reference is set at the middle level. On the exported item (which is the bottom level), the US qualifier ID shows that the asset was set at both one level above and again at two levels above. The 'Inherited="2"' XML attribute value indicates that the distance to that local value is 2 because it is set at the top level, which is two objects above the one exported. Additionally, the 'Inherited="1"' XML attribute value indicates that the value is a local setting for the middle level, which is a distance of one object above the one exported.

For a user reviewing the export file, it is clear that although only a single product was exported, all values inherited from any level in the hierarchy and for all selected contexts are included along with the inheritance distance indicators. The user can use the distance indicator to determine which value is desired. And the same is true for downstream systems.

- If the recipient of this output file is a user or system in Germany, the Qualifier ID="Germany" with Inherited="2" is the value of choice. The fact that the US context has a closer local reference is not important for German systems.
- If the recipient of this output file is a user or system in the US, the selection made closest to the exported object is desired, so the Qualifier ID ="US" with Inherited="1" is the value of choice.

Data Containers in STEPXML

By using data container types, users are able to model a composite entity (e.g., customer record) containing data containers (addresses, contact names, contact information, email addresses, phone numbers, etc.) with relevant attributes.

When importing or exporting entity objects, a number of tags are used to identify data containers upon export, and to build out the data container structure for import. Additionally, data container types and data containers can be set up using STEPXML tags.

For details on tags, elements, and their XML attributes, refer to the available XSD (XML Schema Definition) files in the **STEPXML Tags and Examples** topic.

Setting Up Data Containers

The **DataContainerTypes** tag houses all of the data container details, and the **DataContainerType** tag builds out the structure of the data container.

The below STEPXML shows an example setup that includes the data container types: Shipping Address, Email, and Main Address.

- The <AttributeGroupLink AttributeGroupID="EntityDataContainers"/> tag indicates the ID of the attribute group in which the data containers reside. In this case, it is the EntityDataContainers attribute group.
- The <AttributeLink AttributeID="Country"/> tag indicates the ID of the attributes are associated with each data container type. Country is just one example.
- <UserTypeLink UserTypeID="CD_Customer"/> indicates which entity object type is valid for the data container type. The ID is shown (e.g., CD_Customer). If this tag combo does not appear in the STEPXML, then validity has not been set on that data container type (shown in the Email data container type example STEPXML).

```
<DataContainerTypes>
   <DataContainerType ID="ShippingAddress" MultiValued="true" IDPattern="[id]"
Referenced="true">
       <Name>Shipping Address</Name>
       <AttributeGroupLink AttributeGroupID="EntityDataContainers"/>
       <AttributeLink AttributeID="Street"/>
       <AttributeLink AttributeID="Country"/>
       <AttributeLink AttributeID="State"/>
       <AttributeLink AttributeID="Zip"/>
       <AttributeLink AttributeID="CountryISOCode"/>
       <AttributeLink AttributeID="City"/>
       <UserTypeLink UserTypeID="CD_Customer"/>
   </DataContainerType>
   <DataContainerType ID="Email" MultiValued="false">
       <Name>Email</Name>
       <AttributeGroupLink AttributeGroupID="CustomerData"/>
       <AttributeGroupLink AttributeGroupID="EntityDataContainers"/>
       <AttributeLink AttributeID="S-Email"/>
```

```
   </DataContainerType>
   <DataContainerType ID="MainAddress" MultiValued="false" IDPattern="MainAddress-
[id]" Referenced="true">
       <Name>Main Address</Name>
       <AttributeGroupLink AttributeGroupID="EntityDataContainers"/>
       <AttributeLink AttributeID="Street"/>
       <AttributeLink AttributeID="Country"/>
       <AttributeLink AttributeID="State"/>
       <AttributeLink AttributeID="Zip"/>
       <AttributeLink AttributeID="CountryISOCode"/>
       <AttributeLink AttributeID="City"/>
       <UserTypeLink UserTypeID="CD_Customer"/>
   </DataContainerType>
</DataContainerTypes>
```

DataContainerType Tag and XML Attributes

The **DataContainerType** tag can include the following valid XML attributes:

MultiValued	• False - Indicates that the data container type only allows for a single data container • True - Indicates that the data container type allows for multiple data containers
ID	• Displays the ID for the data container type or data container. When importing, the ID attribute can be omitted if autogenerated IDs are enabled. **Note:** Autogenerated IDs are only supported with single container types. This option is disabled for multi-container types.
IDPattern	• Displays the ID pattern set up for the data container type, if applicable. This pattern will apply to all data containers of that type.
Referenced	• When exporting data and choosing to include Data Container Type Definitions, Referenced="true" displays within the DataContainerType tag to indicate that a data container of that type is referenced in other data within the same export. • If there is no Referenced info shown, then a data container of that type is not being exported in the same data set.

Adding and Maintaining Data Containers

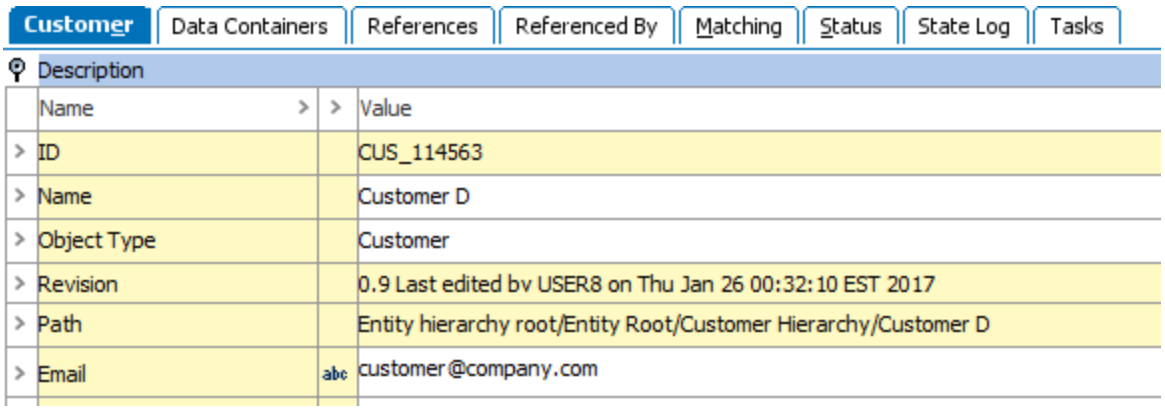
The **DataContainers** tag indicates where, within the entity object tags, the data container information starts.

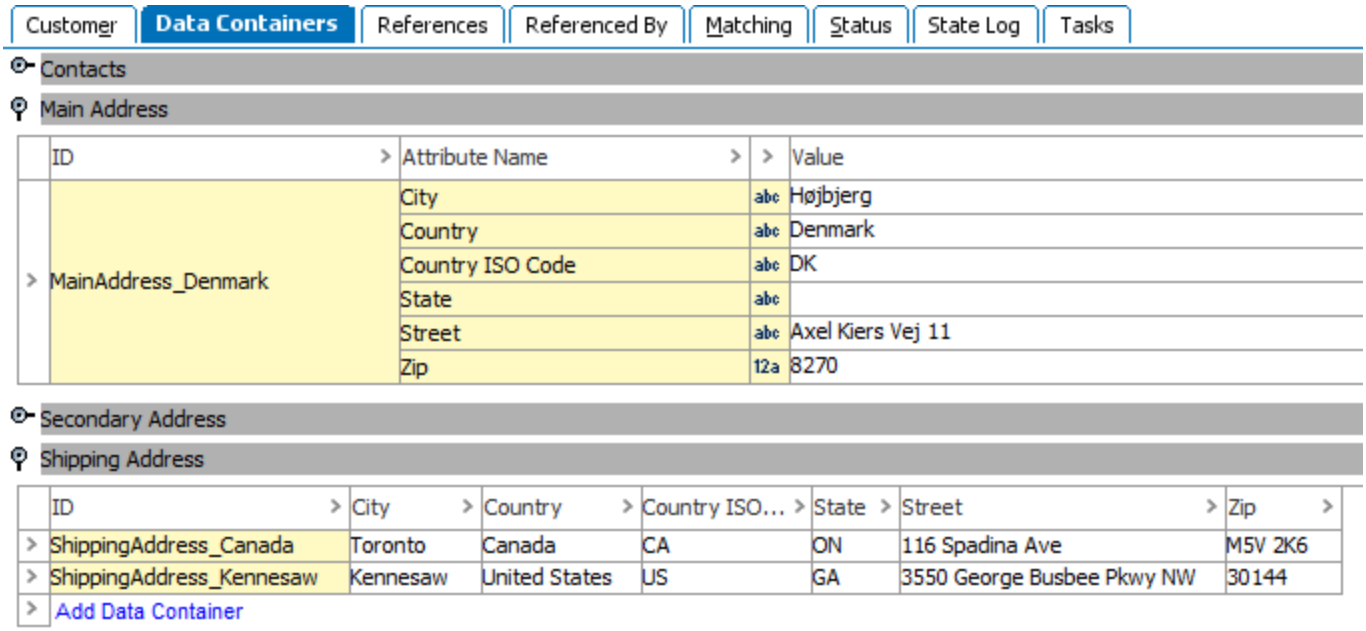
If a data container type allows for multiple containers, then it is indicated by the **MultiDataContainer** tag. Individual data containers are identifiable by the **DataContainer** tag. The **Value** tag, with the AttributeID XML attribute, is used to show / set up the attributes and attribute values.

Example STEPXML code is shown below:

```
<Entity ID="CUS_114563" UserTypeID="CD_Customer">
   <Name>Customer D</Name>
       <EntityCrossReference EntityID="ADD_120042" Type="CustomerToAddress"/>
   <Values>
       <Value AttributeID="S-Email">customer@company.com</Value>
   </Values>
   <DataContainers>

   <MultiDataContainer Type="ShippingAddress">
       <DataContainer ID="ShippingAddress_Kennesaw">
          <Values>
               <Value AttributeID="Street">3550 George Busbee Pkwy NW</Value>
               <Value AttributeID="Country">United States</Value>
               <Value AttributeID="Zip">30144</Value>
               <Value AttributeID="State">GA</Value>
               <Value AttributeID="City">Kennesaw</Value>
               <Value AttributeID="CountryISOCode">US</Value>
          </Values>
       </DataContainer>
       <DataContainer ID="ShippingAddress_Canada">
          <Values>
               <Value AttributeID="Street">116 Spadina Ave</Value>
               <Value AttributeID="Country">Canada</Value>
               <Value AttributeID="Zip">M5V 2K6</Value>
               <Value AttributeID="State">ON</Value>
               <Value AttributeID="City">Toronto</Value>
               <Value AttributeID="CountryISOCode">CA</Value>
          </Values>
          </DataContainer>
   </MultiDataContainer>
   <DataContainer ID="MainAddress_Denmark" Type="MainAddress">
       <Values>
               <Value AttributeID="Street">Axel Kiers Vej 11</Value>
               <Value AttributeID="Country">Denmark</Value>
               <Value AttributeID="Zip">8270</Value>
               <Value AttributeID="City">Højbjerg</Value>
               <Value AttributeID="CountryISOCode">DK</Value>
       </Values>
       </DataContainer>
   </DataContainers>
</Entity>
```

This is how the above code renders in the workbench:



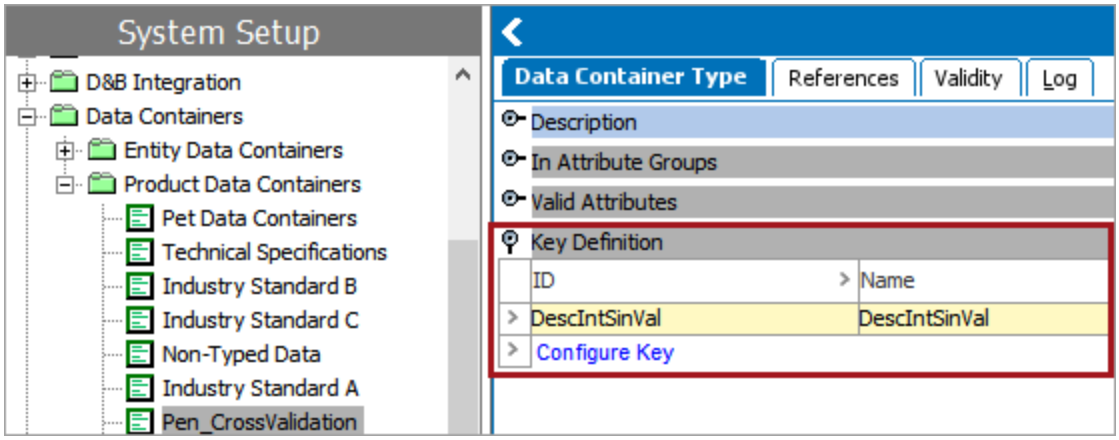


<KeyDefinition> Tag for Data Container Keys

Users can make use of the <KeyDefinition> tag when importing data containers that contain key definitions. In the example below, the attribute "DescIntSinVal" is designated as a key definition for the data container 'Pen_CrossValidation.'

```
<DataContainerType ID="261032" MultiValued="true" Inherited="false"
  <Name>Pen_CrossValidation</Name>
  <AttributeGroupLink AttributeGroupID="CrossValidation_PenAttr"/>
  <AttributeGroupLink AttributeGroupID="ProductDataContainers"/>
  <AttributeLink AttributeID="NibType"/>
  <AttributeLink AttributeID="Model"/>
  <AttributeLink AttributeID="DescIntSinVal"/>
  <AttributeLink AttributeID="PDS Channel"/>
  <AttributeLink AttributeID="PenBrand"/>
  <AttributeLink AttributeID="getObjectType"/>
  <UserTypeLink UserTypeID="Item"/>
  <KeyDefinition>
  <Attribute ID="DescIntSinVal"/>
  </KeyDefinition>
</DataContainerType>
```

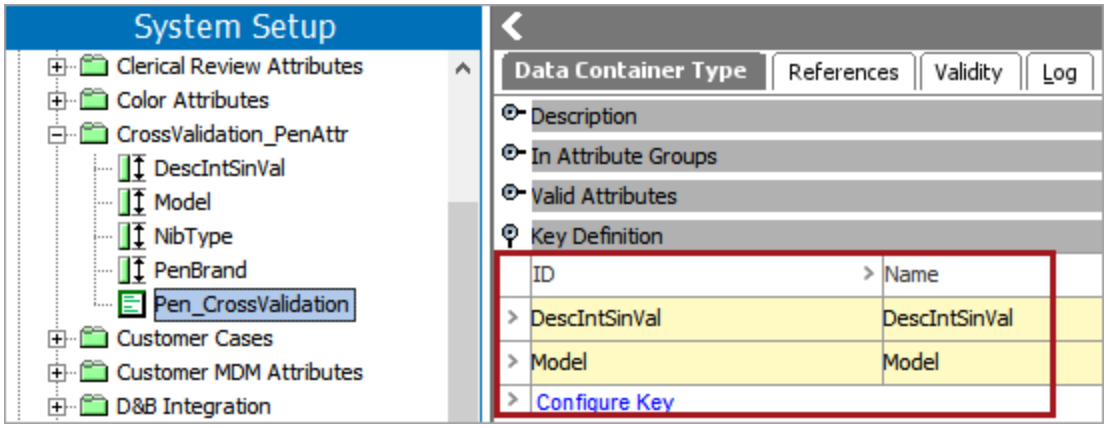
This is how the above code renders in workbench:



In the example below, the attribute 'Model' has been added as part of the key definition for the 'Pen_CrossValidation' data container.

```
<DataContainerType ID="261032" MultiValued="true" Inherited="false"
  <Name>Pen_CrossValidation</Name>
  <AttributeGroupLink AttributeGroupID="CrossValidation_PenAttr"/>
  <AttributeGroupLink AttributeGroupID="ProductDataContainers"/>
  <AttributeLink AttributeID="NibType"/>
  <AttributeLink AttributeID="Model"/>
  <AttributeLink AttributeID="DescIntSinVal"/>
  <AttributeLink AttributeID="PDS Channel"/>
  <AttributeLink AttributeID="PenBrand"/>
  <AttributeLink AttributeID="getObjectType"/>
  <UserTypeLink UserTypeID="Item"/>
  <KeyDefinition>
  <Attribute ID="Model"/>
  <Attribute ID="DescIntSinVal"/>
  </KeyDefinition>
</DataContainerType>
```

This is how the above code renders in workbench:



For more information on data container keys, including the specific criteria necessary for attributes and/or reference types to be included as part of a key definition, refer to the **Data Container Keys** topic in the **System Setup** documentation.

Delete Objects in STEPXML

Products, entities, classifications, assets, and attributes can be deleted via STEPXML. The delete action varies based on the object type and the revisability setting, as follows:

- For workspace revisable object types (all object types except entities), delete moves the object to the Recycle Bin.
- For attributes and global revisable entities, delete purges the object.

Note: LOVs and units cannot be deleted by importing a STEPXML file.

For more information on revisability, refer to **Revisions** or **Revisability on Entity Object Type**.

For details on tags, elements, and their XML attributes, refer to the available XSD (XML Schema Definition) files in the **STEPXML Tags and Examples** topic.

For example, the following STEPXML moves four product objects (which are always workspace revisable) to the Recycle Bin.

```
<?xml version="1.0" encoding="utf-8"?>
<STEP-ProductInformation ContextID="Context1" WorkspaceID="Main">
  <DeleteProducts>
    <DeleteProduct ID="L7629"/>
    <DeleteProduct ID="L7321"/>
    <DeleteProduct ID="L7847"/>
    <DeleteProduct ID="L7623"/>
  </DeleteProducts>
</STEP-ProductInformation>
```

Event-Based OIEP Triggered by Deleting Products, Classifications, and Assets

When an outbound integration endpoint (OIEP) is configured to listen for Delete events, use the **DeleteProducts**, **DeleteClassifications**, or **DeleteAssets** tags to output information about deleted objects. After deleting the object, the OIEP is triggered by the Approve Deletion action within the Recycle Bin.

```
<DeleteProducts/>
<DeleteClassifications/>
<DeleteAssets/>
```

Note: The Force Delete option is not available for products, classifications, or assets when an OIEP is configured to deliver deletion information.

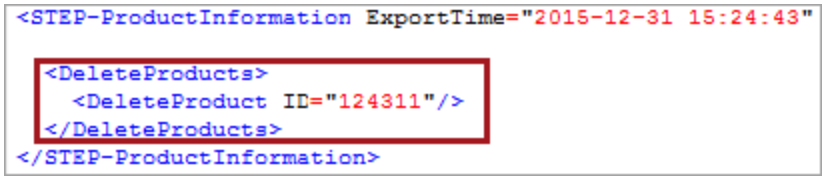
When using the Mongo delivery method, specific settings are required to deliver delete events. For details, refer to the **Delivering Delete Events** section of the **Mongo Delivery Method** topic in this guide.

Template

```
<STEP-ProductInformation ExportDeletedData="true">
  <DeleteProducts/>
</STEP-ProductInformation>
```

Results

Once the deleted Product (Classification or Asset) displays in the Recycle Bin, use Approve Deletion to trigger the OIEP. The ID of the deleted object is included in the OIEP output.



For more information on event-based OIEPs, refer to the **Outbound Integration Endpoints** documentation.

Delete Product Reference in STEPXML

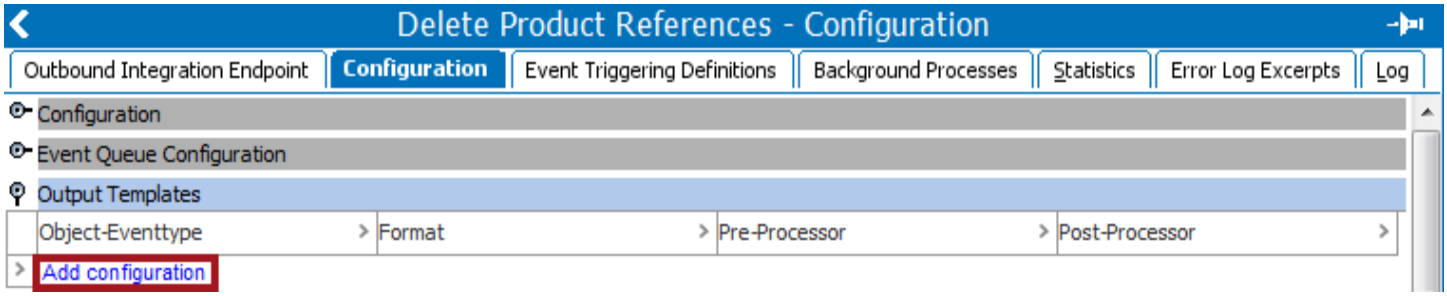
When an outbound integration endpoint (OIEP) is configured to listen for Modify events, use the **STEP-ProductInformation** tag with the **ExportDeletedData** STEPXML attribute to output information about deleted product references. After deleting the reference, approving the product triggers the OIEP.

This tag is not valid for inbound STEPXML.

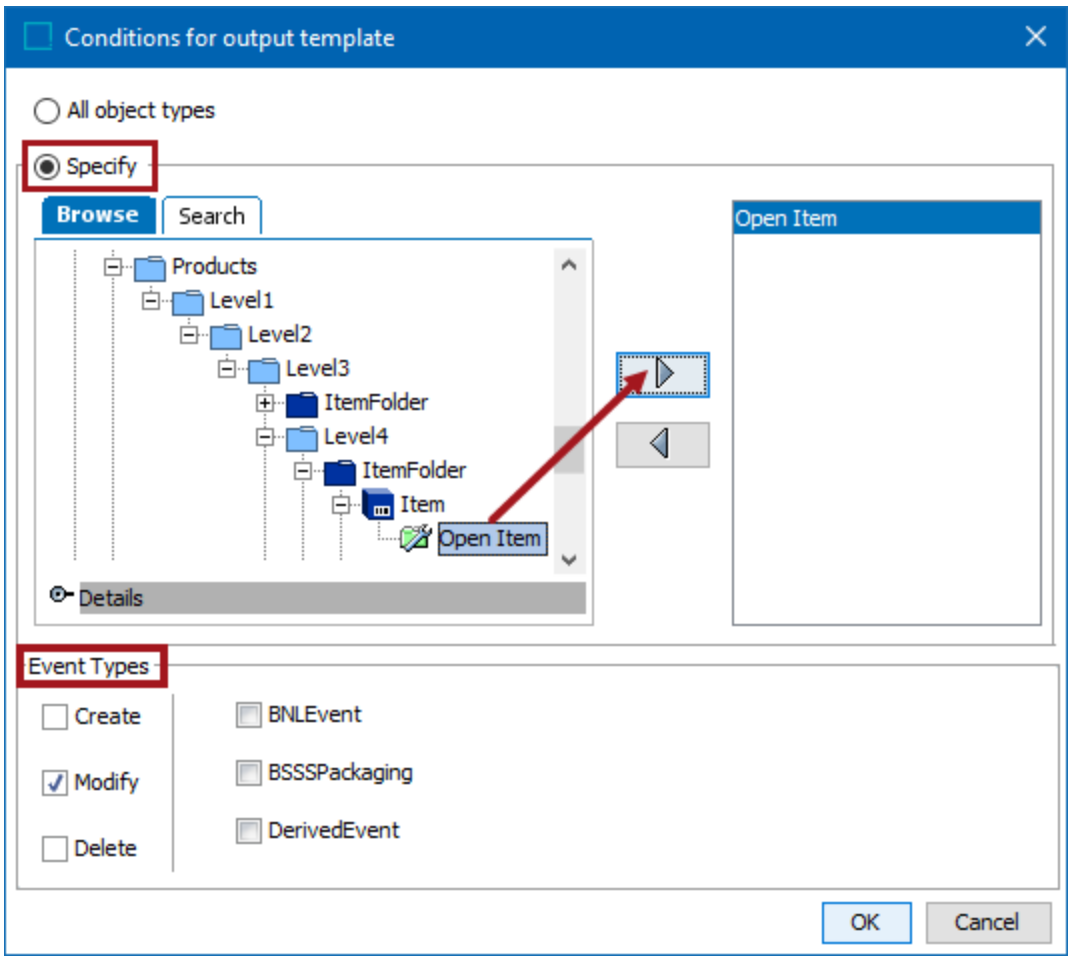
For details on tags, elements, and their XML attributes, refer to the available XSD (XML Schema Definition) files in the **STEPXML Tags and Examples** topic.

Object Type and Event Type Selection

View the OIEP and open the Configuration tab. Under the Output Templates flipper, click the **Add configuration** link.



Set the Object Types and Event Types.

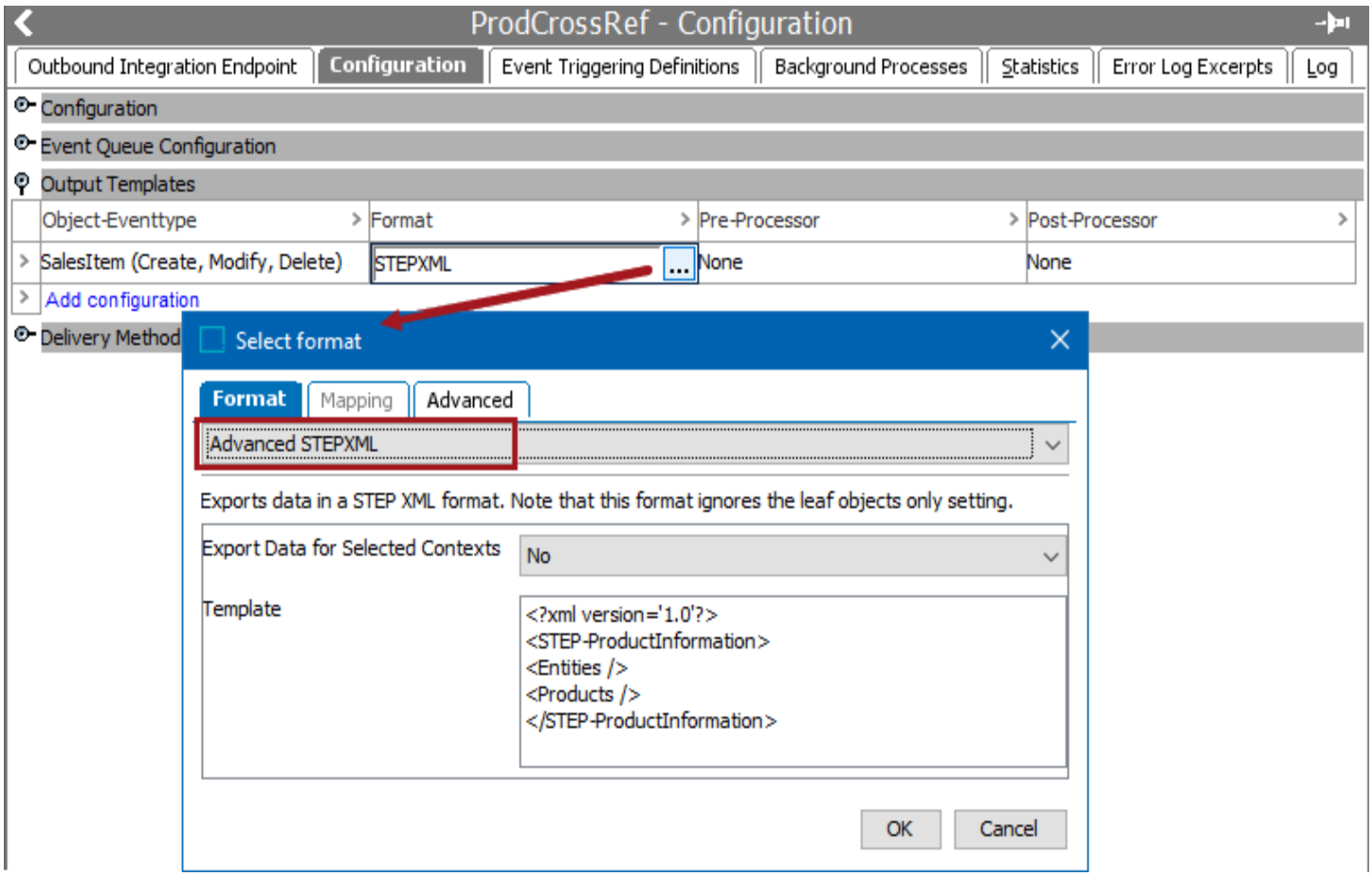


Template

In the Output Template, click the ellipsis button (...), and set the format to **Advanced STEPXML**.

Provide the following text in the Template field.

```
<?xml version='1.0'?>
<STEP-ProductInformation ExportDeletedData="true">
<Products/>
</STEP-ProductInformation>
```



Result

When a Product Reference is deleted and the product is approved, the **DeleteProductCrossReference** tag indicates that the Product Reference is intended to be removed in the downstream system.

The previous settings generated the following Advanced STEPXML output document.

```
<STEP-ProductInformation ExportTime="2015-09-30 11:12:22" ExportContext="Context1" ContextID="Cont

  <Products>
    <Product ID="22163" UserTypeID="SalesItem" ParentID="22157">
      <Name>22163-01</Name>
      <ProductCrossReference ProductID="22155" Type="PrimaryDataSource"/>

      <AssetCrossReference AssetID="22215" Type="PrimaryProductImage"/>

      <Values>
        <Value AttributeID="AnnualSalesForecastMaximum">300</Value>
        <Value AttributeID="AnnualSalesForecast,Minimum">100</Value>
        <Value AttributeID="FeatureBullet3">26 slats of layer-glued birch adjust to your body weig
        <Value AttributeID="SellingPrice" UnitID="iso4217.unit.USD">1299</Value>
        <Value AttributeID="FeatureBullet1">Made of solid wood, which is a durable and warm natura
        <Value AttributeID="DescriptionTable">Comfy bed, expresso, Queen</Value>
        <Value AttributeID="FeatureBullet2">Adjustable bed sides allow you to use mattresses of di
        <Value AttributeID="DescriptionLong">Made of solid wood, which is a durable and warm natu
        <Value AttributeID="DescriptionWeb">Made of solid wood, which is a durable and warm natura
        <Value AttributeID="SalesItemShortDescription">Bed, Expresso, Comfy, Queen</Value>
        <Value AttributeID="SellingPriceUOM">EA</Value>
        <Value AttributeID="Parent" Derived="true">Beds Sales Items</Value>
        <Value AttributeID="Path" Derived="true">Products | Furniture | Bedroom | Beds | Beds Sale
        <Value AttributeID="Category" Derived="true">Primary Product Hierarchy | Products | Furnit
      </Values>
      <DeleteProductCrossReference ProductID="22168" Type="PrimarySupplierItem"/>
    </Product>
  </Products>
</STEP-ProductInformation>
```

Delete Values in STEPXML

Upon import, to delete an existing value, include the appropriate value element and leave the value part blank.

For information on deleting values in other formats, refer to the **Delete Values - Map Inbound** topic in this guide.

For details on tags, elements, and their XML attributes, refer to the available XSD (XML Schema Definition) files in the **STEPXML Tags and Examples** topic.

The example below updates the product with ID 'L6576' by setting the value for 'Horizontal Pixels' to 1920 and removing any existing value for 'Manufacturer's Description.'

```
<?xml version="1.0" encoding="utf-8"?>
<STEP-ProductInformation ContextID="Context1" WorkspaceID="Main">
  <Products>
    <Product ID="L6576">
      <Name>Panasonic Viera TC-P65VT50</Name>
      <Values>
        <Value AttributeID="Manufacturer's Description"/>
        <Value AttributeID="Horizontal Pixels">1920</Value>
      </Values>
    </Product>
  </Products>
</STEP-ProductInformation>
```

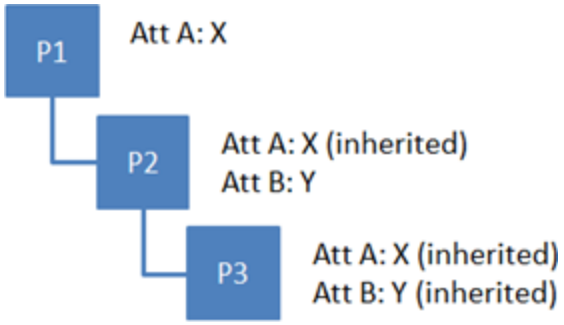
Export Inheritance in STEPXML Example

By default, inherited values and references / links are not exported in STEPXML.

For details on tags, elements, and their XML attributes, refer to the available XSD (XML Schema Definition) files in the **STEPXML Tags and Examples** topic.

In the sample hierarchy below:

- Product 1 (P1) includes the local value 'X' for attribute A (Att A).
- Product 2 (P2) inherits the value 'X' from attribute A (Att A), and has the local value 'Y' for attribute B (Att B).
- Product 3 (P3) inherits the value 'X' from attribute A (Att A) and inherits the value 'Y' from attribute B (Att B).



Using default settings to export the sample hierarchy, only local values are output and results in the following:

- If only P3 is exported, no values are exported since no attribute values are local (both are inherited).
- If P2 and P3 are exported, the only value is Att B from P2. No values for P3 are exported since no attribute values are local (both are inherited).
- If P1, P2, and P3 are exported, the only values are Att A from P1, Att B from P2. No values for P3 are exported since no attribute values are local (both are inherited).

Override Default Behavior

Overriding the default behavior means that an inherited attribute value is exported once when the object that holds the local value is not exported.

Using the override and continuing with the above example data, the values are output as follows:

- If only P3 is exported, the value for Att A (inherited) and Att B (inherited) are exported.
- If P2 and P3 are exported, for P2, the value for AttA (inherited) and the value for Att B (local) are exported. For P3, no values are exported since both values already appear on P2.
- If only P2 is exported, the value for AttA (inherited) and the value for Att B (local) are exported.
- If P1, P2, and P3 are exported, for P1, the value for Att A (local) is exported. For P2, the value for Att B (local) is exported. For P3, no values are exported since both values already appear on other objects.

For more information on setting the override, refer to the following topics:

- For STEPXML format, use the Data Objects parameter **Include Inherited Data** = 'Yes' as defined in the **STEPXML Outbound Parameters** topic.
- For Advanced STEPXML format, use the **IncludeInherited** tag options as defined in the **Inherit Values in STEPXML** topic.
- For Advanced STEPXML format, data for multiple contexts can be exported efficiently as defined in the **Cross Context Inheritance in STEPXML** topic.

Filter Data Containers in STEPXML

When the output from STEP should include only specific data container types, use the **DataContainer** tag with the **Type** attribute. Only data containers identified by the ID are exported. The data container filtering capabilities are available on entities and products.

For details on tags, elements, and their XML attributes, refer to the available XSD (XML Schema Definition) files in the **STEPXML Tags and Examples** topic.

For single instance data containers, the following is used:

```
<DataContainers>
   <DataContainer Type="AddrMainAddressDataContainer"/>
</DataContainers>
```

For multi-instance data containers, the following is used:

```
<DataContainers>
    <MultiDataContainer Type="ContEmailDataContainer"/>
</DataContainers>
```

Consider the scenario where a template delivers the parent structure for the approved entities.

Template

Single Reference Data Container

```
<?xml version='1.0'?>
<STEP-ProductInformation ResolveInlineRefs="true" FollowOverrideSubProducts="true">
    <Entities ExportSize="Minimum">
        <Entity>
        <Name/>
            <Values>
                <Value AttributeID="IndLastName"/>
            </Values>
            <DataContainers>
                <DataContainer Type="AddrMainAddressDataContainer"/>
            </DataContainers>
        </Entity>
    </Entities>
</STEP-ProductInformation>
```

In the following multi-instance data container export will only export the 'ContEmail' attribute in the 'ContEmailDataContainer' data container, ignoring all other attributes in this data container.

Multi-Instance Data Container

```
<?xml version='1.0'?>
<STEP-ProductInformation FollowOverrideSubProducts="true" ResolveInlineRefs="true">
    <Entities ExportSize="Minimum">
        <Entity>
           <Name/>
               <Values>
                   <Value AttributeID="IndLastName"/>
               </Values>
                <DataContainer>
                 <MultiDataContainer Type="ContEmailDataContainer">
                    <Values>
                         <Value AttributeID="ContEmail"/>
                     </Values>
                 </MultiDataContainer>
             </DataContainer>
        </Entity>
    </Entities>
</STEP-ProductInformation>
```

Known Limitation

Any Data Container filtering done on Embedded and Referenced objects in an Advanced STEPXML template will be ignored. Instead, the Data Container filtering done on the exported object itself will be used on the Embedded and Referenced objects. For more information, refer to the **Referenced and Embedded XML Attributes in STEPXML** in this documentation.

Filter Products in STEPXML

When the output from STEP should include only specific objects, use the **FilterUserType** tag with the **ID** XML attribute. Only objects identified by the ID are exported.

```
<FilterUserType ID="ItemFolder"/>
```

For details on tags, elements, and their XML attributes, refer to the available XSD (XML Schema Definition) files in the **STEPXML Tags and Examples** topic.

Consider the scenario where a template delivers the parent structure for the approved product. Although six (6) product levels exist, only three (3) of the levels and their parents are required.

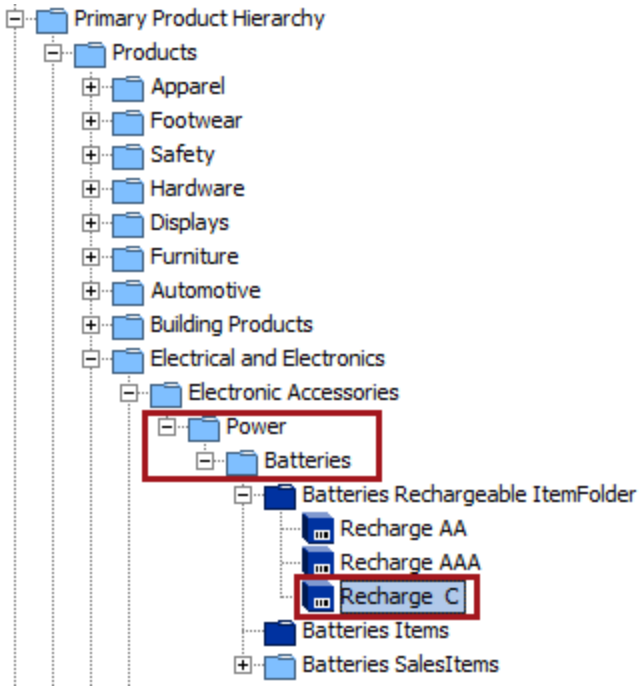
Use multiple instances of the **FilterUserType** tag to include all of the required objects and to filter out the objects that are not needed.

Template

Without Filtering	With Filtering
<?xml version="1.0" encoding="UTF-8"?> <STEP-ProductInformation> <Products> <Product IncludeParent="true"> <Name/> <ProductCrossReference/> <Product/> </Product> </Products> </STEP-ProductInformation>	<?xml version="1.0" encoding="UTF-8"?> <STEP-ProductInformation> <Products> <FilterUserType ID="Level3"/> <FilterUserType ID="Level4"/> <FilterUserType ID="Level5"/> <Product IncludeParent="true"> <Name/> <ProductCrossReference/> <Product/> </Product> </Products> </STEP-ProductInformation>

Modified Object

For each template, the same single item (Recharge C) is modified and approved, which could trigger an OIEP.


Primary Product Hierarchy
Products
Apparel
Footwear
Safety
Hardware
Displays
Furniture
Automotive
Building Products
Electrical and Electronics
Electronic Accessories
Power
Batteries
Batteries Rechargeable ItemFolder
Recharge AA
Recharge AAA
Recharge C
Batteries Items
Batteries SalesItems

Results

Without Filtering

```
<STEP-ProductInformation ExportTime="2015-12-17 15:13:24" ExportContext="Context1" ContextID="C

  <Products>
    <Product ID="Product hierarchy root" UserTypeID="Product user-type root" Selected="false">
      <Name>Primary Product Hierarchy</Name>
      <Product ID="ProductsRoot" UserTypeID="Products" Selected="false">
        <Name>Products</Name>
        <Product ID="8302" UserTypeID="Level1" Selected="false">
          <Name>Electrical and Electronics</Name>
          <Product ID="20875" UserTypeID="Level2" Selected="false">
            <Name>Electronic Accessories</Name>
            <Product ID="20876" UserTypeID="Level3" Selected="false">
              <Name>Power</Name>
              <Product ID="20878" UserTypeID="Level4" Selected="false">
                <Name>Batteries</Name>
                <Product ID="123942" UserTypeID="ItemFolder" Selected="false">
                  <Name>Batteries Rechargable Items</Name>
                  <Product ID="123963" UserTypeID="Level5">
                    <Name Changed="true">Recharge  C</Name>
                      <ProductCrossReference ProductID="20882" Type="PrimaryDataSource"/>
                      <ProductCrossReference ProductID="123943" Type="CrossReference"/>
                      <ProductCrossReference ProductID="123944" Type="CrossReference"/>

                  </Product>
                </Product>
              </Product>
            </Product>
          </Product>
        </Product>
      </Product>
    </Product>
  </Products>
</STEP-ProductInformation>
```

With Filtering

```
<STEP-ProductInformation ExportTime="2015-12-17 15:23:43" ExportContext="Context

  <Products>
    <Product ID="20876" UserTypeID="Level3" ParentID="20875" Selected="false">
      <Name>Power</Name>
      <Product ID="20878" UserTypeID="Level4" Selected="false">
        <Name>Batteries</Name>
        <Product ID="123963" UserTypeID="Level5" ParentID="123942">
          <Name Changed="true">Recharge C</Name>
            <ProductCrossReference ProductID="20882" Type="PrimaryDataSource"/>
            <ProductCrossReference ProductID="123943" Type="CrossReference"/>
            <ProductCrossReference ProductID="123944" Type="CrossReference"/>

        </Product>
      </Product>
    </Product>
  </Products>
</STEP-ProductInformation>
```

Filter References in STEPXML

When a downstream system requires a limited set of STEP references, use a reference tag with the **Type** XML attribute to indicate the required asset, classification, or product reference.

Use multiple instances of each reference tag to include all of the required objects.

For details on tags, elements, and their XML attributes, refer to the available XSD (XML Schema Definition) files in the **STEPXML Tags and Examples** topic.

Asset References

For example, to output asset references with an ID of 'PrimaryProductImage' and 'ProductImage', multiple **AssetCrossReference** tags are used with the **Type** XML attribute as follows:

```
<AssetCrossReference Type="PrimaryProductImage"/>
<AssetCrossReference Type="ProductImage"/>
```

Classification References

For example, to output only the classification reference with an ID of 'Supplier', the **ClassificationReference** tag is used with the **Type** XML attribute as follows:

```
<ClassificationReference Type="Supplier"/>
```

Product References

For example, to output only the product reference with an ID of 'CrossReference', the **ProductCrossReference** tag is used with the **Type** XML attribute as follows:

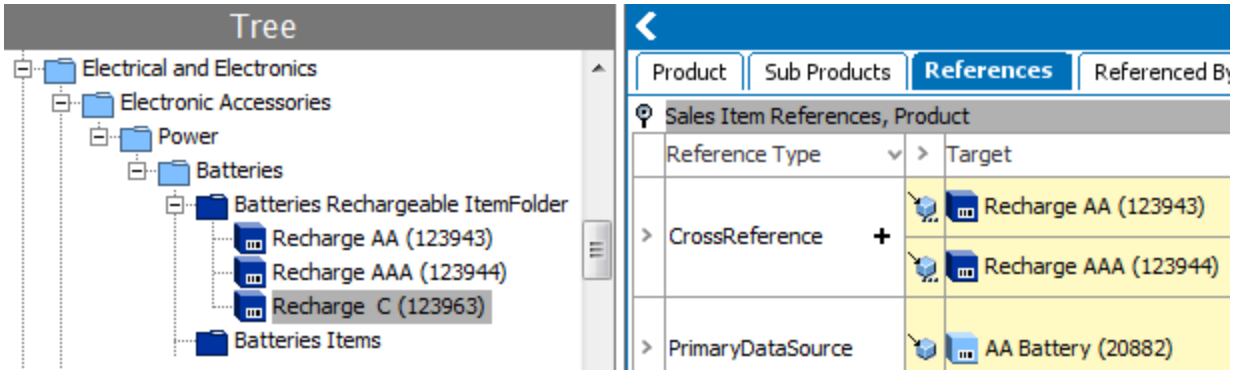
```
<ProductCrossReference Type="CrossReference"/>
```

Template

Without Filtering	With Filtering
<?xml version="1.0" encoding="UTF-8"?> <STEP-ProductInformation> <Products> <Product> <Name/> <ClassificationReference/> <ProductCrossReference/> </Product> </Products> </STEP-ProductInformation>	<?xml version="1.0" encoding="UTF-8"?> <STEP-ProductInformation> <Products> <Product> <Name/> <ClassificationReference/> <ProductCrossReference Type="CrossReference"/> </Product> </Products> </STEP-ProductInformation>

Modified Object

For each template, the same single item (Recharge C) is modified and approved, which can trigger the OIEP.



Results

Without filtering, all references for the exported product are output.

Without Filtering

```
<STEP-ProductInformation ExportTime="2015-12-16 16:23:02" ExportContext="Cxt

  <Products>
    <Product ID="123963" UserTypeID="Item" ParentID="123942">
      <Name Changed="true">Recharge  C</Name>
      <ClassificationReference ClassificationID="SupplyAll" Type="Supplier">
        <MetaData>
  <Value AttributeID="QtyOfItems" Derived="true">1</Value>
</MetaData>
      </ClassificationReference>

      <ProductCrossReference ProductID="20882" Type="PrimaryDataSource"/>
      <ProductCrossReference ProductID="123943" Type="CrossReference"/>
      <ProductCrossReference ProductID="123944" Type="CrossReference"/>

    </Product>
  </Products>
</STEP-ProductInformation>
```

With filtering, only references of the specified type for the exported product are output.

With Filtering

```
<STEP-ProductInformation ExportTime="2015-12-16 16:23:02" ExportContext="Cxt

  <Products>
    <Product ID="123963" UserTypeID="Item" ParentID="123942">
      <Name Changed="true">Recharge  C</Name>
      <ClassificationReference ClassificationID="SupplyAll" Type="Supplier">
        <MetaData>
  <Value AttributeID="QtyOfItems" Derived="true">1</Value>
</MetaData>
      </ClassificationReference>

      <ProductCrossReference ProductID="123943" Type="CrossReference"/>
      <ProductCrossReference ProductID="123944" Type="CrossReference"/>

    </Product>
  </Products>
</STEP-ProductInformation>
```

Filter Values in STEPXML

Use the following values tag to output all the attribute values:

```
<Values/>
```

Or specify them individually or by group using:

```
<Values>
    <Value AttributeID="Colour"/>
    <Value AttributeID="Size"/>
    <Value AttributeID="AllWebAttributes"/>
</Values>
```

For details on tags, elements, and their XML attributes, refer to the available XSD (XML Schema Definition) files in the **STEPXML Tags and Examples** topic.

To filter the attribute values, use the **AttributeID** XML attribute.

- When the output should include a specific attribute's value, include the attribute ID.
- When the output should include attributes within a particular attribute group, include the ID of the attribute group.

For an example of this messaging template, refer to the online version of this topic.

Inherit References in STEPXML

When exporting, references within STEP are output against the object type where the references are defined. However, it is possible to inherit references to a specific product type when data is exported from STEP.

For details on tags, elements, and their XML attributes, refer to the available XSD (XML Schema Definition) files in the **STEPXML Tags and Examples** topic.

Note: Using this tag can cause an import to fail stating that each attribute is not valid for Object when attribute values are attempted on object types where they are not legal.

Inheriting references to all product types

To inherit references to all object types, use IncludeInherited.

For example:

```
<ClassificationReference IncludeInherited="true"/>
```

Inheriting references to a particular product type

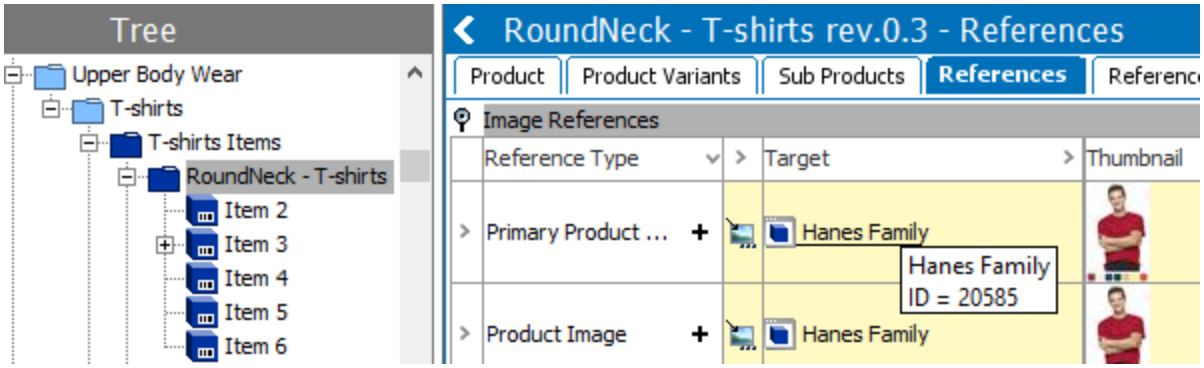
To inherit references to a particular object type, the following tag must include the product object type that references should be inherited to. If the type is not added, then all reference types will be exported.

For example:

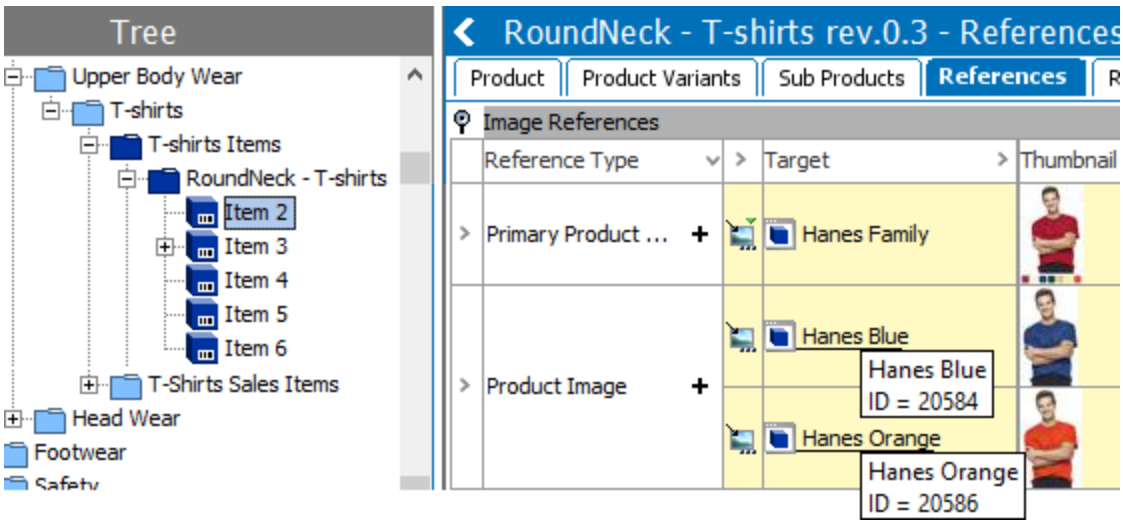
```
<AssetCrossReference Type="ProductImage" IncludeInheritedReferences="Level 3"/>
```

Data Example

The following sample data will be exported using the template below. The parent object includes the image references below:



The child object includes the following two local references (which override the inherited values from the parent):



Template

The boldface text includes two tags. One should be removed before mapping.

```
<?xml version="1.0" encoding="UTF-8"?>
<STEP-ProductInformation ExportDeletedData="true" ResolveInlineRefs="true"
ExportDerivedAttrs="true">
    <Products>
        <Product IncludeParent="true">
            <Name/>
            <Values>
                <Value AttributeID="Brand"/>

                            <Value AttributeID="ProductStatusAttributes"/>

            </Values>
            <AssetCrossReference Type="Main"/>
            <ProductCrossReference Type="Accessory"/>
            <ClassificationReference Type="Website"/>
        </Product>
    </Products>
</STEP-ProductInformation>
```

Result

The following image shows an excerpt of the export using inheritance for only the ProductImage asset reference type using the template line: <AssetCrossReference Type="ProductImage" IncludeInheritedReferences="Level 3"/>.

```
...
<Product ID="18203" UserTypeID="ItemFolder" Selected="false">
  <Name>T-shirts Items</Name>
  ...
  <Product ID="124625" UserTypeID="ItemFamily">
    <Name>RoundNeck - T-shirts</Name>
    <AssetCrossReference AssetID="20585" Type="ProductImage"/>

    ...
    <Product ID="124626" UserTypeID="Item">
      <Name>Item 2</Name>
        <AssetCrossReference AssetID="20586" Type="ProductImage"/>
        <AssetCrossReference AssetID="20584" Type="ProductImage"/>

       ...
```

The following image shows an excerpt of the export using inheritance for the all reference types using the template line: <AssetCrossReference IncludeInherited="true"/>.

```
...
<Product ID="18203" UserTypeID="ItemFolder" Selected="false">
  <Name>T-shirts Items</Name>
  ...
  <Product ID="124625" UserTypeID="ItemFamily">
    <Name>RoundNeck - T-shirts</Name>
      <AssetCrossReference AssetID="20585" Type="ProductImage"/>
      <AssetCrossReference AssetID="20585" Type="PrimaryProductImage"/>

    ...
    <Product ID="124626" UserTypeID="Item">
      <Name>Item 2</Name>
        <AssetCrossReference AssetID="20586" Type="ProductImage"/>
        <AssetCrossReference AssetID="20584" Type="ProductImage"/>
        <AssetCrossReference AssetID="6900" Type="OwnersManual"/>
        <AssetCrossReference AssetID="20700" Type="ADA"/>

       ...
```

Inherit Values in STEPXML

The Values tag is valid within the assets, classifications, and products tags and by default outputs only local values. When no local value is available for an attribute, even when an inherited value exists, no value is output. For more information, refer to the 'Inherited and Local Values' section of the **Editing Attribute Values** topic in the **System Setup** documentation.

Note: The **Only export leaf objects** checkbox on outbound tools is ignored when exporting STEPXML format. Refer to the **Export Manager - Select Objects** topic, the **OIEP - Event-Based - Output Templates Flipper** topic, or the **OIEP - Select Objects - Output Templates Flipper** topic, all in this guide.

The IncludeInherited and IncludeInheritedAttributes XML attributes allow inherited values to output to top-level objects or to specified object types.

For details on tags, elements, and their XML attributes, refer to the available XSD (XML Schema Definition) files in the **STEPXML Tags and Examples** topic.

IncludeInherited

Local values are included for all objects in the output. To avoid duplicate values throughout the file, inherited values are deliberately exported only once at the top-most level of the object type being exported.

```
<Values IncludeInherited="true"/>
```

IncludeInheritedAttributes

Inherits attribute values down to objects of the specified object type. Use multiple instances of this tag when duplicate values are required for inherited attributes. This frequently causes redundant (repetitious) data to be exported.

```
<Values IncludeInheritedAttributes="ItemFolder"/>
<Values IncludeInheritedAttributes="Item"/>
```

Note: IncludeInheritedAttributes may cause attribute values to be exported on object types where they are not valid. For this reason they cannot be imported back into STEP. If attempted, the import process fails with errors declaring that each attribute is not valid for the object.

Considerations

The following scenarios can be problematic if not implemented correctly:

<Product/>

When nested within the 'root' <Product>……</Product> structure, the <Product/> tag includes all child products of the selected product node, as shown in the 'Correct Template' in the table below. Without this tag, no children are output.

This tag cannot be used to build a secondary product structure. All product tags should be located within the 'root' <Product>……</Product> structure.

The 'Incorrect Template' below includes a second <Product>……</Product> structure within the 'root' <Product> ……</Product> structure. This invalid structure does not generate values for children objects.

Correct Template	Incorrect Template
<?xml version="1.0" encoding="UTF-8"?> <STEP-ProductInformation ResolveInlineRefs="true"> <Products ExportSize="Minimum"> <FilterUserType ID="ItemFolder"/> <FilterUserType ID="Item"/> <FilterUserType ID="SpecialItem"/> <Product> <Name/> <Values IncludeInheritedAttributes="Item"/> <Values IncludeInheritedAttributes="SpecialItem"/> <Product/> </Product> </Products> </STEP-ProductInformation>	<?xml version="1.0" encoding="UTF-8"?> <STEP-ProductInformation ResolveInlineRefs="true"> <Products ExportSize="Minimum"> <FilterUserType ID="ItemFolder"/> <FilterUserType ID="Item"/> <FilterUserType ID="SpecialItem"/> <Product> <Name/> <Product> <Values IncludeInheritedAttributes="Item"/> <Values IncludeInheritedAttributes="SpecialItem"/> </Product> </Product> </Products> </STEP-ProductInformation>

<Values>……<Values/>

The correct output is achieved by placing all required attributes within the first <Values>……</Values> construction as shown in the 'Correct Template' below.

When using multiple instances of the <Values>……</Values> tag, attribute values must be included in the first structure.

In the 'Incorrect Template' below, only the 'Country Of Origin' attribute is defined in the first <Values>……</Values> tag and so will be exported with its value. The attributes 'Brand' and 'PrimaryColor' which are specified in the second <Values>……</Values> tag are ignored.

Correct Template	Incorrect Template
<?xml version="1.0" encoding="UTF-8"?> <STEP-ProductInformation ResolveInlineRefs="true"> <Products ExportSize="Minimum"> <Product> <Name/> <Values IncludeInheritedAttributes="Item"> <Value AttributeID="CountryOfOrigin"/> <Value AttributeID="Brand"/> <Value AttributeID="PrimaryColor"/> </Values> <Values IncludeInheritedAttributes="SpecialItem"/> <Product/> </Product> </Products> </STEP-ProductInformation>	<?xml version="1.0" encoding="UTF-8"?> <STEP-ProductInformation ResolveInlineRefs="true"> <Products ExportSize="Minimum"> <Product> <Name/> <Values IncludeInheritedAttributes="Item"> <Value AttributeID="CountryOfOrigin"/> </Values> <Values IncludeInheritedAttributes="SpecialItem"> <Value AttributeID="Brand"/> <Value AttributeID="PrimaryColor"/> </Values> <Product/> </Product> </Products> </STEP-ProductInformation>

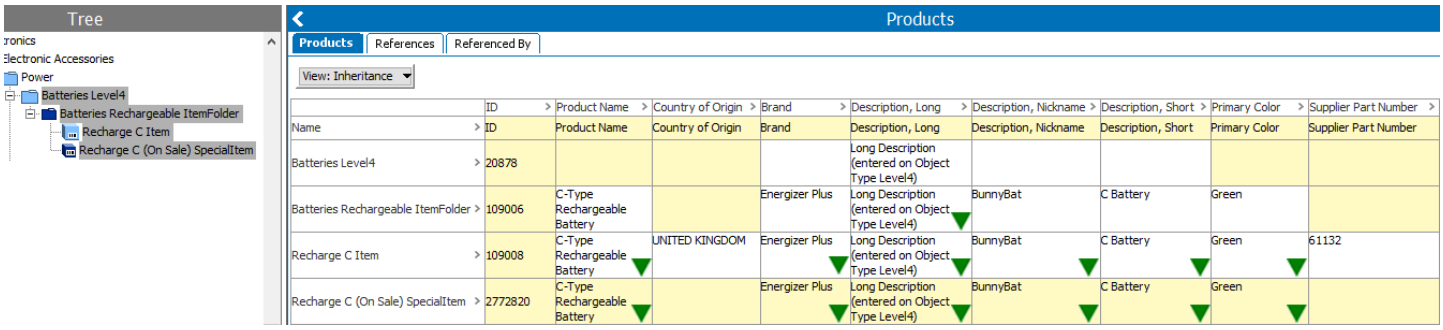
Examples

Review the examples in these linked topics to understand the differences in output of child objects, inherited values, and how to restrict output based on object type.

1. Export only the selected product, no child products - **Inherit Values: Selected Only**
2. Export the selected product and all child products regardless of level - **Inherit Values: Selected and Children**
3. Do not export the selected product, export all child products including all local values, and inherited values only by object type - **Inherit Values: Exclude Selected, All Local on Children, Inherited on Children by Object Type**
4. Export only local values for the selected product, export local and/or inherited values for child products by object type - **Inherit Values: Local on Selected, All on Children by Object Type**
5. Do not export the selected product, export local and/or inherited values for child products by object type - **Inherit Values: Exclude Selected, All on Children**
6. Export the selected product with local values, export the same attributes (but only inherited values) for the child objects - **Inherit Values: Local on Selected, Inherited on Children**

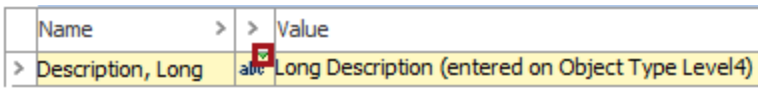
Inherit Values: Selected Only Example

The following objects and attribute values are exported in the template below. The object type has been included in the name of each product to allow for clarity in the output file. Although not displayed in the workbench product view user interface, a green triangle (▼) has been manually added to this image to identify the inherited values.



Name	ID	Product Name	Country of Origin	Brand	Description, Long	Description, Nickname	Description, Short	Primary Color	Supplier Part Number
Batteries Level4	20878				Long Description (entered on Object Type Level4)				
Batteries Rechargeable ItemFolder	109006	C-Type Rechargeable Battery		Energizer Plus	Long Description (entered on Object Type Level4) ▼	BunnyBat	C Battery	Green	
Recharge C Item	109008	C-Type Rechargeable Battery ▼	UNITED KINGDOM	Energizer Plus ▼	Long Description (entered on Object Type Level4) ▼	BunnyBat ▼	C Battery ▼	Green ▼	61132
Recharge C (On Sale) SpecialItem	2772820	C-Type Rechargeable Battery ▼		Energizer Plus ▼	Long Description (entered on Object Type Level4) ▼	BunnyBat ▼	C Battery ▼	Green ▼	

When viewing a single product in workbench, inherited values are indicated with a small green triangle as shown below.



The following template is used to export the 'Batteries Rechargeable Item Folder' product. The output includes:

- Only the selected product. No child products are exported because the <Product/> tag is not included.
- All local values.
- One inherited value (the 'Description, Long' (ID: DescriptionLong) value).

Template

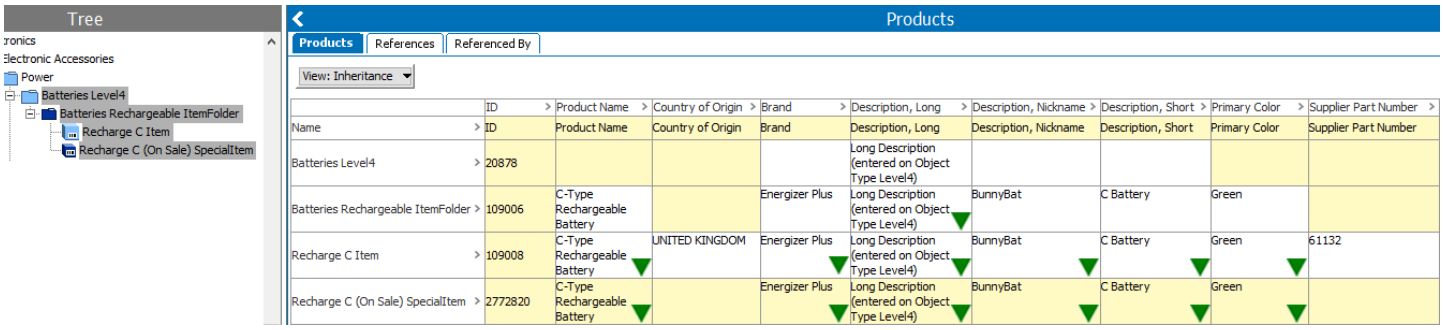
```
<?xml version="1.0" encoding="UTF-8"?>
<STEP-ProductInformation>
 <Products ExportSize="Minimum">
  <Product>
   <Name/>
   <Values IncludeInherited="true"/>
  </Product>
 </Products>
</STEP-ProductInformation>
```

Output

```
- <STEP-ProductInformation ExportTime="2021-02-10 10:53:16" ExportContext="Context1" ContextID="Context
  - <Products>
    - <Product ID="109006" UserTypeID="ItemFolder" ParentID="20878">
        <Name>Batteries Rechargeable ItemFolder</Name>
      - <Values>
          <Value AttributeID="Brand">Energizer Plus</Value>
        - <MultiValue AttributeID="ProductName">
            <Value>C-Type Rechargeable Battery</Value>
          </MultiValue>
          <Value AttributeID="DescriptionLong">Long Description (entered on Object Type Level4)</Value>
          <Value AttributeID="PrimaryColor" ID="124327">Green</Value>
          <Value AttributeID="DescriptionShort">C Battery</Value>
          <Value AttributeID="DescriptionNickname">BunnyBat</Value>
          <Value AttributeID="Print_TitleLevel1" Derived="true">Electronics</Value>
        </Values>
      </Product>
    </Products>
  </STEP-ProductInformation>
```

Inherit Values: Selected and Children Example

The following objects and attribute values are exported in the template below. The object type has been included in the name of each product to allow for clarity in the output file. Although not displayed in the workbench product view user interface, a green triangle (▼) has been manually added to this image to identify the inherited values.



Name	ID	Product Name	Country of Origin	Brand	Description, Long	Description, Nickname	Description, Short	Primary Color	Supplier Part Number
Batteries Level4	20878				Long Description (entered on Object Type Level4)				
Batteries Rechargeable ItemFolder	109006	C-Type Rechargeable Battery		Energizer Plus	Long Description (entered on Object Type Level4)	BunnyBat	C Battery	Green	
Recharge C Item	109008	C-Type Rechargeable Battery	UNITED KINGDOM	Energizer Plus	Long Description (entered on Object Type Level4)	BunnyBat	C Battery	Green	61132
Recharge C (On Sale) SpecialItem	2772820	C-Type Rechargeable Battery		Energizer Plus	Long Description (entered on Object Type Level4)	BunnyBat	C Battery	Green	

When viewing a single product in workbench, inherited values are indicated with a small green triangle as shown below.

Name	Value
Description, Long	Long Description (entered on Object Type Level4)

The following template is used to export the 'Batteries Rechargeable Item Folder' product. The output includes:

- The selected product node.
- All levels of child products ('**Item**' and '**SpecialItem**') via the <Product/> (shown in red text), which is nested within the <Product>……</Product> structure (shown in blue text).
- The inherited 'Description, Long' (ID: DescriptionLong) attribute value for the selected product only. By default, inherited values are not included for child products.

Template

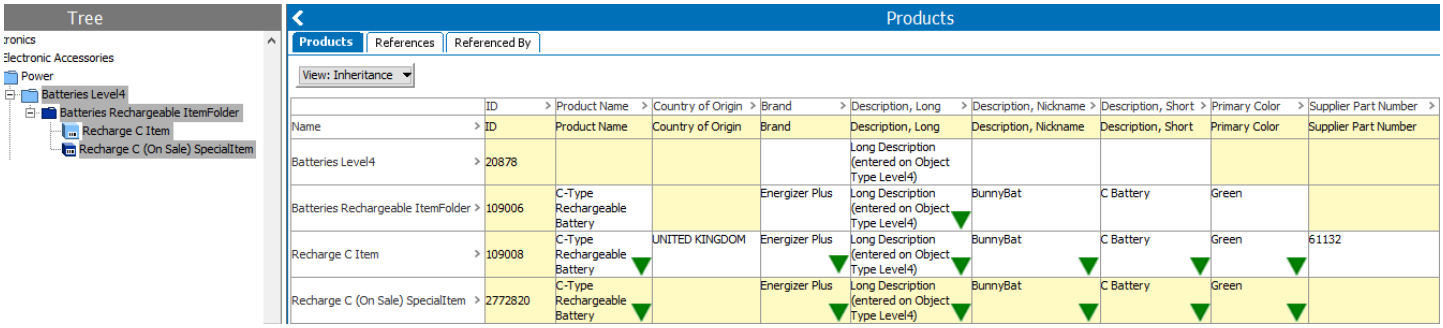
```
<?xml version="1.0" encoding="UTF-8"?>
<STEP-ProductInformation>
 <Products ExportSize="Minimum">
  <Product>
   <Name/>
   <Values IncludeInherited="true"/>
  <Product/>
  </Product>
 </Products>
</STEP-ProductInformation>
```

Output

```
- <STEP-ProductInformation ExportTime="2021-02-10 11:23:40" ExportContext="Context1" ContextID="Context1" WorkspaceID="Main" UseContextLocale="false">
  - <Products>
    - <Product ID="109006" UserTypeID="ItemFolder" ParentID="20878">
        <Name>Batteries Rechargeable ItemFolder</Name>
      - <Values>
          <Value AttributeID="Brand">Energizer Plus</Value>
        - <MultiValue AttributeID="ProductName">
            <Value>C-Type Rechargeable Battery</Value>
          </MultiValue>
          <Value AttributeID="DescriptionLong">Long Description (entered on Object Type Level4)</Value>
          <Value AttributeID="PrimaryColor" ID="124327">Green</Value>
          <Value AttributeID="DescriptionShort">C Battery</Value>
          <Value AttributeID="DescriptionNickname">BunnyBat</Value>
          <Value AttributeID="Print_TitleLevel1" Derived="true">Electronics</Value>
        </Values>
      - <Product ID="109008" UserTypeID="Item">
          <Name>Recharge C Item</Name>
        - <Values>
            <Value AttributeID="SupplierPartNumber">61132</Value>
          - <MultiValue AttributeID="CountryOfOrigin">
              <Value ID="128396">UNITED KINGDOM</Value>
            </MultiValue>
            <Value AttributeID="Path" Derived="true">Electronics | Electronic Accessories | Power | Batteries Level4 | Batteries Rechargeable ItemFolder</Value>
            <Value AttributeID="Category" Derived="true">Primary Product Hierarchy | Products | Electronics | Electronic Accessories | Power | Batteries Level4 |
              Batteries Rechargeable ItemFolder | Recharge C Item | Recharge C Item</Value>
            <Value AttributeID="Parent" Derived="true">Batteries Rechargeable ItemFolder</Value>
          </Values>
        </Product>
      - <Product ID="2772820" UserTypeID="SpecialItem">
          <Name>Recharge C (On Sale) SpecialItem</Name>
        </Product>
      </Product>
    </Products>
  </STEP-ProductInformation>
```

Inherit Values: Exclude Selected, All Local on Children, Inherited on Children by Object Type Example

The following objects and attribute values are exported in the template below. The object type has been included in the name of each product to allow for clarity in the output file. Although not displayed in the workbench product view user interface, a green triangle (▼) has been manually added to this image to identify the inherited values.



Name	ID	Product Name	Country of Origin	Brand	Description, Long	Description, Nickname	Description, Short	Primary Color	Supplier Part Number
Batteries Level4	20878				Long Description (entered on Object Type Level4)				
Batteries Rechargeable ItemFolder	109006	C-Type Rechargeable Battery		Energizer Plus	Long Description (entered on Object Type Level4)	BunnyBat	C Battery	Green	
Recharge C Item	109008	C-Type Rechargeable Battery	UNITED KINGDOM	Energizer Plus	Long Description (entered on Object Type Level4)	BunnyBat	C Battery	Green	61132
Recharge C (On Sale) SpecialItem	2772820	C-Type Rechargeable Battery		Energizer Plus	Long Description (entered on Object Type Level4)	BunnyBat	C Battery	Green	

When viewing a single product in workbench, inherited values are indicated with a small green triangle as shown below.

Name	Value
Description, Long	Long Description (entered on Object Type Level4)

The following template is used to export the 'Batteries Level4' product. The output includes:

- Only object types allowed by the <FilterUserType> tags. In this case, the selected object type '**Level4**' is not output. For more information, refer to the **Filter Products in STEPXML** topic in this guide.
- All child object types (due to the nested <Product/> tag shown in red text) but restricted to the '**ItemFolder**', '**Item**', and '**SpecialItem**' object types based on the <FilterUserType> tags.
- All local values for object type '**ItemFolder**', but not the inherited value for 'Description, Long' (ID: DescriptionLong) because the object type is not included as an 'IncludeInheritedAttributes' tag.
- All values for object type '**Item**' local values, plus any inherited attribute values from any product node above it in the product structure.
- No local values for object type '**SpecialItem**' because there are none, but all inherited attribute values from any product node above it in the product structure.

Template

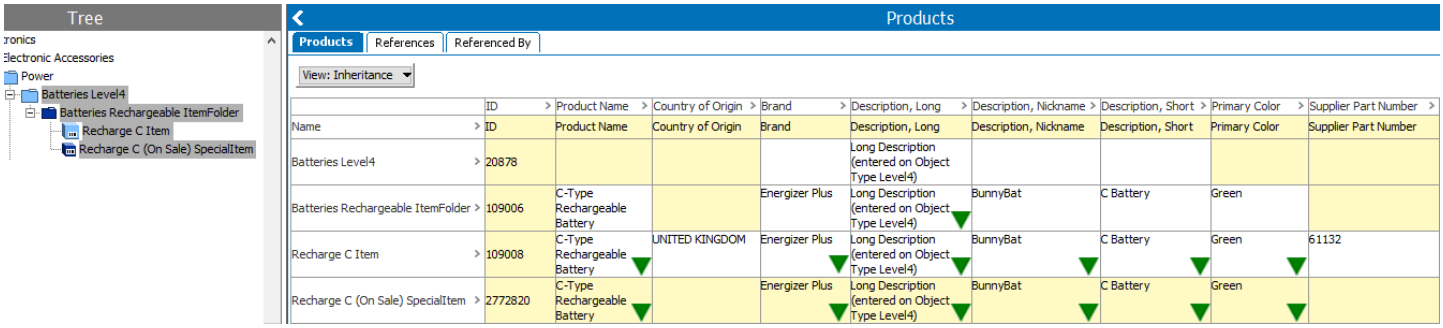
```
<?xml version="1.0" encoding="UTF-8"?>
<STEP-ProductInformation ResolveInlineRefs="true">
 <Products ExportSize="Minimum">
    <FilterUserType ID="ItemFolder"/>
    <FilterUserType ID="Item"/>
    <FilterUserType ID="SpecialItem"/>
  <Product>
   <Name/>
   <Values IncludeInheritedAttributes="Item"/>
   <Values IncludeInheritedAttributes="SpecialItem"/>
  <Product/>
  </Product>
 </Products>
</STEP-ProductInformation>
```

Output

```
- <STEP-ProductInformation ExportTime="2021-02-10 12:02:17" ExportContext="Context1" ContextID="Context1" WorkspaceID="Main" UseContextLocale="false">
  - <Products>
    - <Product ID="109006" UserTypeID="ItemFolder" ParentID="20878">
        <Name>Batteries Rechargeable ItemFolder</Name>
      - <Values>
          <Value AttributeID="Brand">Energizer Plus</Value>
        + <MultiValue AttributeID="ProductName">
          <Value AttributeID="PrimaryColor" ID="124327">Green</Value>
          <Value AttributeID="DescriptionShort">C Battery</Value>
          <Value AttributeID="DescriptionNickname">BunnyBat</Value>
          <Value AttributeID="Print_TitleLevel1" Derived="true">Electronics</Value>
        </Values>
      - <Product ID="109008" UserTypeID="Item">
          <Name>Recharge C Item</Name>
        - <Values>
            <Value AttributeID="Brand">Energizer Plus</Value>
          - <MultiValue AttributeID="CountryOfOrigin">
              <Value ID="128396">UNITED KINGDOM</Value>
            </MultiValue>
            <Value AttributeID="DescriptionNickname">BunnyBat</Value>
            <Value AttributeID="DescriptionShort">C Battery</Value>
          - <MultiValue AttributeID="ProductName">
              <Value>C-Type Rechargeable Battery</Value>
            </MultiValue>
            <Value AttributeID="DescriptionLong">Long Description (entered on Object Type Level4)</Value>
            <Value AttributeID="PrimaryColor" ID="124327">Green</Value>
            <Value AttributeID="SupplierPartNumber">61132</Value>
            <Value AttributeID="Path" Derived="true">Electronics | Electronic Accessories | Power | Batteries Level4 | Batteries Rechargeable ItemFolder</Value>
            <Value AttributeID="Category" Derived="true">Primary Product Hierarchy | Products | Electronics | Electronic Accessories | Power | Batteries Level4 |
              Batteries Rechargeable ItemFolder | Recharge C Item | Recharge C Item</Value>
            <Value AttributeID="Parent" Derived="true">Batteries Rechargeable ItemFolder</Value>
          </Values>
        </Product>
      - <Product ID="2772820" UserTypeID="SpecialItem">
          <Name>Recharge C (On Sale) SpecialItem</Name>
        - <Values>
            <Value AttributeID="Brand">Energizer Plus</Value>
          - <MultiValue AttributeID="ProductName">
              <Value>C-Type Rechargeable Battery</Value>
            </MultiValue>
            <Value AttributeID="DescriptionLong">Long Description (entered on Object Type Level4)</Value>
            <Value AttributeID="PrimaryColor" ID="124327">Green</Value>
            <Value AttributeID="DescriptionNickname">BunnyBat</Value>
            <Value AttributeID="DescriptionShort">C Battery</Value>
          </Values>
        </Product>
      </Product>
    </Products>
  </STEP-ProductInformation>
```

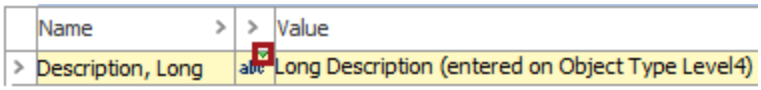
Inherit Values: Local on Selected, All on Children by Object Type Example

The following objects and attribute values are exported in the template below. The object type has been included in the name of each product to allow for clarity in the output file. Although not displayed in the workbench product view user interface, a green triangle (▼) has been manually added to this image to identify the inherited values.



Name	ID	Product Name	Country of Origin	Brand	Description, Long	Description, Nickname	Description, Short	Primary Color	Supplier Part Number
Batteries Level4	20878				Long Description (entered on Object Type Level4)				
Batteries Rechargeable ItemFolder	109006	C-Type Rechargeable Battery		Energizer Plus	Long Description (entered on Object Type Level4)	BunnyBat	C Battery	Green	
Recharge C Item	109008	C-Type Rechargeable Battery	UNITED KINGDOM	Energizer Plus	Long Description (entered on Object Type Level4)	BunnyBat	C Battery	Green	61132
Recharge C (On Sale) SpecialItem	2772820	C-Type Rechargeable Battery		Energizer Plus	Long Description (entered on Object Type Level4)	BunnyBat	C Battery	Green	

When viewing a single product in workbench, inherited values are indicated with a small green triangle as shown below.



Name	Value
Description, Long	Long Description (entered on Object Type Level4)

The following template is used to export the 'Batteries Rechargeable ItemsFolder' product. The output includes:

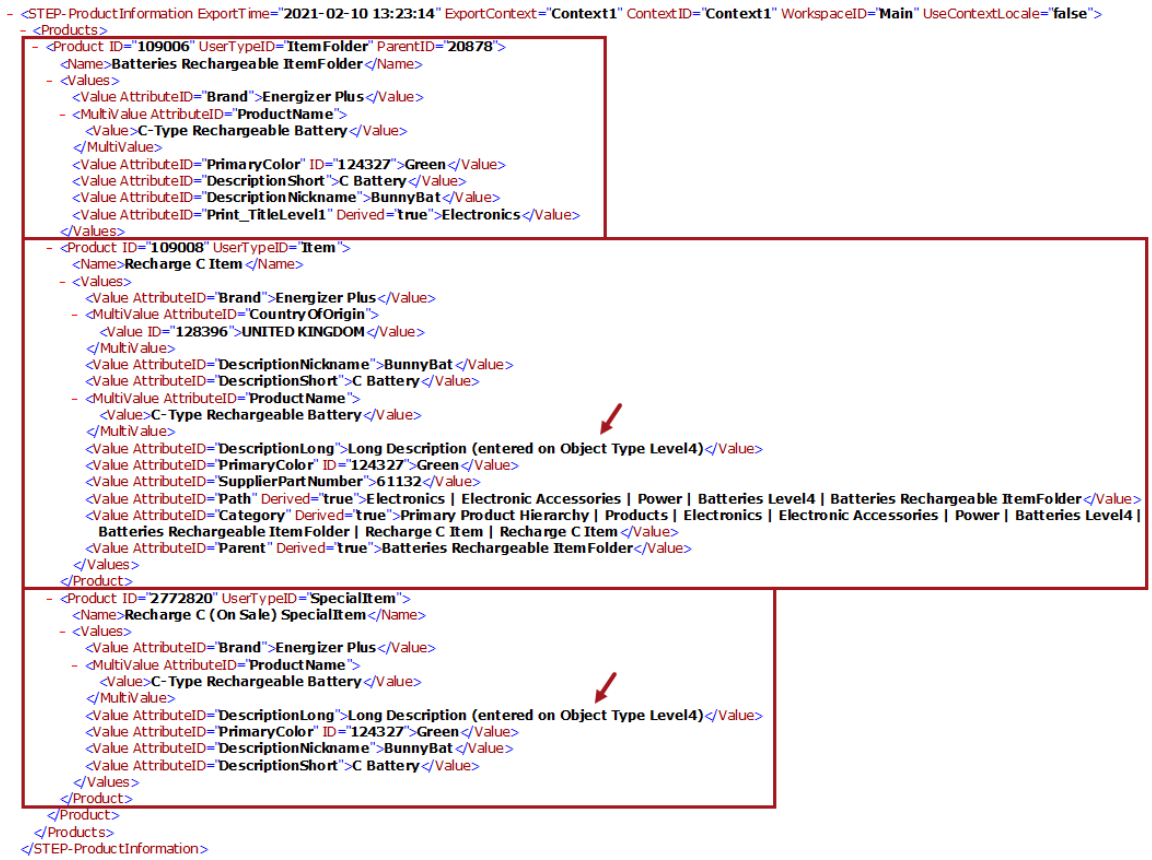
- All local values for '**ItemsFolder**' object type but no inherited values because the object type is not included as an <IncludeInheritedAttributes> tag.
- All child object types (due to the nested <Product/> tag shown in red text) but restricted to the '**ItemFolder**', '**Item**', and '**SpecialItem**' object types based on the <FilterUserType> tags.
- All values for object type '**Item**' local values, plus any inherited attribute values from any product node above it in the product structure.
- No local values for object type '**SpecialItem**' because there are none.
- All inherited attribute values for object type '**SpecialItem**' from any product node above it in the product structure (the inherited value for 'Description, Long' (ID: DescriptionLong)).

Template

```
<?xml version="1.0" encoding="UTF-8"?>
<STEP-ProductInformation ResolveInlineRefs="true">
 <Products ExportSize="Minimum">
    <FilterUserType ID="ItemFolder"/>
    <FilterUserType ID="Item"/>
```

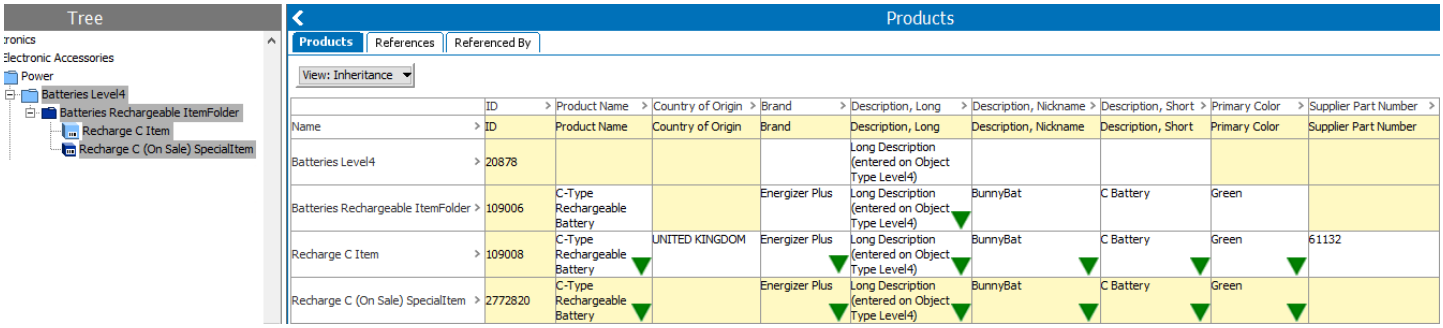
```
    <FilterUserType ID="SpecialItem"/>
  <Product>
   <Name/>
   <Values IncludeInheritedAttributes="Item"/>
   <Values IncludeInheritedAttributes="SpecialItem"/>
  <Product/>
  </Product>
 </Products>
</STEP-ProductInformation>
```

Output

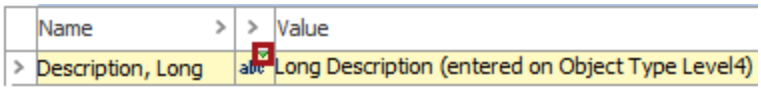


Inherit Values: Exclude Selected, All on Children Example

The following objects and attribute values are exported in the template below. The object type has been included in the name of each product to allow for clarity in the output file. Although not displayed in the workbench product view user interface, a green triangle (▼) has been manually added to this image to identify the inherited values.



When viewing a single product in workbench, inherited values are indicated with a small green triangle as shown below.



The following template is used to export the 'Batteries Level4' product. The output includes:

- Only object types allowed by the <FilterUserType> tags. The selected object type (Level4) is not output because '**Level4**' does not have a <FilterUserType> tag.
- One local value for the '**ItemFolder**' object type on 'Primary Color' (ID = PrimaryColor), but no inherited values since the '**ItemFolder**' object type is not a 'IncludeInheritedAttributes' tag.
- No local values for the '**Item**' or '**SpecialItem**' object types, but the inherited attribute value for 'Primary Color' (ID=PrimaryColor) for both of these object types.

Template

```
<?xml version="1.0" encoding="UTF-8"?>
<STEP-ProductInformation ResolveInlineRefs="true">
<Products ExportSize="Minimum">
   <FilterUserType ID="ItemFolder"/>
    <FilterUserType ID="Item"/>
    <FilterUserType ID="SpecialItem"/>
  <Product>
   <Name/>
   <Values IncludeInheritedAttributes="Item">
```

```
   <Value AttributeID="PrimaryColor"/>
   </Values>
   <Values IncludeInheritedAttributes="SpecialItem"/>
      <Product/>
  </Product>
 </Products>
</STEP-ProductInformation>
```

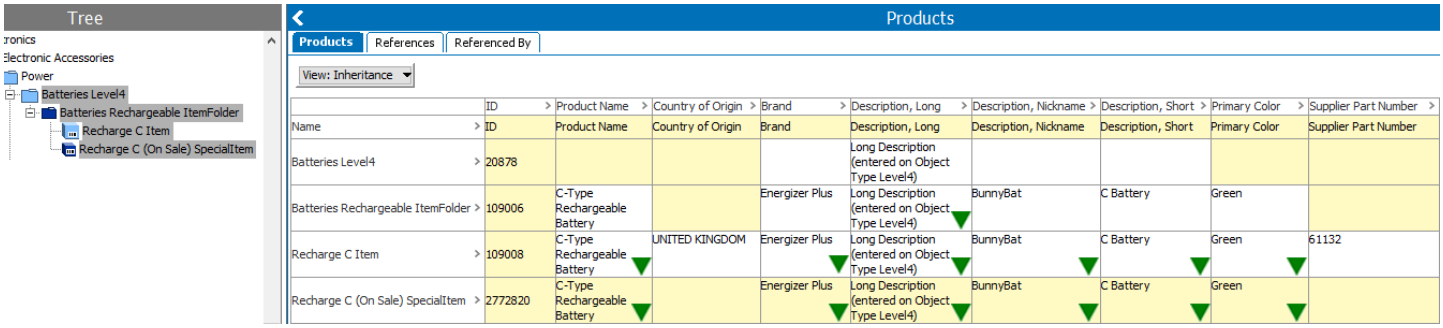
Output

```
- <STEP-ProductInformation ExportTime="2021-02-10 13:55:59" ExportContext="Cor
  - <Products>
    - <Product ID="109006" UserTypeID="ItemFolder" ParentID="20878">
        <Name>Batteries Rechargeable ItemFolder</Name>
      - <Values>
          <Value AttributeID="PrimaryColor" ID="124327">Green</Value>
        </Values>
      - <Product ID="109008" UserTypeID="Item">
          <Name>Recharge C Item</Name>
        - <Values>
            <Value AttributeID="PrimaryColor" ID="124327">Green</Value>
          </Values>
        </Product>
      - <Product ID="2772820" UserTypeID="SpecialItem">
          <Name>Recharge C (On Sale) SpecialItem</Name>
        - <Values>
            <Value AttributeID="PrimaryColor" ID="124327">Green</Value>
          </Values>
        </Product>
      </Product>
    </Products>
  </STEP-ProductInformation>
```

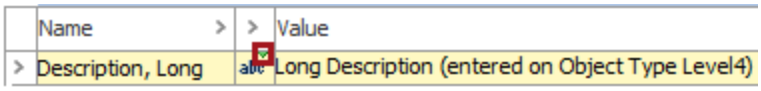
Inherit Values: Local on Selected, Inherited on Children Example

This example exports the selected product with local values, and then exports only inherited values for the same attributes for the child objects.

The following objects and attribute values are exported in the template below. The object type has been included in the name of each product to allow for clarity in the output file. Although not displayed in the workbench product view user interface, a green triangle (▼) has been manually added to this image to identify the inherited values.



When viewing a single product in workbench, inherited values are indicated with a small green triangle as shown below.



The following template is used to export the 'Batteries Rechargeable ItemsFolder' product. The output includes:

- Only two local attribute values for the '**ItemFolder**' object type ('Brand' (ID: Brand) and 'Primary Color' (ID: PrimaryColor))
- The same inherited attribute values from the '**ItemFolder**' for the '**Item**' and '**SpecialItem**' object types because of the 'IncludeInheritedAttributes' tag.
- No local values for '**Item**' or '**SpecialItem**' object types.

Template

```
<?xml version="1.0" encoding="UTF-8"?>
<STEP-ProductInformation ResolveInlineRefs="true">
 <Products ExportSize="Minimum">
  <Product>
   <Name/>
   <Values IncludeInheritedAttributes="ItemFolder">
     <Value AttributeID="Brand"/>
```

```
      <Value AttributeID="PrimaryColor"/>
    </Values>
    <Values IncludeInheritedAttributes="Item"/>
     <Values IncludeInheritedAttributes="SpecialItem"/>
     <Product/>
   </Product>
 </Products>
</STEP-ProductInformation>
```

Output

```
- <STEP-ProductInformation ExportTime="2021-02-10 14:06:11" ExportContext="Co
  - <Products>
    - <Product ID="109006" UserTypeID="ItemFolder" ParentID="20878">
        <Name>Batteries Rechargeable ItemFolder</Name>
      - <Values>
          <Value AttributeID="Brand">Energizer Plus</Value>
          <Value AttributeID="PrimaryColor" ID="124327">Green</Value>
        </Values>
      - <Product ID="109008" UserTypeID="Item">
          <Name>Recharge C Item</Name>
        - <Values>
            <Value AttributeID="Brand">Energizer Plus</Value>
            <Value AttributeID="PrimaryColor" ID="124327">Green</Value>
          </Values>
        </Product>
      - <Product ID="2772820" UserTypeID="SpecialItem">
          <Name>Recharge C (On Sale) SpecialItem</Name>
        - <Values>
            <Value AttributeID="Brand">Energizer Plus</Value>
            <Value AttributeID="PrimaryColor" ID="124327">Green</Value>
          </Values>
        </Product>
      </Product>
    </Products>
  </STEP-ProductInformation>
```

ImageConversionConfiguration Tag in STEPXML

Image Conversion configurations can be imported / exported via the STEPXML and Advanced STEPXML formats using the following tags:

- **<ImageConversionConfigurations>**

  The <ImageConversionConfigurations> tag is used to contain all the Image Conversion configurations included within the STEPXML / Advanced STEPXML file.

- **<ImageConversionConfiguration>**

  The <ImageConversionConfiguration> tag is used to contain a single Image Conversion configuration included within the STEPXML / Advanced STEPXML file.

Note: While Image Conversion configurations can be exported using the STEPXML and Advanced STEPXML formats, users would use the STEPXML format if importing an Image Conversion configuration that was originally created using the Advanced STEPXML format. For more information, refer to the **Advanced STEPXML Format** topic in the **Data Exchange** documentation.

For an example of the <ImageConversionConfiguration> tag used within STEPXML, access the online version of this topic.

For more information on Image Conversion configurations, refer to the **Image Conversion Configuration** topic in the **Digital Assets** documentation. For details on tags, elements, and their attributes, refer to the available XSD (XML Schema Definition) files in the **STEPXML Tags and Examples** topic.

IncludeKey Tag in STEPXML

A unique key is a database item that represents a specific product in the STEP system. The item is composed of (transformed) product data. A key is a definition of how unique keys are composed. It specifies which object data is used and how it should be combined and transformed. The combination and transformation is done using the function framework (key formula). There can be exactly one unique key for each object for each defined key. No two objects can have the same unique key for an active key.

For more information, refer to the **Unique Keys** section of the **System Setup** documentation.

For details on tags, elements, and their XML attributes, refer to the available XSD (XML Schema Definition) files in the **STEPXML Tags and Examples** topic.

Unique keys are especially used when STEPXML is delivered to external systems like ERP systems, websites, and so on. These systems do not have access to the STEP ID of the exported products and therefore need a specific unique ID. For example, a SAP ID must be a 7-digit number.

In the following example, an advanced STEPXML template is defined to export the key ID based on the unique key 'SAP_ID.'

To export a key (with ID SAP_ID) through the Export Manager or the Event Messenger, you must include the XML tag <IncludeKey KeyID="SAP_ ID"/> after the XML tag <STEP-ProductInformation ….>.

Template

```
<STEP-ProductInformation ExportDerivedAttrs="false"
FollowOverrideSubProducts="true">

            <IncludeKey KeyID="SAP_ID"/>
<Products ExportSize="Minimum">
<Product>
   <Name/>
   <AttributeLink/>
   <ClassificationReference/>
   <ProductCrossReference/>
   <AssetCrossReference/>
   <Values/>
</Product>
</Products>
</STEP-ProductInformation>
```

Result

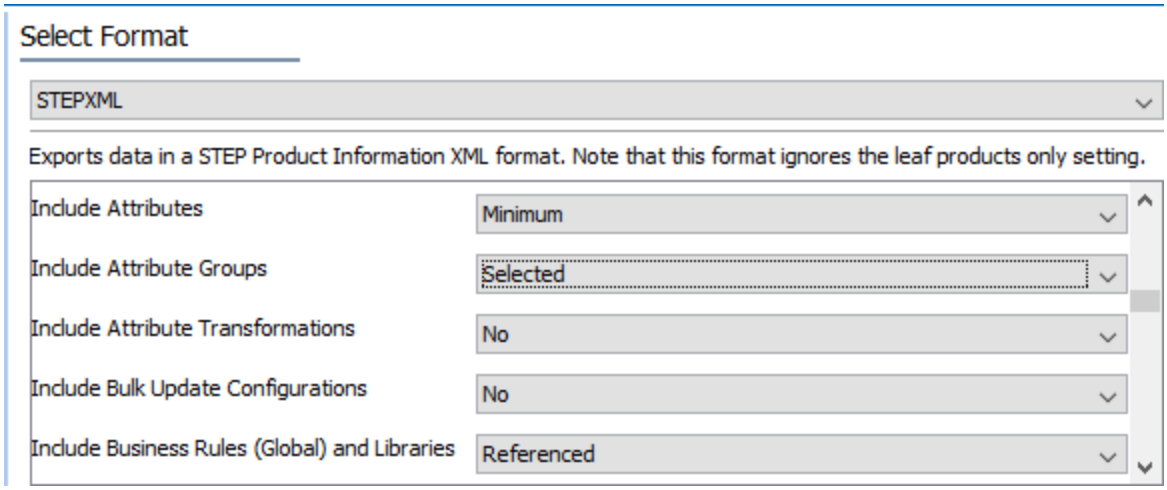
The template above instructs the exporter to also export keys of the corresponding IDs:

```
<STEP-ProductInformation
   ...
   <DeleteProduct ID="77">
```

```
      <KeyValue KeyID="SAP_ID">SAP_77</KeyValue>
      ...
   <Product ID="77" ...>
      <KeyValue KeyID="SAP_ID">SAP_77</KeyValue>
      ...
      <ProductCrossReference ProductID="77" ...>
         <KeyValue KeyID="SAP_ID">SAP_77</KeyValue>
         ...
   <Entity ID="77" ...>
      <KeyValue KeyID="SAP_ID">SAP_77</KeyValue>
      ...
   ...
```

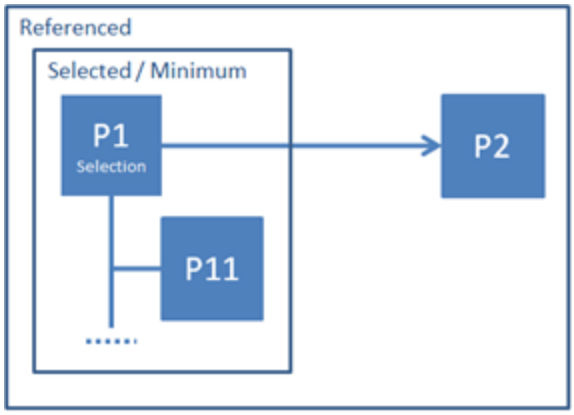
Minimum, Referenced, and Selected in STEPXML

Many of the parameters for STEPXML format on the Select Format step of the wizard include the options Minimum, Referenced, and/or Selected. The following illustrates the difference in the output of these options.



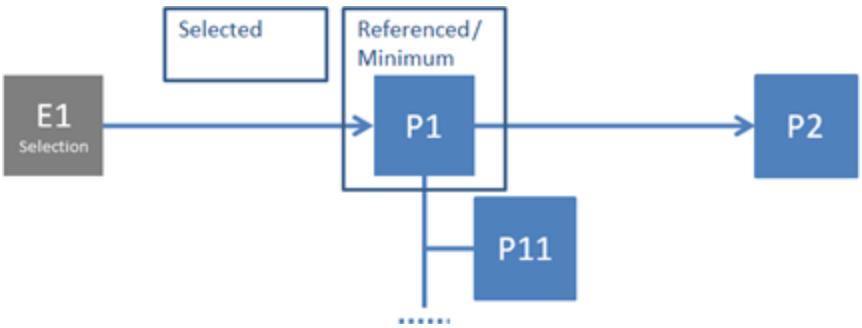
For details on tags, elements, and their XML attributes, refer to the available XSD (XML Schema Definition) files in the **STEPXML Tags and Examples** topic.

In the example below, product P1 has been selected as root node for the export. The Selected and Minimum options cause P1 and its descendants to be exported. The Referenced setting will also include P2 since it is referenced from P1.

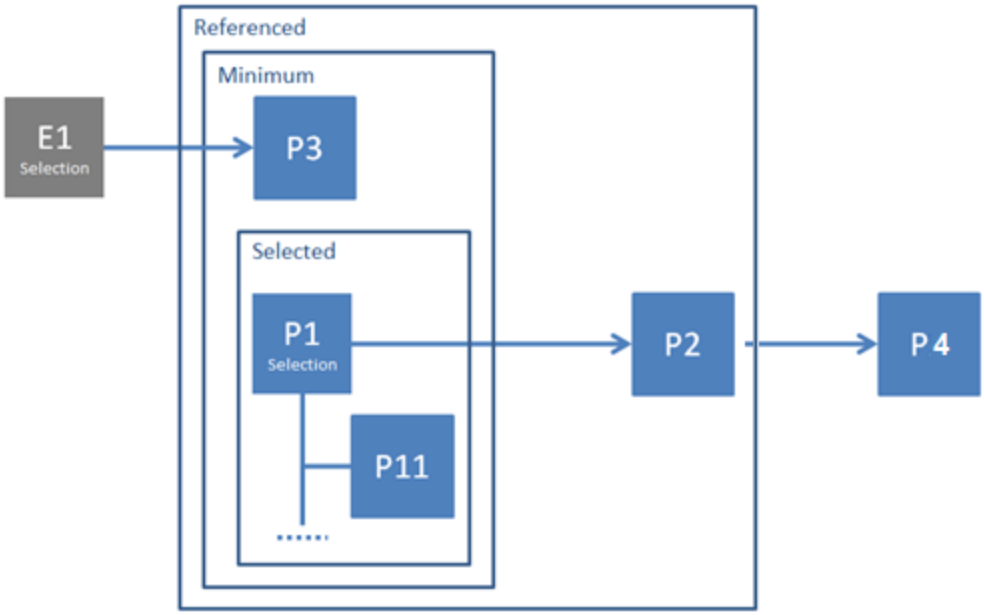


When exporting a product hierarchy, selecting Minimum outputs only the objects used in the selected product hierarchy.

In the following example, the root node selected for export is a non-product (E1) that references a product (P1). The Selected setting (for 'Include Products') will not cause any products to be exported. The Referenced and Minimum will instead have the same meaning, and both cause P1 to be exported, as shown below.



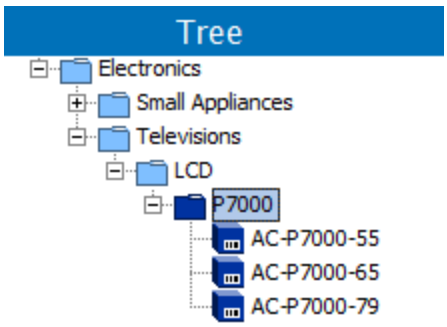
In the following example, the reason for having three options is demonstrated. Both a non-product (E1) and a product (P1) are selected as root nodes for the export. With this selection, Referenced, Minimum, and Selected will have three different meanings.



Referenced means the object and its referenced objects (P2). Referenced does not include all additional references beyond the initial level, so (P4) is not included.

Hierarchy Export Including Objects Referenced Example

The following example exports the hierarchy shown below. In addition to data for these objects, the export also includes the product, classification, and asset objects referenced from these objects.



Product P7000 is selected as the root and the export uses the following settings:

- **Include Assets: Minimum** – The minimum set of asset objects related to the initial selection (the referenced assets).
- **Include Classifications: Minimum** – The minimum set of classification objects related to the initial selection (the linked classifications and, as a special feature for classifications, all ancestors up to the top level classification node).
- **Include Products: Referenced** – The selected product, its descendants, and referenced products.
- **Include Product Attribute Values: All** – Local values (inherited values are not exported by default).

By default, products and classifications will be exported in two nested structures. To get a flat structure (no product elements inside product elements and no classification elements inside classification elements), set the Data Objects > **Flatten Hierarchies** parameter to yes.

The image below shows the exported XML, stripped of values, names, and references. Notice the following elements of the export file:

1. Linked classifications
2. Referenced assets
3. Referenced products
4. Export selection

```
<?xml version="1.0" encoding="utf-8"?>
<STEP-ProductInformation ContextID="Context1" WorkspaceID="Main" UseContextLocale="false">
  <Classifications>
    <Classification ID="Classification 1 root" UserTypeID="Classification 1 user-type root" Selected="false">
      <Classification ID="Websites" UserTypeID="WebHierarchyRoot" Selected="false">
        <Classification ID="I-WebsiteRoot" UserTypeID="WebsiteRoot" Selected="false">
          <Classification ID="I-WebLevel1-2" UserTypeID="WebLevel1" Selected="false">
1           <Classification ID="I-WebLevel2-21" UserTypeID="WebLevel2" Selected="false" Referenced="true"/>
            <Classification ID="I-WebLevel2-22" UserTypeID="WebLevel2" Selected="false" Referenced="true"/>
            <Classification ID="I-WebLevel2-24" UserTypeID="WebLevel2" Selected="false" Referenced="true"/>
          </Classification>
        </Classification>
      </Classification>
    </Classification>
  </Classifications>
  <Assets>
2   <Asset ID="ProductImage-5" UserTypeID="ProductImage" Selected="false" Referenced="true"/>
    <Asset ID="Manual-7" UserTypeID="Manual" Selected="false" Referenced="true"/>
    <Asset ID="Manual-8" UserTypeID="Manual" Selected="false" Referenced="true"/>
    <Asset ID="ProductImage-6" UserTypeID="ProductImage" Selected="false" Referenced="true"/>
  </Assets>
  <Products>
3   <Product ID="I-SalesItem-1111" UserTypeID="SalesItem" ParentID="I-Level3-111" Selected="false" Referenced="true"/>
    <Product ID="I-SalesItem-1121" UserTypeID="SalesItem" ParentID="I-Level3-112" Selected="false" Referenced="true"/>
    <Product ID="I-SalesItemFamily-1311" UserTypeID="SalesItemFamily" ParentID="I-Level3-131">
      <Name>P7000</Name>
      <ProductCrossReference ProductID="I-SalesItem-1111" Type="Accessory"/>
      <ProductCrossReference ProductID="I-SalesItem-1121" Type="Accessory"/>
      <AssetCrossReference AssetID="Manual-8" Type="ManualEN"/>
      <AssetCrossReference AssetID="Manual-7" Type="ManualDE"/>
4     <AssetCrossReference AssetID="ProductImage-6" Type="SituationImage"/>
      <AssetCrossReference AssetID="ProductImage-5" Type="PrimaryProductImage"/>
      <Product ID="I-SalesItem-13113" UserTypeID="SalesItem"/>
      <Product ID="I-SalesItem-13112" UserTypeID="SalesItem"/>
      <Product ID="I-SalesItem-13111" UserTypeID="SalesItem"/>
    </Product>
  </Products>
</STEP-ProductInformation>
```

Product Export Including Definitions, LOVs, and Units for Attributes Example

As another example, to export a single product from the same structure above, and also export the definitions for all attributes used by the product and all LOVs, and units used by these attributes.

A single product is selected as the root, and the export uses the following settings:

- **Configuration > Include Units: Minimum** – The Units used by the Attributes being exported.
- **Configuration > Include List of Values: Minimum** – The LOVs used by the exported Attributes.
- **Configuration > Include Attributes: Minimum** – The set of Attributes used by the objects being exported.
- **Data Objects > Include Products: Selected** – Only the selected product.
- **Data Objects > Include Product Attribute Values: Yes**

Below is a modified version of the results. All but one unit, one list of values group, one list of values and one attribute has been removed. Also, the exported product has again been stripped of name, references / links, and values.

```
<?xml version="1.0" encoding="utf-8"?>
<STEP-ProductInformation ContextID="Context1" WorkspaceID="Main"
UseContextLocale="false">
<UnitList>
  <Unit ID="unece.unit.CMT" Referenced="true">
  <Name>cm</Name>
  <UnitConversion BaseUnitID="unece.unit.MTR" Factor="0.01" Offset="0"/>
  </Unit>
</UnitList>
<ListsOfValues>
  <ListOfValue ID="ScreenSizes" ParentID="List Of Values group root"
   AllowUserValueAddition="false" UseValueID="true" Referenced="true">
    <Name>ScreenSizes</Name>
    <Validation BaseType="text" MinValue="" MaxValue="" MaxLength="" InputMask=""/>
    <Value ID="55">55'' (139 cm)</Value>
    <Value ID="65">65'' (164 cm)</Value>
    <Value ID="79">79'' (200 cm)</Value>
  </ListOfValue>
</ListsOfValues>
<AttributeList>
  <Attribute ID="ScreenSize" MultiValued="false" ProductMode="Normal"
  FullTextIndexed="false" ExternallyMaintained="true" Derived="false"
  HierarchicalFiltering="false" ClassificationHierarchicalFiltering="false"
  Referenced="true">
    <Name>Screen Size</Name>
    <ListOfValueLink ListOfValueID="ScreenSizes"/>
    <AttributeGroupLink AttributeGroupID="CategorySpecificAttributes"/>
    <UserTypeLink UserTypeID="SalesItem"/>
  </Attribute>
</AttributeList>
<Products>
  <Product ID="I-SalesItem-13111" UserTypeID="SalesItem" ParentID="I-
SalesItemFamily-1311"/>
</Products>
</STEP-ProductInformation>
```

Products Tag in STEPXML

The Products tag can include multiple options. When an option is absent, the default setting is used.

```
<Products ExportSize="Minimum">
  <Product IncludeParent="true">
  </Product>
</Products>
```

For details on tags, elements, and their XML attributes, refer to the available XSD (XML Schema Definition) files in the **STEPXML Tags and Examples** topic.

The following XML attributes are available:

ExportSize

For details about export size, refer to the **Minimum, Referenced, and Selected in STEPXML** section.

IncludeParent

- False - default setting
- True - when a product is exported from STEP, the file includes the approved product as well as its parents.

The Product tag can include multiple additional tags, each is described below the following sample STEPXML. When an option is absent, the data is excluded.

```
<?xml version="1.0" encoding="UTF-8"?>
<STEP-ProductInformation>
 <Products>
  <Product>
  <Name/>
  <Values IncludeInherited="true"/>
  </Product>
 </Products>
</STEP-ProductInformation>
```

Additional STEPXML Tags

The Product tag can include multiple additional internal tags, each is described below the following sample STEPXML. When an option is absent, the default setting is used.

Name

The <Name/> tag returns the object name as output.

Values

The <Values/> tag returns the attribute values as output. For more information about values output, refer to the **Inherit Values in STEPXML** section, the **Delete Values in STEPXML** and the **Filter Values in STEPXML** section.

AssetCrossReference

The <AssetCrossReference/> tag returns all asset cross references as output. For more information about cross reference output, refer to the **Filter References in STEPXML** section.

ProductCrossReference

The <ProductCrossReference/> tag returns all product cross references as output. For more information about cross reference output, refer to the **Filter References in STEPXML** section.

ClassificationReference

The <ClassificationReference/> tag returns all classification cross references as output. For more information about cross reference output, refer to the **Filter References in STEPXML** section.

Additionally, refer to the following topics for more information on modifying product data:

- Auto Approve in STEPXML
- Filter Products in STEPXML
- Referenced and Embedded XML Attributes in STEPXML
- SequenceProduct Tag in STEPXML

Product Nesting Example

In the following STEPXML file, product data reflects the Tree hierarchy structure. SalesItem-13112 and SalesItem-13113 are children of SalesItemFamily-1311. It can be represented in both a nested and non-nested structure.

Nested structure

```
<?xml version="1.0" encoding="utf-8"?>
<STEP-ProductInformation ContextID="GL" WorkspaceID="Main">
  <Products>
    <Product ID="SalesItemFamily-1311" UserTypeID="SalesItemFamily"
ParentID="Level3-131">
    <Product ID="SalesItem-13112" UserTypeID="SalesItem"/>
    <Product ID="SalesItem-13113" UserTypeID="SalesItem"/>
    </Product>
```

```
  </Products>
</STEP-ProductInformation>
```

Non-nested structure

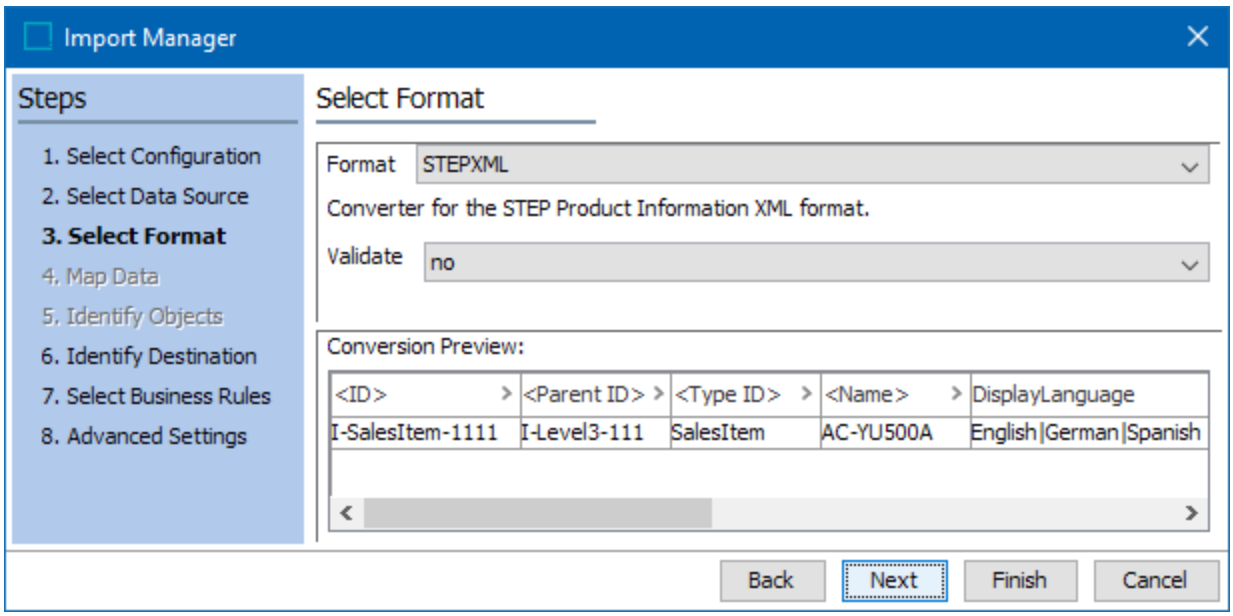
```
<?xml version="1.0" encoding="utf-8"?>
<STEP-ProductInformation ContextID="GL" WorkspaceID="Main">
  <Products>
    <Product ID="SalesItemFamily-1311" UserTypeID="SalesItemFamily"
ParentID="Level3-131"/>
    <Product ID="SalesItem-13112" UserTypeID="SalesItem" ParentID="SalesItemFamily-
1311"/>
    <Product ID="SalesItem-13113" UserTypeID="SalesItem" ParentID="SalesItemFamily-
1311"/>
  </Products>
</STEP-ProductInformation>
```

Product with Units, References, and Multi-Values Attributes Example

The following STEPXML file being imported shows a single product of the object type SalesItem (UserTypeID in the XML). The XML mirrors the STEP model so that all data owned by the product is represented in the product element.

Notice that units, references, and multi-valued attributes are represented. Also, the data in the product element does not include information about attribute, unit, and reference target names.

```
<?xml version="1.0" encoding="utf-8"?>
<STEP-ProductInformation ContextID="Context1" WorkspaceID="Main">
  <Products>
    <Product ID="I-SalesItem-1111" UserTypeID="SalesItem" ParentID="I-Level3-111">
    <Name>AC-YU500A</Name>
    <ClassificationReference ClassificationID="I-WebLevel2-23" Type="WebsiteLink"/>
    <AssetCrossReference AssetID="ProductImage-2" Type="PrimaryProductImage"/>
    <AssetCrossReference AssetID="Manual-6" Type="ManualEN"/>
    <Values>
      <MultiValue AttributeID="DisplayLanguage">
        <Value>English</Value>
        <Value>German</Value>
        <Value>Spanish</Value>
      </MultiValue>
      <Value AttributeID="LongAdvertisingCopy">Active 3D Glasses</Value>
      <Value AttributeID="Color" ID="BK">Black</Value>
      <Value AttributeID="Weight" UnitID="unece.unit.KGM">0.036</Value>
      <Value AttributeID="EAN">4905524977623</Value>
    </Values>
    </Product>
  </Products>
</STEP-ProductInformation>
```



Product to Classification Links Owned by the Classification

By default the product to classification links owned by the classification are exported in two places:

1. Under the Product as a reference to the classification.
2. Under the Classification as a reference to the product. For an XML example of the classification links, refer to the online version of this topic.

The reason the links are exported under the products is historical, and it has not been changed due to backwards compatibility. However, logically they should be exported under the classification, as they are owned by the classification. And when the links are modified, they affect the revision of the classification - not the product. It is possible to stop the classification owned links from being exported under the products by using the recorder option 'IncludeClassificationOwned' on the 'Products' element. By setting this to "false" the classification links will not be exported under the products:

```
    <STEP-ProductInformation>
        <Products IncludeClassificationOwned="false">
            <Product>
            </Product>
        </Products>
    </STEP-ProductInformation>
```

For information on IncludeClassificationOwned, refer to the **Owns Product Links on Alternate Classifications Object Type** topic in the **System Setup** documentation.

ReplacementRules Tag in STEPXML

The STEPXML import properties generally do not impact data that is missing in the XML file. The ReplacementRules tag provides special processing instructions to express that properties that are not present in the import file should be removed from the system as part of the import.

Replacement Rules allow the super type data in the import file to determine the 'end state' of the STEP data after the import (not for all objects, but only those objects in the file), rather than adding the import file data to the existing data.

Replacement is especially relevant for 'list properties' — those where multiple instances of the same XML element are present at the same level. For example, 'Value' elements within the 'Values' element for a product or 'TargetUserTypeLink' elements for a reference type definition. For list properties, use ReplacementRules to remove existing data by including only the data that should remain.

In the import file, for each super type included in the ReplacementRules tag, the data specified in the individual super type sections outside of the ReplacementRules tag defines the end state of the data. This means that after a successful import, only the data specified in the import file will exist for the elements included as ReplacementRules. Parameters on the ReplacementRules restrict the impact of the process.

For details on tags, elements, and their XML attributes, refer to the available XSD (XML Schema Definition) files in the **STEPXML Tags and Examples** topic.

Recommended Process

When using the ReplacementRules tag, follow these steps and test against a few objects until the results are successful. Then, run the rules against the whole dataset.

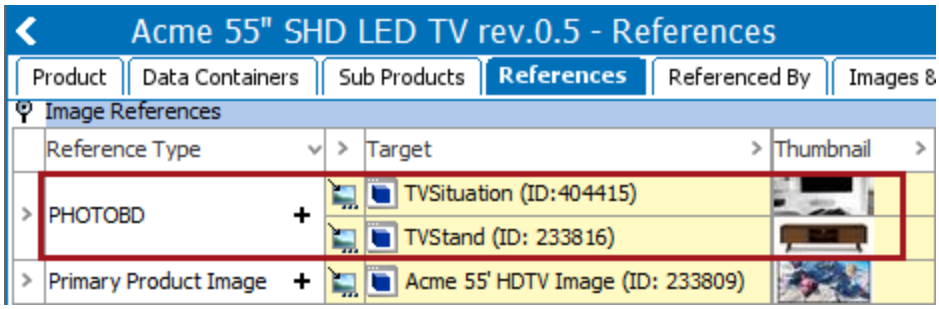
Note: When using replacement rules across contexts with suppressions, include all contexts in an export for best results.

Important: There is no 'undo' option for import. Any mistakes caused by an import must be repaired manually or by running another import with the desired results for the data.

1. Select a few objects that are representative of all objects that need to be modified.
2. To limit the data in the STEPXML file, export the objects to be modified via STEPXML by setting the 'Include <super type>' parameter to 'Selected', and set all other parameters to 'No' or 'None'.
3. Modify the export file to add the appropriate ReplacementRules tags.
4. Modify the export file super type sections to show only the data expected as a result of a successful import.
5. Save the file with a new name.
6. Import the new file to modify the objects.
   - If the results are not as expected, review the import file, resolve any issues, save the file, and import again.
   - If the results are as expected, add the remaining object IDs to the appropriate super type tag, save the file, and import again to update the data.

ReplacementRules Tag Example

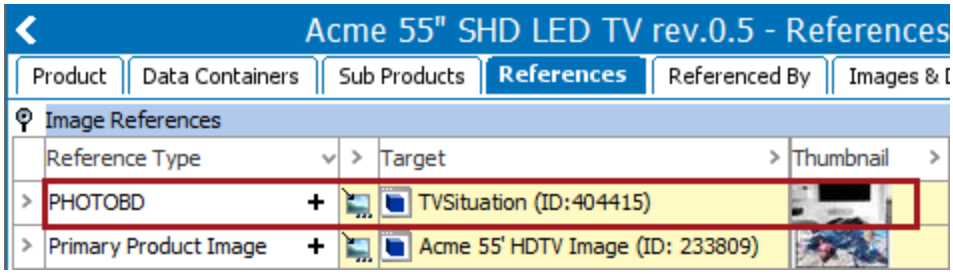
The product with ID 249024 is of the 'Item' object type and has assets with reference type of PHOTOBD and an asset of Primary Product Image reference type. An external system has more current data and will update STEP so the object reflects the true PHOTOBD asset references. The 'TVStand' asset will be removed.



Below is a STEPXML import using ReplacementRules to remove the incorrect 'PHOTOBD' reference type while leaving the correct 'PHOTOBD' reference and the 'PrimaryProductImage' reference type unchanged:

```
01 <?xml version="1.0" encoding="UTF-8" ?>
02 <STEP-ProductInformation UseContextLocale="false" WorkspaceID="Main"
03 ContextID="Context1" ExportContext="Context1"
04 ExportTime="2020-06-26 10:32:25">
05
06 <ReplacementRules>
07   <Products>
08     <ReplaceCrossReferences ReferenceTypeID="PHOTOBD" />
09   </Products>
10 </ReplacementRules>
11
12 <Products>
13   <Product ID="249024" UserTypeID="Item" ParentID="179620">
14     <Name>Acme 55" SHD LED TV</Name>
15     <AssetCrossReference Type="PHOTOBD" AssetID="404415" />
16   </Product>
17 </Products>
18 </STEP-ProductInformation>
```

The result of the import is evident on the References tab of the product editor.



While this is an example for a single product and a single referenced asset type, this method can be effective for updating many objects using search and replace to modify the STEPXML file to define the expected end result.

For detailed examples including the STEPXML template, and the data before and after the use of the ReplacementRules tag, refer to the example topics identified in the following table.

ReplacementRules Tag Structure

This table defines the ReplacementRules elements, applicable object super types, and the description, parameters, and examples.

Important: To ensure you do not remove unintended items, use the ID parameter when available to define changes.

Element	Super Type	Description
ReplaceAttributeGroupLinks	Attributes	For attribute definitions included in the import file, replace existing attribute group links with only those provided in the file. **Note:** Since an attribute must always be linked to / included in at least one attribute group, the last link cannot be removed. No parameters are valid for this element. Examples included in **ReplacementRules in STEPXML for Attributes Examples**
ReplaceAttributeLinks	Classifications Products	For classifications or products included in the import file, replace existing attribute links with those provided in the file. **CrossContext** parameter: • Only dimension dependent objects are impacted by the CrossContext parameter. • If a specified object matches multiple ReplacementRules tags, the 'CrossContext' setting from the most specific rule will be used. For example, if an object being modified matches a replacement rule for all objects of the specified type and also matches a replacement rule with a specified TypeID, the TypeID rule is applied.

Element	Super Type	Description
		• When absent (default), or defined as **CrossContext="N"**, modifies included objects local to the import context only. • When defined as **CrossContext="Y"**, modifies included objects for all contexts. Examples included in: • **ReplacementRules in STEPXML for Classifications Examples** • **ReplacementRules in STEPXML for Products Examples**
ReplaceClassificationReferences	Products	For products included in the import file where the linked alternative classification object type has 'Own Product Links' parameter set to 'No', replace existing product-to-classification link types with those provided in the file. For more information, refer to the **Owns Product Links on Alternate Classifications Object Type** topic. Refer to **ReplaceProductReferences** for classification object types with 'Own Product Links' parameter set to 'Yes'. Refer to **ReplaceCrossReferences** and **ReplaceProductReferences** for other reference types. **TypeID** parameter: • When defined, specifies the product-to-classification link to be modified for the objects in the import file. • When absent, applies to all product-to-classification link for all objects in the import file. **CrossContext** parameter: • Only dimension dependent objects are impacted by the CrossContext parameter. • If a specified object matches multiple ReplacementRules tags, the 'CrossContext' setting from the most specific rule will be used. For example, if an object being modified matches

Element	Super Type	Description
		a replacement rule for all objects of the specified type and also matches a replacement rule with a specified TypeID, the TypeID rule is applied. • When absent (default), or defined as **CrossContext="N"**, modifies included objects local to the import context only. • When defined as **CrossContext="Y"**, modifies included objects for all contexts. Examples included in: • **ReplacementRules in STEPXML for Classifications Examples** • **ReplacementRules in STEPXML for Products Examples**
ReplaceCrossReferences	Assets Classifications Entities Products	For objects included in the import file, replace existing cross references owned by the objects (excluding product-to-classification links - refer to **ReplaceClassificationReferences** and **ReplaceProductReferences**) with those provided in the file. **ReferenceTypeID** parameter: • When defined, specifies the reference type to be modified for the objects in the import file. • When absent, applies to all reference types for all objects in the import file. **CrossContext** parameter: • Only dimension dependent objects are impacted by the CrossContext parameter. • If a specified object matches multiple ReplacementRules tags, the 'CrossContext' setting from the most specific rule will be used. For example, if an object being modified matches a replacement rule for all objects of the specified type and also matches a replacement rule with a specified TypeID, the TypeID rule is applied. • When absent (default), or defined as **CrossContext="N"**, modifies included objects local to the import context only.

Element	Super Type	Description
		• When defined as **CrossContext="Y"**, modifies included objects for all contexts. Examples included in: • **ReplacementRules in STEPXML for Assets Examples** • **ReplacementRules in STEPXML for Entities Examples** • **ReplacementRules in STEPXML for Products Examples**
ReplaceDataContainers	Entities Products	For products or entities included in the import file, replace existing data container objects owned by the objects with those provided in the file. **DataContainerTypeID** parameter: • When defined, specifies the data container types to be modified for the objects in the import file. • When absent, applies to all data container types for all objects in the import file. Examples included in **ReplacementRules in STEPXML for Entities Examples**
ReplaceOverrideSubProducts	Products	For product overrides included in the import file, replace override sub products with those provided in the file. For more information, refer to the **Product-Override Object Types and Commercial Object Types** topic in the **System Setup** documentation. **Note:** Product override children are represented via the 'OverrideSubProduct' elements in STEPXML. Product override objects that only live in the product override structure and are not adopted from elsewhere in the product hierarchy will not be replaced. Examples included in **ReplacementRules in STEPXML for Products Examples**
ReplacePrivilegeRules	UserGroups	For user group definitions included in the import file,

Element	Super Type	Description
		replace existing privilege rules with those provided in the file. No parameters are valid for this element. **Important:** Particularly with this instruction, beware of unintentional results. If **all** privilege rules are removed for **all** users, no users are able to access the STEP system on which the file is imported. Examples included in **ReplacementRules in STEPXML for User Groups Examples**
ReplaceProductReferences	Classifications	For classification object types included in the import file where the 'Own Product Links' parameter is set to 'Yes', replace existing product-to-classification link types with those provided in the file. For more information, refer to the **Owns Product Links on Alternate Classifications Object Type** topic in the **System Setup** documentation. Refer to **ReplaceClassificationReferences** for classification object types with 'Own Product Links' parameter set to 'No'. Refer to **ReplaceCrossReferences** for other reference types. **TypeID** parameter: • When defined, specifies the reference type to be modified for the objects in the import file. • When absent, applies to all references for all objects in the import file. **CrossContext** parameter: • Only dimension dependent objects are impacted by the CrossContext parameter. • If a specified object matches multiple ReplacementRules tags, the 'CrossContext' setting from the most specific rule will be used. For example, if an object being modified matches

Element	Super Type	Description
		a replacement rule for all objects of the specified type and also matches a replacement rule with a specified TypeID, the TypeID rule is applied. • When absent (default), or defined as **CrossContext="N"**, modifies included objects local to the import context only. • When defined as **CrossContext="Y"**, modifies included objects for all contexts. Examples included in **ReplacementRules in STEPXML for Classifications Examples**
ReplaceTargetUserTypeLinks	LinkTypes	**Note:** Target UserTypeID links defined in the import file cannot have values for the definitions being modified. Refer to the **ReplaceUserTypeLinks** element for replacing a link's source. For **link type** definitions (reference types and classification product link types) included in the import file, replace existing target user type links with those provided in the file. Target user type links are links between a link type and an object type that make instances of the object type valid targets for the link type. For **classification product link types**, 'user type links' represent the product and 'target user type links' represent the classification, regardless of the link ownership. For more information, refer to the **Reference and Link Types** topic in the **System Setup Guide** documentation. No parameters are valid for this element. Examples included in **ReplacementRules in STEPXML for User Type Links Examples**
ReplaceUnitLinks	Attributes	For attribute definitions included in the import file, replace existing unit links with those provided in the file.

Element	Super Type	Description
		Note: Units being made invalid cannot be included on the 'Values for Attribute' flipper on the References tab of the Attribute editor. For more information, refer to the **Units** topic in **System Setup** documentation. No parameters are valid for this element. Examples included in **ReplacementRules in STEPXML for Attributes Examples**.
ReplaceUserTypeLinks	Attributes LinkTypes UserTypes	In STEPXML, object types are called 'user types'. **Note:** Source UserTypeID links defined in the import file cannot have values for the definitions being modified. Refer to the **ReplaceTargetUserTypeLinks** element for replacing a link's target. For **attribute** definitions included in the import file, replace existing user type links with those provided in the file. User type links are displayed on the 'Validity' tab of the attribute editor. These are the links between an attribute and an object type that make the attribute valid for instances of the object type. For more information, refer to the **Validity on Description Attributes** topic or the **Validity on Specification Attributes** topic. For **link type** definitions (reference types and classification product link types) included in the import file, replace existing user type links with those provided in the file. User type links are displayed as a Source on the 'Validity' tab of the appropriate editor. These are links between a link type and an object type that make instances of the object type valid sources for the link type. For classification product link types, 'user type links' represent the product and 'target user type links' represent the classification, regardless of the link ownership.

Element	Super Type	Description
		For **user type** definitions (i.e., object type definitions) included in the import file, replace existing user type links with those provided in the file. User type links are displayed on the References tab of the object editor within the Parents flipper. These are links between object types that make an object type legal (relation is owned by the child type). Since user type defined object types must have at least one parent, the last user type link cannot be removed. No parameters are valid for this element. Examples included in **ReplacementRules in STEPXML for User Type Links Examples**.
ReplaceValues	Assets Classifications Entities Products	For objects included in the import file, replace existing values owned by the object with those provided in the file. **AttributeID** parameter: • When defined, specifies the attribute to be modified for the objects in the import file. **AttributeGroupID** parameter: • When defined, specifies the attribute group and modifies all attributes in the group for the objects in the import file. **Important:** When both the AttributeID and AttributeGroupID parameters are absent, all attributes are modified for the objects of the identified super type sections of the import file. **CrossContext** parameter: • Only dimension dependent objects are impacted by the CrossContext parameter. • If a specified object matches multiple ReplacementRules tags, the 'CrossContext' setting from the most specific rule will be used. For example, if an object being modified matches a replacement rule for all objects of the specified

Element	Super Type	Description
		type and also matches a replacement rule with a specified TypeID, the TypeID rule is applied. • When absent (default), or defined as **CrossContext="N"**, modifies included objects local to the import context only. • When defined as **CrossContext="Y"**, modifies included objects for all contexts. Examples included in: • **ReplacementRules in STEPXML for Assets Examples** • **ReplacementRules in STEPXML for Classifications Examples** • **ReplacementRules in STEPXML for Products Examples**

ReplacementRules in STEPXML for Assets Examples

For simplicity, this example uses minimal data. Once an import file is verified for a small dataset, the use case can be expanded to include additional objects using search / replace as necessary within a larger import file.

Important: To ensure you do not remove unintended items, use the ID parameter when available to define changes.

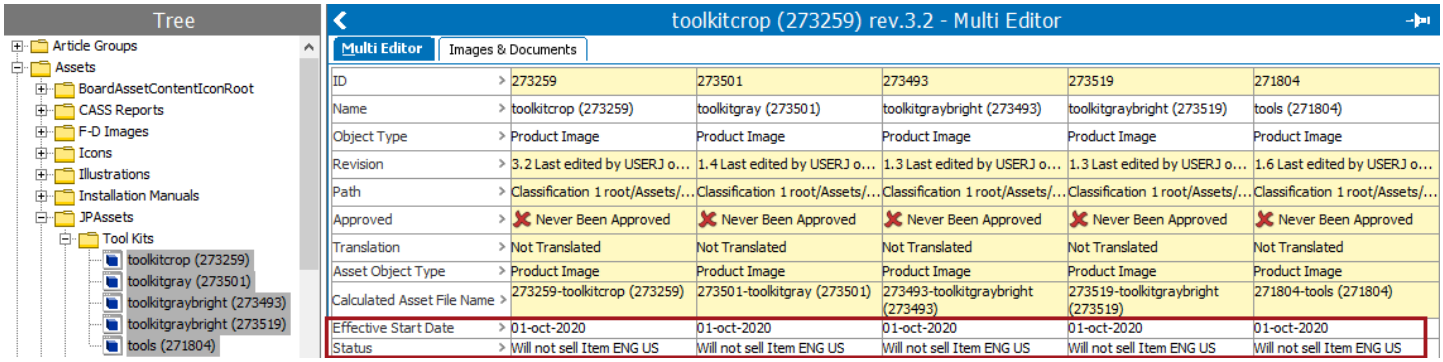
This example demonstrates using the ReplacementRules tag via inbound STEPXML to:

- Remove asset values

For more information, refer to the **ReplacementRules Tag in STEPXML** topic.

Remove Asset Values

The attribute values should be removed for each asset in the 'Tool Kits' folder. The 'Effective Start Date' and 'Status' attribute values are shown in the image below. Any other existing values for these assets would also be removed by this example XML.



ID	273259	273501	273493	273519	271804
Name	toolkitcrop (273259)	toolkitgray (273501)	toolkitgraybright (273493)	toolkitgraybright (273519)	tools (271804)
Object Type	Product Image	Product Image	Product Image	Product Image	Product Image
Revision	3.2 Last edited by USERJ o...	1.4 Last edited by USERJ o...	1.3 Last edited by USERJ o...	1.3 Last edited by USERJ o...	1.6 Last edited by USERJ o...
Path	Classification 1 root/Assets/...	Classification 1 root/Assets/...	Classification 1 root/Assets/...	Classification 1 root/Assets/...	Classification 1 root/Assets/...
Approved	Never Been Approved	Never Been Approved	Never Been Approved	Never Been Approved	Never Been Approved
Translation	Not Translated	Not Translated	Not Translated	Not Translated	Not Translated
Asset Object Type	Product Image	Product Image	Product Image	Product Image	Product Image
Calculated Asset File Name	273259-toolkitcrop (273259)	273501-toolkitgray (273501)	273493-toolkitgraybright (273493)	273519-toolkitgraybright (273519)	271804-tools (271804)
Effective Start Date	01-oct-2020	01-oct-2020	01-oct-2020	01-oct-2020	01-oct-2020
Status	Will not sell Item ENG US	Will not sell Item ENG US	Will not sell Item ENG US	Will not sell Item ENG US	Will not sell Item ENG US

Online help includes a condensed version of the export file for the assets generated as STEPXML in the Export Data option on the File menu (not the Images & Documents export option). Set the 'Include Assets' parameter to 'Selected' and set all other parameters to 'No' or 'None'.

Note: Sections of the XML file that do not impact this example have been removed for brevity.

The export file for assets can include types of special attributes that are not affected by replacement rules but are illustrated in bold for Asset ID="271804" above. These attributes cannot be modified upon import:

- System Properties attributes are shown with AttributeID that starts with 'asset.'. For example:

```
<Value AttributeID="asset.uploaded">2018-12-07 09:31:38</Value>
```

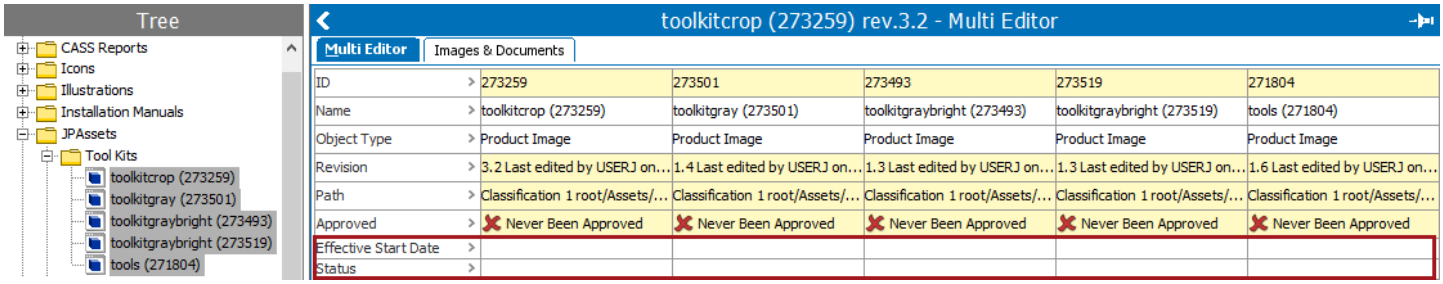
- Calculated attributes are shown with 'Derived="true"'. For example:

```
<Value AttributeID="Asset Object Type" Derived="true">Product Image</Value>
```

Online help includes the export file which has been modified as follows for import:

- Add the ReplacementRules <ReplaceValues> tag for the <Assets> super type.
- Remove the <Name> and <Values> sections, leaving only the asset IDs.

As shown in the asset editor, all attribute values on the assets are removed by importing the modified file.



ReplacementRules in STEPXML for Attributes Examples

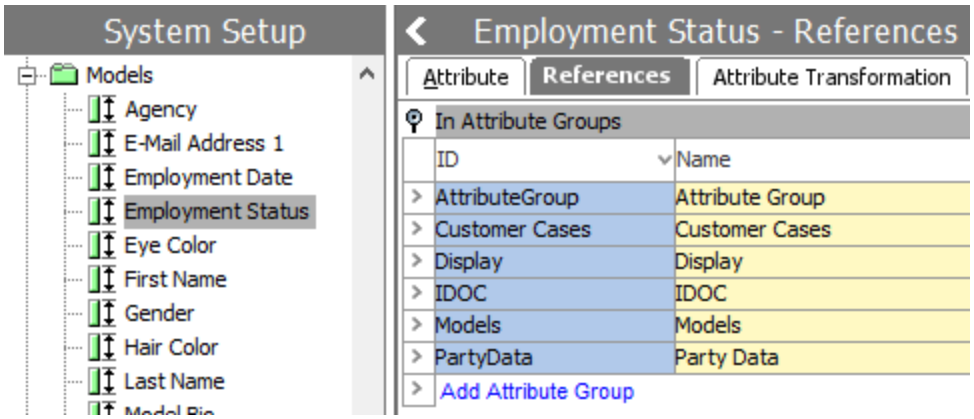
These examples demonstrate using the ReplacementRules tag via inbound STEPXML to:

- Remove attribute group links
- Replace unit links

For more information, refer to the **ReplacementRules Tag in STEPXML** topic.

Remove Attribute Group Links

The 'Employment Status' attribute is in multiple attribute groups, but should only be displayed in the Models group.

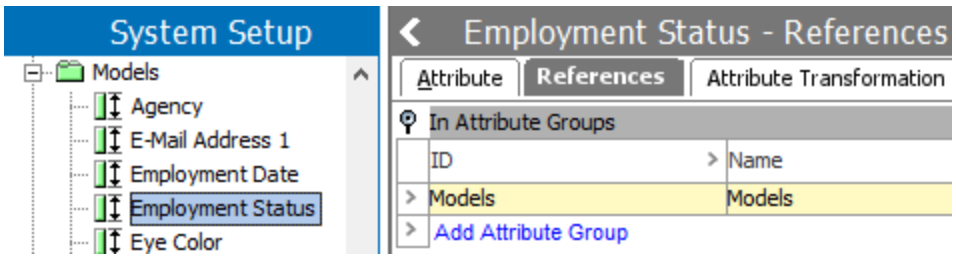


Online help includes the export file for the 'Employment Status' attribute generated as STEPXML by setting the 'Include Attributes' parameter to 'Selected', and setting all other parameters to 'No' or 'None'.

Online help includes the export file which has been modified as follows for import:

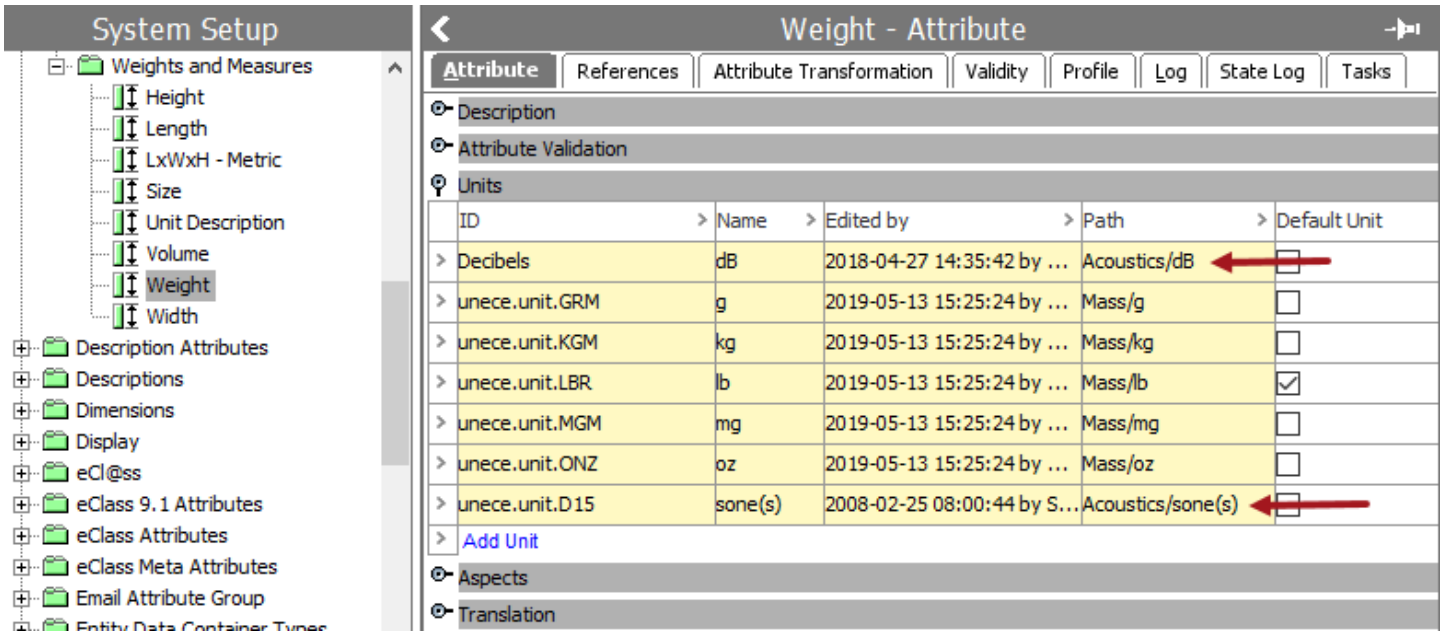
- Add the ReplacementRules <ReplaceAttributeGroupLinks> tag for the <Attributes> super type.
- Remove all of the AttributeGroupLink data except the group 'Models' from the <AttributeList> super type section.

The result of importing the modified file is shown in the System Setup attribute editor.

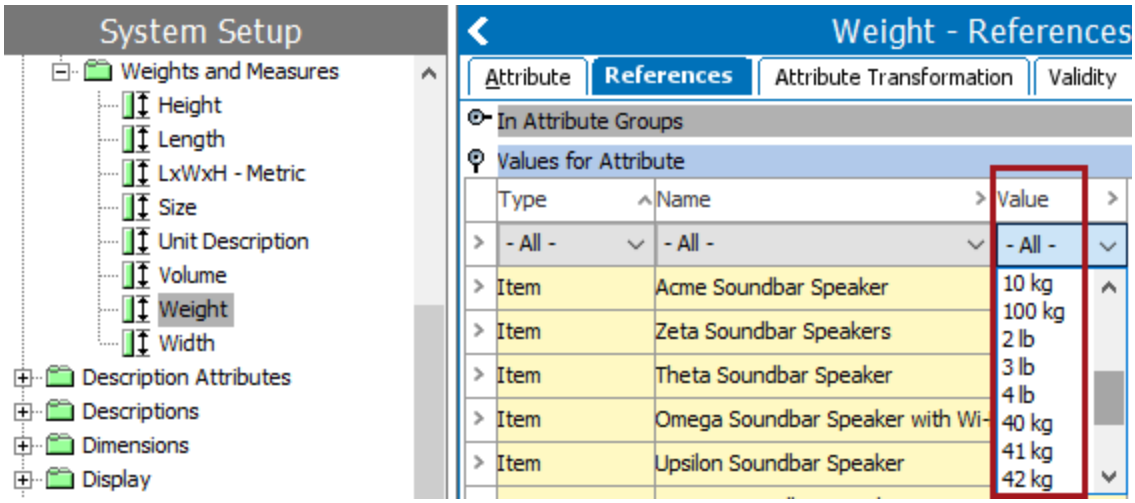


Replace Unit Links

The 'Weight' attribute has multiple valid units, but the 'acoustics' units are not accurate and should be removed. Additionally, the missing 'mass' units should be added.



In the attribute editor, the References tab 'Values for Attribute' flipper includes all units being used. In this example, the filter option on the Value column shows that only 'kg' and 'lb' units are applied. For larger data sets, this validation option may be not feasible.



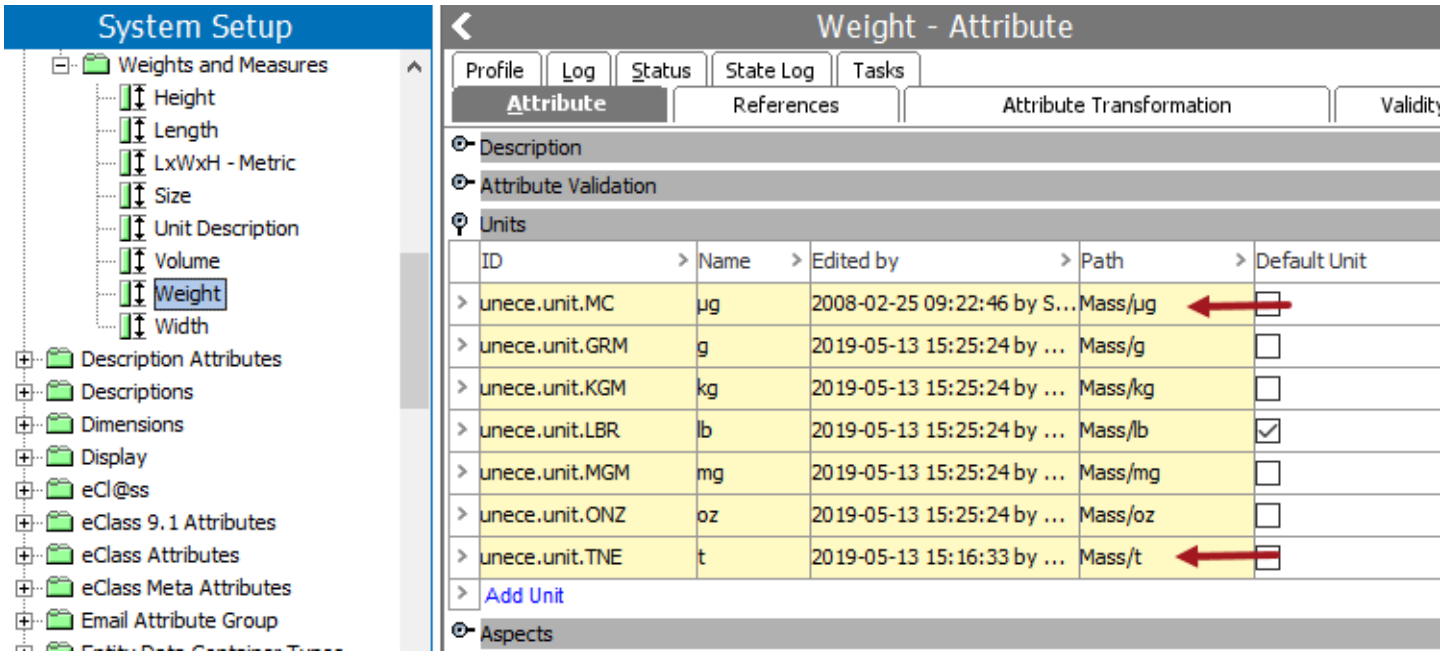
Note: Any unit applied to the newest revision of an object is not removed upon import regardless of the ReplacementRules setting. If a unit is removed and was in use for an older revision, attempting to restore the effected revision displays an error and allows a valid unit to be selected.

Online help includes the export file for the 'Weight' attribute as STEPXML by setting the 'Include Attributes' parameter to 'Selected', and setting all other parameters to 'No' or 'None'.

Online help includes the export file which has been modified as follows for import:

- Add the ReplacementRules <ReplaceUnitLinks> tag for the <Attributes> super type.
- Remove the unnecessary units 'Decibels' and 'D15'.
- Add missing units 'MC' and 'TNE'.
- Remove tags for data originally output that is not being modified (such as, <MetaData> and <AttributeGroupLink >.

The result of importing the modified file is shown in the System Setup attribute editor.



ReplacementRules in STEPXML for Classifications Examples

For simplicity, examples use minimal data. Once an import file is verified for a small dataset, the use case can be expanded to include additional objects using search / replace as necessary within a larger import file.

Important: To ensure you do not remove unintended items, use the ID parameter when available to define the changes.

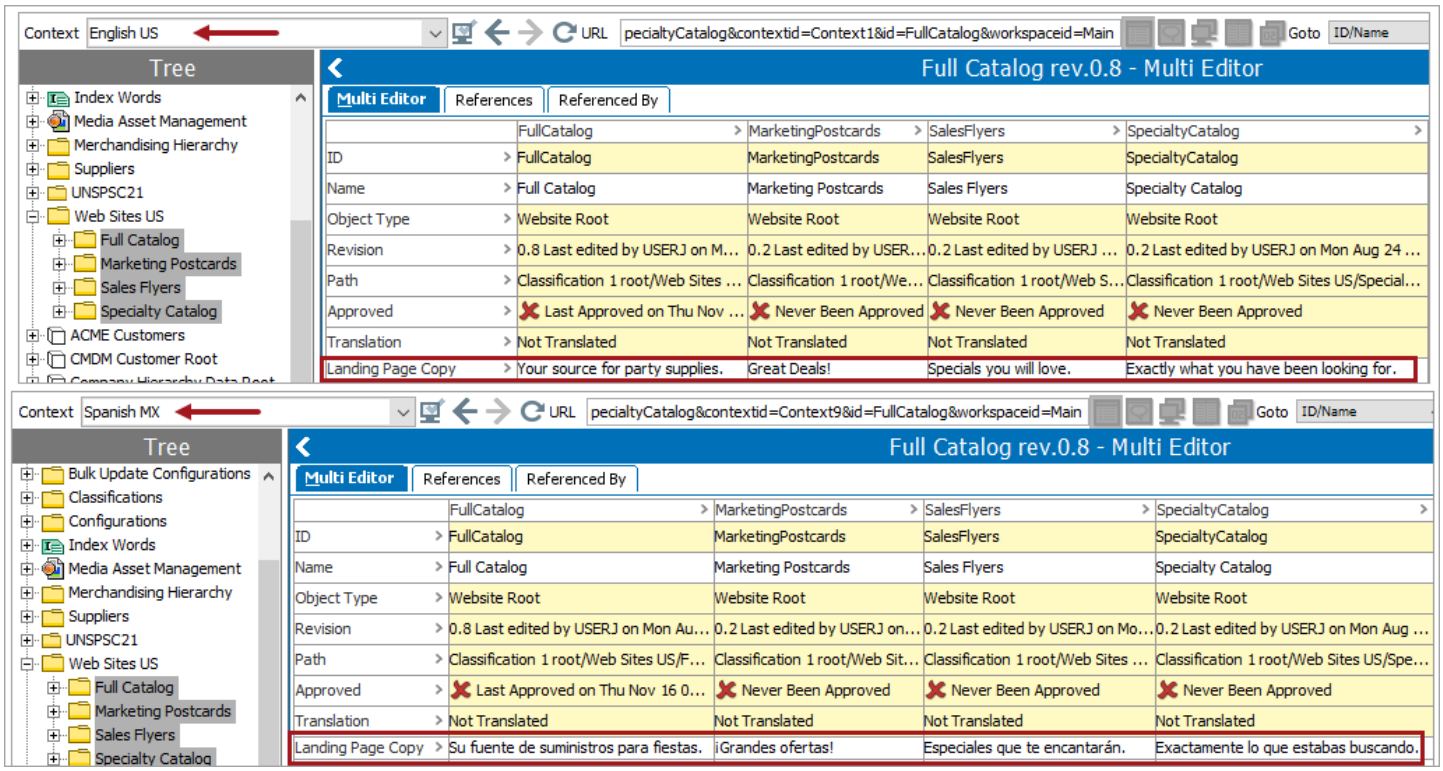
These examples demonstrate using the ReplacementRules tag via inbound STEPXML to:

- Remove values on classification folders
- Replace product references

For more information, refer to the **ReplacementRules Tag in STEPXML** topic.

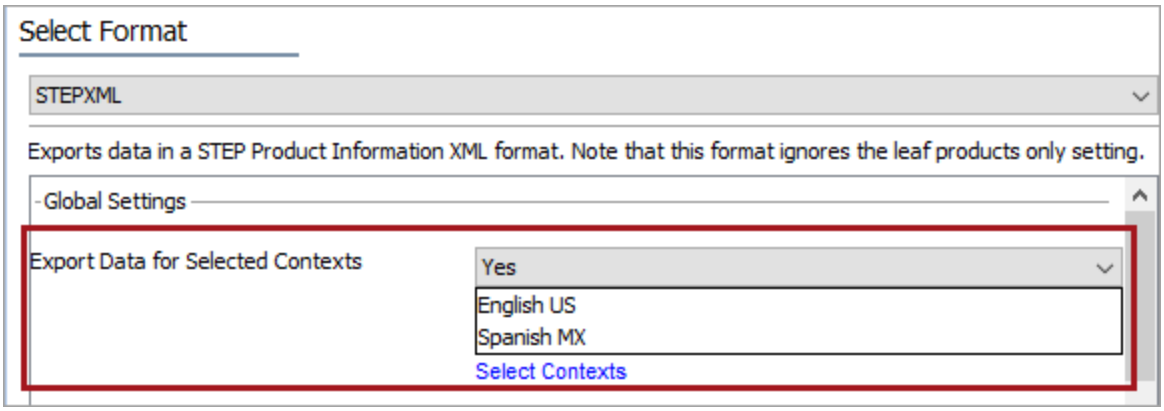
Remove Values On Classification Folders

The 'Landing Page Copy' and 'Print Tag Line' dimension dependent attributes are valid for classification folders. However, the text in 'Landing Page Copy' for both contexts should be removed for all contexts for all classifications except for the SalesFlyers.



Exporting objects using STEPXML includes the parameter to select contexts and gives access to dimension-dependent attribute values. The following STEPXML format selections are used for exporting the classification objects. For details, refer to the **STEPXML Outbound Parameters** topic.

- On the Select Objects step, select 'Classification' from the Export dropdown and add the classification objects.
- The 'Export Data for Selected Contexts' parameter is set to 'Yes' and the necessary contexts are selected.



- The 'Include Classifications' parameter is set appropriately, in this case, as 'Selected'.

All other STEPXML format parameter selections are set to 'No' or 'None' in this example.

Note: Qualifier IDs are generated at the time of export and are valid for the current file. The 'Qualifier ID' assigned in the 'Qualifiers' element of the output file corresponds to the context displayed in workbench or Web UI. For more information, refer to the **Multi-Context Data in STEPXML** topic.

Online help includes a condensed version of the generated export file showing only one of the classifications that will be modified.

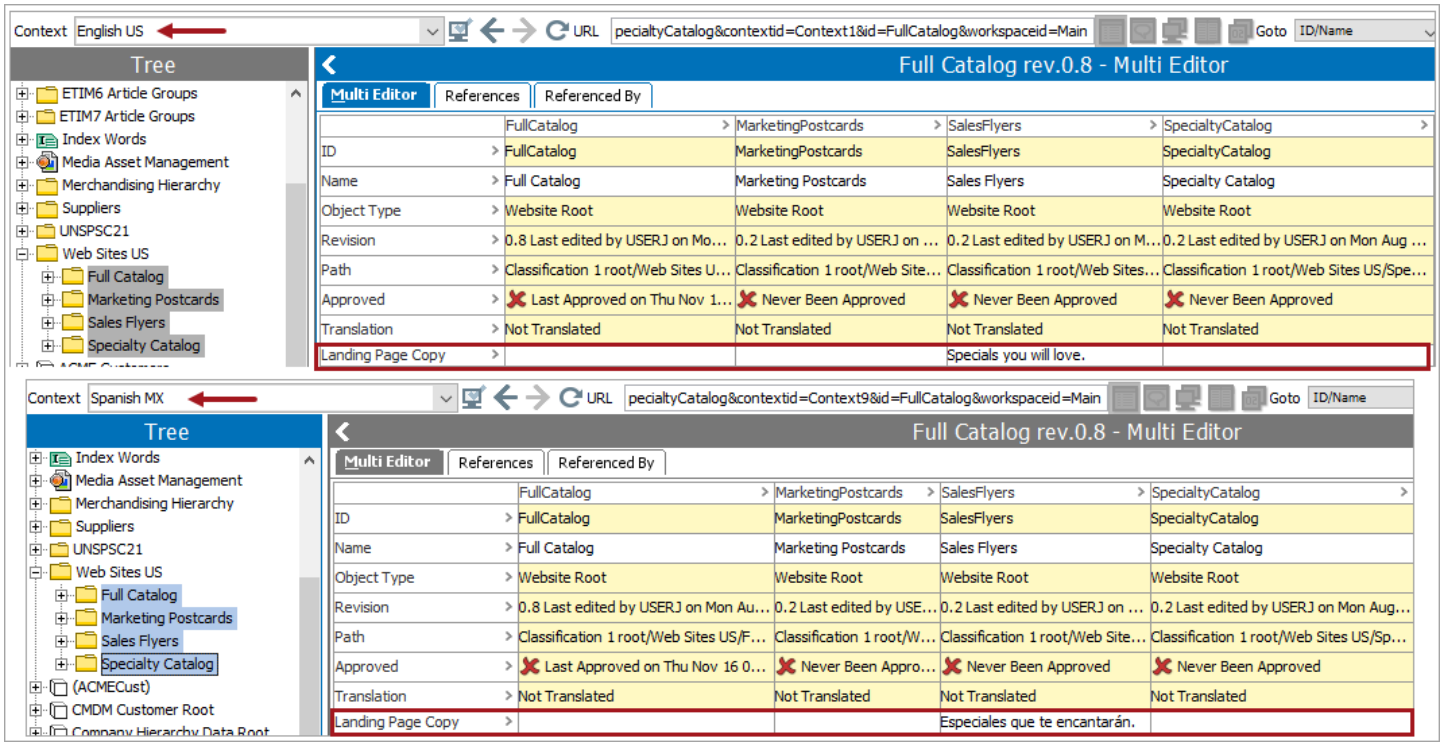
Note: Sections of the XML file that do not impact this example have been removed for brevity.

Online help includes a condensed version of the export file which has been modified as follows for import:

- Add the ReplacementRules <ReplaceValues AttributeID="LandingPageCopy" CrossContext="Y"/> tag for the <Classifications> super type.
- Remove the <ValueGroup> tags for the 'LandingPageCopy' attribute on Classification IDs "FullCatalog", "MarketingPostcards", and "SpecialtyCatalog". The "SalesFlyers" values will remain.

Note: Sections of the XML file that do not impact this example have been removed for brevity.

The result of importing the modified file is shown in the classification multi editor.

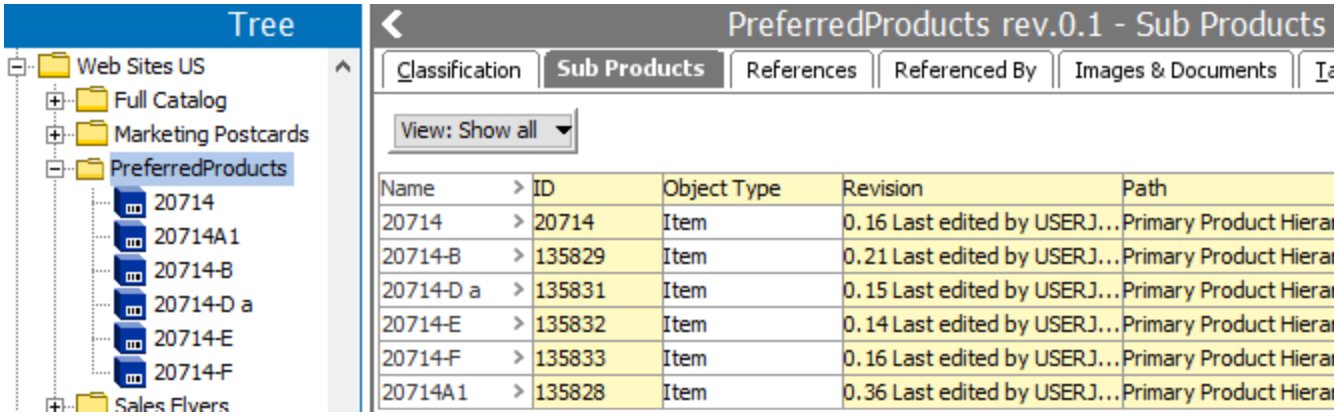


Replace Product References

The **Product to Classification Link Types** topic describes linking products to classifications while the actual products are additionally displayed as children of the classification. The 'Owns Product Link' setting on the classification object type determines where the linked products are included in a STEPXML export.

ReplaceProductReferences for classifications only affects links owned by the classification, indicated by the 'Yes' setting on the 'Own Product Links' parameter on the classification object type. When the classification object type parameter is set to 'No' use ReplaceClassificationReferences. For more information, refer to the **Owns Product Links on Alternate Classifications Object Type** topic in the **System Setup** documentation.

In this example, the 'Preferred' product-to-classification link type has been used to classify multiple products, which displays the products as Sub Products. Some sub products are no longer 'preferred'.

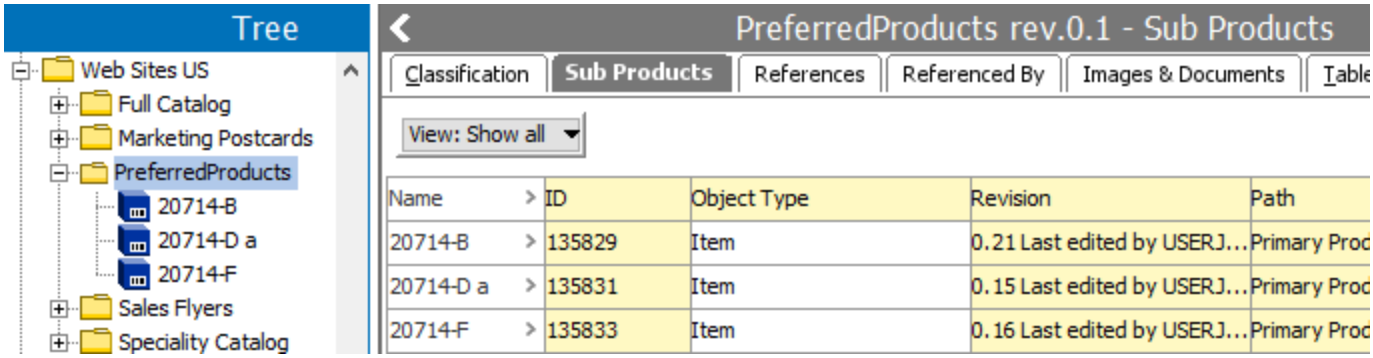


Online help includes the export file for the classifications as STEPXML generated by using the Export parameter set to 'Classification' and selecting the necessary classifications. The 'Include Classifications' parameter is set to 'Selected' and all other parameters are set to 'No' or 'None'.

Online help includes the export file which has been modified as follows for import:

- Add the ReplacementRules <ReplaceProductReferences> tag for the <Classifications> super type.
- Remove the product IDs that should no longer be classified as 'preferred'.

The result of importing the modified file is shown in the classification editor.



ReplacementRules in STEPXML for Entities Examples

For simplicity, examples use minimal data. Once an import file is verified for a small dataset, the use case can be expanded to include additional objects using search / replace as necessary within a larger import file.

Important: To ensure you do not remove unintended items, use the ID parameter when available to define the changes.

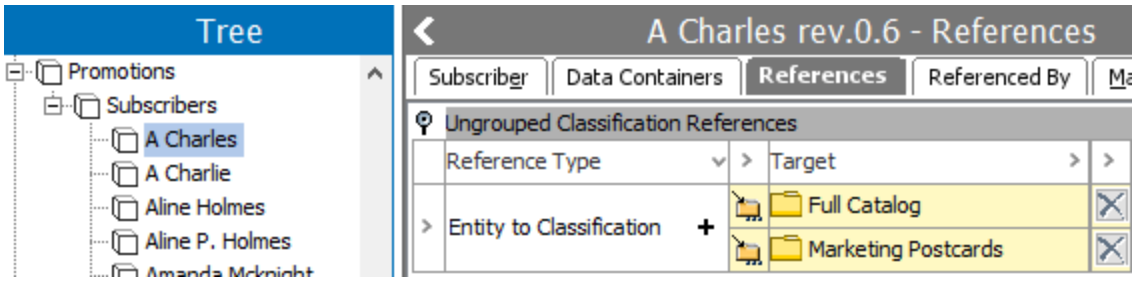
These examples demonstrate using the ReplacementRules tag via inbound STEPXML to:

- Replace cross references
- Replace data containers

For more information, refer to the **ReplacementRules Tag in STEPXML** topic.

Replace Cross References

The following entity includes items for the multivalued 'Entity to Classification' reference:



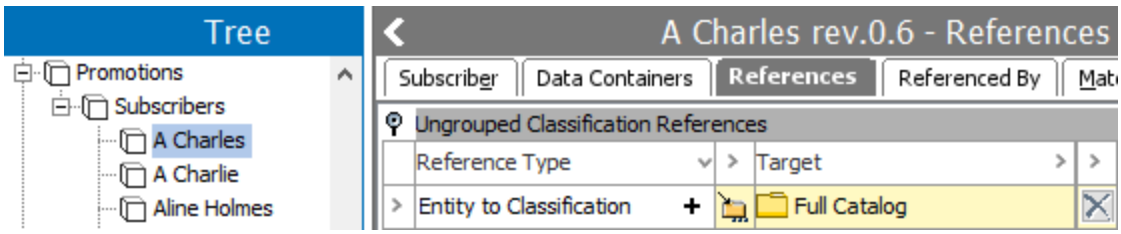
Online help includes the export file for the entity as STEPXML by setting the 'Include Entities' parameter to 'Selected' and setting all other parameters to 'No' or 'None'.

The subscriber A Charles only wants to receive the Full Catalog so the Marketing Postcards classification should be removed.

Online help includes the export file which has been modified as follows for import:

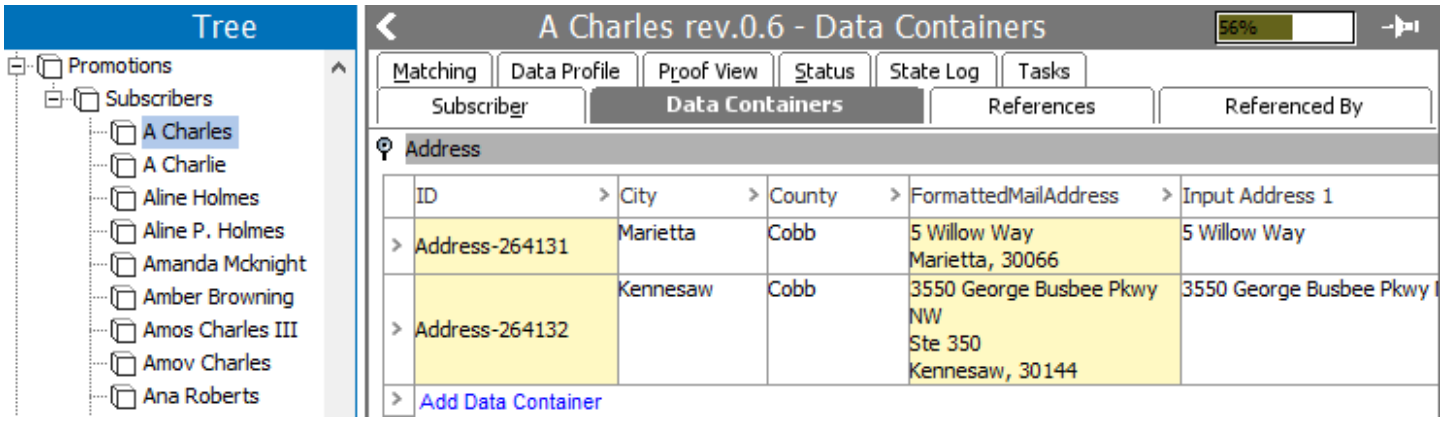
- Add the ReplacementRules <ReplaceCrossReferences> tag for the <Entities> super type.
- Remove data from the <Entities> tag until only the required data is left.

The result of importing the modified file is shown in the entity editor.



Replace Data Containers

The following entity includes multiple items for the multivalued Address data container:



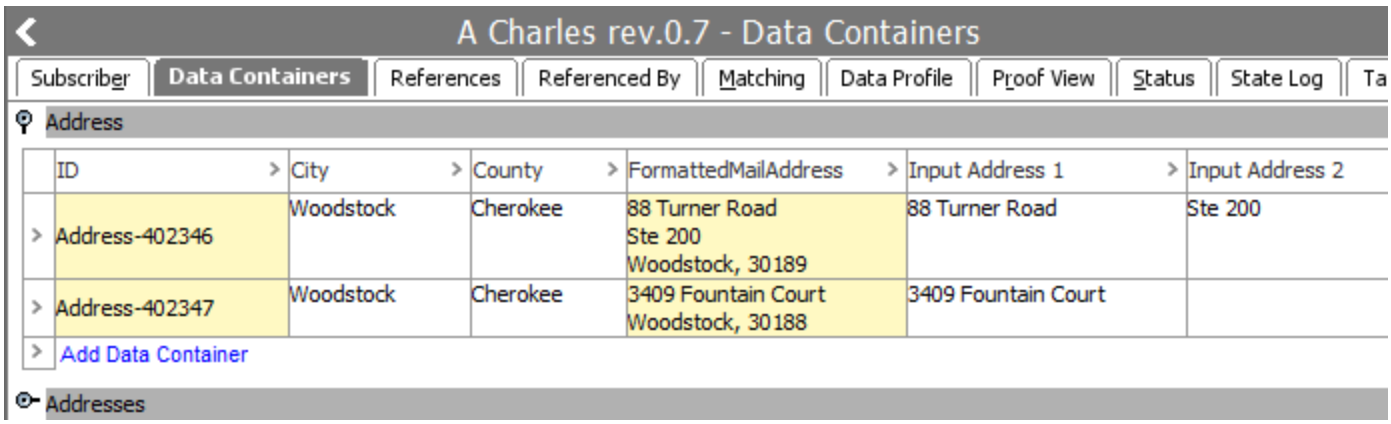
Online help includes the export file for the data containers on the selected entity as STEPXML. Set the 'Include Data Containers' parameter to 'Yes' and set all other parameters to 'No' or 'None'.

The subscriber entity has new addresses that should replace both of the current addresses.

Online help includes the export file which has been modified as follows for import:

- Add the ReplacementRules <ReplaceDataContainers> tag for the <Entities> super type.
- Remove entity tags not being modified.
- Remove ID from the <DataContainer> tag to create new data containers.
- Remove the <FormattedMailAddress> calculated attribute that combines values from the attributes.
- Add the new address information.

The result of importing the modified file is shown in the 'Subscriber' entity editor.


A Charles rev.0.7 - Data Containers
Subscriber
Data Containers
References
Referenced By
Matching
Data Profile
Proof View
Status
State Log
Ta
Address
ID
City
County
FormattedMailAddress
Input Address 1
Input Address 2
Address-402346
Woodstock
Cherokee
88 Turner Road
Ste 200
Woodstock, 30189
88 Turner Road
Ste 200
Address-402347
Woodstock
Cherokee
3409 Fountain Court
Woodstock, 30188
3409 Fountain Court
Add Data Container
Addresses

ReplacementRules in STEPXML for Products Examples

For simplicity, examples use minimal data. Once an import file is verified for a small dataset, the use case can be expanded to include additional objects using search / replace as necessary within a larger import file.

Important: To ensure you do not remove unintended items, use the ID parameter when available to define the changes.

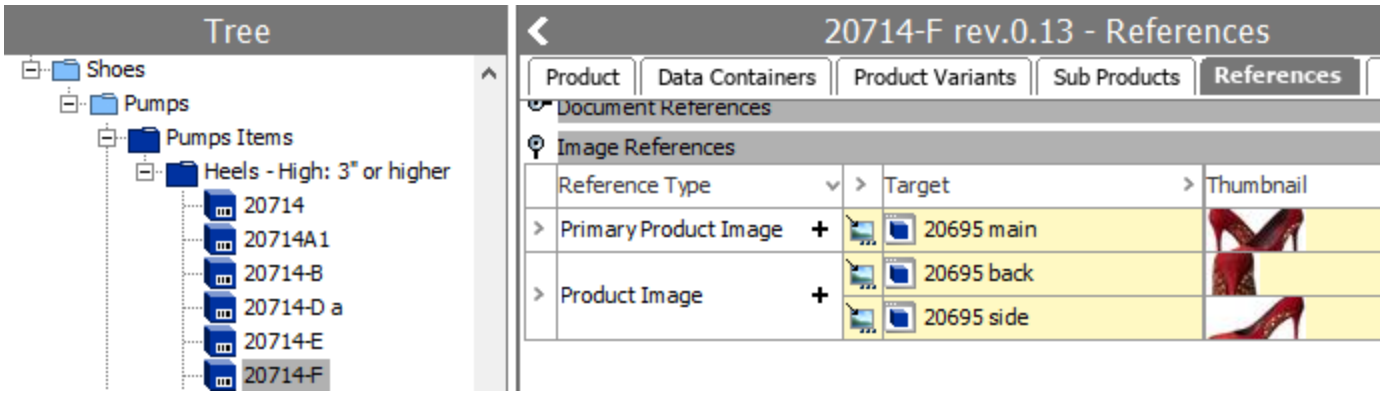
These examples demonstrate using the ReplacementRules tag via inbound STEPXML to:

- Remove references (by type or by all)
- Remove unnecessary attribute links
- Replace attribute values by context
- Update override sub products
- Replace classification references

For more information, refer to the **ReplacementRules Tag in STEPXML** topic.

Remove References

The following product includes multiple Image and Document (asset) references using the 'Product Image' and the 'Primary Product Image' reference types.



Online help includes the export file for the product as STEPXML generated by setting the 'Include Products' parameter to 'Selected' and setting all other parameters to 'No' or 'None'.

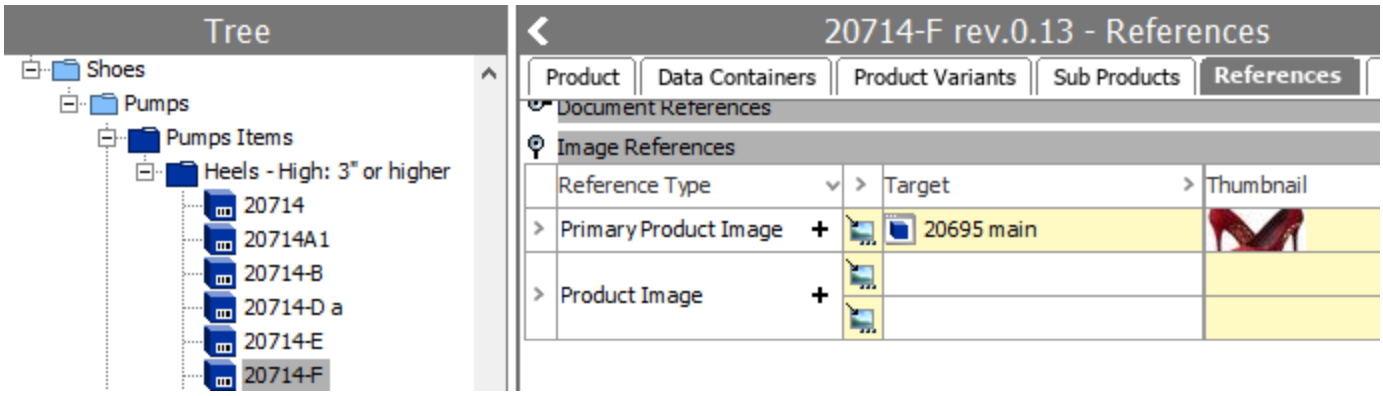
Remove References by Type

This example removes references of a single type. Alternatively, all references, regardless of type, can be removed using the next process.

Online help includes the export file which has been modified as follows for import:

- Add the ReplacementRules <ReplaceCrossReferences> tag for the <Products> super type.
- Remove the reference types no longer needed in the <Products> super type.

The result of importing the modified file is shown in the product editor:

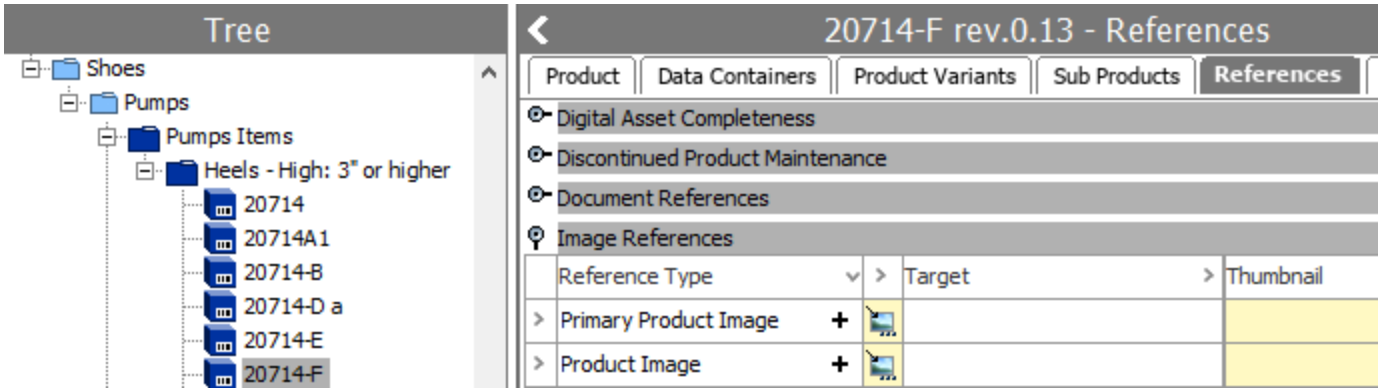


Remove All References

Online help includes the alternative export file which has been modified as follows for import:

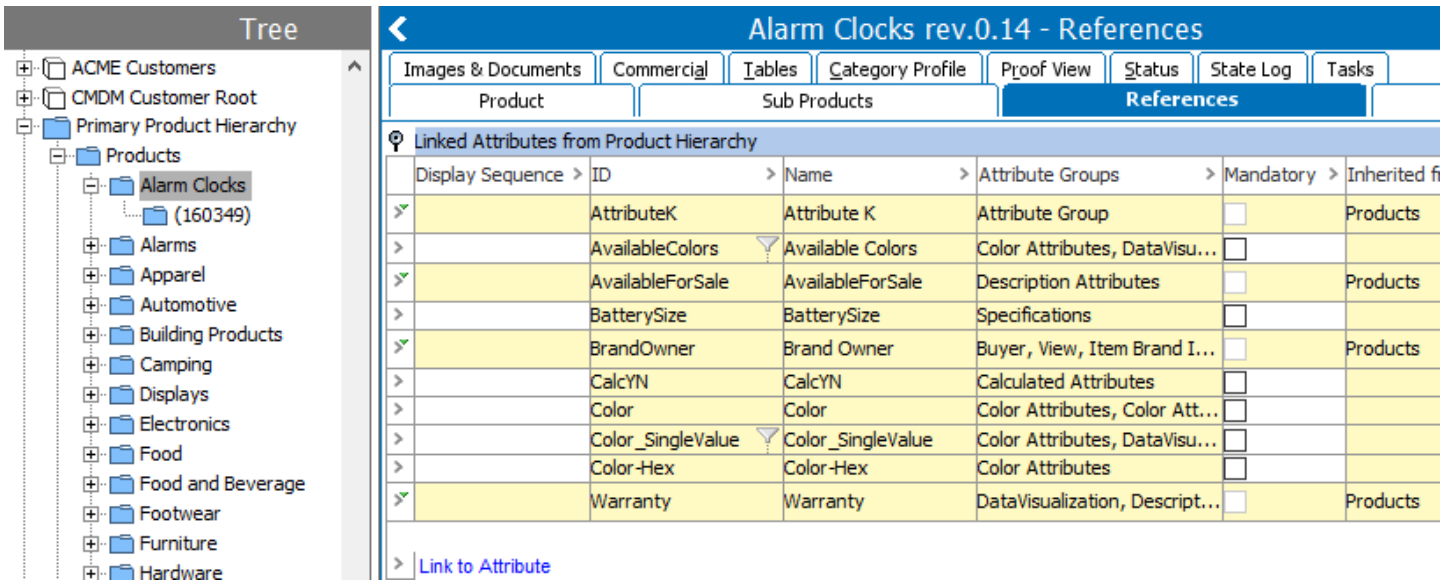
- Add the ReplacementRules <ReplaceCrossReferences> tag without a reference ID for the <Products> super type to remove all references from the products in the import file.
- Remove all reference types in the <Products> super type.

The result of importing the modified file is shown in the product editor:



Remove Unnecessary Attribute Links

The 'Alarm Clocks' object includes multiple local attribute links, but only the 'Color' local attribute is required. Attribute links for AvailableColors, Color_SingleValue, and Color-Hex should be removed.

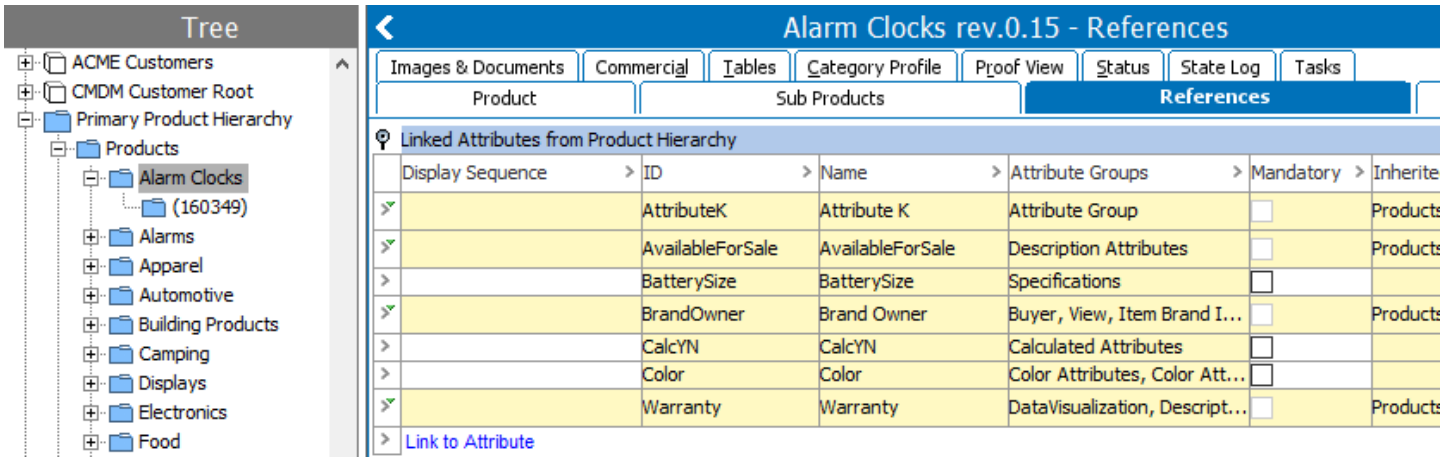


Online help includes the export file for the product as STEPXML generated by setting the 'Include Products' parameter to 'Minimum' and setting all other parameters to 'No' or 'None'.

Online help includes the export file which has been modified as follows for import:

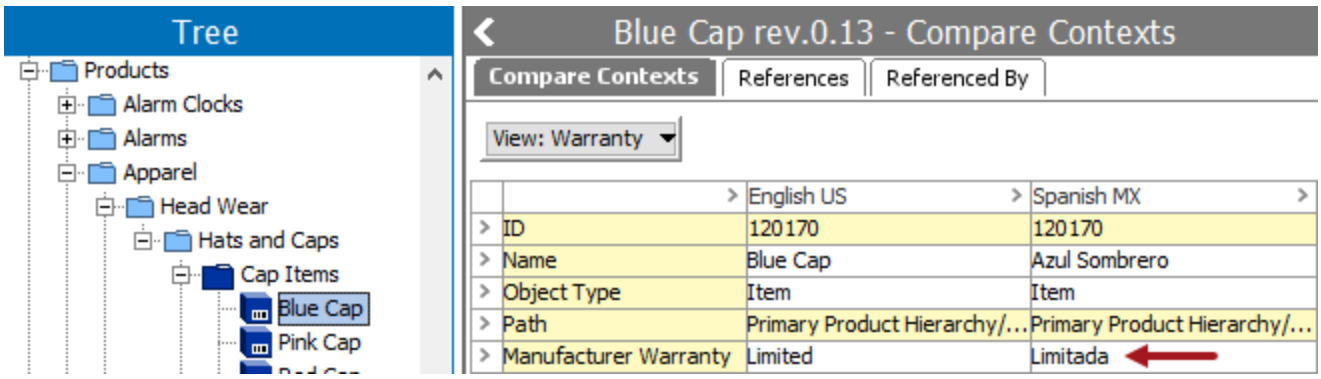
- Add the ReplacementRules <ReplaceAttributeLinks> tag for the <Products> super type.
- Remove attribute link data from the <Products> super type section to be deleted.

The result of importing the modified file is shown in the product editor.



Replace Attribute Values by Context

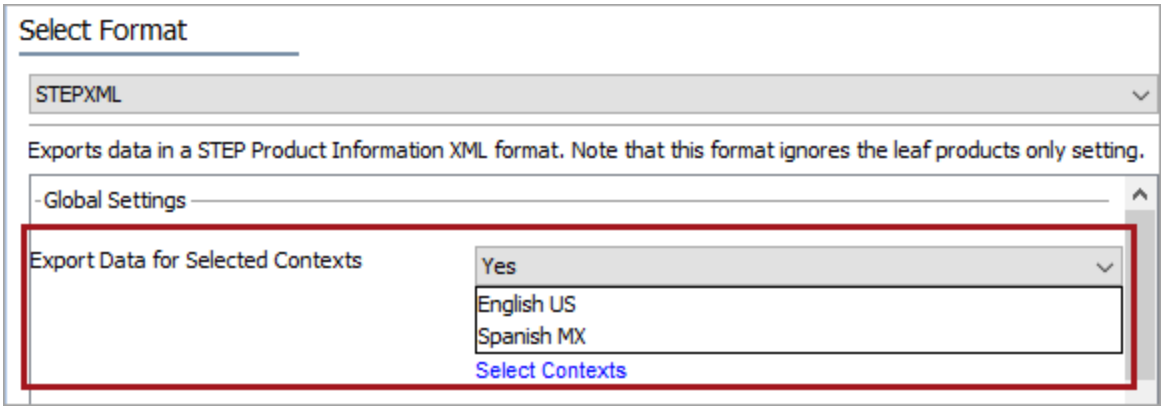
The 'Blue Cap' product includes different values for the language-dependent 'Manufacturer Warranty' attribute. The values in the Spanish MX context should be removed.



Exporting objects using STEPXML includes the option to select contexts and gives access to dimension dependent attribute values.

The following STEPXML format selections are used for exporting the 'Blue Cap' object. For details, refer to the **STEPXML Outbound Parameters** topic.

- Export Data for Selected Contexts parameter is set 'Yes' and the necessary contexts are selected.



- Include Products parameter is set appropriately, in this case, 'Selected'.
- Include Product Attribute Values parameter is set to 'Yes'.

Set all other STEPXML format parameters to 'No' or 'None'.

Note: Qualifier IDs are generated at the time of export and are valid for the current file. The 'Qualifier ID' assigned in the 'Qualifiers' element of the output file corresponds to the context displayed in workbench or Web UI. For more information, refer to the **Context Data in STEPXML** topic.

Online help includes a condensed version of the generated export file showing only one of the classifications that will be modified.

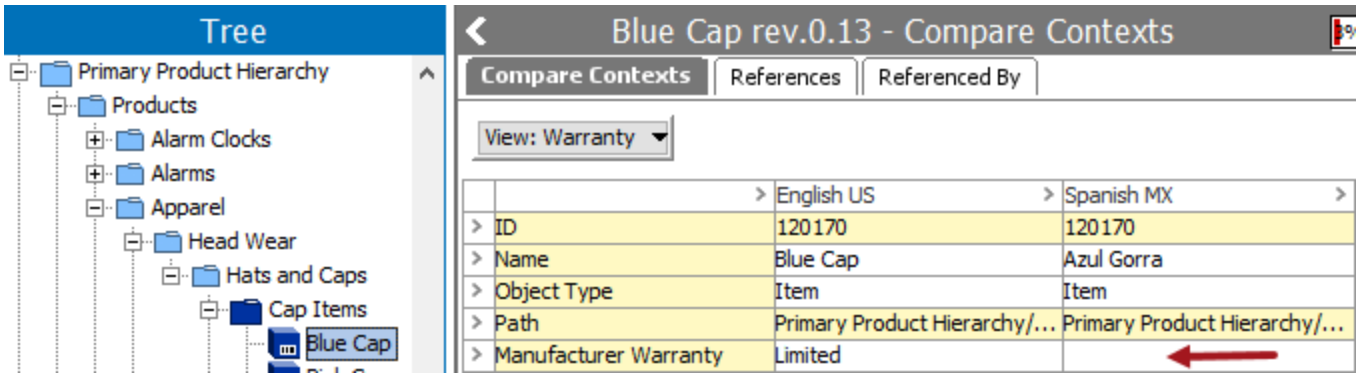
Note: Sections of the XML file that do not impact this example have been removed for brevity.

Online help includes the export file which has been modified as follows:

- Add the ReplacementRules <ReplaceValues CrossContext="Y"> tag for the <Products> super type.
- Remove the value of the 'Warranty' attribute for the appropriate Qualifier ID for the Spanish MX context.

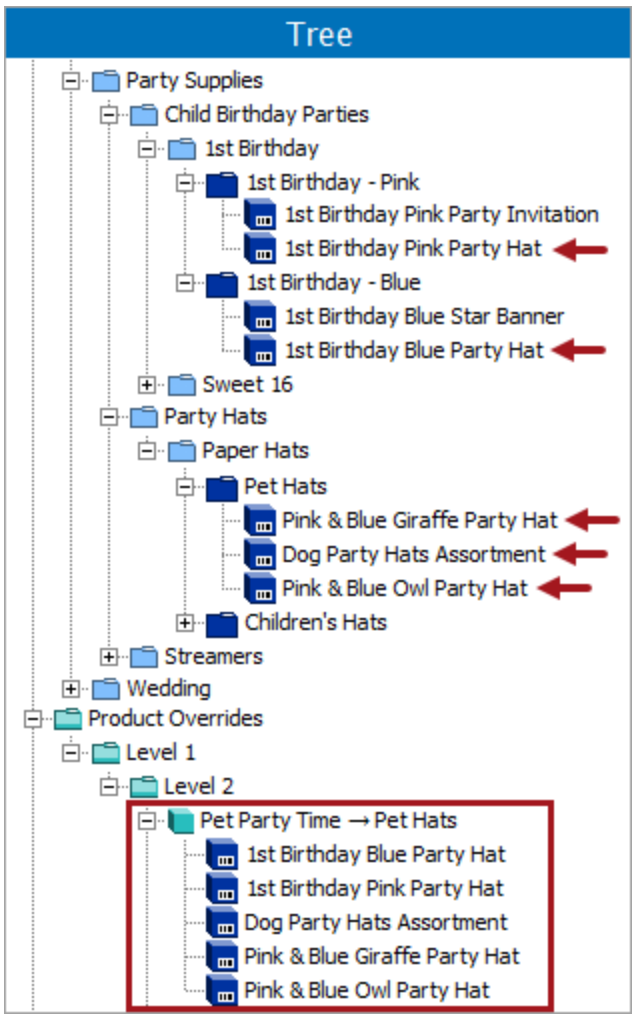
Note: Sections of the XML file that do not impact this example have been removed for brevity.

The result of importing the modified file is shown in the product editor.



Update Override Sub Products

The 'Pet Party Time -> Pet Hats' override object includes overrides for several products. Both of the '1st Birthday...' products from the Child Birthday Parties node should be removed as overrides. For more information on overrides, refer to the **Product-Override Object Types and Commercial Object Types** topic in the **System Setup** documentation.



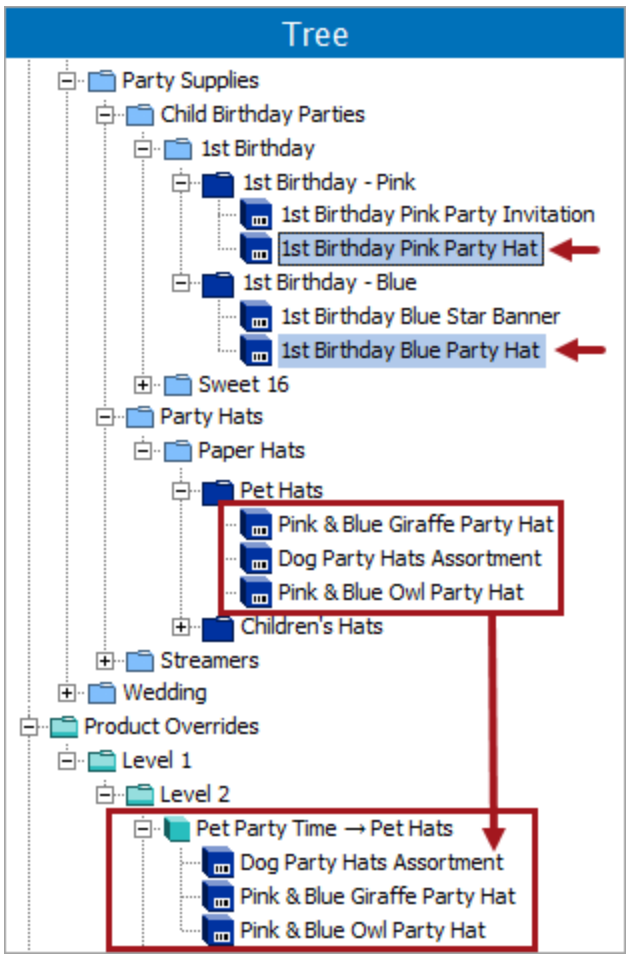
Online help includes a condensed version of the STEPXML export file generated by selecting the override product as well as the overridden products. Set the 'Include Overridden Products' parameter to 'Yes', the 'Include Products' parameter to 'Selected', and set all other parameters to 'No' or 'None'.

Note: Sections of the XML file that do not impact this example have been removed for brevity.

Online help includes the export file that has been modified as follows:

- Add the ReplacementRules <ReplaceOverrideSubProducts> tag for the <Products> super type.
- Remove the unneeded override items.

The result of importing the modified file is shown in the Tree.



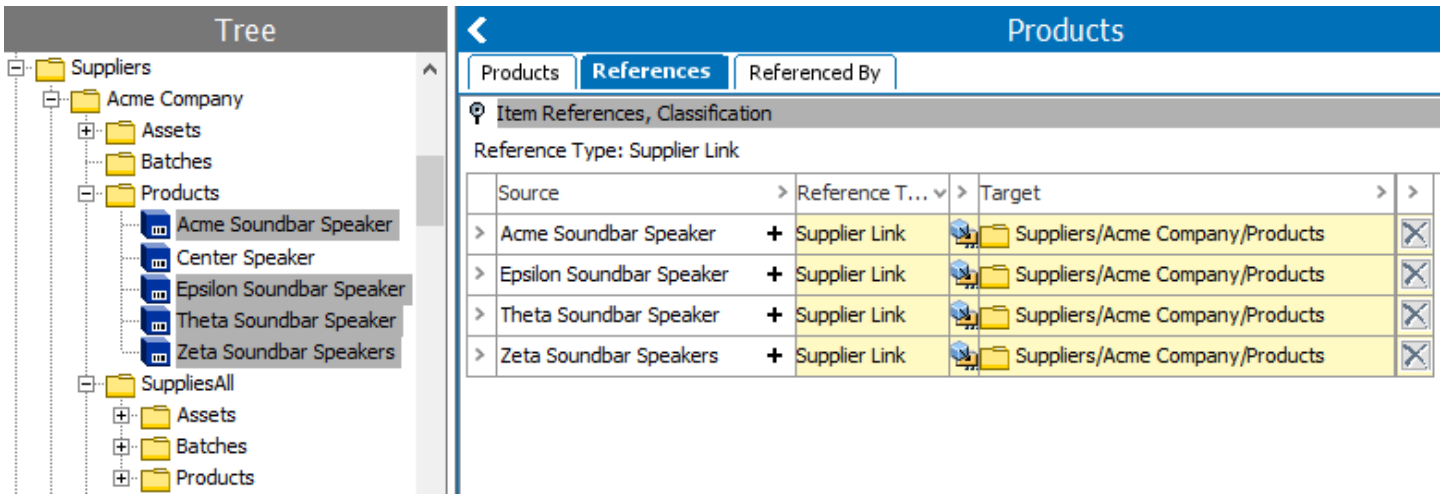
Replace Classification References

The **Product to Classification Link Types** topic describes linking products to classifications while the actual products are additionally displayed as children of the classification. The 'Owns Product Link' setting on the classification object type determines where the linked products are included in a STEPXML export.

For the Products supertype ReplaceClassificationReferences replacement rule, only links owned by the product side (the 'Owns Product Links' = No is the setting for the classification object type that is valid for the link type) are impacted. Links owned by the classification side must be modified via the Classifications supertype ReplaceProductReferences replacement rule. For more information, refer to the **Owns Product Links on Alternate Classifications Object Type** topic in the **System Setup** documentation.

Important: To ensure you do not remove unintended items, use the ID parameter when available to define the changes.

In this example, the soundbar speaker objects include references to the Acme Company supplier Products classification folder. Instead, they should reference the SuppliesAll supplier's Products classification folder.



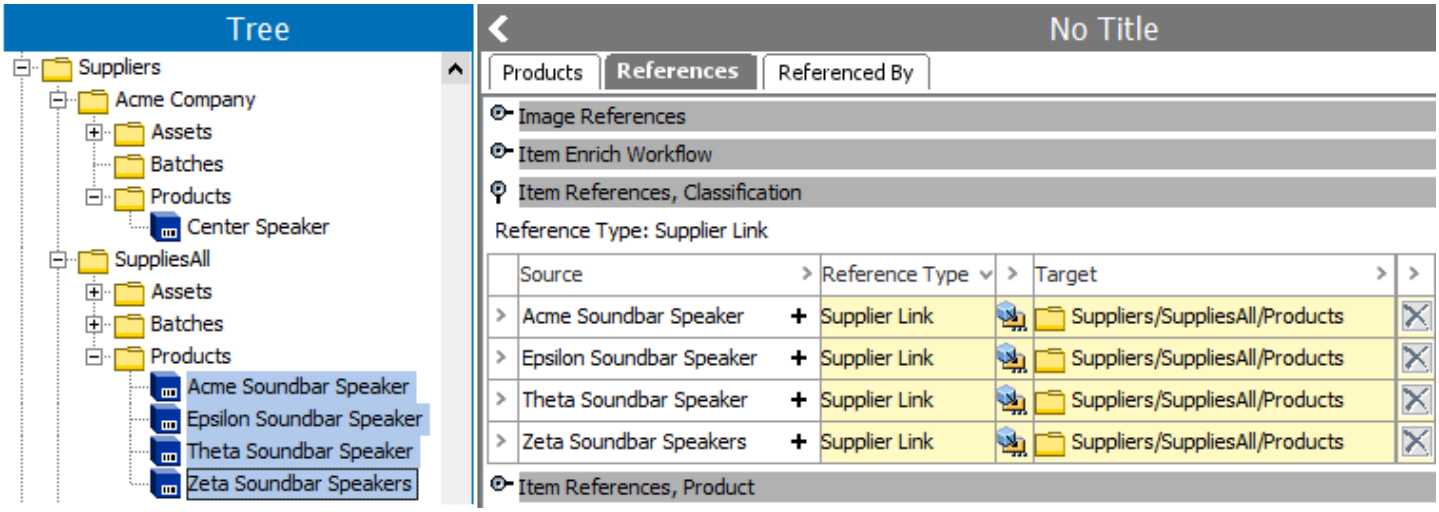
Online help includes a condensed version of the STEPXML export file generated by selecting the products. Set the 'Include Products' parameter to 'Selected' and set all other parameters to 'No' or 'None'.

Note: Sections of the XML file that do not impact this example have been removed for brevity.

Online help includes the export file that has been modified as follows:

- Add the ReplacementRules <ReplaceClassificationReferences> tag for the <Products> super type.
- Remove the <Classifications> tag, leaving only the <Products> tag.
- Change the classification ID from 'Acme_3Products' to 'SuppliesAllProducts' to link the products to a different location.

The result of importing the modified file is shown in the Tree.



ReplacementRules in STEPXML for User Groups Examples

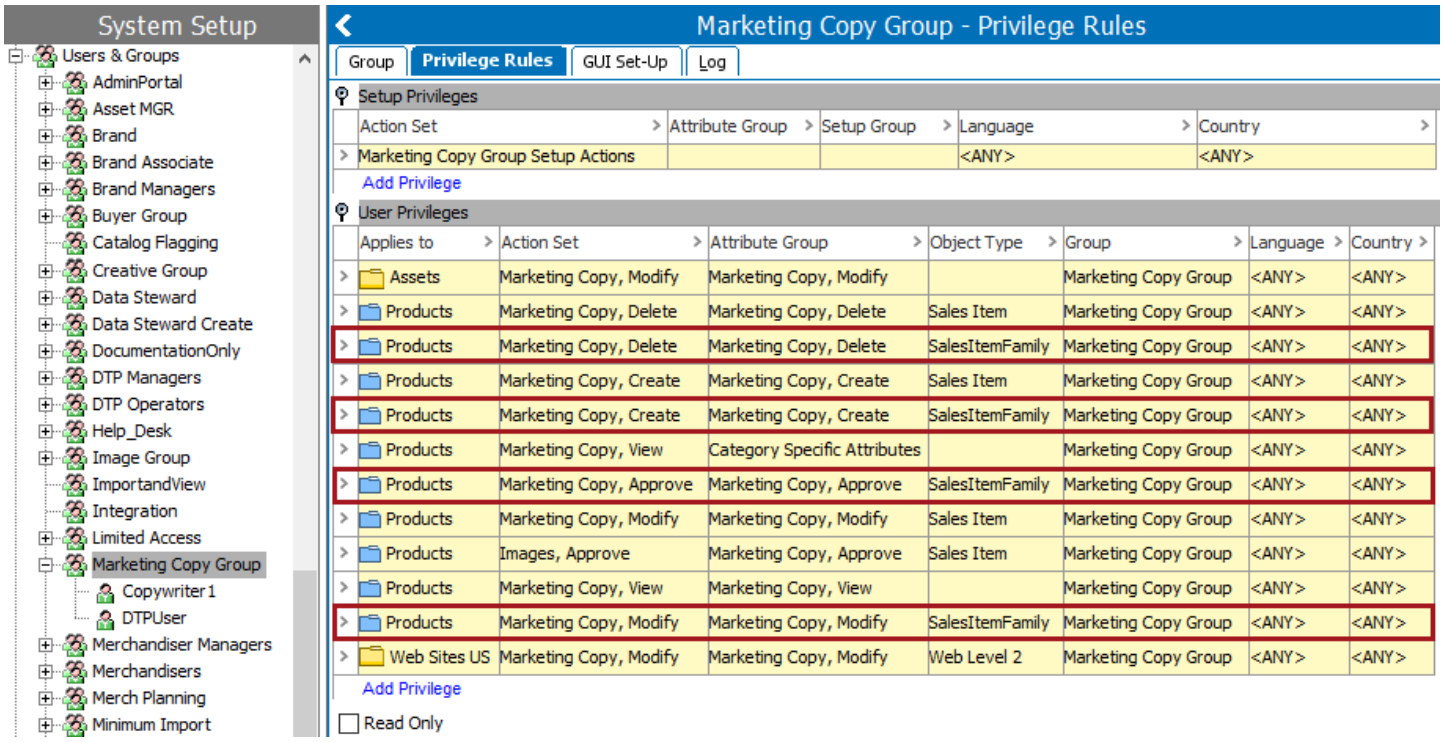
This example demonstrates using the ReplacementRules tag via inbound STEPXML to:

- Update privileges rules for a user group.

For more information, refer to the **ReplacementRules Tag in STEPXML** topic.

Update Privilege Rules for a User Group

The 'Marketing Copy Group' is assigned privileges for the 'SalesItemFamily' object type. Due to a change in the role, that privilege is no longer needed and will be replaced.

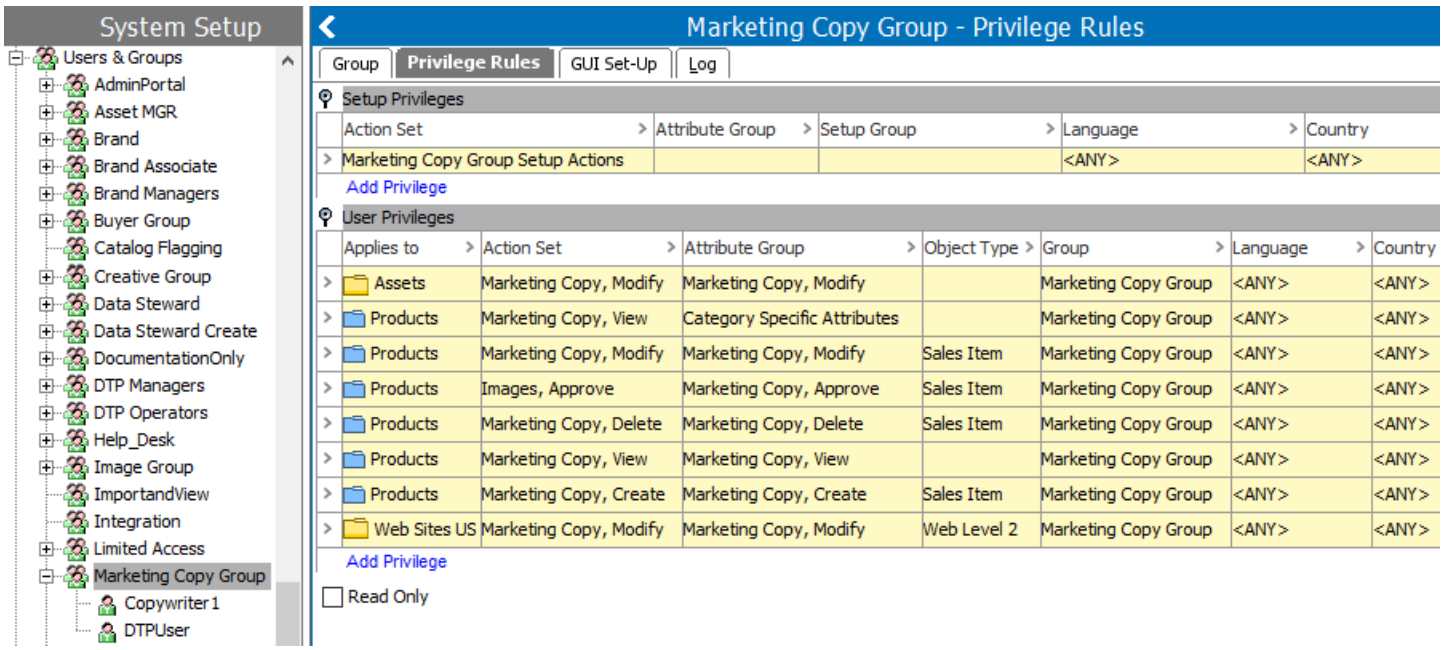


Online help includes the STEPXML export file generated using the Export parameter set to 'All' before selecting the user group. Set the 'Include Users and User Groups' parameter to 'Selected' and set all other parameters to 'No' or 'None'.

Online help includes the export file which has been modified as follows:

- Add the ReplacementRules <ReplacePrivilegeRules> tag for the <UserGroups> super type.
- Remove the <UserGroupList> tag data for 'SalesItemFamily'.

The result of importing the modified file is shown in the user group editor.


System Setup
Users & Groups
AdminPortal
Asset MGR
Brand
Brand Associate
Brand Managers
Buyer Group
Catalog Flagging
Creative Group
Data Steward
Data Steward Create
DocumentationOnly
DTP Managers
DTP Operators
Help_Desk
Image Group
ImportandView
Integration
Limited Access
Marketing Copy Group
Copywriter1
DTPUser
Marketing Copy Group - Privilege Rules
Group
Privilege Rules
GUI Set-Up
Log
Setup Privileges
Action Set
Attribute Group
Setup Group
Language
Country
Marketing Copy Group Setup Actions
<ANY>
<ANY>
Add Privilege
User Privileges
Applies to
Action Set
Attribute Group
Object Type
Group
Language
Country
Assets
Marketing Copy, Modify
Marketing Copy, Modify
Marketing Copy Group
<ANY>
<ANY>
Products
Marketing Copy, View
Category Specific Attributes
Marketing Copy Group
<ANY>
<ANY>
Products
Marketing Copy, Modify
Marketing Copy, Modify
Sales Item
Marketing Copy Group
<ANY>
<ANY>
Products
Images, Approve
Marketing Copy, Approve
Sales Item
Marketing Copy Group
<ANY>
<ANY>
Products
Marketing Copy, Delete
Marketing Copy, Delete
Sales Item
Marketing Copy Group
<ANY>
<ANY>
Products
Marketing Copy, View
Marketing Copy, View
Marketing Copy Group
<ANY>
<ANY>
Products
Marketing Copy, Create
Marketing Copy, Create
Sales Item
Marketing Copy Group
<ANY>
<ANY>
Web Sites US
Marketing Copy, Modify
Marketing Copy, Modify
Web Level 2
Marketing Copy Group
<ANY>
<ANY>
Add Privilege
Read Only

ReplacementRules in STEPXML for User Type Links Examples

'User type' is the term for an 'object type' in STEPXML. 'User Type Links' include attribute, link types, and object types.

For simplicity, examples use minimal data. Once an import file is verified for a small dataset, the use case can be expanded to include additional objects using search / replace as necessary within a larger import file.

These examples demonstrate using the ReplacementRules tag via inbound STEPXML to:

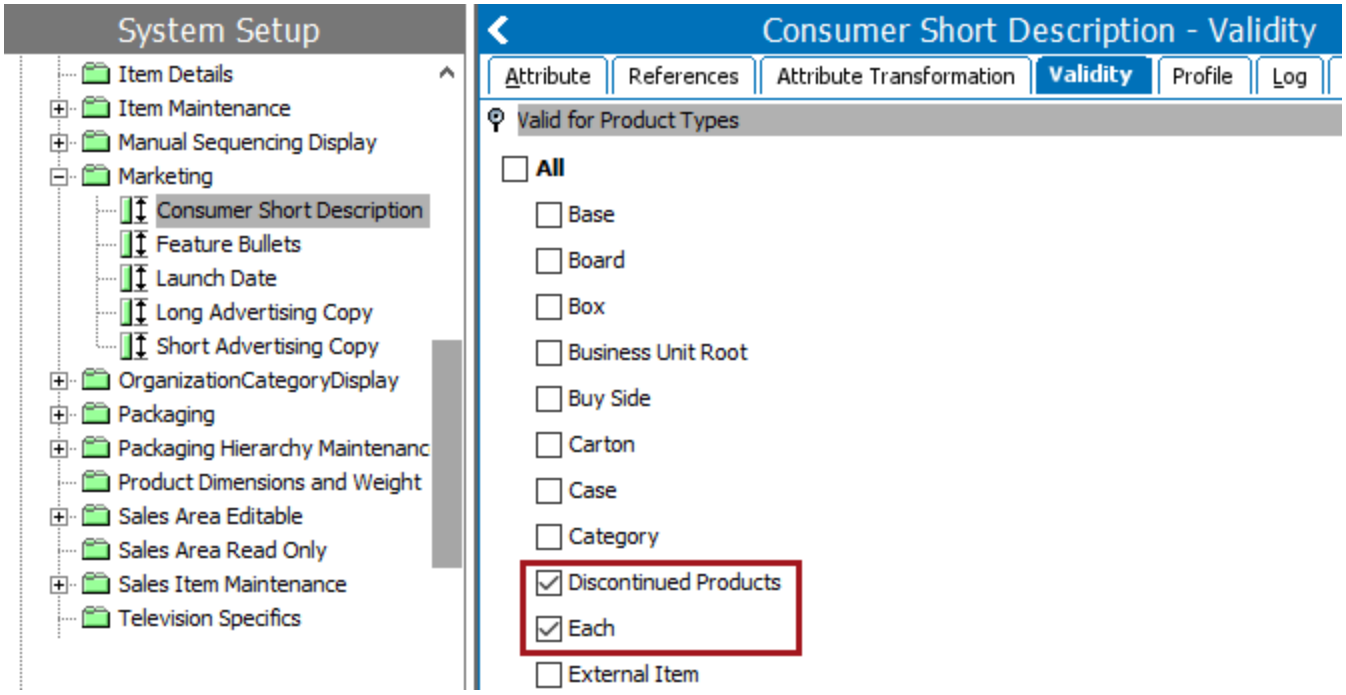
- Replace links on attribute object types
- Replace reference source and target for link types
- Replace links on object types

For more information, refer to the **ReplacementRules Tag in STEPXML** topic.

Replace Links on Attribute Object Types

The 'Customer Short Description' attribute is valid for multiple product object types.

Note: Only replacements in the import file for link types without values are processed successfully. The import fails for any replacement link type when the newest revision in any context or workspace has values. The background process has a status of 'Completed with Errors' and the execution report shows errors for the object types with values.

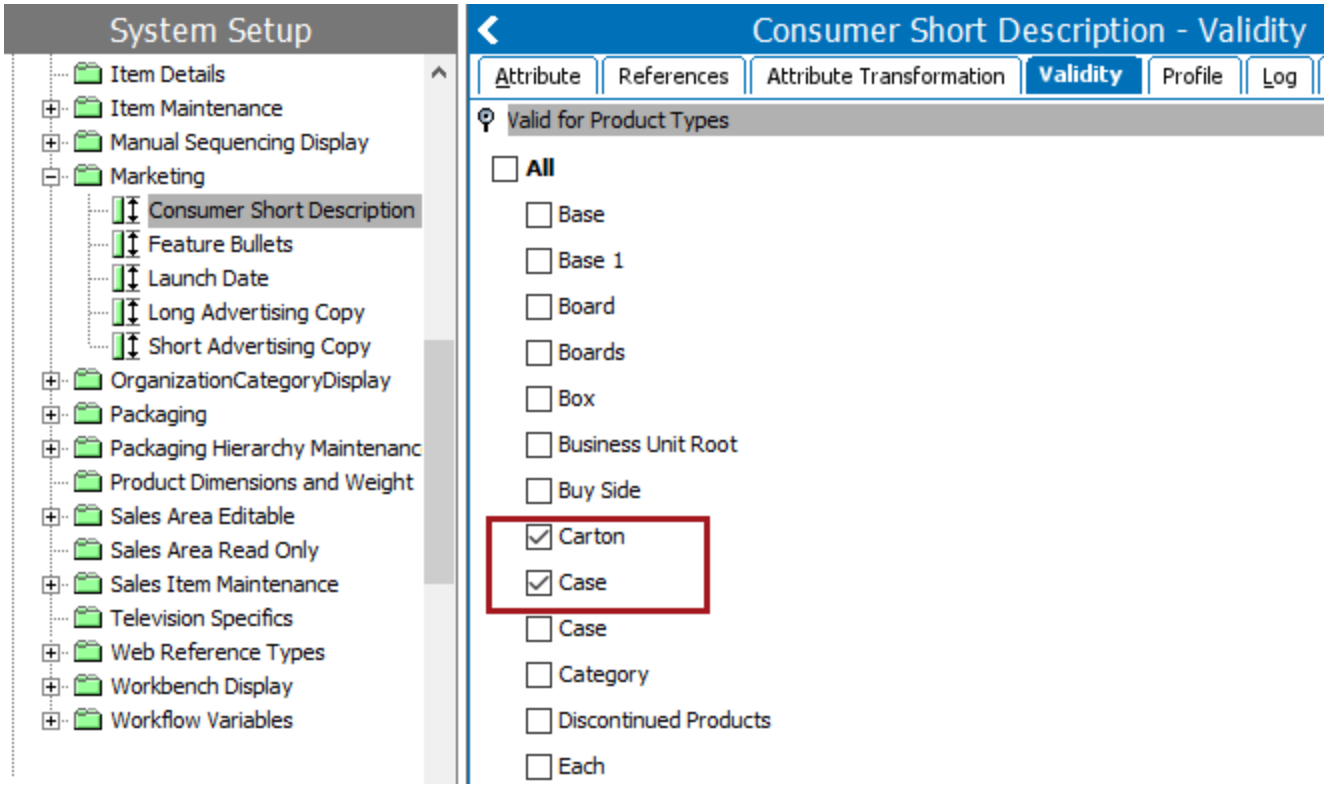


Online help includes the export file for the attribute as STEPXML generated by setting the 'Include Attributes' parameter to 'Selected', and setting all other parameters to 'No' or 'None'.

Online help includes the export file which has been modified as follows:

- Add the ReplacementRules <ReplaceUserTypeLinks> tag for the <Attributes> super type.
- Update user type link data within the <AttributeList> tag to be modified.

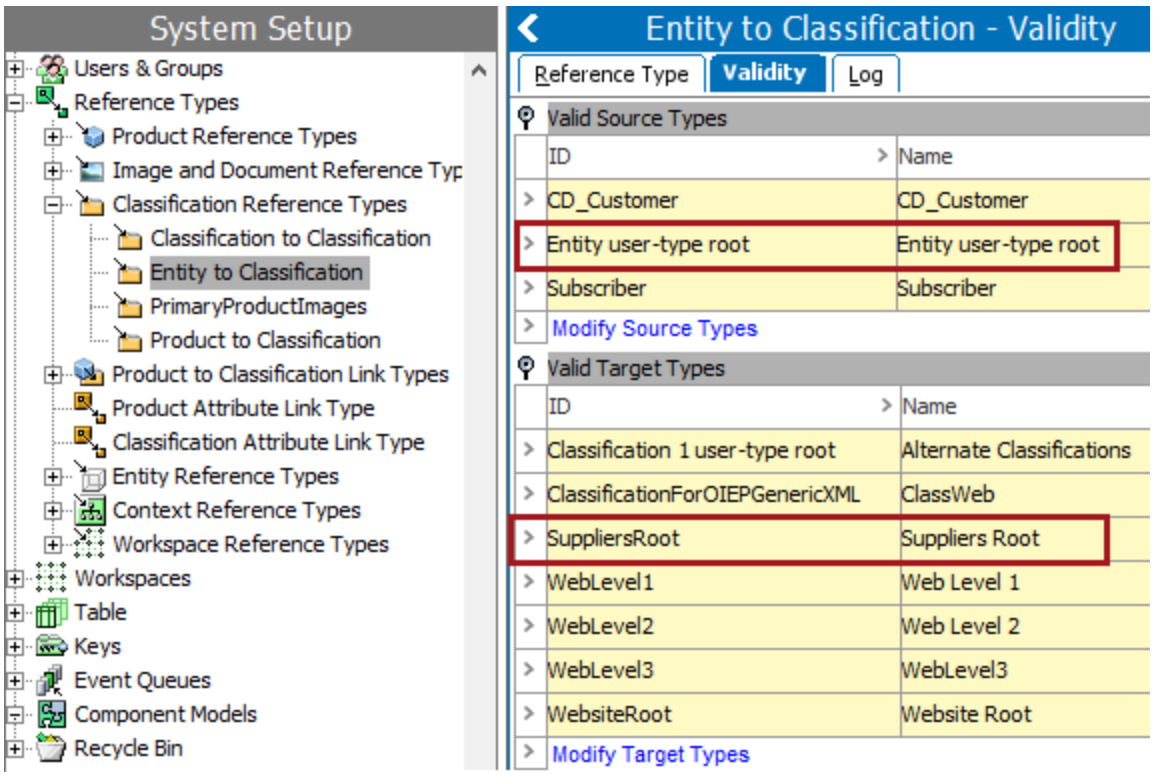
The result of importing the modified file is shown in the attribute editor:



Replace Reference Source and Target for Link Types

The 'Entity to Classification' reference is valid for multiple source and target object types.

Note: Only replacements in the import file for link types without values are processed successfully. The import fails for any replacement link type when the newest revision in any context or workspace has values. The background process has a status of 'Completed with Errors' and the execution report shows errors for the object types with values.



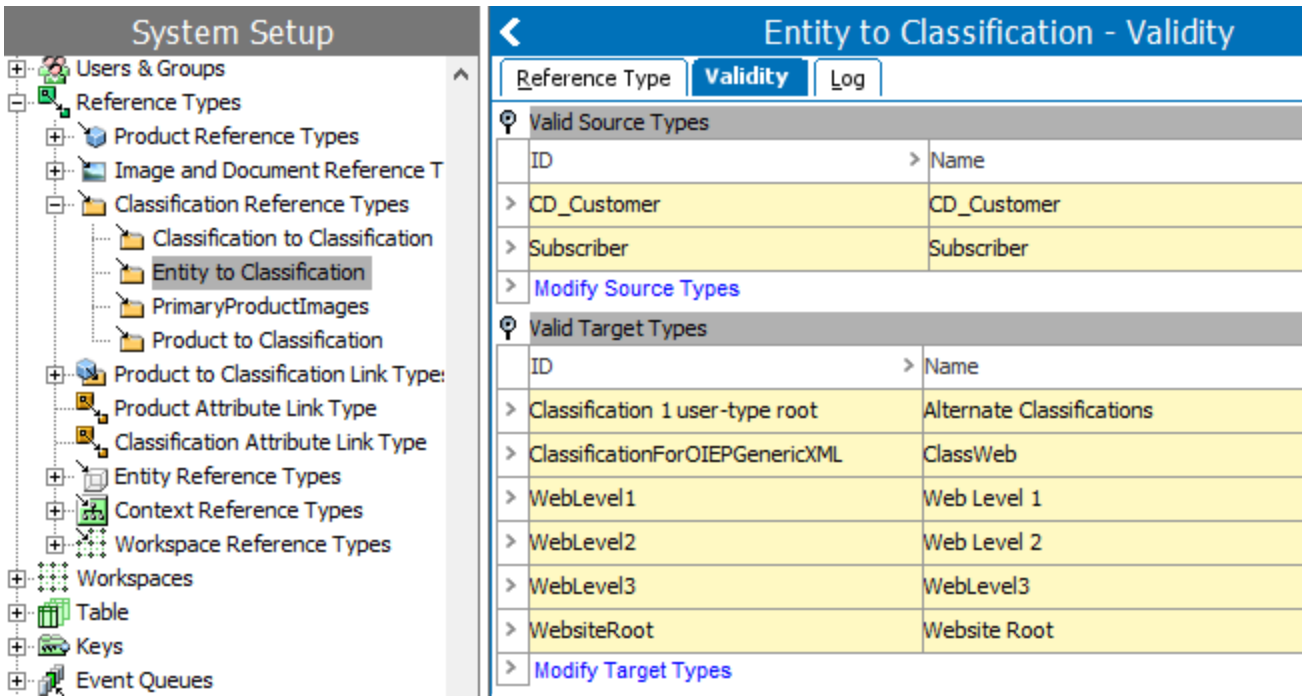
Online help includes a condensed version of export file for the attribute as STEPXML generated by using the Export parameter set to 'All' before selecting the desired reference(s) / link(s). Set the 'Include Link, Reference and Object Types' parameter to 'Selected' and set all other parameters to 'No' or 'None'.

Note: Sections of the XML file that do not impact this example have been removed for brevity.

Online help includes the export file that uses the <UserTypeLink> tag for source types and the <TargetUserTypeLink> tag for target types. The export file has been modified as follows:

- Add the ReplacementRules <ReplaceUserTypeLinks> tag and the <ReplaceTargetUserTypeLinks> tag for the <LinkTypes> super type.
- Remove the user type link 'Entity user-type root' as a source.
- Remove the user type link 'SuppliersRoot' as a target.

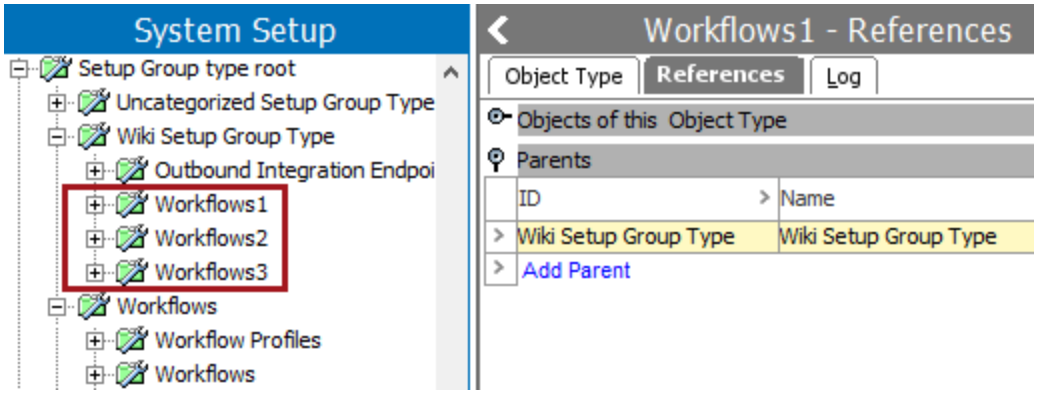
The result of importing the modified file is shown in the reference type editor. One source and one target have been removed.



Replace Links on Object Types

Within the 'Object Types & Structures' node, the 'Setup Group type root' node includes multiple object types for workflows. Some of the object types have an incorrect parent of 'Wiki Setup Group Type' as shown in the image below and as evidenced by the STEPXML export file. The parent 'Workflows' will replace 'Wiki Setup Group Type' on these object types: Workflows1, Workflows2, and Workflows3.

Note: Only replacements in the import file for link types without values are processed successfully. The import fails for any replacement link type when the newest revision in any context or workspace has values. The background process has a status of 'Completed with Errors' and the execution report shows errors for the object types with values.



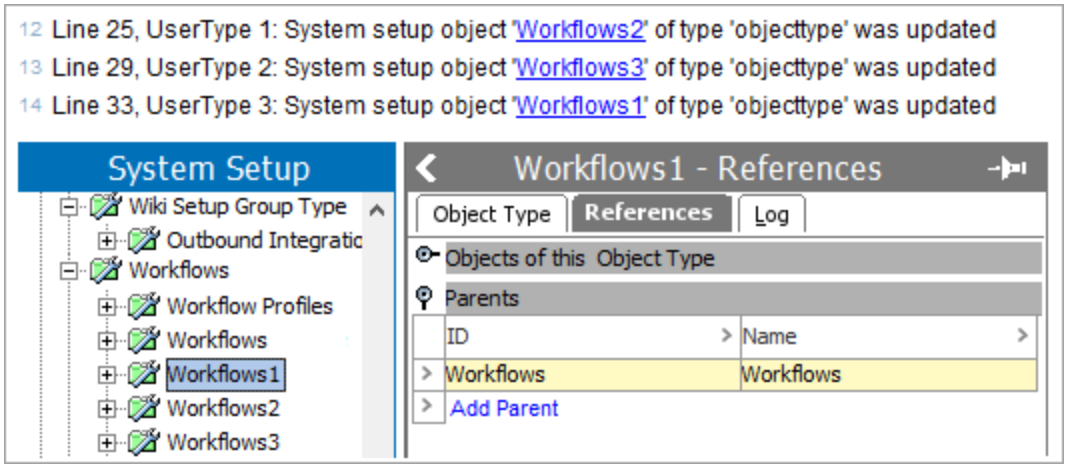
Online help includes a condensed version of export file as STEPXML generated by using the Export parameter set to 'All' before selecting the desired 'workflow' object types. Set the 'Include Link, Reference and Object Types' parameter to 'Selected' and set all other parameters to 'No' or 'None'.

Note: Sections of the XML file that do not impact this example have been removed for brevity.

Online help includes the export that has been modified as follows for import:

- Add the ReplacementRules <ReplaceUserTypeLinks> tag for the <UserTypes> super type.
- Replace the user type link 'Wiki Setup Group Type' with 'Workflows'.

Importing the modified object types result in them being children to the 'Workflows' nodes, as evidenced by the BGP Execution Report.



Referenced and Embedded XML Attributes in STEPXML

The ID and Type of an object's references can be exported using the **Referenced** XML attribute.

```
<Product Referenced="true">
```

In order to export details about the reference itself, use the **Embedded** XML attribute.

```
<Entity Referenced="true" Embedded="true">
```

Note: Embedded referenced objects cannot be imported. Any changes to the embedded object, like its name, will not update the data in STEP if loaded for import.

When required by a downstream system, the **Referenced** and **Embedded** XML attributes are used together to provide additional details about the reference in line with the product, classification, or asset that owns the reference.

For details on tags, elements, and their XML attributes, refer to the available XSD (XML Schema Definition) files in the **STEPXML Tags and Examples** topic.

For example, if STEP allows a product-to-product cross reference between products A and B, then the data for both products A and B can be delivered via the product reference. The cross referenced product details are exported within the product that references it.

Template

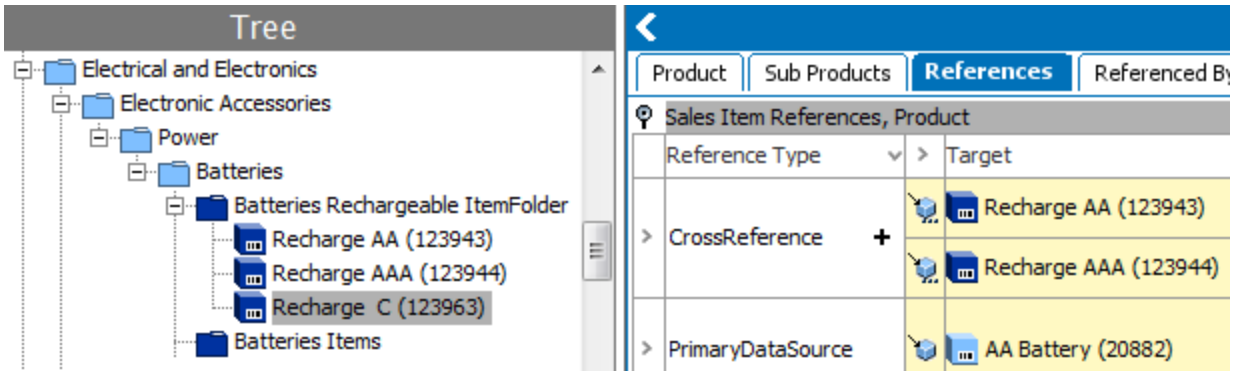
The MetaData tag is used to output attribute values maintained on the product reference itself.

Referenced Without Embedded	Referenced With Embedded
<?xml version="1.0" encoding="UTF-8"?> <STEP-ProductInformation ResolveInlineRefs="true"> <Products> <FilterUserType ID="Item"/> <Product IncludeParent="true"> <Name/> <ProductCrossReference Type="CrossReference"> <MetaData/> <Product Referenced="true"> <Name/> <Values/>	<?xml version="1.0" encoding="UTF-8"?> <STEP-ProductInformation ResolveInlineRefs="true"> <Products> <FilterUserType ID="Item"/> <Product IncludeParent="true"> <Name/> <ProductCrossReference Type="CrossReference"> <MetaData/> <Product Referenced="true" Embedded="true"> <Name/>

Referenced Without Embedded	Referenced With Embedded
</Product> </ProductCrossReference> </Product> </Products> </STEP-ProductInformation>	<Values/> </Product> </ProductCrossReference> </Product> </Products> </STEP-ProductInformation>

Modified Object

For each template, the same single item (Recharge C) is modified and approved, which triggers the OIEP.



Results

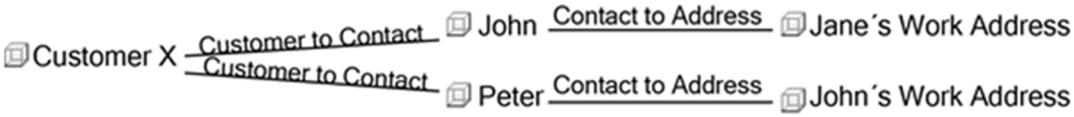
Referenced Without Embedded

```
<STEP-ProductInformation ExportTime="2015-12-21 15:14:13" ExportContext="Context1" ContextID="Co

  <Products>
    <Product ID="123944" UserTypeID="Item" ParentID="123942" Selected="false" Referenced="true">
      <Name>Recharge AAA</Name>
      <Values>
        <Value AttributeID="SupplierPartNumber">65165165</Value>
        <Value AttributeID="Path" Derived="true">Electrical and Electronics | Electronic Accesso
        <Value AttributeID="WWW1" Derived="true">http://www.stibosystems.com</Value>
      </Values>
    </Product>
    <Product ID="123943" UserTypeID="Item" ParentID="123942" Selected="false" Referenced="true">
      <Name>Recharge AA</Name>
      <Values>
        <Value AttributeID="SupplierPartNumber">32153563</Value>
        <Value AttributeID="Path" Derived="true">Electrical and Electronics | Electronic Accesso
        <Value AttributeID="WWW1" Derived="true">http://www.stibosystems.com</Value>
      </Values>
    </Product>
    <Product ID="123963" UserTypeID="Item" ParentID="123942">
      <Name Changed="true">Recharge C</Name>
        <ProductCrossReference ProductID="123943" Type="CrossReference"/>
        <ProductCrossReference ProductID="123944" Type="CrossReference"/>

    </Product>
  </Products>
</STEP-ProductInformation>
```

Referenced With Embedded

```
<STEP-ProductInformation ExportTime="2015-12-21 15:11:51" ExportContext="Context1" ContextID="Co

  <Products>
    <Product ID="123944" UserTypeID="Item" ParentID="123942" Selected="false" Referenced="true">
      <Name>Recharge AAA</Name>
      <Values>
        <Value AttributeID="SupplierPartNumber">65165165</Value>
        <Value AttributeID="Path" Derived="true">Electrical and Electronics | Electronic Accesso
        <Value AttributeID="WWW1" Derived="true">http://www.stibosystems.com</Value>
      </Values>
    </Product>
    <Product ID="123943" UserTypeID="Item" ParentID="123942" Selected="false" Referenced="true">
      <Name>Recharge AA</Name>
      <Values>
        <Value AttributeID="SupplierPartNumber">32153563</Value>
        <Value AttributeID="Path" Derived="true">Electrical and Electronics | Electronic Accesso
        <Value AttributeID="WWW1" Derived="true">http://www.stibosystems.com</Value>
      </Values>
    </Product>
    <Product ID="123963" UserTypeID="Item" ParentID="123942">
      <Name Changed="true">Recharge  C</Name>
        <ProductCrossReference ProductID="123943" Type="CrossReference">
<Product ID="123943" UserTypeID="Item" ParentID="123942">
      <Name>Recharge AA</Name>
      <Values>
        <Value AttributeID="SupplierPartNumber">32153563</Value>
        <Value AttributeID="Path" Derived="true">Electrical and Electronics | Electronic Accesso
        <Value AttributeID="WWW1" Derived="true">http://www.stibosystems.com</Value>
      </Values>
    </Product></ProductCrossReference>
        <ProductCrossReference ProductID="123944" Type="CrossReference">
<Product ID="123944" UserTypeID="Item" ParentID="123942">
      <Name>Recharge AAA</Name>
      <Values>
        <Value AttributeID="SupplierPartNumber">65165165</Value>
        <Value AttributeID="Path" Derived="true">Electrical and Electronics | Electronic Accesso
        <Value AttributeID="WWW1" Derived="true">http://www.stibosystems.com</Value>
      </Values>
    </Product></ProductCrossReference>

    </Product>
  </Products>
</STEP-ProductInformation>
```

As another example, the following Advanced STEPXML template is defined to export a customer entity hierarchy. Each customer has references to contacts, and each contact has one or more references to addresses. All objects are in this example are modeled with entity objects.



The following is a general Advanced STEPXML template that exports all used entity objects below the customer hierarchy.

```
<?xml version='1.0'?>
<STEP-ProductInformation>
<Entities>
   <Entity>
      <EntityCrossReference Type="Customer to Contact">
         <Entity Referenced="true">
            <Name />
            <EntityCrossReference Type="Contact to Address">
               <Entity Referenced="true">
                  <Name />
               </Entity>
            </EntityCrossReference>
        </Entity>
      </EntityCrossReference>
   </Entity>
</Entities>
</STEP-ProductInformation>
```

The template exports customer entity objects and their references of the type **Customer to Contact** and **Contact to Address**. However, the referenced objects are not exported as embedded in the customer hierarchy. To export the referenced contacts and their address objects embedded in the customer hierarchy, you have to insert <Entity Referenced="true" Embedded="true"> in the advanced template as illustrated in the following.

```
<?xml version='1.0'?>
<STEP-ProductInformation>
<Entities>
   <Entity>
      <EntityCrossReference Type="Customer to Contact">
         <Entity Referenced="true" Embedded="true">
            <Name />
            <EntityCrossReference Type="Contact to Address">
               <Entity Referenced="true" Embedded="true">
                  <Name />
               </Entity>
            </EntityCrossReference>
         </Entity>
      </EntityCrossReference>
   </Entity>
</Entities>
</STEP-ProductInformation>
```

SequenceProduct Tag in STEPXML

The sequence number is the order of the products in STEP. When used with the **IncludeParent** tag, the **SequenceProduct** tag exports the sequence numbers of all exported products.

```
<SequenceProduct/>
```

For details on tags, elements, and their XML attributes, refer to the available XSD (XML Schema Definition) files in the **STEPXML Tags and Examples** topic.

Prerequisites

This option is only effective for objects that have the **Manually Sorted** parameter activated as follows:

- In System Setup on an attribute group, when the Manually Sorted parameter is checked, instances of the attributes in the group are output in the order that is displayed in System Setup.
- In System Setup, when a product object type or classification object type has the Manually Sorted parameter set to Yes, the export uses the order of the instances of the objects as they are displayed in the Tree.

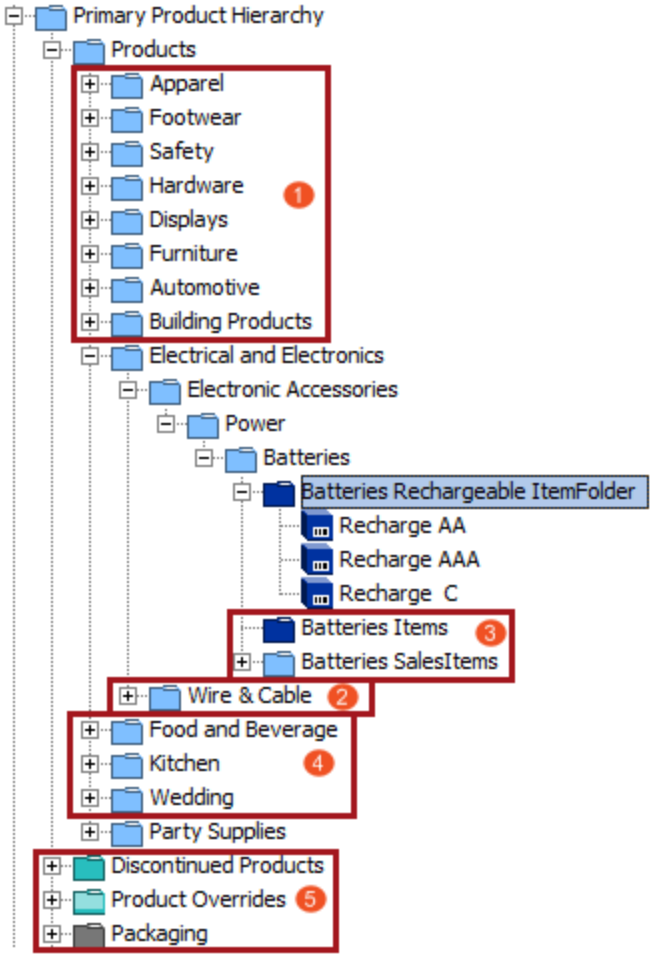
Note: When the Manually Sorted option is not activated, objects are output in what may appear to be a random order since they are not ordered by ID or name.

Template

With SequenceProduct	Without SequenceProduct
<?xml version="1.0" encoding="UTF-8"?> <STEP-ProductInformation> <Products> <Product IncludeParent="true"> <SequenceProduct/> <Name/> </Product> </Products> </STEP-ProductInformation>	<?xml version="1.0" encoding="UTF-8"?> <STEP-ProductInformation> <Products> <Product IncludeParent="true"> <Name/> </Product> </Products> </STEP-ProductInformation>

Modified Object

For each template, the same single item (Batteries Rechargeable ItemFolder) is modified and approved, which triggers the OIEP.



Results

All parents for the exported product are output.

Sequence numbers for all nodes under each of the parents are output.

Sequence numbers for nodes equal to the changed object are output.

With SequenceProduct

```
<STEP-ProductInformation ExportTime="2016-01-04 10:04:58" ExportContext="Context1" ContextID="

  <Products>
    <Product ID="Product hierarchy root" UserTypeID="Product user-type root" Selected="false">
      <Name>Primary Product Hierarchy</Name>
      <Product ID="ProductsRoot" UserTypeID="Products" Selected="false">
        <Name>Products</Name>
        <SequenceProduct ID="18200"/>
        <SequenceProduct ID="20688"/>
        <SequenceProduct ID="20718"/>
        <SequenceProduct ID="20864"/>
        <SequenceProduct ID="22005"/>  1
        <SequenceProduct ID="22150"/>
        <SequenceProduct ID="8298"/>
        <SequenceProduct ID="8299"/>
        <Product ID="8302" UserTypeID="Level1" Selected="false">
          <Name>Electrical and Electronics</Name>
          <SequenceProduct ID="8313"/>  2
          <Product ID="20875" UserTypeID="Level2" Selected="false">
            <Name>Electronic Accessories</Name>
            <Product ID="20876" UserTypeID="Level3" Selected="false">
              <Name>Power</Name>
              <Product ID="20878" UserTypeID="Level4" Selected="false">
                <Name>Batteries</Name>
                <SequenceProduct ID="20879"/>
                <SequenceProduct ID="20880"/>  3
                <Product ID="123942" UserTypeID="ItemFolder">
                  <Name Changed="true">Batteries Rechargeable  ItemFolder</Name>
                </Product>
              </Product>
            </Product>
          </Product>
        </Product>
        <SequenceProduct ID="8303"/>
        <SequenceProduct ID="8304"/>  4
        <SequenceProduct ID="109946"/>
      </Product>
      <SequenceProduct ID="DiscontinuedProductsRoot"/>
      <SequenceProduct ID="ProductOverridesRoot"/>  5
      <SequenceProduct ID="PackagingRoot"/>
    </Product>
  </Products>
</STEP-ProductInformation>
```

All parents for the exported product are output.

Without SequenceProduct

```
<STEP-ProductInformation ExportTime="2016-01-04 10:13:31" ExportContext="Context1" ContextID="(

  <Products>
    <Product ID="Product hierarchy root" UserTypeID="Product user-type root" Selected="false">
      <Name>Primary Product Hierarchy</Name>
      <Product ID="ProductsRoot" UserTypeID="Products" Selected="false">
        <Name>Products</Name>
        <Product ID="8302" UserTypeID="Level1" Selected="false">
          <Name>Electrical and Electronics</Name>
          <Product ID="20875" UserTypeID="Level2" Selected="false">
            <Name>Electronic Accessories</Name>
            <Product ID="20876" UserTypeID="Level3" Selected="false">
              <Name>Power</Name>
              <Product ID="20878" UserTypeID="Level4" Selected="false">
                <Name>Batteries</Name>
                <Product ID="123942" UserTypeID="ItemFolder">
                  <Name Changed="true">Batteries Rechargeable ItemFolder</Name>
                </Product>
              </Product>
            </Product>
          </Product>
        </Product>
      </Product>
    </Product>
  </Products>
</STEP-ProductInformation>
```

SingleUpdateMode in STEPXML

Importing certain changes requires that the STEP system enters single-update mode. This ensures that during import, only the import process is allowed to modify data in the database. When the import process enters single-update mode, no user can write any data; although viewing data is still allowed.

For more information, refer to **Single-Update Mode**.

For details on tags, elements, and their XML attributes, refer to the available XSD (XML Schema Definition) files in the **STEPXML Tags and Examples** topic.

The following import changes require single-update mode:

- Adding a dimension to attributes, object types, reference types, units, LOVs, etc.
- Changing attributes from being internally maintained to being externally maintained
- Changing validation base type settings on attributes or term object types.

Note: If the import process cannot enter single-update mode because users are working in STEP, the import process goes into a wait state until single-update mode is possible.

Insert **SingleUpdateMode="Y"** in the **STEP-ProductInformation** tag to cause the import process to enter single-update mode, as illustrated in the following example:

```
<STEP-ProductInformation ContextID="Context1" WorkspaceID="Main"
SingleUpdateMode="Y">
```

For more information, refer to **STEP-ProductInformation Tag in STEPXML**.

STEP-ProductInformation Tag in STEPXML

The **STEP-ProductInformation** tag determines the data exported for Products. It allows multiple XML attributes (described below) to be listed on a single line, each separated by a single space. XML attributes that are absent use the default setting.

```
<STEP-ProductInformation ExportDeletedData="true" ResolveInlineRefs="true"
FollowOverrideSubProducts="true" ExportDerivedAttrs="true" Validation="XSD"
DefinitionsAsComments="true" EventsAsComments="false">
```

Important: The true and false settings for all XML attributes are case-sensitive and must be lower case as shown in the above example.

For details on tags, elements, and their XML attributes, refer to the available XSD (XML Schema Definition) files in the **STEPXML Tags and Examples** topic.

DefinitionsAsComments

- false - default setting
- true - configuration definitions for Business Rules, Web UIs, IEPs, or Workflows are exported as comments in the output

For more details, refer to the **Export Configuration Definitions as Comments** section of the **Configuration Management** documentation.

EventsAsComments

- true - default setting
- false - events are not exported as comments in the output

The following image shows the comments section of a STEPXML export that included events:

ExportDeletedData

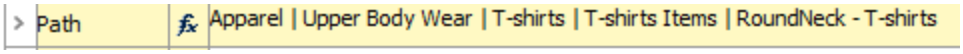
- false - default setting
- true - change markers indicate if data has been added to or removed from a product. This includes attribute values, product references and classification references.

For more information, refer to the **Event-Based OIEP Triggered by Deleting Products, Classifications, and Assets** section of the **Delete Objects in STEPXML** topic.

ExportDerivedAttrs

- false - default setting
- true - calculated attribute values are derived when an object is output from STEP

With the ExportDerivedAttrs XML attribute set to true in the Advanced STEPXML Template parameter, the following calculated attribute is exported:

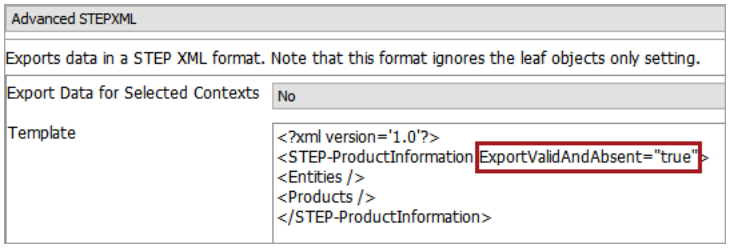


An excerpt from the output generated includes this data:

```
106 <Value AttributeID="Path" Derived="true">Apparel | Upper Body Wear | T-shirts | T-shirts Items | RoundNeck - T-shirts</Value>
```

ExportValidAndAbsent

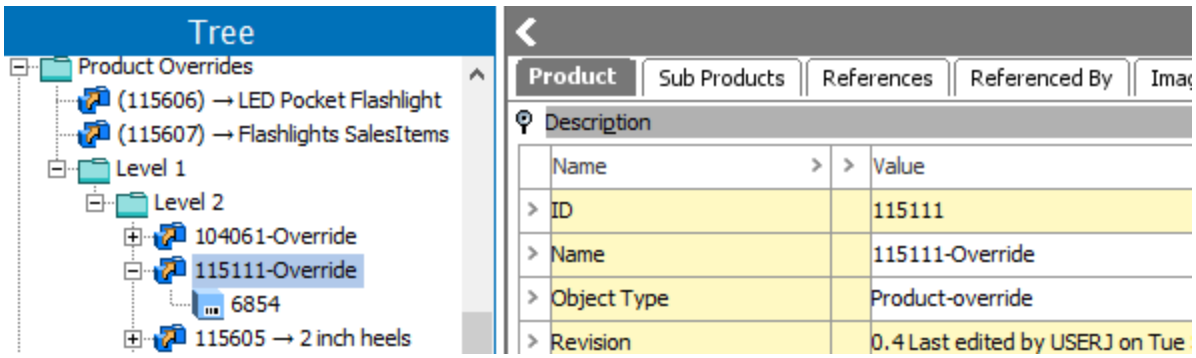
- true - empty attribute values and/or data containers and/or references or links (including metadata) are exported, including all valid and linked data, whether the exported object has a value or not
- false - default setting; empty values are not exported



FollowOverrideSubProducts

- false - default setting
- true - all Sub-Products of Product Overrides are included in the output XML

With the FollowOverrideSubProducts XML attribute set to true in the Advanced STEPXML Template parameter, the following product override is exported:



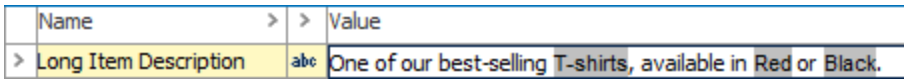
An excerpt from the output generated includes this data:

```
...
<Product ID="115111" UserTypeID="std.AdaptorType" OverridesProductID="" ParentID="101608">
  <Name>115111-Override</Name>
  <Values>
  </Values>
  <OverrideSubProduct ProductID="6854"/>
  ...
```

ResolveInlineRefs

- false - default setting
- true - all inline references are resolved within the attributes values that are exported

With the ResolveInlineRefs XML attribute set to true in the Advanced STEPXML Template parameter, the following attribute with inline references is exported:



An excerpt from the output generated includes this data:

```
<Values>
  ...
  <Value AttributeID="LongItemDescription">One of our best-selling T-shirts, available in Red or Black.</Value>
 ...
</Values>
```

STEPXML Configuration Export Format

The STEPXML Configuration Export format plugin is a utility feature for easily creating a cross-context STEPXML file that contains the full STEP configuration (minus user objects) and the data nodes that are required for being able to import the configuration on to an empty STEP system.

Format Availability

STEPXML Configuration Export is available for selection in:

- Export Manager - refer to **Creating a Data Export**
- OIEP - refer to **Creating an Outbound Integration Endpoint**

Mapping

Since STEPXML is the native format for STEP, mapping data is not required and the Map Data step / Mapping tab is disabled. Additional configuration not related to mapping should be completed as desired.

Outbound Data

Behind the scenes, the plugin makes use of the STEPXML output template (included later into this topic).

The template allows the following configuration objects to be exported:

- All contexts, dimensions, and dimension points
- All object types
- All edge, classification product links and reference types
- All units
- All lists of values and list of values groups
- All attributes and attribute groups
- All data container types
- All attribute transformations and attribute transformation groups
- All keys
- All derived event types
- All collections (definitions) and collection groups
- All setup groups
- All setup entities

- All business rules and business libraries
- All workflows, status flags, and workflow profiles
- All match codes and matching algorithms
- All action sets
- All user groups
- All tag groups and tags
- All import configurations
- All export configurations
- All translation configurations
- All bulk update configurations
- All transformation lookup tables
- All image conversion configurations
- All asset push and non-asset push event queues
- All eCatalogs
- All Web UI configurations
- All integration endpoints
- All event processors
- All table types, table type groups, table rules, and table colors
- All component models
- Global settings and system setup
- Selected products, classifications, and entities (name only) and their ancestors up to and including the supertype specific roots
- Selected assets, and if using the domain exporter, their “parent” classifications including ancestors up to and including the classification hierarchy root

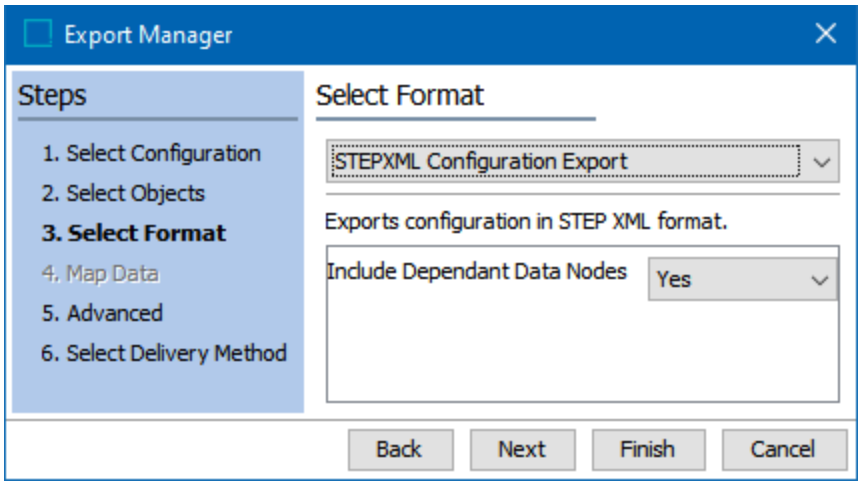
The template could be used with the Advanced STEPXML format plugin, which would allow for the same configuration objects to be exported. What the STEPXML Configuration Export adds is the ability to get data node dependencies included in the exported file. Dependencies that are not present in the target system or included in the STEPXML file will cause errors when importing the file.

Specifically, the plugin will add the following objects to the selection passed to the exporter:

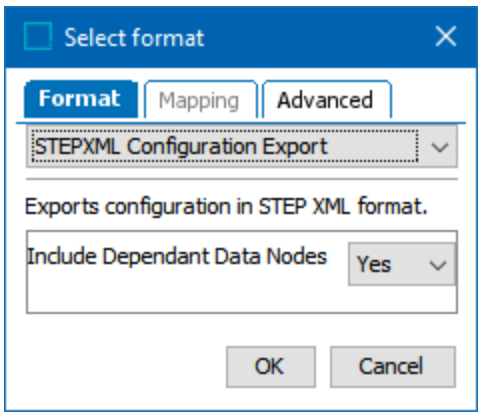
- All classifications, entities, products, and assets referenced from user group privilege rules
- All classifications referenced from supplier user groups
- All products and entities referenced from match codes
- For the following configurations types that live in the Tree, their parent classifications:
  - Export configurations
  - Import configurations
  - Bulk update configurations
  - Transformation lookup tables
  - Image conversion configurations
  - Translation configurations

Note: Whether or not to have these data node dependents included is determined via the 'Include Dependent Data Nodes' parameter for the 'Select Format' step when using the Export Manager or configuring an outbound integration endpoint (OIEP).

Export Manager example:



OIEP example:



The plugin will produce a cross-context STEPXML file containing data from all system contexts.

Output Template

```
<?xml version="1.0" encoding="utf-8"?>
<STEP-ProductInformation>
  <TagGroupList/>
  <TagList/>
  <ContextList/>
  <GlobalSettings/>
  <UserTypes ExportSize="All"/>
  <SystemSetup ExportSize="All"/>
  <Keys/>
  <DerivedEventTypes/>
  <EdgeTypes/>
  <CrossReferenceTypes ExportSize="All"/>
  <DimensionList/>
  <UnitList ExportSize="All"/>
  <CollectionList ExportSize="All"/>
  <ListOfValuesGroupList/>
  <ListsOfValues ExportSize="All"/>
  <AttributeGroupList ExportSize="All"/>
  <AttributeList ExportSize="All"/>
  <AttributeTransformationGroups/>
  <SetupGroups ExportSize="All"/>
  <SetupEntities ExportSize="All"/>
  <BusinessLibraries ExportSize="All"/>
```

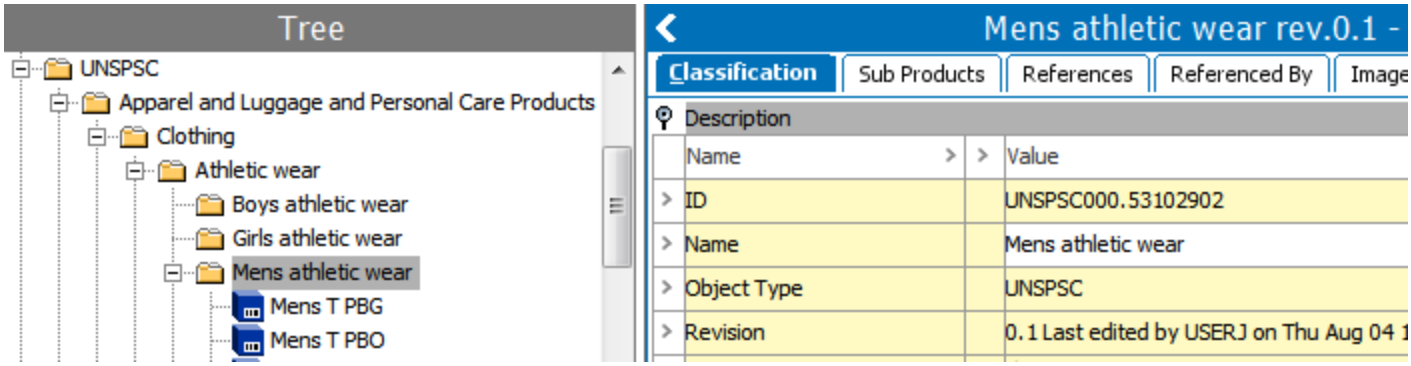
```
<BusinessRules ExportSize="All"/>
<STEPWorkflows ExportSize="All"/>
<STEPWorkflowProfiles ExportSize="All"/>
<MatchCodes/>
<MatchingAlgorithms ExportSize="All"/>
<Classifications ExportSize="Selected">
  <Classification IncludeParent="true">
    <Name/>
  </Classification>
</Classifications>
<Assets ExportSize="Selected" DontFollowClassificationToAssetReference="true">
  <Asset IncludeParentClassifications="true">
    <Name/>
    <ClassificationReference/>
  </Asset>
</Assets>
<Entities ExportSize="Selected">
  <Entity IncludeParent="true">
    <Name/>
  </Entity>
</Entities>
<DataContainerTypes ExportSize="All"/>
<Products ExportSize="Selected" DontFollowClassificationToProductReference="true">
  <Product IncludeParent="true">
    <Name/>
  </Product>
</Products>
<ActionSetList/>
<UserGroupList ExportSize="All"/>
<ImportConfigurations ExportSize="All"/>
<ExportConfigurations ExportSize="All"/>
<TranslationConfigurations ExportSize="All"/>
<BulkUpdateConfigurations ExportSize="All"/>
<TransformationLookupTableConfigurations ExportSize="All"/>
<ImageConversionConfigurations ExportSize="All"/>
<AssetPushEventQueues ExportSize="All"/>
<ECatalogs/>
```

```
  <PortalConfigurations ExportSize="All"/>
  <EventQueues/>
  <IntegrationEndpoints ExportSize="All"/>
  <EventProcessors ExportSize="All"/>
  <StatusFlags ExportSize="All"/>
  <TableColors ExportSize="All"/>
  <TableRules ExportSize="All"/>
  <TableTypeGroupList ExportSize="All"/>
  <TableTypeDefinitions ExportSize="All"/>
  <ComponentModels/>
</STEP-ProductInformation>
```

UNSPSC Format

United Nations Standard Products and Services Code® (UNSPSC) is a global classification framework for products and services in all industry sectors. Importing a UNSPSC file into STEP creates a set of multi-level classification folders, depending on the contents of the import file. For detailed information about UNSPSC, search the web.

When a UNSPSC classification exists in STEP, you can make product / service identification easier by using Bulk Update to add a 'product-to-classification link type' reference between your products / services and the UNSPSC classification system. This displays your products / services within the UNSPSC folder as shown in the image below. For more information, refer to the **Bulk Updates** documentation.



Format Availability

UNSPSC is available for selection in:

- IIEP - refer to **Creating an Inbound Integration Endpoint**
- Import Manager - refer to **Creating a Data Import**

Mapping

This format does not require a data map.

Inbound Data

The UNSPSC file to be imported into STEP can be retrieved from the member's area of the UNSPSC website. The Download Type selection should be 'Prepackaged complete download.'

All data included in the file selected for import will be imported; selection of particular classifications is not available.

The content and required preparation of a UNSPSC file depends on the UNSPSC version, as described below:

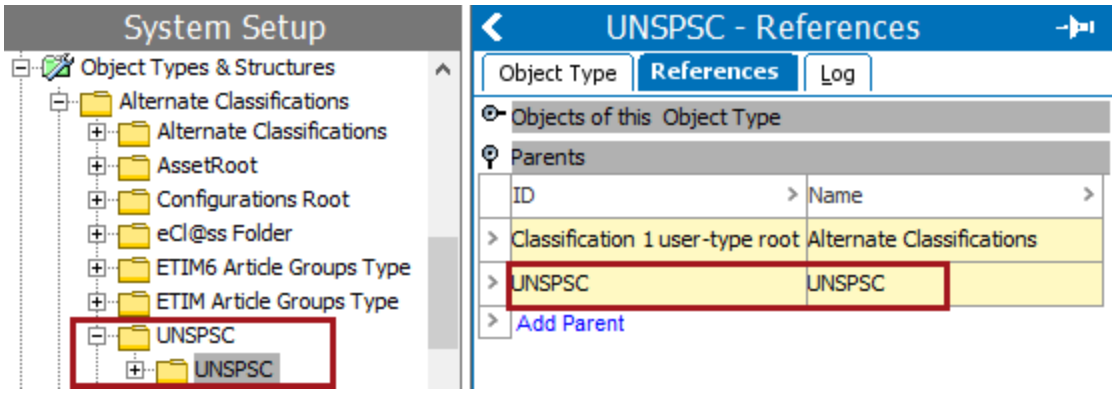
Version	File Types Included in .ZIP	File Prep Required	File to Import
13	One (1) Excel file with six (6) tabs	Extract before import	Excel file
17, 18, 19, 20, 21, 22, 23, 24, 25 and 26	Six (6) individual .txt files	None	.ZIP file

Multiple versions of UNSPSC classifications can exist simultaneously in STEP, provided different UNSPSC classification ID prefixes are used.

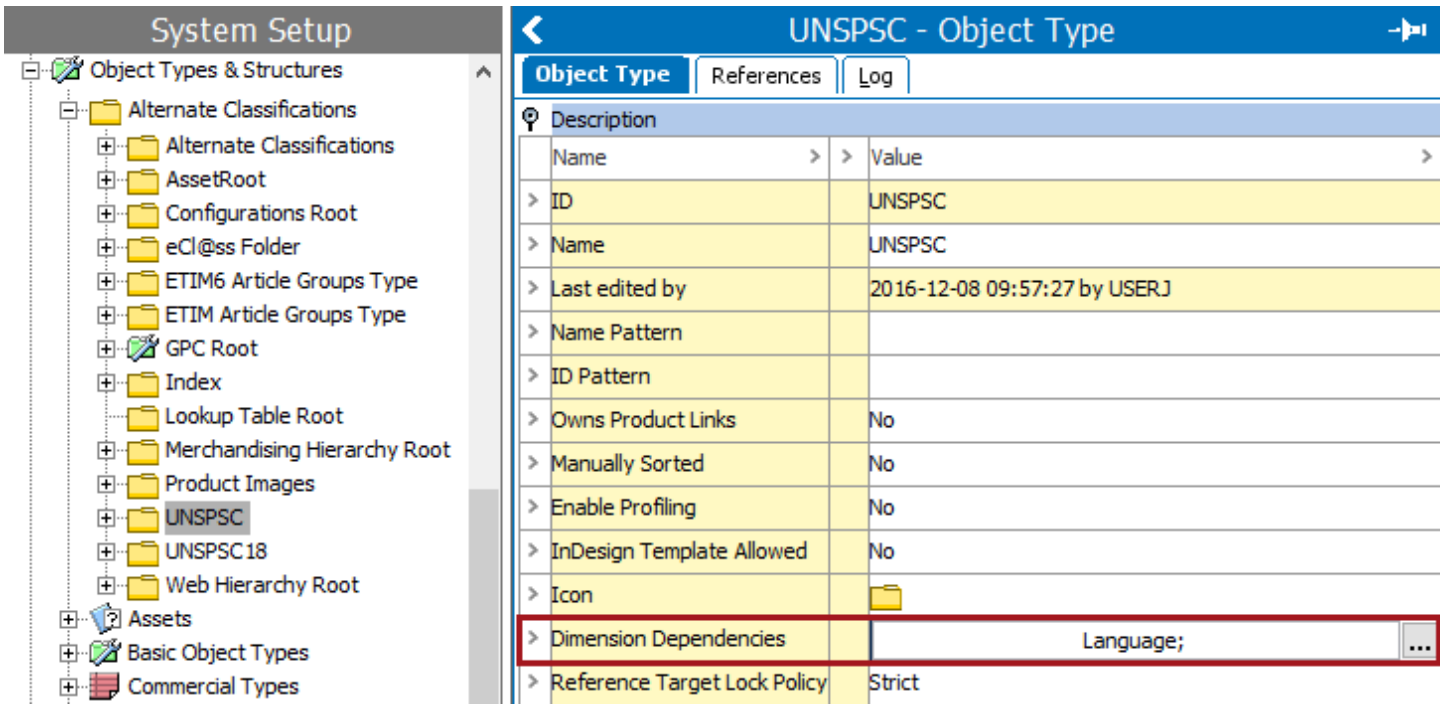
Prerequisites

To accommodate importing multiple levels within a single classification, the UNSPSC classification object type must exist and have itself as a parent, as defined in the following steps.

1. In System Setup, create the UNSPSC alternate classification object type.
2. On the alternate classification editor Reference tab, click the **Add Parent** link and select the UNSPSC object. The hierarchy becomes an endless chain where UNSPSC is repeated for every level.



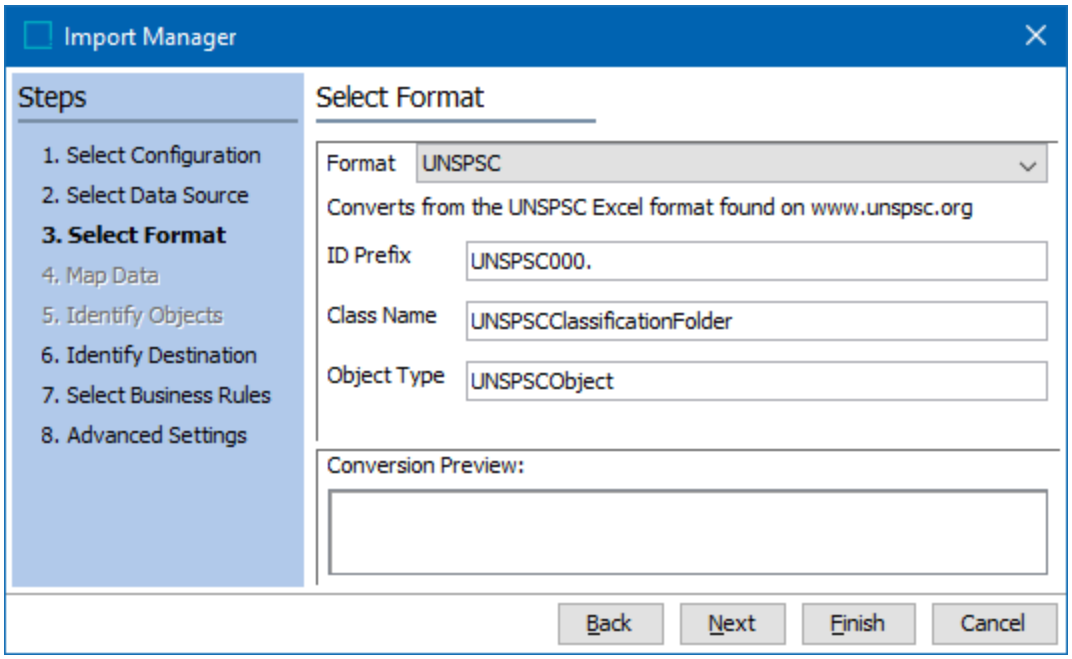
To import multiple UNSPSC languages for different contexts, on the top-level UNSPSC folder, set the Dimension Dependencies parameter to Language.



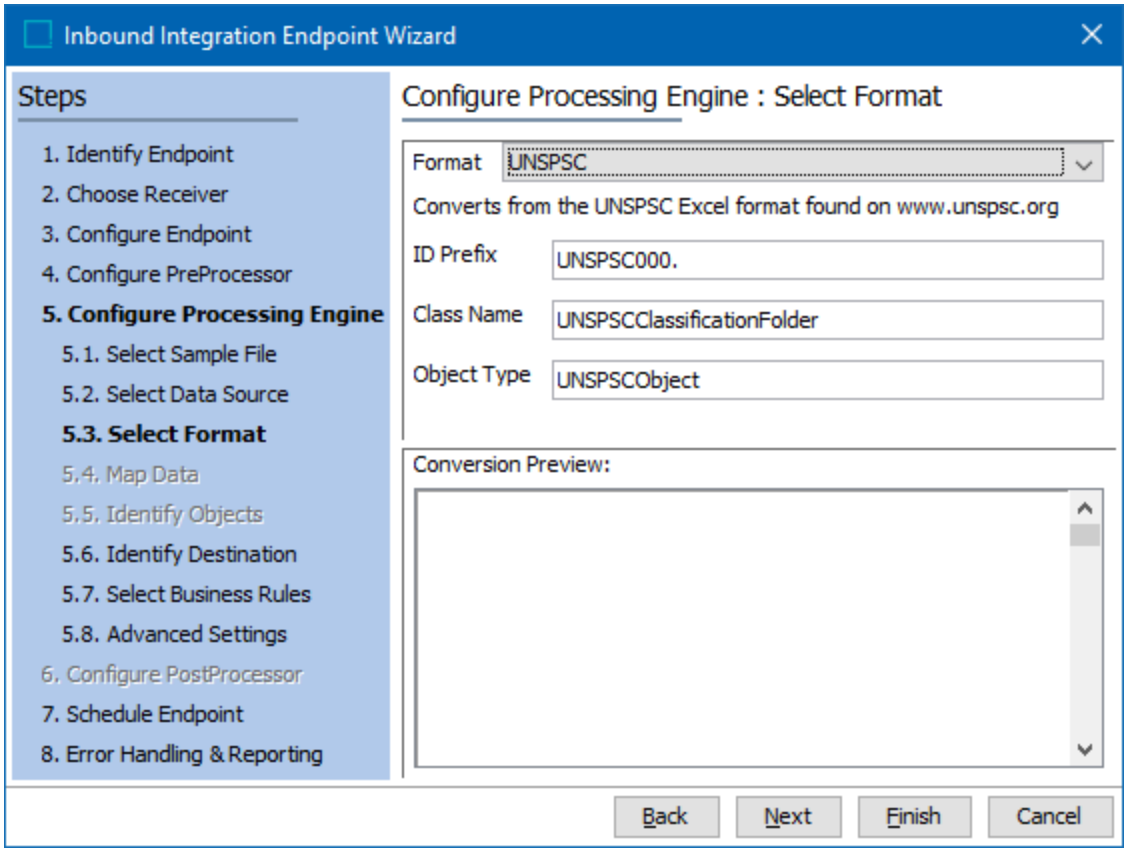
Inbound Parameters

- **ID prefix** is the text you want to appear preceding the classification 'code' from the UNSPSC file. The 'code' is imported as a STEP classification ID (refer to the image above). 'UNSPSC000.' is the default text prefixed to the ID.
- **Class Name** field, enter a name for the root node of the classification. A new classification folder with this name will be created on the tree. The default setting is UNSPSC.
- **Object Type** field, enter the object type ID of your UNSPSC object. This object type must already exist in System Setup. The default setting is UNSPSC. Update this field to match your object type ID if it is not 'UNSPSC.'
- **Conversion Preview** information is not available for UNSPSC format.

Import Manager



IIEP



xCBL Format

The xCBL format exports data using the CommerceOne XML-based language. For more information, search the web.

Format Availability

xCBL is available for selection in:

- Export Manager - refer to **Creating a Data Export**
- OIEP - refer to **Creating an Outbound Integration Endpoint**

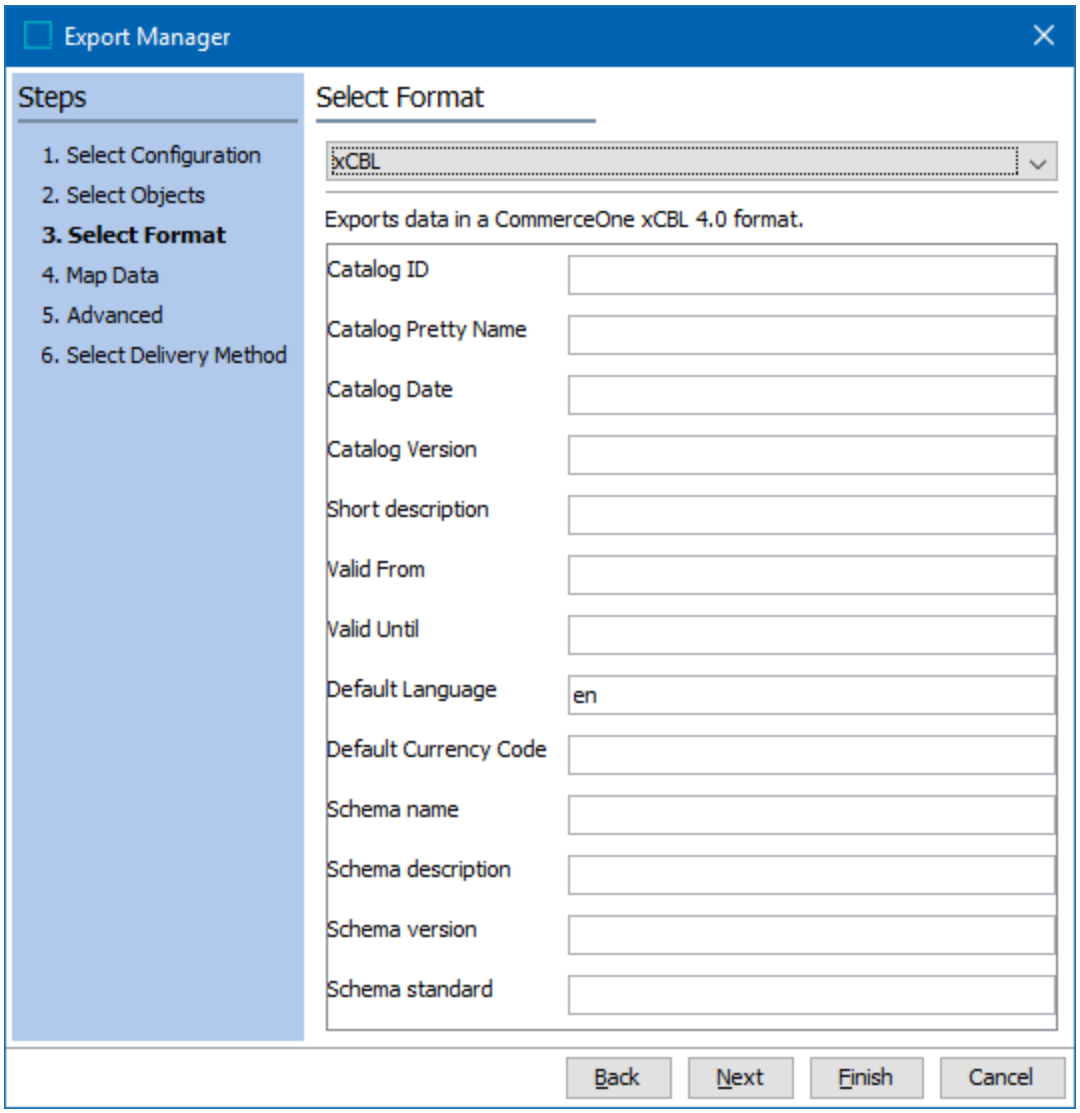
Mapping

This format requires creating a data map between STEP and the data being processed, and may also include data transformations. For details, refer to **Data Mapping**.

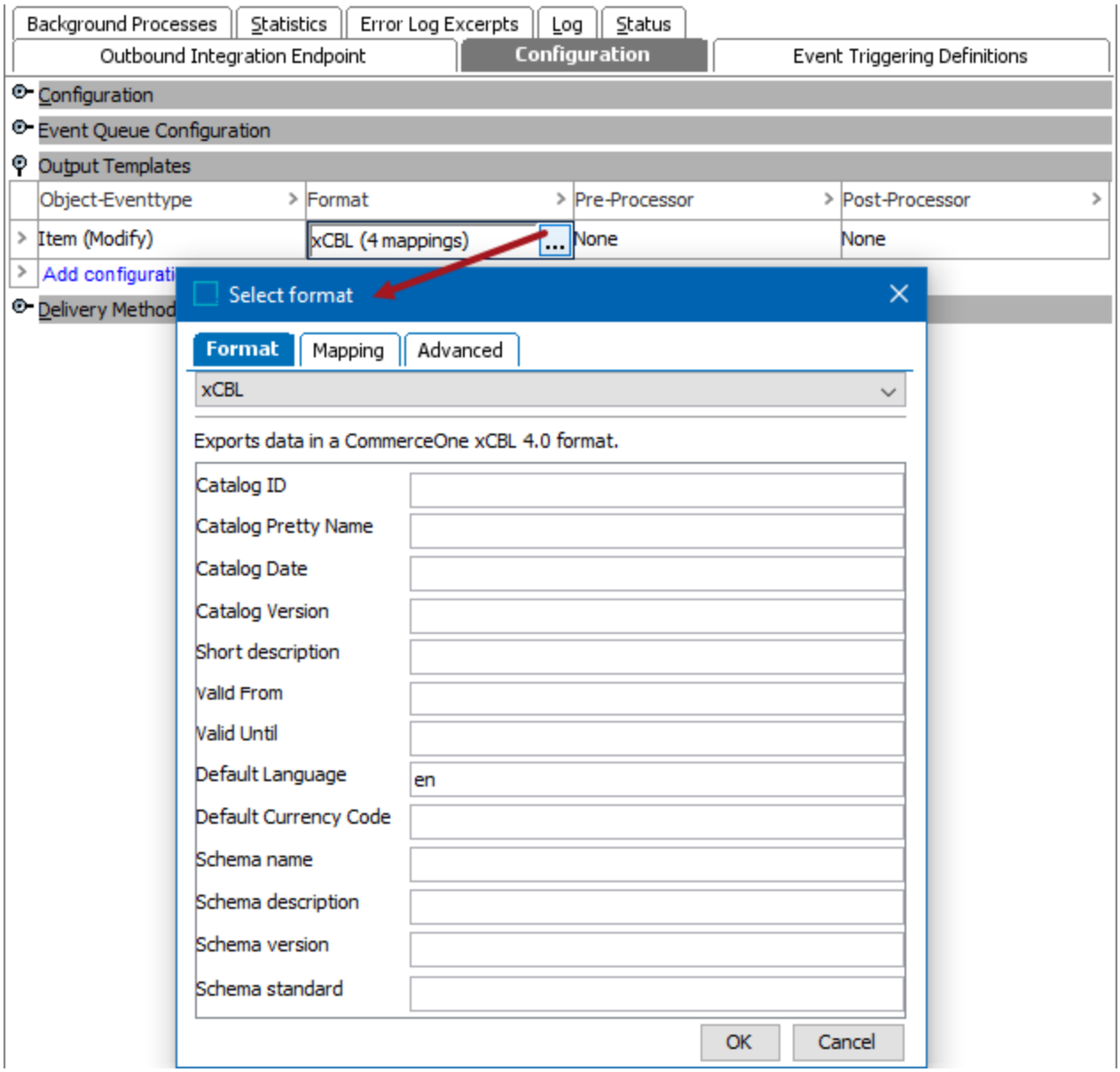
Outbound Data

The same parameters are available in both Export Manager and OIEP.

Export Manager



OIEP


Background Processes
Statistics
Error Log Excerpts
Log
Status
Outbound Integration Endpoint
Configuration
Event Triggering Definitions
Configuration
Event Queue Configuration
Output Templates
Object-Eventtype
Format
Pre-Processor
Post-Processor
Item (Modify)
xCBL (4 mappings)
None
None
Add configurati
Delivery Method
Select format
Format
Mapping
Advanced
xCBL
Exports data in a CommerceOne xCBL 4.0 format.
Catalog ID
Catalog Pretty Name
Catalog Date
Catalog Version
Short description
Valid From
Valid Until
Default Language
en
Default Currency Code
Schema name
Schema description
Schema version
Schema standard
OK
Cancel

Data Mapping

The data format selected determines if data mapping is required to sync data from external systems with STEP data. The mapping functionality also includes transformations, which allow the data to be modified.

Mapping functions are comparable for inbound (Import Manager and IIEPs) and outbound (Export Manager and OIEPs) tools, however there are two user interfaces.

Inbound Data Mapping

Data mapping for import allows you to determine where the incoming data belongs in STEP. For information about the available inbound options, refer to **Inbound Map Data Options**.

Outbound Data Mapping

For export, data mapping identifies the location of STEP data being output. For information about the available outbound options, refer to **Outbound Map Data Options**.

Inbound Map Data Options

The mapping step for the Import Manager and IIEP wizards include the same user interface.

For more information about the additional import steps, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Formats

Mapping differs slightly based on the type of data being imported.

Tabular Formats

The following tabular formats require mapping for inbound data:

- CSV Format
- Excel Format
- FixedWidth Format

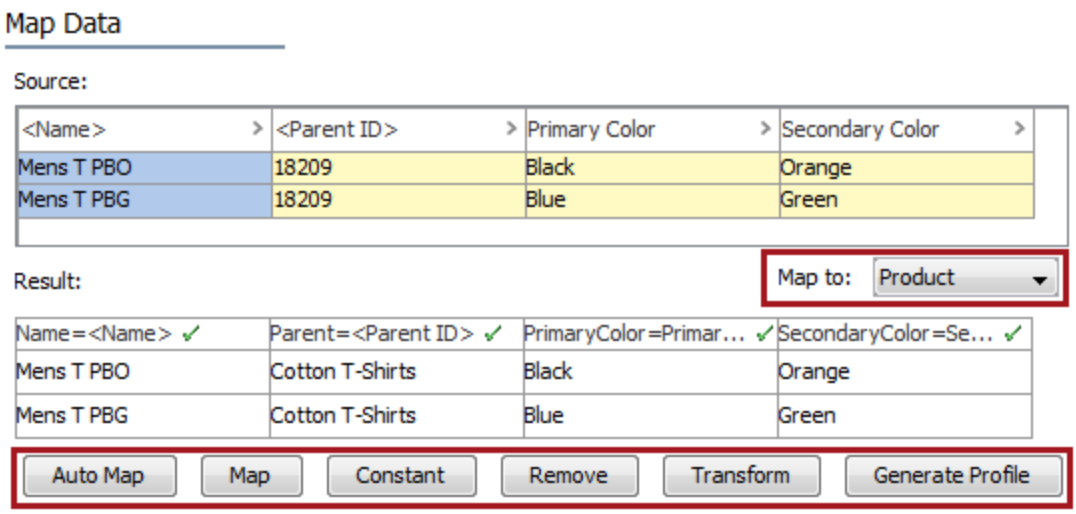
XML Formats

The following XML formats require mapping for inbound data:

- BMEcat Format
- BMEcat2005 Format
- Generic XML Format
- IDoc MATMAS 05 Format

As defined below, in addition to mapping data, you can also select the object type being imported, modify data before import, and verify the quality of the inbound data.

Once mapped, source columns are shown with an yellow background. The result section displays a green check for a data column that has been validated for the object type.



For mapping examples using common column data, refer to the **Inbound Map Data Examples**.

Map To Object Type

The 'Map to' field determines the object type of the data being imported. For more information, refer to the **Inbound Map Data - Map To** section.

Mapping Data

Mapping can be accomplished using one of these three ways:

1. Auto Map - click the **Auto Map** button, STEP reads the header row of the import file, and relates all matches to corresponding objects in STEP. For more information, refer to the **Inbound Map Data - Auto Map** section.
2. Map - manually select an import column and use the **Map** button to relate a STEP object. For more information, refer to the **Inbound Map Data - Map** section.
3. Auto Map and Map manually - after auto mapping an import file with a header row, manually update or correct the results.

If the Result panel includes a column if you do not want to import, click the **Remove** button to eliminate the mapping. For more information, refer to the **Inbound Map Data - Remove** section.

For additional guidance, refer to the **Tips for Mapping Data** below.

Modifying Data

The following options are available to modify data being imported into STEP:

- **Constant** allows you to add an attribute value to all records. Common setup uses this feature for filling some object attributes. For more information, refer to the **Inbound Map Data - Constant** section.

- **Transform** provides numerous ways to modify data before it is loaded into STEP, such as split one data column into multiple attribute values, append and prepend data to the values, perform search and replace, and so on. For more information, refer to the **Inbound Map Data - Transform** section.

Profiling Data

The **Generate Profile** data quality option enables you to locate errors in the data you want to import. The functionality applies to row and column based import formats such as Excel, CSV, and FixedWidth. For more information, refer to the **Inbound Map Data - Generate Profile** section.

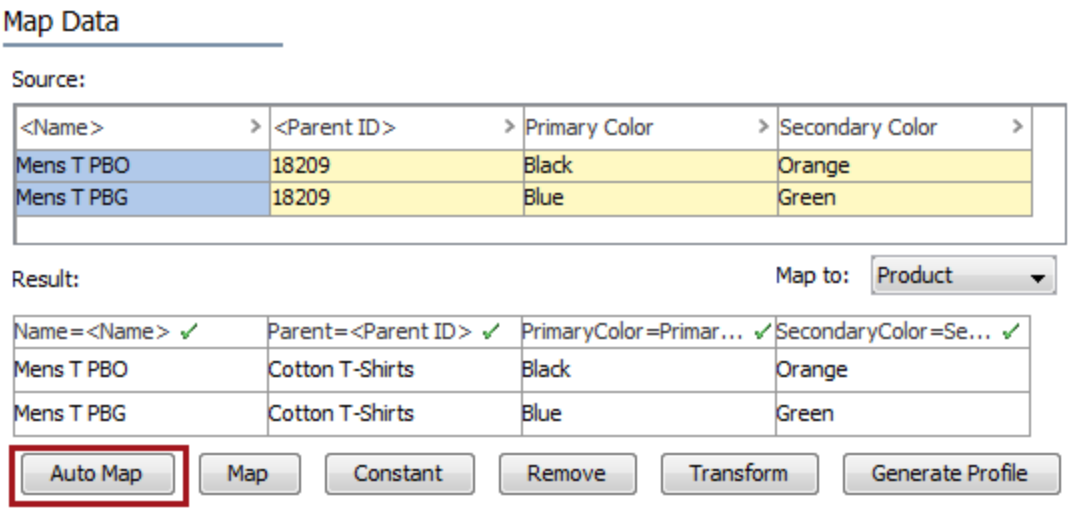
Tips for Mapping Data

Keep the following information in mind when mapping import files:

- Not every column of the input file has to be mapped. For example, if there are five (5) pieces of data to load, but the import file has 20 columns, only map the five (5) relevant columns.
- Mapped columns must correspond to a selected STEP object. Columns that are not mapped are not loaded into STEP.
- Each row (item) of an input file results in one object created or updated. STEP attempts to import every row of the file. Ensure your file contains only the objects that should be created or updated in STEP and remove any unwanted rows before importing.
- Every node and level within the Primary Product Hierarchy is regarded as being a STEP product. The term 'product' includes product families, subcategories, minor categories, major categories, and so on.
- Import file columns can be in any order. Auto Map allows you to automatically match data based on the name or ID of an existing attribute, or you can Map to map columns manually.
- Units are not required in the import file for values for STEP attributes with a default unit. For example, when loading values for the attribute 'Maximum Operating Temperature' with the default unit of °F, there is no need to include the °F in the load file if they are all specified in Fahrenheit. The system imports the value and assigns the °F automatically.
- All mapped columns must be mapped against only one object type (product, asset, classification, entity, or attribute). If the file contains a more than one object type (for example products and entities), you must import the same file several times, each time choosing a single object type. This procedure is often used when building a new product hierarchy.

Inbound Map Data - Auto Map

Auto Map compares data in the input file's header row with the IDs or names of the attributes in STEP. When the inbound tool finds a match, the column is automatically mapped to the corresponding object attributes. The Auto map feature only works when the input file has a header row. You can add a header row if one does not exist, otherwise, you can manually map the data.



For more information about the additional import steps, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Predefined Header Text

Auto Map recognizes the following **case-sensitive** header text to indicate these special columns:

Header Text	Description
<Data Type>	Identifies data on the row of an Excel or CSV file as • NODE (the object being processed) • REFERENCE (a reference owned by the NODE object) • DATA_CONTAINER (a data container owned by the NODE object) Other values in this column are processed as NODE entries.
<Data Container ID>	ID of the data container object of the type identified in the <Data Container Type ID> field and owned by the object in the preceding NODE row.

Header Text	Description
<Data Container Type ID>	ID of the data container type identified in the <Data Container ID> field and owned by the object in the preceding NODE row.
<Data Owner Node>	ID of the object that owns the data on the row. This header text can be mapped during output but is not available for inbound mapping.
<ID>	STEP ID of object
<ID KeyID=..>	ID of unique key for object For example, <ID KeyID=SupplierPartNumber> will automap to the unique key with the ID 'SupplierPartNumber.'
<Name>	STEP name of object
<Object Type ID>	STEP object type ID **Note:** The default selection for exporting object type is 'Object Type Name,' however the importer expects the 'Object Type ID.' To generate an export file that is prepared for reimport, either manually map the object type, or use the Transformation Aspect to map object type ID in the exported file. For more information, refer to the **Aspect - Transform Outbound** section of the **Data Mapping** documentation.
<Parent ID>	STEP ID of object's parent
<Parent KeyID=..>	ID of unique key for object's parent For example, <Parent KeyID=Key3> will automap to the unique key with the ID 'Key3.'
<Reference Type ID>	ID of the reference type for the object identified in the Reference Target field and owned by the object in the preceding NODE row.
<Reference Target ID> or	ID or Key of the target object of the identified in the <Reference Type ID> field and owned by the object in the preceding NODE row. For example, <Reference Target Key KeyID=DescriptionKey> will automap to the unique key

Header Text	Description
<Reference Target Key KeyID=..>	with the ID 'DescriptionKey.'

The import process attempts to auto map all other columns in the input file using Name or ID of existing attributes. Both successful and unsuccessful mappings are shown in the message dialog. If an auto mapping is unsuccessful, you can map that column manually as defined in the **Inbound Map Data - Map** topic.

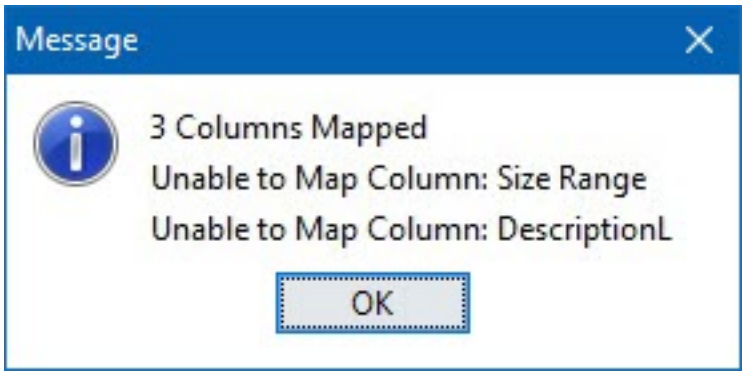
Auto Map Example

The file being imported includes the following columns:

Source:

<Name>	<Parent ID>	Primary Color	Size Range	DescriptionL
Mens T PBO	18209	Black	XS - XXXL	Comfortable fitting 100% cotton tee.
Mens T PBG	18209	Blue	S - XL	Comfortable fitting 100% cotton tee.
Mens T PGS	18209	Green	XS - XXXL	Comfortable fitting tee of cotton/poly blend.
Mens T PGW	18209	Gray	S - XL	Comfortable fitting tee of cotton/poly blend.
Mens T POY	18209	Orange	S - XL	Comfortable fitting 100% cotton tee.

1. In the inbound Map Data step, click the **Auto Map** button. After matching header text to the attributes in STEP, a message displays with the results. The 'Unable to Map Column' error indicates that no attribute exists in STEP with the name. Click **OK**.



Important: If your system has multiple attributes with the same name, use caution when auto mapping. STEP will find a match, but it may not be the attribute you expected. STEP will match exactly if the input file contains the Attribute ID in the header.

2. The Map Data page is updated to show:

- Successfully mapped columns are displayed with a yellow background in the Source panel and a green check mark in the Result panel.
- Columns that are not mapped have a white background in the Source panel.

Source:

<Name>	<Parent ID>	Primary Color	Size Range	DescriptionL
Mens T PBO	18209	Black	XS - XXXL	Comfortable fitting 100% cotton tee.
Mens T PBG	18209	Blue	S - XL	Comfortable fitting 100% cotton tee.
Mens T PGS	18209	Green	XS - XXXL	Comfortable fitting tee of cotton/poly blend.
Mens T PGW	18209	Gray	S - XL	Comfortable fitting tee of cotton/poly blend.
Mens T POY	18209	Orange	S - XL	Comfortable fitting 100% cotton tee.

Result: Map to: Product

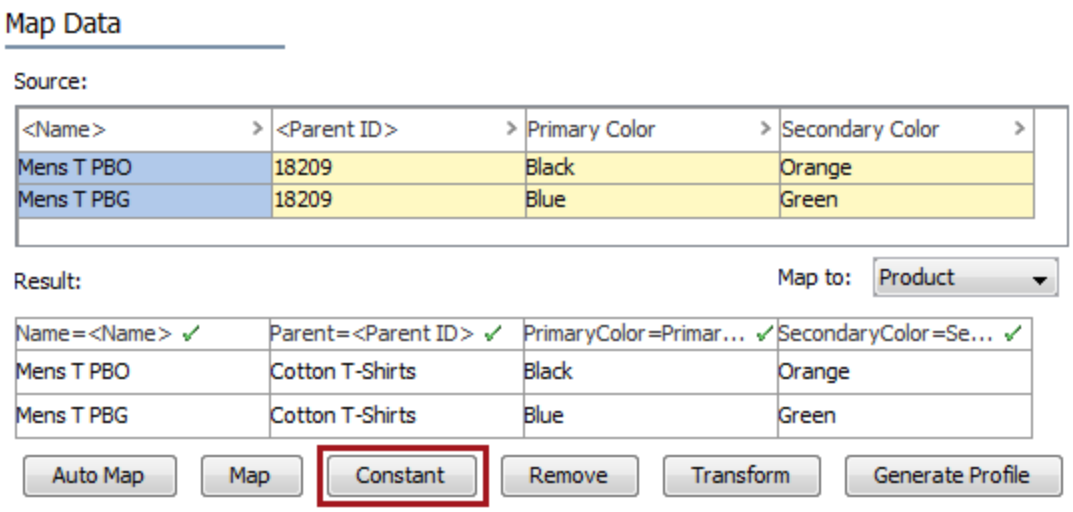
Name=<Name> ✓	Parent=<Parent ID> ✓	PrimaryColor=Primary Color ✓
Mens T PBO	Cotton T-Shirts	Black
Mens T PBG	Cotton T-Shirts	Blue
Mens T PGS	Cotton T-Shirts	Green
Mens T PGW	Cotton T-Shirts	Gray
Mens T POY	Cotton T-Shirts	Orange

Note: ETIM users have the option to use the Auto Map feature, which is enabled when the system identifies industry standard attributes when importing various import formats. Auto Map allows you to map these attributes with attributes that are part of an already-installed ETIM version.

3. Review the **Result** panel. Make any adjustments or transformations if needed.
4. When the mapping is correct, click **Next**.

Inbound Map Data - Constant

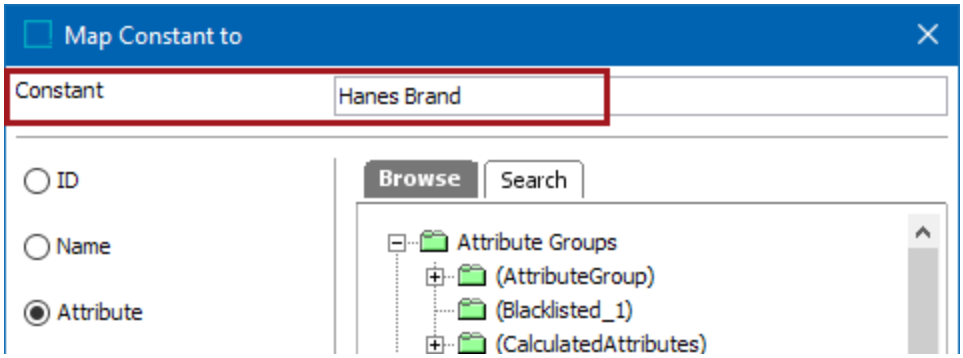
You can add an attribute value to all records by mapping a constant value. Common setup uses this feature for filling some object attributes.



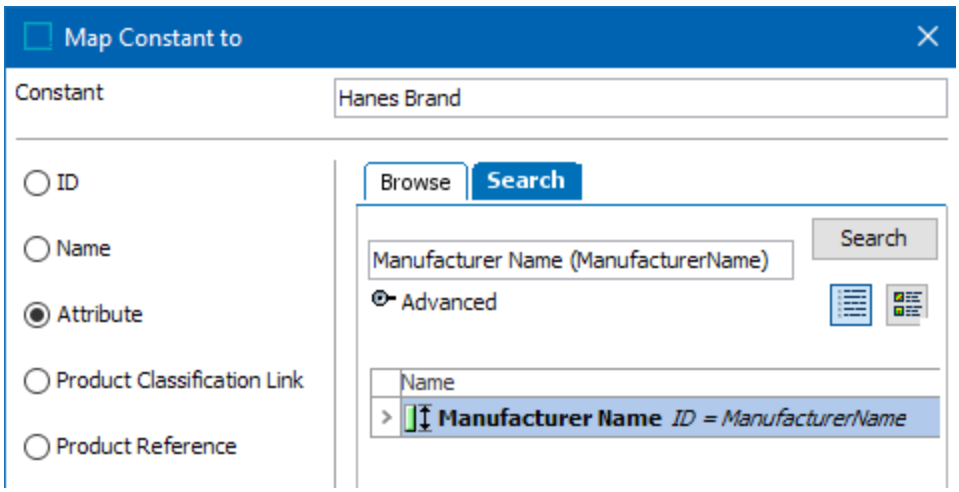
For more information about the additional import steps, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Mapping a Constant

1. Below the Result panel, click the **Constant** button.
2. On the **Map Constant To** dialog, in the **Constant** field, enter text that should be assigned to all records.



3. Select the **Attribute** radio button and use **Browse** or **Search** to select the attribute to be modified. Click **OK**.

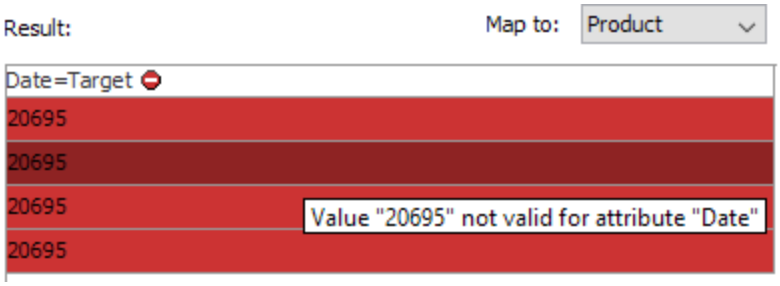


4. The **Result** panel displays the constant value as if it exists in the input file.

Result: Map to: Product

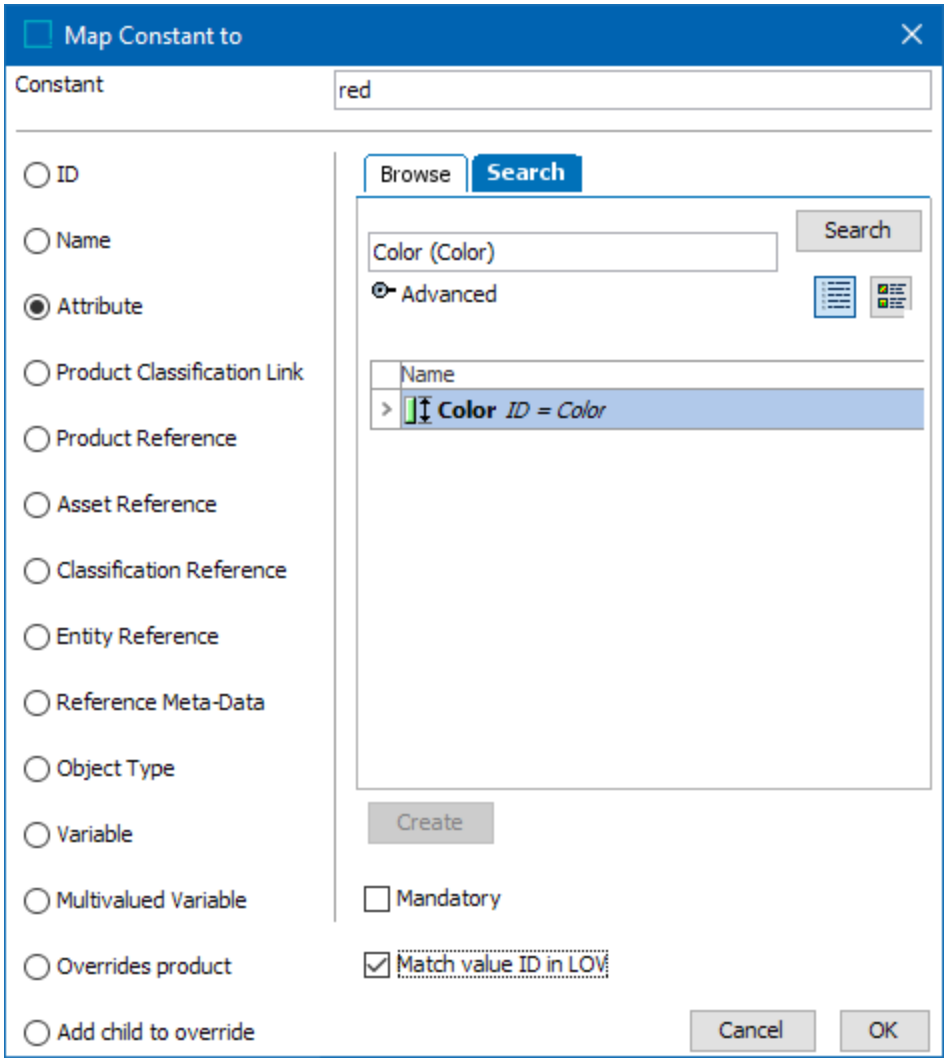
ID=<ID> ✓	Name=<Name> ✓	Parent=<Parent ID> ✓	PrimaryColor=(PrimaryColor) ✓	ManufacturerName=Hanes Brand ✓
18210	18210 M B	Cotton T-shirts	Blue	Hanes Brand
18212	18212 L B	Cotton T-shirts	Red	Hanes Brand
18213	18213 M O	Cotton T-shirts	Black	Hanes Brand
18216	18216 L O	Cotton T-shirts	Green	Hanes Brand
100703	12-GGK79	T-shirts Items	Red	Hanes Brand
112456		T-shirts Items	Orange	Hanes Brand
113202		Polo T-shirt	Green	Hanes Brand

Once a constant value has been mapped to an attribute, for the first 200 rows of the import file, the data is validated against the data model. If the data is not applicable for an attribute due to mismatch with the configured validation will be highlighted as below.

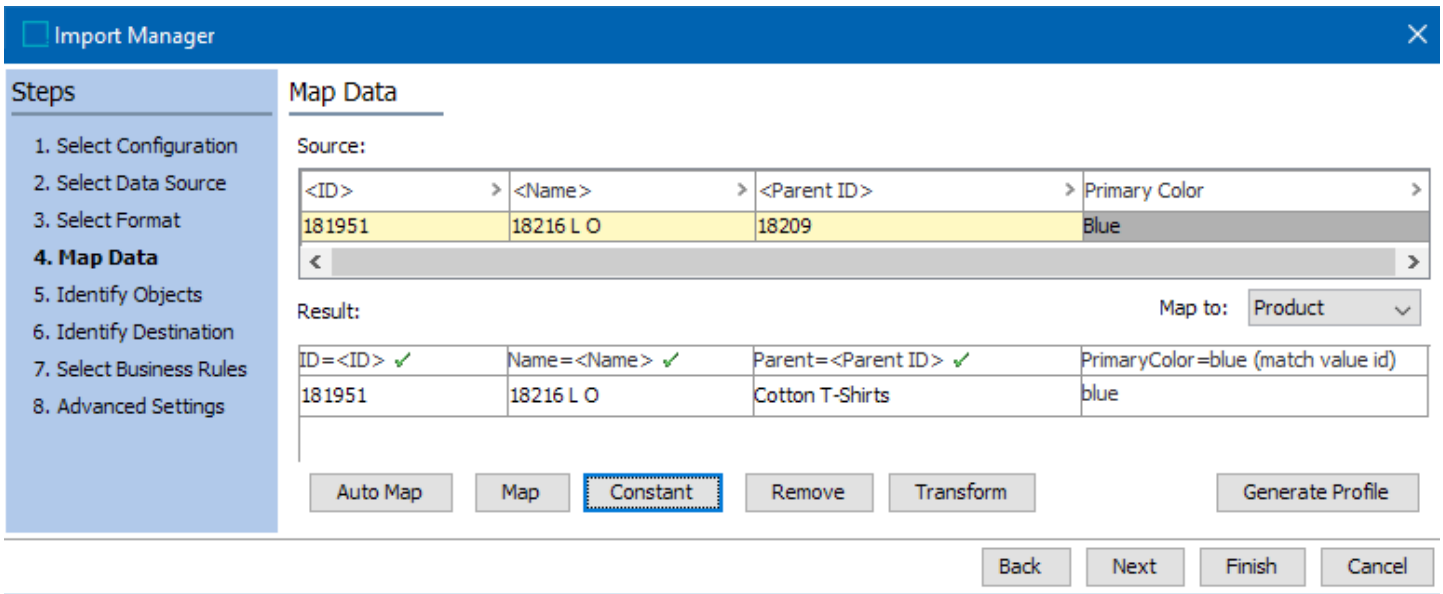


A user can also add the constant value to the LOV attribute, they must enter the value of the LOV, which is case sensitive.

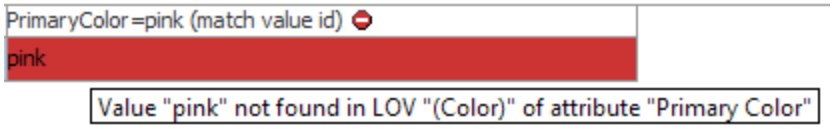
If the LOV attribute holds an LOV with value IDs, then the user can enter the value ID. If the option 'Match value ID in LOV' is checked, the IDs are matched against the IDs in the LOV, and values are applied based on the match, as shown below.



The Result panel displays the mapped values below, and matches the LOV ID.

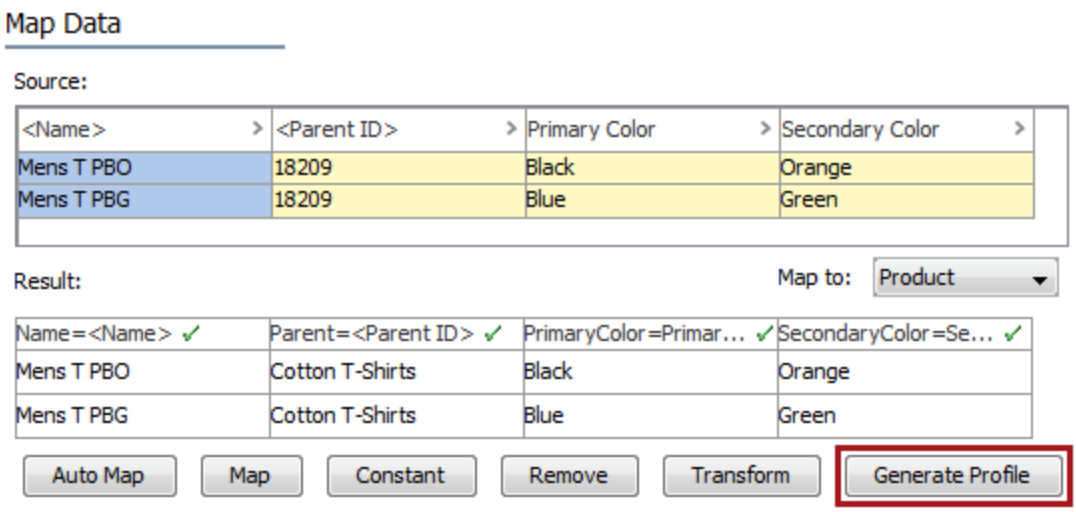


Note: The LOV ID is case sensitive, if there is a mismatch with the LOV ID that exists in STEP, validation will return an error as shown below.



Inbound Map Data - Generate Profile

The import functionality provides a data quality option that enables you to locate errors in the data you want to import. The functionality applies to row and column based import formats such as Excel, CSV, and FixedWidth.



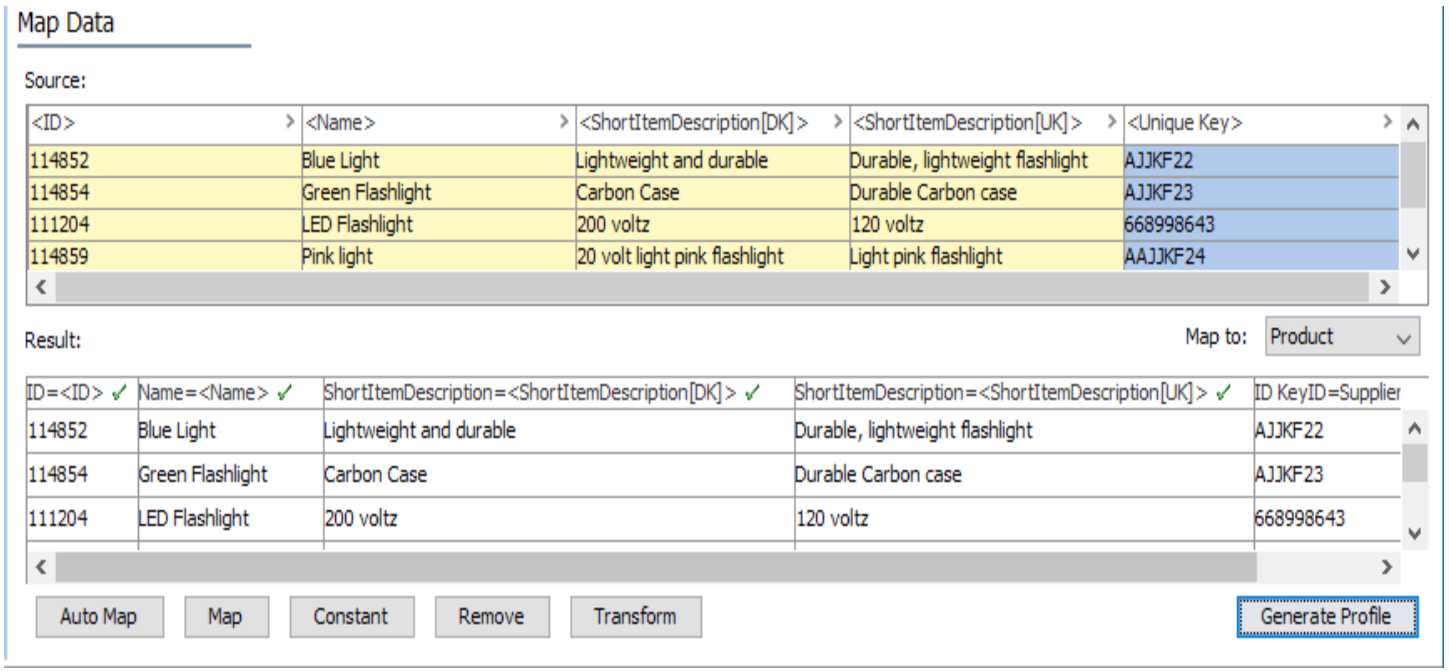
For more information about the additional import steps, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Profiling Setup

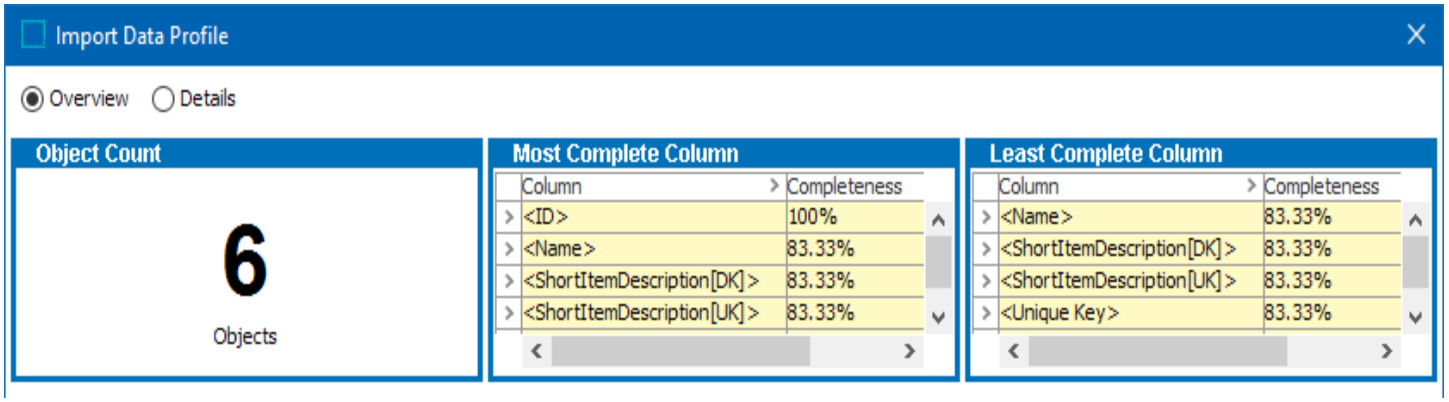
To limit the number of profiled rows and prevent out of memory exceptions, use the property **Import.Profile.Max.Products** in the sharedconfig.properties file. The default value for the property is 10,000. If the number of rows in the import file exceeds the limit, the remaining rows are not profiled.

Profiling Data

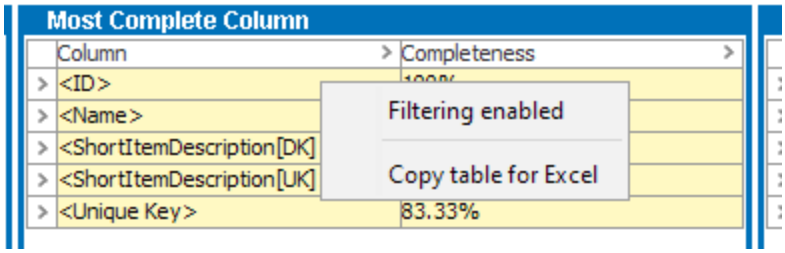
1. Map all required data on the Map Data page and click the Generate Profile button to display the Import Data profile.



2. The Import Data Profile page includes an Overview and a Details view page.
3. Select the **Overview** radio button to display the following three widgets:
   - **Object Count**: displays the number of profiled objects
   - **Most Complete**: displays the 10 most complete columns
   - **Least Complete**: displays the 10 least complete columns



Not that when you select and right-click on either the Most Complete or the Least Complete Column you will find 2 options which are shown below:

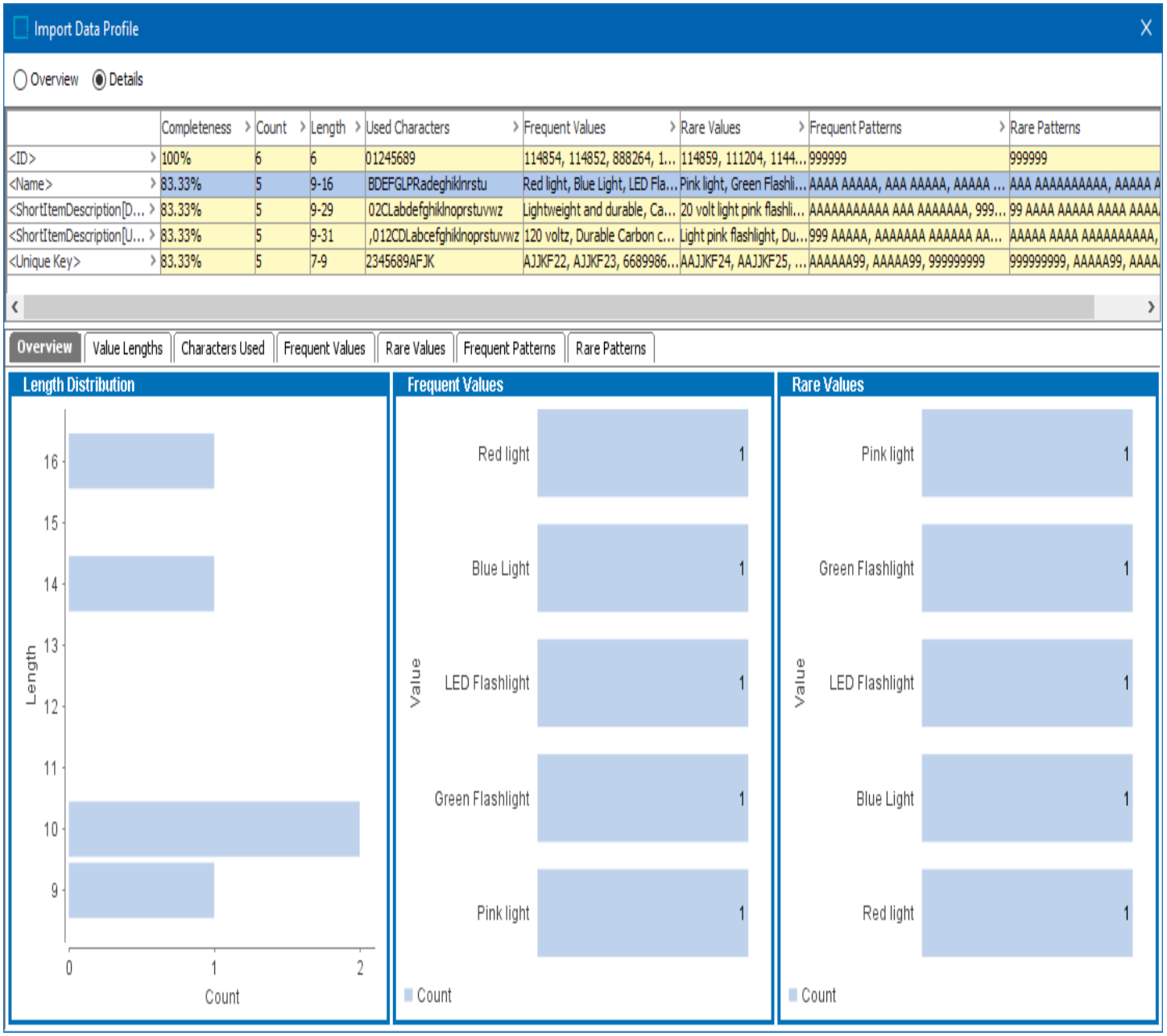


When **Filtering Enabled** is selected, you will get a dropdown list which allows you to filter the based on: Empty Values, Non Empty Values, Equals, Begins with, Ends with, Contains, '<' (lesser than), and '>' (Greater than) options which is nothing but a criteria. Once used, based on the criteria selected, the results will be displayed.

Similarly, when the option Copy table for Excel is selected, both the columns are selected and can be pasted in an excel sheet.

4. Select the **Details** radio button to display a master section at the top and a details section at the bottom.

   In the master section, every column of the import file is represented as a row.



The columns in the master section contain the following information.

Column	Description
Completeness	Shows the degree to which the attribute is populated in the category rounded to the nearest whole percent. Inherited values are included in the calculation.

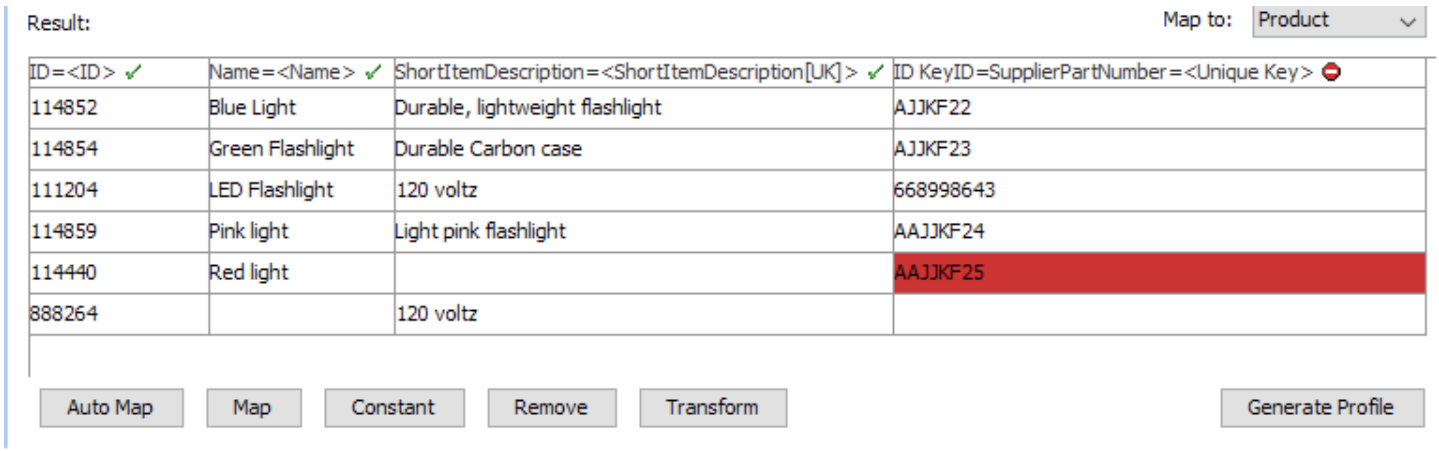
Column	Description
Count	Lists the number of products below the category where the attribute has a value (including inherited values)and the number of products below the category where the attribute could possibly have a value.
Used Characters	List of the characters used for values in the column
Length	Displays value lengths, for example, 3-99 or 10.
Frequent Values	Displays a comma-separated list of the most frequently appearing values. The list is sorted ascending on frequency.
Rare Values	Comma-separated list of the least frequently appearing values. Sorted descending on frequency.
Frequent Patterns	This tab lists the most frequent patterns of the values of a given attribute together with the number values that match the pattern. The pattern describes the structure of a value.
Rare Patterns	This tab lists the least frequent patterns of the values of a given attribute together with the number values that match the pattern. The pattern describes the structure of a value.

5. Select a row in the master section to populate the details section at the bottom. The five tabs present a different view of the information that is displayed in the master section.
   - **Overview** - Displays widgets for length distribution, frequent values, and rare values bar charts. If all values are of the same length, the length distribution bar chart is not displayed. Likewise, if only a single unique value exists in the import column, the frequent and rare values widgets are not be displayed.
   - **Value Lengths** - Displays the number of occurrences of each value length.
   - **Characters Used** - Displays the number of occurrences of each character.
   - **Frequent Values** - Lists the most frequently appearing values.
   - **Rare Values** - Lists the least frequently appearing values.
   - **Frequent Patterns** - Lists the frequent pattern in which the values are entered
   - **Rare Patterns** - List the most rare pattern in which the values are entered

For more information on Data Profiling, refer to the **Data Profiling** documentation.

On Demand Data Validation

When you map columns in the import file to data in STEP with the Data Quality functionality, the mapped data is validated on demand. The data validation status is displayed with an icon on the header of each column in the Result area of the **Map Data** step. Cells with validation errors are highlighted with red background color.



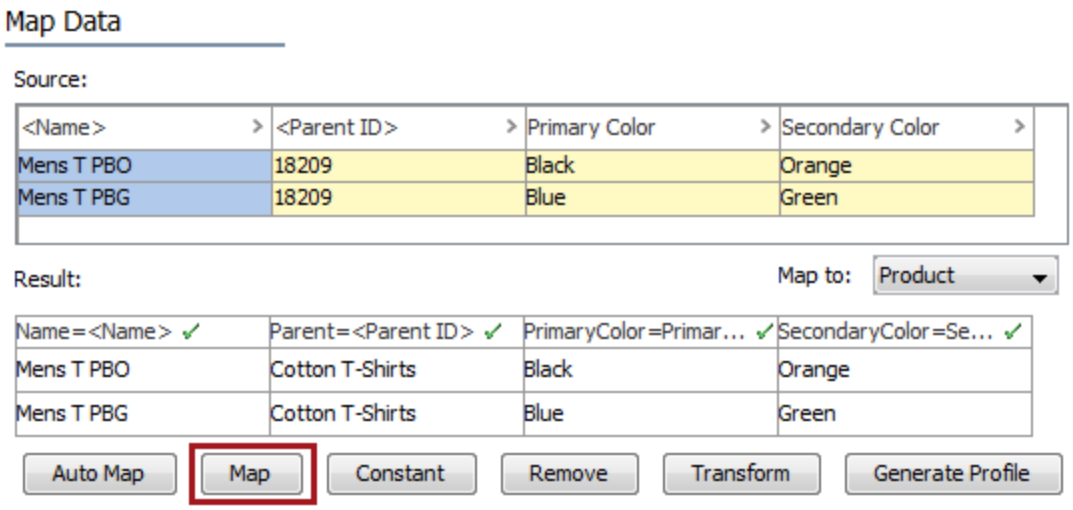
Mouse over a cell with validation errors and the hover text provides information about the problem. If you apply transformations to a result column, the validation feature is also available in the Transformation Preview. Data is re-validated with every change. Refer to the **Inbound Map Data - Transform** topic in the **Data Exchange** documentation for more details.

Important: Only data for the first 200 rows in the import file is validated and displayed in the result panel.

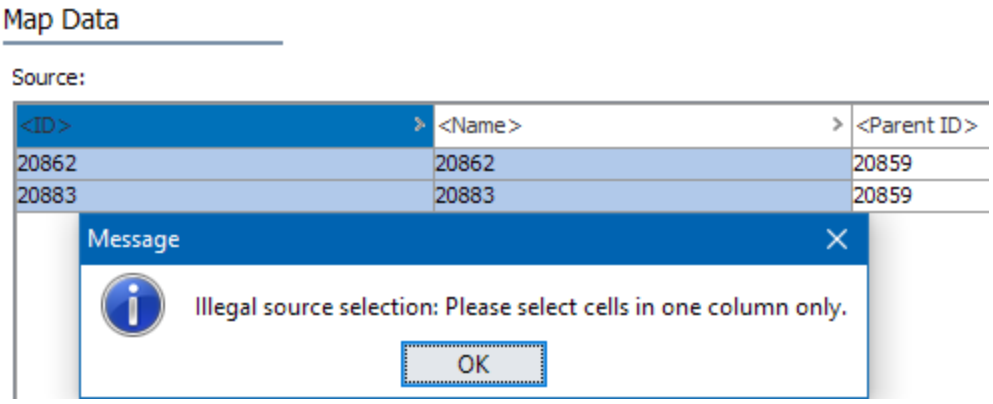
It is possible to start an import process even if validation errors have been discovered and the same errors are displayed in the background process in the execution report.

Inbound Map Data - Map

When an inbound file does not contain a header row, you must use the Map button to map the data manually. Manual mapping can also be used to modify the mappings generated with the Auto Map feature. For information about automatic mapping, refer to the **Inbound Map Data - Auto Map** documentation.



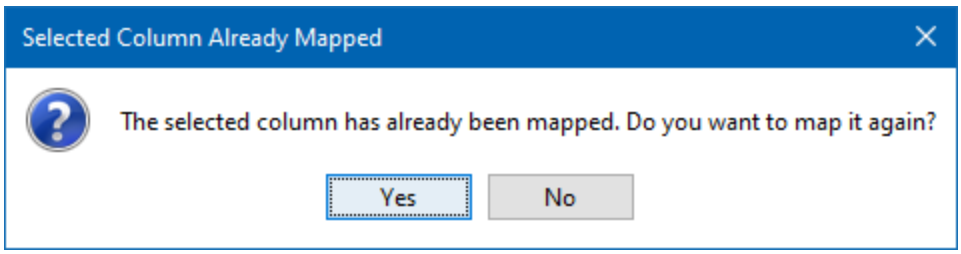
Source columns must be mapped individually. If multiple source columns are selected prior to clicking the Map button, a message is displayed saying the source selection is not valid.



Mapping the Same Column Multiple Times

A single source column can be mapped as many times as needed. For example, if an import column contains three dimensions for height, length, and width, you might want to map it three times so that the elements can be saved to three separate STEP attributes. For an example, refer to **Split and Extract Data Example**.

When you attempt to map a column that is already mapped, although a warning displays, you can click **Yes** to map it again, or click **No** and select a different column.



The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Mapping Options

These mapping options are available both in IIEP and Import Manager.

Refer to the following topics for the specific options available for manual mapping, based on the object type selected in the **Map to** dropdown parameter.

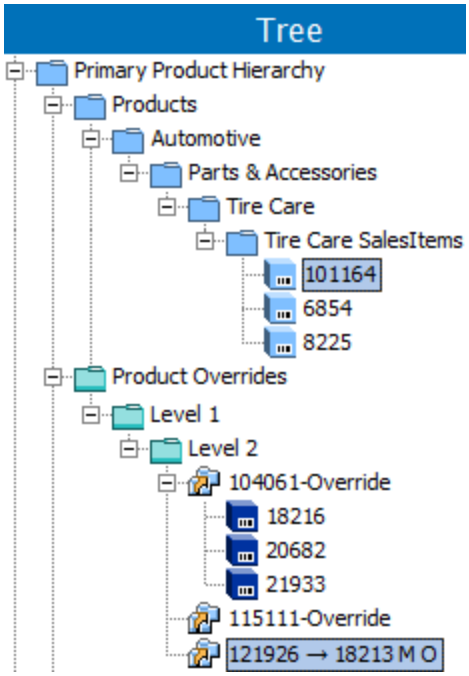
- Add Child to Override - Map Inbound
- Asset Reference - Map Inbound
- Attribute - Map Inbound
- Attribute Info - Map Inbound
- Attribute Validation Info - Map Inbound
- Classification Reference - Map Inbound
- Data Container - Map Inbound
- Data Container Type ID - Map Inbound - within Inserted References / Data Containers
- Data Container ID - Map Inbound - within Inserted References / Data Containers
- Data Type - Map Inbound - within Inserted References / Data Containers
- Delete Values - Map Inbound
- Entity Reference - Map Inbound
- ID or Key - Map Inbound
- Multivalued Variable - Map Inbound
- Name - Map Inbound
- Object Type - Map Inbound
- Overrides Product - Map Inbound

- Parent - Map Inbound
- Product Classification Links - Map Inbound
- Reference Meta-Data - Map Inbound
- Reference Target - Map Inbound - within Inserted References / Data Containers
- Reference Type ID - Map Inbound - within Inserted References / Data Containers
- Variable - Map Inbound

Add Child to Override - Map Inbound

A product override folder can include products from various 'standard' (non-override) product families. Similarly, a product override leaf object (child to override) can live under a different product folder (override or standard) than the folder in which its corresponding standard object resides.

A single import file can include multiple products added as a child to a single Product Override or to multiple Product Overrides. For this example, two 'salesitem' product objects are added as children to two separate Product Override objects.



The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Prerequisites

The required steps include:

1. Setting up the Product Override object type in System Setup. Refer to the **Product-Override Object Types and Commercial Object Types** topic in the **System Setup** documentation.
2. Verifying the import file before starting the import as defined in the **Verify Import File** section below.
3. Mapping 'Add child to override' as defined in the **Map Add Child to Override** section below. To add a child during creation of an initial product override, refer to the **Product Overrides** topic in the **Getting Started** documentation.

Verify Import File

To ensure that the object is imported successfully and correctly, before starting the import, open the import file and ensure that the following fields exist:

- **ID:** The ID of the parent that the child product override will live under.
- **Add Child to Override:** The ID of the product in STEP that will get an override version of itself as a child to a parent override.

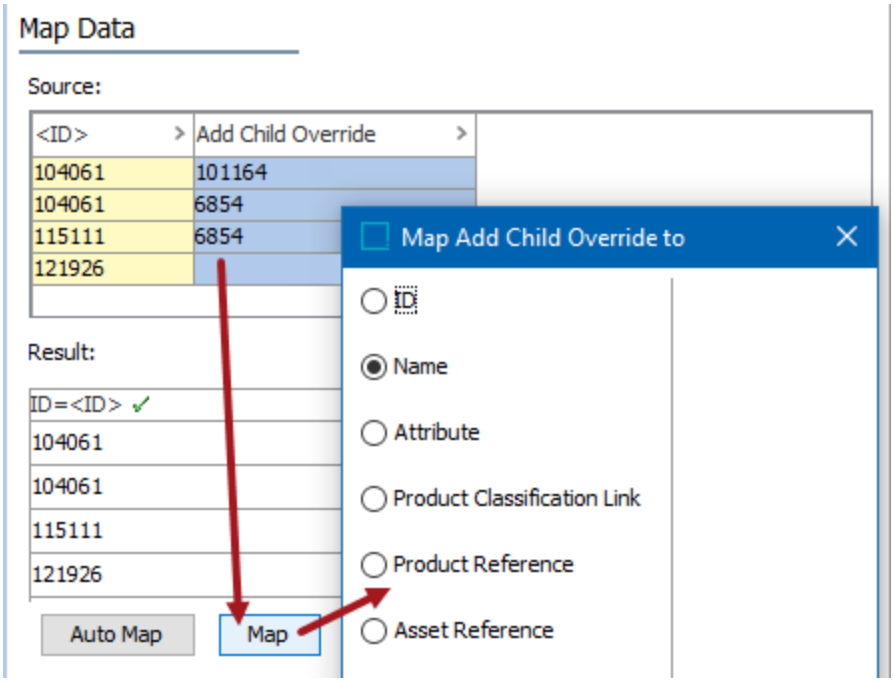
This example shows an Excel import file (although any inbound format can be used). The file adds two objects as children for one product override (ID 104061), and one of those objects is also added as a child on another product override (ID 115111). Any additional data on the final row in this file will be applied to the product override (ID 121926) based on the setting of the Mandatory parameter as defined below.

A	B
<ID>	Add Child Override
104061	101164
104061	6854
115111	6854
121926	

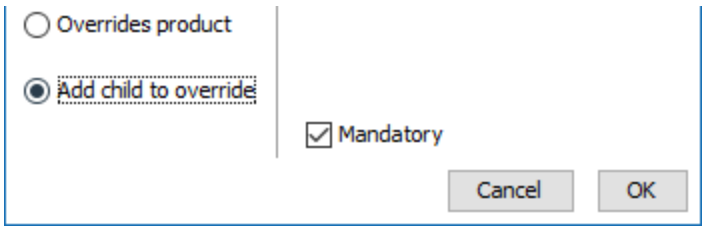
Map Add Child to Override

Once System Setup is configured, an object can be imported as a child to a product override using Import Manager or Inbound Integration Endpoint.

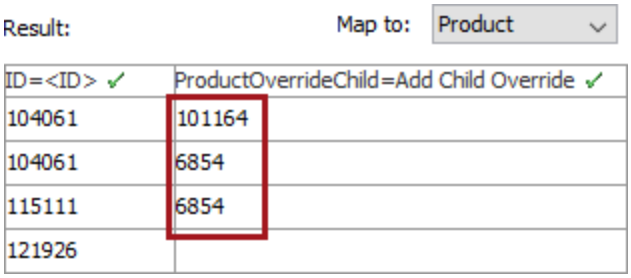
1. Use Auto Map to map the ID, or map it manually. For more information, refer to **ID or Key - Map Inbound**.
2. In the Source section, first select the Add Child Override column and then click the **Map** button to display the 'Map ...to' dialog. The label of the field is used as the title of this dialog.



3. On the Map ... to dialog, select the **Add child to override** radio button.



4. Check the **Mandatory** option appropriately:
   - If **checked**, imported objects must have an Add Child Override value in the file. If an Add Child Override value does not exist for an object, the object is skipped and not imported.
   - If **unchecked**, all objects are imported regardless if an Add Child Override value is specified or not.
5. Click **OK** and the **Result** panel displays your Add Child Override values. A validated data column is marked with a green check mark as shown below.

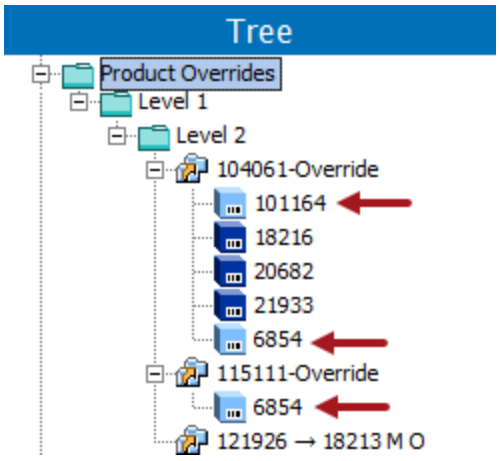


ID=<ID> ✓	ProductOverrideChild=Add Child Override ✓
104061	101164
104061	6854
115111	6854
121926	

6. Complete the mapping and initiate the import.

Result

When import is successful, the objects display under the designated Product Override parent in Tree, as indicated by the arrows in the following image. The last object was not modified since mandatory was checked and no ID was included for the Add Child Override field in the import file.



Asset Reference - Map Inbound

Asset references are available for product, classification, and entity objects. The reference types available for mapping during import are based on the asset reference types that exist in STEP.

Before you create mappings, note the following:

- The existing reference types for the super type selected in the wizard are available for mapping. New reference types cannot be specified or created during import. The reference type is not read from the input file.
- The same column can be mapped multiple times when a different asset reference type is needed for the same target.

Below are the steps required to map the following asset references:

- Product to Asset References
- Classification to Asset References
- Entity to Asset References

The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Prerequisites

To ensure that the object is imported successfully and correctly, before starting the import, open the import file and ensure that the following data exists:

- The inbound file must contain the ID or key (not the name) of the source object for the reference (product, classification, or entity).
- The inbound file must contain the ID or key (not the name) of the target object for the reference (asset).
- Objects in the inbound file must already exist in STEP.
- The desired reference type must be valid for the object types being referenced.

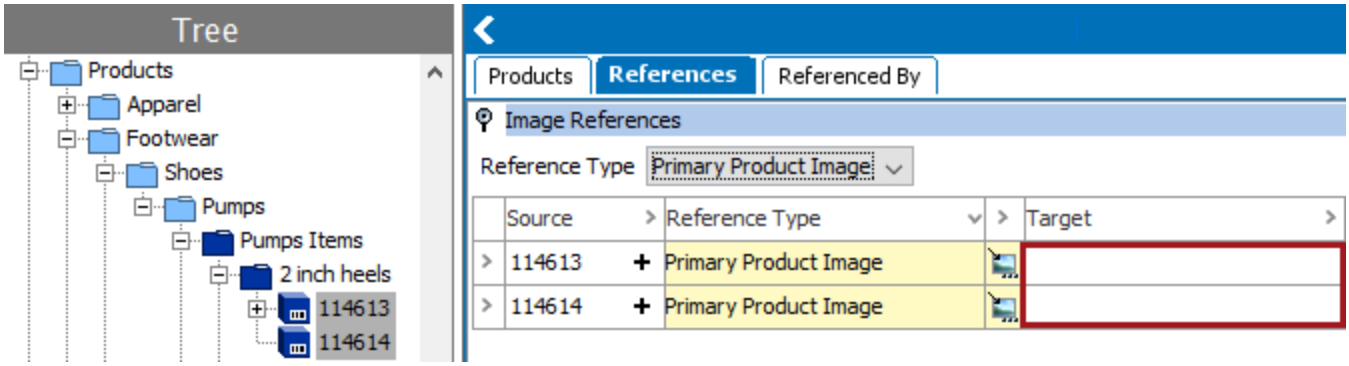
If the objects to be imported or the Target to be mapped is identified via a unique key, you must verify that the Key required is available and activated. For more information, refer to the **Activating and Deactivating Keys** topic of the **System Setup** documentation.

Map Product to Asset Reference

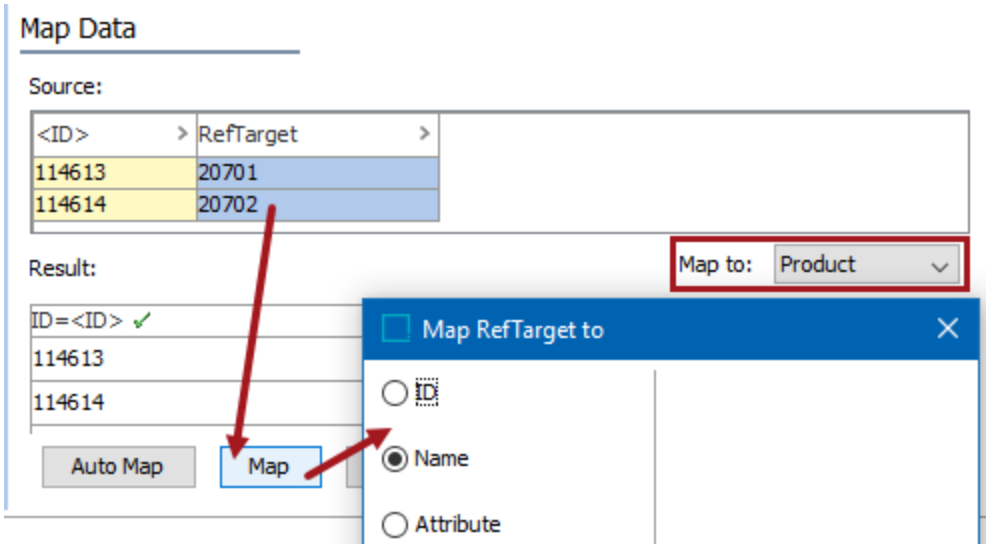
This example shows an Excel import file (although any inbound format can be used). The file includes target asset objects to be used as the primary product image for two products (indicated by the ID column).

A	B
<ID>	RefTarget
114613	20701
114614	20702

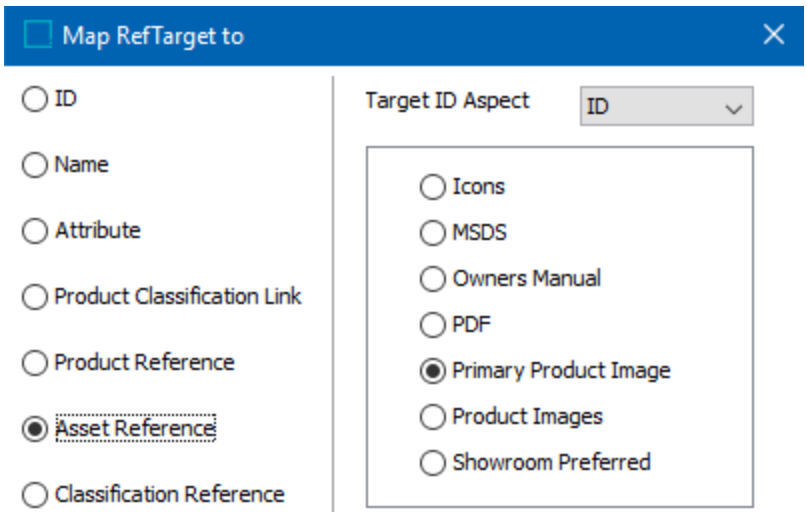
The import will specify an asset that should reference a product using the reference type 'Primary Product Image.' Before the import, there are no image references for either source product, as shown simultaneously in the image below.



1. In the Source window, select the ID or Key column and map it. For more information, refer to the **ID or Key - Map Inbound** documentation.
2. Below the Source section, in the **Map to** parameter, select the **Product** super type.
3. In the Source section, select the reference target column and then click the **Map** button to display the Map ... to dialog. The label of the field is used within the title of this dialog.



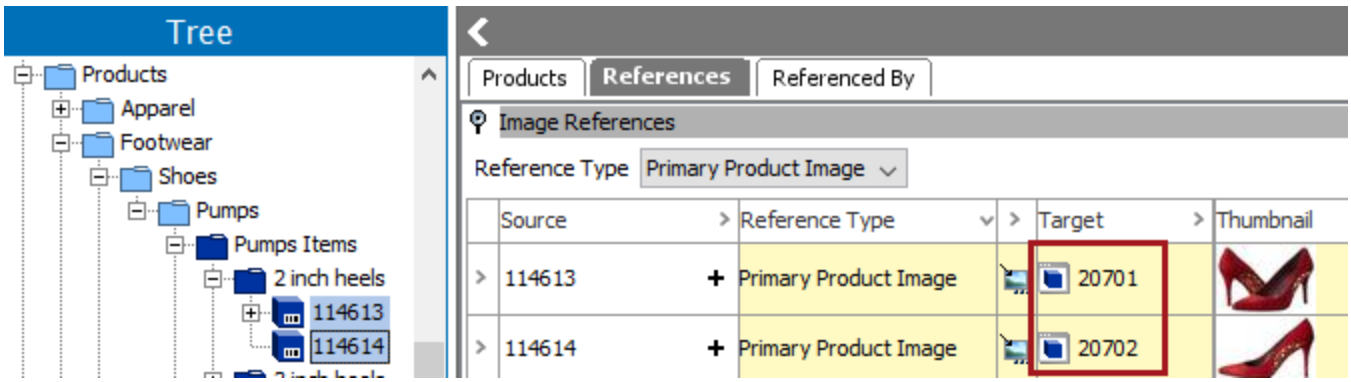
4. On the 'Map ...to' dialog, select the **Asset Reference** radio button to display the list of available reference types on your system. Select the radio button of the desired reference type. Because the Product super type was selected previously, only product-to-asset references are displayed.



5. Check the **Mandatory** option appropriately:
   - If **checked**, imported objects must have an ID for the selected reference in the file. If an ID for the reference does not exist for an object, the object is skipped and not imported.
   - If **unchecked**, all objects are imported regardless if an ID for the reference is specified or not.

6. Click **OK** and the **Result** panel displays your asset reference values. The Result section shows the selected product-to-asset reference type. A validated data column is marked with a green check mark as shown below.

7. Complete the mapping and initiate the import. When the import is successful, the imported references display on the References tab of the product within the appropriate flipper.



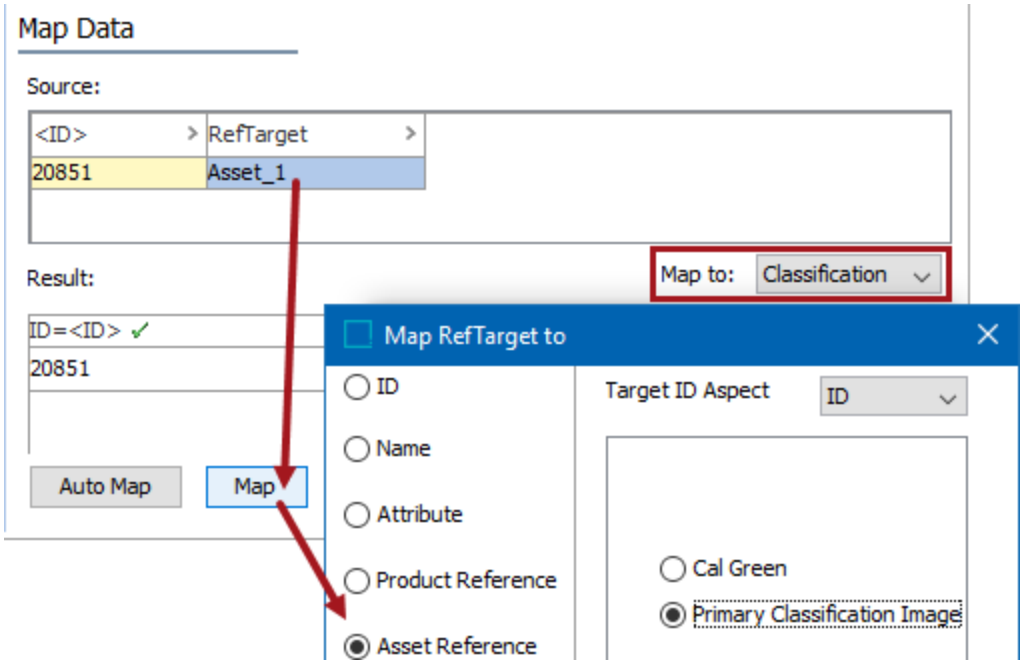
Map Classification to Asset Reference

Importing a classification-to-asset reference is similar to the previous steps for a product-to-asset reference, with the following differences:

- If you want to make references from the classification objects to an entity, asset, or classification, verify that the object and entity, asset, or classification exists in STEP and also the reference type is valid for the object types in the input file. Lastly, verify that the classification ID is specified in the input file, not the name.
- The import file must include the ID of the classification and the ID of the asset.

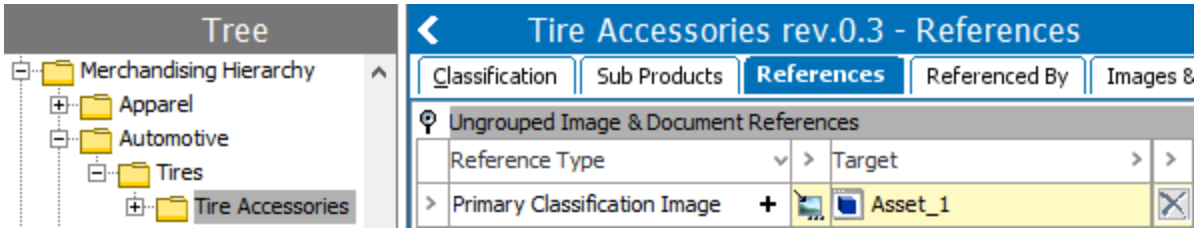
A	B
<ID>	RefTarget
20851	Asset_1

- The **Map to** parameter must be set to the Classification super type prior to selecting the reference target and clicking the Map button.



- Only classification-to-asset references are displayed after selecting the Asset Reference radio button on the **Map ... to** dialog. The Result section shows the selected classification-to-asset reference type.

- When the import is successful, the imported references display on the References tab of the classification within the appropriate flipper.



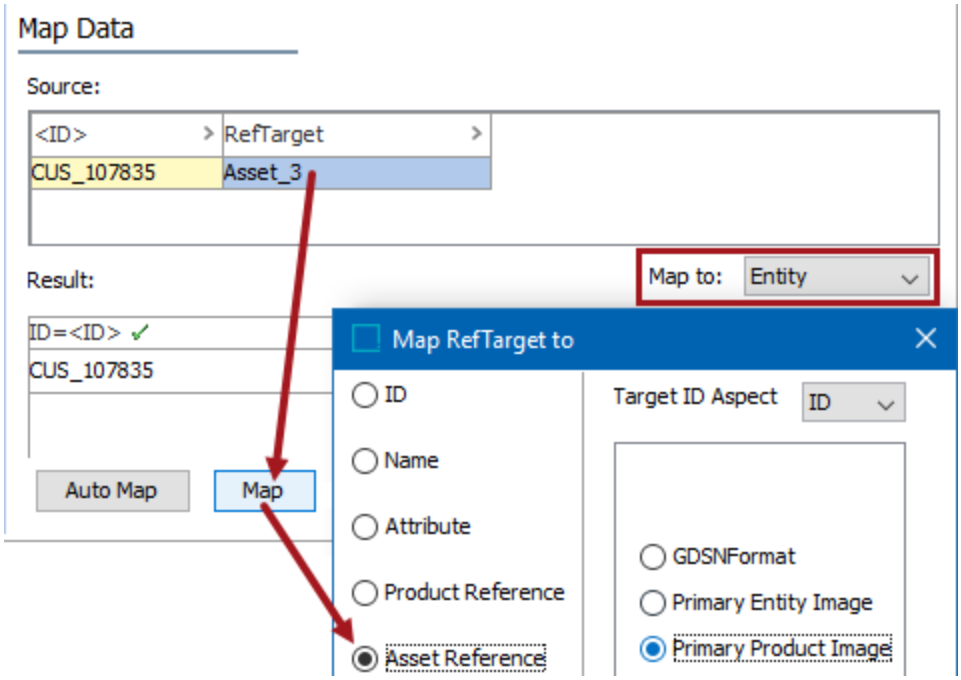
Map Entity to Asset Reference

Importing an entity-to-asset reference is similar to the previous steps for a product-to-asset reference, with the following differences:

- The import file must include the ID of the entity and the ID of the asset.

A	B
<ID>	RefTarget
CUS_107835	Asset_3

- The **Map to** parameter must be set to the Entity super type prior to selecting the reference target and clicking the Map button.

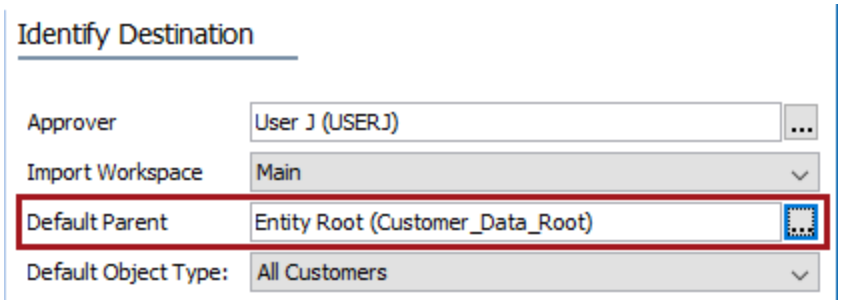


- Only entity-to-asset references are displayed after selecting the Asset Reference radio button on the **Map ... to** dialog. The Result section shows the selected entity-to-asset reference type.

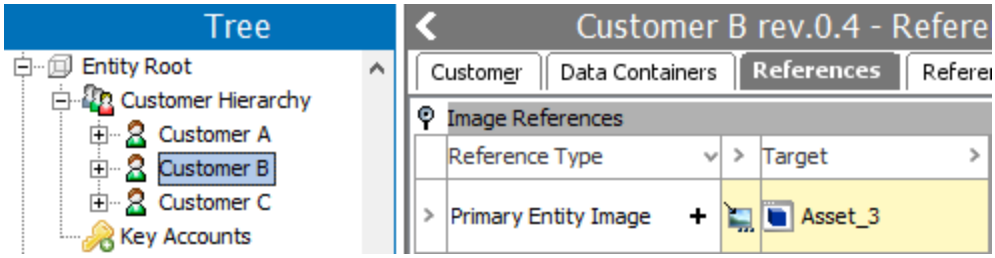
Result: Map to: Entity

ID=<ID> ✓	Primary Entity Image AssetReference=RefTarget ✓
CUS_107835	Asset_3

- On the Identify Destination step of the import, on the **Default Parent** parameter, click the ellipsis button (...) to display the valid entity parent nodes. Select a node and click the **Select** button.



- When the import is successful, the imported references display on the References tab of the entity within the appropriate flipper.



Attribute - Map Inbound

Mapping for attributes is available for both attributes that already exist in STEP and those that need to be created, as defined below.

The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Map to an Existing Attribute

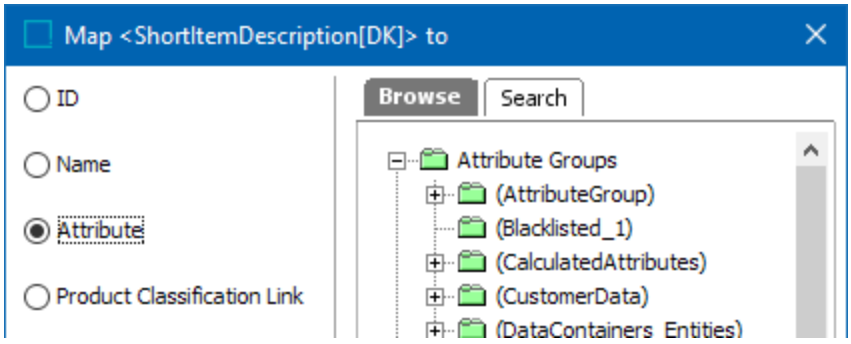
You can map a data column in your input file to an attribute that already exists in STEP. For Classification(s) you can map the data to a description attribute only.

1. In the Source section, select the column of data that you want to map to an attribute and click the **Map** button.

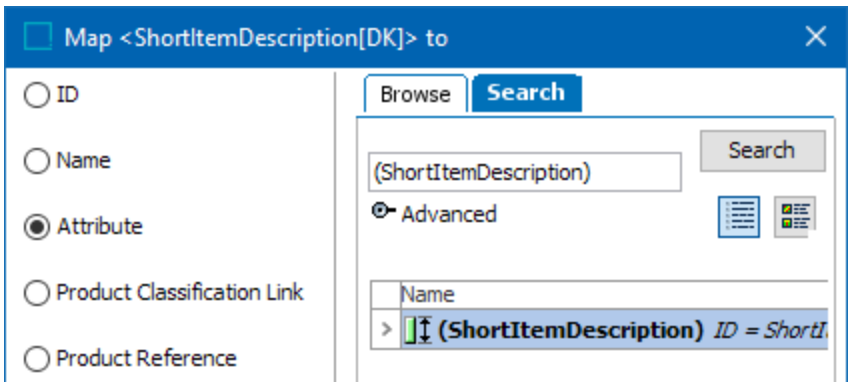
Source:

<ID>	<Name>	<Parent ID>	Primary Color	Secondary Color
MT18400	Mens T PBO	18209	Black	Orange
MT18401	Mens T PBG	18209	Blue	Green
MT18402	Mens T PGS	18209	Green	Silver
MT18403	Mens T PGW	18209	Gray	White
MT18404	Mens T POY	18209	Orange	Yellow

2. In the 'Map ... to' dialog, select the **Attribute** radio button.



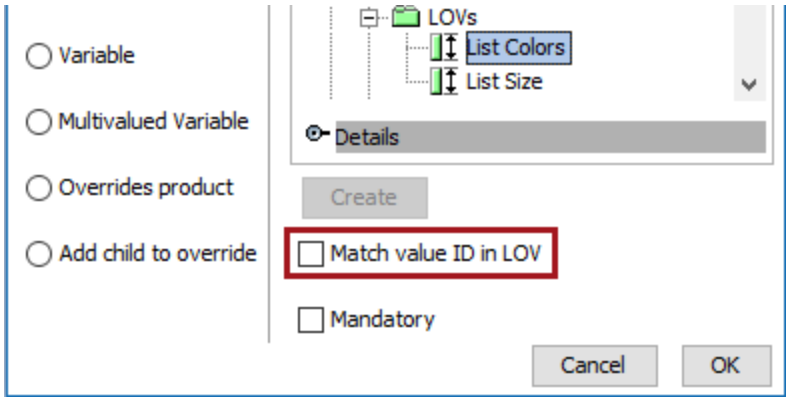
3. In the displayed attribute selector, use Browse or Search to select the STEP attribute to be mapped.



Search is available on all or part of the attribute's name or ID. Click the advanced search flipper to use other options, such as searching for attributes with a specific value.

Note: Only description attribute(s) can be mapped to a classification. Selecting a specification attribute displays an error.

4. If the attribute selected is an LOV, set the **Match value ID in LOV** appropriately:
   - If **checked**, the ID of the LOV must be included in the inbound file.
   - If **unchecked**, the LOV ID is not required.



5. Check the **Mandatory** option appropriately:
   - If **checked**, imported objects must have an attribute value in the file. If an attribute value does not exist for an object, the object is skipped and not imported.
   - If **unchecked**, all objects are imported regardless if an attribute value is specified or not.

6. Click **OK** and the **Result** panel displays your selection of column and attribute. A validated data column is marked with a green check mark as shown below.

Result: Map to: Product

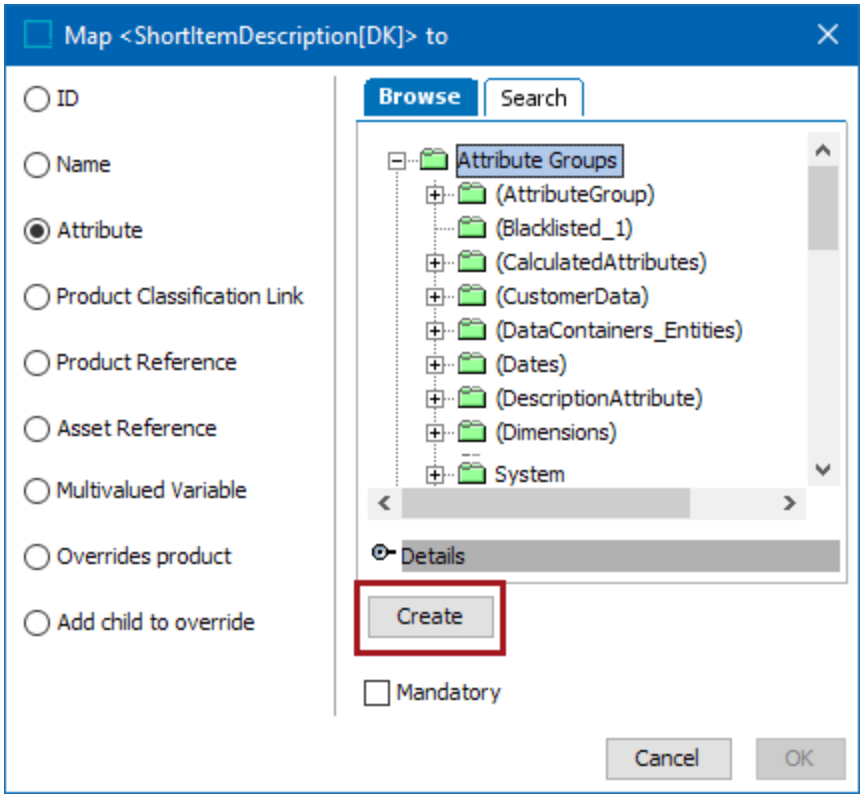
ID=<ID> ✓	Name=<Name> ✓	Parent=<Parent ID> ✓	PrimaryColor=Primary Color ✓
MT18400	Mens T PBO	Cotton T-Shirts	Black
MT18401	Mens T PBG	Cotton T-Shirts	Blue
MT18402	Mens T PGS	Cotton T-Shirts	Green
MT18403	Mens T PGW	Cotton T-Shirts	Gray
MT18404	Mens T POY	Cotton T-Shirts	Orange

7. Complete the mapping and initiate the import.

Create a New Attribute and Map

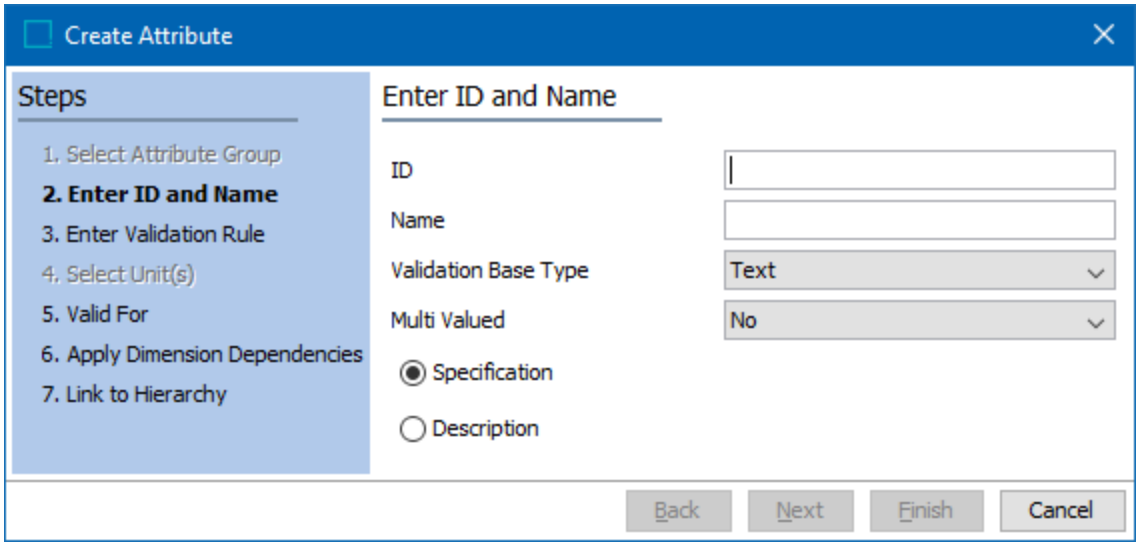
When a data column in your input file is an attribute that does not yet exist in STEP, you can create the new STEP attribute during the import process, and then map the data column to it.

1. In the 'Map to' window, select the attribute radio button and navigate to the appropriate attribute group to house the new attribute. Click the **Create** button.

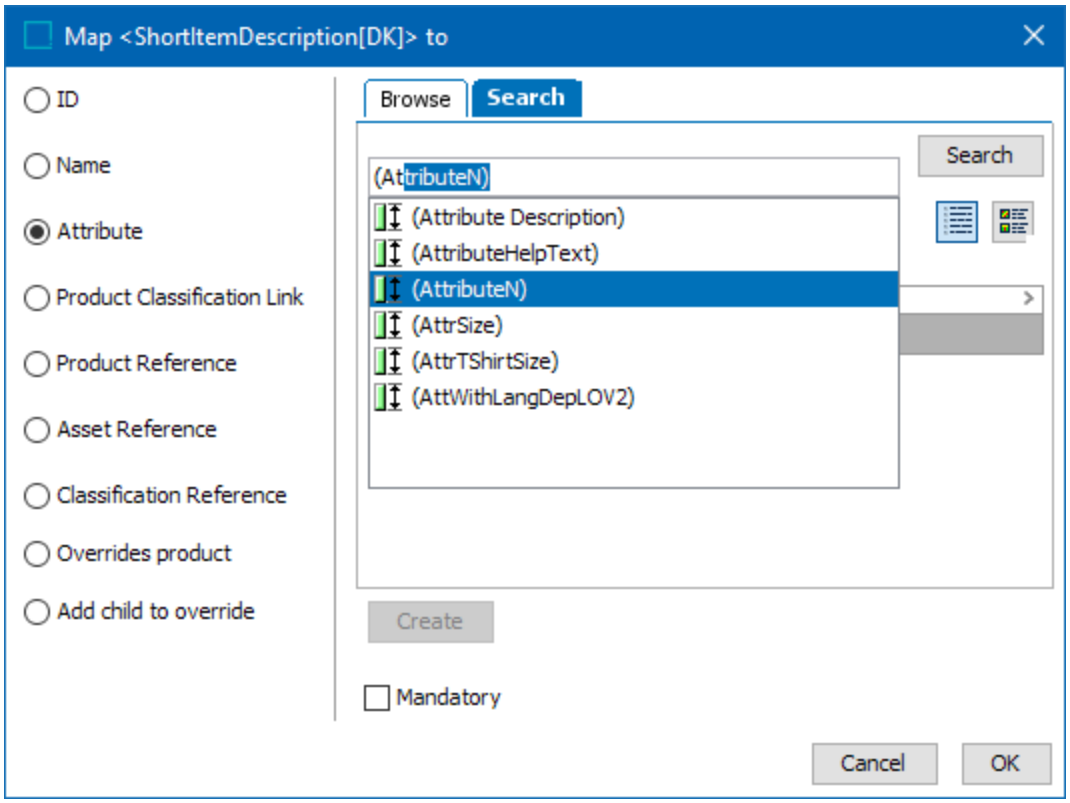


2. In the Create Attribute window, provide required information for the new attribute. For details on how to create an attribute, refer to the **Creating Attributes** topic in the **System Setup** documentation.

Note: Only description attributes are valid for classifications



3. Click **Finish** on the Create Attribute window to close it and return to the Map To dialog.
4. In the attribute selector window, use Browse or Search to select the STEP attribute just created.



5. Check the **Mandatory** option appropriately:

- If **checked**, imported objects must have an attribute value in the file. If an attribute value does not exist for an object, the object is skipped and not imported.
- If **unchecked**, all objects are imported regardless if an attribute value is specified or not.

6. Click **OK** to map this new attribute to the previously selected data column.
7. Complete the mapping and initiate the import.

Attribute Info - Map Inbound

The Attribute Info mapping options are used to create new attributes or update existing attribute definitions. This option is available only when Attribute is selected from the object super type 'Map to' dropdown. The attribute definitions available for import can be found in the attribute editor as defined in the **Attribute Info File Structure Requirements** section below.

This example shows an Excel import file (although any inbound format can be used). The sample file updates the attribute's Type and Calculated parameters for three attributes. Additional attribute info columns can be added to the import file as needed.

A	B	C	D
<ID>	<Name>	<Attribute Type>	<Attribute Calculated>
BrandName	Brand Name	Normal	false
Fit	Fit	Normal	false
ShippingRestrictions	Shipping Restrictions	Property	true

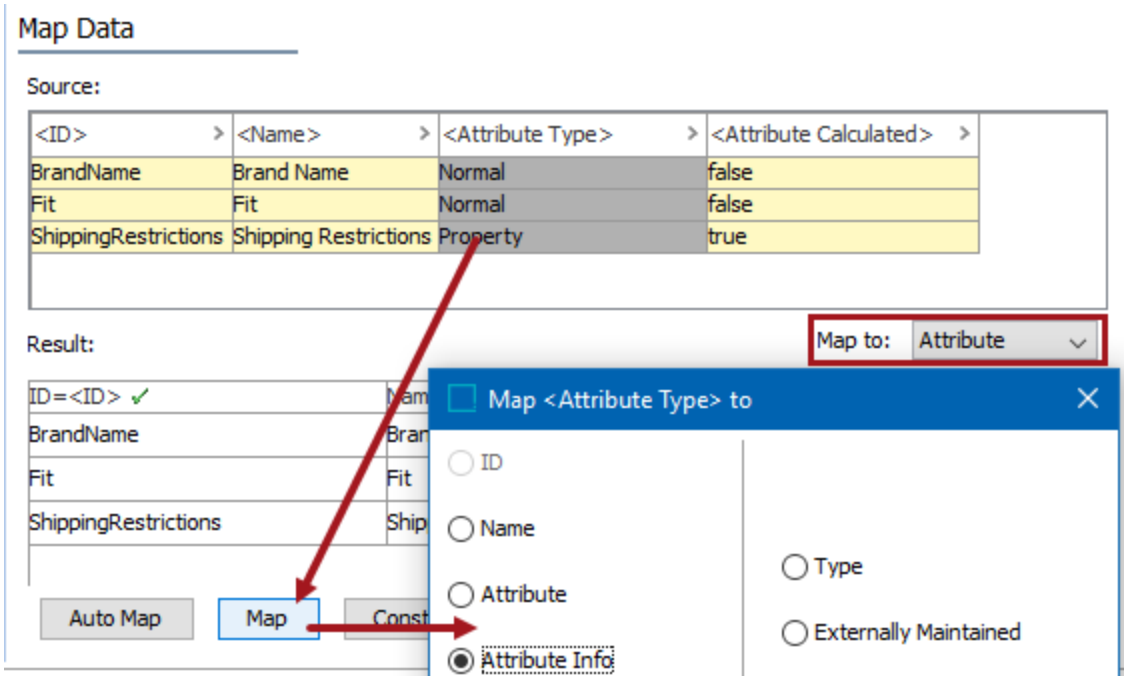
The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Verify Import File

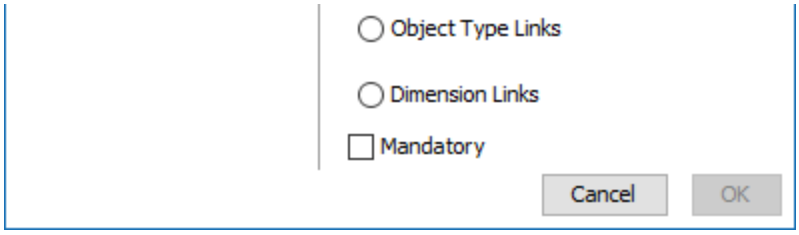
The import file must include the attribute ID or key, and must following the format defined in the **Attribute Info File Structure Requirements** section below.

Map Attribute Info

1. Below the Source section, in the **Map to** parameter, select the **Attribute** super type.
2. Use Auto Map to map the ID, or map it manually. For more information, refer to **ID or Key - Map Inbound**.
3. In the Source section, first select the attribute option column and then click the **Map** button to display the 'Map ... to' dialog. The label of the field is used as the title of this dialog.



4. Select a specific Attribute Info option from the radio buttons on the right panel of the 'Map ... to' dialog. This selection enables the OK button. Each of the available options are defined in the **Attribute Info File Structure Requirements** section below.
5. Check the **Mandatory** option appropriately:
   - If **checked**, imported objects must have a value for the selected Attribute Info in the file. If a value does not exist for an object, the object is skipped and not imported.
   - If **unchecked**, all objects are imported regardless if the selected Attribute Info is specified or not.



6. Click **OK** and the **Result** panel displays your Add Child Override values. A validated data column is marked with a green check mark as shown below.

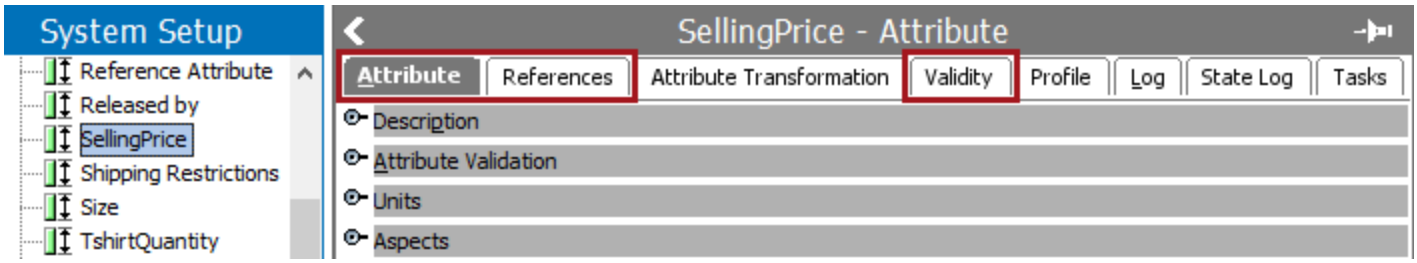
Result: Map to: Attribute

ID=<ID> ✓	Name=<Name> ✓	Attribute Type=<Attribute Type> ✓	Attribute Calculated=<Attribute Calculated> ✓
BrandName	Brand Name	Normal	false
Fit	Fit	Normal	false
ShippingRestrictions	Shipping Restrictions	Property	true

7. Complete the mapping and initiate the import.

Attribute Info File Structure Requirements

On the Map ... to dialog, after selecting the Attribute Info radio button, the following options are displayed on the right side. These options correspond to parameters available in the attribute editor under the identified flippers.



Important: The predefined values shown below are case-sensitive. An error is displayed during mapping if the case is not as defined. For example, a value of 'FALSE' is not allowed when a value of 'false' is expected.

Data on the attribute editor can be set for parameters in the following tabs and flippers as defined below:

1. Attribute Tab, Description flipper
2. Attribute Tab, Attribute Validation flipper
3. Attribute Tab, Units flipper
4. References Tab, Attribute Groups flipper
5. Validity Tab

Note: No updates are made in STEP to existing values when an attribute definition is imported with an empty value.

Attribute Tab Description Flipper Parameters

- **Type** sets an attribute as description when the value is **Property**, or specification when the value is **Normal**.
- **Externally maintained** sets an attribute as externally maintained when the value is **true**, or internally maintained when value is **false**. When updating an existing attribute that has a value for this parameter, special handling is required. For Oracle databases, this action requires single-update mode (SUM), as defined in the **Single-Update Mode** topic. For Cassandra databases, this action uses Lock-free Schema Change (LFSC) functionality, as defined in the **Lock-free Schema Change** topic.
- **Full Text Indexed** sets an attribute as full text indexed when the value is **true**, or not when the value is **false**. When updating an existing attribute that has a value for this parameter, special handling is required. For Oracle databases, this action requires single-update mode (SUM), as defined in the **Single-Update Mode** topic. For Cassandra databases, this action uses Lock-free Schema Change (LFSC) functionality, as defined in the **Lock-free Schema Change** topic.
- **Mandatory** sets an attribute as mandatory when the value is **true** or not when the value is **false**. This field is never output in any export file, and must be added manually to an input file.
- **Completeness** previously used to set the completeness of an attribute but is a legacy option. This field is no longer used by STEP. To learn about attribute and data completeness, and how to calculate them now, refer to the **Data Profiling** documentation.

- **Calculated** sets an attribute as calculated when the value is **true**, or not when the value is **false**. For more information on calculated attributes, refer to the **Calculated Attributes** section of the **System Setup** documentation.
- **Calculated Value Template** can only be used when the Calculated field is set to 'yes' in the workbench (refer to previous bullet) and sets the value template. This field is never output in any export file, and must be added manually to an input file. For details on configuring a calculated value template for importing with Excel, refer to the **Importing Calculated Attribute Value Templates with Excel** topic. For more information on calculated attributes, refer to the **Calculated Attributes** section of the **System Setup** documentation.
- **Calculated Unit Template** can only be used when the Calculated field is set to 'yes' in the workbench. This is only available for attributes with a validation base type of Number, Embedded Number, Fraction, Integer, Number Range, Numeric Text, Numeric Text (exclude tags), or Regular Expression. This field is never output in any export file, and must be added manually to an input file. For more information on calculated attributes, refer to the **Calculated Attributes** section of the **System Setup** documentation.
- **Dimension links** sets the Dimension Dependencies of an attribute. For example, language, country, or market for an attribute. Updating a language dimension to a country dimension is not allowed and displays an error for incompatible dimensions. When adding country dimension to a language dimension attribute, special handling is required. For Oracle databases, this action requires single-update mode (SUM), as defined in the **Single-Update Mode** topic. For Cassandra databases, this action uses Lock-free Schema Change (LFSC) functionality, as defined in the **Lock-free Schema Change** topic.

Attribute Tab Attribute Validation Flipper Parameters

- **Multi Valued** sets an attribute as multivalued when the value is **true**, or single valued when the value is **false**.

Attribute Tab Units Flipper Parameters

- **Default Unit ID** sets a default unit and is only available for attributes with a validation base type of Number, Embedded Number, Fraction, Integer, Number Range, Numeric Text, Numeric Text (exclude tags), or Regular Expression. This determines which unit displays with a check mark in the Default Unit column of the Units flipper. This field is never output in any export file, and must be added manually to an input file.

  When creating an attribute with a default unit via import, the unit must already exist in STEP.

  For more information on validation base type, refer to the **Validation Rules** section of the **System Setup** documentation.

  For more information on units, refer to the **Units** section of the **System Setup** documentation.

  For more information on regular expression, refer to the **Regular Expression** topic in **Resource Materials** online help.

References Tab Attribute Groups Flipper Parameters

- **Attribute group links** specifies under which attribute group this new attribute should be created or linked. The attribute group ID must be included in the input file. When linking to more than one attribute group,

separate ID with a semicolon delimiter. This data is added to the attribute editor's References tab under the In Attribute Groups flipper.

Validity Tab Parameters

- **Object type links** sets the validity of an attribute which defines in which object type level this attribute should be valid. This field is never output in any export file, and must be added manually to an input file. This data is

Attribute Validation Info - Map Inbound

The Attribute Validation mapping options are used to set parameters for attributes. This option is available only when Attribute is selected from the object super type 'Map to' dropdown. The attribute definitions available for import can be found in the attribute editor as defined in the **Attribute Validation Info File Structure Requirements** section below.

This example shows an Excel import file (although any inbound format can be used). The sample file creates multiple attributes and sets the Validation Base Type and ID for an LOV attribute. Additional attribute validation information columns can be added to the import file as needed.

A	B	C	D	E
<ID>	<Name>	<AttributeGroup Ref ID>	<Attribute Validation Base Type>	<LOV ID>
TextAttrb	Text Attribute	ItemBrandInformation	Text	
LOVAttrb	LOV Attribute	ItemBrandInformation		Colour
EmbNumAttrb	Embedded Number At	ItemBrandInformation	Embedded_Number	

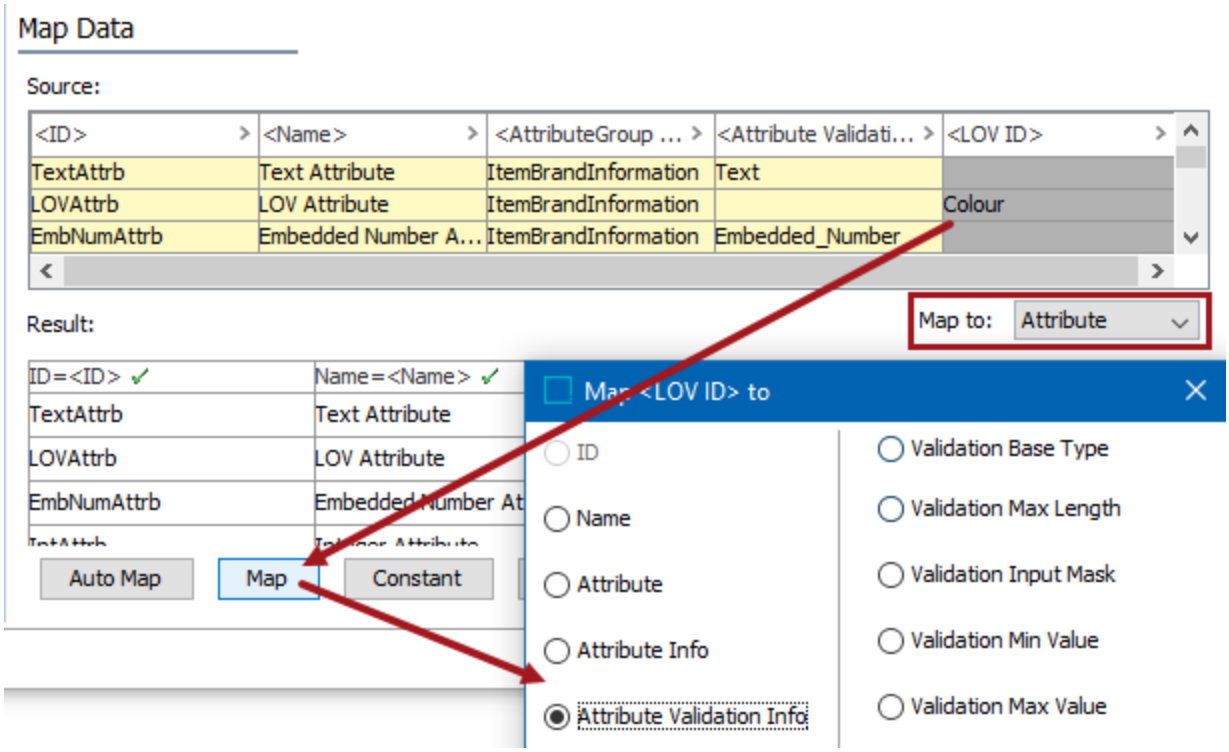
The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Verify Import File

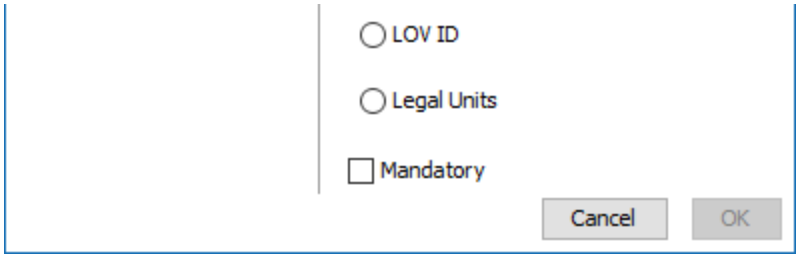
The import file must include the attribute ID, and must follow the format defined in the **Attribute Validation Info File Structure Requirements** section below.

Map Attribute Validation Info

1. Below the Source section, in the **Map to** parameter, select the **Attribute** super type.
2. Use Auto Map to map the ID, or map it manually. For more information, refer to **ID or Key - Map Inbound**.
3. If mapping manually, in the Source section, first select the attribute option column and then click the **Map** button to display the 'Map ... to' dialog. The label of the column is used within the title of the dialog.



4. Select the Attribute Validation Info option, then select a specific type of validation information from the radio buttons on the right panel of the 'Map ... to' dialog to enable the OK button. Each of the available options are defined in the **Attribute Validation Info File Structure Requirements** section below.
5. Check the **Mandatory** option appropriately:
   - If **checked**, imported objects must have a value for the selected attribute information in the file. If a value does not exist for an object, the object is skipped and not imported.
   - If **unchecked**, all objects are imported regardless if the selected attribute information is specified or not.



6. Click **OK** and the **Result** panel displays your attribute information values. A validated data column is marked with a green check mark as shown below.

Result:

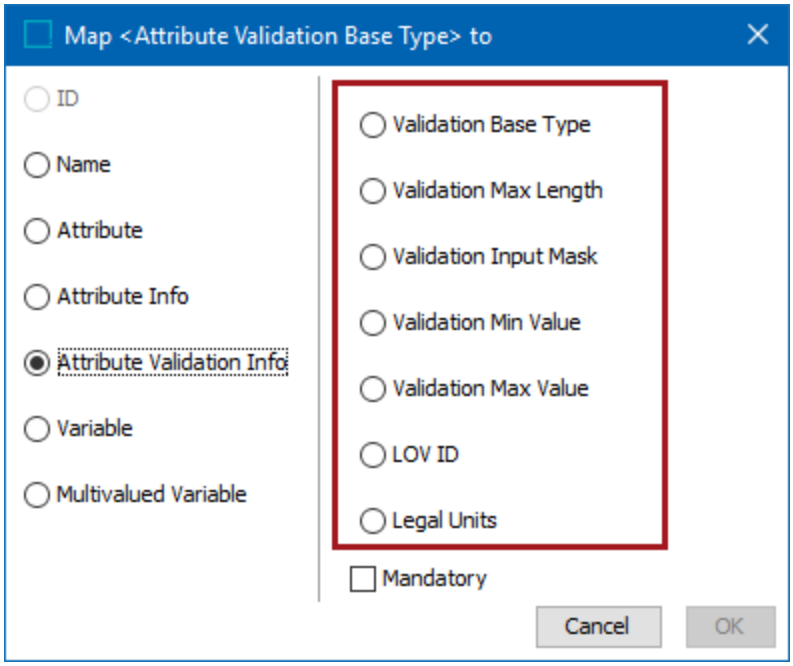
Map to: Attribute

ID=<ID> ✓	Name=<Name> ✓	AttributeGroup Ref ID=<...✓	Attribute Validation Base Type=<Attribute Validation Base Type>✓	Attribute Validation LOV ID=<LOV ID>✓
TextAttrb	Text Attribute	Item Brand Information	Text	
LOVAttrb	LOV Attribute	Item Brand Information		Color
EmbNumAttrb	Embedded Number ...	Item Brand Information	Embedded_Number	

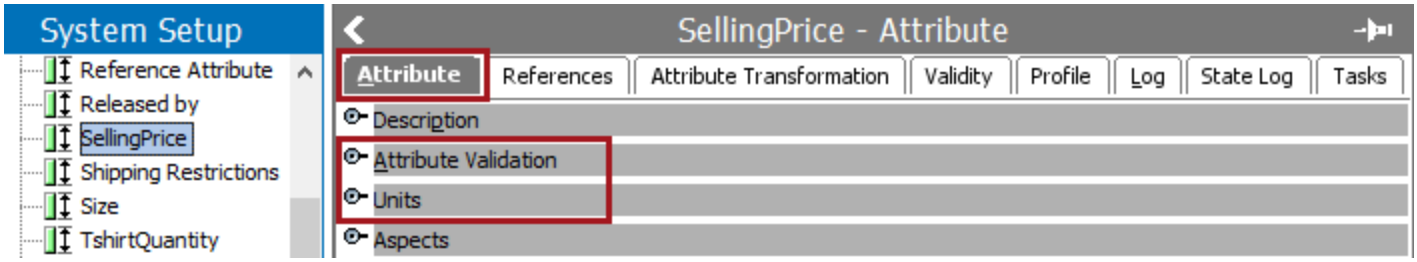
7. Complete the mapping and initiate the import.

Attribute Validation Info File Structure Requirements

On the Map ... to dialog, after selecting the Attribute Validation Info radio button, the following options are displayed.



These options correspond to parameters available in the attribute editor Attribute tab, which can be set in the Attribute Validation flipper and the Units flipper.



Each available parameter for this mapping option is defined below.

1. **Validation Base Type** sets the Validation Base Type parameter on an attribute. The following text is not case sensitive, but is required to set the validation base type. For LOVs, refer to the **LOV ID** section below.

   Note: No spaces are included in types with multiple words. Instead, use the following list to include an underscore (_) or exclude the spaces.

   - Condition
   - Date
   - Embedded_Number
   - Fraction
   - Integer
   - ISODate
   - ISODateTime
   - Number
   - NumberRange
   - Numeric_Text
   - Numeric_Text_Exclude_Tags
   - RegExp
   - Text
   - Text_Exclude_Tags
   - Url

   For more information on validation base type, refer to the **Validation Rules** section of the **System Setup** documentation.

2. **Validation Max Length** sets the Maximum Length parameter which determines the number of characters allowed on an attribute. Primarily, this is used for Text, Text (exclude tags), Numeric Text, and Numeric Text (exclude tags) validation base type attributes.
3. **Validation Input Mask** sets the Mask parameter on an attribute. For more on masks, refer to the **Input Masks** topic in the **System Setup** documentation.
4. **Validation Min Value** sets the Minimum Value parameter for an attribute. Primarily, this is used for Number and Integer validation base type attributes.
5. **Validation Max Value** sets the Maximum Value parameter for an attribute. Primarily, this is used for Number and Integer validation base type attributes.
6. **LOV ID** can only be used when the Validation Base Type is List Of Values (LOV). Specify the ID of a list of values that already exists in System Setup. For more on LOVs, refer to the **List of Value (LOVs)** topic in the **System Setup** documentation.

   Note: The Validation Base Type column will be empty for an LOV since it is identified by the ID.

7. **Legal Units** sets the Units parameter options and is only available attribute with Validation Base Types of number, embedded number, fraction, integer, number range, numeric text, or regular expression. For more

information, refer to the **Units** topic in **System Setup** documentation.

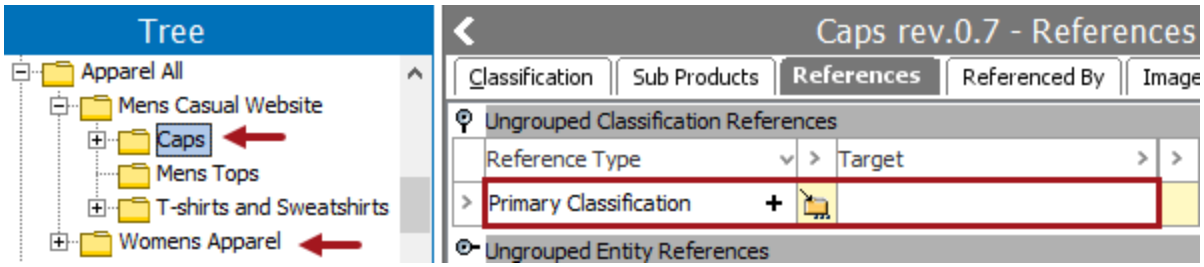
This field is never output in any export file, and must be added manually to an input file. If more than one unit type is being imported for an attribute, then IDs should be separated with a delimiter, for example a semicolon.

Classification Reference - Map Inbound

The classification reference option allows creating references in STEP from classifications to products, classifications, or entities. Mapping a classification reference includes specifying the reference type from those exist in STEP.

The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

The steps below will create a 'Primary Classification' reference between the 'Caps' source classification folder (ID 22585) and the 'Womens Apparel' target classification folder (ID 22584).



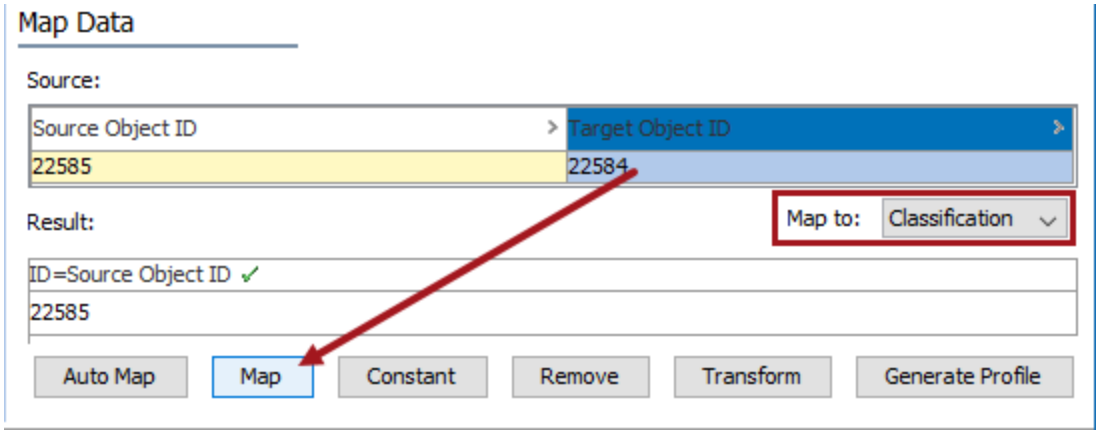
Prerequisites

Prior to import:

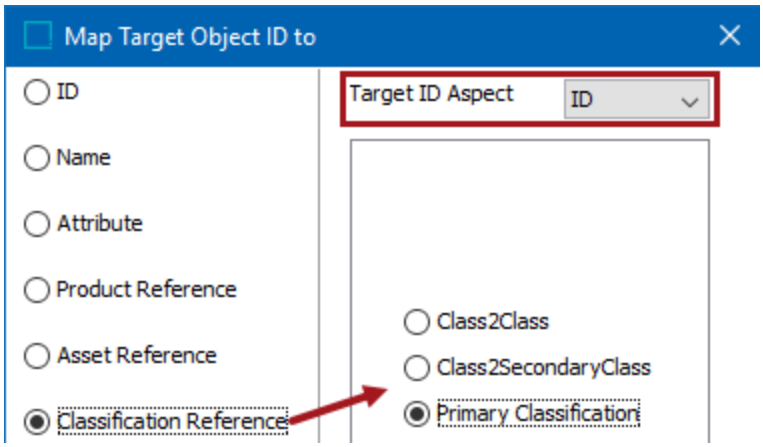
1. The object types and classification types must exist in STEP.
2. The reference type must be valid in STEP for the object types in the import file.
3. The import file must include at least two columns: one with the ID or unique key of the source object, and the other with the ID or unique key of the target object. (The reference type will be specified in the wizard.)
4. The import file must include the classification ID, not the classification name.

Map Classification Reference

1. Below the Source section, use the **Map to** dropdown to select the **Classification** object super type.
2. Map the source ID or key column as the ID using the Auto Map button, or manually. For more information, refer to the **ID or Key - Map Inbound** topic.
3. In the Source section, select the target column of data and click the **Map** button.



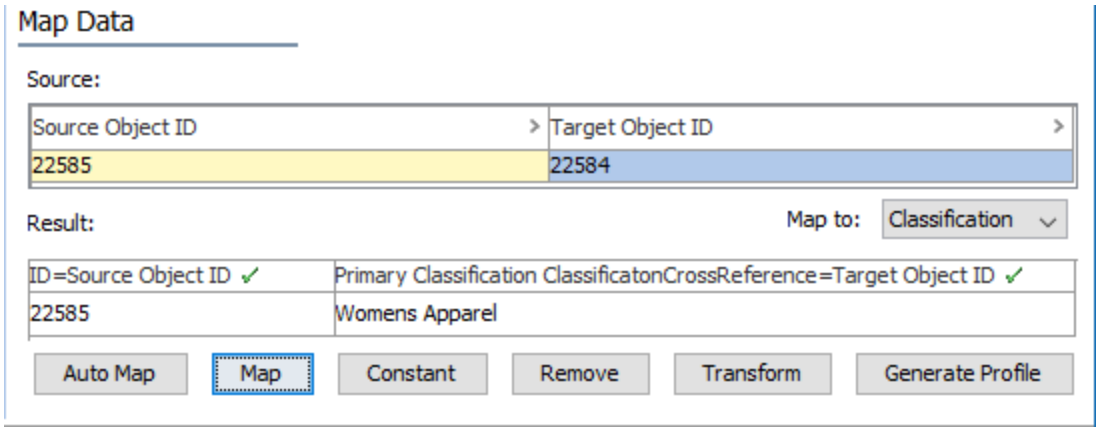
4. In the **Map to** window, select the **Classification Reference** radio button, and then select the reference type. Only existing reference types are listed. You cannot specify or create new reference types during the import.
5. For the **Target ID Aspect** parameter:
   - Select **ID** when the inbound file includes the ID for the source.
   - Select the **Key** from the dropdown when the inbound file includes a key for the target.



Note: Selecting a reference type that is not valid for the objects being referenced results in a background process error.

6. Check the **Mandatory** option appropriately:
   - If **checked**, imported objects must have an attribute value in the file. If an attribute value does not exist for an object, the object is skipped and not imported.
   - If **unchecked**, all objects are imported regardless if an attribute value is specified or not.

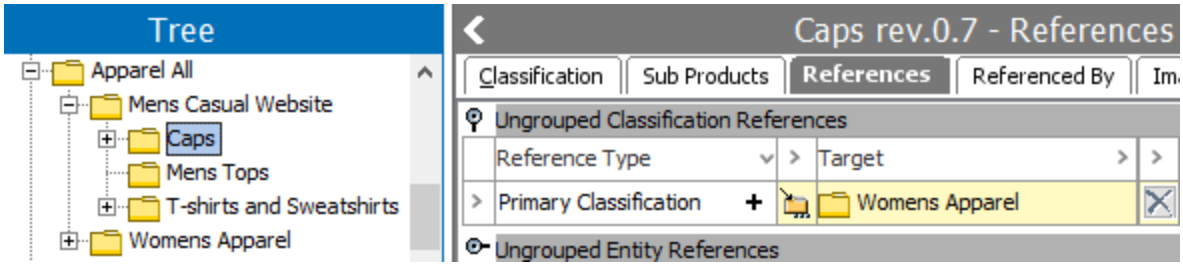
7. Click **OK** and the **Result** panel displays the name of the selected reference target object. A validated data column is marked with a green check mark as shown below.



8. Complete any additional mapping.
9. Initiate the import.

Results

The result of the import is the new 'Primary Classification' reference on the source classification.



Data Container - Map Inbound

The Data Container Attribute option allows you to map data container attributes and references types to specific (new or existing) data containers when mapping to entity or product objects.

The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Map Data Container

1. Below the Source section, in the **Map to** parameter, select the **Entity** or **Product** super type.
2. In the Source section, select the column of data containing the first attribute of the data container, and click the **Map** button.

   In the example below, the 'PM_Model' column will be mapped to the appropriate data container type and attribute.

<ID>	<PM_Name>	PM_Model	PM_Shape	PM_Colors	Product Pa...
100	Franklin	Anthem	Round	Green	Cardboard
101	Franklin	Epic	Round	Green	Cardboard
102	Franklin	Arena	Round	Green	Cardboard
103	Franklin	Touring	Round	Green	Cardboard

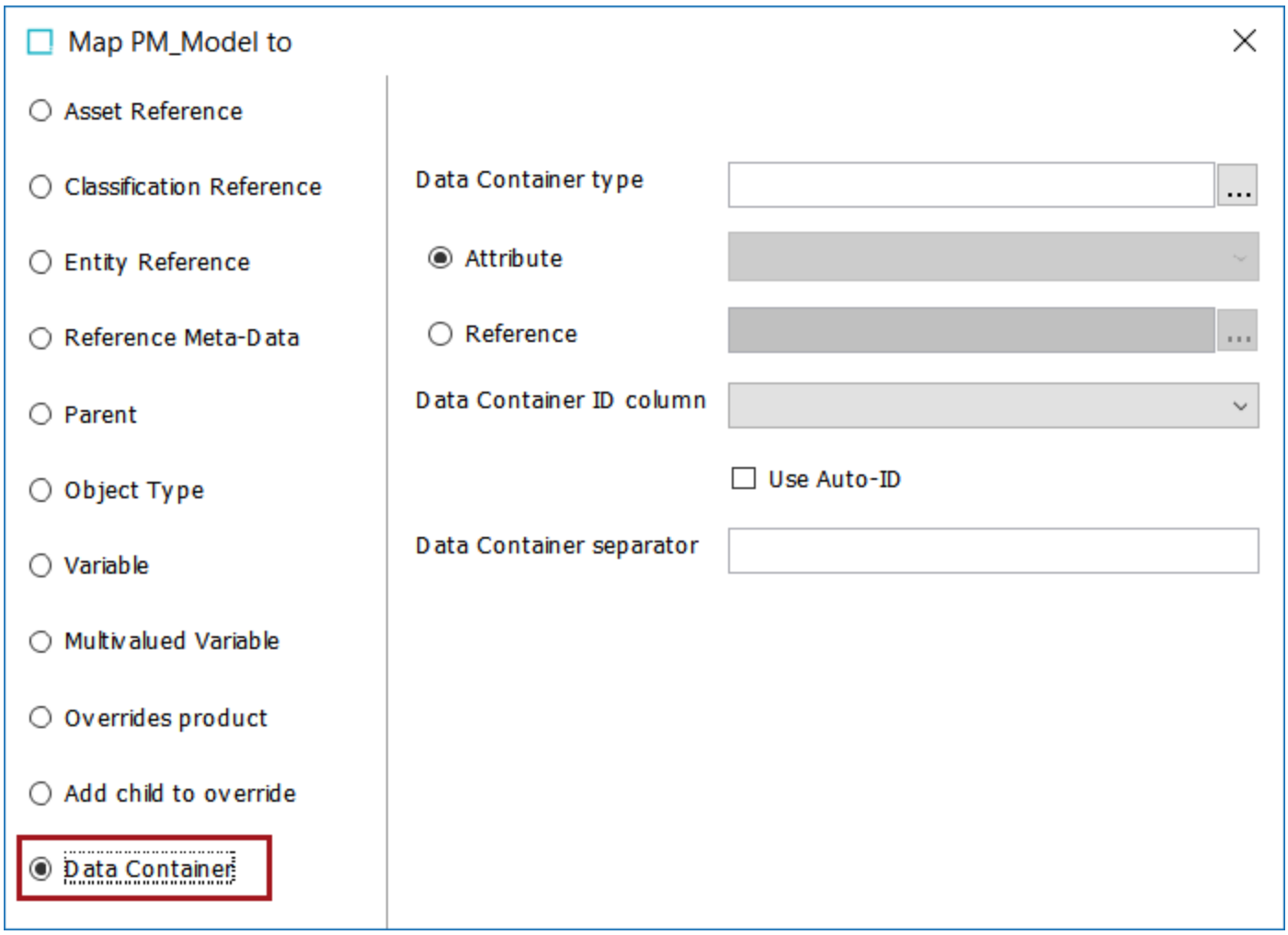
   Unless configured with auto-ID, a data container ID source column (or variable) must also be mapped to the Data Container ID column. In this example, the '<ID>' column represents the data container IDs being imported.

<ID>	<PM_Name>	PM_Model	PM_Shape	PM_Colors	Product Pa...
100	Franklin	Anthem	Round	Green	Cardboard
101	Franklin	Epic	Round	Green	Cardboard
102	Franklin	Arena	Round	Green	Cardboard
103	Franklin	Touring	Round	Green	Cardboard

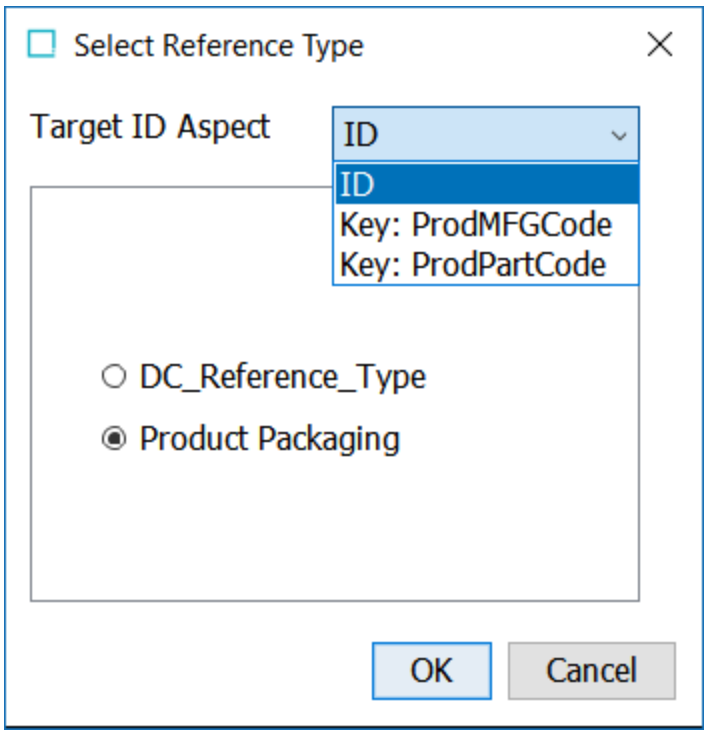
   Explicitly setting an ID for each data container in this way allows for subsequent imports to update the data container instances. If data container IDs are not provided as a source column in the imported file, a variable may be created to serve in its place. This is especially helpful when importing and updating single instance data containers.

   For more information on variables, refer to the **Variable - Map Inbound** section of the **Data Exchange** documentation.

3. In the 'Map ... to' dialog, select the **Data Container** radio button. This option is only available when the Entity or Product super type is specified in the 'Map To' dropdown.



4. For **Data Container type**, Click the ellipsis button (...) and select the data container type associated with the attribute column being mapped.
5. With the attribute radio button engaged, select the **Attribute** using the dropdown. Only attributes valid for the data container type mapped in the previous step will display.
6. With the reference radio button engaged, click the ellipsis button (...) and select the reference type that should be associated with the source column being mapped. Use the appropriate target ID aspect for identifying the target object of the selected reference type. Only reference types valid for the data container type mapped in the previous step will display.



Note: If an attribute and/or reference type that was selected as a data container key for the selected data container type is not mapped and the user selects 'Next' to advance to the 'Identify Objects' dialog in the Import Manager, a warning dialog pop-up will appear. For more information on data container keys, refer to the **Data Container Keys** topic in the **Data Containers** documentation.

7. The **Match value ID in LOV** parameter (shown in the image below), is *only* displayed if the data container attribute holds an LOV where:
   - Users are not allowed to add values (a hard domain LOV) **and**
   - The LOV uses Value IDs

   In this case, the attribute column in the import file can include the value IDs. Check the box if the IDs in the Excel sheet should be matched against the IDs in the LOV, and values applied based on the match. For more information on hard domain LOV, refer to the **List of Values (LOVs)** section of the **System Setup** documentation.
8. For **Data Container ID column**, use the dropdown to select the header of the data container ID column applicable to the data container attribute being mapped.

   Note: All source column headers and configured variables are listed in the dropdown.
9. For the **Use Auto-ID** parameter, check the box to assign automatic IDs to data containers if you do not want to manually map an ID column. If a data container ID column is selected this box will become unchecked.
10. Add a separator symbol in the **Data Container separator** field, if applicable. This only impacts data container types configured to allow multiple data containers. The separator signals to the STEP system that there are multiple data containers for a single entity so that attributes are created or updated accurately.

Note: All attributes being mapped to a specific data container type must use the same ID column or an error will display.

11. Check the **Mandatory** option appropriately:
    - If **checked**, imported objects must have a value in the file. If a value does not exist for an object, the object is skipped and not imported.
    - If **unchecked**, all objects are imported regardless if a value is specified or not.

12. Click **OK** and the **Result** panel displays your selection.

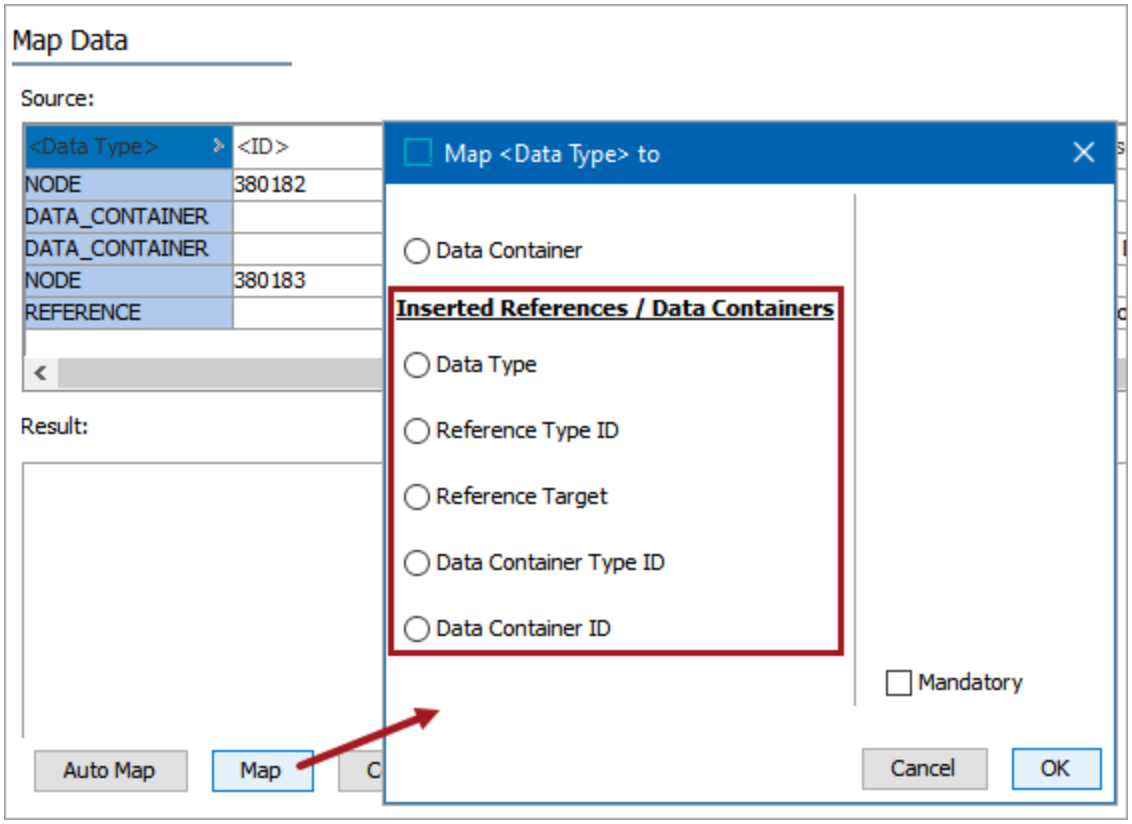
Result:

ProdModel.PM_Co...	ProdModel.PM_Mo...	ProdModel.Produc...	ProdModel.PM_Sh...
Green	Anthem	Cardboard	Round
Green	Epic	Cardboard	Round
Green	Arena	Cardboard	Round
Green	Touring	Cardboard	Round

Data Container ID - Map Inbound

Representing data container objects in separate rows provides a simplified layout with clear relationships between the objects. The 'Data Container ID' source can be used to identify existing data container objects by ID or create new ones with a specific ID. Alternatively, values for Data Container Key attributes can be used to create new data containers upon import.

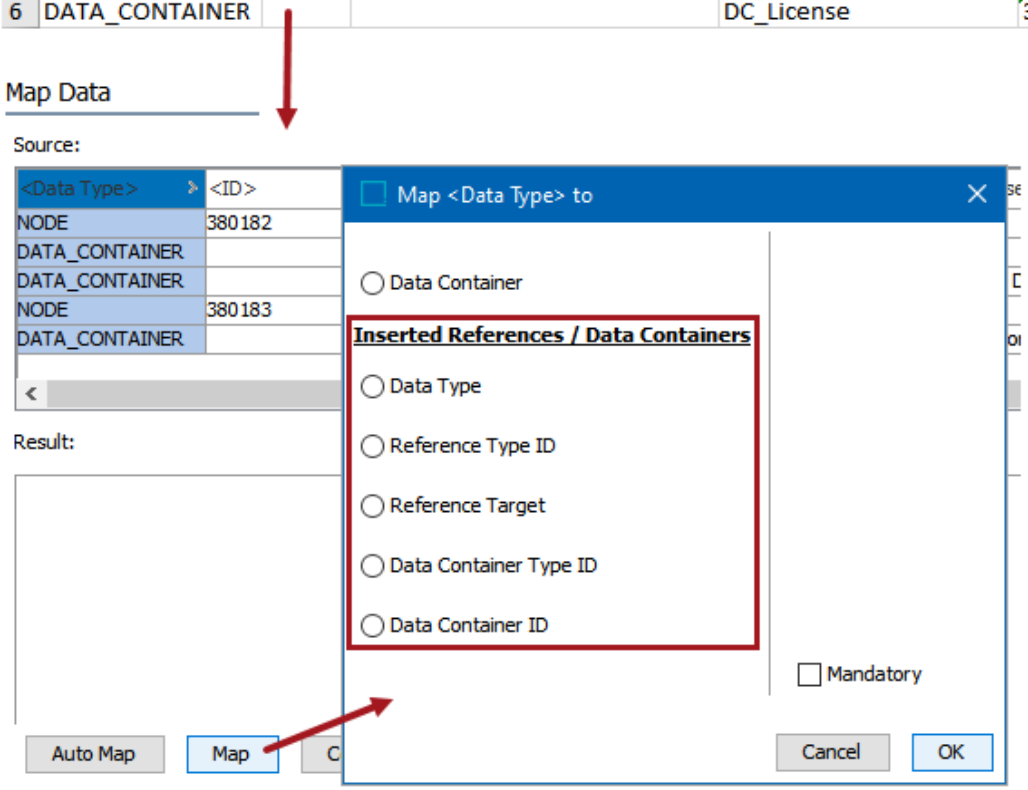
This source mapping option is available in the 'Inserted References / Data Containers' section on the 'Map To' dialog. For details on using the options in this group, refer to the **Reference Type ID - Map Inbound** topic and the **Data Container Type ID - Map Inbound** topic.



Data Container Type ID - Map Inbound

Simplified Excel or CSV file layouts use a single row for each data container owned by the imported product or entity object. This type of file is mapped with the 'Inserted References / Data Containers' section of the 'Map...to' dialog. Refer to the **Prerequisites** section below for required use of this mapping option.

	A	B	C	D	E	F
1	<Data Type>	<ID>	<Name>	<Data Container Type ID>	<Data Container ID>	License Type
2	NODE	380182	Crayola Colored Pens, 12 Count			
3	DATA_CONTAINER			DC_License	380207	Trial
4	DATA_CONTAINER			DC_License	380206	Fixed Duration
5	NODE	380183	Crayola Colored Pens, 24 Count			
6	DATA_CONTAINER			DC_License	380214	Temporary



The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Prerequisites

References and data containers owned by the imported object can be included in the same file. Refer to the **Reference Type ID - Map Inbound** topic for detail on importing data containers.

The following must be included in the import file and mapped during import configuration:

1. The **Data Type** source with one of the following expected values on each row:

   - NODE - the row contains the product, asset, classification, or entity being imported. NODE must be followed by row(s) for related references and data containers.

     Important: The NODE row is the object being imported. The references and data containers below a NODE row in the file are added to or updated on the imported NODE object.

   - REFERENCE - the row contains data for a reference owned by the preceding NODE.
   - DATA_CONTAINER - the row contains data for a data container owned by the preceding NODE.

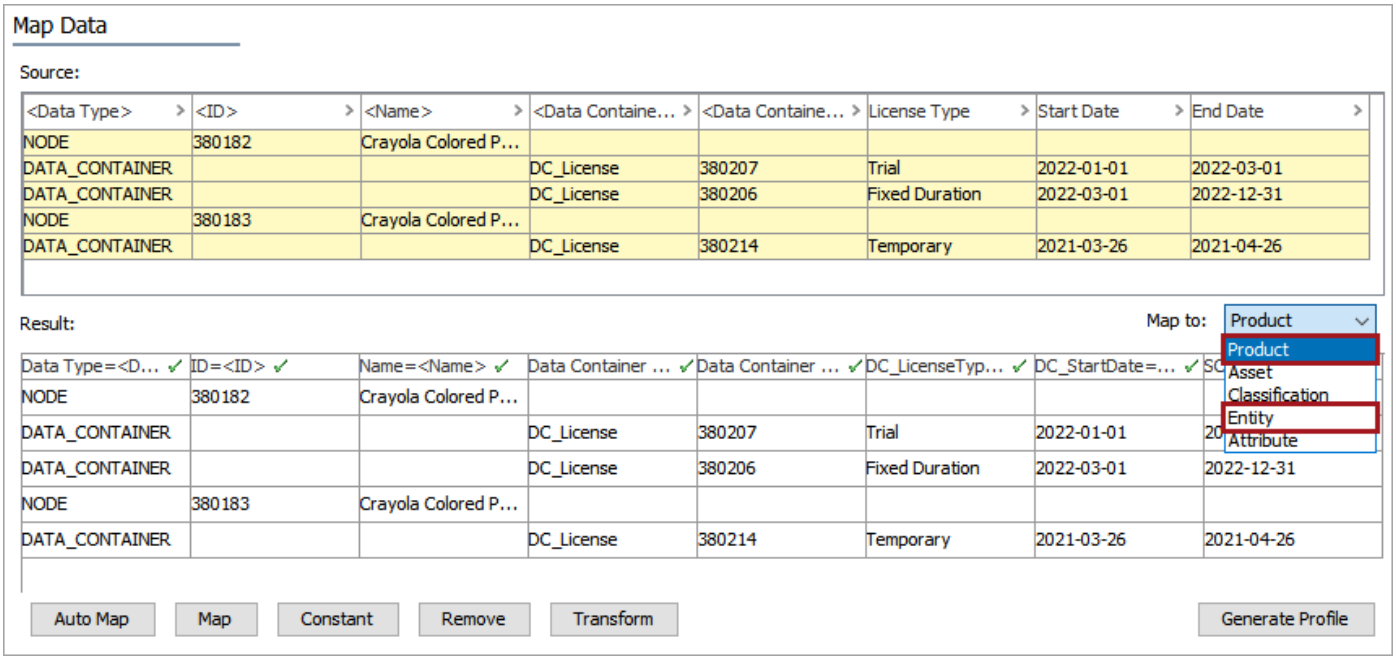
   Note: Other values in the 'Data Type' column of the import file are imported as NODE and are processed as an object to import.

2. The **ID** of the NODE.
3. The **Data Container Type ID**, which is the STEP ID of the data container type, is required when data container objects are represented in a separate row to identify the data container type. Data Container Type ID must be used in combination with Data Type.
4. The **Data Container ID**, the STEP ID of the data container object. Alternatively, values for the Data Container Key attributes can be supplied.

Map Inserted Data Containers

Using the following steps to map the Excel or CSV file for import:

1. Below the Source section, use the **Map to** dropdown to select product or entity as the super type for the objects being imported. Data Containers are not valid for assets, classifications, or attributes. Refer to the **Data Containers** topic in the **System Setup** documentation.



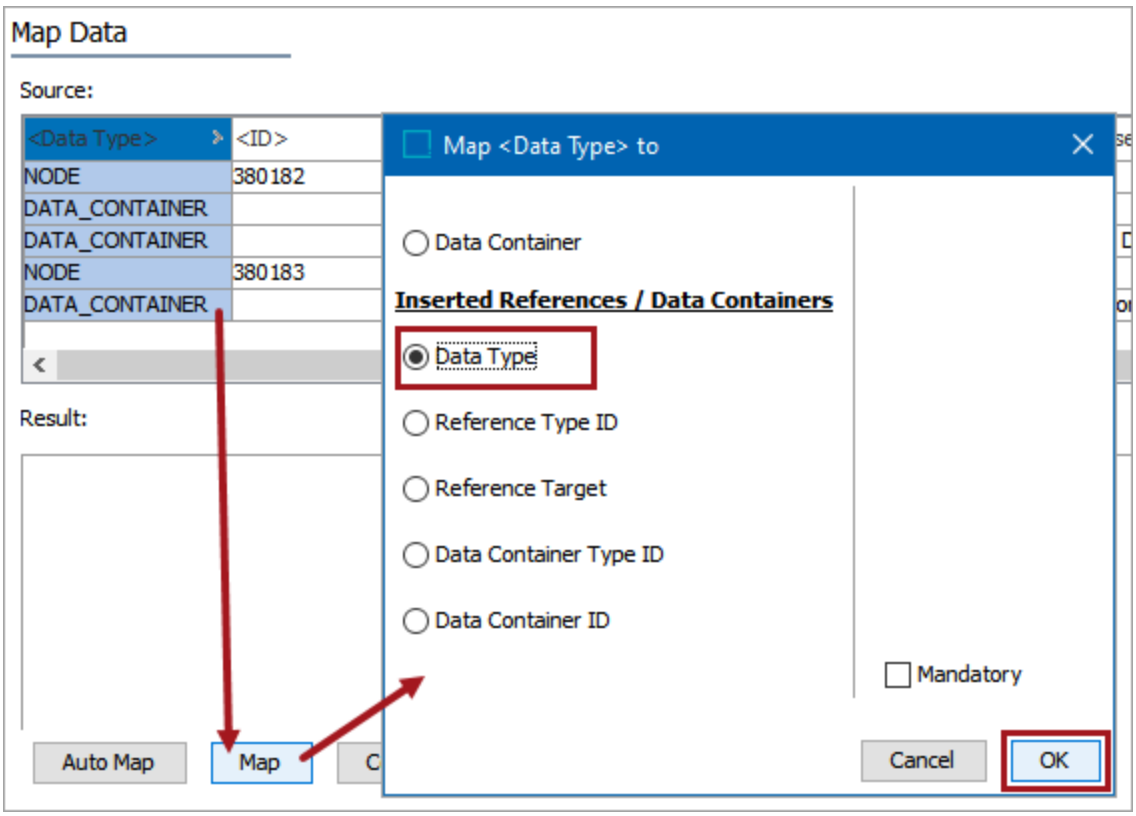
2. Map the source information in the import file:

   - Columns that include Predefined Header Text can be automatically mapped as defined in the **Inbound Map Data - Auto Map** topic.
   - Columns can also be manually mapped as defined in the following steps.

   Note: Although the <Data Owner Node> can be mapped during output, it is not needed during import. Refer to the **Data Owner Node - Data Source Outbound** topic.

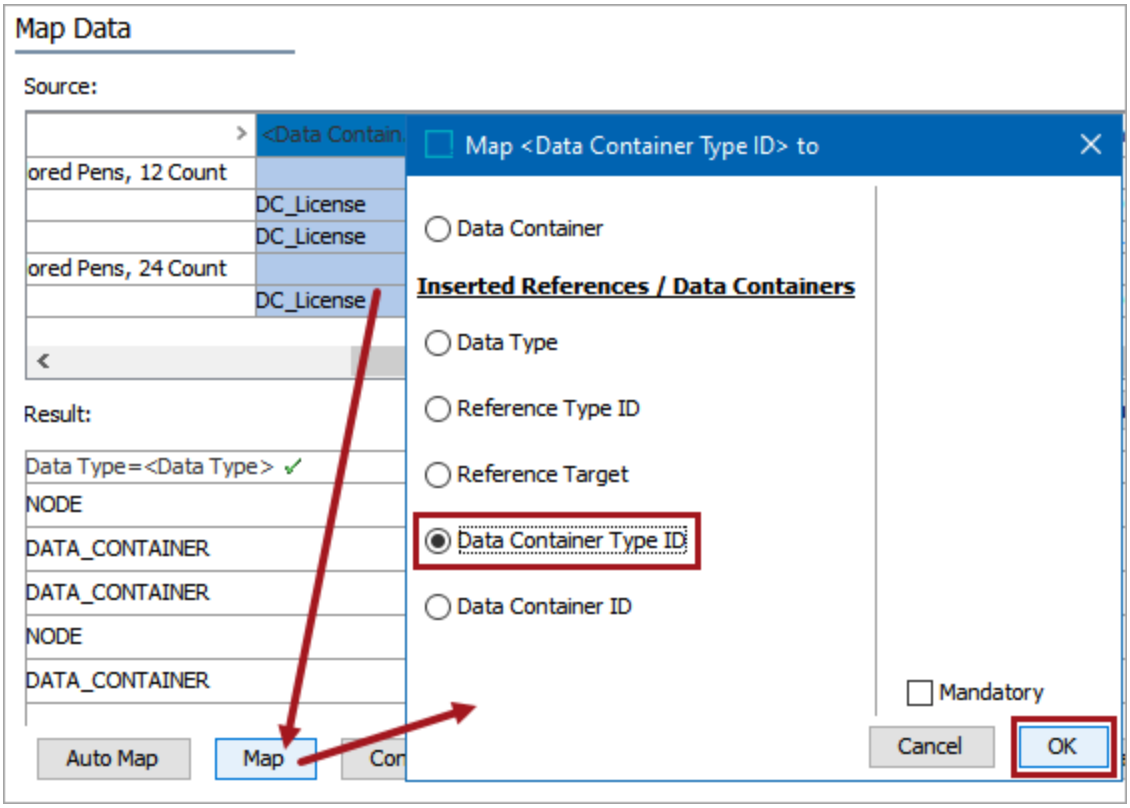
3. Map (or verify the mapping of) the required **Data Type** column (refer to the **Prerequisites** section above for requirements) and check the **Mandatory** option appropriately.

   - If **checked**, imported objects must have an attribute value in the file. If an attribute value does not exist for an object, the object is skipped and not imported.
   - If **unchecked**, all objects are imported regardless if an attribute value is specified or not.



Important: The 'Data Type' source must be mapped to successfully import data containers owned by the node. No values are shown in the Result panel for data containers until the 'Data Type' source is mapped.

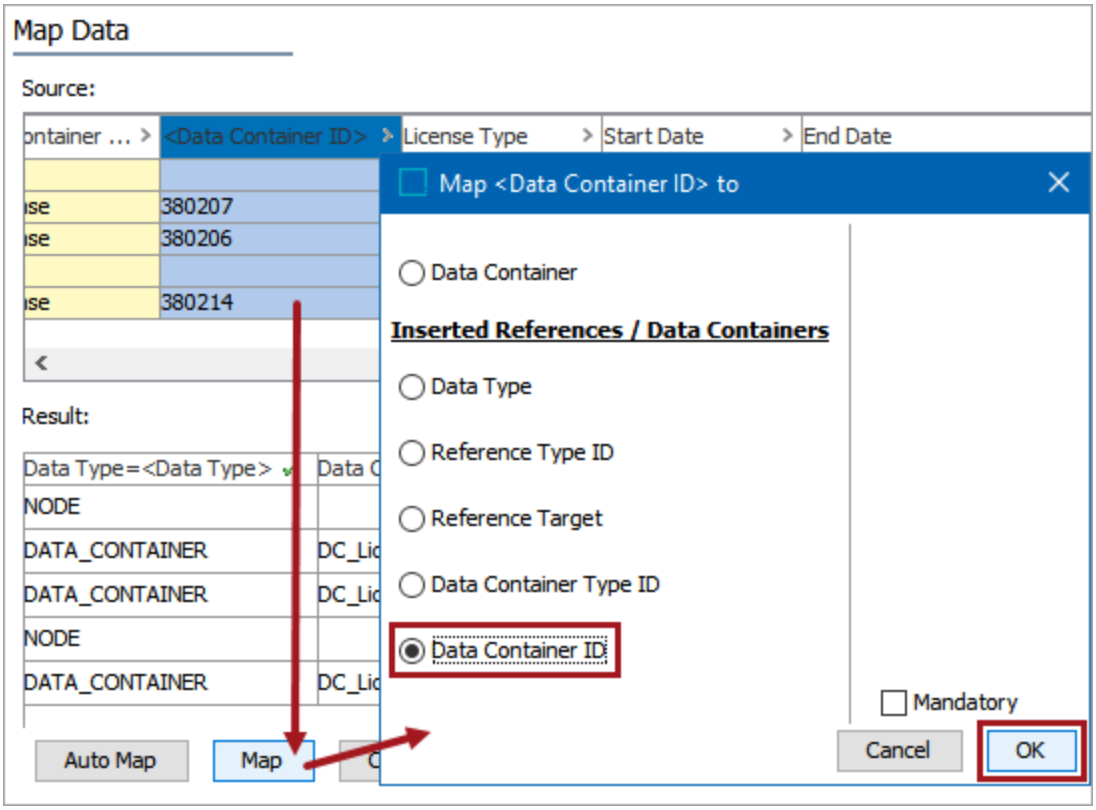
4. Map or verify the mapping of the required **Data Container Type ID** column and check the **Mandatory** option appropriately (defined in a previous step).



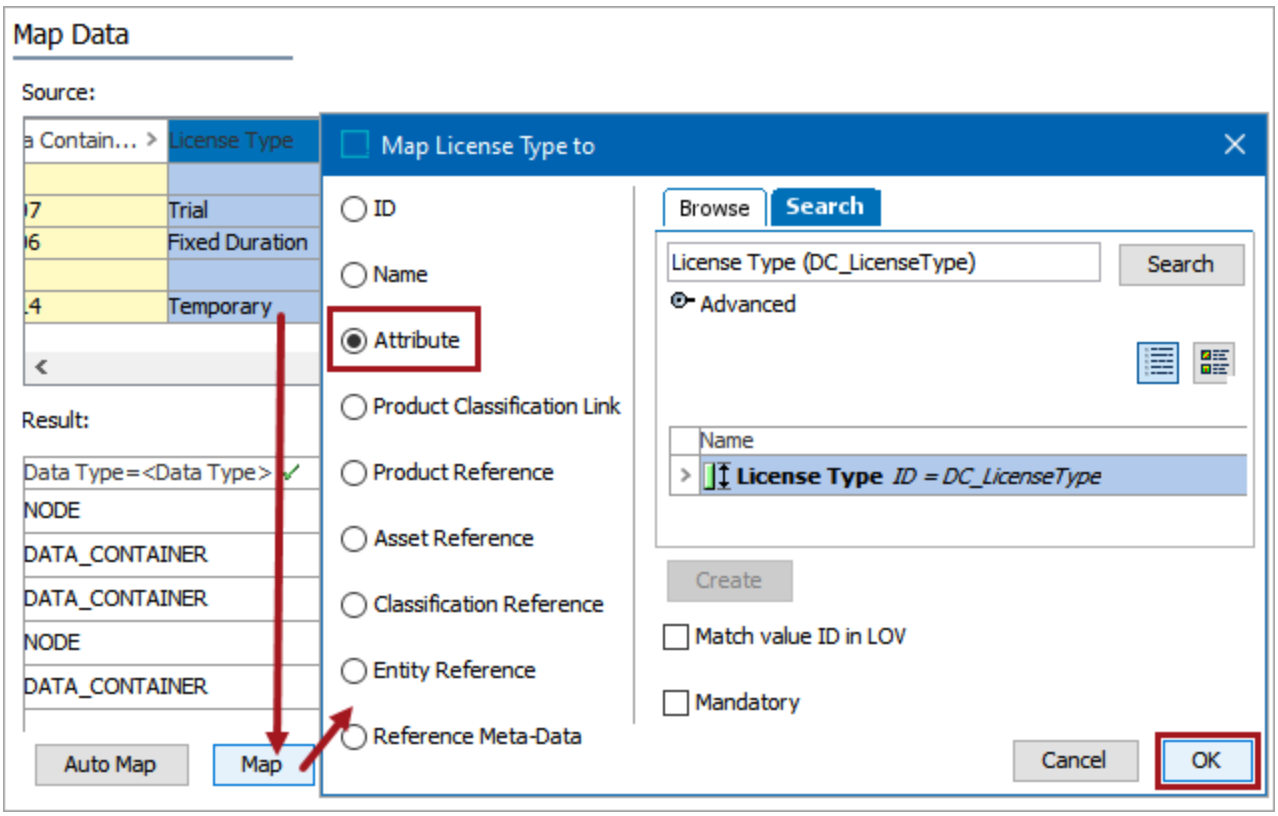
5. If necessary, map (or verify the mapping of):

   - No value is required to import new data containers if the data container object uses an ID Pattern (as shown in the **Examples** section below). Refer to the **Autogenerate Using Name Pattern and ID Pattern** topic in the **System Setup** documentation.
   - A value is required to import new data containers for types without an ID Pattern or to update existing data containers. Map either the **Data Container ID** column or the **Data Container Key** attributes (as shown in the metadata attributes step below), which is defined in the **Data Container Keys** topic.

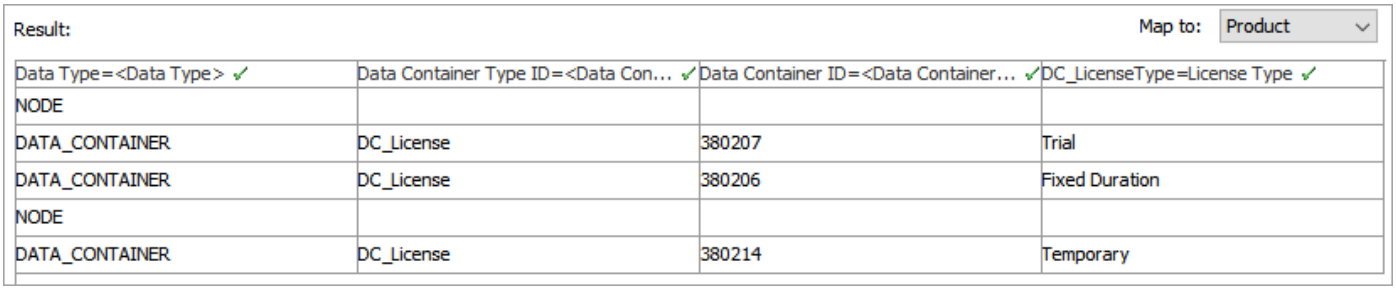
   and check the **Mandatory** option appropriately (defined in a previous step).



6. Map (or verify the mapping of) the desired metadata using the Attribute option on the 'Map...to' dialog. With this single mapping, values are imported for the selected attribute on all NODE, REFERENCE, and DATA_CONTAINER rows of the import file.



7. Review that the **Result** panel displays the inserted references as expected. Validated data columns are marked with a green check mark.



Result: Map to: Product

Data Type=<Data Type> ✓	Data Container Type ID=<Data Con... ✓	Data Container ID=<Data Container... ✓	DC_LicenseType=License Type ✓
NODE			
DATA_CONTAINER	DC_License	380207	Trial
DATA_CONTAINER	DC_License	380206	Fixed Duration
NODE			
DATA_CONTAINER	DC_License	380214	Temporary

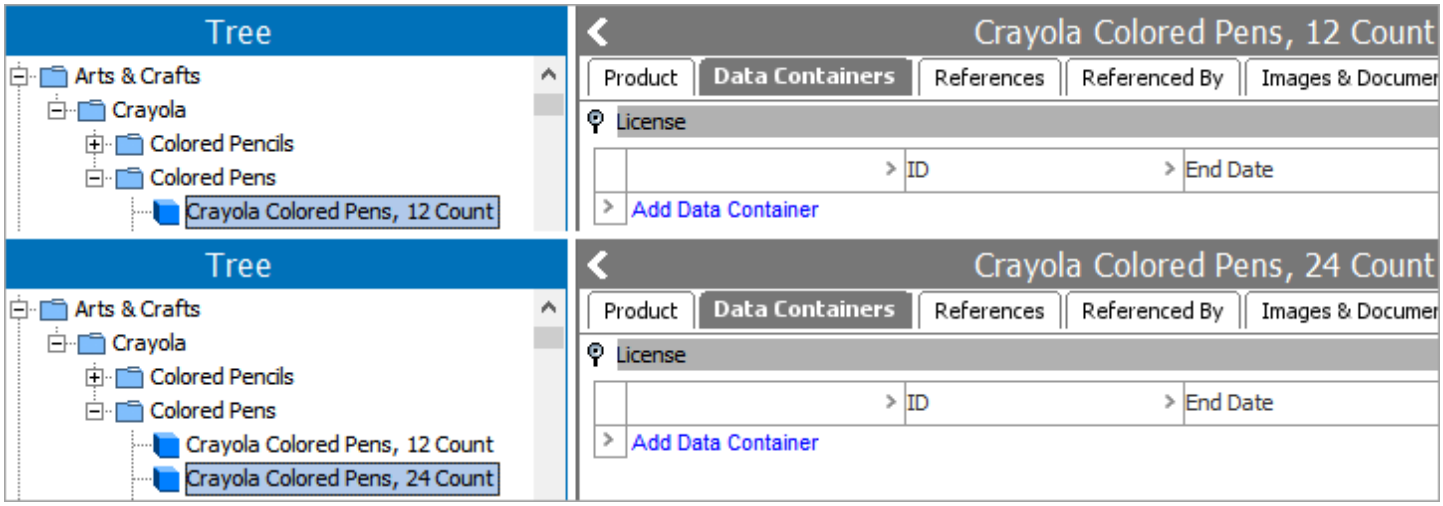
8. Complete any additional mappings or transformations.
9. Initiate the import.

Example

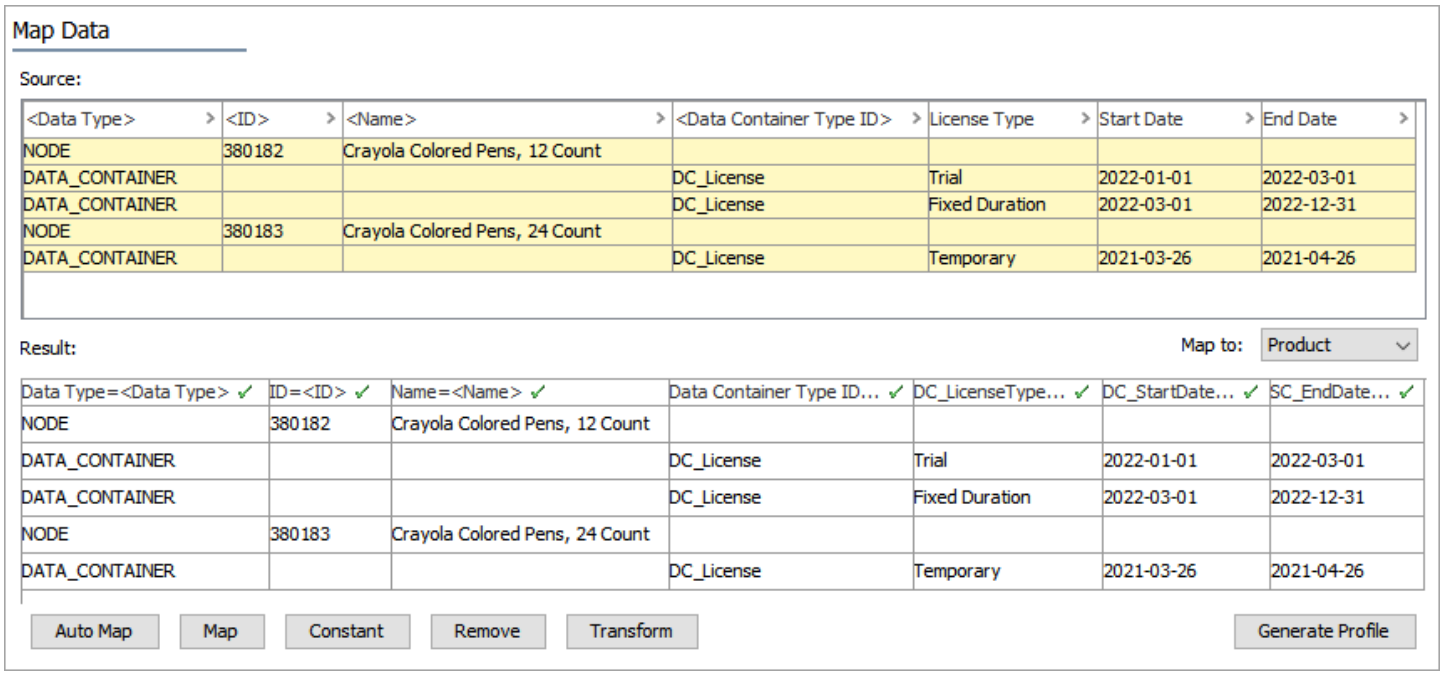
This sample file is used to create new data containers with metadata for the NODE object. The 'DC_License' data container uses an 'ID Pattern' so the <Data Container ID> information is not included in the file.

	A	B	C	D	E	F	G
1	<Data Type>	<ID>	<Name>	<Data Container Type ID>	License Type	Start Date	End Date
2	NODE	380182	Crayola Colored Pens, 12 Count				
3	DATA_CONTAINER			DC_License	Trial	2022-01-01	2022-03-01
4	DATA_CONTAINER			DC_License	Fixed Duration	2022-03-01	2022-12-31
5	NODE	380183	Crayola Colored Pens, 24 Count				
6	DATA_CONTAINER			DC_License	Temporary	2021-03-26	2021-04-26

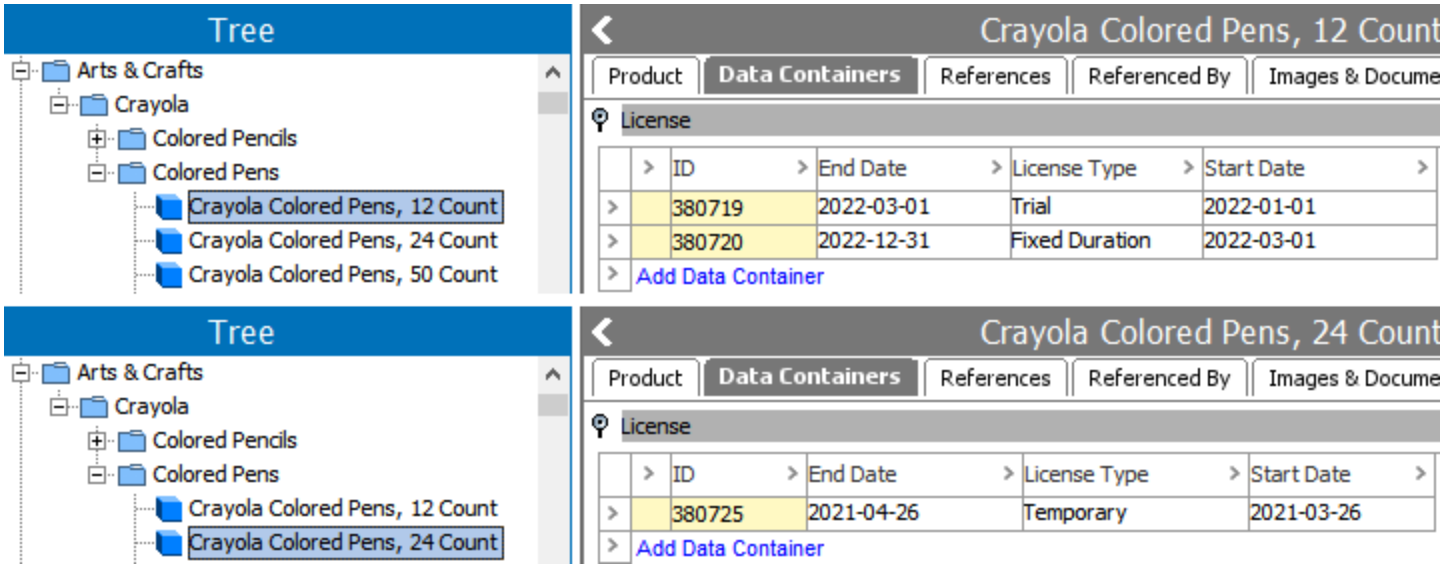
Before the import, the product has no values for the 'License' (DC_License) data container.



The import is set to use the product object super type, the sources are mapped, and the import is invoked.



Once the import background process completes without error, the NODE objects display the new data containers including their metadata values.

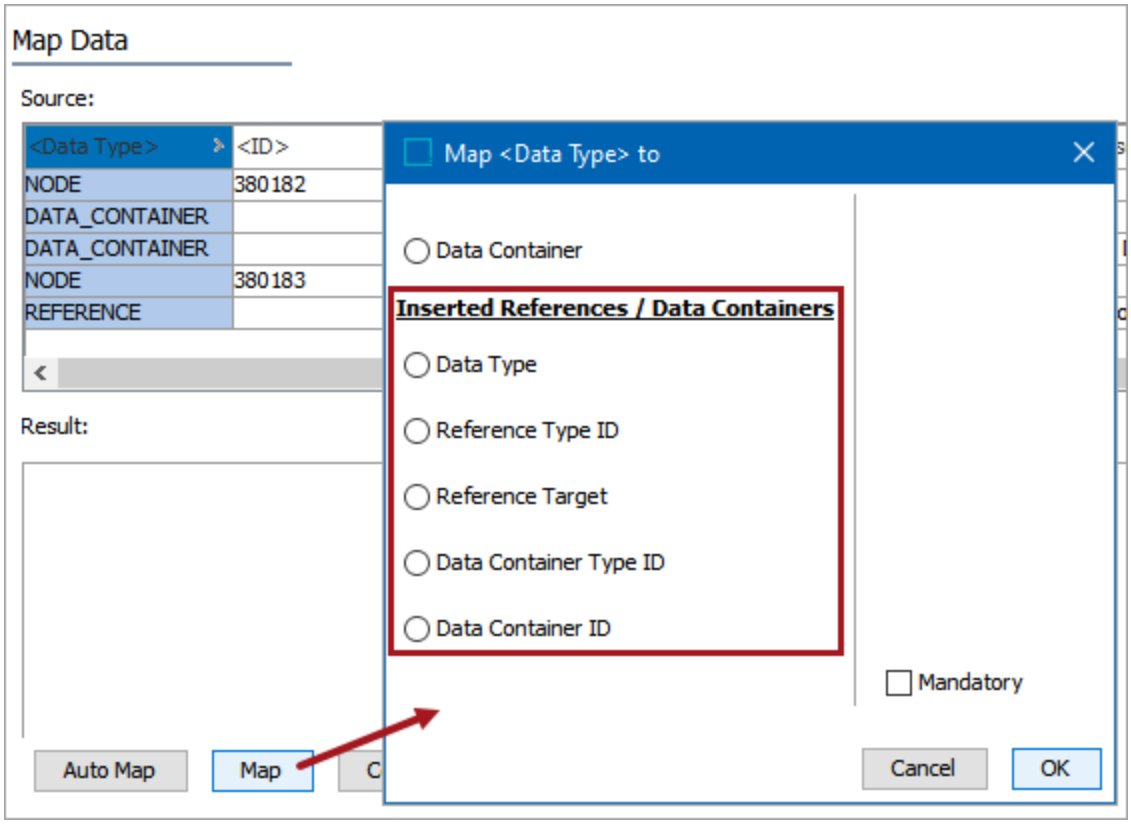


Data Type - Map Inbound

Mapping the 'Data Type' source displays the text from the import file to identify the relationship between the rows. 'Data Type' is included in the 'Inserted References / Data Containers' section on the 'Map To' dialog. This group of options expects each object in the import file to be on a separate row, which provides a simplistic layout with clear relationships between the objects.

The 'Data Type' option is required when data containers and references are represented in separate rows and identifies the type of data in the row. Valid values are:

- NODE - a row representing data for a STEP node.
- DATA_CONTAINER - a row with data for a data container object owned by the preceding NODE. For further mapping requirements, refer to the **Data Container Type ID - Map Inbound** topic.
- REFERENCE - a row with data for a reference or classification product link owned by the preceding NODE. For further mapping requirements, refer to the **Reference Type ID - Map Inbound** topic.



Delete Values - Map Inbound

When updating existing objects, use the **[delete]** command to remove values for attributes or the name of the object that exists in STEP. The delete command works for both tabular and XML formats.

The formats supported are:

- CSV Format
- Excel Format
- FixedWidth Format
- BMEcat Format
- BMEcat2005 Format
- Generic XML Format
- IDoc MATMAS 05 Format

Note: For information on deleting values via STEPXML, refer to the **Delete Values in STEPXML** topic.

Keep in mind the following points:

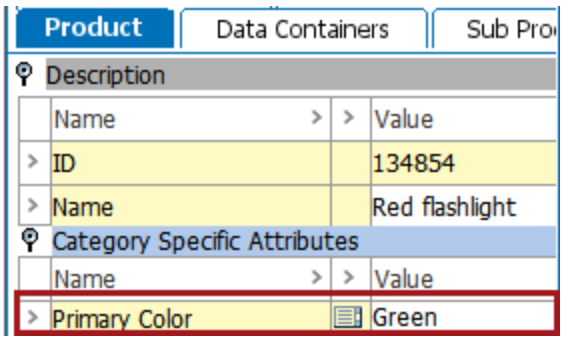
- The delete command is case-sensitive and must be included in the import file as: **[delete]**
- The object name can be deleted for attributes, products, assets, classifications, and entities.
- The attribute value for most Validation Base Types can be deleted, excluding 'Condition' and 'Calculated Attribute.' Only the value of an attribute is deleted, and not the object itself.
- If the value being deleted was inherited, the result is not a blank field, but restoration of the inherited value.
- Selected multivalued attributes can be deleted by mapping to 'Multivalued Variable.' For more information, refer to the **Multivalued Variable - Map Inbound** topic.

Map to Delete a Value

1. In the inbound file, add the **[delete]** command for the cells where existing data should be deleted.
2. Map the inbound data following the steps required for the type of data. Details for mapping each object type are included in the **Inbound Map Data - Map** topic.
3. Initiate the import.

Excel Example

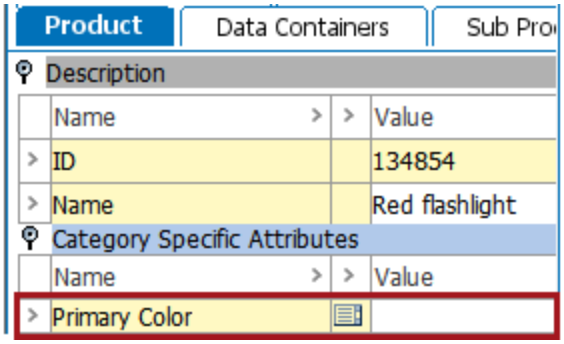
The following object has an attribute, 'Primary Color,' with a value of 'Green', that should be removed.



By importing this tabular format import file, the attribute value is removed by mapping to the appropriate STEP object.

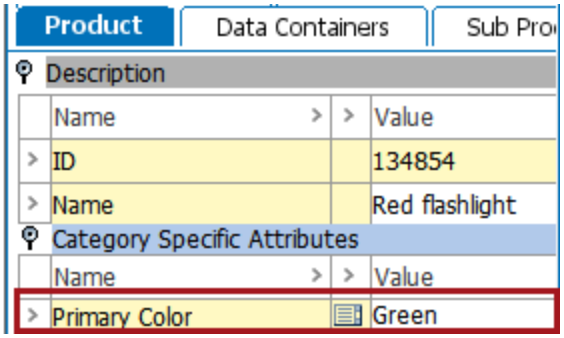
	A	B
1	<ID>	<PrimaryColor>
2	134854	[delete]

After the successful mapping and import, the 'Primary Color' attribute value is blank.



Generic XML Example

By importing the following Generic XML input document and using the corresponding Generic XML template, the existing value of the attribute 'PrimaryColor' is removed for the mapped column.



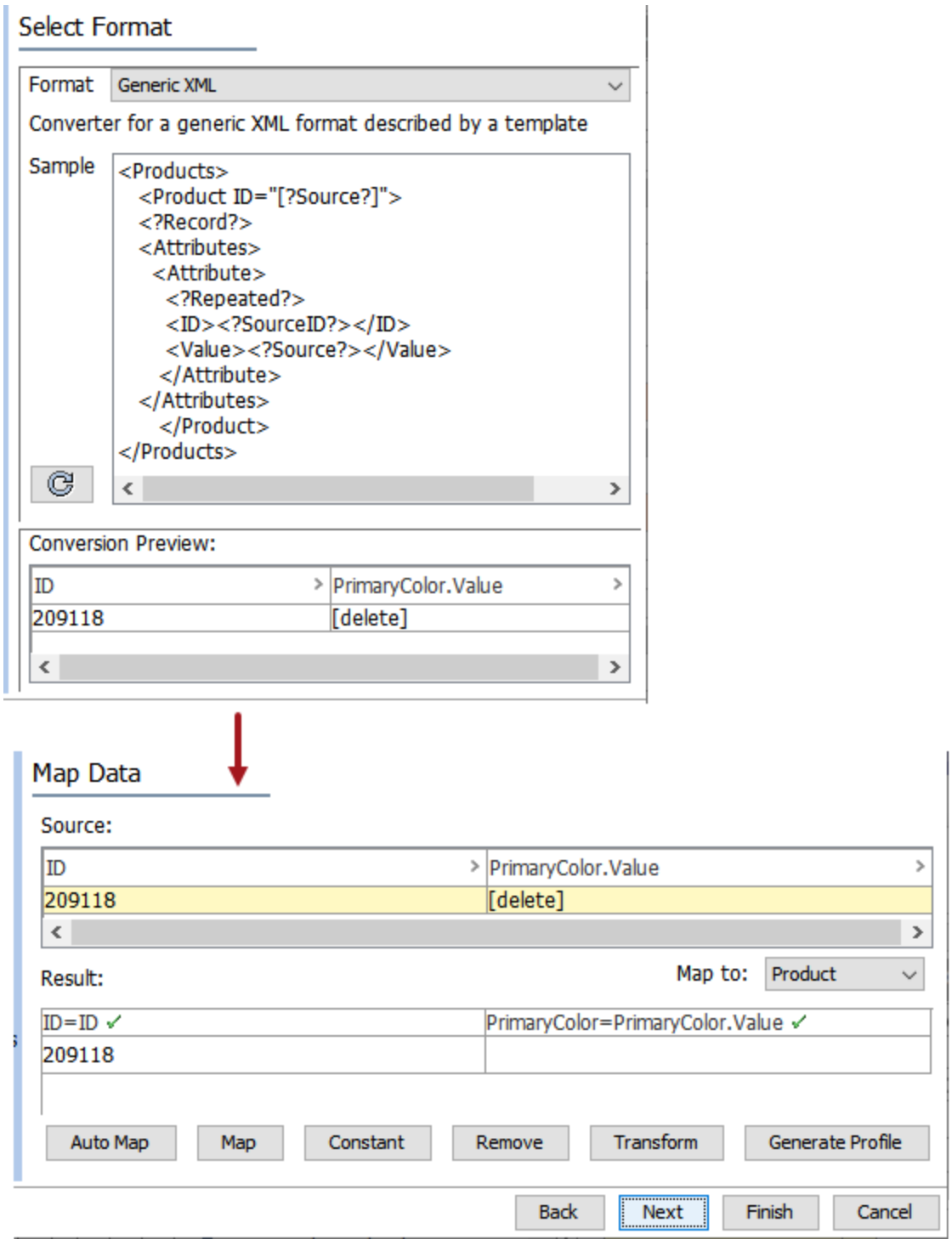
Generic XML input document:

```
<Products>
   <Product ID="209118">
      <Attributes>
        <Attribute>
          <ID>PrimaryColor</ID>
          <Value>[delete]</Value>
         </Attribute>
      </Attributes>
   </Product>
</Products>
```

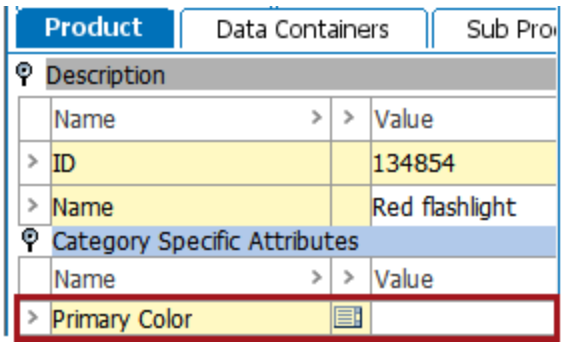
Generic XML template (pasted into the Sample field of the Select Format step on Import Manager):

```
<Products>
   <Product ID="[?Source?]">
   <?Record?>
   <Attributes>
     <Attribute>
       <?Repeated?>
       <ID><?SourceID?></ID>
       <Value><?Source?></Value>
      </Attribute>
   </Attributes>
      </Product>
</Products>
```

Using the above files, the Import Manager Conversion Preview shows the delete command, and the Result section of the Map Data steps shows the value is removed.



After the successful mapping and import, the 'Primary Color' attribute value is blank.


Product
Data Containers
Sub Pro
Description
Name
Value
ID
134854
Name
Red flashlight
Category Specific Attributes
Name
Value
Primary Color

Entity Reference - Map Inbound

Entity references can be created for products, classifications, or entities. The available reference types are based on the entity reference types that exist in STEP. The mapping steps are the same, regardless of the reference type needed. Some examples of an input file and results for entity reference types are included below.

The import creates a reference on the source object's editor References tab. It can also be viewed from the target object's editor References By tab.

The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Prerequisites

The inbound file must include:

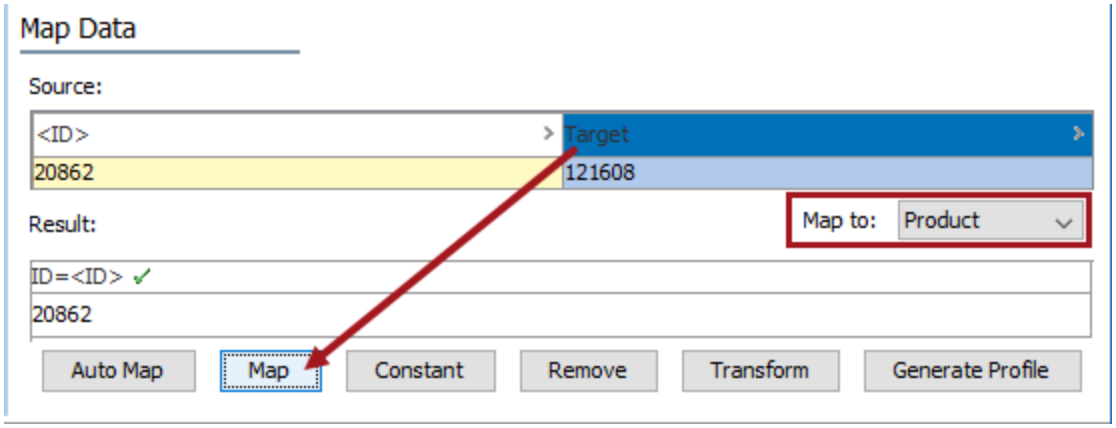
1. The ID or key (not the name) of the reference source: product, classification, or entity.
2. The ID or key (not the name) of the reference target: entity.
3. Objects that already exist in STEP.

Map Entity References

1. Below the Source section, use the **Map to** dropdown to select the object super type defined by the source of the reference type. For example, on a product-to-entity, the source is product, and on an classification-to-entity reference, the source is classification.
2. Map the source ID or key column using the Auto Map button, or manually. For more information, refer to the **ID or Key - Map Inbound** topic.

   Columns that include Predefined Header Text can be automatically mapped as defined in the **Inbound Map Data - Auto Map** topic. For the sample below, this includes the <ID> column.

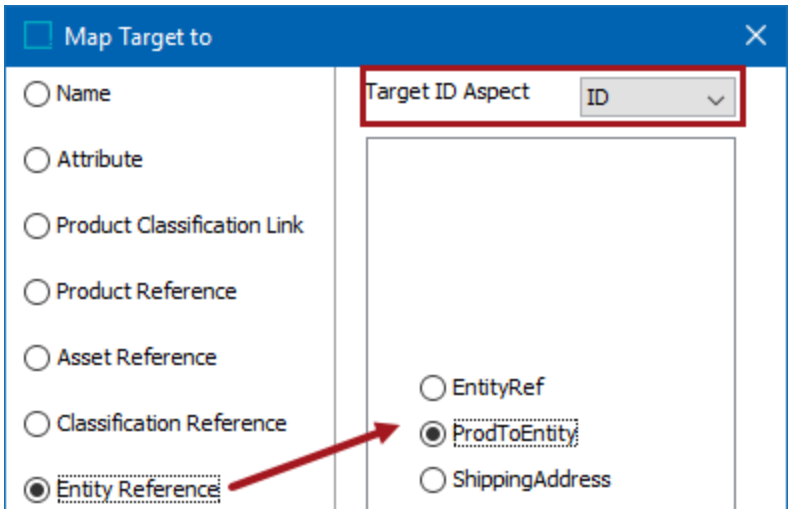
3. In the Source section, select the target column of data and click the **Map** button.



4. In the 'Map ... to' dialog, select the **Entity Reference** radio button, and then select the reference type.

   Only existing reference types are listed. You cannot specify or create new reference types during the import.

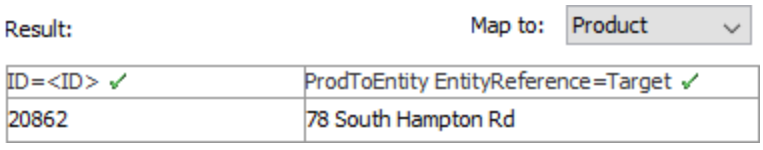
5. For the **Target ID Aspect** parameter:
   - Select **ID** when the inbound file includes the ID for the target.
   - Select the **Key** from the dropdown when the inbound file includes a key for the target.



Note: Selecting a reference type that is not valid for the objects being referenced results in a background process error.

6. Check the **Mandatory** option appropriately:
   - If **checked**, imported objects must have an attribute value in the file. If an attribute value does not exist for an object, the object is skipped and not imported.
   - If **unchecked**, all objects are imported regardless if an attribute value is specified or not.

7. Click **OK** and the **Result** panel displays your selection of reference target. A validated data column is marked with a green check mark as shown below.



8. Complete any additional mapping.

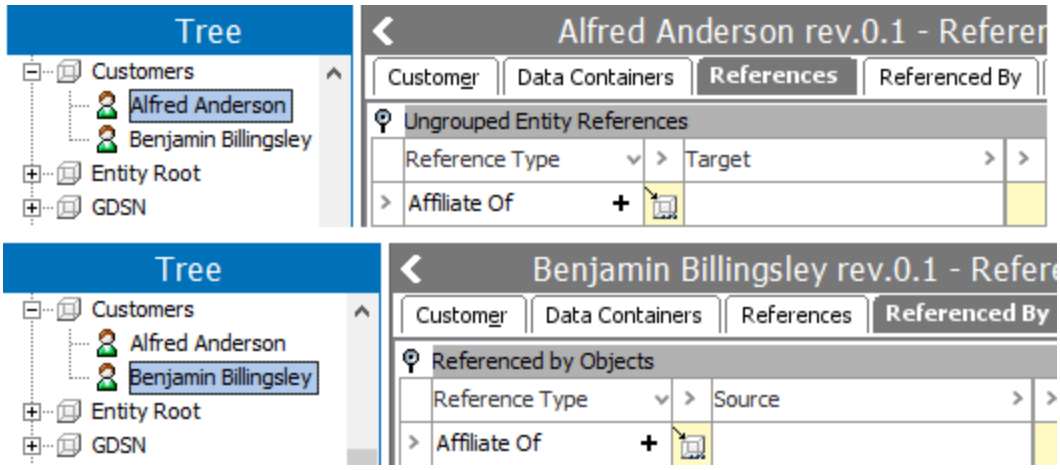
9. If the Parent ID is not included in the inbound file, set the **Default Parent** parameter on the **Identify Destination** step. The parameter is the same for IIEP and Import Manager as defined in the **Import Manager - Identify Destination** topic.
10. Initiate the import.

Entity-to-Entity Reference Example

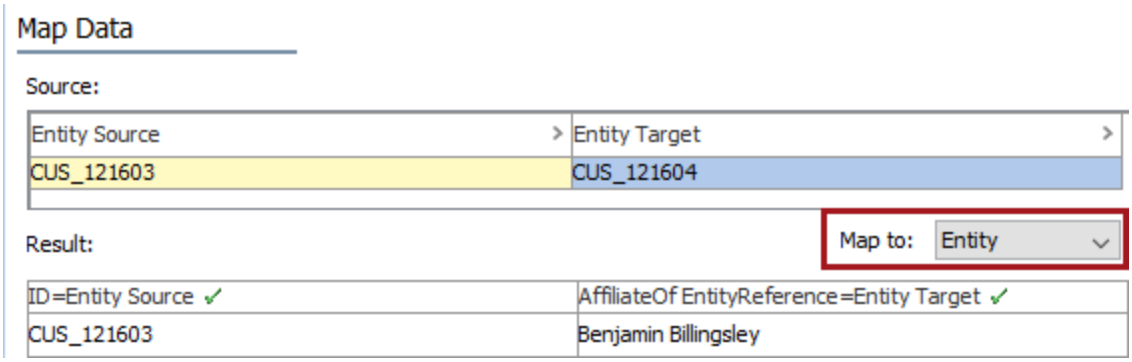
This sample file will be used to create an Affiliate Of reference between two entity objects holding customer data.

	A	B
1	Entity Source	Entity Target
2	CUS_121603	CUS_121604

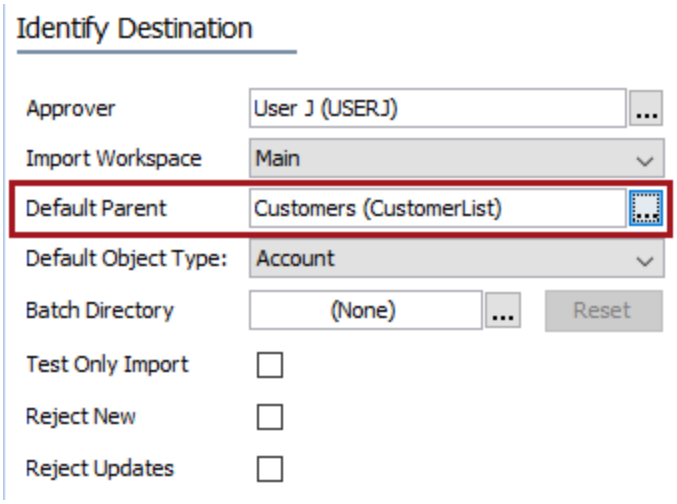
Before the import, the source entity has no Affiliate Of references.



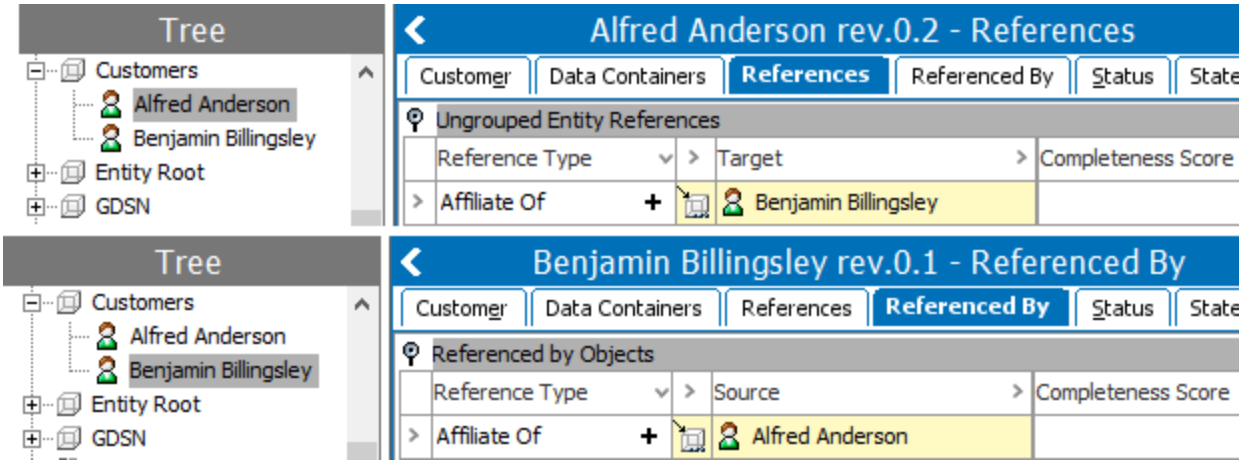
Select the Entity object super type before mapping the source and target entities.



Since the Parent ID was not included in the import file, the Default Parent is selected on the Identify Destination step.



Once the import background process completes without error, the new reference is displayed on both the source and the target objects.



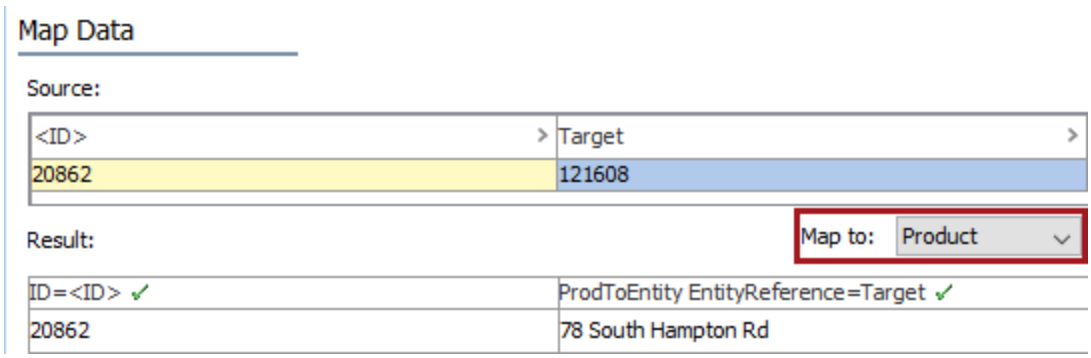
Entity-to-entity references can also be defined by the target entity's Source Record ID. For more information on mapping entity-to-entity references in this way, refer to the **Entity Reference via Source Record ID - Map Inbound** section of the **Data Exchange** documentation.

Product-to-Entity Reference Example

In this sample file, a product ID will be mapped as the source, and an entity ID is the target of the imported reference.

	A	B
1	<ID>	Target
2	20862	121608

The mapped source and target are displayed in the Result panel.


Map Data
Source:
<ID>
Target
20862
121608
Result:
Map to:
Product
ID=<ID>
ProdToEntity EntityReference=Target
20862
78 South Hampton Rd

Entity Reference via Source Record ID - Map Inbound

In the event that a STEP ID is not available, it is possible to import entities with cross references to other entities via the target entity's Source Record ID. This allows incoming objects to reference other objects that either already exist in STEP or are available in the same import.

This feature is intended for use in business-to-business customer data solutions that utilize a Match and Merge strategy, and can be included in an inbound integration endpoint used by a Match and Merge configuration. For more information on configuring an IIEP for Match and Merge, refer to the **IIEP - Configure Match and Merge Importer** section of the **Data Exchange** documentation.

The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Prerequisites

In addition to the steps outlined in the **Map Entity References** section below, the following setup is required to display the Match & Merge Importer option on the Target ID Aspect parameter:

1. The 'Matching - Merge Golden Record' component model must be valid.
2. At least one matching algorithm must be configured.
3. At least one entity-to-entity reference type must have a target of a 'Golden Record Object Type' as defined by the component model, but NOT be used in the component model as the 'Source Relation Reference Type' or the 'Merged-Into Relation Reference Type.'

The inbound file must include:

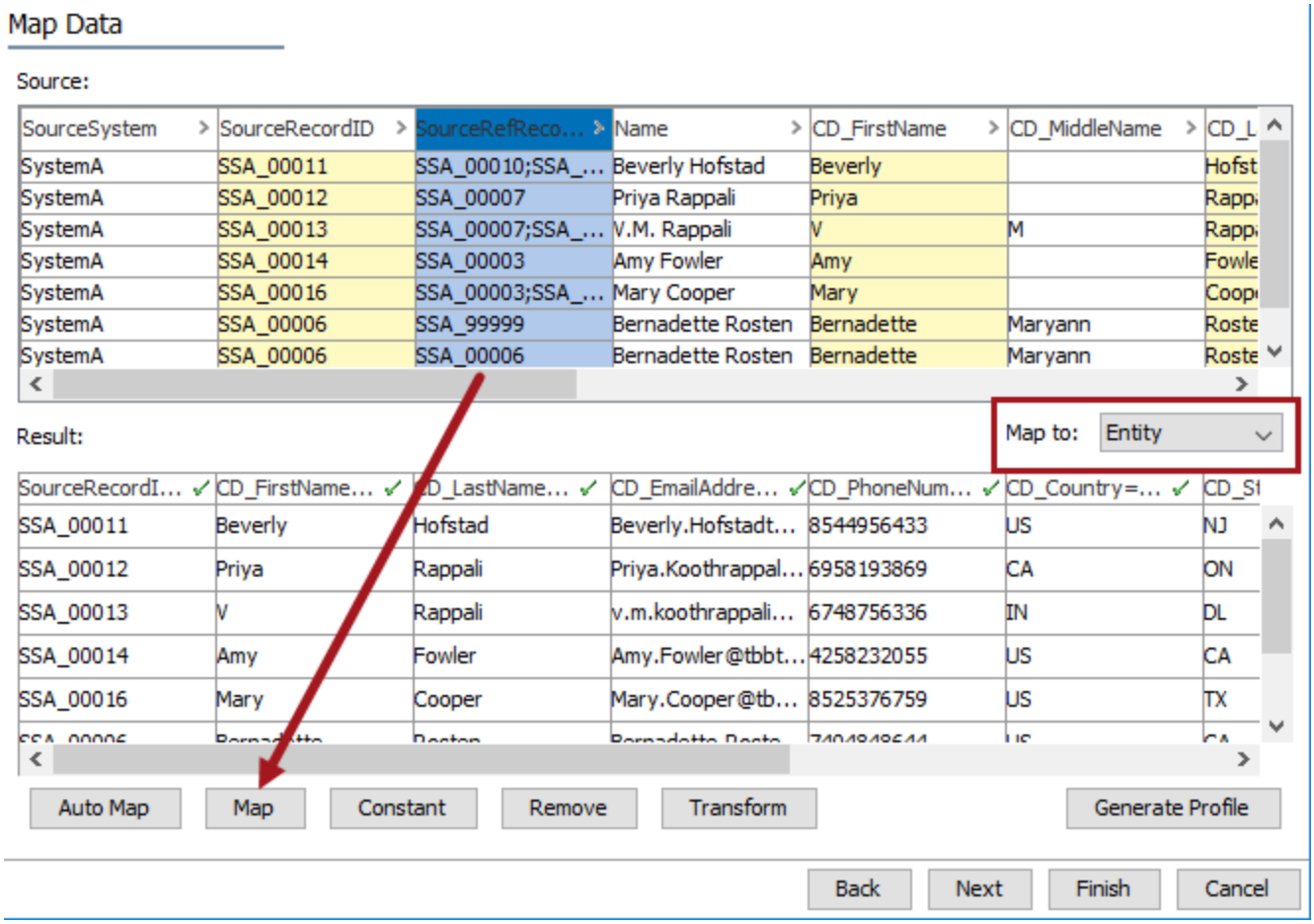
1. The ID or key (not the name) of the reference source: entity.
2. The Source Record ID key (not the name) of the reference target: entity.

Map Entity References

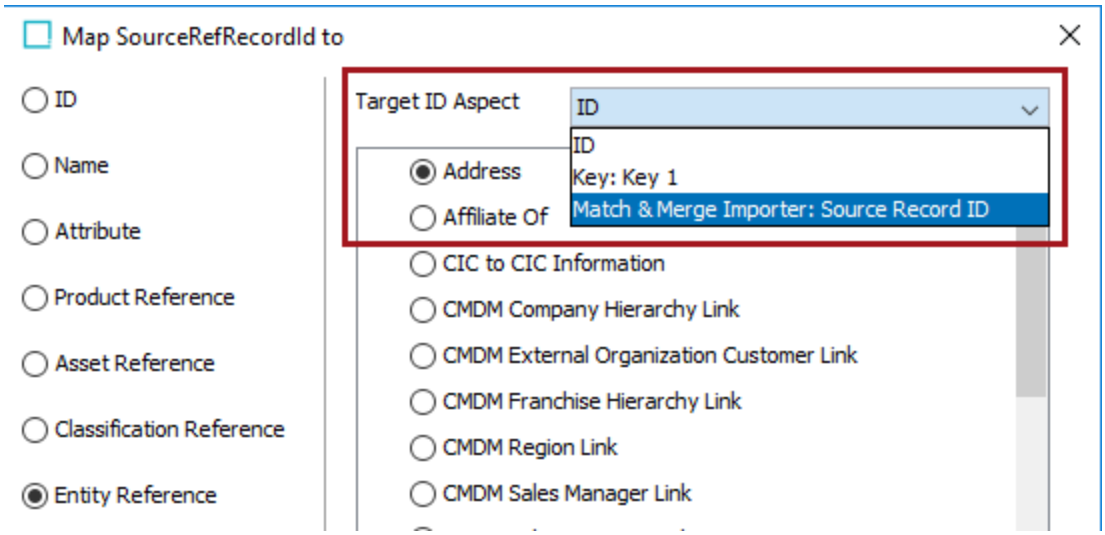
1. Below the Source section, use the **Map to** dropdown to select the object super type defined by the source of the reference type. For example, on an entity-to-entity, the source is entity.
2. Map the source ID or key column using the **Auto Map** button, or do so manually. For more information, refer to the **ID or Key - Map Inbound** topic.

   Columns that include Predefined Header Text can be automatically mapped as defined in the **Inbound Map Data - Auto Map** topic.

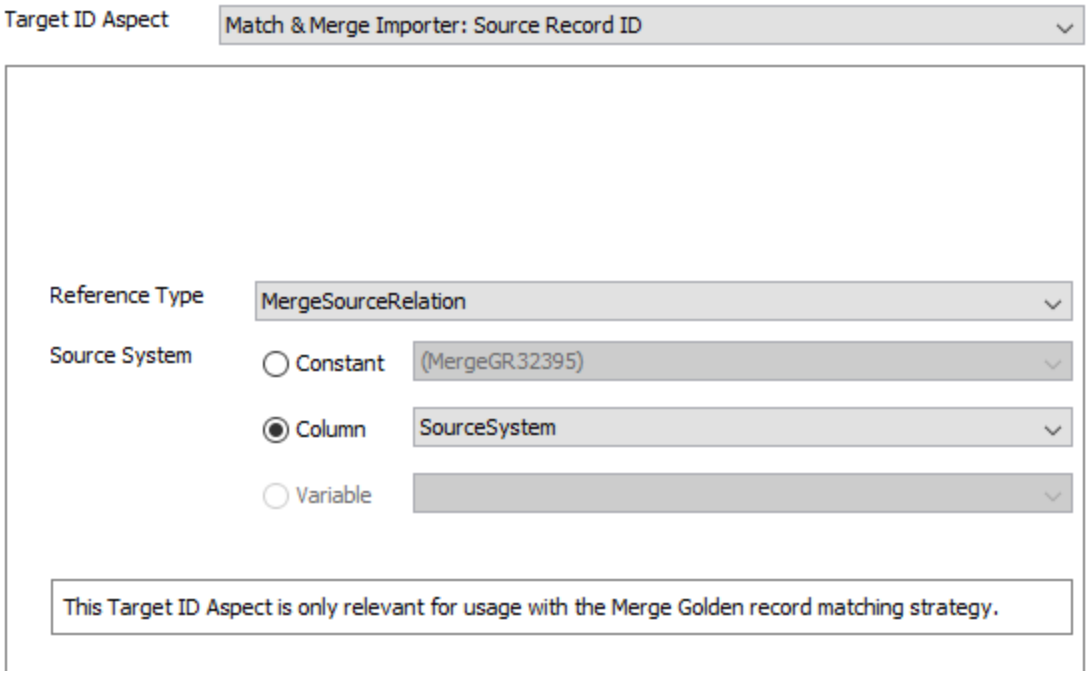
3. In the Source section, select the target column of data and click the **Map** button.



4. In the **Map to** window, select the **Entity Reference** radio button, and then select the **Target ID Aspect** parameter:
   - Select **Match & Merge Importer: Source Record ID** when the inbound file includes the Source Record ID for the target.

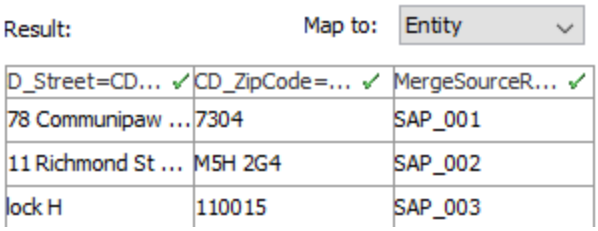


5. For the **Reference Type** parameter, choose the relevant Reference Type.
6. For the **Source System** parameter, choose the relevant Source System:
   - 'Constant' provides a constant value that can be chosen. The values include all the source system found in the system.
   - 'Column' lists the set of columns in the Excel or CSV file in question. Choose the column that contains the source system values.
   - 'Variable' lists the variables that could have been generated in a previous step.



7. Check the **Mandatory** option appropriately:
   - If **checked**, imported objects must have an attribute value in the file. If an attribute value does not exist for an object, the object is skipped and not imported.
   - If **unchecked**, all objects are imported regardless if an attribute value is specified or not.

8. Click **OK** and the **Result** panel displays your selection of column and attribute. A validated data column is marked with a green check mark as shown below.



9. Complete any additional mapping.
10. Save the import configuration and add it to the relevant IIEP used for the Match & Merge use case in question.

ID or Key - Map Inbound

The STEP ID is the most common way to uniquely identify an object and it is required for updating values for existing objects. This type of mapping is described below in the **Mapping ID** section.

However, when creating new objects, and a STEP ID is not included in the input file, you can match an existing object using its key instead. This type of mapping is described below in the **Mapping Key** section.

The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

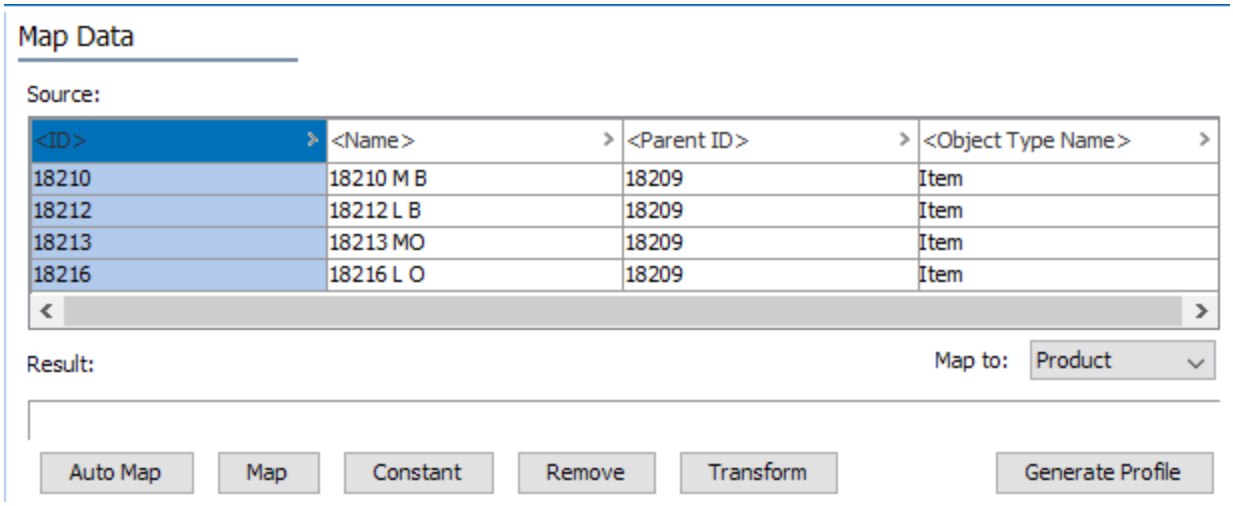
Mapping ID

IDs must be unique and can be up to 40 characters long. Once saved in STEP, an ID can never be changed. Common setup is to avoid using special characters in the ID. When the object is a family, subcategory, or major category, etc., a common setup is to use a number for the ID and use the actual category name as the name.

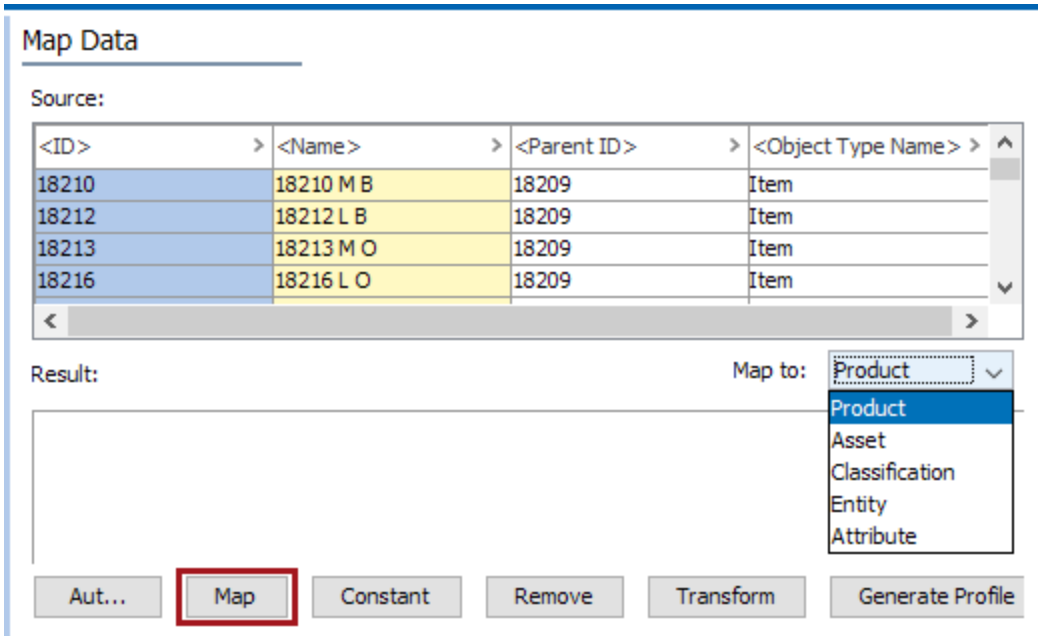
When updating existing products, you must specify the object ID as it exists in STEP.

When STEP is configured to generate IDs automatically, the import can generate object IDs when no match in STEP is found to exist in the import file. In this case, a blank column in the import file will be mapped. When your object ID is an actual part number, common setup is to not configure automatically generated IDs similar to the actual part number, since it can cause confusion when identifying data.

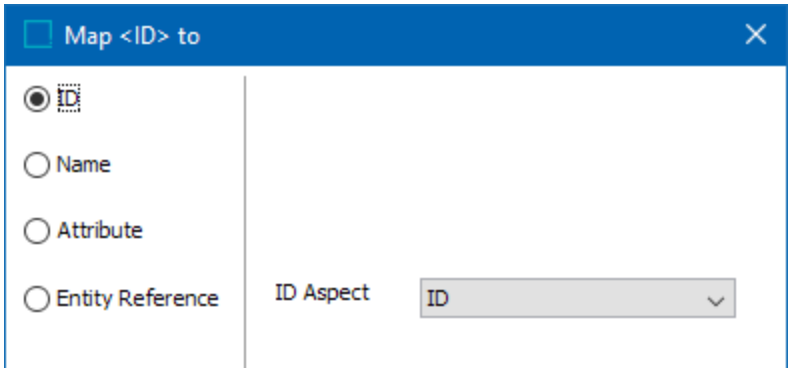
1. In the Source section, select the column that contains the object ID.



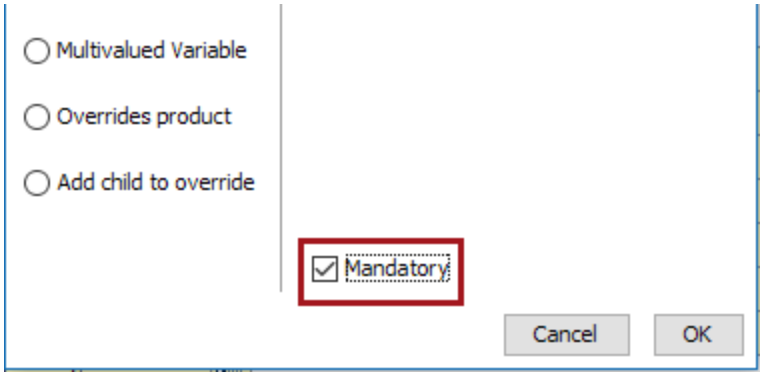
2. From the **Map to** dropdown, choose the option for the type of data being imported, and then click the **Map** button. In this example, products are being imported.

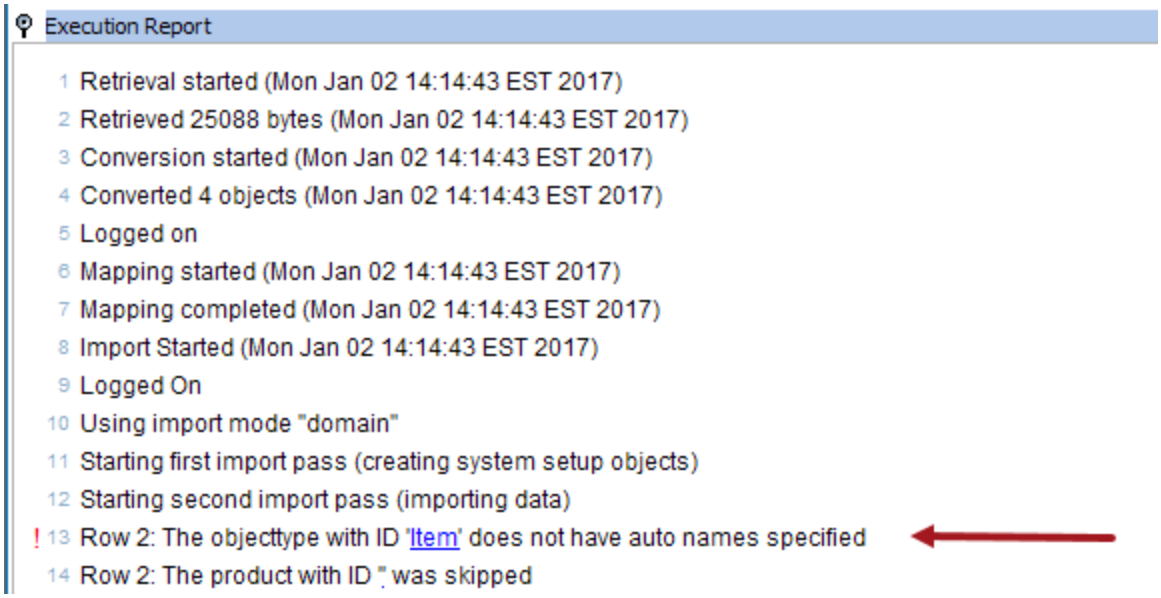


3. In the 'Map ... to' dialog, select **ID** radio button, and select the **ID** in the 'ID Aspect' dropdown list.

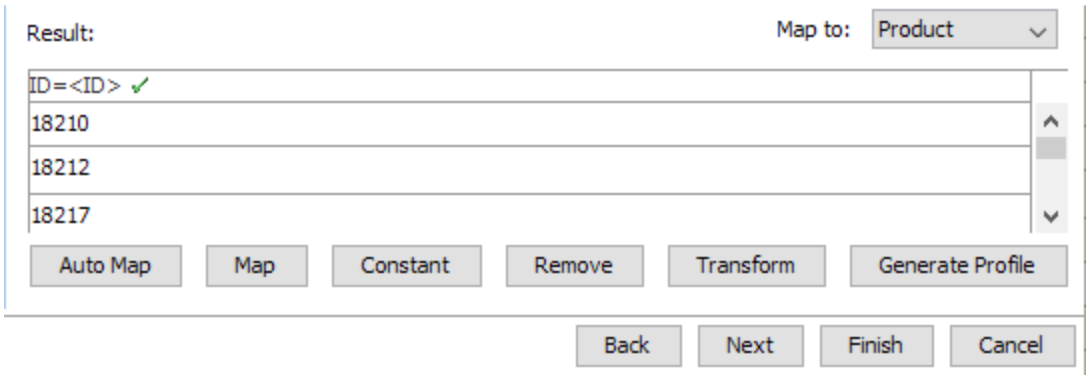


4. Determine if the 'Mandatory' option needs to be checked and do so accordingly:
   - **Check** the 'Mandatory' option only if each object being imported has an ID. If they do not all have IDs, then the object will be skipped, will not display in the Results field, and will not be imported.
   - **Unchecked** the 'Mandatory' option if any objects are missing the ID but still have other values. Doing this allows all objects to be imported, provided their object type is configured to generate IDs automatically. If there is no configuration in the 'ID Pattern', the object missing the ID will be skipped and not import.





5. Click **OK** and the **Result** section shows the mapped column with the ID of the objects to be created and/or have their attributes modified in STEP.



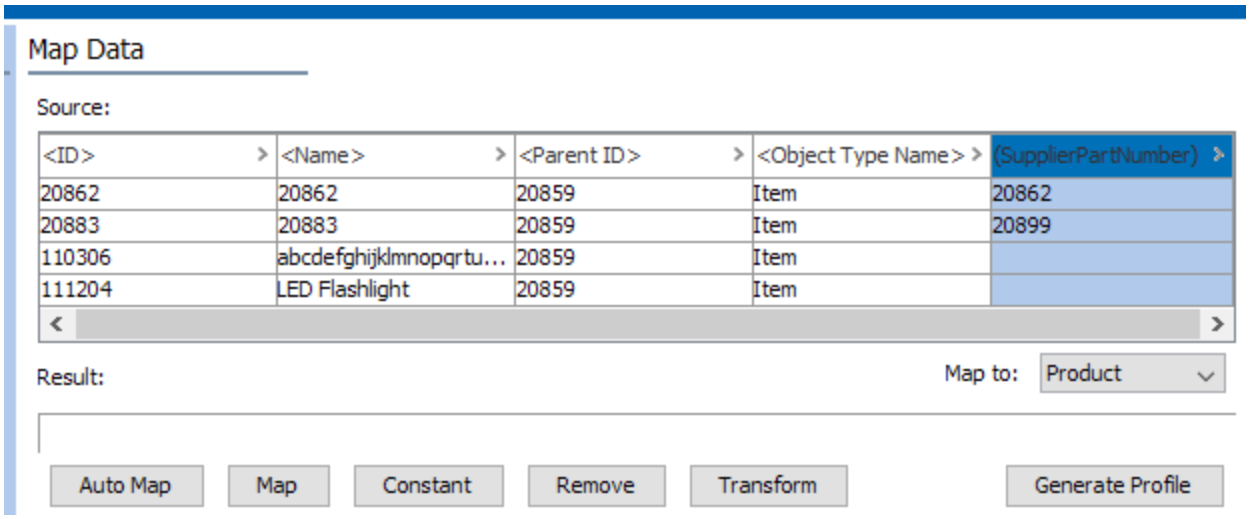
6. Complete the mapping and initiate the import.

Mapping Key

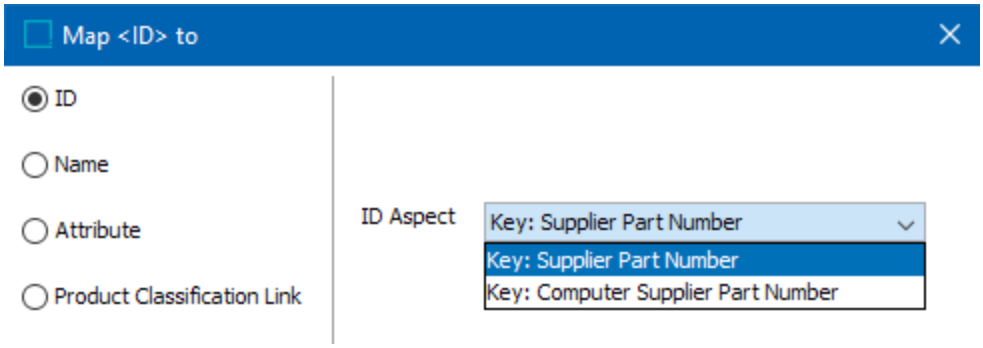
Objects, parent objects, and referenced objects can also be identified using keys. Keys are automatically generated when the underlying attributes are mapped during import.

If an existing object has a key that is active and has a specified object type, it can be mapped to identify an object during import. For more information, refer to the **Unique Keys** section of the **System Setup** documentation.

1. Prior to importing, verify unique keys are activated in STEP so that they are available for mapping.
2. In the Source table, select a column which contains unique key information, and click the **Map** button.



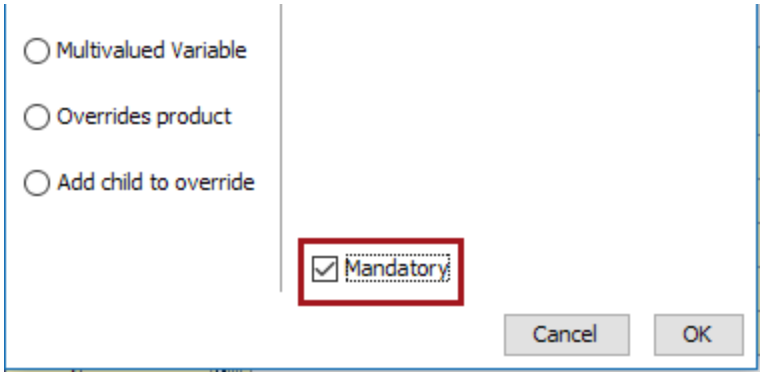
3. In the 'Map ... to' dialog, select the ID radio button option, and then select the respective key in the ID Aspect dropdown list. This indicates the ID of the current object is represented by the specified key.



Note: Keys will only be available for selection if they are active and have an object type specified.

4. Determine if the 'Mandatory' option needs to be checked, and do so accordingly:
   - **Check** the 'Mandatory' option only if each object being imported has an unique key. If they do not all have unique keys, then the object will be skipped, will not display in the Results field, and will not be imported.

- **Unchecked** the 'Mandatory' option if any objects are missing the unique key, but still have other values that need to be imported.



Note: If no matching object is found (based on keys) and the import configuration allows for the creation of new objects, a new object will be created. To accomplish this, STEP IDs must be omitted from the import, and all attributes that make up the key definition must be mapped and populated on the current object. Additionally, the object type being created must use autogenerated IDs. Failing to provide the underlying key attributes will result in an error.

5. Complete the mapping and initiate the import.

Multivalued Variable - Map Inbound

When mapping to multivalued variables, a user is able to perform the same transformation on each value within a multivalued attribute, asset, classification, entity, or product. Mapping to this option enables the user to map without having to split each part of the multivalued attribute, asset, classification, entity, or product out, perform multiple mappings and transformations, and then map all of them back into one variable again.

Note that it is only possible to map to multivalued variables during the Map Data process when importing, and multivalued variables are not accessible elsewhere in STEP. It does not have a permanent place or usage outside of being a way to transform a multivalued attribute, asset, classification, entity, or product during import.

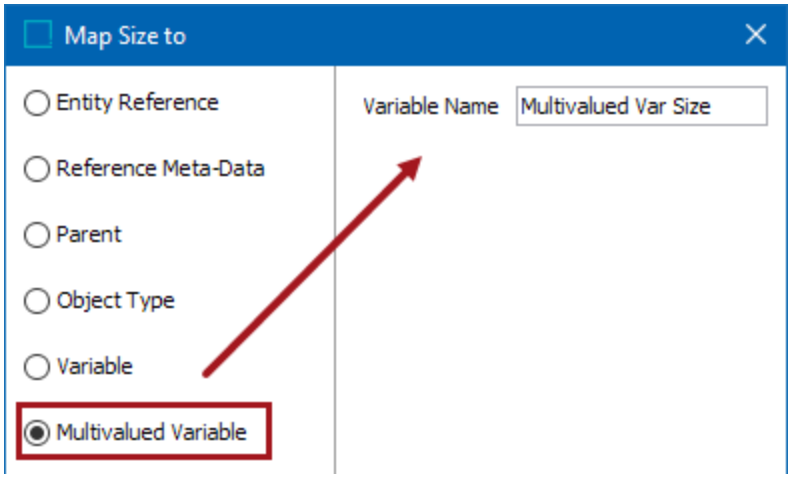
To map using a multivalued variable, follow the steps below. Note that the below example shows an Excel file being imported, though all standard inbound formats are accepted:

	A	B	C	D
1	ID	Name	Size	Color
2	26658	box 1	10;20;30	brown

The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Map Multivalued Variable

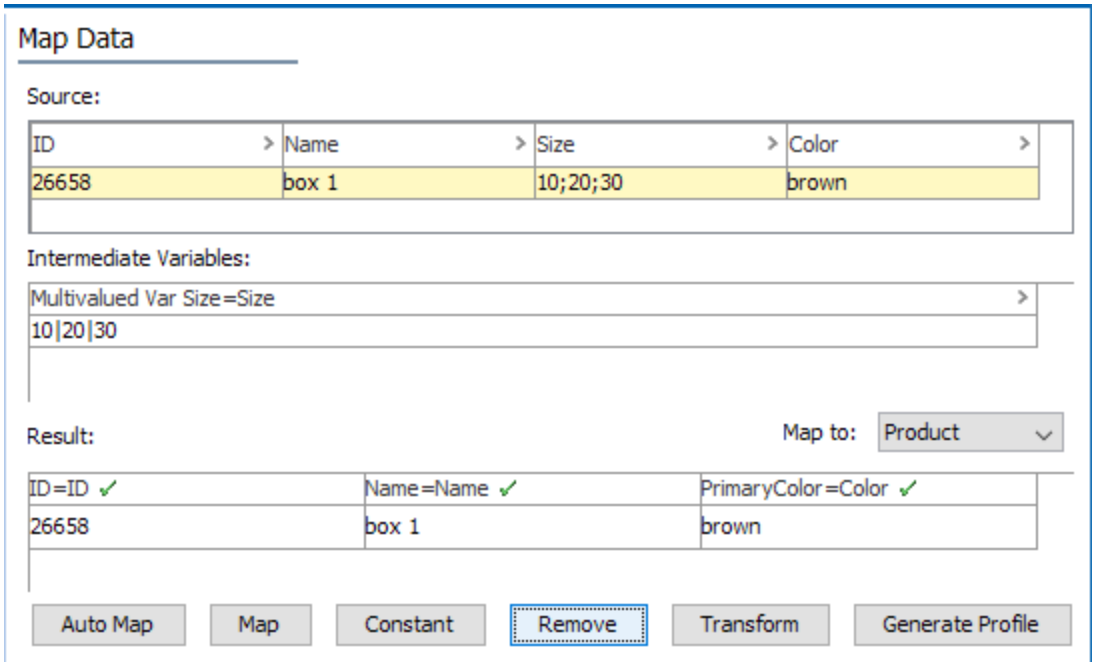
1. Use Auto Map to map the ID, or map it manually. For more information, refer to **ID or Key - Map Inbound**.
2. Map all other data required for import. For more information, refer to **Inbound Map Data - Map**.
3. Map the column that holds data that needs transformations on each value to the Multivalued Variable option, and type the name in the Variable Name field.



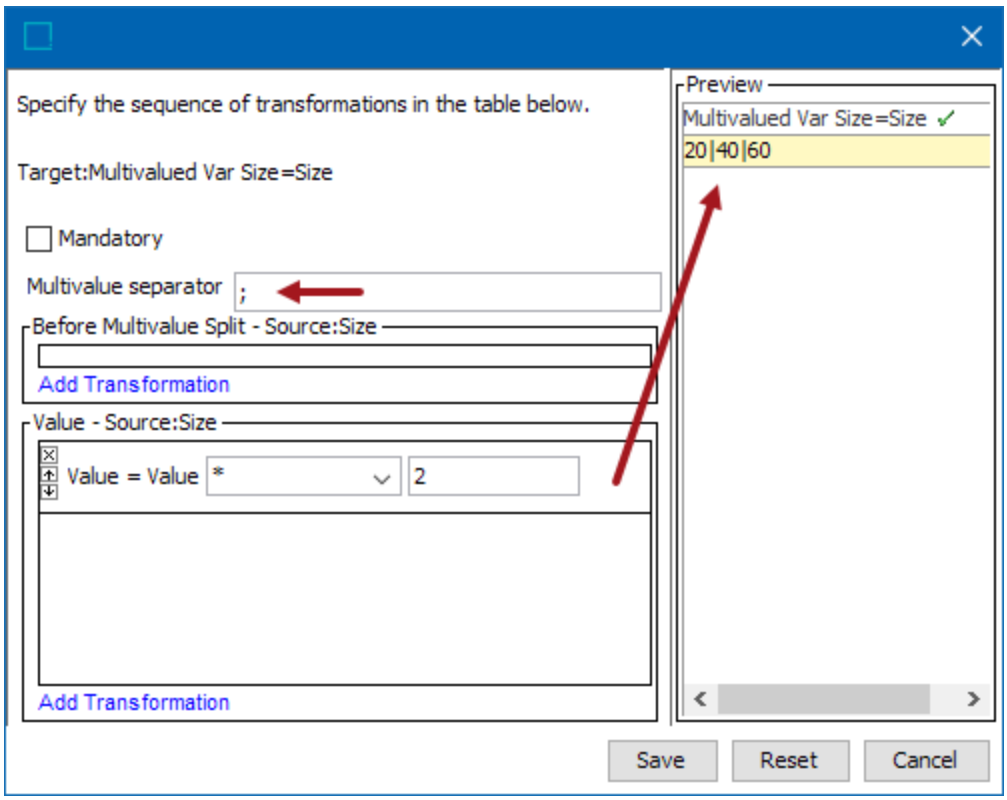
4. Check the **Mandatory** option appropriately:
   - If **checked**, imported objects must have an attribute value in the file. If an attribute value does not exist for an object, the object is skipped and not imported.

- If **unchecked**, all objects are imported regardless if an attribute value is specified or not.

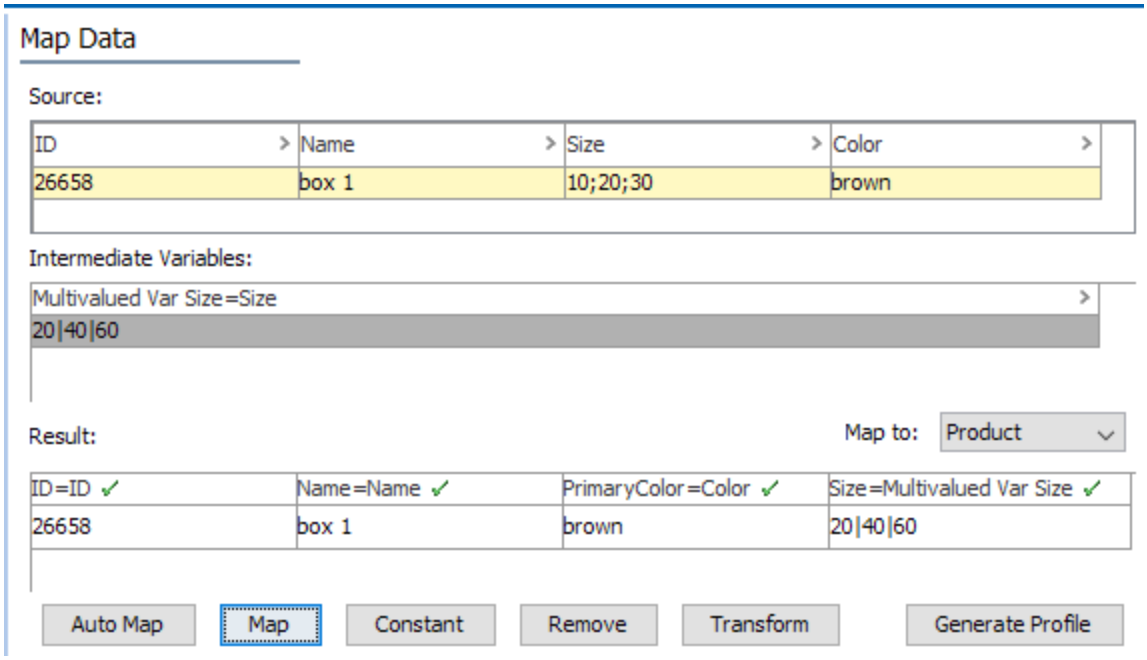
5. Click **OK** and the **Intermediate Variables** section displays the multivalued variable to temporarily hold modified data for additional transformations.



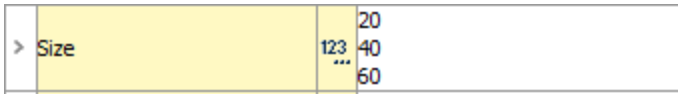
6. Click on the multivalued variable in the Intermediate Variables section, and click the **Transform** button.
7. Select any needed transformations. The Preview pane shows how the transformations modify each part of the value in the multivalued variable, as 'multivalued parts' are designated by the **Multivalued separator** field. In this example the semi-colon (;) is used as the separator and each portion of multiplied by 2.



8. Once transformed, select the multivalued variable in the Intermediate Variables section and click on Map. Map appropriately.



Once the import is completed successfully, the attribute displays the transformed values.



Mappings, multivalued variables, and transformations are saved in an import configuration for use each time the same import is run.

The transformations that can use the variable during import are the following:

- Append from multivalue source
- Append from source
- If equals source
- Math - Add source
- Math – Divided by source
- Math – Multiply by source
- Math – Subtract source
- Prepend from Source

For more on the above transformations, refer to the **Transformations** topic in the **Resource Materials** online help.

For an example of the 'Append from source' transformation being used, refer to the **Concatenate Data Using Variables Example** topic in the **Data Exchange** documentation.

Name - Map Inbound

Names should be used to properly to identify the object (not as a description). Avoid using special characters in the name. Unique names are not required, but do aid in identifying data. When the object is a family, subcategory, or major category, etc., a common setup is to use a number for the ID and use the actual category name as the name. When updating attribute values on existing objects in STEP, mapping the object name is not required, even if it exists in the input file.

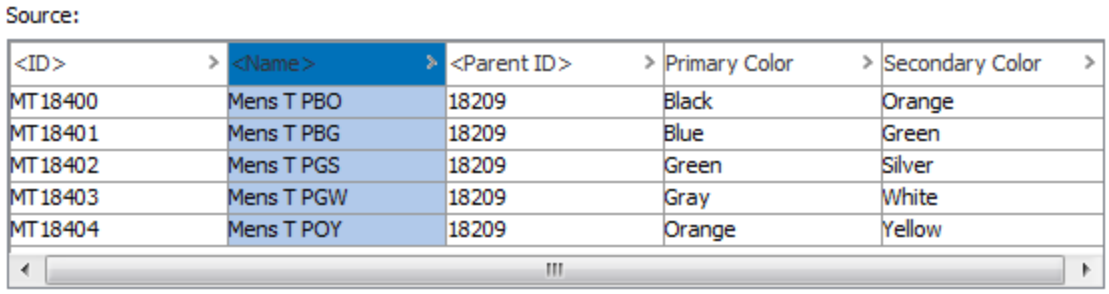
Common setup involves mapping the object name when one of the following conditions is true:

- New objects are being imported into STEP.
- Existing names of all objects in the import file should be changed.
- The import file includes object names for a specific context and the objects are dimension dependent. For example, you sell the same product in three different markets, but identify them by a different part number in each market. Although you might typically hold a single product in STEP with a unique ID, in this case, for each market's context, you could enter the appropriate 'part number' into the Name. For more information on how to import content in different context from the same import file, refer to the **Dimension Point for Single Dimension Data Example** topic.

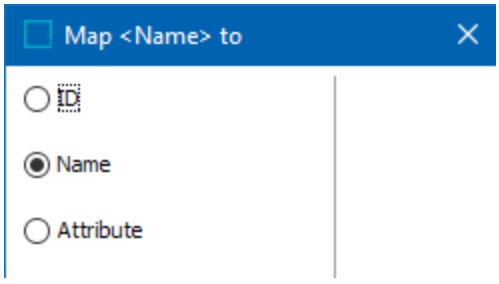
The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Map a Name

1. In the Source section, select the data column to map to the object name, and then click **Map**.



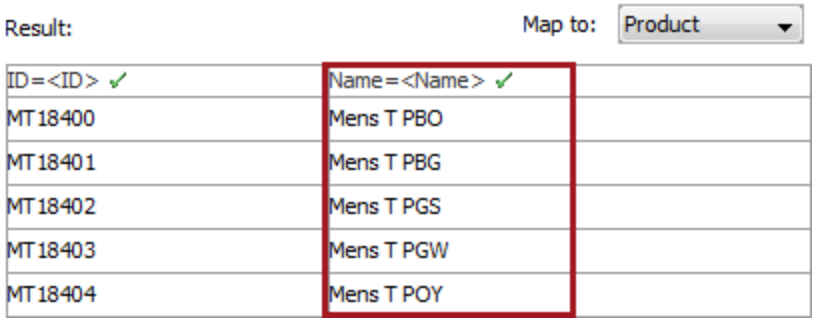
2. In the 'Map ... to' dialog (the label of the field is used as the title of this dialog.), select the **Name** radio button.



3. Check the **Mandatory** option appropriately:

- If **checked**, imported objects must have a name in the file. If an object does not have a name, it is skipped and not imported.
- If **unchecked**, all objects are imported, even those without a name.

4. Click **OK** and the **Result** panel displays that the mapped column will specify the Name of the objects in the input file.



5. Complete the mapping and initiate the import.

Object Type - Map Inbound

This option is used whenever a new object is created or to change the existing object type of an object being imported. Object type can be changed for products, classifications, assets, and entities (with the limitations defined below).

Although these examples use Excel, any format that is valid for both inbound and outbound exchange is allowed.

The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Limitations for Entities

The following limitations exist for changing the object type on an entity:

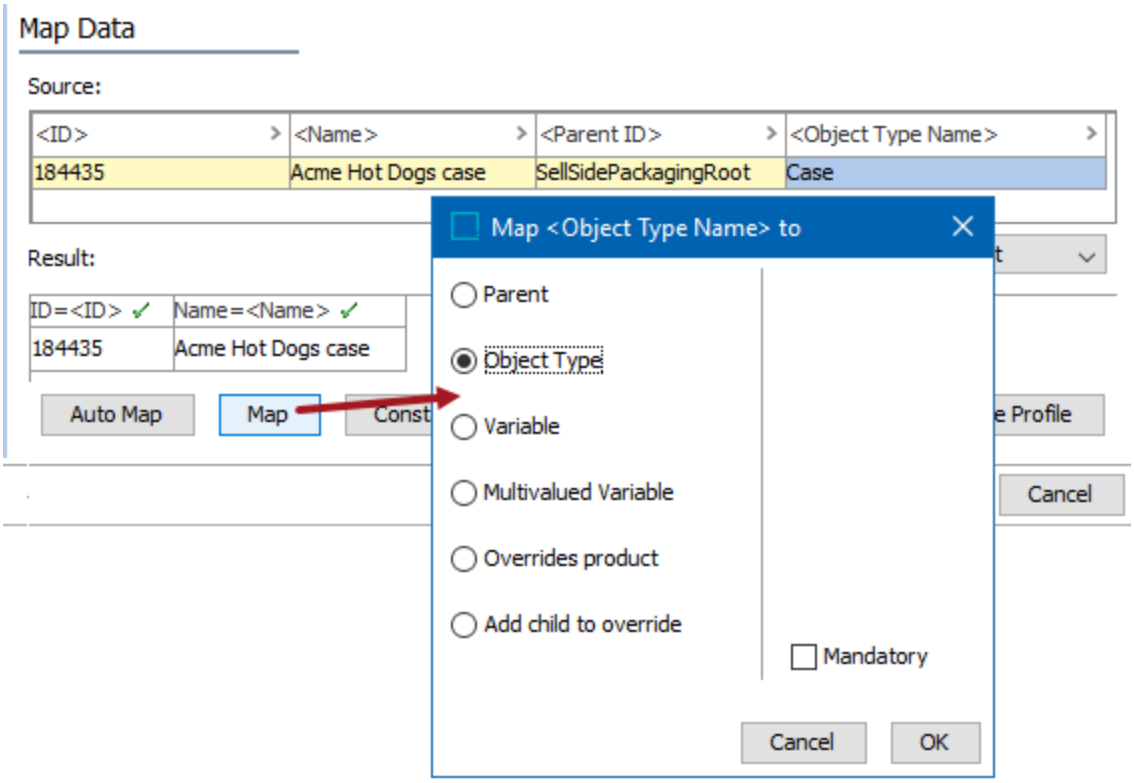
- Changing a globally revisable entity object type requires special handling. For Oracle databases, this action requires single-update mode (SUM), as defined in the **Single-Update Mode** topic. For Cassandra databases, this action uses Lock-free Schema Change (LFSC) functionality, as defined in the **Lock-free Schema Change** topic.
- Changing the object type of a child entity if the parent is globally revisable is not allowed.
- Changing the object type of the entity when the Term List is different is not allowed.

Mapping

The steps for mapping to create new objects or to update existing objects are the same as follow. For prerequisites and examples of creating or updating, refer to the following sections below:

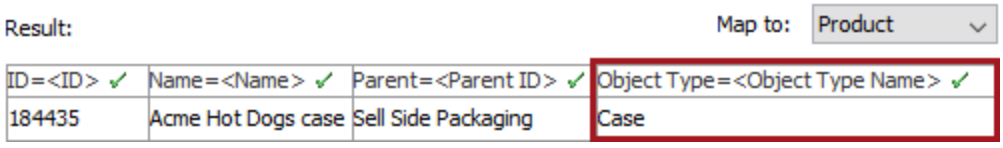
- **Map to Create New Items with Object Type**
- **Map to Change the Existing Object Type**

1. Import the file to make the update in STEP and map the **ID (or key)** and **object type**. For more information on mapping ID or Key, refer to the **ID or Key - Map Inbound** topic.
2. In the Source section, select the object type column, click the **Map** button.
3. In the 'Map ... to' dialog, select the **Object Type** radio button. No additional configuration is necessary.



4. Check the **Mandatory** option appropriately:
   - If **checked**, imported objects must have an attribute value in the file. If an attribute value does not exist for an object, the object is skipped and not imported.
   - If **unchecked**, all objects are imported regardless if an attribute value is specified or not.

5. Click **OK** and the **Result** panel displays your selection of column and attribute. A validated data column is marked with a green check mark as shown below.



6. Complete the mapping and initiate the import.

Map to Create New Items with Object Type

The inbound field mapped to object type defines the object type being imported and also defines what parent type is allowed.

Prerequisites

The following conditions must be met in order to set an object type during import:

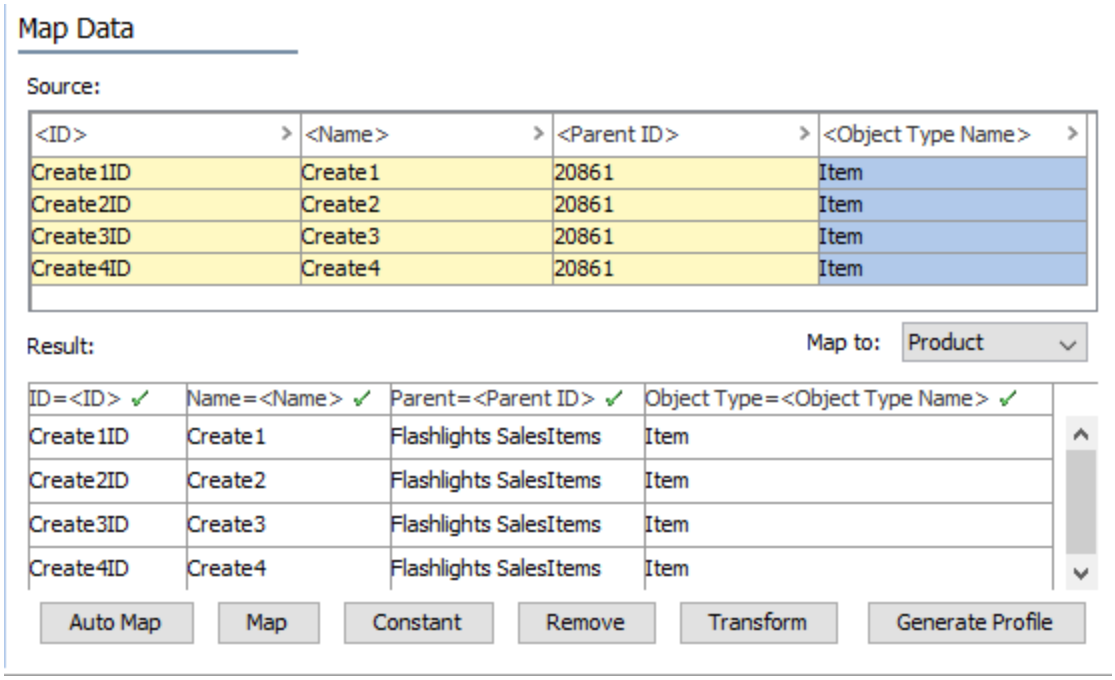
1. The inbound file must include: object ID (or key), parent ID, and object type.
2. The object type in the inbound file must already exist in STEP.
3. When the parent ID is not in the inbound file, and the object type is not valid, an error is displayed in the background process.

Inbound Data

An import file includes multiple new objects, the parent ID, and the desired object type.

	A	B	C	D
1	<ID>	<Name>	<Parent ID>	<Object Type Name>
2	Create1ID	Create1	20861	Item
3	Create2ID	Create2	20861	Item
4	Create3ID	Create3	20861	Item
5	Create4ID	Create4	20861	Item

Once all columns have been mapped, the Result section shows the objects that will be created.



Map to Change the Existing Object Type

During import, it is possible to change the object type of one or more existing objects. This is useful during data migration, data cleanup activities, for example when the wrong object type was originally assigned.

Important: When changing object types, common setup is to keep the import simple and update changes to attributes separately. Since not all attributes are valid for all object types, import errors will occur if attempting to update invalid attributes.

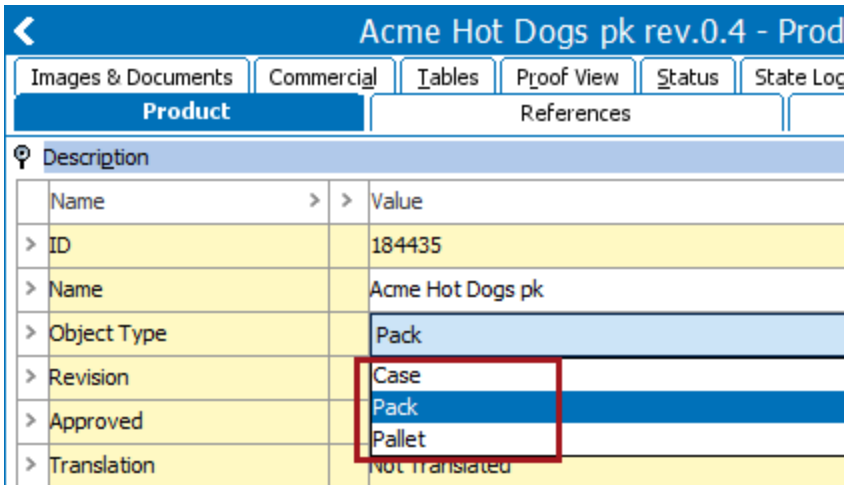
Prerequisites

The following conditions must be met in order to change an object type during import:

1. The inbound file must include the object's ID (or unique key) and the ID of the new object type.
2. The existing objects being modified must be updated to one of the valid options available on the Object Type parameter dropdown (as shown in the image below).
3. The parent node must allow children of the new object type being selected or an error is displayed in the background process.
4. The attribute validations must be the same between the original object type and the new object type or an error is displayed in the background process.
5. The dimension dependencies must be the same between the original object type and the new object type or an error is displayed in the background process.

Existing Data

Consider an object which has the valid object type options of Case, Pack, and Pallet (as shown in the image below). In our example steps below, assume that the current selection of 'pack' should be changed to 'case.'



The import file is created by first exporting the objects that need to have a new object type.

	A	B	C	D
1	<ID>	<Name>	<Parent ID>	<Object Type Name>
2	184435	Acme Hot Dogs pk	SellSidePackagingRoot	Pack

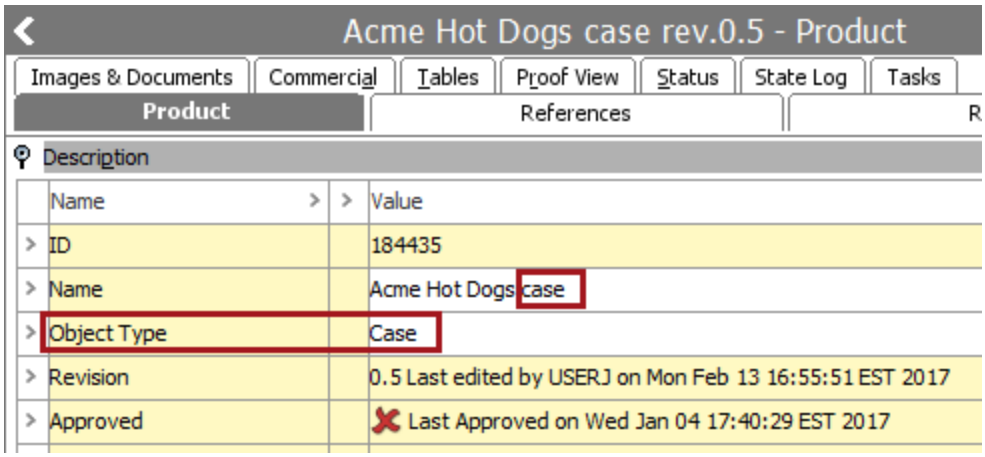
Inbound Data

Update the export file to include the data needed for import:

- Change the name to reflect the object type change
- Change the object type
- If necessary, include a new Parent ID in the file to also reparent the object (change the location in the hierarchy). For this example, we left the object in its current hierarchy location.

	A	B	C	D
1	<ID>	<Name>	<Parent ID>	<Object Type Name>
2	184435	Acme Hot Dogs case	SellSidePackagingRoot	Case

After a successful import, refresh the object and review it in the Tree. The original object type of 'Pack' has been changed to 'Case,' and the name has been updated.



Overrides Product - Map Inbound

Product overrides are alternate versions of products and product families that may have differing values, references, links, and structures. Attributes and values applied to the product family are inherited to the product override and can be replaced with local values and references on the product override.

The Overrides Product option is only available for the Product object super type.

For more information on Product Overrides, refer to the **Product Override** topic in the **Getting Started** documentation or the **Override Scenario** use case topic in the **Calculated Attributes** section of the **System Setup** documentation.

The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Prerequisites

1. Before importing a product override, it must first be setup in System Setup. Refer to the **Product-Override Object Types and Commercial Object Types** topic in the **System Setup** documentation.
2. Verify the import file includes the following fields:
   - **ID:** The ID of the new override product to be imported
   - **Overrides:** The ID of the product in STEP that will get an override version of itself
   - **Object Type ID:** The ID of the object type in System Setup that the override product will be
   - **Parent ID**: The ID of the parent the override product will live under

While the example shows an Excel import file, all inbound formats are allowed.

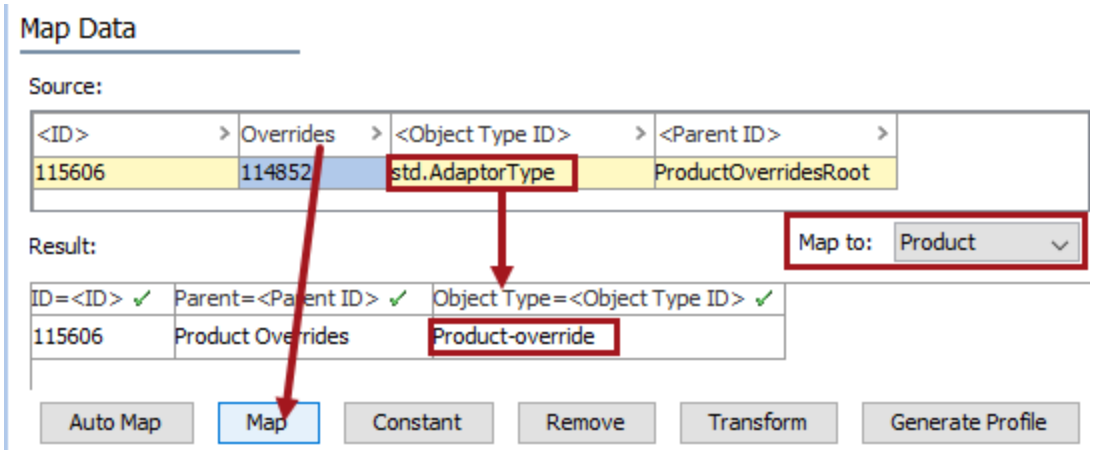
	A	B	C	D
1	<ID>	Overrides	<Object Type ID>	<Parent ID>
2	115606	114852	std.AdaptorType	ProductOverridesRoot

Map Overrides Product

1. Below the Source section, use the **Map to** dropdown to select the **Product** object super type.
2. Map the ID or key column using the Auto Map button, or manually. For more information, refer to the **ID or Key - Map Inbound** topic.

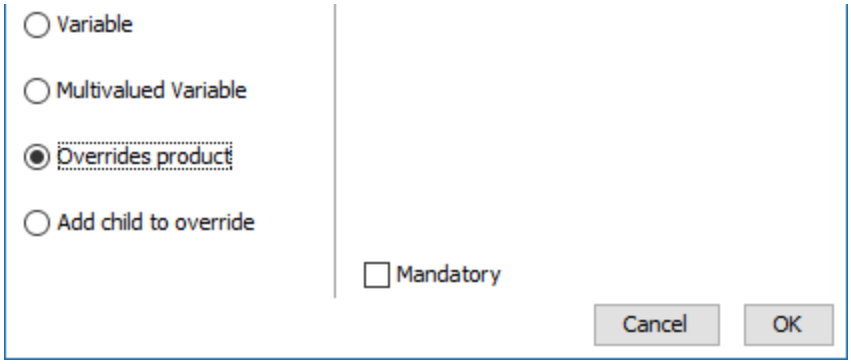
   Columns that include Predefined Header Text can be automatically mapped as defined in the **Inbound Map Data - Auto Map** topic. For this sample file, this includes the <ID>, <Object Type ID>, and <Parent ID> columns.

3. In the Source section, select the Overrides column of data and click the **Map** button.



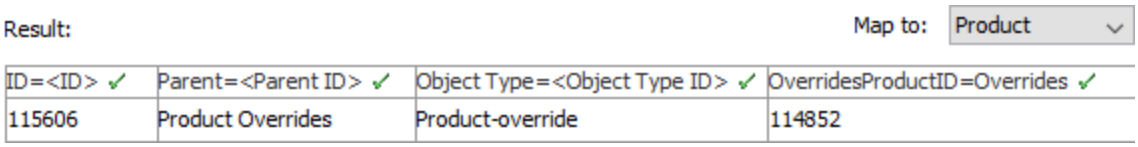
Note: The Object Type ID in System Setup is standard in STEP for Product Override and thus displays as 'std.AdaptorType.' However, once mapped, STEP automatically changes to the object type that will be displayed in Tree after import.

4. In the 'Map ... to' dialog, select the **Overrides product** radio button.



5. Check the **Mandatory** option appropriately:
   - If **checked**, imported objects must have an attribute value in the file. If an attribute value does not exist for an object, the object is skipped and not imported.
   - If **unchecked**, all objects are imported regardless if an attribute value is specified or not.

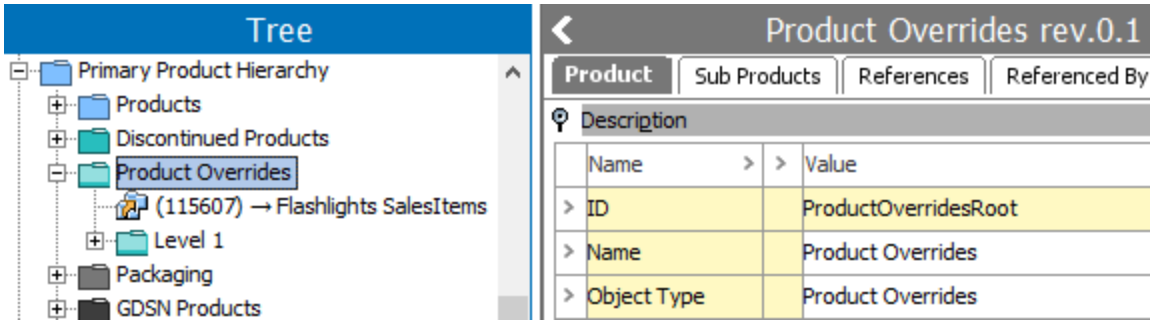
6. Click **OK** and the **Result** panel displays your selection of column and attribute. A validated data column is marked with a green check mark as shown below.



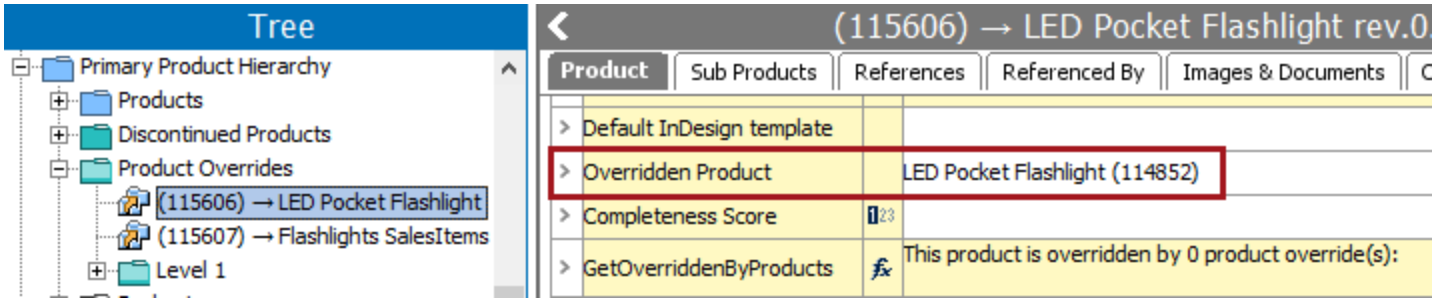
7. Complete the mapping and initiate the import.

Example

Prior to import, the Product Overrides node in Tree displays a single override.



After a successful import, a new product override object is displayed and the overridden product is shown.



Parent - Map Inbound

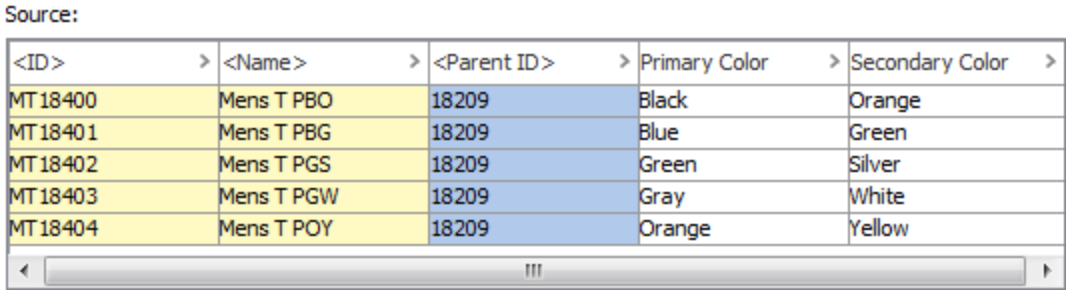
The parent column indicates where to locate, or relocate, the imported object(s). When updating attribute values for existing objects, mapping the parent column is not required, even if it exists in the input file. Common setup involves mapping the parent column when one of the following conditions is true:

- new objects are being imported into STEP
- existing objects within the hierarchy need to be relocated

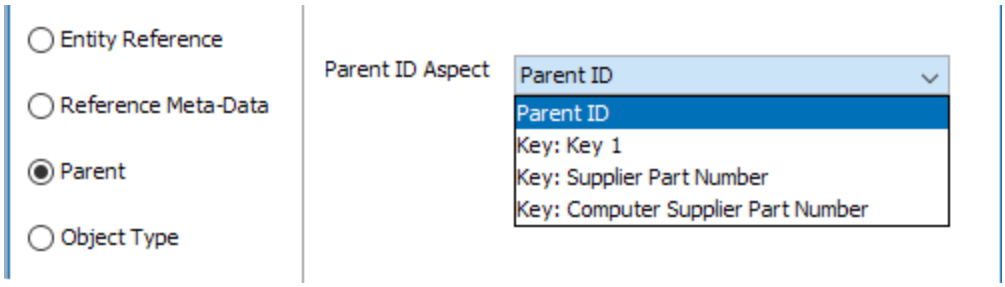
The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Map Parent

1. In the Source section, select the column of data that includes valid STEP IDs for parents, and click the **Map** button.



2. In the 'Map ... to' dialog (the label of the field is used as the title of this dialog.), select the **Parent** radio button. Select the 'Parent ID' option from the **Parent ID Aspect** dropdown list. To use a Key, instead of the Parent ID, select the appropriate option from the Parent ID Aspect dropdown.



3. Check the **Mandatory** option appropriately:
   - If **checked**, imported objects must have a parent ID in the file. If a parent ID does not exist for an object, the object is skipped and not imported.
   - If **unchecked**, all objects are imported regardless if a parent ID is specified or not.

4. Click **OK** and the **Result** panel displays your selection.

Result: Map to: Product

ID=<ID> ✓	Name=<Name> ✓	Parent=<Parent ID> ✓
MT18400	Mens T PBO	Cotton T-Shirts
MT18401	Mens T PBG	Cotton T-Shirts
MT18402	Mens T PGS	Cotton T-Shirts
MT18403	Mens T PGW	Cotton T-Shirts
MT18404	Mens T POY	Cotton T-Shirts

Note that if valid STEP IDs were not found in the first 200 rows of the file, the Result panel will highlight in red color, and a tool tip will show when hovered over the row, that the object was not found in STEP.

Parent=<Parent ID>
111204
20899
668998643
3585275

Product with ID "668998643" not found

5. Complete the mapping and initiate the import.

Product Classification Links - Map Inbound

A product in the inbound file can be linked to one or more classifications in STEP using the valid product-to-classification link types.

Prerequisites

The inbound file must include:

1. The ID or Key of the product.
2. The ID or Key of the classification.
3. A separate field for each classification ID when linking the same product to multiple classifications.

Although an Excel file is shown in the examples, any inbound format is allowed.

	A	B	C	D	E	F
1	<ID>	<Name>	<Parent ID>	<Object Type ID>	Link 1	Link 2
2	22624	L23-RP41000	ItemFolder_6805	Item	SuppliesAllProducts	
3	8046	RP4100	ItemFolder_6805	Item	SuppliesAllProducts	SAProducts

The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

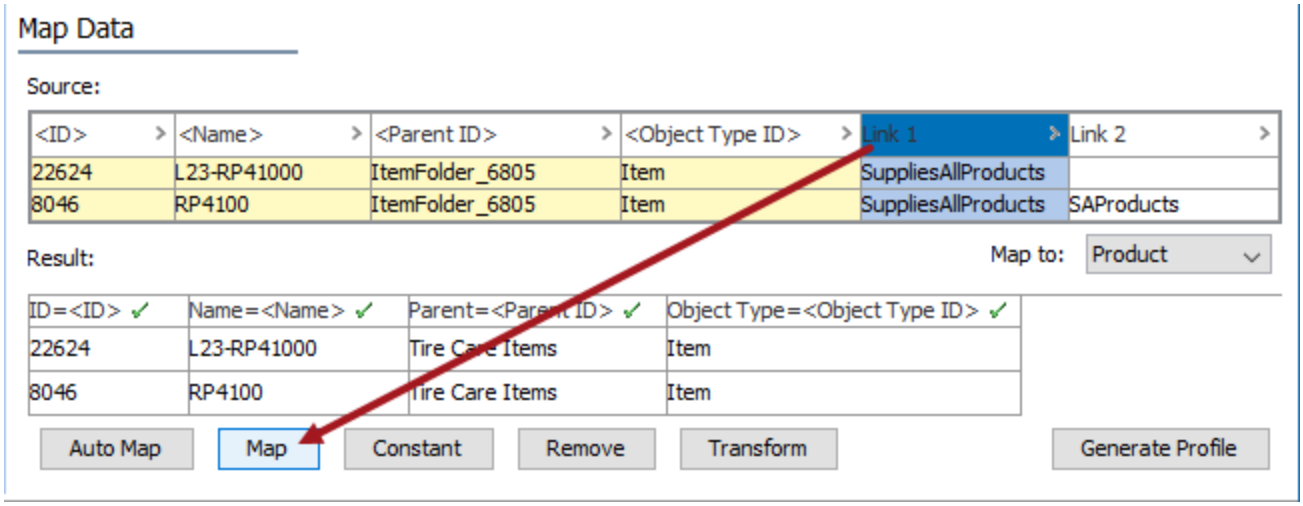
Note: Replacing a single Product Classification link during import requires the use of the ReplacementRules tag in STEPXML. Replacement during import is not available for other formats. Refer to the ReplacementRules Tag in STEPXML in the Data Exchange documentation.

Map Product Classification

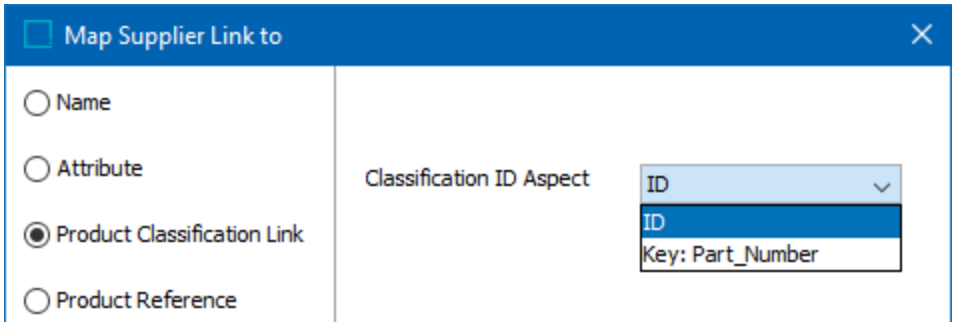
1. Below the Source section, use the **Map to** dropdown to select the **Product** object super type.
2. Map the ID or key column using the Auto Map button, or manually. For more information, refer to the **ID or Key - Map Inbound** topic.

   Columns that include Predefined Header Text can be automatically mapped as defined in the **Inbound Map Data - Auto Map** topic. For this sample file, this includes the <ID>, <Name>, <Object Type ID>, and <Parent ID> columns.

3. In the Source section, select the classification ID column of data and click the **Map** button.



4. In the 'Map ... to' dialog, select the **Product Classification Link** radio button.
   For the **Classification ID Aspect** dropdown, select:
   - **ID** if the file includes the object ID
   - **Key** if the file includes a unique key. Only activated keys are displayed.



Note: Each classification object type can have only one product-to-classification link type assigned. STEP identifies the classification by the ID or key and applies the single valid link type available for the mapped product.

5. Check the **Mandatory** option appropriately:
   - If **checked**, imported objects must have an attribute value in the file. If an attribute value does not exist for an object, the object is skipped and not imported.
   - If **unchecked**, all objects are imported regardless if an attribute value is specified or not.

6. Map additional product-to-classification links if necessary.
7. Click **OK** and the **Result** panel displays your selection of column and attribute. A validated data column is marked with a green check mark as shown below.

Result: Map to: Product

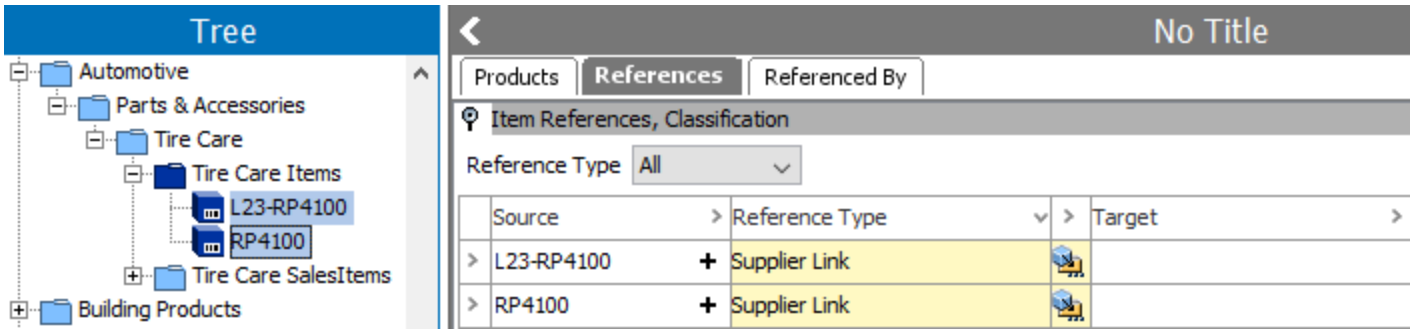
ID=<ID> ✓	Name=<Name> ✓	Parent=<Parent ID> ✓	Object Type=<Object Type ID> ✓	Classification=Link 1 ✓	Classification=Link 2 ✓
22624	L23-RP41000	Tire Care Items	Item	SASProducts	
8046	RP4100	Tire Care Items	Item	SASProducts|SAProducts	

Note: Although the data is not displayed in the Results panel for links after the first one, it is imported as demonstrated in the example below.

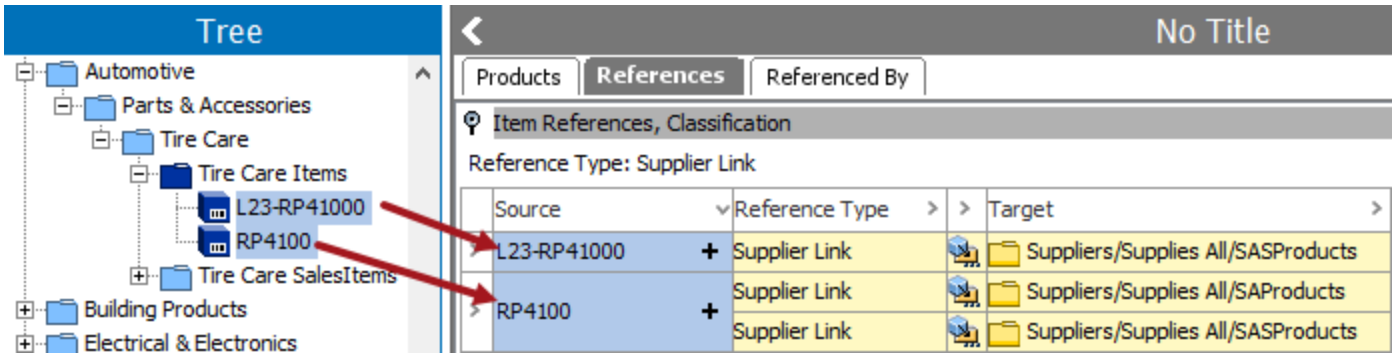
8. Complete the mapping and initiate the import.

Example

Prior to import, the products show no product-to-classification links in the editor.



After a successful import, new links are displayed on the product's References tab.



Product Reference - Map Inbound

You can reference products, classifications, or entities IDs or Unique keys listed in one column of the input file to another column of products IDs or Unique keys in that file, and make a product reference between them in STEP. When you make this selection, you must specify which reference type to use and the reference type should be valid for the object types. The available reference types are based on the product reference types that exist in STEP.

When using the Product Reference option, consider the following information:

- When mapping references, you must have at least two columns in your input file. One with the ID or unique key of the source object, and the other wit the ID or unique key of the target object. When you map these columns, you can specify the reference type that should be using to link the objects.
- The reference types listed in the wizard are the list of existing reference types. You cannot specify or create new reference types at this point (during the import).
- When making product-to-product references, you must have at least two columns in your input file. One with the ID of the source product, and the other with the ID of the target product. When you map these columns, you can specify the reference type that should be used to link the products.
- When making product-to-product references, you cannot specify the product reference type in the input file. You must select the product reference type in the wizard screen. The product reference type should be valid for the object types in the input file. In a single load operation, you can only make one type of product-to-product reference. You can, however, load the same file again and specify a different product reference type.
- For product-to-product and product-to-asset references, the list of reference types listed in the wizard is the list of existing reference types. You cannot specify or create new reference types at this point.

The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

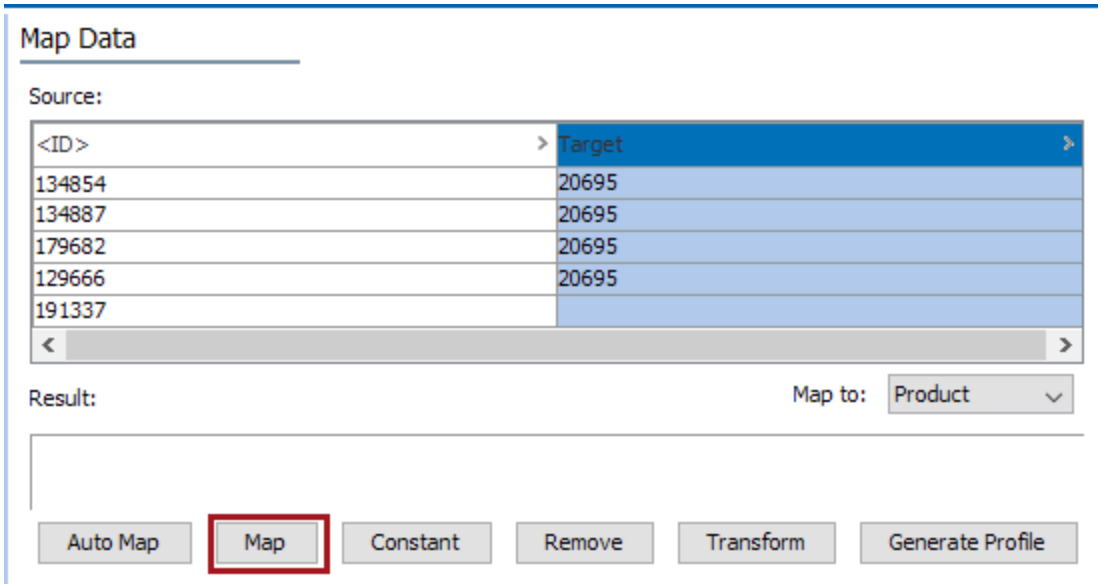
Prerequisites

If the objects to be imported or the Target to be mapped is identified via a unique key, you must verify that the Key required is available and activated. For more information, refer to the **Activating and Deactivating Keys** topic of the **System Setup** documentation.

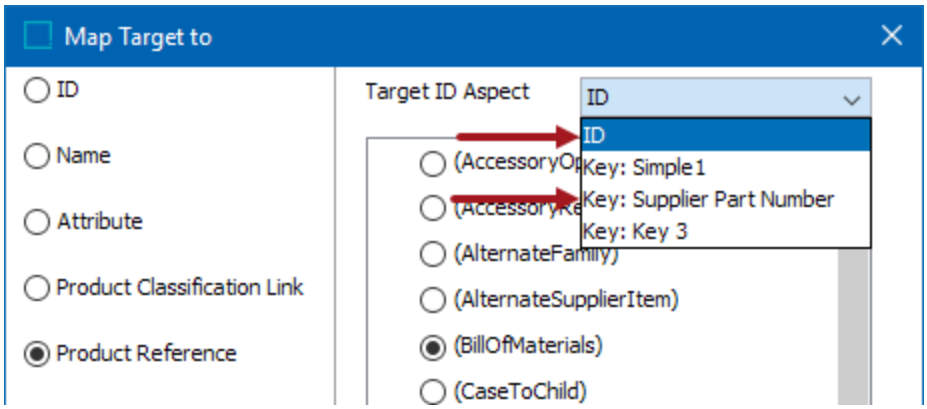
The following example assumes that the 'Allow multiple references parameter' on the Reference Type is set to Yes, which means that multiple products can be linked to the same product reference. For more information, refer to the **Multiple References for the Same Objects** topic within the **System Setup** documentation.

Map Product Reference

1. In the Source window, select the ID or Key column and map it. For more information refer to the **ID or Key - Map Inbound** documentation.
2. Select the reference target column in the Source section and then click the 'Map' button.

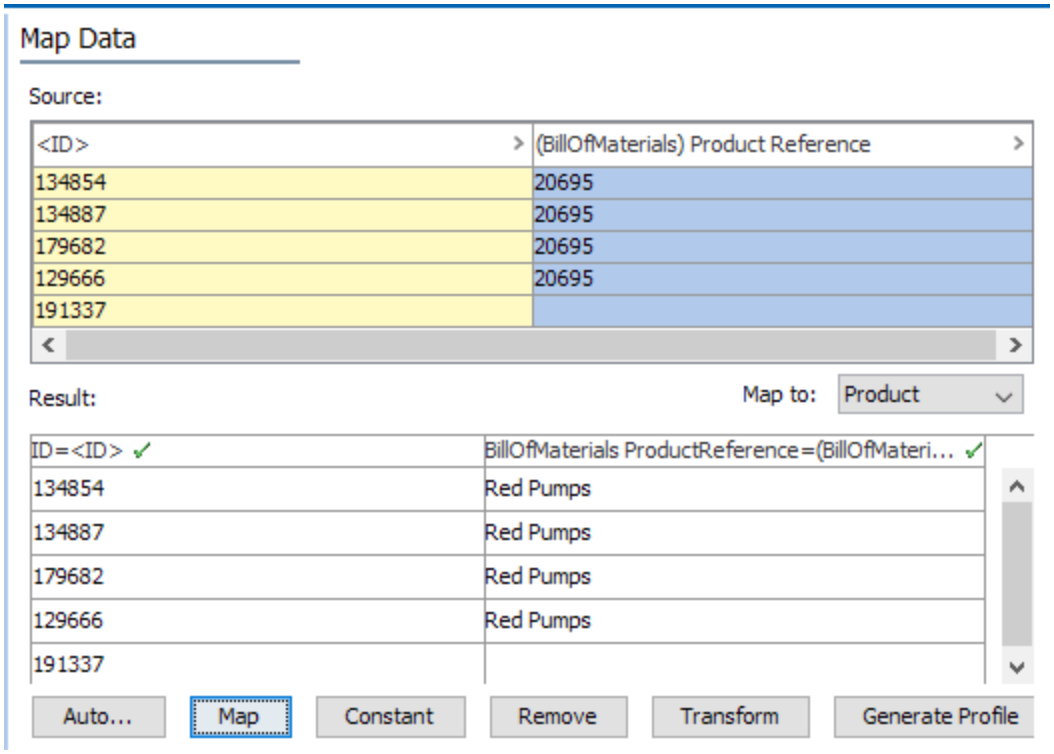


3. In the 'Map ... to' dialog, the label of the import field is used as the title. Make the following selections:
   - Select the **Product Reference** radio button option.
   - In the 'Target ID Aspect' dropdown list, select either ID or the appropriate key. Refer to the **Prerequisites** section above if mapping by key.
   - Select the desired Reference type.

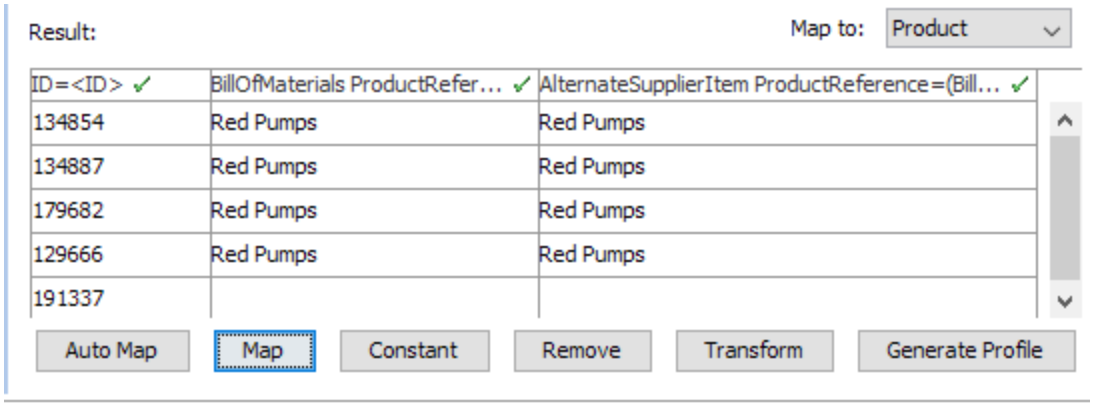


4. Check the **Mandatory** option appropriately:
   - If **checked**, imported objects must have a Target ID in the file. If a Target ID does not exist for an object, the object is skipped and not imported.
   - If **unchecked**, all objects are imported regardless if a Target ID is specified or not.

5. Click the **OK** button to display the mapping in the Result section.



6. If needed, map the same Target column again to a different product reference type.

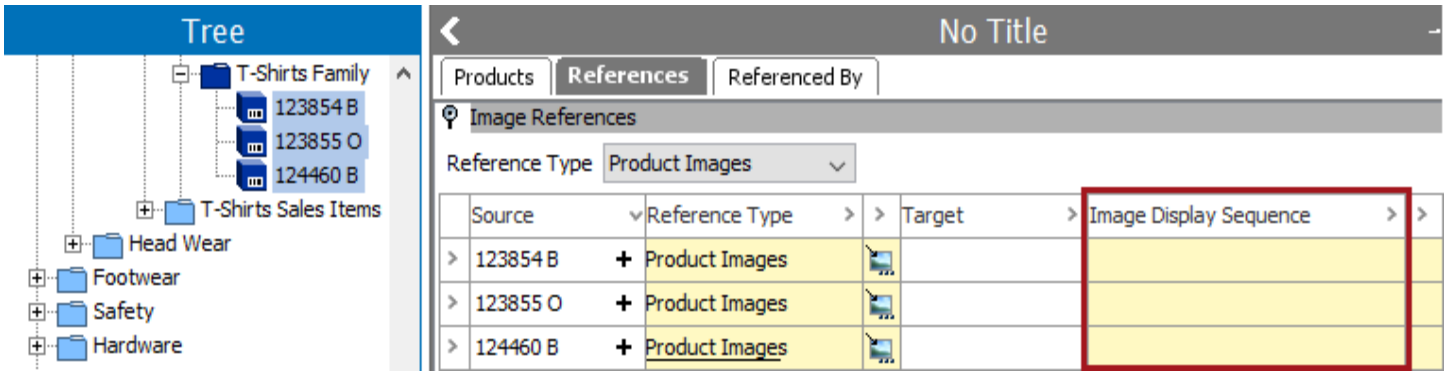


Reference Meta-Data - Map Inbound

Attribute values can be imported as metadata for all reference types or link types.

The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

For example, viewing the References tab on the selected products shows the 'Image Display Sequence' metadata attribute on the 'Product Images' reference type.



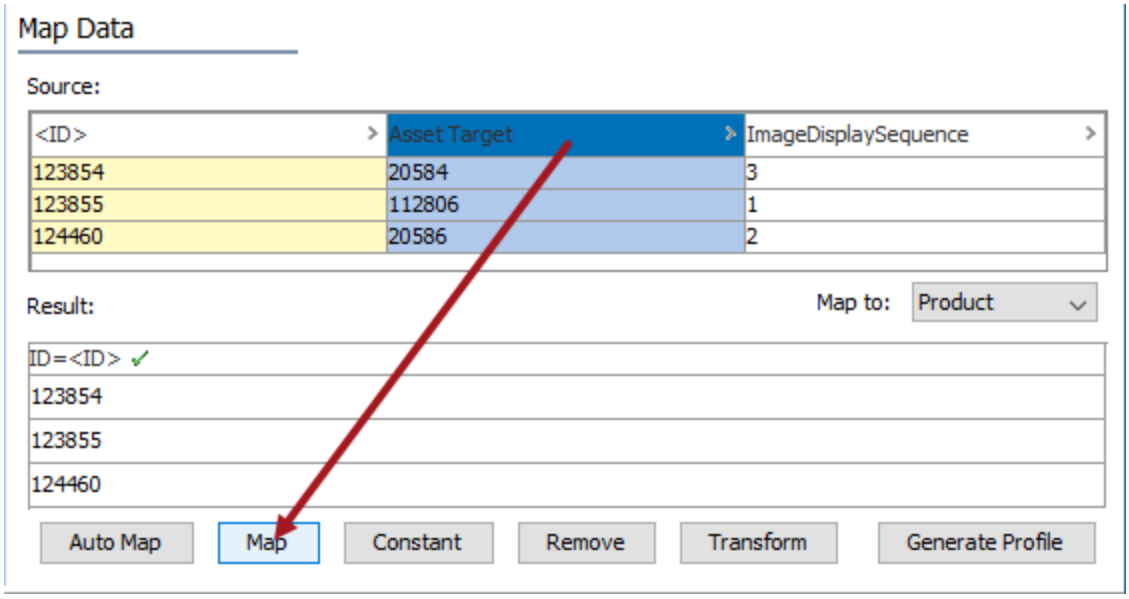
To import metadata attribute values into the 'Image Display Sequence' metadata attribute, an input file must contain columns holding the metadata attribute values, the Product IDs, and the referenced target asset IDs.

Map Reference Metadata

1. Map the source ID or key column using the Auto Map button, or manually. For more information, refer to the **ID or Key - Map Inbound** topic.

   Columns that include Predefined Header Text can be automatically mapped as defined in the **Inbound Map Data - Auto Map** topic. For the sample below, this includes the <ID> column.

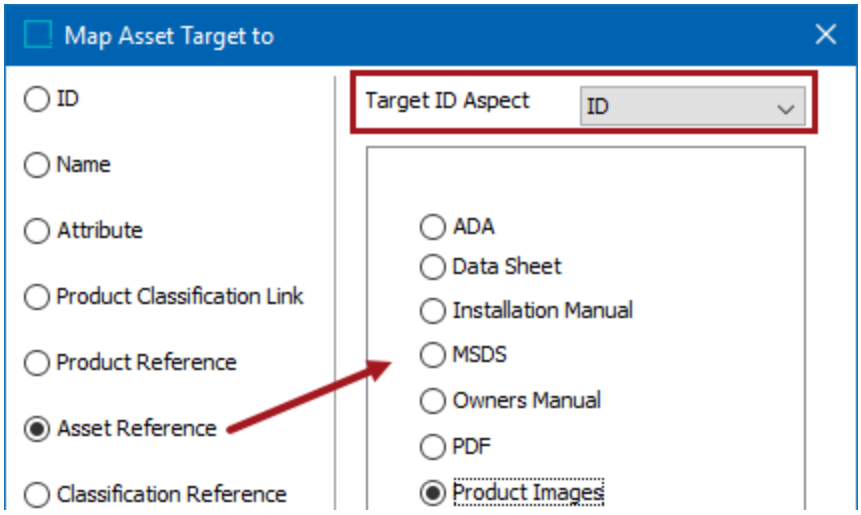
2. In the Source section, select the target column of data and click the **Map** button.



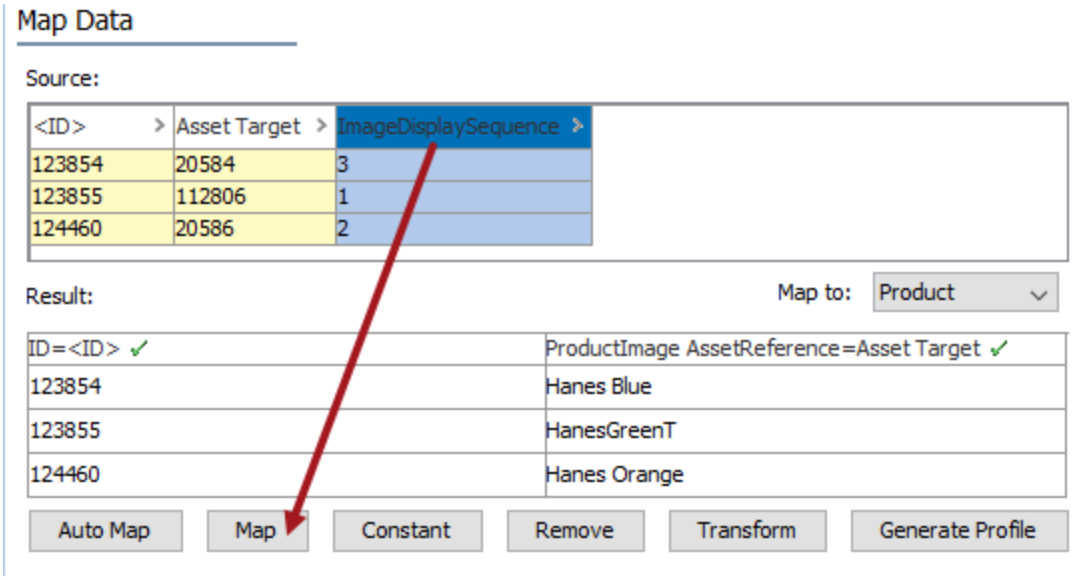
3. In the 'Map ... to' dialog (below), select the **Asset Reference** radio button, and then select the reference type.

   Only existing reference types are listed. You cannot specify or create new reference types during the import.

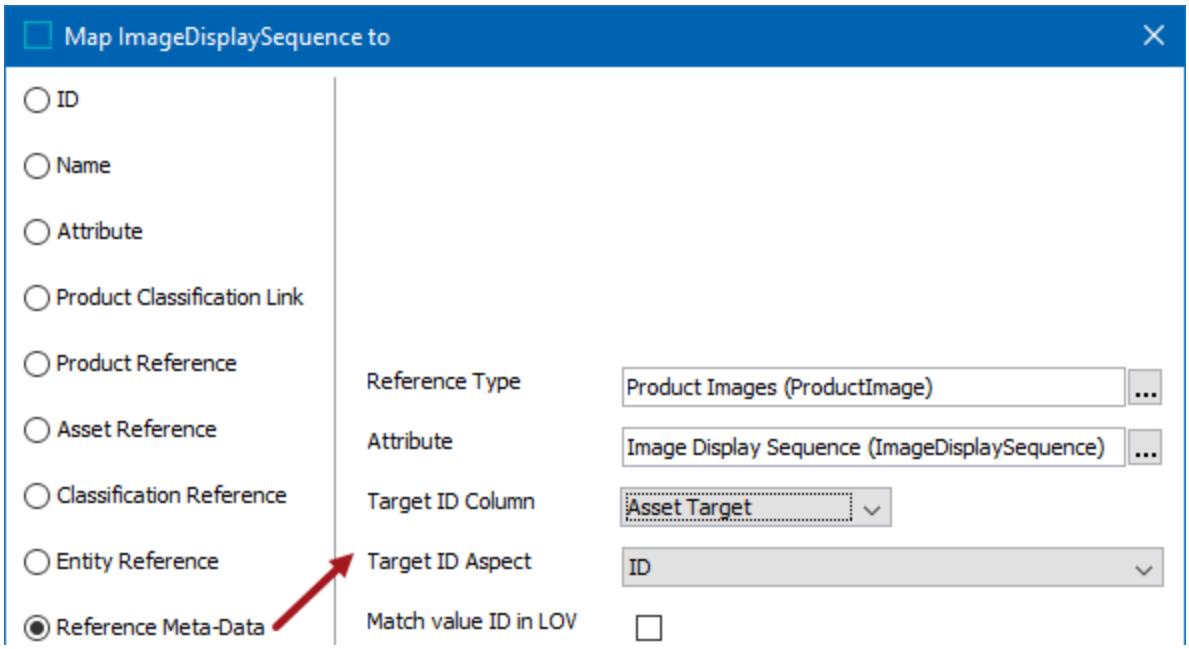
4. For the **Target ID Aspect** parameter:
   - Select **ID** when the inbound file includes the ID for the target.
   - Select the **Key** from the dropdown when the inbound file includes a key for the target.



5. Click **OK** and the **Result** panel displays the selection of reference asset target. A validated data column is marked with a green check mark as shown below.
6. In the Source section, select the metadata attribute column of data and click the **Map** button.



7. In the 'Map ... to' window, select the **Reference Meta-Data** radio button to display the additional parameters.

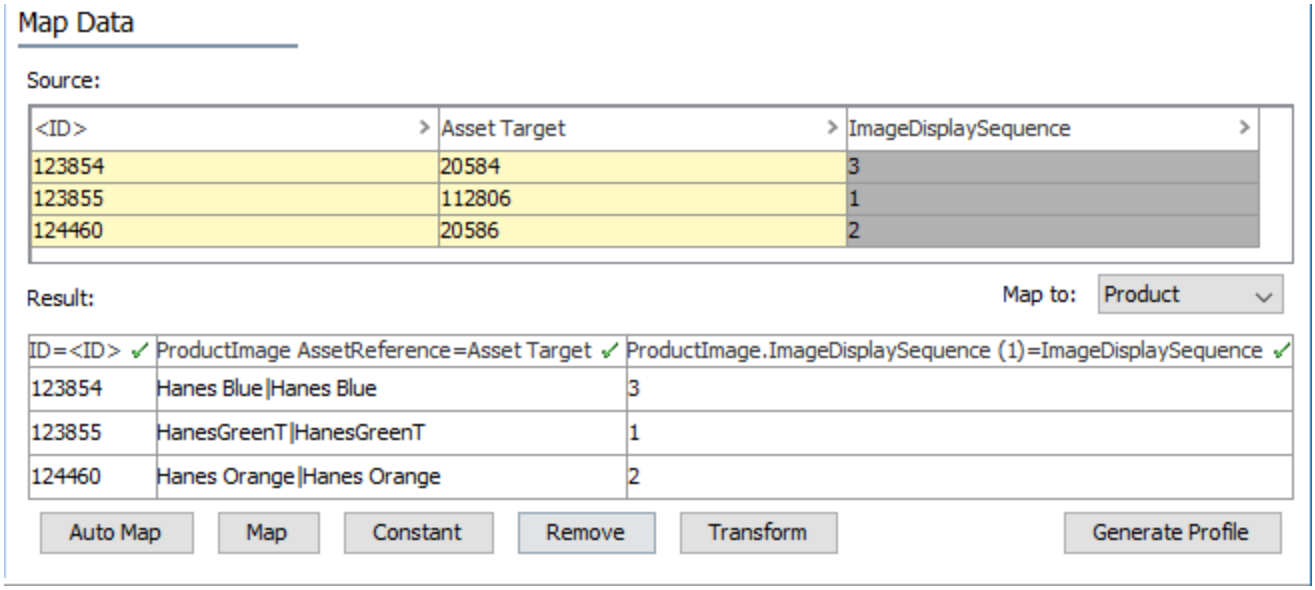


8. Complete the additional parameters as follows:
   - For **Reference Type**, select the type of reference that should receive the metadata values. In this example, the asset reference type 'Product Images' is selected.
   - For **Attribute**, select the attribute that should receive the metadata values. In this example, the attribute 'Image Display Sequence' is selected.
   - For **Target ID Column**, from the dropdown, select the column in the import file that includes the ID of the reference target. In this example, the 'Asset Target' column in the input file is selected.

- For **Target ID Aspect**, from the dropdown, select **ID** when the inbound file includes the ID for the target; select the **Key** when the inbound file includes a key for the target.
- For **Match value ID in LOV**, if the metadata attribute holds an LOV with value IDs, the import file can also hold the value IDs. If checked, the ID of the LOV must be included in the inbound file; if unchecked, the LOV ID is not required.

9. Check the **Mandatory** option appropriately:
   - If **checked**, imported objects must have an attribute value in the file. If an attribute value does not exist for an object, the object is skipped and not imported.
   - If **unchecked**, all objects are imported regardless if an attribute value is specified or not.

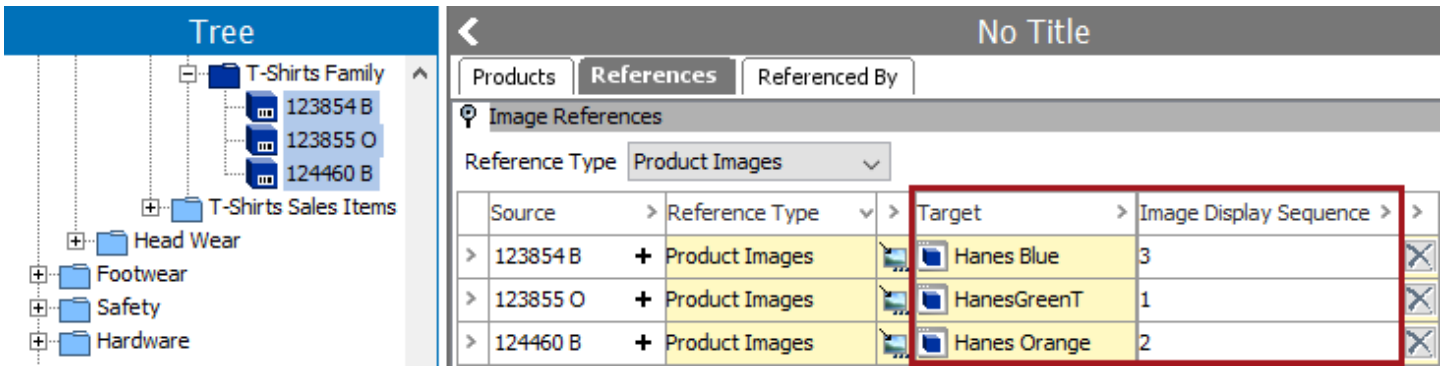
10. Click **OK** and the **Result** panel displays your selection of reference target. A validated data column is marked with a green check mark as shown below.



11. Complete any additional mapping.
12. Initiate the import.

Results

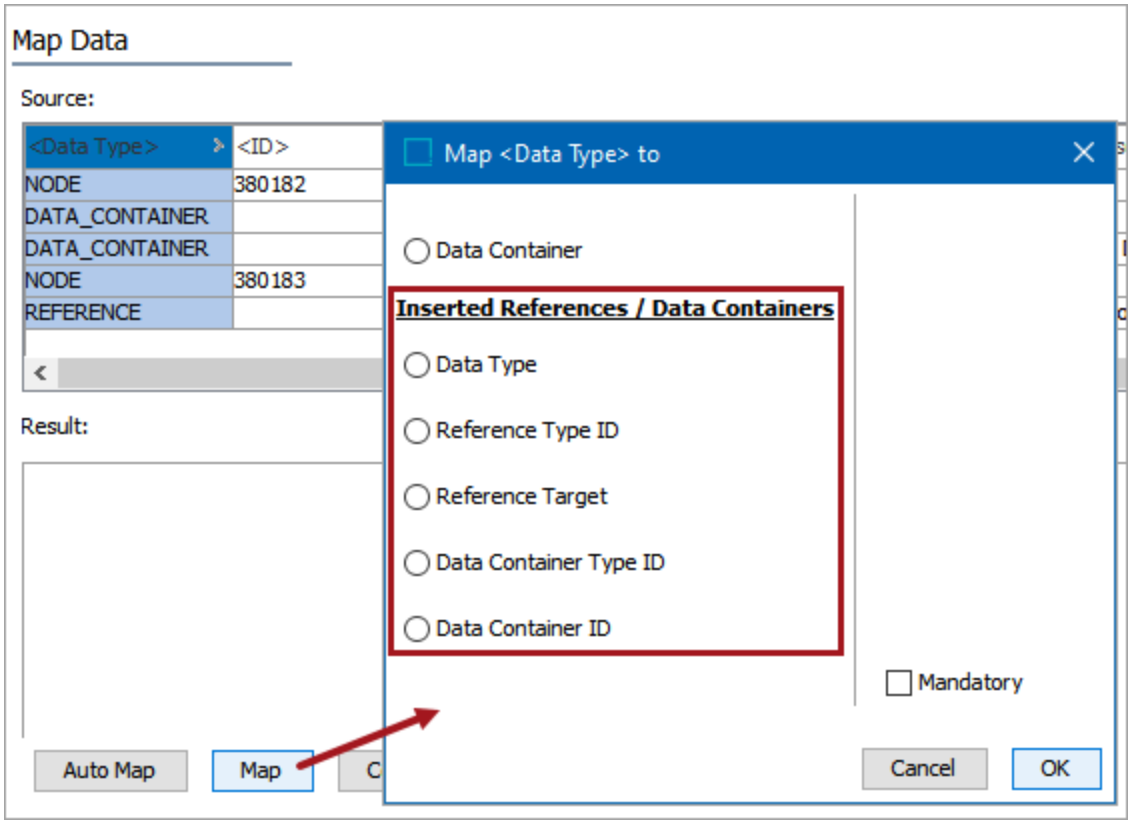
In the Tree, selecting the updated products and displaying the 'Product Images' reference type shows the new referenced objects as well as the metadata on the reference.


Tree
T-Shirts Family
123854 B
123855 O
124460 B
T-Shirts Sales Items
Head Wear
Footwear
Safety
Hardware
No Title
Products
References
Referenced By
Image References
Reference Type
Product Images
Source
Reference Type
Target
Image Display Sequence
123854 B
Product Images
Hanes Blue
3
123855 O
Product Images
HanesGreenT
1
124460 B
Product Images
Hanes Orange
2

Reference Target - Map Inbound

Representing references or product classification links in separate rows provides a simplified layout with clear relationships between the objects. The 'Reference Target' source can be used to identify the reference target for the reference or link to create or update.

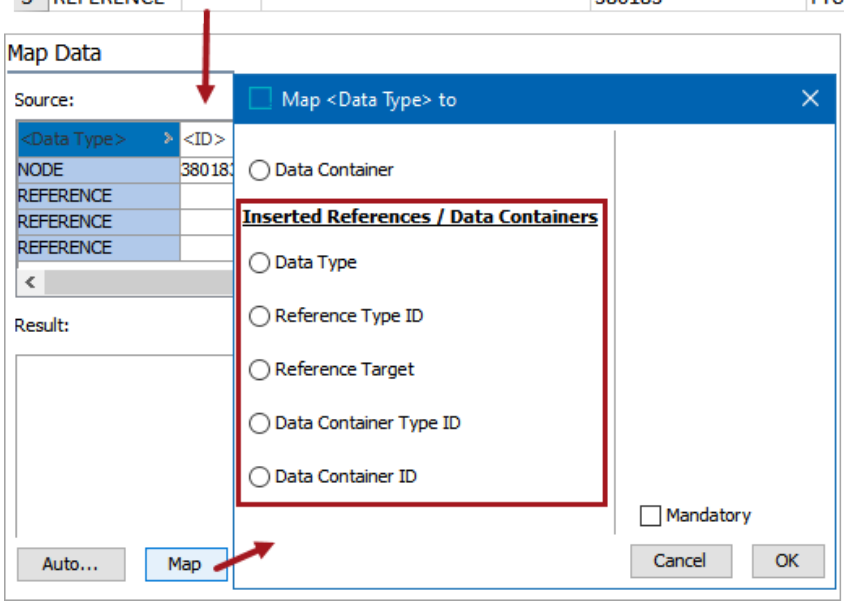
This source mapping option is available in the 'Inserted References / Data Containers' section on the 'Map To' dialog. For details on using the options in this group, refer to the **Reference Type ID - Map Inbound** topic and the **Data Container Type ID - Map Inbound** topic.



Reference Type ID - Map Inbound

Simplified Excel or CSV file layouts use a single row for each reference owned by the imported product, asset, classification, or entity object. This type of file is mapped with the 'Inserted References / Data Containers' section of the 'Map...to' dialog. Refer to the **Prerequisites** section below for required use of this mapping option.

	A	B	C	D	E	F	G
1	<Data Type>	<ID>	<Name>	<Data Owner Node>	<Reference Type ID>	<Reference Target ID>	Effective Date
2	NODE	380183	Crayola Colored Pens, 24 Count				2022-09-01
3	REFERENCE			380183	ProductToSupplierLink	Supplier_Acme	2022-01-01
4	REFERENCE			380183	ProductToSupplierLink	Supplier_Alpha	2022-06-01
5	REFERENCE			380183	ProductToSupplierLink	Supplier_Beta	2022-06-01



The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Prerequisites

References and data containers owned by the imported object can be included in the same file. Refer to the **Data Container Type ID - Map Inbound** topic for details on importing data containers.

The following must be included in the import file and mapped during import configuration:

1. The **Data Type** source with one of the following expected values on each row:
   - NODE - the row contains the product, asset, classification, or entity being imported. NODE must be followed by row(s) for related references and data containers.

Important: The NODE row is the object being imported. The references and data containers below a NODE row in the file are added to or updated on the imported NODE object.

- REFERENCE - the row contains data for a reference owned by the preceding NODE.
- DATA_CONTAINER - the row contains data for a data container owned by the preceding NODE.

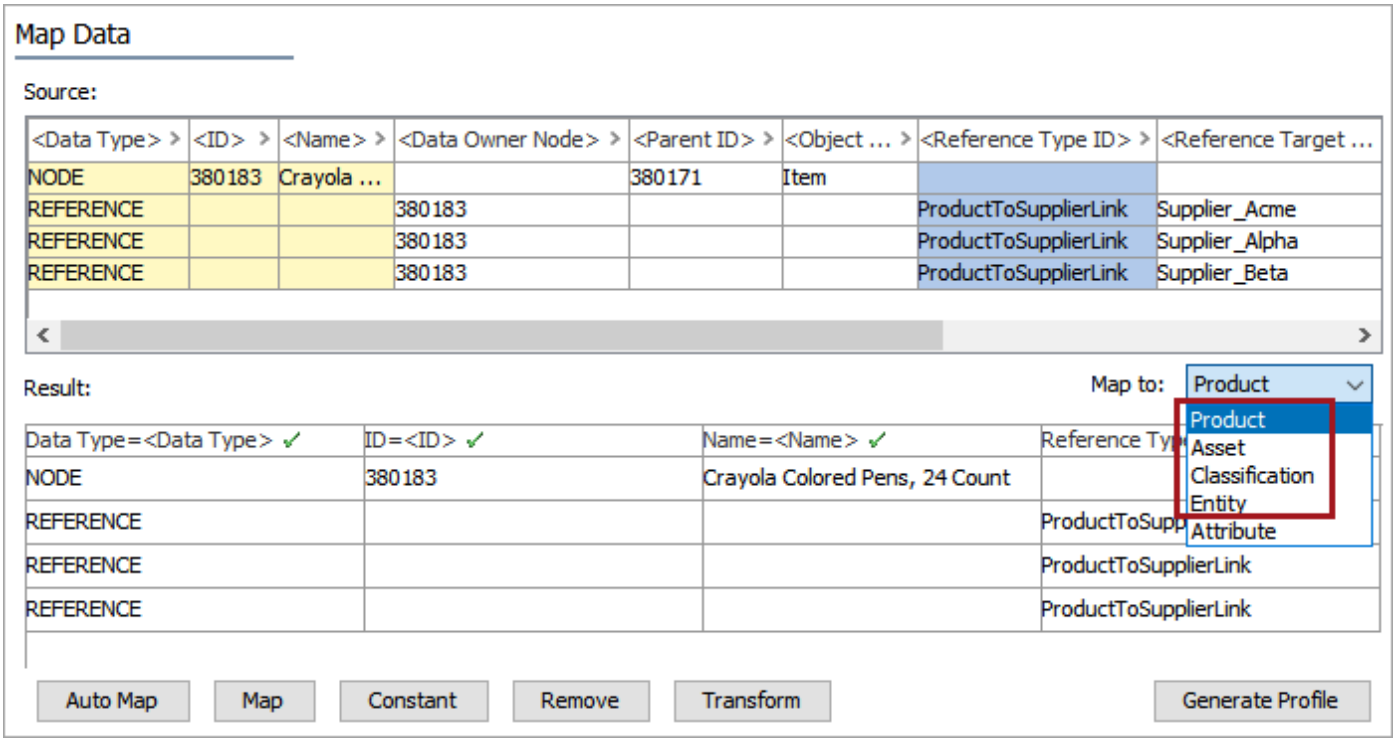
Note: Other values in the 'Data Type' column of the import file are imported as NODE and are processed as an object to import.

2. The **ID** of the NODE.
3. The **Reference Type ID**, which is the STEP ID of the reference type, is required when reference or classification product links are represented in a separate row to identify the reference or classification product link type. Reference Type ID must be used in combination with Data Type and Reference Target.
4. The **Reference Target**, the STEP ID or Key of the reference target object.

Map Inserted References

Using the following steps to map the Excel or CSV file for import:

1. Below the Source section, use the **Map to** dropdown to select product, asset, classification, or entity as the super type for the objects being imported. The 'Inserted References / Data Containers' mapping options are not intended for use when importing attributes.



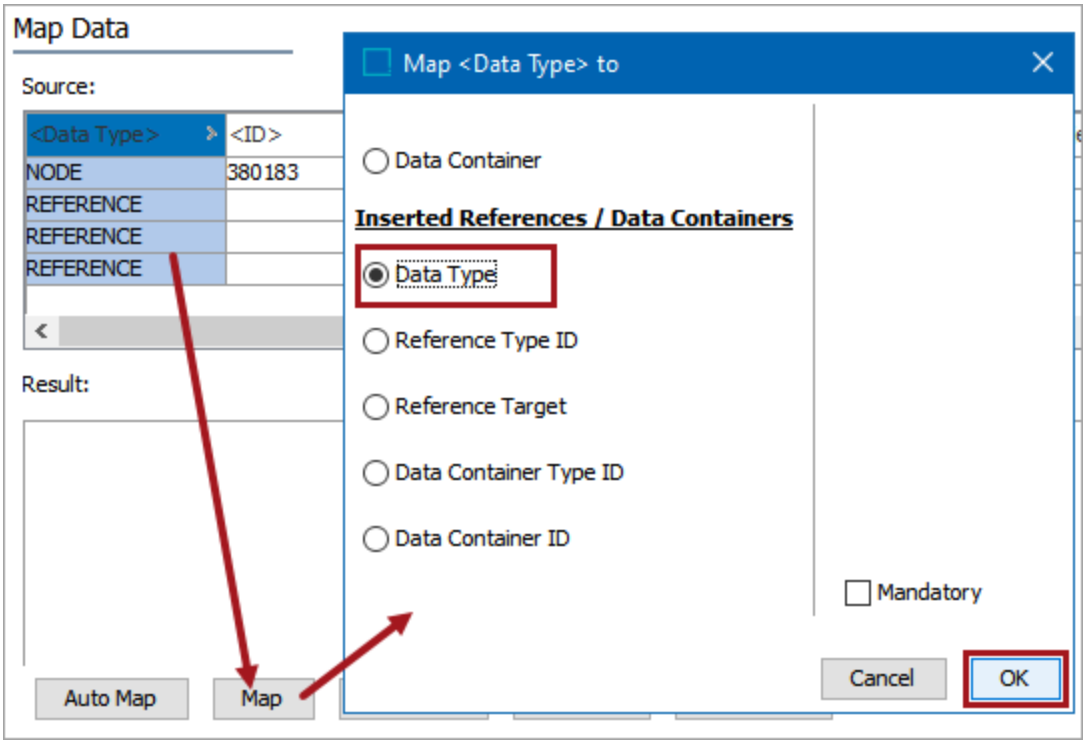
2. Map the source information in the import file:

   - Columns that include Predefined Header Text can be automatically mapped as defined in the **Inbound Map Data - Auto Map** topic.
   - Columns can also be manually mapped as defined in the following steps.

Note: Although the <Data Owner Node> can be mapped during output, it is not needed during import. Refer to the **Data Owner Node - Data Source Outbound** topic.

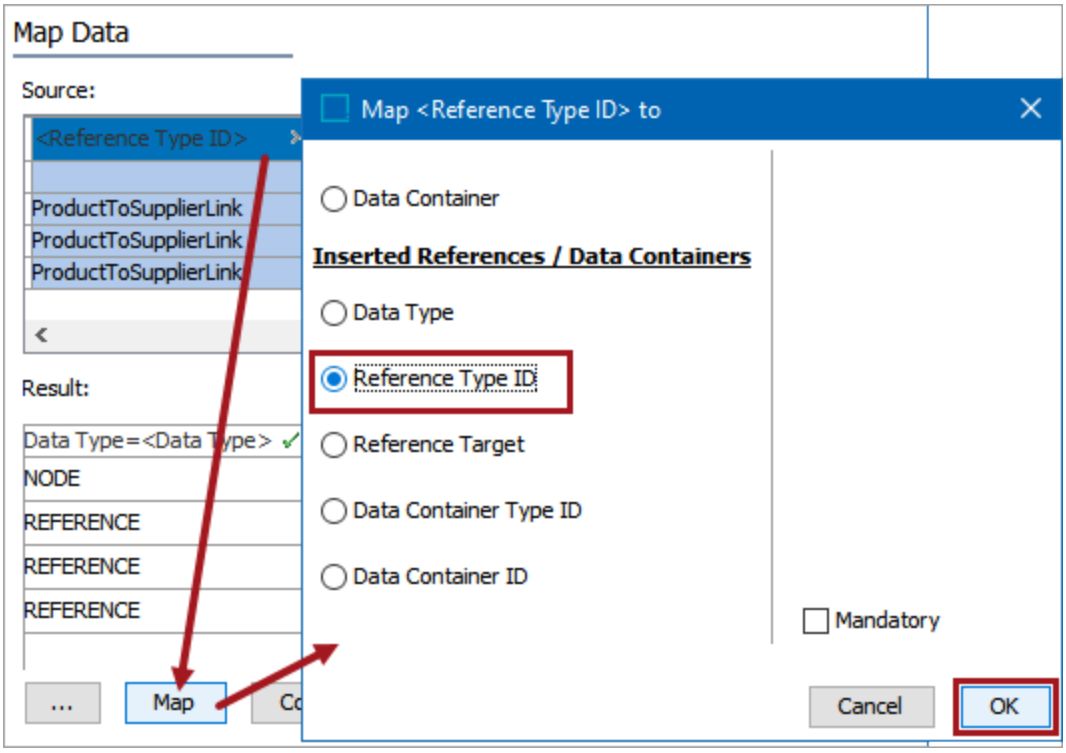
3. Map (or verify the mapping of) the required **Data Type** column (refer to the **Prerequisites** section above for requirements) and check the **Mandatory** option appropriately.

   - If **checked**, imported objects must have an attribute value in the file. If an attribute value does not exist for an object, the object is skipped and not imported.
   - If **unchecked**, all objects are imported regardless if an attribute value is specified or not.

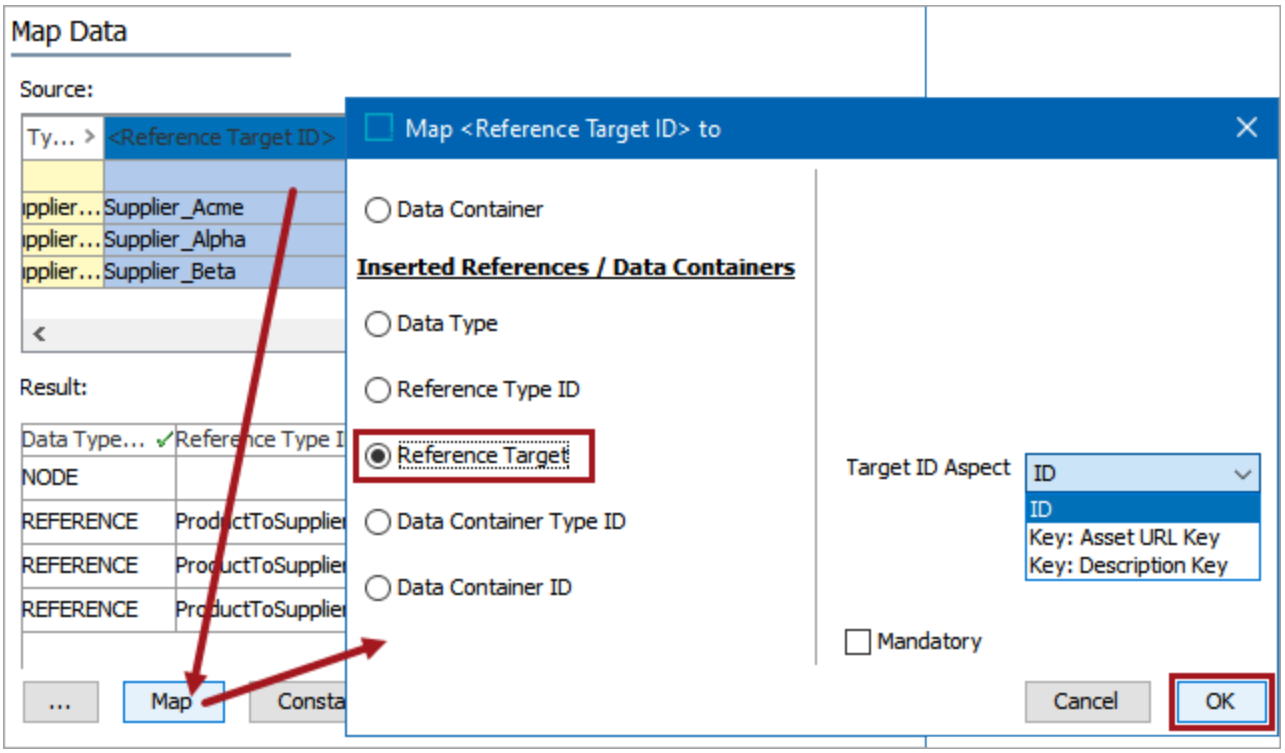


Important: The 'Data Type' source must be mapped to successfully import references owned by the node. No values are shown in the Result panel for references until the 'Data Type' source is mapped.

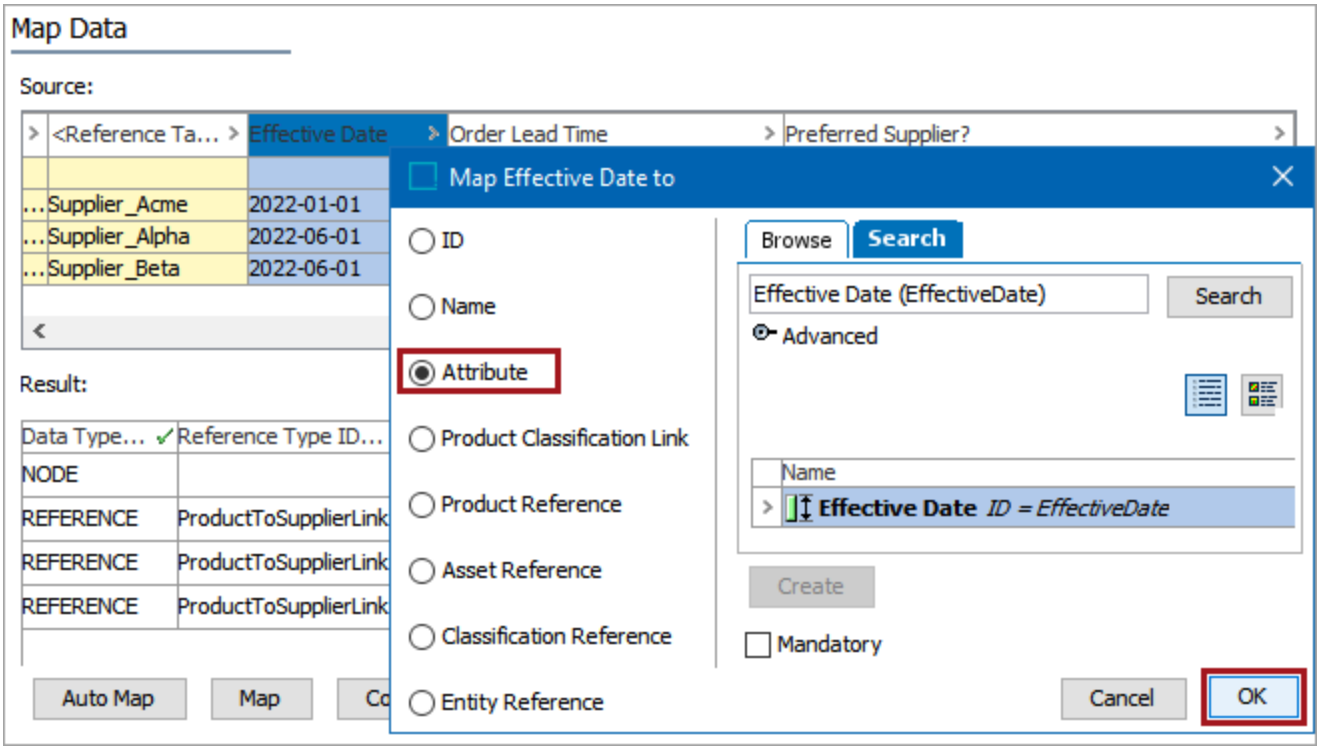
4. Map or verify the mapping of the required **Reference Type ID** column and check the **Mandatory** option appropriately (defined above).



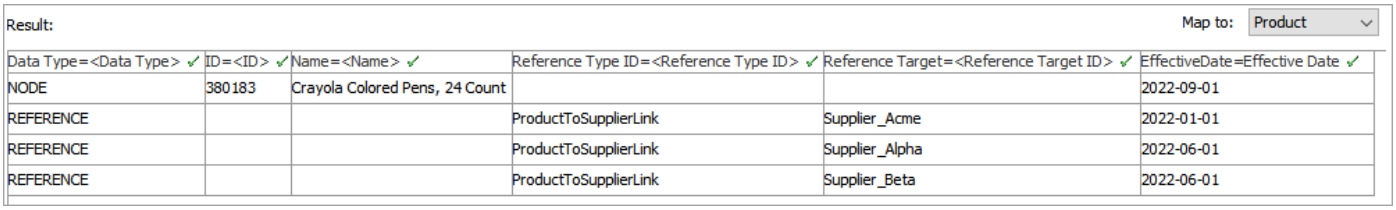
5. Map (or verify the mapping of) the required **Reference Target** column, from the **Target ID Aspect** dropdown, select the ID or Key (based on the data in the import file), and check the **Mandatory** option appropriately (defined above).



6. Map (or verify the mapping of) the desired metadata using the Attribute option on the 'Map...to' dialog. With this single mapping, values are imported for the selected attribute on all NODE, REFERENCE, and DATA_CONTAINER rows of the import file.



7. Review that the **Result** panel displays the inserted references as expected. Validated data columns are marked with a green check mark.



Result:

Map to: Product

Data Type=<Data Type> ✓	ID=<ID> ✓	Name=<Name> ✓	Reference Type ID=<Reference Type ID> ✓	Reference Target=<Reference Target ID> ✓	EffectiveDate=Effective Date ✓
NODE	380183	Crayola Colored Pens, 24 Count			2022-09-01
REFERENCE			ProductToSupplierLink	Supplier_Acme	2022-01-01
REFERENCE			ProductToSupplierLink	Supplier_Alpha	2022-06-01
REFERENCE			ProductToSupplierLink	Supplier_Beta	2022-06-01

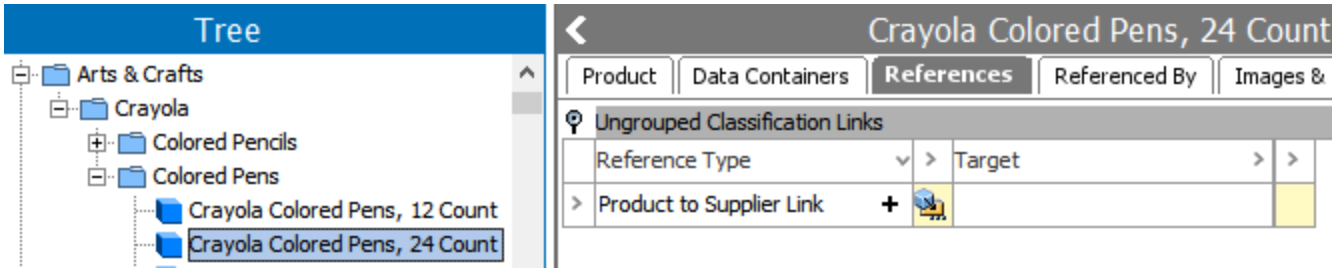
8. Complete any additional mappings or transformations.
9. Initiate the import.

Example

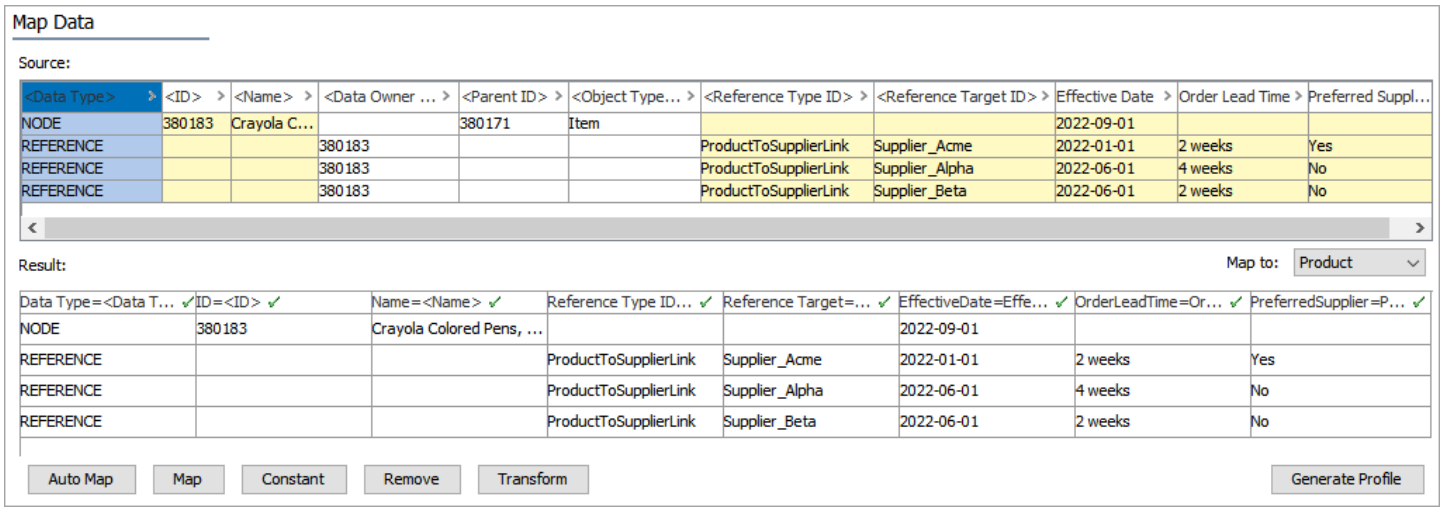
This sample file will be used to create product to classification links with metadata for the NODE object. Metadata is also imported to the 'Effective Date' attribute for the NODE object.

	A	B	C	D	E	F	G	H	I	J	K
1	<Data Type>	<ID>	<Name>	<Data Owner Node>	<Parent ID>	<Object Type ID>	<Reference Type ID>	<Reference Target ID>	Effective Date	Order Lead Time	Preferred Supplier?
2	NODE	380183	Crayola Colored Pens, 24 Count		380171	Item			2022-09-01		
3	REFERENCE			380183			ProductToSupplierLink	Supplier_Acme	2022-01-01	2 weeks	Yes
4	REFERENCE			380183			ProductToSupplierLink	Supplier_Alpha	2022-06-01	4 weeks	No
5	REFERENCE			380183			ProductToSupplierLink	Supplier_Beta	2022-06-01	2 weeks	No

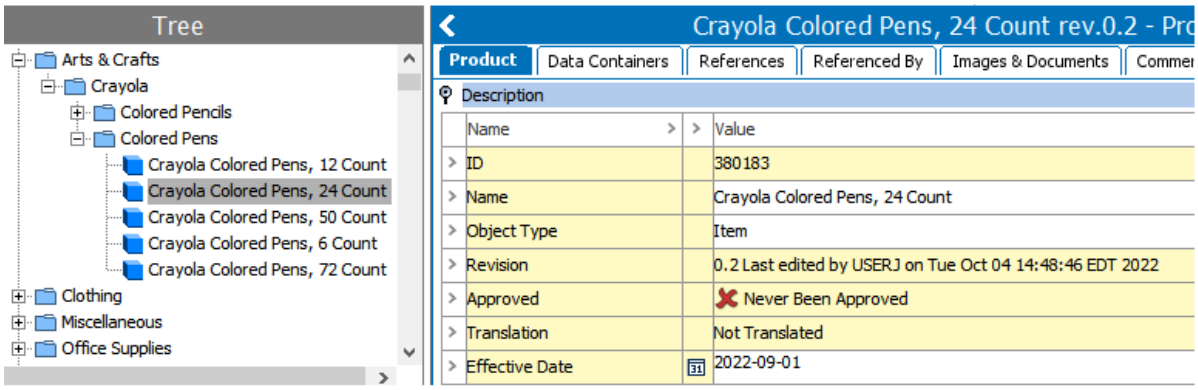
Before the import, the product has no values for the 'Product to Supplier Link' (ProductToSupplierLink).

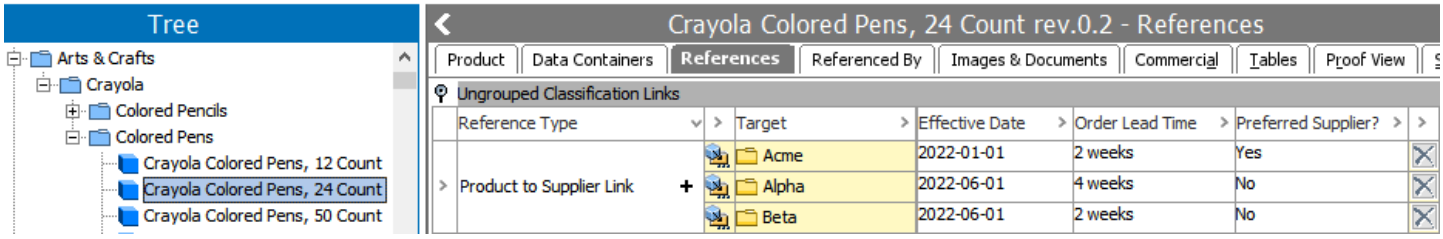


Select the product object super type, map the required sources, and invoke the import.



Once the import background process completes without error, the NODE object displays the new metadata value and the new links including their metadata values.


Tree
Arts & Crafts
Crayola
Colored Pencils
Colored Pens
Crayola Colored Pens, 12 Count
Crayola Colored Pens, 24 Count
Crayola Colored Pens, 50 Count
Crayola Colored Pens, 6 Count
Crayola Colored Pens, 72 Count
Clothing
Miscellaneous
Office Supplies
Crayola Colored Pens, 24 Count rev.0.2 - Pro
Product
Data Containers
References
Referenced By
Images & Documents
Description
Name
Value
ID
380183
Name
Crayola Colored Pens, 24 Count
Object Type
Item
Revision
0.2 Last edited by USERJ on Tue Oct 04 14:48:46 EDT 2022
Approved
Never Been Approved
Translation
Not Translated
Effective Date
2022-09-01


Tree
Arts & Crafts
Crayola
Colored Pencils
Colored Pens
Crayola Colored Pens, 12 Count
Crayola Colored Pens, 24 Count
Crayola Colored Pens, 50 Count
Crayola Colored Pens, 24 Count rev.0.2 - References
Product
Data Containers
References
Referenced By
Images & Documents
Commercial
Tables
Proof View
Ungrouped Classification Links
Reference Type
Target
Effective Date
Order Lead Time
Preferred Supplier?
Product to Supplier Link
Acme
2022-01-01
2 weeks
Yes
Alpha
2022-06-01
4 weeks
No
Beta
2022-06-01
2 weeks
No

Variable - Map Inbound

A variable temporarily holds modified data for additional transformation. Variables are only available when importing a format that requires mapping, and are not accessible or stored elsewhere in STEP. They are used to include specified math operations, or to append values to the input data before importing the new value into STEP.

For example, consider a file that includes values for a multivalued attribute, but each value is in a column of its own. A variable can hold the values, and a separator character, so that upon import, the attribute is updated with a true multivalued value.

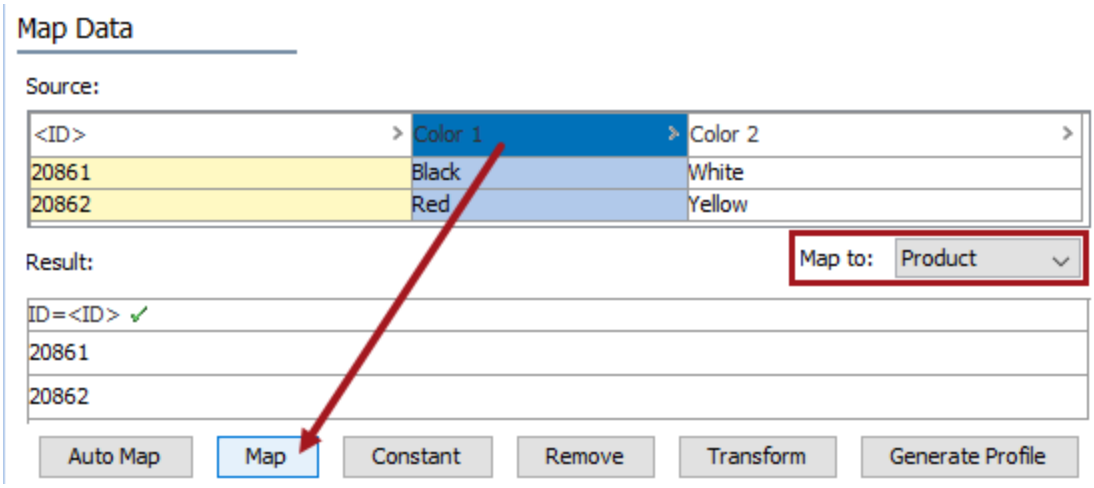
The Variable option is available for use with assets, attributes, classifications, entities, and products.

Variables, transformations, and mappings are included when saving an import configuration This makes it possible to use them again without having to recreate them. For more information about saving an import configuration, refer to the **Running a Data Import** topic in the **Import Manager** documentation. or the **IIEP - Configure STEP Importer Processing Engine** topic in the **IIEP** documentation.

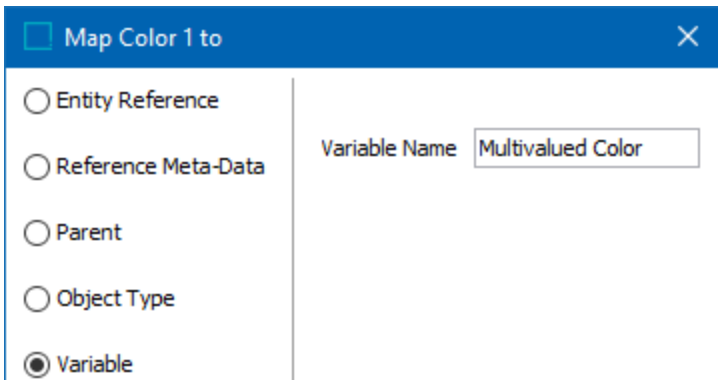
The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Map a Variable

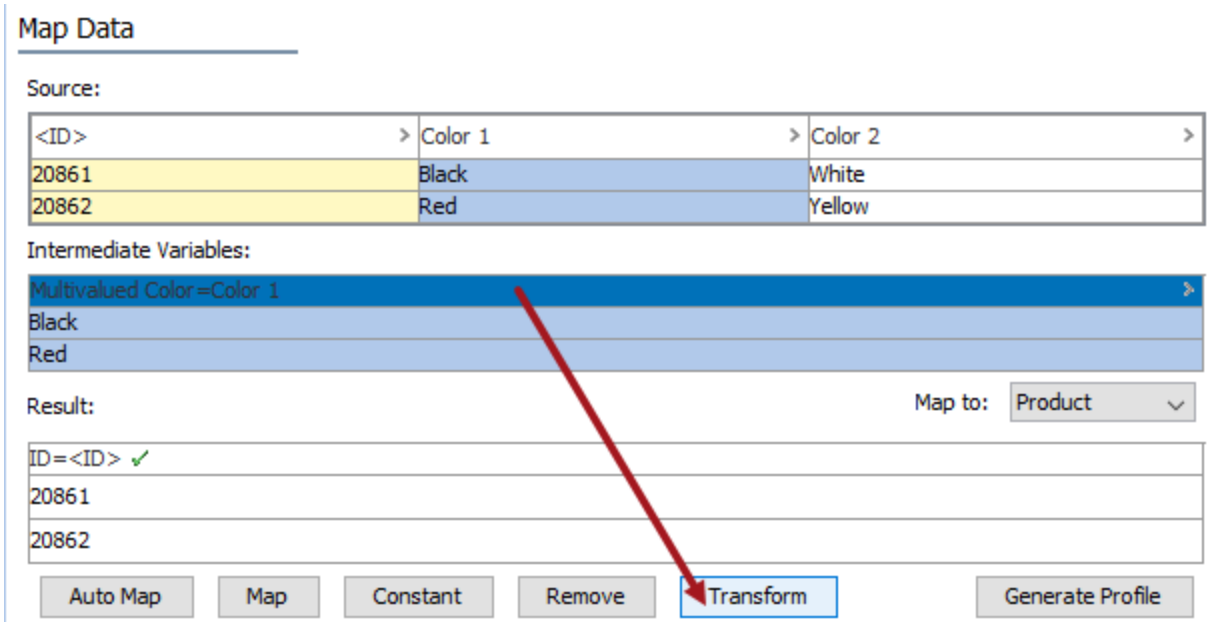
1. Below the Source section, use the **Map to** dropdown to select the object super type required by the import file.
2. In the Source section, select the column of data that you want to map to the variable and click the **Map** button.



3. In the 'Map ... to' dialog, select the **Variable** radio button and provide text to describe the purpose of the variable.

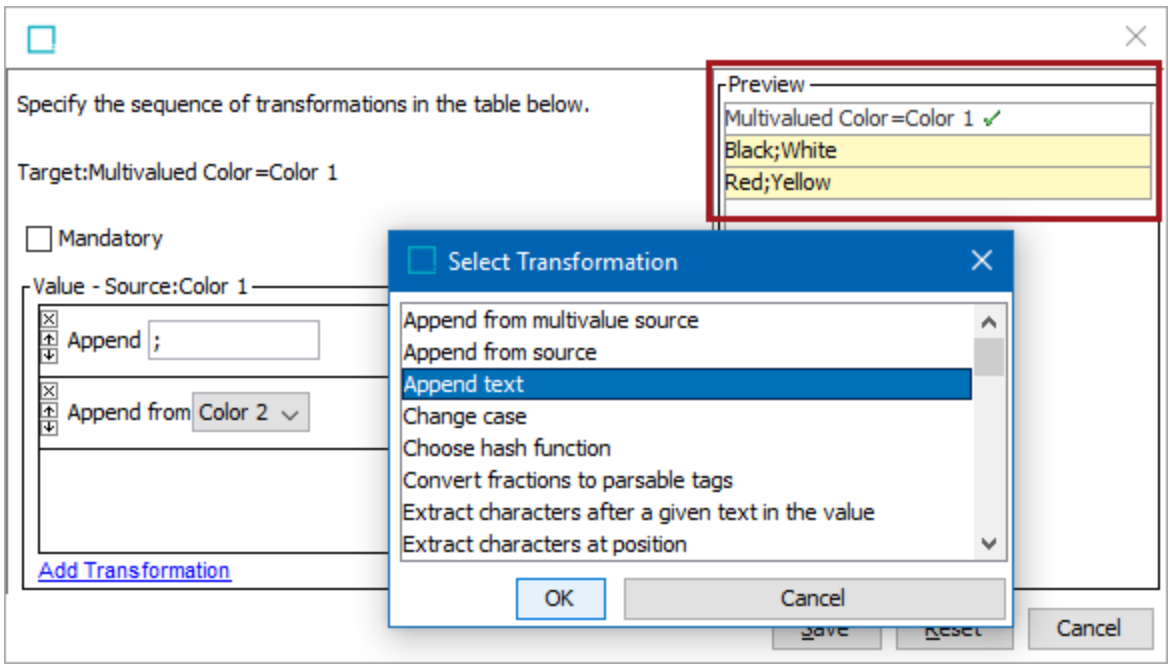


4. Click **OK** and the **Intermediate Variables** section is displayed on the Map Data dialog. The originally selected Source column is shown as mapped to the variable text provided. The data is now available for multiple transformations.

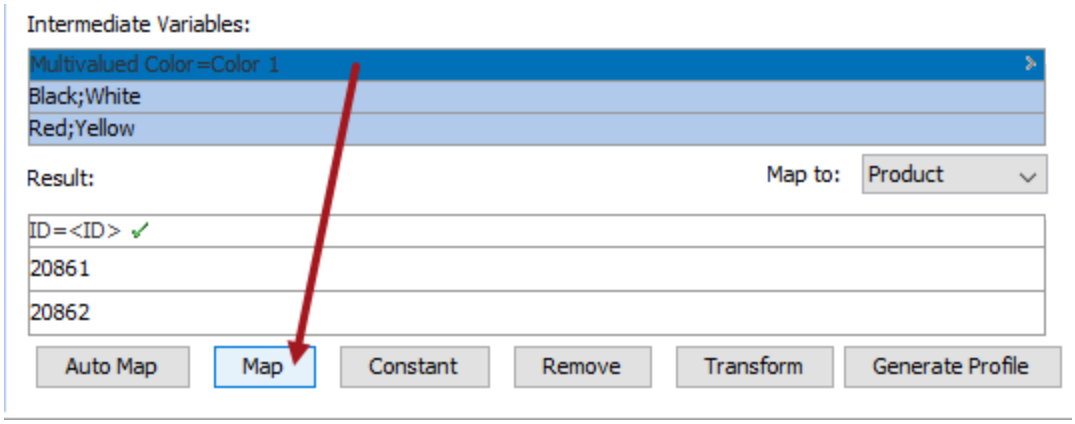


5. Select the intermediate variable column and click the **Transform** button. The transformations dialog is displayed. For more on the transformations, refer to the **Inbound Map Data - Transform** topic in the **Data Exchange** documentation.
6. Click the **Add Transformation** link to display the Select Transformation dialog.
7. Select the necessary transformation and provide any additional required information. For this exercise, we use **Append Text** and **Append from Source** as shown below.

   Continue to add all required transformations until the **Preview** panel shows the desired results.



8. Click **Save** on the transformations button to display the Preview results in the Intermediate Variables section.
9. Select the intermediate variable column and click the **Map** button.



10. In the 'Map ... to' dialog, select the **Attribute** radio button and select the attribute that will hold the transformed values, setting the Mandatory option as necessary. In this exercise, we select the Color attribute. For more information on mapping to attributes, refer to the **Attribute - Map Inbound** topic.
11. Click **OK** and the **Result** panel displays your selection of column (the intermediate variable Multivalued Color) and attribute (Color). A validated data column is marked with a green check mark as shown below.

Intermediate Variables:

Multivalued Color=Color 1
Black;White
Red;Yellow

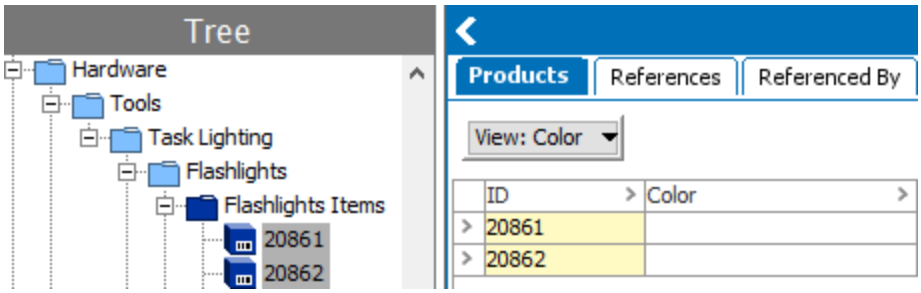
Result: Map to: Product

ID=<ID> ✓	Color=Multivalued Color ✓
20861	Black|White
20862	Red|Yellow

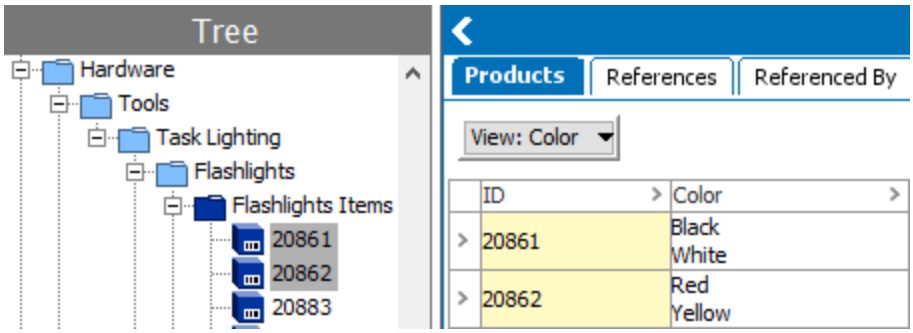
12. Complete the mapping and initiate the import.

Example

Consider two objects with a multivalued attribute named Color.



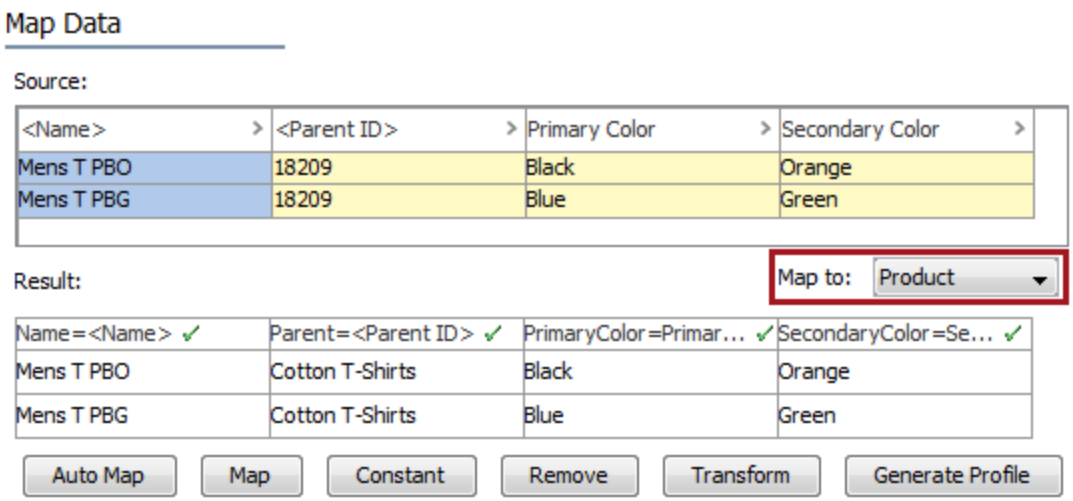
The import file includes two colors for each object, and the variable is used to hold the transformation that adds both colors to the same attribute.



For an example of the 'Append from source' transformation being used, refer to the **Concatenate Data Using Variables Example** topic in the **Data Exchange** documentation.

Inbound Map Data - Map To

The 'Map to' field allows you to import objects of the types product, asset, classification, entity, and attribute. Setting this field determines the options available when you click the Map button.



For more information about the additional import steps, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Mapping Based on Object Type

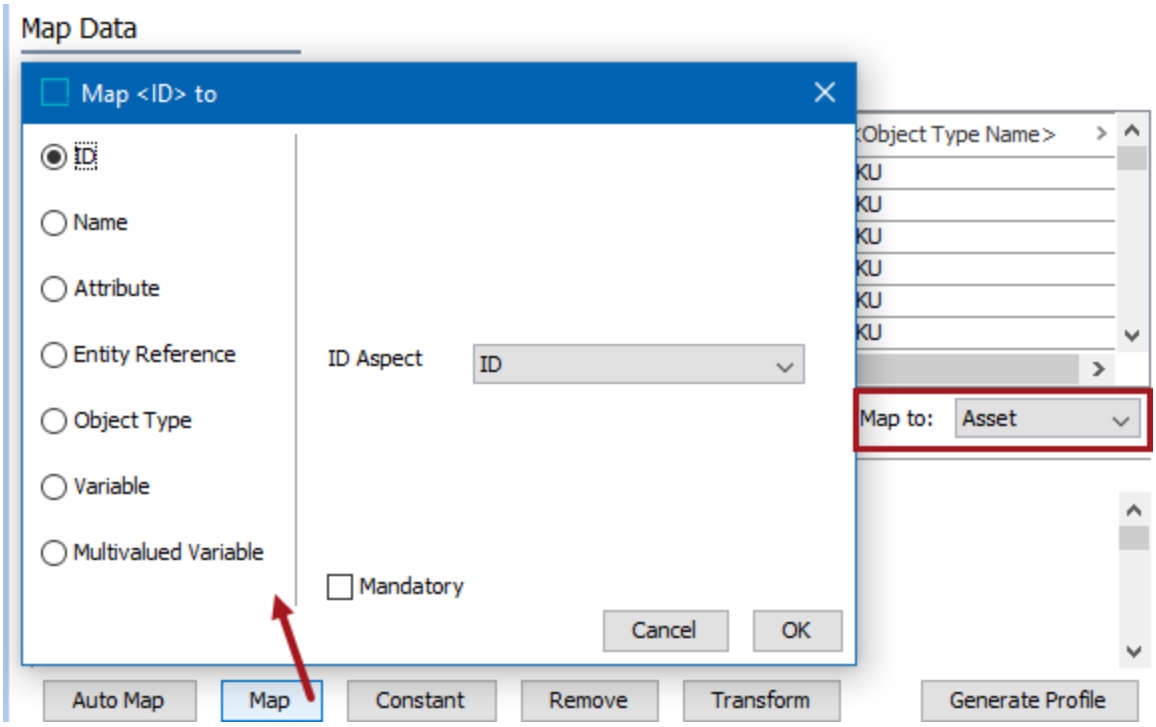
The following object types can be mapped:

- Asset Object Type - Map To Inbound
- Attribute Object Type - Map To Inbound
- Classification Object Type - Map To Inbound
- Entity Object Type - Map To Inbound
- Product Object Type - Map To Inbound

Asset Object Type - Map To Inbound

For inbound data, after selecting the Asset super type in the **Map to** dropdown, click the **Map** button to import information for assets. Asset content is not included in this data import, but can be added separately using the **Digital Assets** documentation.

The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.



For more information, refer to the **Map Inbound** topics on the available options:

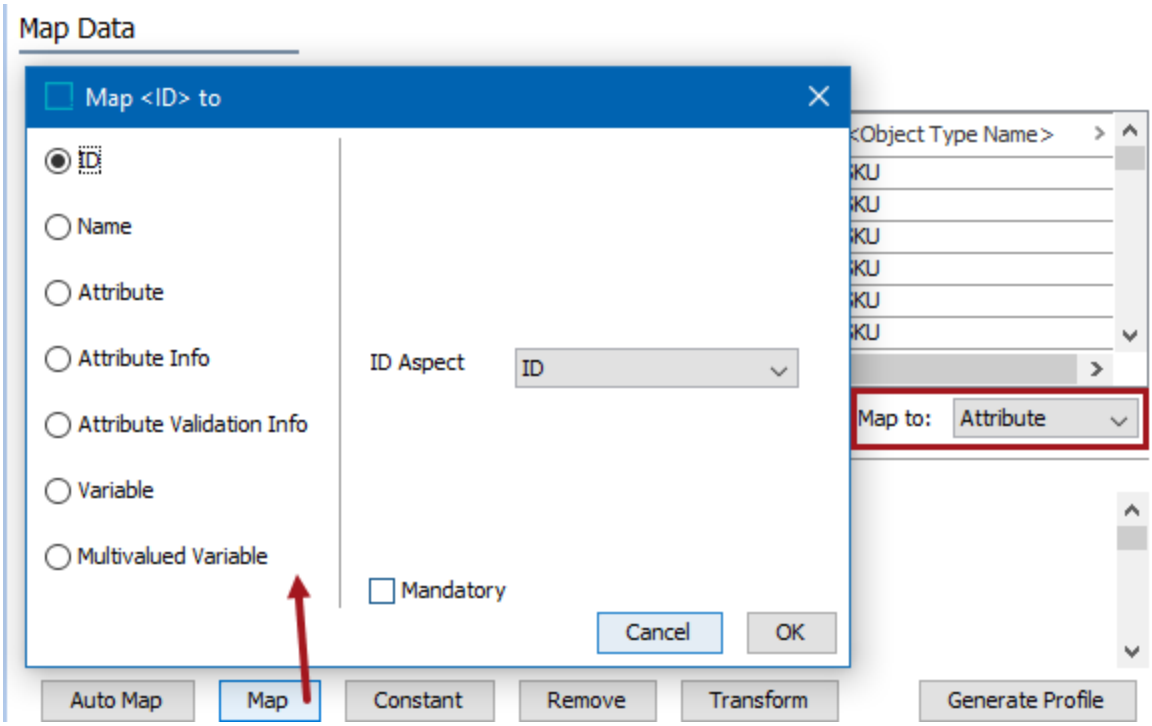
- ID or Key
- Name
- Attribute
- Entity Reference
- Object Type
- Variable
- Multivalued Variable

Attribute Object Type - Map To Inbound

For inbound data, after selecting the Attribute super type in the **Map to** dropdown, click the **Map** button to import information for attributes.

Note: When mapping to an attribute, it is common setup to only change the object type of a list of objects when those objects are all at the same level in the STEP hierarchy.

The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.



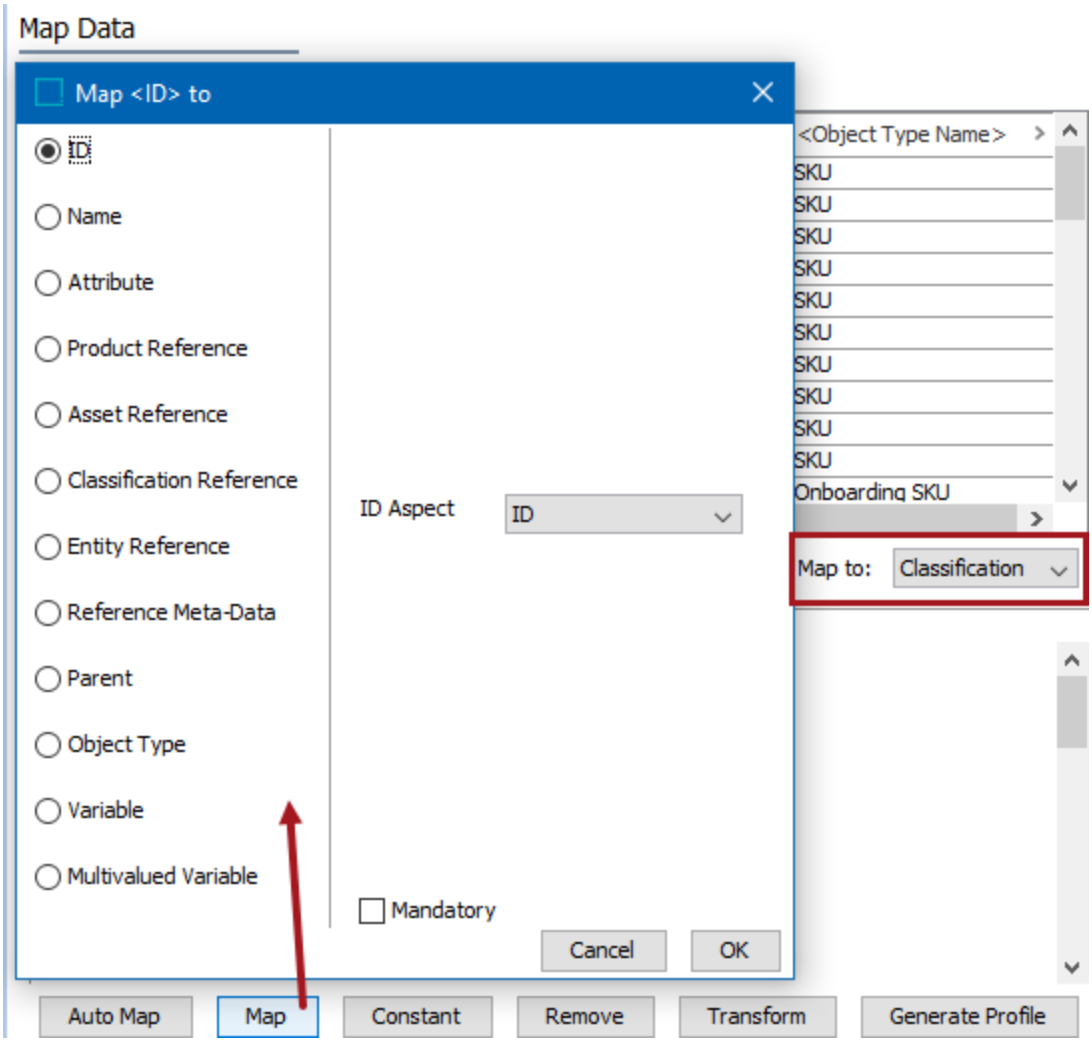
For more information, refer to the **Map Inbound** topics on the available options:

- ID or Key
- Name
- Attribute
- Attribute Info
- Attribute Validation Info
- Variable
- Multivalued Variable

Classification Object Type - Map To Inbound

For inbound data, after selecting the Classification super type in the **Map to** dropdown, click the **Map** button to import information for classifications.

The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.



For more information, refer to the **Map Inbound** topics on the available options:

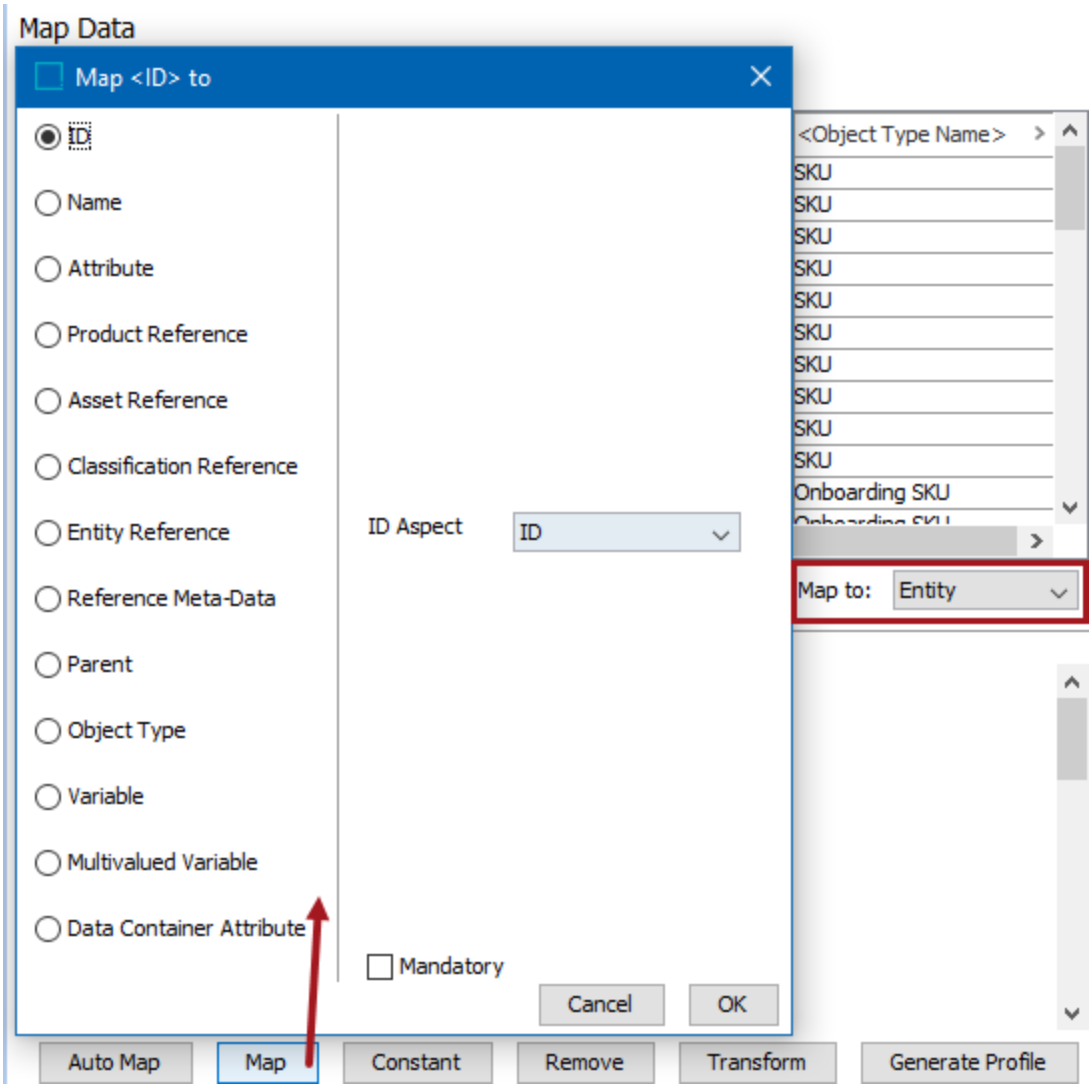
- ID or Key
- Name
- Attribute
- Product Reference
- Asset Reference

- Classification Reference
- Entity Reference
- Reference Meta-Data
- Parent
- Object Type
- Variable
- Multivalued Variable

Entity Object Type - Map To Inbound

For inbound data, after selecting the Entity super type in the **Map to** dropdown, click the **Map** button to import information for entities.

The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.



For more information, refer to the **Map Inbound** topics on the available options:

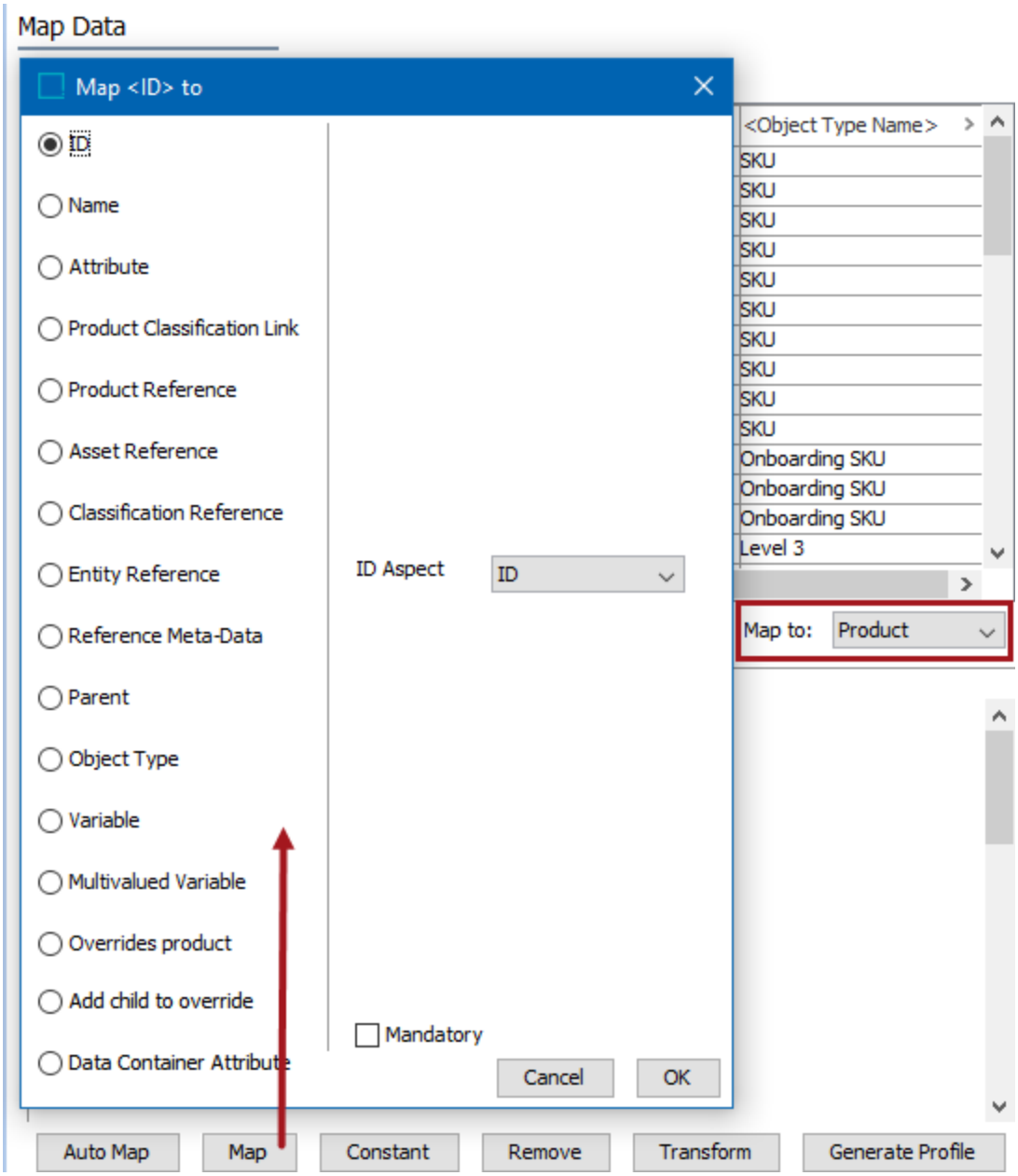
- ID or Key
- Name
- Attribute
- Product Reference

- Asset Reference
- Classification Reference
- Entity Reference
- Reference Meta-Data
- Parent
- Object Type
- Variable
- Multivalued Variable
- Data Container

Product Object Type - Map To Inbound

For inbound data, after selecting the Product super type in the **Map to** dropdown, click the **Map** button to import information for products.

The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.



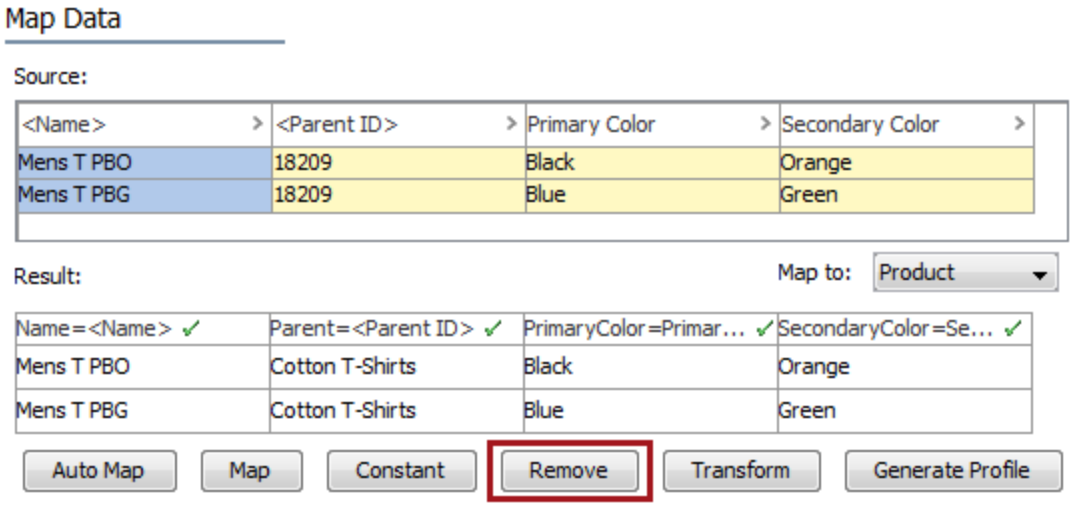
For more information, refer to the **Map Inbound** topics on the available options:

- ID or Key
- Name
- Attribute
- Product Classification Links
- Product Reference
- Asset Reference
- Classification Reference
- Entity Reference
- Reference Meta-Data
- Parent
- Object Type
- Variable
- Multivalued Variable
- Overrides Product
- Add child to override

When mapping products for commercial data, refer to the **Importing Commercial Data via IIEP** topic in the **Publisher (Adobe InDesign Integration)** documentation.

Inbound Map Data - Remove

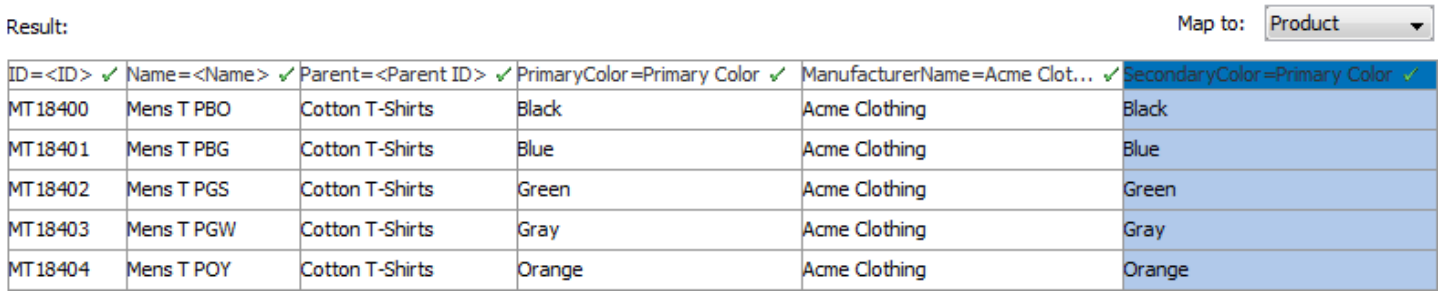
If the Result panel displays a column that you do not want to import, you can remove it.



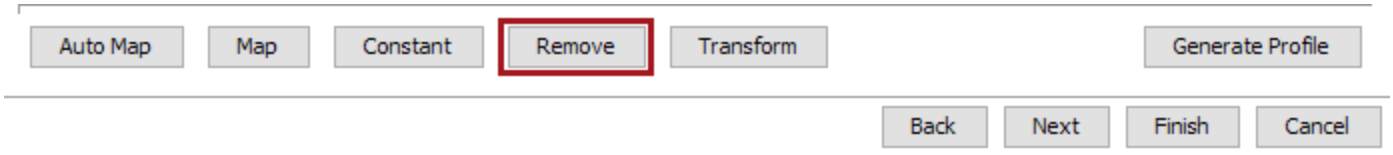
For more information about the additional import steps, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Remove a Mapped Column

1. In the **Result** panel, select the mapped column to remove.



2. Click the **Remove** button.



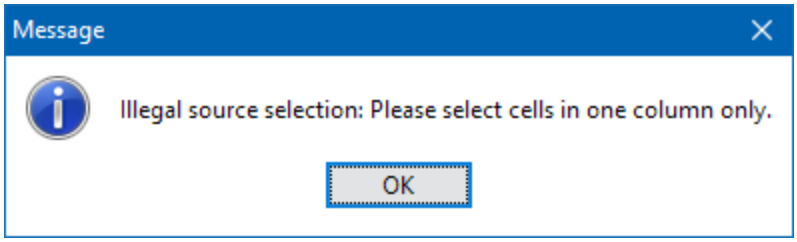
The **Result** panel no longer displays the column, and the removed data will not be imported.

Result:

Map to: Product

ID=<ID> ✓	Name=<Name> ✓	Parent=<Parent ID> ✓	PrimaryColor=Primary Color ✓	ManufacturerName=Acme ... ✓
MT18400	Mens T PBO	Cotton T-Shirts	Black	Acme Clothing
MT18401	Mens T PBG	Cotton T-Shirts	Blue	Acme Clothing
MT18402	Mens T PGS	Cotton T-Shirts	Green	Acme Clothing
MT18403	Mens T PGW	Cotton T-Shirts	Gray	Acme Clothing
MT18404	Mens T POY	Cotton T-Shirts	Orange	Acme Clothing

Attempting to remove more than one column at a time is not allowed and displays the following message.



Inbound Map Data - Transform

There are numerous ways to use the transformation options to modify the data before it is loaded into STEP. With transformations, you can split one data column into multiple attribute values, append and prepend data to the values, perform search and replace, and so on.

Transformations can also be used to clean data prior to importing. For example, for an attribute named 'Color', the values for Black are represented as 'Black', 'black', 'B', and 'BLK.' You can use a transformation to ensure a single value is added for all of the original values.

Keep the following in mind when using transformations:

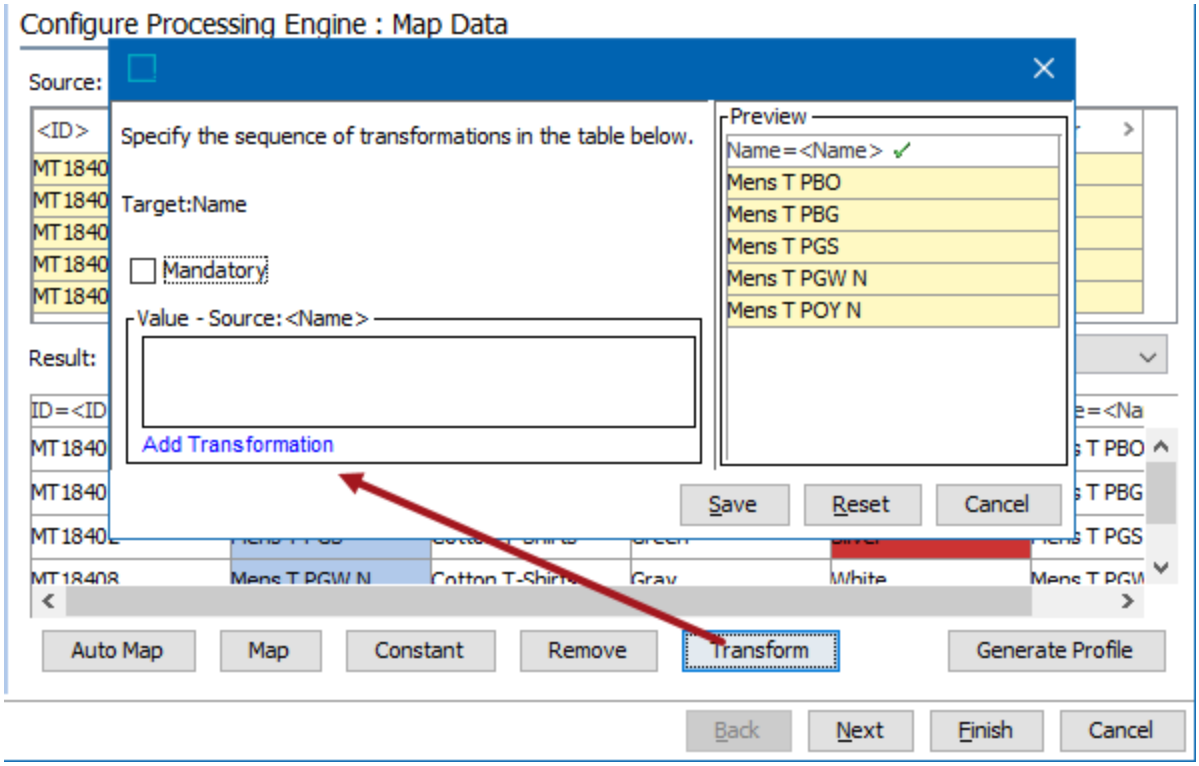
- You can apply multiple transformations to the columns of data, but be aware that they are processed sequentially, from top to bottom.
- Import transformations can manage multiple data container instances at once.
- When setting up transformations that are implemented with a saved configuration, it is recommended that you work with smaller load files first. You can then build up the complexity of your transformations in a step-by-step mode.
- Several of the transformations allow you to manipulate units mm, kg. etc., including manipulations such as removing the unit from the value. Other transformations allow you to extract only the unit from a value. In these cases, the attribute that you are mapping to must be one that is allowed to have units and further, must be one that already has those units assigned to it.

For examples of the most common transformations, refer to **Transformation Examples**.

To use transformations on attributes in STEP, or on data in tables and publications, refer to **Attribute Transformations** in the **System Setup** documentation and **Table Formatting Transformations** in the **Tables** documentation.

Transforming Data

The mapping step for Import Manager and IIEPs include the same options after clicking the **Transform** button.



For details on each of the available transformation options, refer to the **Transformations** topic in the **Resource Materials** online help.

For more information about the additional import steps, refer to **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Dimension Dependent Data

When mapping data from a file to an attribute that is dependent on a single dimension (for example, language or country), transformations make it possible to specify which Dimension Point is required. This allows values for multiple languages or countries to be imported from a single file. For more information, refer to **Dimension Point for Single Dimension Data Example** documentation.

Outbound Map Data Options

Mapping allows you to specify data sources that you want to extract for each record. When mapping is active, only data that is mapped is exported. For Excel or CSV, each STEP data source mapped is displayed as a column in the output.

Mapping is not required for the STEPXML or Advanced STEPXML formats.

Formats

Mapping differs slightly based on the type of data being output.

Tabular Formats

The following tabular formats require mapping for outbound data:

- CSV Format
- Excel Format
- Excel Smartsheet Format

XML Formats

The following XML formats require mapping for outbound data:

- BMEcat Format
- BMEcat 2005 Format
- BMEcat 2005.1 Format
- Generic XML Format
- IDoc MATMAS 05 Format

JSON Format

The following JSON format requires mapping for outbound data:

- Generic JSON Format

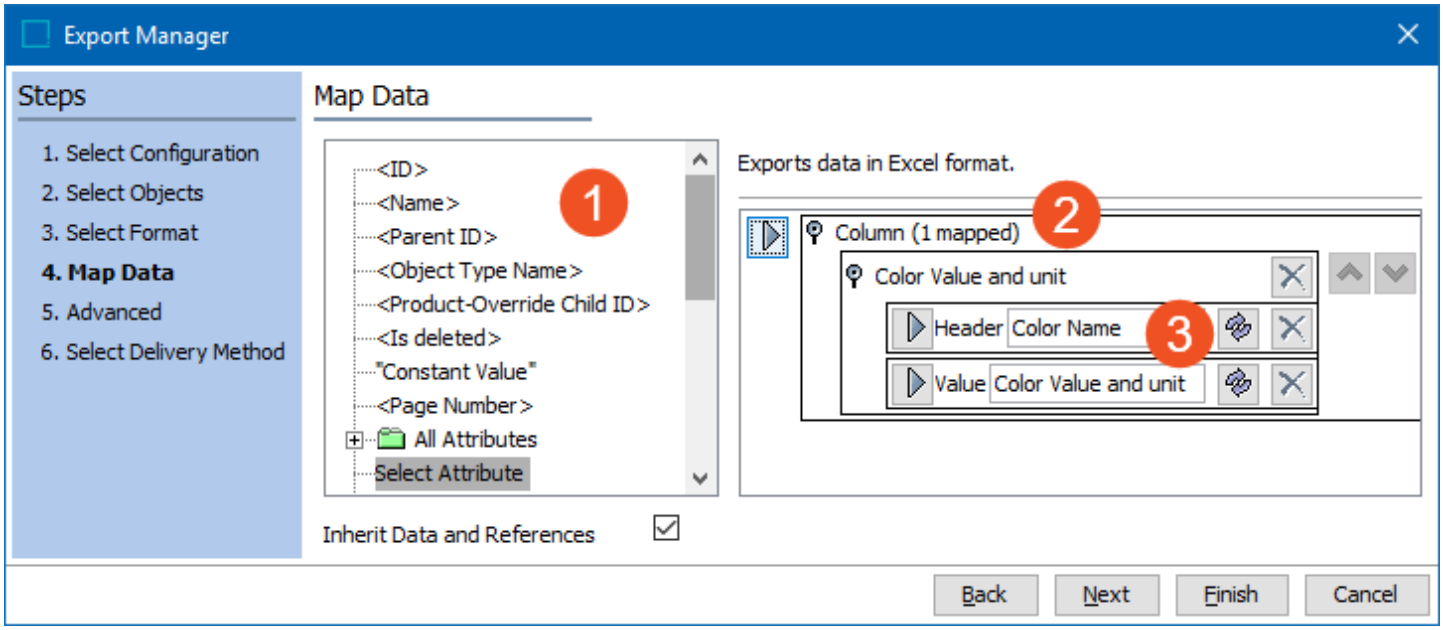
Mapping in Outbound Tools

The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic.

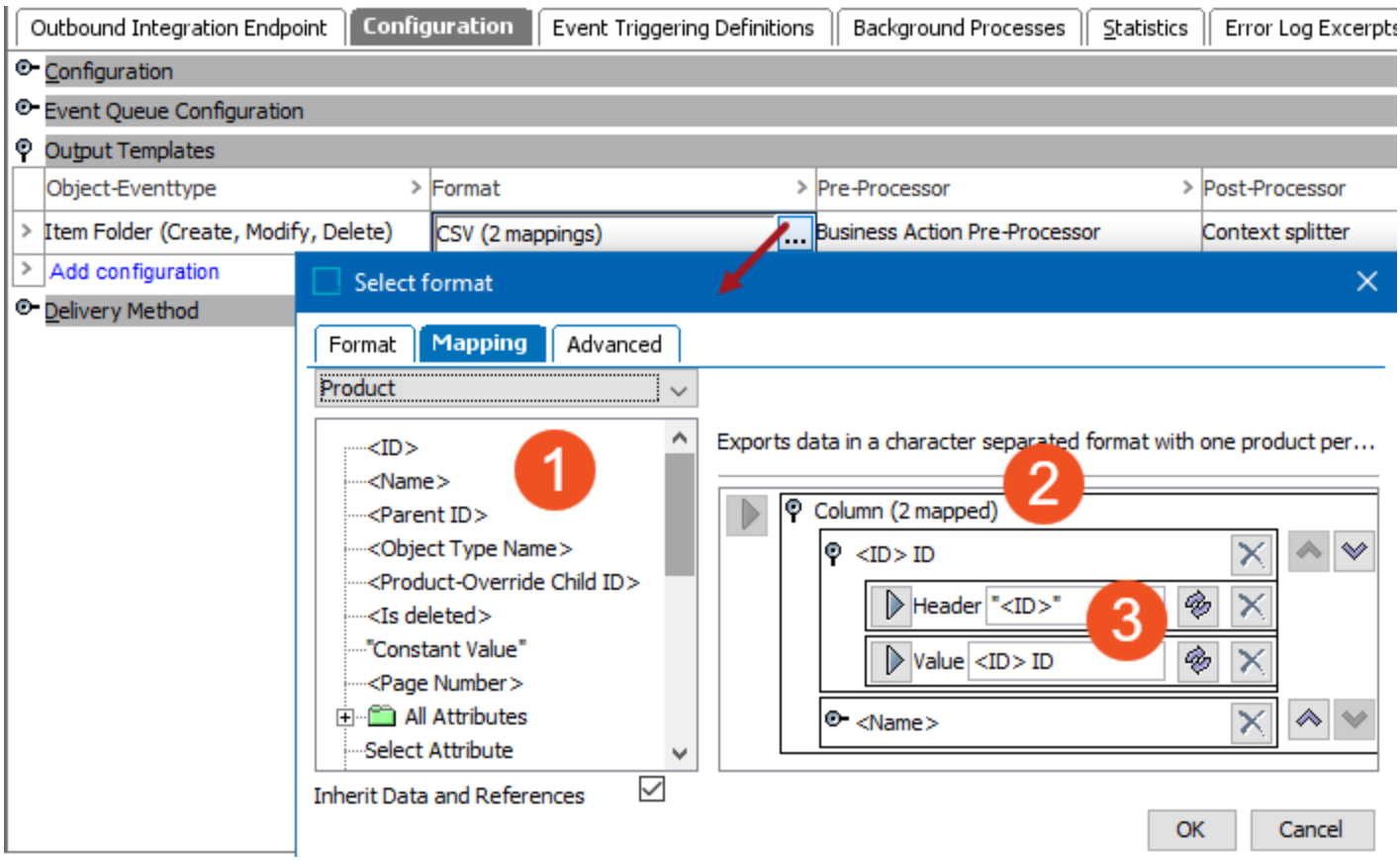
For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

For more information about the additional wizard steps, refer to **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

The Map Data step is displayed in the Export Manager wizard based on the selected format.



For outbound integration endpoints, based on the format selected in the Format parameter, mapping is available on a Mapping tab.



Mapping

The numbered elements in the images above include:

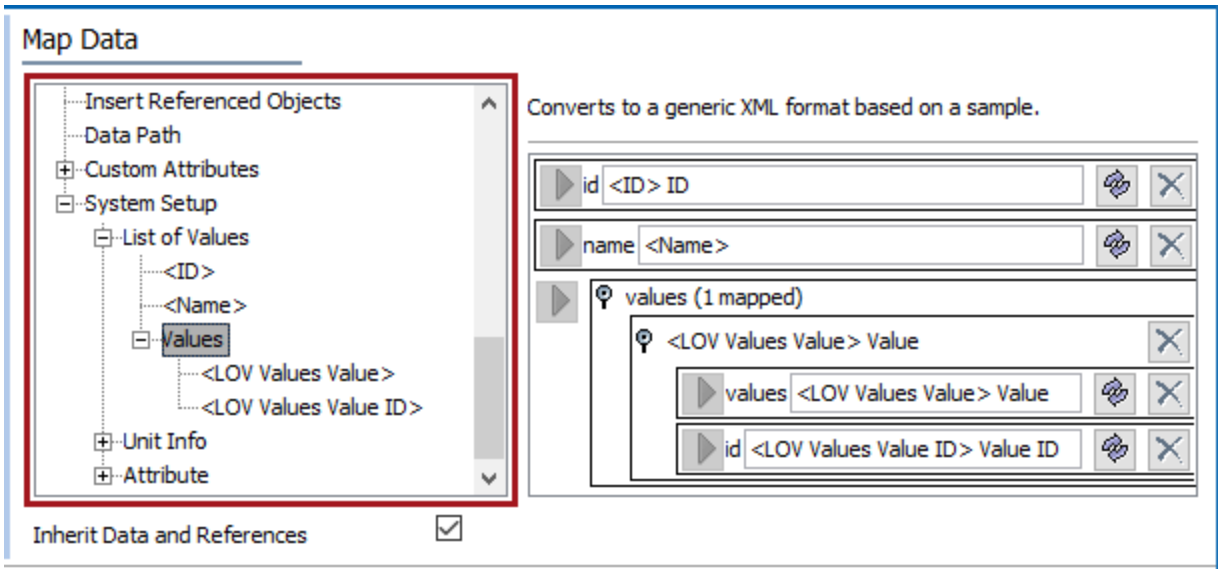
1. **Data Sources** indicate the original location of the information to be exported. The mapping page provides a way to associate the mapping targets with the STEP information available for the selected object types being exported. For more information on each source option, refer to the **Outbound Map Data - Data Source** documentation.
2. **Mapping Targets** are placeholders for the actual data being exported. The selected format determines which targets are available. Refer to the mapping section of your format documentation (listed above) and the **Outbound Map Data - Mapping Targets** documentation.
3. **Mapping Rules** represent the relationship, also called a binding, between a data source and a mapping target. Bindings can be configured in a number of ways, adding a constant value, applying a transformation (including choosing a different aspect), or mapping additional STEP data.

Transformations

The transform button allows you to modify the mapped data before export. For details, refer to the **Outbound Map Data - Transform** documentation.

Outbound Map Data - Data Source

Data Sources (outlined in the Map Data image below) indicate the original location of the information being exported. A data source provides a way to associate the mapping targets (defined by the template and displayed on the right side) with the STEP information available for the selected object types being exported.



The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

For more information about the additional wizard steps, refer to **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

The process to map data differs only slightly based on the data source selected. Generally, mapping data involves selecting one or more data sources from the left panel and then clicking the mapping target arrow in the right panel. The selected data appears in the list of columns to export.

Available Data Source Options

The following options are available for all formats that require outbound mapping, with the exceptions noted:

- All Attributes - Data Source Outbound
- Asset References - Data Source Outbound
- Attribute Information - Data Source Outbound
- Attributes Inherited from Classification - Data Source Outbound

- Attribute Links - Data Source Outbound or - for Generic XML and IDoc MATMAS 05 only
- Business Functions - Data Source Outbound
- Classification References - Data Source Outbound
- Classifications - Data Source Outbound
- Constant Value - Data Source Outbound
- Custom Attributes - Data Source Outbound
- Data Container Type ID - Data Source Outbound - within Insert References / Data Containers
- Data Owner Node - Data Source Outbound - within Insert References / Data Containers
- Data Path - Data Source Outbound - for Generic XML and IDoc MATMAS 05 only
- Data Type - Data Source Outbound - within Insert References / Data Containers
- Entity References - Data Source Outbound
- ID or Key - Data Source Outbound
- Index Words - Data Source Outbound - for Print Publisher only
- Inherit Data and References - Data Source Outbound
- Insert Data Containers - Data Source Outbound - within Insert References / Data Containers
- Insert References - Data Source Outbound - within Insert References / Data Containers
- Insert Referenced Objects - Data Source Outbound
- Is Deleted - Data Source Outbound - for Generic XML only
- Multi Level Parent Attributes - Data Source Outbound
- Multi Level References - Data Source Outbound
- Name - Data Source Outbound
- Object Type Name - Data Source Outbound
- Page Number - Data Source Outbound - for Print Publisher only
- Parent ID - Data Source Outbound
- Product Classification Links - Data Source Outbound
- Product-Override Child ID - Data Source Outbound
- Product References - Data Source Outbound
- Reference Type ID - Data Source Outbound - within Insert References / Data Containers

- Select Attribute - Data Source Outbound
- STEP Workflow Task Info - Data Source Outbound
- System Setup - Data Source Outbound

Asset References - Data Source Outbound

Asset references can be exported when they reference products, classifications, or entities. Asset references can be mapped via their ID or a unique key. Each selected asset reference is extracted into a separate field in the output.

The Aspect option is available for Asset References and allows a variety of data on the asset reference to be exported.

The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

For more information about the additional wizard steps, refer to the **Creating a Data Export** topic or the **Creating an Outbound Integration Endpoint** topic.

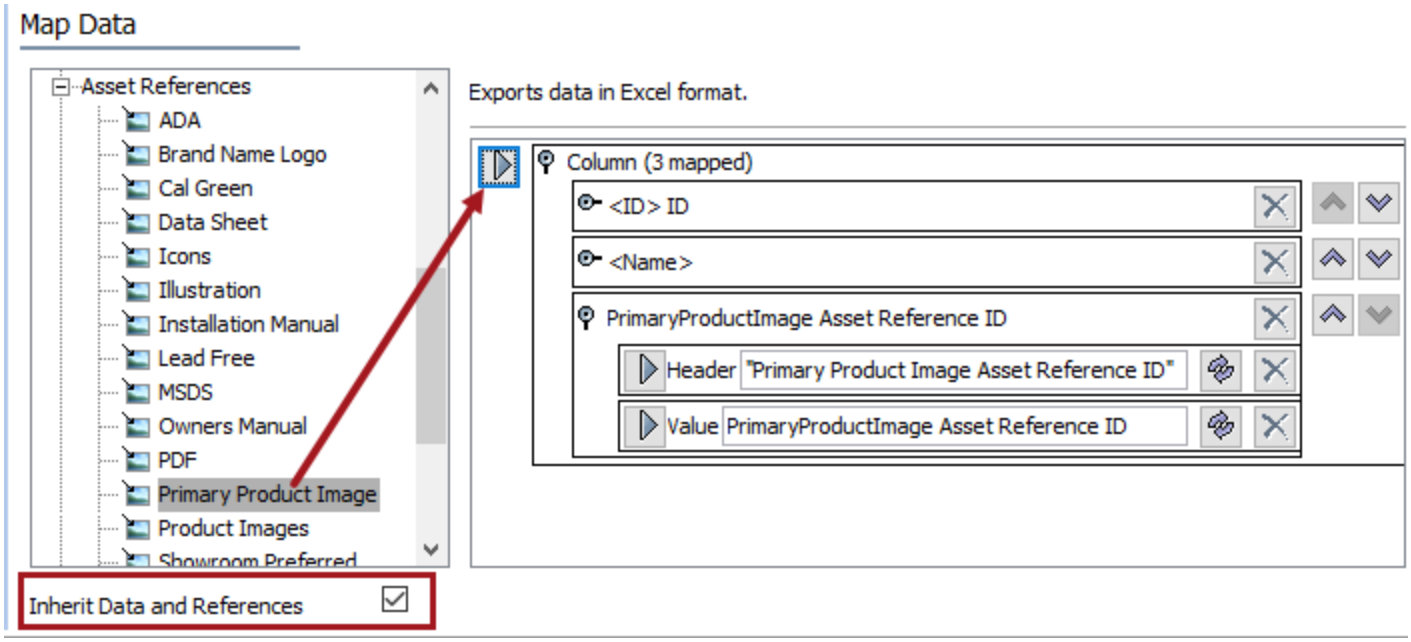
After mapping, most output options can be altered using transformations. For more information, refer to the **Outbound Map Data - Transform** topic.

Mapping Asset References

1. Select the data to be exported for the output tool. For more information, refer to the **Data Exchange** topic.
2. On the Map Data step, in the left panel, expand **Asset References**, and select the desired asset reference type.

   Note: For an OIEP, you must first select Product from the dropdown at the top of the Mapping tab.

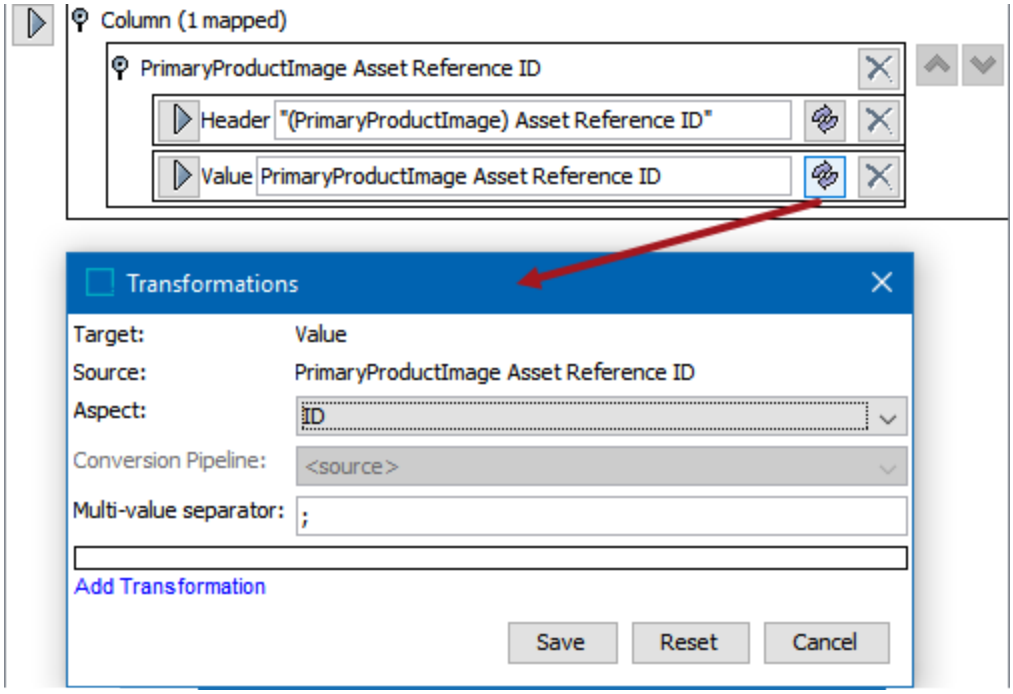
3. Click the right arrow button ( ) in the right panel to add the selected asset reference type.



4. Check the **Inherit Data and References** option appropriately:
   - If **checked**, inherited data and references are included in the export.
   - If **unchecked**, inherited data and references is not included in the export.

   For information on inheritance, refer to the **Inherit Data and References - Data Source Outbound** topic.

5. If needed, change the reference Value Aspect to output data other than the reference ID. For examples, refer to the **Results** section below. For details on the options, refer to the **Aspect - Transform Outbound** topic.



- Open the asset reference flipper to display the Header and Value elements.
- Click the transformation button ( ) for the Value element to display the Transformations dialog and the Aspects parameter.
- Click the **Aspect** dropdown to display and select an option.
- Click the **Save** button to apply any aspect changes.

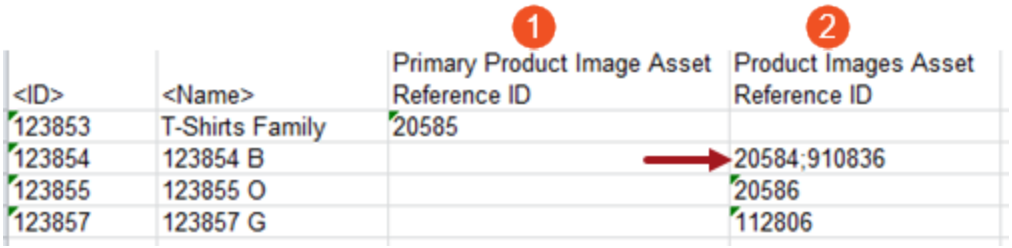
5. Apply any transformations necessary to change the output without changing the original data. Refer to the **Outbound Map Data - Transform** topic for details.
6. Complete any additional mappings and initiate the export.

Results

The exported file includes Product ID, Name, Asset Reference Type ID as header and referenced asset ID as value.

<ID>	<Name>	Primary Product Image Asset Reference ID ()
123853	T-Shirts Family	20585

When exporting multiple products that are linked to assets with different asset reference types, each asset type is displayed in its own column. When multiple references for the same reference type exist, they are separated by a semicolon.



When the Name aspect is used, the output includes product ID, name, and mapped Asset Reference Type Name as header and referenced asset's name as value for all exported rows.

<ID>	<Name>	Product Images Asset Reference Name ()
123854	123854 B	Hanes Blue;Blue Shirt

When the MIME Type aspect is used, the output includes asset reference type MIME Type as the header and referenced asset's MIME Type as its value in the exported file. For more information, refer to the **MIME Types** section of the **System Setup** documentation.

<ID>	<Name>	Primary Product Image Asset Reference MIME Type ()
123854	123854 B	image/tiff
123855	123855 O	image/tiff
123857	123857 G	image/tiff

When the 'File named by asset ID' aspect is selected, multiple files are output in a .ZIP file:

- The digital media files associated with the referenced assets. If the referenced asset is an image, the image can be converted using the conversion pipeline.
- The selected export format file. This file includes product ID, name and ‘Asset reference file named by Asset ID’ as header and asset ID with corresponding asset extension as value for all the exported rows.

<ID>	<Name>	Product Images Asset Reference File named by asset ID ()
123854	123854 B	20584.tif
123855	123855 O	20586.tif
123857	123857 G	112806.jpg

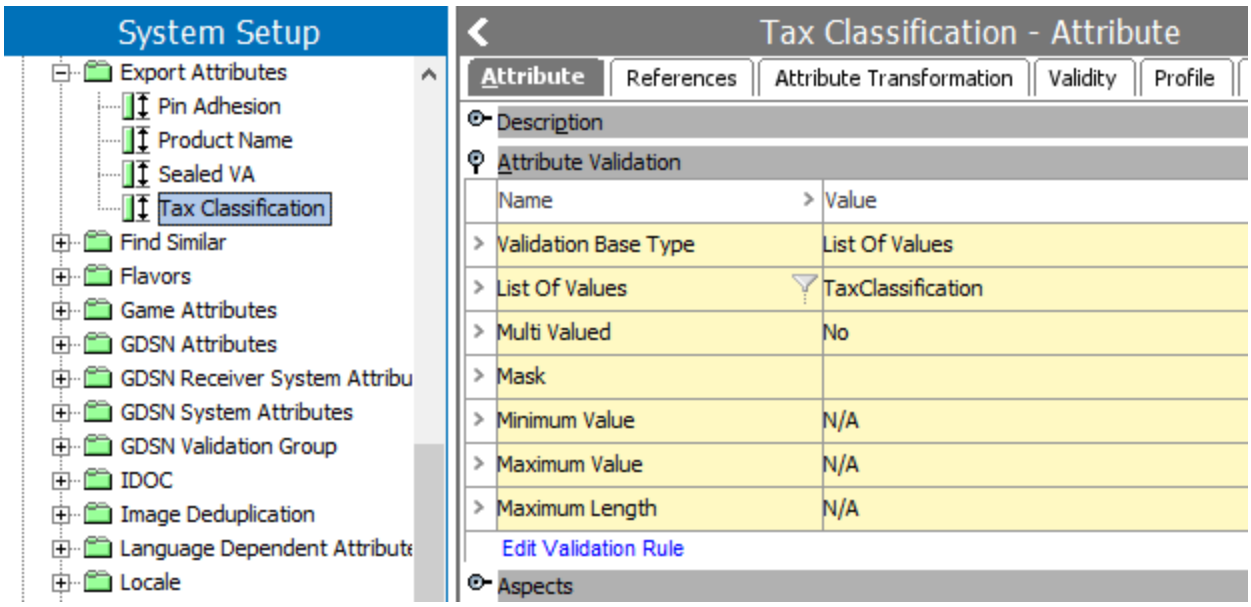
When the 'File named by asset Name' aspect is selected, multiple files are output in a .ZIP file:

- The digital media files associated with the referenced assets. If the referenced asset is an image, the image can be converted using the conversion pipeline.
- The selected export format file. This file includes product ID, name and ‘Asset reference file named by Asset Name’ as header and asset name with corresponding asset extension as value for all the exported rows.

<ID>	<Name>	Product Images Asset Reference File named by asset Name ()	Primary Product Image Asset Reference File named by asset Name ()
123853	T-Shirts Family		Hanes Family.tif
123854	123854 B	Hanes Blue.tif	
123855	123855 O	Hanes Orange.tif	
123857	123857 G	HanesGreenT.jpg	

Attribute Information - Data Source Outbound

The option to map attribute details is available when the object super type selected for the export is Attribute. The option is also included within the System Setup data source group, which is available for use with the Generic XML format, as defined in the **System Setup - Data Source Outbound** topic. This option can be used to output attribute details like ID, name, and parameter settings from System Setup.



Additionally, the export file can be modified and then imported to create new attributes in STEP. For information about using this option to create new attributes, refer to the **Managing Attribute Parameters with Excel** topic.

The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

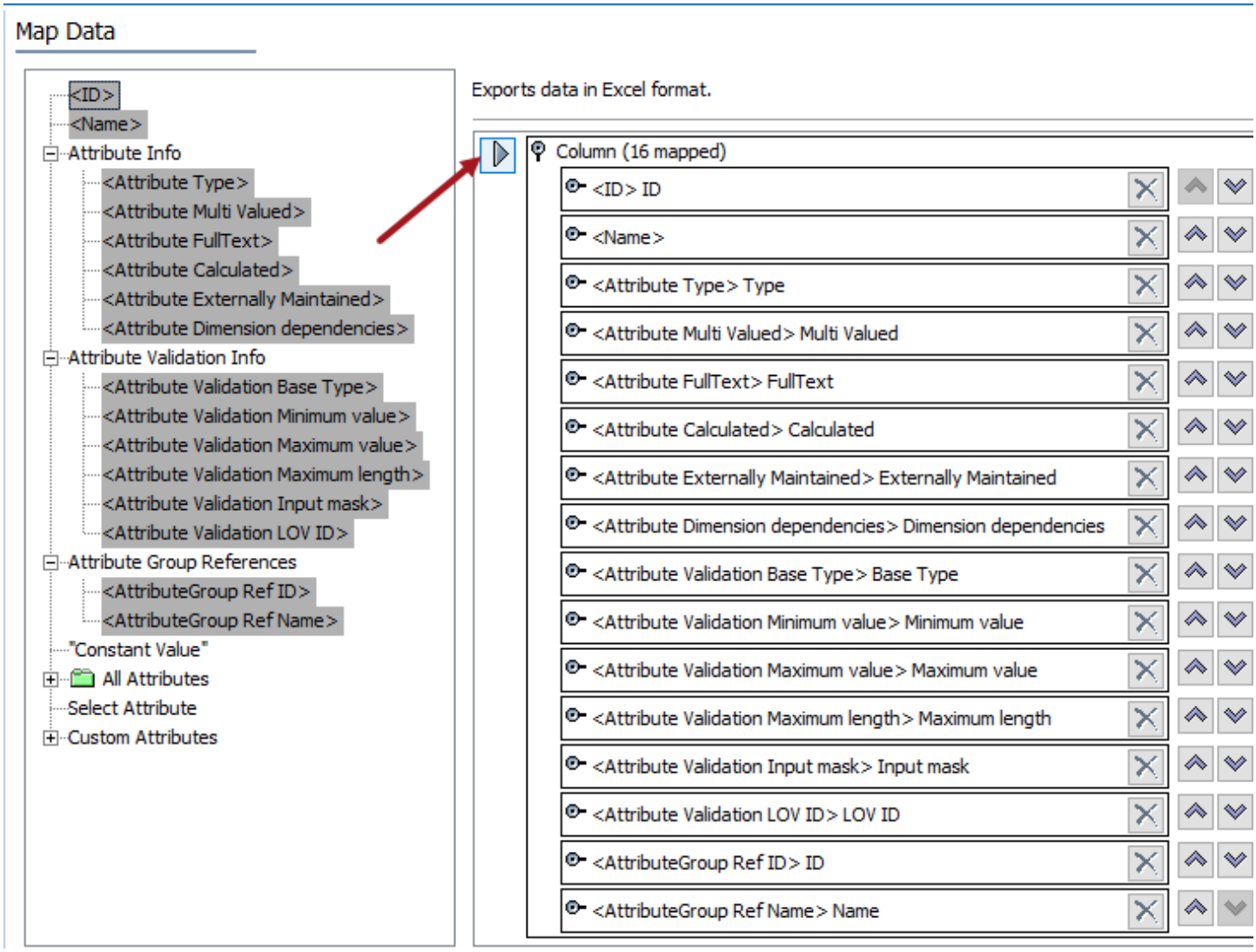
For more information about the additional wizard steps, refer to the **Creating a Data Export** topic or the **Creating an Outbound Integration Endpoint** topic.

After mapping, most output options can be altered using transformations. For more information, refer to the **Outbound Map Data - Transform** topic.

Mapping Attribute Details

1. Select the data to be exported for the output tool. For more information, refer to the **Data Exchange** topic.
2. Select the attribute super object type based on your output tool. For more information, refer to the **Export Manager - Select Objects** topic, the **Configure the Format** section of the **OIEP - Event-Based - Output Templates Flipper** topic, or the **Configure the Format** section of the **OIEP - Select Objects - Output Templates Flipper** topic.

3. On the Map Data step, in the left panel, select one or more attribute details node(s) to export and click the right arrow button ( ). In the image below, all possible attribute detail options were selected and mapped.



4. Apply any transformations, which can change the output without changing the original data. Refer to the **Outbound Map Data - Transform** topic.
5. Complete any additional mappings and initiate the export.

Results

The output includes headers that can be auto-mapped upon import, and the values for each attribute exported. For display purposes, the image below is shown in two sections. This export actually created a single Excel file.

	A	B	C	D	E	F	G	H	I
1	**<ID>**	**<Name>**	**<Attribute Type>**	**<Attribute Multi Valued>**	**<Attribute FullText>**	**<Attribute Calculated>**	**<Attribute Externally Maintained>**	**<Attribute Dimension dependencies>**	**<Attribute Validation Base Type>**
2	ProductName	Product Name	Normal	false	false	false	false	Language	text
3	SealedVA	Sealed VA	Property	false	false	false	false		numeric_text
4	PinAdhesion	Pin Adhesion	Property	false	false	false	false		number
5	TaxClassification	Tax Classification	Normal	false	false	false	false		LOV

	J	K	L	M	N	O	P
1	**<Attribute Validation Minimum value>**	**<Attribute Validation Maximum value>**	**<Attribute Validation Maximum length>**	**<Attribute Validation Input mask>**	**<Attribute Validation LOV ID>**	**<AttributeGroup Ref ID>**	**<AttributeGroup Ref Name>**
2			10000			ManualSequencing;WarehouseView;ImageReviewView;EcomView;ManualSequencingDisplay;ItemDescriptionInformation;BuyerView;ExportAttrs;BuildAll;DataVisualization	Manual Sequencing;Warehouse, View;Image Review View;Ecom, View;Manual Sequencing Display;Item Description Information;Buyer, View;Export Attributes;Build
3			100			ExportAttrs	ExportAttrs
4						ExportAttrs	ExportAttrs
5					TaxClassification	ForecastingView;ItemImportExportInformation;SupplierModify;ExportAttrs	Forecasting, View;Item Import/Export Information;Supplier, Modify;ExportAttrs

Attributes (and Data Containers) - Data Source Outbound

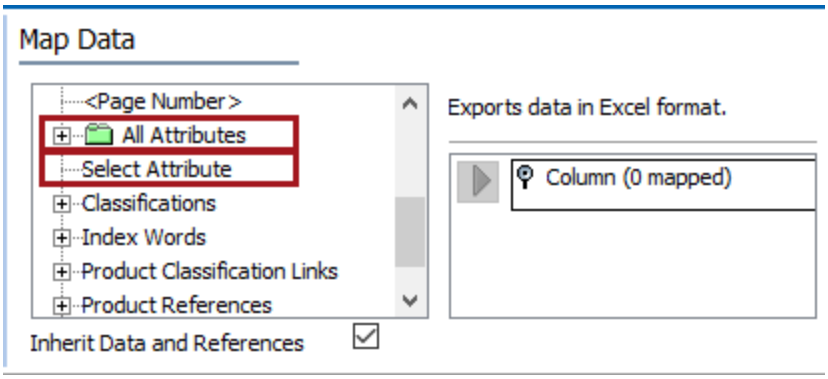
Attributes and data containers can be selected and exported for the objects being exported.

The following options are available to export attributes:

- **All Attributes** shows attributes that are relevant for the objects that are selected for export. This includes specification attributes linked to the selected object(s) and description attributes that are valid for the selected object(s). You can select and map attribute groups in the hierarchy, allowing you to map all attributes in the group. The top level node All Attributes can also be mapped, allowing you to map all attributes relevant for the selected object(s).

  The All Attributes option is also used to map **data containers**. For details, refer to the **Mapping Entity Data Containers via All Attributes** section below.

- **Select Attribute** allows you to browse or search for an attribute or attribute group, and also Force Calculation when the attribute is calculated.



The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

For more information about the additional wizard steps, refer to the **Creating a Data Export** topic or the **Creating an Outbound Integration Endpoint** topic.

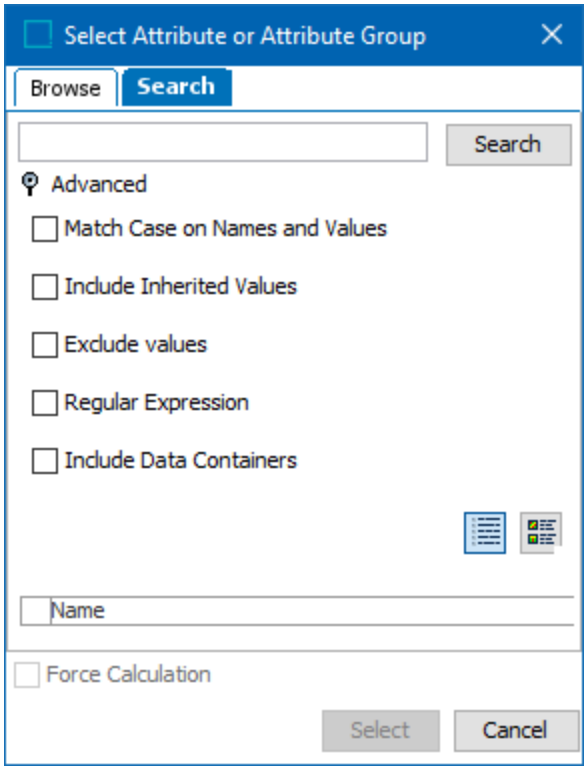
After mapping, most output options can be altered using transformations. For more information, refer to the **Outbound Map Data - Transform** topic.

Mapping Attributes via Select Attribute

Browse or Search can be used to find any attribute.

1. Select the data to be exported for the output tool. For more information, refer to the **Data Exchange** topic.

2. On the Map Data step, in the left panel, click **Select Attribute**.
3. In the right panel, click the right arrow ( ) to display the **Select Attribute or Attribute Group** dialog.



4. Find the attribute being mapped using one of the following methods:
   - The Browse tab allows finding an item via the hierarchy.
   - The Search tab allows entering part of the name of the attribute to display matches. Click the **Search** button to display all matches and choose an attribute. Open the Advanced flipper to use the advanced search options. For more information, refer to the **Search: Advanced Options** topic in the **Getting Started** documentation.

   When selecting a calculated attribute, the Force Calculation checkbox is enabled. Checking this box ensures the value of the calculation will be resolved upon export. When unchecked, calculated values can still be generated via the **Include Calculated Attribute Values** option on the Advanced step.

   Note: Calculated attributes can be mapped and resolved even when they are not valid for the objects being exported. However, the elements of the calculation (for example, the value of an attribute) must be valid and have a value in order to display as expected in the final output.

5. Select the relevant attribute, and then click **Select**. If Force Calculation was checked, the attribute transformation dialog is displayed with the Force Calculation option checked.
6. Repeat these steps to add all other required attributes.
7. Apply any transformations, which can change the output without changing the original data. Refer to the

Aspect - Transform Outbound topic.

8. Complete any additional mappings and initiate the export.

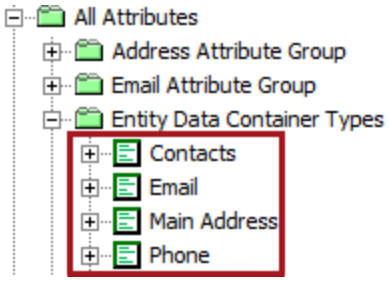
Mapping Attributes or Attribute Groups via All Attributes

Only attributes that are relevant for the objects that are selected for export are available.

1. Select the data to be exported for the output tool. For more information, refer to the **Data Exchange** topic.
2. On the Map Data step, in the left panel, expand the **All Attributes** node and select one or more attributes or attribute groups.
3. In the right panel, click the right arrow button ( ) to add the selections.
4. Apply any transformations, which can change the output without changing the original data. Refer to the **Aspect - Transform Outbound** topic.
5. Complete any additional mappings and initiate the export.

Mapping Entity Data Containers via All Attributes

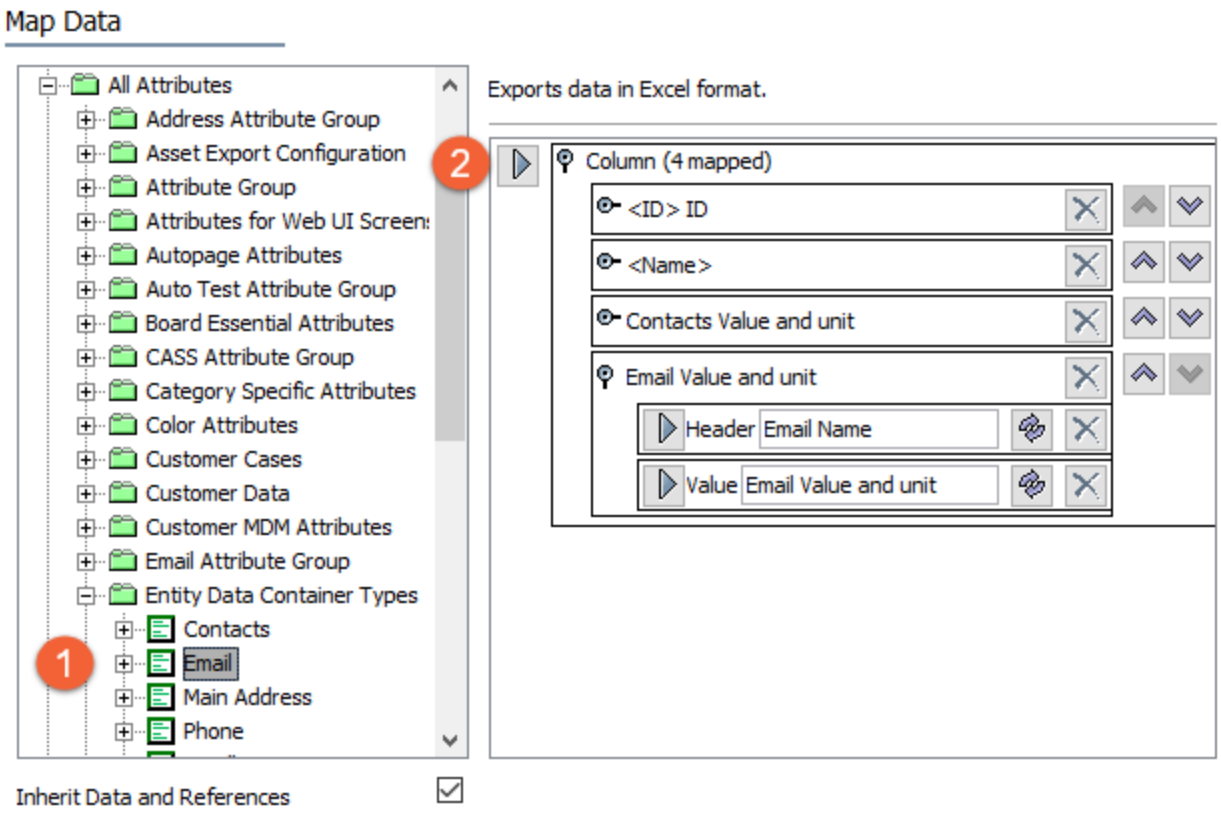
For exporting entity data containers into CSV or Excel formats (which require mapping), only **All Attributes** can be used. Using this option, data containers can be mapped directly to the export or included in a higher level selection.



For information on exporting entity and product data containers without mapping, refer to the **Data Containers in STEPXML** section of the **Data Exchange** documentation.

The following steps define mapping for data containers.

1. Select the data to be exported for the output tool. For more information, refer to the **Data Exchange** topic.
2. On the Map Data step, in the left panel, click **All Attributes** and select the desired entity data container type (s).



3. Click the right arrow button ( ) in the right panel to add the selected data container type as a group of columns.
4. Add additional entity data container types, if needed.
5. Apply any transformations, which can change the output without changing the original data. Refer to the **Outbound Map Data - Transform** topic.
6. Complete any additional mappings and initiate the export.

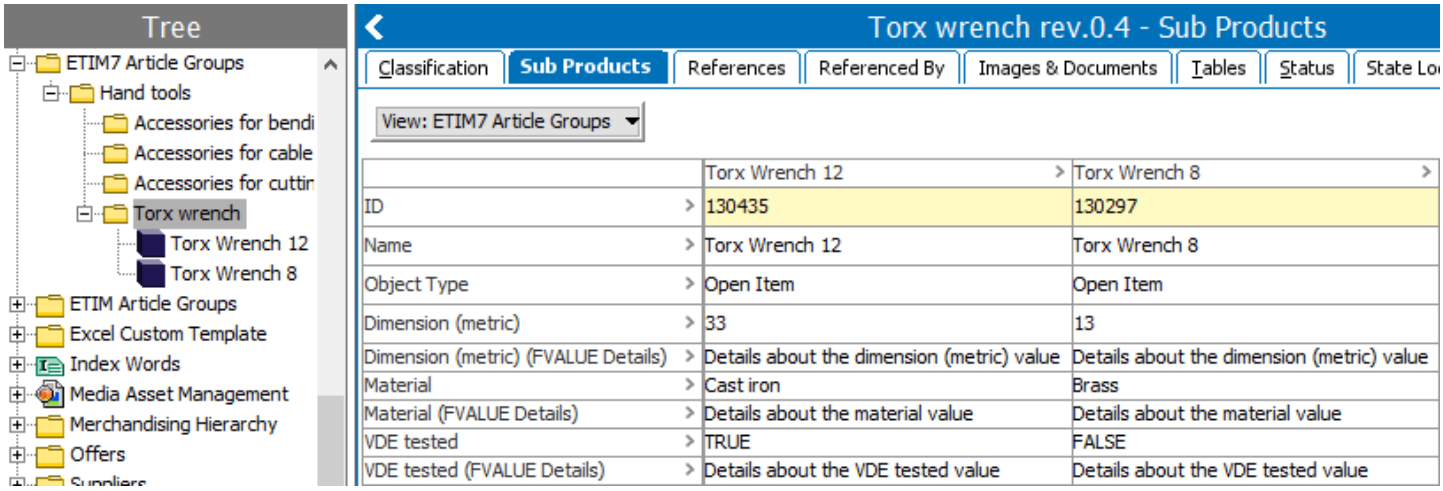
For information on handling entity data container transformations during export, refer to the **Data Container Separator - Transform Outbound** section of the **Data Exchange** documentation.

Attributes Inherited from Classification - Data Source Outbound

The 'Attributes Inherited from Classification' data source is only available when the object type selected for the export is Product and the format is BMEcat 2005 or FAB-DIS. This option is intended for use with ETIM classification data when mapping the BMEcat 2005 'FEATURE' mapping target.

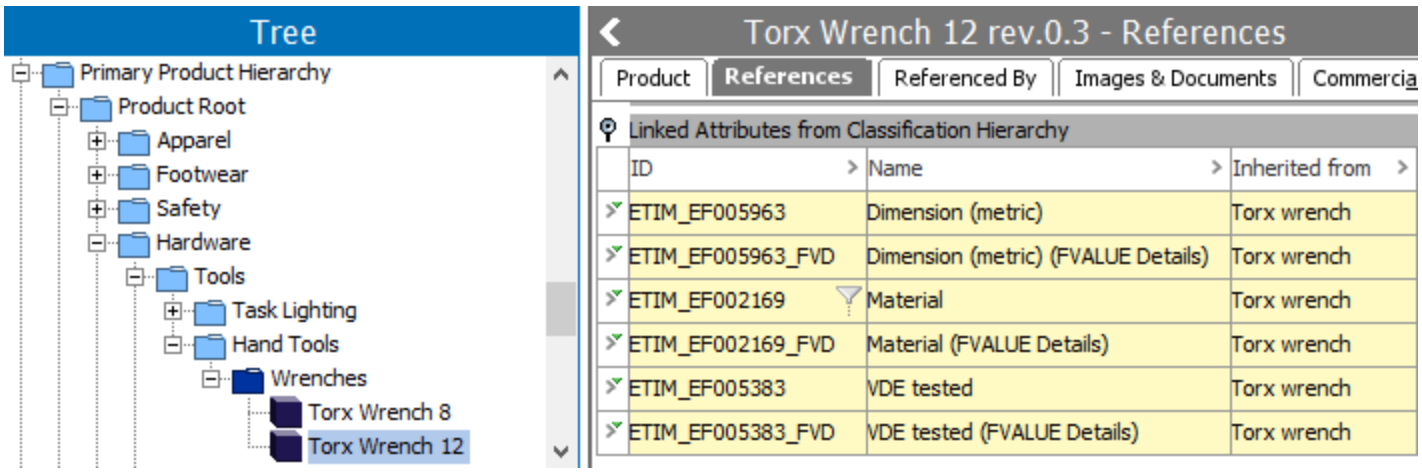
The option to map attributes inherited from a classification extracts attributes that are inherited from a selected classification (or child classification of the selected classification) to the product. So instead of mapping to an attribute group, you map to an ETIM classification, and the system analyzes which attributes are valid for the classification and inherited to the product that was selected for the export.

For example, the following products have ETIM feature specification attribute links that are inherited from the classification. This inheritance is achieved via the Product to Classification Link Type using the Inheritance of Links and Inheritance of Specification Attributes parameters as defined in the **Product to Classification Link Types** topic. The local values for the linked attributes can be viewed on the SubProducts tab of the 'Torx wrench' classification shown below.



	Torx Wrench 12	Torx Wrench 8
ID	130435	130297
Name	Torx Wrench 12	Torx Wrench 8
Object Type	Open Item	Open Item
Dimension (metric)	33	13
Dimension (metric) (FVALUE Details)	Details about the dimension (metric) value	Details about the dimension (metric) value
Material	Cast iron	Brass
Material (FVALUE Details)	Details about the material value	Details about the material value
VDE tested	TRUE	FALSE
VDE tested (FVALUE Details)	Details about the VDE tested value	Details about the VDE tested value

The References tab of the product confirms that the values of the product are coming from the classification. Although not shown below, both products have inherited the same links from the ETIM classification.



The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

For more information about the additional wizard steps, refer to the **Creating a Data Export** topic or the **Creating an Outbound Integration Endpoint** topic.

After mapping, most output options can be altered using transformations. For more information, refer to the **Outbound Map Data - Transform** topic.

Limitations

For ETIM product data, the data source can only be used to extract the following sub-fields in the FEATURE target via aspect parameters:

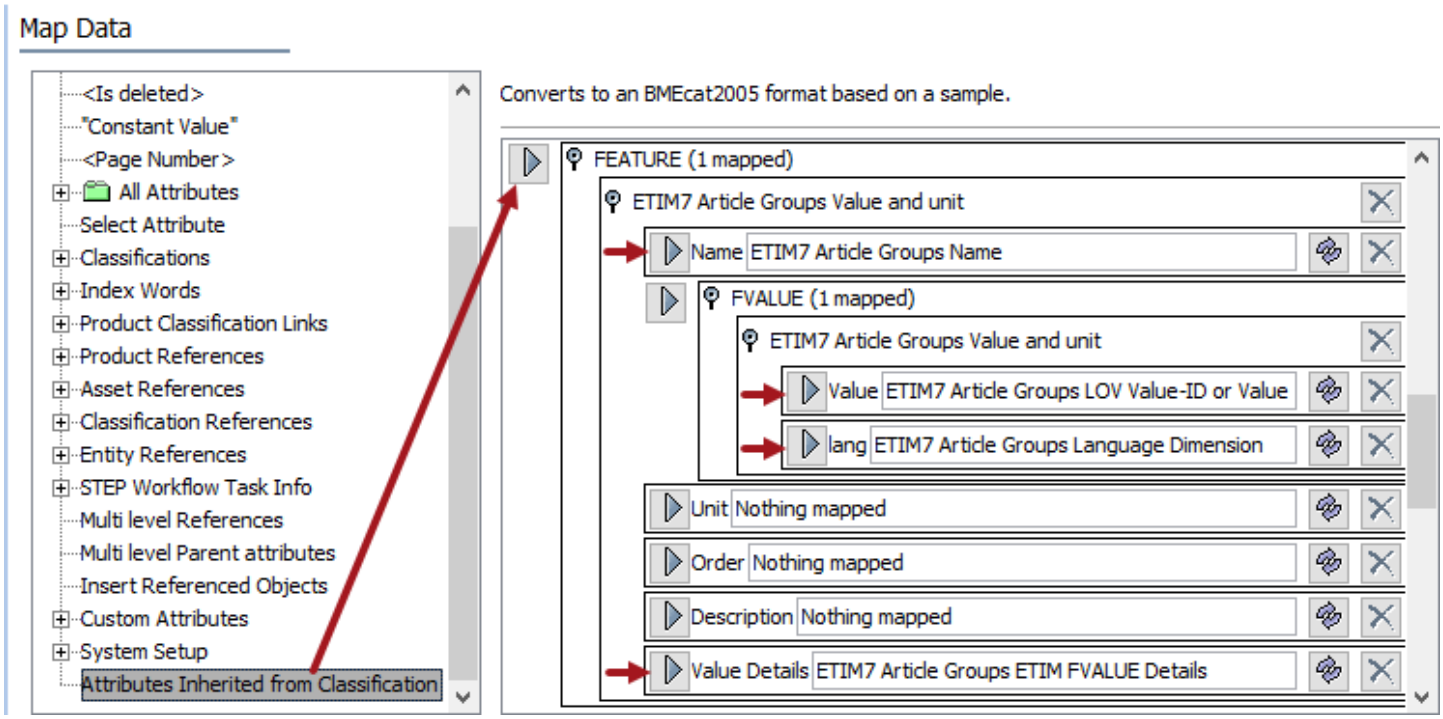
- Name: using the 'Name' aspect
- FVALUE Value: using the 'LOV Value-ID or Value' aspect
- FVALUE lang: using the 'Language Dimension' aspect
- Value Details: using the 'ETIM FVALUE Details' aspect

Mapping Attributes Inherited from Classification

1. Select the data to be exported for the output tool. For more information, refer to the **Data Exchange** topic.
2. If values from multiple contexts are desired, on the Select Format step:
   - Set the 'Export data for selected contexts' parameter to 'Yes.'
   - Click the **Select Contexts** link to display the Select Contexts dialog.
   - Select the relevant contexts and click the **Select** button.

   For more information, refer to the **BMEcat 2005 Format** topic.

3. On the Map Data step, in the left panel, select the 'Attributes Inherited from Classification' data source.



   - Click the right arrow button ( ) for the **FEATURE** mapping target.
   - On the classifications dialog, use the Browse or Search tab to select the desired ETIM classification folder and click the **Select** button.
   - Open the ETIM classification 'Value and unit' flipper.
4. For the **Name** target, click the transformation button ( ), in the Aspect dropdown choose 'Name' option, and click the **Save** button.
5. Open the FVALUE target flipper, and then open the flipper for the ETIM 'Value and unit' group to display the 'Value' and 'lang' targets.
6. For the **Value** target, click the transformation button ( ), in the Aspect dropdown choose 'LOV Value-ID or Value' option, and click the **Save** button.
7. For the **lang** target, click the transformation button ( ), in the Aspect dropdown choose 'Language Dimension' option, and click the **Save** button.
8. For the **Value Details** target, click the transformation button ( ), in the Aspect dropdown choose 'ETIM FVALUE Details' option, and click the **Save** button.
9. Complete any additional mappings, apply any transformations, and initiate the export. Transformations can change the output without changing the original data. Refer to the **Outbound Map Data - Transform** topic.

Results

The attribute links inherited from the classification are included in the export for each product. The following images include only the product tag portion of the export file based on the export configuration.

```
<PRODUCT>
  <SUPPLIER_PID>130297</SUPPLIER_PID>
  <PRODUCT_DETAILS>
    <DESCRIPTION_SHORT>Torx Wrench 8</DESCRIPTION_SHORT>
    <BUYER_PID type="buyer_specific"/>
    <SPECIAL_TREATMENT_CLASS type="NOT_RELEVANT"/>
  </PRODUCT_DETAILS>
  <PRODUCT_FEATURES>
    <FEATURE>
      <FNAME>Dimension (metric)</FNAME>
      <FVALUE>13</FVALUE>
    </FEATURE>
    <FEATURE>
      <FNAME>Material</FNAME>
      <FVALUE lang="en-US">EV000149</FVALUE>
      <FVALUE_DETAILS>Majority material content (51% or more)</FVALUE_DETAILS>
    </FEATURE>
    <FEATURE>
      <FNAME>VDE tested</FNAME>
      <FVALUE>FALSE</FVALUE>
    </FEATURE>
  </PRODUCT_FEATURES>
  <PRODUCT_REFERENCE type="others"/>
</PRODUCT>
<PRODUCT>
  <SUPPLIER_PID>130435</SUPPLIER_PID>
  <PRODUCT_DETAILS>
    <DESCRIPTION_SHORT>Torx Wrench 12</DESCRIPTION_SHORT>
    <BUYER_PID type="buyer_specific"/>
    <SPECIAL_TREATMENT_CLASS type="NOT_RELEVANT"/>
  </PRODUCT_DETAILS>
  <PRODUCT_FEATURES>
    <FEATURE>
      <FNAME>Dimension (metric)</FNAME>
      <FVALUE>33</FVALUE>
    </FEATURE>
    <FEATURE>
      <FNAME>Material</FNAME>
      <FVALUE lang="en-US">EV000117</FVALUE>
      <FVALUE_DETAILS>Majority material content (51% or more)</FVALUE_DETAILS>
    </FEATURE>
    <FEATURE>
      <FNAME>VDE tested</FNAME>
      <FVALUE>TRUE</FVALUE>
    </FEATURE>
  </PRODUCT_FEATURES>
  <PRODUCT_REFERENCE type="others"/>
</PRODUCT>
```

Business Functions - Data Source Outbound

The Business Functions data source is available for Excel and CSV export configurations, both in the Export Manager and in outbound integration endpoints (OIEPs). This option uses business functions to export attribute values from child, sibling, and other groupings of indirectly referenced product objects in STEP. Excel and CSV are the only supported formats, as the intention is for these exports to be 'human readable.' Additionally, only **product** objects are supported.

Business functions work in a similar fashion to multi-level references in Excel and CSV exports, except the intended use of business functions is to pull values from product groupings that are 'implicitly' connected and not directly linked through references.

> **Note:** The Business Functions data source is only supported in In-Memory enabled STEP systems. For more information about the In-Memory Database Component, refer to the **In-Memory Database Component for STEP** section of the **Resource Materials** documentation.

Mapping Business Functions

The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

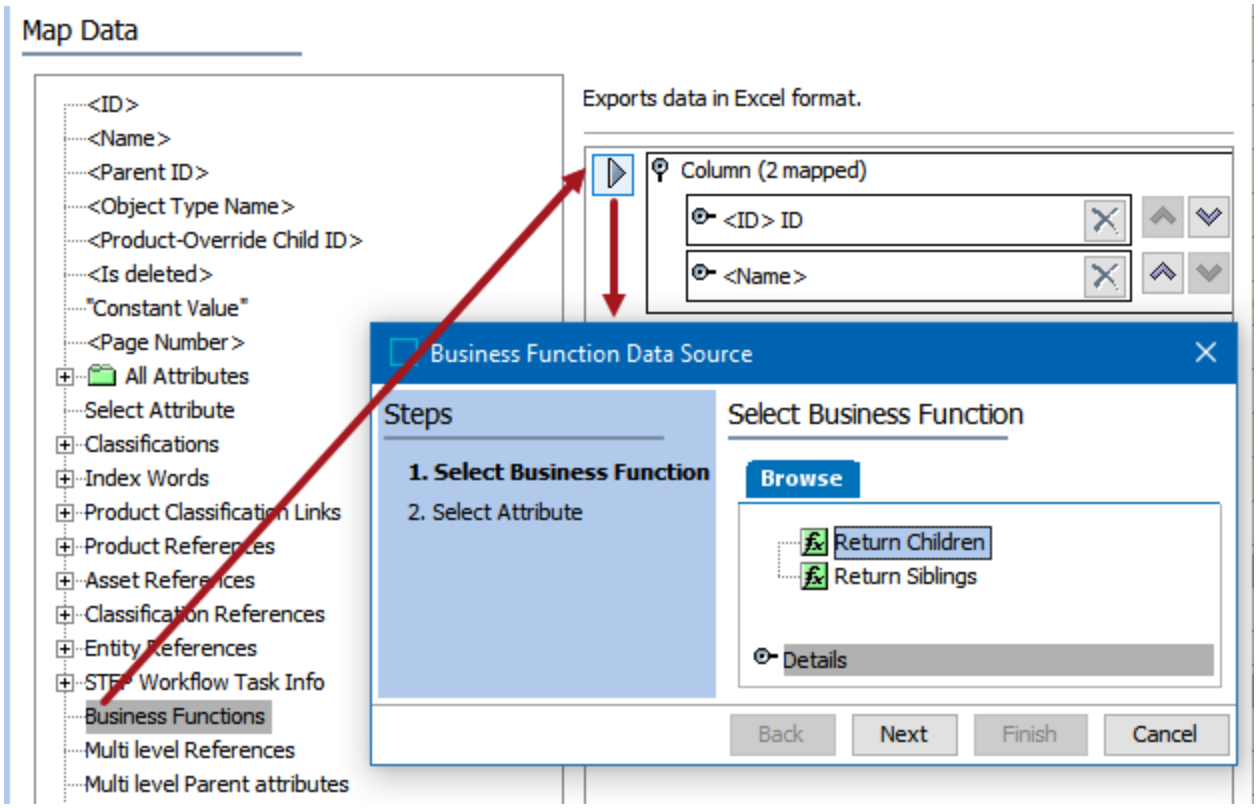
For more information about the additional wizard steps, refer to the **Creating a Data Export** topic or the **Creating an Outbound Integration Endpoint** topic.

After mapping, most output options can be altered using transformations. For more information, refer to the **Outbound Map Data - Transform** topic.

1. Select the data to be exported for the output tool. For more information, refer to the **Data Exchange** topic.
2. On the Map Data step, in the left panel, select the **Business Functions** data source.

   > **Note:** For an OIEP, you must first select 'Product' from the dropdown at the top of the Mapping tab.

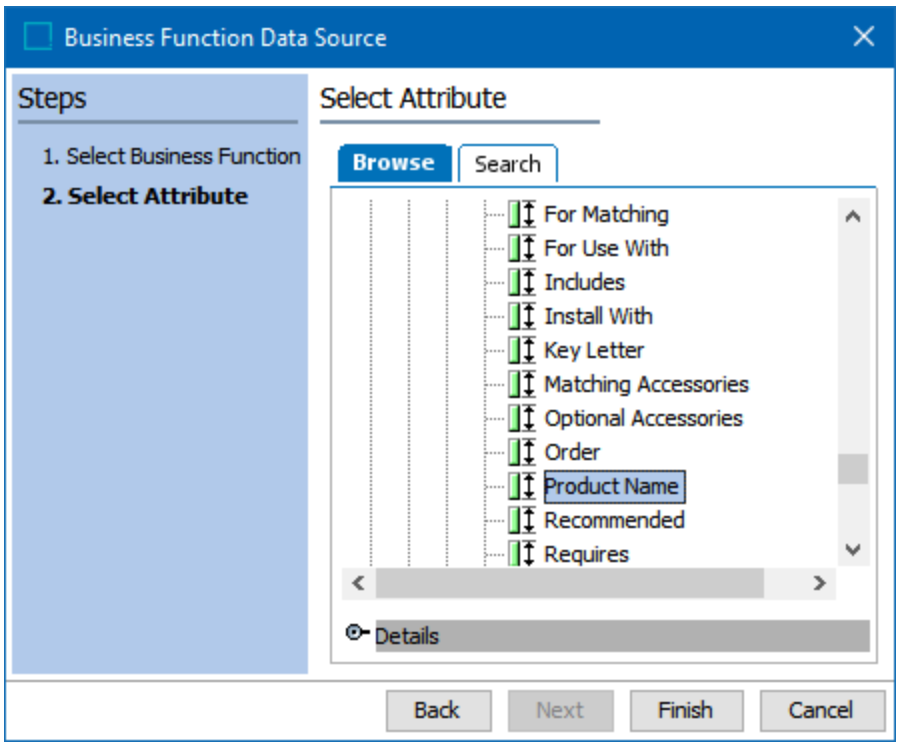
3. In the right panel, click the right arrow icon to display the **Business Function Data Source** wizard.



4. On the **Select Business Function** screen, select the relevant business function, then click **Next**.

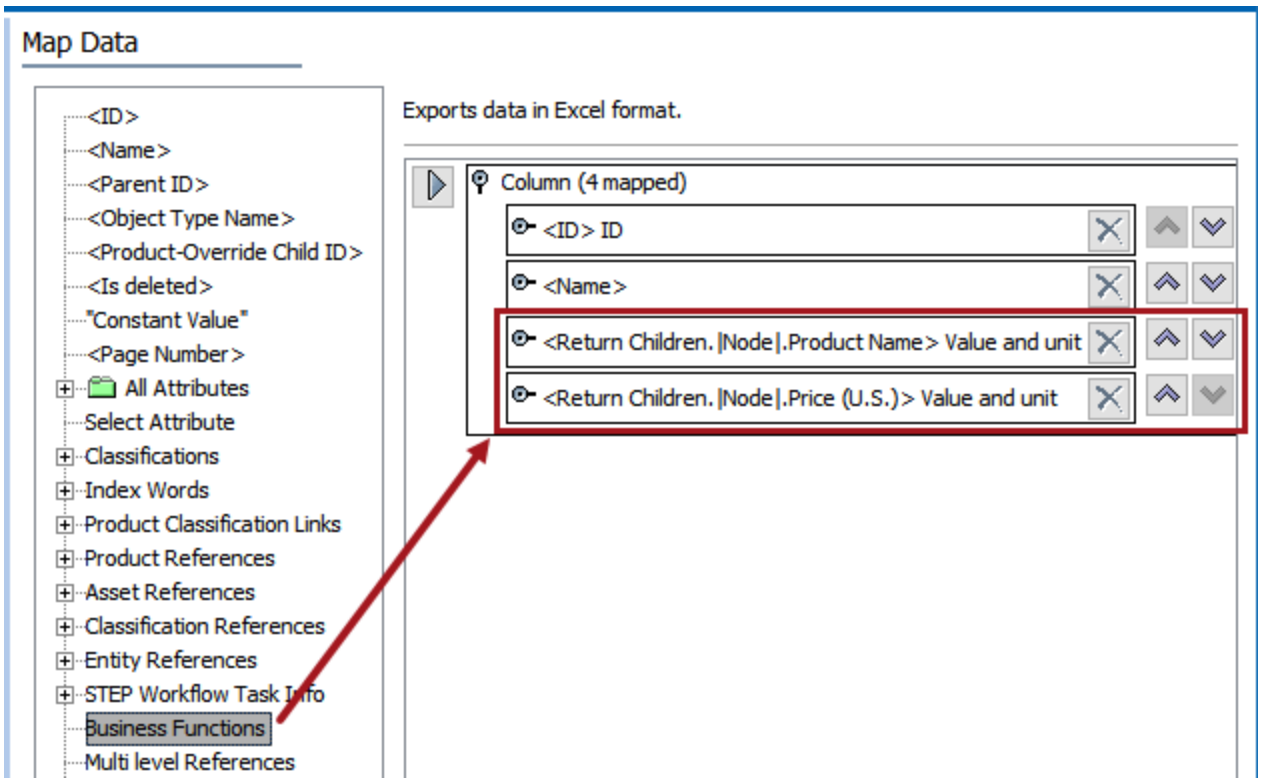
   Note: Only business functions that have **Node** as input and either **Node** or **List of Nodes** as output are valid for this functionality. As such, only these business functions display in the wizard.

5. On the **Select Attribute** screen, select the relevant attribute(s), then click **Finish**. To select more than one attribute, press Shift or Ctrl while making selections. An individual mapping will be created for each selected attribute.



6. The business function mapping is displayed on the **Map Data** screen. Follow the previous steps to map additional business functions. In the below screenshot, two business function columns have been mapped by mapping two different attribute selections from the same business function.

   In the exported Excel or CSV sheet, one column is returned per mapped business function. If the business function returns multiple nodes, then the attribute value will be found for each node, and the values will display within the column, separated by a multi-separator, e.g., **;**.

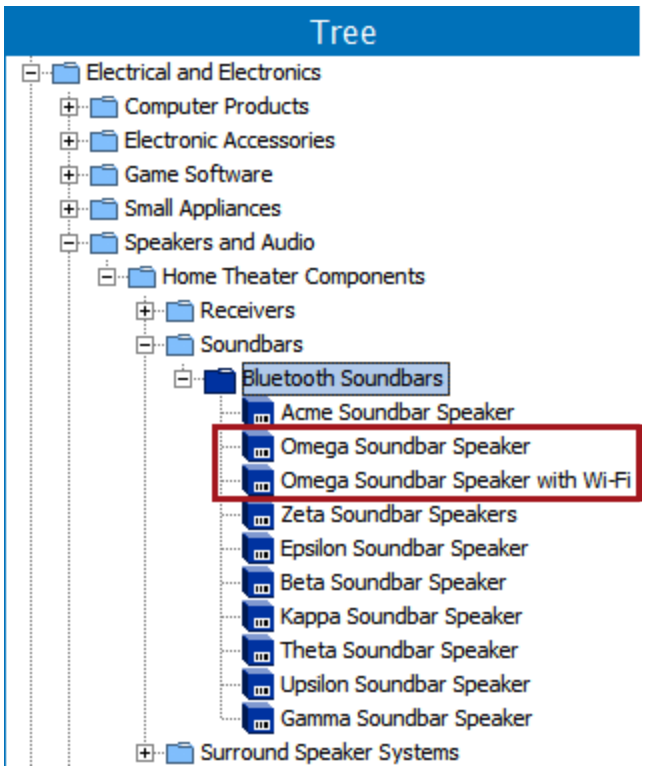


Example Use Cases and Results

Two sample use cases for this functionality are as follows:

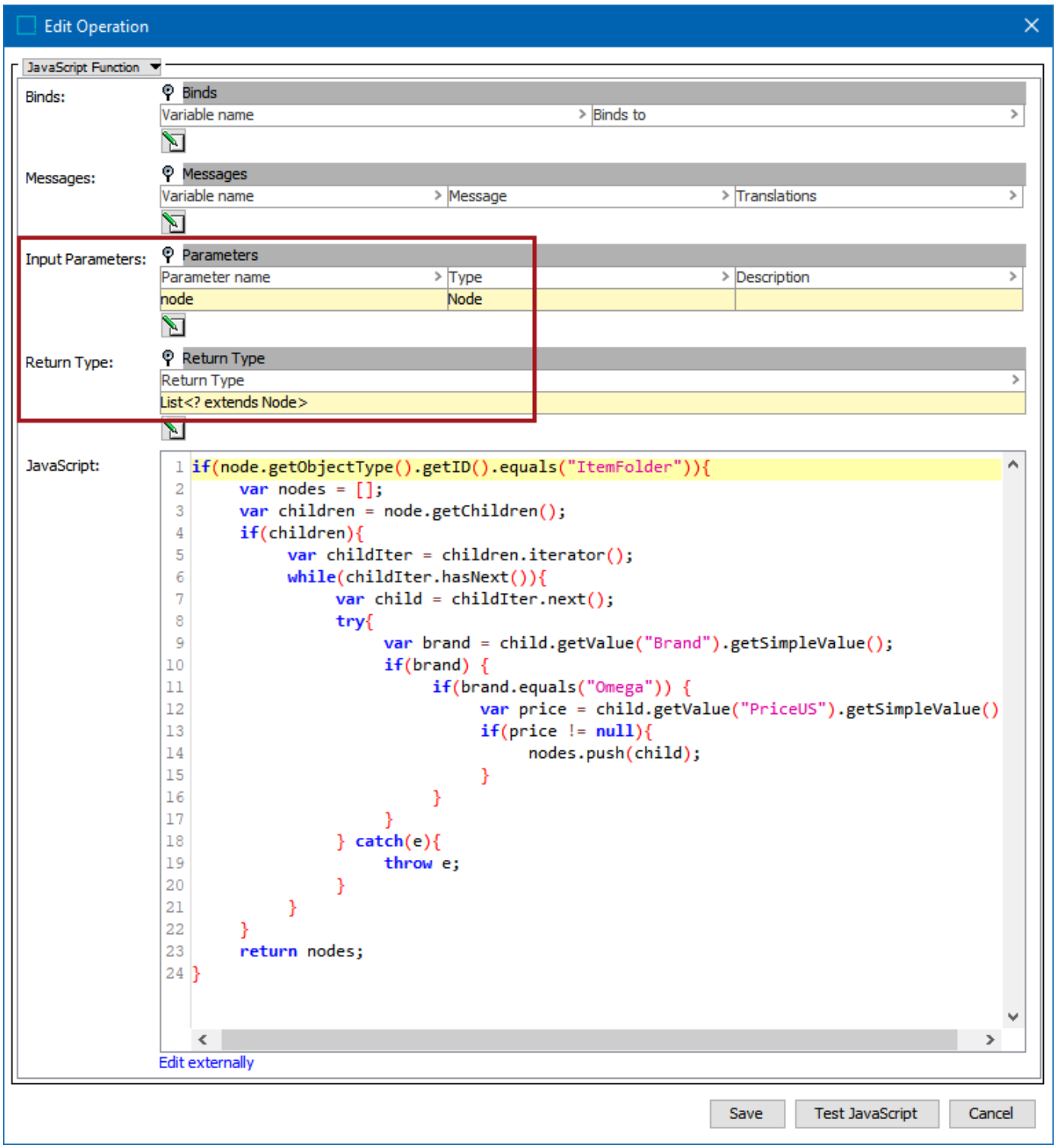
- Return a list of *child* product objects where the 'Brand' attribute equals a specified value (e.g., Brand = Omega). If a price value also exists for these objects, return the attribute value(s) from the node(s) to be used in an export.
- Return a list of *sibling* product objects where Brand = Omega. If a price value exists for these objects, return the attribute value(s) from the node(s) to be used in an export.

For both of these examples, the following sample data model will be used. The two Omega-branded products will be returned in the export for each example.



Example 1 - 'Return Children' Business Function

For the 'Return Children' example, a business function is used with the Input Parameter of 'Node' and Return Type of 'List of Nodes.' The function is intended to be used for exports performed at the ItemFolder level and will return a list of *child* product objects where the 'Brand' attribute = Omega and a 'PriceUS' value exists.



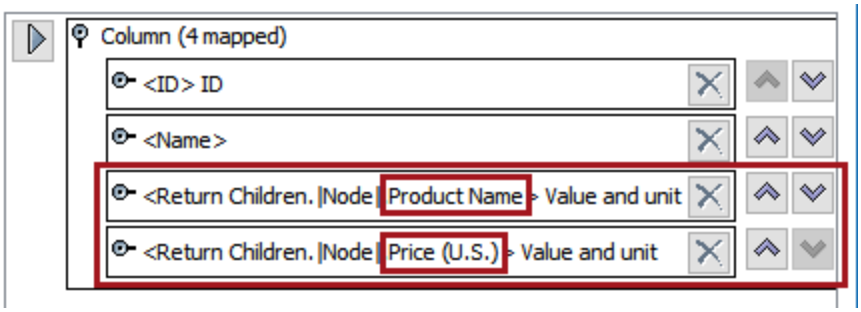
The code used in the above example is:

```
if(node.getObjectType().getID().equals("ItemFolder")){
    var nodes = [];
```

```
    var children = node.getChildren();
    if(children){
        var childIter = children.iterator();
        while(childIter.hasNext()){
            var child = childIter.next();
            try{
                var brand = child.getValue("Brand").getSimpleValue();
                if(brand) {
                    if(brand.equals("Omega")) {
                        var price = child.getValue("PriceUS").getSimpleValue();
                        if(price != null){
                            nodes.push(child);
                        }
                    }
                }
            } catch(e){
                throw e;
            }
        }
    }
    return nodes;
}
```

Example 1 - 'Return Children' Excel Sheet Result

Building upon the *mapping* shown in the configuration steps, the exported sheet will return the attribute values for 'Product Name' and 'Price (U.S.)' for all child objects where Brand = Omega and a value exists for Price (U.S.).

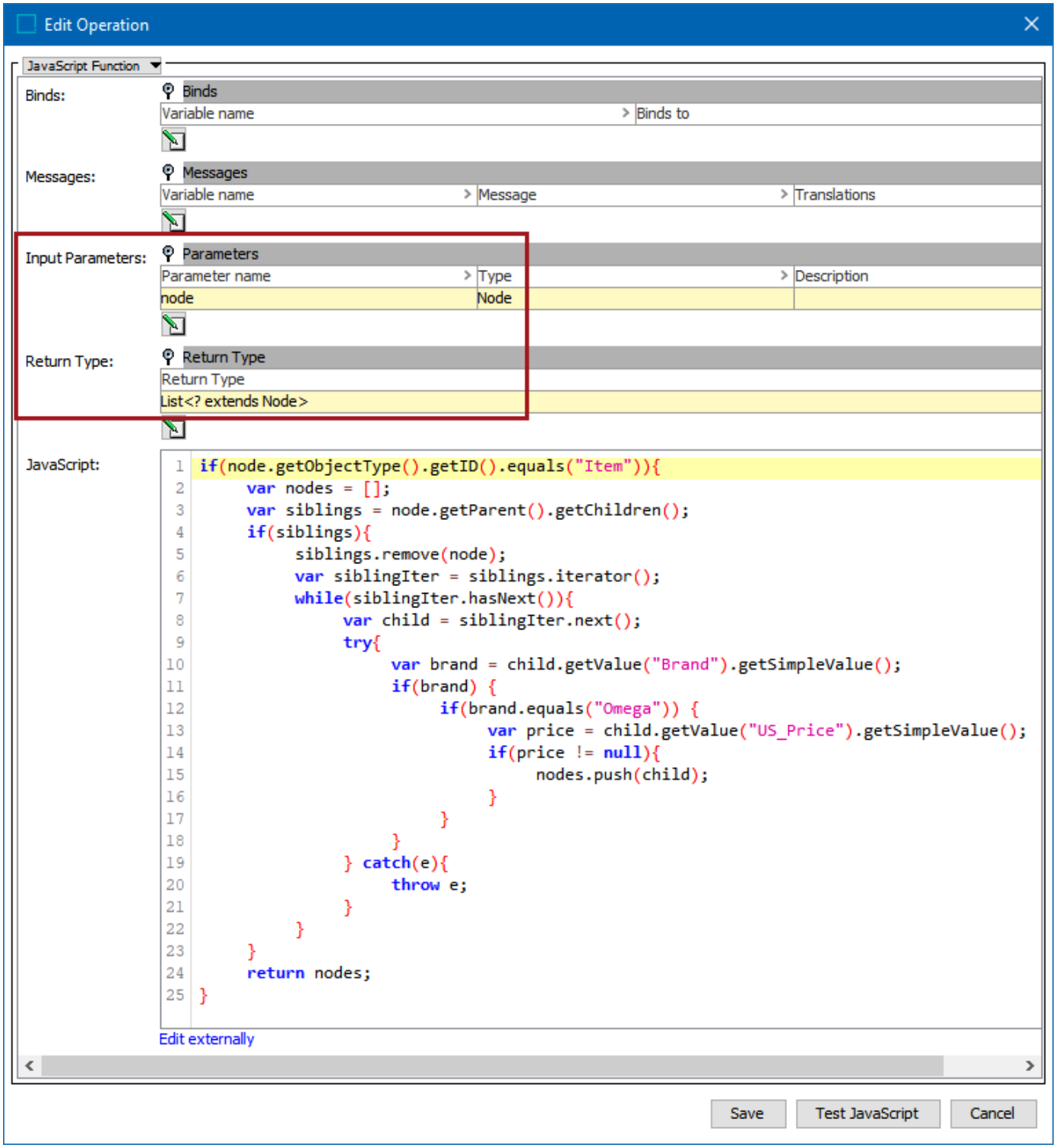


In the below example, the export has been performed from the Item Folder level (Bluetooth Soundbars) and has returned the Product Name and Price (U.S.) of the two Omega-branded children.

	A	B	C	D
1	<ID>	<Name>	<Return Children.|Node|.Product Name>	<Return Children.|Node|.Price (U.S.)>
2	306442	Bluetooth Soundbars	Omega Soundbar Speaker;Omega Soundbar Speaker with Wi-Fi	299.79 $;329.99 $
3				

Example 2 - 'Return Siblings' Business Function

For the 'Return Siblings' example, a business function is used with the Input Parameter of 'Node' and Return Type of 'List of Nodes.' The function is intended to be used for exports performed at the Item level and will return a list of *sibling* product objects where the 'Brand' attribute = Omega and a 'PriceUS' value exists.

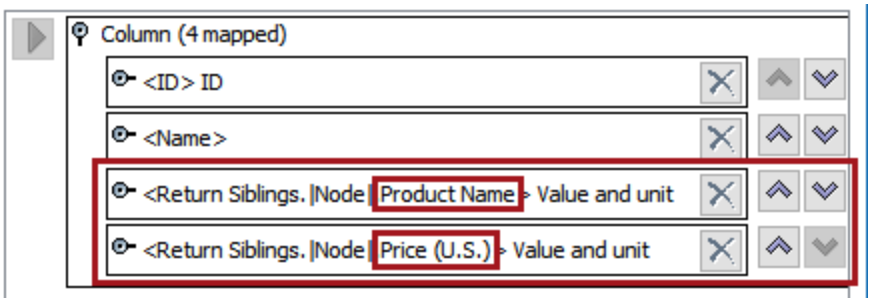


The code used in the above example is:

```
if(node.getObjectType().getID().equals("Item")){
    var nodes = [];
    var siblings = node.getParent().getChildren();
    if(siblings){
        siblings.remove(node);
        var siblingIter = siblings.iterator();
        while(siblingIter.hasNext()){
            var child = siblingIter.next();
            try{
                var brand = child.getValue("Brand").getSimpleValue();
                if(brand) {
                    if(brand.equals("Omega")) {
                        var price = child.getValue("US_Price").getSimpleValue();
                        if(price != null){
                            nodes.push(child);
                        }
                    }
                }
            } catch(e){
                throw e;
            }
        }
    }
    return nodes;
}
```

Example 2 - 'Return Siblings' Excel Sheet Result

Based on the *mapping* in the export configuration, the exported sheet will return the attribute values for Product Name and Price (U.S.) for all *sibling* objects where Brand = Omega and a value exists for Price (U.S.).



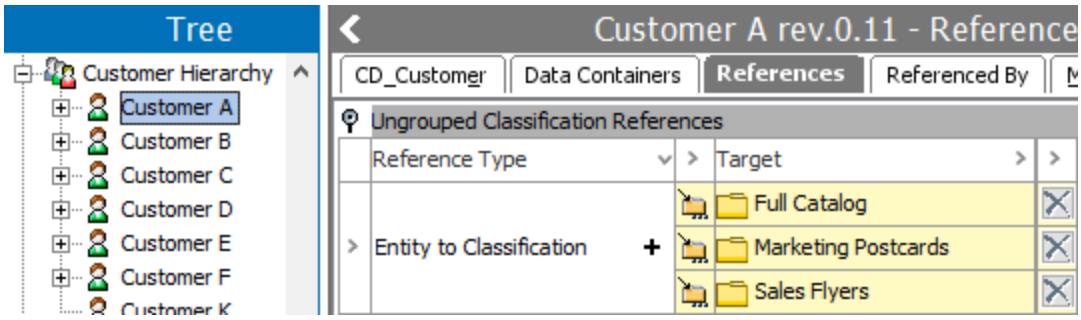
In the below example, the export has been performed from the Item level (Acme Soundbar Speaker) and has returned the Product Name and Price (U.S.) of the two Omega-branded siblings.

	A	B	C	D
1	<ID>	<Name>	<Return Siblings.|Node|.Product Name>	<Return Siblings.|Node|.Price (U.S.)>
2	298229	Acme Soundbar Speaker	Omega Soundbar Speaker;Omega Soundbar Speaker with Wi-Fi	299.79 $;329.99 $

Classification References – Data Source Outbound

The option to map a classification is only available when the object super type selected for the export is Product or Entity. Only the classification reference types displayed are those that match the selected object type. Classification-to-classification references are not available for output.

When mapping a classification reference, users can include or exclude inherited references.



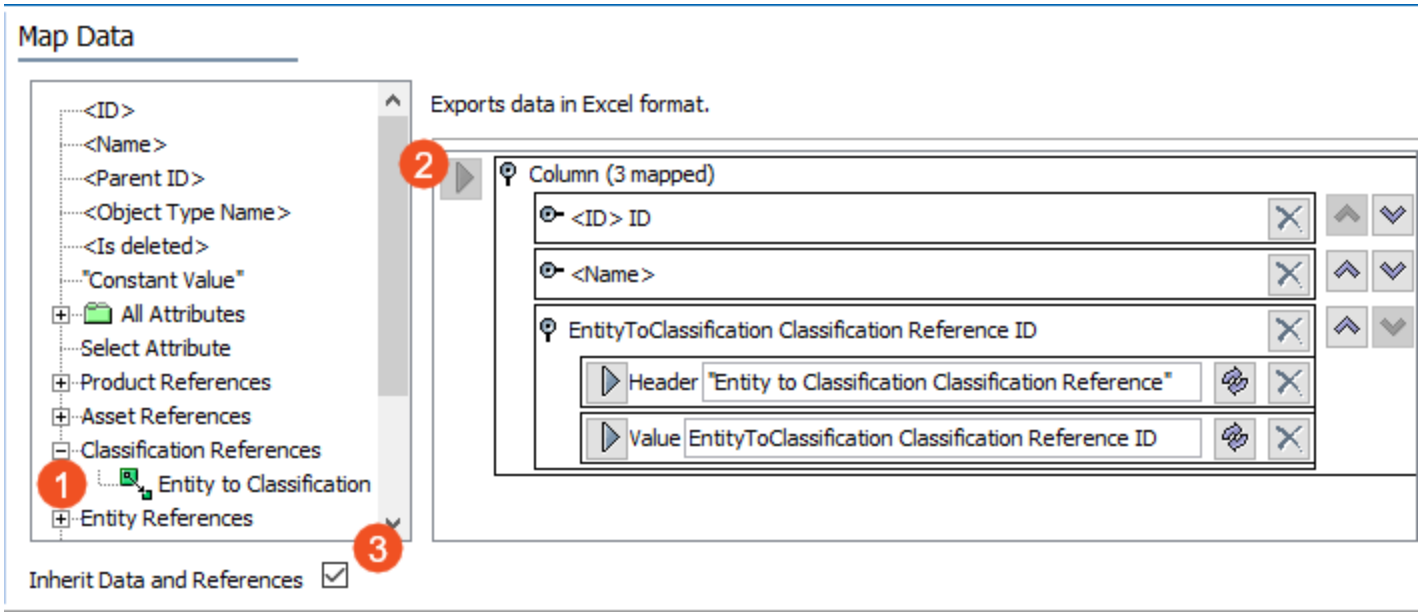
The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

For more information about the additional wizard steps, refer to the **Creating a Data Export** topic or the **Creating an Outbound Integration Endpoint** topic.

After mapping, most output options can be altered using transformations. For more information, refer to the **Outbound Map Data - Transform** topic.

Mapping Classification References

1. Select the data to be exported for the output tool. For more information, refer to the **Data Exchange** topic.
2. On the Map Data step, in the left panel, expand Classification References, and select the desired Classification Reference type.



3. Click the right arrow button in the right panel to add the selected classification reference type as a column. Add additional classification reference types, if needed.
4. Check the **Inherit Data and References** option appropriately:
   - If **checked**, inherited data and references are included in the export.
   - If **unchecked**, inherited data and references is not included in the export.

   For information on inheritance, refer to the **Inherit Data and References - Data Source Outbound** topic.
5. Apply any transformations, which can change the output without changing the original data. Refer to the **Outbound Map Data - Transform** topic.
6. Complete any additional mappings and initiate the export.

Results

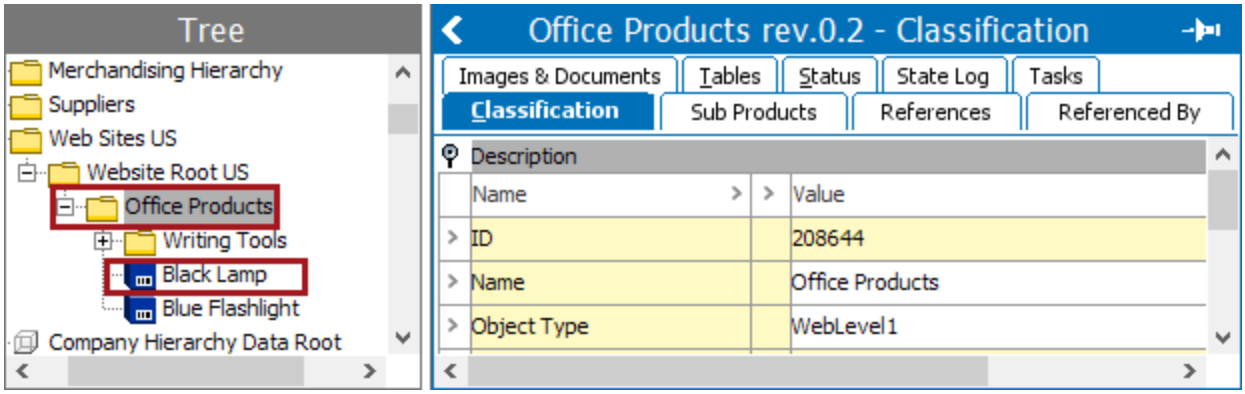
The output includes the classification reference type name as a header and the ID for each row.

When the product references multiple classifications, the classification reference IDs are displayed in a semi-colon separated list.

	A	B	C
1	<ID>	<Name>	Entity to Classification Classification Reference
2	CUS_101545	Customer A	FullCatalog;MarketingPostcards;SalesFlyers
3	CUS_107835	Customer B	SalesFlyers

Classifications - Data Source Outbound

The option to map a classification is only available when the object type selected for the export is Product. Additionally, the ID of a classification is only output when the product being exported has a reference to the mapped classification or to one of the descendant classifications. For example, by mapping the top level classification, details of all the classifications referenced by product are exported.



The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

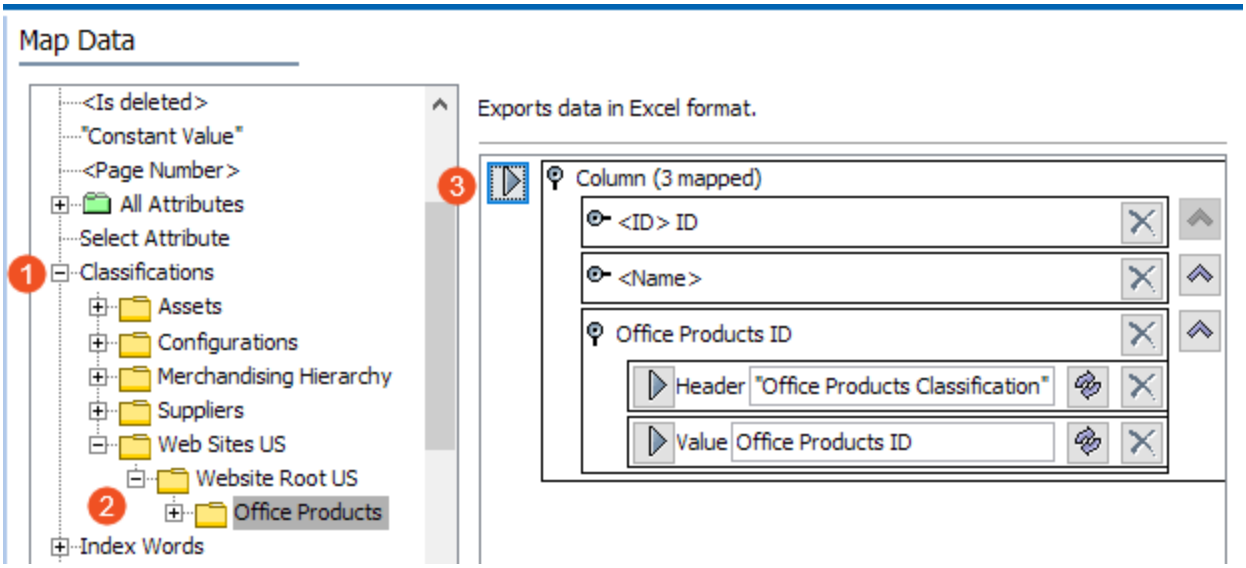
For more information about the additional wizard steps, refer to the **Creating a Data Export** topic or the **Creating an Outbound Integration Endpoint** topic.

After mapping, most output options can be altered using transformations. For more information, refer to the **Outbound Map Data - Transform** topic.

Mapping Classifications

1. Select the data to be exported for the output tool. For more information, refer to the **Data Exchange** topic.
2. On the Map Data step, in the left panel, expand the Classification node to show all the classifications.

3. Select the desired classification and click the right arrow button.



4. Apply any transformations, which can change the output without changing the original data. Refer to the **Outbound Map Data - Transform** topic.
5. Complete any additional mappings and initiate the export.

Results

The output includes the classification reference type name as a header and the ID for each row.

	A	B	C
1	<ID>	<Name>	Office Products Classification
2	208653	Black Lamp	208644

When the product references multiple classifications, the classification IDs are displayed in a semi-colon separated list.

	A	B	C
1	<ID>	<Name>	Suppliers Classification
2	134887	Blue Flashlight	20851;208645

Constant Value - Data Source Outbound

Mapping a constant value allows you to replace the default data with your own text. For example, use a constant value to change a header from the default of <Name> to 'Product Name', or to add static text to the value.

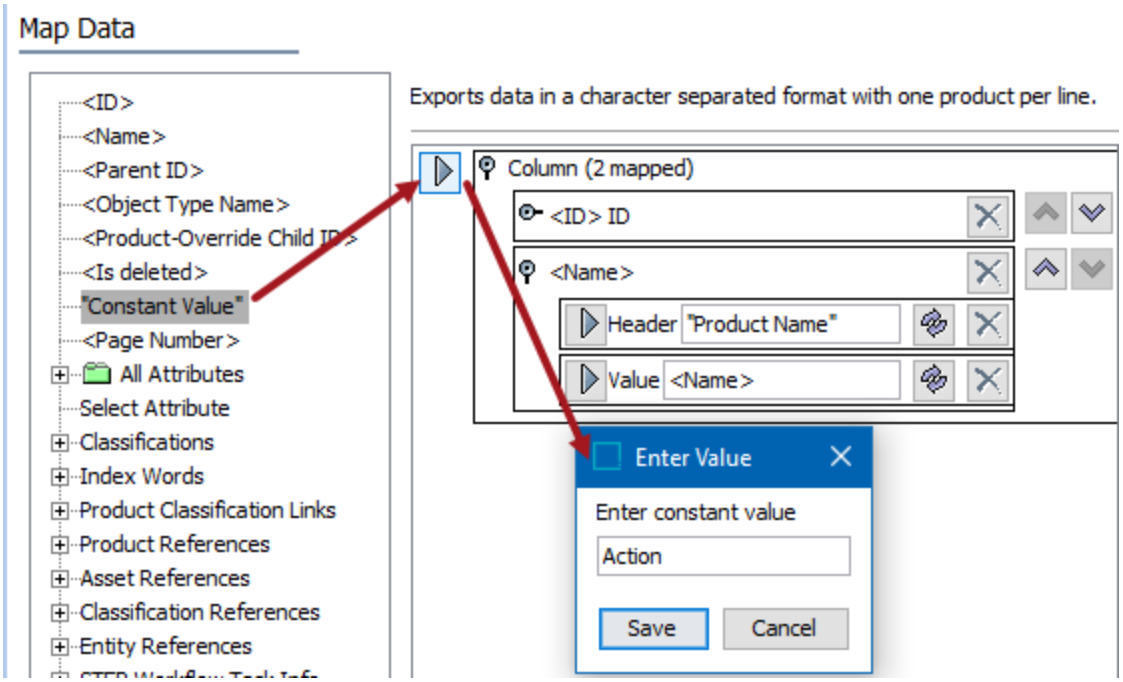
The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

For more information about the additional wizard steps, refer to the **Creating a Data Export** topic or the **Creating an Outbound Integration Endpoint** topic.

After mapping, most output options can be altered using transformations. For more information, refer to the **Outbound Map Data - Transform** topic.

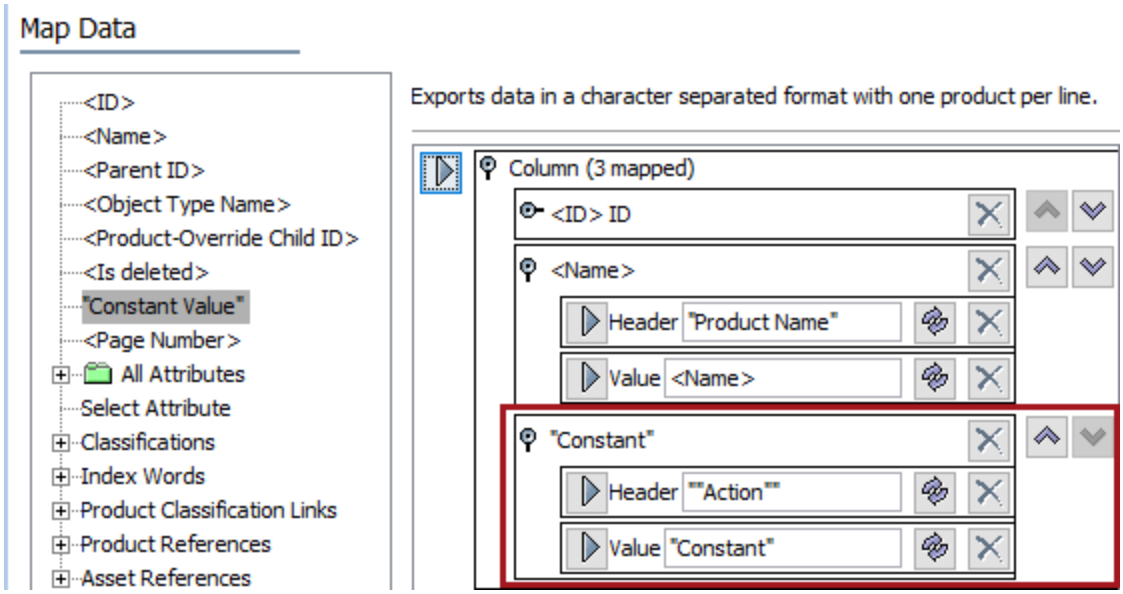
For an expanded example of mapping aspects and constants, refer to the **Header and Value Aspects Example** documentation.

Mapping Constant Value as a New Field

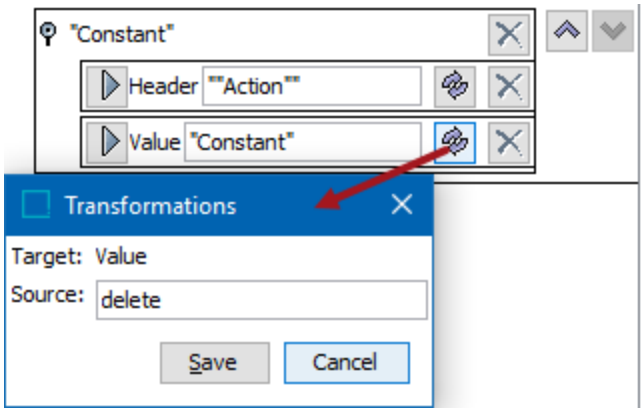


1. Select the data to be exported for the output tool. For more information, refer to the **Data Exchange** topic.
2. On the Map Data step, in the left panel, select **Constant Value**.
3. In the right panel, click the top-most right arrow ( ) to add a new field to the export file.

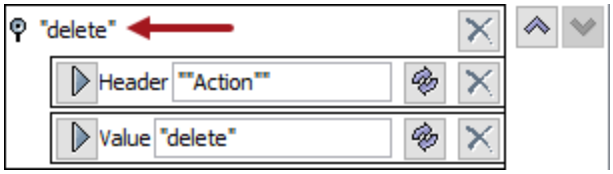
4. In the **Enter Value** dialog, enter the constant value to assign to all records in the export file, and click **Save**.
5. The new "Constant" filed is displayed using the text you entered as the Header.



6. Use the transformation button to update the Value field, if necessary. You can also use the transformation to remove the extra set of double quotes displayed by default.



7. Click **Save** and the value is then displayed as the field name.



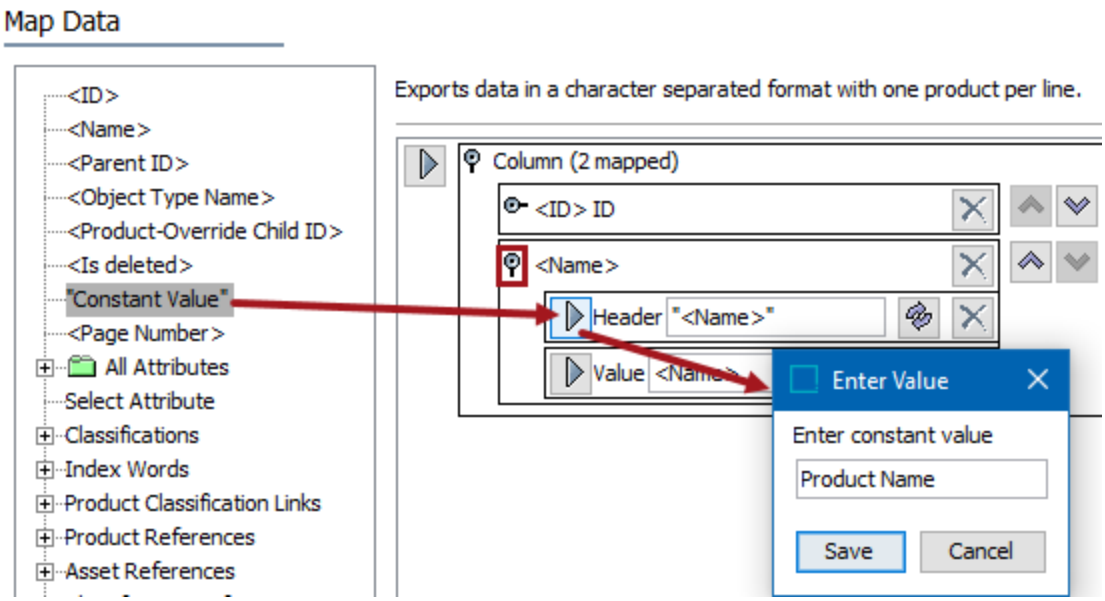
8. Complete the mapping and initiate the export.

Results

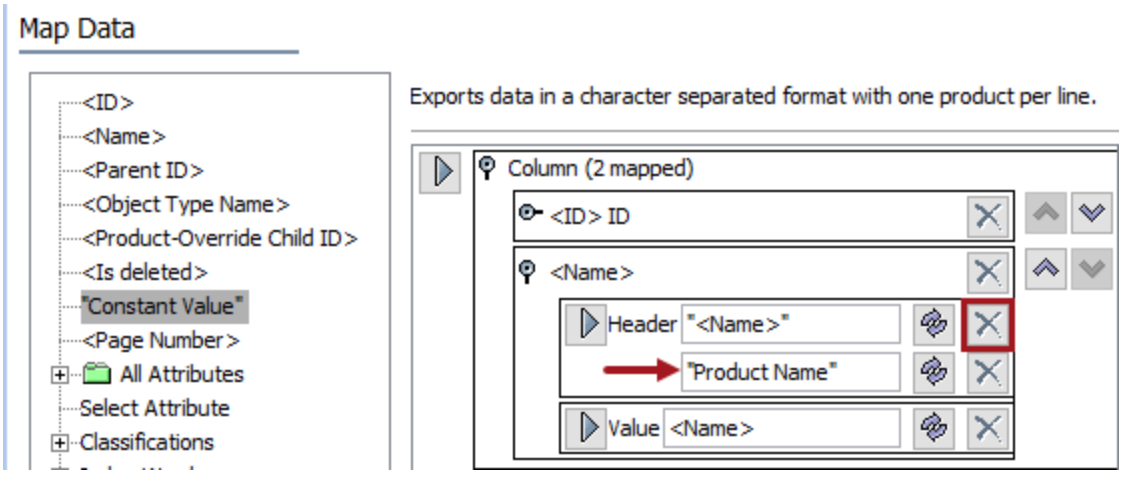
The exported file includes ID, product name and an action column with constant value “delete”.

<ID>	Product Name	"Action"
123853	T-Shirts Family	delete
123854	123854 B	delete
123855	123855 O	delete
123857	123857 G	delete

Mapping Constant Value Within a Field



1. In the left panel, select **Constant Value**.
2. In the right panel, click the right arrow ( ) for the desired field.
3. In the **Enter Value** dialog, enter the constant value to assign to all records in the export file, and click **Save**.
4. The constant value is displayed as a new element in the right panel. If necessary, remove any duplicate data by clicking the X button



5. Complete any additional mappings and initiate the export.

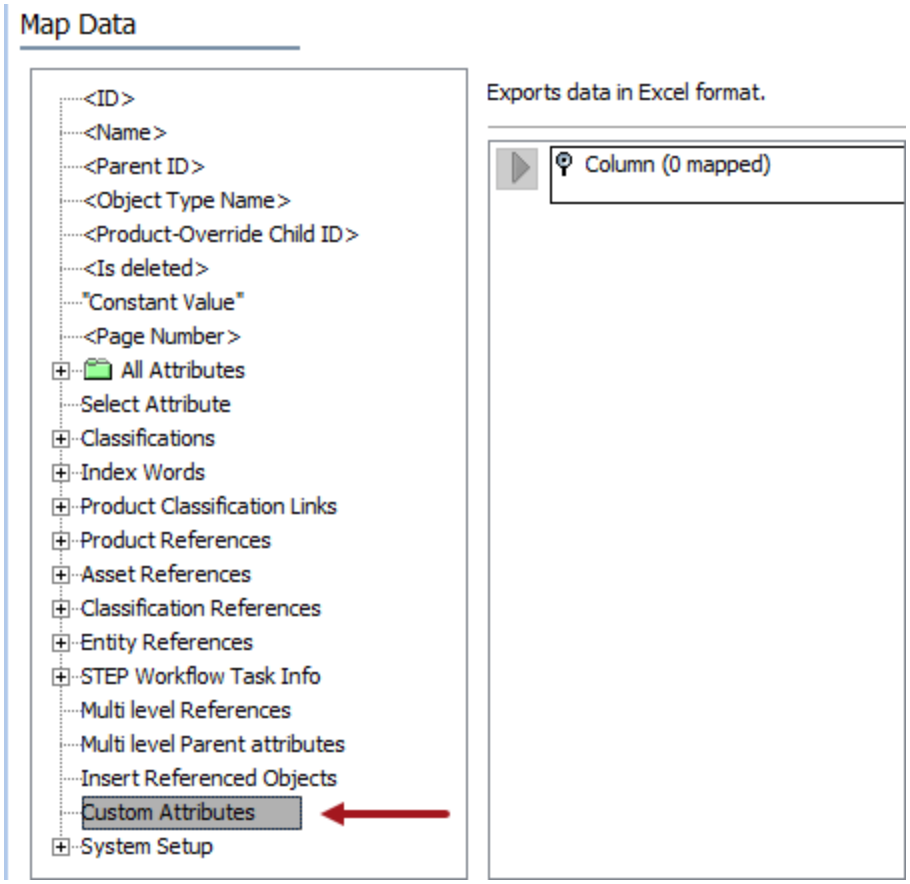
Results

The exported file includes existing name column header to be modified with a constant value as 'Product Name.'

<ID>	Product Name
123853	T-Shirts Family
123854	123854 B
123855	123855 O
123857	123857 G

Custom Attributes - Data Source Outbound

Custom attributes are legacy functionality only available via custom code written by a STEP developer. When custom attributes exist, the Custom Attributes data source is used to export the data.



The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

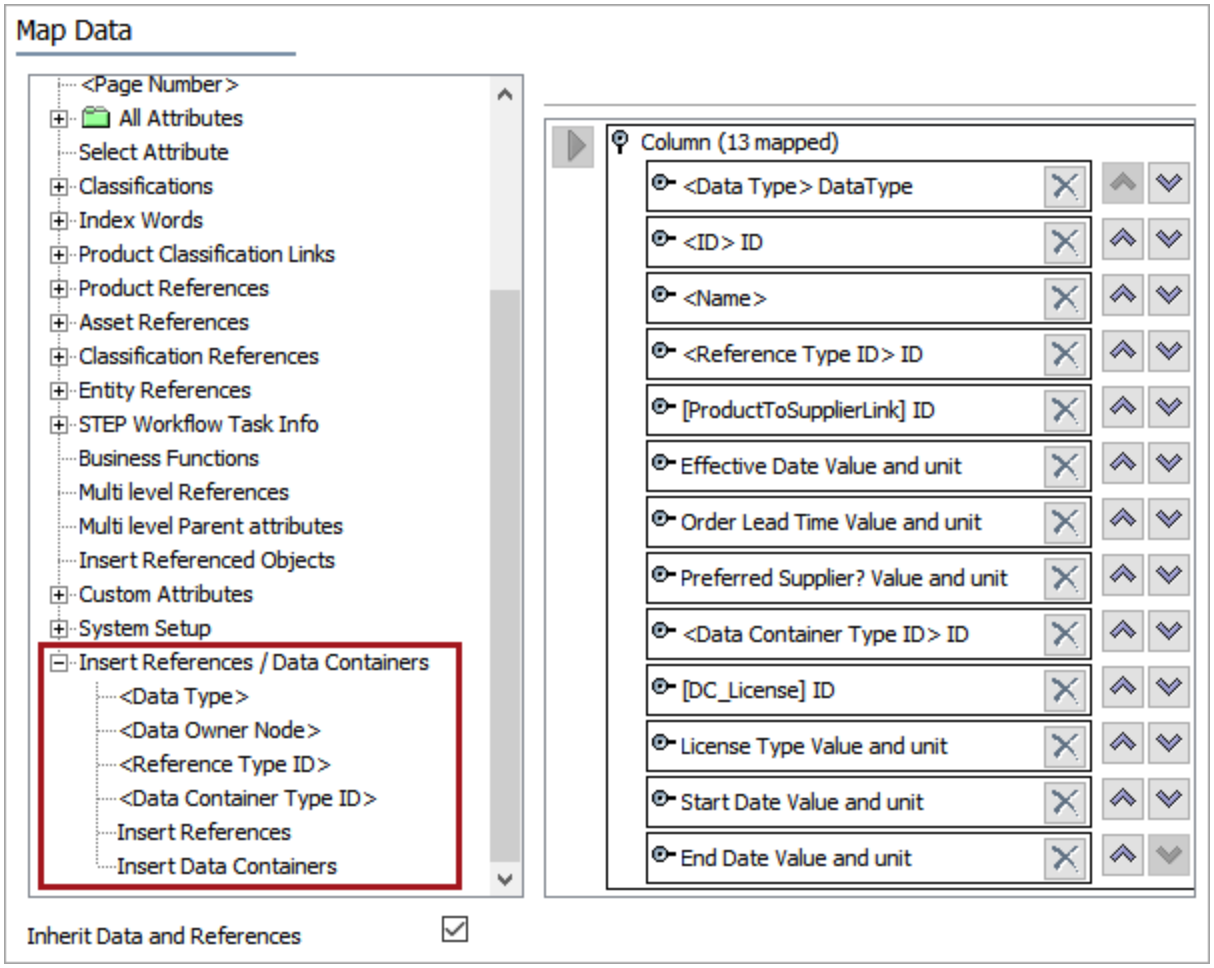
For more information about the additional wizard steps, refer to the **Creating a Data Export** topic or the **Creating an Outbound Integration Endpoint** topic.

After mapping, most output options can be altered using transformations. For more information, refer to the **Outbound Map Data - Transform** topic.

Data Container Type ID - Data Source Outbound

The 'Data Container Type ID' option allows output of the ID of the data container type for rows representing a data container object. This option is primarily relevant to export data containers of different types or to be able to import the exported file again.

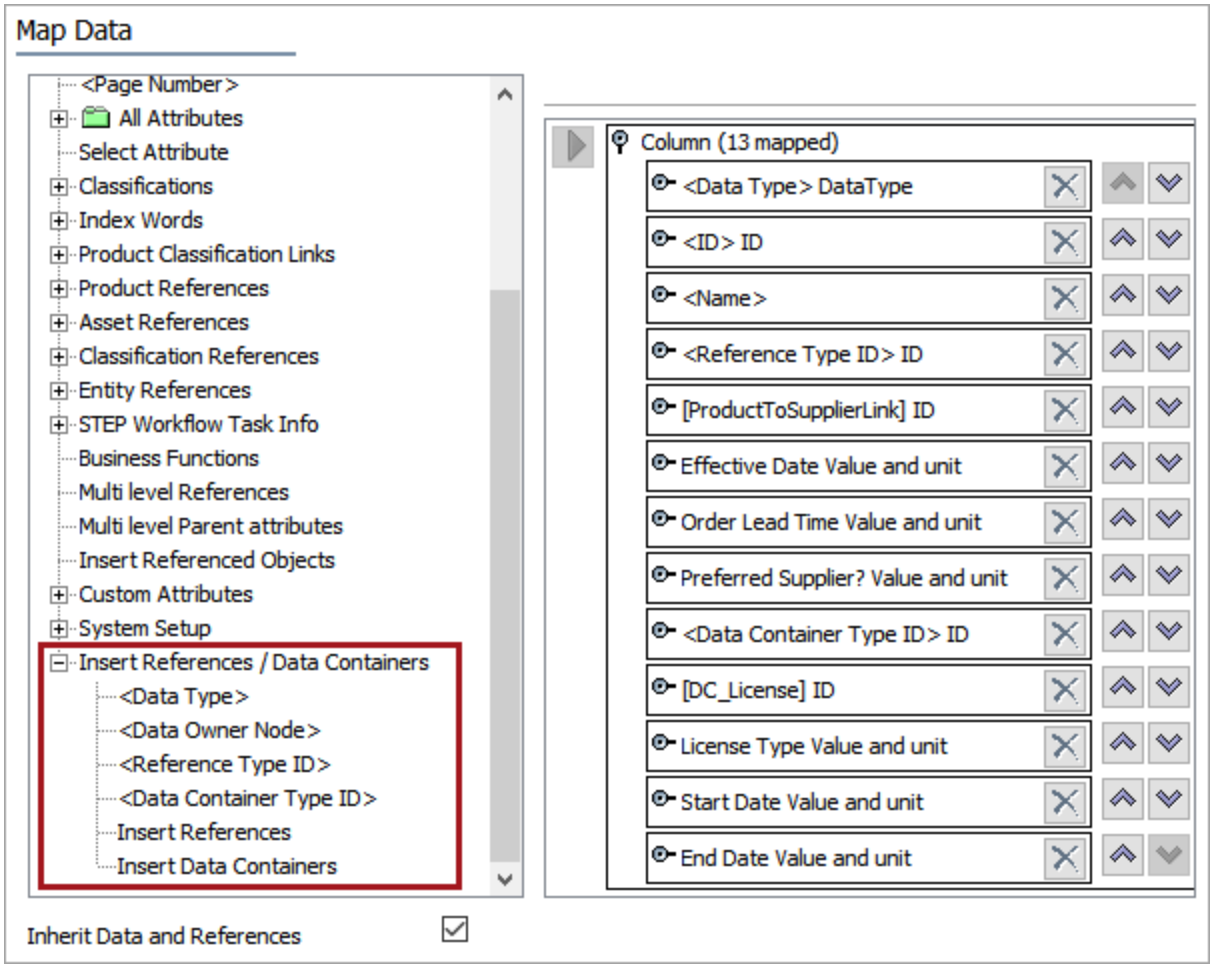
For details on using the options in the 'Insert References / Data Containers' group, refer to the **Insert References - Data Source Outbound** topic and the **Insert Data Container - Data Source Outbound** topic.



Data Owner Node - Data Source Outbound

When using the 'Insert References / Data Containers' functionality, by convention, all non-NODE rows following a NODE row, represent data owned by the NODE. The 'Data Owner Node' option allows you to make the relationship between the NODE and non-NODE rows more explicit. For each row representing a data container object, a reference object, or a classification product link object, the owner node ID can be included in the 'Data Owner Node' column.

For details on using the options in the 'Insert References / Data Containers' group, refer to the **Insert References - Data Source Outbound** topic and the **Insert Data Container - Data Source Outbound** topic.



Data Path - Data Source Outbound

The Data Path data source is only available with the Generic XML, Generic JSON, and IDoc MATMAS 05 formats. The following mapping examples use the 'Data Path' data source:

- Nestable Data Paths in Generic XML
- Nestable Data Paths in Generic XML Example
- Mapping Data Path Example in Generic JSON Format
- Mapping for Nestable Data Paths in Generic XML Example

The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

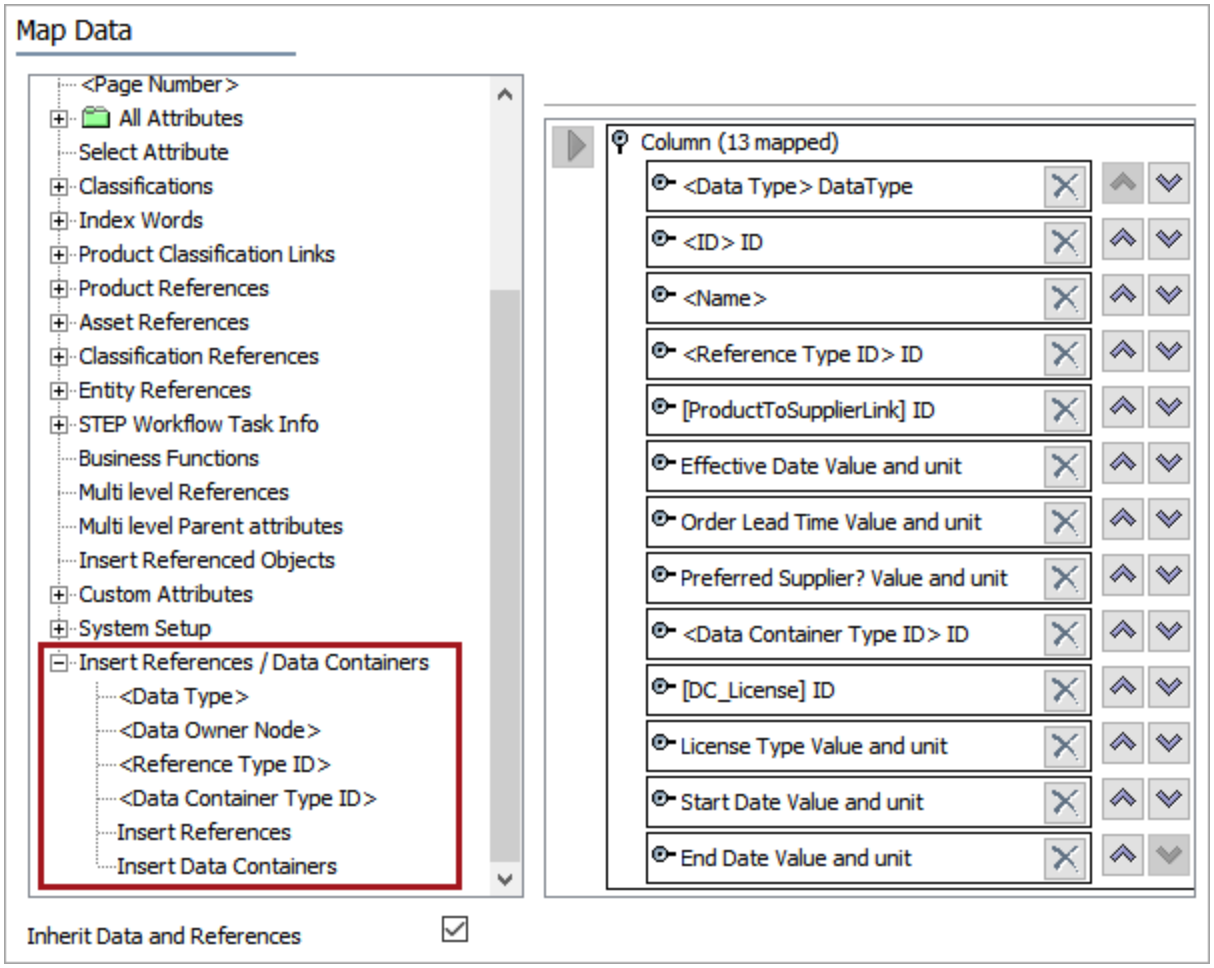
For more information about the additional wizard steps, refer to the **Creating a Data Export** topic or the **Creating an Outbound Integration Endpoint** topic.

After mapping, most output options can be altered using transformations. For more information, refer to the **Outbound Map Data - Transform** topic.

Data Type - Data Source Outbound

The 'Data Type' data source exports static text values of 'NODE', 'DATA_CONTAINER', or 'REFERENCE' to identify the reference or data container information on the row.

For details on using the options in the 'Insert References / Data Containers' group, refer to the **Insert References - Data Source Outbound** topic and the **Insert Data Container - Data Source Outbound** topic.



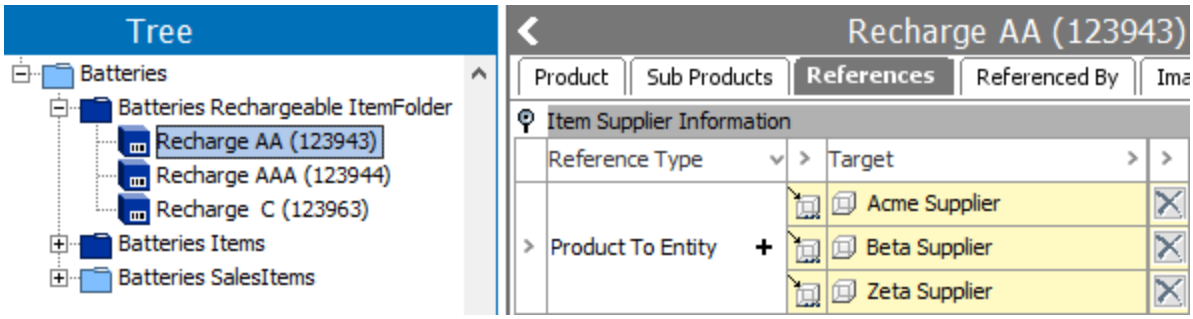
Entity References – Data Source Outbound

The option to map an entity reference is available when the object super type selected for the export is Asset, Classification, Entity, or Product. The only displayed entity reference types for mapping are those that match the selected super type. Publication-to-entity references are not available for output.

For information on selecting an object super type, refer to the following topics:

- For Export Manager, use the Export dropdown as described in the **Export Manager - Select Objects** topic.
- For an event-based OIEP, use the object super type dropdown as described in the **OIEP - Event-Based - Output Templates Flipper** topic.
- For a select objects OIEP, use the object super type dropdown as described in the **OIEP - Select Objects - Output Templates Flipper** topic.

For example, consider details of various suppliers of a product, where suppliers are the entity object within STEP. The entity reference mapping would allow you to extract the list of suppliers for a product.



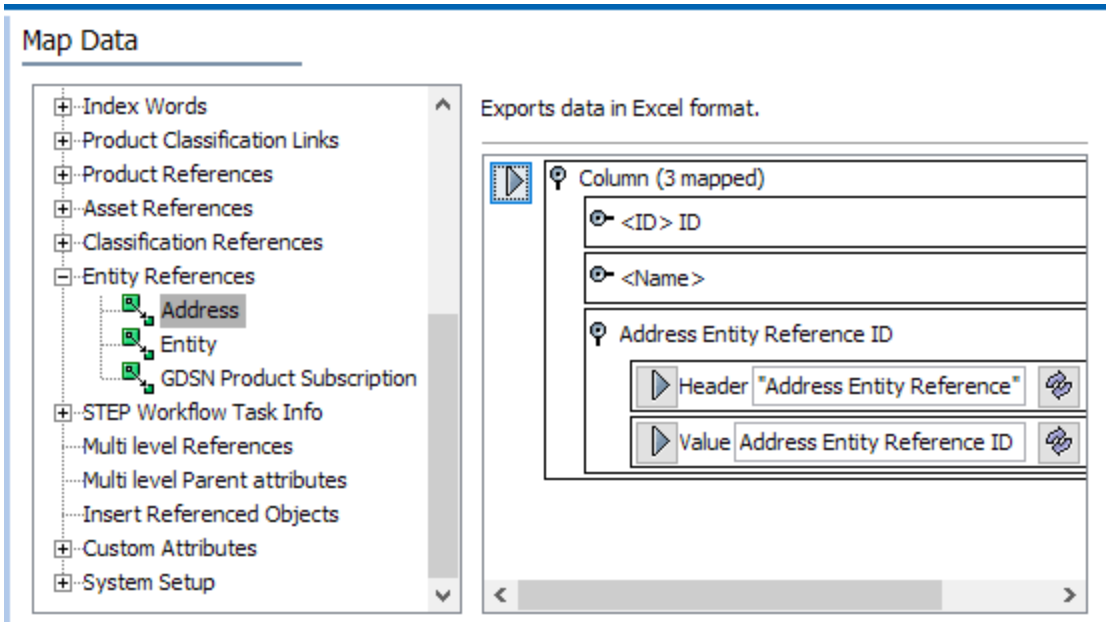
The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

For more information about the additional wizard steps, refer to the **Creating a Data Export** topic or the **Creating an Outbound Integration Endpoint** topic.

After mapping, most output options can be altered using transformations. For more information, refer to the **Outbound Map Data - Transform** topic.

Mapping Entity References

1. Select the data to be exported for the output tool. For more information, refer to the **Data Exchange** topic.
2. On the Map Data step, in the left panel, select **Entity References** to expand the node.
3. Select the desired entity reference type.



4. In the right panel, click the top-most right arrow to add the selected entity reference type to the export file.
5. Apply any transformations, which can change the output without changing the original data. Refer to the **Outbound Map Data - Transform** topic.
6. Complete any additional mappings and initiate the export.

Results

The output includes the entity reference type name as a header and the ID for each row.

When the object references multiple entities, the entity reference IDs are displayed in a semi-colon separated list.

	A	B	C
1	<ID>	<Name>	Product To Entity Entity Reference
2	123943	Recharge AA (123943)	AcmeSupplier;BetaSupplier;ZetaSupplier
3	123944	Recharge AAA (123944)	ZetaSupplier

ID or Key - Data Source Outbound

When the STEP ID is used to identify an object, the ID data source is used to identify the mapping target intended to hold ID. All objects in STEP can be exported using the ID data source.

When a key is used in place of the STEP ID by external systems, prior to exporting, keys must be activated in STEP so they can be displayed in the Map Data step of an export tool. For more on keys, refer to the **Unique Keys** section of the **System Setup** documentation.

While one image below shows Generic XML, and the steps below show Excel, this topic applies to any of the formats that require mapping as defined in the **Outbound Map Data Options** topic.

The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

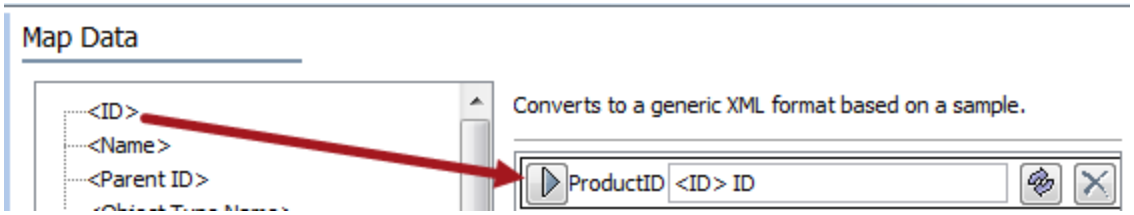
For more information about the additional wizard steps, refer to the **Creating a Data Export** topic or the **Creating an Outbound Integration Endpoint** topic.

After mapping, most output options can be altered using transformations. For more information, refer to the **Outbound Map Data - Transform** topic.

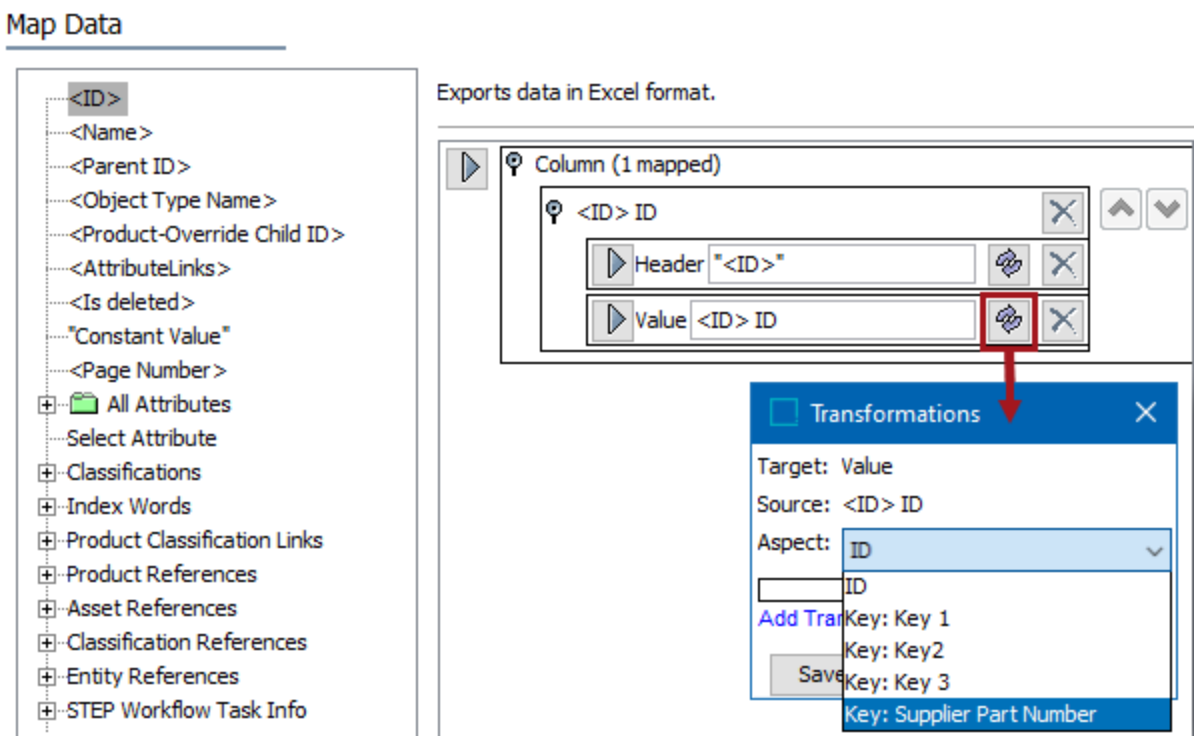
Mapping ID or Key

The steps for mapping IDs are very similar to those for mapping a key.

1. Select the data to be exported for the output tool. For more information, refer to the **Data Exchange** topic.
2. On the Map Data step, in the left panel, select **<ID>**.



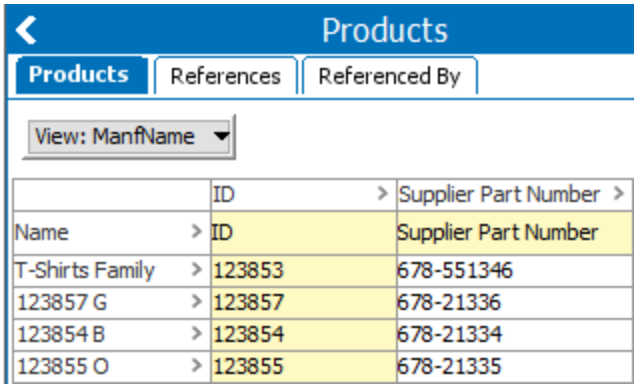
3. Click the right arrow button ( ) in the right panel to add the ID flipper as a column. If mapping the ID, skip to the last step.
4. If mapping a key, open the <ID> flipper, and under **Value**, click the **Transformation** icon ( ). In the **Transformations** dialog, select the relevant key from the **Aspect** list and click **Save**.



5. For both ID and Key mappings, finally, complete any additional mappings and initiate the export.

Results

The following data is exported with the results displayed below, based on mapping the ID or 'Supplier Part Number' key.



	ID	Supplier Part Number
Name	ID	Supplier Part Number
T-Shirts Family	123853	678-551346
123857 G	123857	678-21336
123854 B	123854	678-21334
123855 O	123855	678-21335

If only the ID is mapped, the exported file includes the IDs of the objects.

<ID>
123853
123854
123855
123857

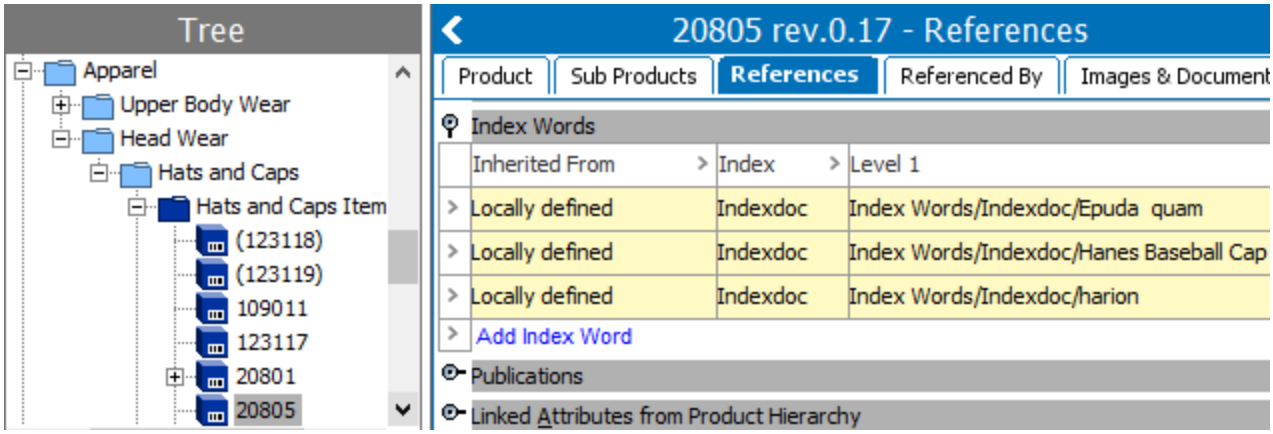
If only the 'Supplier Part Number' active key is mapped, the exported file includes the ID KeyID=SupplierPartNumber as header and its corresponding key values for each of the exported objects.

<ID KeyID=SupplierPartNumber>
678-551346
678-21334
678-21335
678-21336

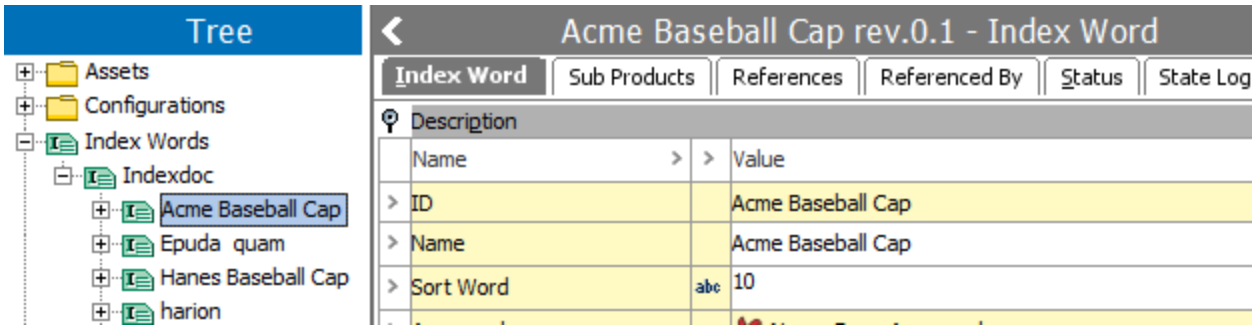
Index Words – Data Source Outbound

The option to map index words is available for an index hierarchy. For more information on indexes, refer to **Creating Document Indexes** in the **Publisher (Adobe InDesign Integration)** documentation.

Index words that are linked to products display on the product's References Tab under the Index Words flipper.



Index words modeled as classification objects are displayed in the classifications section of the Tree. The Sort Word attribute can be used to order an index non-alphabetically.



The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

For more information about the additional wizard steps, refer to the **Creating a Data Export** topic or the **Creating an Outbound Integration Endpoint** topic.

After mapping, most output options can be altered using transformations. For more information, refer to the **Outbound Map Data - Transform** topic.

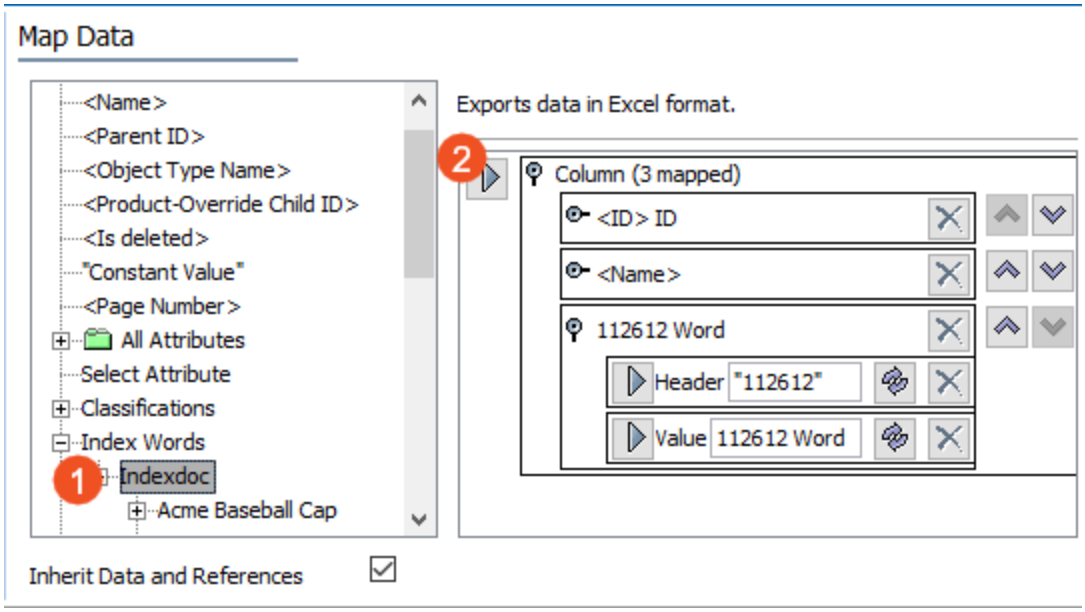
The index words of an index hierarchy can be exported as follows:

- Index Words Linked to Products Using Word - exports the index word
- Index Words Linked to Products Using Sort Word - exports the sort word attribute value
- Index Words from Classification - exports all index words in the selected hierarchy

Each of these options is defined below.

Mapping Index Words Linked to Products Using Word

1. Select the data to be exported for the output tool. For more information, refer to the **Data Exchange** topic.
2. Select the Product super type. For more information, refer to **Export Manager - Select Objects**, the **Configure the Format** section of the **OIEP - Event-Based - Output Templates Flipper**, or the **Configure the Format** section of the **OIEP - Select Objects - Output Templates Flipper**.
3. Ensure products with index word references are selected for output.
4. On the Map Data step, in the left panel, expand Index Words, and select the desired index words.



5. Click the right arrow button ( ) in the right panel to add the selected index words type as a column. The added column displays that Word is being used for export. Add additional index words, if needed.
6. Apply any transformations, which can change the output without changing the original data. Refer to the **Outbound Map Data - Transform** topic.
7. Complete any additional mappings and initiate the export.

Results of Index Words Linked to Products Using Word

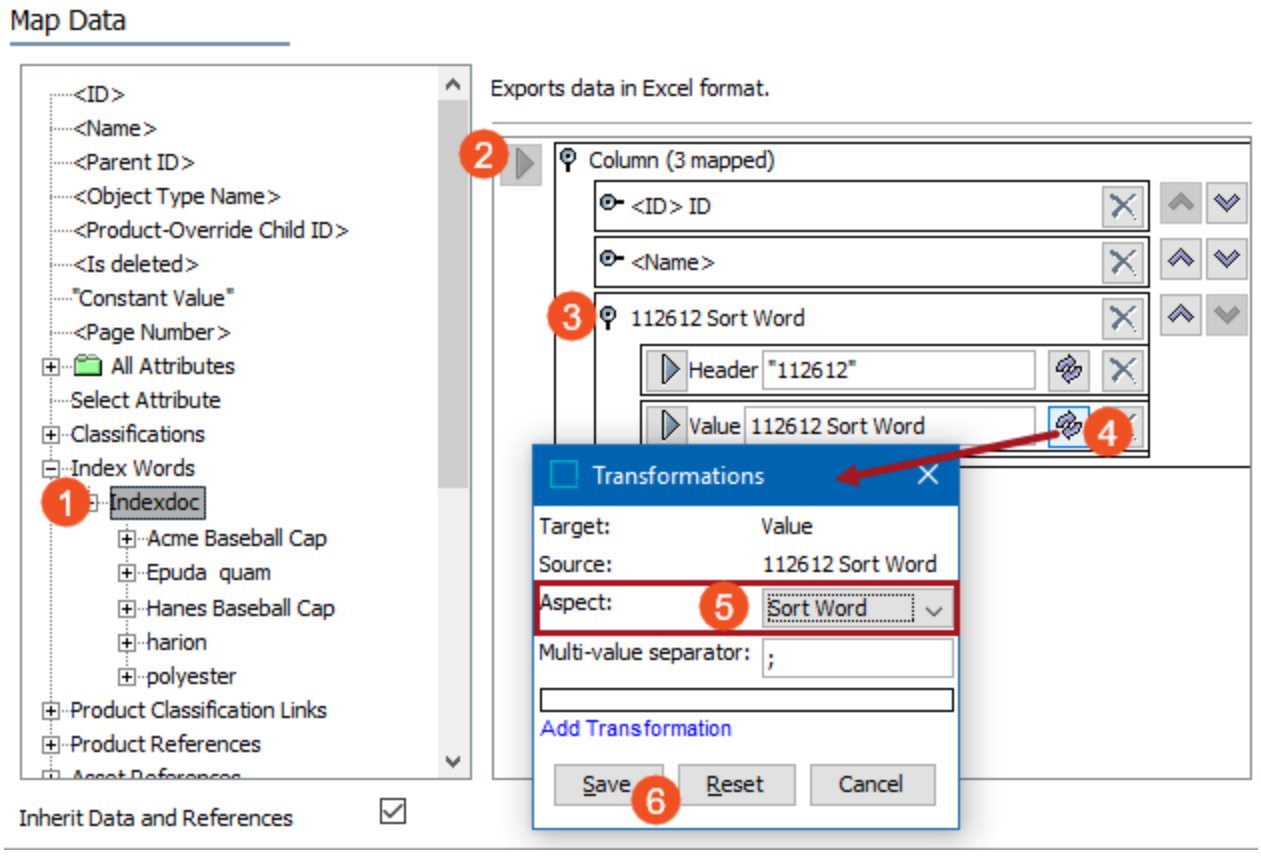
The output includes the ID and name (of product), and index word ID as headers.

When the multiple index words are linked to a product, the index words names are displayed in a semi-colon separated list.

	A	B	C
1	<ID>	<Name>	112612
2	121933	121933	Acme Baseball Cap
3	20805	20805	Hanes Baseball Cap;Epuda quam;harion

Mapping Index Words Linked to Products Using Sort Word

1. Select the data to be exported for the output tool. For more information, refer to the **Data Exchange** topic.
2. Select the Product super type. For more information, refer to **Export Manager - Select Objects**, the **Configure the Format** section of the **OIEP - Event-Based - Output Templates Flipper**, or the **Configure the Format** section of the **OIEP - Select Objects - Output Templates Flipper**.
3. Ensure products with index word references are selected for output.
4. On the Map Data step, in the left panel, expand Index Words, and select the desired index words.



5. Click the right arrow button in the right panel to add the selected index words type as a column. Add additional index words, if needed.
6. Open the flipper for the index words that should use the sort word parameter. On the Value element, click the **Transformation** button ( ).
7. On the Transformations dialog, for the Aspect parameter, select **Sort Word** from the dropdown. Click the **Save** button. The mapped column now displays Sort Word.
8. Complete the mapping and initiate the export.

Results of Index Words Linked to Products Using Sort Word

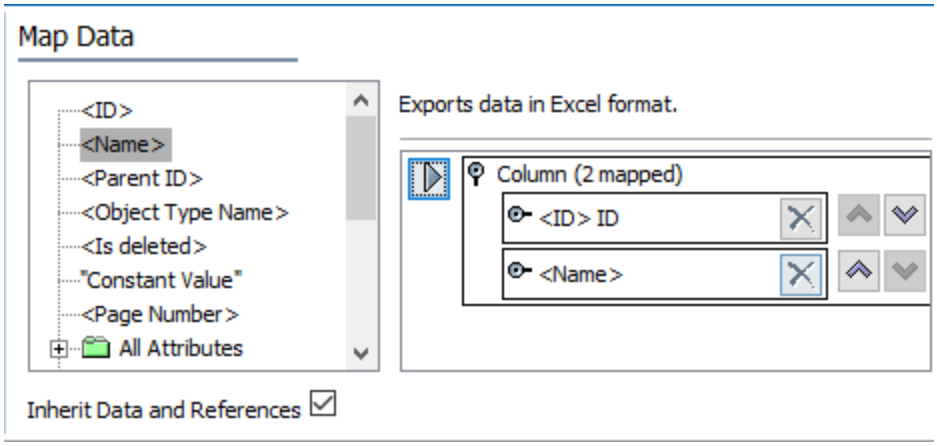
The output includes the ID and name of product, and index word ID as headers. Data includes the sort word attribute value defined on the index word that is linked to the product.

When the multiple index words are linked to a product, the sort words values are displayed in a semi-colon separated list.

	A	B	C
1	<ID>	<Name>	112612
2	121933	121933	10
3	20805	20805	15;80;50

Mapping Index Words from Classification

1. Select the data to be exported for the output tool. For more information, refer to the **Data Exchange** topic.
2. Select the Classification super type. For more information, refer to **Export Manager - Select Objects**, the **Configure the Format** section of the **OIEP - Event-Based - Output Templates Flipper**, or the **Configure the Format** section of the **OIEP - Select Objects - Output Templates Flipper**.
3. On the Map Data step, in the left panel, select any information necessary for the classification



4. Complete the mapping and initiate the export.

Results of Index Words from Classification

The output includes the mapped information of all index words in the selected hierarchy.

	A	B
1	<ID>	<Name>
2	Acme Baseball Cap	Acme Baseball Cap
3	Hanes Baseball Cap	Hanes Baseball Cap
4	Epuda quam	Epuda quam
5	harion	harion
6	polyester	polyester

Inherit Data and References - Data Source Outbound

The 'Inherit Data and References' option shown below is the default setting and is global, in that it applies to all attribute values or references selected in the Map Data window. When checked, the output includes values that are inherited from objects at a higher level in the hierarchy.

For general information on inheritance, refer to the **Product Hierarchy** topic in the **Getting Started** documentation.

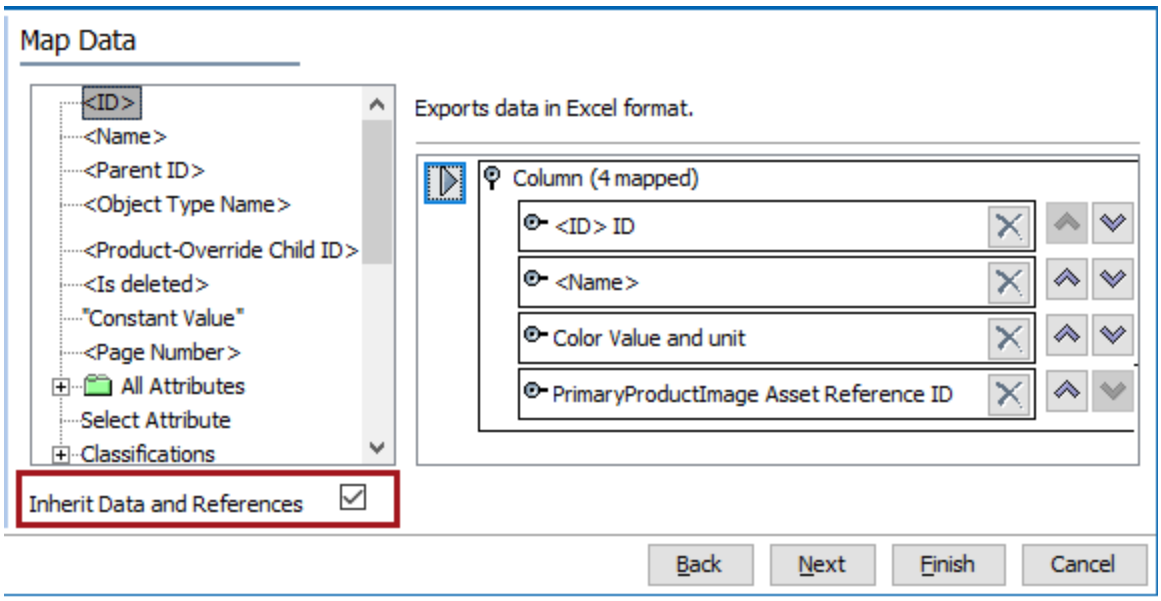
The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

For more information about the additional wizard steps, refer to the **Creating a Data Export** topic or the **Creating an Outbound Integration Endpoint** topic.

After mapping, most output options can be altered using transformations. For more information, refer to the **Outbound Map Data - Transform** topic.

Mapping Inherited Data and References

1. Select the data to be exported for the output tool. For more information, refer to the **Data Exchange** topic.
2. On the Map Data step, below the left panel, check the **Inherit Data and References** option appropriately to affect all attribute values or references mapped:
   - If **checked**, inherited data and references are included in the export.
   - If **unchecked**, inherited data and references is not included in the export.

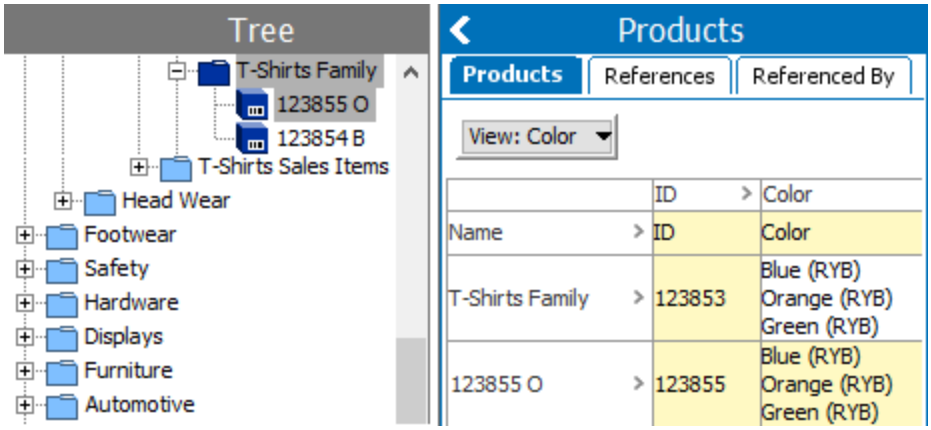


3. Use the necessary data sources to map the required data. For information, refer to **Outbound Map Data - Data Source**.
4. Initiate the export.

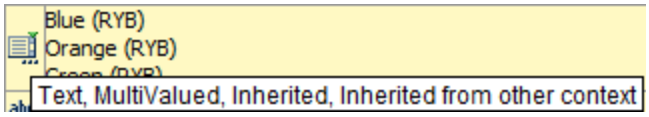
Results

The following data is exported with the results displayed below, based on the 'Inherit Data and References' parameter.

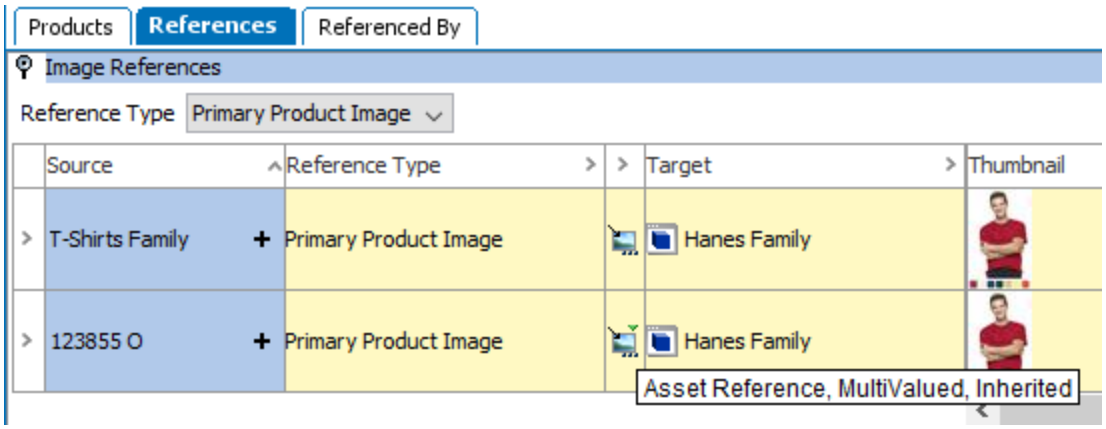
The Color attribute has a local value on the family object, and the values are inherited to the same attribute of the child object. The following shows a view created to show the values available for both the parent and the child.



Inheritance can be verified by the viewing the attribute on the editor for the child, and hovering over the icon on the attribute.



The 'Primary Product Image' reference value is also being inherited to the child level in the hierarchy.



When the 'Inherit Data and References' parameter is not checked, and the child object is exported, only the values and the references local on the child object are exported. In this case, all values were inherited, so no values are exported.

<ID>	<Name>	Color	Primary Product Image Asset Reference ID
123855	123855 O		

When the 'Inherit Data and References' parameter is checked, and the child object is exported, the values and the references are inherited to the child object.

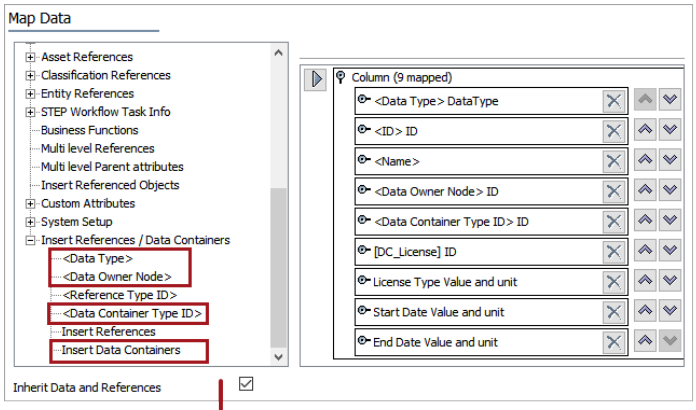
<ID>	<Name>	Color	Primary Product Image Asset Reference ID
123855	123855 O	Blue (RYB);Orange (RYB);Green (RYB)	20585

Insert Data Containers - Data Source Outbound

The 'Insert Data Containers' data source can export data container values and reference target IDs and names, that are owned by the exported object in a separate row. This option is available in the CSV and Excel formats and is effective when exporting product or entity objects, both of which can own data containers.

The following data sources are useful for understanding the relationships between the objects selected for output and in the output data containers and all should be mapped for export:

- Data Type
- Data Owner Node
- Data Container Type ID
- Insert Data Containers - the mapping in itself causes data containers of the selected types to be output in separate rows in the exported file and produces a column holding the ID of each data container object.



	A	B	C	D	E	F	G	H	I
1	<Data Type>	<ID>	<Name>	<Data Owner Node>	<Data Container Type ID>	<Data Container ID>	License Type	Start Date	End Date
2	NODE	380182	Crayola Colored Pens, 12 Count						
3	DATA_CONTAINER			380182	DC_License	380207	Trial	2022-01-01	2022-03-01
4	DATA_CONTAINER			380182	DC_License	380206	Fixed Duration	2022-03-01	2022-12-31
5	NODE	380183	Crayola Colored Pens, 24 Count						
6	DATA_CONTAINER			380183	DC_License	380214	Temporary	2021-03-26	2021-04-26

The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

For more information about the additional wizard steps, refer to the **Creating a Data Export** topic or the **Creating an Outbound Integration Endpoint** topic.

After mapping, most output options can be altered using transformations. For more information, refer to the **Outbound Map Data - Transform** topic.

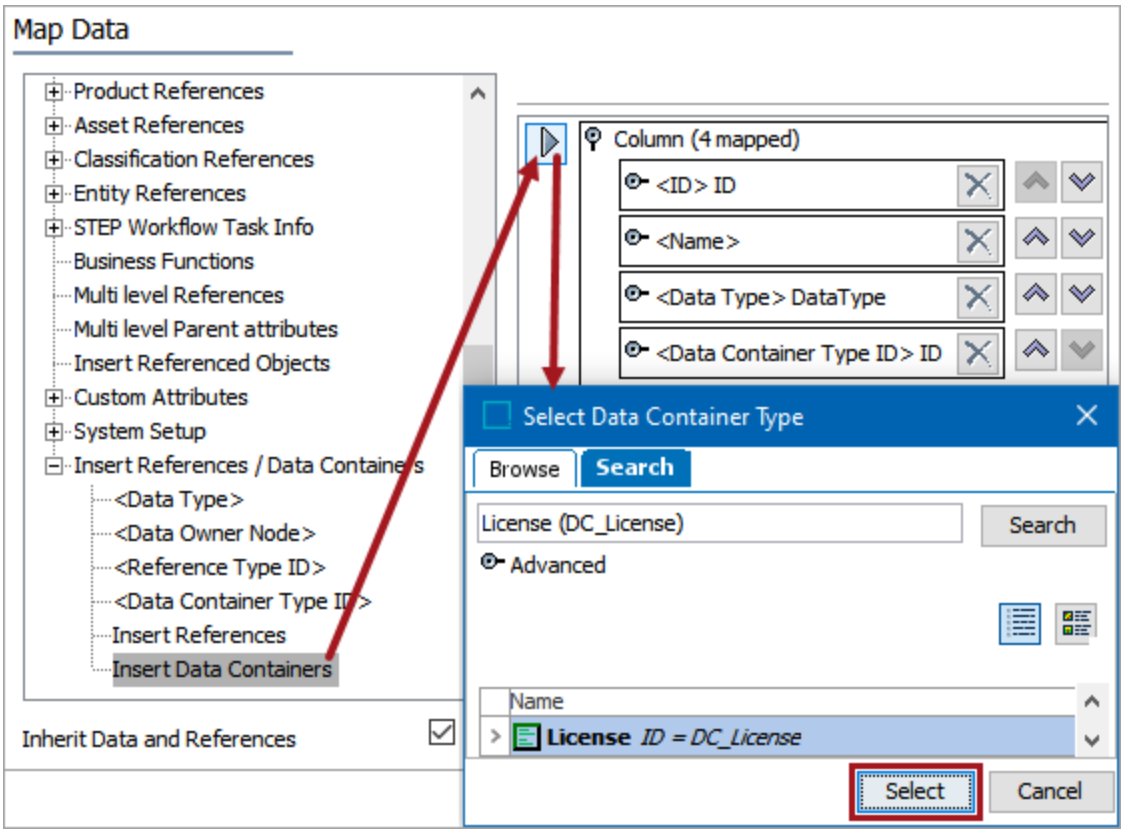
Mapping the Insert Data Containers data source

An example export is included in the **Results** section below.

1. Open the output tool and select the product or entity data to be exported.

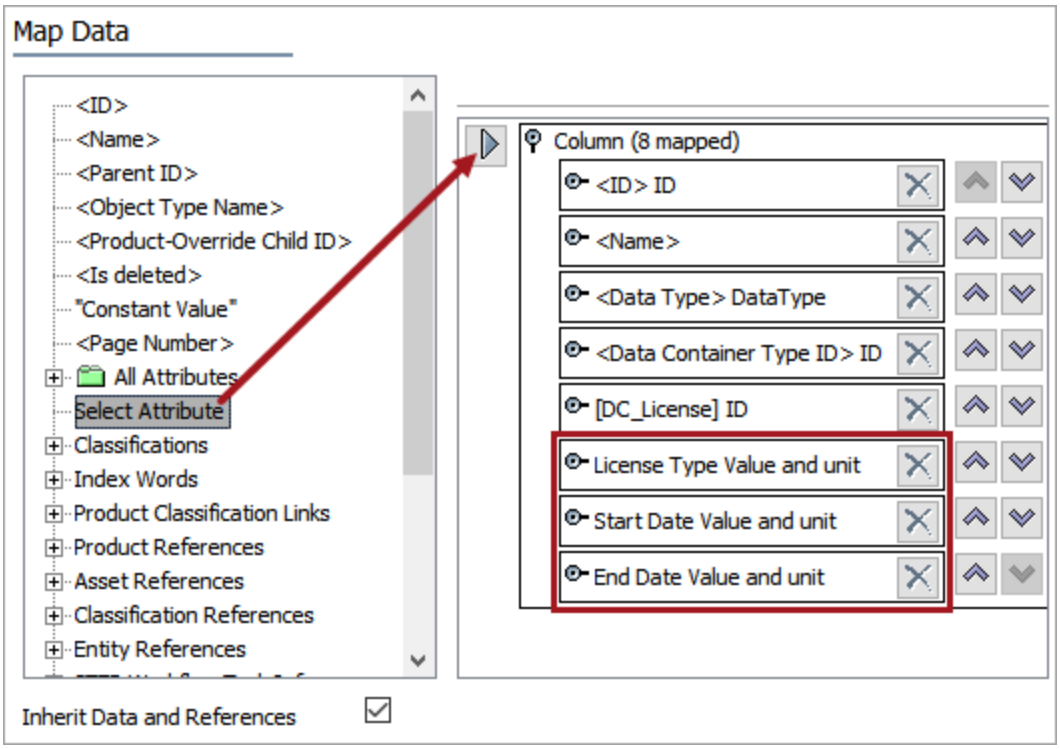
   For more information on the output tools, refer to the **Data Exchange** topic. For more information on data containers, refer to the **Data Containers** topic in the **System Setup** documentation.

2. On the Select Format step, select **Excel** or **CSV**.
3. On the Map Data step, in the left panel, expand the **Insert References / Data Containers** node.
4. Optionally, select the **<Data Type>** source and click the right arrow button ( ) in the right panel to supply the static NODE or DATA_CONTAINER text, which identifies the data on the row.
5. Select the **<Data Container Type ID>** source and click the right arrow button ( ) in the right panel to supply the ID of the selected data container.
6. Select the **Insert Data Containers** source, click the right arrow button ( ) in the right panel, select the desired data container type, and click the **Select** button.

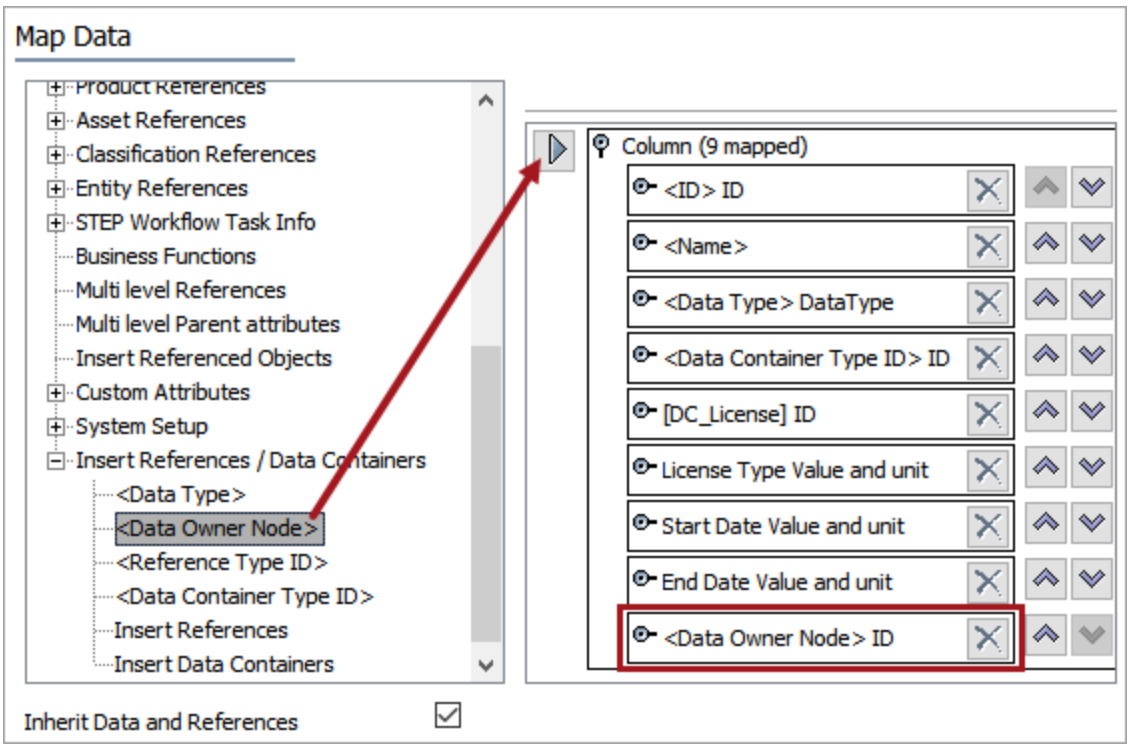


7. Optionally, map metadata attributes on the data container using an attribute data source option (defined in the **Attributes (and Data Containers) - Data Source Outbound** topic).

> **Note:** Mapped metadata attributes export values that exist on both the exported object(s) and the data container(s).



8. Optionally, map the **<Data Owner Node>** data source to clearly identify the relationship between the rows in the output file.

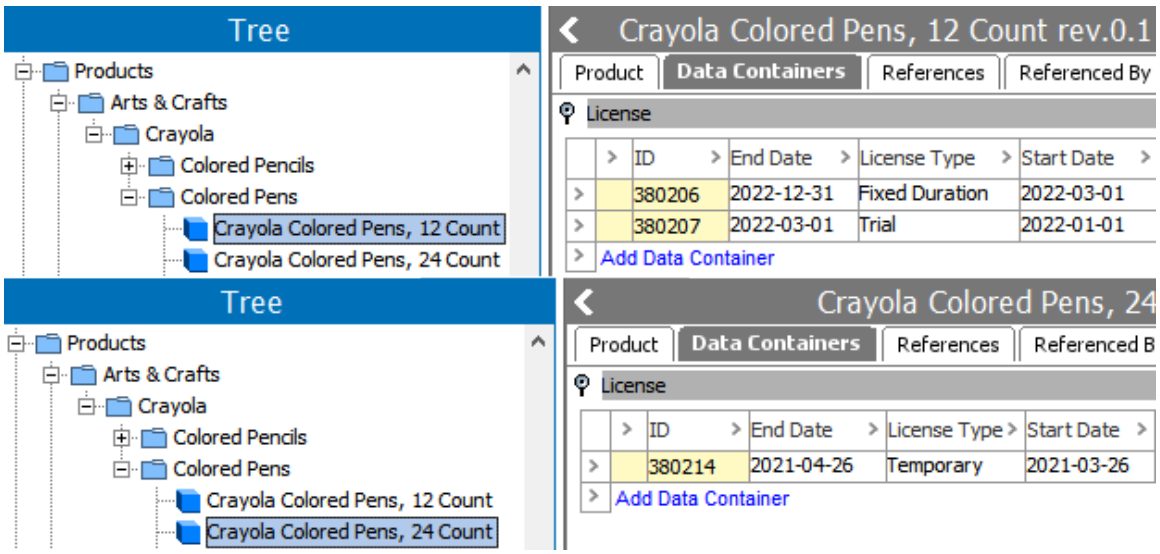


9. Check the **Inherit Data and References** option appropriately:
   - If **checked**, inherited data and references are included in the export.
   - If **unchecked**, inherited data and references are not included in the export.

   For information on inheritance, refer to the **Inherit Data and References - Data Source Outbound** topic.
10. Apply any transformations to change the output without changing the original data. For example, exporting the data container type name as a header instead of ID. For more information on transformations, refer to the **Aspect - Transform Outbound** topic.
11. Complete any additional mappings and initiate the export.

Results

In this example, the exported products have 'License (DC_License)' data containers:



The output file sorts data with the object selected for export first, and then any options mapped from the 'Insert References / Data Containers' section.

Note: When using the 'Insert References / Data Containers' data sources to create an export file that will be modified and re-imported, new data rows added to the import file must be added below the appropriate NODE row.

The Excel or CSV file includes the following information based on the mapping:

- The <ID> and <Name> columns identify the exported object.
- The <Data Type> column uses the static text 'NODE' to identify the exported product or entity. The static text 'DATA _CONTAINER' identifies the type of data on the row.
- The <Data Container Type ID> column identifies the ID of the selected data container owned by the exported object.
- The <Data Container ID> column identifies the ID of the data container identified in the <Data Container Type ID> column.
- The 'License Type', 'Start Date', and 'End Date' columns hold the values of the metadata attributes.
- The <Data Owner Node> column identifies the ID of the object that owns the data on the row.

	A	B	C	D	E	F	G	H	I
1	<ID>	<Name>	<Data Type>	<Data Container Type ID>	<Data Container ID>	License Type	Start Date	End Date	<Data Owner Node>
2	380182	Crayola Colored Pens, 12 Count	NODE						
3			DATA_CONTAINER	DC_License	380207	Trial	2022-01-01	2022-03-01	380182
4			DATA_CONTAINER	DC_License	380206	Fixed Duration	2022-03-01	2022-12-31	380182
5	380183	Crayola Colored Pens, 24 Count	NODE						
6			DATA_CONTAINER	DC_License	380214	Temporary	2021-03-26	2021-04-26	380183

Insert Referenced Objects - Data Source Outbound

The 'Insert Referenced Objects' data source exports all objects referenced from the selected reference type before the object that is selected to be exported. The column added by this mapping contains the ID of the owner of the reference to display the objects belong together.

This option can be used to export objects referenced from asset references, product references, classification references, and entity references. All the reference types from the System Setup > Reference Types except Link Types are available for export with this option.

The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

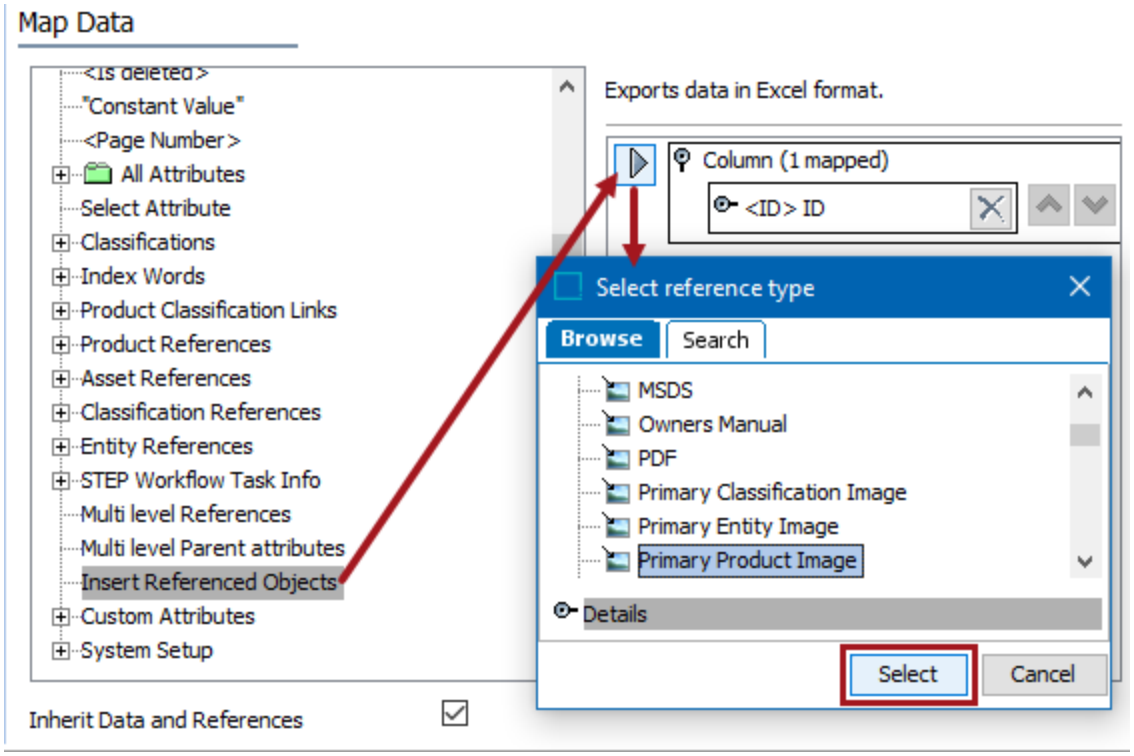
For more information about the additional wizard steps, refer to the **Creating a Data Export** topic or the **Creating an Outbound Integration Endpoint** topic.

After mapping, most output options can be altered using transformations. For more information, refer to the **Outbound Map Data - Transform** topic.

Mapping Insert Referenced Object

Examples of a variety of exports are included in the **Results** section below.

1. Select the data to be exported for the output tool. For more information, refer to the **Data Exchange** topic.
2. On the Map Data step, in the left panel, select **Insert Referenced Object**, click the right arrow button () in the right panel, select the desired reference type, and click the **Select** button.

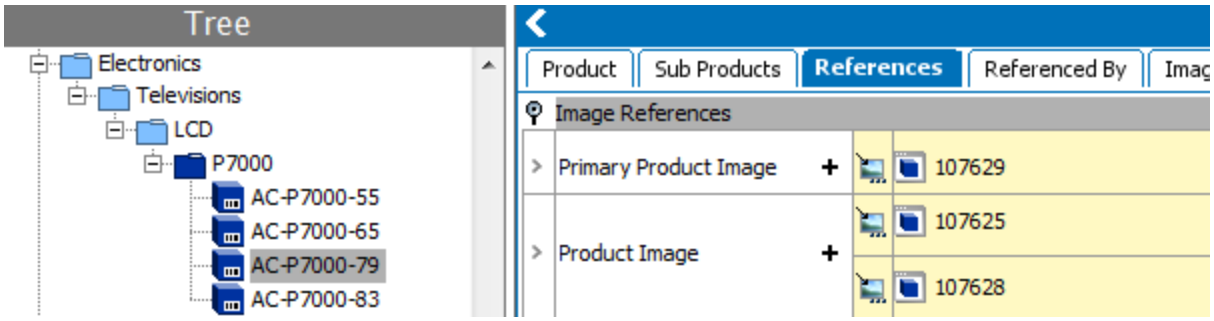


3. Check the **Inherit Data and References** option appropriately:
   - If **checked**, inherited data and references are included in the export.
   - If **unchecked**, inherited data and references is not included in the export.

   For information on inheritance, refer to the **Inherit Data and References - Data Source Outbound** topic.
4. Apply any transformations, which can change the output without changing the original data. For example, exporting the asset reference type name as a header and asset name as the value. Refer to the **Aspect - Transform Outbound** topic. Example outputs are included in the **Results** section below.
5. Complete any additional mappings and initiate the export.

Results

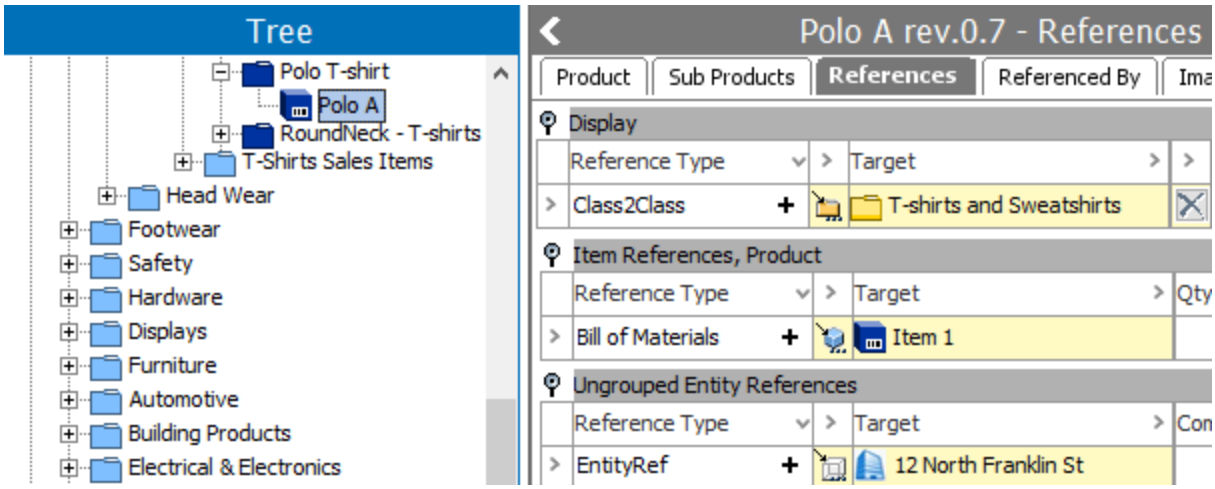
In this example of Asset References, the product being exported has three references to images:



Using the 'Insert Referenced Objects' option, the export shows the product ID once under the <ID> column and again for each of the two Ref Owner ID columns.

	A	B	C	D
1	<ID>	<Name>	Ref Owner ID of PrimaryProductImage	Ref Owner ID of ProductImage
2	107628	107628	179626	179626
3	107625	107625	179626	179626
4	107629	107629	179626	179626
5	179626	AC-P7000-79	179626	179626

In this example, the product being exported has a Classification Reference of type 'Class2Class', a Product Reference of type 'Bill of Materials', and an Entity Reference of type 'EntityRef.'



Using the 'Insert Referenced Objects' option, the export shows the product ID once under the <ID> column and again for each of the three Ref Owner ID columns.

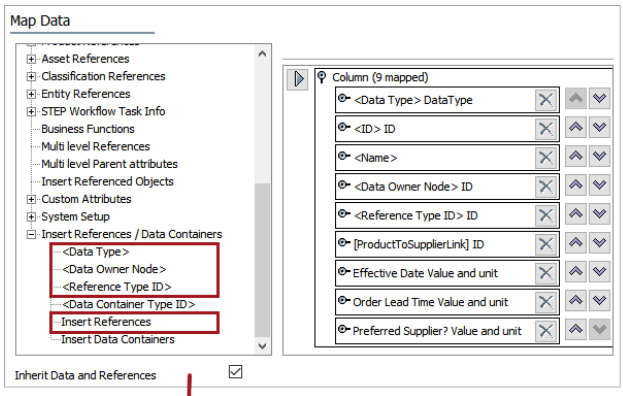
	A	B	C	D	E
1	<ID>	<Name>	Ref Owner ID of ClassificationReference	Ref Owner ID of BillOfMaterials	Ref Owner ID of EntityRef
2	18212	Item 1	113202	113202	113202
3	121605	12 North Franklin St	113202	113202	113202
4	22586	T-shirts and Sweatshirts	113202	113202	113202
5	113202	Polo A	113202	113202	113202

Insert References - Data Source Outbound

The 'Insert References' data source can export target IDs, names, keys, and metadata on the reference owned by the exported object in a separate row. This option is available on the CSV and Excel formats when exporting products, entities, classifications, and assets. All reference types from the System Setup 'Reference Types' node are available for export, including product to classification link types.

The following data sources are useful for understanding the relationships between the data selected for output and the owned references and all should be mapped for export:

- Data Type
- Data Owner Node
- Reference Type ID
- Insert References - the mapping in itself causes references and classification product links of the selected types to be output in separate rows in the exported file and produces a column holding the identifier for each reference or link (target ID, name or key).



	A	B	C	D	E	F	G	H	I
1	<Data Type>	<ID>	<Name>	<Data Owner Node>	<Reference Type ID>	<Reference Target ID>	Effective Date	Order Lead Time	Preferred Supplier?
2	NODE	380182	Crayola Colored Pens, 12 Count				2012-12-12		
3	REFERENCE			380182	ProductToSupplierLink	Supplier_AllGoods	2022-01-01	4 weeks	No
4	REFERENCE			380182	ProductToSupplierLink	AcmeSupplier	2022-01-01	2 weeks	Yes
5	NODE	380183	Crayola Colored Pens, 24 Count						
6	REFERENCE			380183	ProductToSupplierLink	Supplier_AllGoods	2022-01-01	2 weeks	Yes
7	REFERENCE			380183	ProductToSupplierLink	Supplier_AtoZ	2022-06-01	4 weeks	No

Although the 'Multi Level References' data source (defined in the **Multi Level References - Data Source Outbound** topic) gives access to this same data, the 'Multi Level References' may be less effective because it concatenates values for multiple instances of the same data container type into a single column, which can make the relationships between the data difficult to decipher.

The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

For more information about the additional wizard steps, refer to the **Creating a Data Export** topic or the **Creating an Outbound Integration Endpoint** topic.

After mapping, most output options can be altered using transformations. For more information, refer to the **Outbound Map Data - Transform** topic.

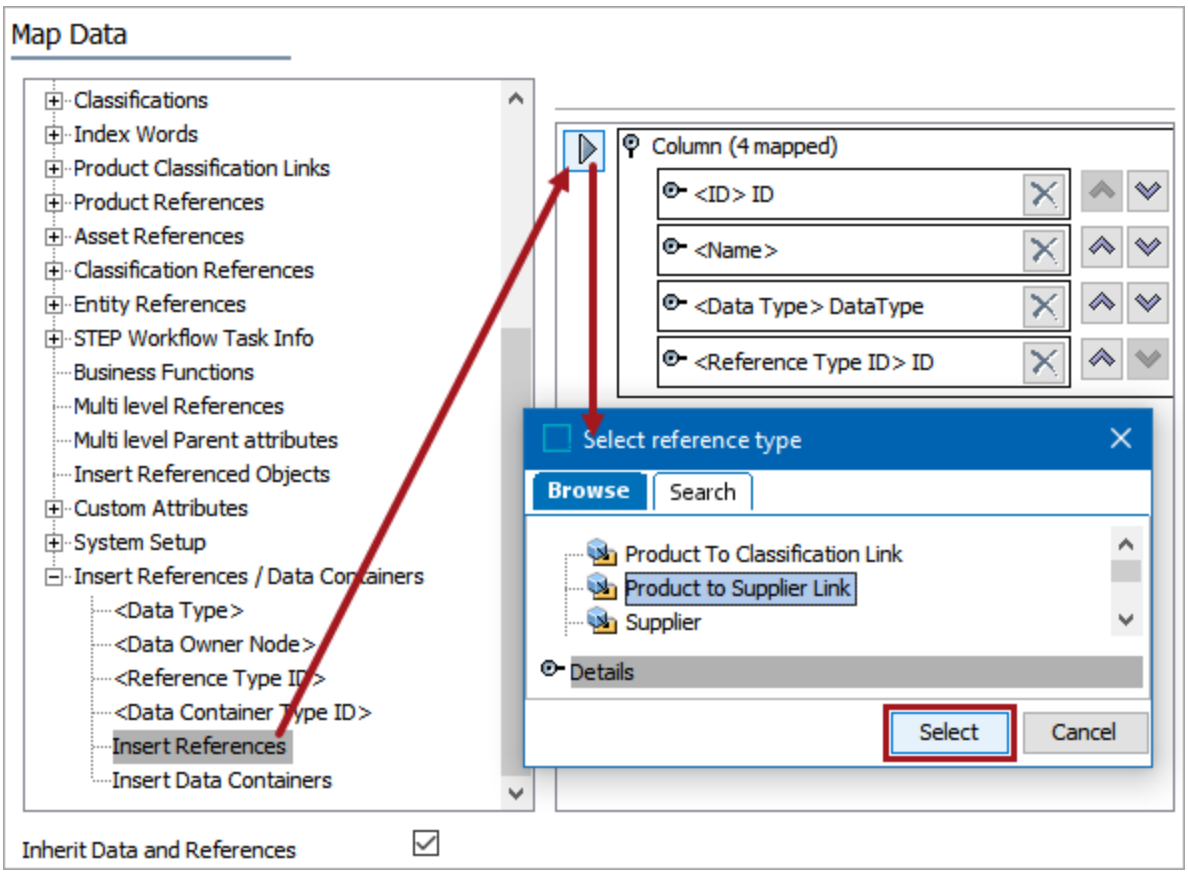
Mapping the Insert References data source

An example export is included in the **Results** section below.

1. Open the output tool and select the product, entity, classification, or asset data to be exported.

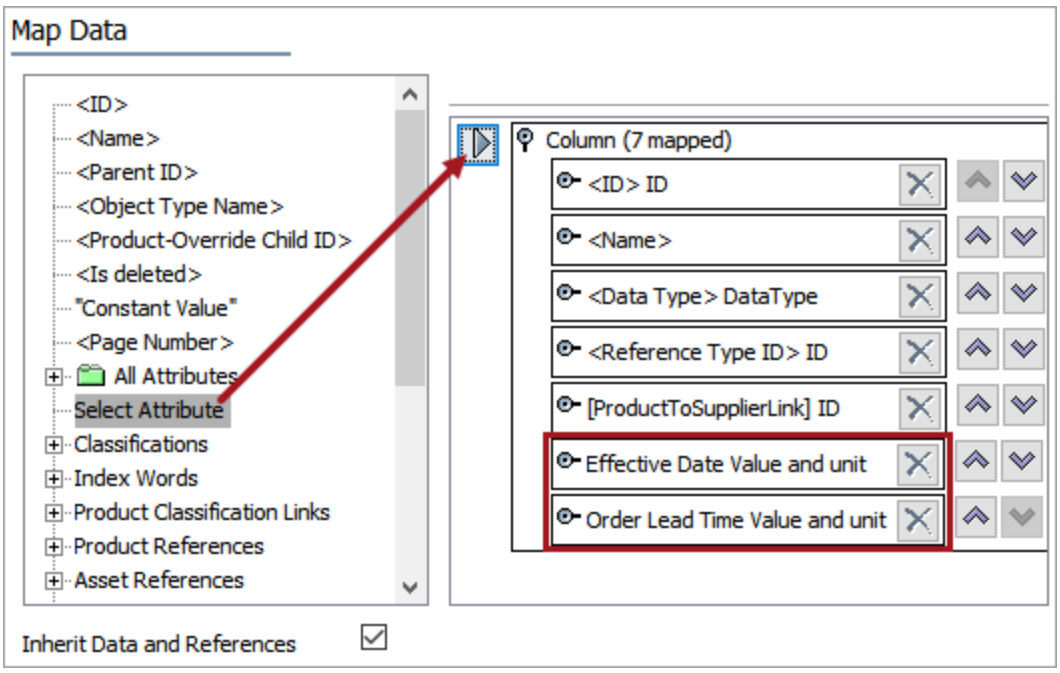
   For more information on output tools, refer to the **Data Exchange** topic.

2. On the Select Format step, select **Excel** or **CSV**.
3. On the Map Data step, in the left panel, expand the **Insert References / Data Containers** node.
4. Select the **<Data Type>** source and click the right arrow button ( ) in the right panel to supply the static NODE or REFERENCE text, which identifies the data on the row.
5. Select the **<Reference Type ID>** source and click the right arrow button ( ) in the right panel to supply the ID of the selected reference.
6. Select the **Insert References** source, click the right arrow button ( ) in the right panel, select the desired reference or product to classification link type, and click the **Select** button.

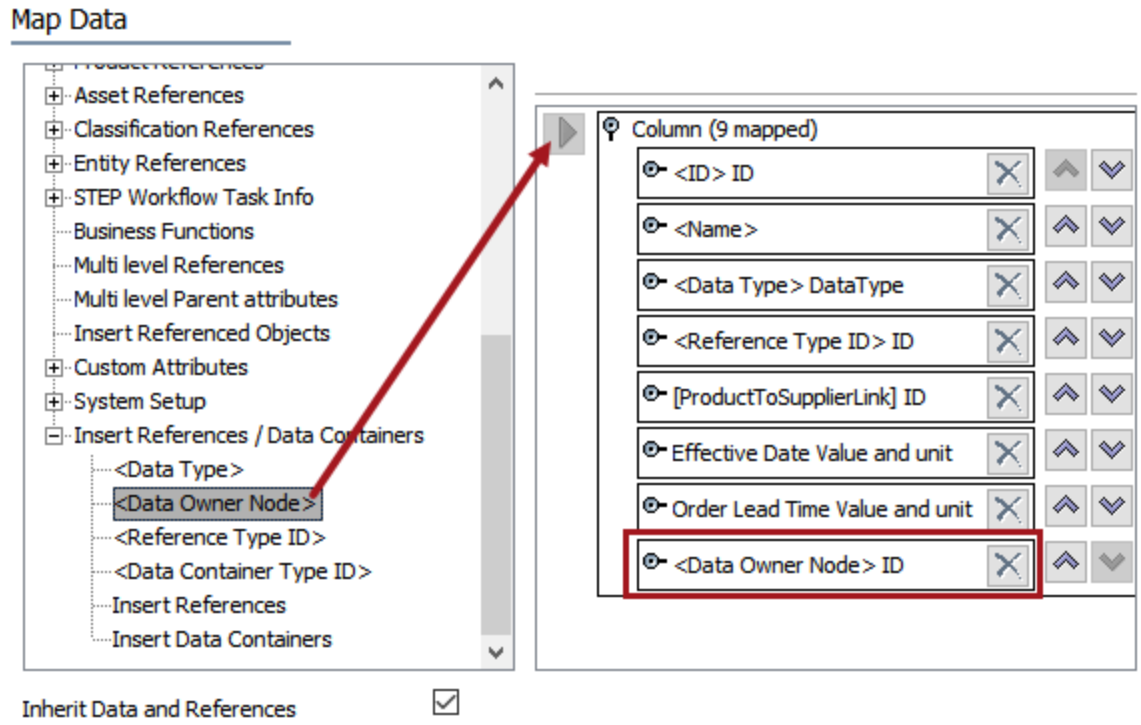


7. Optionally, map metadata attributes on the reference using an attribute data source option (defined in the **Attributes (and Data Containers) - Data Source Outbound** topic).

> **Note:** Mapped metadata attributes export values that exist on both the exported object(s) and the reference(s).



8. Optionally, map the **<Data Owner Node>** data source to clearly identify the relationship between the rows in the output file.



9. Check the **Inherit Data and References** option appropriately:

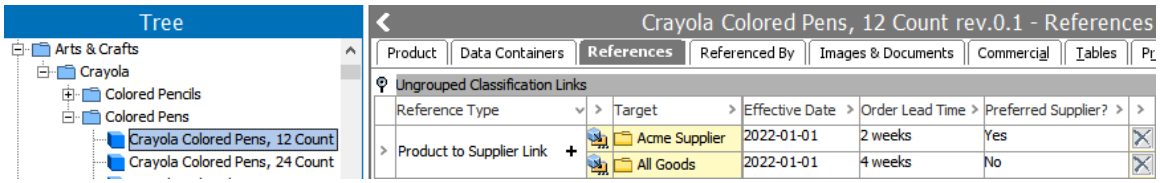
- If **checked**, inherited data and references are included in the export.
- If **unchecked**, inherited data and references are not included in the export.

For information on inheritance, refer to the **Inherit Data and References - Data Source Outbound** topic.

10. Apply any transformations to change the output without changing the original data. For example, exporting the reference type name as a header and reference name as the value. For more information on transformations, refer to the **Aspect - Transform Outbound** topic.
11. Complete any additional mappings and initiate the export.

Results

In this example, the exported product has values for the 'Product to Supplier Link (ProductToSupplierLink)' product to classification link type:



The output file sorts data with the object selected for export first, and then any options mapped from the 'Insert References / Data Containers' section.

Note: When using the 'Insert References / Data Containers' data sources to create an export file that will be modified and re-imported, new data rows added to the import file must be added below the appropriate NODE row.

The Excel or CSV file includes the following information based on the mapping:

- The <ID> and <Name> columns identify the exported object.
- The <Data Type> column uses the static text 'NODE' to identify the exported product, entity, classification, or asset. The static text 'REFERENCE' identifies the type of data on the row.
- The <Reference Type ID> column identifies the ID of the selected reference or product to classification link type owned by the exported object.
- The <Reference Target ID> column identifies the ID of the reference or link identified in the <Reference Type ID> column.
- The 'Effective Date' and 'Order Lead Time' columns hold the values of the metadata attributes.
- The <Data Owner Node> column identifies the ID of the object that owns the data on the row.

	A	B	C	D	E	F	G	H
1	<ID>	<Name>	<Data Type>	<Reference Type ID>	<Reference Target ID>	Effective Date	Order Lead Time	<Data Owner Node>
2	380182	Crayola Colored Pens, 12 Count	NODE			2012-12-12		
3			REFERENCE	ProductToSupplierLink	Supplier_AllGoods	2022-01-01	4 weeks	380182
4			REFERENCE	ProductToSupplierLink	AcmeSupplier	2022-01-01	2 weeks	380182

Is Deleted - Data Source Outbound

Although displayed for all formats, the '<Is deleted>' data source can be used by Generic XML in OIEPs as described in the following mapping examples:

- Reporting Deleted Assets with a Generic XML OIEP
- Reporting Deleted Classifications with a Generic XML OIEP
- Reporting Deleted Entities with a Generic XML OIEP
- Reporting Deleted Product References with a Generic XML OIEP

For additional 'is deleted' functionality, refer to the 'Is Deleted' option available in the **Aspect - Transform Outbound** topic.

The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

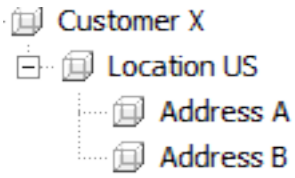
For more information about the additional wizard steps, refer to the **Creating a Data Export** topic or the **Creating an Outbound Integration Endpoint** topic.

After mapping, most output options can be altered using transformations. For more information, refer to the **Outbound Map Data - Transform** topic.

Multi Level Parent Attributes - Data Source Outbound

The Multi Level Parent Attributes option enables the exporting of data on parent objects as a separate column for each object being exported. This option is available for exporting parent attribute values for product, entity, and classification objects.

For example, a customer hierarchy is created in STEP using entity nodes. Addresses are grouped into locations.



When this customer hierarchy is exported, you can export attribute values on parent nodes. This means that the row that holds Address A and Address B could also include values that only exist on the Location US or Customer X entity objects.

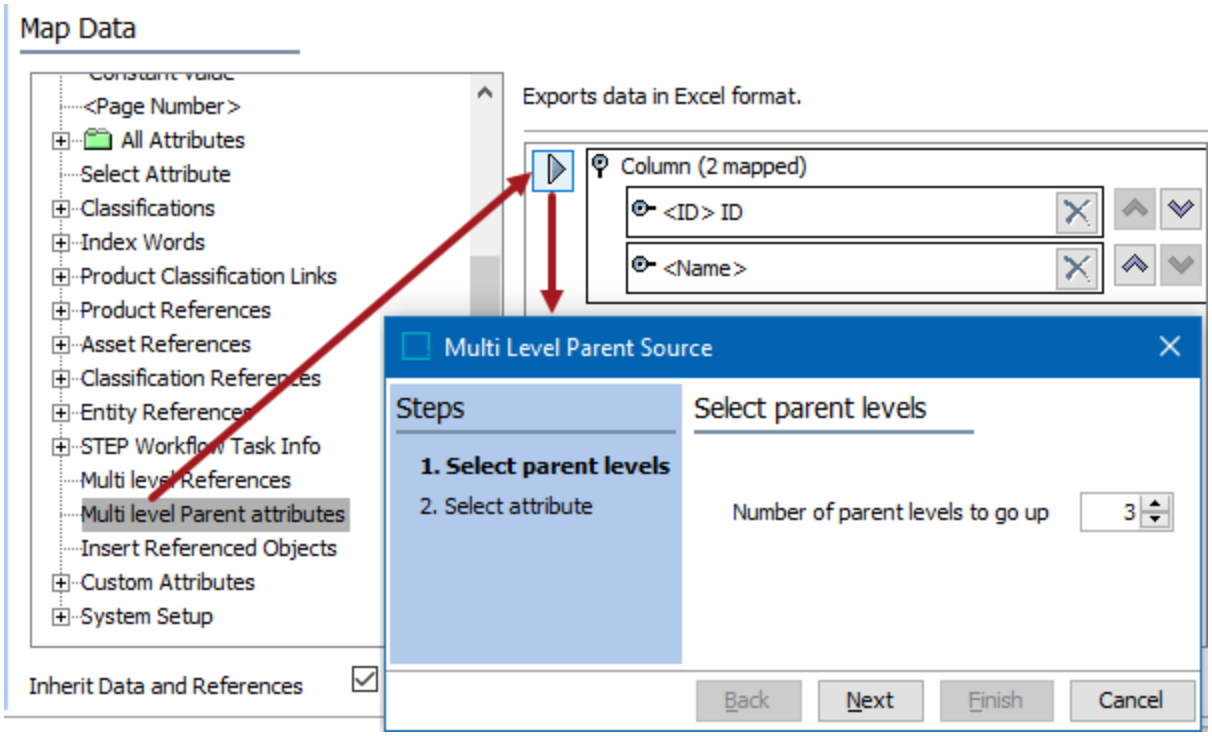
The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

For more information about the additional wizard steps, refer to the **Creating a Data Export** topic or the **Creating an Outbound Integration Endpoint** topic.

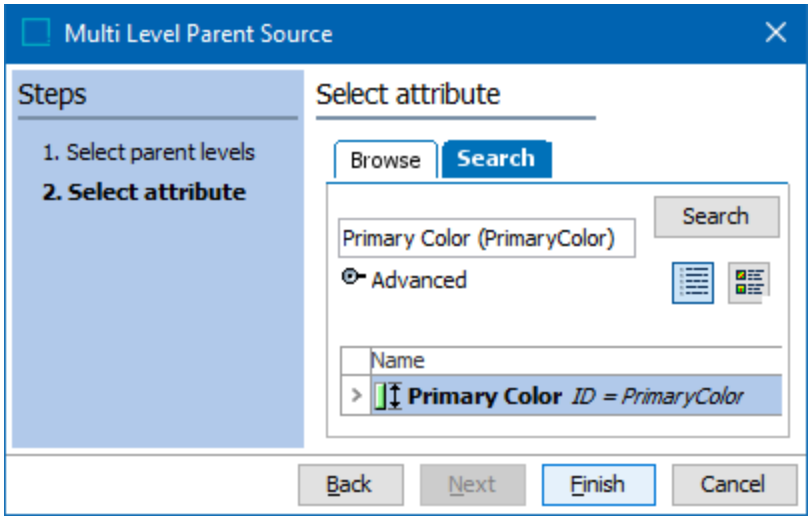
After mapping, most output options can be altered using transformations. For more information, refer to the **Outbound Map Data - Transform** topic.

Mapping Multi Level Parent Attributes

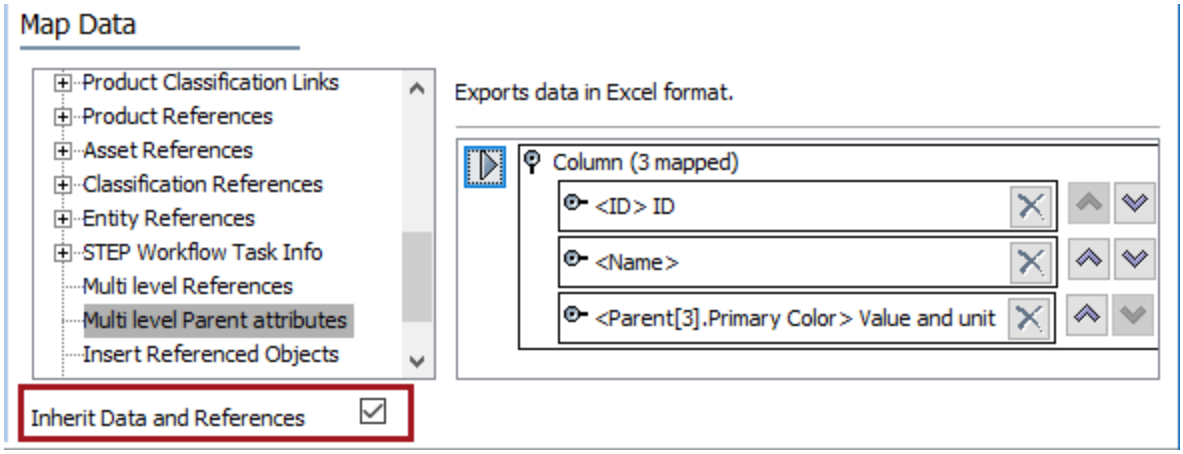
1. Select the data to be exported for the output tool. For more information, refer to the **Data Exchange** topic.
2. On the Map Data step, in the left panel, select the **Multi level Parent attributes** option, and in the right panel, click the arrow mapping button ( ) to display the Multi Level Parent Source dialog.



3. For the **Select parent levels** step, specify the number of parent levels to go up within the hierarchy, and click **Next**. The default of '1' indicates that only the value from the immediate parent is exported. Increasing this setting exports the value for the parent at the specified level, as shown above with the setting of '3.'
4. For the **Select attribute** step, use **Browse** or **Search** to locate the attribute to be exported, and click **Finish**.

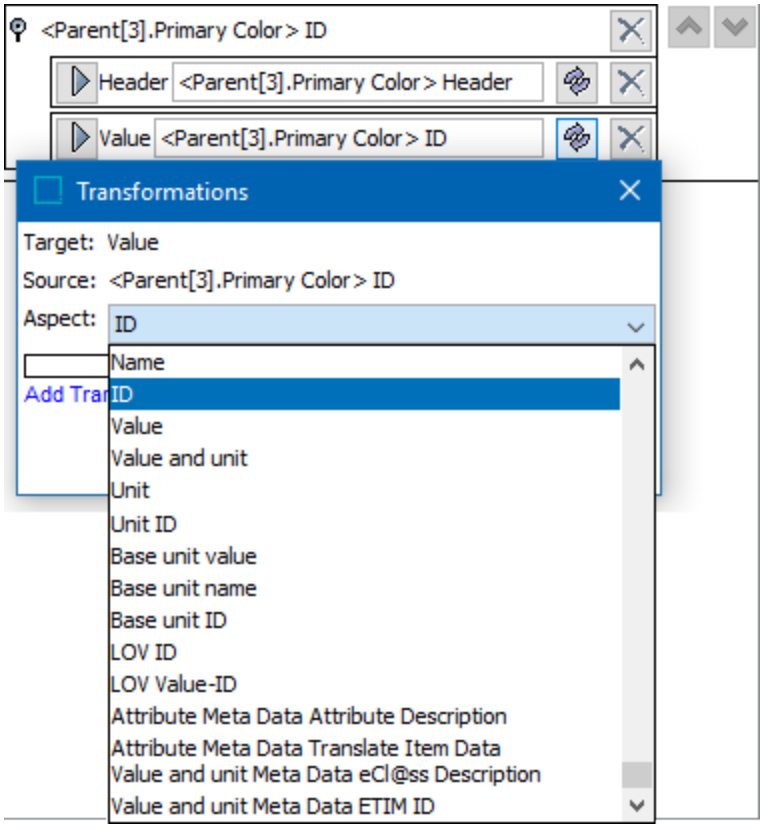


5. Back on the Map Data dialog, check the **Inherit Data and References** option appropriately:
   - If **checked**, inherited data and references are included in the export.
   - If **unchecked**, inherited data and references is not included in the export.



For information on inheritance, refer to the **Inherit Data and References - Data Source Outbound** topic.

6. If needed, change the asset reference Value Aspect to output data other than the asset reference ID. For details on the options, refer to the **Aspect - Transform Outbound** topic.



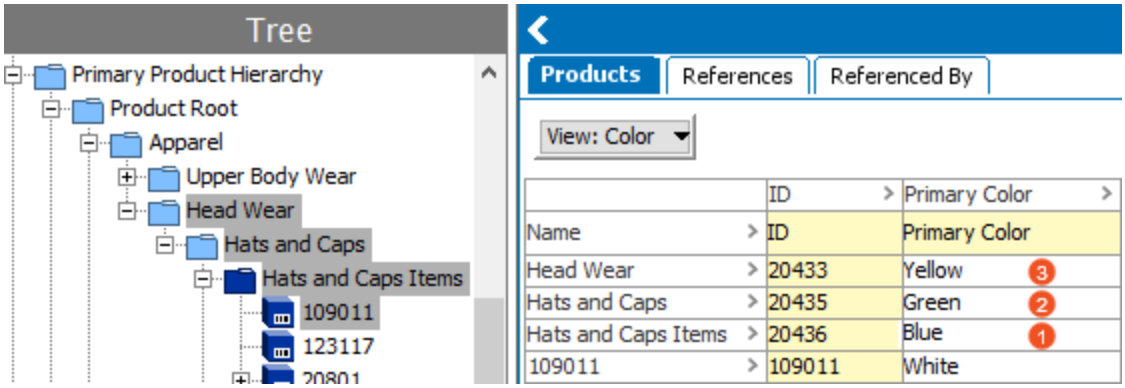
- Open the asset reference flipper to display the Header and Value elements.
- Click the transformation button ( ) for the Header or for the Value element to display the Transformations dialog and the Aspects parameter.

- Click the **Aspect** dropdown to display and select an option.
- Click the **Save** button to apply any aspect changes.

7. Apply any transformations necessary to change the output without changing the original data. Refer to the **Outbound Map Data - Transform** topic for details.
8. Complete any additional mappings and initiate the export.

Results

The 'Primary Color' attribute holds different data for each of the selected levels in the hierarchy, as displayed in the view named 'Color.' The three parent levels are indicated by the white numbers in the orange circles.



When exported for a multi-level parent setting of '3', the following is output, which includes the value of the exported object, as well as the third parent level up the hierarchy.

<ID>	<Name>	<Parent[3].Primary Color>
109011	109011	Yellow

Multi Level References - Data Source Outbound

The multi-level references data source option allows you to map data from referenced or ancestor objects. You can also map so that one or more additional rows are generated for referenced objects. This option is available and mapped the same for product, classification, asset, and entity objects.

Note: The multi level references data source is available for most formats that require mapping. This option is not supported for the Excel Smartsheet format. This option is also not supported in the STEPXML format since mapping is not available.

For example, a customer hierarchy is created in STEP using entity nodes. Each customer has a reference of the type Region to a Location. Each Location has a reference of type Address to specific addresses. When the customer hierarchy is exported, it is possible to follow the references and export data from the address objects.

Using the Multi Level References option allows the following information from the object to be exported:

- Names of referenced objects.
- Attribute values on referenced objects.
- Meta data attributes on references with values.

The output of the 'Insert References' data source (defined in the **Insert References -** Data **Source Outbound** topic) gives access to the same data as the Multi Level Reference data source, but 'Insert References' writes each object on an individual row to produce a simplistic layout with clear relationships between the objects.

The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

For more information about the additional wizard steps, refer to the **Creating a Data Export** topic or the **Creating an Outbound Integration Endpoint** topic.

After mapping, most output options can be altered using transformations. For more information, refer to the **Outbound Map Data - Transform** topic.

When multiple references exist, the options for the output are:

- All references in a single field, separated - This is the default setting for the 'Multi Valued' option.
- One reference per field - Use the 'Item [ ]' option as a separate mapping to output each reference in an individual field.

Examples

The following topics include examples using this data source option with:

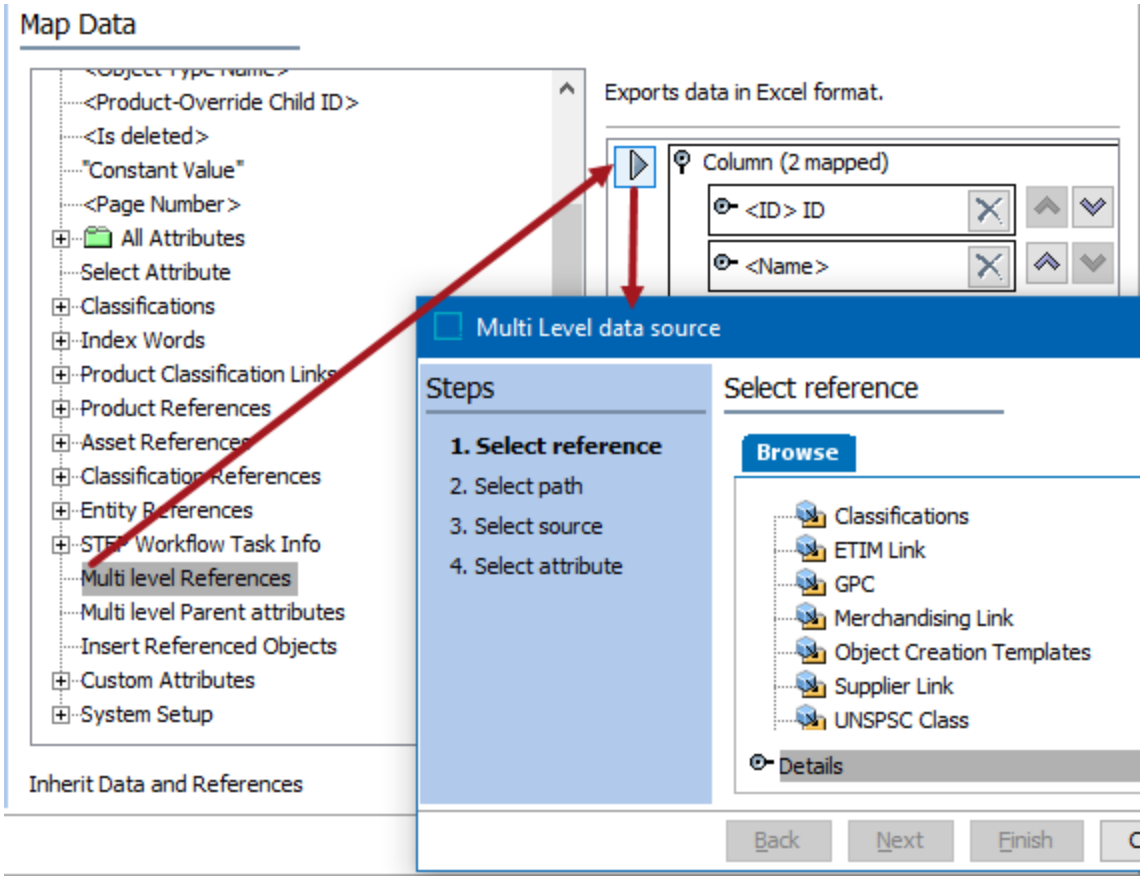
- Single value reference types
- Multi-value reference types
- Entities
- Products

Mapping Multi Level References

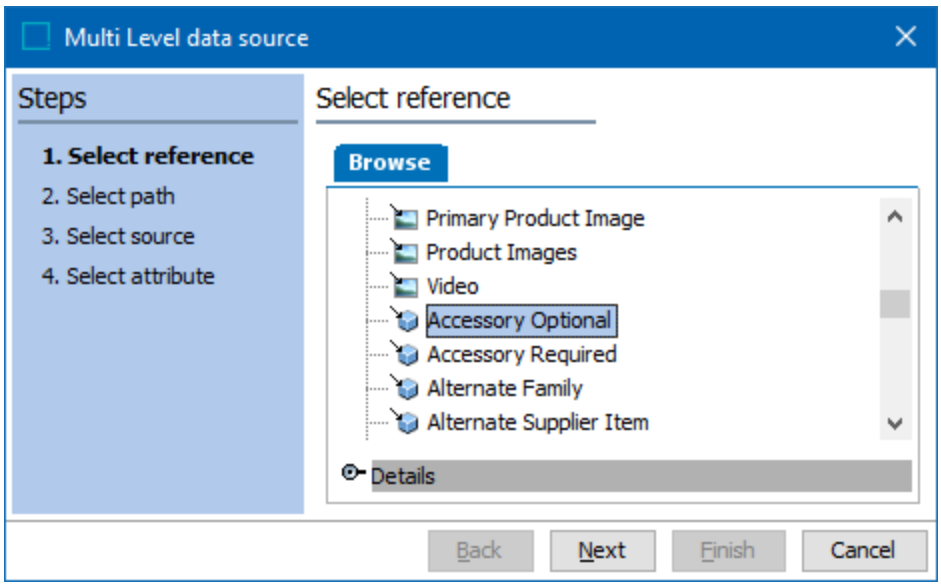
1. Select the data to be exported for the outbound tool. For more information, refer to the **Data Exchange** topic.

   Note: Export Manager includes the ability to restrict the output based on several options regardless of mapping. For more information, refer to the **Export Manager - Select Objects** topic.

2. On the Map Data step, in the left panel, select **Multi level References** data source.
3. In the right panel, click the right arrow icon to display the **Multi Level data source** wizard.

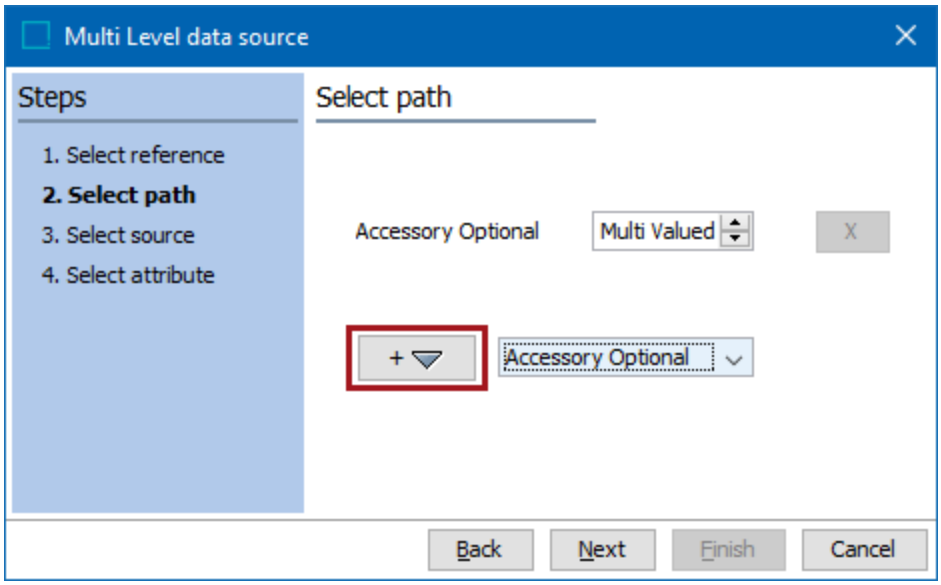


4. In **Select reference**, highlight the relevant source reference type and click **Next**.



5. In **Select path**, the legal reference types available are displayed in the dropdown. Select the required reference type.

   To add an additional reference type, click the 'plus down arrow' button, .

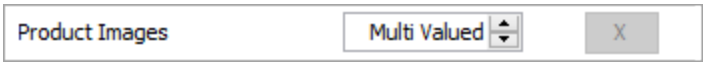


6. Select the necessary reference levels and click **Next**.

   Based on selected reference, the following options may be available:

   - **Single valued** - exports data from the referenced object of the selected type where only one is allowed.

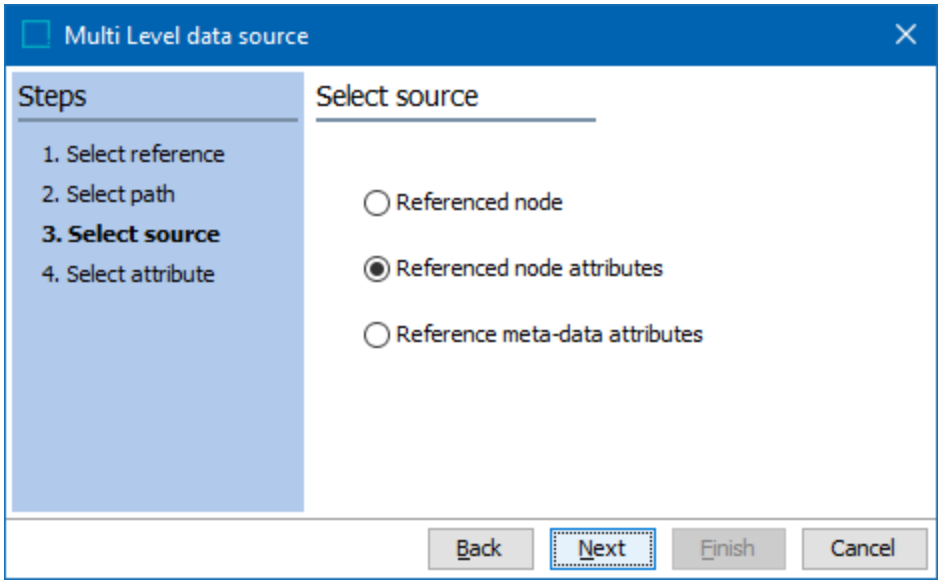
- **Multivalued** - exports data from all referenced objects of the selected type where multiple references are allowed. All reference values are output in a single field, with a separator character.



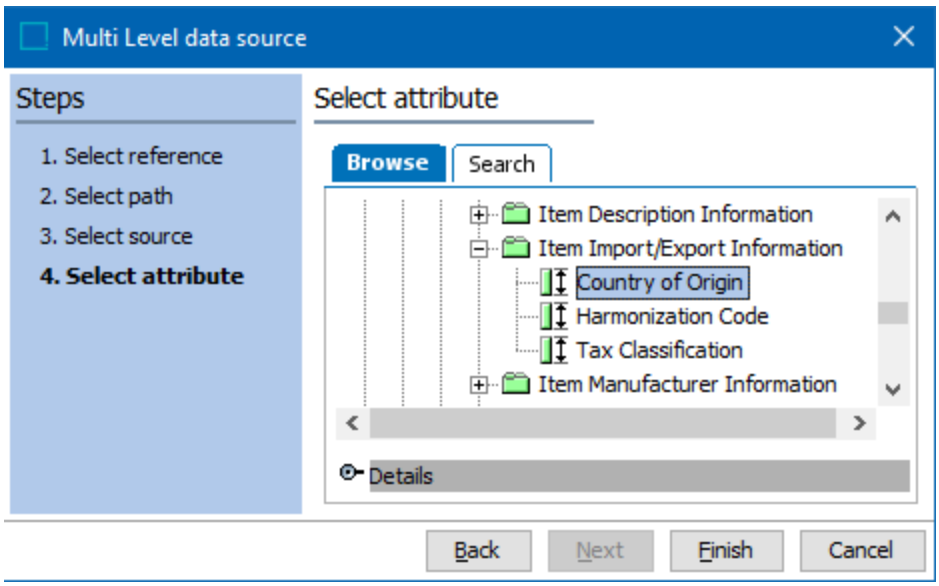
- **Item [ ]** - exports data from only a single referenced object, determined by the selected Item number. For example, Item [1] only exports data from the first referenced object.

- **Item [ ] with multiple references** - use the previous 'Item [ ]' option to create a separate mapping for each of the available references. Each mapping will show a unique number within the brackets, for example: Item [1], Item [2], Item [3] will produce information for three references, each in a column. For details, refer to the **Multiple Reference Metadata in Separate Fields** section of the **Multi Level References - Examples with Multi-Value Reference Types** topic.
- **X** - Click the delete button to remove any unnecessary levels.

7. In **Select Source**, choose the type of data to be exported:
   - **Referenced node** exports the names of the objects. The Select Attribute screen is disabled.
   - **Referenced node attributes** exports attribute values on the referenced objects for the selected attribute. Click **Next** to display the **Select Attribute** screen.
   - **Reference meta-data attributes** exports attribute values applied to the reference. Click **Finish**.

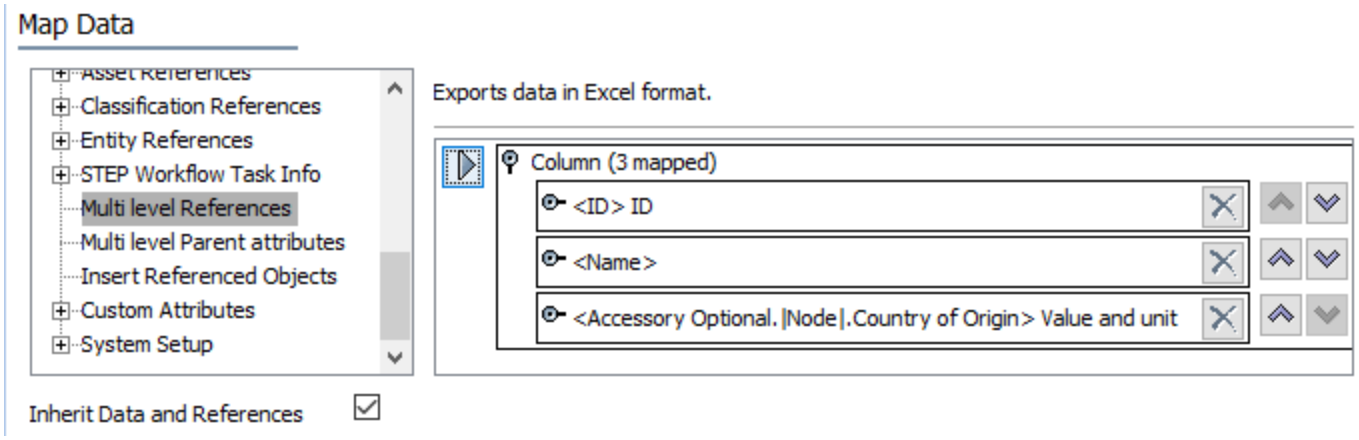


8. If needed, in **Select Attribute**, use **Browse** or **Search** to locate the attribute that holds the values to be exported and click **Finish**.



The multi-level reference mapping is complete and displayed on the Map Data dialog.

9. Complete any additional mappings and initiate the export.



The single object output includes the values for both of the Accessory Optional references. Although this example used the Excel format, any other format would produce the same output. In the following image, '.|Node|.' is the referenced product of the exported product. This is included since on the **Select source** step, 'Referenced node attributes' was selected and then for the **Select attribute** step, 'Country of Origin' was selected.

	A	B	C
1	<ID>	<Name>	<Accessory Optional.|Node|.Country of Origin>
2	20805	Baseball Cap	SPAIN;CHINA

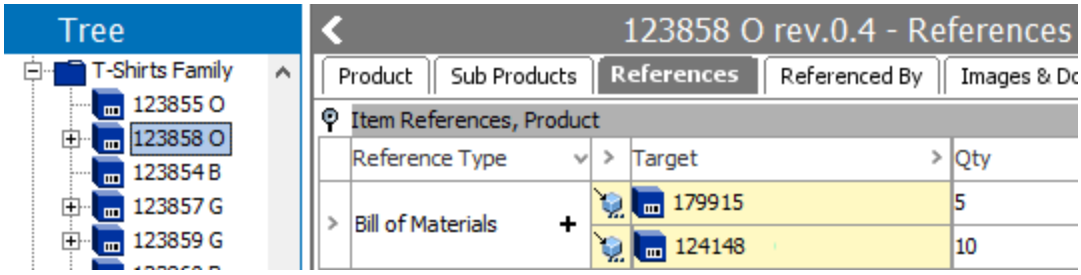
Multi Level References - Examples with Multi-Value Reference Types

General information about this data source and mapping multi-level references is in the **Multi Level References - Data Source Outbound** topic.

Note: The multi level references data source is available for most formats that require mapping. This option is not supported for the Excel Smartsheet format. This option is also not supported in the STEPXML format since mapping is not available.

For each of the following examples, the same product is exported with ID and name, and then includes a different source or path option in the 'Multi Level data source' wizard. The output results are included to illustrate the differences.

Export a product with references to a multi-valued reference type. The product shown below has multiple references via the 'Bill of Materials' reference type, and includes a value for the metadata attribute 'Qty' on the reference.

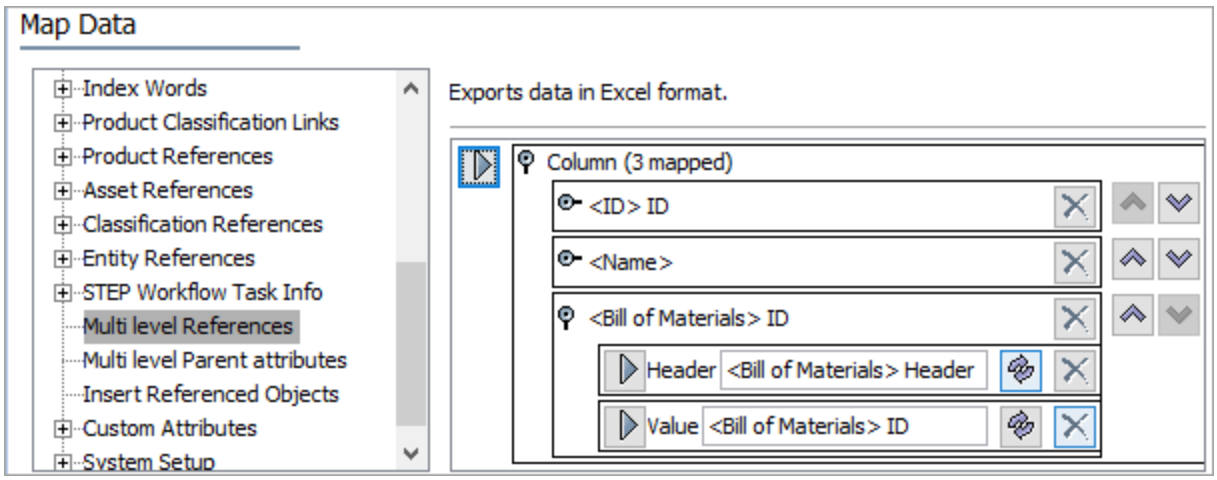


Multiple Reference Node in the Same Field

When mapping the **Multi Level References** data source:

- On the **Select reference** step, choose the 'Bill of Materials' reference type.
- On the **Select path** step, set 'Bill of Materials' as 'Multi Valued.'
- On the **Select source** step, choose 'Referenced node.'

The final mapping is displayed as follows:



Once the export is initiated, the output includes the originally selected object ID and name (since those are mapped individually), as well as the name for each of the multiple references, separated by a semicolon (;).

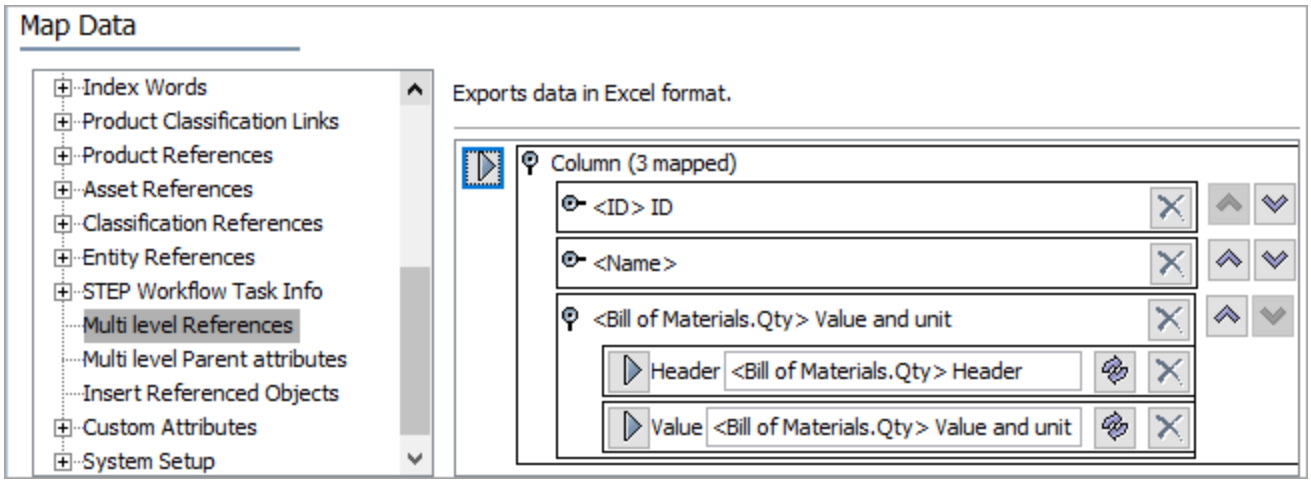
	A	B	C
1	<ID>	<Name>	<Bill of Materials>
2	124458	123858 O	179915;124148

Multiple Reference Metadata in the same Field

Export the same product again, but this time when mapping the **Multi Level References** data source:

- On the **Select reference** step, choose the 'Bill of Materials' reference type, as set previously.
- On the **Select path** step, set 'Bill of Materials' as 'Multi Valued', as set previously.
- On the **Select source** step, now choose 'Reference meta-data attributes.'
- On the **Select attribute** step, now choose the 'Qty' metadata attribute.

The final mapping is displayed as follows:



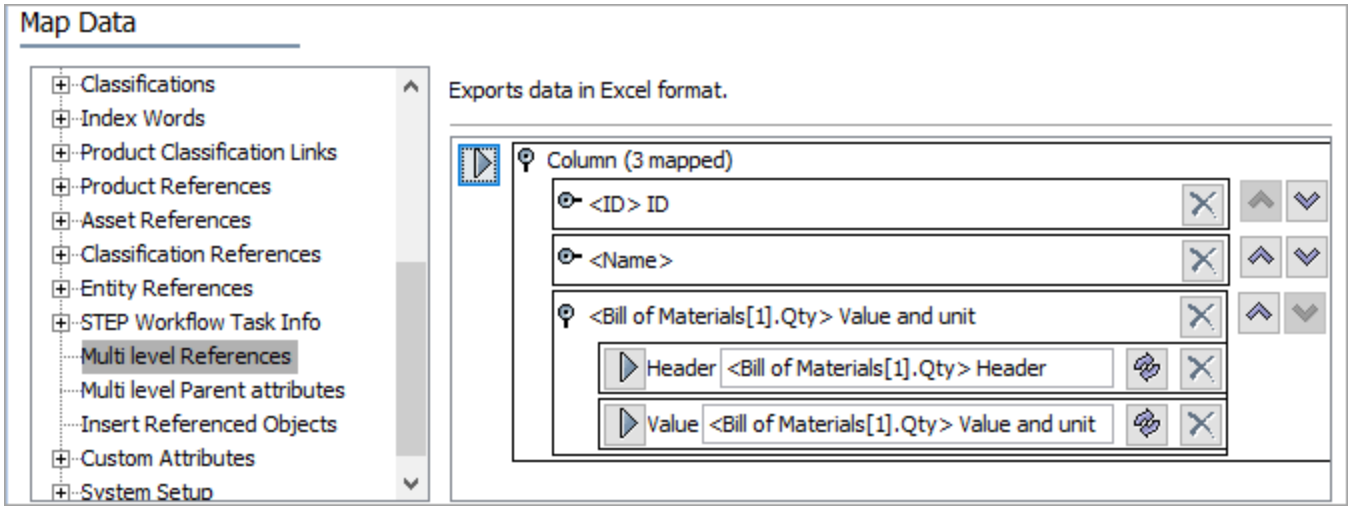
	A	B	C
1	<ID>	<Name>	<Bill of Materials.Qty>
2	124463	124463	5;10

Single Reference Metadata Value

Export the same product again, but this time when mapping the **Multi Level References** data source:

- On the **Select reference** step, choose the 'Bill of Materials' reference type.
- On the **Select path** step, set 'Bill of Materials' as 'Item [1].'
- On the **Select source** step, choose 'Reference meta-data attributes.'
- On the **Select attribute** step, choose the 'Qty' metadata attribute.

The final mapping is displayed as follows:



Once the export is initiated, the output includes the originally selected object ID and name (since those are mapped individually), as well as the value for the first 'Qty' attribute metadata.

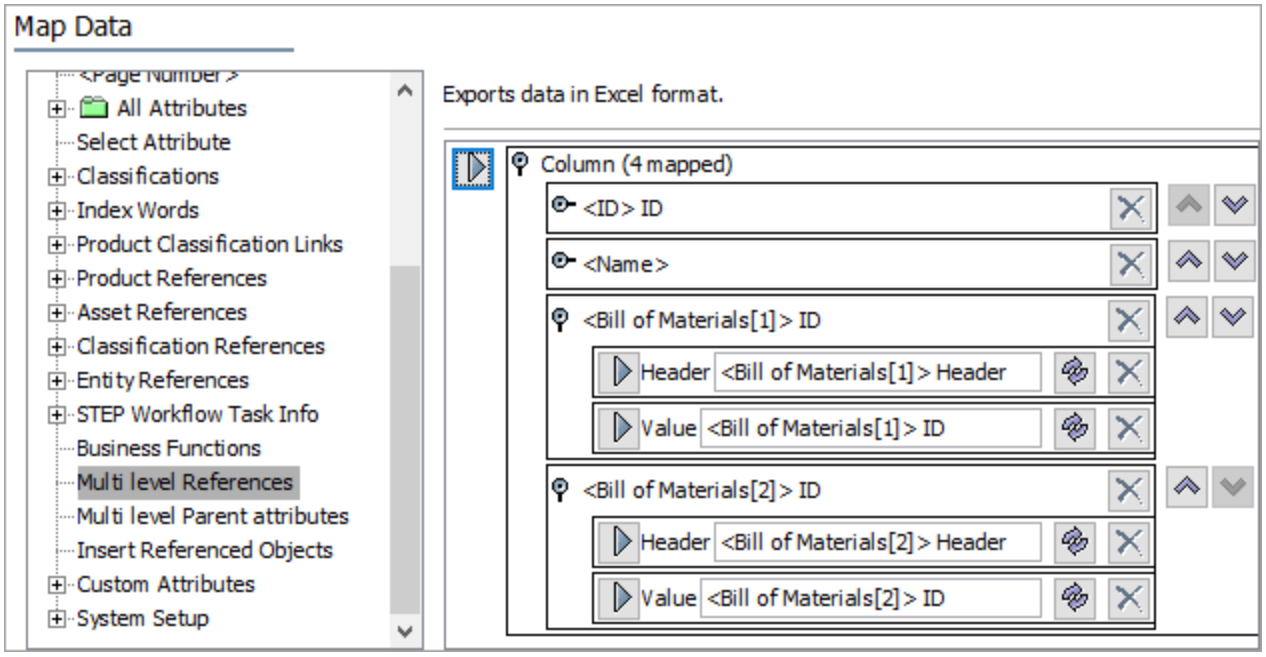
	A	B	C
1	<ID>	<Name>	<Bill of Materials[1].Qty>
2	124458	123858 O	5

Multiple Reference Metadata in Separate Fields

Export the same product again, but this time when mapping the **Multi Level References** data source:

- On the **Select reference** step, choose the 'Bill of Materials' reference type.
- On the **Select path** step, set 'Bill of Materials' as 'Item [1].'
- On the **Select source** step, choose 'Referenced node.'
- Again, use the Multi Level References data source and on the **Select reference** step, choose the 'Bill of Materials' reference type.
- On the **Select path** step, set 'Bill of Materials' as 'Item [2].'
- On the **Select source** step, choose 'Referenced node.'

The final mapping is displayed as follows:



Once the export is initiated, the output includes the originally selected object ID and name (since those are mapped individually), as well as the value for each of the names of the targets of the references, in their own column.

	A	B	C	D
1	<ID>	<Name>	<Bill of Materials[1]>	<Bill of Materials[2]>
2	124463	124463	124148	179915

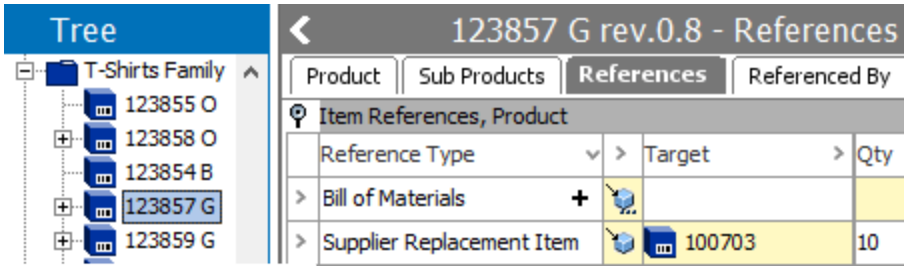
Multi Level References - Examples with Single Value Reference Types

General information about this data source and mapping multi-level references is in the **Multi Level References - Data Source Outbound** topic.

Note: The multi level references data source is available for most formats that require mapping. This option is not supported for the Excel Smartsheet format. This option is also not supported in the STEPXML format since mapping is not available.

For each of the following examples, the same product is exported with ID and name, and then includes a different source or path option in the 'Multi Level data source' wizard. The output results are included to illustrate the differences.

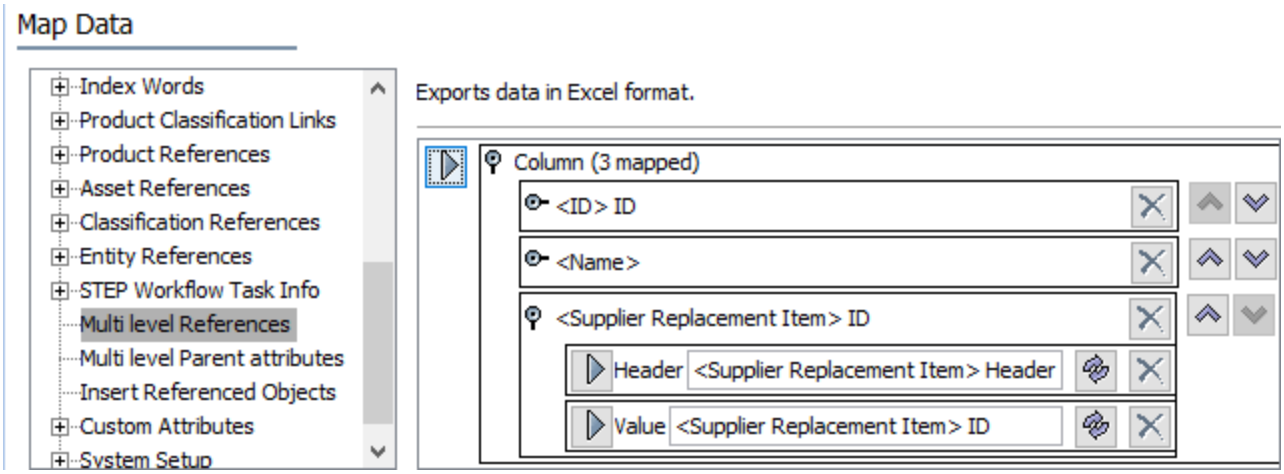
The following product includes a reference to another product with name 100703 via the 'Supplier Replacement Item' reference type. No additional references of this type are allowed, as indicated by the absence of the + sign following the reference type. Additionally, the 'Qty' metadata attribute on the reference has a value.



When mapping the **Multi Level References** data source:

- On the **Select reference** step, choose the 'Supplier Replacement Item' reference type.
- On the **Select path** step, leave 'Supplier Replacement Item' as 'Single Valued.'
- On the **Select source** step, choose 'Referenced node' to return the name of the reference target.

The final mapping displays as follows:



Once the export is initiated, the output includes the originally selected object ID and name (since those are mapped individually), as well as the name of the 'Supplier Replacement Item' referenced target object.

	A	B	C
1	<ID>	<Name>	<Supplier Replacement Item>
2	123857	123857 G	100703

Export the object again. This time, on the **Select source** step choose 'Referenced node attributes' and for the **Select attribute** step select 'Country of Origin' to generate the following output. As before, '.|Node|.' indicates the selected source was 'Referenced node attributes.'

	A	B	C
1	<ID>	<Name>	<Supplier Replacement Item.|Node|.Country of Origin>
2	123857	123857 G	ANTIGUA AND BARBUDA

Export this object again. Set the **Select source** step to 'Reference meta-data attributes' and for the **Select attribute** step select the 'Qty' metadata attribute. This results in the following.

	A	B	C
1	<ID>	<Name>	<Supplier Replacement Item.Qty>
2	123857	123857 G	10

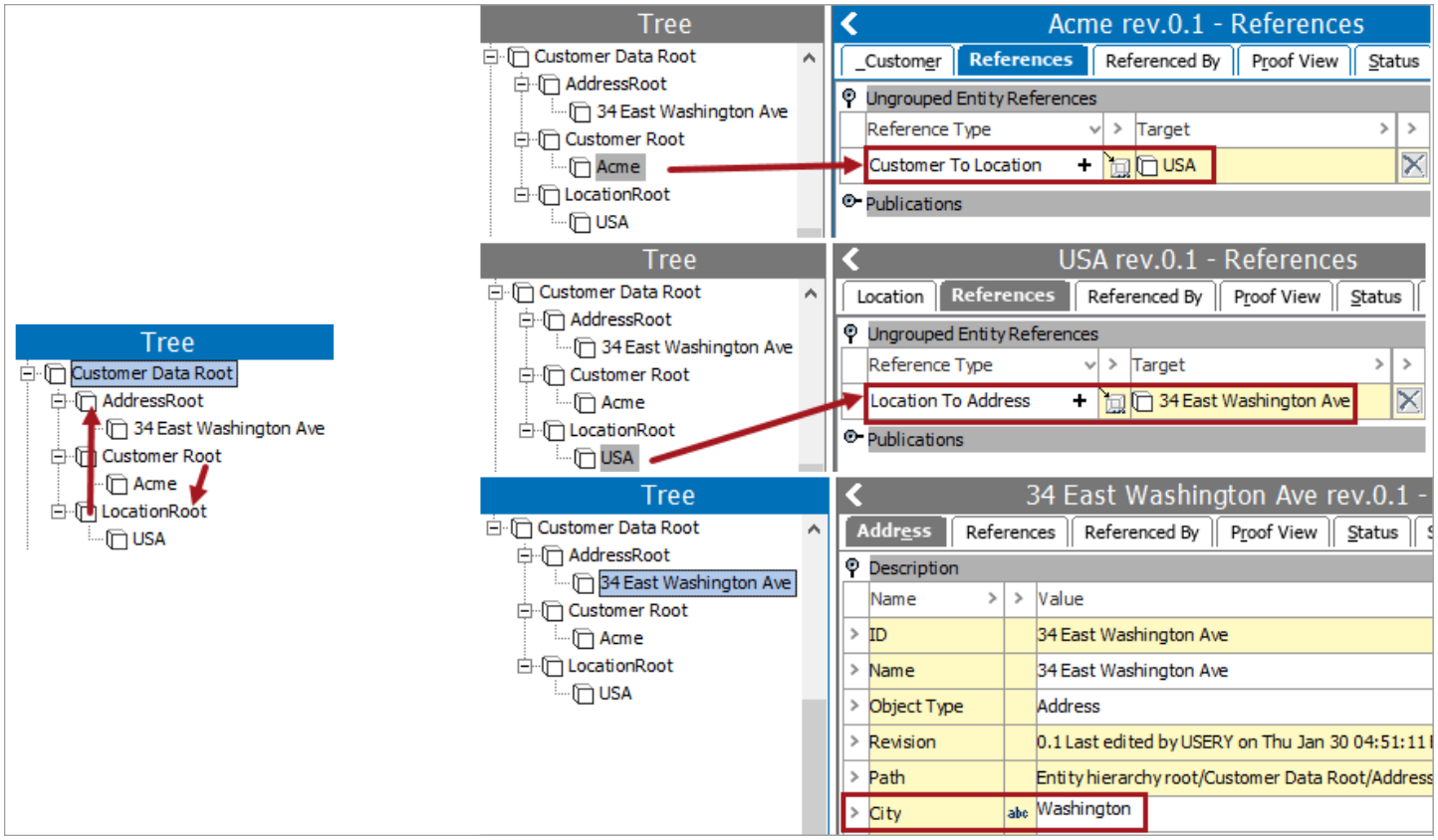
Multi Level References - Examples with Entities

General information about this data source and mapping multi-level references is in the **Multi Level References - Data Source Outbound** topic.

Note: The multi level references data source is available for most formats that require mapping. This option is not supported for the Excel Smartsheet format. This option is also not supported in the STEPXML format since mapping is not available.

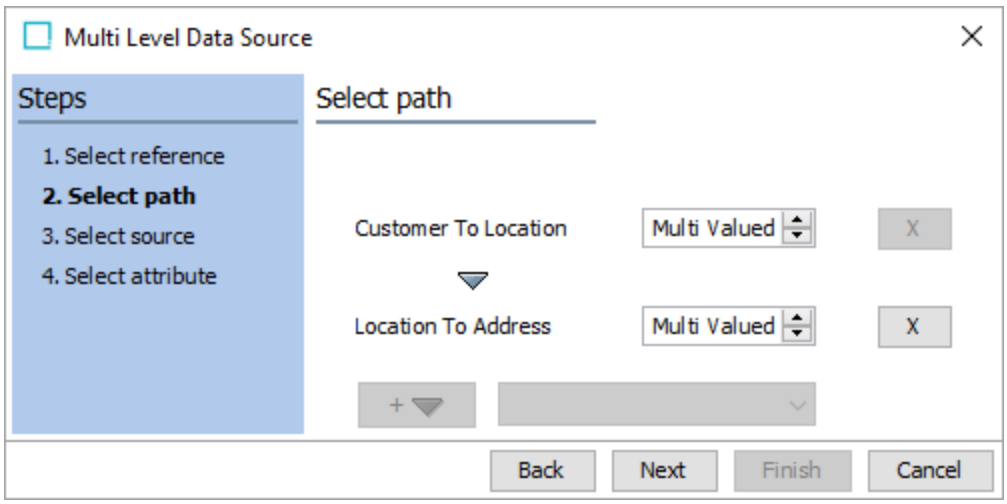
This simple customer hierarchy of entities includes address, customer, and location objects.

- The 'Customer' object references the 'Location' object via the 'Customer To Location' reference type.
- The 'Location' object references the 'Address' object via the 'Location To Address' reference type.
- The 'Address' object includes the valid 'City' attribute, with the 'Washington' value, as shown below.



The user can export the 'Customer' object and also export the 'City' attribute value by following the path of the references. The export can also include any data along the path of the references, as shown below.

Using a format that requires mapping, select the 'Customer' object for export, map the ID and name, and the Multi Level References data source option as shown below.

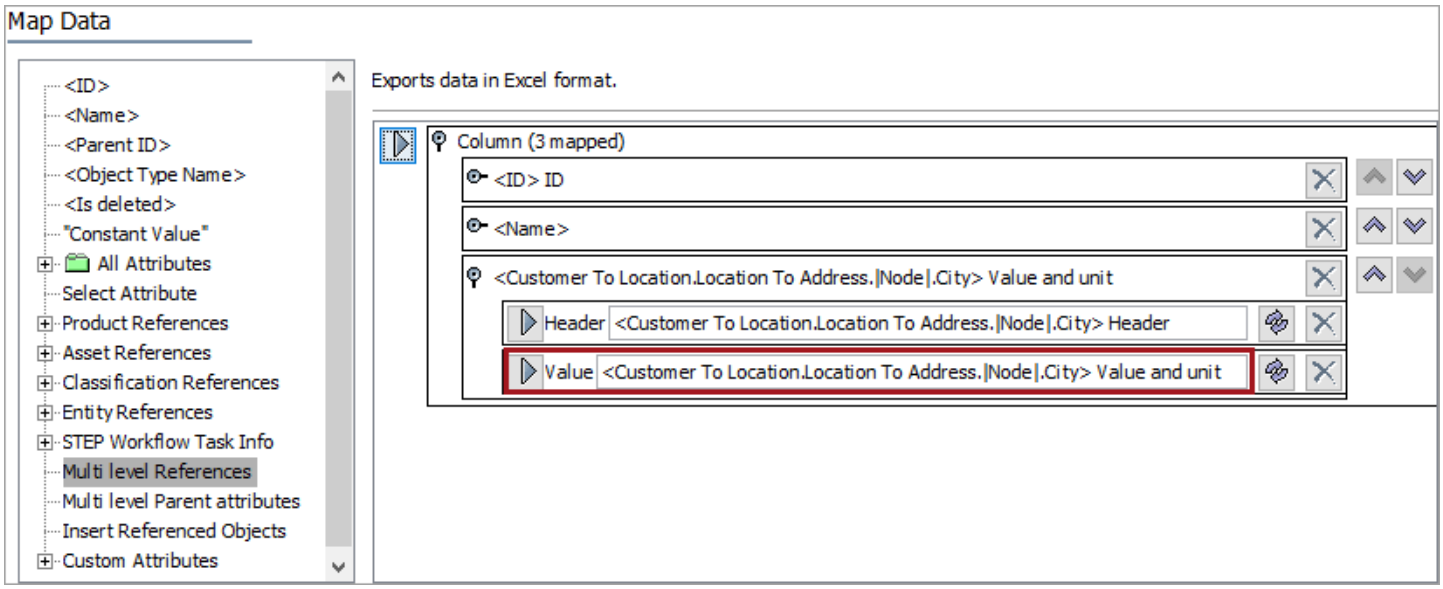


On the 'Select source' step, select the 'Referenced node attributes' option and on the 'Select attribute' step select the 'City' attribute.

This results in the final mapping shown below, including the highlighted **Customer to Location.Location to Address.|Node|.City** entry for the Value mapping target.

Each element in the highlighted mapping target is separated by a 'period' (.) can be translated as:

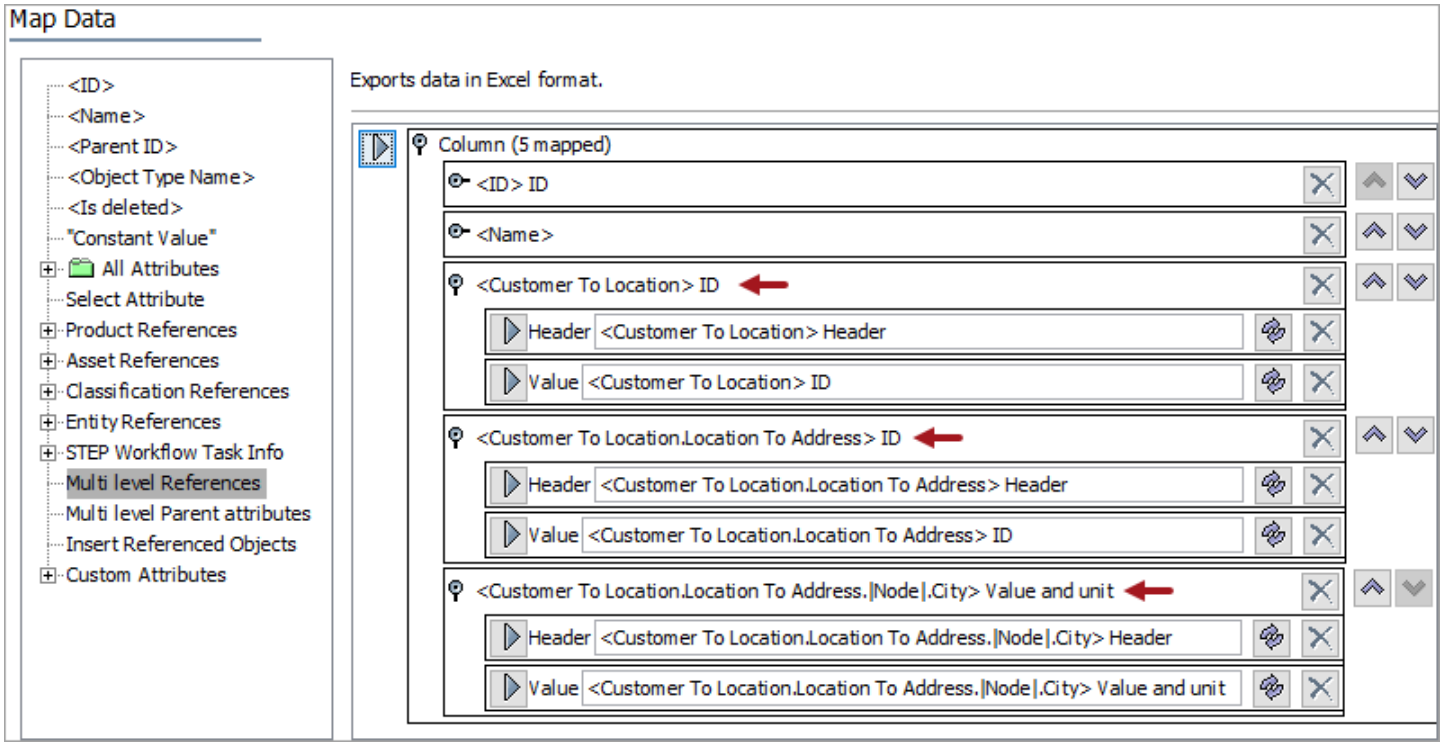
- Customer to Location = the first reference
- Location to Address = the second reference
- |Node| = Referenced node attributes option
- City = the selected attribute



Initiate the export and view the output which includes the ID and name of the 'Customer' and the value of the 'City' attribute on the final target reference.

	A	B	C
1	<ID>	<Name>	<Customer To Location.Location To Address.|Node|.City>
2	Acme	Acme	Washington

Repeating this same example, map the 'Reference node' once for each of the references to show the full path of the references. And then map the 'Referenced node attributes' to get the attribute value on the target of the final reference (as also done in the previous example).



When the export is initiated, the output includes the names of each node in the reference path in addition to the 'City' attribute value as follows:

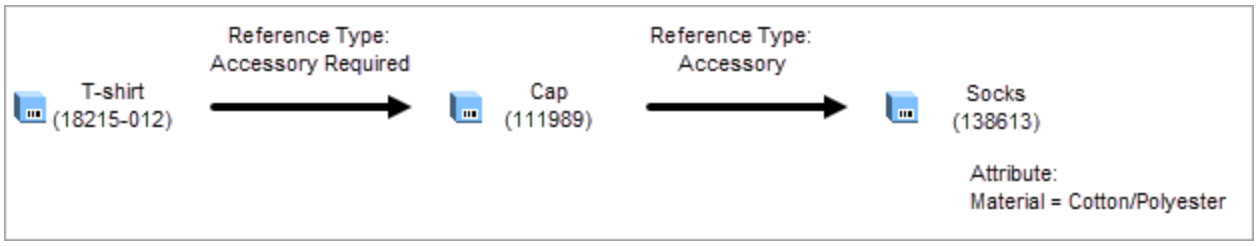
	A	B	C	D	E
1	<ID>	<Name>	<Customer To Location>	<Customer To Location.Location To Address>	<Customer To Location.Location To Address.|Node|.City>
2	Acme	Acme	USA	34 East Washington Ave	Washington

Multi Level References - Examples with Products

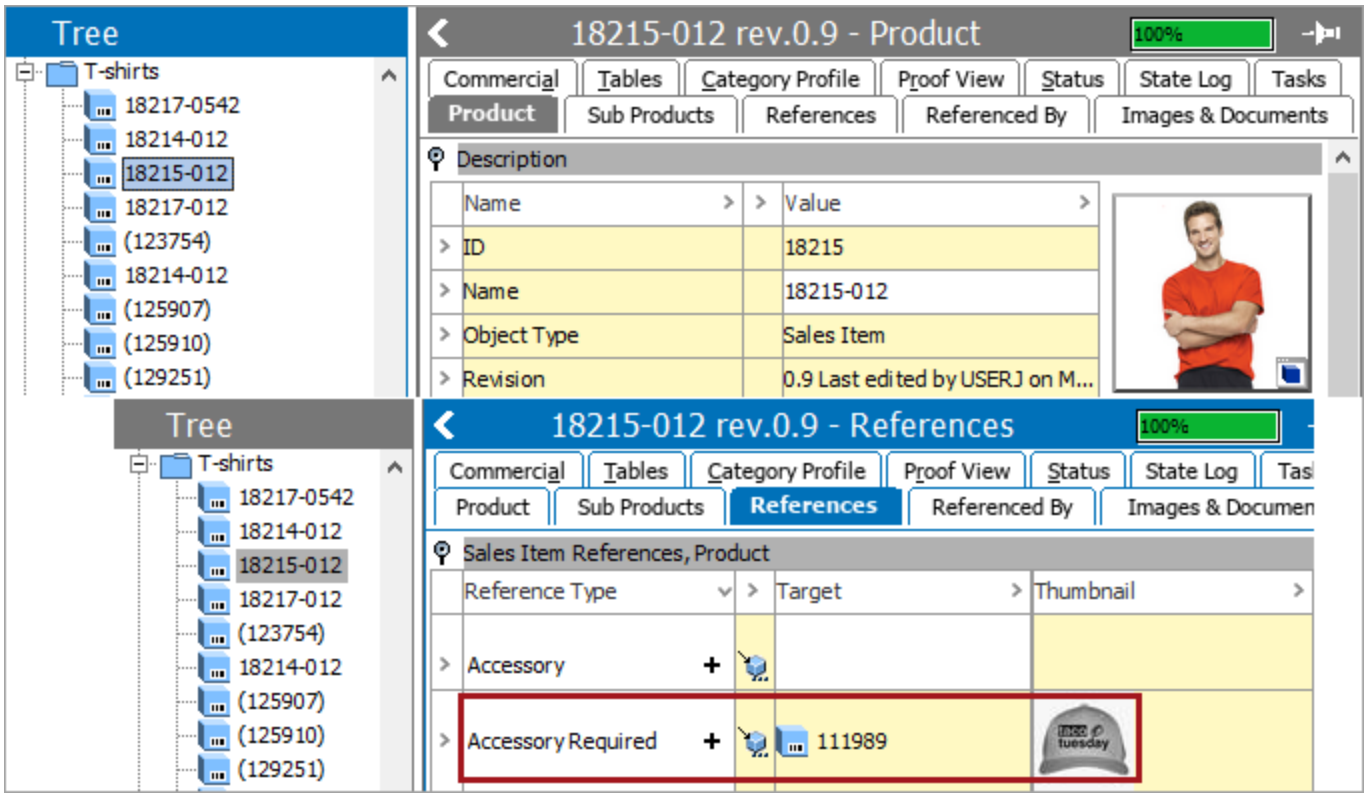
General information about this data source and mapping multi-level references is in the **Multi Level References - Data Source Outbound** topic.

Note: The multi level references data source is available for most formats that require mapping. This option is not supported for the Excel Smartsheet format. This option is also not supported in the STEPXML format since mapping is not available.

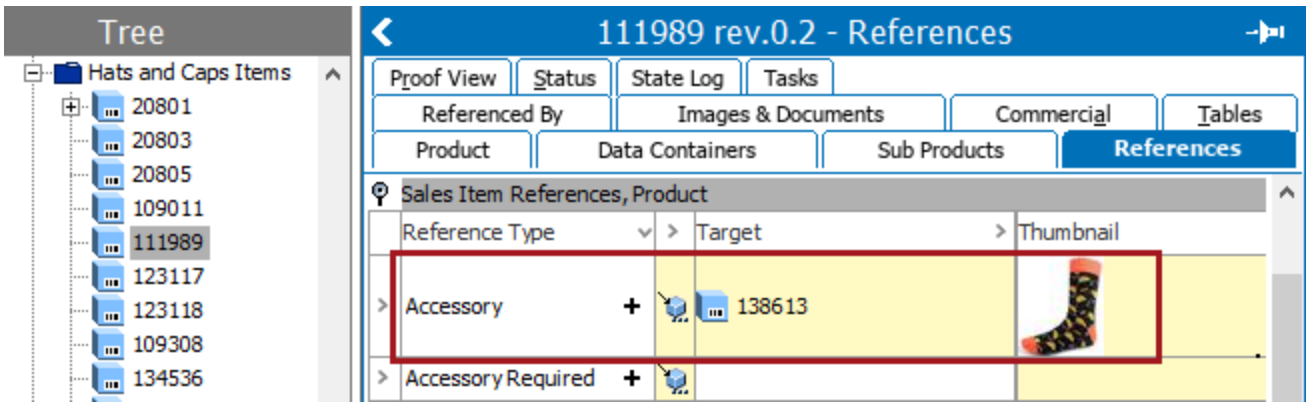
This simple hierarchy of products related by references including a t-shirt, a cap, and a pair of socks.



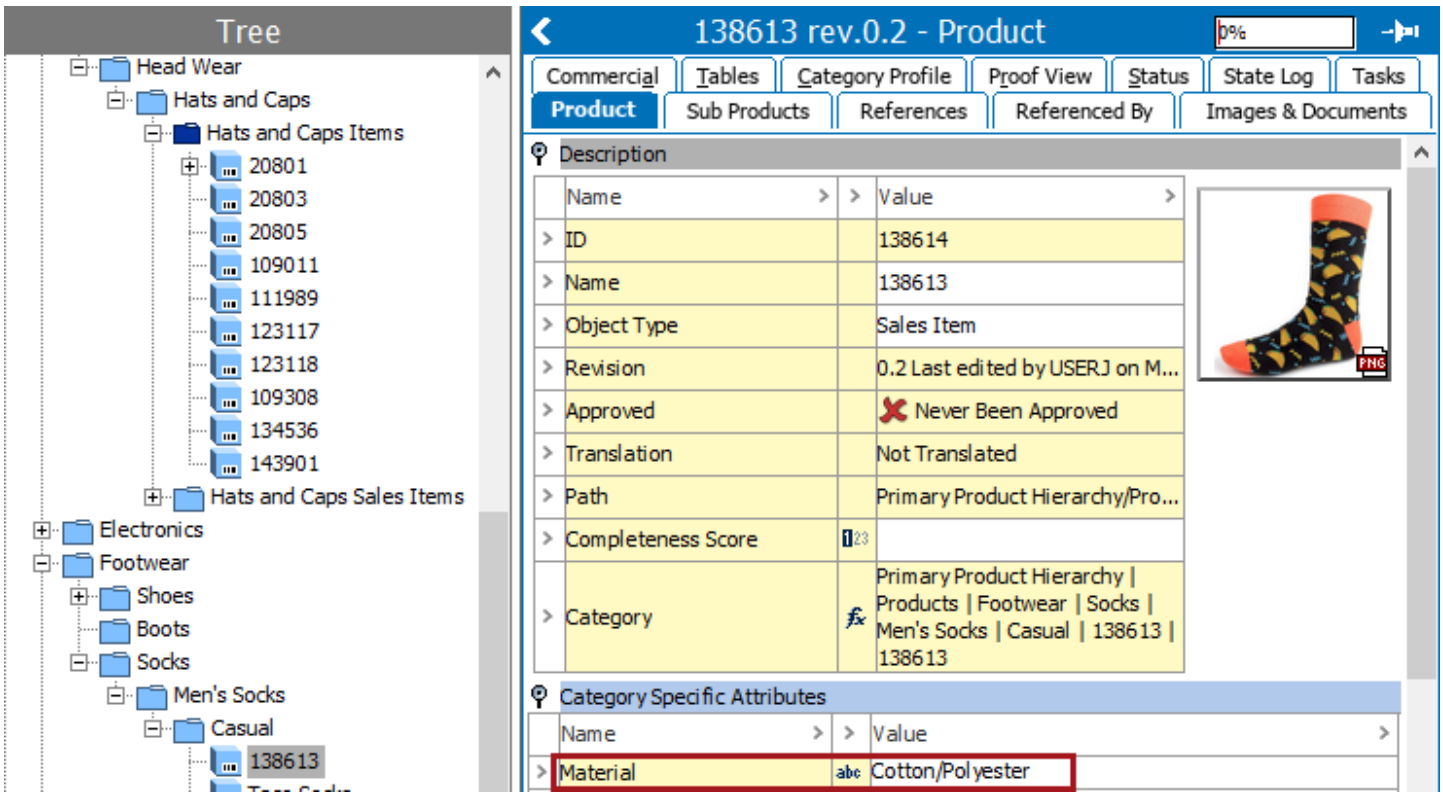
The user can export the 'T-shirt' object and can also export data along the path of the references.



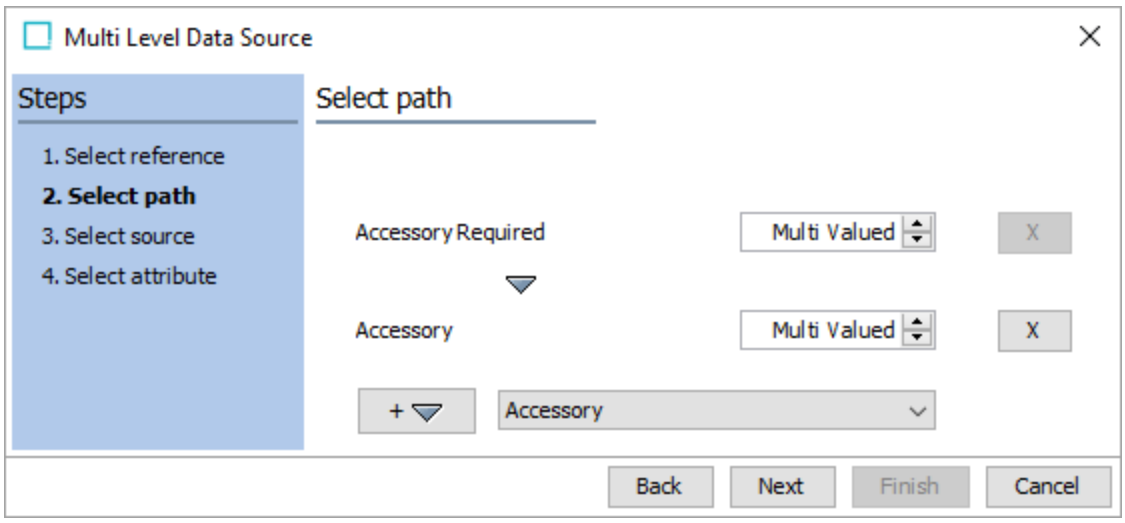
The output should be an attribute value on the final reference target, which is the object with '138613' as the name.



The 'Material' attribute value is available for export by following the path of the references.



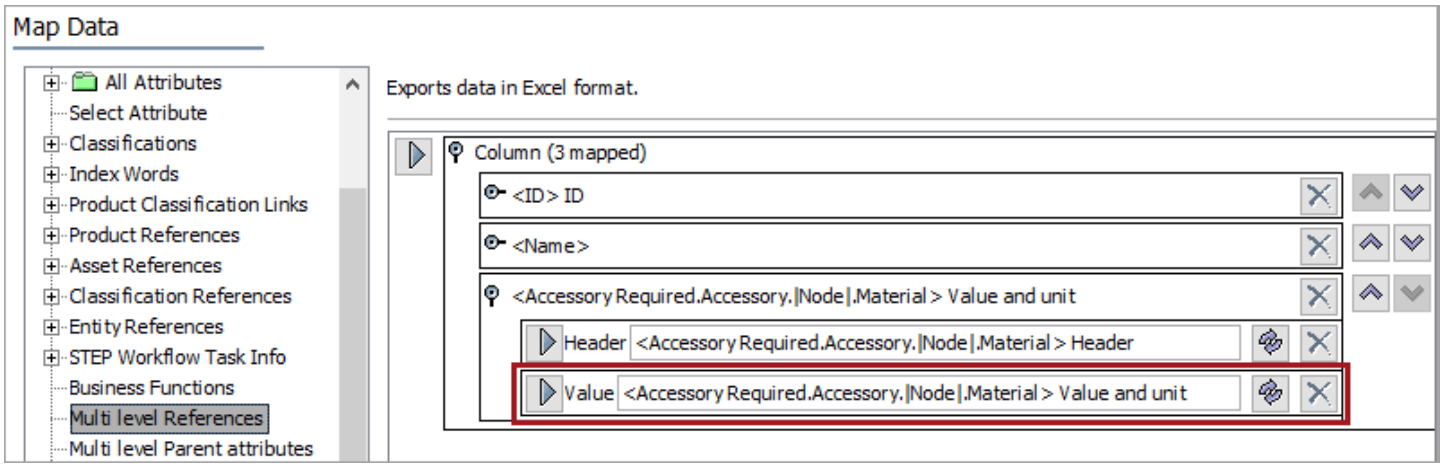
Using a format that requires mapping, select the 'T-shirt' object for export, map the ID and name, and the Multi Level References data source option as shown below.



On the 'Select source' step, select the 'Referenced node attributes' option and on the 'Select attribute' step select the 'Material' attribute.

This results in the final mapping shown below, including the highlighted **AccessoryRequired.Accessory.|Node|.Material** entry for the Value mapping target. Each element in the highlighted mapping target is separated by a 'period' (.) can be translated as:

- AccessoryRequired= the first reference
- Accessory = the second reference
- |Node| = Referenced node attributes option
- Material = the selected attribute

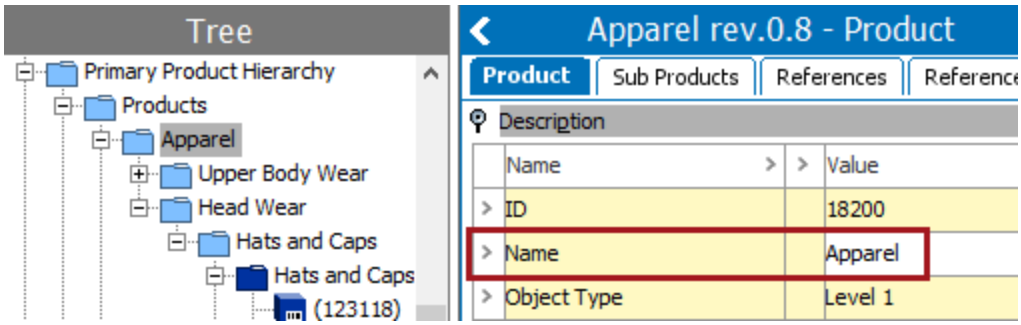


Initiate the export and the output includes the ID and name of the 'T-shirt' and the value of the 'Material' attribute on the final target reference.

	A	B	C
1	<ID>	<Name>	<Accessory Required.Accessory.|Node|.Material>
2	18215	18215-012	Cotton/Polyester

Name - Data Source Outbound

The option to map a name allows output of the value of the Name attribute for the selected objects.



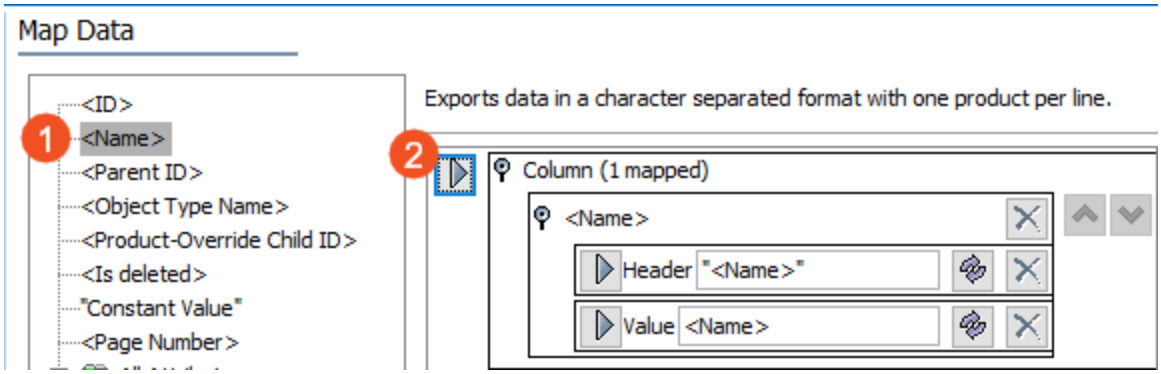
The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

For more information about the additional wizard steps, refer to the **Creating a Data Export** topic or the **Creating an Outbound Integration Endpoint** topic.

After mapping, most output options can be altered using transformations. For more information, refer to the **Outbound Map Data - Transform** topic.

Mapping Name

1. Select the data to be exported for the output tool. For more information, refer to the **Data Exchange** topic.
2. On the Map Data step, in the left panel, select the <Name> node and click the mapping button ( ).



3. Apply any transformations, which can change the output without changing the original data. Refer to the **Outbound Map Data - Transform** topic.
4. Complete any additional mappings and initiate the export.

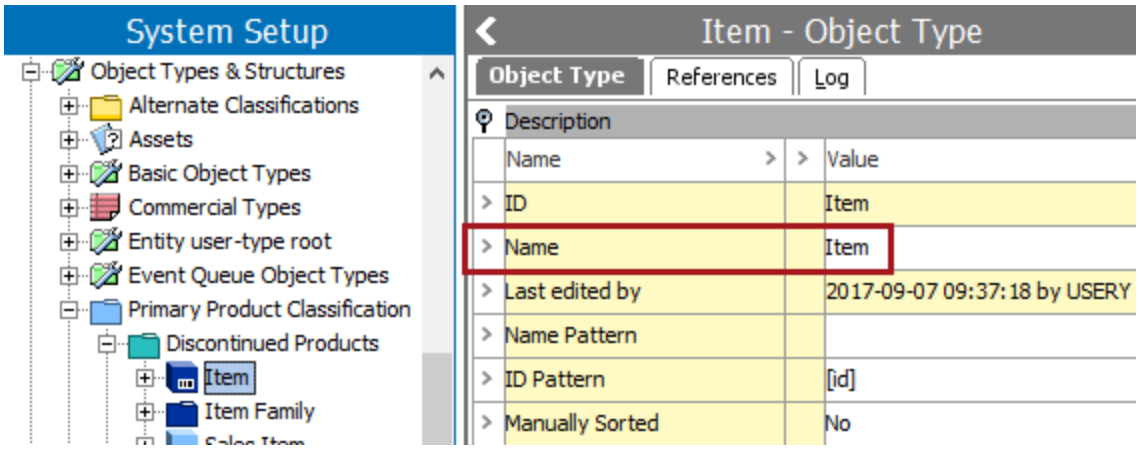
Results

The output includes <Name> as a header and the value for each row.

	A
1	<Name>
2	18210 M B_EN
3	18212 L B
4	18213 M O
5	18216 L O

Object Type Name - Data Source Outbound

The option to map the name of a STEP object type can allow a downstream system to identify the type of data being received.



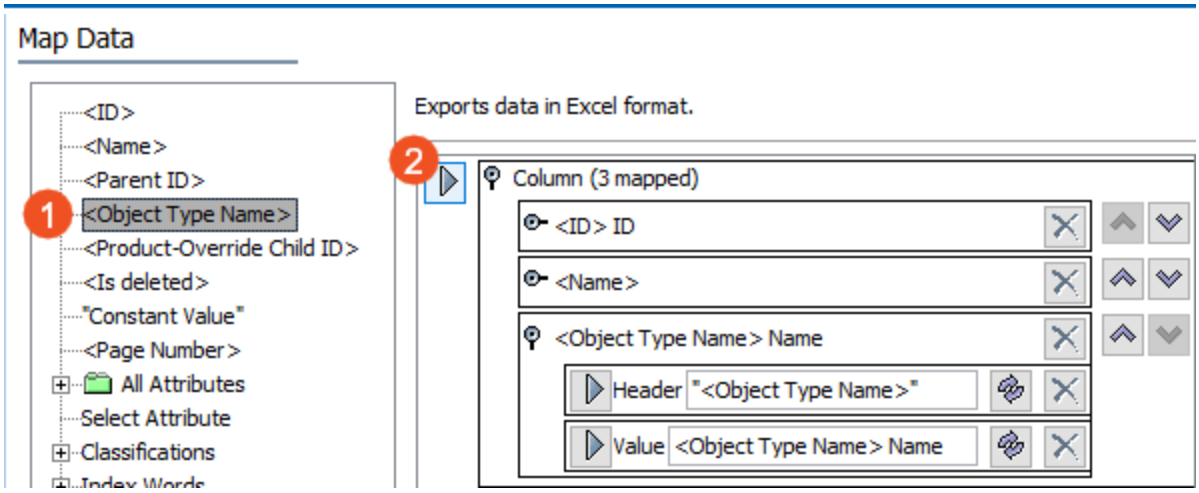
The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

For more information about the additional wizard steps, refer to the **Creating a Data Export** topic or the **Creating an Outbound Integration Endpoint** topic.

After mapping, most output options can be altered using transformations. For more information, refer to the **Outbound Map Data - Transform** topic.

Mapping Object Type Name

1. Select the data to be exported for the output tool. For more information, refer to the **Data Exchange** topic.
2. Select the super type for the desired object types. For more information, refer to **Export Manager - Select Objects**, the **Configure the Format** section of the **OIEP - Event-Based - Output Templates Flipper**, or the **Configure the Format** section of the **OIEP - Select Objects - Output Templates Flipper**.
3. On the Map Data step, in the left panel, select the <Object Type Name> node and click the right arrow button ( ).



4. Click the Transformation button ( ) to apply any transformations, which can change the output without changing the original data.

> **Note:** While the default selection is 'Object Type Name' for export, when importing the same file, the auto-mapping feature expects the 'Object Type ID.' To generate an export file that is prepared for reimport, use the Transformation Aspect to map object type ID. For more information, refer to the **Aspect - Transform Outbound** section of the **Data Mapping** documentation.

5. Complete any additional mappings and initiate the export.

Results

The output includes <Object Type Name> as a header (or Object Type ID if transformed) and the value for each row.

	A	B	C
1	<ID>	<Name>	<Object Type Name>
2	121933	121933	Open Item
3	20803	Red Baseball Cap	Item
4	20805	20805	Item
5	101567	20808-013	Sales Item
6	109013	20803-04	Sales Item
7	110604	20808-08	Sales Item
8	110606	20808-09	Sales Item
9	121932	SalesItem 121932	Open Sales Item

Page Number – Data Source Outbound

After all InDesign documents for a publication have been finalized and saved back to STEP, data about the mounted products and the pages on which they appear can be exported from the publication in order to generate an **Alphabetical Index - XML** file. This XML file format is used to create both types of indexes—the basic 'stock number' index and the more advanced 'alphabetical index.'

Details about this option can be found in the **Exporting Index Data from STEP** section of the **STEP'n'design** documentation.

The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

For more information about the additional wizard steps, refer to the **Creating a Data Export** topic or the **Creating an Outbound Integration Endpoint** topic.

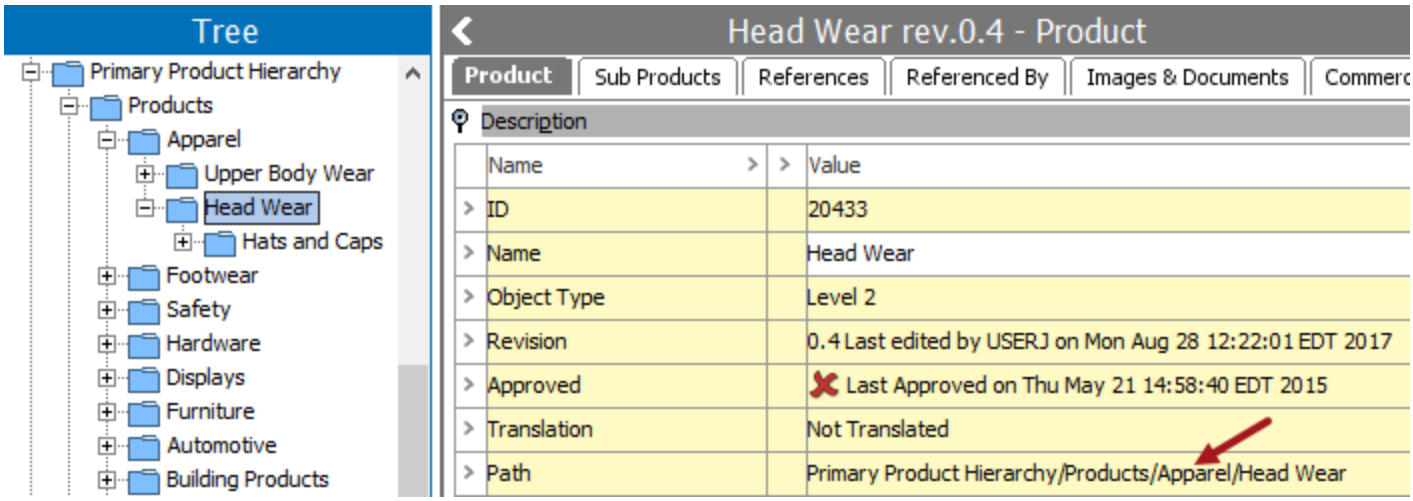
After mapping, most output options can be altered using transformations. For more information, refer to the **Outbound Map Data - Transform** topic.

Parent ID - Data Source Outbound

The option to map the parent ID of an object identifies the object's place in the hierarchy.

When a key is used in place of the STEP parent ID by external systems, the key(s) must be mapped instead. For more on keys, refer to the **Unique Keys** section of the **System Setup** documentation.

The parent is displayed as part of the Path parameter on an object editor, and indicates the node directly above the selected object.



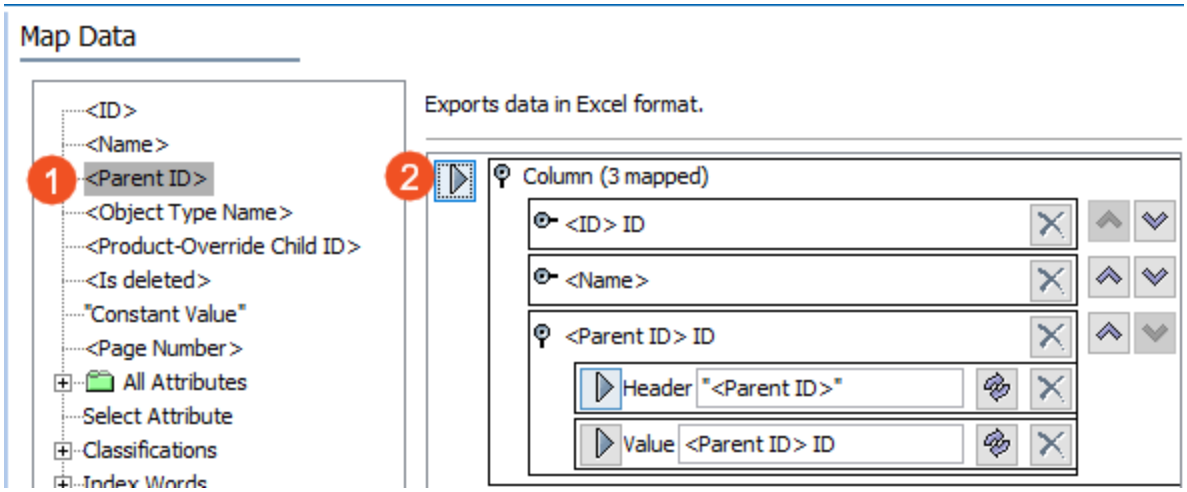
The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

For more information about the additional wizard steps, refer to the **Creating a Data Export** topic or the **Creating an Outbound Integration Endpoint** topic.

After mapping, most output options can be altered using transformations. For more information, refer to the **Outbound Map Data - Transform** topic.

Mapping Parent ID or Key

1. Select the data to be exported for the output tool. For more information, refer to the **Data Exchange** topic.
2. On the Map Data step, in the left panel, select the <Parent ID> node and click the right arrow button.



3. Click the Transformation button ( ) to apply any transformations, which can change the output without changing the original data. For example, change the output from Name to ID using the Aspect parameter. Refer to the **Outbound Map Data - Transform** topic.
4. Complete any additional mappings and initiate the export.

Results

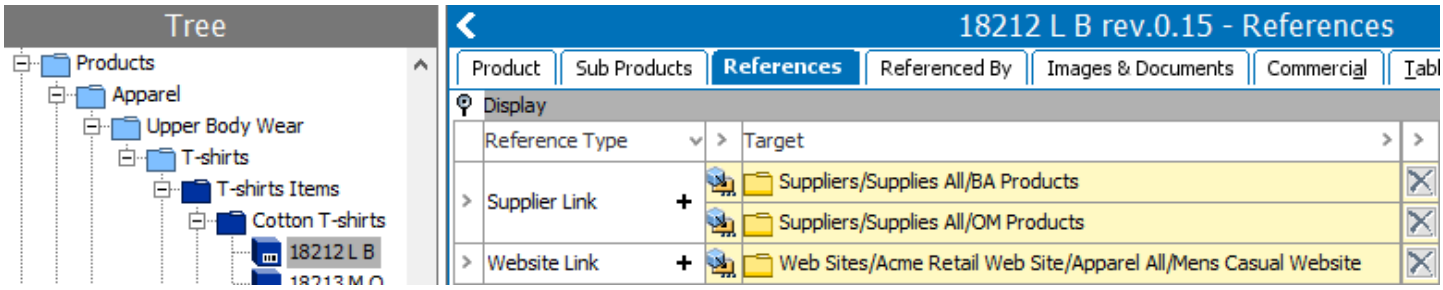
The output includes <Parent ID> as a header (or Object Type ID if transformed) and the value for each row.

	A	B	C
1	<ID>	<Name>	<Parent ID>
2	18210	18210 M B_EN	18209
3	18212	18212 L B	18209
4	18213	18213 M O	18209
5	18216	18216 L O	18209
6	100703	12-GGK799	113201

Product Classification Links – Data Source Outbound

The option to map a Product Classification Link is only available when the object super type selected for the export is Product.

When mapping a Product Classification Link, users can include or exclude inherited references.



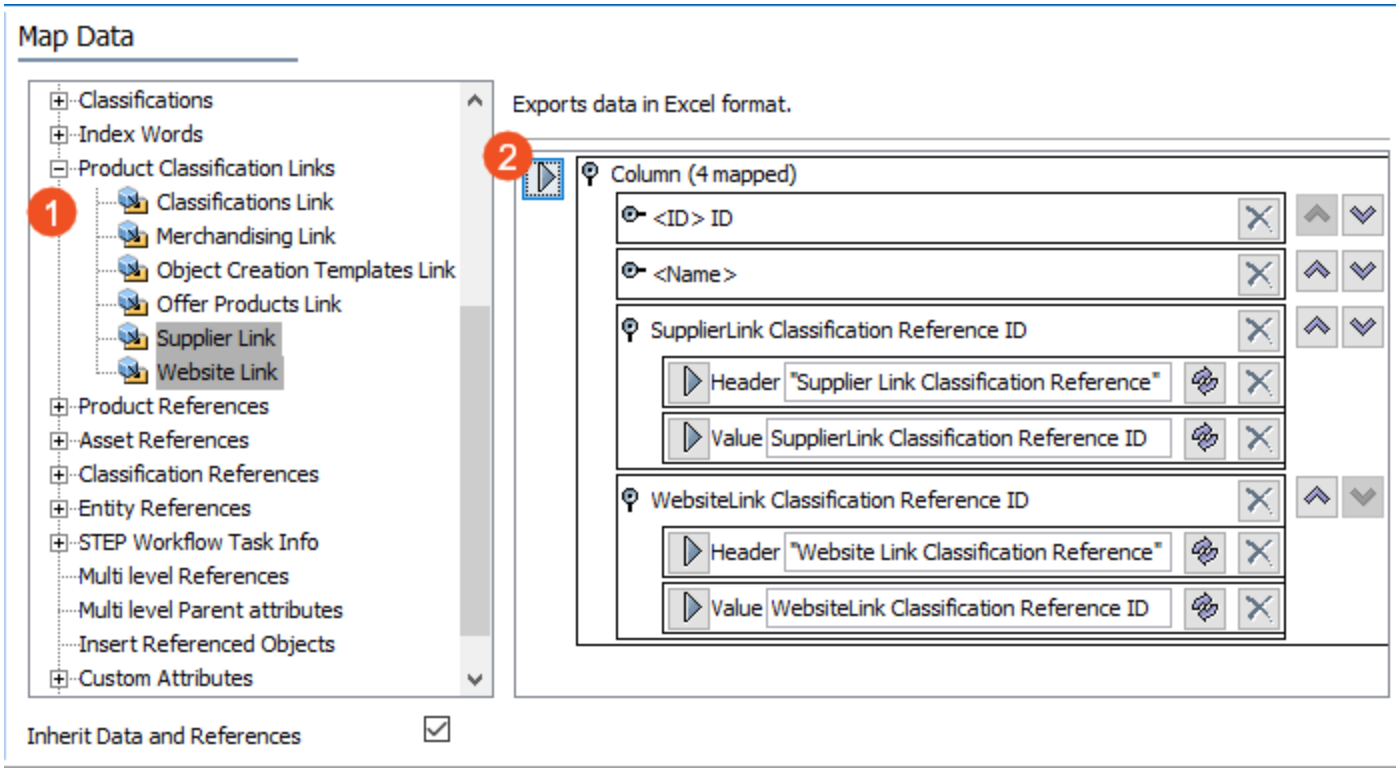
The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

For more information about the additional wizard steps, refer to the **Creating a Data Export** topic or the **Creating an Outbound Integration Endpoint** topic.

After mapping, most output options can be altered using transformations. For more information, refer to the **Outbound Map Data - Transform** topic.

Mapping Product Classification Links

1. Select the data to be exported for the output tool. For more information, refer to the **Data Exchange** topic.
2. On the Map Data step, in the left panel, expand Product Classification Links, and select one or more product classification link types.



3. Click the right arrow button in the right panel to add the selected product classification link type as a column.
4. Check the **Inherit Data and References** option appropriately:
   - If **checked**, inherited data and references are included in the export.
   - If **unchecked**, inherited data and references is not included in the export.

   For information on inheritance, refer to the **Inherit Data and References - Data Source Outbound** topic.
5. Click the Transformation button ( ) to apply any transformations, which can change the output without changing the original data. Refer to the **Outbound Map Data - Transform** topic.
6. Complete any additional mappings and initiate the export.

Results

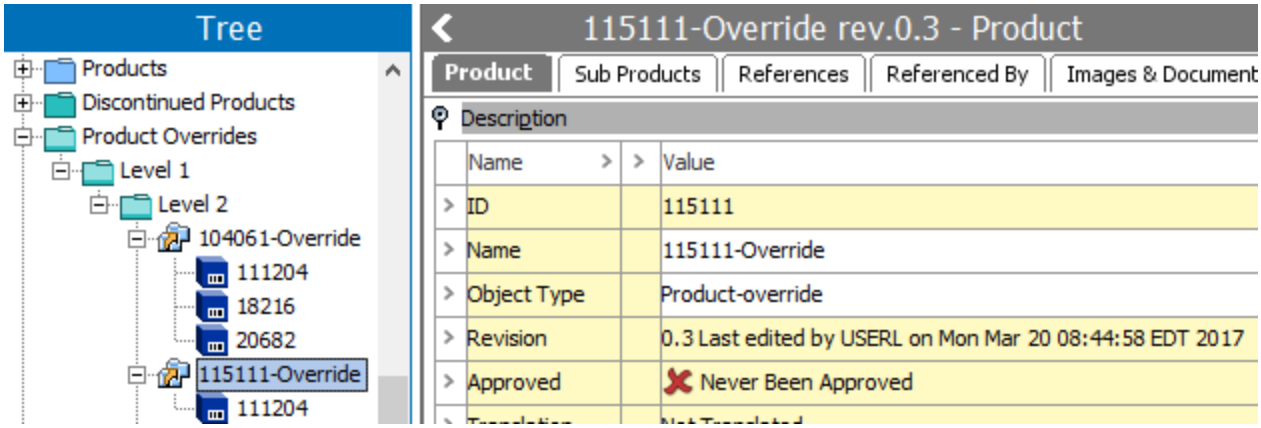
The output includes the product classification link type name as a header and the ID for each row.

When the product references multiple product classification links, the product classification link IDs are displayed in a semi-colon separated list.

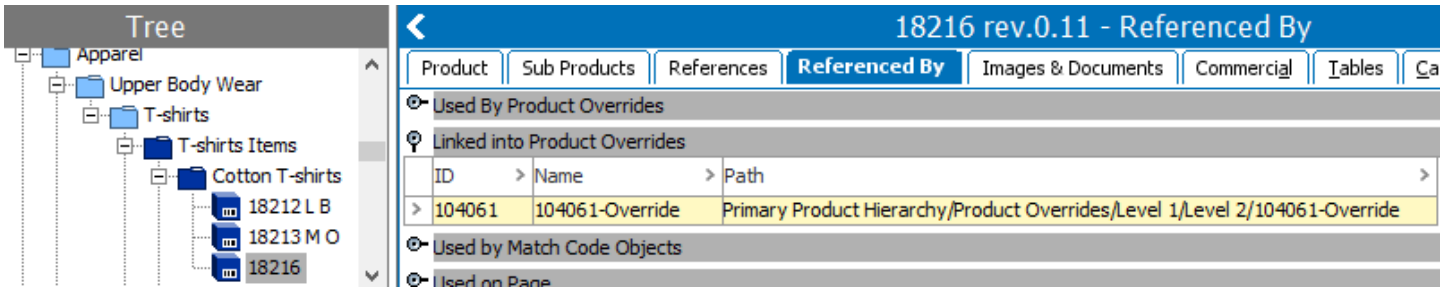
	A	B	C	D
1	<ID>	<Name>	Supplier Link Classification Reference	Website Link Classification Reference
2	18212	18212 L B	BAProducts;OMProducts	22583

Product-Override Child ID – Data Source Outbound

The option to map a Product-Override Child ID is only available when the object super type selected for the export is Product. This allows you to get child products associated to product-overrides, as shown in the following image.



When viewing a product that has been overridden, the Referenced By tab displays the overrides in the Linked into Product Overrides flipper.



When mapping a product-override child ID, users can include or exclude inherited references.

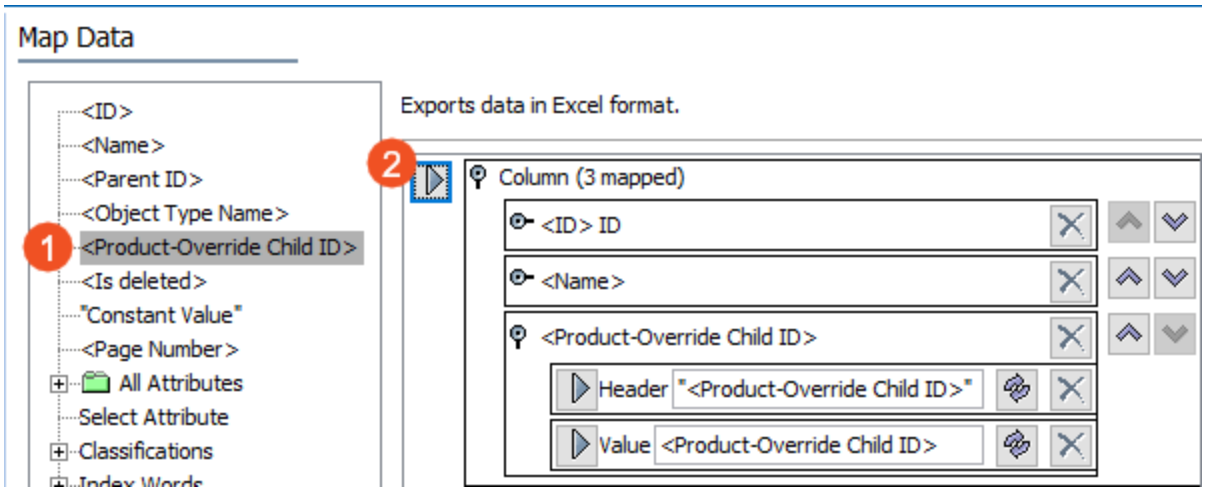
The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

For more information about the additional wizard steps, refer to the **Creating a Data Export** topic or the **Creating an Outbound Integration Endpoint** topic.

After mapping, most output options can be altered using transformations. For more information, refer to the **Outbound Map Data - Transform** topic.

Mapping Product-Override Child ID

1. Select the data to be exported for the output tool. For more information, refer to the **Data Exchange** topic.
2. Select the product super type and the product override objects for export. For more information, refer to **Export Manager - Select Objects**, the **Configure the Format** section of the **OIEP - Event-Based - Output Templates Flipper**, or the **Configure the Format** section of the **OIEP - Select Objects - Output Templates Flipper**.
3. On the Map Data step, in the left panel, select the Product-Override Child ID option.



4. Click the right arrow button ( ) in the right panel to add the product-override child ID as a column.
5. Check the **Inherit Data and References** option appropriately:
   - If **checked**, inherited data and references are included in the export.
   - If **unchecked**, inherited data and references is not included in the export.

   For information on inheritance, refer to the **Inherit Data and References - Data Source Outbound** topic.
6. Click the Transformation button ( ) to apply any transformations, which can change the output without changing the original data. For example, change the output from Name to ID using the Aspect parameter. Refer to the **Outbound Map Data - Transform** topic.
7. Complete any additional mappings and initiate the export.

Results

The output includes the product IDs selected for export and the product-override child ID as a header.

When the product-override has multiple children, the child IDs are displayed in a semi-colon separated list.

	A	B	C
1	<ID>	<Name>	<Product-Override Child ID>
2	104061	104061-Override	20682;18216;111204
3	115111	115111-Override	111204

Product References - Data Source Outbound

Product references can be exported when they reference products, classifications, or entities. Product references can be mapped via their ID or a unique key. Each selected product reference is extracted into a separate field in the output.

The Aspect option is available for Product References and allows a variety of data on the product reference to be exported.

The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

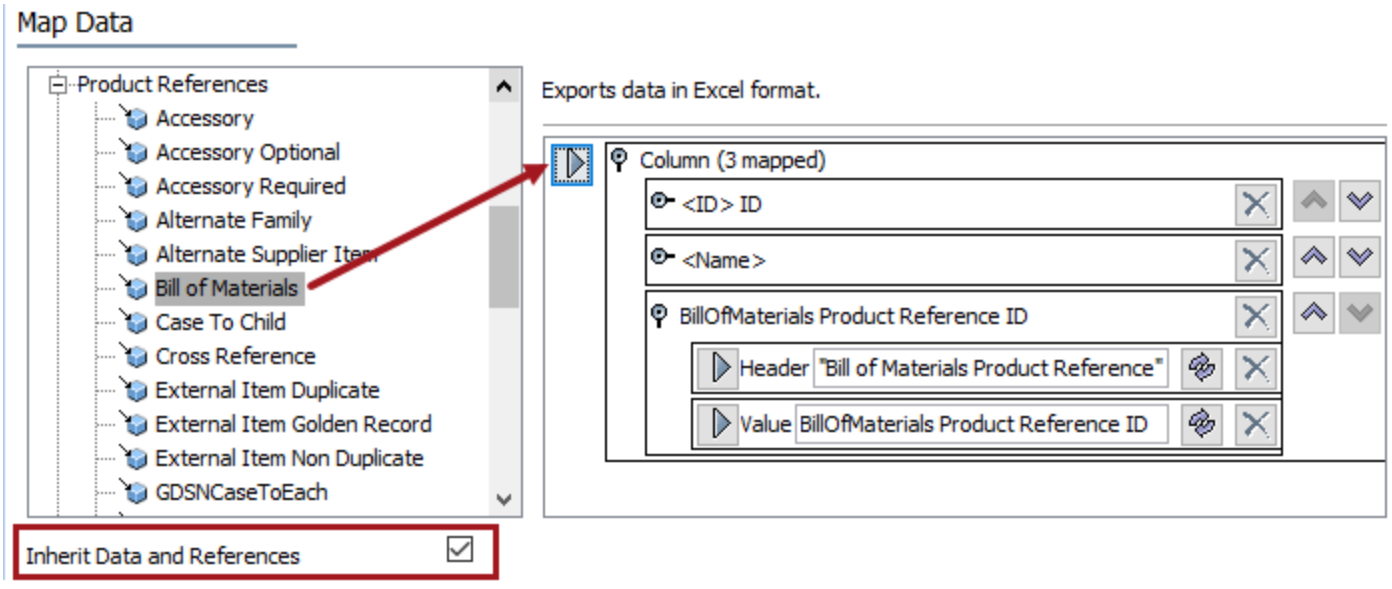
For more information about the additional wizard steps, refer to the **Creating a Data Export** topic or the **Creating an Outbound Integration Endpoint** topic.

After mapping, most output options can be altered using transformations. For more information, refer to the **Outbound Map Data - Transform** topic.

1. Select the data to be exported for the output tool. For more information, refer to the **Data Exchange** topic.
2. On the Map Data step, in the left panel, expand **Product References**, and then select the preferred reference type from the list of those available in your system.

   Note: For an OIEP, you must first select 'Product' from the dropdown at the top of the Mapping tab.

3. In the right panel, click the right arrow button ( ) to add the selected reference type.

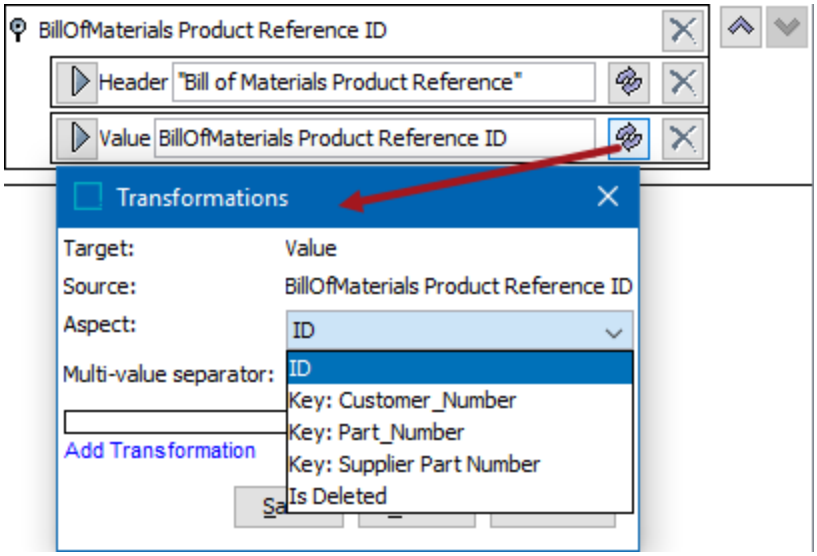


4. Check the **Inherit Data and References** option appropriately:

- If **checked**, inherited data and references are included in the export.
- If **unchecked**, inherited data and references is not included in the export.

For information on inheritance, refer to the **Inherit Data and References - Data Source Outbound** topic.

5. If needed, change the reference Value Aspect to output data other than the reference ID. For details on the options, refer to the **Aspect - Transform Outbound** topic.



- Open the asset reference flipper to display the Header and Value elements.
- Click the transformation button ( ) for the Header or for the Value element to display the Transformations dialog and the Aspects parameter.
- Click the **Aspect** dropdown to display and select an option.
- Click the **Save** button to apply any aspect changes.

6. Click the transformation button ( ) to apply any transformations necessary to change the output without changing the original data. Refer to the **Outbound Map Data - Transform** topic for details.
7. Complete any additional mappings and initiate the export.

Results

The following output is generated using the mapping shown above. The file includes the ID of the object selected for export, the name of that object, and the selected product reference type ID, as well as values for the object.

	A	B	C
1	<ID>	<Name>	Supplier Replacement Item Product Reference
2	18212	Item 1	100703

When product-to-product references exist for the exported object, a column is output for each product reference type ID, as well as the value for the reference type, as shown below. Multiple references for the same reference type are output in a semi-colon-separated list, as shown below.

	A	B	C	D
1	<ID>	<Name>	Bill of Materials Product Reference	Supplier Replacement Item Product Reference
2	18212	Item 1		100703
3	18213	18213 M O		124148
4	18216	18216	123857;123854	123855

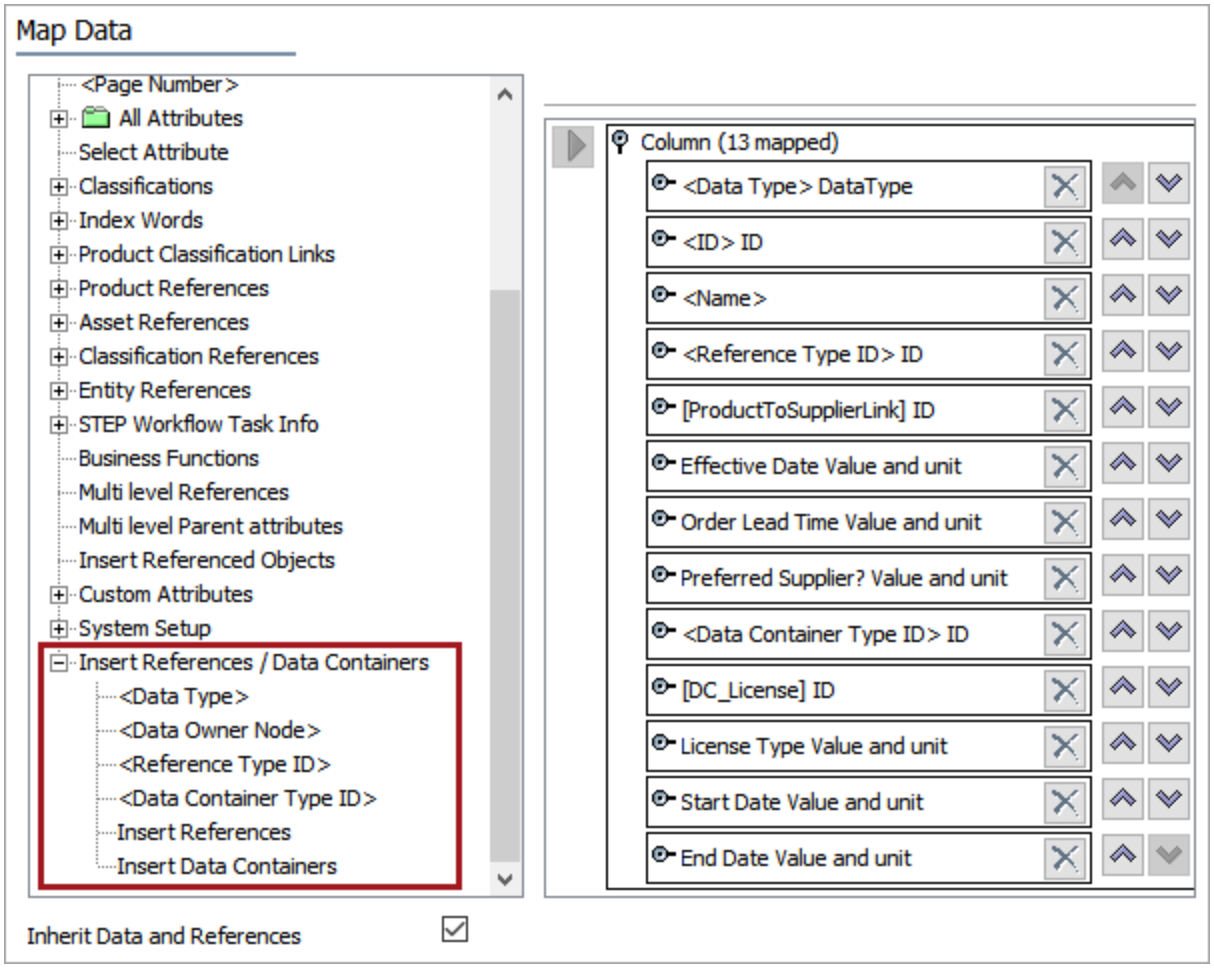
When the transformation Aspect parameter is used and a Key is selected to identify a reference type, the output includes the key in the reference type column, as shown below.

	A	B	C
1	<ID>	<Name>	Bill of Materials Product Reference KeyID=SupplierPartNumber
2	18212	Item 1	179911

Reference Type ID - Data Source Outbound

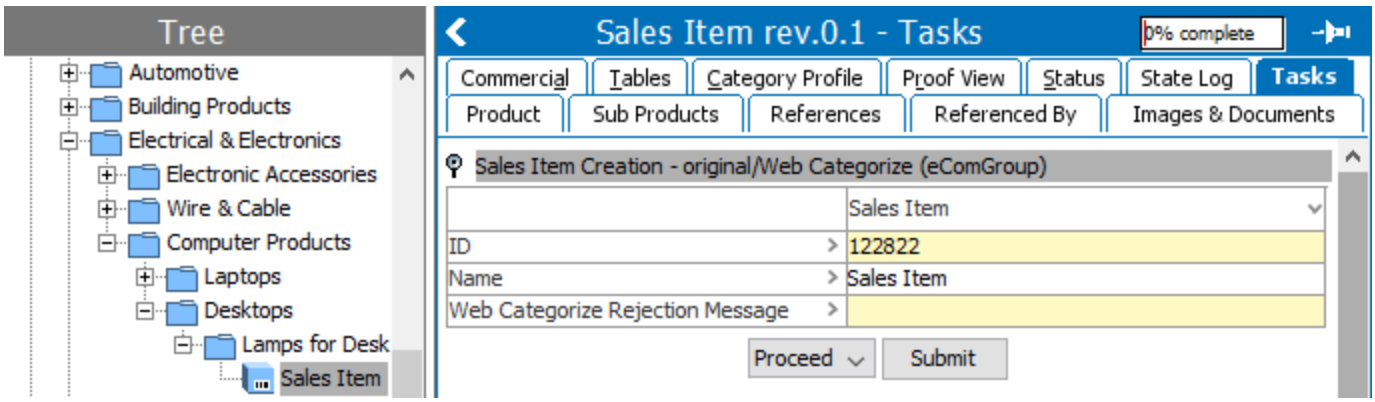
The 'Reference Type ID' option allows output of the ID of the reference or classification product link type for rows representing a reference or classification product link object. This option is primarily relevant to export references and links of different types to be able to import the exported file again.

For details on using the options in the 'Insert References / Data Containers' group, refer to the **Insert References - Data Source Outbound** topic and the **Insert Data Container - Data Source Outbound** topic.



STEP Workflow Task Info - Data Source Outbound

The option to map STEP Workflow Task Info is only available when the object type selected for the export is Product. This option reports on a product's existence and status in the selected workflow(s) at the time of the export. A number of aspects are available to give access to different workflow information. Each aspect must be mapped individually.



Note: This same information can be exported via STEPXML which requires no mapping and includes all aspects and all workflows. STEPXML output can be used to further explain the output generated with a format that requires mapping. For more on STEPXML output, refer to the **Exporting Workflow Statuses** topic in the **Workflows** documentation.

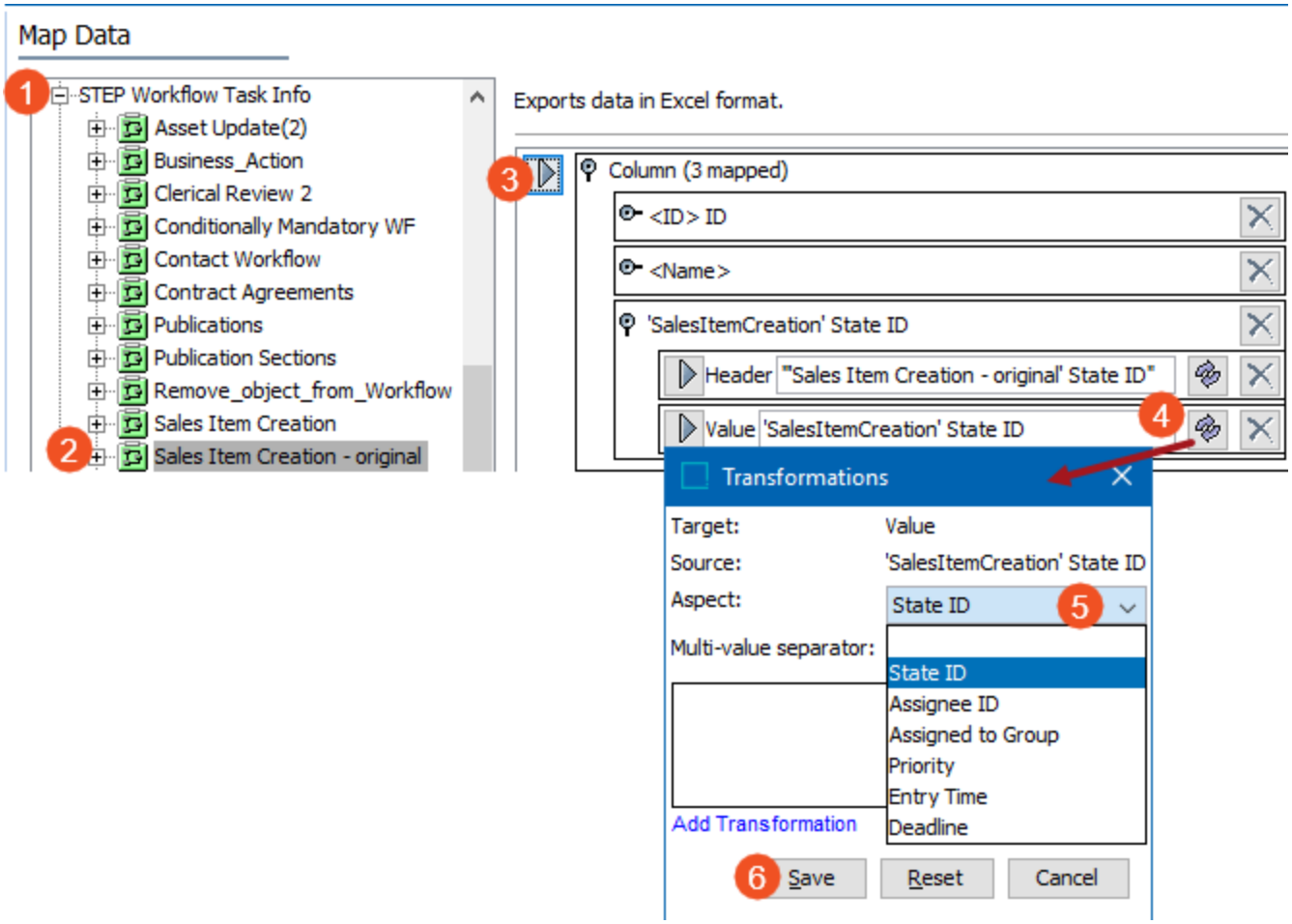
The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

For more information about the additional wizard steps, refer to the **Creating a Data Export** topic or the **Creating an Outbound Integration Endpoint** topic.

After mapping, most output options can be altered using transformations. For more information, refer to the **Outbound Map Data - Transform** topic.

Mapping STEP Workflow Task Info

1. Select the data to be exported for the output tool. For more information, refer to the **Data Exchange** topic.
2. On the Map Data step, in the left panel, expand the STEP Workflow Task Info node to show all the workflows.
3. Select one or more workflows and click the right arrow button ( ).



4. The default aspect is State ID at the time of the export. To output a different aspect for the time of the export, click the Transformation button ( ), and choose one of the following aspect options on the Transformations dialog:
   - **State ID** - outputs the IDs of the current states of the product.
   - **Assignee ID** - outputs current assignees.
   - **Assigned to Group** - outputs 'true' or 'false' based on the selection of a group as assignee.
   - **Priority** - is a legacy feature that has been replaced with Status Flag functionality.
   - **Entry Time** - outputs entry time of all current states, and records only one time for a parallel or cluster.
   - **Deadline** - when applicable, outputs deadline specified for the state.

   For more information, refer to the **Outbound Map Data - Transform** topic. Each aspect output is illustrated in the **Results** section below.
5. Complete any additional mappings and initiate the export.

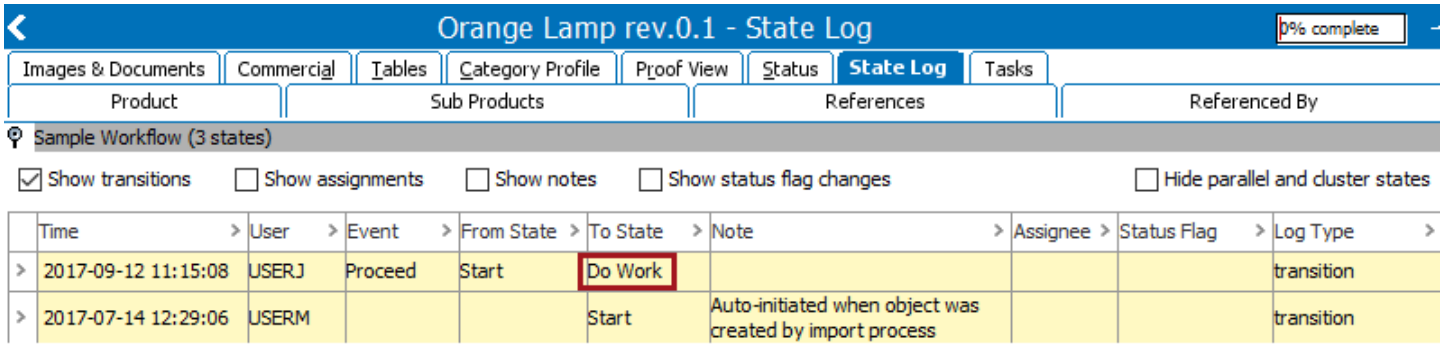
Results

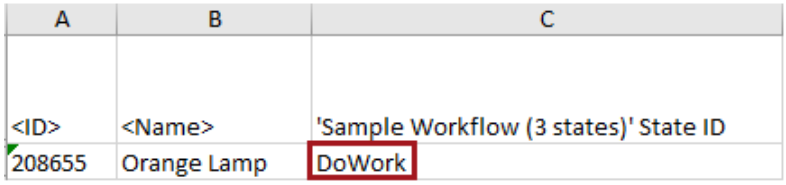
The images below include the product exported and the Excel result based on the selected aspect. Columns A and B show the ID and Name for the product. Column C shows the workflow task information. When the product has multiple values for the aspect, the data is displayed in a semi-colon separated list.

When multiple workflows are mapped, each is displayed it a single column as defined by the header.

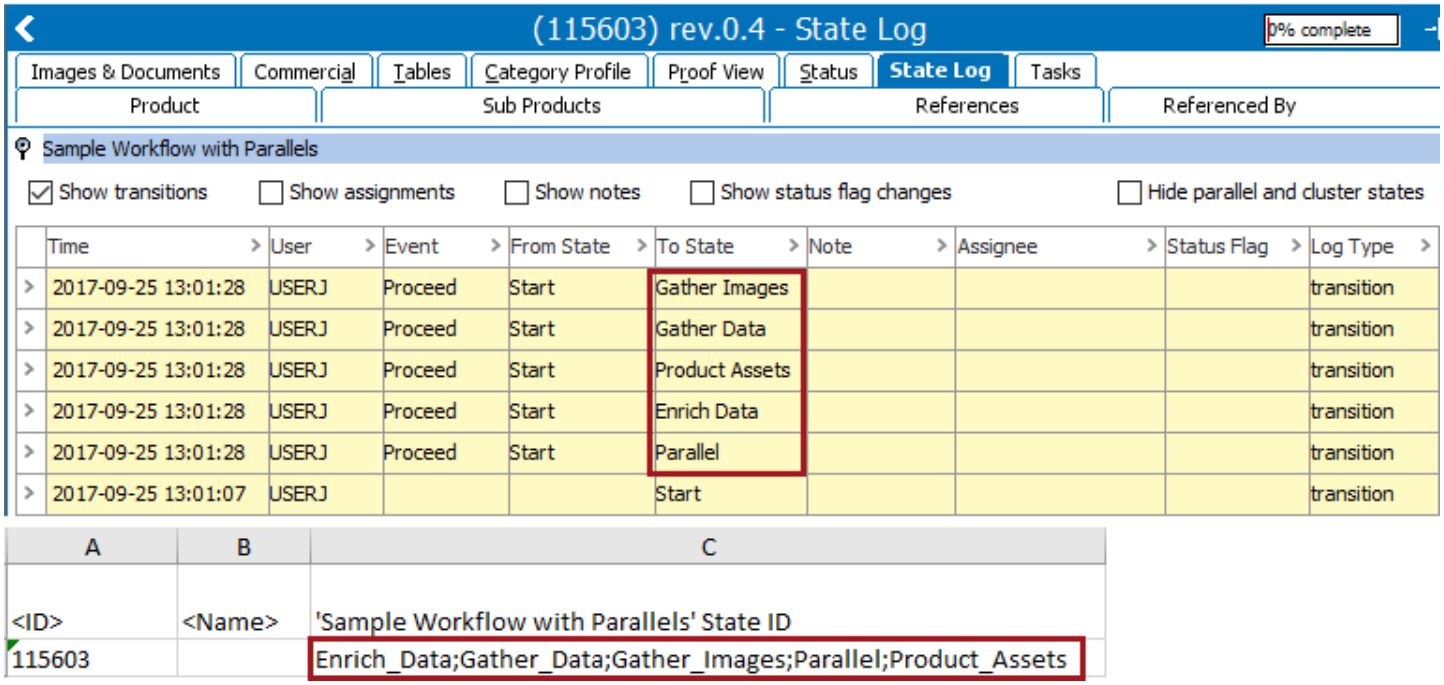
- **State ID** Data and Output

  When a product exists in a single workflow state for the workflow as shown below, the workflow state is displayed.





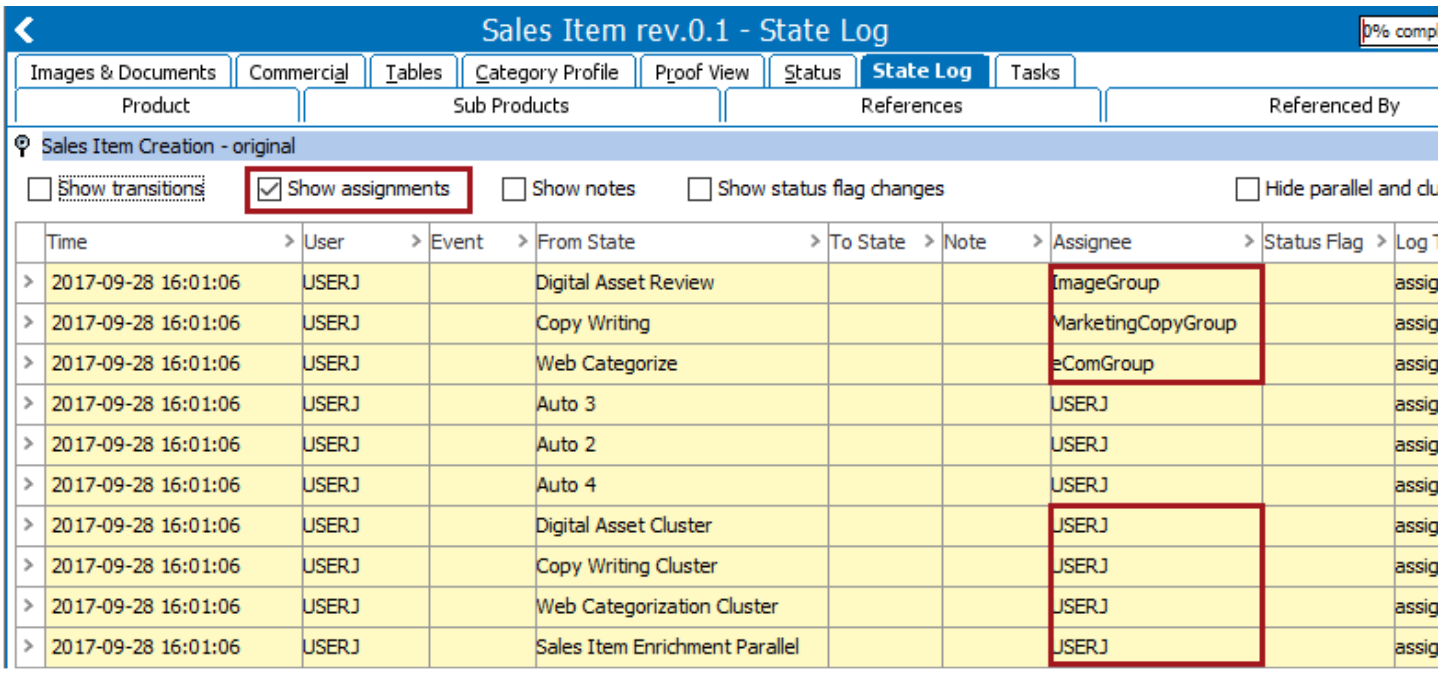
  When a product exists in multiple workflow states for the same workflow as shown below, all the relevant workflow states are displayed along with the Cluster and Parallel State IDs that the product exists within.



A	B	C
<ID>	<Name>	'Sample Workflow with Parallels' State ID
115603		Enrich_Data;Gather_Data;Gather_Images;Parallel;Product_Assets

- **Assignee ID** Data and Output

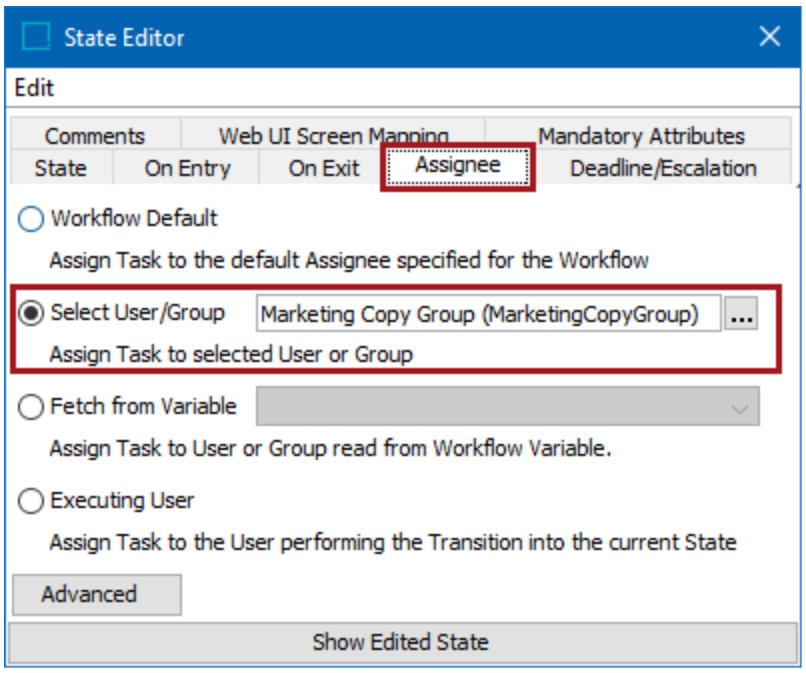
  When a workflow task has an assignee (set in the workflow State Editor on the Assignee tab), for the current states, the assignees are output. The following images show results for a workflow with a parallel that includes multiple clusters, so the product is assigned to multiple users at the same time.



A	B	C
<ID>	<Name>	'Sales Item Creation - original' Assignee ID
122822	Sales Item	MarketingCopyGroup;USERJ;USERJ;ImageGroup;USERJ;USERJ;eComGroup

- **Assigned to Group** Data and Output

  This allows you to verify if any groups are involved in the workflow. When a workflow task has a group as an assignee (set in the workflow State Editor on the Assignee tab), the output displays 'true.' If only single users are assigned, the output is 'false.'



A	B	C
<ID>	<Name>	'Sales Item Creation - original' Assigned to Group
122822	Sales Item	false;false;false;true;false;false;true

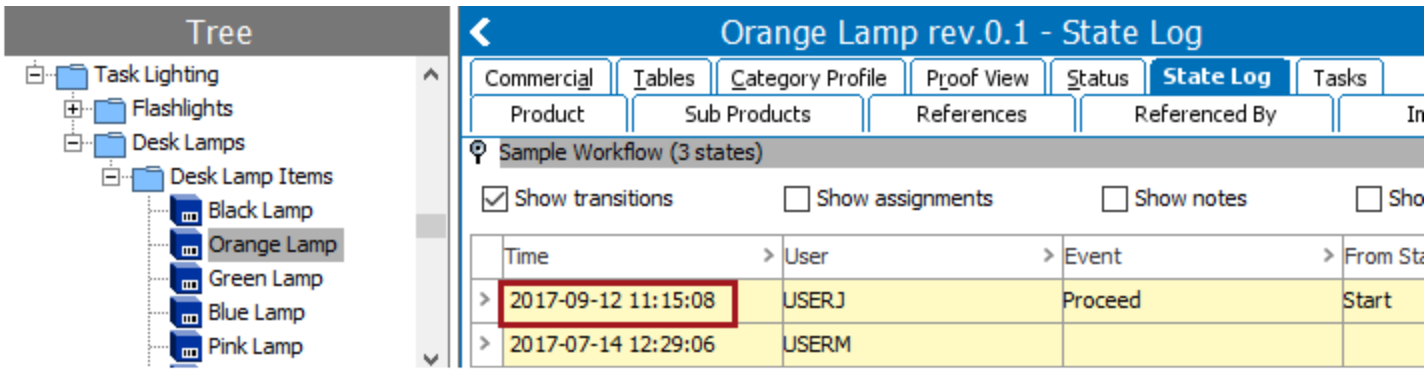
- **Priority** Data and Output

  The output includes only -1, which indicates the lowest priority. This legacy feature has been replaced with status flag functionality. For more information, refer to the **Status Flags** topics in the **Workflow** documentation.

A	B	C
<ID>	<Name>	'Sales Item Creation - original' Priority
122822	Sales Item	-1;-1;-1;-1;-1;-1;-1

- **Entry Time** Data and Output

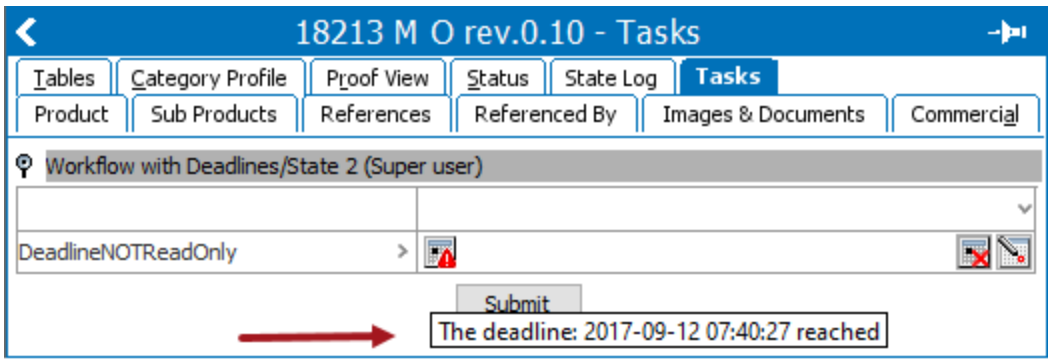
  When an object enters a workflow task state, the server time is recorded and can be output using this aspect. In the output, a single time is logged upon entry into a parallel or cluster, so the entry times do not correspond one-to-one with the times displayed on the State Log tab of the object for workflows with parallels and clusters.



A	B	C
		'Sample Workflow (3 states)'
<ID>	<Name>	Entry Time
208655	Orange Lamp	2017-09-12 11:15:08

- **Deadline** Data and Output

  When a deadline is set on a workflow using the Deadlines/Escalations tab in the Workflow Editor or the WorkflowDeadline component available in Views and Mappings, the date can be output using this aspect. For information on setting deadlines, refer to the **Deadlines and Escalations in Workflows** topic and **Views and Mappings for Workflows** in the **Workflows** documentation.



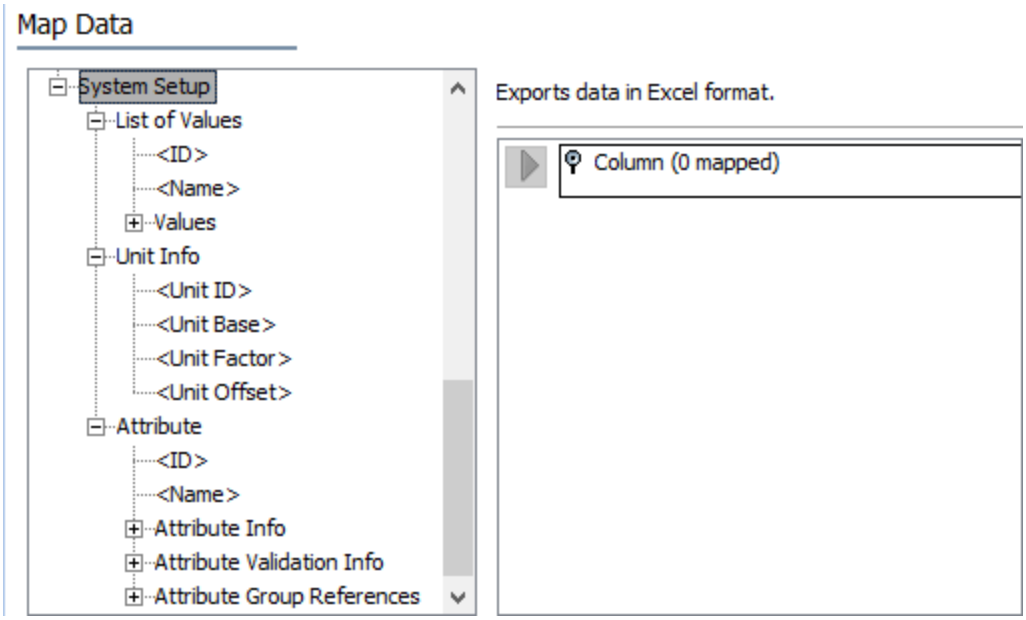
A	B	C
		'Workflow with Deadlines' Deadline
<ID>	<Name>	
18213	18213 M O	2017-09-12 07:40:27

System Setup - Data Source Outbound

Although displayed for all formats, the System Setup data source gives access to details on List of Values and Unit Info only when the Generic XML format is used. For examples using this data source, refer to:

- Attribute Links and LOVs Processing Instructions in Generic XML
- Attribute Links and Units Processing Instructions in Generic XML

The Attribute group within the System Setup data source is also available when exporting attributes, and is defined in the **Attribute Information - Data Source Outbound** topic.



The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

For more information about the additional wizard steps, refer to the **Creating a Data Export** topic or the **Creating an Outbound Integration Endpoint** topic.

After mapping, most output options can be altered using transformations. For more information, refer to the **Outbound Map Data - Transform** topic.

Outbound Map Data - Mapping Targets

Mapping targets, displayed in the right panel, indicate the order and details of the data being exported. The process to map data differs only slightly based on the data source selected.

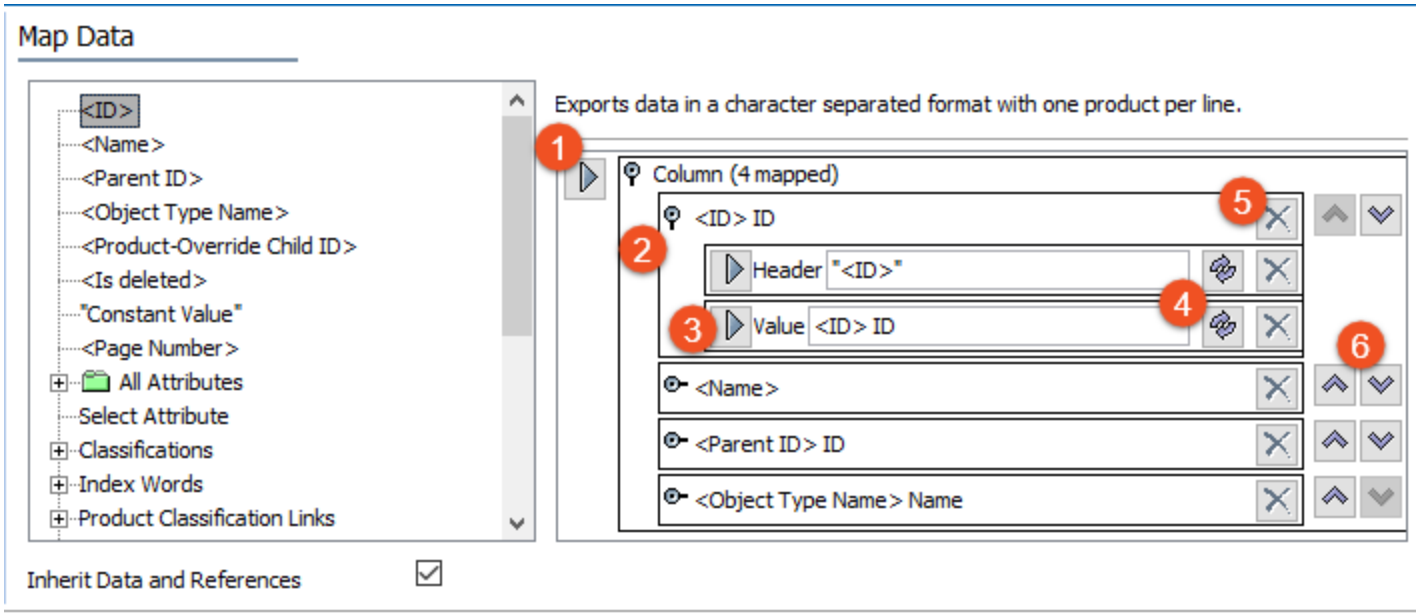
Generally, mapping data involves selecting a data source from the left panel and clicking the mapping button ( ) in the right panel. Based on the selected data source, the mapping rule appears immediately, or additional configuration is required. Ultimately, all data being exported is displayed as a mapping rule.

The user interface for the mapping step of the Export Manager tool and the outbound integration endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter as described in the **OIEP - Event-Based - Output Templates Flipper** topic or the **OIEP - Select Objects - Output Templates Flipper** topic.

For more information about the additional wizard steps, refer to **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

Tabular Format Targets

For tabular formats (as identified in the **Outbound Map Data Options** topic), the numbered options in the following image are available for creating mapping targets for outbound data.



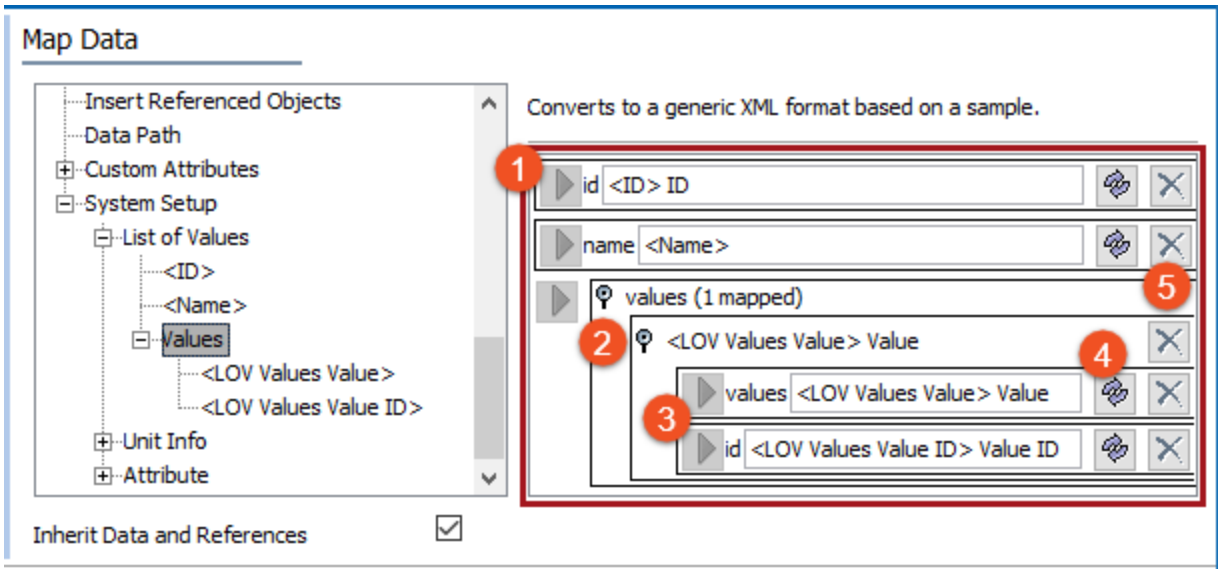
1. Mapping button assigns the selected data source to the selected mapping target.
2. A flipper is available when multiple mapping options are available for a single mapping target.
3. Mapping button assigns the selected data source to the selected mapping target within a flipper.
4. The transform button allows you to modify the STEP data prior to export.
5. The delete button removes a mapping target, eliminating the data from the only export only. This does not

affect the actual data in STEP.

6. Up and down buttons allow you to resort the mapping targets.

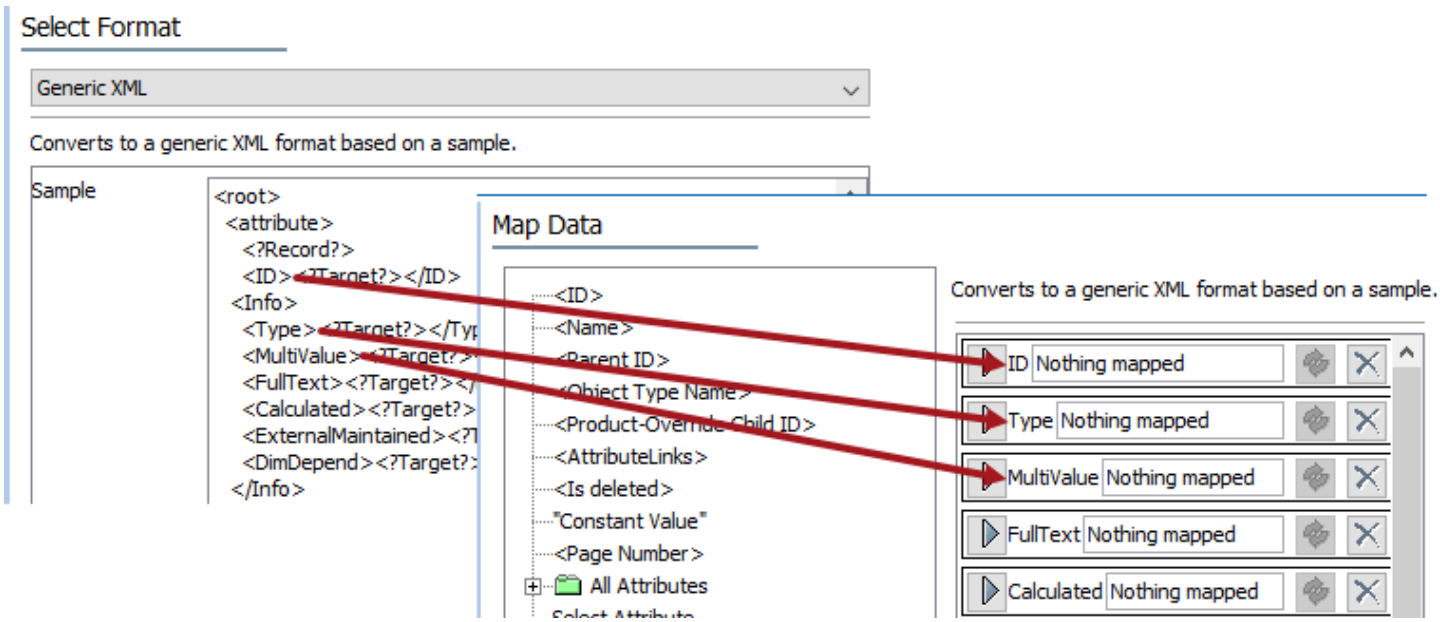
XML Format Targets

For XML-based formats (as identified in the **Outbound Map Data Options** topic), the numbered options in the following image are available for creating mapping targets for outbound data.



1. Mapping button assigns the selected data source to the selected mapping target.
2. A flipper is available when multiple mapping options are available for a single mapping target.
3. Mapping button assigns the selected data source to the selected mapping target within a flipper.
4. The transform button allows you to modify the STEP data prior to export.
5. The delete button removes a mapping target, eliminating the data from the only export only. This does not affect the actual data in STEP.

Additionally, for XML, the sample file displayed on the Select Format step determines the fields available for export in the Map Data step. To modify the order of the mapping targets, edit the sample file.



JSON Format Targets

Refer to the **Generic JSON Format** topic for configuration examples.

Outbound Map Data - Transform

During the Map Data process, transformations modify how data is displayed when it is exported from STEP. For example, with transformations, you can split one STEP data column into multiple values, append and prepend data to values, and perform search and replace.

Transformations are cumulative and are carried out in strict sequence order. That is, if you specify many transformations, each one takes place on the result of the prior transformation, not on the original data. Since there is no preview of the effect of transformation instructions, it is a good idea to implement one or two transformations at a time, save the configuration, and gradually build the complete set of transformations.

Transformations that are textual in nature cannot be applied to attribute values that are not textual.

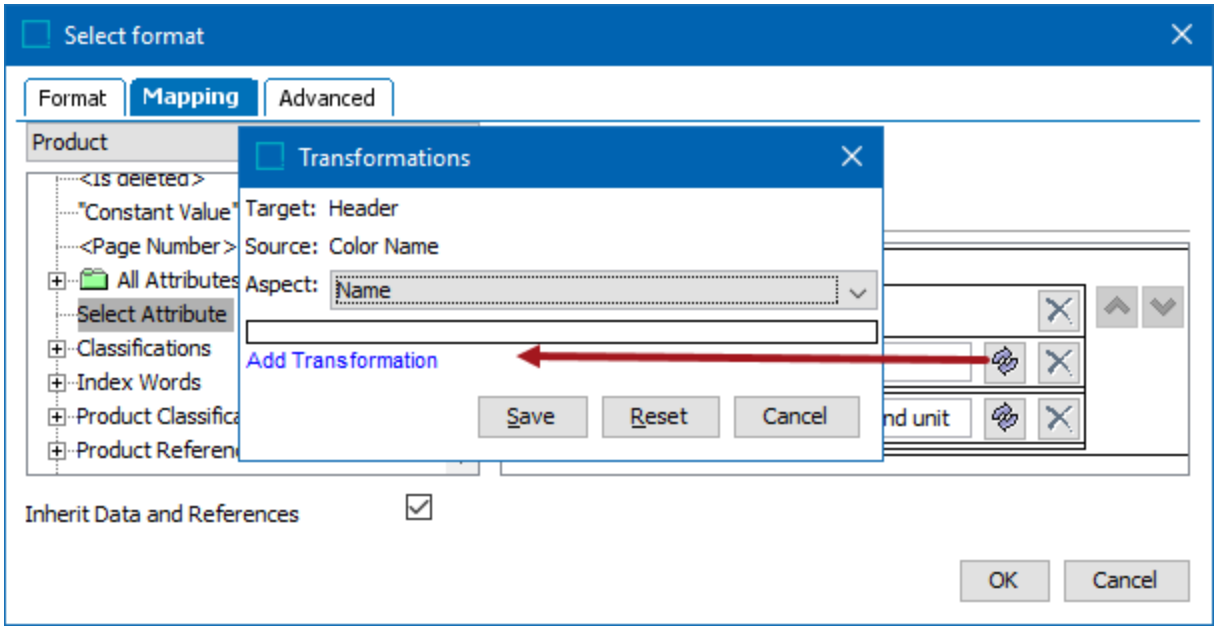
For examples of the most common transformations, refer to **Mapping Transformation Examples**.

To use transformations on attributes in STEP, or data in tables and publications, refer to **Attribute Transformations** in the **System Setup** documentation.

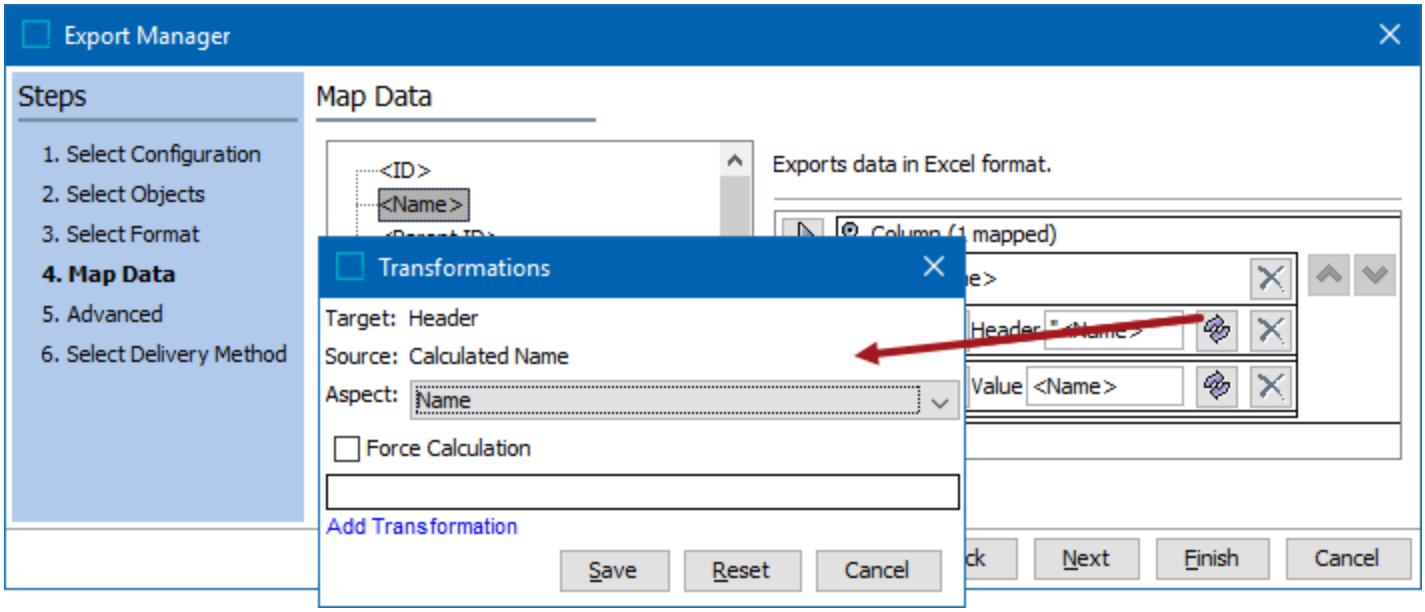
Transforming Data

The mapping step for Export Manager and OIEPs include the same options after clicking the **Transform** button.

In OIEPs, the transformations dialog is accessed from the OIEP editor Configuration tab > > Output Templates flipper > edit the Format column > select the Mapping tab.



In the Export Manager, the transformations dialog is accessed the Map Data step.



Based on the item being transformed, the following parameters are displayed on the Transformations dialog, and allow you to complete the transformation setup:

- **Aspect**
- **Data Container Separator**
- **Force Calculation** is only available when a calculated attribute has been selected. Checking this box ensures the value of the calculation will be resolved upon export. When unchecked, calculated values can still be generated via the **Include Calculated Attribute Values** option on the Advanced step.
- **Multi-Value Separator** text box
- **Add Transformation** link (options are shown below)

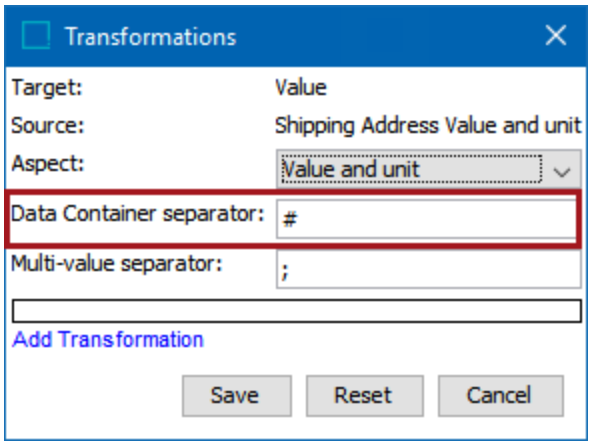
For details on each of the available transformation options, refer to the **Transformations** topic in the **Resource Materials** online help.

For more information about the additional wizard steps, refer to **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

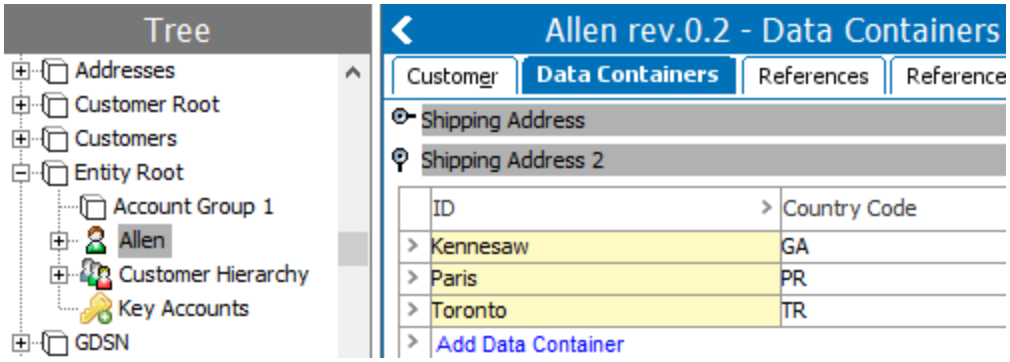
Data Container Separator - Transform Outbound

If a data container attribute is being mapped, the Transformations dialog will show a field for adding in a separator between data container attribute values. Any character entered in this field will be added to the data set that is exported. Inheritance is not applicable for data containers. Transformations are available for any format that requires mapping.

Important: 'Data Container separator' is only visible in the Transformations dialog for Entity object types with data container types that allow for multiple data containers.



In this example, the entity ‘Allen’ has 'Shipping Address 2' as a data container with the 'Country Code' attribute, which includes values. The 'Data Container separator' parameter is used to determine the text that is exported between the values of the data container attribute.

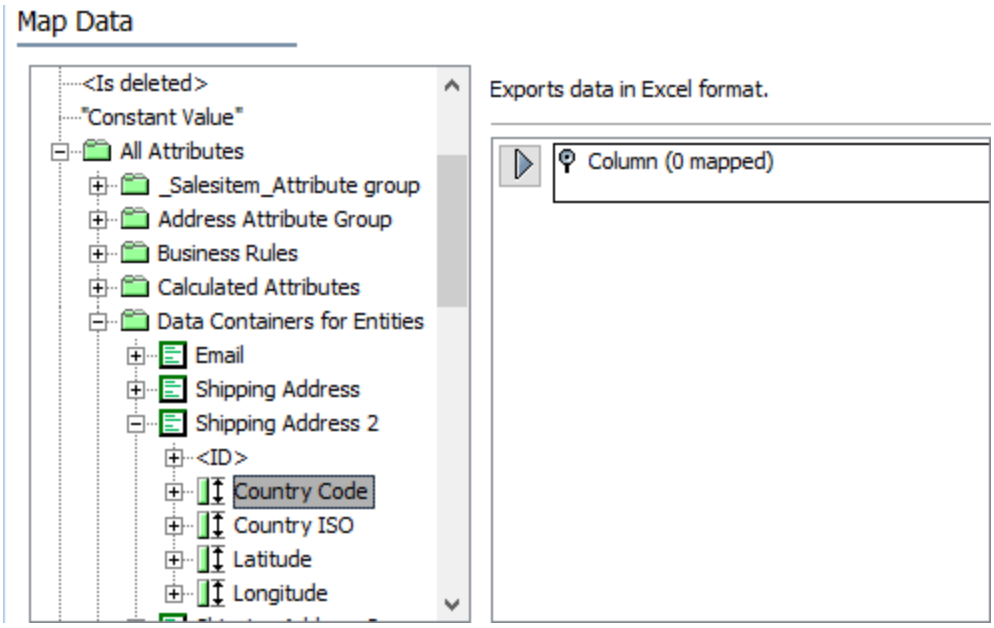


Configuration

1. In the output tool, choose Entity from the object type dropdown, and select the entity to be exported. For more information, refer to the **Data Exchange** topic.
2. On the Map Data step, in the left panel, expand the **All Attributes** data source, then expand the **Data**

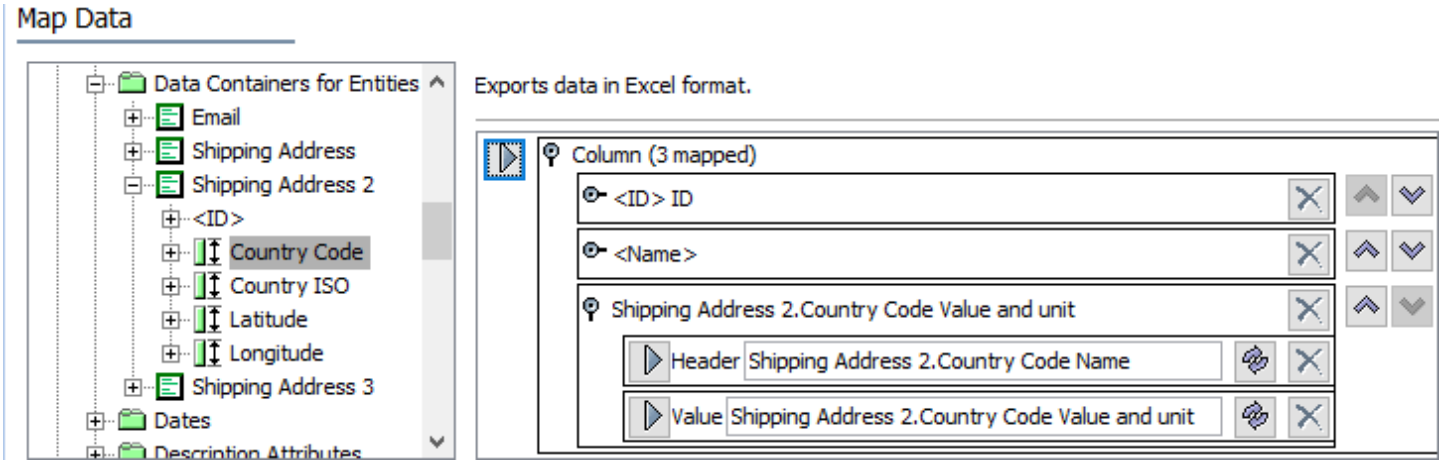
Containers for Entities group.

3. Expand the required data container, and then select the required attribute.



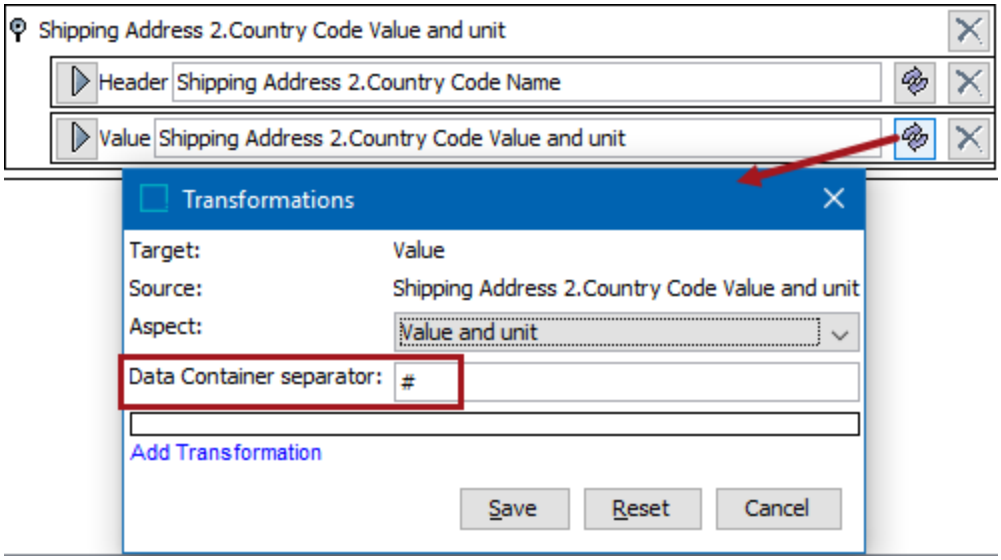
Important: The **All Attributes** data source can also be used to map the Data Container itself and get all of the attribute values associated with the data container. The **Select Attribute** data source does not give access to the data container type, nor the attributes.

4. In the right panel, click the mapping arrow ( ) to display the attribute as an item for export and open the flipper.



5. On the Value element, click the transformation button ( ) to display the Transformations dialog. By default, the '#' is displayed in the **Data Container separator** parameter. If required, specify a different separator.

Important: When exporting a tabular format, select a different character from the one used to delimit columns.



6. If necessary, click the **Reset** button to revert back to the dialog default settings.
7. Click the **Save** button.
8. If necessary, click the Transformation button ( ) again to apply any additional transformations, which can change the output without changing the original data. Refer to the **Outbound Map Data - Transform** topic.
9. Complete any additional mappings and initiate the export.

Results

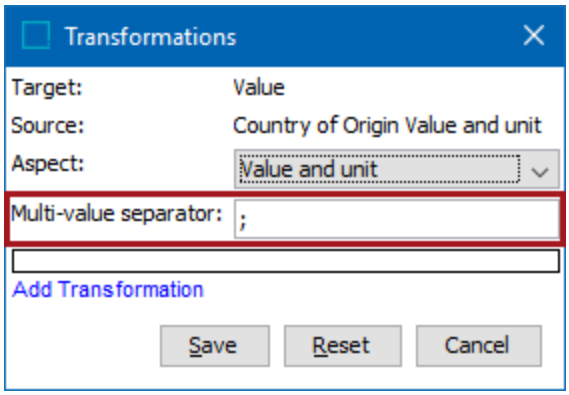
Using the data and configuration in the example above, the output of the entity object includes the entity ID and name, as well as the values in the 'Shipping Address 2' data container, separated by the default separator.

	A	B	C
1	<ID>	<Name>	Shipping Address 2.Country Code
2	CUS_114563	Allen	GA#TR#PR

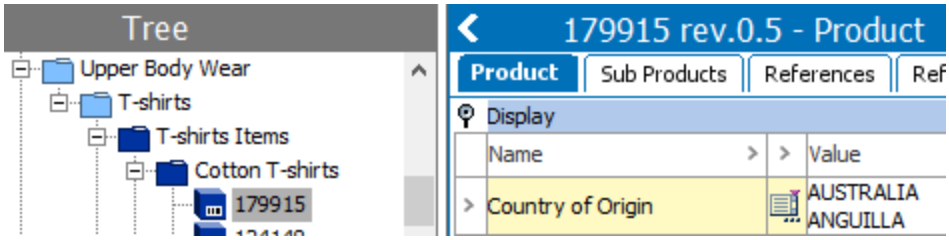
Multi-Value Separator - Transform Outbound

When multivalued attributes are mapped, use a transformation to specify a character to be used as value delimiter. Transformations are available for any format that requires mapping.

Important: 'Multi-value separator' is only visible in the Transformations dialog for attributes with the Multi Valued attribute validation set to Yes.

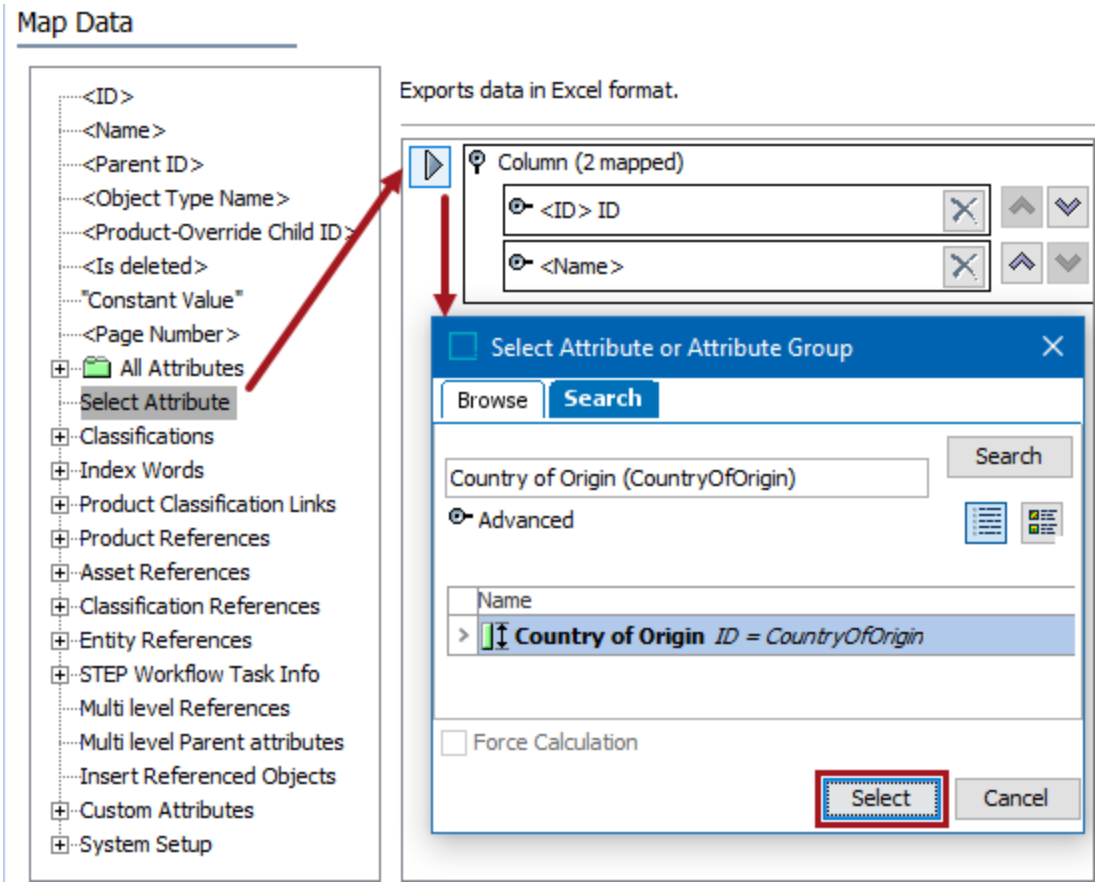


In this example, the product ‘179915’ has a multivalued attribute 'Country Of Origin', which includes multiple values. The 'Multi-value separator' parameter is used to determine the text that is exported between the values.



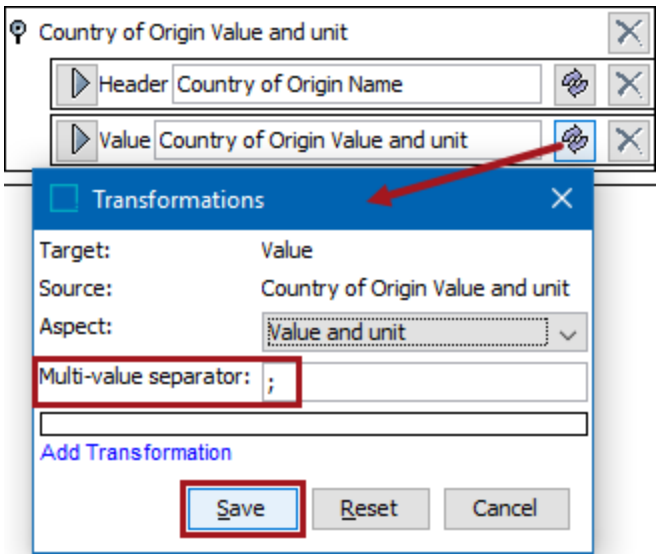
Configuration

1. In the output tool select the data to be exported. For more information, refer to the **Data Exchange** topic.
2. On the Map Data step, in the left panel, click the **Select Attribute** data source.
3. In the right panel, click the mapping arrow ( ) to display the Select Attribute or Attribute Group dialog. Use the **Browse** or **Search** option to find and select the 'Country of Origin' attribute. Click the **Select** button.



4. Open the flipper for the mapped attribute.
5. On the Value element, click the transformation button ( ) to display the Transformations dialog. By default, the '#' is displayed in the **Multi-value separator** parameter. If required, specify a different separator.

Important: When exporting a tabular format, select a different character from the one used to delimit columns.



6. If necessary, click the **Reset** button to revert back to the dialog default settings.
7. Click the **Save** button.
8. If necessary, click the Transformation button ( ) again to apply any additional transformations, which can change the output without changing the original data. Refer to the **Outbound Map Data - Transform** topic.
9. Complete any additional mappings and initiate the export.

Results

Using the data and configuration in the example above, the output of the product includes the ID and name, as well as the values in the 'Country of Origin' attribute, separated by the selected separator.

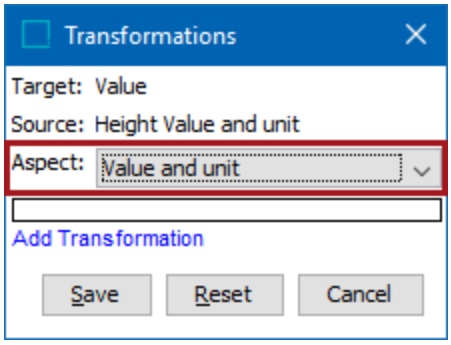
	A	B	C
1	<ID>	<Name>	Country of Origin
2	179915	179915	AUSTRALIA;ANGUILLA

Aspect - Transform Outbound

The Aspect determines the data element that is exported, for example, name, ID, value, value and unit, LOV ID, and metadata. (All data elements are listed in the table below.) Using aspects allows you to generate multiple columns of different data elements for the same attribute. The options available for an aspect are based on the data source selected. Transformations are available for any format that requires mapping.

When mapping attributes and values, both default to the 'Value and unit' aspect but not all attributes or values include units. It is recommended to change the attribute ID to the 'Name' or 'ID' aspect instead of leaving the default 'Value and unit'. Otherwise, 'Value and unit' data is repeated for both the attribute ID and the value, which requires editing the template again after export.

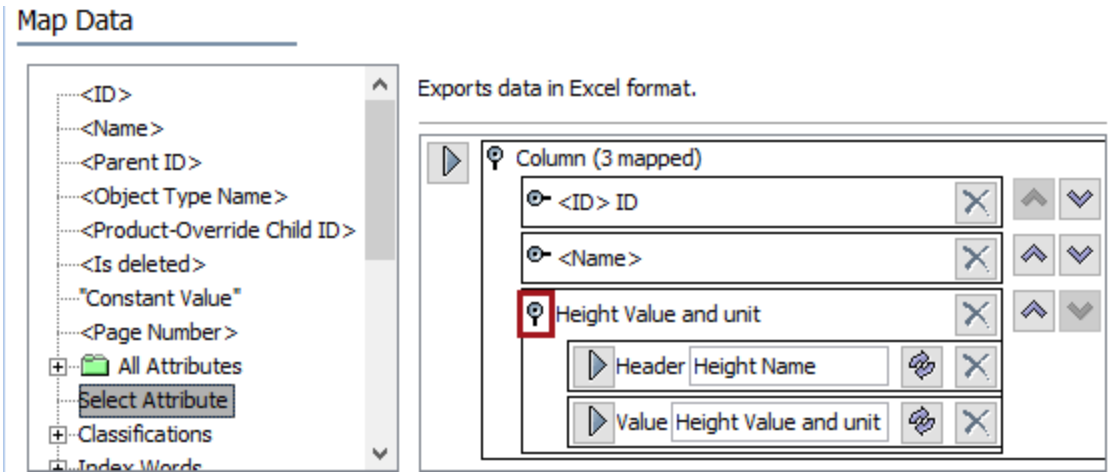
Important: 'Aspect' is only visible in the **Transformations** dialog for the data sources included in the **Data Sources with Aspect** below.



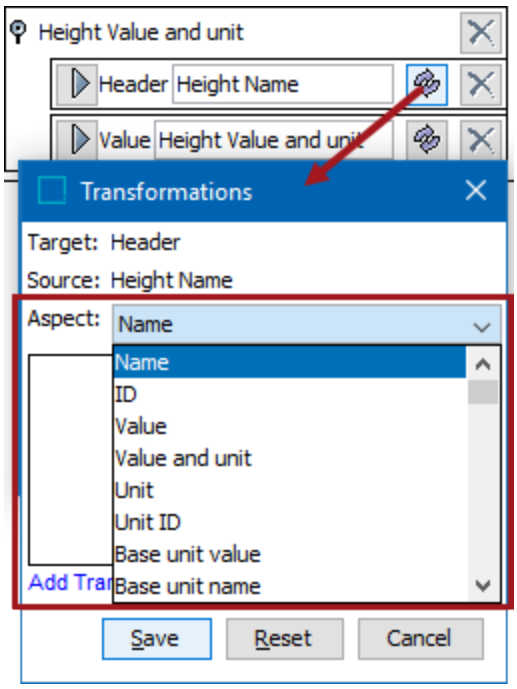
For a general example of mapping aspects, refer to the **Header and Value Aspects Example**. For specific examples of the available aspects, refer to the topics referenced in the **Aspect Data Elements Options** table below.

Configuration

1. In the output tool, select the data to be exported. For more information, refer to the **Data Exchange** topic.
2. On the Map Data step, in the left panel, select the data source. For more information, refer to the **Outbound Map Data - Data Source** topic.
3. In the right panel, click the mapping arrow ( ) and provide any required selections to add the data source as a mapping target. For more information, refer to the **Outbound Map Data - Mapping Targets** topic.
4. Open the flipper for the mapping target.



5. On the Header or Value element, click the **Transformation** button ( ) to display the Transformations dialog with the Aspect parameter.



6. Select an Aspect from the dropdown and click the **Save** button. For details on the options, refer to the topics referenced in the **Aspect Data Elements Options** table below.
7. If necessary, click the **Transformation** button ( ) again to apply any additional transformations, which can change the output without changing the original data. Refer to the **Outbound Map Data - Transform** topic.
8. Complete any additional mappings and initiate the export.

Aspect Data Element Options

The following options are available in the Aspect dropdown, and will vary based on the selected data source.

Aspect Data Element	Description
Attribute Meta Data [*attribute name*]	Extracts metadata (Description) attributes that are available on the mapped attribute and are displayed with the prefix 'Attribute Meta Data,' as described in **Attribute Meta Data Aspect**.
Attribute Type	Extracts the Validation Base Type of the attribute, as described in **Attribute Type Aspect**.
Base Unit ID	Extracts the ID of the base unit that is associated with the unit of the product attribute value, as described in **Base Unit ID, Name, and Value Aspects**.
Base Unit Name	Extracts the name of the base unit that is associated with the unit of the object attribute value, as described in **Base Unit ID, Name, and Value Aspects**.
Base Unit Value	Extracts the attribute value according to the base unit. This aspect can be applied when established associations exist between units, and where one unit is the base unit, as described in **Base Unit ID, Name, and Value Aspects**.
Closest Attribute Link Meta Data [*attribute name*]	For BMEcat 1.2 and BMEcat 2005 formats only, extracts Product Attribute Link metadata attribute values, as described in **Closest Attribute Link Meta Data Aspect**. The 'closest' value is defined as the first value encountered, either explicit or inherited, while moving up the hierarchy from the product being exported.
Dimension Dependencies	Extracts the names of the dimension dependencies on the selected attribute, as described in **Dimension Dependencies Aspects**.
[*name*] Dimension	Extracts the ID of the dimension point selected by name for the selected attribute based on the exported context, as described in **Dimension Dependencies Aspects**.
ETIM FVALUE Details	For use with the ETIM IXF format and the 'ETIM Value Details Attributes' group. Requires that FValue_Details have been imported via the ETIM IXF format. Allows the values from these attributes to be output in the FVALUE_DETAILS tag, as defined in **ETIM IXF Format**

Aspect Data Element	Description
File name by asset ID	Extracts multiple files in a .ZIP file including the ID of the digital media files associated with the referenced assets (which can be converted using the conversion pipeline) and the selected export format file.
File name by asset Name	Extracts multiple files in a .ZIP file including the name of the digital media files associated with the referenced assets (which can be converted using the conversion pipeline) and the selected export format file.
ID	Extracts the ID of the object. For example, if an object has the ID '1234' and the name 'Weight,' then '1234' is extracted for object.
Is Deleted	For event-based OIEP only, outputs a value to indicate if the target for the reference has been deleted, as described in **Is Deleted Aspect**.
Key: [*key name*]	Extracts the key specified for the object, as described in **Key Aspect**.
LOV ID	Extracts the ID of the LOV used by the mapped attribute, as described in **LOV ID Aspect**. For example, mapping the ItemColor attribute, which uses the ColorLOV, outputs the ID of ColorLOV.
LOV Value-ID	Extracts the Value ID of the LOV value used by the mapped attribute, as described in **LOV Value-ID Aspect**. For example, mapping the ItemColor attribute, which uses the ColorLOV, outputs the Value ID of value selected from ColorLOV on the attribute.
LOV Value-ID or Value	When mapping Attributes or an Attribute Group, this aspect extracts the LOV Value-ID when present, otherwise it will extract the values in the attributes, as described in **LOV Value-ID or Value Aspect**.
MIME Type	Extracts the MIME Type of an asset.

Aspect Data Element	Description
	For example, when an asset with the MIME Type of 'image/png' is referenced to an object. Mapping the Asset Reference data source and using the MIME Type aspect, it extracts the MIME type of 'image/png' for the asset reference. For more information, refer to the **MIME Types** topic of the **System Setup** documentation.
Name	Extracts the name of the object. For example, if an attribute has the ID '1234' and the name 'Weight,' then the word 'Weight' is extracted per object.
Unit	Extracts the name of the unit associated with the object attribute value, but not the actual value, as described in **Unit Aspect**.
Unit ID	Extracts the ID of the unit associated with the object attribute value, but not the actual value, as described in **Unit ID Aspect**.
Value	Extracts the attribute value, as described in **Value Aspect**. If the attribute has units assigned to it and the values have units assigned to them, the units are not extracted.
Value and unit	Extracts the object attribute value and the related unit (if any), as described in **Value and Unit Aspect**.
Value and unit Meta Data [*attribute name(ID)*]	Extracts the value and unit (if any) on the selected metadata attribute that exists on the mapped object, as described in **Value and Unit Aspect Meta Data**. For example, if the metadata attribute 'ETIM description' is valid on the ColorsListLOV object, the mapping a product attribute that uses ColorListLOV and using this aspect exports the value and unit on that metadata attribute.

Data Sources with Aspect

The following outbound data source options include the Aspect parameter on the Map Data step:

- Asset References - Data Source Outbound
- Attributes (and Data Containers) - Data Source Outbound

- Classification References - Data Source Outbound
- Classifications - Data Source Outbound
- Constant Value - Data Source Outbound
- Entity References - Data Source Outbound
- ID or Key - Data Source Outbound
- Index Words - Data Source Outbound
- Insert Referenced Objects - Data Source Outbound
- Multi Level Parent Attributes - Data Source Outbound
- Multi Level References - Data Source Outbound
- Object Type Name - Data Source Outbound
- Page Number - Data Source Outbound
- Parent ID - Data Source Outbound
- Product Classification Links - Data Source Outbound
- Product References - Data Source Outbound
- STEP Workflow Task Info - Data Source Outbound
- System Setup - Data Source Outbound

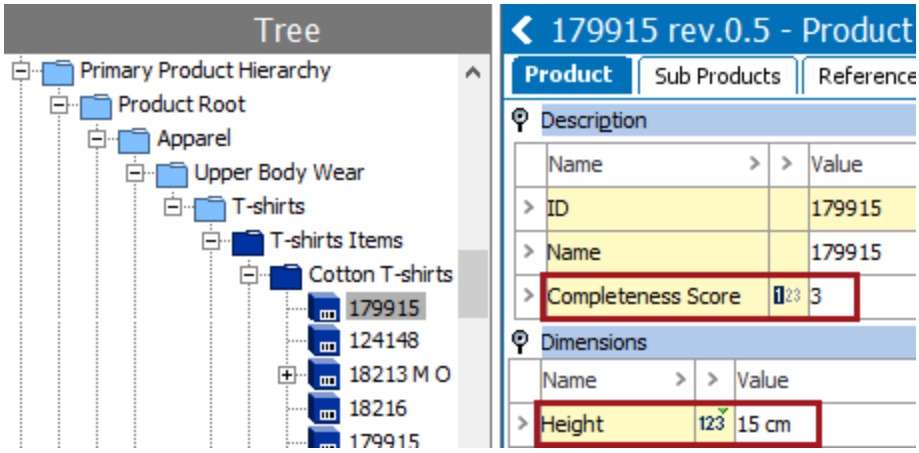
Attribute Meta Data Aspect

Metadata (description) attributes that are available on the mapped attribute are displayed with the prefix 'Attribute Meta Data.'

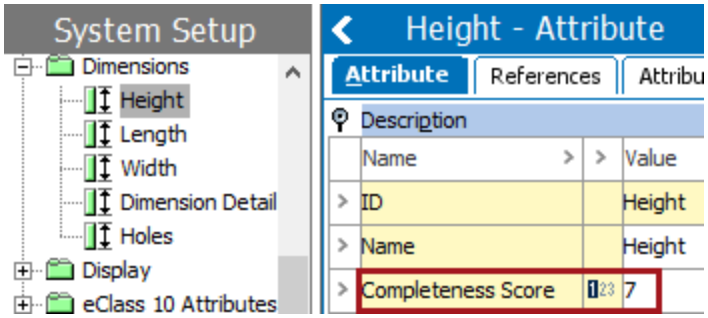
For general information on setting a Transformation aspect, refer to the **Aspect - Transform Outbound** topic.

For example, the description attribute 'Completeness Score' is valid on attributes and on products.

In the Tree, the product 179915 has the metadata Completeness Score attribute with a value of 3, which indicates that the product is 30 percent 'complete' when measured against the company's data standards. The Height attribute has a value of 15.



In System Setup, the 'Height' attribute has the metadata Completeness Score attribute with a value of 7, which indicates that the attribute is 70 percent 'complete' when measured against the company's configuration standards.

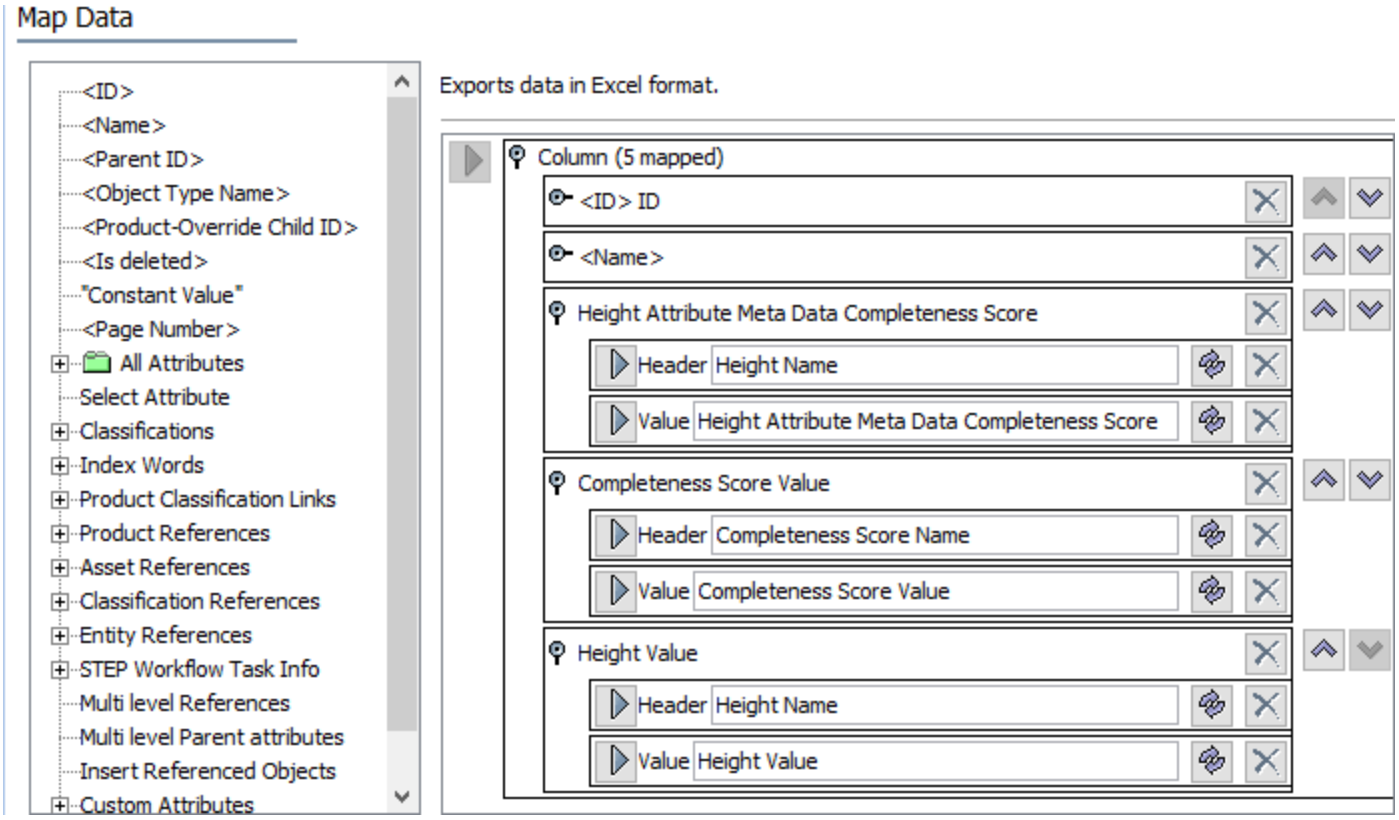


Configuration

In addition to the ID and name of the product, the values identified above are mapped using the Select Attribute data source and the specified aspect.

Note: The name of the selected aspect is displayed after the name of the mapped attribute.

- Height attribute with the value aspect of 'Attribute Meta Data Completeness Score.'
- Completeness Score attribute with the value aspect of 'Value.'
- Height attribute with the value aspect of 'Value.'



Results

Product 179915 is exported with the following output:

- ID is 179915.
- Name is 179915.
- Height attribute with the value aspect of 'Attribute Meta Data Completeness Score' is 7.
- Completeness Score attribute with the value aspect of 'Value' is 3.
- Height attribute with the value aspect of 'Value' aspect is 15.

	A	B	C	D	E
1	<ID>	<Name>	Height	Completeness Score	Height
2	179915	179915	7	3	15

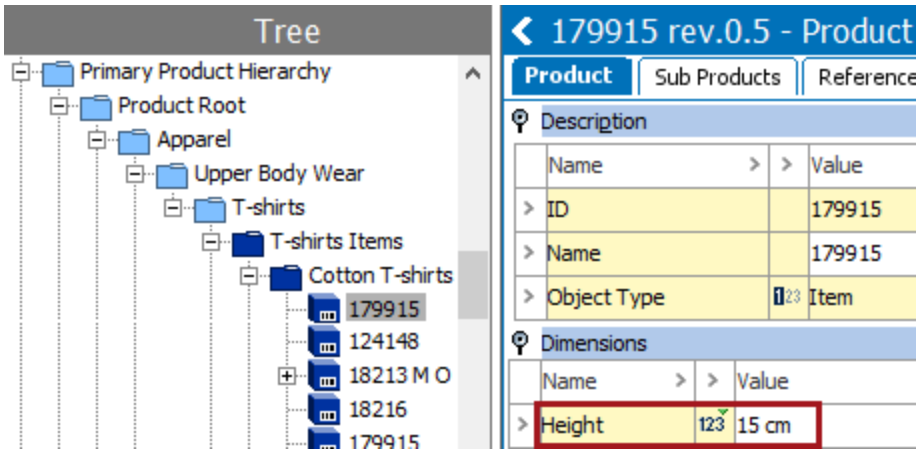
Attribute Type Aspect

The Validation Base Type of an attribute can be extracted using the 'Attribute Type' aspect. For more information on validation base types, refer to the **Validation Rules** topic in the **System Setup**.

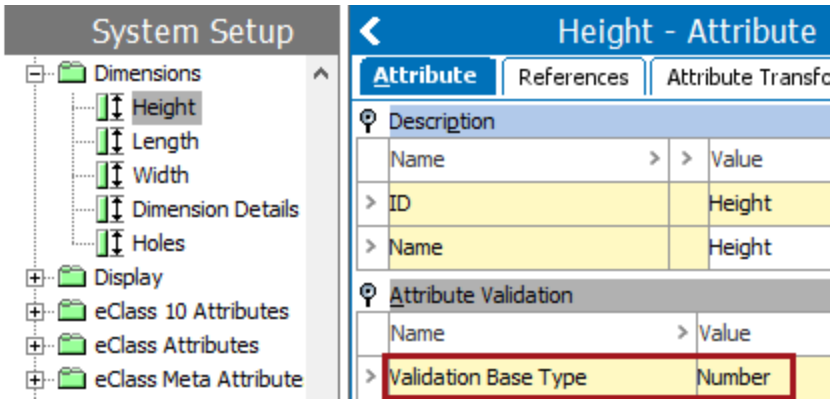
For general information on setting a Transformation aspect, refer to the **Aspect - Transform Outbound** topic.

For example, the attribute 'Height' is valid on products.

In the Tree, the product 179915 has the value of 15 for the Height attribute.



In System Setup, the 'Height' attribute has a Validation Base Type of Number.

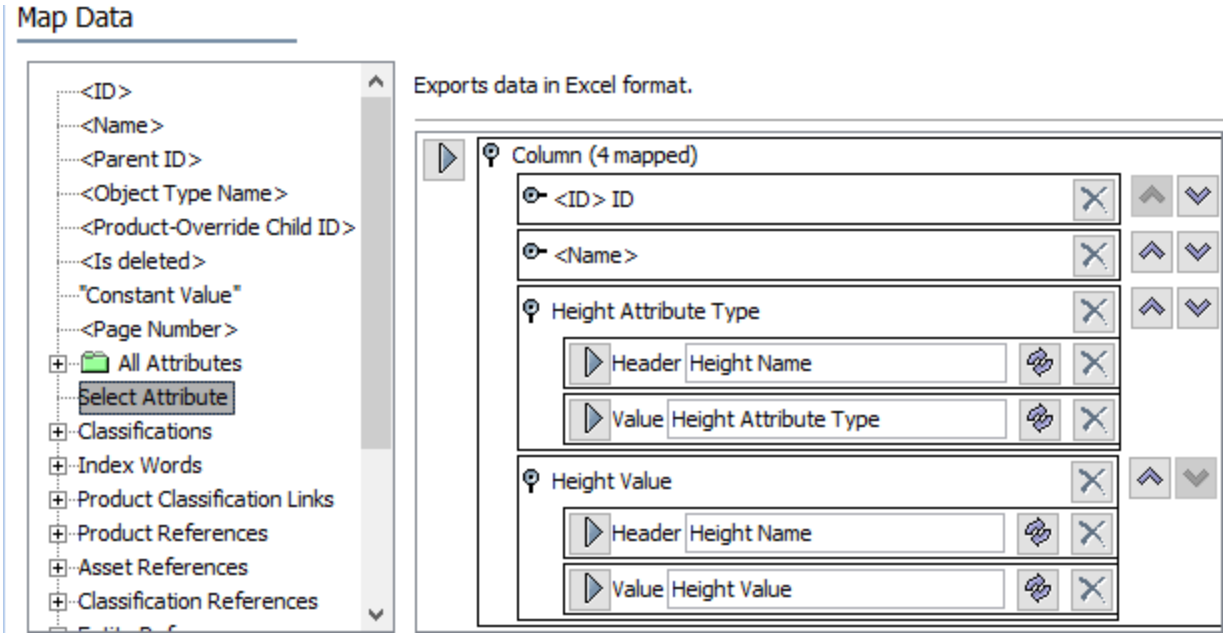


Configuration

In addition to the ID and name of the product, the values identified above are mapped using the Select Attribute data source and the specified aspect.

Note: The name of the selected aspect is displayed after the name of the mapped attribute.

- Height attribute is mapped with the aspect of 'Attribute Type.'
- Height attribute with the value aspect of 'Value.'



Results

Product 179915 is exported with the following output:

- ID is 179915.
- Name is 179915.
- Height attribute with the value aspect of 'Attribute Type' is 'number.'
- Height attribute with the value aspect of 'Value' aspect is 15.

	A	B	C	D
1	<ID>	<Name>	Height	Height
2	179915	179915	number	15

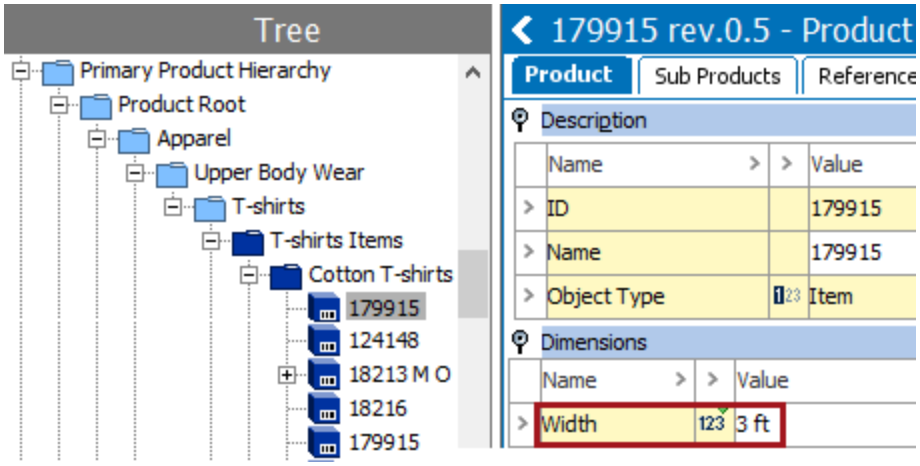
Base Unit ID, Name, and Value Aspects

Data for attributes that include a unit can be extracted using the following base unit aspects:

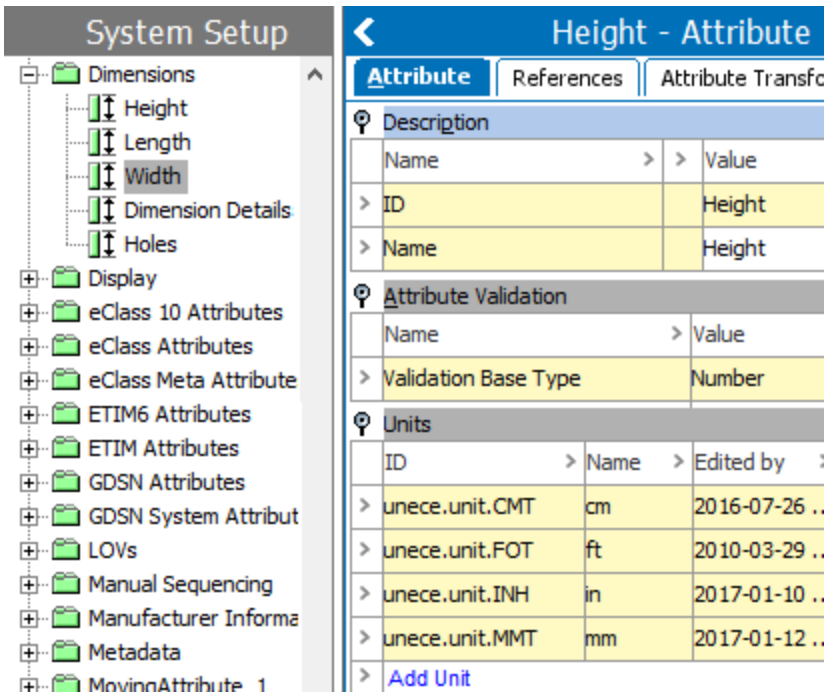
- **Base Unit ID** - Extracts the ID of the base unit that is associated with the unit of the attribute value.
- **Base Unit Name** - Extracts the name of the base unit that is associated with the unit of the attribute value.
- **Base Unit Value** - Extracts the attribute value according to the base unit. This aspect can be applied when established associations exist between units, and where one unit is the base unit.

For general information on setting a Transformation aspect, refer to the **Aspect - Transform Outbound** topic.

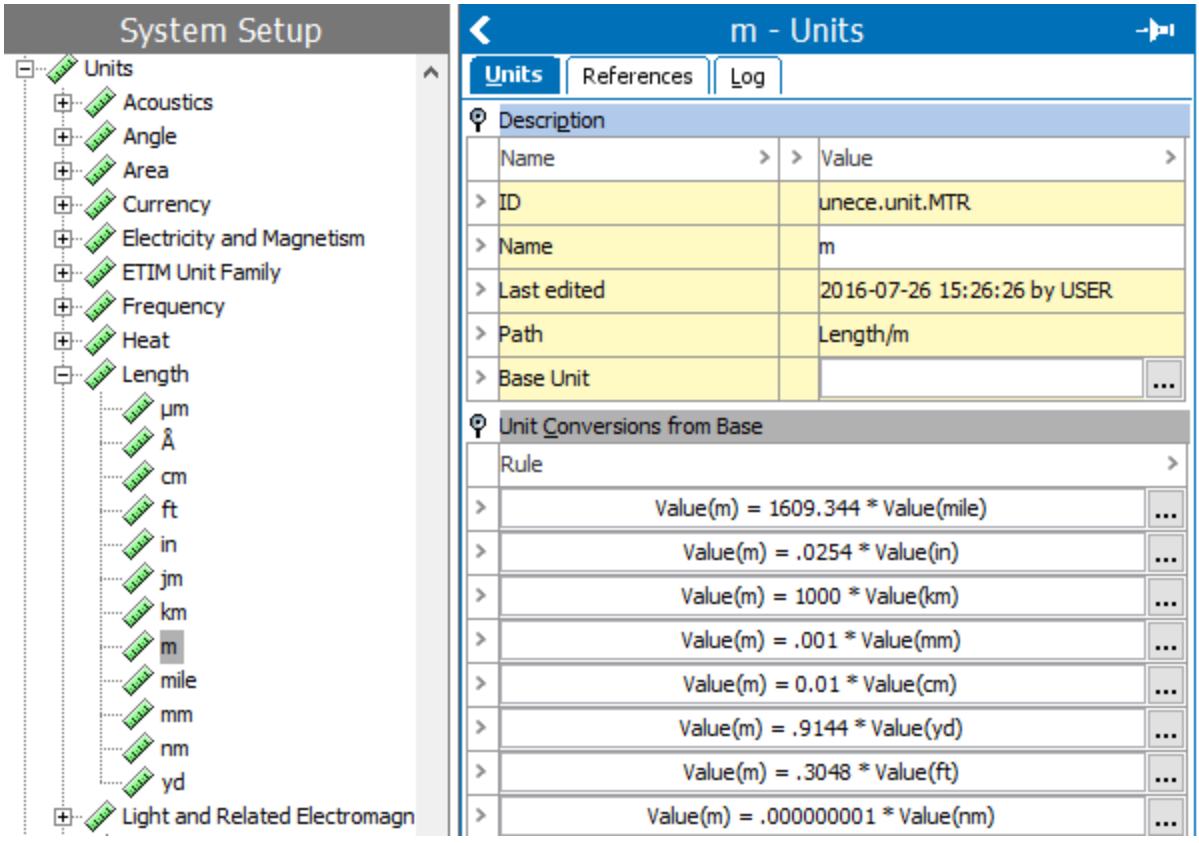
For example, for the Width attribute on a product, the value '3' is associated with the unit 'ft.'



The Width attribute has a Number validation base type, and allows units of cm (centimeters), mm (millimeters), in (inches), and ft (feet).



Each of the units is associated with the same base unit of m (meters) and include a conversion factor. For example, feet are multiplied by .3048 to get meters. The base unit of meters includes all required conversion rules. For more information on base units, refer to the **Units** topic in the **System Setup**.

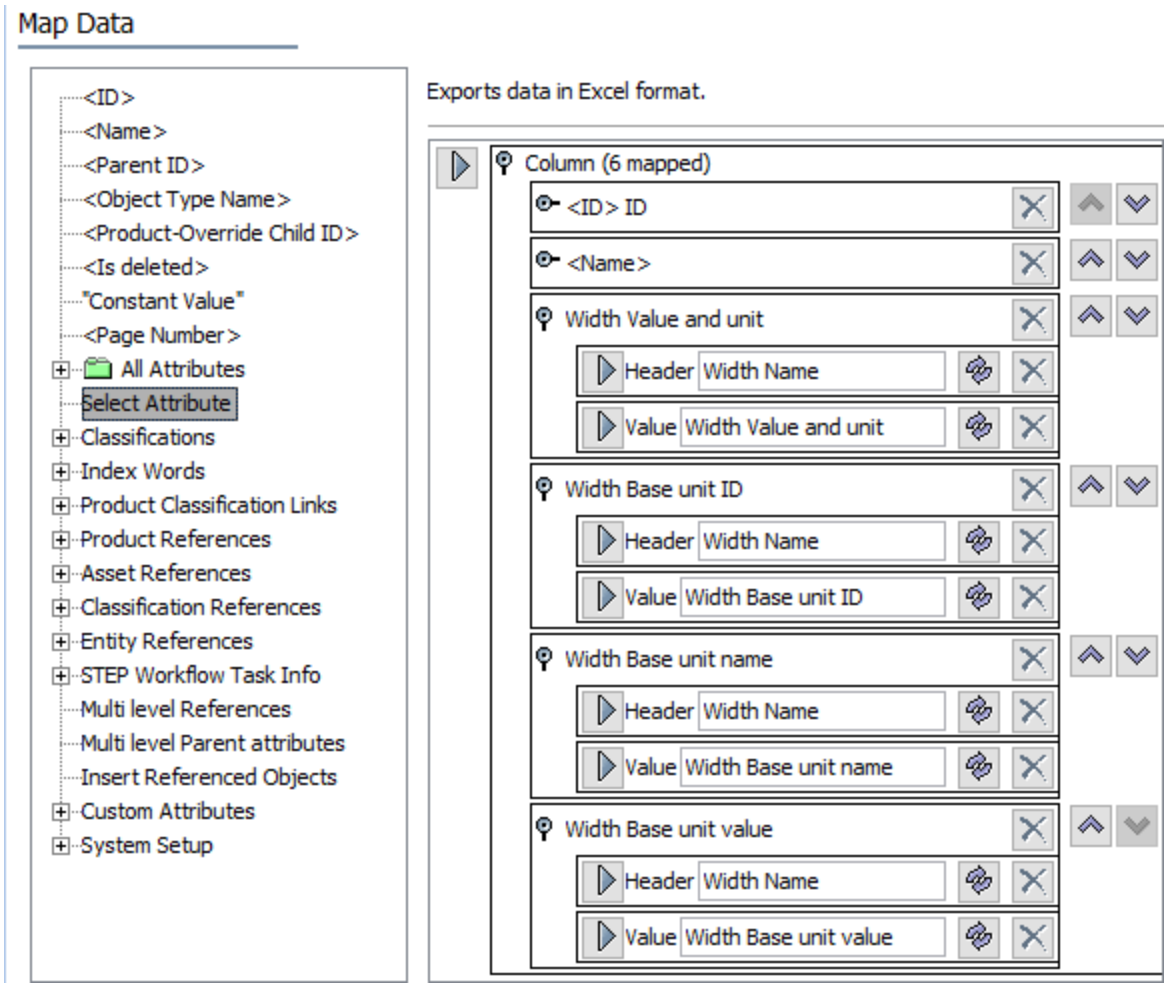


Configuration

In addition to the ID and name of the product, the values identified above are mapped using the Select Attribute data source and the specified aspect.

Note: The name of the selected aspect is displayed after the name of the mapped attribute.

- Width attribute is mapped with the aspect of 'Value and unit.'
- Width attribute is mapped with the aspect of 'Base Unit ID.'
- Width attribute is mapped with the aspect of 'Base Unit Name.'
- Width attribute is mapped with the aspect of 'Base Unit Value.'



Results

Product 179915 is exported with the following output:

- ID is 179915.
- Name is 179915.
- Width attribute with the value aspect of 'Value' is '3 ft.'
- Width attribute with the value aspect of 'Base Unit ID' aspect is 'unece.unit.MTR.' Since attribute value uses the 'ft' unit, and it is associated with the meters base unit, the meters ID is used.
- Width attribute with the value aspect of 'Base Unit Name' aspect is blank.
- Width attribute with the value aspect of 'Base Unit Value' aspect is '0.9144.' The attribute value is converted from 3 ft to the corresponding number in m, so the foot conversion of .3048 is multiplied by the attribute value of 3.

	A	B	C	D	E	F
1	<ID>	<Name>	Width	Width	Width	Width
2	179915	179915	3 ft	unece.unit.MTR		0.9144000000000001

Closest Attribute Link Meta Data Aspect

BMEcat 1.2 and BMEcat 2005 are the only formats that allow access to Product Attribute Link metadata attribute values, via this transformation aspect. The 'closest' value is defined as the first value encountered, either explicit or inherited, while moving up the hierarchy from the product being exported.

The options available for this aspect include the names of the Product Attribute Link type reference types.

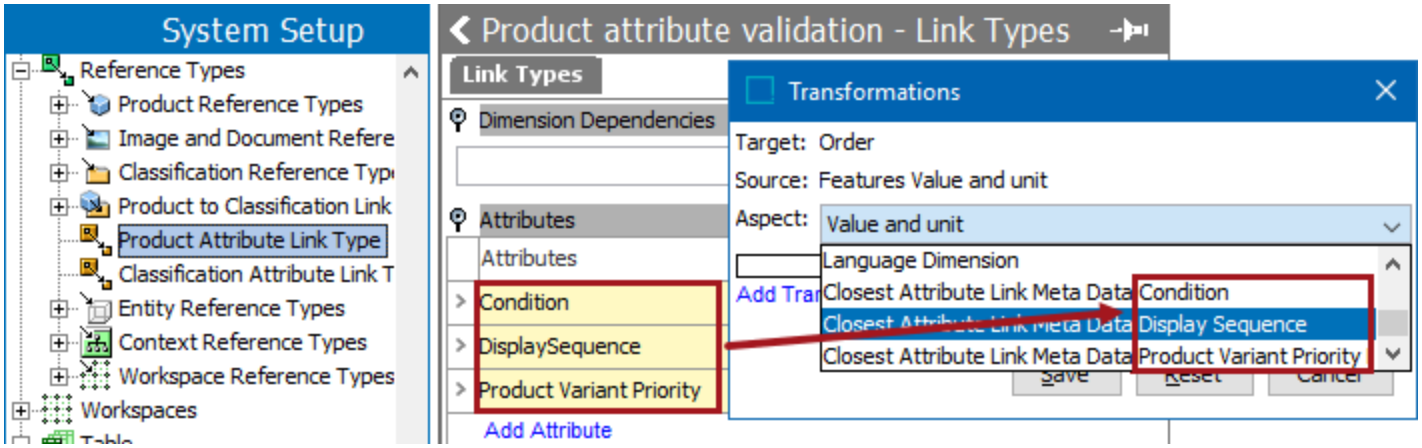
For more information on BMEcat 1.2, refer to the **BMEcat Format** topic. For more information on BMEcat 2005, refer to the **BMEcat 2005 Format** topic.

Limitations

- Product Attribute Link metadata attribute values cannot be imported.
- Classification Attribute Link Type metadata and attributes inherited from classifications are not accessible.
- For a specific configuration, there is a limitation on exporting inherited values for the Display Sequence attribute, as described in the **Closest Attribute Link Meta Data and Display Sequence** topic.

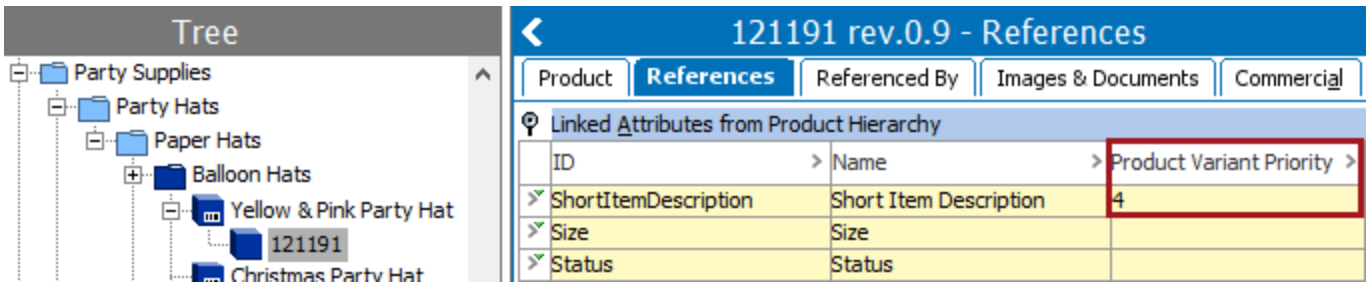
For general information on setting a Transformation aspect, refer to the **Aspect - Transform Outbound** topic.

For example, as shown in System Setup, the following product attribute link type includes the metadata attributes Condition, DisplaySequence, and Product Variant Priority. These same metadata attributes are available via the transformation Aspect parameter.



Note: The name of the selected aspect is displayed after the name of the mapped attribute.

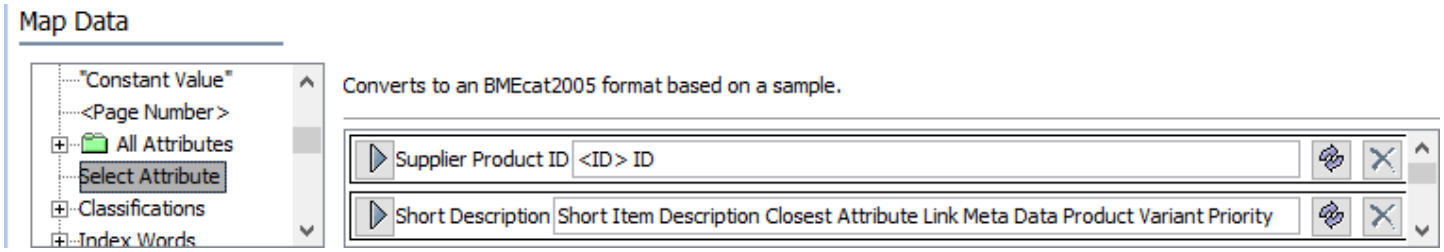
In the tree, product 121191 has an inherited value of '4' for the Product Variant Product metadata attribute displayed on the References tab under the Linked Attributes from Product Hierarchy flipper.



Configuration

On the Select Format step of the outbound data tool, select BMEcat 1.2 or BMEcat 2005.

In addition to the mapping targets required by the BMEcat formats, the Select Attribute data source is used for the Short Description mapping target. The 'Short Item Description' attribute is mapped, and the transformation Aspect is set to Closest Attribute Link Meta Data Product Variant Priority.



Results

Product 121191 is exported with the following output:

- Supplier_PID is 121191 because it was mapped to the product ID.
- The 'Short Item Description' attribute with the value aspect of 'Closest Attribute Link Meta Data Product Variant Priority' aspect is 4, and is displayed in the preformatted DESCRIPTION_SHORT tag.

```
<PRODUCT>
  <SUPPLIER_PID>121191</SUPPLIER_PID>
  <PRODUCT_DETAILS>
    <DESCRIPTION_SHORT>4</DESCRIPTION_SHORT>
```

Closest Attribute Link Meta Data Aspect and Display Sequence

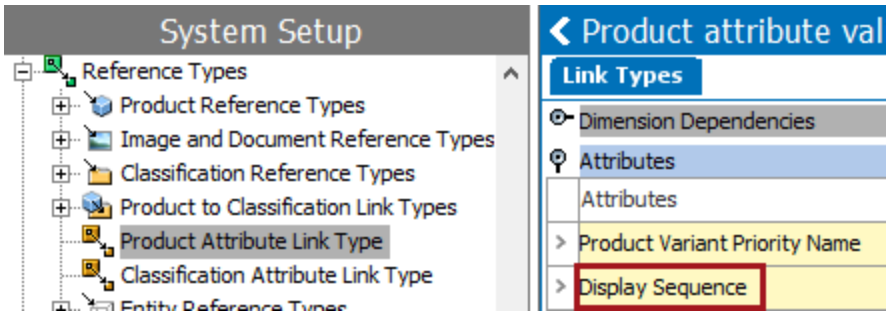
BMEcat 2005 is the only format that allows access to Product Attribute Link metadata attribute values, via the Closest Attribute Link Meta Data transformation aspect.

Inheritance of the Display Sequence attribute value used as metadata varies based on the configuration as follows:

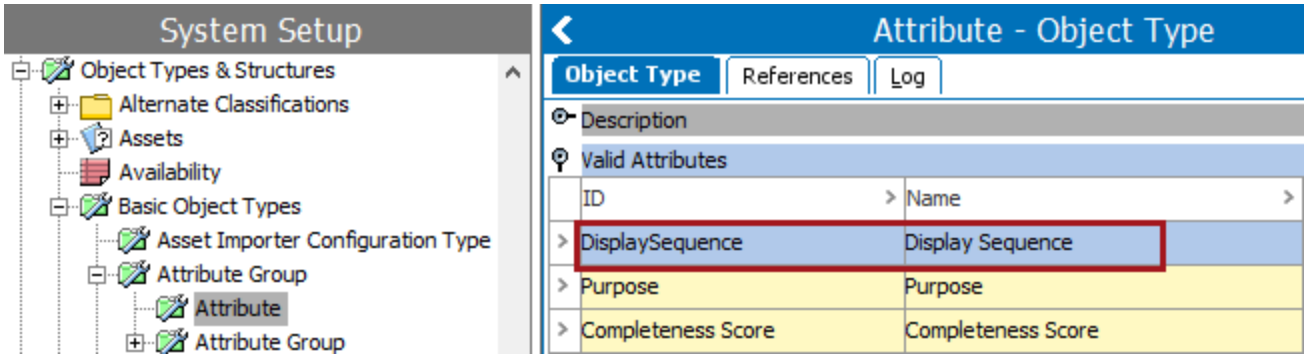
- Using the standard setup of a display sequence attribute, both explicit and inherited values via the Product Attribute Link are exported as expected. Standard setup is defined in the **Attribute Display Sequence** topic of the **Getting Started** documentation.
- Configuring the Display Sequence attribute on an attribute node, and setting the value directly on the attribute in System Setup, results in both explicit and inherited values being displayed via the Product Attribute Link, but inherited values are not exported.

For an example of setting the Display Sequence on the attribute, consider the following:

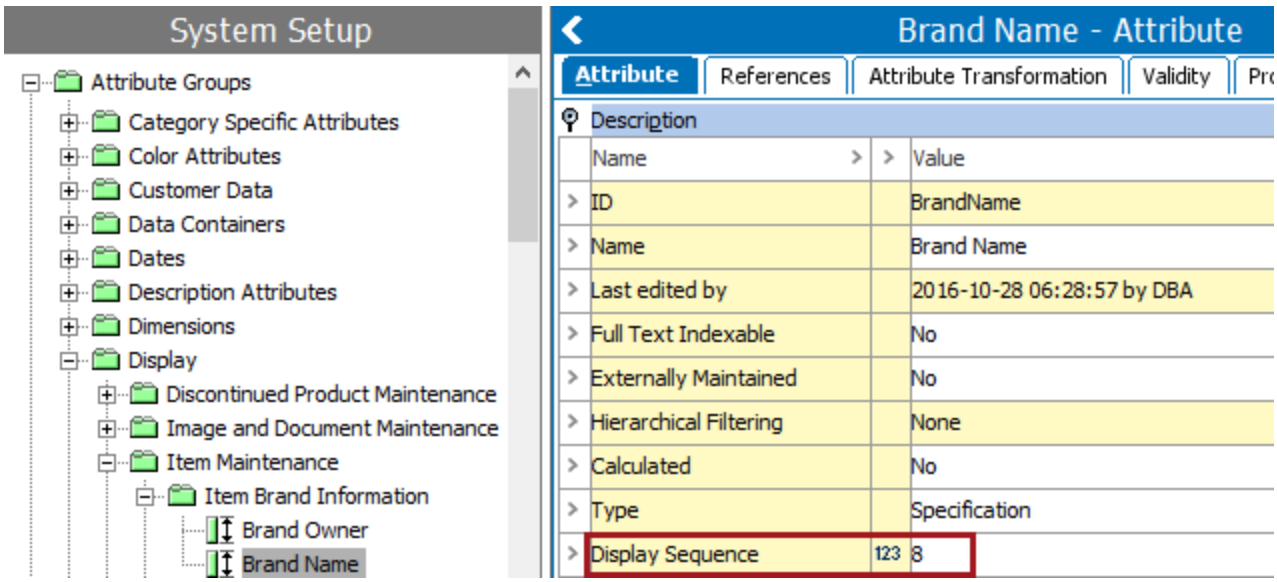
The Product Attribute Link Type includes a metadata attribute with the name Display Sequence. This is the attribute identified in System Setup as **Default Attribute to use as Attribute Display Sequence Attribute**.



The same attribute is valid as metadata on the attribute object type (below the Object Types & Structures node in System Setup).



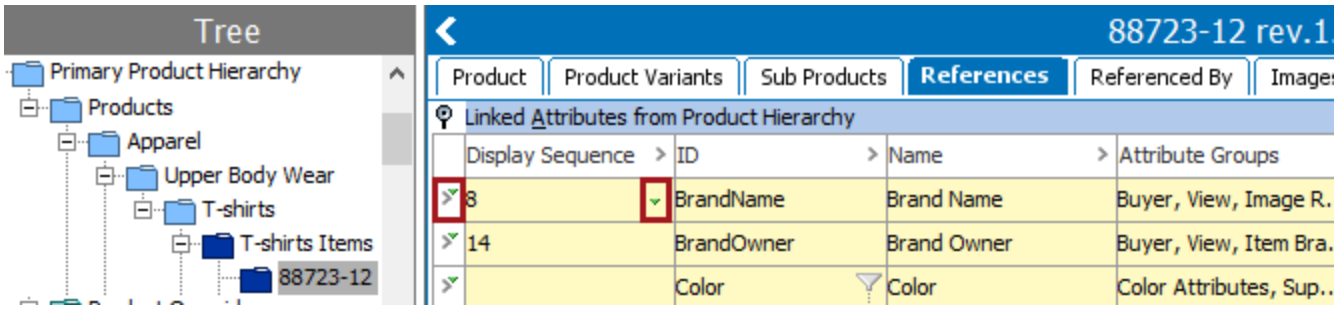
The value is then set on an attribute that is valid for a product.



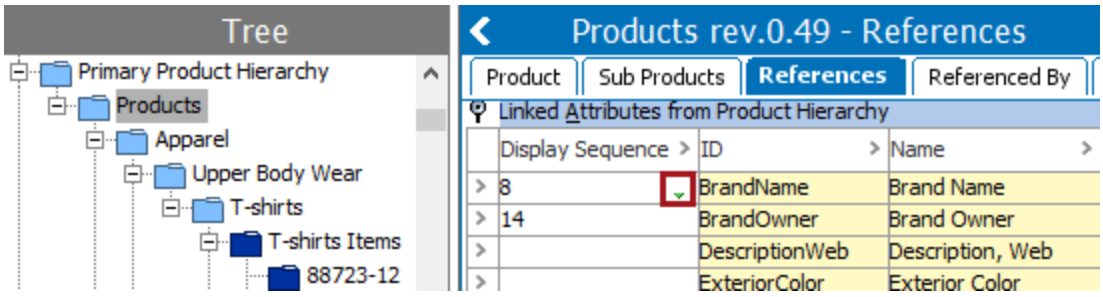
The value from the attribute is displayed on a product on the References tab, under the Linked Attributes from Product Hierarchy flipper.

Notice the two small green down arrows:

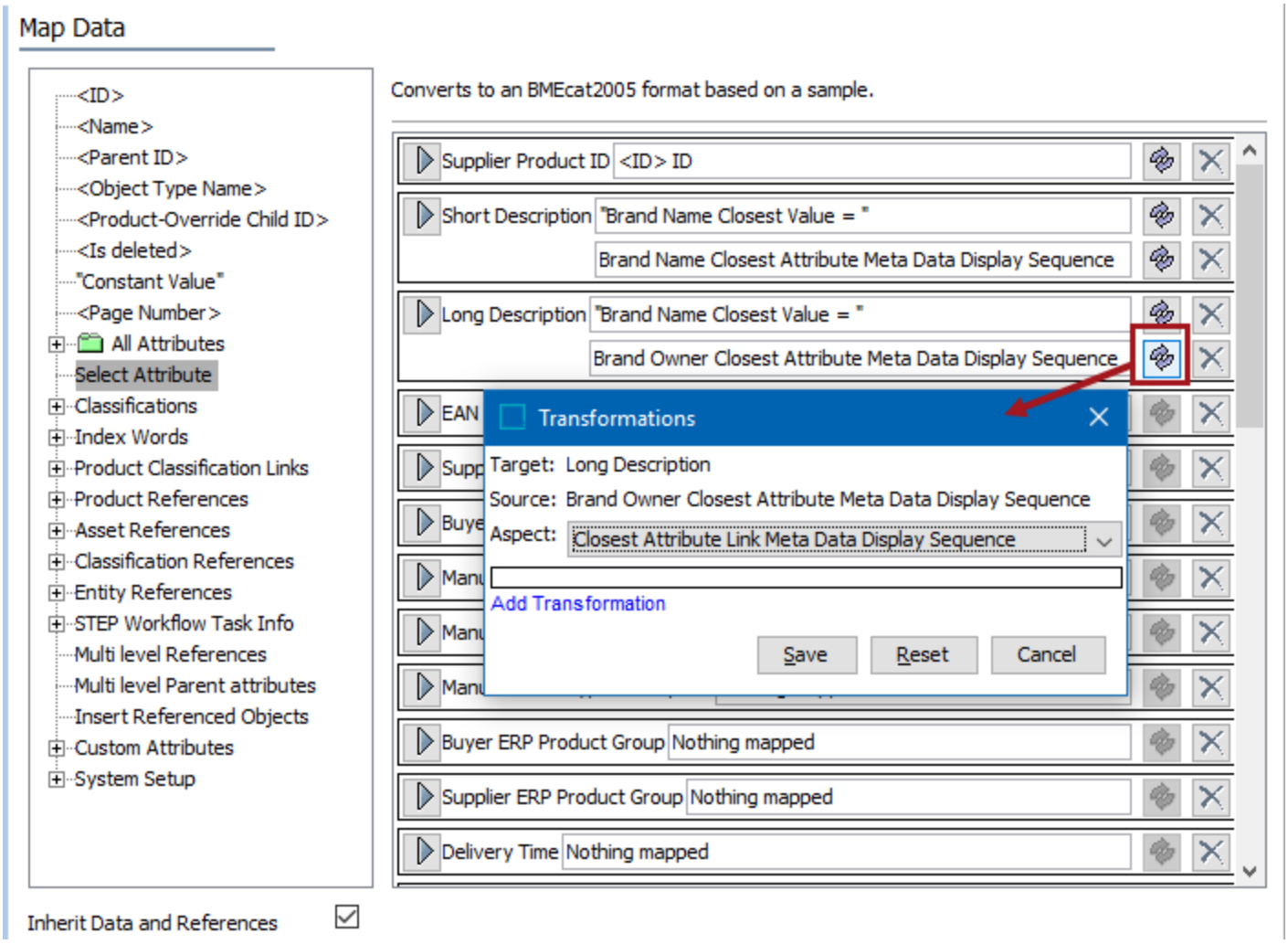
- The arrow in the first column (to the left of the Display Sequence column) indicates that the value is inherited from the location higher up the Tree that is identified by the Inherited From column.
- The arrow within the attribute value field indicates that the value is inherited from the attribute.



In this example, viewing the Products node (indicated as Inherited From) for the Brand Name attribute continues to show the arrow that indicates the value is inherited from the attribute. Notice that the Brand Owner linked attribute is not inheriting from the attribute, indicated by the missing green arrow.



With this configuration, the Brand Name and Brand Owner attributes are mapped. In this mapping, a constant value has been added to both Brand Name and Brand Owner to clearly identify it in the output file. The 'Closest Attribute Link Meta Data Display Sequence' transformation aspect is also selected for each mapped attribute.



The output shows that the value inherited from the attribute (Brand Name) is not exported, while the value inherited from the Tree (Brand Owner) is exported.

The following portion of the Generic XML output file shows the constant value text and the inherited values:

```
<T_NEW_CATALOG>
  <PRODUCT>
    <SUPPLIER_PID>100914</SUPPLIER_PID>
    <PRODUCT_DETAILS>
      <DESCRIPTION_SHORT>Brand Name Closest Value = </DESCRIPTION_SHORT>
      <DESCRIPTION_LONG lang="en">Brand Name Closest Value = 14</DESCRIPTION_LONG>
      <EAN/>
```

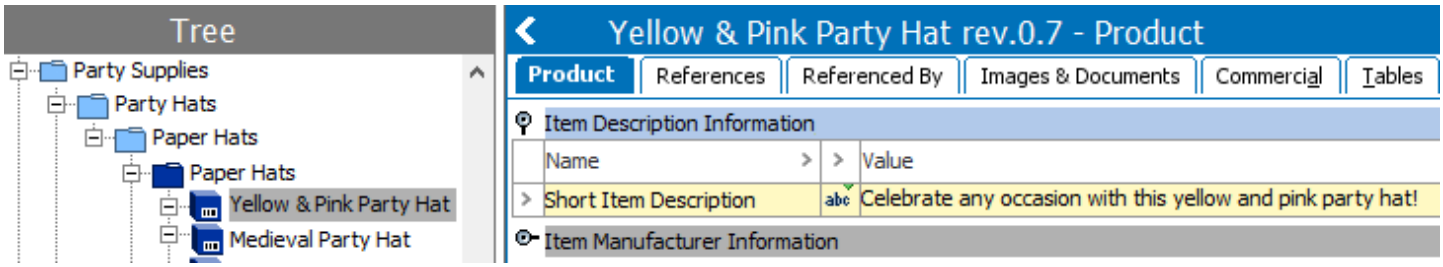
Dimension Dependencies Aspects

Data for the dimension dependencies parameter on attributes can be extracted using the following dimension dependency aspects:

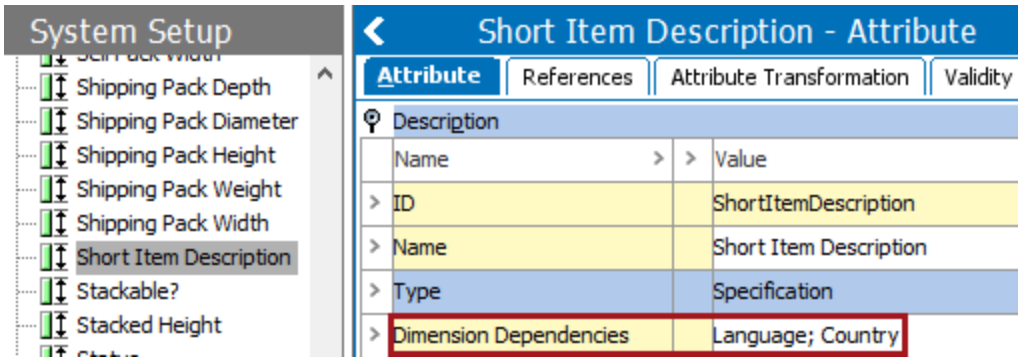
- **Dimension Dependencies** - Extracts the names of the dimension dependencies on the selected attribute.
- **[*name*] Dimension** - Extracts the ID of the dimension point selected by name, for the selected attribute, based on the exported context.

For general information on setting a Transformation aspect, refer to the **Aspect - Transform Outbound** topic.

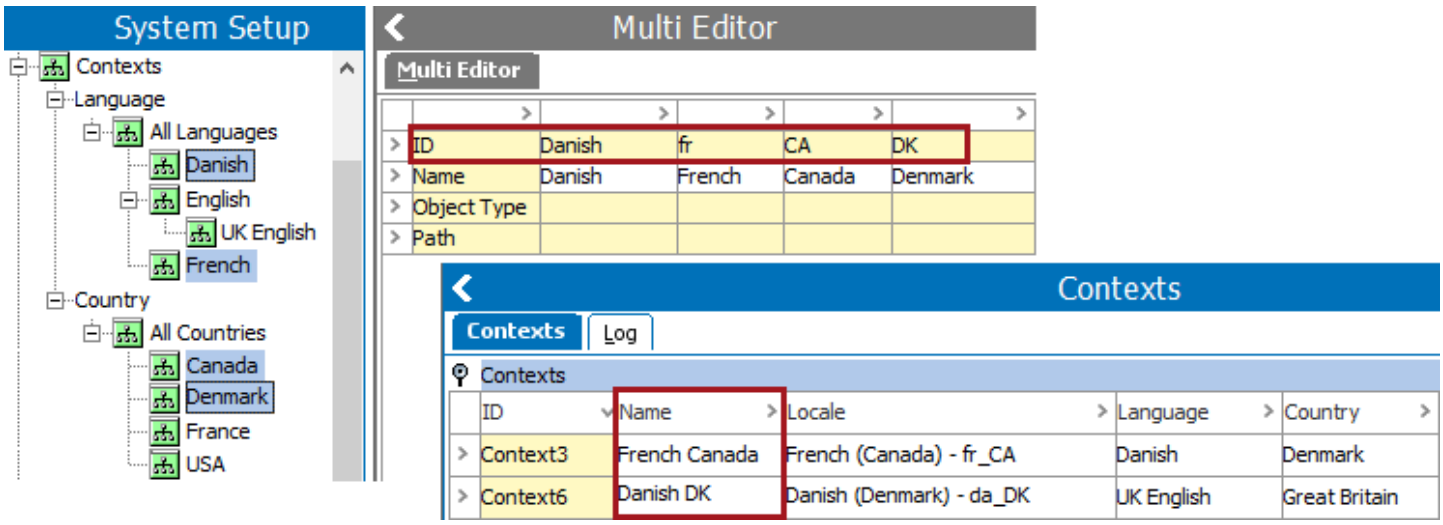
For example, the Tree displays that the Short Item Description attribute is valid on a product with ID 121179 and name 'Yellow & Pink Party Hat' and has a value in the contexts being exported.



In System Setup, the 'Short Item Description' attribute shows that two dimensions have been selected: 'Language' and 'Country name.' A mapping option is displayed for each of the Dimension Dependencies applied to the attribute.



For the dimension points, for the context selected during the export, the ID of the dimension point is used. In this example, one export is from the 'Danish DK' context (with Language Danish and Country Denmark) and the other is from the 'French Canada' context (with Language French and Country Canada). For more information, refer to the **Dimensions, Dimension Points, and Contexts** topic in the **System Setup**.

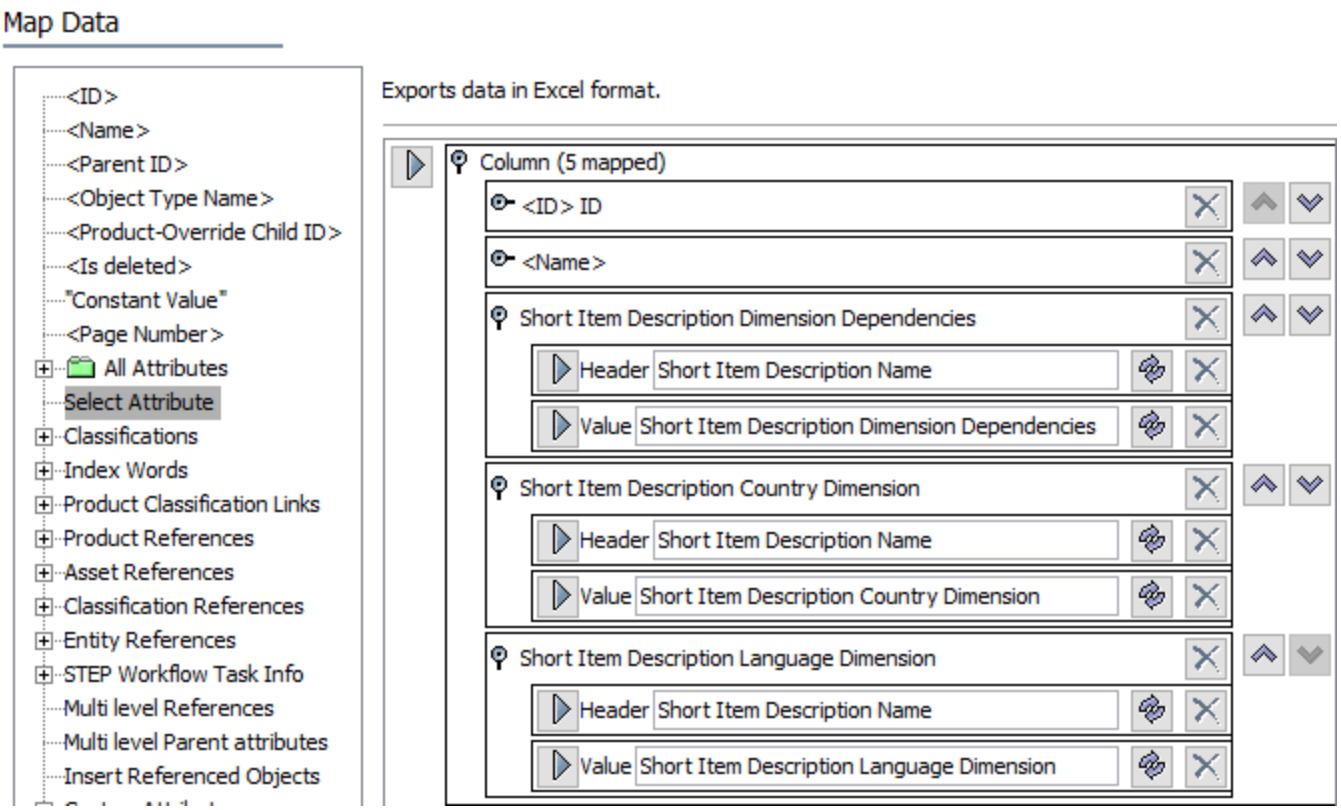


Configuration

In addition to the ID and name of the product, the values identified above are mapped using the Select Attribute data source and the specified aspect.

Note: The name of the selected aspect is displayed after the name of the mapped attribute.

- Short Item Description attribute is mapped with the aspect of 'Dimension Dependencies.'
- Short Item Description attribute is mapped with the aspect of 'Country Dimension.'
- Short Item Description attribute is mapped with the aspect of 'Language Dimension.'



Results

The product is exported two times, once for each context, with the following output:

- For both contexts: ID is 121179.
- For both contexts: Name is Yellow & Pink Party Hat.
- For both contexts: Short Item Description attribute with the value aspect of 'Dimension Dependencies' is Language and Country, separated by a semicolon.
- Short Item Description attribute with the value aspect of 'Language Dimension' is the ID of the Language object, namely 'Danish' for the Danish DK context and 'fr' for the French Canada context.
- Short Item Description attribute with the value aspect of 'Country Dimension' is the ID of the Country object, namely 'DK' for the Danish DK context and 'CA' for the French Canada context.

	A	B	C	D	E
1	<ID>	<Name>	(ShortItemDescription)	(ShortItemDescription)	(ShortItemDescription)
2	121179	Yellow & Pink Party Hat	Language;Country	Danish	DK

	A	B	C	D	E
1	<ID>	<Name>	Courte Description de l'Article	Courte Description de l'Article	Courte Description de l'Article
2	121179	Yellow & Pink Party Hat	Language;Country	fr	CA

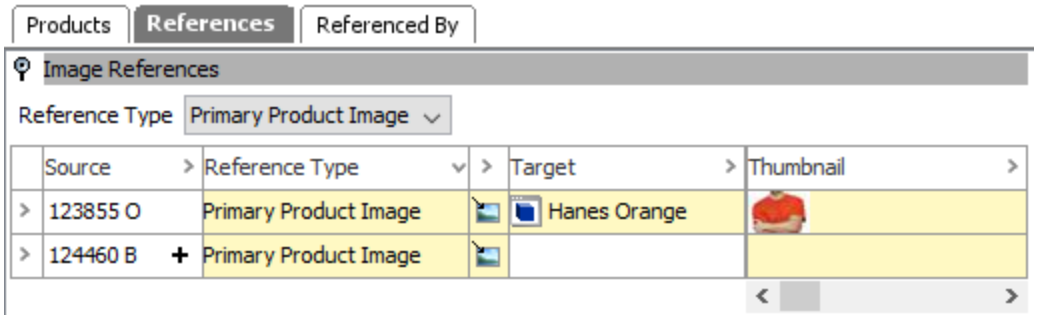
Is Deleted Aspect

This aspect is valid only for an event-based OIEP when using formats that require mapping (i.e., Generic XML, Excel, and CSV), as those are the formats that allow transformations. The OIEP must be configured to listen for a change on the specified object type and reference type, and then sends the 'IsDeleted' value for the deleted target ID to downstream systems.

Note: Although mapping is allowed in Export Manager, no value is output for the 'Is Deleted' aspect.

For general information on setting a Transformation aspect, refer to the **Aspect - Transform Outbound** topic.

In the following example, when the 'Primary Product Image' reference type target is deleted from a product (image ID '20584' was deleted from product '124460 B'), approving the product triggers an event for the OIEP. For comparison, image ID '20586' was added as the primary product image to product '123855 O'.

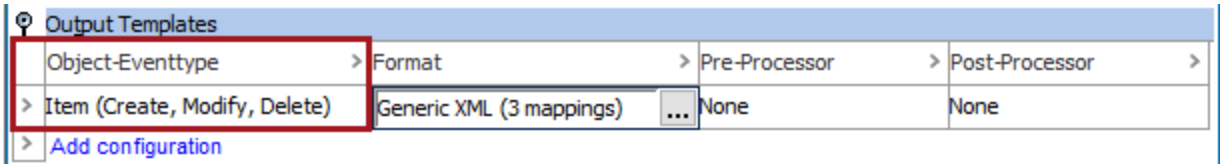


The Generic XML format output includes the 'IsDeleted' value only when the ID of the 'Primary Product Image' was deleted from the product.

Configuration

Create an event-based OIEP as defined in the **Creating an Event-Based Outbound Integration Endpoint** topic, including the following specific settings.

1. On the Configuration tab, open the Output Templates flipper, click into the **Object-Eventtype** parameter to display and click the ellipsis button (...). Set the Object type(s) and the Event Type(s).

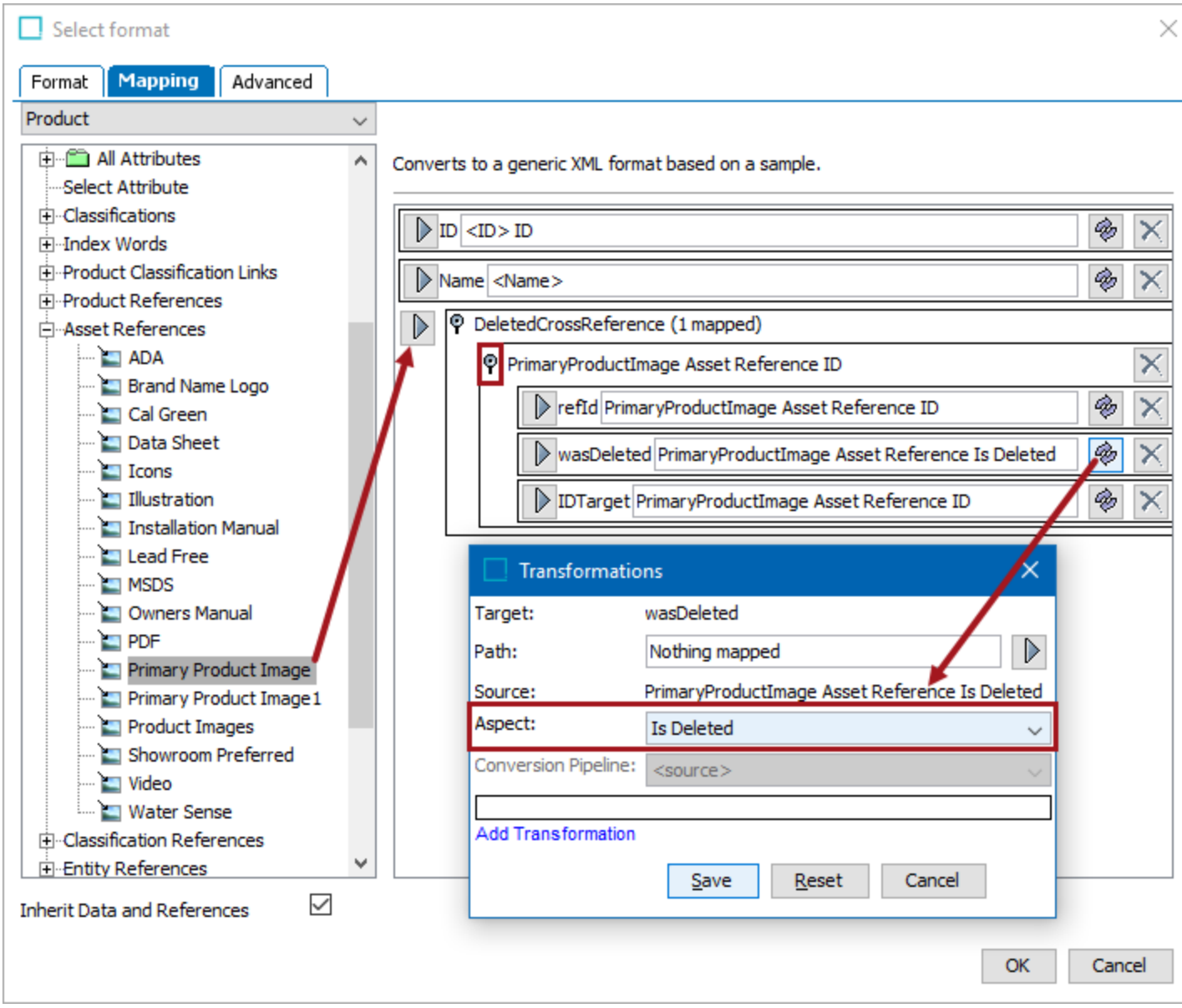


2. On the Configuration tab, on the Output Templates flipper, click into the **Format** parameter to display and click the ellipsis button (...).
   - On the Format tab, set the desired format and include a sample if required.

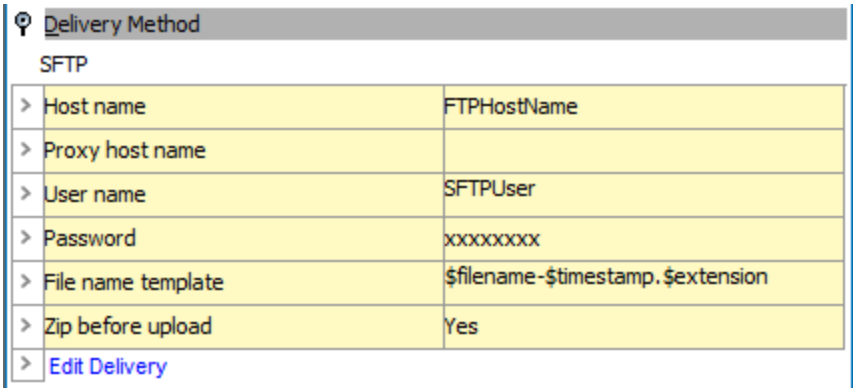
     For this example, the following Generic XML sample document is used:

```
<ExportProducts>
   <Products>
       <Product ID="[?Target ID?]">
          <?Record?>
          <Name>
               <?Target?>
          </Name>
          <DeletedCrossReferences>
             <DeletedCrossReference refTypeID="[?Target refId?]" wasDeleted="[?Target
wasDeleted?]">
             <?MultiTarget?>
                  <IDTarget>
                       <?Target?>
                  </IDTarget>
             </DeletedCrossReference>
          </DeletedCrossReferences>
       </Product>
   </Products>
</ExportProducts>
```

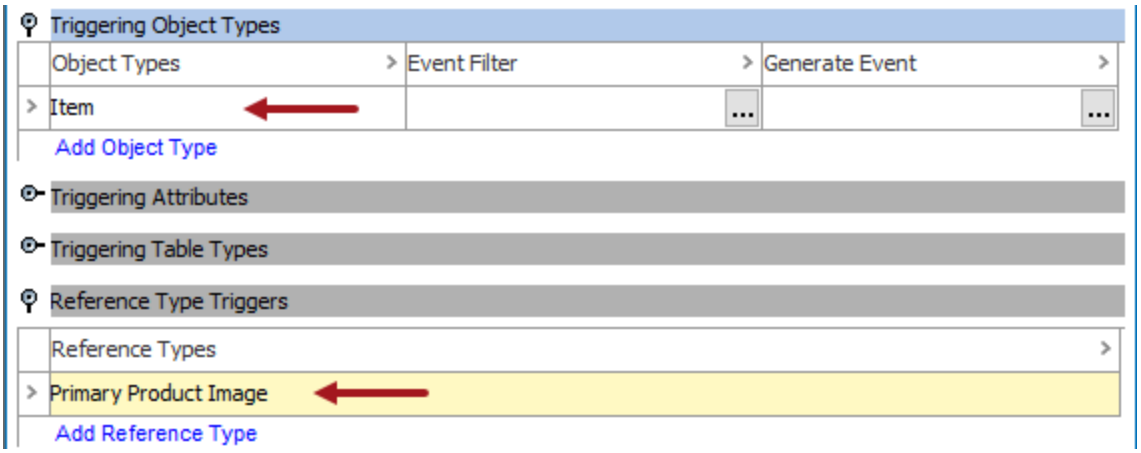
- On the Mapping tab, map the attributes, the reference type, and the transformation.



- On the Configuration tab, on the **Delivery Method** flipper, set the delivery method parameters.



- On the **Event Triggering Definitions** tab add the required object type(s) and reference type(s).



- Complete the final OIEP setup as defined in the **Running an Outbound Integration Endpoint** topic and invoke the endpoint.

Results

For the sample data shown above, the image target '20584' was removed for product '124460 B'. The output Generic XML includes the 'IsDeleted' value.

For product '123855 O', no value is included on the hard-coded 'wasDeleted=' element supplied by the Generic XML sample.

```
<?xml version="1.0" encoding="utf-8"?>
<ExportProducts>
  <Products>
    <Product ID="123855">
      <Name>123855 O</Name>
      <DeletedCrossReferences>
        <DeletedCrossReference refTypeID="20586" wasDeleted="">
          <IDTarget>20586</IDTarget>
        </DeletedCrossReference>
      </DeletedCrossReferences>
    </Product>
    <Product ID="124460">
      <Name>124460 B</Name>
      <DeletedCrossReferences>
        <DeletedCrossReference refTypeID="20584" wasDeleted="IsDeleted">
          <IDTarget>20584</IDTarget>
        </DeletedCrossReference>
      </DeletedCrossReferences>
    </Product>
  </Products>
</ExportProducts>
```

Key Aspect

When the STEP ID is used to identify an object, the ID data source is used to identify the mapping target intended to hold the ID. All objects in STEP can be exported using the ID data source.

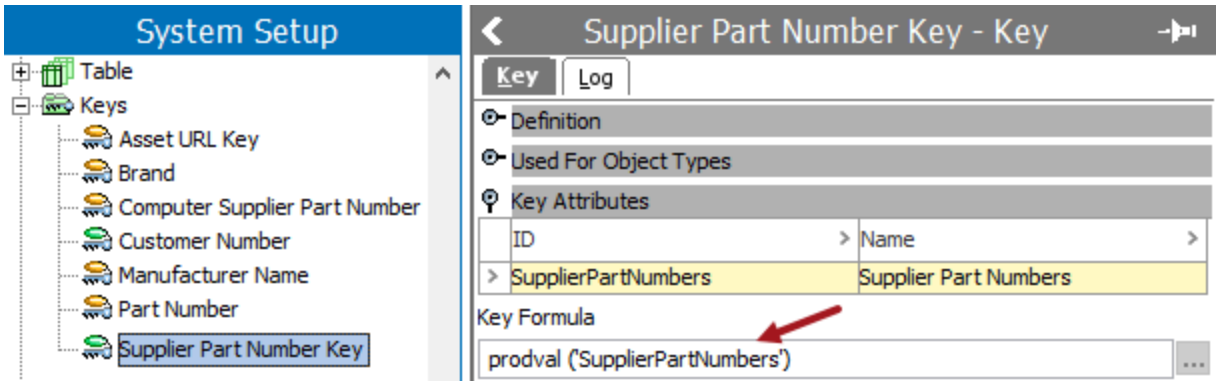
The ID data source is also used to define a unique key (in place of, or in addition to, the STEP ID) as required by external systems.

For general information on setting a Transformation aspect, refer to the **Aspect - Transform Outbound** topic.

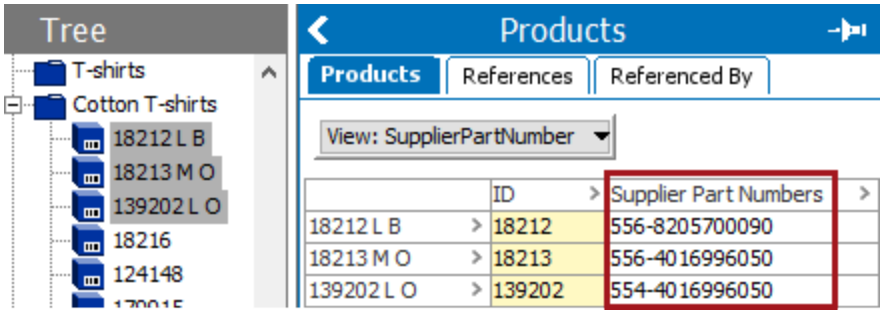
Prerequisite

Keys must be activated before they are displayed in the Map Data step of an export tool. For more information, refer to the **Unique Keys** topic of the **System Setup** documentation.

For example, on System Setup, the 'Supplier Part Number Key' key is active and is displayed in the 'Supplier Part Numbers (SupplierPartNumbers)' attribute that is valid on products.



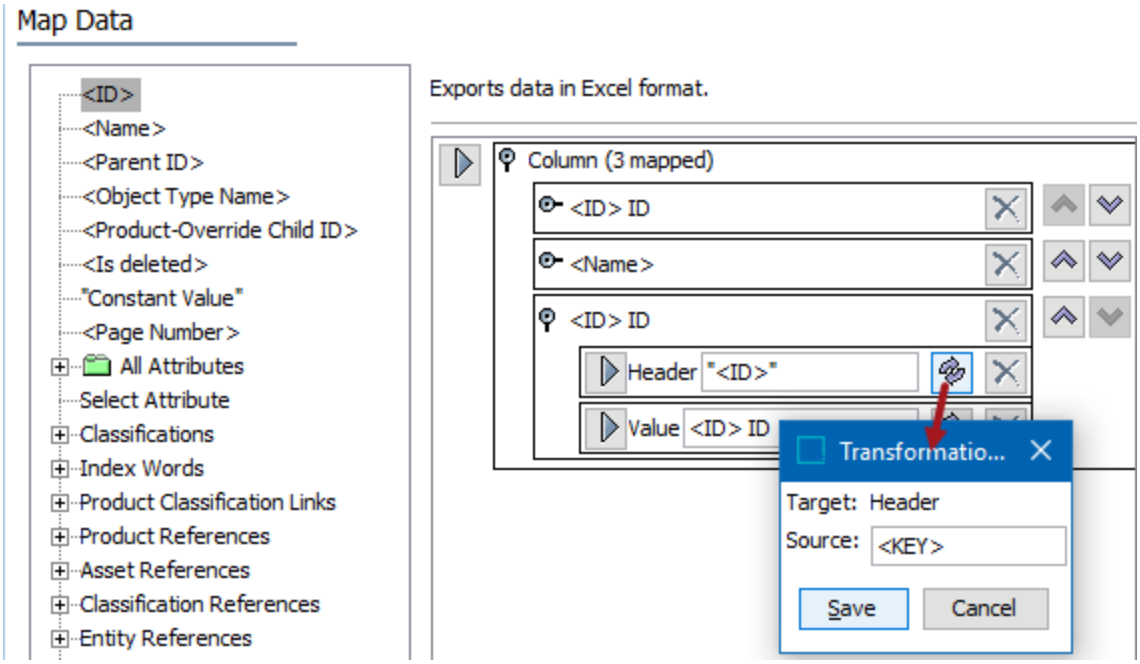
In the Tree, the products 18212, 18213, and 139202 have values for the attribute 'Supplier Part Numbers (SupplierPartNumbers)' which is used in the active unique key.



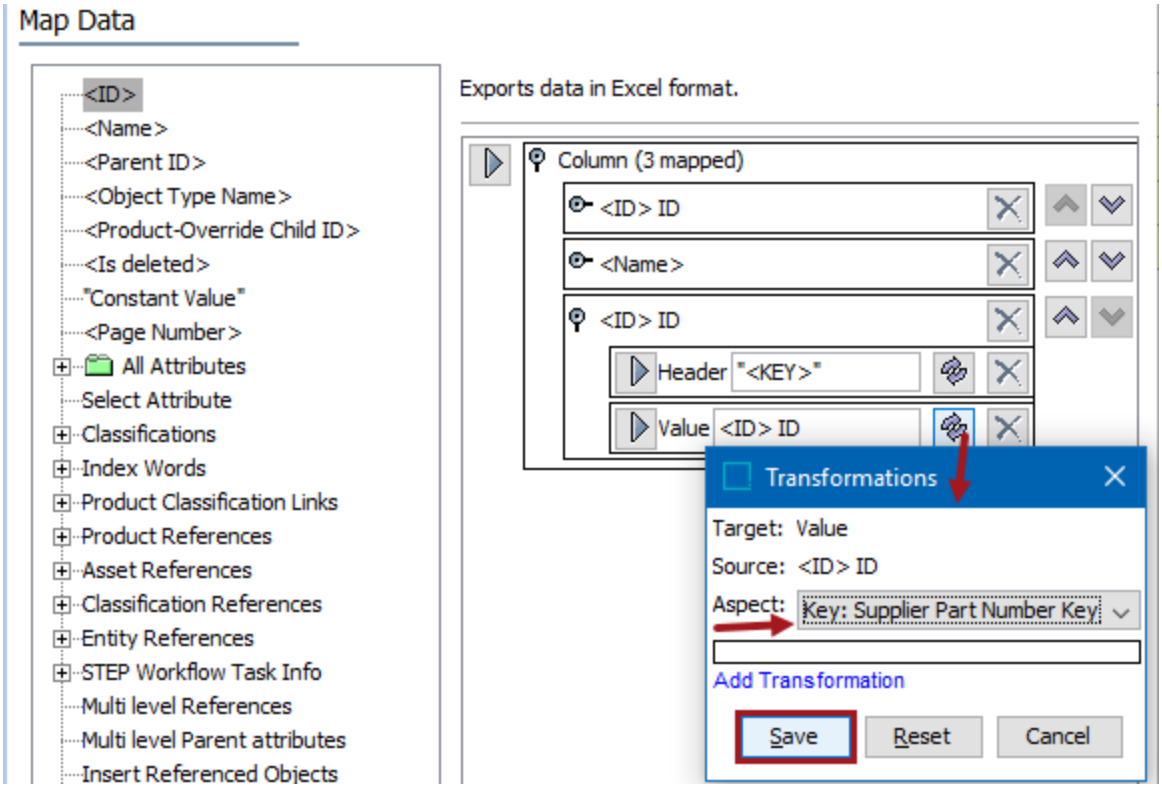
Configuration

In addition to the ID and name of the product, the specified key is mapped using the value aspect of the ID data source as defined below.

For the Key, an optional first transformation is used to modify the Header from '<ID>' to '<KEY>.' Any text can be entered for the header, for example, the name of the key: <Key: Supplier Part Numbers>.



For the Key, a second transformation is used to modify the Value from '<ID> ID' to '<ID> Key: Supplier Part Number Key.'



Results

Products 18212, 18213, and 139202 are exported with STEP ID, Name, and Key:

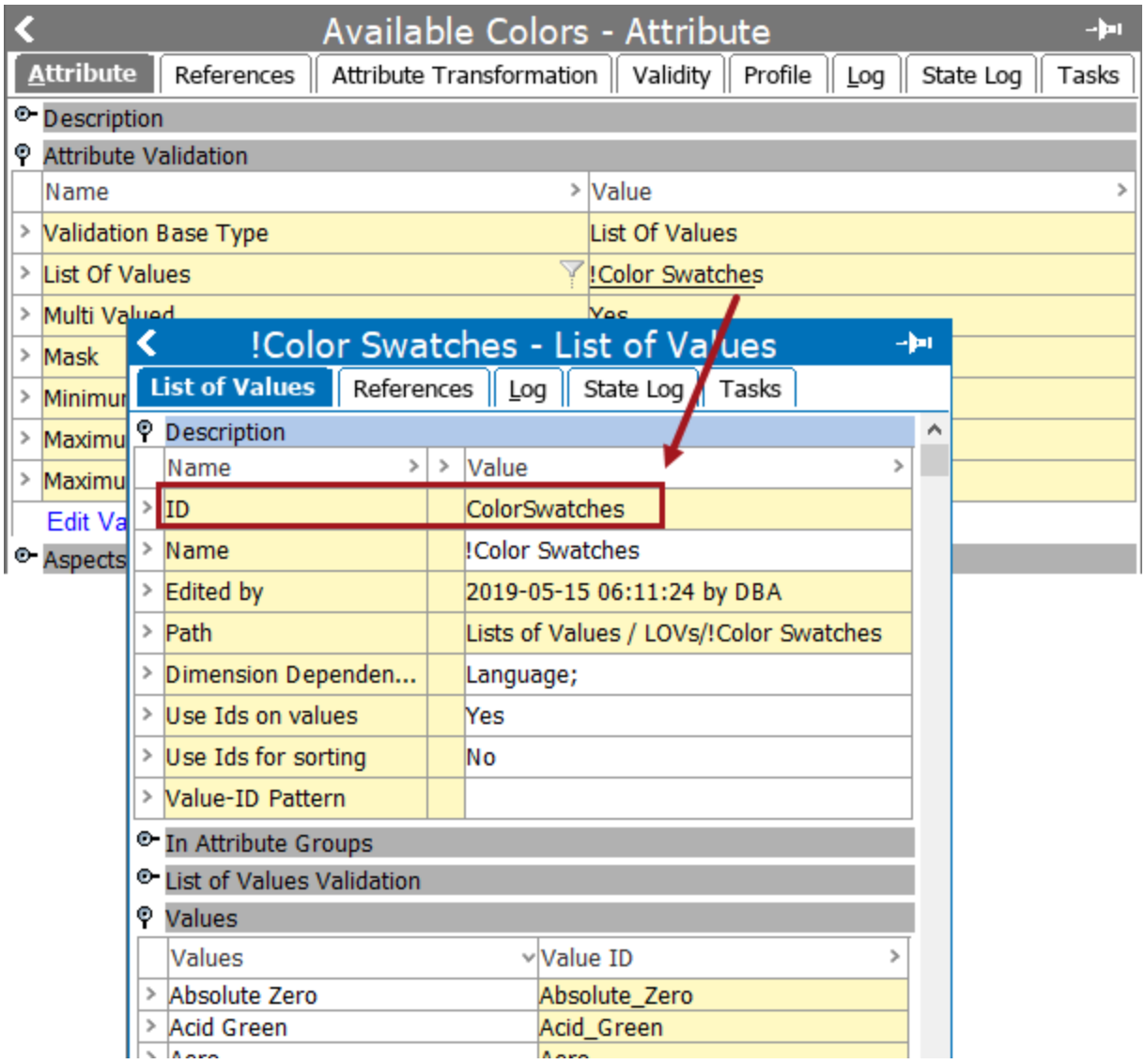
	A	B	C
1	<ID>	<Name>	<KEY>
2	18212	18212 L B	556-8205700090
3	18213	18213 M O	556-4016996050
4	139202	139202 L O	554-4016996050

LOV ID Aspect

The 'LOV ID' aspect allows for LOV IDs to be exported for the mapped LOV attributes. For details on LOVs, refer to the **List of Values (LOVs)** topic in the **System Setup** documentation.

For general information on setting a Transformation aspect, refer to the **Aspect - Transform Outbound** topic.

Consider the 'Available Colors' attribute shown below. It uses the '!Color Swatches' LOV. The 'LOV ID' aspect exports the ID of the LOV itself.

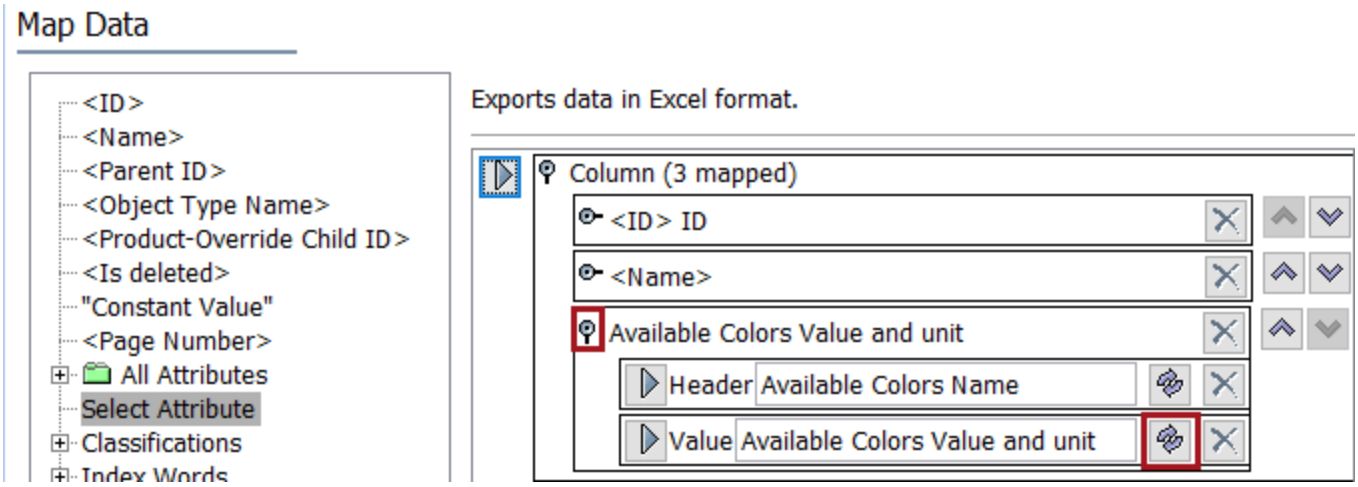


Additional options for exporting LOV aspect information are included in the following topics:

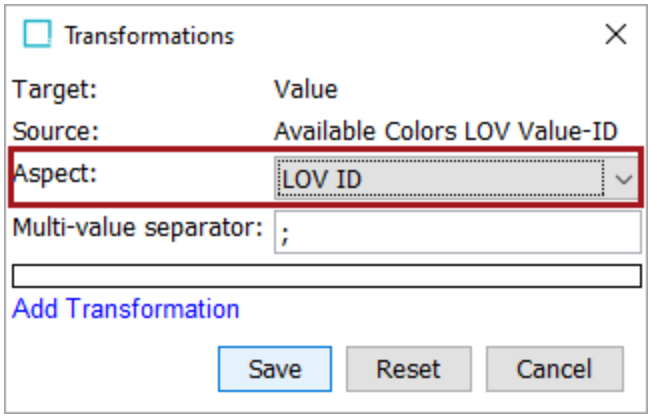
- LOV Value-ID Aspect
- LOV Value-ID or Value Aspect

Configuration

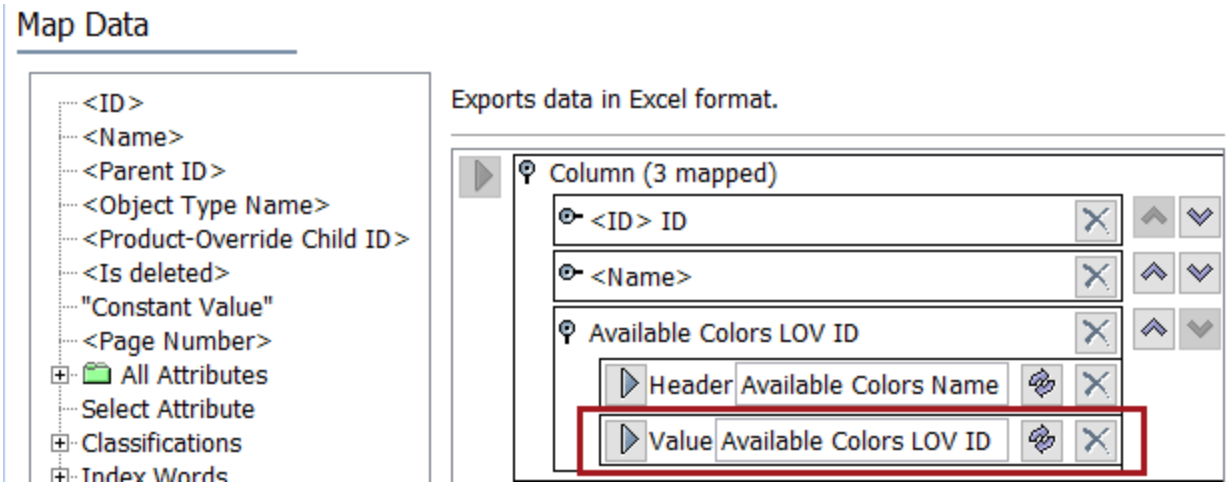
1. In the output tool, select the data to be exported. For more information, refer to the **Data Exchange** topic.
2. On the Map Data step, map the required attribute or attribute group using the **Select Attribute** data source. For more information, refer to **Mapping Attributes via Select Attribute** section of the **Attributes (and Data Containers) - Data Source Outbound** topic.
3. In the right panel, click the flipper for the attribute that should be extracted using LOV ID. In the example below, the 'AvailableColors' attribute has been mapped.
4. On the Value row, click the **Transformation** button to display the Transformations dialog.



5. In the **Transformations** dialog, from the **Aspect** list, select the **LOV ID** and click the **Save** button.



The Value element is displayed with the Available Colors attribute using the 'LOV ID' aspect.



6. Complete any additional mappings and initiate the export.

Results

The product IDs and names are exported, along with the ID for the 'Available Colors' LOV attribute.

	A	B	C
1	<ID>	<Name>	Available Colors
2	18212	18212 L B	ColorSwatches
3	18213	18213 M O	ColorSwatches
4	18216	18216 L O	ColorSwatches

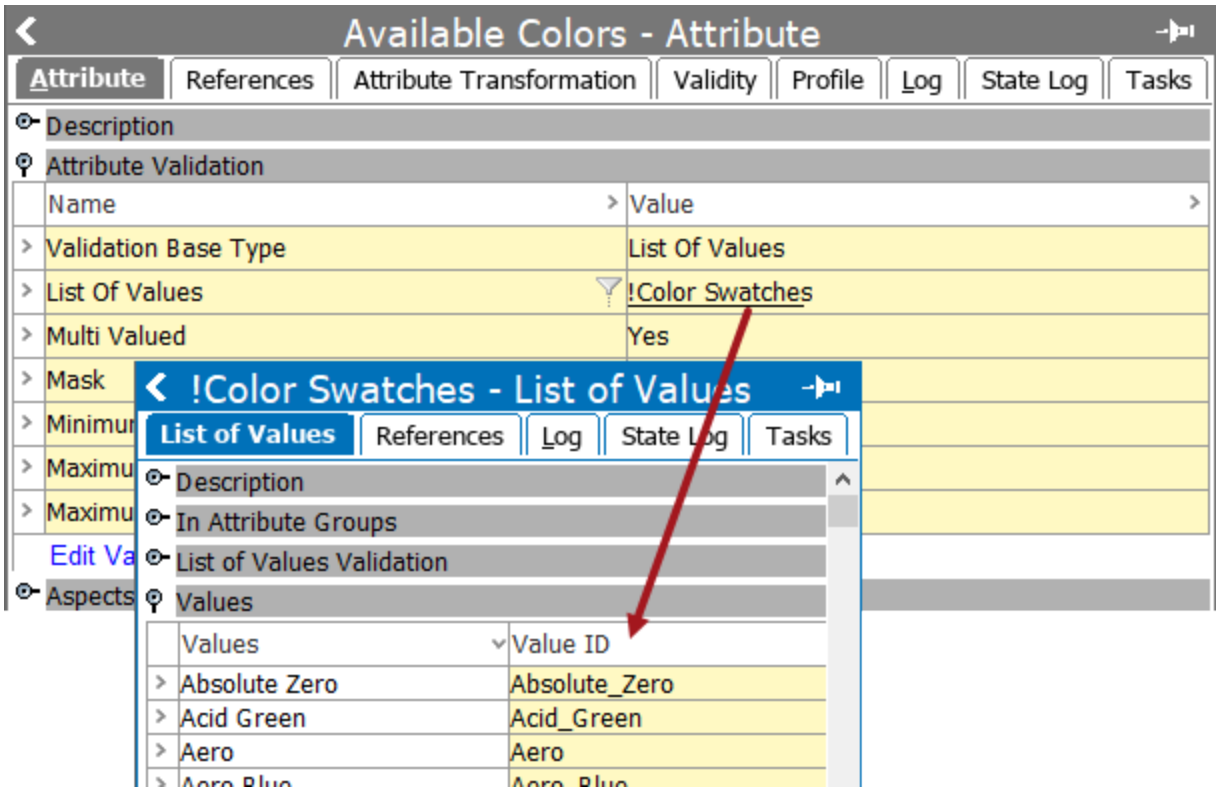
Note: If the LOV attribute allows multiple values, the LOV Value IDs are exported with the multi-value separator defined in the Transformations dialog. For more information, refer to the **Multi-Value Separator - Transform Outbound** topic.

LOV Value-ID Aspect

The 'LOV Value-ID' aspect allows for LOV Value IDs to be exported for the mapped LOV attributes. For details on LOV Value IDs, refer to the **Use IDs on values** section of the **Creating an LOV** topic in the **System Setup** documentation.

For general information on setting a Transformation aspect, refer to the **Aspect - Transform Outbound** topic.

Consider the 'Available Colors' attribute shown below. It uses the '!Color Swatches' LOV, which uses Value IDs. In this example, for identification of the Value ID compared to the Value, notice that the Value IDs include an underscore instead of a space between words. The 'LOV Value-ID' aspect exports the 'Value ID' instead of the Values.



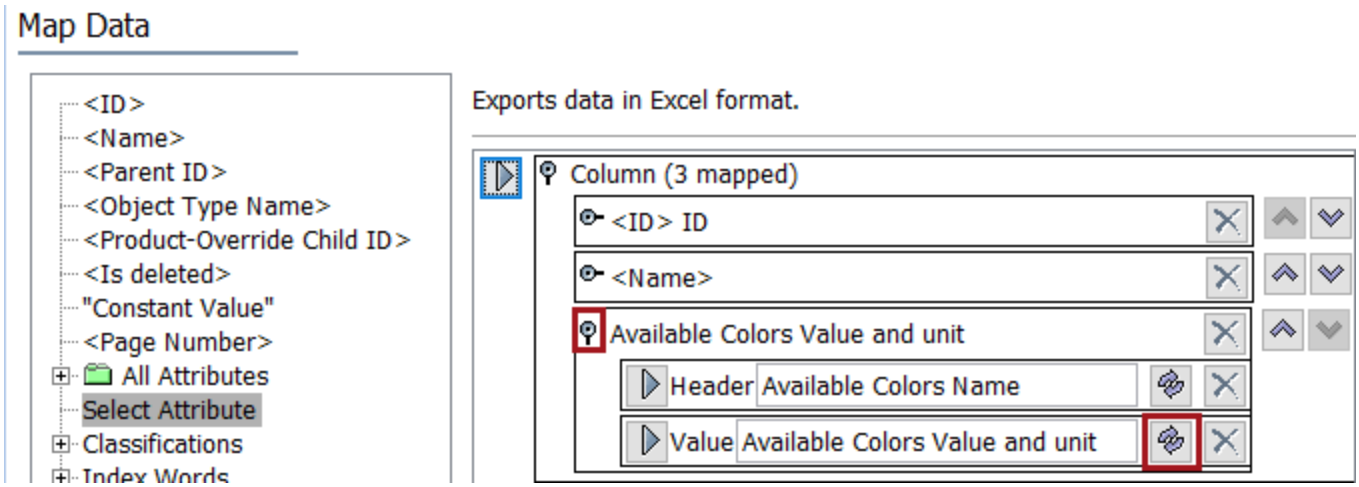
Additional options for exporting LOV aspect information are included in the following topics:

- LOV ID Aspect
- LOV Value-ID or Value Aspect

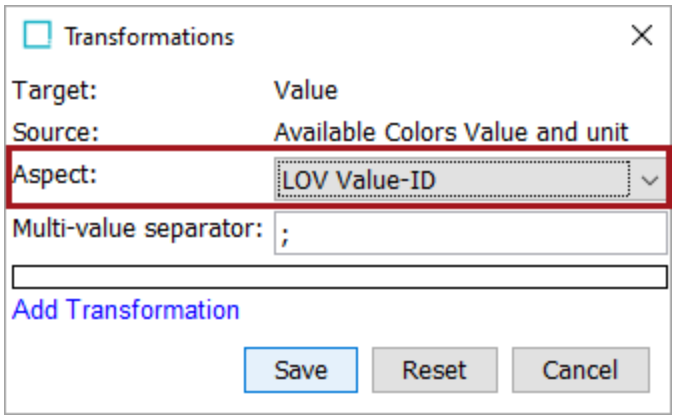
Configuration

1. In the output tool select the data to be exported. For more information, refer to the **Data Exchange** topic.
2. On the Map Data step, map the required attribute or attribute group using the **Select Attribute** data source. For more information, refer to **Mapping Attributes via Select Attribute** section of the **Attributes (and Data Containers) - Data Source Outbound** topic.

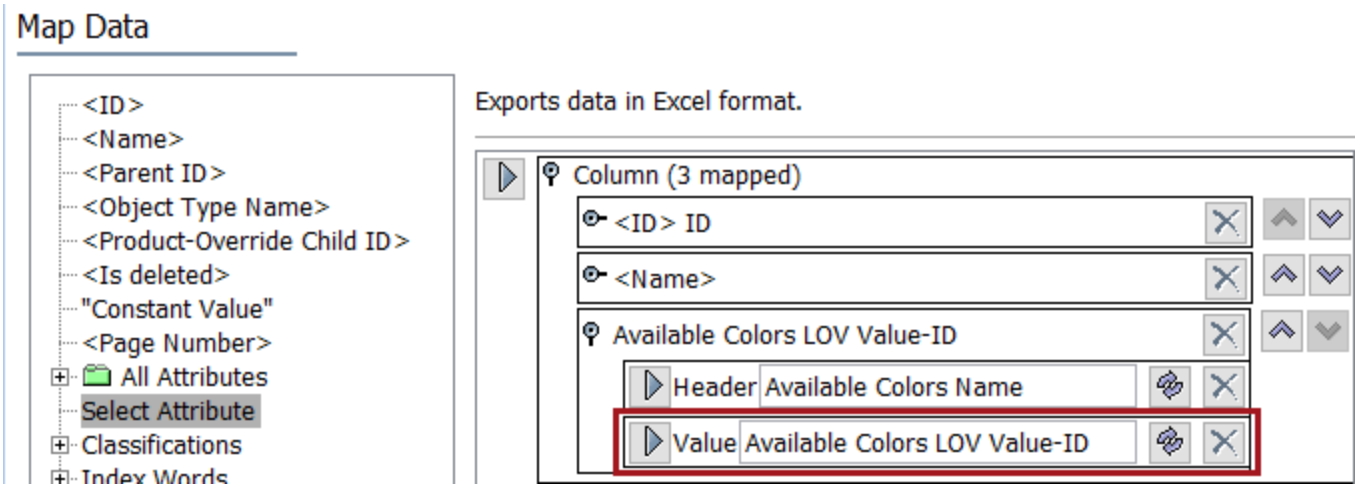
3. In the right panel, click the flipper for the attribute or group that should be extracted using LOV Value-ID. In the example below, the 'AvailableColors' attribute has been mapped.
4. On the Value row, click the **Transformation** button to display the Transformations dialog.



5. In the **Transformations** dialog, from the **Aspect** list, select the **LOV Value-ID** and click the **Save** button.



The Value element is displayed with the Available Colors attribute using the 'LOV Value-ID' aspect.



6. Complete any additional mappings and initiate the export.

Results

The product ID and name are exported, along with the Value IDs for the 'Available Colors' LOV attribute.

	A	B	C
1	<ID>	<Name>	Available Colors
2	18212	18212 L B	Acid_Green
3	18213	18213 M O	Aero
4	18216	18216 L O	Absolute_Zero

Note: If the LOV attribute allows multiple values, the LOV Value IDs are exported with the multi-value separator defined in the Transformations dialog. For more information, refer to the **Multi-Value Separator - Transform Outbound** topic.

LOV Value-ID or Value Aspect

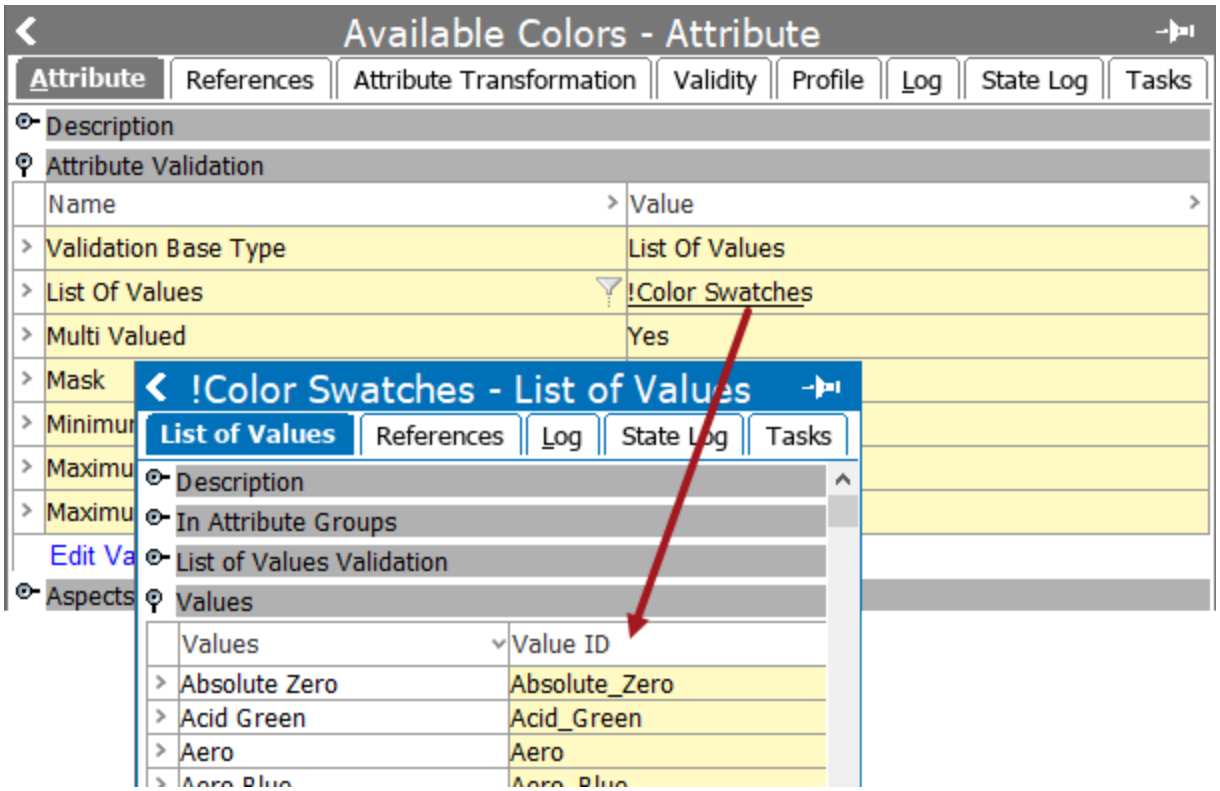
When mapping attributes or attribute groups, this aspect extracts the LOV Value-ID when present, otherwise it extracts the values in the attributes. For details on LOV Value IDs, refer to the **Use IDs on values** section of the **Creating an LOV** topic in the **System Setup** documentation.

The 'LOV Value-ID or Value' aspect allows for values to be exported for attributes and allows for Value IDs to be exported if it is an LOV attribute with a Value ID.

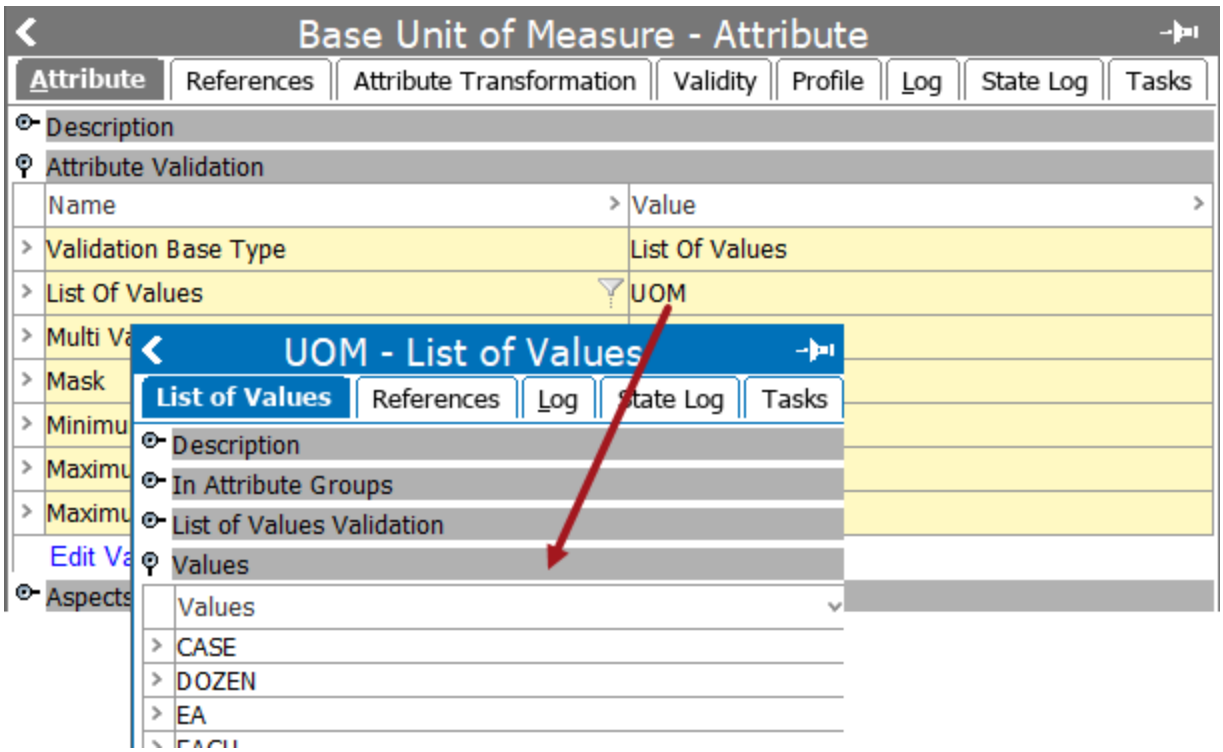
A common use for this aspect is when exporting ETIM via BMEcat2005 format, map an attribute group to the feature field and export the LOV Value IDs. Refer to the **BMEcat 2005 Format Example** section below.

For general information on setting a Transformation aspect, refer to the **Aspect - Transform Outbound** topic.

Consider the 'DataVisualization' attribute group that will be mapped during the Configuration steps below. This group includes two attributes with an LOV Validation Base Type. The 'Available Colors' attribute uses the '!Color Swatches' LOV which uses Value IDs. For this example, for identification of the Value ID compared to the Value, notice that the Value IDs include an underscore instead of a space between words.



The 'Base Unit of Measure' attribute uses the 'UOM' LOV which does not use Value IDs:



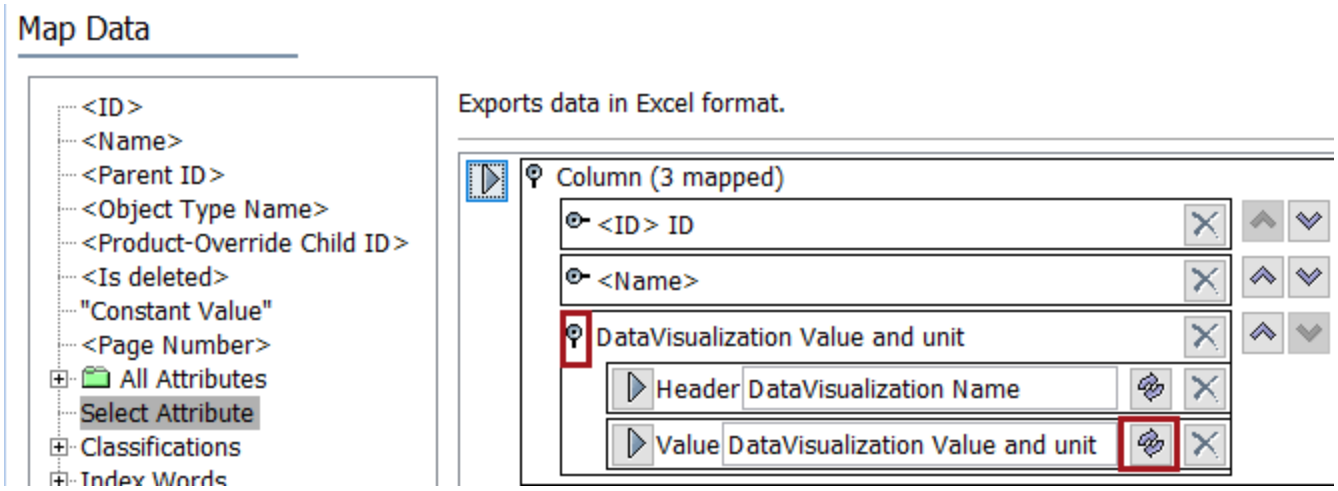
Using the single 'LOV Value-ID or Value' aspect for the attribute group results in exporting the Value IDs for the 'Available Colors' attribute and the Values for the 'Base Unit of Measure' attribute.

Additional options for exporting LOV aspect information are included in the following topics:

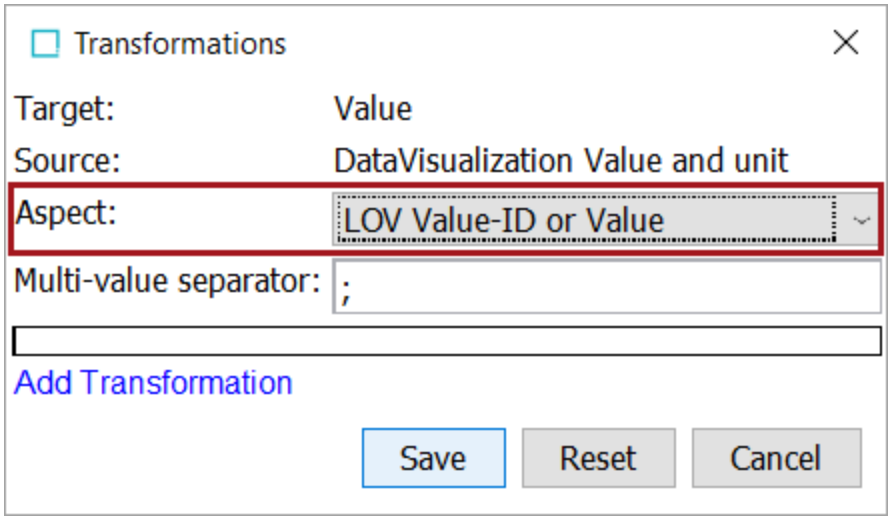
- LOV ID Aspect
- LOV Value-ID Aspect

Configuration

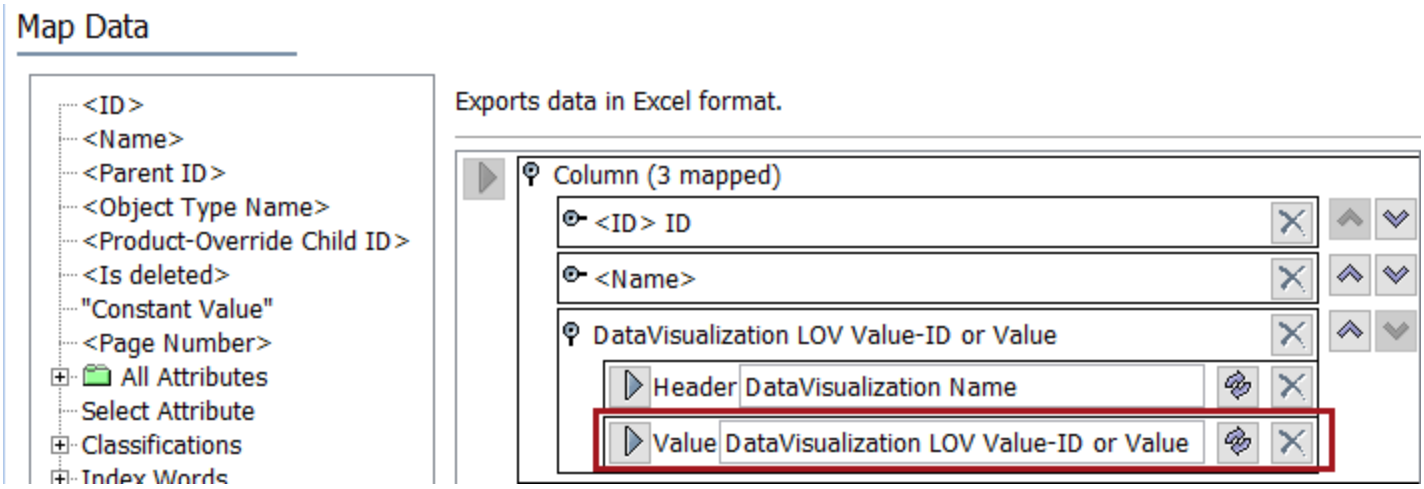
1. In the output tool select the data to be exported. For more information, refer to the **Data Exchange** topic.
2. On the Map Data step, map the required attribute or attribute group using the **Select Attribute** data source. For more information, refer to **Mapping Attributes via Select Attribute** section of the **Attributes (and Data Containers) - Data Source Outbound** topic.
3. In the right panel, click the flipper for the attribute or group that should be extracted using LOV Value ID or Value. In the example below, the 'DataVisualization' group has been mapped.
4. On the Value row, click the **Transformation** button to display the Transformations dialog.



5. In the **Transformations** dialog, from the **Aspect** list, select the **LOV Value-ID or Value** and click the **Save** button.



The Value element is displayed with the DataVisualization attribute using the 'LOV Value-ID or Value' aspect.



6. Complete any additional mappings and initiate the export.

Results

The product ID and name are exported, along with the Value IDs for the LOV attribute that uses them, and Values for the LOV attribute that does not use them.

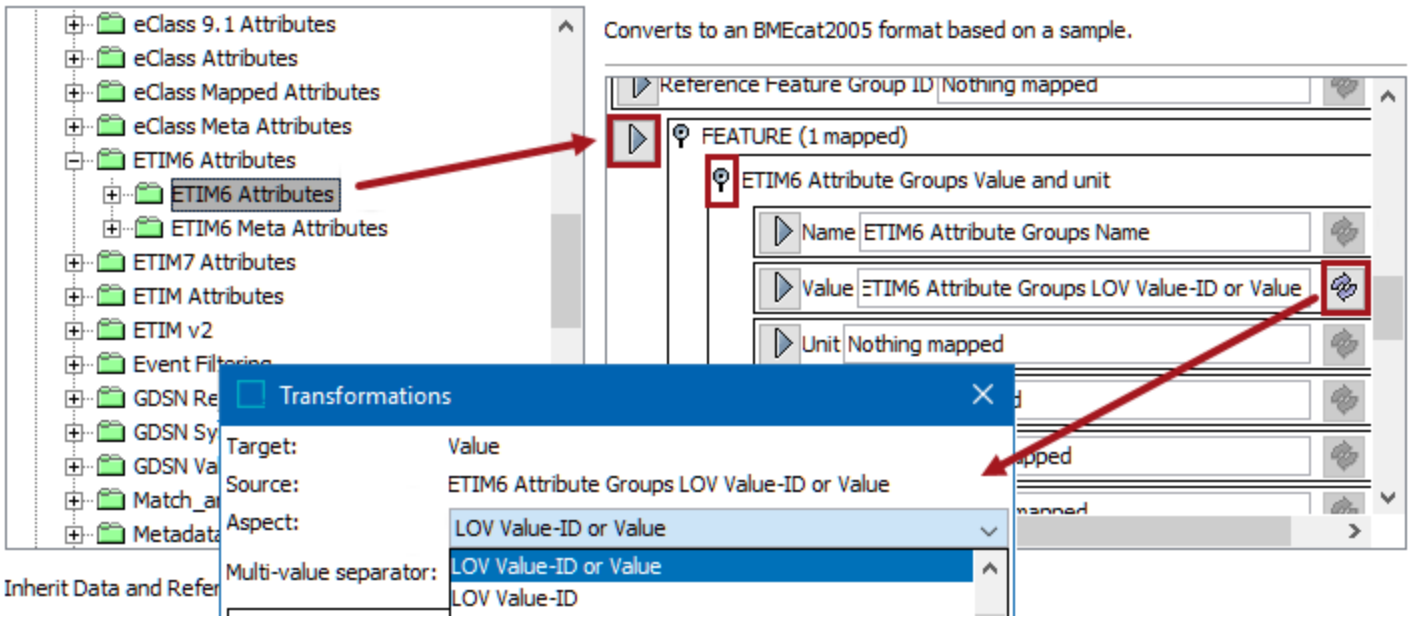
	A	B	C	D
1	<ID>	<Name>	Available Colors	Base Unit of Measure
2	18212	18212 L B	Acid_Green	CASE
3	18213	18213 M O	Aero	EA
4	18216	18216 L O	Absolute_Zero	DOZEN

Note: If the LOV attribute allows multiple values, the LOV Value IDs are exported with the multi-value separator defined in the Transformations dialog. For more information, refer to the **Multi-Value Separator - Transform Outbound** topic.

BMEcat 2005 Format Example

The following image uses the BMEcat 2005 format. The 'ETIM6 Attributes' attribute group is mapped to the BMEcat2005 FEATURE element, which then creates the 'ETIM6 Attribute Groups' flipper with its contents. The **LOV Value-ID or Value** aspect exports values for attributes without an LOV Value ID and exports the LOV Value ID for attributes with an LOV Value ID.

Map Data


eClass 9.1 Attributes
eClass Attributes
eClass Mapped Attributes
eClass Meta Attributes
ETIM6 Attributes
ETIM6 Attributes
ETIM6 Meta Attributes
ETIM7 Attributes
ETIM Attributes
ETIM v2
Converts to an BMEcat2005 format based on a sample.
Reference Feature Group ID
Nothing mapped
FEATURE (1 mapped)
ETIM6 Attribute Groups Value and unit
Name
ETIM6 Attribute Groups Name
Value
ETIM6 Attribute Groups LOV Value-ID or Value
Unit
Nothing mapped
Transformations
Target:
Value
Source:
ETIM6 Attribute Groups LOV Value-ID or Value
Aspect:
LOV Value-ID or Value
Multi-value separator:
LOV Value-ID or Value
LOV Value-ID

Unit Aspect

When working with values that may include units, use the Unit aspect option to extract the name of the unit associated with the object attribute value (not the actual value).

For general information on setting a Transformation aspect, refer to the **Aspect - Transform Outbound** topic.

Consider an attribute with the values '12 mon,' '3 dy,' and '4 wk' for the Lead Time attribute, as shown in the following image. Exporting using the transformation aspect 'Unit' outputs the name of the unit on the exported products respectively as 'mon,' 'dy,' and 'wk.'

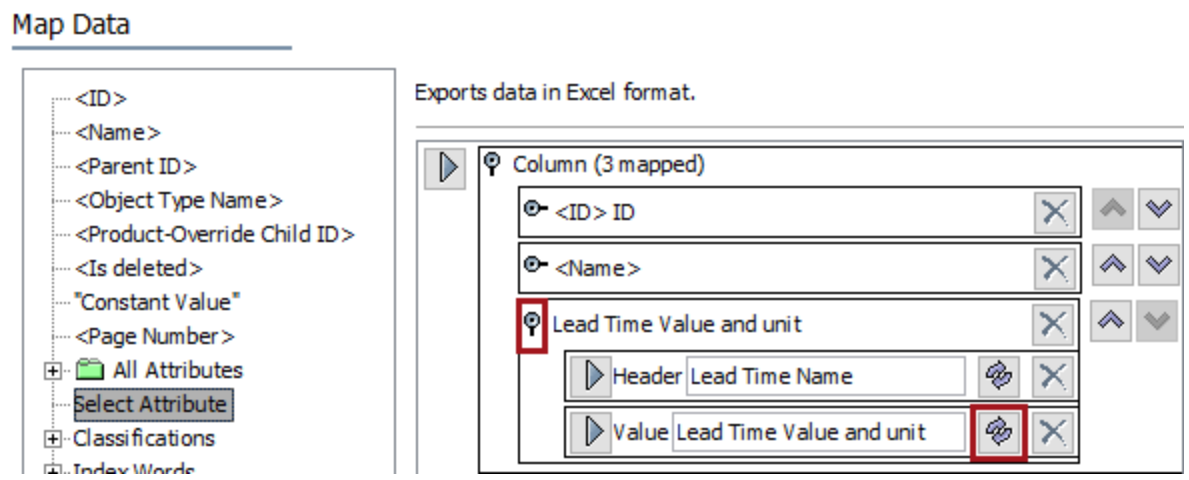
ID	Name	Lead Time
18212	18212 L B	12 mon
18213	18213 M O	3 dy
18216	18216 L O	4 wk

Additional options for exporting unit aspect information are included in the following topics:

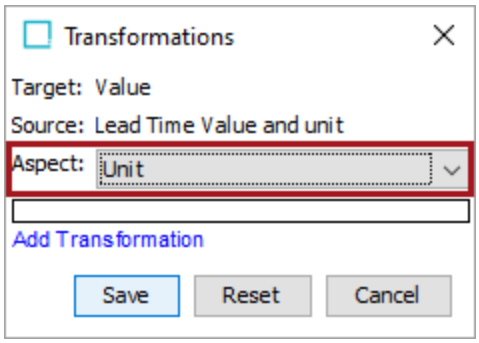
- Unit ID Aspect
- Value and Unit Aspect
- Value and Unit Meta Data Aspect

Configuration

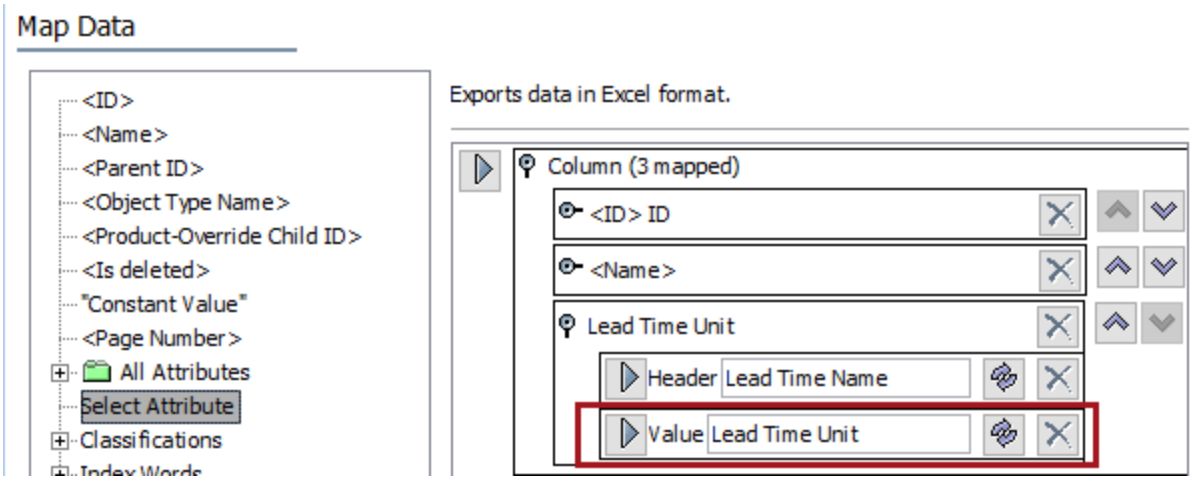
1. In the output tool, select the data to be exported. For more information, refer to the **Data Exchange** topic.
2. On the Map Data step, map the required attribute using the **Select Attribute** data source. For more information, refer to **Mapping Attributes via Select Attribute** section of the **Attributes and Data Containers - Data Source Outbound** topic.
3. In the right panel, click the flipper for the attribute value that should be extracted without the unit to display the Header and Value elements.
4. On the Value row, click the **Transformation** button to display the Transformations dialog. For number attributes with units, 'Value and unit' is the default aspect.



5. In the **Transformations** dialog, from the **Aspect** list, select the **Unit** aspect and click the **Save** button. In our example, the **Value** aspect will export only the number part of '12,' while the **Value and unit** aspect exports both number and unit of '12 mon.'



The Value element is displayed with the Lead Time attribute using the 'Unit' aspect.



6. Complete any additional mappings and initiate the export.

Results

The attribute ID and name are exported, along with only the unit portion of the attribute value.

	A	B	C
1	<ID>	<Name>	Lead Time
2	18212	18212 L B	mon
3	18213	18213 M O	dy
4	18216	18216 L O	wk

Unit ID Aspect

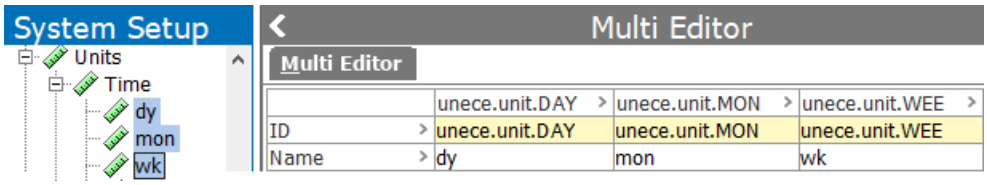
When working with values that may include units, use the Unit ID aspect option to extract the ID of the unit associated with the object attribute value (not the actual value).

For general information on setting a Transformation aspect, refer to the **Aspect - Transform Outbound** topic.

Consider an attribute with the value '12 mon,' '3 dy,' and '4 wk' for the Lead Time attribute, as shown in the following image.

ID	Name	Lead Time
18212	18212 L B	12 mon
18213	18213 M O	3 dy
18216	18216 L O	4 wk

Exporting using the transformation aspect 'Unit ID' outputs the ID of the unit as defined in the Units 'Time' node in System Setup. The exported product values are then exported respectively as 'unece.unit.MON,' 'unece.unit.DAY,' and 'unece.unit.WEE' based on their selected units.

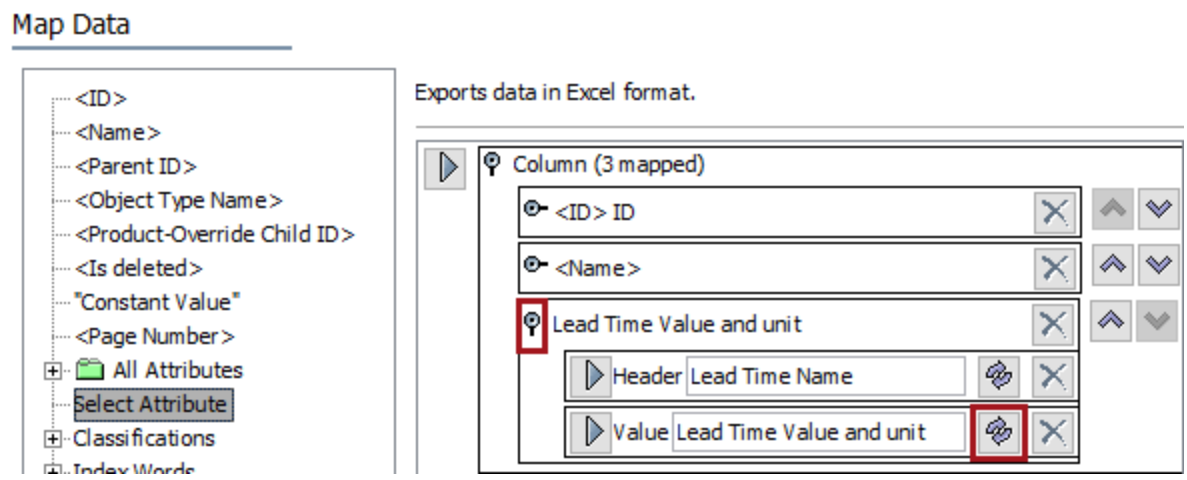


Additional options for exporting unit aspect information are included in the following topics:

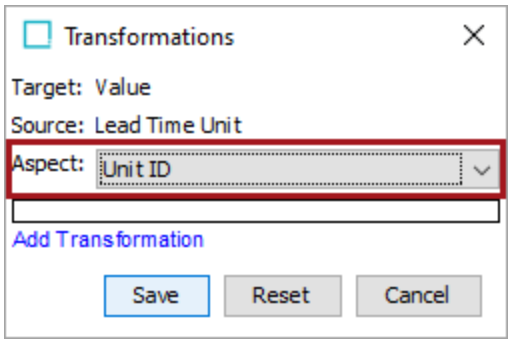
- Unit Aspect
- Value and Unit Aspect
- Value and Unit Meta Data Aspect

Configuration

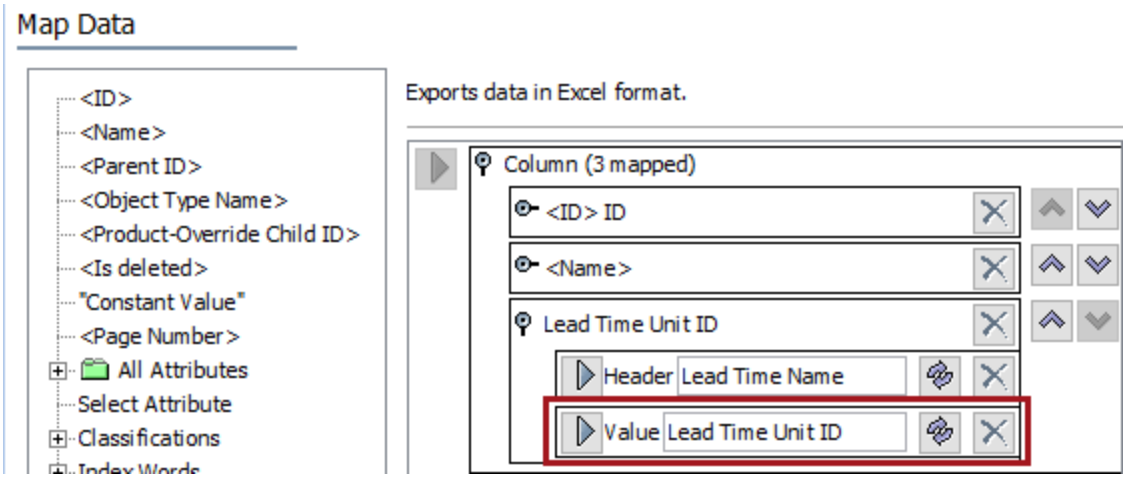
1. In the output tool select the data to be exported. For more information, refer to the **Data Exchange** topic.
2. On the Map Data step, map the required attribute using the **Select Attribute** data source. For more information, refer to **Mapping Attributes via Select Attribute** section of the **Attributes and Data Containers - Data Source Outbound** topic.
3. In the right panel, click the flipper for the attribute value that should be extracted without unit to display the Header and Value elements.
4. On the Value row, click the **Transformation** button to display the Transformations dialog. For number attributes with units, 'Value and unit' is the default aspect.



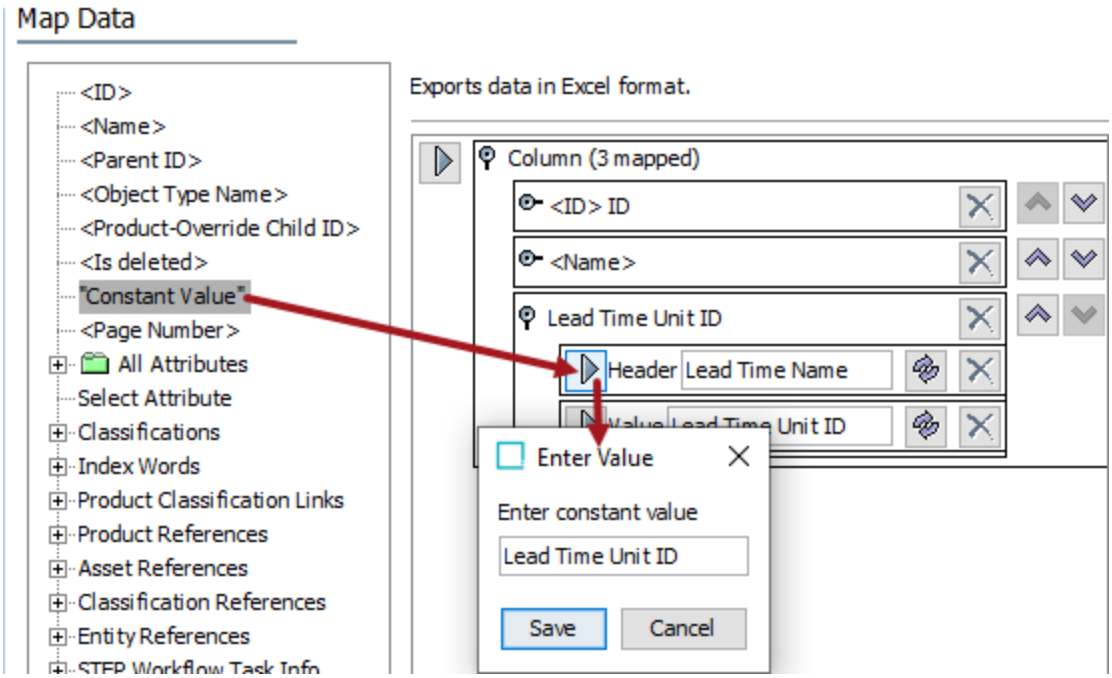
5. In the **Transformations** dialog, from the **Aspect** list, select the **Unit ID** aspect and click the **Save** button. In our example, the Unit ID aspect will export only the ID of the 'month' unit as 'unece.unit.MON', the **Value** aspect will export only the number part of '12,' while the **Value and unit** aspect exports the number and unit of '12 mon.'



The Value element is displayed with the Lead Time attribute using the 'Unit ID' aspect.



6. If desired, use a Constant Value mapping to correctly indicate the contents of the Unit ID column, and click Save. For example, printing 'Lead Time Unit ID' instead of the default 'Lead Time' text. For more information, refer to the **Constant Value - Data Source Outbound** topic.



7. Complete any additional mappings and initiate the export.

Results

The attribute ID and name are exported, along with only the unit ID for the attribute value.

	A	B	C
1	<ID>	<Name>	Lead Time Unit ID
2	18212	18212 L B	unece.unit.MON
3	18213	18213 M O	unece.unit.DAY
4	18216	18216 L O	unece.unit.WEE

Value and Unit Aspect

When working with values that may include units, use the Value and Unit aspect option to extract the actual value and the unit (if any) associated with the object attribute value.

For general information on setting a Transformation aspect, refer to the **Aspect - Transform Outbound** topic.

Consider an attribute with the value '12 ft,' '3 in,' and '4 cm' for the Height attribute, as shown in the following image. Exporting using the transformation aspect 'Value and Unit' outputs the value and the name of the unit on the exported products.

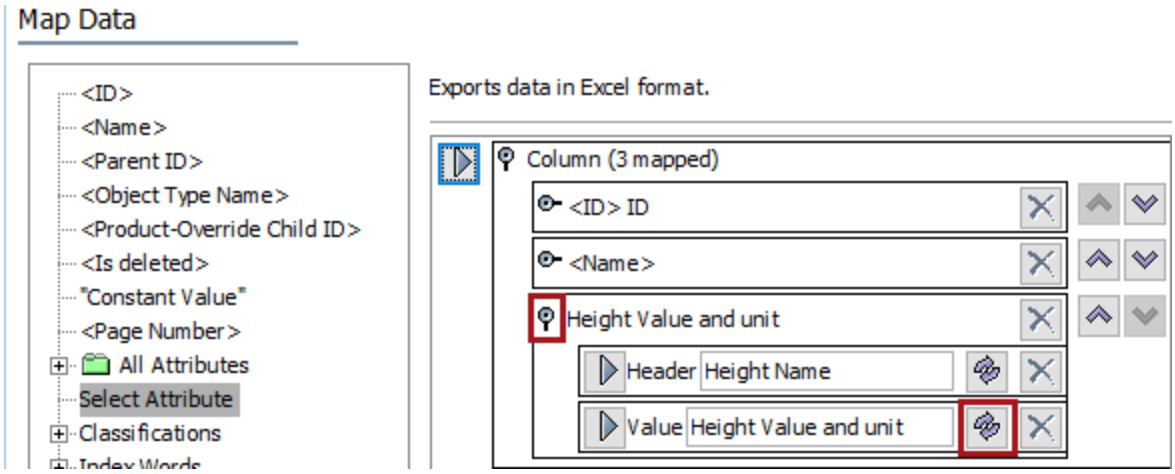
ID	Name	Height
18212	18212 L B	12 ft
18213	18213 M O	3 in
18216	18216 L O	4 cm

Additional options for exporting value aspect information are included in the following topics:

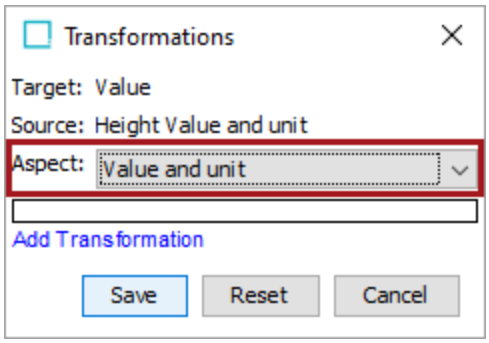
- Value Aspect
- Value and Unit Meta Data Aspect

Configuration

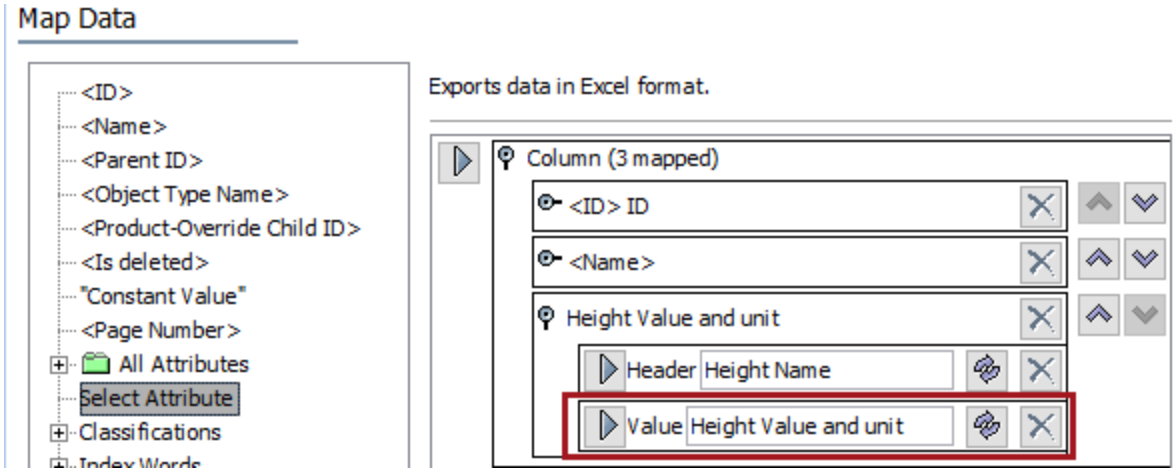
1. In the output tool select the data to be exported. For more information, refer to the **Data Exchange** topic.
2. On the Map Data step, map the required attribute using the **Select Attribute** data source. For more information, refer to **Mapping Attributes via Select Attribute** section of the **Attributes and Data Containers - Data Source Outbound** topic.
3. In the right panel, click the flipper for the attribute value that should be extracted with the unit to display the Header and Value elements.
4. On the Value row, click the **Transformation** button to display the Transformations dialog. For number attributes with units, 'Value and unit' is the default aspect.



5. In the **Transformations** dialog, from the **Aspect** list, select the **Value and Unit** aspect and click the **Save** button. In our example, the **Value** aspect will export only the number part of '12,' while the **Value and unit** aspect exports both number and unit of '12 ft.' For number attributes with units, 'Value and unit' is the default aspect.



The Value element is displayed with the Height attribute using the 'Value and Unit' aspect.



6. Complete any additional mappings and initiate the export.

Results

The attribute ID and name are exported, along with only the value and unit portion of the attribute value.

	A	B	C
1	<ID>	<Name>	Height
2	18212	18212 L B	12 ft
3	18213	18213 M O	3 in
4	18216	18216 L O	4 cm

Value and Unit Meta Data Aspect

When working with values that may include units on metadata attributes, use the 'Value and unit Meta Data [name of attribute]' aspect option to extract the actual value and the unit (if any) associated with the object attribute value.

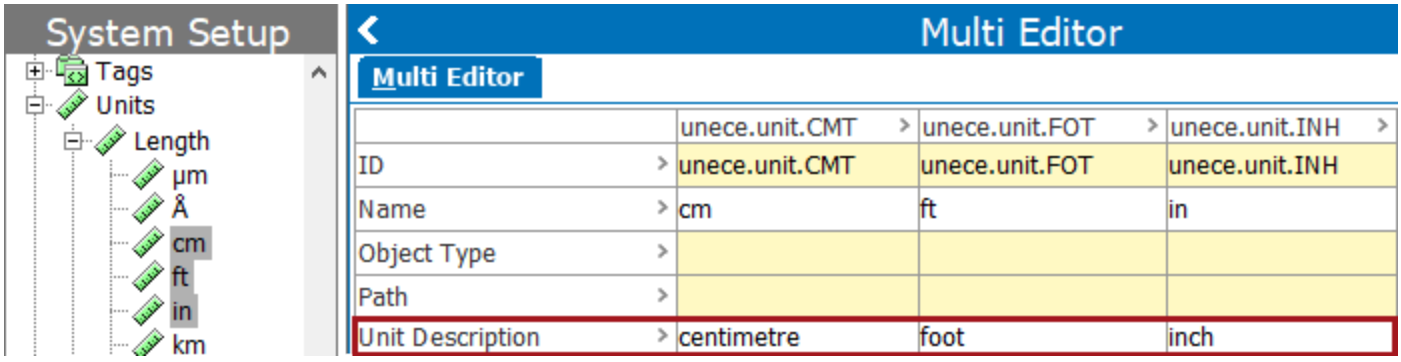
The metadata options available for selection are defined by the metadata attributes that are valid on the objects being exported.

For general information on setting a Transformation aspect, refer to the **Aspect - Transform Outbound** topic.

Consider an attribute with the value '12 ft,' '3 in,' and '4 cm' for the Height attribute, as shown in the following image:

ID	Name	Height
18212	18212 L B	12 ft
18213	18213 M O	3 in
18216	18216 L O	4 cm

The units for the Height attribute are supplied by the Length units as defined in System Setup. A metadata attribute 'Unit Description' is valid on the Length unit object types 'cm,' 'ft,' and 'in.' The metadata values represented the spelled-out versions of the length abbreviations.



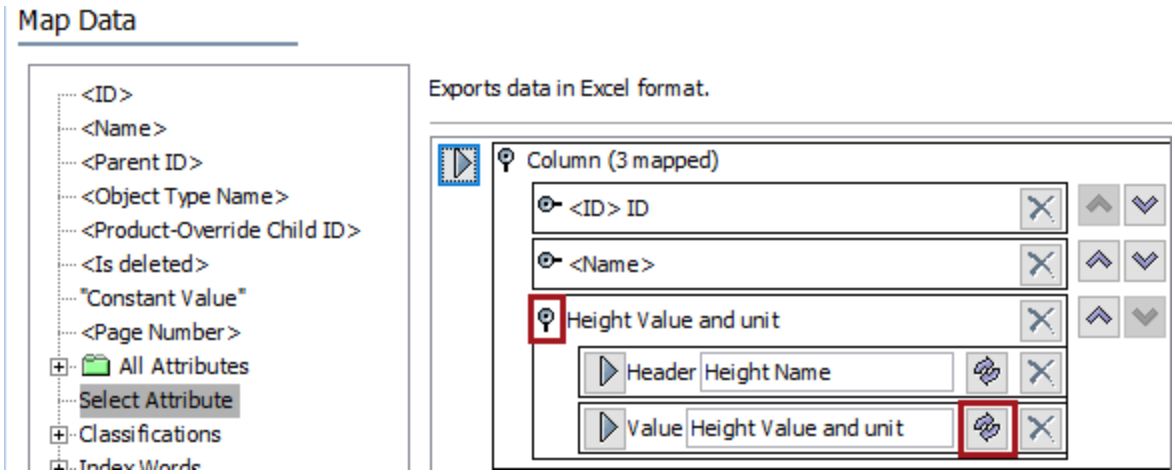
Using the 'Value and unit Meta Data Unit Description' aspect allows products with the values '12 ft,' '3 in,' and '4 cm' to export the 'Unit Description' values of 'feet,' 'inches,' and 'centimeters.'

Additional options for exporting value aspect information are included in the following topics:

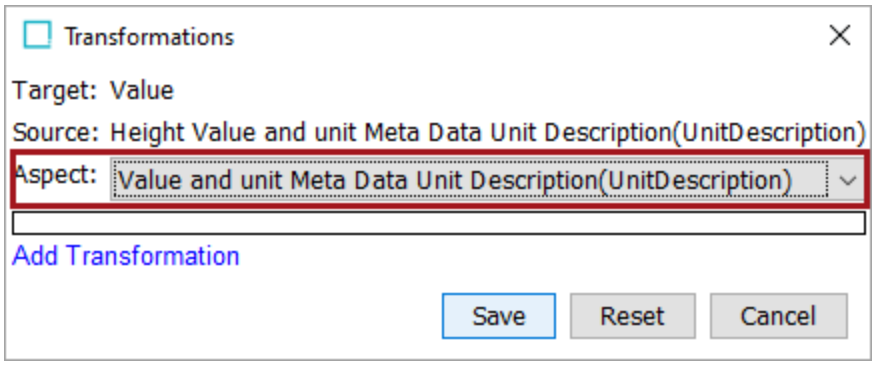
- Value Aspect
- Value and Unit Aspect

Configuration

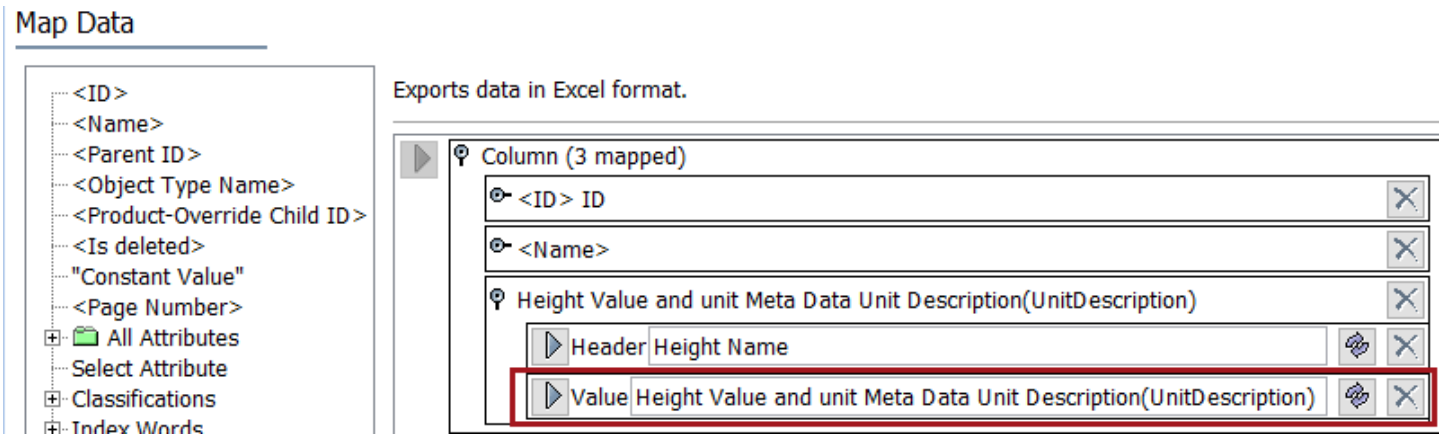
1. In the output tool, select the data to be exported. For more information, refer to the **Data Exchange** topic.
2. On the Map Data step, map the required attribute using the **Select Attribute** data source. For more information, refer to **Mapping Attributes via Select Attribute** section of the **Attributes and Data Containers - Data Source Outbound** topic.
3. In the right panel, click the flipper for the attribute value that should be extracted with the unit to display the Header and Value elements.
4. On the Value row, click the **Transformation** button to display the Transformations dialog. For number attributes with units, 'Value and unit' is the default aspect.



5. In the **Transformations** dialog, from the **Aspect** list, select the **Value and unit Meta Data [name of attribute]** aspect and click the **Save** button. In our example, this aspect will export the number part of '12' and the description part of 'foot.'



The Value element is displayed with the Height attribute using the 'Value and unit Meta Data Unit Description (UnitDescription)' aspect.



6. Complete any additional mappings and initiate the export.

Results

The attribute ID and name are exported, along with only the value and unit description (metadata) of the attribute value.

	A	B	C
1	<ID>	<Name>	Height
2	18212	18212 L B	12 foot
3	18213	18213 M O	3 inch
4	18216	18216 L O	4 centimetre

Value Aspect

When working with values, use the Value aspect option to extract the actual value (if any) associated with the attribute. If the value includes units, they are not exported.

For general information on setting a Transformation aspect, refer to the **Aspect - Transform Outbound** topic.

Consider an attribute with the value '12 ft,' '3 in,' and '4 cm' for the Height attribute, as shown in the following image. Exporting using the transformation aspect 'Value and Unit' outputs the value and the name of the unit on the exported products.

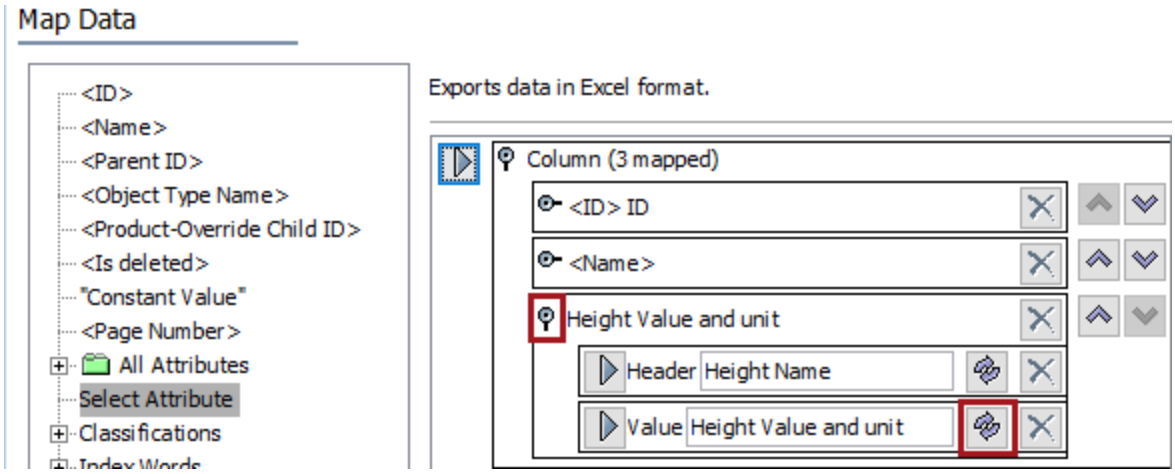
ID	Name	Height
18212	18212 L B	12 ft
18213	18213 M O	3 in
18216	18216 L O	4 cm

Additional options for exporting value aspect information are included in the following topics:

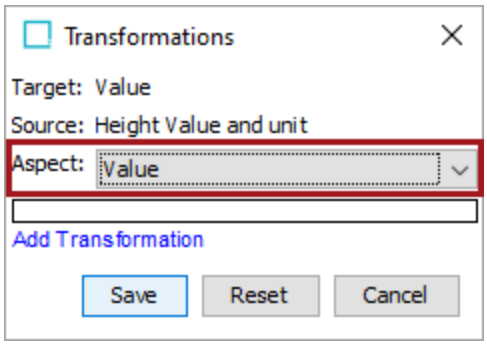
- Value and Unit Aspect
- Value and Unit Meta Data Aspect

Configuration

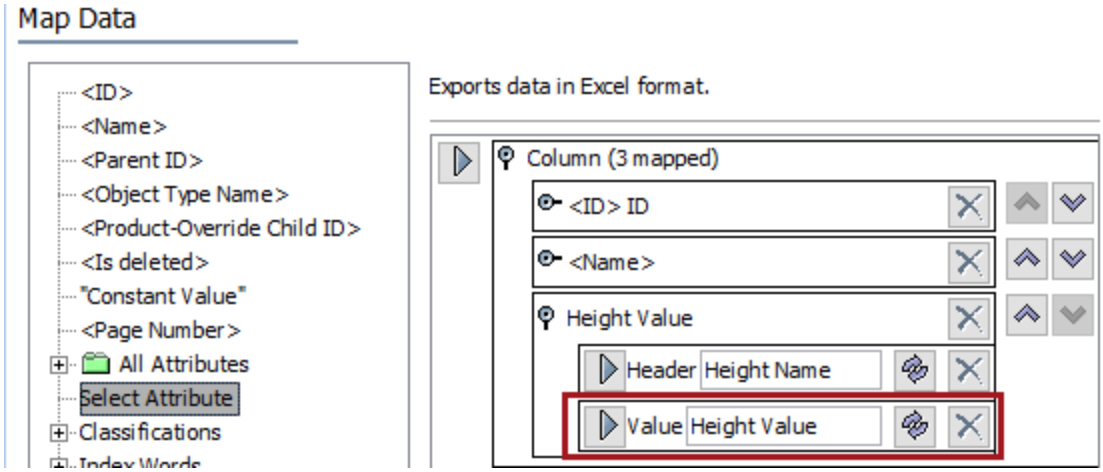
1. In the output tool, select the data to be exported. For more information, refer to the **Data Exchange** topic.
2. On the Map Data step, map the required attribute using the **Select Attribute** data source. For more information, refer to **Mapping Attributes via Select Attribute** section of the **Attributes (and Data Containers) - Data Source Outbound** topic.
3. In the right panel, click the flipper for the attribute value that should be extracted with the unit to display the Header and Value elements.
4. On the Value row, click the **Transformation** button to display the Transformations dialog. For number attributes with units, 'Value and unit' is the default aspect.



5. In the **Transformations** dialog, from the **Aspect** list, select the **Value** aspect and click the **Save** button. In our example, the **Value** aspect will export the number '12' but not the unit 'ft.'



The Value element is displayed with the Height attribute using the 'Value' aspect.



6. Complete any additional mappings and initiate the export.

Results

The attribute ID and name are exported, along with only the value portion of the attribute value.

	A	B	C
1	<ID>	<Name>	Height
2	18212	18212 L B	12
3	18213	18213 M O	3
4	18216	18216 L O	4

Transformation Examples

Regardless if the data is being imported or exported, or is being used within STEP, generally the transformation options have the same effect.

Transformations are available using the STEP Workbench import and export processes for formats that require **Data Mapping** as discussed in the **Data Exchange** documentation, and also using **Attribute Transformations** for attributes in the **System Setup** documentation.

For additional help importing data, refer to the **Creating a Data Import** section of the **Data Exchange** documentation. And, for additional help exporting data, refer to the **Creating a Data Export** section of the Data Exchange documentation.

Transformation options and availability are defined in the **Transformations** topic in the **Resource Materials** online help.

For examples of some of the most commonly used transformations, refer to the sections below:

- Concatenate Column with Unit Data Example
- Concatenate Data Using Variables Example
- Concatenate into Single-Valued Data Example
- Dimension Point for Single Dimension Data Example
- Extract Characters Between and Extract Characters After Example
- Extract Characters at Position Example
- Format Number Example
- Header and Value Aspects Example
- If Equals Source Example
- Insert Text Example
- Math Operation Example
- Merge Attributes and Constant Example
- Merge Data for Multi-Valued Data Example
- Replace Substrings of the Value Example
- Split and Extract Data Example
- Replace The Whole Value Example
- Transform Date by Locale or by Pattern
- Transform Using Attribute Transformation Example
- Transformation Lookup Table Example

Additional examples using the Transformation Aspects parameter, refer to the **Aspect - Transform Outbound** topic.

Concatenate Column with Unit Data Example

Transformations can improve the quality of the data you want to import. For general information about transformations, refer to the **Inbound Map Data - Transform** section.

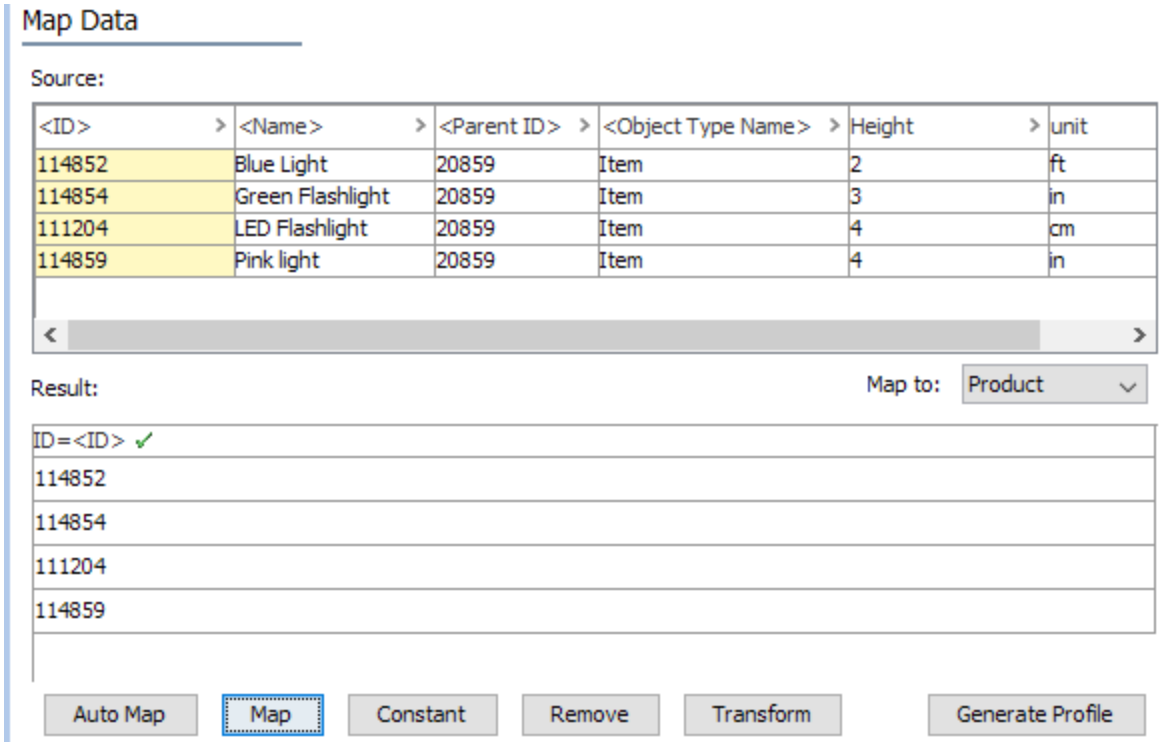
In this example, the source file is a CSV or XLS file with product data where one column has a value and the following column has the corresponding units for that value.

One column has numbers such as 12, 15, 22, and another column has units like mm, cm, and inches. The following options are available to properly load this into STEP.

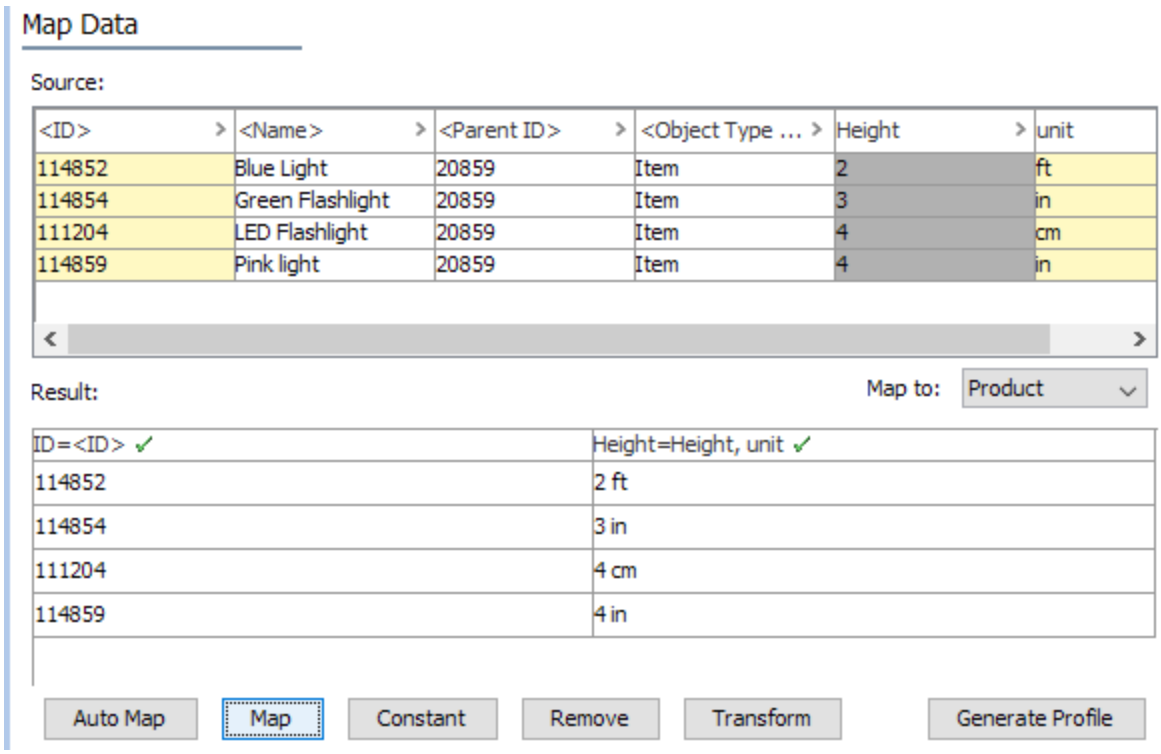
- External update: Concatenate the value and unit columns in the source file, and save the concatenated column values as text. In the import, map the concatenated column to the attribute.
- During import: Concatenate the value and unit columns using the mapping procedure as illustrated below.

Concatenate value and unit during import

1. In the Import Manager 'Map Data' step, map the product ID by selecting the column that contains the product ID, in the Source window as usual.

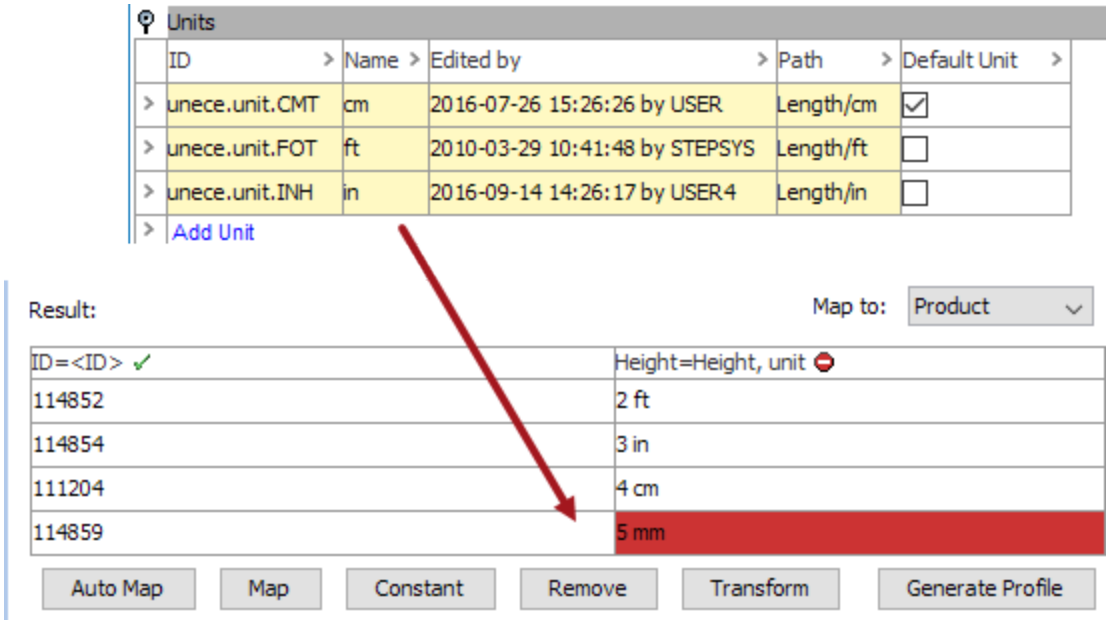


2. Select the column that contains the attribute values, in this case 'Height', and map it to the appropriate attribute.



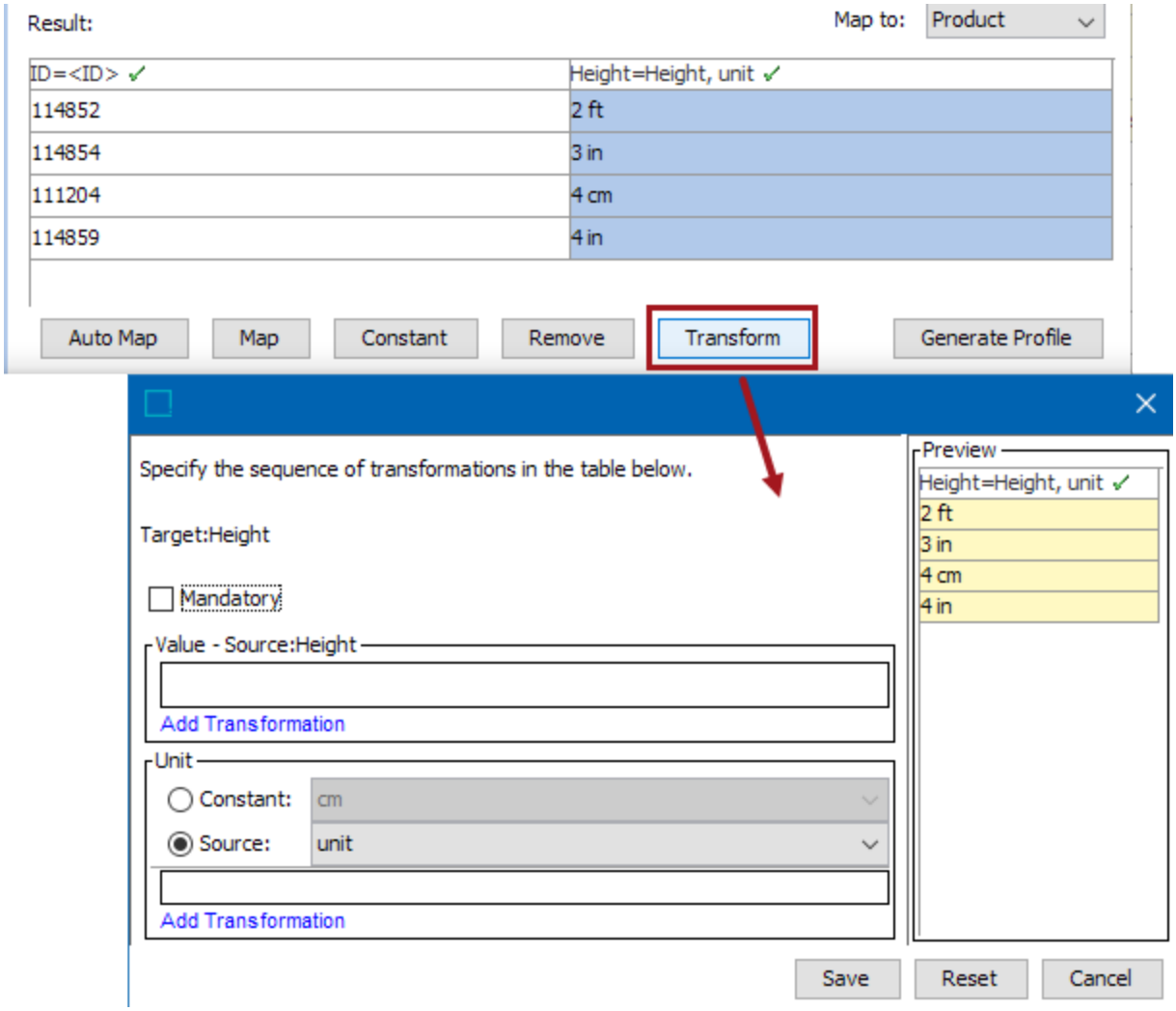
Note that the column has the corresponding units mapped automatically, and that the Result panel shows the value with the corresponding units. It can be observed in the header of the Result panel that **Height = Height, unit**. This mean that it is concatenated with the 'unit' column.

In order for it to concatenate correctly, the attribute must have the corresponding units in STEP. If it does not, an error is displayed.

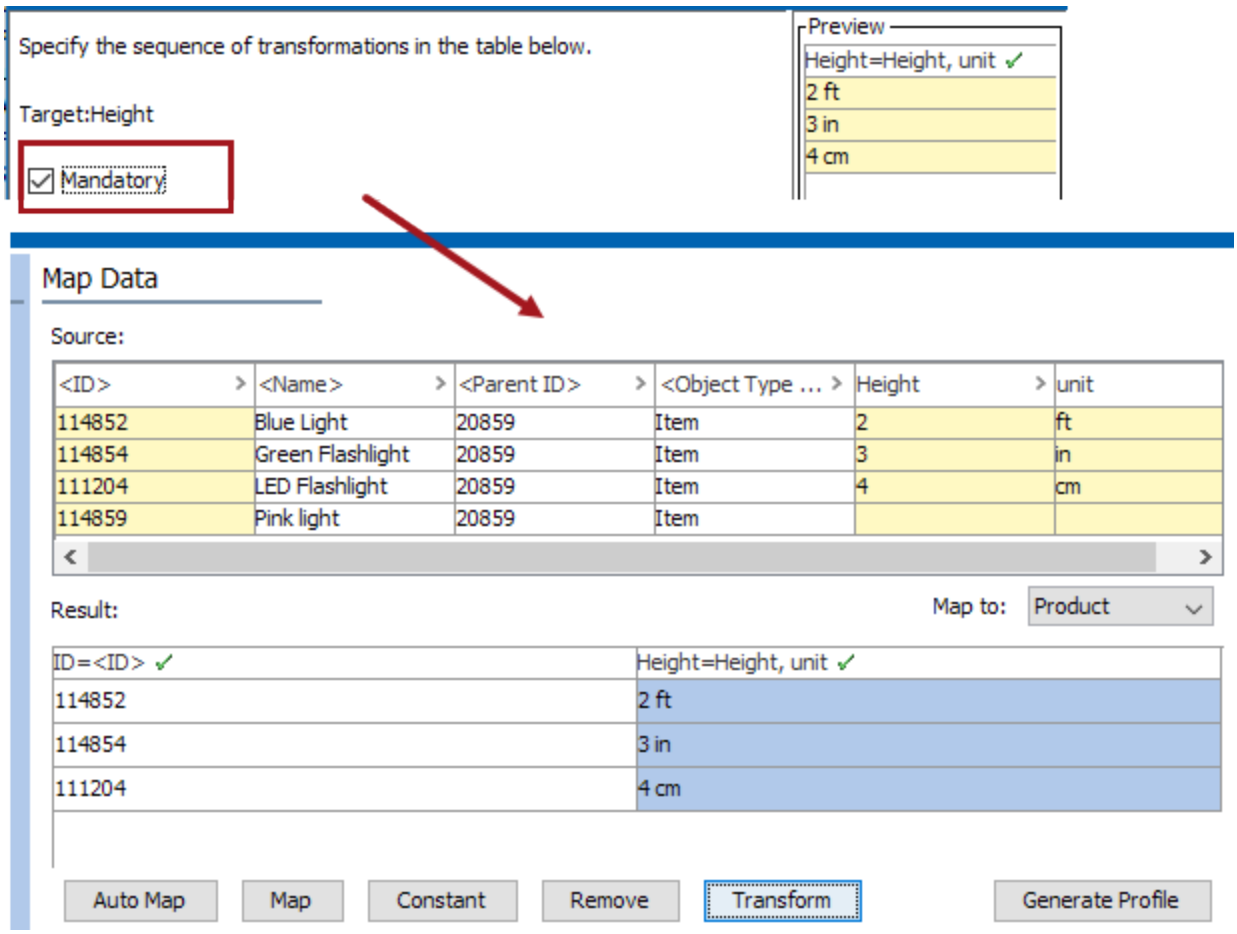


Note: For auto-mapping of the 'units' column to the attribute values, the imported file must include the units in the column following the column that holds the attribute values.

3. If you need to check the concatenation of the selected attribute column, in this case 'Height', select the attribute column in the Result filed, and click the **Transform** button. It will show the applied transformation for the mapped column.



It is worth noting that if the 'Mandatory' option is checked, it will skip the empty values and those objects will not be imported.


Specify the sequence of transformations in the table below.
Target:Height
Mandatory
Preview
Height=Height, unit ✓
2 ft
3 in
4 cm
Map Data
Source:
<ID>
<Name>
<Parent ID>
<Object Type ...
Height
unit
114852
Blue Light
20859
Item
2
ft
114854
Green Flashlight
20859
Item
3
in
111204
LED Flashlight
20859
Item
4
cm
114859
Pink light
20859
Item
Result:
Map to:
Product
ID=<ID> ✓
Height=Height, unit ✓
114852
2 ft
114854
3 in
111204
4 cm
Auto Map
Map
Constant
Remove
Transform
Generate Profile

Concatenate Data Using Variables Example

Transformations can improve the quality of the data you want to import. For general information about transformations, refer to the **Inbound Map Data - Transform** section.

In some cases, a user might need to transform both pieces of data that are to be concatenated before the concatenation. Variables make that type of transformation possible. For more information about variables, refer to the **Variable - Map Inbound** topic.

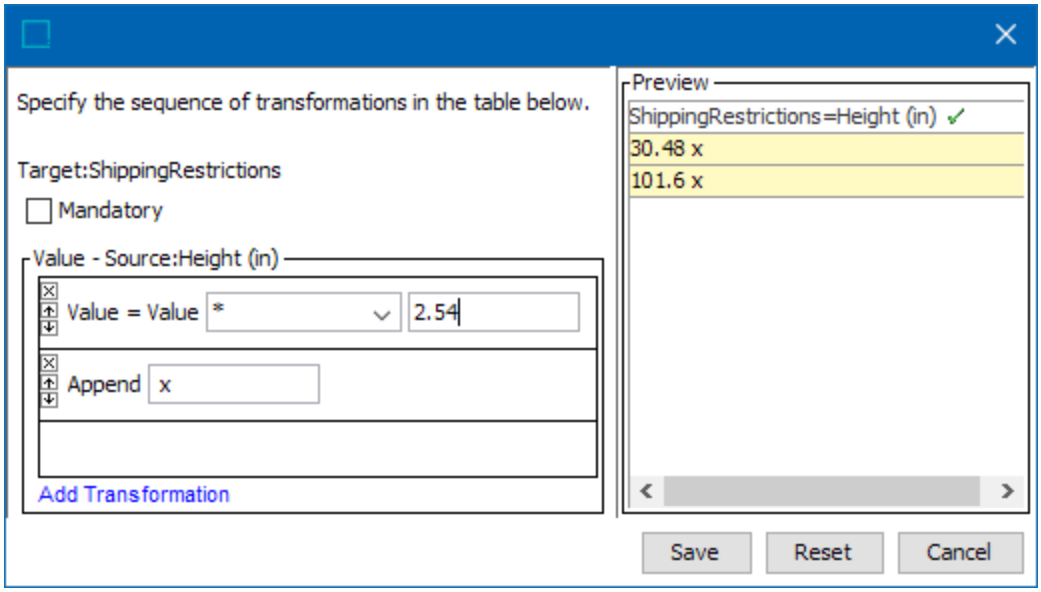
Important: Saving an import configuration saves variables and transformations, as well as the mappings, for use each time the same import is run.

This example includes the following transformations:

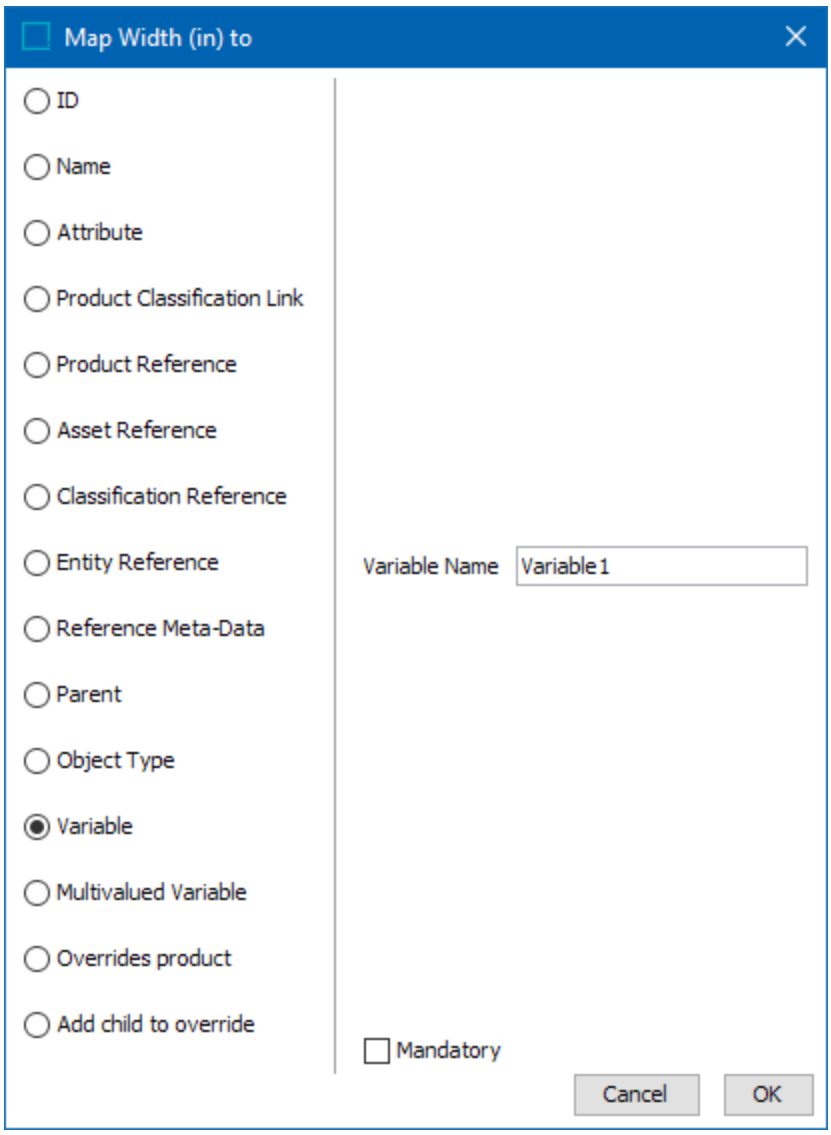
- Math operation (+,-,*,/)
- Append text
- Append from source

In this example, an import file has a 'Height (in)' and 'Width (in)' column. Both hold values that will be merged into a single multi-valued attribute named, 'ShippingRestrictions', which will be in centimeters instead of inches. The format should look like: [Width cm] x [Height cm].

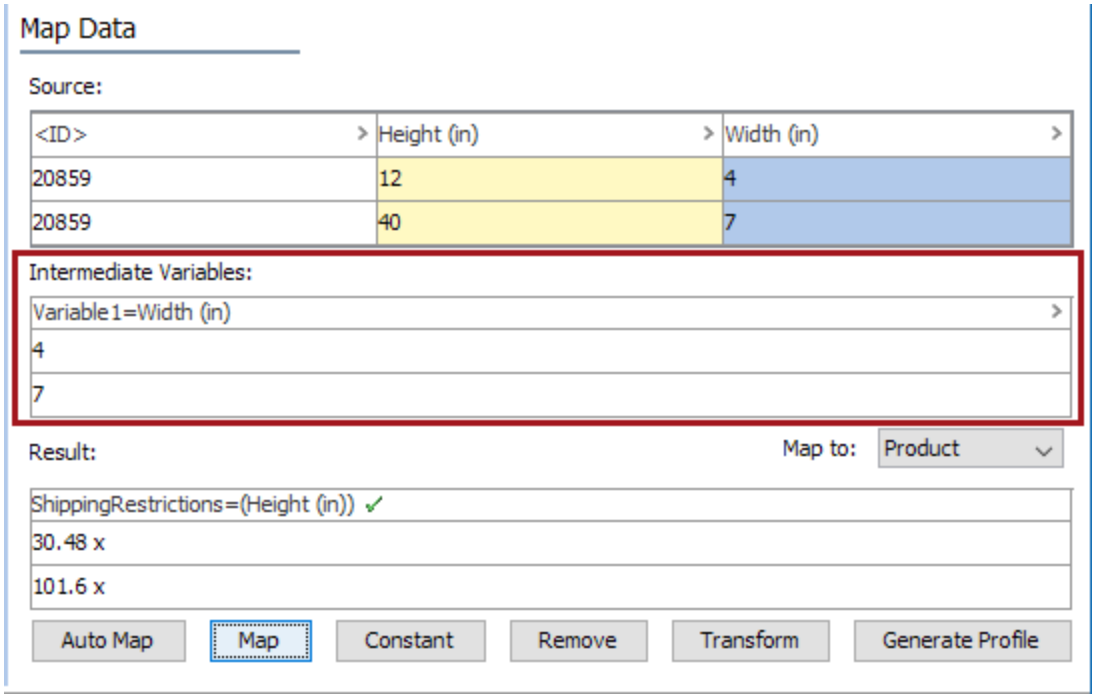
1. Map 'Height (in)' to the 'ShippingRestrictions' attribute and apply needed transformations. In this case the 'Math operation (+,-,*,/)' and the 'Append text' transformation were used.



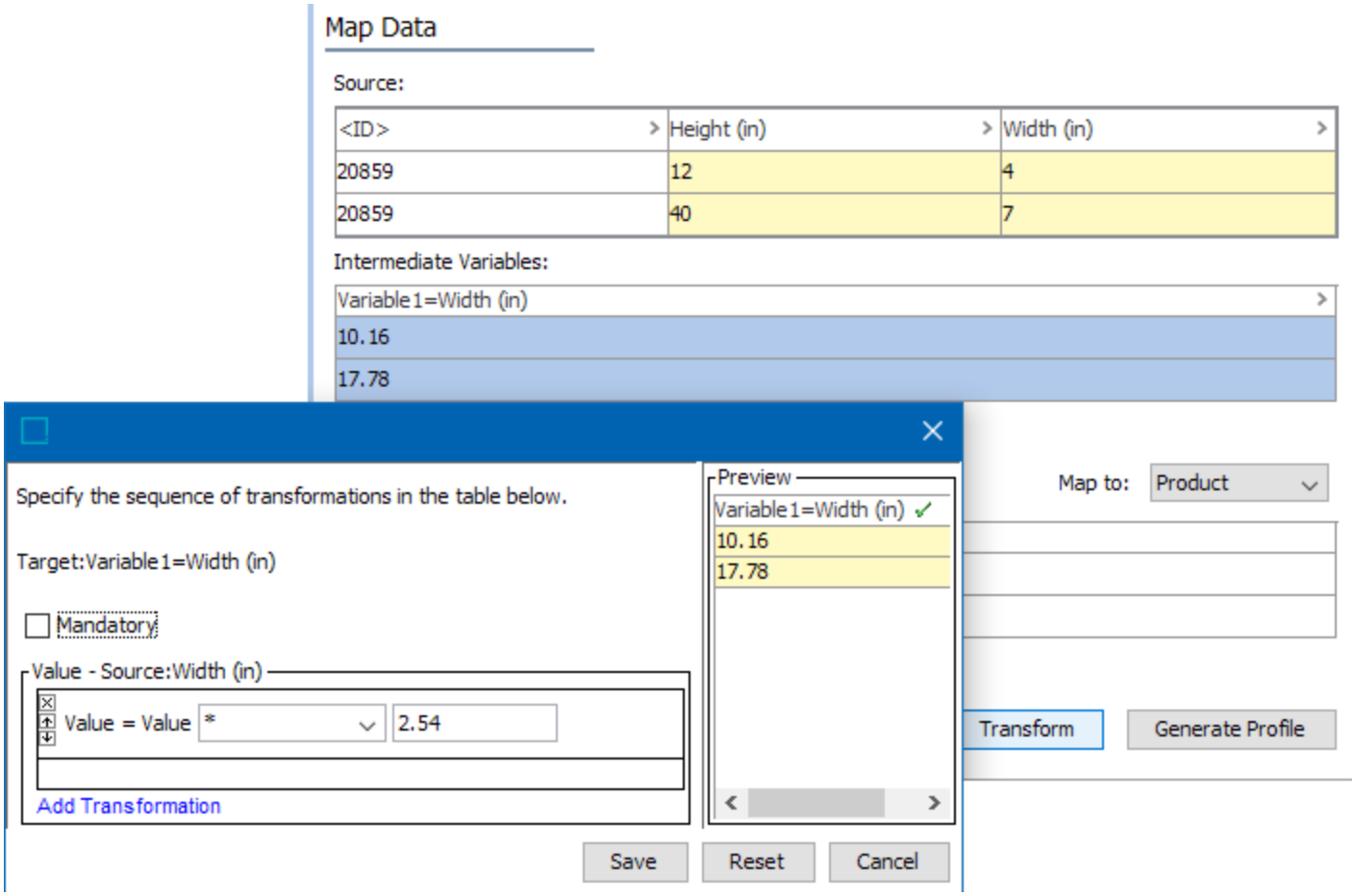
2. Manipulate the data in the 'Width (in)' column before it is appended by saving the transformations made in the first mapping, selecting the 'Width (in)' column in the Source area, and clicking the **Map** button. In the 'Map [column name] to' dialog, select the **Variable** radio button and enter a variable name.



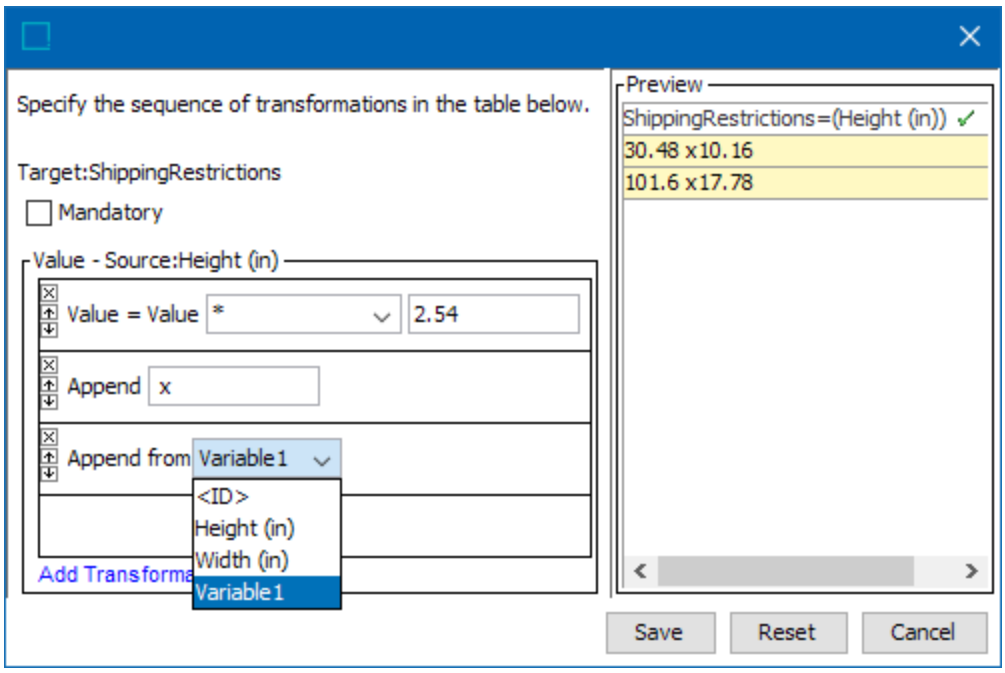
3. Click **OK** to create a new intermediate level, which is displayed between the Source and Result panels.



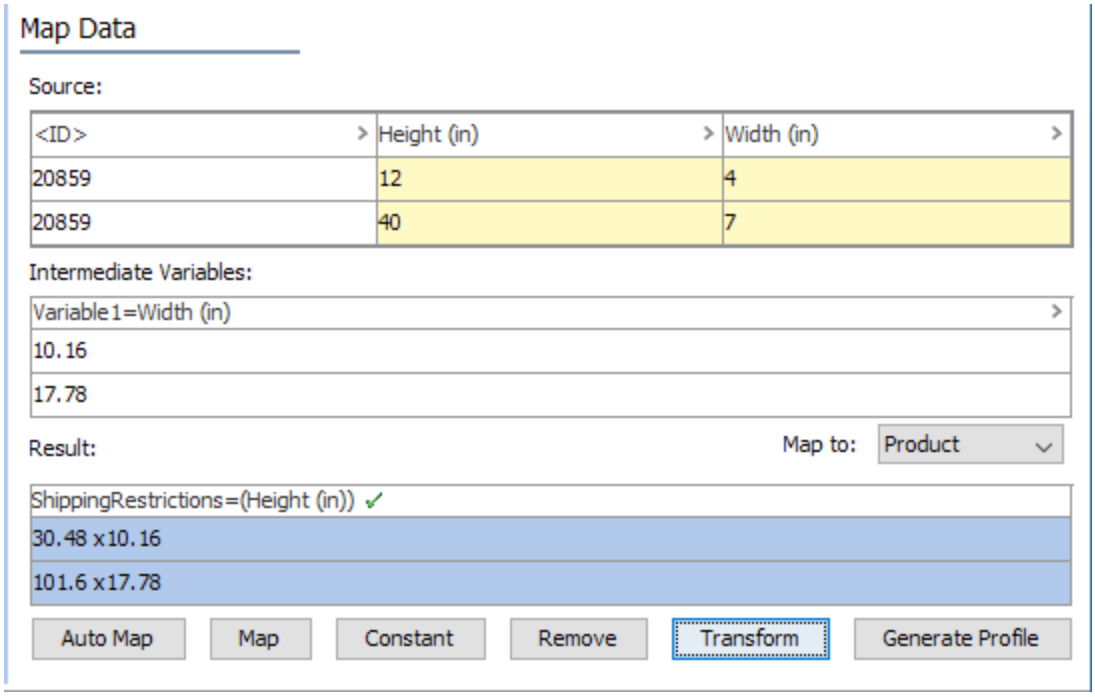
4. You can now transform this data before it is appended to the data already mapped to the 'ShippingRestrictions' attribute. Select the variable column, click **Transform**, and then add the transformation that change inches to centimeters. Click **Save**.



5. In the Result area, select the 'ShippingRestrictions=(Height (in))' column, click **Transform** and add the 'Append from source' transformation where the intermediate variable is now available. Select the variable and click **Save**.



6. The variable now displays the transformed value, which can be mapped in the same manner as source columns are mapped.



Concatenate into Single-Valued Data Example

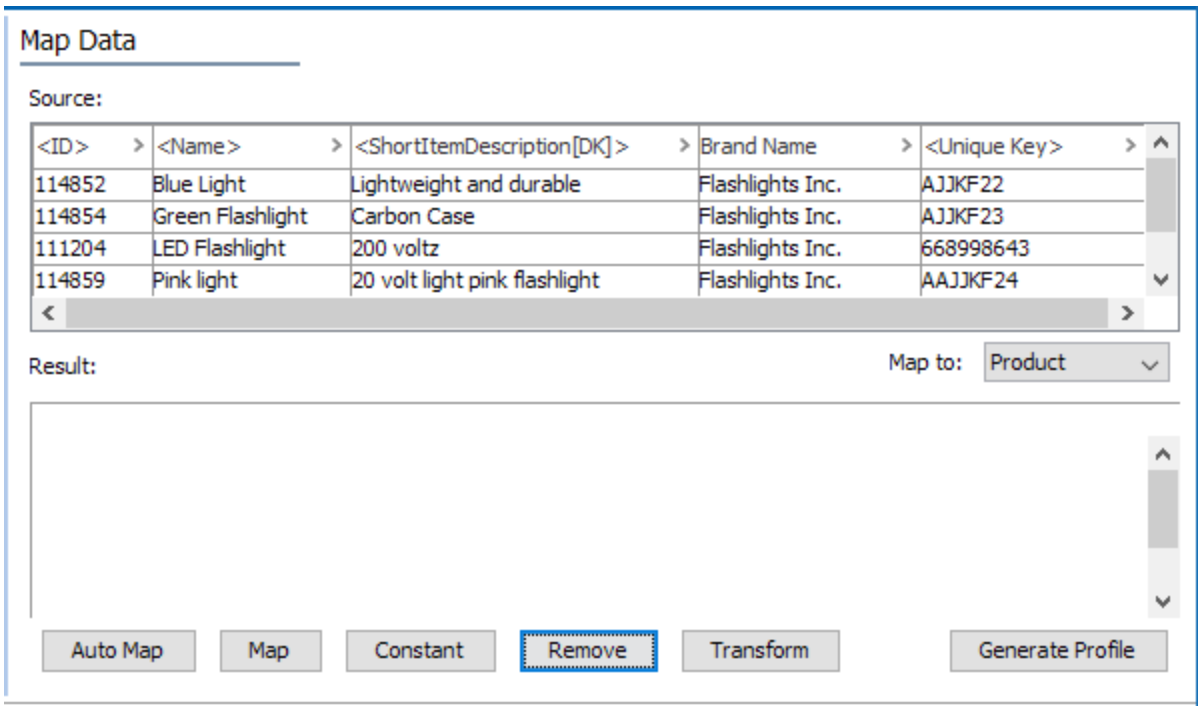
Transformations can improve the quality of the data you want to import. For general information about transformations, refer to the **Inbound Map Data - Transform** section.

In the Map Data step of the Import Manager and the IIEP configuration, you can concatenate data from different columns and map the result to a single element in STEP. You can do this for row and column based import formats such as Excel, CSV, and FixedWidth.

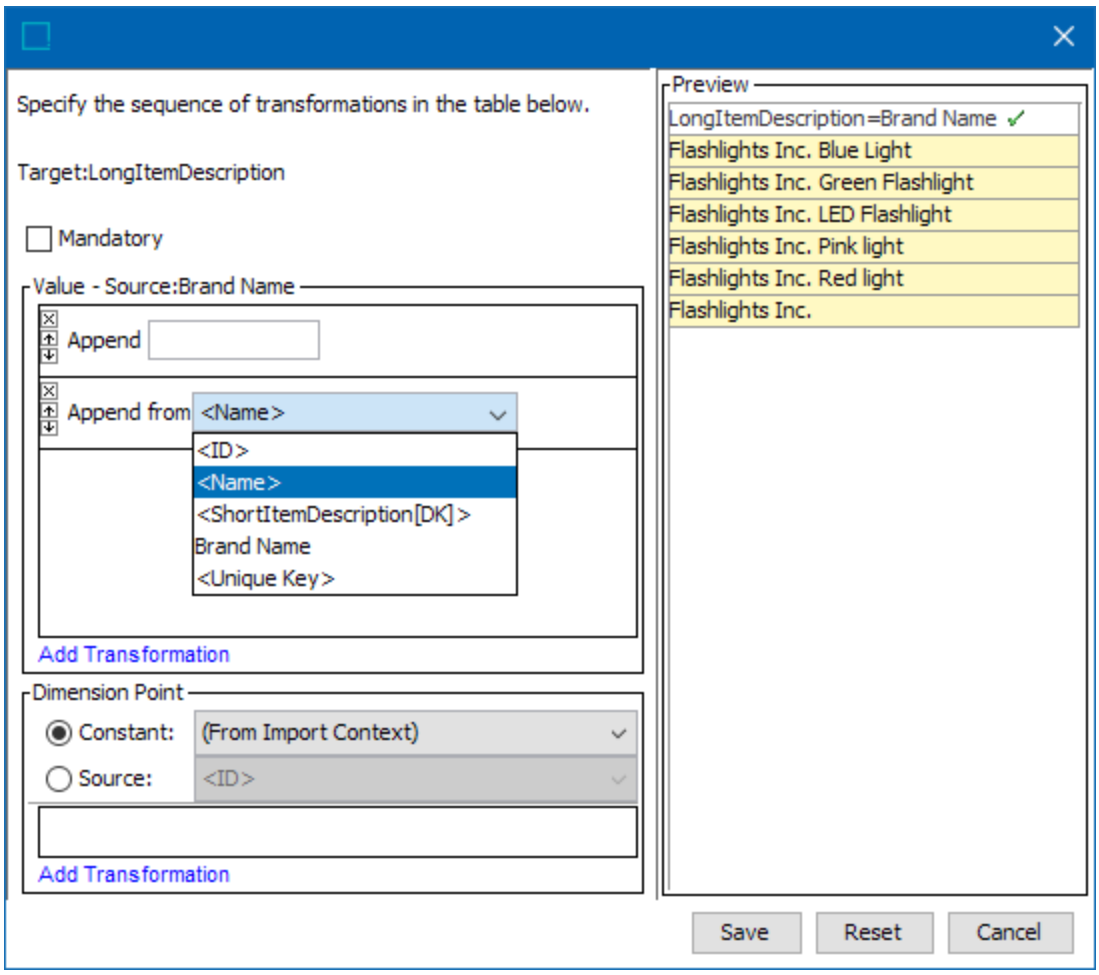
This example includes the following transformations:

- Append text
- Append from source

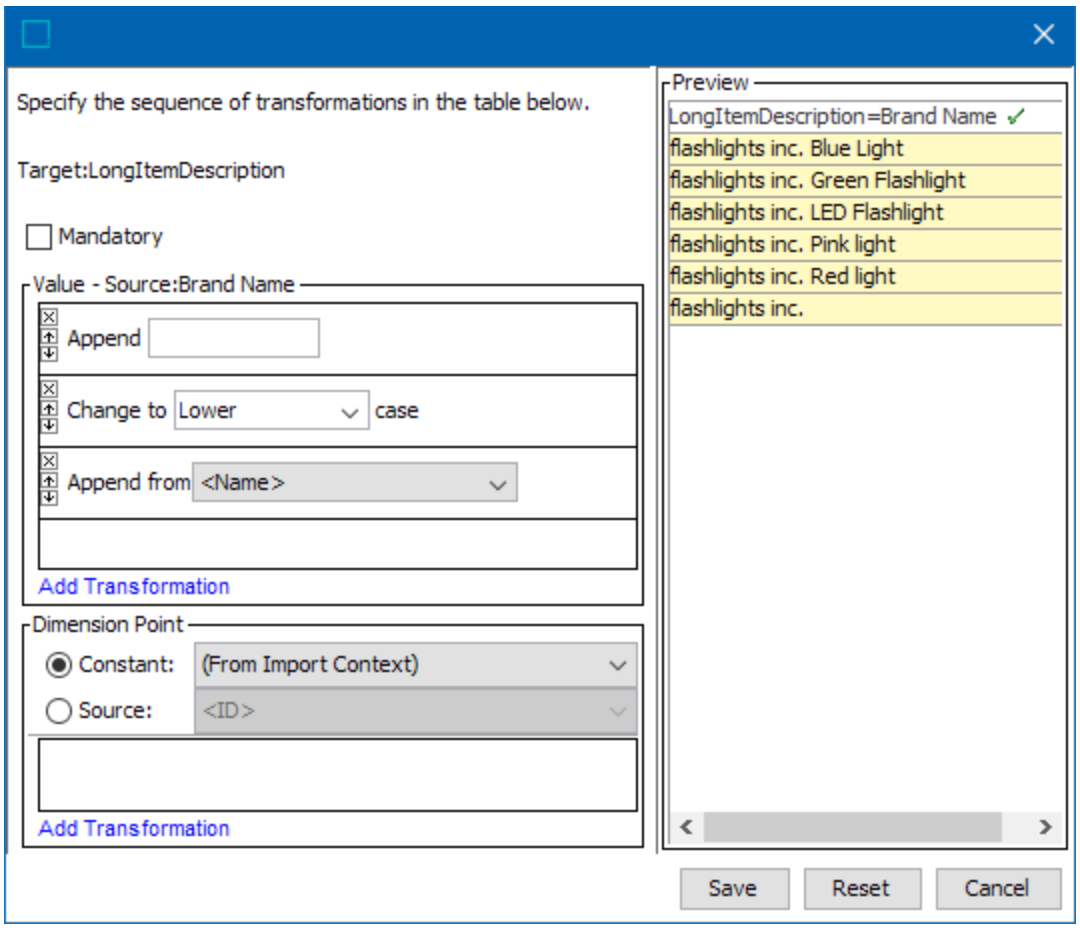
In this example, you have an import file where you want to concatenate the data in the 'Brand Name' column with the data in the '<Name>' column and map it into a attribute named 'LongItemDescription.'



1. Select the 'Brand Name' column in the **Source** area and map it to the 'LongItemDescription' attribute.
2. In the **Result** area select the 'LongItemDescription = Brand Name' column and click **Transform**.
3. In the **Transformations** dialog, add the transformation **Append from source** and select the '<Name>' column.



In this example, a space has been added to the 'Brand Name' value before the **Append from source** transformation to separate the concatenated strings. You can add any number of transformations before the **Append from source** transformation. Also, you can add transformations that are applied to the entire string.



Note: If you need to add a suffix value from a source or intermediate variable column, a **Prepend from source** transformation is available.

Dimension Point for Single Dimension Data Example

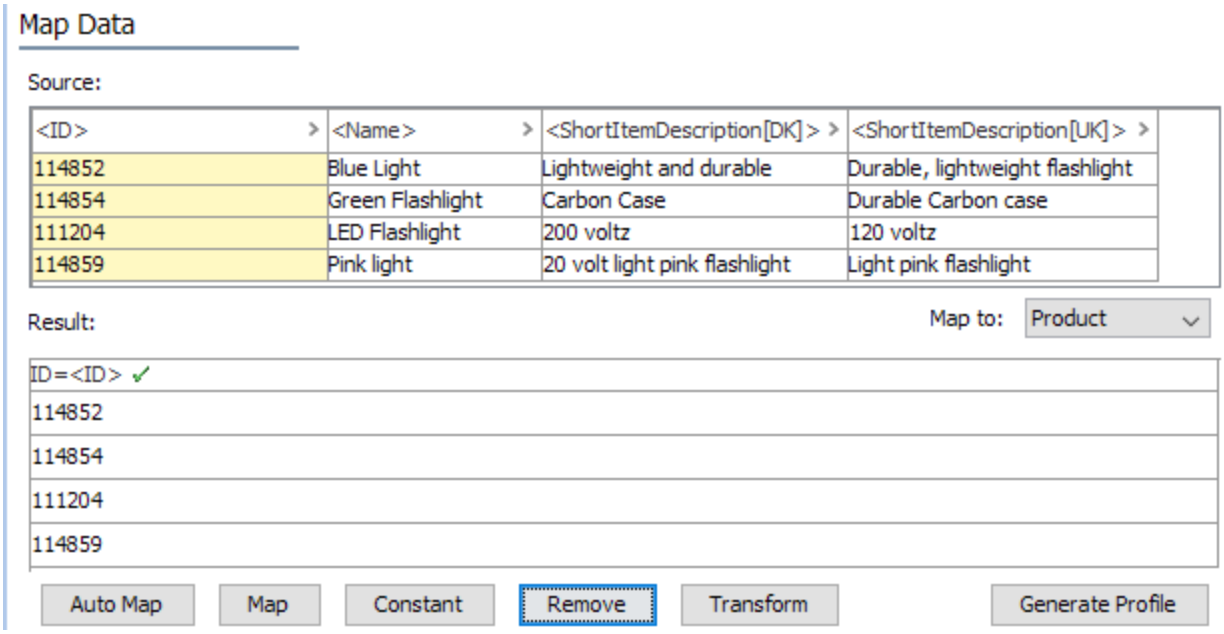
Transformations can improve the quality of the data you want to import. For general information about transformations, refer to the **Inbound Map Data - Transform** section.

This example includes the following transformations:

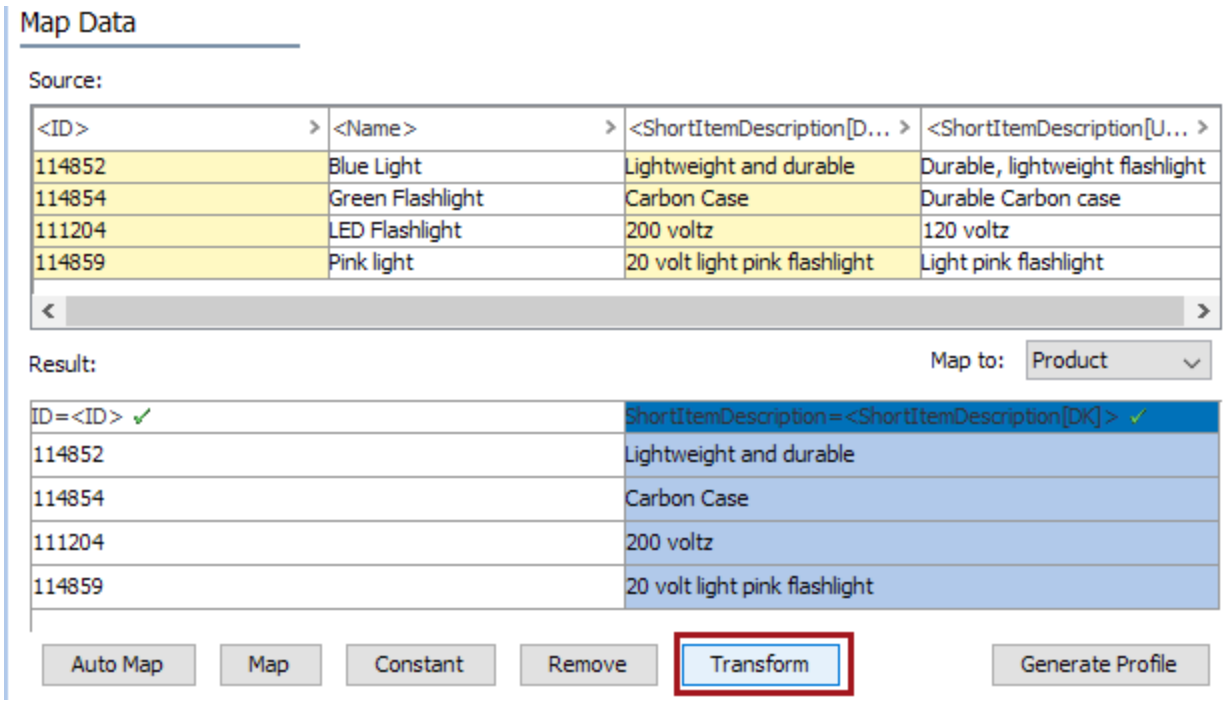
- Dimension Point

This example demonstrates import of single dimension dependent data (language in this case) from either a CSV, Excel, or Generic XML file. The title of the column includes the language of the values (DK and UK). The input data is displayed in the Source section of the Map Data step.

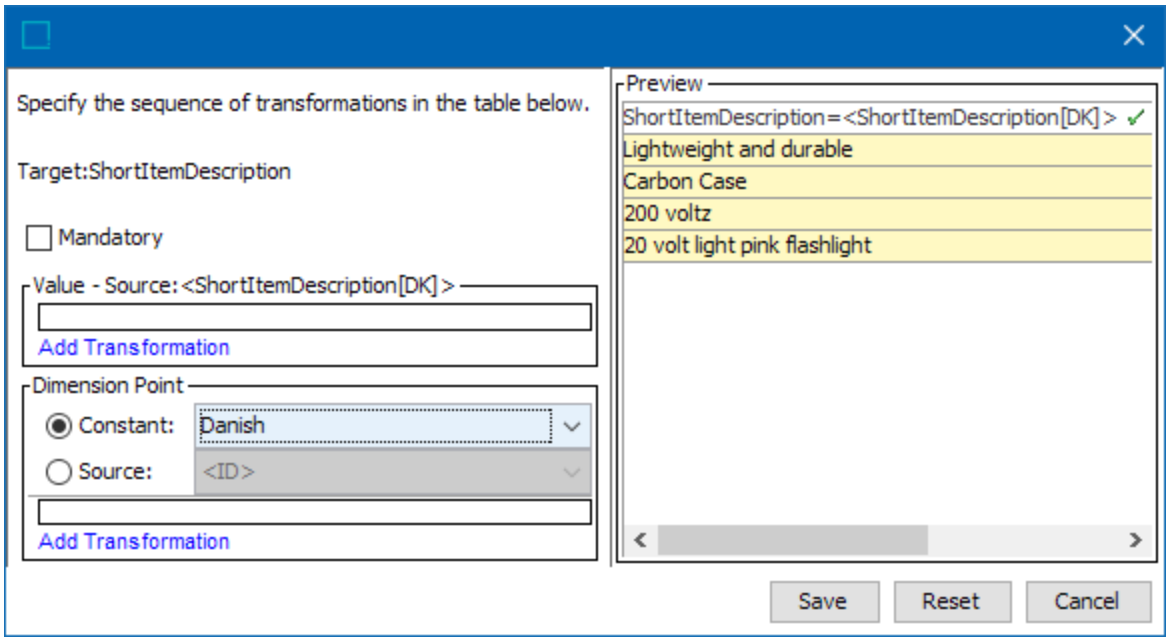
1. Map the object's **ID** in the import tool.



2. Map the column 'ShortItemDescription[DK]' to the desired attribute. In this case it is mapped to 'ShortItemDescription.' Once mapped, select the attribute column in the Results field and click the **Transform** button.



3. In the Transformation window, under the Dimension Point section, select the Constant radio button, and select the desired language from the dropdown menu. In this case **Danish** is chosen. Click **Save**.

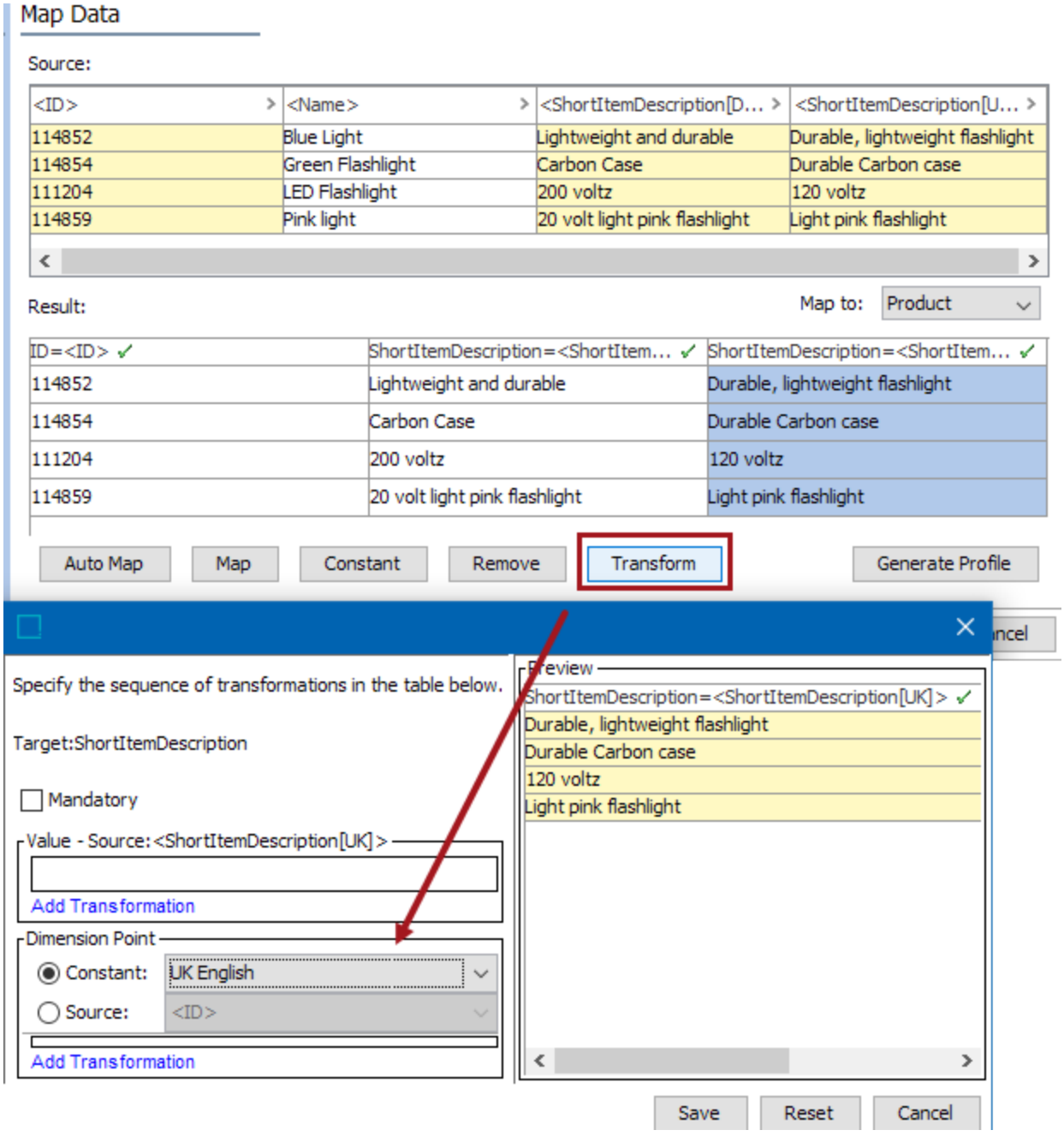


The Dimension Point section is shown because the attribute has a single dimension dependency. The default behavior is to use the language of the import context. The dimension point can either be mapped to:

- **Constant** - when all values of the import are in the same language.
- **Source** - when the language is specified in a separate column. In that case transformations may be needed in order to map the values of the language column correctly to the dimension points.

Note: If the Mandatory option is checked on the Transformation screen, it will skip the objects with empty values, and those objects will not be imported or updated.

4. Select the column 'ShortItemDescription[UK]' and map to the same attribute that was chosen before, 'ShortItemDescription.' Once mapped, select the attribute column in the Results field and click **Transform** > **Constant** > **UK English**. Click **Save** on the Transformations dialog.



5. Complete the remaining steps of the import as usual.

LOV Value-ID or Value Aspect

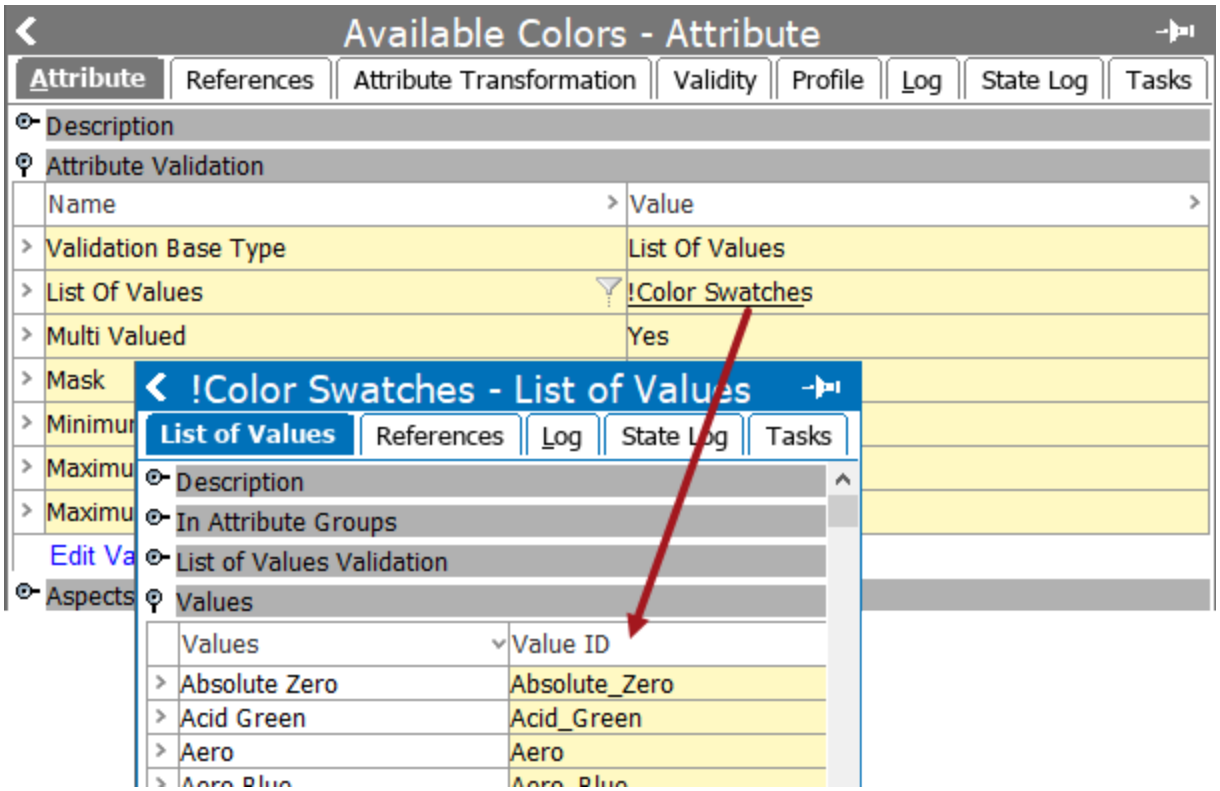
When mapping attributes or attribute groups, this aspect extracts the LOV Value-ID when present, otherwise it extracts the values in the attributes. For details on LOV Value IDs, refer to the **Use IDs on values** section of the **Creating an LOV** topic in the **System Setup** documentation.

The 'LOV Value-ID or Value' aspect allows for values to be exported for attributes and allows for Value IDs to be exported if it is an LOV attribute with a Value ID.

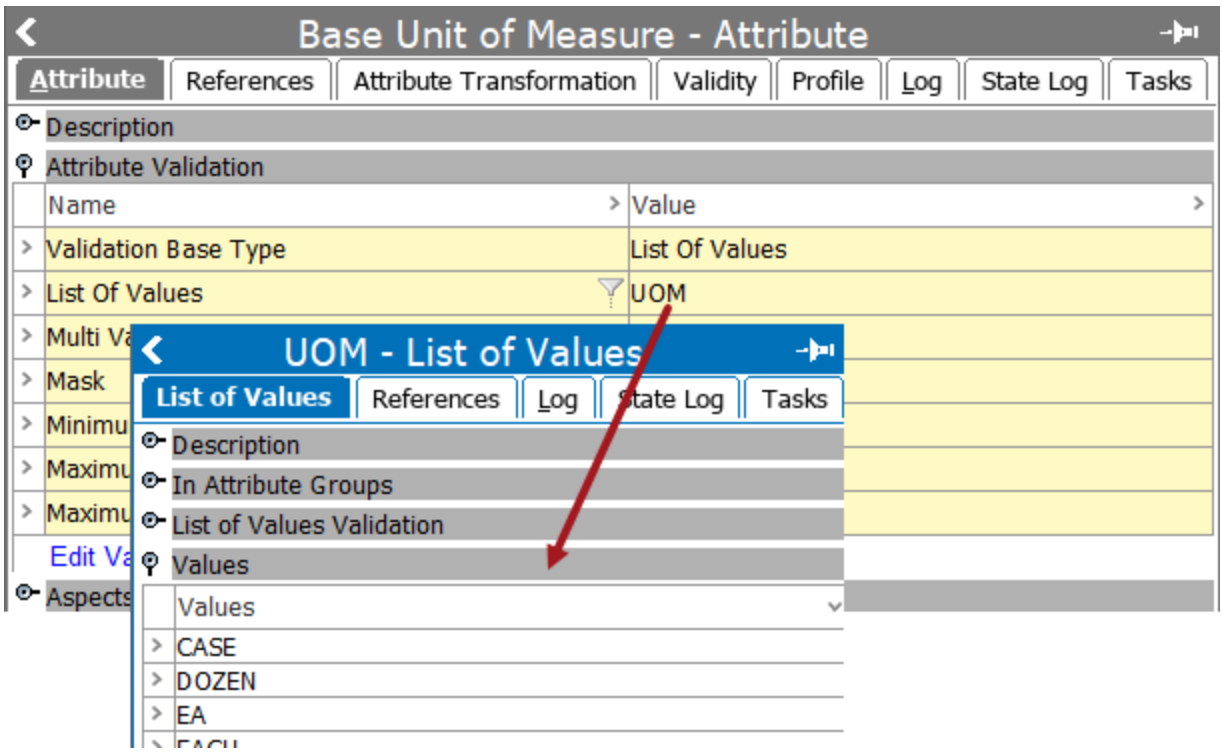
A common use for this aspect is when exporting ETIM via BMEcat2005 format, map an attribute group to the feature field and export the LOV Value IDs. Refer to the **BMEcat 2005 Format Example** section below.

For general information on setting a Transformation aspect, refer to the **Aspect - Transform Outbound** topic.

Consider the 'DataVisualization' attribute group that will be mapped during the Configuration steps below. This group includes two attributes with an LOV Validation Base Type. The 'Available Colors' attribute uses the '!Color Swatches' LOV which uses Value IDs. For this example, for identification of the Value ID compared to the Value, notice that the Value IDs include an underscore instead of a space between words.



The 'Base Unit of Measure' attribute uses the 'UOM' LOV which does not use Value IDs:



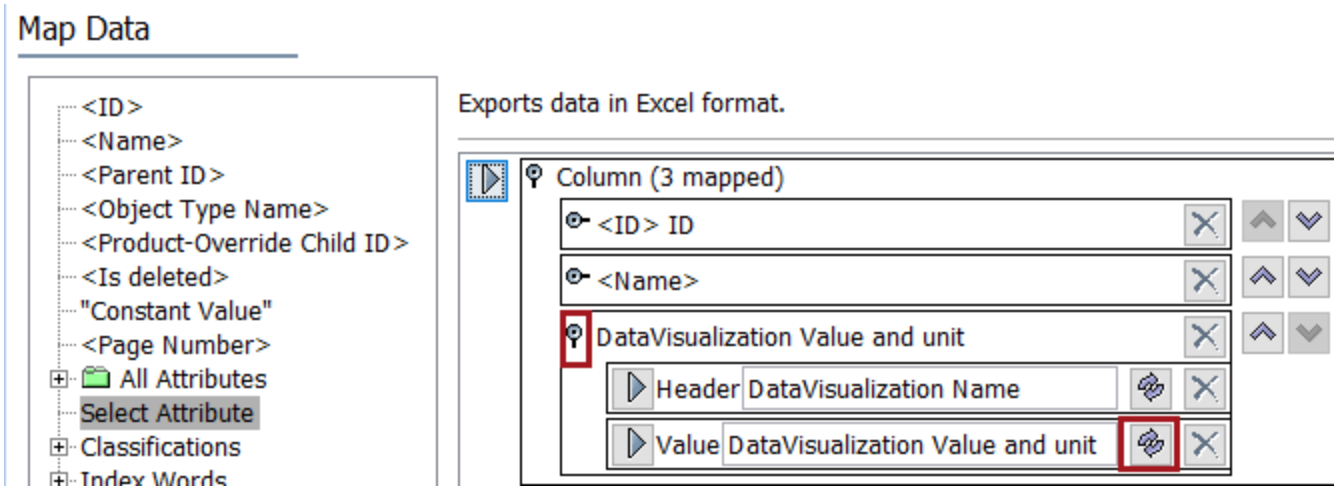
Using the single 'LOV Value-ID or Value' aspect for the attribute group results in exporting the Value IDs for the 'Available Colors' attribute and the Values for the 'Base Unit of Measure' attribute.

Additional options for exporting LOV aspect information are included in the following topics:

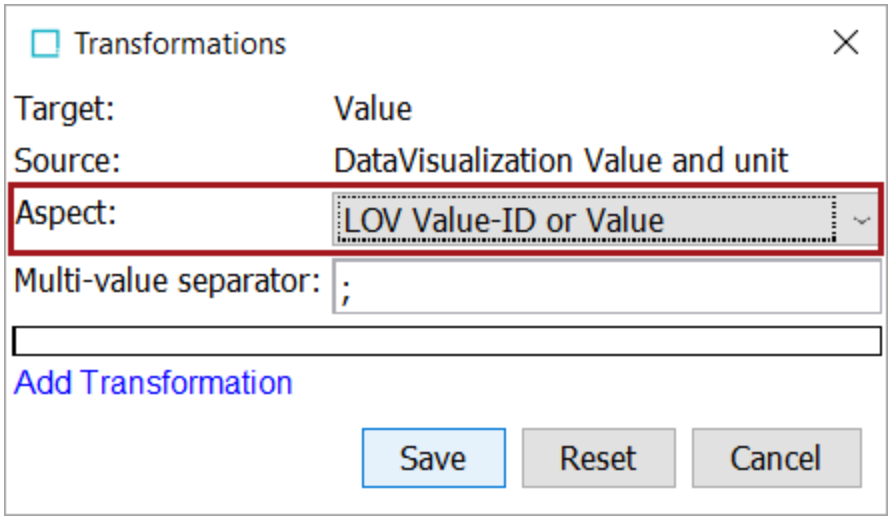
- LOV ID Aspect
- LOV Value-ID Aspect

Configuration

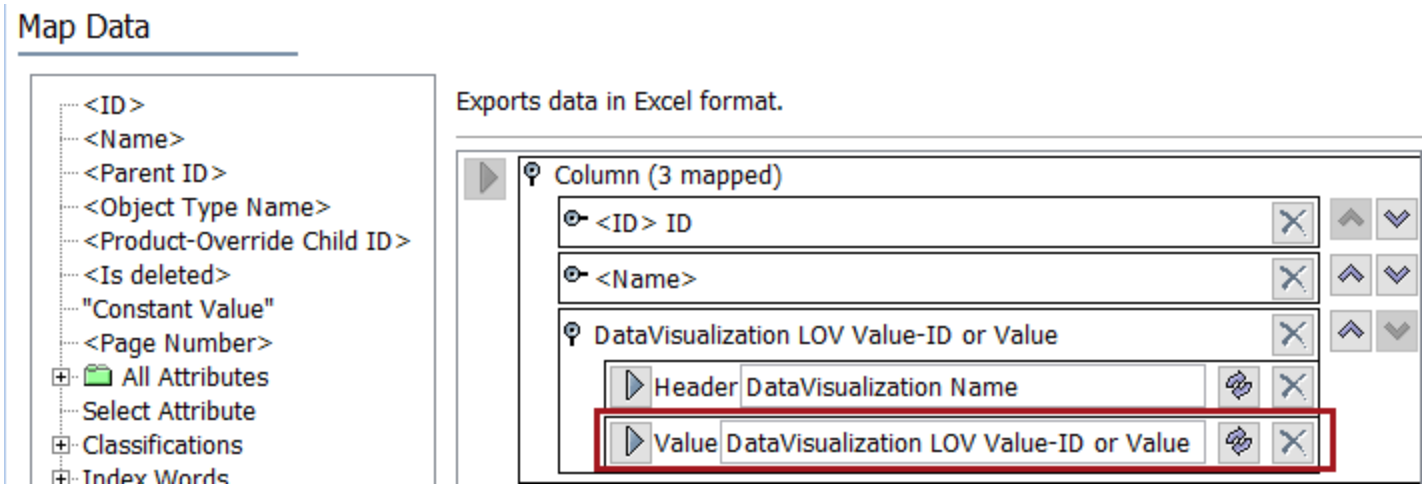
1. In the output tool select the data to be exported. For more information, refer to the **Data Exchange** topic.
2. On the Map Data step, map the required attribute or attribute group using the **Select Attribute** data source. For more information, refer to **Mapping Attributes via Select Attribute** section of the **Attributes (and Data Containers) - Data Source Outbound** topic.
3. In the right panel, click the flipper for the attribute or group that should be extracted using LOV Value ID or Value. In the example below, the 'DataVisualization' group has been mapped.
4. On the Value row, click the **Transformation** button to display the Transformations dialog.



5. In the **Transformations** dialog, from the **Aspect** list, select the **LOV Value-ID or Value** and click the **Save** button.



The Value element is displayed with the DataVisualization attribute using the 'LOV Value-ID or Value' aspect.



6. Complete any additional mappings and initiate the export.

Results

The product ID and name are exported, along with the Value IDs for the LOV attribute that uses them, and Values for the LOV attribute that does not use them.

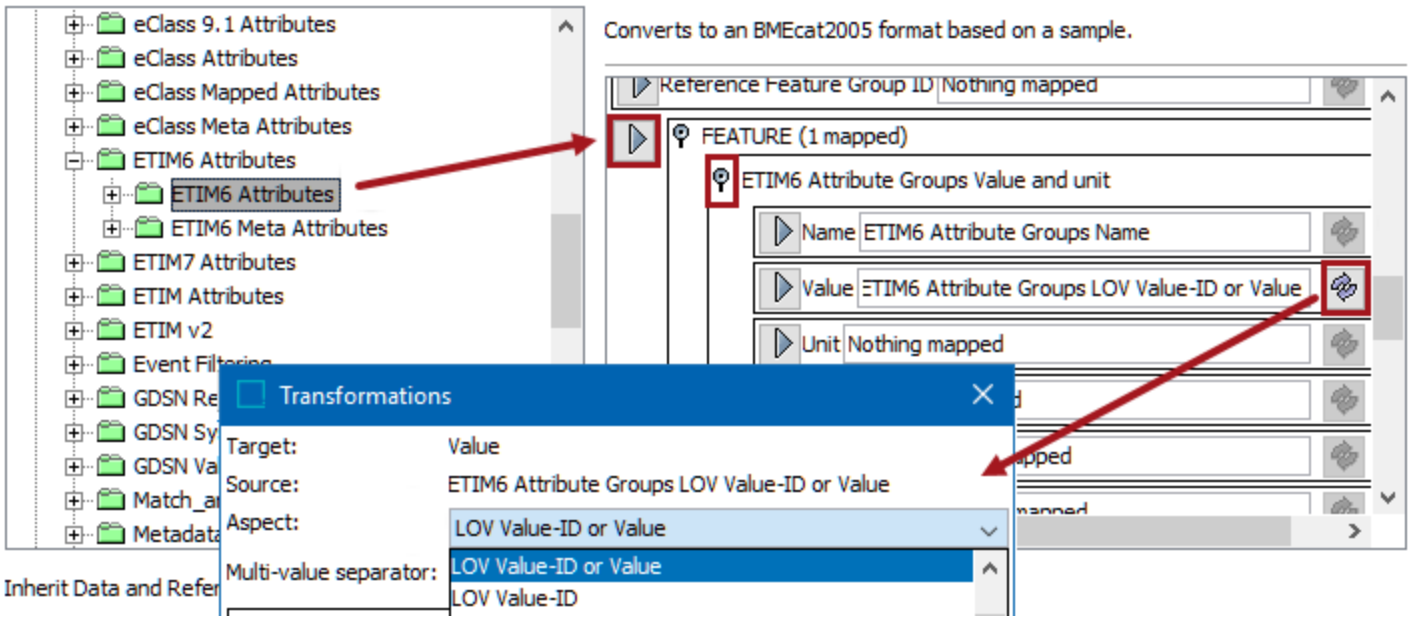
	A	B	C	D
1	<ID>	<Name>	Available Colors	Base Unit of Measure
2	18212	18212 L B	Acid_Green	CASE
3	18213	18213 M O	Aero	EA
4	18216	18216 L O	Absolute_Zero	DOZEN

Note: If the LOV attribute allows multiple values, the LOV Value IDs are exported with the multi-value separator defined in the Transformations dialog. For more information, refer to the **Multi-Value Separator - Transform Outbound** topic.

BMEcat 2005 Format Example

The following image uses the BMEcat 2005 format. The 'ETIM6 Attributes' attribute group is mapped to the BMEcat2005 FEATURE element, which then creates the 'ETIM6 Attribute Groups' flipper with its contents. The **LOV Value-ID or Value** aspect exports values for attributes without an LOV Value ID and exports the LOV Value ID for attributes with an LOV Value ID.

Map Data


eClass 9.1 Attributes
eClass Attributes
eClass Mapped Attributes
eClass Meta Attributes
ETIM6 Attributes
ETIM6 Attributes
ETIM6 Meta Attributes
ETIM7 Attributes
ETIM Attributes
ETIM v2
Converts to an BMEcat2005 format based on a sample.
Reference Feature Group ID
Nothing mapped
FEATURE (1 mapped)
ETIM6 Attribute Groups Value and unit
Name
ETIM6 Attribute Groups Name
Value
ETIM6 Attribute Groups LOV Value-ID or Value
Unit
Nothing mapped
Transformations
Target:
Value
Source:
ETIM6 Attribute Groups LOV Value-ID or Value
Aspect:
LOV Value-ID or Value
Multi-value separator:
LOV Value-ID or Value
LOV Value-ID

Extract Characters at Position Example

Transformations can improve the quality of the data you want to import. For general information about transformations, refer to the **Inbound Map Data - Transform** section.

This example describes how to extract and clean information from a legacy system, or groups of information stored in one column, and load it into separate attributes in STEP. If information is more technical or cryptic, refer to the last section of this topic on how to convert cryptic data.

This example includes the following transformations:

- Starting from position
- Replace whole value
- Change case
- Extract characters at position

Data being transformed

Below we use transformations for each of the columns that require data to be clarified.

Consider data in an input file that is cryptic, meaning that it is represented in a different than the way STEP organizes the data. For example, a user needs to extract the color from the material. In this case bl = blue, gr = green and br = brown. This data needs to be transformed into 'real' attribute values that STEP can recognize.

	A	B
1	<ID>	(Data)
2	107601	blcotton 2
3	20803	grwool 4
4	20805	brwool 3
5	109011	grcotton 2

For more on all available transformations, refer to the **Transformations** topic in the **Resource Materials** online help.

Applying transformations

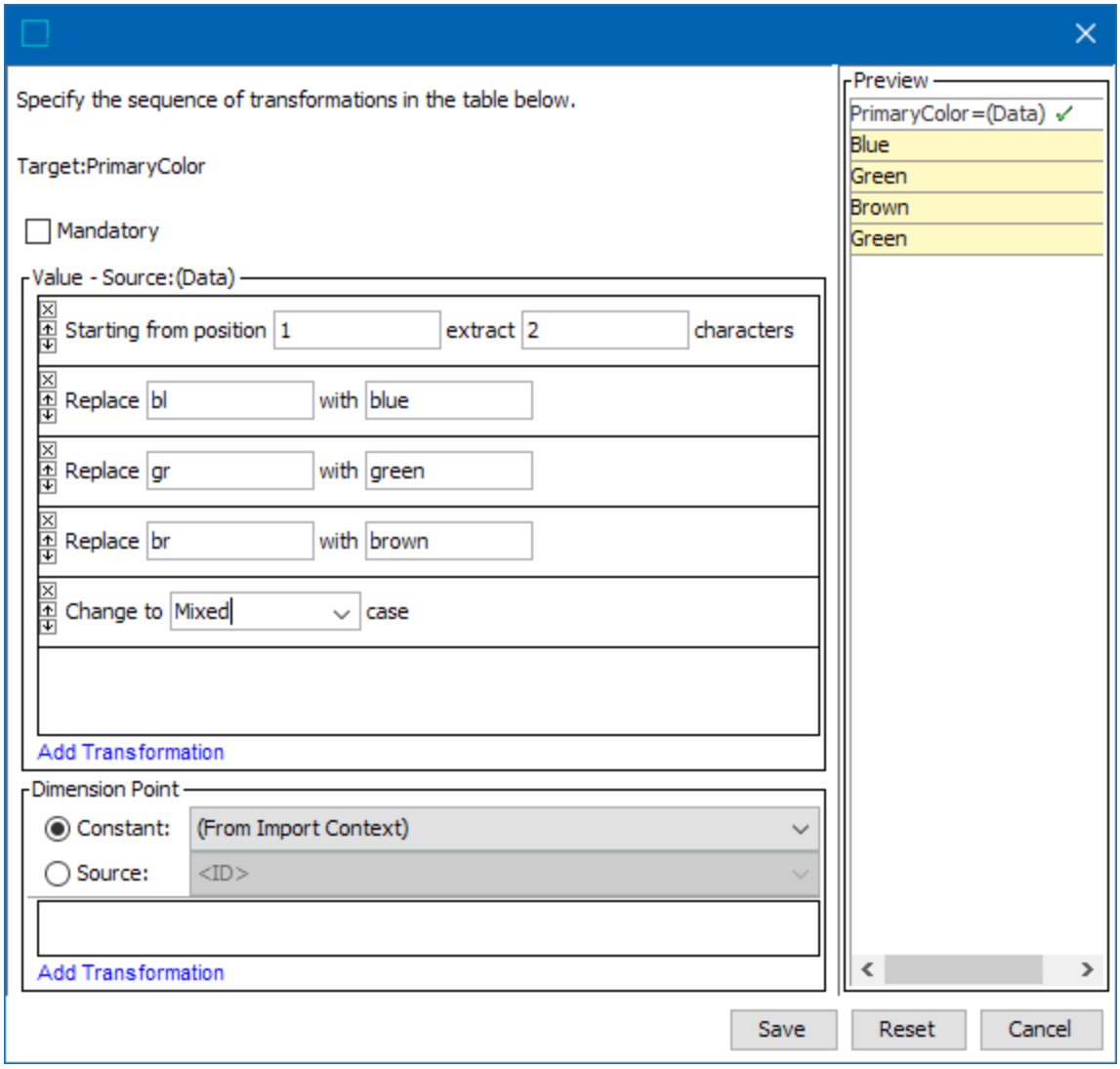
First map the inbound data columns and then apply the needed transformations.

1. Map the '<ID>' column to the ID in STEP. For details, refer to the **ID or Key - Map Inbound** topic.
2. Map the '(Data)' column to the appropriate attributes. In this case, since the column holds both the color and the material it is mapped to the 'Primary Color' attribute and the 'Material' attribute.

   A warning message is displayed saying that the source column has already been mapped. Click 'Yes' to continue Mapping. For details, refer to the **Mapping the Same Column Multiple Times** section of the **Inbound Map Data - Map** topic.

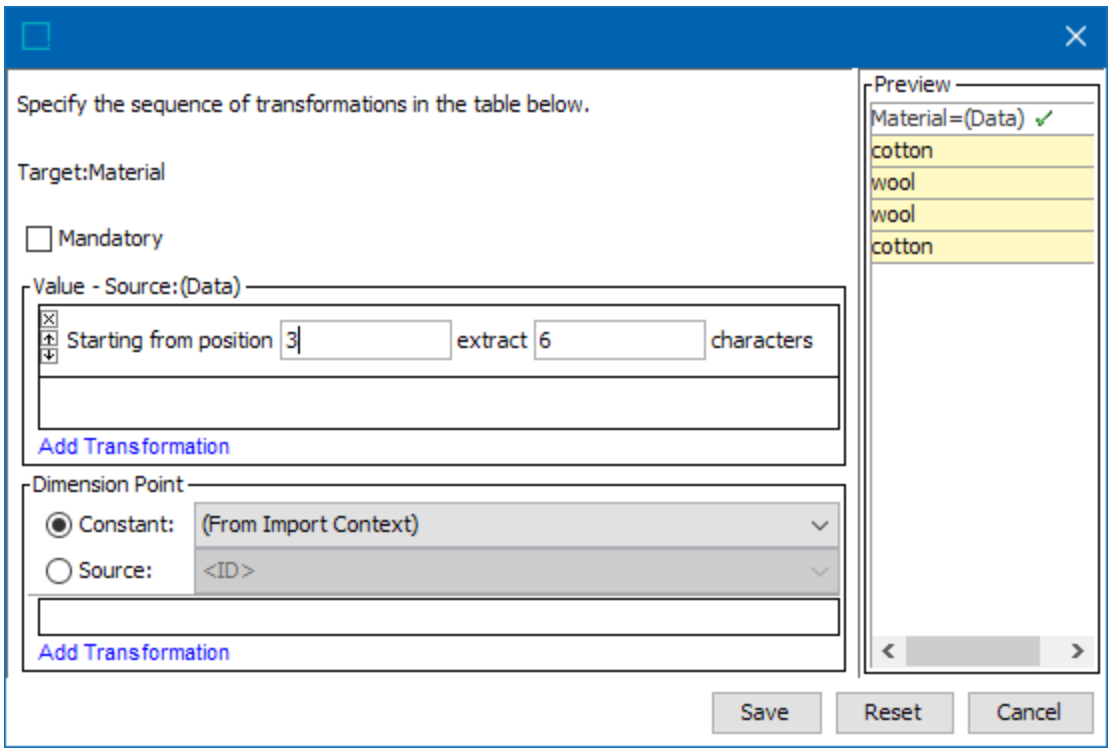
3. In the Results area, select the column mapped to the 'Primary Color' attribute, and click **Transform**.
4. Add required transformations:

- The 'Starting from position' transformation is used so that the exact letters that represent the color can be separated from the material.
- The 'Replace whole value' is used to update the two letter coding data to match the correct color data in STEP.
- The 'Change case' is used to apply a capital letter to color names, with lower case letters that follow.
- Click **Save** to close the Transformations dialog.

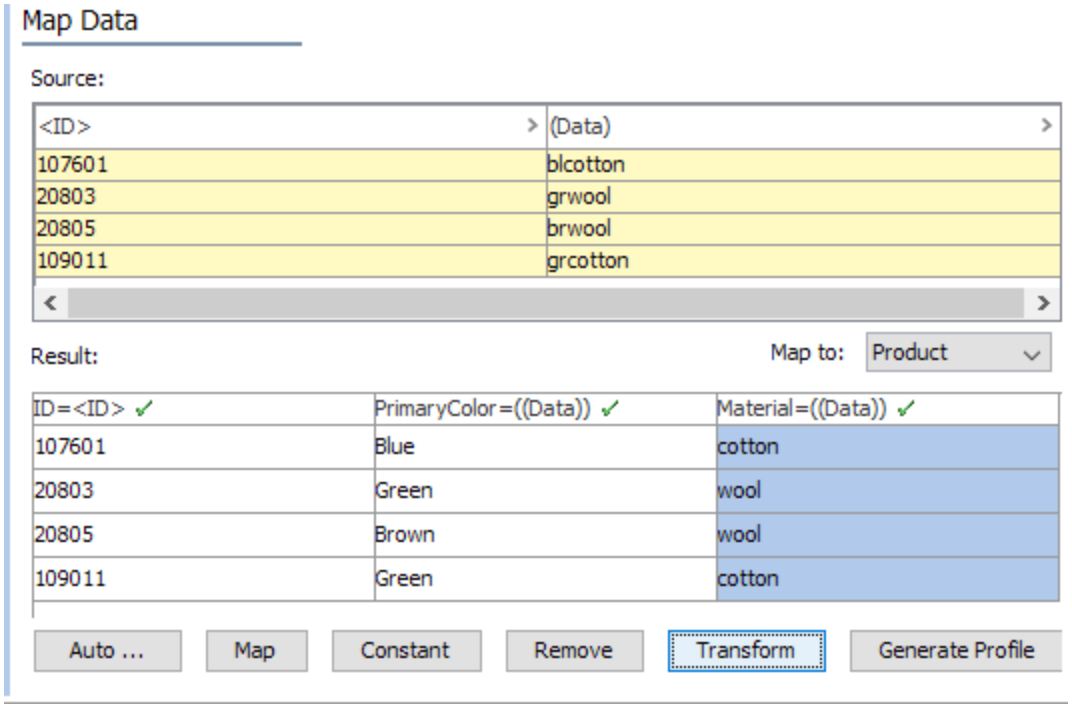


Note: Transformations work in sequence from top to bottom.

5. In the Results area, select the column mapped to the 'Material' attribute, and click **Transform**.
6. Add the 'Extract Characters at position' transformation to indicate that the material information starts with character 3, and that the longest material is 6 letters long.



7. Once each column is properly transformed, the results field should now look accurate, and it is ready to go on to the next steps for import.



Extract Characters Between and Extract Characters After Example

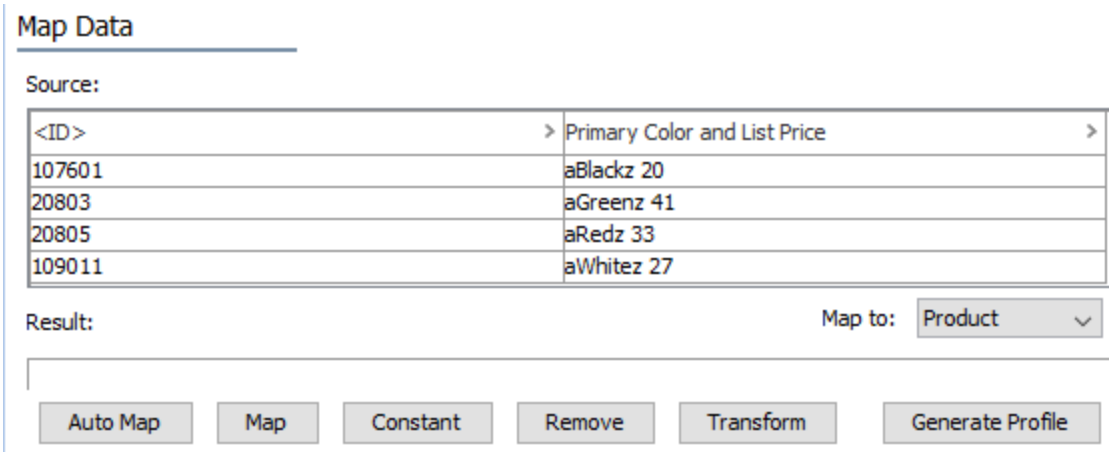
Transformations can improve the quality of the data you want to import. For general information about transformations, refer to the **Inbound Map Data - Transform** section.

Transformations can be used to separate attribute values from a single field of data, and create values for multiple attributes in STEP.

This example includes the following transformations:

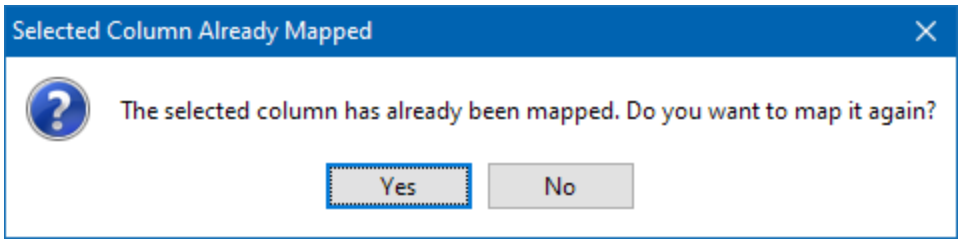
- Extract characters between two pieces of text in the value
- Extract characters after a given text in the value

This example describes how to extract and clean information from one field of an import file, and prepare it to be loaded into separate attributes in STEP. The following data is being imported, and contains data for multiple STEP attributes within a single field 'Primary Color and List Price':



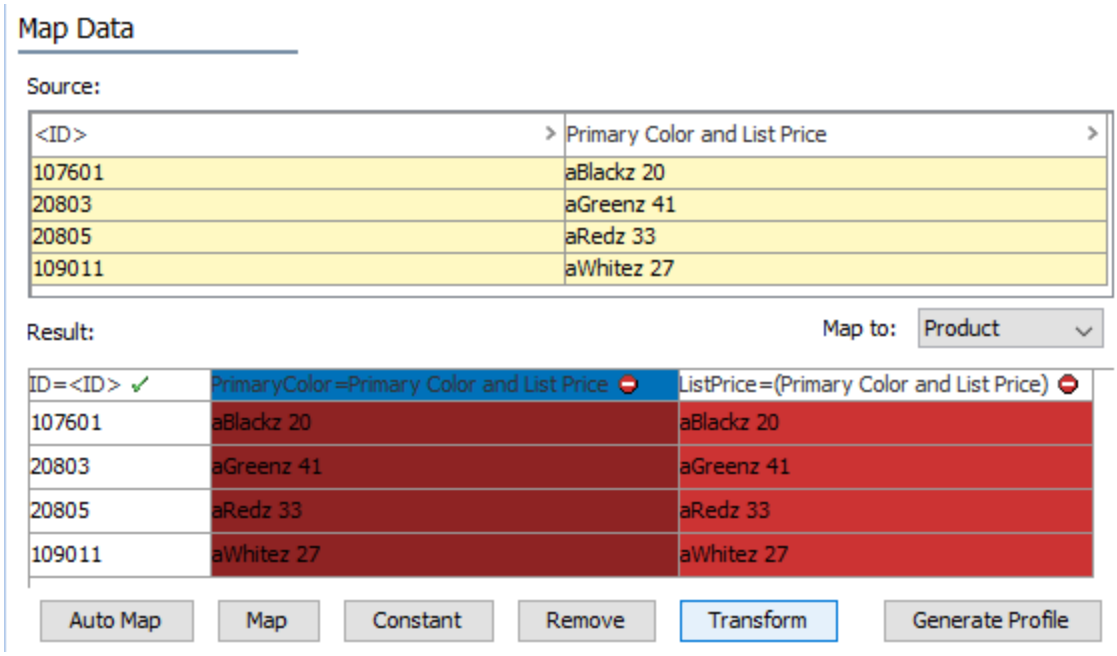
Configuration

1. Load the import file into the inbound tool. For more information, refer to the **Data Exchange** topic.
2. Map the object ID in the import tool. Refer to **ID or Key - Map Inbound** for details.
3. Map the source column that holds multiple values to the required attributes. When mapping the same source column more than once, you must click Yes to confirm that you want to map it again.

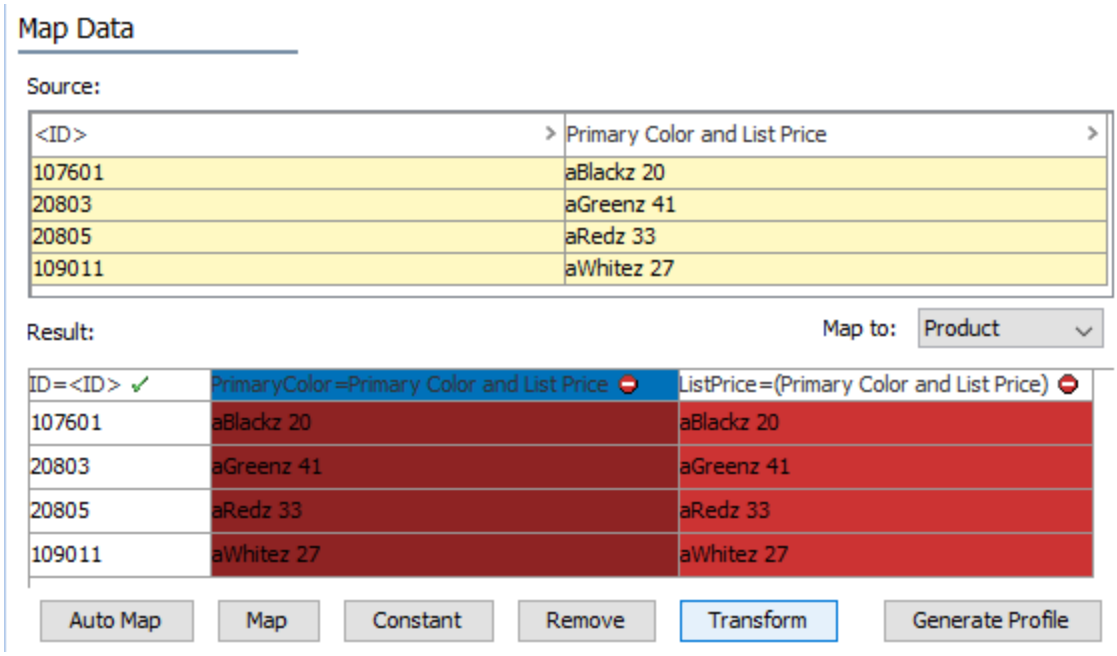


In this example, we map 'Primary Color and List Price' two times, once to the Primary Color attribute, and again to the List Price attribute. In our example, the Result section shows the newly mapped items in red because the current values are not valid.

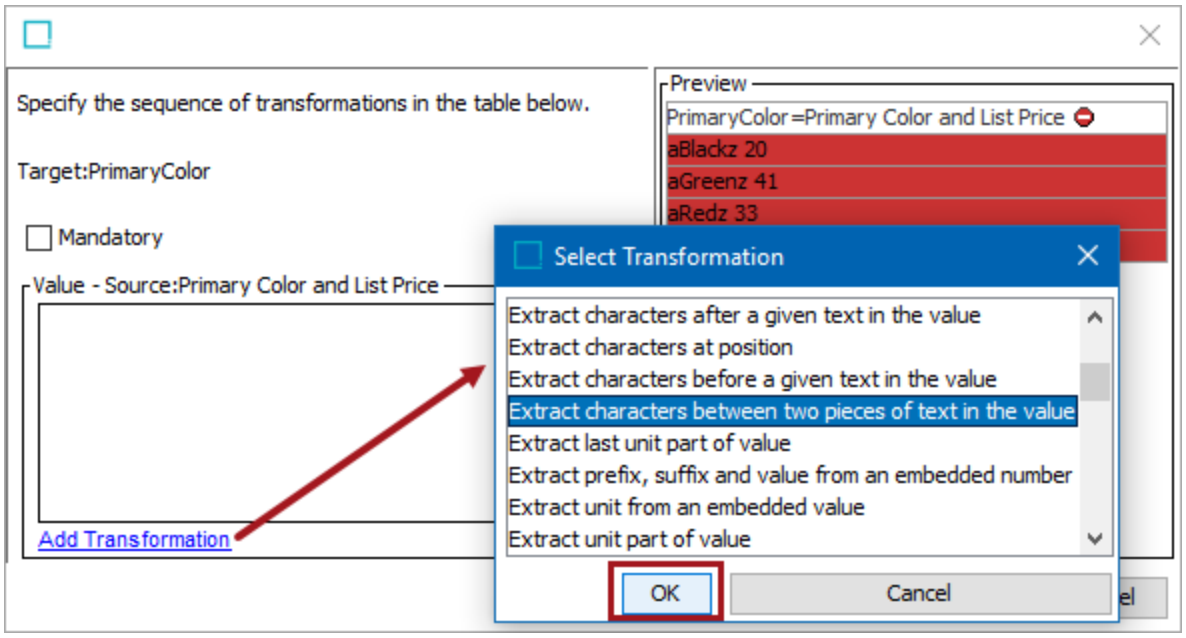
Note: Hover over a red cell to display the error message.



4. Select the one of the result columns mapped to an attribute and click the **Transform** button.



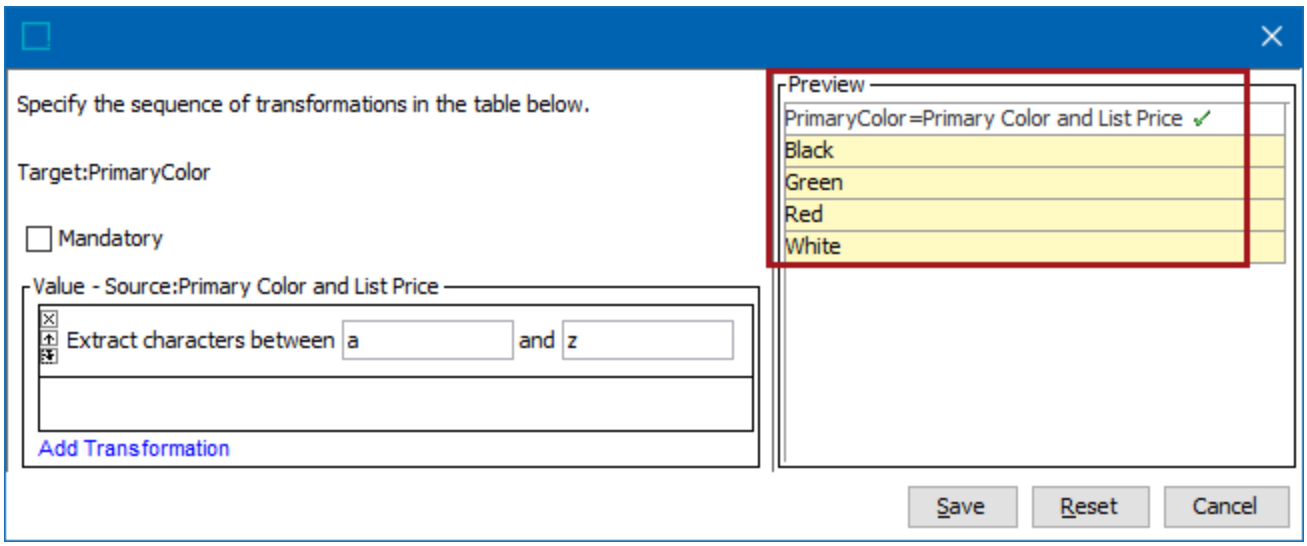
5. In the Transformation window, click on the **Add Transformation** link, select the **Extract characters between two pieces of text in the value** transformation, and click **OK**.



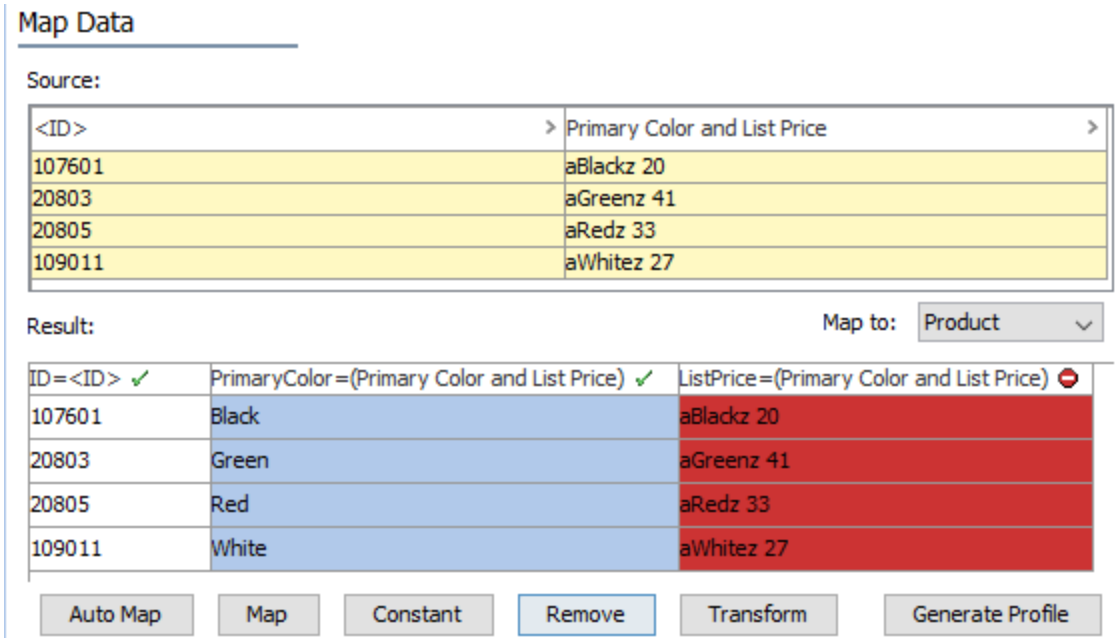
6. Enter the characters that should be used to determine that data that needs to be extracted and click out of the text fields. In this example, the color is always between the letters 'a' and 'z.' Click or tab out of the parameter to complete the configuration. The Preview pane shows the results of the transformation.

   Click or tab out of the parameter to complete the configuration. The Preview pane shows the results of the transformation. If any additional transformations are needed for this data, click the **Add Transformation** to add them.

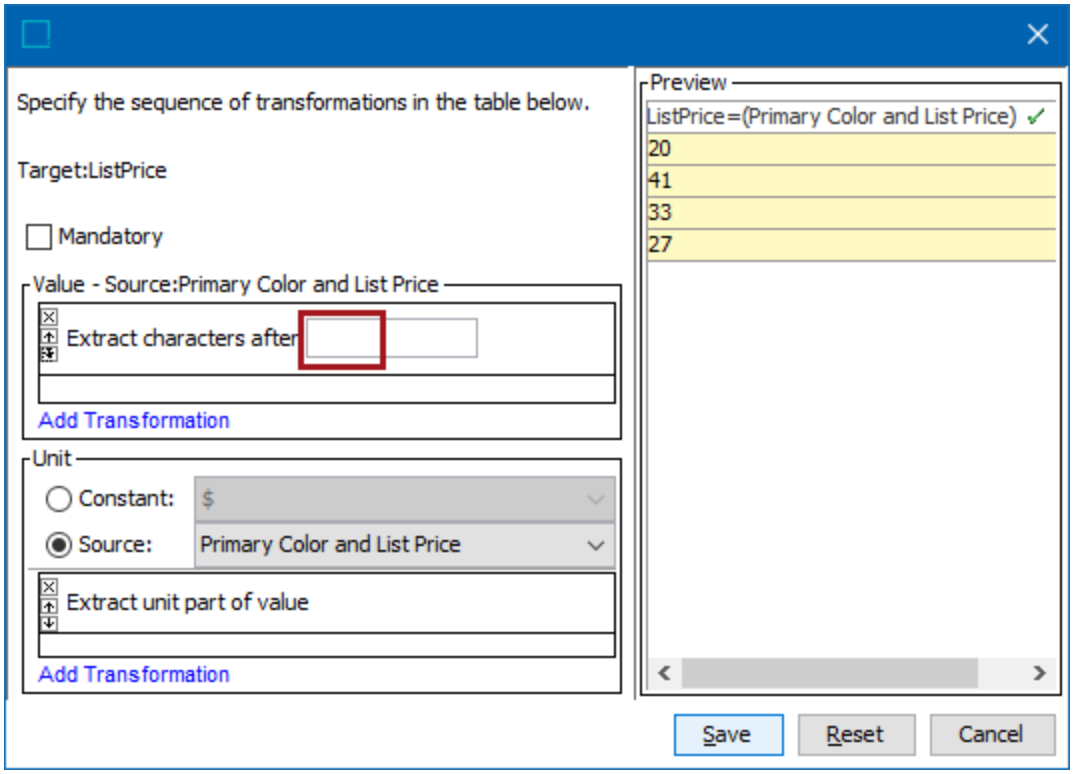
Important: Transformations are evaluated from the top down. Carefully consider the order of the transformations applied. To remove or rearrange transformations, use the Delete, Up arrow, and Down arrow buttons .



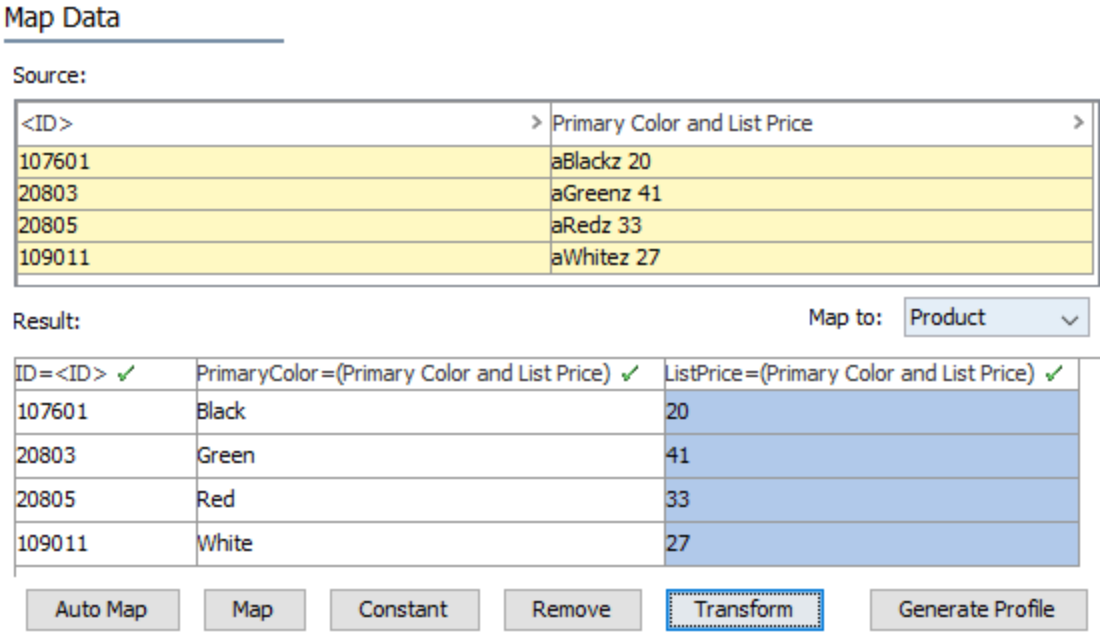
7. If necessary, check the 'Mandatory' option to skip objects with empty values during import.
8. Click **Save** to display the transformed data in the Map Data step.



9. Repeat step 3 for the remaining mapped column in the result area.
10. Use the **Extract characters after a given text in the value** transformation in order to remove all text except the price. Although not visible in the following image, a 'space' character was added for the 'Extract characters after' parameter, and results in extracting only the price as shown below.



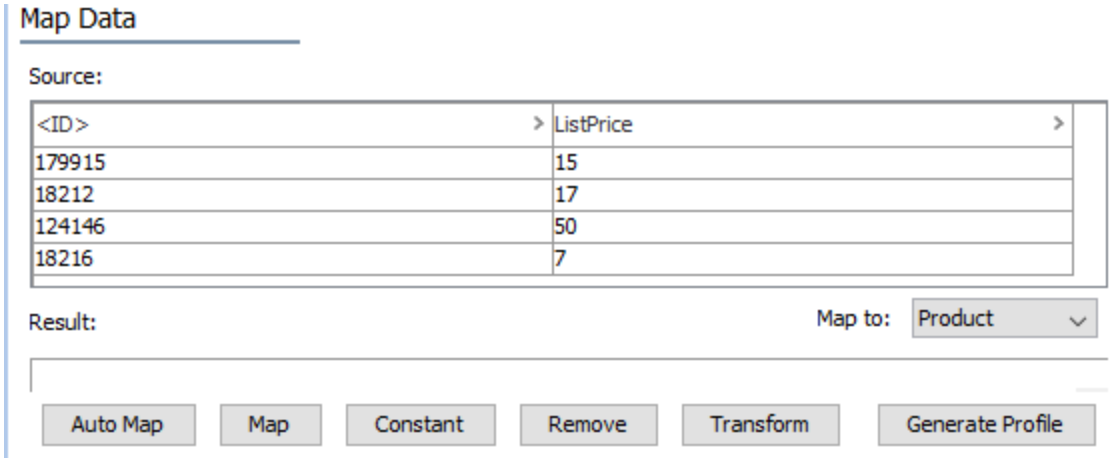
11. Click **Save** to display all transformed data in the Map Data step.



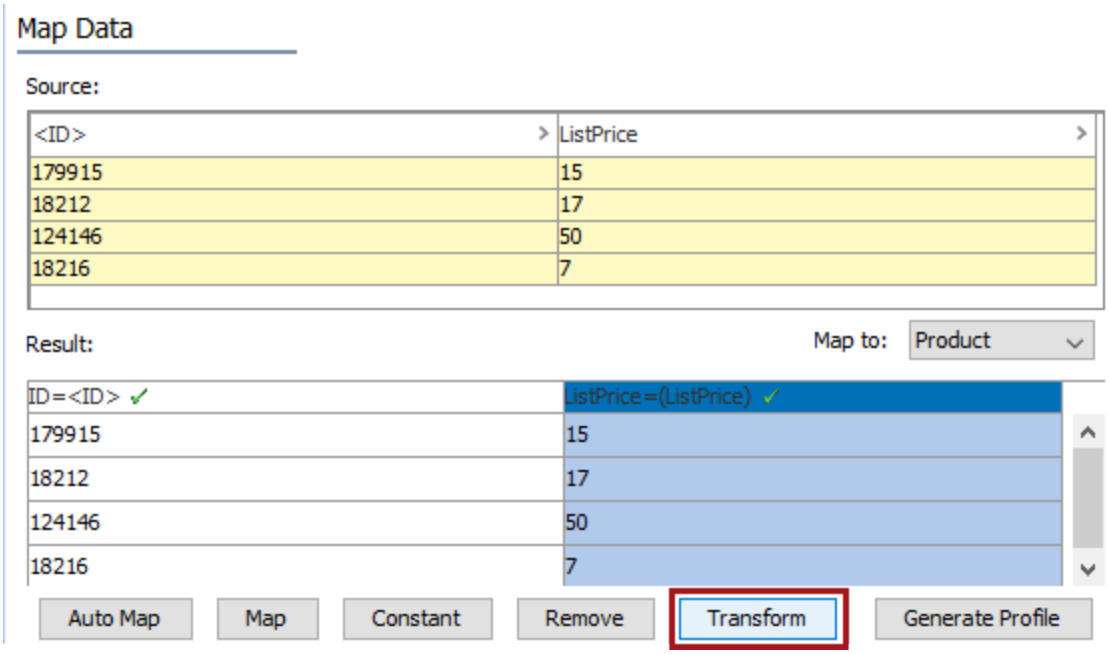
Format Number Example

Transformations can improve the quality of the data you want to import. For general information about transformations, refer to the **Inbound Map Data - Transform** section.

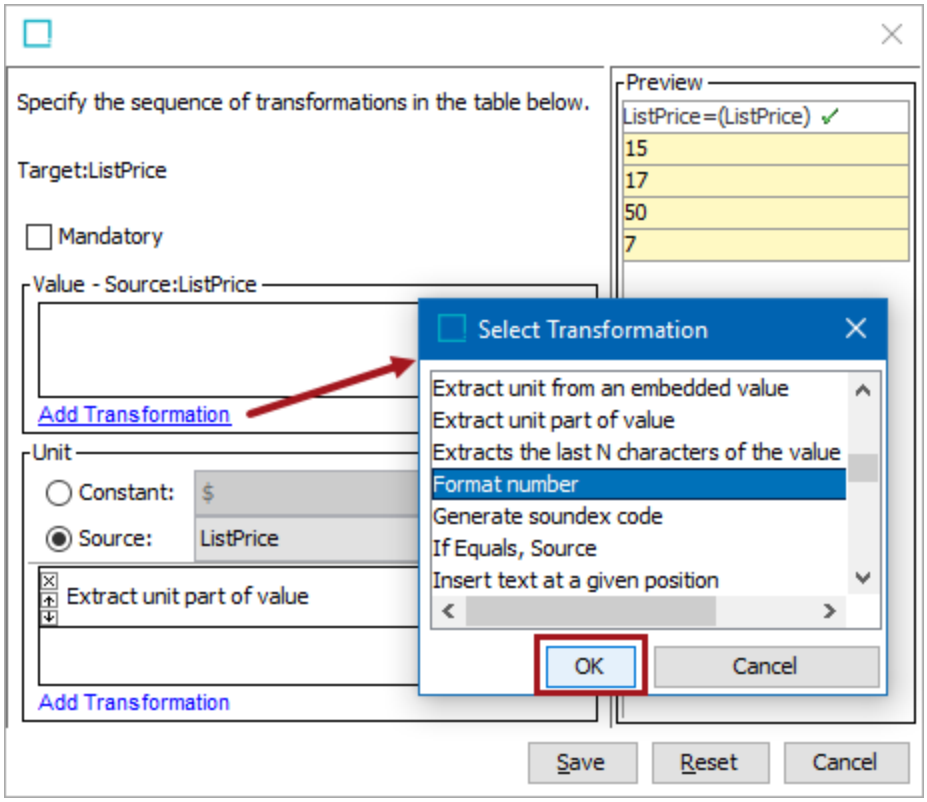
Transformations can be used to format numbers, for example, to set a specified number of decimals, as required by an attribute. For example, the following data is being imported, and contains no decimal places:



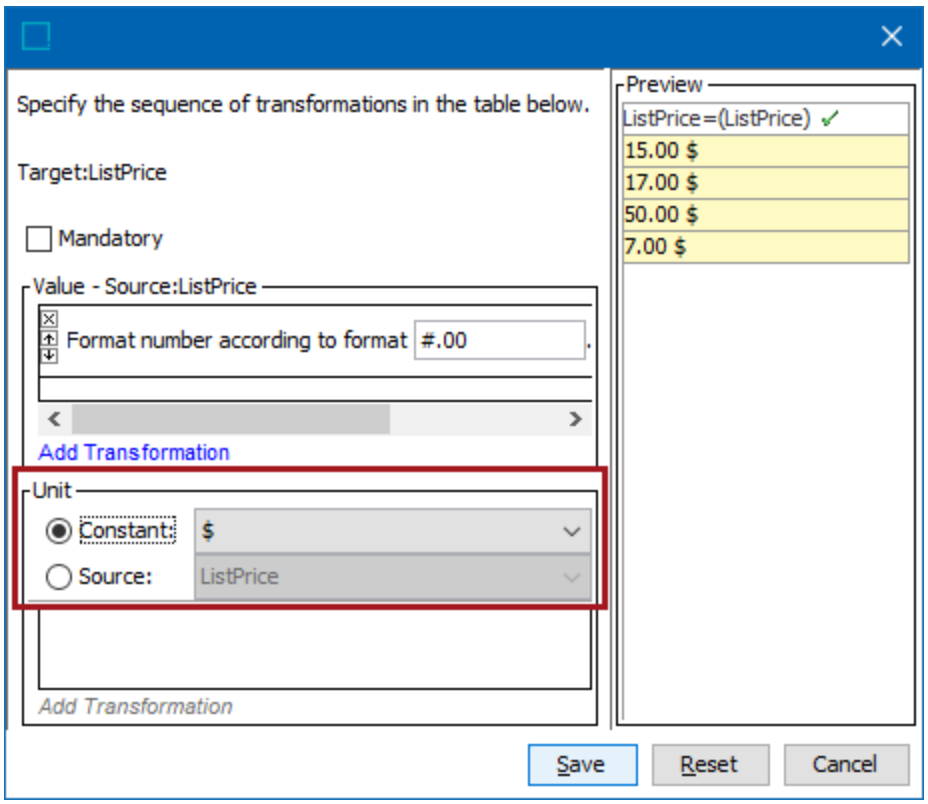
1. Map the object ID in the import tool.
2. Map the source column that holds the data to the required attribute. In this example, we map 'ListPrice' to the attribute of the same name.
3. Select the mapped column with the price data and click the **Transform** button.



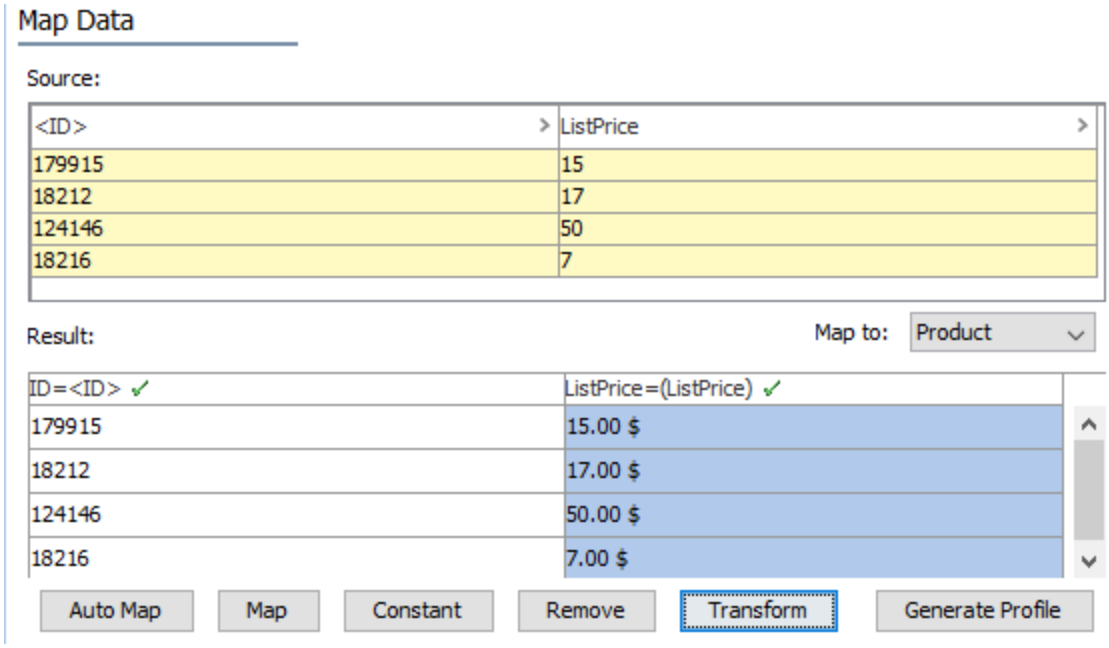
4. Click the **Add Transformation** link, and choose the 'Format number' transformation. Click **OK**.



5. For the Unit section, select the **Constant** radio button, and use the dropdown to add the correct currency symbol to the value.



6. Click **Save** to display the results in the Map Data step.



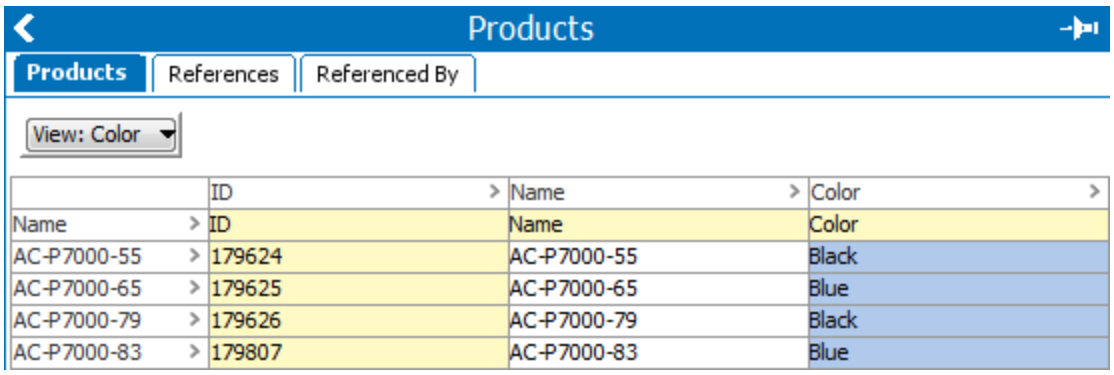
Header and Value Aspects Example

This example shows a variety of mappings and exports using a single attribute, named 'Color.' Although the format illustrated is Excel, using other formats would produce the same results. For more information on each outbound mapping option, refer to the **Outbound Map Data - Data Source** documentation.

For more information on the available aspects, refer to the **Aspect - Transform Outbound** documentation.

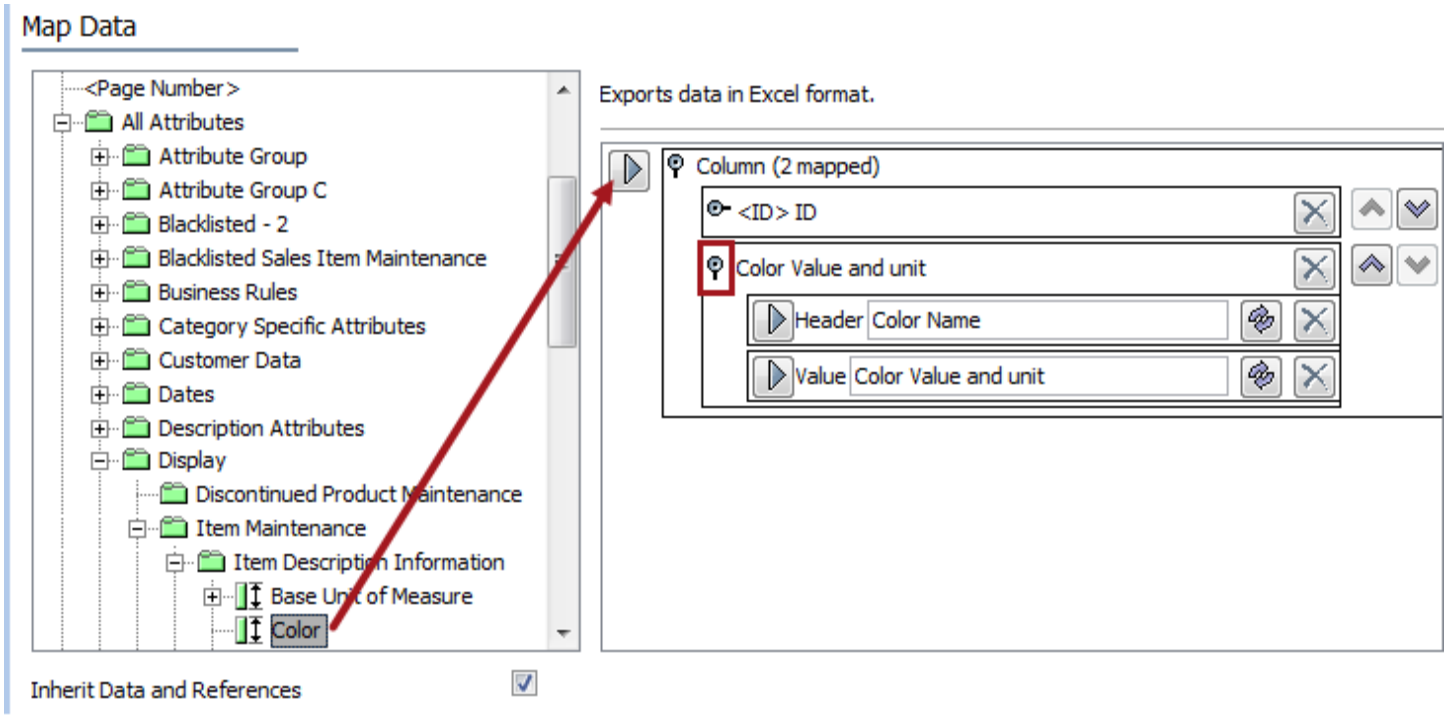
Default Header Aspect and Default Value Aspect

The following products have values for the Color attribute and are displayed using the multi object editor:



	ID	Name	Color
Name	ID	Name	Color
AC-P7000-55	179624	AC-P7000-55	Black
AC-P7000-65	179625	AC-P7000-65	Blue
AC-P7000-79	179626	AC-P7000-79	Black
AC-P7000-83	179807	AC-P7000-83	Blue

The ID and the Color attribute are mapped using the default settings. When the flipper for the mapped column is opened (), separate header and value parts are displayed. When mapping an attribute, by default, the attribute name is mapped to the header part and the value of the attribute is mapped to the value part.



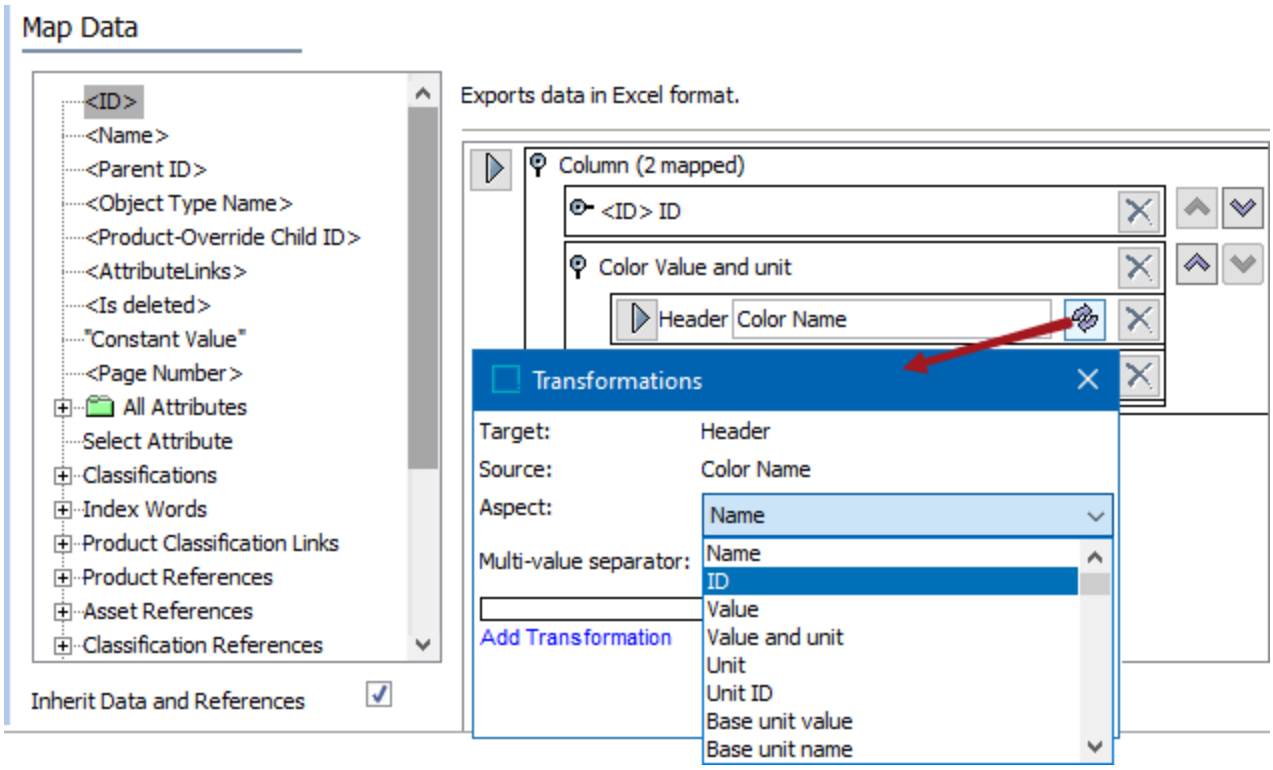
Results

Exporting with this mapping produces the following result file:

	A	B
1	<ID>	Color
2	179624	Black
3	179625	Blue
4	179626	Black
5	179807	Blue

ID Header Aspect and Default Value Aspect

The data exported as header and value is determined by the aspect setting. You can include the attribute ID in the header using the transform button ( ) and selecting the ID aspect for the header column.



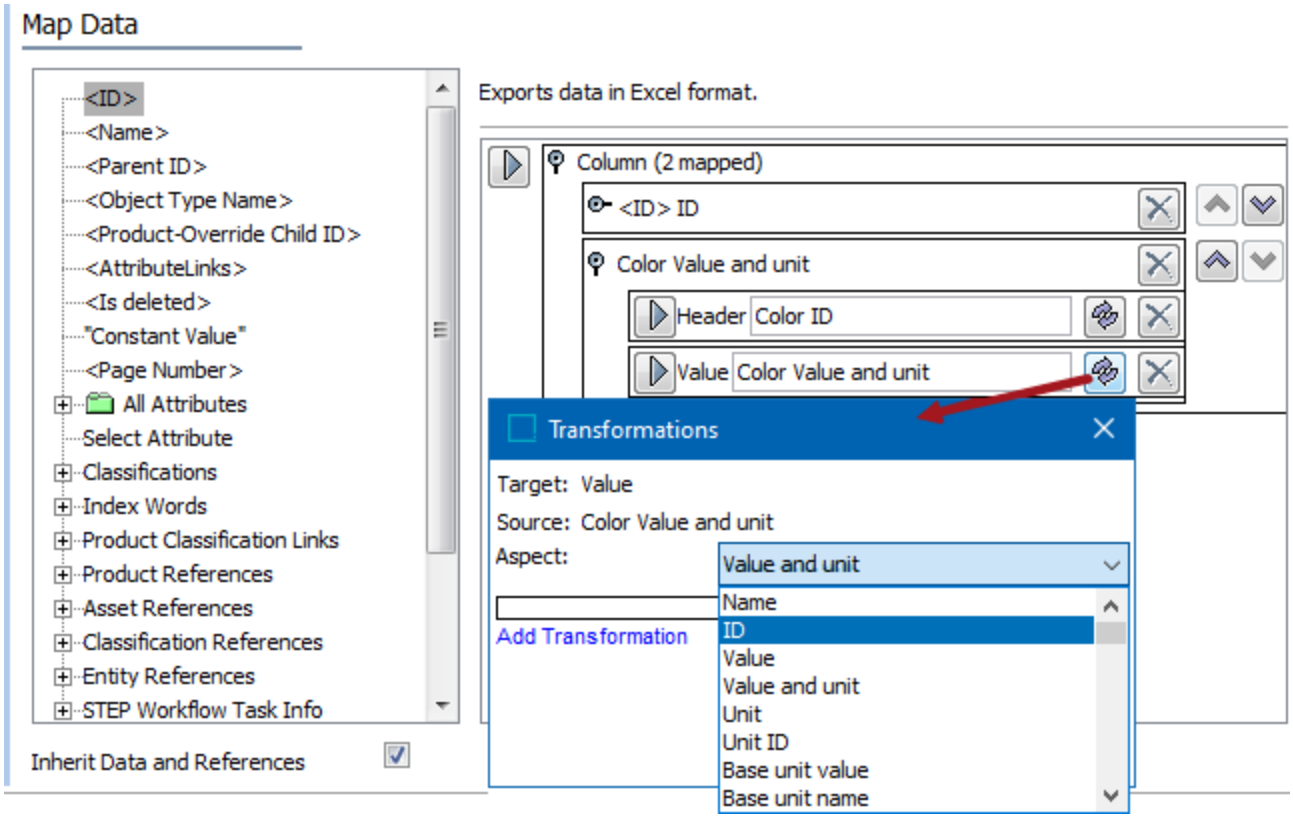
Results

Exporting with the transformation produces the following result file:

	A	B
1	<ID>	ColorID
2	179624	Black
3	179625	Blue
4	179626	Black
5	179807	Blue

ID Header Aspect and ID Value Aspect

You could also generate a file output with an additional data column like product ID, attribute ID, and the attribute value by adding the ID aspect for the value part of the mapping.



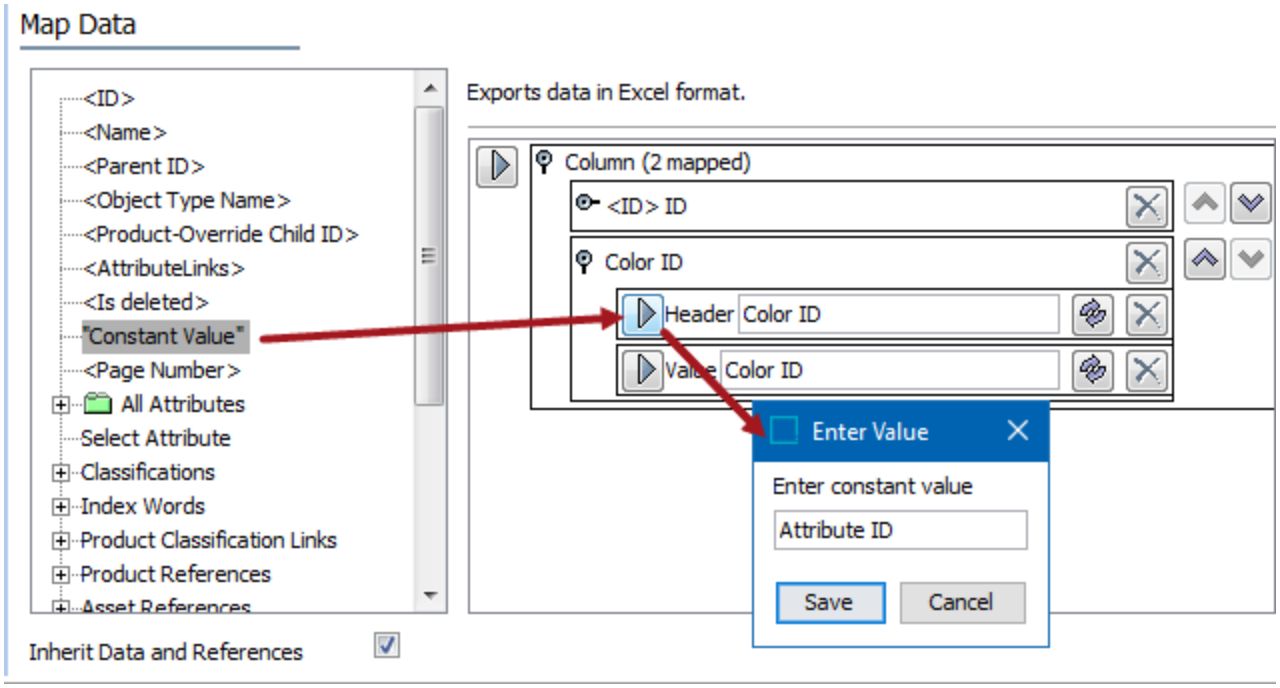
Results

Exporting with this mapping results in the following file, which includes the attribute ID in both the header and the value for each product:

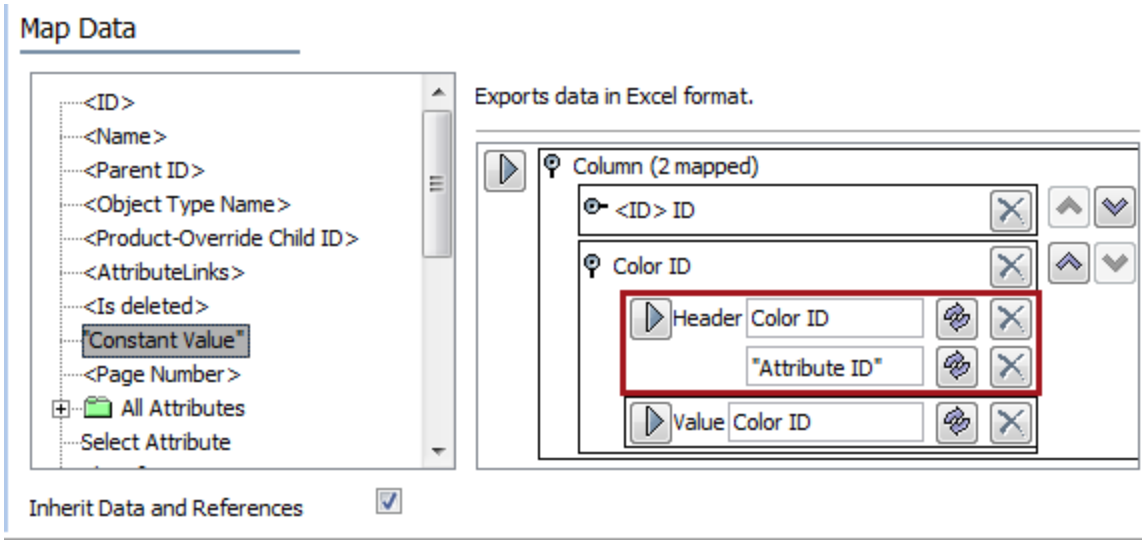
	A	B
1	<ID>	ColorID
2	179624	ColorID
3	179625	ColorID
4	179626	ColorID
5	179807	ColorID

Constant Value, ID Header Aspect, and ID Value Aspect

Now you can map a Constant Value to the header part by selecting "Constant Value" data source on the left and map it to the header as shown below.



After saving the constant value, there are two elements for the header.



For more information on using the Constant Value data source, refer to the **Constant Value - Data Source Outbound** documentation.

Results

Exporting with this mapping results in a file where the header is a concatenation of the attribute ID (ColorID) and the constant value 'Attribute ID.'

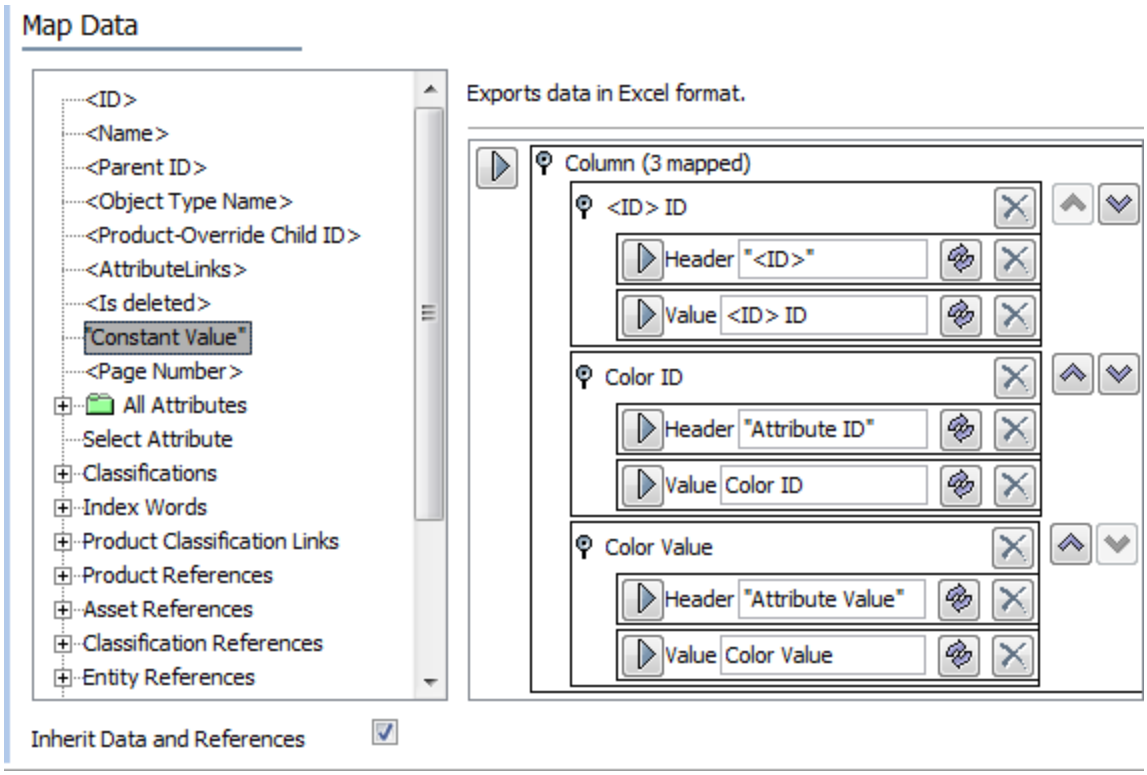
	A	B
1	<ID>	ColorIDAttribute ID
2	179624	ColorID
3	179625	ColorID
4	179626	ColorID
5	179807	ColorID

Constant Value and Value Aspect

Click the delete ( ) button for the Header > Color ID element to remove the actual attribute ID from the header.

Add the third column by mapping the Color attribute again, and replace the header part with an 'Attribute Value' constant.

The full mapping now includes three columns, that contain information for the product ID, the ID for the attribute named Color, and the value for the attribute named Color.



Results

Exporting with this mapping results in the following file:

	A	B	C
1	<ID>	Attribute ID	Attribute Value
2	179624	ColorID	Black
3	179625	ColorID	Blue
4	179626	ColorID	Black
5	179807	ColorID	Blue

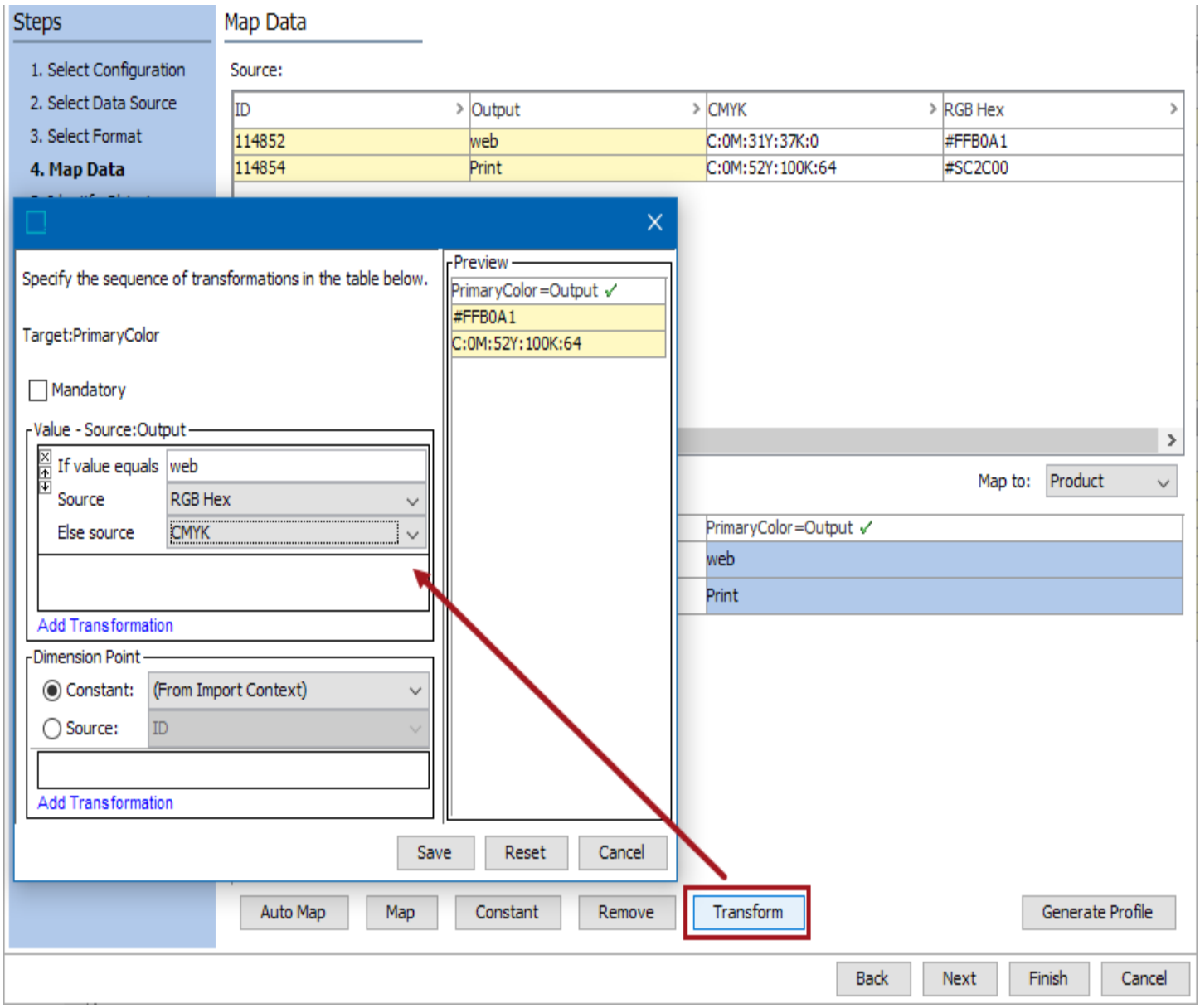
If Equals Source Example

Transformations can improve the quality of the data you want to import. For general information about transformations, refer to the **Inbound Map Data - Transform** section.

Using the **If Equals, Source** transformation, you can make specific columns in a row or column based import file dependent on the content of a mapped column.

For example, a column in your import file specifies whether a product is for print or web publication, and another column contains color information in RGB Hex and CMYK. You now want to populate an attribute with CMYK information if the product is for print and RGB if it is for web.

First, map the column that contains the web and print information to the PrimaryColor attribute in STEP. Next, apply the If Equals, Source transformation. The transformation specifies to get data from the RGB Hex column if the value equals web and otherwise to get data from the CMYK column.

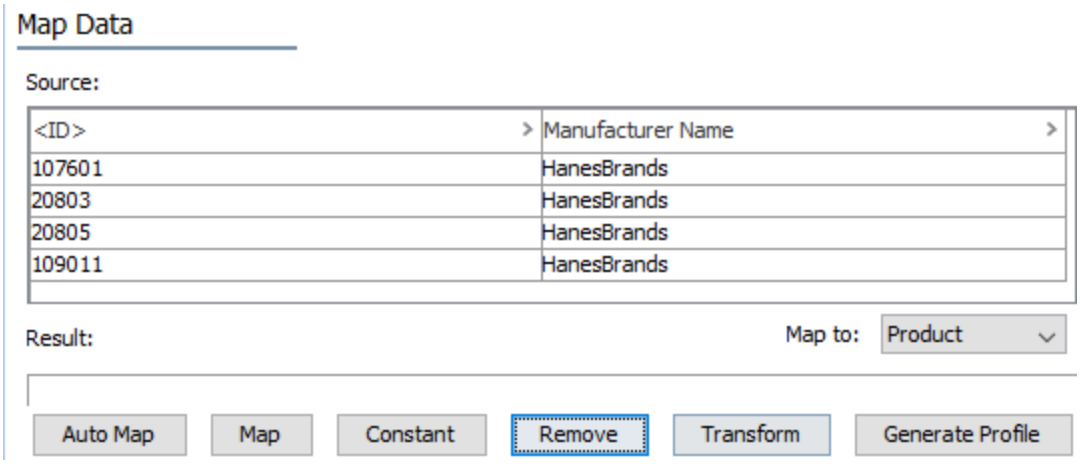

Steps
1. Select Configuration
2. Select Data Source
3. Select Format
4. Map Data
Map Data
Source:
ID
Output
CMYK
RGB Hex
114852
web
C:0M:31Y:37K:0
#FFB0A1
114854
Print
C:0M:52Y:100K:64
#SC2C00
Specify the sequence of transformations in the table below.
Target:PrimaryColor
Mandatory
Value - Source:Output
If value equals
web
Source
RGB Hex
Else source
CMYK
Add Transformation
Dimension Point
Constant:
(From Import Context)
Source:
ID
Add Transformation
Preview
PrimaryColor=Output
#FFB0A1
C:0M:52Y:100K:64
Save
Reset
Cancel
Map to:
Product
PrimaryColor=Output
web
Print
Auto Map
Map
Constant
Remove
Transform
Generate Profile
Back
Next
Finish
Cancel

Insert Text Example

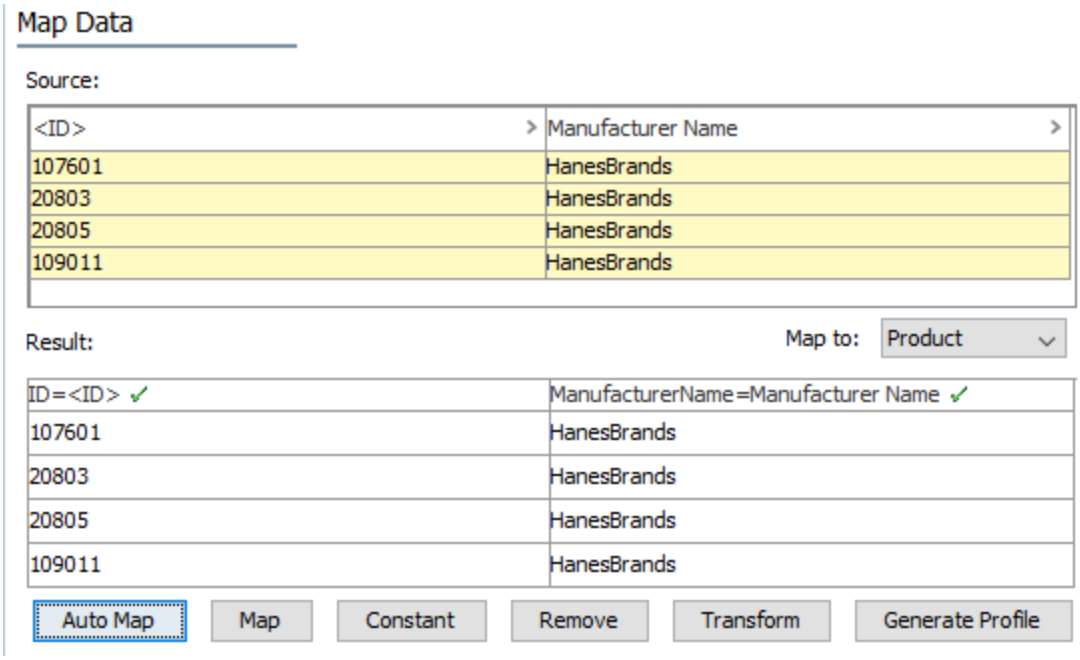
Transformations can improve the quality of the data you want to import. For general information about transformations, refer to the **Inbound Map Data - Transform** section.

Inserting text into the attribute values while importing data is possible using the **Insert text before** and the **Insert text at a given position** transformations.

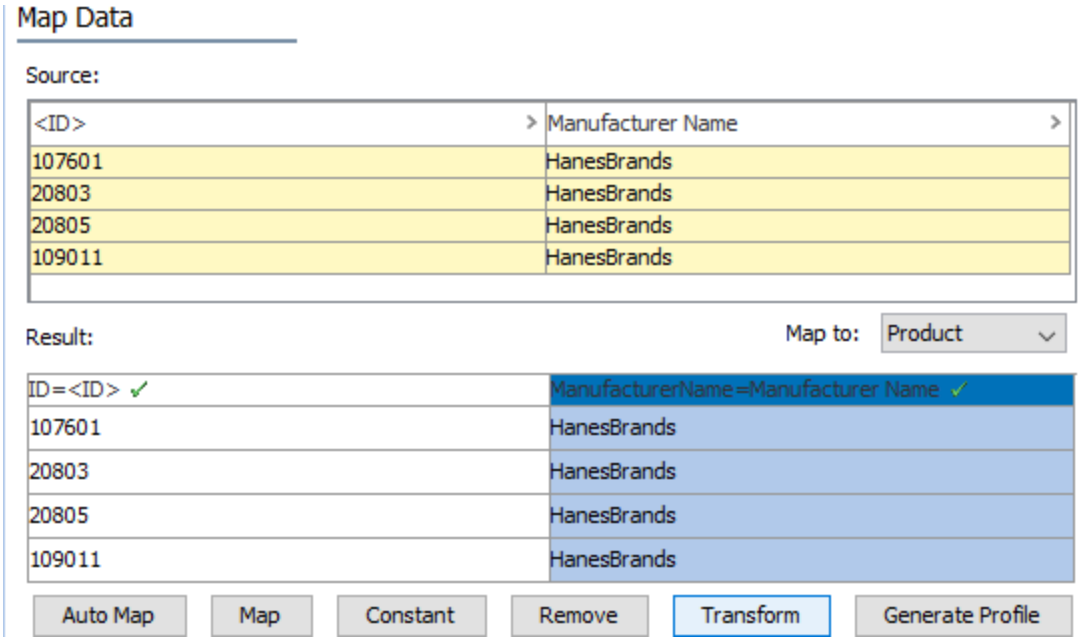
This example describes how to add new text to a field in an import file, and prepare it to be loaded into an attribute in STEP. The following data is being imported and contains incomplete data for the 'Manufacturer Name' attribute:



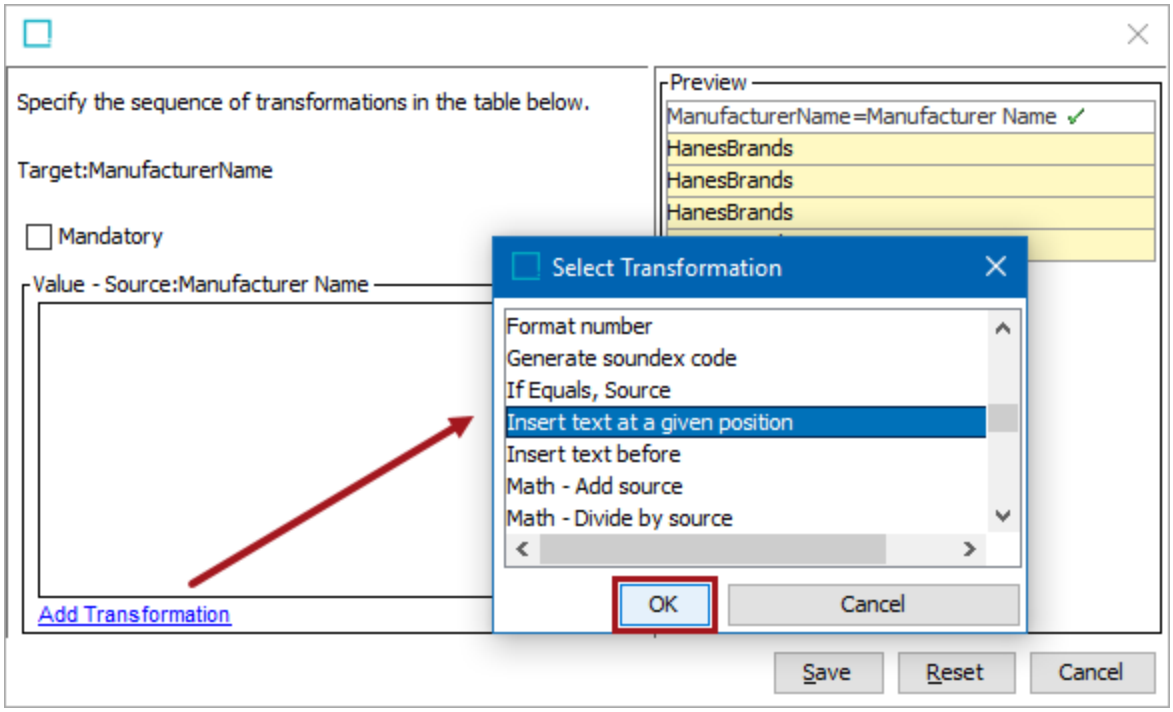
1. Load the import file into the inbound tool. For more information, refer to the **Data Exchange** topic.
2. Map the object ID in the import tool. Refer to **ID or Key - Map Inbound** for details.
3. Map the next source column to the required attribute. In this example, we map 'Manufacturer Name' to the Manufacturer Name attribute.



4. Select the result column that contains data to be modified and click the **Transform** button.

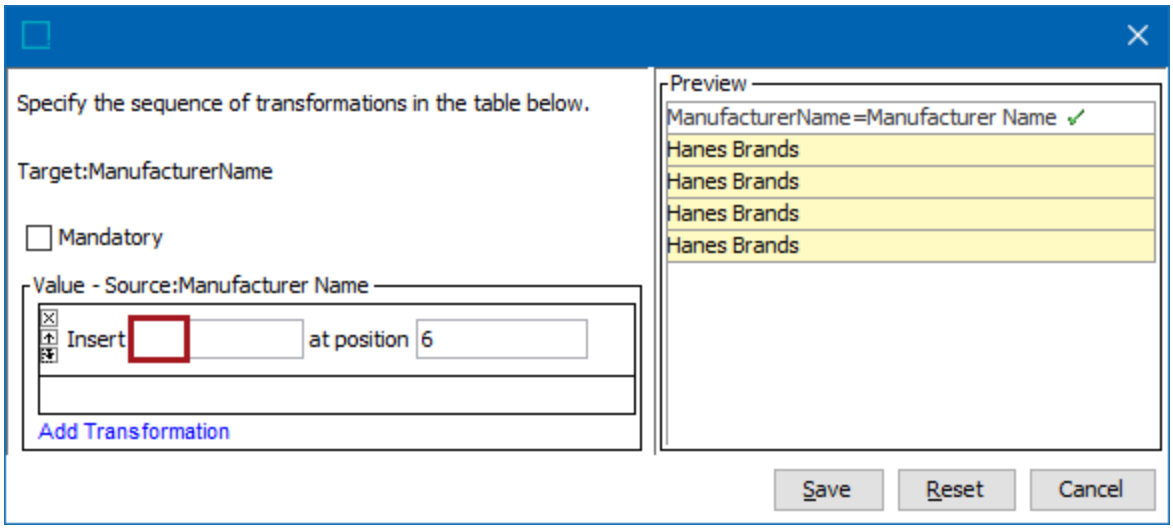


5. In the Transformation window, click on the **Add Transformation** link, select the **Insert text at a given position** transformation, and click **OK**.



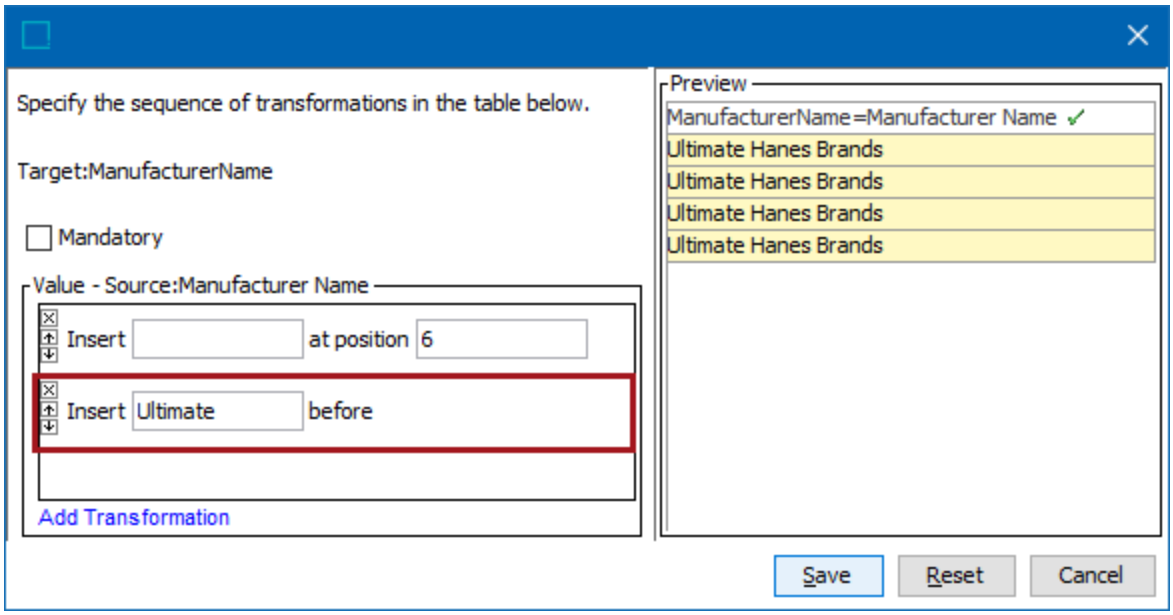
6. Although not obvious in the following image, add a space in the **Insert** parameter, and indicate the position of the space by adding a number in the **at position** parameter. In this example, since the space belongs between the words 'Hanes' and 'Brands', we enter the number 6 for the position so the space is inserted as character 6.

   Click or tab out of the parameter to complete the configuration. The Preview pane shows the results of the transformation.



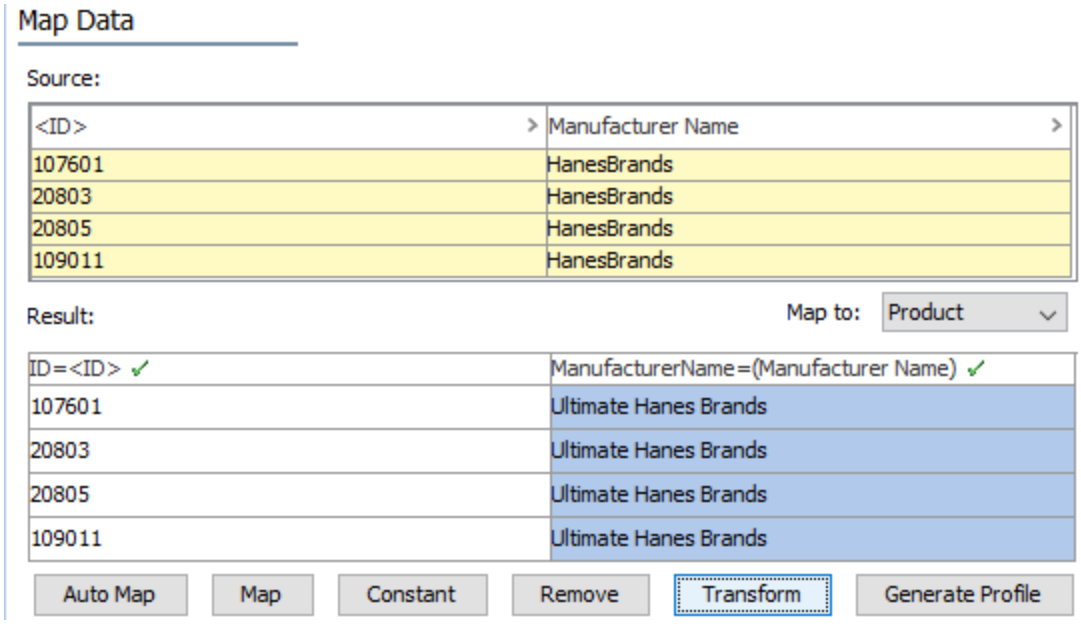
7. In the Transformation window, click on the **Add Transformation** link again, select the **Insert text before** transformation, and click **OK**.

8. In the **Insert before** parameter, add the text that should precede the existing value. Remember to add any necessary space characters. Click or tab out of the parameter to update the Preview pane.



Important: Transformations are evaluated from the top down. Carefully consider the order of the transformations applied. To remove or rearrange transformations, use the Delete, Up arrow, and Down arrow buttons .

9. If necessary, check the 'Mandatory' option to skip objects with empty values during import.
10. Click **Save** to display the transformed data in the Map Data step.

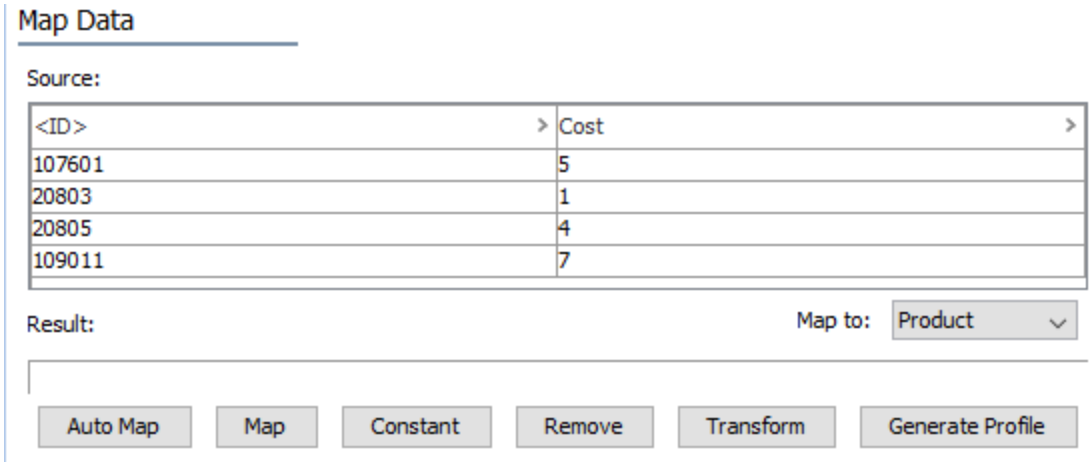


Math Operation Example

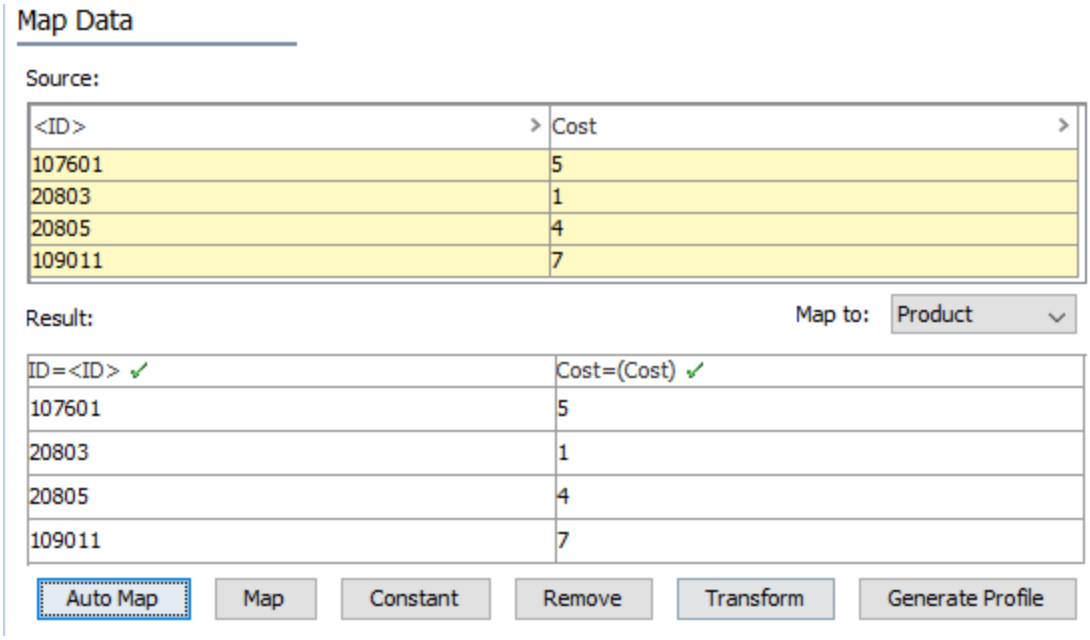
Transformations can improve the quality of the data you want to import. For general information about transformations, refer to the **Inbound Map Data - Transform** section.

Using the **Math operation (+,-,*,/)** transformation, basic math operations can be applied to inbound data mapped to attributes with Validation Base Types for numbers (for example, integer or number). Values can then be imported.

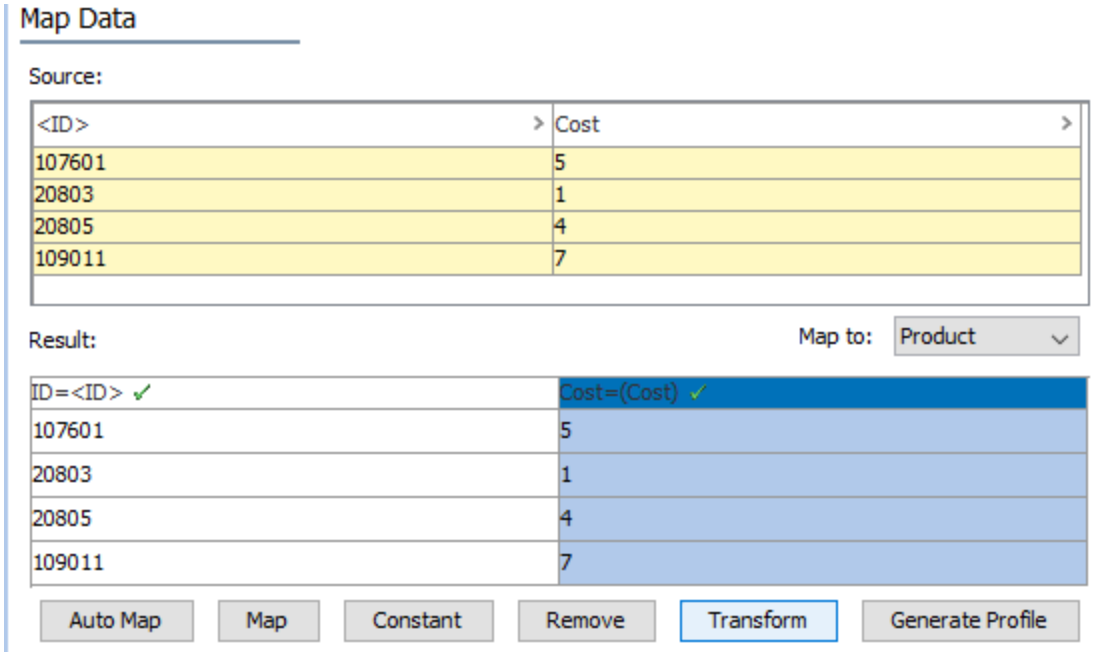
This example applies each of the math operations to a single field of the import file as shown below.



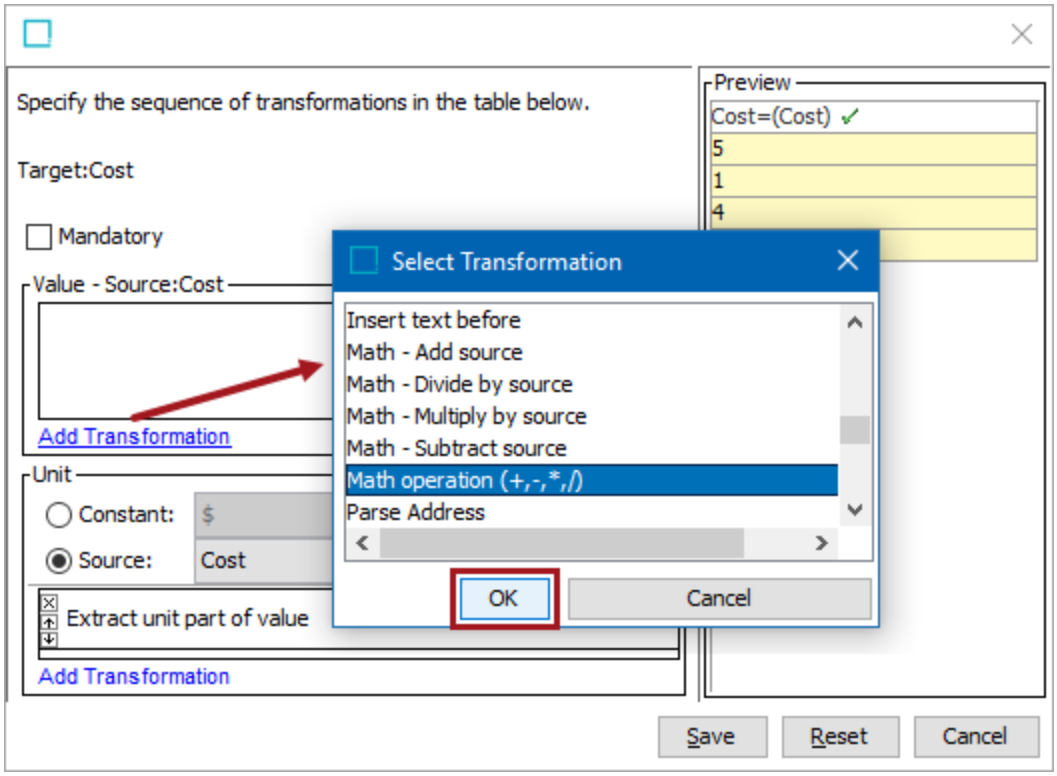
1. Load the import file into the inbound tool. For more information, refer to the **Data Exchange** topic.
2. Map the object ID in the import tool. Refer to **ID or Key - Map Inbound** for details.
3. Map the next source column to the required attribute. In this example, we map 'Cost' to the Cost attribute.



4. Select the result column that contains data to be modified and click the **Transform** button.

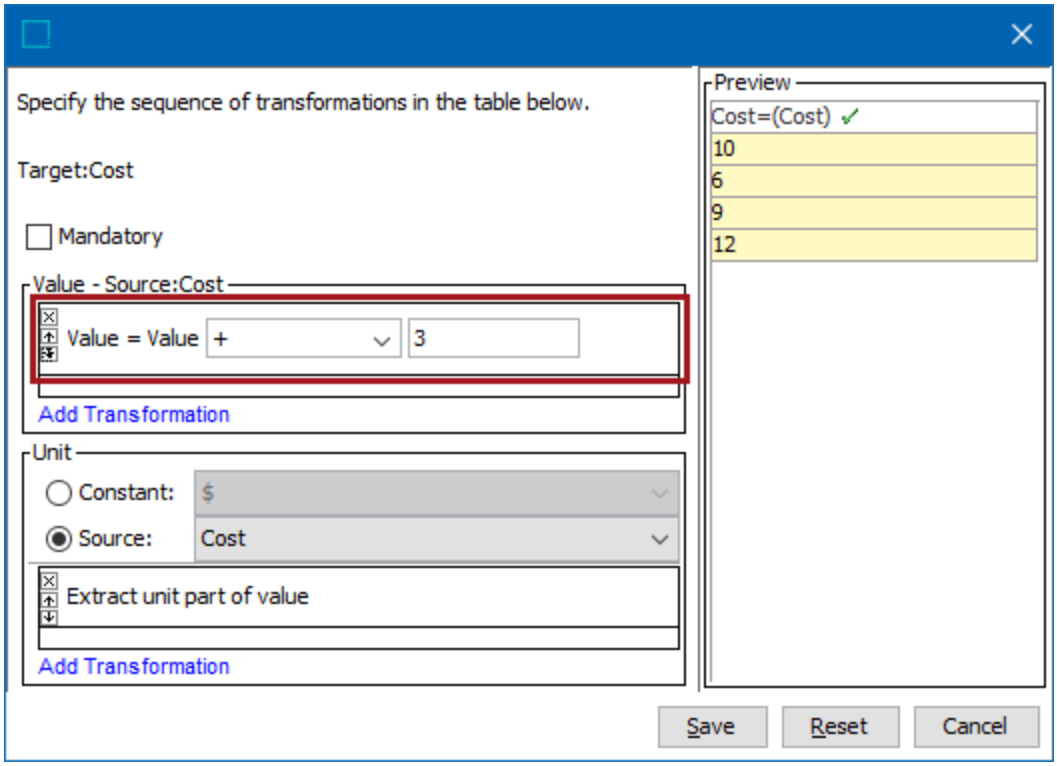


5. In the Transformation window, click on the **Add Transformation** link, select the **Math operation (+,-,*,/)** transformation, and click **OK**.

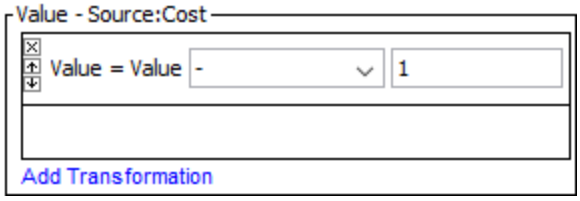


6. In the **Value** parameter, use the dropdown to choose the **+** math operation, and in the text field, enter the number that should be added to the inbound value. In this example, 3 is added to the value.

   Click or tab out of the parameter to complete the configuration. The Preview pane shows the results of the transformation.



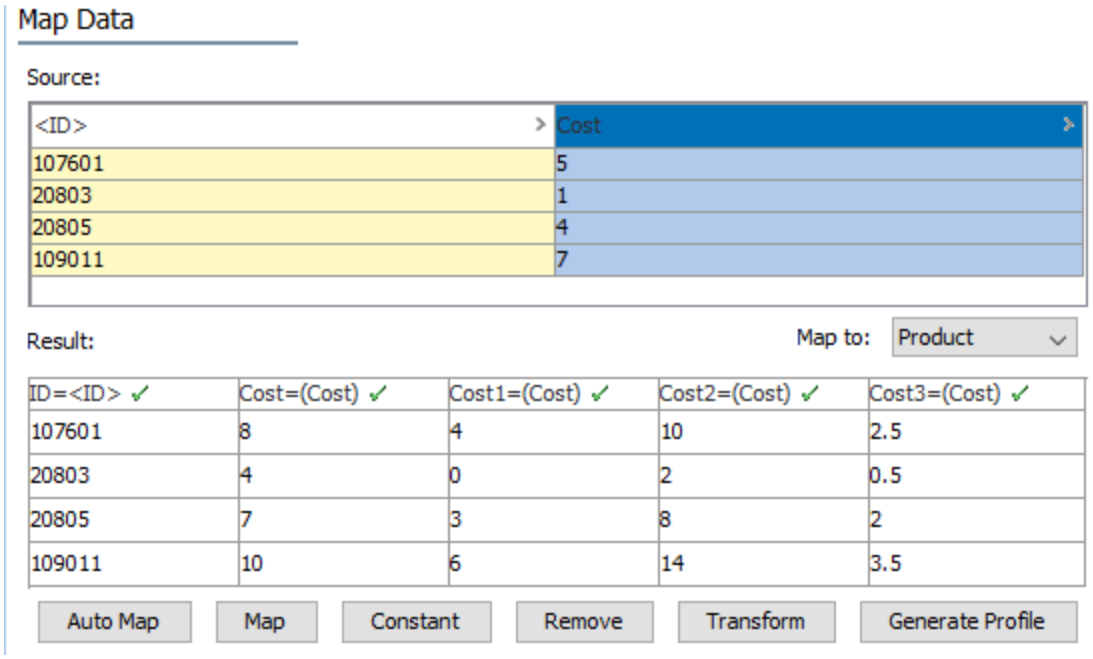
7. Click **Save** to display the transformed data in the Map Data step.
8. To illustrate each math operation, three additional attributes for Cost are used: Cost1, Cost2, and Cost3. Repeat steps 3 - 6, selecting one of the additional Cost attributes each time, confirming to map the same column again, and selecting a different math operation each time as follows:
   - In the **Value** parameter, use the dropdown to choose the **-** math operation, and in the text field, enter the number that should be subtracted from the inbound value. In this example, 1 is subtracted from the value.



   - In the **Value** parameter, use the dropdown to choose the ***** math operation, and in the text field, enter the number that should be multiplied with the inbound value. In this example, the value is multiplied by 2.

- In the **Value** parameter, use the dropdown to choose the **/** math operation, and in the text field, enter the number that should be used to divide the inbound value. In this example, the value is divided by 2.

9. If necessary, check the 'Mandatory' option to skip objects with empty values during import.
10. Click **Save** to display all of the transformations in the Map Data step Result section as follows:
    - The **Cost = (Cost)** column shows where the original value had 3 added.
    - The **Cost1 = (Cost)** column shows where the original value had 1 subtracted.
    - The **Cost2 = (Cost)** column shows where the original value was multiplied by 2.
    - The **Cost3 = (Cost)** column shows where the original value was divided by 2.



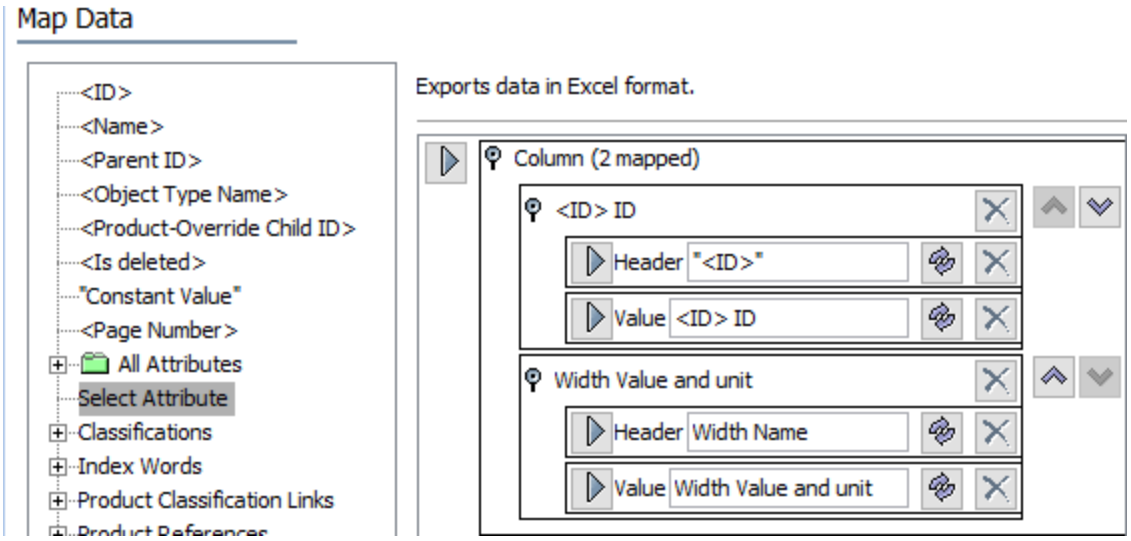
Merge Attributes and Constant Example

You can modify an output by concatenating separate pieces of data from STEP to display in the same field upon export.

For example, you can merge three attributes with the names width, depth, and height into one export field in the format 'width x depth x height', and also show only the value, not the unit. This mapping will include concatenating values (the measurement) with constant values (an abbreviation to indicate width, depth, or height).

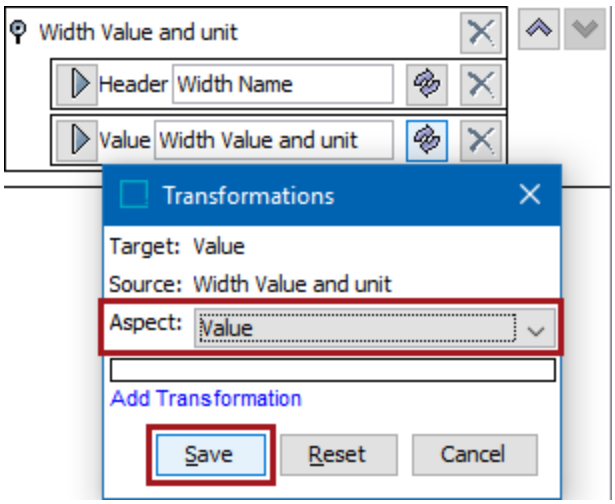
1. Map **<ID>**. Refer to the **ID or Key - Data Source Outbound** topic for details.
2. Map **Select Attribute**, and search for the Width attribute. Refer to the **Attributes and Data Containers - Data Source Outbound** topic for details.

   Two items have been mapped and display in the right panel.

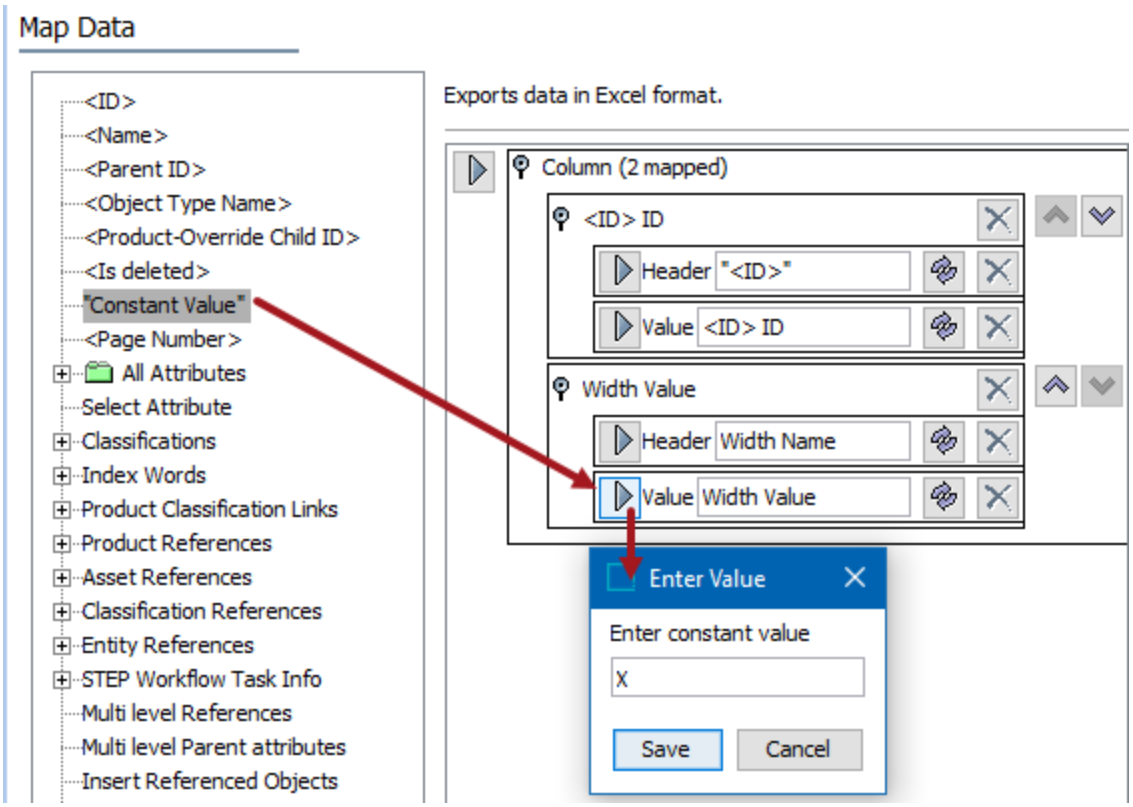


3. Open the Width attribute flipper to display the Header and Value elements.
4. Click the **Transformation** button ( ) for the width attribute Value element to display the Transformations dialog.
5. From the **Aspect** parameter dropdown select **Value**, and click **Save** to apply the change.

> **Note:** Although you removed the 'Value And Unit' aspect by choosing 'Value' aspect, if the attributes have valid units assigned to them, you can extract the units in the same manner.

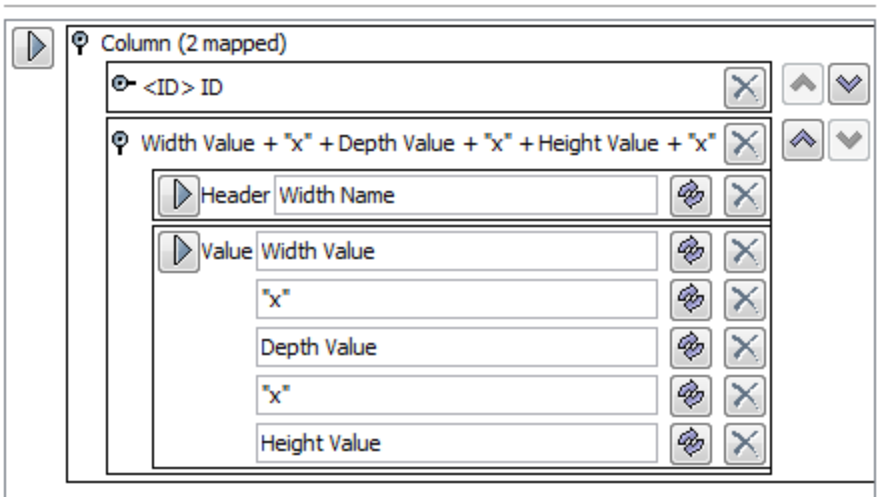


6. Map **Constant Value** to the Value element of your Width attribute in order to merge elements.
7. In the **Enter Value** dialog type **X** and click the **Save** button.



8. Repeat steps 2-7 and add the attributes depth and height, excluding the final X constant.
   The right panel now looks like the following image.

Exports data in Excel format.



9. Complete any additional mappings and initiate the export.

Results

Using the mapping shown above, the following output is generated and includes the ID of the object and the values for the width, depth, and height attributes in a single column. When no value of attribute being merged is used along with a constant, the value will be skipped while the other attributes and constants appear as defined, as shown below when the depth value is not available.

	A	B
1	<ID>	Width
2	18212	13x10x12
3	124188	15xx16
4	124148	15x16x20
5	18216	12x11x12

When exporting in a tabular format, you can also add a user-defined column header by mapping a Constant Value for the Header element. Additionally, the header could be a combination of multiple pieces of data, using the merging steps above.

For more information on each outbound mapping option, refer to the **Outbound Map Data - Data Source** documentation.

Merge Data for Multi-Valued Data Example

Transformations can improve the quality of the data you want to import. For general information about transformations, refer to the **Inbound Map Data - Transform** section.

Multiple values for a multi-valued attribute and multiple targets for a reference or link type can be imported from a single field in the import file. In this case when using a CSV, the values or Target IDs must be separated by a delimiter that is different from the delimiter used for separating columns (alternately, the value can be enclosed in quotes).

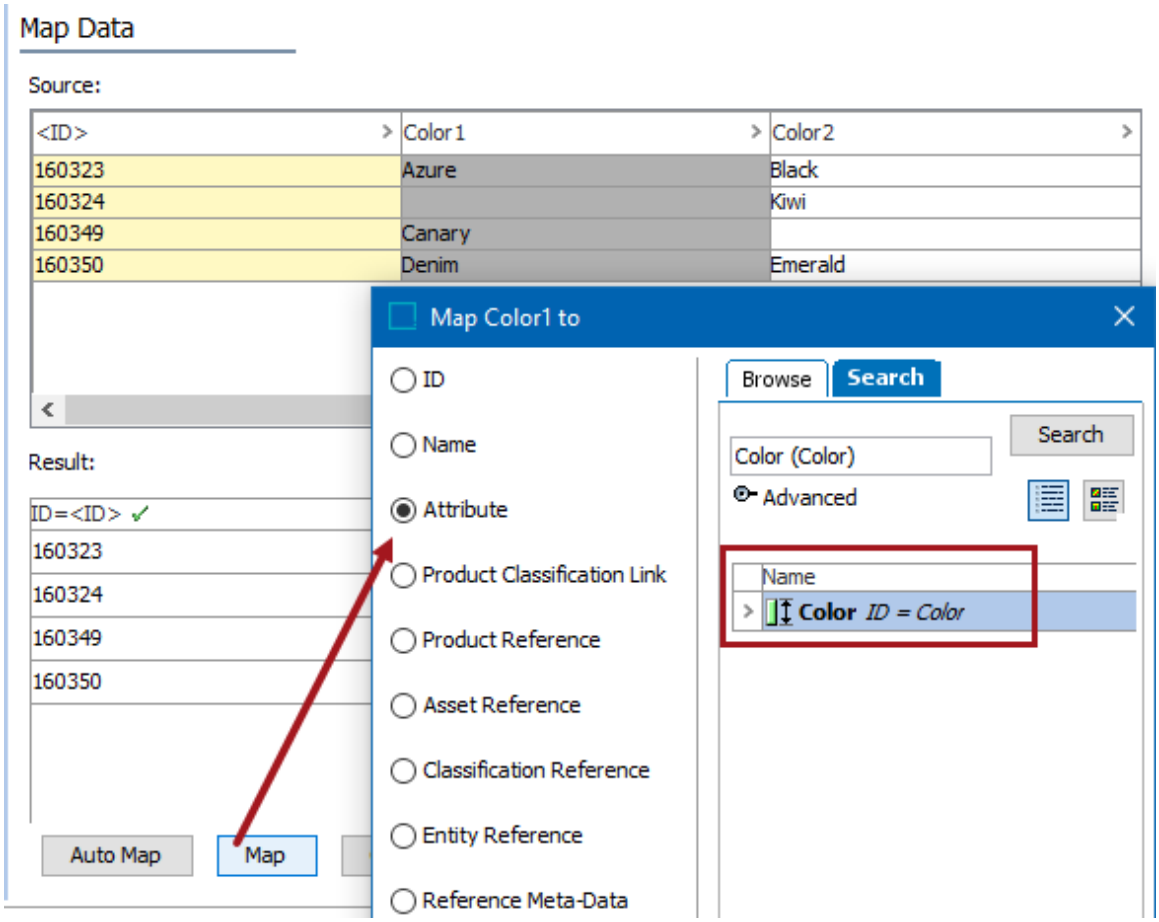
Below are the following examples of merging import data to result in values for a multi-valued attribute and/or multiple targets for a reference or link type:

- **Merge single values from multiple columns** - in this scenario, the import file contains the values for a multi-valued attribute in more than one column. Transformations are used to combine the data into a single STEP attribute.
- **Merge single values from multiple columns using an Intermediate Variable** - in this scenario, the import file is the same as the first one (it contains the values for a multi-valued attribute in more than one column). Transformations are used on an intermediate variable to combine the data into a single STEP attribute. While the method is different, the result is the same as the first scenario.
- **Merge multiple values from multiple columns** - in this scenario, the import file includes two columns that each have multiple values in a single cell. The values from the first column should be paired with the corresponding value in the second column.
- **Merge multiple values from a single column** - In this scenario, the import file contains a single column that has multiple targets for a reference type within a single cell.

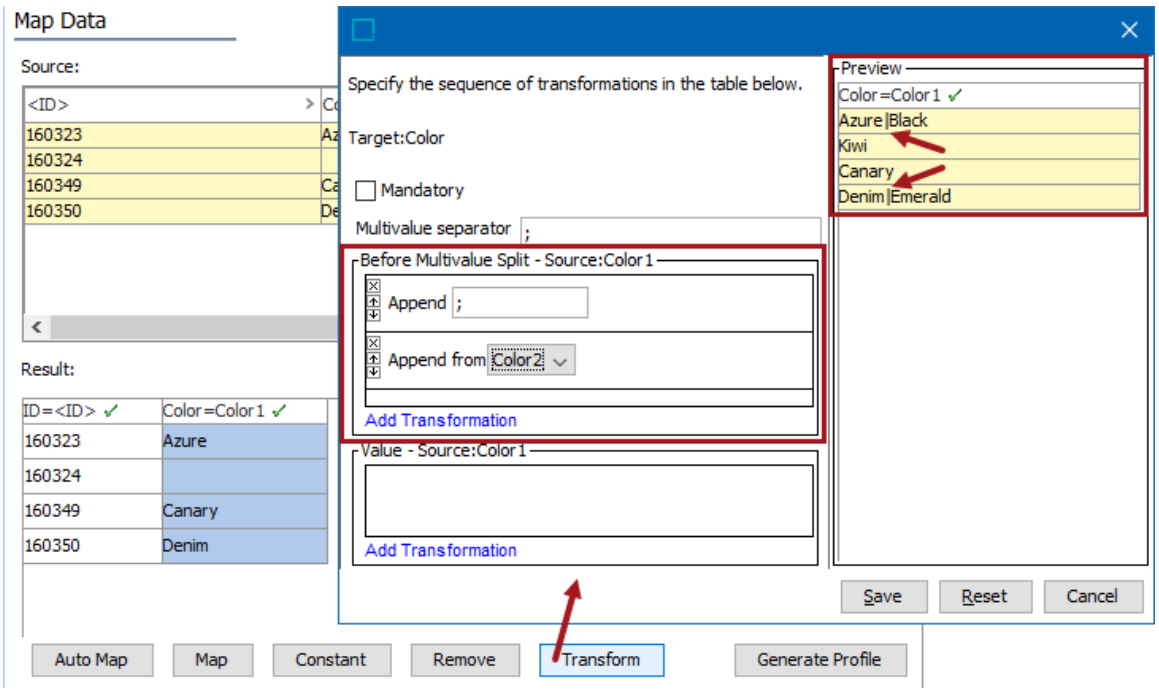
Merge single values from multiple columns

In this example, the import file contains the values for a multi-valued attribute in more than one column. Transformations are used to combine the data into a single STEP attribute. The end result is that the values of columns 'Color 1' and 'Color 2' are merged into a single multi-valued attribute named 'Color.'

1. Map the 'Color 1' column to the 'Color' attribute.

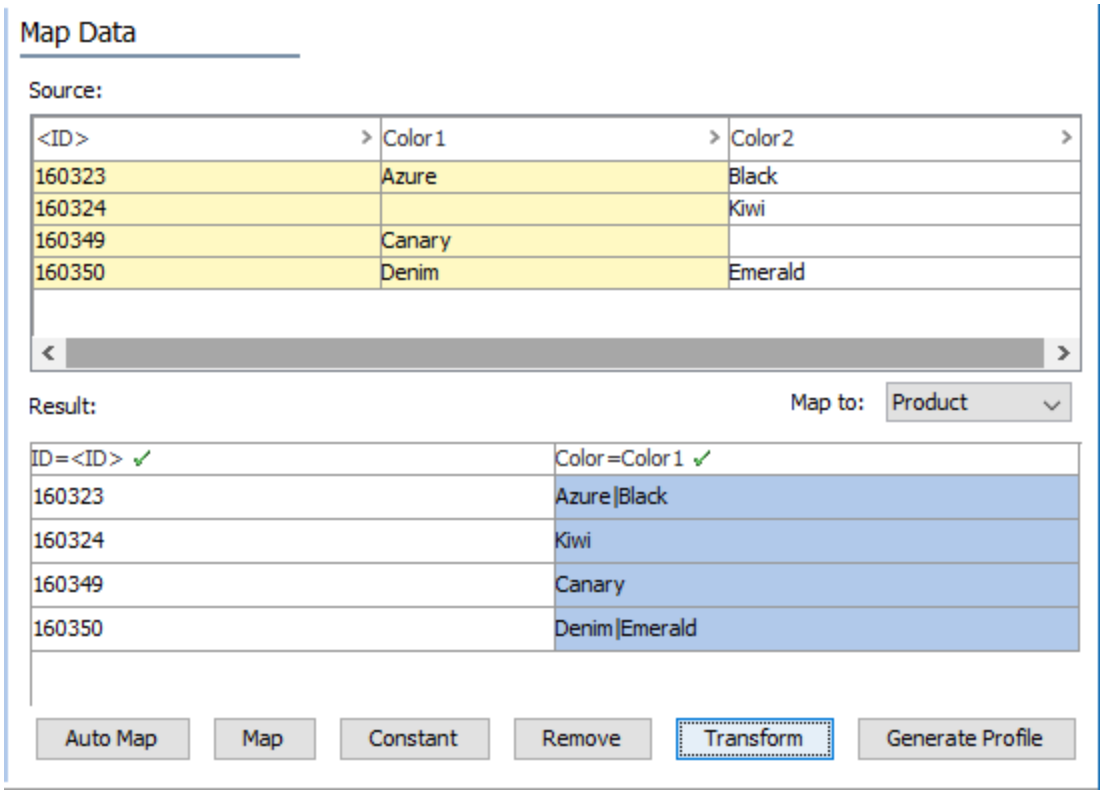


2. In the **Result** area, select the 'Color=Color 1' column and click the **Transform** button to display the transformation dialog.
3. In the **Before Multivalue Split** area, add the **Append text** with a semi-colon (;), and the **Append from source** transformation selecting the 'Color2' column from the dropdown.

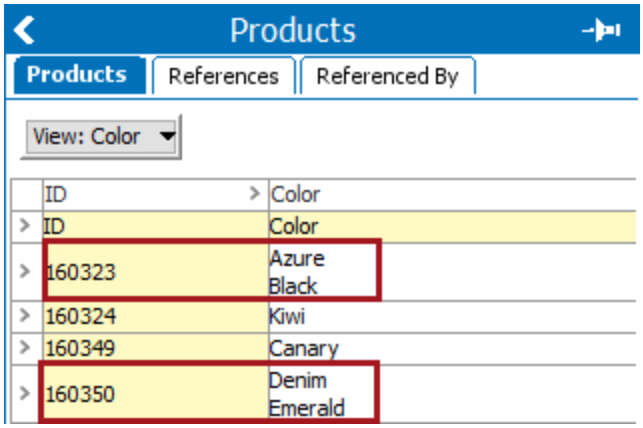


In the Preview area, verify that the rows with multiple values are displayed correctly with a pipe (|) separator. This is required for a successful import of multi-values.

4. Click **Save** to display the result of the transformation in the Result area. Continue navigating through the mappings and rest of the wizard.



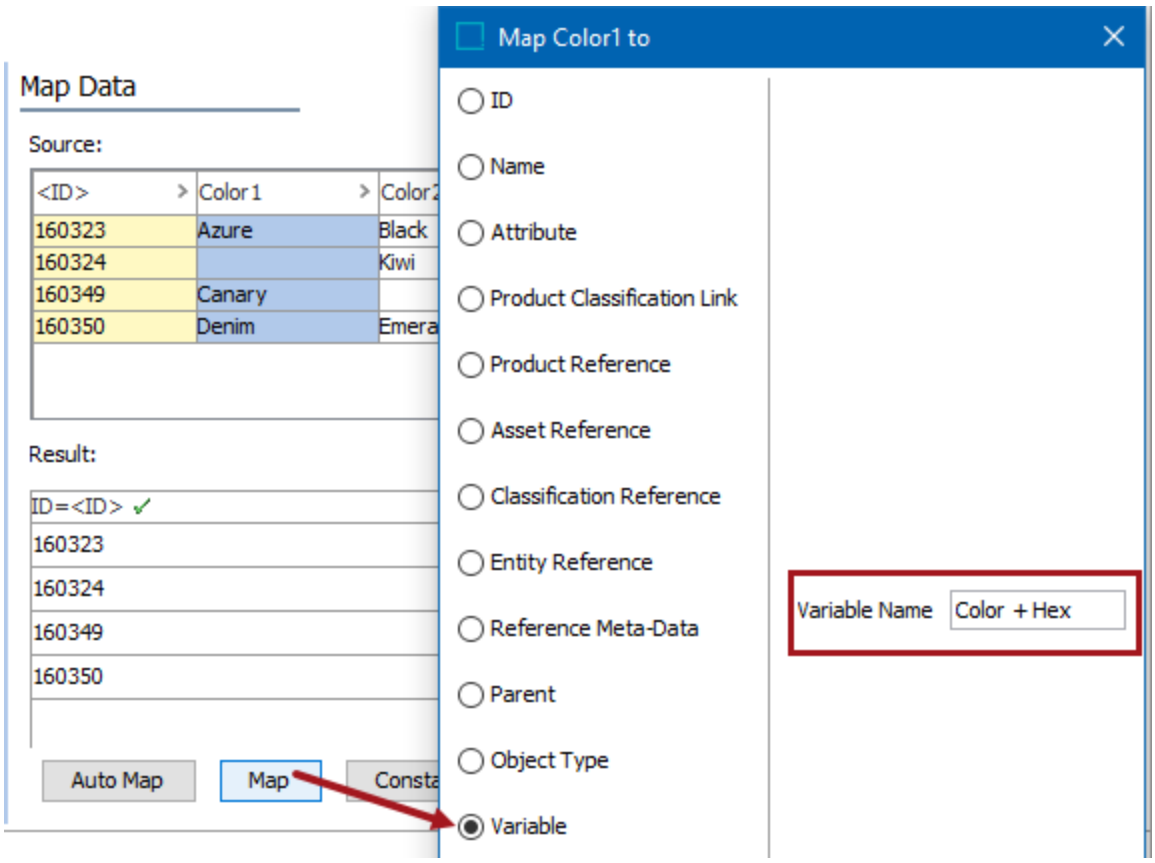
Viewing the products that were modified by the import, shows that the multi-valued Color attribute correctly includes multiple colors when available.



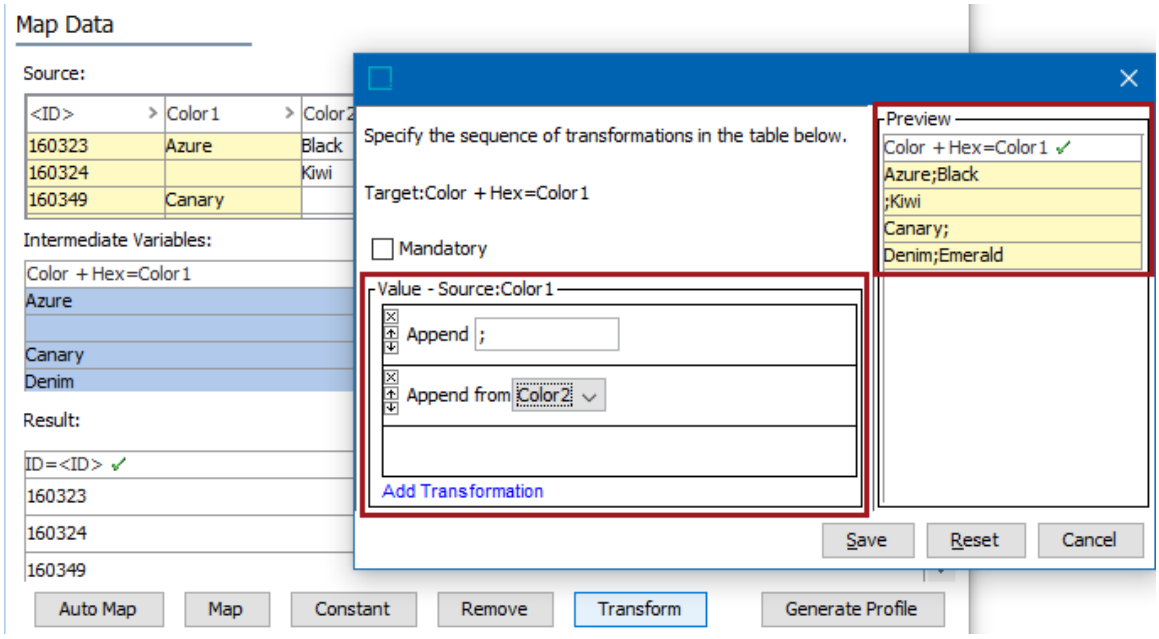
Merge single values from multiple columns using an Intermediate Variable

In this example, the import file is the same as the one before (it contains the values for a multi-valued attribute in more than one column). Transformations are used on an intermediate variable to combine the data into a single STEP attribute. While the method is different, the result is the same as the first scenario, namely that the values of columns 'Color 1' and 'Color 2' are merged into a single multi-valued attribute named 'Color.'

1. Map the 'Color 1' column to the 'Color + Hex' variable.

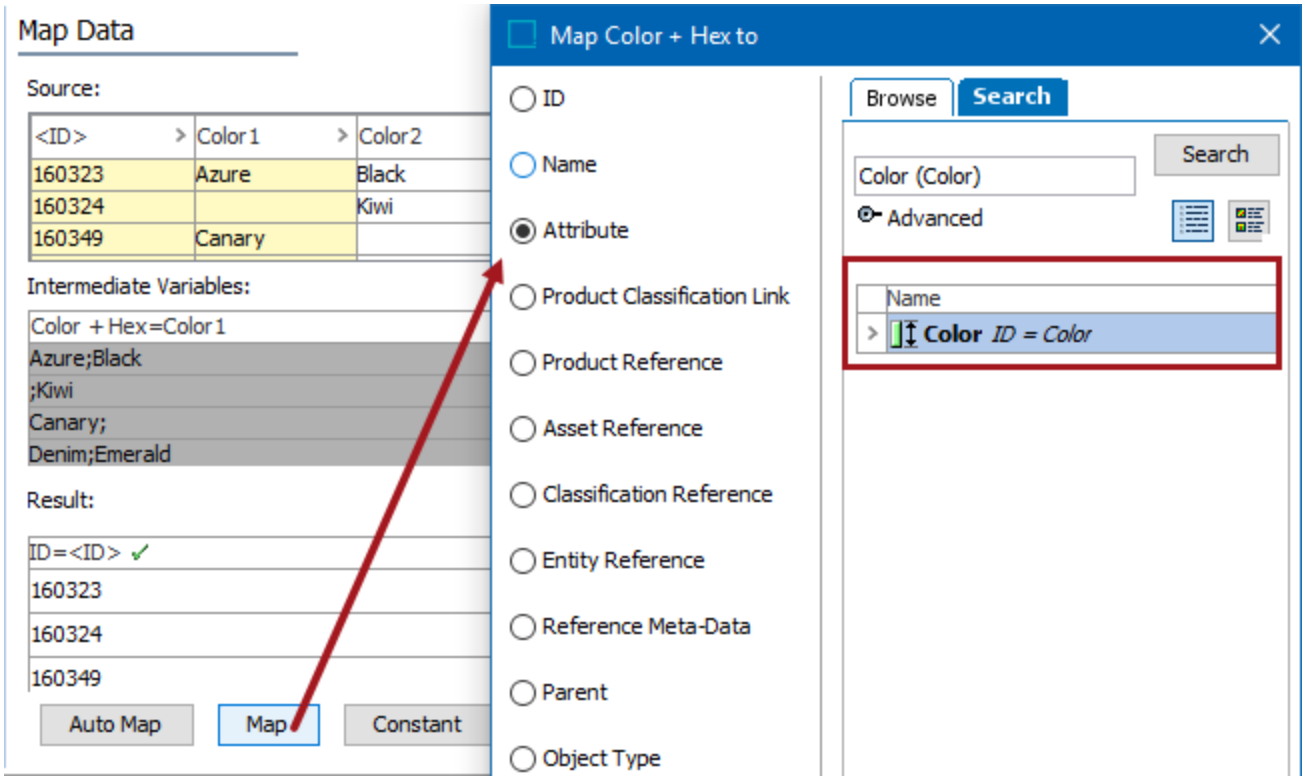


2. In the **Intermediate Variables** area, select the 'Color + Hex =Color 1' column and click the **Transform** button to display the transformation dialog.
3. In the **Value** area, add the **Append text** with a semi-colon (;), and the **Append from source** transformation selecting the 'Color2' column from the dropdown.

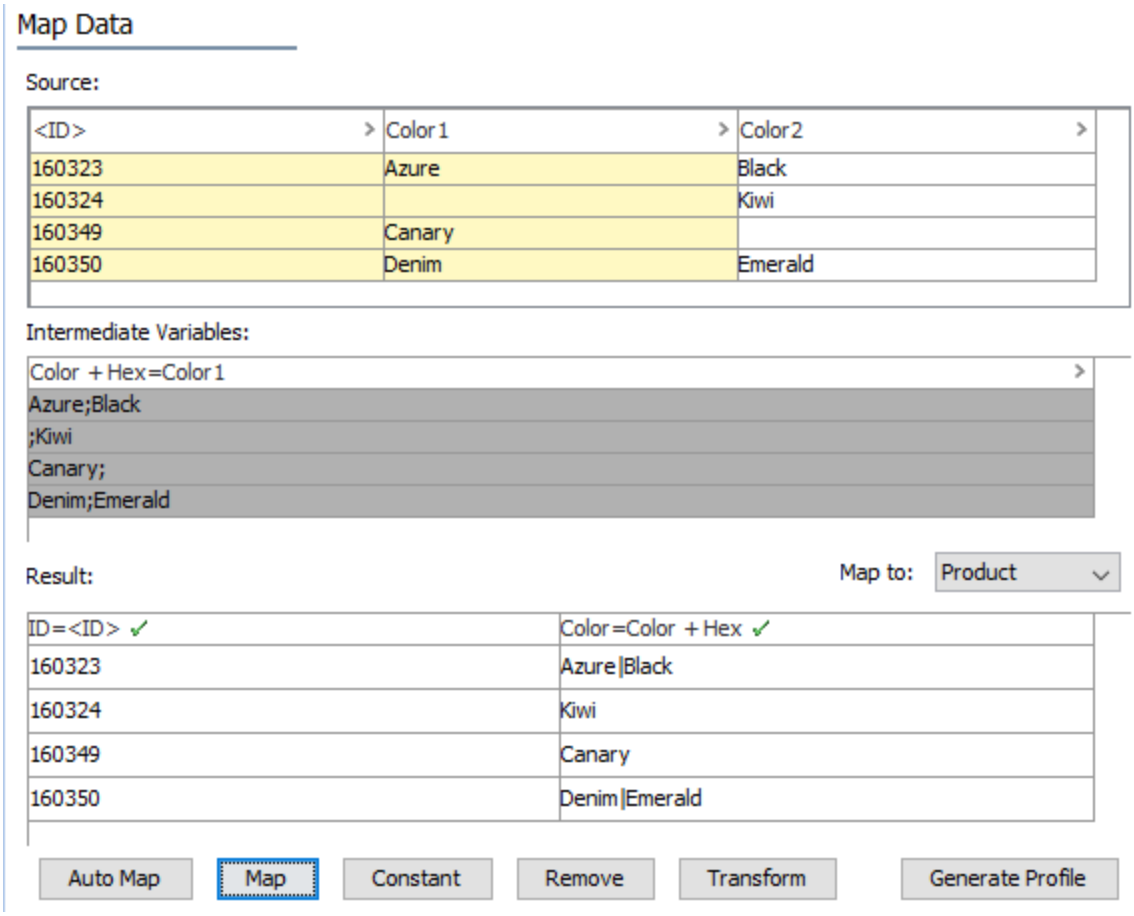


In the Preview area, verify that the rows with multiple values are displayed correctly with a semi-colon (;) separator.

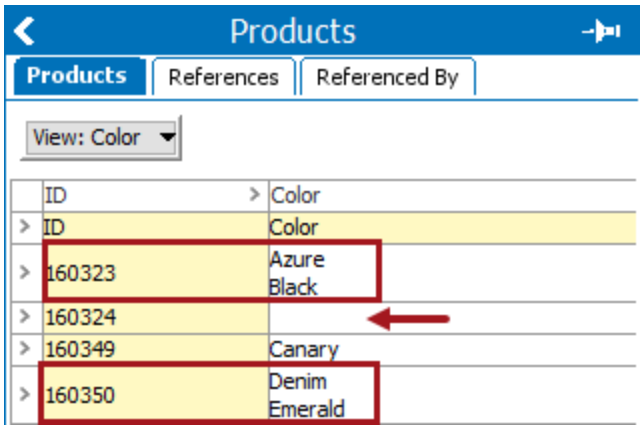
4. Click **Save** to display the result of the transformation in the Intermediate Variables area, select the 'Color + Hex =Color 1' variable column, and map it to the 'Color' attribute.



5. Click **Save** to display the result of the transformation and variable mapping in the Result area. Continue navigating through the mappings and rest of the wizard.



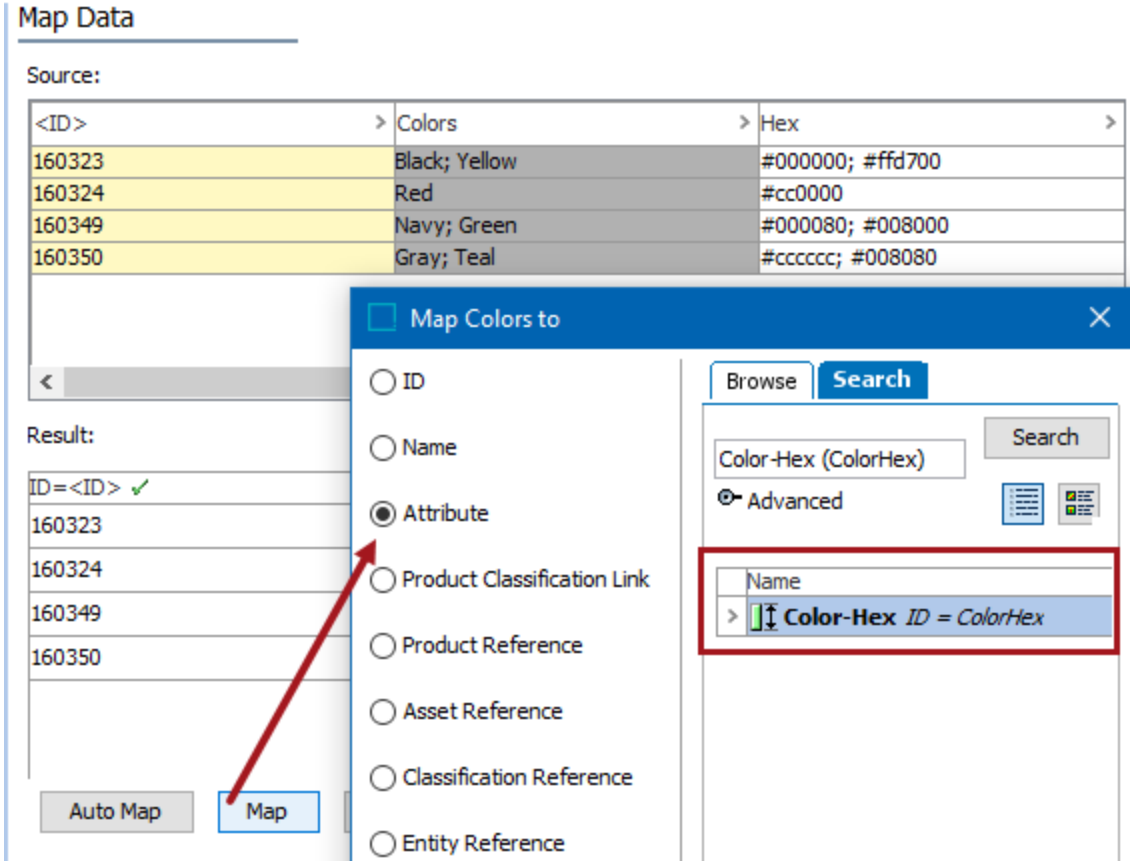
Viewing the products that were modified by the import, shows that the multi-valued Color attribute correctly includes multiple colors when available. However, note that when first column mapped has no value, indicated by the arrow below, that the import does not include the second value.



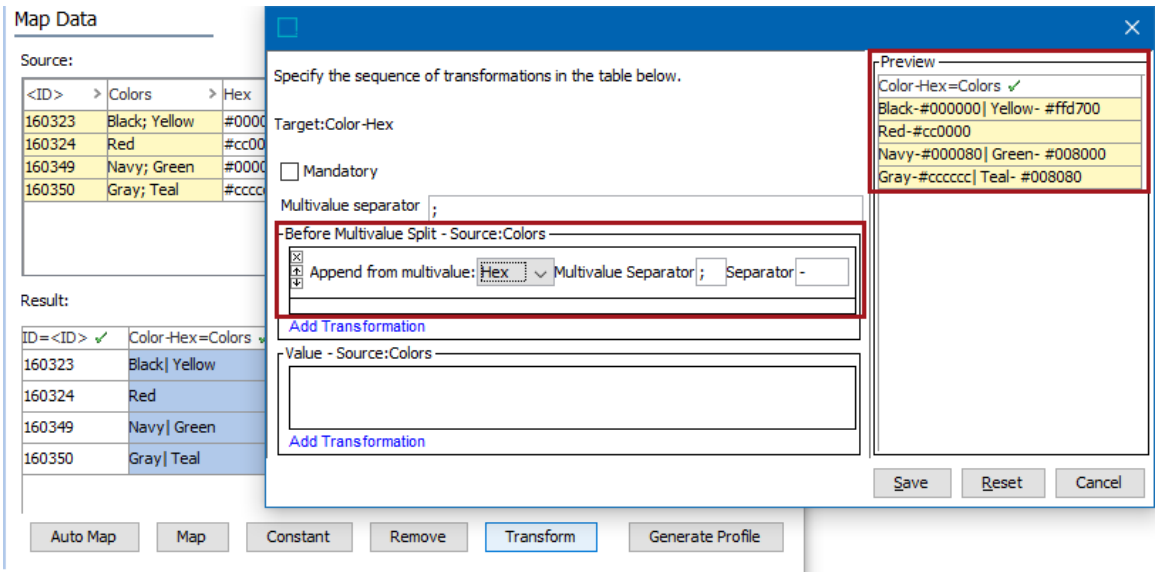
Merge multiple values from multiple columns

In this example, the import file includes two columns that each have multiple values in a single cell. The color values from the first column ('Colors') should be paired with the corresponding hexadecimal values in the second column ('Hex'). The end result is that the 'Colors' and 'Hex' columns are merged into a multi-valued attribute ('Color-Hex'), displaying each as a color-hexadecimal pair.

1. Map 'Colors' to 'Color-Hex.'

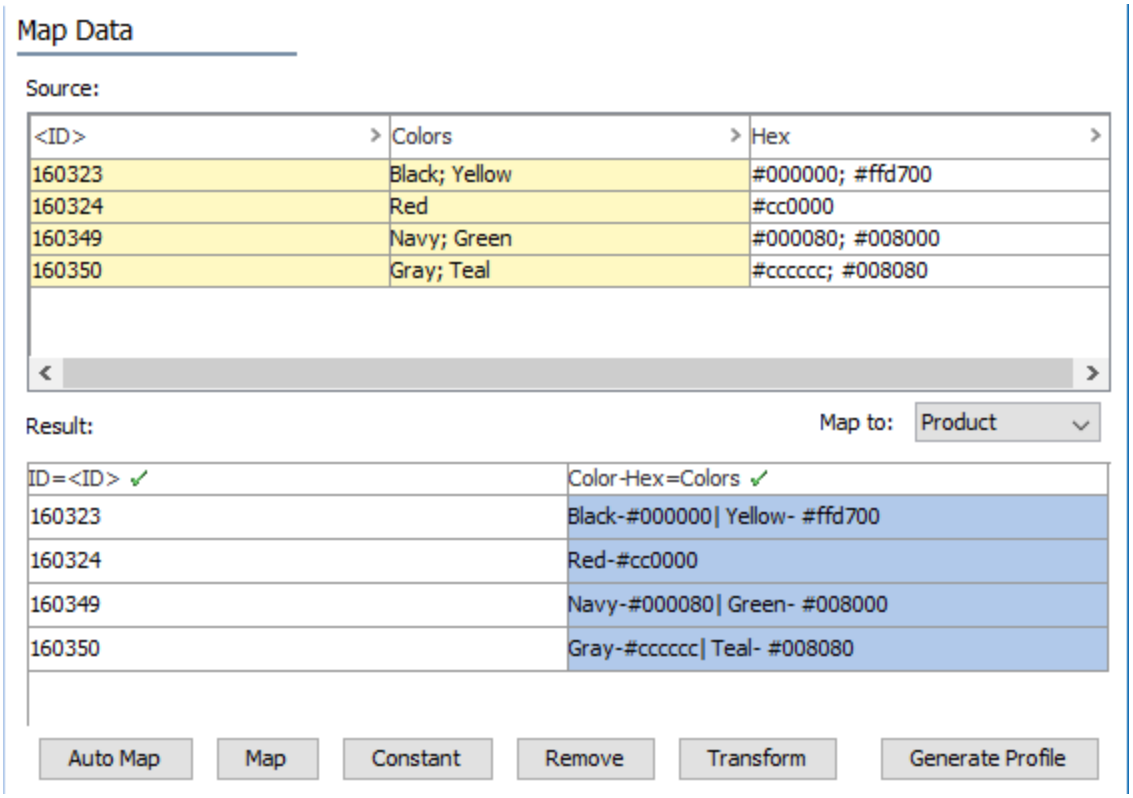


2. In the **Result** area, select the mapped 'Color-Hex=Colors' column, and click the **Transform** button to display the transformation dialog.
3. In the **Before Multivalue Split** area, add the **Append from multivalue source** transformation selecting the 'Hex' column from the dropdown. The separator, '-' in the example, is optional.

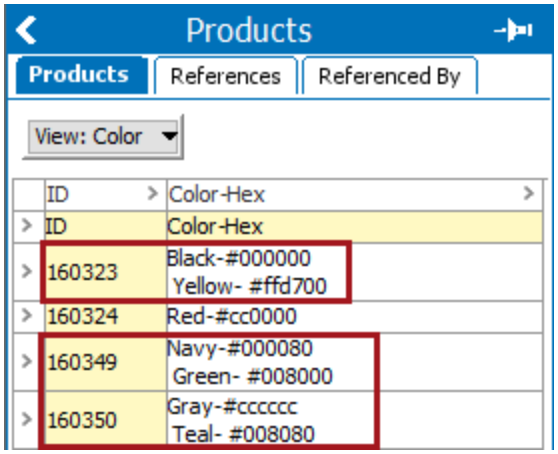


In the Preview area, verify that the rows with multiple values are displayed correctly with a pipe (|) separator. This is required for a successful import of multi-values.

4. Click **Save** to display the result of the transformation in the Result area. Continue navigating through the mappings and rest of the wizard.



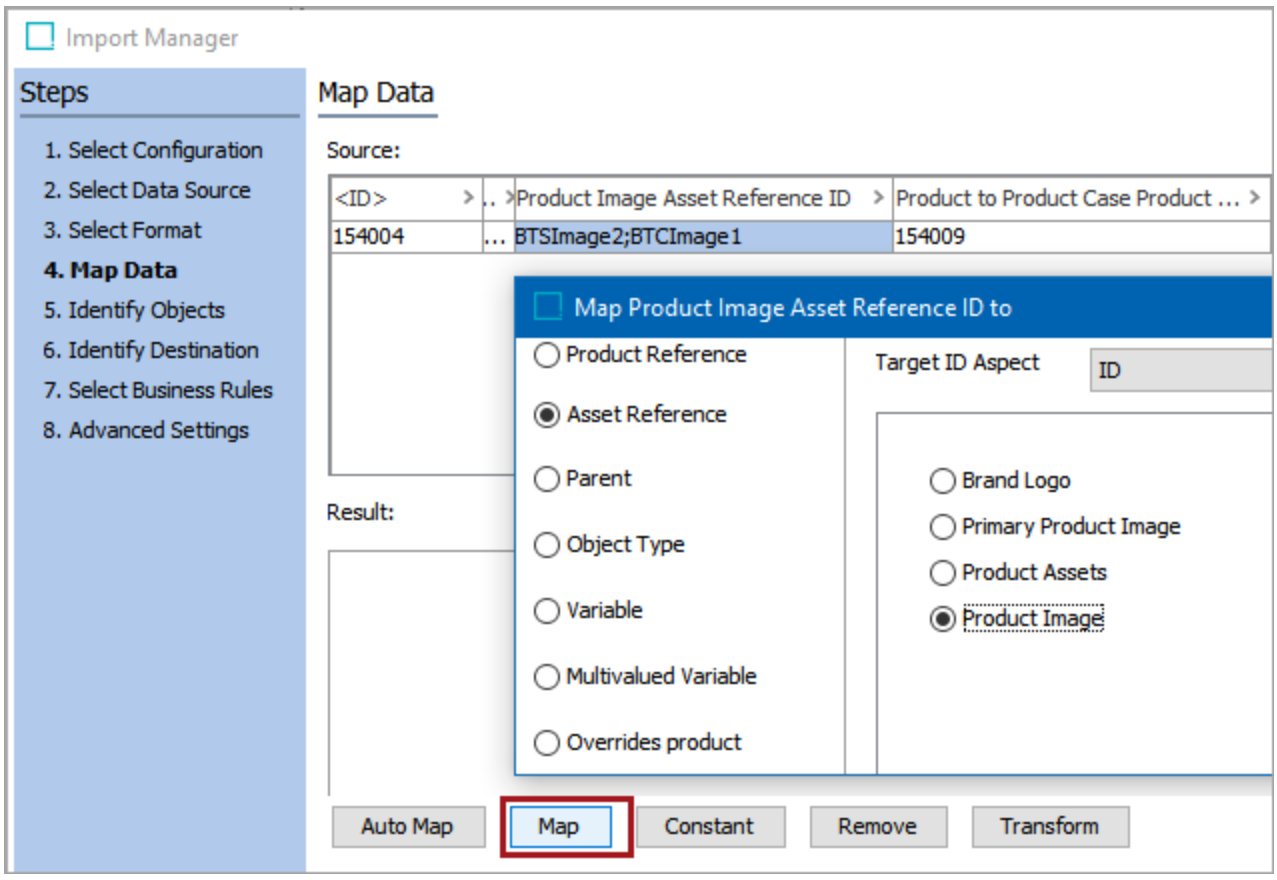
Viewing the products that were modified by the import, shows that the multi-valued Color-Hex attribute correctly includes multiple colors and hexadecimal values when available.



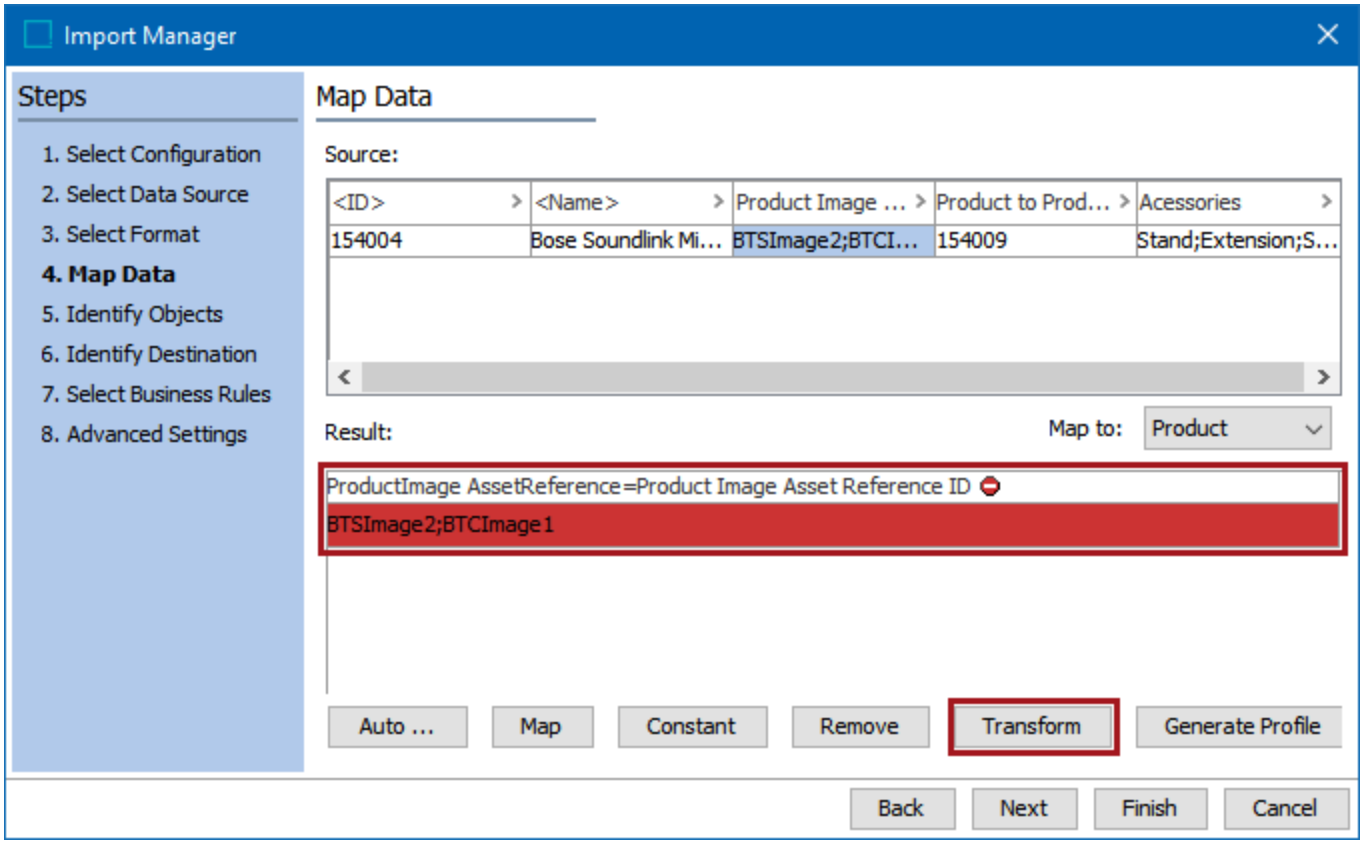
Merge multiple values from a single column

In this example, the import file contains a single column that contains multiple targets for reference types. The end result is that file can be imported without moving the targeted references values into different columns or rows by using the 'Transformation' feature.

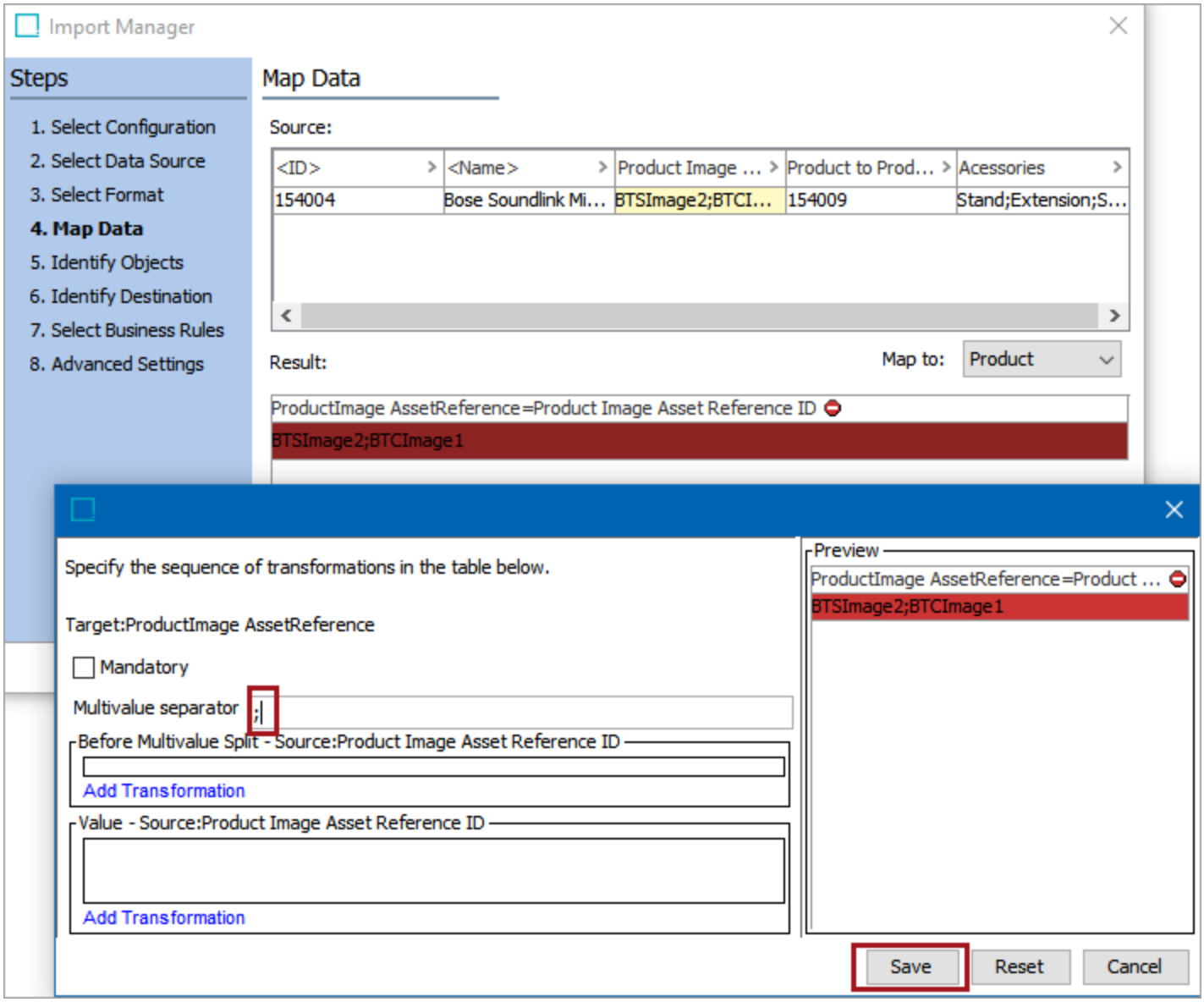
1. With **Asset Reference** selected, click the **Map** button to map Product Image Asset Reference ID to Product Image**.**



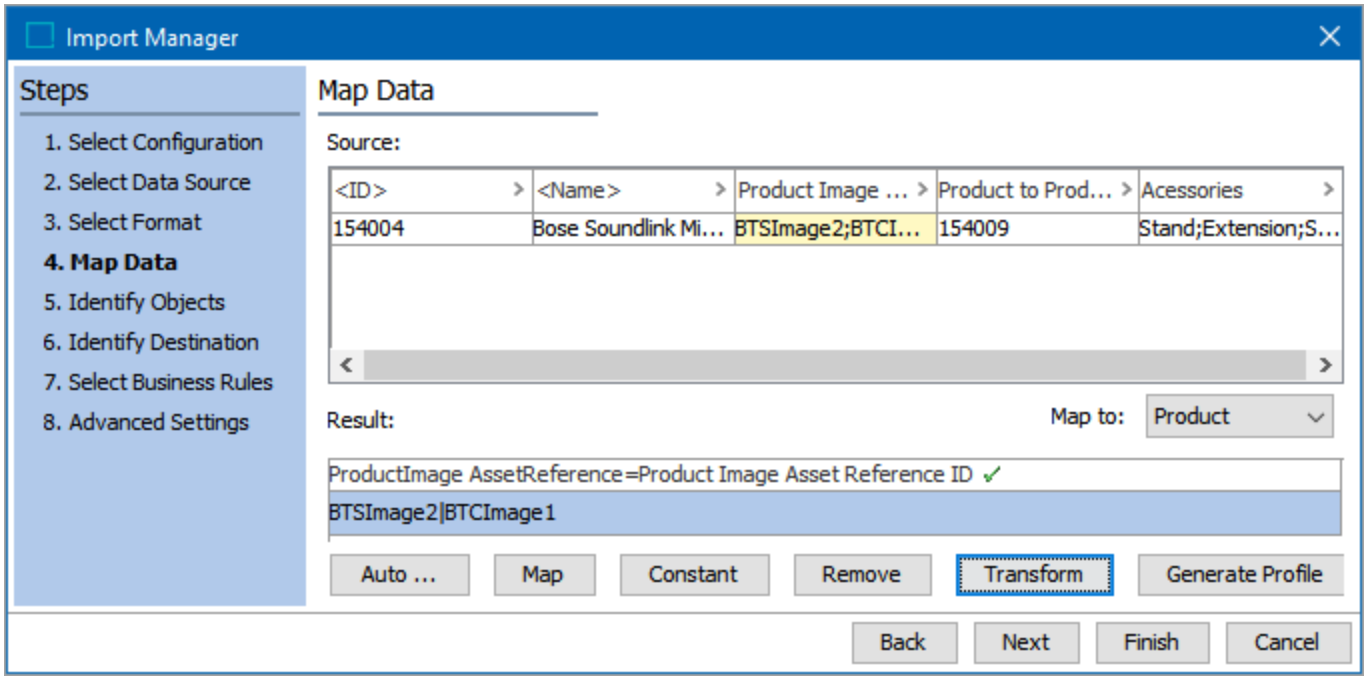
2. In the **Result** area, select the ProductImage AssetReference = Product Image Asset Reference ID column and click the **Transform** button.



3. In the Transform dialog, because the values are separated by a semicolon in the import file, add a semicolon within the **Multivalue separator** text field and click **Save**.



The single column with multiple reference targets is now successfully mapped.

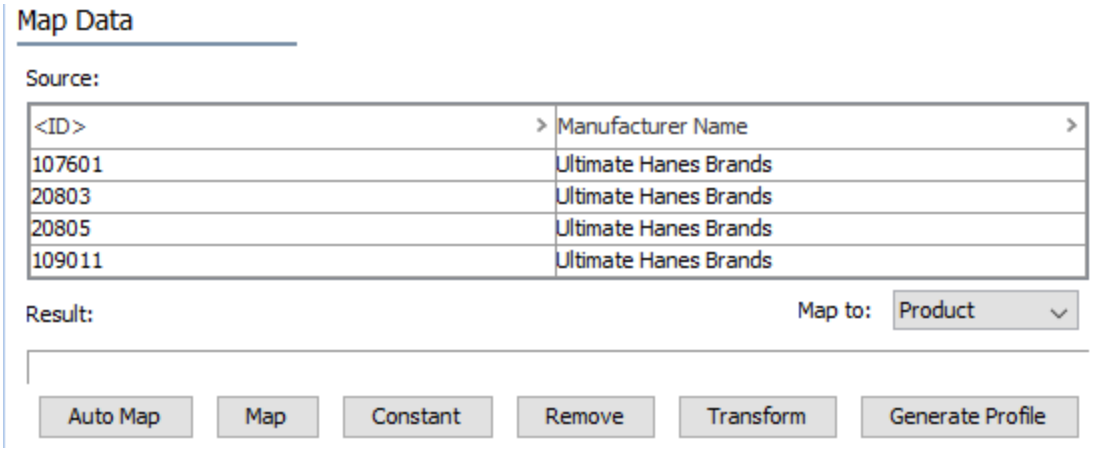

Import Manager
Steps
1. Select Configuration
2. Select Data Source
3. Select Format
4. Map Data
5. Identify Objects
6. Identify Destination
7. Select Business Rules
8. Advanced Settings
Map Data
Source:
<ID>
<Name>
Product Image ...
Product to Prod...
Acessories
154004
Bose Soundlink Mi...
BTSImage2;BTCI...
154009
Stand;Extension;S...
Result:
Map to:
Product
ProductImage AssetReference=Product Image Asset Reference ID
BTSImage2|BTCImage1
Auto ...
Map
Constant
Remove
Transform
Generate Profile
Back
Next
Finish
Cancel

Replace Substrings of the Value Example

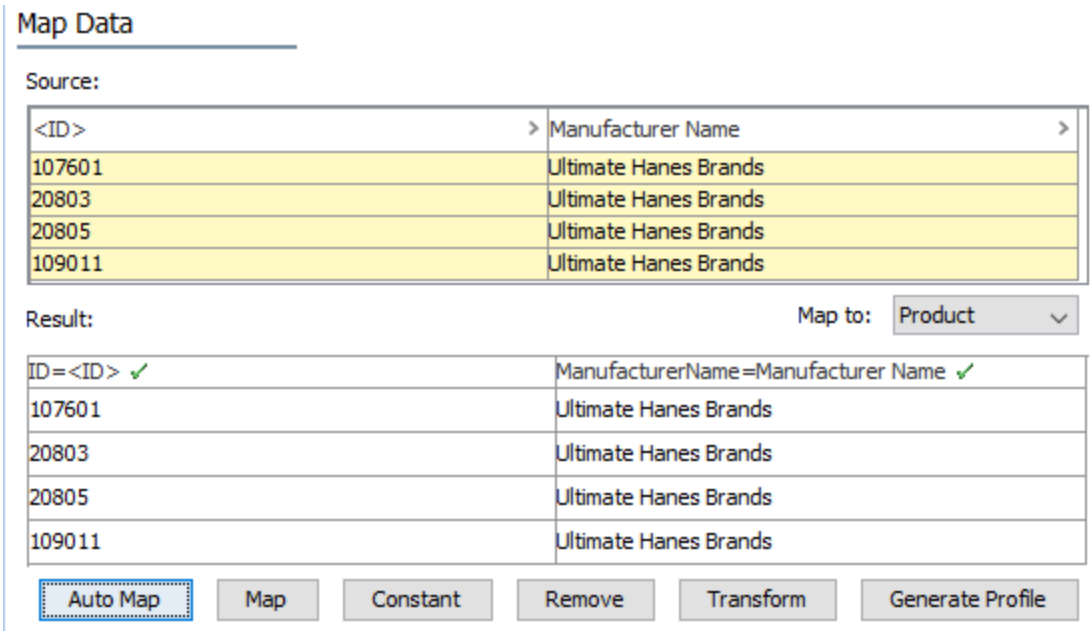
Transformations can improve the quality of the data you want to import. For general information about transformations, refer to the **Inbound Map Data - Transform** section.

The **Replace substrings of the value** transformation can be used to find and replace all matches of a specified text within a value.

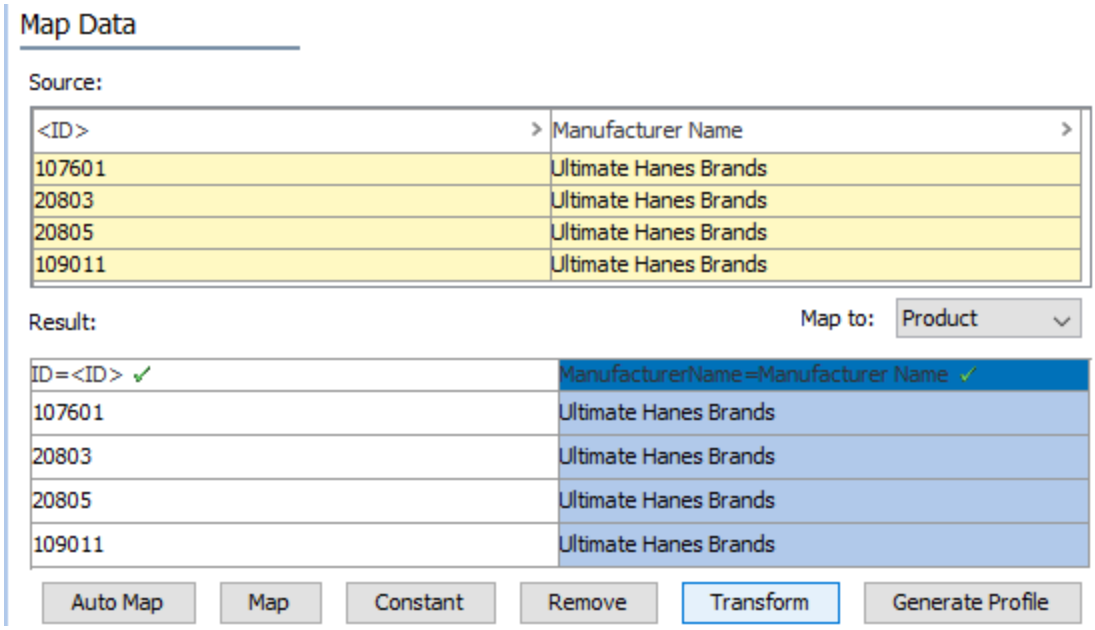
This example describes how text within a field in an import file is modified to be loaded into an attribute in STEP. The following data is being imported, and contains outdated data for the 'Manufacturer Name' attribute:



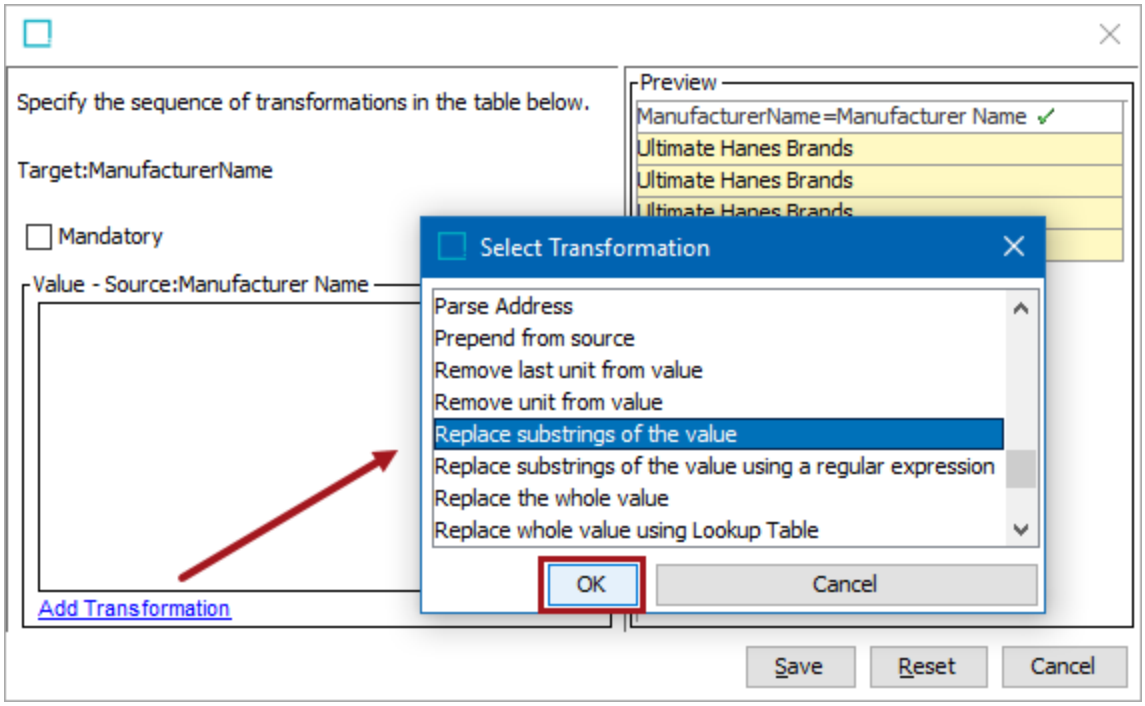
1. Load the import file into the inbound tool. For more information, refer to the **Data Exchange** topic.
2. Map the object ID in the import tool. Refer to **ID or Key - Map Inbound** for details.
3. Map the next source column to the required attribute. In this example, we map 'Manufacturer Name' to the Manufacturer Name attribute.



4. Select the result column that contains data to be modified and click the **Transform** button.

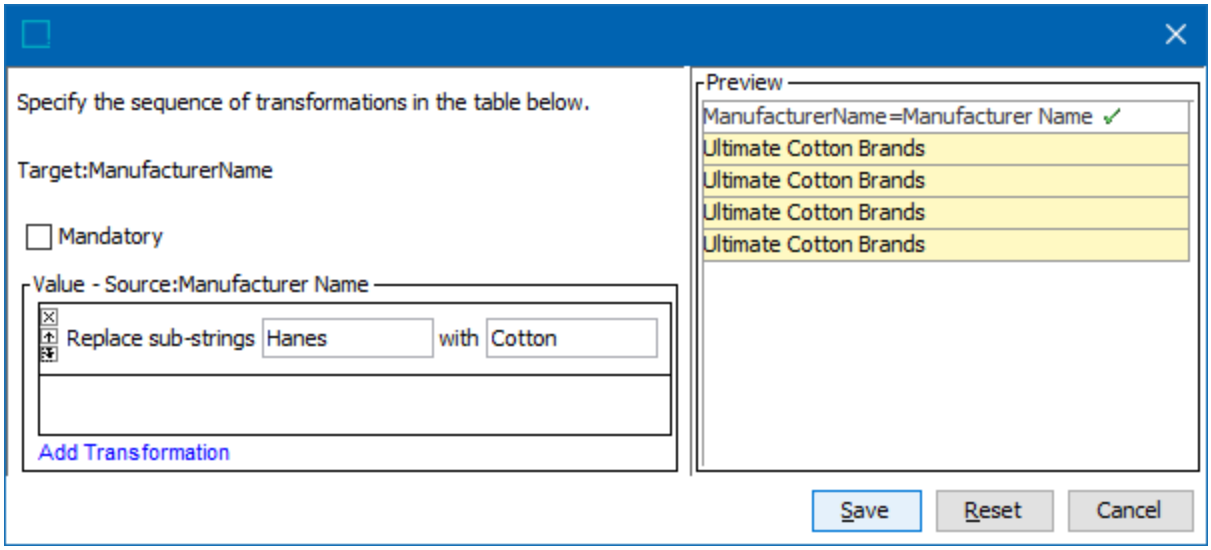


5. In the Transformation window, click on the **Add Transformation** link, select the **Replace substrings of the value** transformation, and click **OK**.



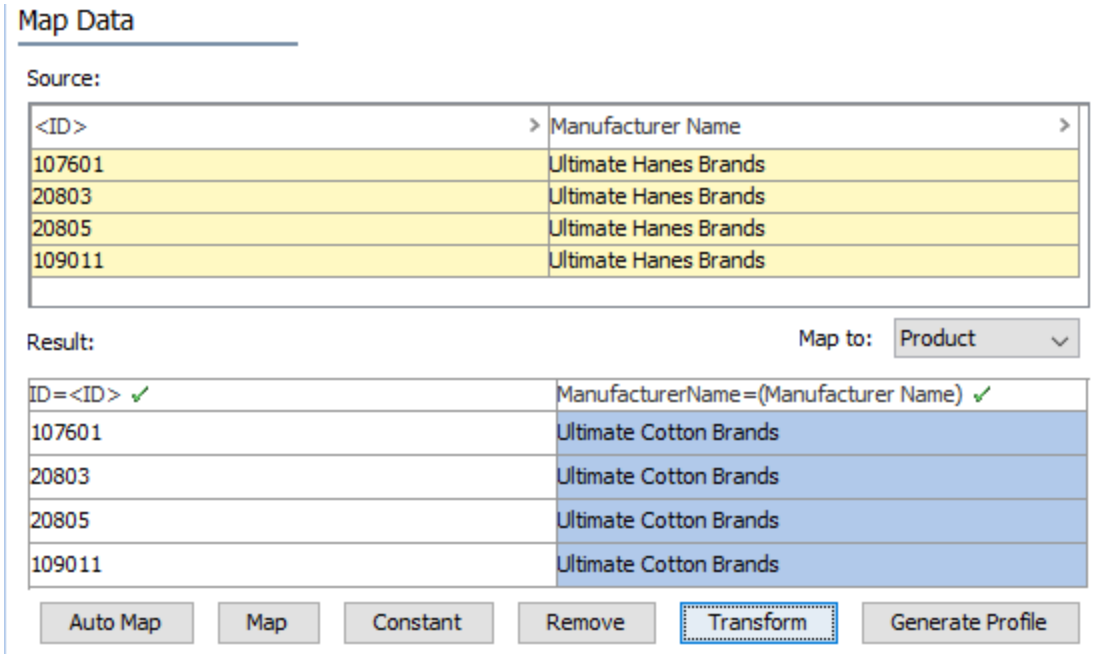
6. In the **Replace sub-strings** parameter, add the text that should be removed.
7. In the **with** parameter, add the replacement text. In this example, the word 'Hanes' will be replaced with the word 'Cotton.' If multiple instances of the word 'Hanes' existed, all would be replaced.

Click or tab out of the parameter to complete the configuration. The Preview pane shows the results of the transformation.



Important: Transformations are evaluated from the top down. Carefully consider the order of the transformations applied. To remove or rearrange transformations, use the Delete, Up arrow, and Down arrow buttons .

8. If necessary, check the 'Mandatory' option to skip objects with empty values during import.
9. Click **Save** to display the transformed data in the Map Data step.



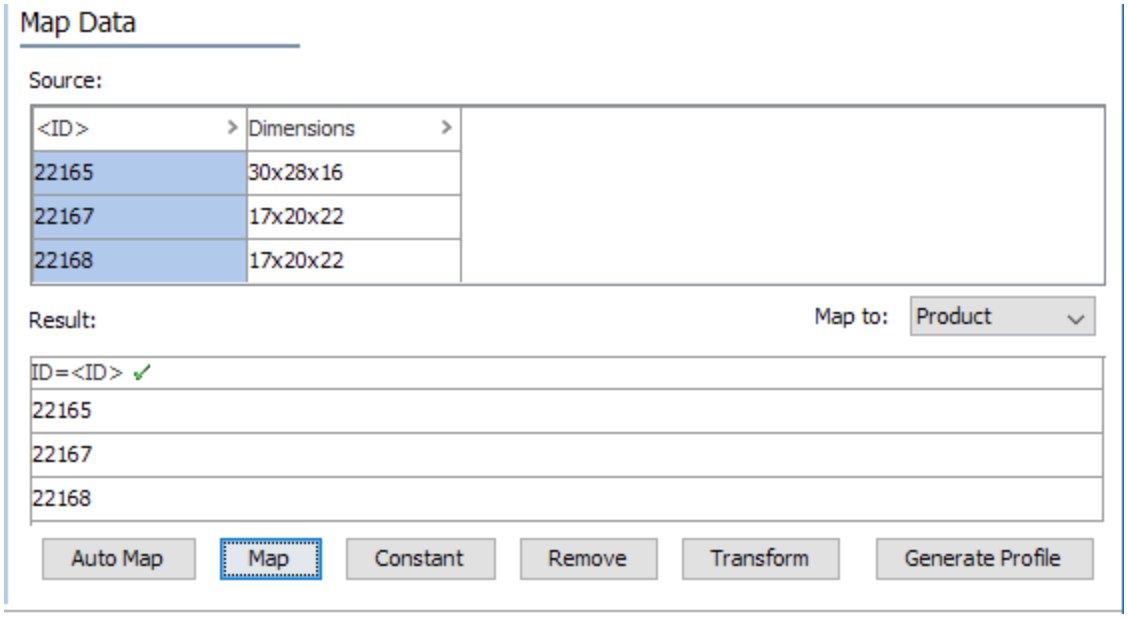
Another example of a Replace Substrings of the Value transformation is transforming an attribute that has a mixture of text and one or more images in the same attribute. This information is touched on in the **Attribute Transformation in Print Publisher** topic in the **Publisher (Adobe InDesign Integration)** documentation.

Split and Extract Data Example

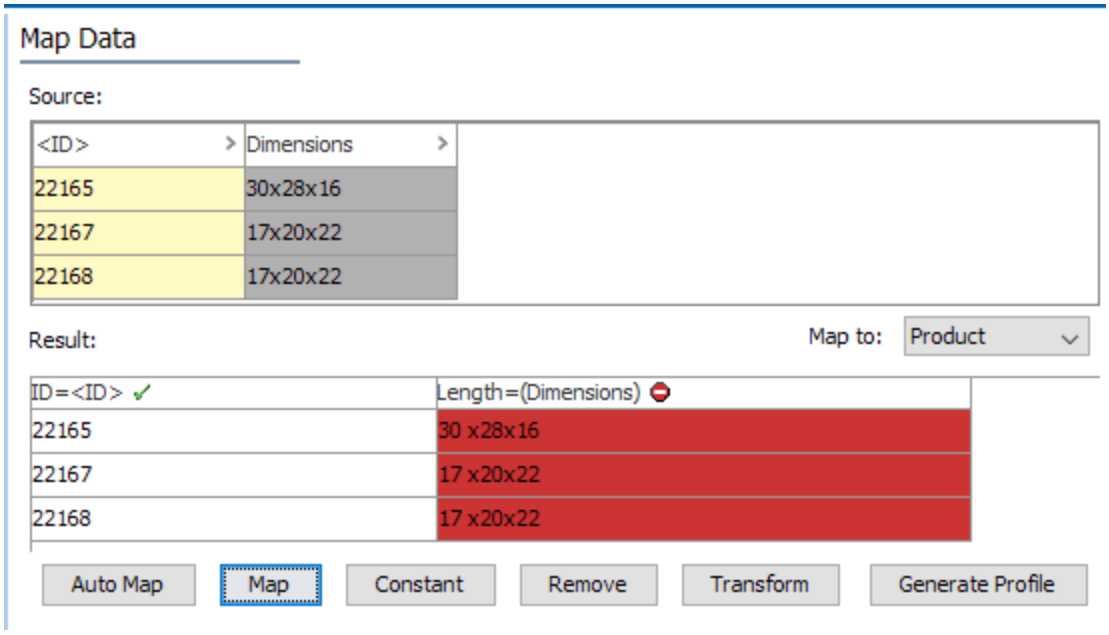
Transformations can improve the quality of the data you want to import. For general information about transformations, refer to the **Inbound Map Data - Transform** section.

In this example, you have a file that has dimensional data in one of the columns. Now, you want to split that dimensional information into three separate values for loading into STEP.

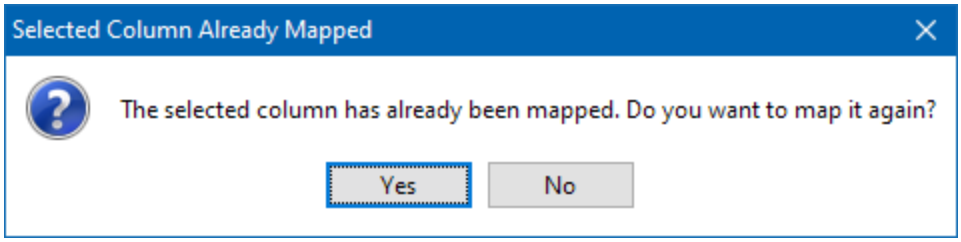
1. Map the product ID in the import tool. Then select the column that holds the dimensional data and begin to map it to the desired three separate attributes in STEP.



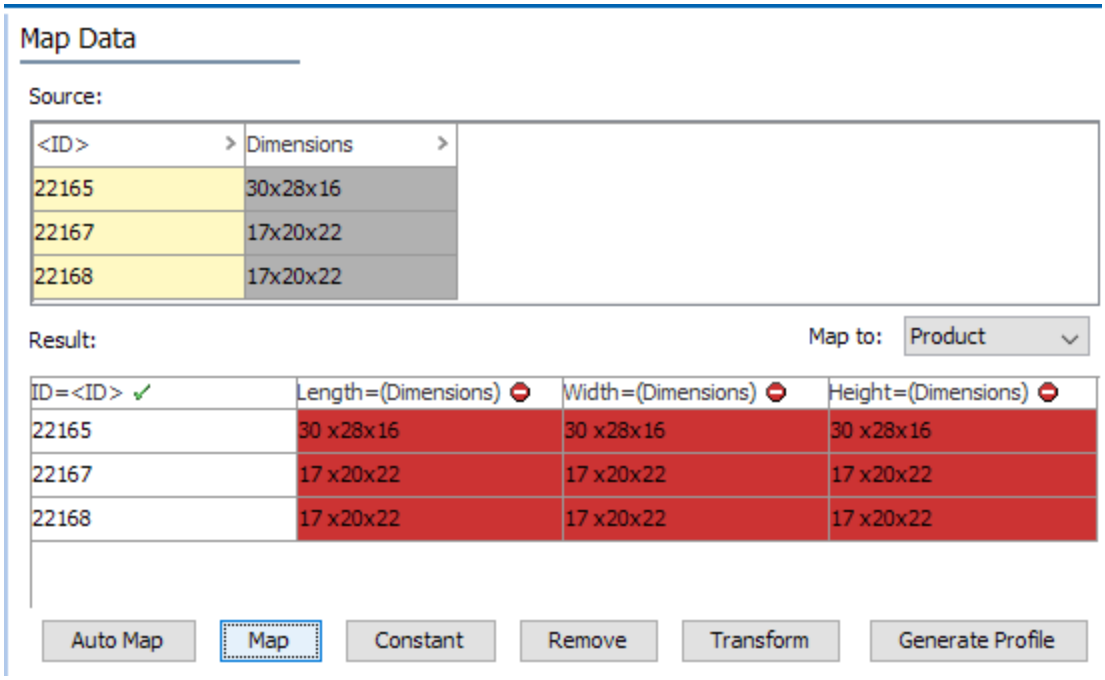
2. Manually map the column to the first attribute that will be populated with some part of the data. In this case, the fist group of numbers are mapped to the attribute 'Length.' Notice that the Results field shows it in red- this is expected until proper transformation is set.



3. Map the column that holds the dimension data again to the next two attributes that are needed. Each time a warning message is displays saying that the source column has already been mapped and asks if you want to map it again. Click **Yes**.

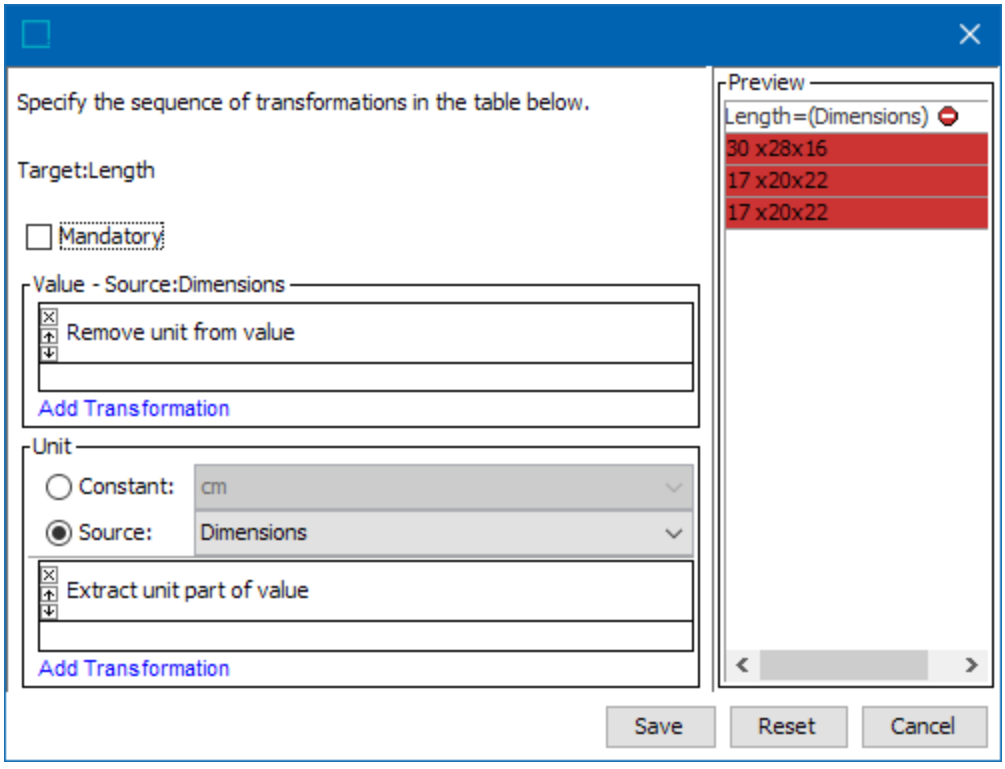


In this example, the same column has been mapped to three different attributes: Length, Width, and Height.

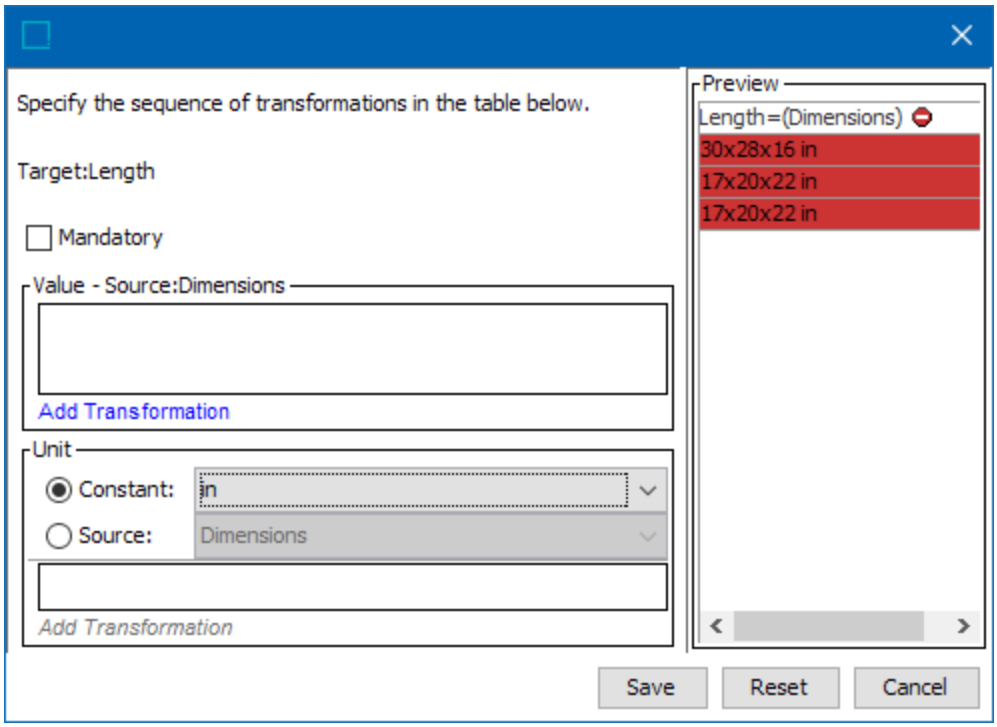


Note that in this particular case, each of the three attributes mapped to have a default unit in centimeters, 'cm.'

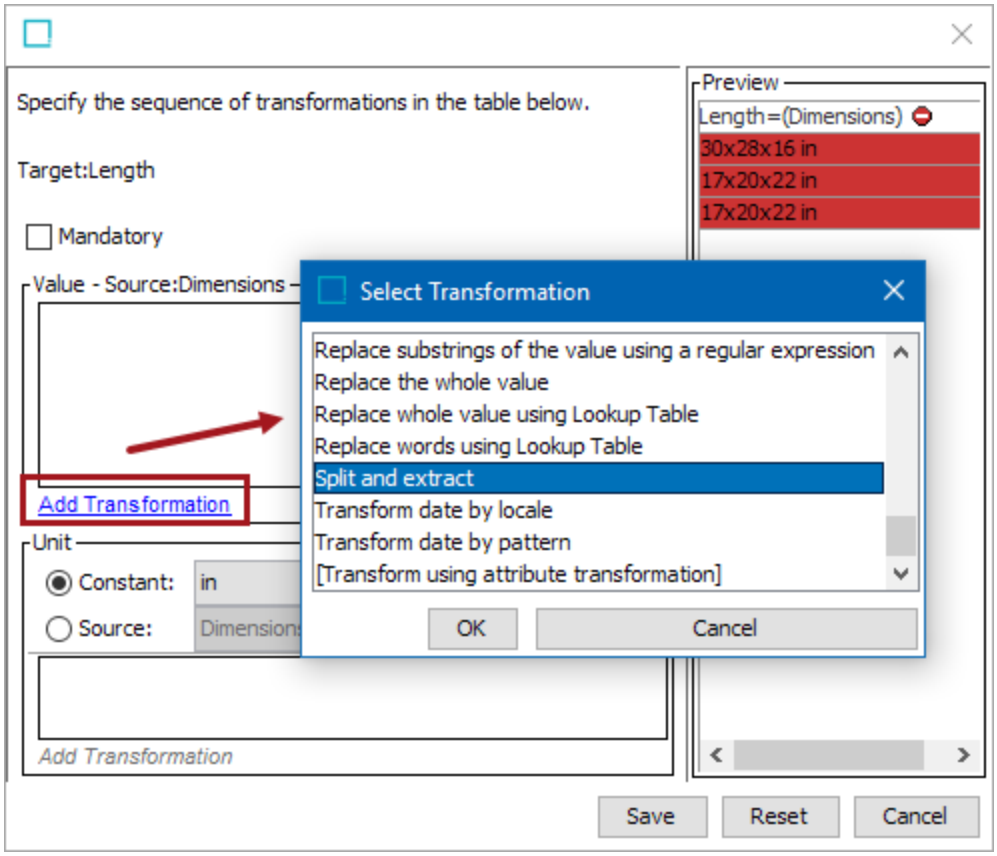
4. In the **Results** pane, select the first attribute that you want to perform a transformation on. In this example that is 'Length.' Click **Transform**. The following window appears.



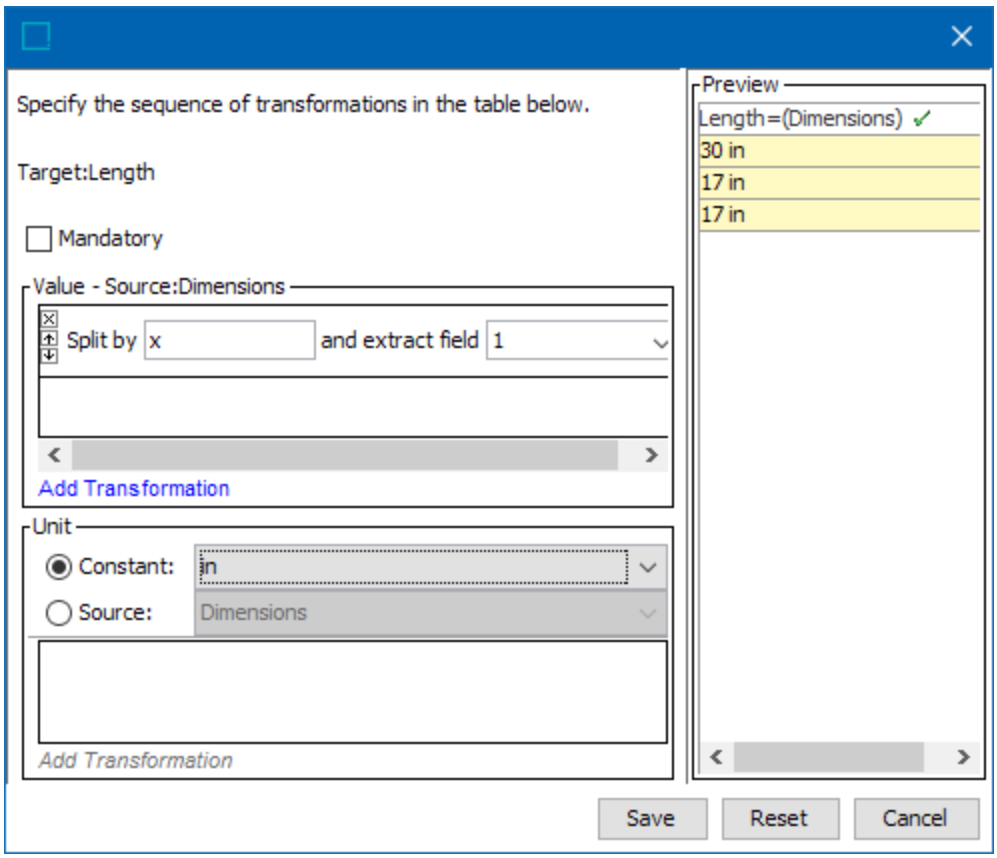
5. In the Unit field, select **Constant**, and choose the desired measurement from the dropdown if there are choices. In this case inches, 'in' are chosen. In the Value-Source: Dimensions field, remove the transformation **Remove unit from value**.



6. Under the Value-Source: Dimensions field, click **Add Transformation**. A list of available transformation functions appears. Select the **Split and extract** transformation.

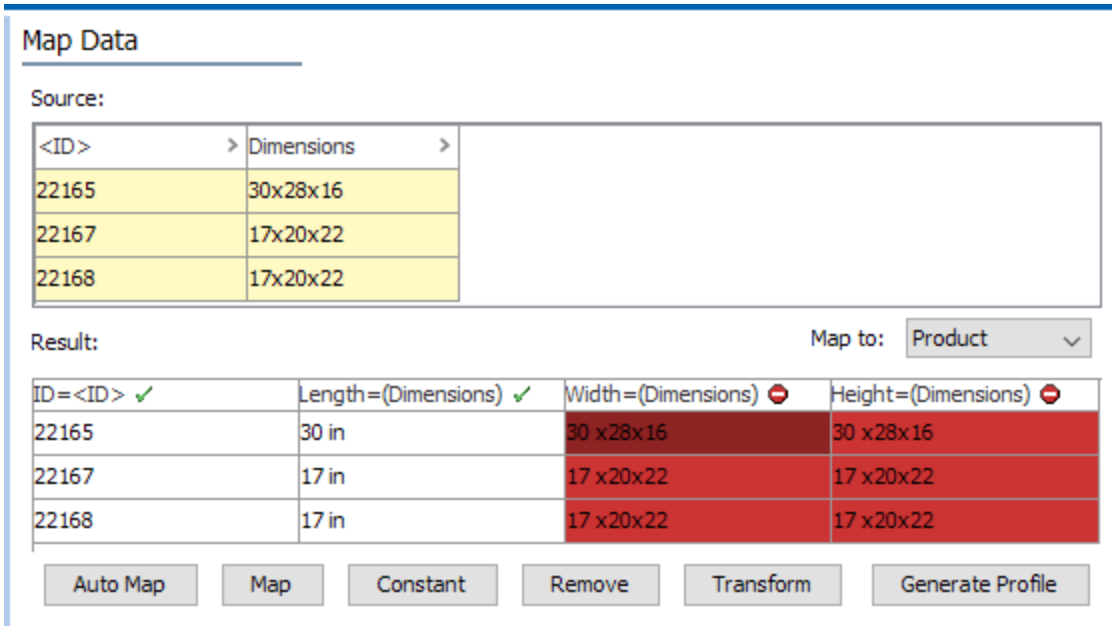


7. In the **Split and extract** transformation, specify how the data in the original file's dimensional data column should be split. In this example, you want to split the data by looking at the information between the " x " (include the spaces), and extracting the data before the first occurrence of this string. Select the dropdown list in the extract field and choose '1.' This represents the first group of numbers in the series. Choose 2 to represent the second group of numbers, and so forth. (The maximum extract field is 10). The right panel gives a preview of the data that is extracted. If the preview is not displayed, click F2. Click **Save**.

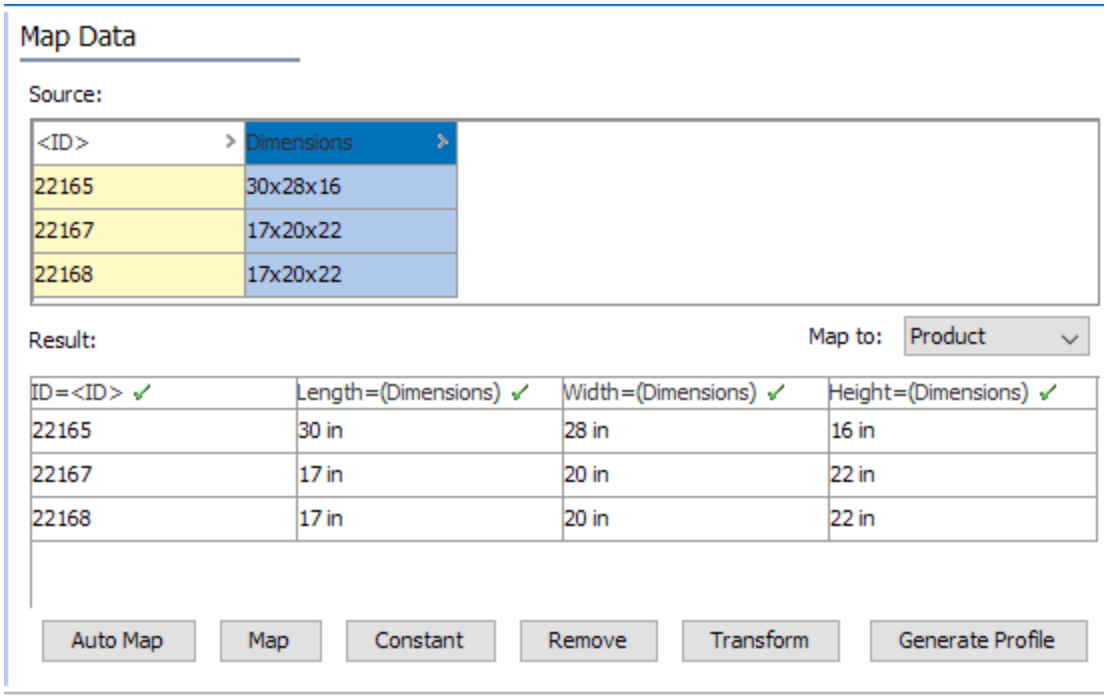


Note: If the 'Mandatory' option is checked, it will skip the empty values, and those objects will not be updated or imported into STEP.

8. In the **Map Data** screen, the column for Length shows a preview of the data that is loaded into STEP.



9. Repeat the transformation steps for the other two needed attributes, in this case 'Width' and 'Height.'



Replace the Whole Value Example

Transformations can improve the quality of the data you want to import. For general information about transformations, refer to the **Inbound Map Data - Transform** section.

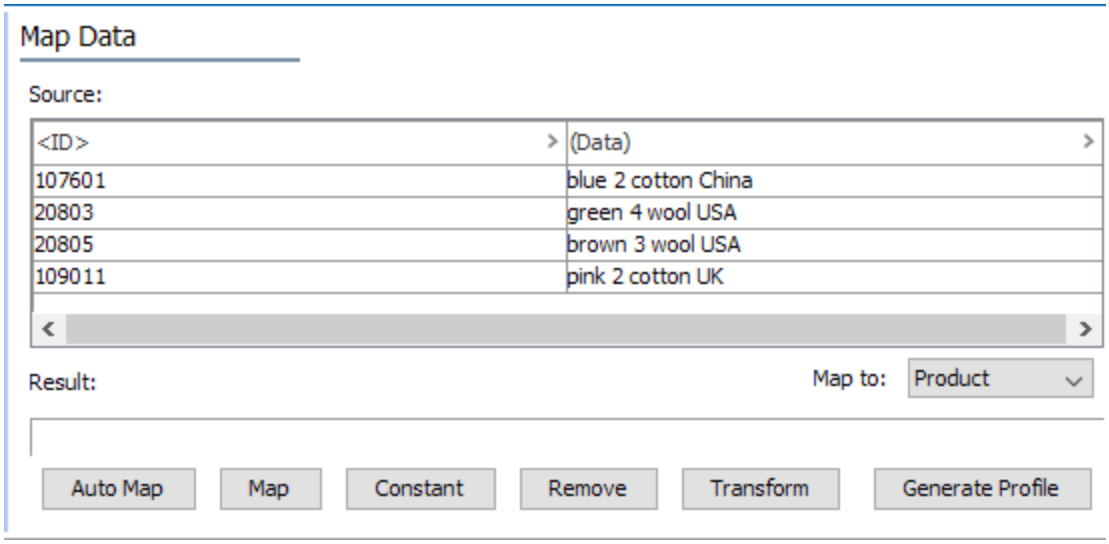
This example describes how to extract and clean information from a legacy system, or groups of information stored in one column, and load it into separate attributes in STEP. If information is more technical or cryptic, refer to the last section of this topic on how to convert cryptic data.

This example includes the following transformations:

- Extract characters before a given text in the value (Extract Characters before)
- Change case
- Split and extract
- Replace the whole value

Data being transformed

Note: The preview displays how the transformations function on a sample of data, typically the first 20 rows. Prior to starting the import, verify that the first 20 objects contain the entries with transformations you want to verify.

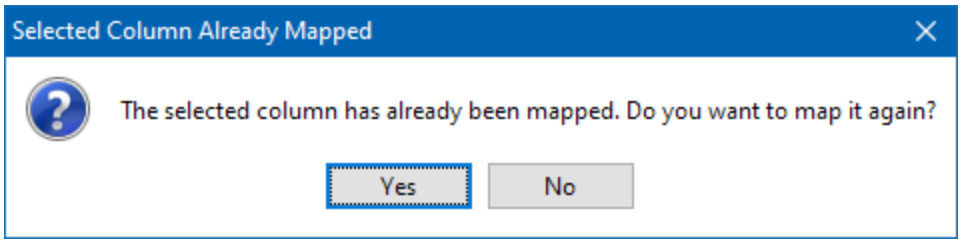


A single import column contains four pieces of data that are to be stored separately in STEP. Using transformations, separate values can be created for the attributes. In this case it is: Primary Color, Hat Size, Material, and Country of Origin. For the first source row, the object should have Color = blue, Hat Size = 2, Material = Cotton, and Country of Origin = CN for China.

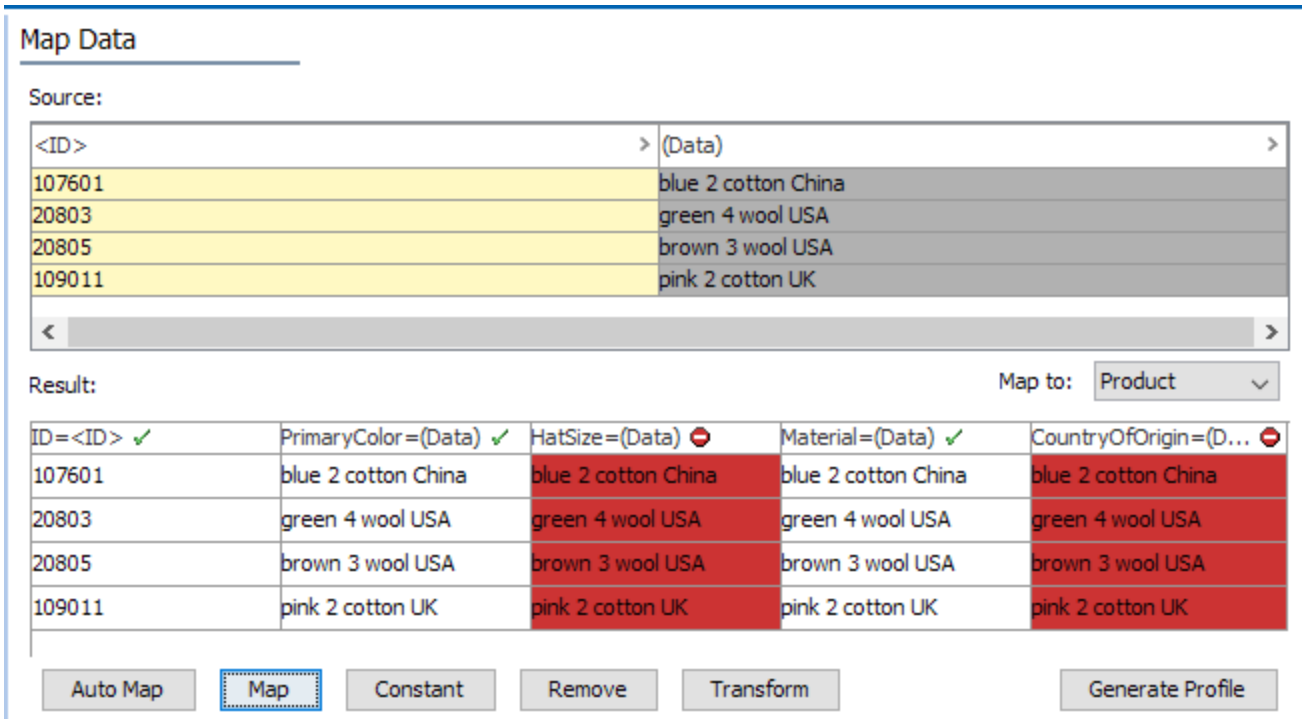
1. In the 'Map Data' step of an import wizard, map the object's ID. Then, map the column that holds the data to the four needed attributes. This means that the same column will be mapped 4 times. For more on how to

map data, Refer to the **Import Manager - Map Data** topic or the **IIEP - Configure Processing Engine** topic in the **Data Exchange** documentation.

2. A warning message will display saying that the source column you are trying to map has already been mapped. Click 'Yes' to continue mapping to the needed attributes.



3. When finished with mapping, the Results field should look similar to the image below. Some columns may or may not be red depending on the attributes mapped and the validation base type for those attributes. Each column now needs to be transformed.

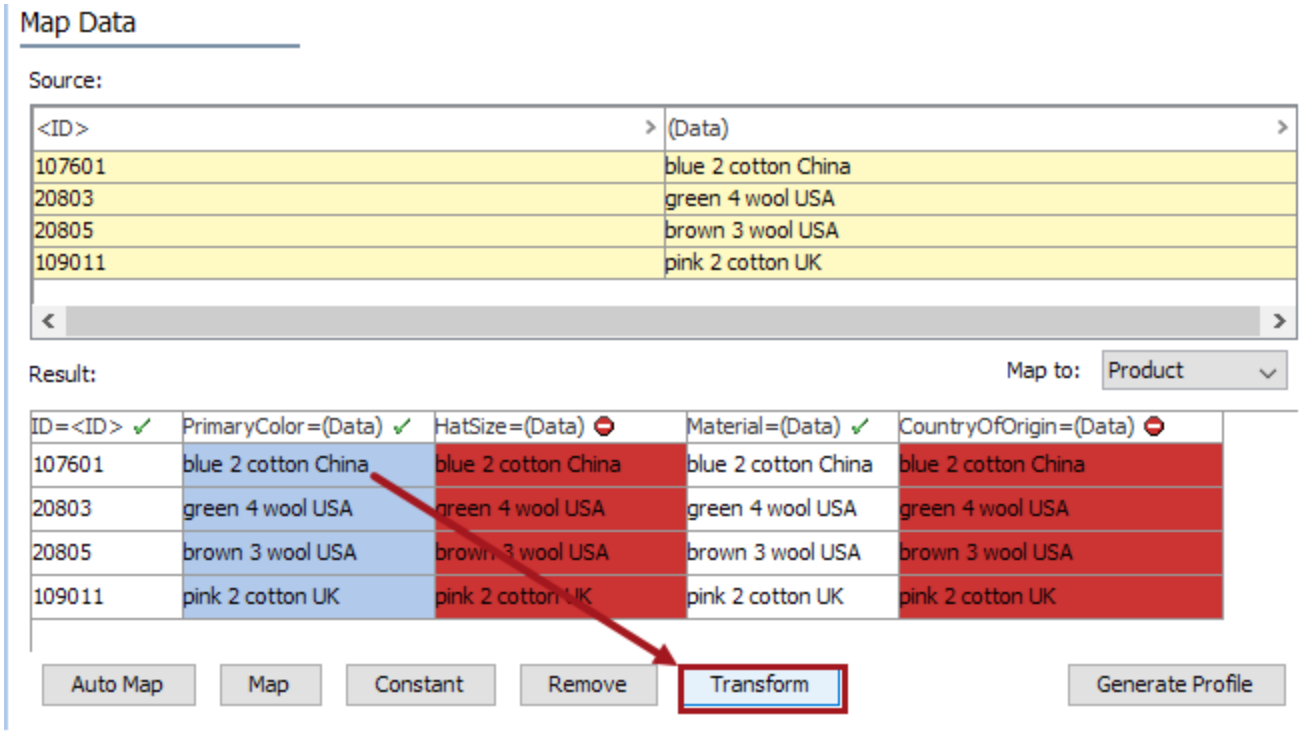


Applying transformations

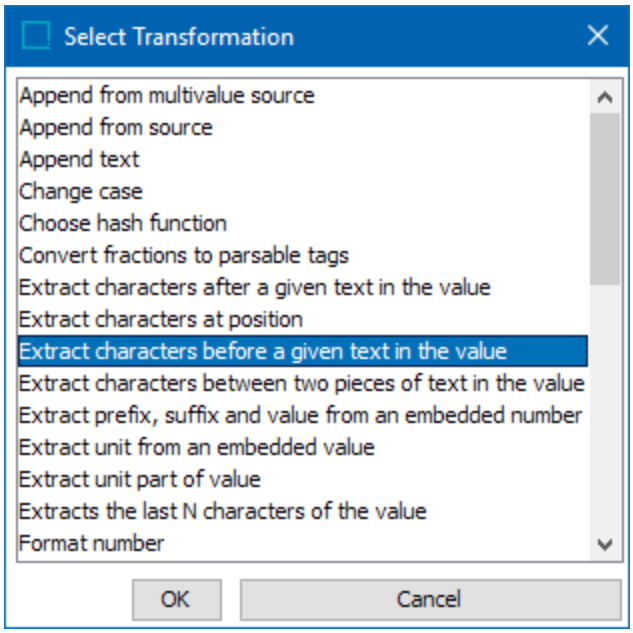
Below are examples of how transformations can be used for each of the columns to clarify the data.

For more on all available transformations, Refer to the **Transformations** topic in the **Resource Materials** online help.

1. Select the first column that needs to be transformed, in this case it is 'PrimaryColor=(Data)', in the Results field, and click **Transform**.

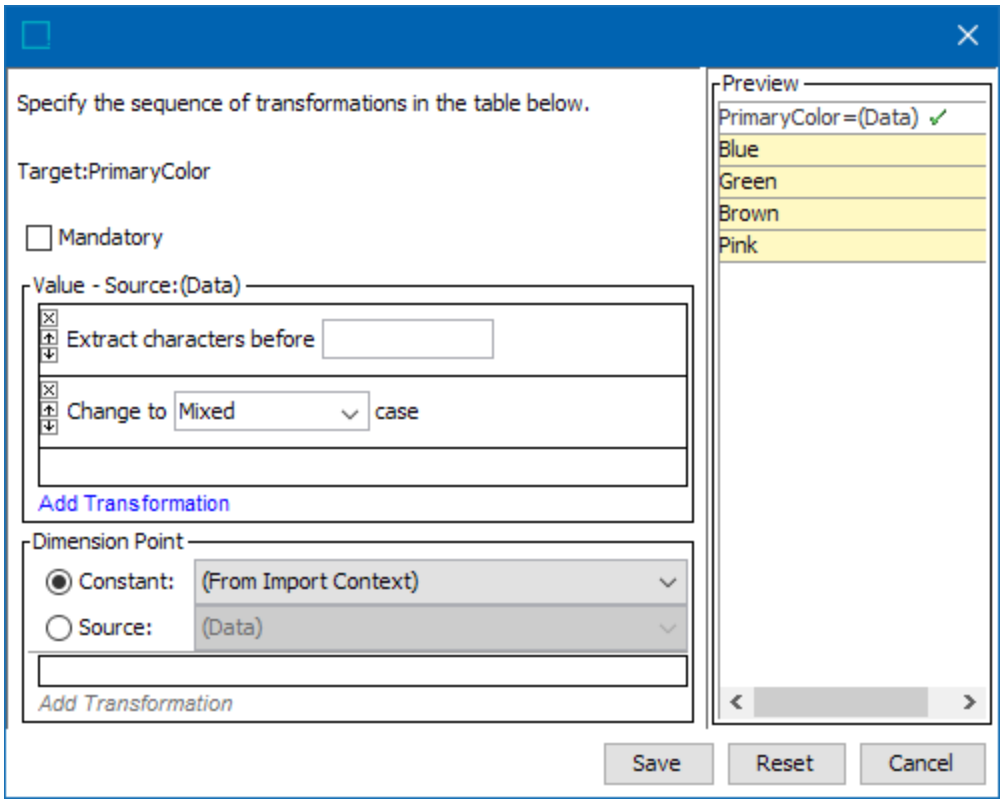


2. In the Transformation window, click on the blue **Add Transformation** link, to open the list of available transformations.



3. Choose the needed transformation and enter in any necessary data. For example, two transformations were added for 'PrimaryColor=(Data).' The first one, 'Extract characters before a given text in the value', shows up in the transformation table as 'Extract Characters before.' A space is put in the text field, though it is not visible

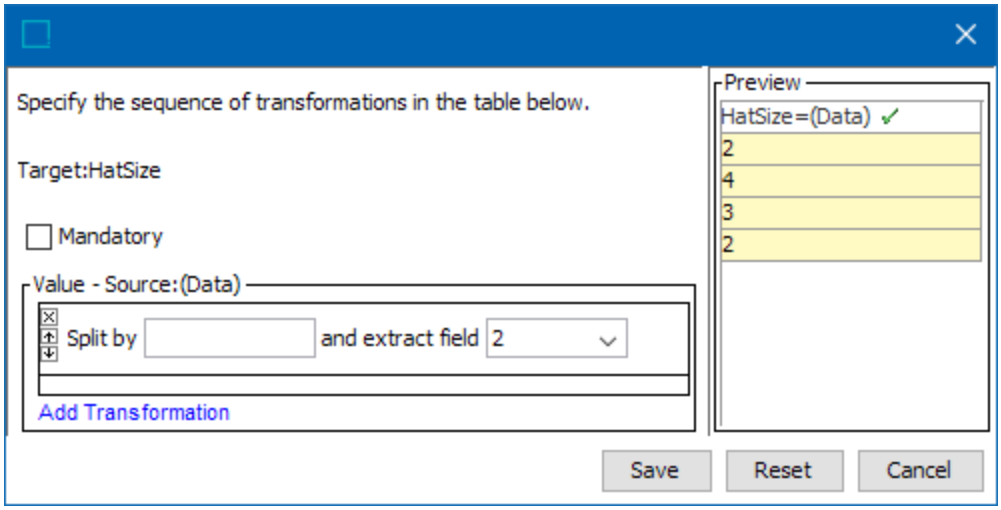
in the picture. Also, the 'Change case' transformation was added. 'Mixed' was chosen from the dropdown so that the first letter of the colors are capitalized and the rest appear lower case.



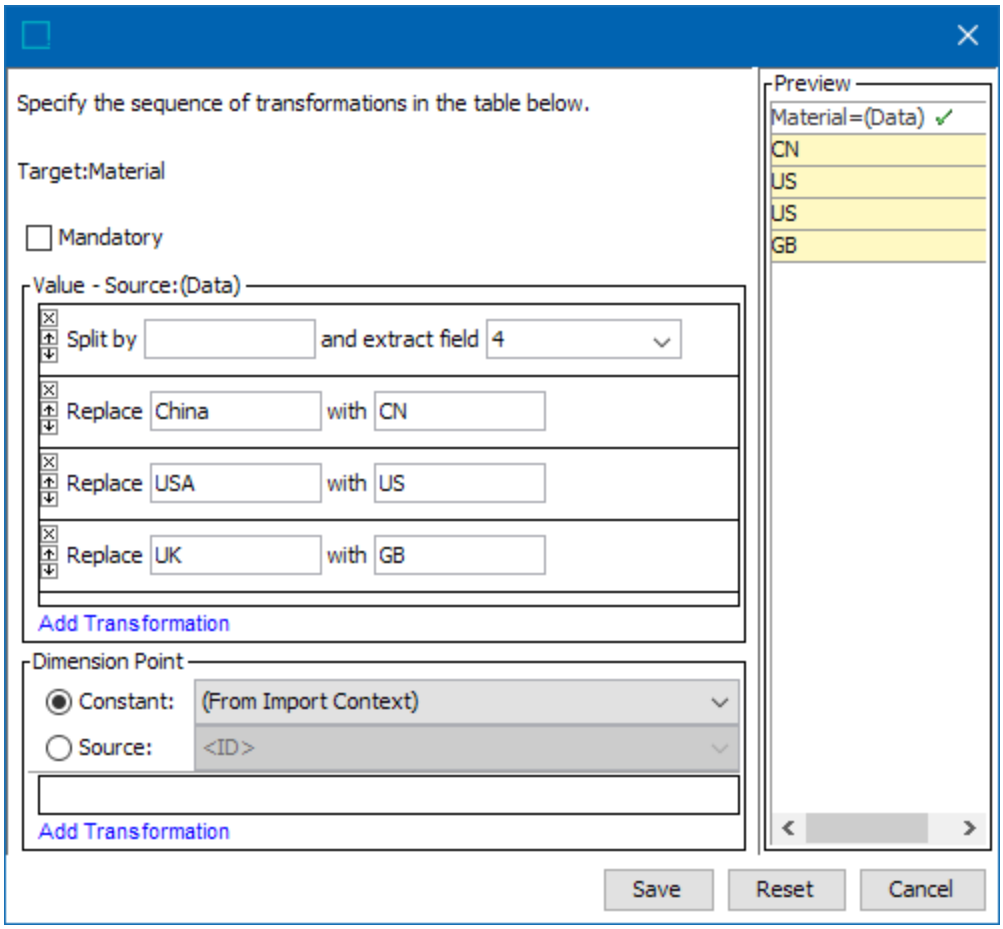
It is important to note that transitions are evaluated from the top down. Careful consideration needs to go into the order of the transformations applied. Each time a new transformation needs to be applied, **Add Transformation** needs to be clicked.

Note: If the 'Mandatory' option is checked, it will skip the empty values, and those objects will not be imported.

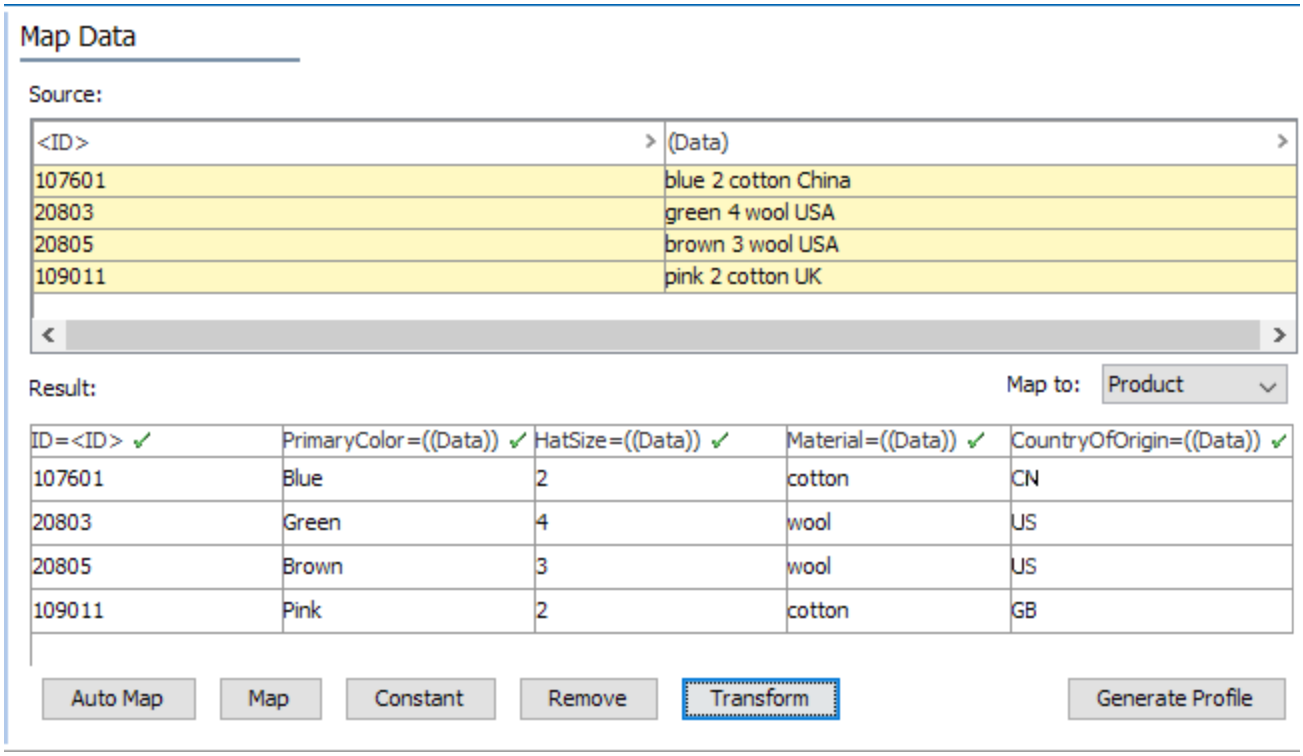
4. Continue with transforming the next column. For example, in the 'HatSize=(Data)' column, the 'Split and extract' transformation can be selected to show the correct numbers. In the first field, a space, which is not included in the picture, was added. In the second field, the number 2 was selected from the dropdown menu. This is because the second grouping of data is what is desired to put into this field.



5. In the third column, again select the 'Split and extract' transformation and select to extract field 3.
6. For the last column, select the 'Split and extract' transformation and select to extract field 4. Then select the 'Replace whole value with' transformation to change the countries to the needed abbreviations for STEP. You must select this for each country type.



7. Once each column is properly transformed, the results field should now look accurate, and it is ready to go on to the next steps for import.



Transform Date by Locale or by Pattern

Attributes with a Validation Base Type of Date, 'ISO Date', and 'ISO Date and Time' can be transformed by locale (using the 'Transform date by locale' option) or by pattern (using the 'Transform date by pattern' option).

While it is recommended that users do STEP search, compare, and order operations with ISO date attributes, users may want to transform date attributes by locale or another recognizable format to meet a business requirement. Using the locale and pattern transformations does not impact the attribute Validation Base Type.

Transform date by locale

Dates converted to locale will follow the DateFormat.SHORT Java formatting style. Times converted to locale will be the DateFormat.MEDIUM formatting style. This locale information is based on the context being used in workbench at the time of export or import.

Date and Time (DateFormat) Samples	English (United States)	Danish (Denmark)	French (France)
Date (SHORT)	8/14/15	14-08-15	14/08/15
Time (MEDIUM)	11:25:40 AM	11:25:40	11:25:40

Transform date by pattern

A few examples are shown in the following table. If time zone is included in a pattern used to export data, the time zone abbreviation is determined by the location of the STEP server.

Date (and Time) Pattern Examples	United States Locale Example
yyyy.MM.dd G 'at' HH:mm:ss z	2015.08.14 AD at 11:25:40 EDT
EEE, MMM d, "yy	Fri, Aug 14, '15
dd.MM.yy	14.08.15

Step-by-step instructions for export and import transformations are included below.

Considerations and Limitations

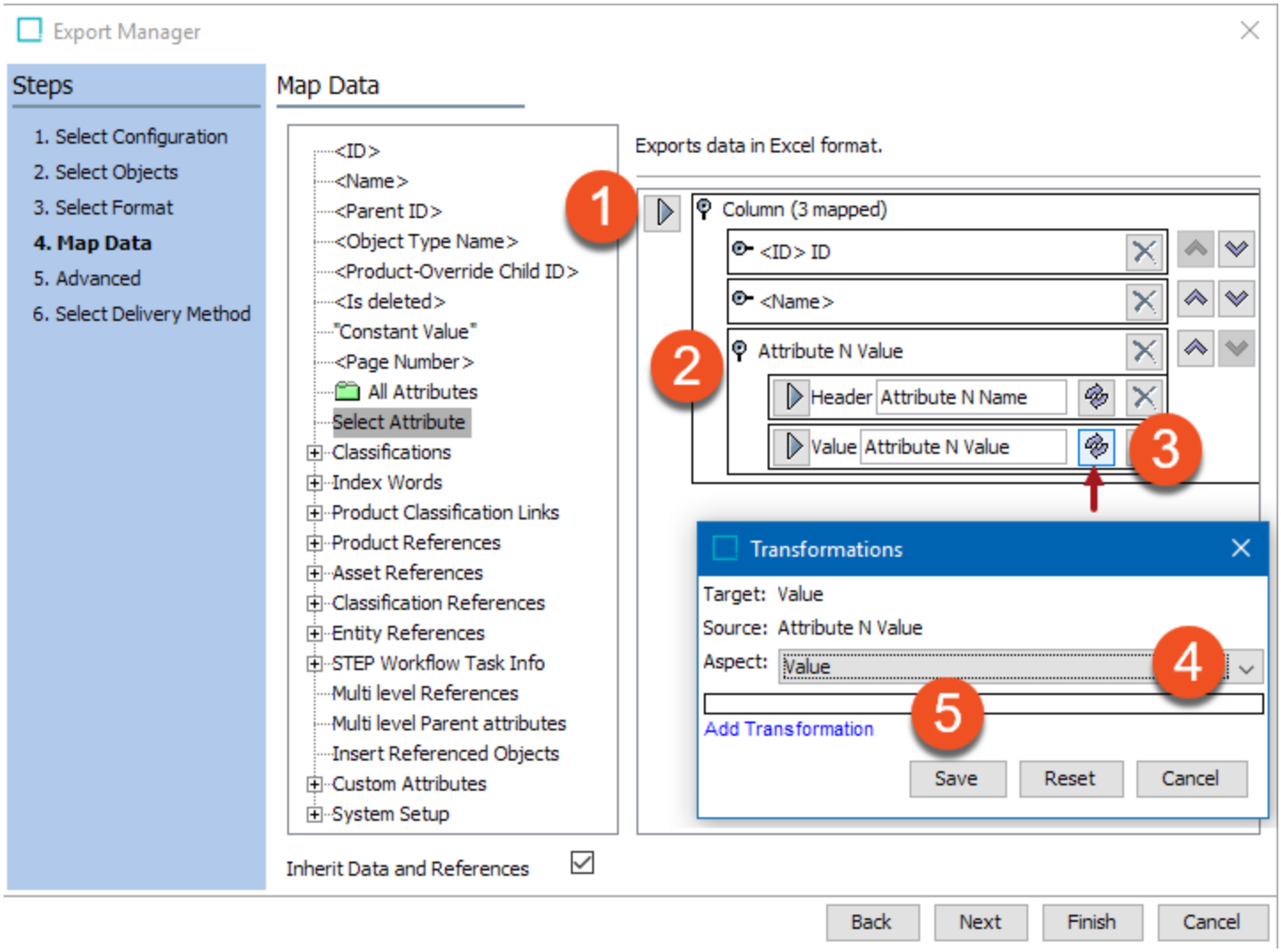
- Date localization and transformation works on attributes with these validation base types: Date (DD-MON-YYYY), 'ISO Date' (YYYY-MM-DD), and 'ISO Date and Time' (YYYY-MM-DD HH24:MI:SS).
- The steps for import and export must be followed carefully or the process may fail.

For information regarding converting attributes from Date type to 'ISO Date' type, refer to the **Maintaining Attributes** section of the **System Setup** documentation.

Export Transformations

The steps for exporting data using date transformations should be followed very carefully. If not, the export may fail and the entire process will need to be restarted.

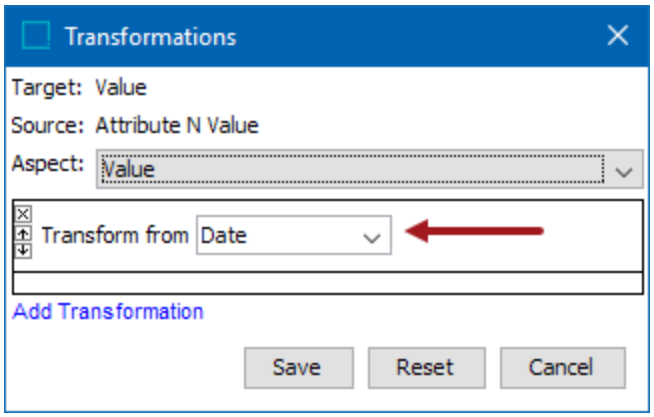
1. Verify that the STEP Workbench Context is set correctly. This will impact locale transformations.
2. Choose a way to begin the export process using the wizard as described in **Creating a Data Export** and select the objects if necessary.
3. Select a format and continue to the Map Data step. Transformations are set up in Map Data step. If your selected format does not require mapping, this step is disabled. For details, refer to **Export Manager - Map Data**.
4. Begin by mapping the attributes to be exported. Pick the attributes from the tree and click the mapping icon.
5. Click the flipper to expand the options for the attribute to transform. For 'Attribute N' values, a Header row and a Value row displays.
6. Click the transformations button to the right of the Value row.
7. A Transformations dialog box appears. Change the Aspect field to 'Value' using the dropdown.
8. Click the 'Add Transformation' link and make a 'Transform date' selection: Transform date by locale or Transform date by pattern. Follow the directions for the applicable selection and then continue on to the next step in these directions.



Transform date by locale

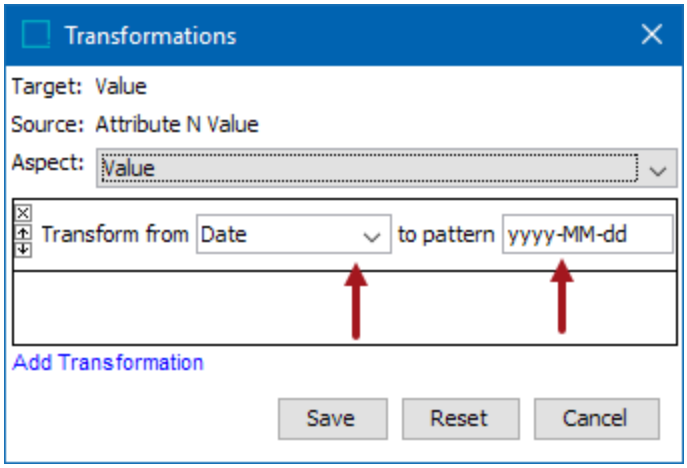
- Click on **Transform date by locale** and then click OK.
- From the Transformations screen, use the 'Transform from' dropdown to choose the validation base type

of the date attribute being exporting.



Transform date by pattern

- Click on **Transform date by pattern** and then click OK.
- From the Transformations screen, choose the validation base type of the date attribute you are exporting (Date, 'ISO Date', 'ISO Date and Time'). And, enter the desired pattern manually or copy and paste it from an external source. Click Save.



Once you have made a selection and picked the source date validation base type and pattern (if applicable), continue on with the rest of the steps.

9. Select Next to move to the Advanced step. For details, refer to **Export Manager - Advanced**. Make any additional selections, if desired. However, no additional configurations are required to complete the process, click Finish.
10. Optional: Make any Select Delivery Method selections and click Finish. For details, refer to **Export Manager - Select Delivery Method**.
11. Make selections on the Save Export Configuration screen, if desired. Click OK. For details, refer to **Running a Data Export**.
12. The 'Starting process Exporting' dialog will appear. For details, refer to **Monitoring a Data Export**. Choose

'Go to process' to access the file.

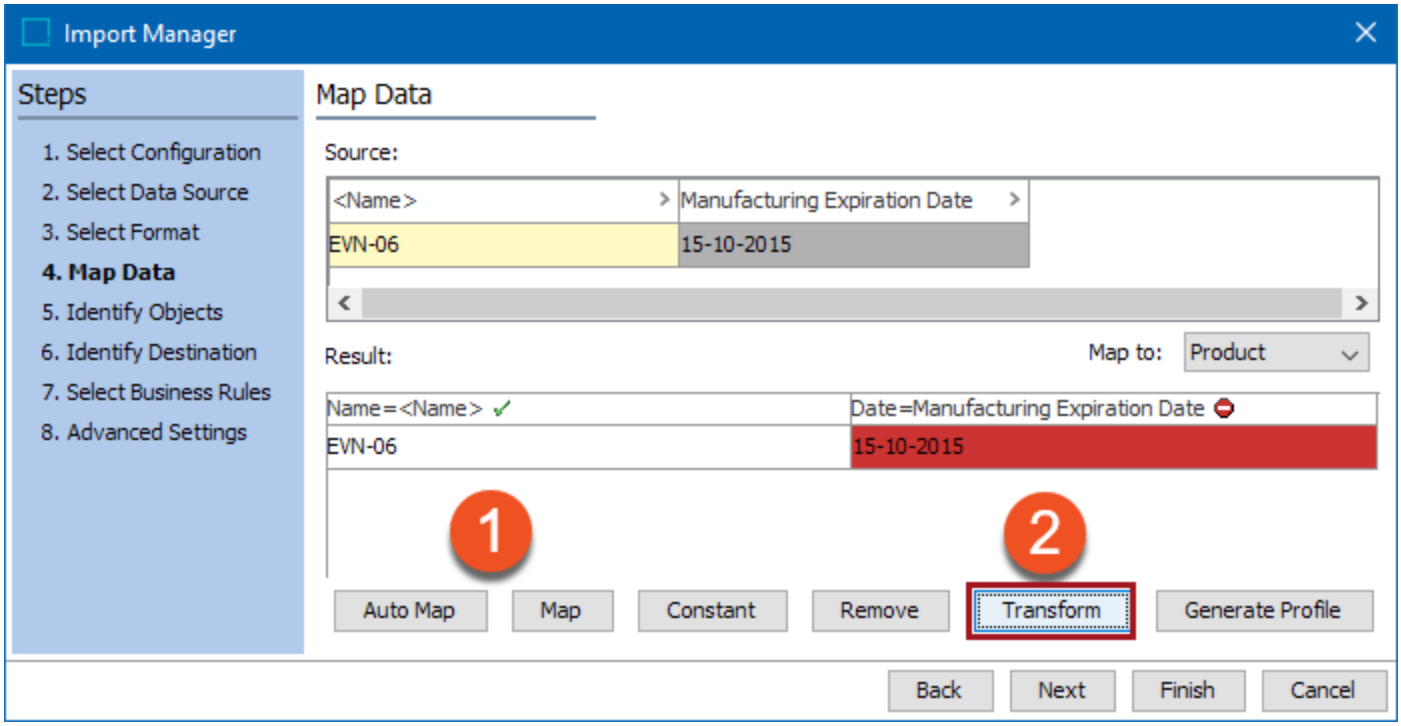
13. On the Background Process screen, determine if the export was successful or not.
    - If the process failed, go back and try the steps again (making sure the correct attribute validation base type was selected on the Transformations screen).
    - If the process was successful and the delivery method was set to file, an export file (expand the Result flipper, if needed) will be available at the bottom of the Background Process page. Double-click the file name to open the file, or click the Save icon to the right of the file name to save the file without immediately opening it.
14. Verify the exported data is correct.
    - If transforming by locale, the correct locale format should be showing. The locale is determined by the context set in the workbench at the time the export process begins.
    - If doing a pattern transformation, verify the date format matches the pattern entered. If the date pattern includes a time zone, the time zone is determined by the location of your server. It is not impacted by context set in the workbench.

Import Transformations

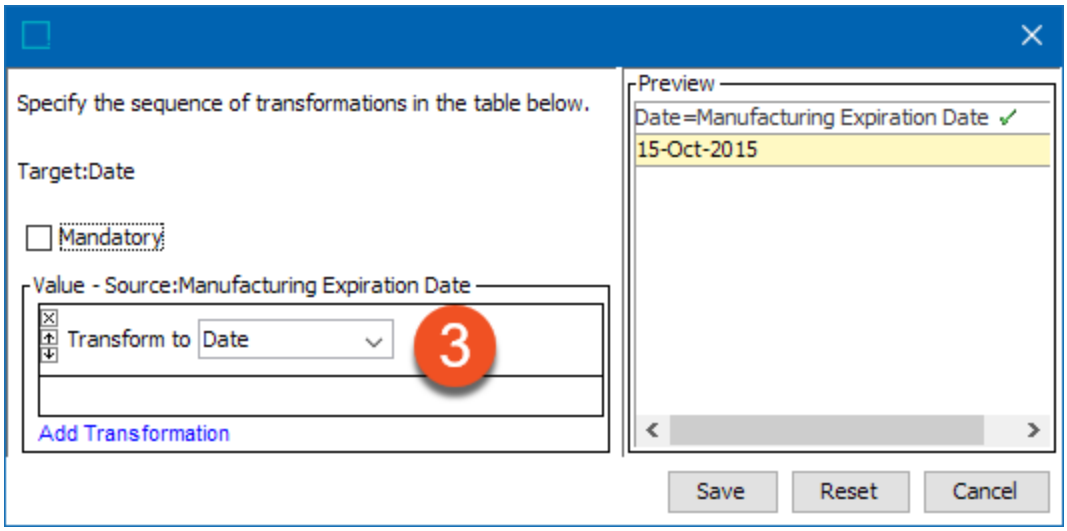
The steps for importing data using date transformations should be followed very carefully. If not, the import may fail and the entire process will need to be restarted.

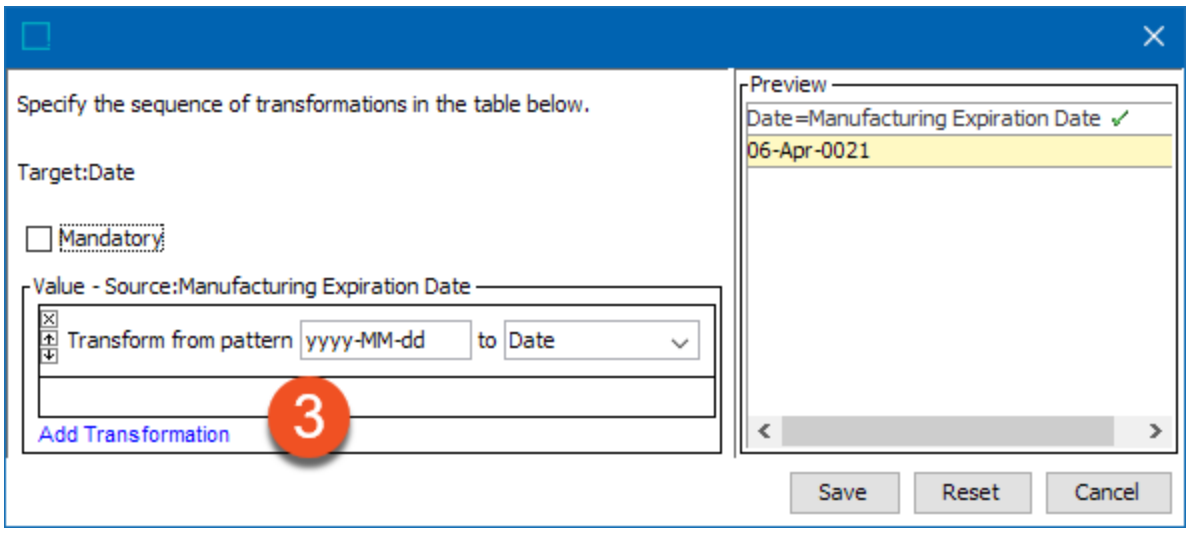
Important: Begin by making sure the date fields in the input file do not have any custom formatting. The best results will occur when importing dates that are in text format. Users should convert the date fields to text format prior to starting the import process.

1. Choose a way to begin the import process using the wizard as described in **Creating a Data Import**.
2. In the Select Data Source step, link the file you are importing by searching or selecting or manually typing in (or pasting) the file location. For details, refer to **Import Manager - Select Data Source**.
3. Make a format selection in the Select Format step. For details, refer to **Import Manager - Select Format**.
4. Transformations are set up in Map Data step. If your selected format does not require mapping, this step is disabled. For details, refer to **Import Manager - Map Data**.
5. Begin data mapping by selecting Auto Map. If the date column is unmapped, then use Map to manually tie the attribute values to be imported to an existing attribute.
6. Under Result, select the column you want to transform during the import process. While the column is selected, click on the Transform button.



7. Click the blue 'Add Transformation' link in the bottom left of the transformations screen. Make a selection: Transform date by locale or Transform date by pattern. Follow the directions for the applicable selection and then continue on with the remaining steps.





Transform date by locale

- Click on **Transform date by locale** and then click OK.
- Choose the validation base type of the target date attribute (Date, 'ISO Date', 'ISO Date and Time'). Once you have picked the correct option, the Preview data will turn yellow. And, click Save.
- If you try all three selections and the Preview remains red, then you will need to use the Transform date by pattern option. To use the locale import option, the date values must match the locale format for the context that the workbench is currently set to.

Transform date by pattern

- Click on **Transform date by pattern** and then click OK.
- From the Transformations screen, enter the pattern of the date values you are importing. Then, tab over to or click on the dropdown for the validation base type of the target attribute. Make a selection from the dropdown or leave as-is. Click Save.
- The Preview section of the screen will turn yellow when the validation base type is correct. Before continuing make sure the Preview date is showing correctly and verify that the correct pattern is entered.

Once the transformation is set up and screen information is validated, continue on with the rest of the steps.

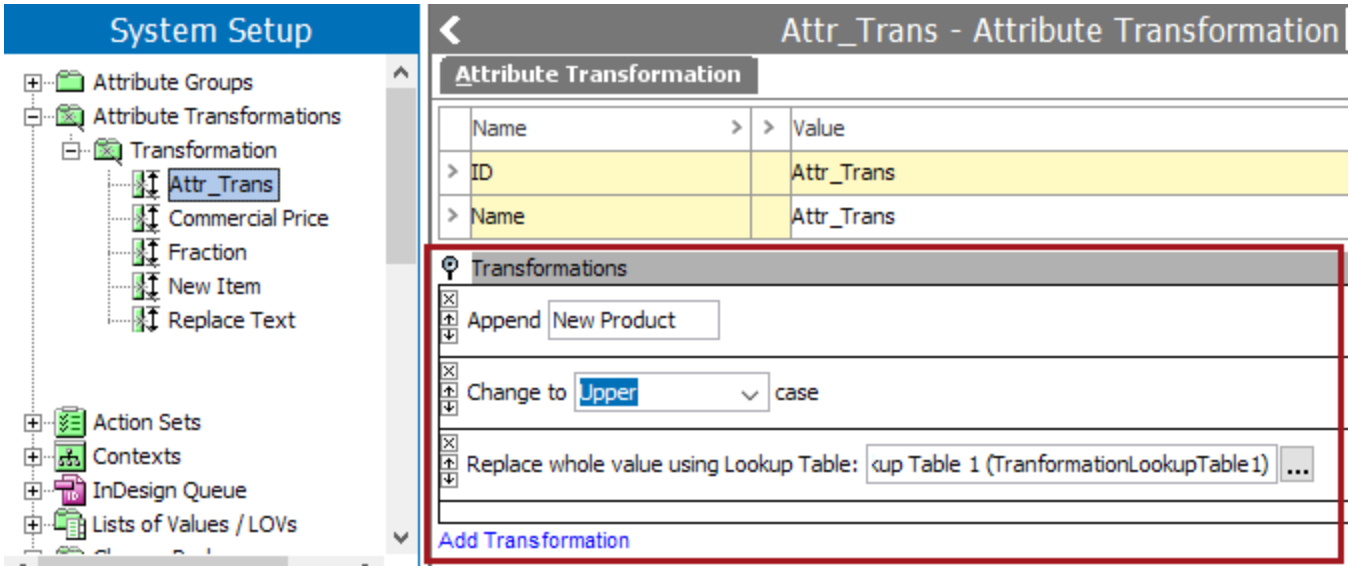
8. Select Next to move to the Identifying Objects step. For details, refer to **Import Manager - Identify Objects**. If you are importing an ID field with the objects, the data will be automatically matched up with existing data. If you are importing data without an ID field, click a Source column and then press the Match button to match the source data up with existing STEP data. If you skip this step, you will create new objects in the workbench instead of importing into existing ones. Click Next.
9. Optional: Make any **Identifying Destination**, **Select Business Rules**, and **Advanced Settings** selections. While selections are not required, it is recommended that the end user review each screen.
10. Click Finish when ready for the import process to begin.
11. Choose Start Import Process on the Save Import Configuration dialog box. For details, refer to **Running a Data Import**.
12. The 'Starting process Importing' dialog will appear. Choose 'Go to process.' For details, refer to **Monitoring a Data Import**.
13. On the Background Process screen, users can determine if the import was successful or not.
    - If the process was successful, an 'import completed' message displays at the bottom of the process log.
    - If the process failed, go back and try the steps again (making sure the correct attribute validation base type was selected on the Transformations screen and the identification was done properly in Step 5 of the Import Manager).
14. Verify the imported data is correct.

Transform Using Attribute Transformation Example

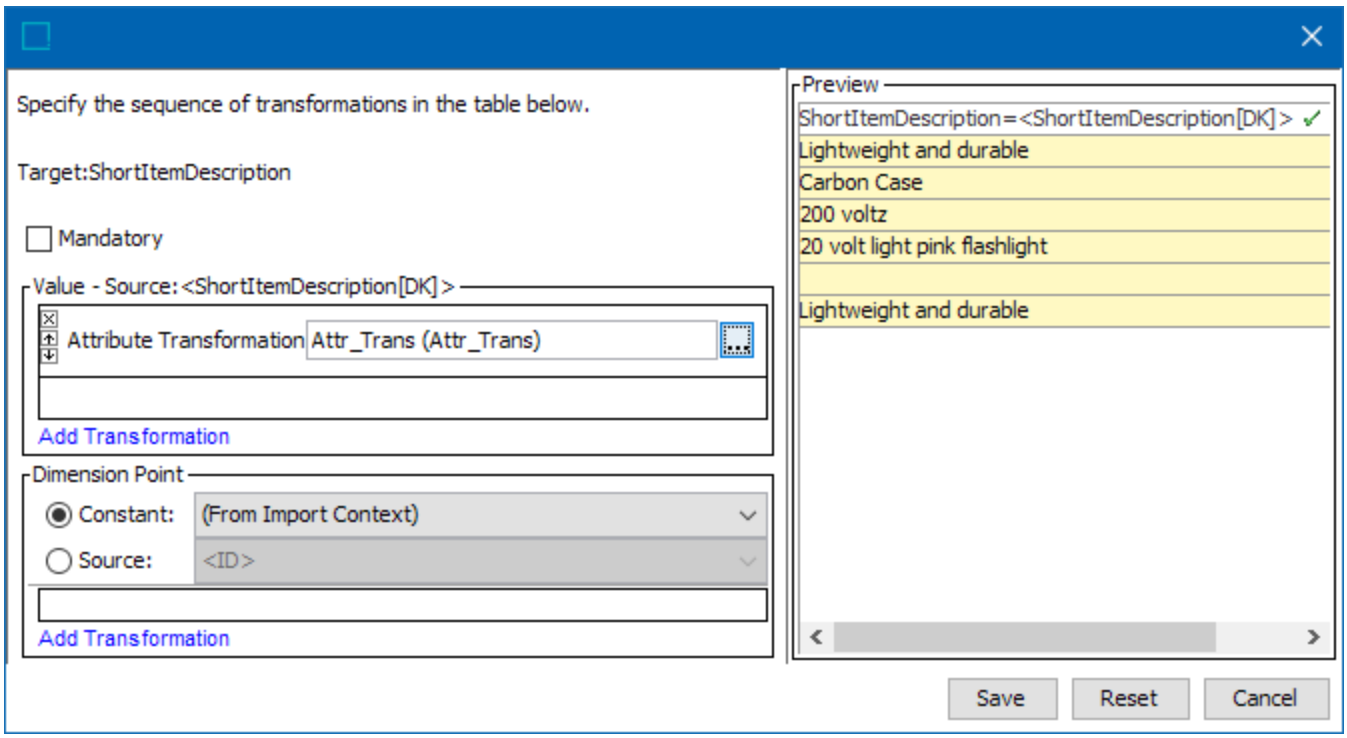
Transformations can improve the quality of the data you want to import. For general information about transformations, refer to the **Inbound Map Data - Transform** section.

Using the '**[Transform Using Attribute Transformation]**' transformation, it is possible to apply a set of transformations specified in an attribute transformation to a value. The transformation takes a predefined attribute transformation as argument. This can be treated as a convenient setup to avoid duplication of such transformations, and it is easy to maintain once created. **[Transform Using Attribute Transformation]** are defined in System Setup, and are predominantly used for print and tables.

For example, a user can apply the required transformations.



Then select the attribute transformation to apply the full set of transformations during a data import.


Specify the sequence of transformations in the table below.
Target:ShortItemDescription
Mandatory
Value - Source:<ShortItemDescription[DK]>
Attribute Transformation
Attr_Trans (Attr_Trans)
Add Transformation
Dimension Point
Constant:
(From Import Context)
Source:
<ID>
Add Transformation
Preview
ShortItemDescription=<ShortItemDescription[DK]>
Lightweight and durable
Carbon Case
200 voltz
20 volt light pink flashlight
Lightweight and durable
Save
Reset
Cancel

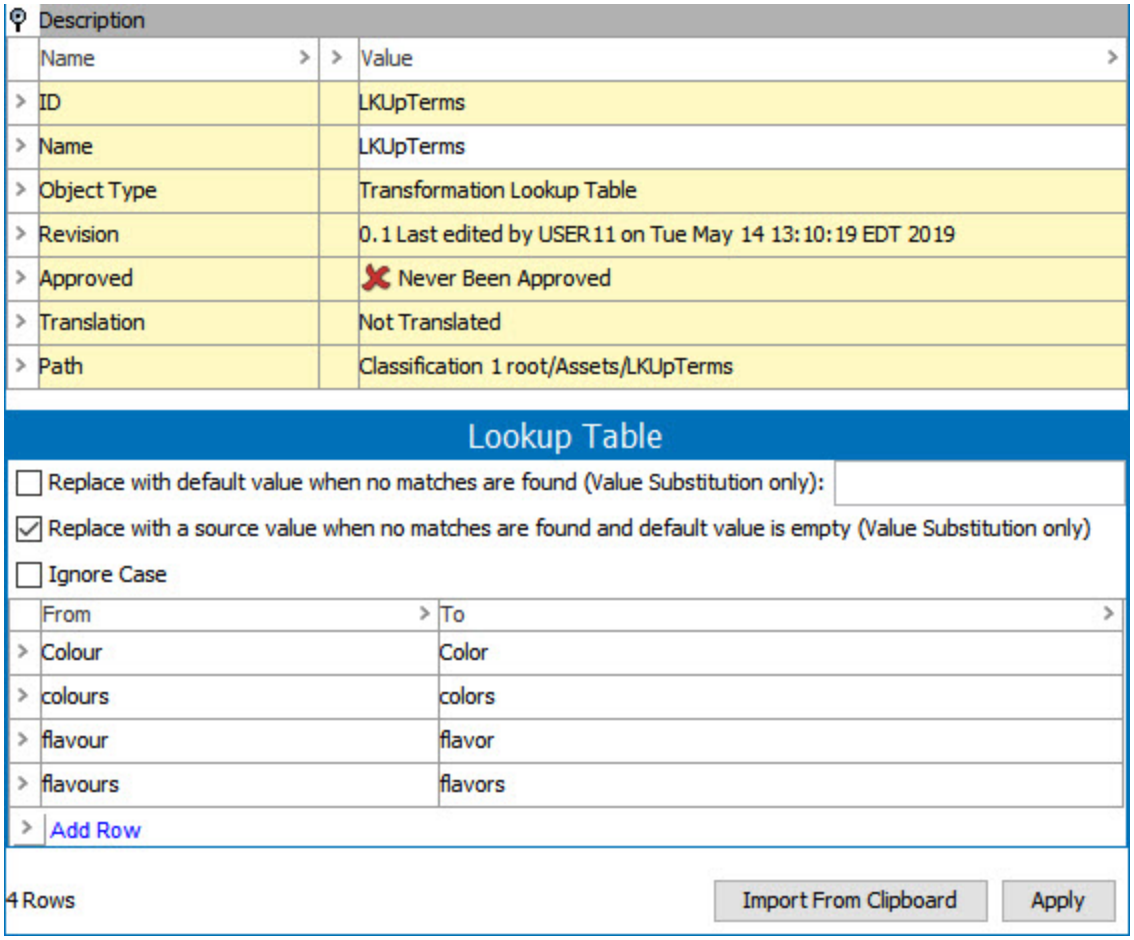
Transformation Lookup Table Example

Transformations can improve the quality of the data you want to import. For general information about transformations, refer to the **Inbound Map Data - Transform** section.

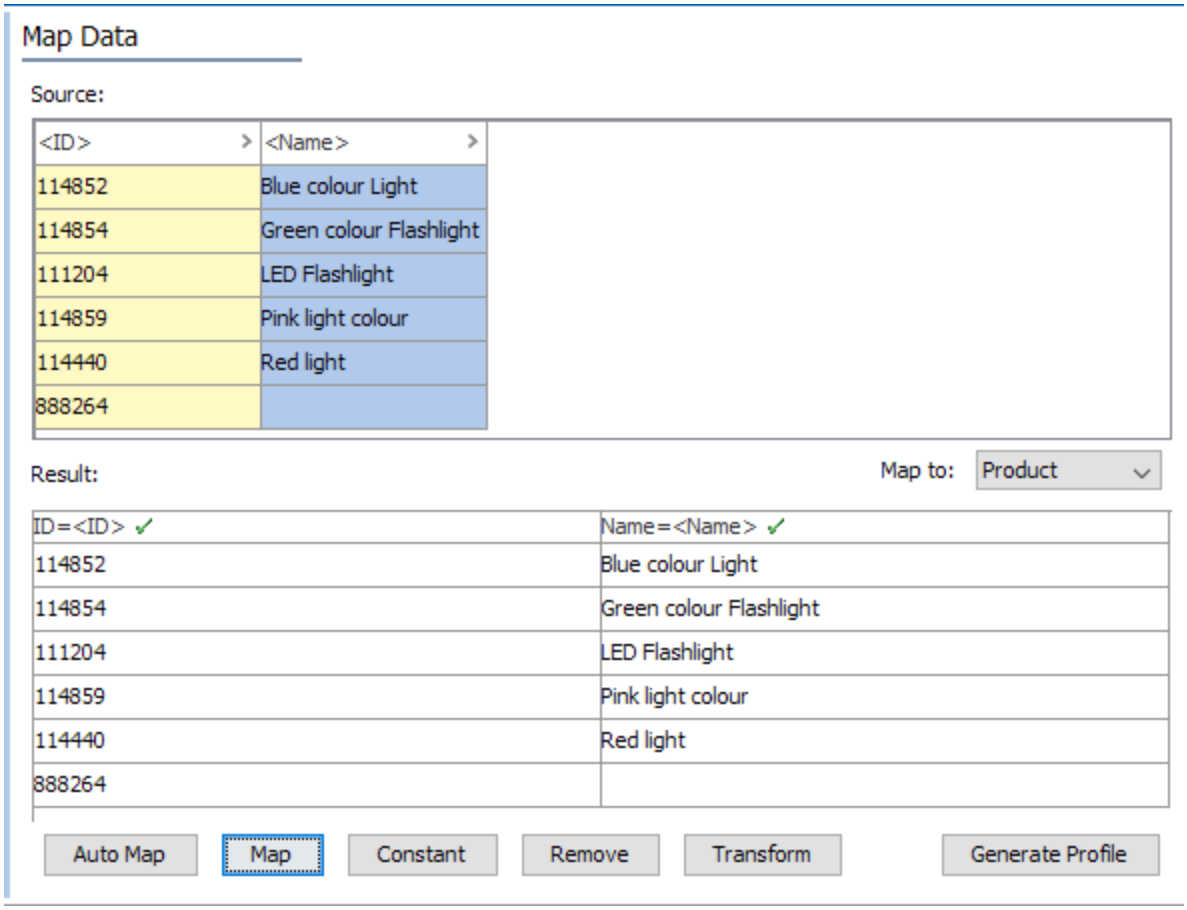
Transformation Lookup Tables can be used to transform words and entire values when importing tabular-based data. For information on lookup tables, refer to the **Transformation Lookup Tables** topic in **Resource Materials** online help.

Note: Transformations that use Lookup Tables can be saved as an Import Manager configuration. To use a saved Lookup Table in a configuration, the referenced Transformation Lookup Table must not be deleted and must have content in the context of the import.

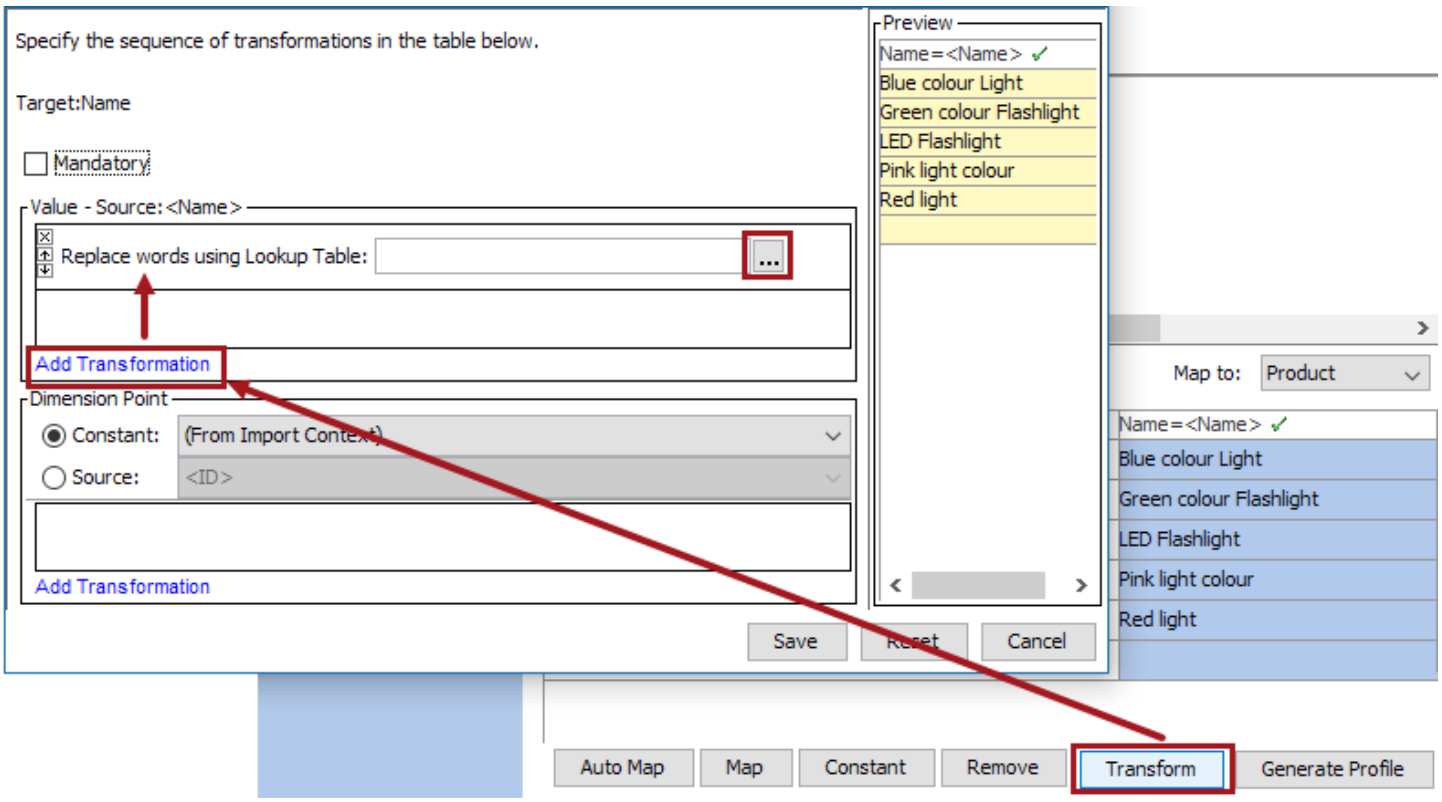
For example, it is possible to replace all occurrences of a word with another word before data is imported into an attribute. Using a Transformation Lookup Table, you can add a 'Replace words using Lookup Table' transformation on the result column, and select the relevant Lookup Table. For this case, all instances of 'colour' are being changed to 'color' by using this transformation table.



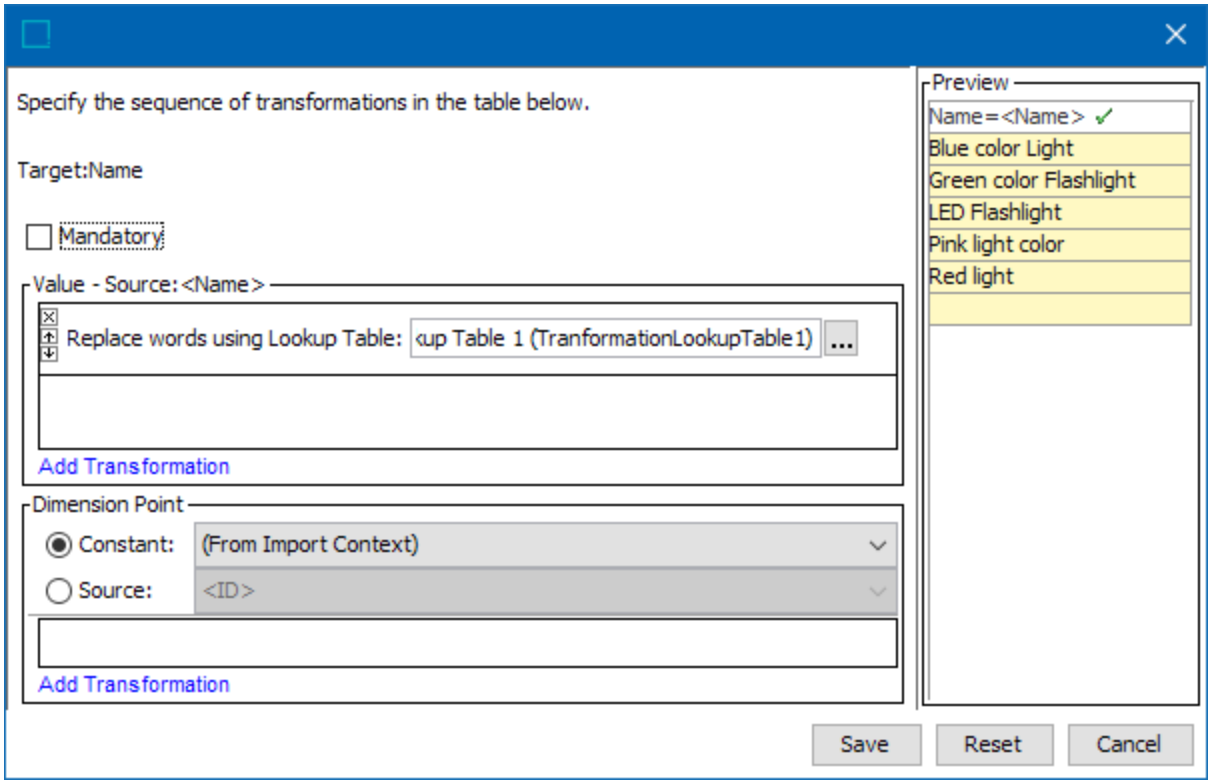
1. In the **Preview**, before any transformations are added to the Results field, the data looks like the table below:



2. Select the 'Name' field in the Source table, then click **Transform**. Click the **Add Transformation** link and select the 'Replace words using Lookup Table' option, and choose the needed transformation table.

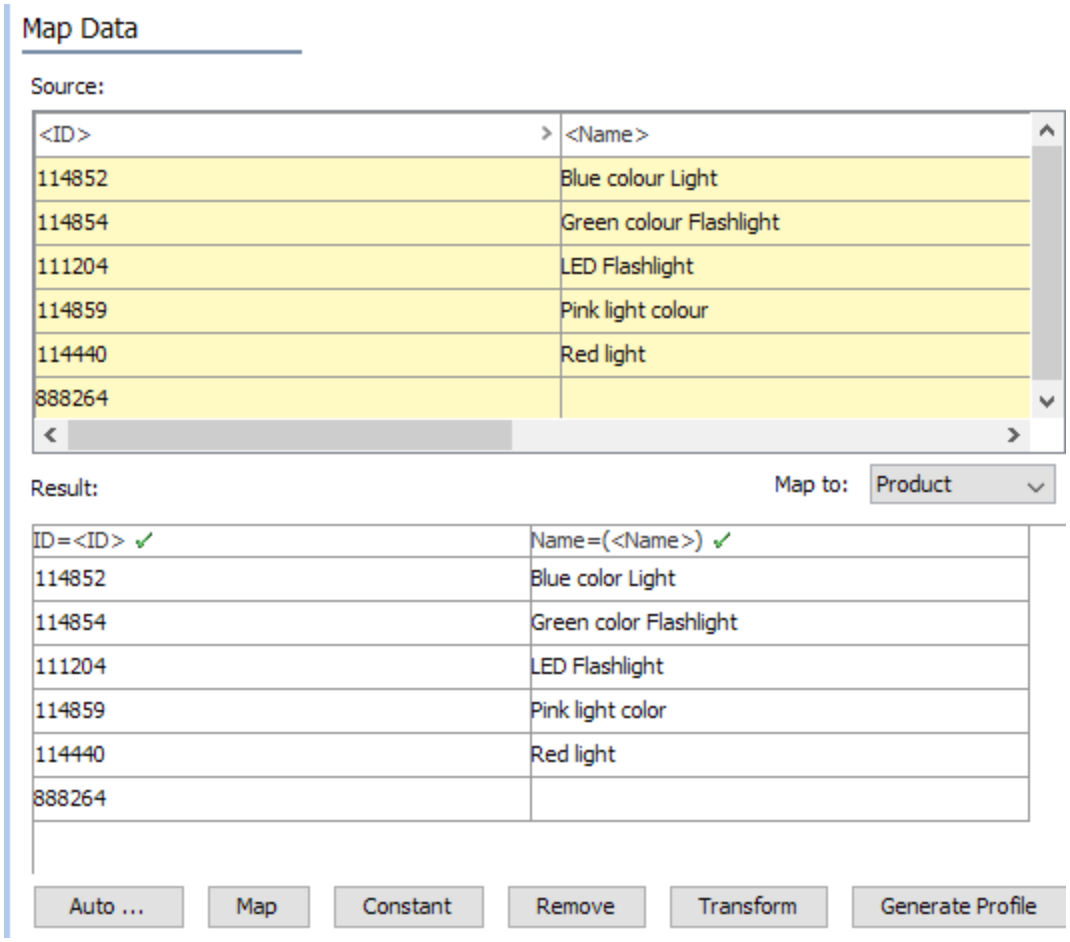


3. When properly added, the transformation changes the input text to the proper format.



Note: When whole values are replaced with multi-valued attribute data, the replacement works on the separate entries in the multi-value when added in the Value section. When added in the Before Multivalue Split section it works on the entire string from the import file including the multi-value separators.

4. Click **Save** to exit the Transformations dialog and update the Map Data step contents.



Export Manager

The Export Manager allows you to export data on demand or at scheduled intervals and to save or modify an export configuration. The final step of the wizard allows you to save the export configuration. This is useful for frequently-used exports where the parameters and options are always, or nearly always, the same. A saved export configuration saves setup time, and is required for scheduled exports.

Setup Requirements

Setting up and using the Export Manager involves the following steps:

1. Based on the data being exported, determine how to start the Export Manager wizard as described in **Creating a Data Export**.
2. In the wizard, choose a previously saved configuration or start a new export as described in **Export Manager - Select Configuration**.
3. In the wizard, select one or more hierarchy nodes or modify the nodes selected via a configuration as described in **Export Manager - Select Objects**.
4. In the wizard, select the file format in which the extracted information is delivered as described in **Export Manager - Select Format**.
5. In the wizard, when required, select which pieces of information are extracted per object exported as described in **Export Manager - Map Data**.
6. In the wizard, when required, specify if exported data should be converted to another output format as described in **Export Manager - Advanced**.
7. In the wizard, specify how information is to be delivered as described in **Export Manager - Select Delivery Method**.
8. In the wizard, save the configuration (if necessary), and run the export as described in **Running a Data Export**.
9. Monitor the export and download the output file as described in **Monitoring a Data Export**.

Additional Information

The following information is useful once an export configuration is saved:

1. Maintain a saved configuration as described in **Maintaining a Saved Export Configuration**.
2. Perform the selected export configuration one or more times, according to the scheduling parameters, via a background process as described in **Scheduling a Data Export**.
3. Remove a scheduled export process, which is required to run current parameters if the original configuration is modified, as described in **Deleting a Scheduled Background Process** within the **System Setup** documentation.

Creating a Data Export

The Export Manager wizard allows you to export data on demand and to create or modify an export configuration. The final step of the wizard allows you to save the export configuration. This is useful for frequently-used exports where the parameters and options are always, or nearly always, the same.

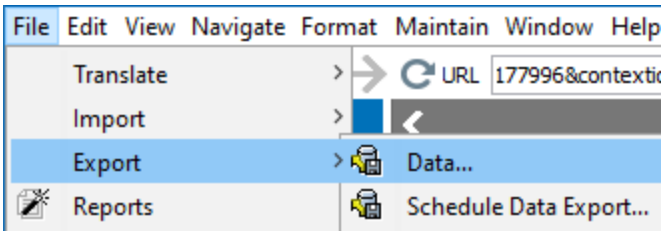
Export data using the Export Manager

1. Select the appropriate STEP context. All data is exported using the currently selected context and the current workspace, typically Main.

   Note: When exporting products, refer to the **Considerations for Exporting Products** section below.

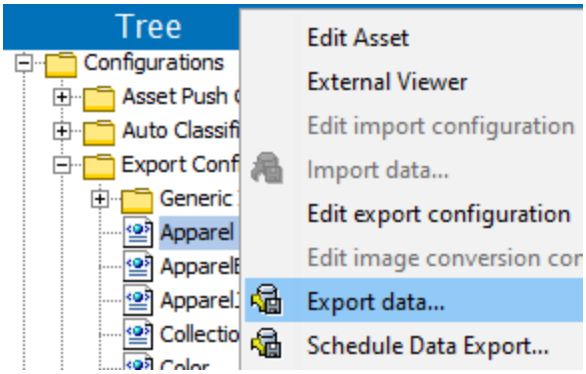
2. Use one of the following methods to launch the Export Manager wizard, noting that method of starting the wizard results in exporting different objects. Regardless of the method selected, the same wizard is launched.
   - **From the File Menu**: Click the File menu > Export > **Data**.

     The wizard opens in Step 1 and allows you to select an existing configuration or create a new one.

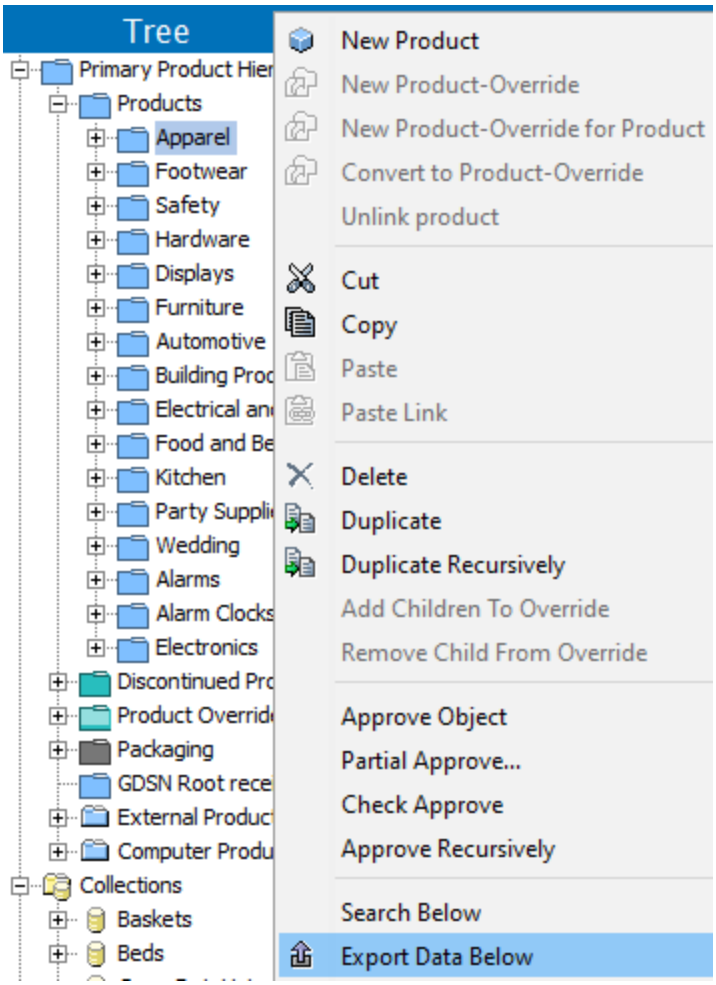
   - **From an Export Configuration**: On the Tree, select an export configuration, right-click, and then select **Export Data**.

     The wizard opens in Step 2 with the configuration and objects included in the configuration already selected.

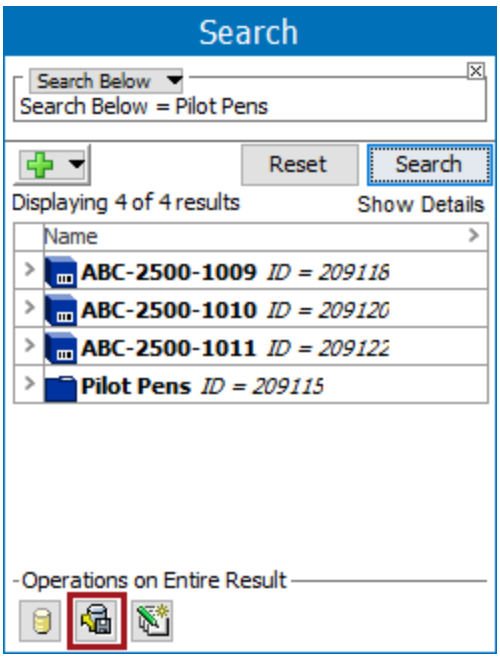
- **From an Object**: On the Tree, select a classification, product group, publication, or entity, right-click, and then select **Export Data Below**.

  The wizard opens in Step 2 with the objects below the selection displayed. This selection is non-inclusive, meaning that the selected object is not exported. For example, when selecting a product node in the primary product hierarchy, the selected node is not exported, but the children are exported. When selecting one or more individual products that have no children, only the selected products are output.



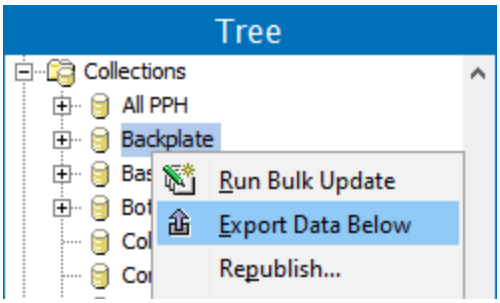
- **From a Search Operation**: On the Search tab, perform a search with results displayed. In the **Operations on Entire Result** section, click the export button.

  The wizard opens in Step 2 with 'Product selection made by search URL' displayed.



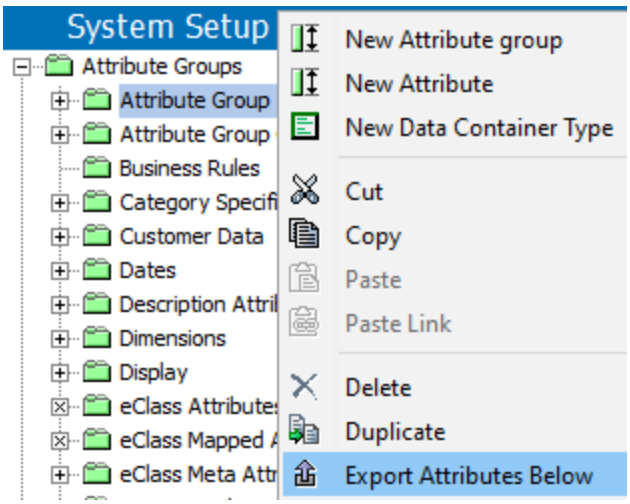
- **From a Collection**: On the Tree, select a collection, right-click, and then select **Export Data**. All object in the collection are included in the export.

  The wizard opens in Step 3 ready to choose a format.

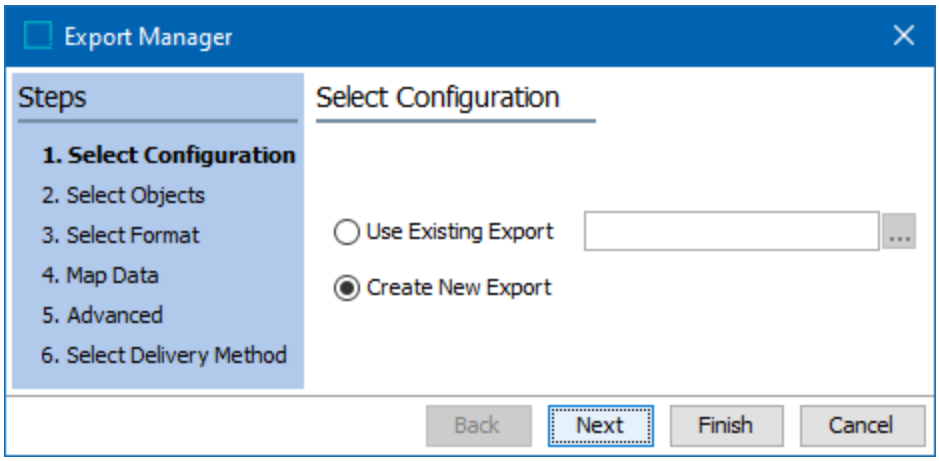


- **From an Attribute Group**: On System Setup, select an attribute group, right-click and then select **Export Attributes Below**.

  The wizard opens in Step 2 with the attributes below the selection displayed.



3. After selecting one of the above ways to export, the Export Manager wizard displays and can involve the following steps:



- **Select Configuration** allows you to select a previously stored configuration or start a new export.
- **Select Objects** allows you to select one or more hierarchy nodes or modify the nodes selected via a configuration.
- **Select Format** allows you to select the file format in which the extracted information is delivered.
- **Map Data** allows you to select which pieces of information are extracted per object exported.
- **Advanced** allows you to specify if exported data should be converted to another output format.
- **Select Delivery Method** allows you to specify how information is to be delivered.

Considerations for Exporting Products

When exporting products, it is common setup to first review exactly the products and attributes that you want to extract. It is important to choose the exported attributes carefully since there is no distinction in the output between a missing (or blank) attribute value and a value that is empty because the attribute is not valid for a product. Including invalid attributes in an export results in output with blank attribute data and in many cases, the

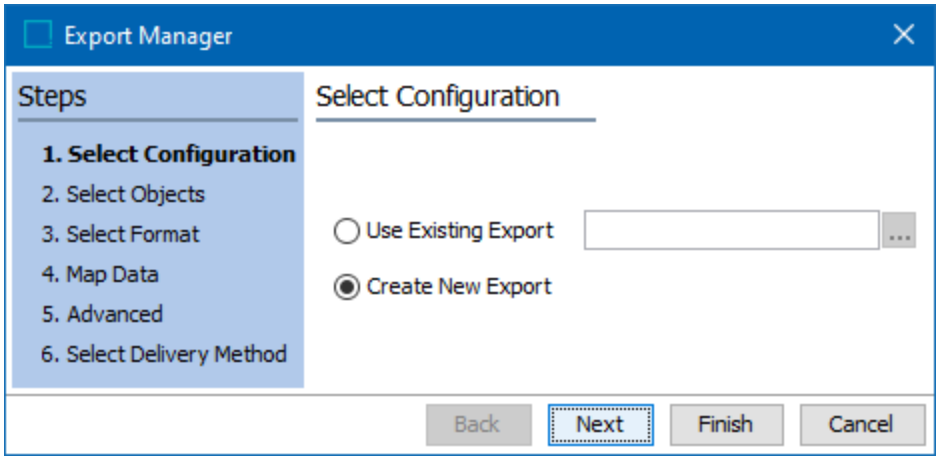
cause of the missing data is not apparent. The Include Empty Values export only includes valid and linked attributes to prevent orphans and filter out invalid attributes when selecting attribute groups versus individual attributes to export.

Determine the following information about your required export enables you to accurately set the wizard parameters:

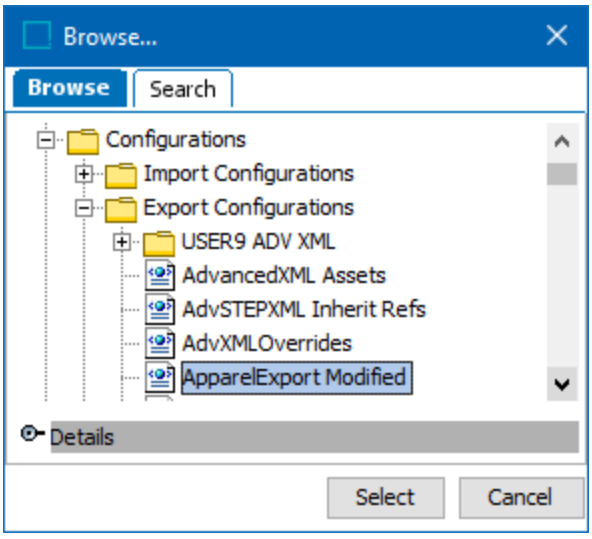
- If the attributes you want to extract are global, and for example, relevant for nearly all products, you can select products from disparate product lines, since the attributes would be valid for all those products.
- If the product lines you want to extract do not have common attributes, it can be useful to extract multiple files with separate sets of attributes.
- If your goal is to create a spreadsheet that manufacturers can return to you with missing data, extract the product lines, including the manufacturer's name as an attribute, and then split the extracted file into separate files for each manufacturer. In this way, the attributes are valid for the products in each file.
- If you know the attributes you want to extract before you launch the Export Manager wizard, it is easier to select the appropriate attributes on the Data Map step.
- If you want valid, empty data to be included in the export when exporting via Advanced STEPXML, CSV, Excel, or STEPXML, this can be accomplished via the 'Include Empty Fields' option in the Export Manager wizard.

After completing the wizard steps, save the configuration so you can modify it during testing. Initially, use a small group of products to test your export settings. Refine the wizard parameters to resolve any issues and save the updated configuration.

Export Manager - Select Configuration

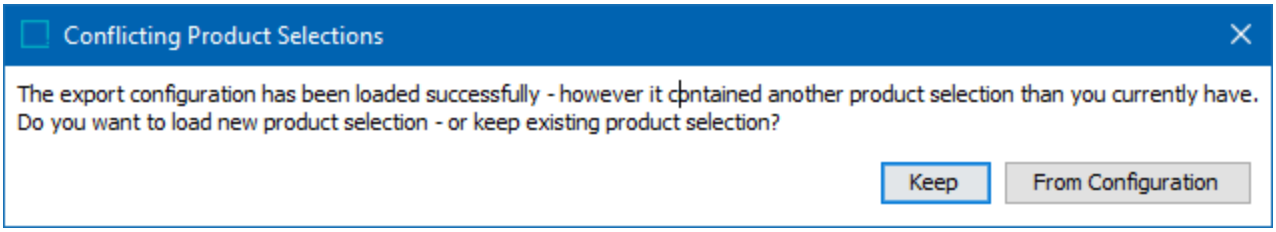


1. Select a radio button to use an existing export configuration, or create a new one. The following options are available when using an existing configuration:
   - To use an export configuration previously saved, select the **Use Existing Export** radio button, click the ellipsis button (...), and either Browse or Search to locate the configuration in STEP. For more information on saved configurations, refer to the **Maintaining a Saved Export Configuration** topic.



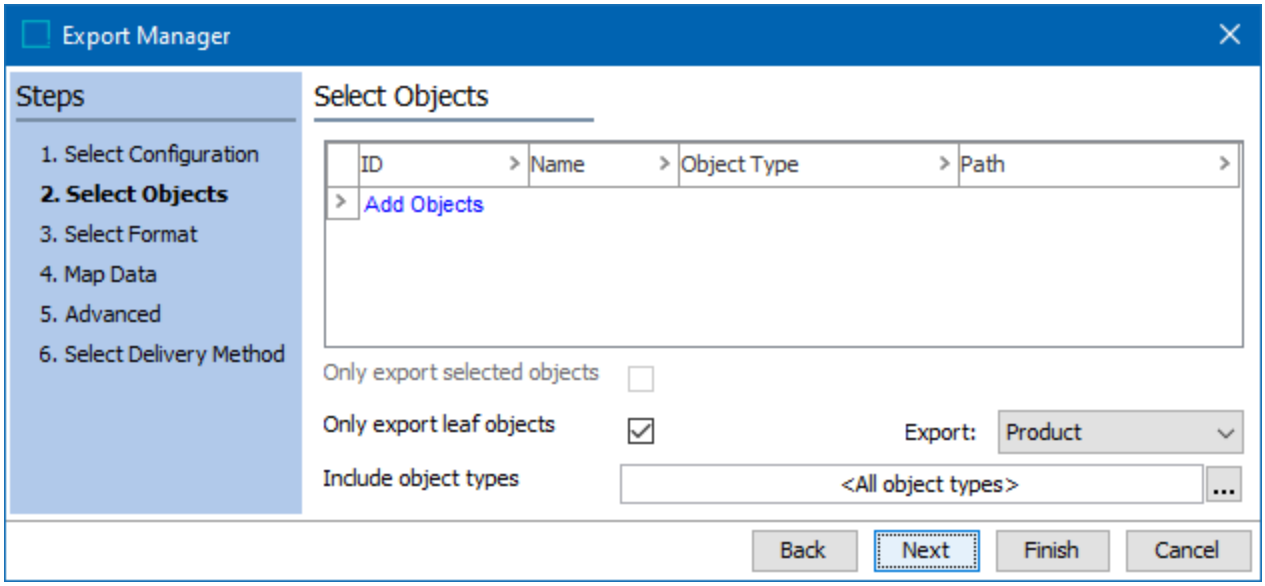
   - To create a new export configuration based on an existing export configuration, select the **Use Existing Export** radio button, click the ellipsis button (...), and either Browse or Search to locate the existing configuration, use the other wizard steps to change the configuration to meet your current requirements, and then save the modified configuration in the final step of the wizard.
   - When returning to the Select Configuration step after making a selection on the Select Objects step, select **Use Existing Export** to apply a saved configuration. In this scenario, if the objects selected differ from those in the saved configuration, the Conflicting Product Selections dialog is presented. Click **Keep** to

continue with the manually selected objects, or click **From Configuration** to use the object selection in the saved configuration.



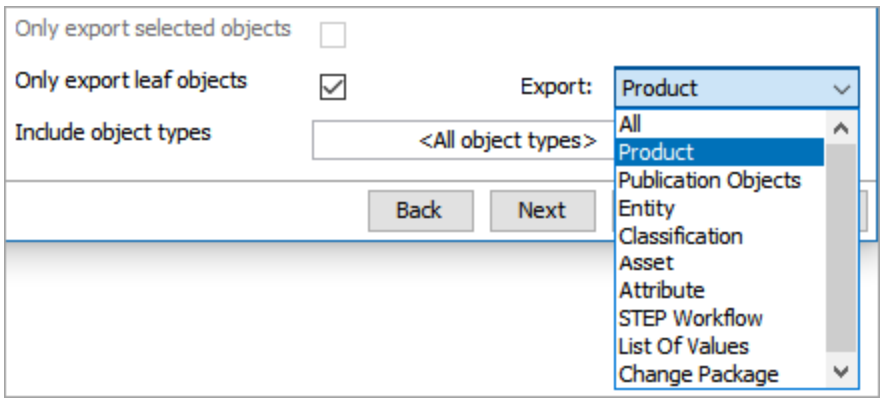
2. Click **Next** to display **Export Manager - Select Objects**, or click **Finish** to use an existing configuration without changes, and display the Save Export Configuration window as defined in **Running a Data Export**.

Export Manager - Select Objects



1. In the **Export** dropdown, select the object super type to export and allow the next step of the wizard to show only legal export formats.
   - **All** - all objects of the selected node are exported. Most formats are available. Use this option to choose one or more specific configuration objects from the System Setup tab, such as reference types, to be exported with STEPXML.
   - **Asset** - only metadata on assets is exported from the selected node, export of the actual digital asset (asset content) is not determined by this selection. Most formats are available. For information on asset content exports, refer to **Exporting Assets** in the **Digital Assets** documentation.
   - **Attribute** - all attribute objects are exported for the selected attribute group. Most formats are available.
   - **Change Package** - only sealed change package objects are exported. Advanced STEPXML format is available with default template.
   - **Classification** - only classification objects are exported from the selected classification hierarchy. Most formats are available.
   - **Entity** - only entity objects are exported from the selected node. Most formats are available.
   - **List of Values** - only LOV objects are exported. Advanced STEPXML (with default template), STEPXML (with default parameter settings), and Generic XML formats are available.
   - **Product** - only product objects are exported from the selected product hierarchy. All formats are available.
   - **Publication Objects** - all publication-related objects are exported. When a publication group object is selected, this option defaults to Publication Objects and the format defaults to STEPXML, allowing the publication-related parameters to be set. The Publication Excel or Flatplan Excel formats are available.
   - **STEP Workflow** - all available workflows are exported unless individual workflows are selected. Advanced STEPXML (with default template) and STEPXML (with default parameter settings) formats are available.

   For example, when adding objects, you could select a classification hierarchy node, then choose the Product export type, which results in exporting all products linked in below the selected classification.

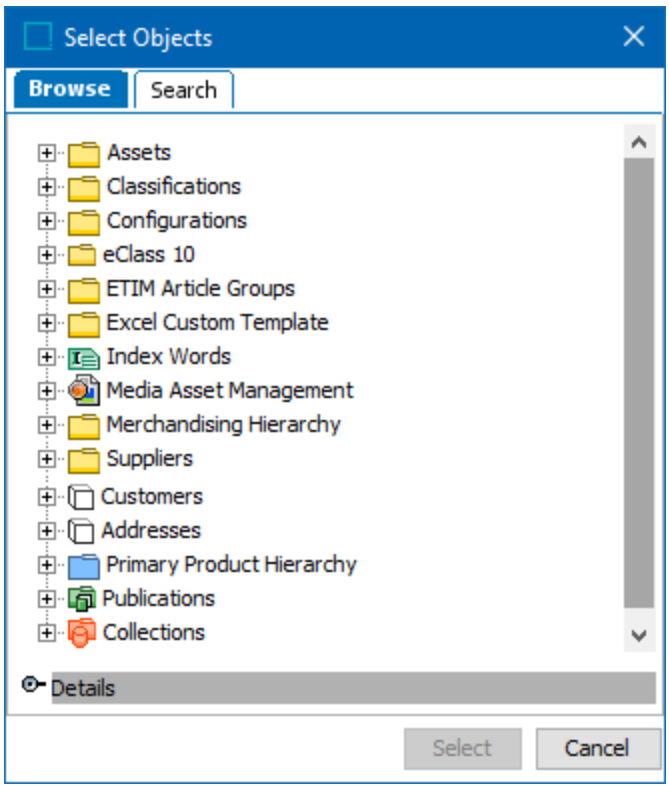


Note: Selecting an option from the Export dropdown specifically impacts formats that require mapping. For STEPXML, which does not require mapping, the Export dropdown modifies the default settings on the STEPXML Format step. For other formats, when the Map Data step is disabled, this selection has no effect.

2. Click the **Add Objects** link to select objects, or leave the Select Object tables empty and export based on the other wizard parameters. If you started the export from a node using Export Below, the objects are selected automatically.

   If exporting an Excel Smartsheet, refer to the **Selecting Objects for Excel Smartsheet Export** section below.

3. In the Select Objects window, use the **Browse** or **Search** tab to locate and select objects for export. The available objects displayed in the Select Objects window are determined by selection you made in the Export dropdown.
   - To export specific branches of the product, entity, collection, or classification hierarchy, select the root node for the export.
   - Multiple objects can be selected at the same time.
   - The selected root node object and any objects below will be exported, if allowed by other parameters in the wizard.

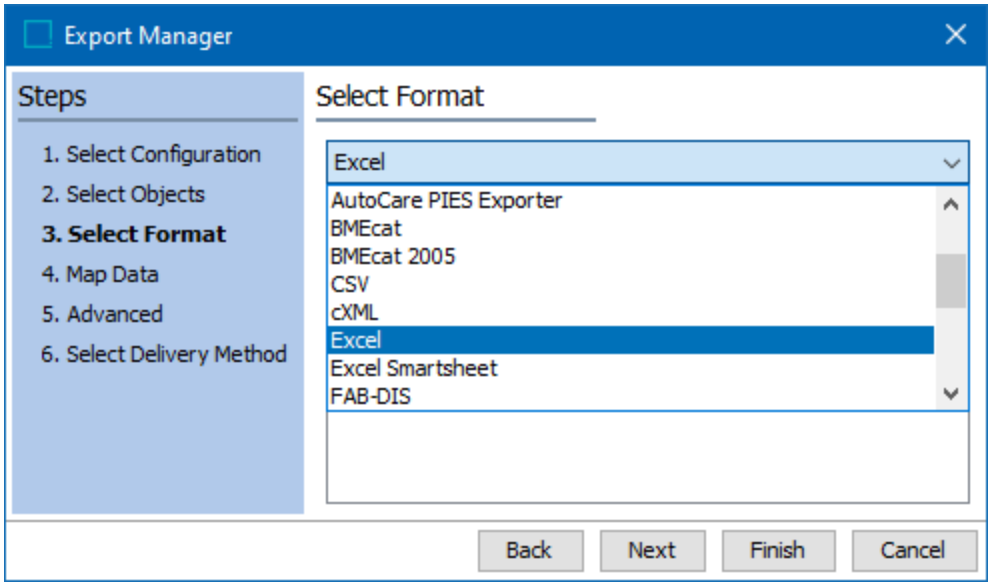


4. Click **Select**. The selected nodes appear in the **Select Objects** window.
5. Select **Only export selected objects** to export only the root nodes, and not the objects that live below the selected node. This option is only available when 'Only export leaf objects' is not selected. If available and unchecked, objects and their children are exported.
6. Select **Only export leaf objects** if you have selected a top hierarchy of products to be exported, but only want the lowest level on the Tree to be included in the export file. Selected objects are only included in the export if they have no children. This option is only available when 'Only export selected objects' is not selected. If available and unchecked, objects and their children are exported.

   Note: This option is ignored when exporting STEPXML format.

7. The **Include object types** parameter is used to limit the export to objects of a specific Object Type. To modify the parameter, click the ellipsis button (...) and select the object types to be exported. Leave this field blank to include all object types.
8. Click **Next** to display **Export Manager - Select Format**.

Export Manager - Select Format

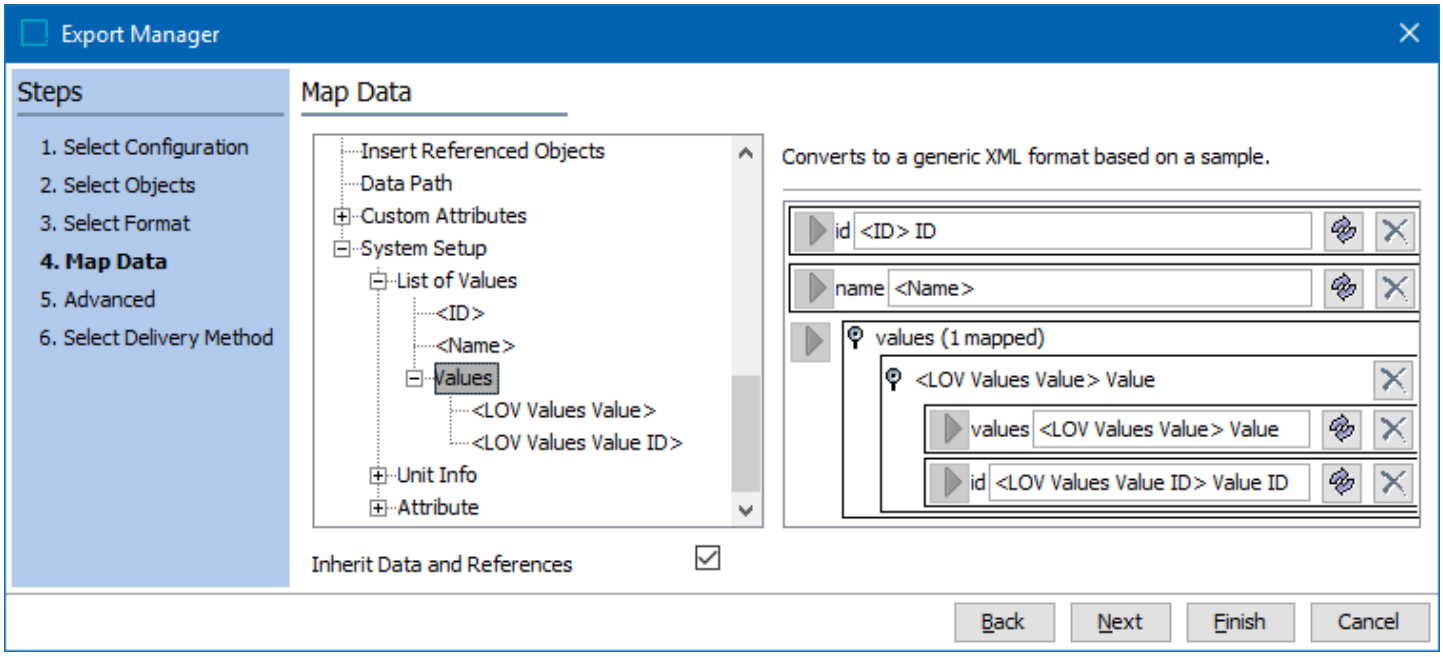


1. The selected option determines the file format of the extracted data.
   - Advanced STEPXML Format
   - Alphabetical Index - XML Format
   - Ariba CIF Format
   - Ariba CIF 3.0 Format
   - BMEcat Format
   - BMEcat 2005 Format
   - BMEcat 2005.1 Format
   - CSV Format
   - cXML Format
   - Excel Format
   - Excel Custom Template
   - Excel List of Values
   - Excel Smartsheet Format
   - FAB-DIS Format
   - Flatplanner Format
   - Generic XML Format
   - Generic JSON Format
   - IDoc MATMAS05 Format
   - Publication Excel Format

- STEPXML Format
- STEPXML Configuration Export Format
- xCBL Format

2. Click the **Next** button to display **Export Manager - Map Data** when required by the format, or **Export Manager - Advanced** if mapping is not required.

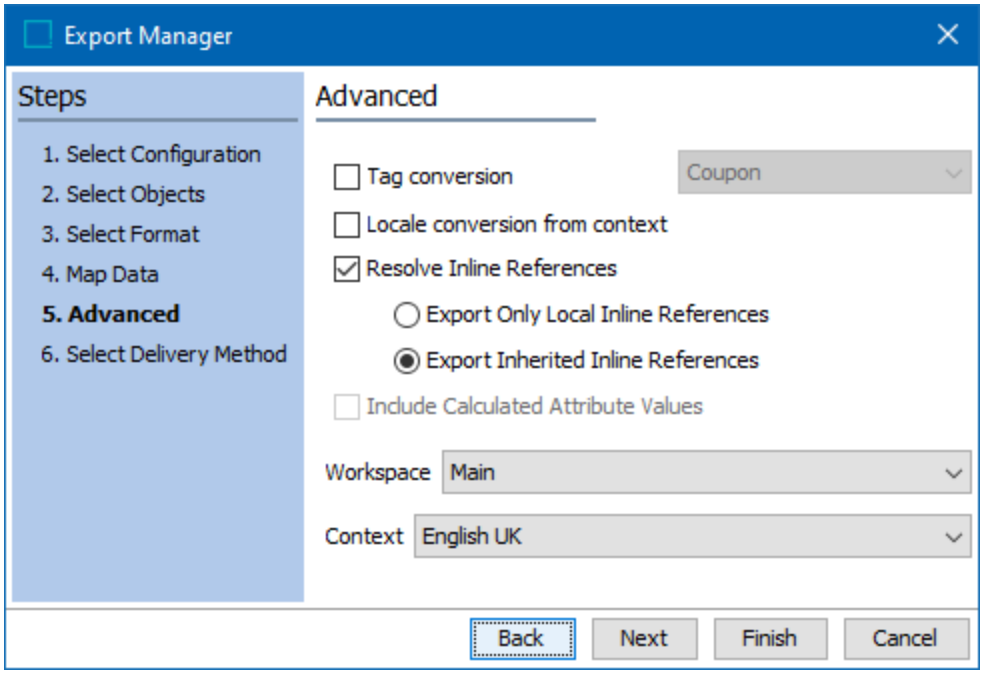
Export Manager - Map Data



Generally, you will create mapping rules by selecting a data source (which indicates what you want to extract for each record) and creating a mapping target (which indicates where it belongs in the output).

1. Select a Data Source from the left panel, as defined in the **Outbound Map Data - Data Source** documentation.
   - The options available for mapping are the same for all formats except Generic XML, which includes additional data sources. Outbound mapping is discussed in detail in the **Outbound Map Data Options** documentation.
   - For details on the inheritance checkbox, refer to **Inherit Data and References - Data Source Outbound** documentation.
2. Click the right arrow ( ) to create a mapping target, as defined in the **Outbound Map Data - Mapping Targets** documentation.
3. If necessary, click the transformation button ( ) to modify the output data as defined in the **Outbound Map Data - Transform** documentation.
4. Click the **Next** button to display **Export Manager - Advanced**.

Export Manager - Advanced



1. Set the following additional export parameters as needed:
   - **Tag conversion**: Optional. Allows for STEP Tags to be converted to match the selected output tag format.

     For example, the STEP bold tag could be converted to the HTML equivalent. Tags and Tag Conversions are configured under Tags in System Setup. The selection in the dropdown determines the output formatting. For more details, refer to the **Tags** topic, and for the available tag conversion formats, refer to the **Configuring Tag Output Formatting** topic in the **System Setup** documentation.

   - **Locale conversion from context**: Optional. When checked, converts dates and numbers to comply with the selected locale for the Context selected before starting the Export Manager wizard according to the locale rules associated with the context.

     Important: Selecting **Locale conversion from context** when exporting a Smartsheet can break the automatic conversion to the client's native decimal separator, causing the import of the Smartsheet to fail.

   - **Resolve Inline References**: Determines how inline references are resolved when exported. By default, this option is checked, which means that if inline references are included in the data, they will be resolved (rather than tagged) in the export. If not checked, inline references are exported with the inline references tagging. Once checked, choose one of the following options:
     - Export Only Local Inline References means only local references are resolved and exported.
     - Export Inherited Inline References means references inherited from a parent are resolved and exported.

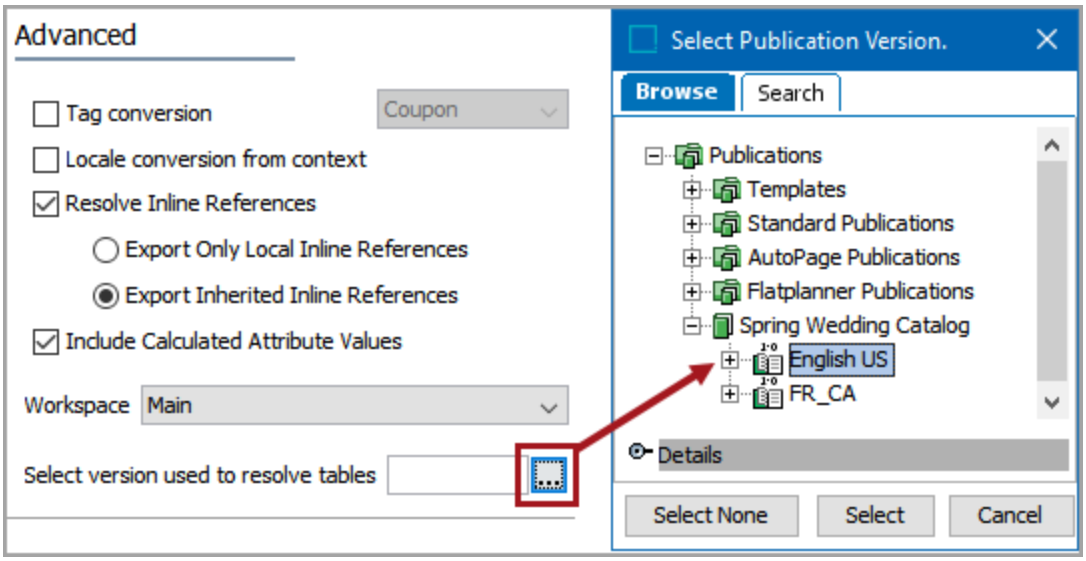
- **Include Calculated Attribute Values**: This option is only enabled (and is checked by default) when calculated attributes have been mapped. When checked, calculated attribute values are resolved upon export.

  If not checked, calculated attribute values are exported with empty values, unless the Force Calculation option has been set. The Force Calculation option is available when mapping a calculated attribute using the Select Attribute data source, and also as a transformation. For more information, refer to **Force Calculation** in **Attribute - Data Source Outbound** and **Outbound Map Data - Transform**.

  The value template is exported for each selected context, including a qualifier ID, which makes it possible to import the same data back into STEP. For information about calculated attribute values, refer to the **Calculated Attributes** topic in **System Setup** documentation.

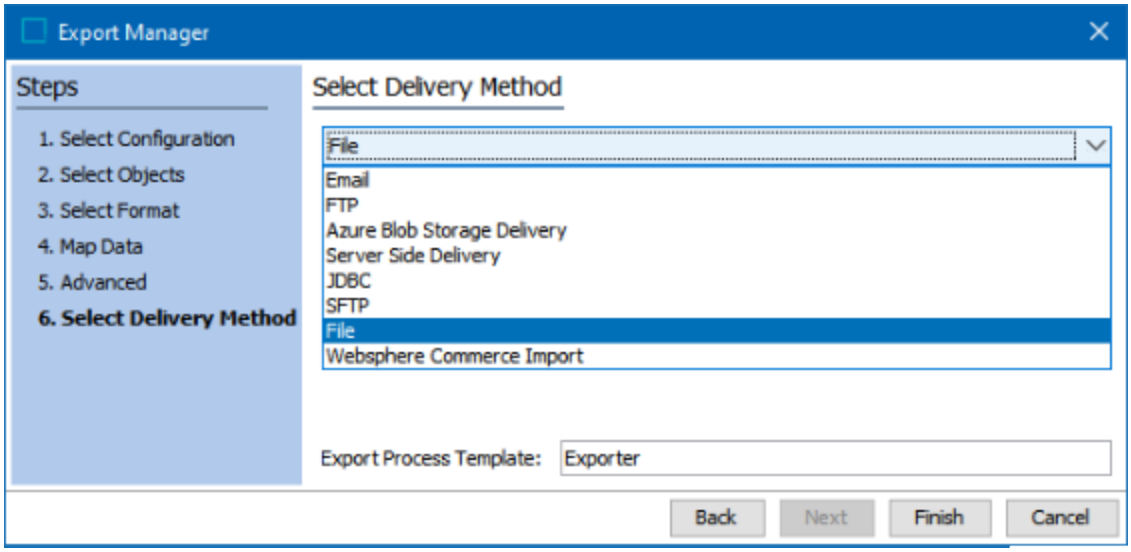
  Important: If many complex calculated attribute values (traversing hierarchies and/or references) are used, consider if they should be exported, since they can impact performance. If required, consider scheduling for non-peak times. Simple calculations are not detrimental to an export, regardless of the quantity.

- **Workspace**: Allows the export to run from a different Workspace than the one selected when the Export was started. Only a single workspace can be selected for export.
- **Context**: Allows the export to run from a different Context than the one selected when the Export was started. For all formats except STEPXML, only a single context can be selected for export. For information about exporting multiple contexts with STEPXML, refer to the **Export Data for Selected** Contexts section of the **STEPXML Outbound Parameters** documentation.
- **Select version used to resolve tables**: This setting is available for STEPXML if 'Include Tables' is set to Yes and for Advanced STEPXML if the template includes version-dependent content. A publication version should be selected if the tables contain content relevant to a particular publication (such as column or row types that are only valid for certain publication types) or a publication version (such as commercial data). For more information, refer to the **Exporting Resolved Tables** topic in the **Tables** documentation.



2. Click the **Next** button to display **Export Manager - Select Delivery Method**.

Export Manager - Select Delivery Method



Delivery methods are used by both outbound data tools but the available options vary. For information on the delivery methods available in OIEPs, refer to **OIEP - Delivery Method Flipper**.

1. In the dropdown displaying the default 'File' option, select a delivery method. The available options are defined in the Delivery Methods section below.
2. For the **Export Process Template** parameter, leave the default setting, which is required to start the background process for the export.
3. Click **Finish** to display the Save Export Configuration window as defined in **Running a Data Export**.

Delivery Methods

Method	Description
Azure Blob Storage Delivery	Delivers files to Azure Blob Storage. Refer to the **Azure Blob Storage Delivery Method** documentation.
Cloud Blob Storage Delivery	Delivers exported files to cloud storage. Amazon S3, Google Cloud Storage, and Microsoft Azure are supported. Refer to the **Cloud Blob Storage Delivery Method** topic.
Email	Delivers files as an email attachment and allows a zipped delivery. Refer to the **Email Delivery Method** topic.

Method	Description
File	Delivers a file via a background process and notifies when the file is ready for download. Refer to the **File Delivery Method** topic.
FTP	Delivers files using file transfer protocol (FTP). Refer to the **FTP Delivery Method** topic.
JDBC	The JDBC delivery option enables delivery of data to RDBMS-type databases like Oracle, MySQL, MS SQL Server, PostgreSQL, etc. Refer to the **JDBC Delivery Method** topic.
Server Side Delivery	Exports a file to a folder located on the STEP system application server. Refer to the **Server Side Delivery Method** topic.
SFTP	Delivers a file using the Secure File Transfer Protocol (SFTP). Refer to the **SFTP Delivery Method** topic.
WebSphere Commerce Import	Delivers a STEPXML file to a WebSphere Commerce server. Refer to the **WebSphere Commerce Import Delivery Method** topic.

Azure Blob Storage Delivery Method

The Azure Blob Storage Delivery method in Export Manager makes it possible to export files to Azure Blob Storage. This delivery method is available in Export Manager and OIEPs.

Note: To deliver export files to Azure blob storage, the **Cloud Blob Storage Delivery Method** can also be used. The differentiation between the setup and functionalities is that the Cloud Blob Storage Delivery plugin is part of the STEP baseline, can be used for Amazon S3, and the Export Manager and OIEP configurations use the gateway integration endpoints instead of having separate delivery method integration properties.

To use this method in an OIEP refer to the **Azure Blob Storage Delivery Method** topic.

Prerequisites

The Azure Blob Storage Delivery plugin is part of the 'cloudstorage-azure' component that must be installed in addition to the STEP baseline. No additional licenses are required.

Prior to configuration, click the **Configuration name** dropdown parameter to display the required configuration to be used. You will provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the following configuration properties:

```
AzureBlobStorageDeliveryPlugin.ConfigurationNames
AzureBlobStorageDeliveryPlugin.ConnectionString.[Configuration Name]
AzureBlobStorageDeliveryPlugin.ContainerName.[Configuration Name]
```

The value for 'AzureBlobStorageDeliveryPlugin.ConfigurationNames' is a comma-separated list of user-defined names for the desired Azure Blob Storage configurations. For each name, corresponding 'AzureBlobStorageDeliveryPlugin.ConnectionString.[Configuration Name]' and 'AzureBlobStorageDeliveryPlugin.ContainerName.[Configuration Name]' properties must be set with the values being the connection string for the configuration and the desired blob container name, respectively.

An example configuration using the storage account access key (*AccountName* / *AccountKey*) method is below:

```
AzureBlobStorageDeliveryPlugin.ConfigurationNames=Azurite

AzureBlobStorageDeliveryPlugin.ConnectionString.Azurite=DefaultEndpointsProtocol=htt
p;AccountName=devstoreaccount1;AccountKey=Eby8vdM02xNOcqFlqUwJPLlmEtlCDXJ1OUzFT50uSR
Z6IFsuFq2UVErCz4I6tq/K1SZFPTOtr/KBHBeksoGMGw==;BlobEndpoint=http://127.0.0.1:10000/d
evstoreaccount1;
AzureBlobStorageDeliveryPlugin.ContainerName.Azurite=productData
```

It is also possible to use a Shared Access Signature (SAS) credential for the ConnectionString.

The SAS token must be created directly on the blob storage account itself (and not the corresponding container); and as a minimum, it must have **Service**, **Container**, and **Object** specified as its 'Allowed resource types' as well as **Read**, **Write**, and **List** for its 'Allowed permissions' to grant the proper access rights to STEP.

These resource types and permissions are required to allow STEP to perform all the needed operations to deliver the content (blobs) to the specified Azure Blob Storage account's container.

Important: If the SAS token has insufficient privileges, the delivery will result in an error message similar to this one:

If you are using a SAS token, and the server returned an error message that says 'Signature did not match', you can compare the string to sign with the one generated by the SDK. To log the string to sign, pass in the context key value pair 'Azure-Storage-Log-String-To-Sign': true to the appropriate generateSas method call. Remember to disable 'Azure-Storage-Log-String-To-Sign' before going to production as this string can potentially contain PII.

Status code 403, "<?xml version="1.0" encoding="utf-8"?><Error><Code>AuthorizationResourceTypeMismatch</Code><Message>This request is not authorized to perform this operation using this resource type. RequestId:836910b1-801e-001a-4da2-900fc9000000 Time:2022-07-05T19:11:07.4796215Z</Message></Error>"

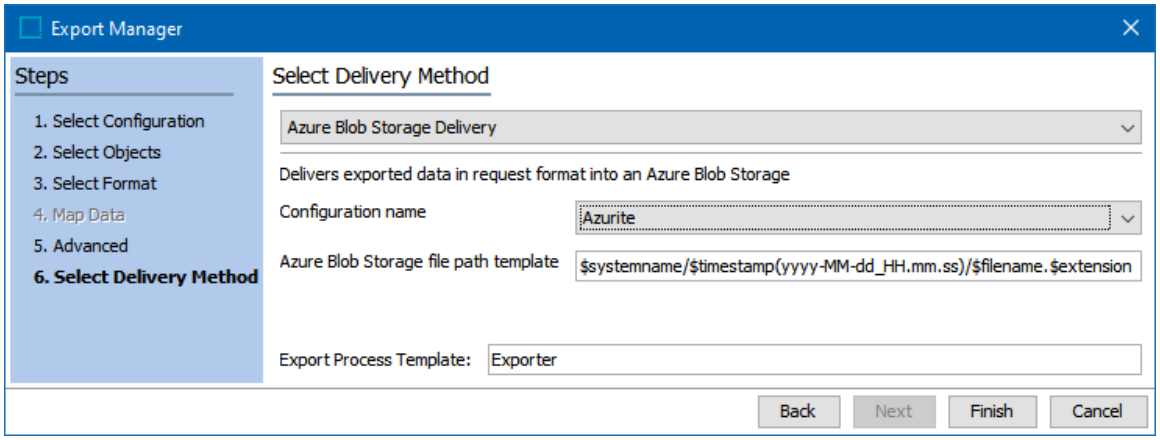
An example using a SAS configuration is below:

```
AzureBlobStorageDeliveryPlugin.ConfigurationNames=besiProducts
AzureBlobStorageDeliveryPlugin.ConnectionString.besiProducts=
BlobEndpoint=https://pimtest.blob.core.windows.net/;SharedAccessSignature=sv=2021-06-08&ss=b&srt=sco&sp=rwlx&se=2023-01-01T05:00:00Z&st=2022-07-05T18:58:41Z&spr=https&sig=N5X7J9tCMscbOTYioR4sb30H7B%2B0j8dk74MSCQ6Gxsw%3D
AzureBlobStorageDeliveryPlugin.ContainerName.besiProducts=productData
```

Once the property configuration is in place and the system has been restarted, the configuration name(s) will appear as selectable options in the STEP Workbench.

Configuration

For information on a parameter, hover over the parameter label to display help text.



1. Click the **Select Delivery Method** parameter to display the dropdown and choose **Azure Blob Storage Delivery**.
2. The **Configuration name** comes from the properties you set above. If you configured the properties to use multiple configuration names within a comma-separated list, then you will choose from the dropdown. *(Based on the example configuration given earlier in this topic, the only option available in this example is 'Azurite.')*
3. Configure the Azure Blob Storage file path template.

   In addition to selecting the appropriate connection string and container, it is possible to make use of Azure Blob Storage virtual directories. Four variables are available:

   - $systemname – The STEP system name. Useful when several STEP systems are delivering to the same blob container.
   - $timestamp([YMDHMS Format]) – Delivery timestamp. Desired format to be supplied in Java SimpleDateFormat compatible format.
   - $filename – Name of the file produced by the export.
   - $extension – The extension of the file produced by the outbound integration endpoint.

If the blob containers and virtual directories do not already exist, the plugin will create them.

Important: Existing files with the same virtual directory path and file name will be overwritten.

4. Click **Finish** to display the Save Export Configuration window as defined in **Running a Data Export**.

Cloud Blob Storage Delivery Method

The Cloud Blob Storage Delivery method delivers exported files to cloud storage. Amazon S3, Google Cloud Storage, and Microsoft Azure are supported. This delivery method is available in Export Manager and OIEPs.

Note: The Cloud Blob Storage Delivery Method is similar to the legacy Azure Blob Storage Delivery Method. The difference between the setup and functionalities is that the Cloud Blob Storage Delivery plugin is part of the STEP baseline, can also be used for Amazon S3 and Google Cloud Storage, and the Export Manager and OIEP configurations use the gateway integration endpoints versus having separate delivery method integration properties.

To use this method in an OIEP refer to the **Cloud Blob Storage Delivery Method** topic.

Prerequisites

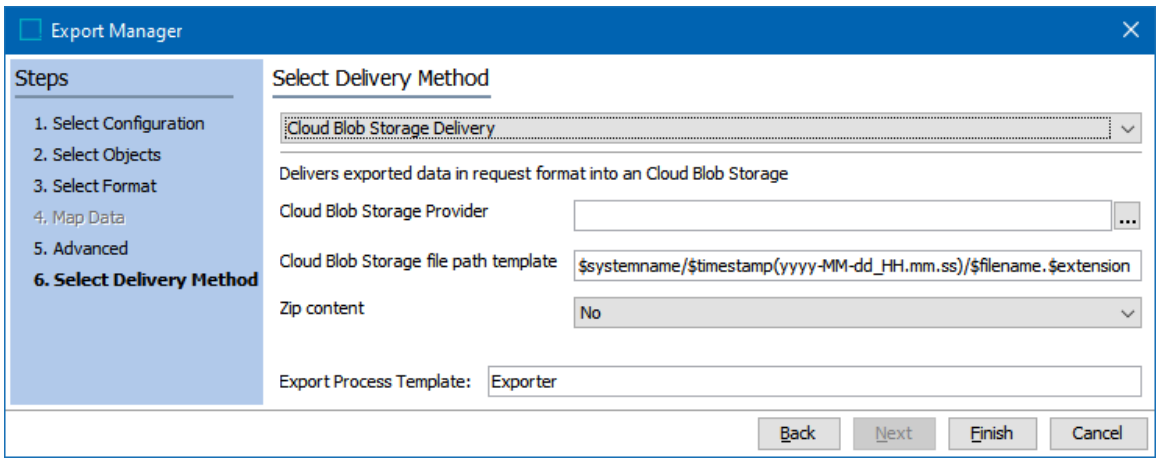
A blob storage gateway integration endpoint must be configured before moving ahead with the Export Manager or OIEP setup.

- For Amazon S3, directions for setting up this endpoint are in the **Configuring a Gateway Integration Endpoint - Amazon S3 Blob Storage** topic in the **Data Exchange** documentation.
- For Google Cloud Storage (GCS), directions for setting up this endpoint can be found in the **Configuring a Gateway Integration Endpoint - Google Cloud Storage** topic in the **Data Exchange** documentation.
- For Microsoft Azure (ABS), directions for setting up this endpoint can be found in the **Configuring a Gateway Integration Endpoint - Microsoft Azure Blob Storage** topic in the **Data Exchange** documentation.

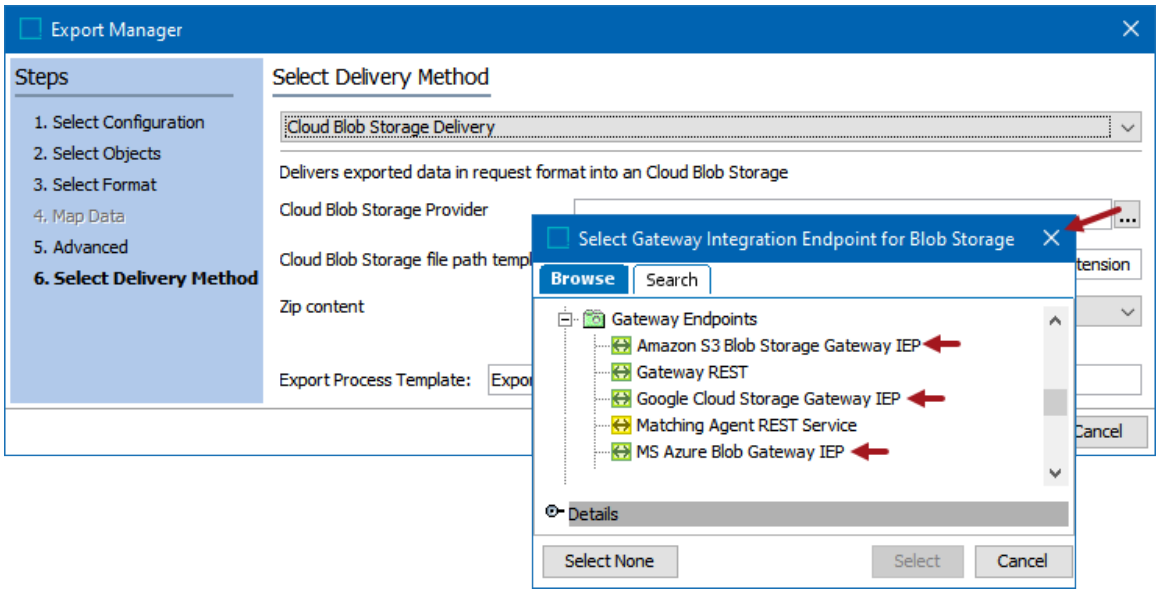
Also, buckets (S3 and GCS) and containers (ABS) must be created. If they do not exist in advance, you will get an exception error during the delivery process.

Configuring the Export Manager Delivery

For information on a parameter, hover over the parameter label to display help text.

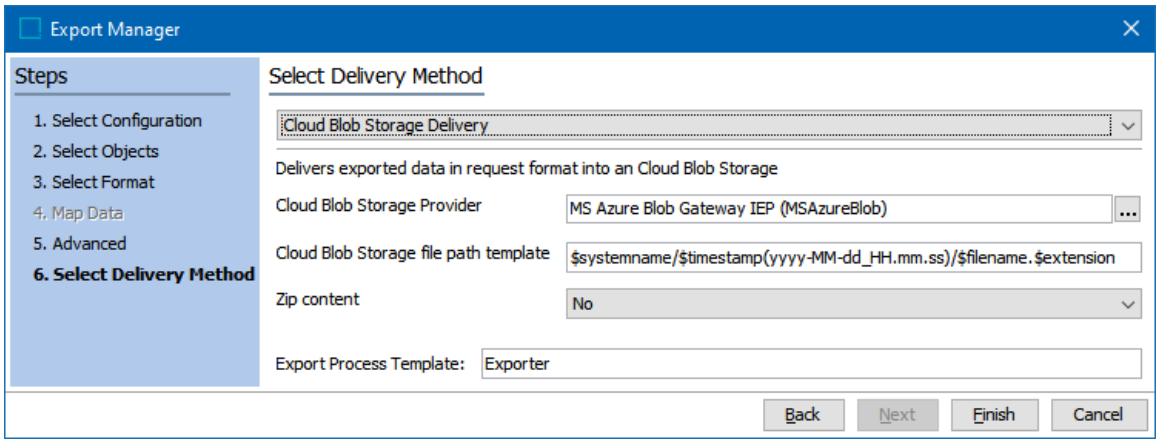


1. From the **Select Delivery Method** parameter dropdown, choose **Cloud Blob Storage Delivery**.
2. Click the ellipsis button (...) to the right of the **Cloud Blob Storage Provider** parameter, select a gateway endpoint configured for blob storage, and click the **Select** button. The name and ID of the selection display in the Edit Delivery Configuration dialog.



3. Configure the **Blob Storage file path template** using the available variables:
   - $systemname – The STEP system name. Useful when several STEP systems are delivering to the same blob container.
   - $timestamp([YMDHMS Format]) – Delivery timestamp. Desired format to be supplied in Java SimpleDateFormat compatible format.

- $filename – Name of the file produced.
- $extension – The extension of the file produced.



In addition to selecting the appropriate connection string and container, you can make use of Azure Blob Storage virtual directories. If the virtual directories do not already exist, the plugin creates them. As mentioned in the **Prerequisites** section, buckets (S3 and GCS) and containers (ABS) must exist in advance, or you will get an exception error during the delivery.

Important: Existing files with the same virtual directory path and file name are overwritten.

4. In **Zip content**, select 'yes' or 'no' from the dropdown to specify if the output file should be delivered in a .ZIP (compressed) file format.
   - **Yes** uses 'export-' before the timestamp variable, and then the extension ZIP. For example, a zipped STEPXML output with the default File Name Template exported on 15 NOV 2016 results in an output .ZIP file named 'export-1479230247017.zip.' The contents of the ZIP file follow the $filename variable applied in the 'File name template' parameter.
   - **No** uses the file type for the file name along with the appropriate extension for the selected data format.
5. Click **Finish** to display the Save Export Configuration window as defined in **Running a Data Export**.

Email Delivery Method

Similar to the 'Copy to directory' delivery method, the email delivery option allows the delivery to be zipped. This method is typically used to test the export output, or to deliver a small export to one or more persons.

To use this method in an OIEP refer to the **Email Delivery Method** topic.

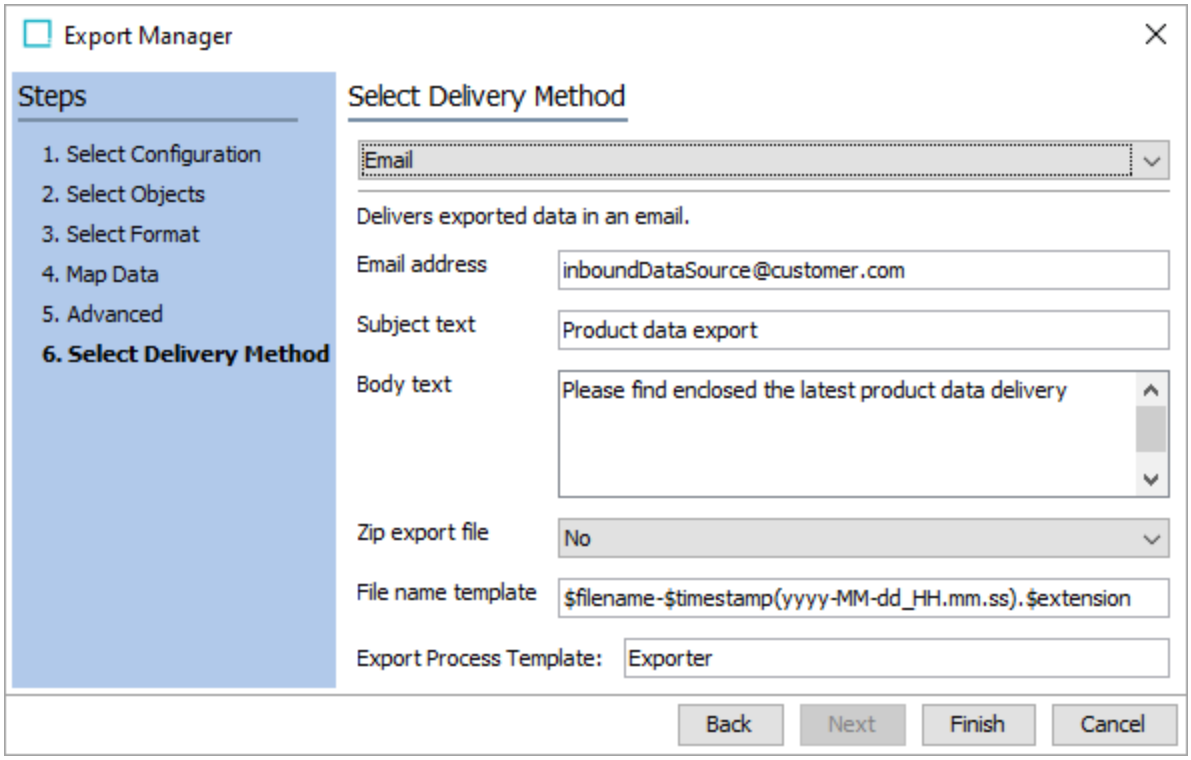
Prerequisites

To deliver an exported file via email, the STEP server must have access to the SMTP server. Be aware of email file size limits, since depending on export contents, the file could be very large.

For information on configuring email from STEP, refer to the **Email from STEP** topic in the **Resource Materials** of online help.

Configuration

For information on a parameter, hover over the parameter label to display help text.



1. Click the **Select Delivery Method** parameter to display the dropdown and choose **Email**.
2. For the **Email** parameter, enter an email address. For multiple recipients, enter the email addresses separated by a semicolon.
3. For the **Subject text** parameter, enter the text that will display for the email subject line. The server name will prepend the subject text for the email.

4. For the **Body text** parameter, enter the text that will display for the email body.
5. In **Zip export file**, select an option from the dropdown to specify if the output file should be delivered as an email attachment in a .ZIP (compressed) file format.
   - **Yes** zips the export file.
   - **No** does not zip the export file.
6. In **File name template**, to create the file name used for the output, enter template variables (identified with a $), static text, or use a combination. The default template is $filename-$timestamp(yyyy-MM-dd_HH.mm.ss).$extension and creates a file name like 'GenericXML-2020-07-30_14.07.44.xml' based on the settings of the configuration. The hyphens (-), underscore (_), and the periods (.) are actual characters that are used to build the complete name of the exported file. Each variable is described below:
   - **$filename**: This variable is replaced with text to indicate the format of the delivered file as specified in the Output Templates flipper. For example, the output file name would include the text 'excel' or 'GenericXML' for those formats, or the Event ID for event-based STEPXML. Using the default file name template, a comma-separated value file would be named 'csv--2020-07-30_14.07.44.csv' and 'GenericXML--2020-07-30_14.07.44.xml' would indicate that Generic XML was used.
   - **$timestamp**: This variable is replaced with the time in year-day-month-hour-minutes-seconds-milliseconds, excluding the dashes. For example, 20161511094709625 is used for an export on 15 NOV 2016 at 9:47:09:625 AM.

     Adding a definition for the arrangement of the date / time elements provides a way to generate a more readable date. Static characters can be included to separate the timestamp elements and individual elements can be ordered as required, as shown below.

     Note: To revert the **$timestamp** variable so that it uses the legacy definition (milliseconds between January 1, 1970, and the time when the file is uploaded), contact Stibo Systems Support for more information.

     $timestamp(ddMMyyyy): The timestamp is modified to display day-month-year (excluding dashes) and is output as '15112016' for 15 NOV 2016.

     $timestamp(yyyy-MM-dd_HH.mm.ss): The timestamp is modified to display year-month-day_hour.minutes.seconds (including the dashes, underscore, and periods) and is output as '2016-11-15_09.32.43' for 2016 NOV 15 at 9:32.43 AM.
   - **$extension**: This variable is replaced with the extension of the output file based on the selected format in the Select Format step. For XML-based formats, the output file is set with the .XML extension, while Excel files use .XLS, and comma-separated values use .CSV.

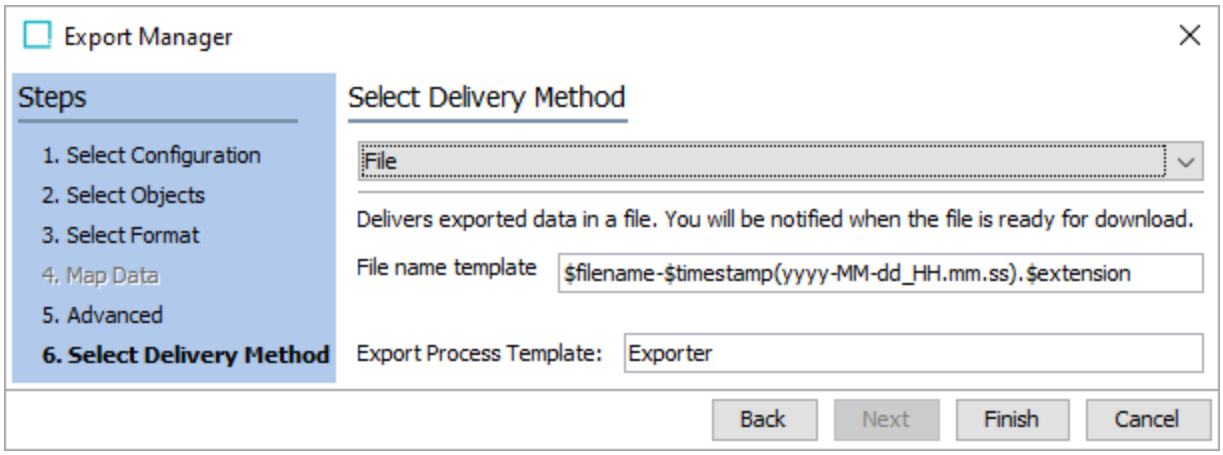
   Note: The File Name Template does not support conversions of file formats and can only be used to deliver files in the format specified in the configuration.
7. For the **Export Process Template** parameter, leave the default setting, which is required to start the background process for the export.
8. Click **Finish** to display the Save Export Configuration window as defined in **Running a Data Export**.

File Delivery Method

The file delivery option is only available in Export Manager. It makes the output available within STEP, via a background process.

Configuration

For information on a parameter, hover over the parameter label to display help text.



1. Select **File** from the dropdown.
2. In **File name template**, to create the file name used for the output, enter template variables (identified with a $), static text, or use a combination. The default template is $filename-$timestamp(yyyy-MM-dd_HH.mm.ss).$extension and creates a file name like 'GenericXML-2020-07-30_14.07.44.xml' based on the settings of the configuration. The hyphens (-), underscore (_), and the periods (.) are actual characters that are used to build the complete name of the exported file. Each variable is described below:
   - **$filename**: This variable is replaced with text to indicate the format of the delivered file as specified in the Output Templates flipper. For example, the output file name would include the text 'excel' or 'GenericXML' for those formats, or the Event ID for event-based STEPXML. Using the default file name template, a comma-separated value file would be named 'csv--2020-07-30_14.07.44.csv' and 'GenericXML--2020-07-30_14.07.44.xml' would indicate that Generic XML was used.
   - **$timestamp**: This variable is replaced with the time in year-day-month-hour-minutes-seconds-milliseconds, excluding the dashes. For example, 20161511094709625 is used for an export on 15 NOV 2016 at 9:47:09:625 AM.

     Adding a definition for the arrangement of the date / time elements provides a way to generate a more readable date. Static characters can be included to separate the timestamp elements and individual elements can be ordered as required, as shown below.

     Note: To revert the **$timestamp** variable so that it uses the legacy definition (milliseconds between January 1, 1970, and the time when the file is uploaded), contact Stibo Systems Support for more information.

$timestamp(ddMMyyyy): The timestamp is modified to display day-month-year (excluding dashes) and is output as '15112016' for 15 NOV 2016.

$timestamp(yyyy-MM-dd_HH.mm.ss): The timestamp is modified to display year-month-day_ hour.minutes.seconds (including the dashes, underscore, and periods) and is output as '2016-11-15_ 09.32.43' for 2016 NOV 15 at 9:32.43 AM.

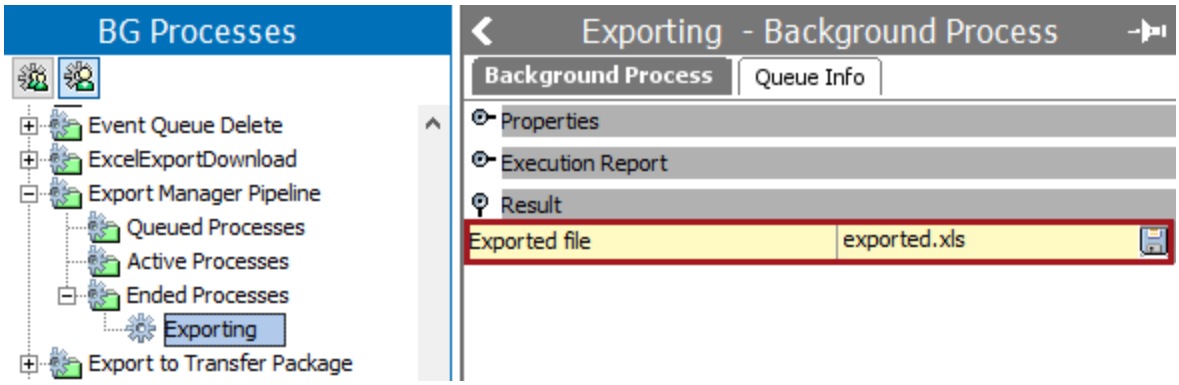
- **$extension**: This variable is replaced with the extension of the output file based on the selected format in the Select Format step. For XML-based formats, the output file is set with the .XML extension, while Excel files use .XLS, and comma-separated values use .CSV.

Note: The File Name Template does not support conversions of file formats and can only be used to deliver files in the format specified in the configuration.

3. For the **Export Process Template** parameter, leave the default setting, which is required to start the background process for the export.
4. Click **Finish** to display the Save Export Configuration window as defined in **Running a Data Export**.

Output

By default, the output is saved on the Background Processes tab, under the Export Manager Pipeline node, which can be easily accessed as described in **Monitoring a Data Export**.



FTP Delivery Method

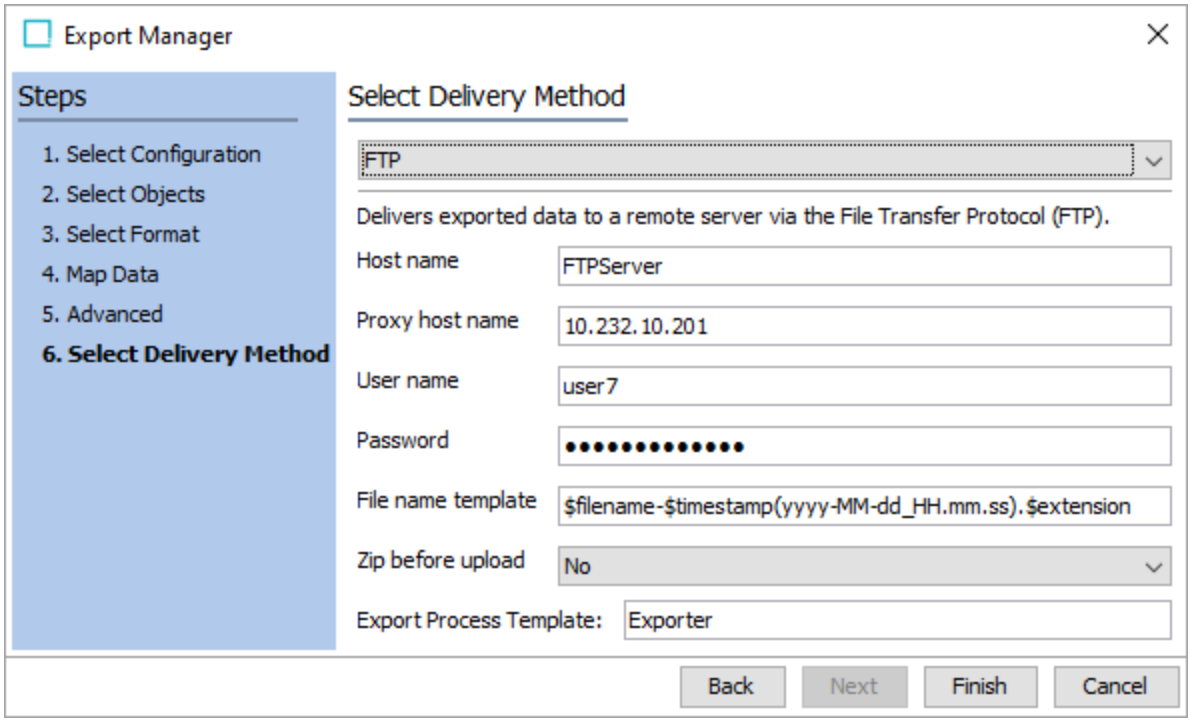
The FTP delivery method allows an exported file to be delivered to an external system and is often used when the output files are large, or when a different or remote system is in use. The preferred method for FTP delivery is the SFTP secure delivery method.

For information on the SFTP delivery method, refer to the **SFTP Delivery Method** topic.

To use this method in an OIEP refer to the **FTP Delivery Method** topic.

Configuration

For information on a parameter, hover over the parameter label to display help text.



1. In **Select Delivery Method**, choose **FTP** from the dropdown.
2. In **Host name**, enter the host name or IP address of the FTP server to be used for the delivery.
3. In **Proxy host name**, add the host name to be used for the server proxy. This field is optional.
4. In **User name**, enter the user name that has access to log on to the FTP server.
5. In **Password**, enter the password that will be used to log on to the FTP server.
6. In **File name template**, to create the file name used for the output, enter template variables (identified with a $), static text, or use a combination of both. The default template is $filename-$timestamp(yyyy-MM-dd_HH.mm.ss).$extension. Depending on the settings of the configuration, the default template outputs a file name similar to 'GenericXML-2020-07-30_14.07.44.xml.' The hyphens (-), underscore (_), and the periods (.) are actual characters that are used to build the complete name of the exported file. Each variable is described below:

- **$filename**: This variable is replaced with text to indicate the format of the delivered file as specified in the Output Templates flipper. For example, the output file name would include the text 'excel' or 'GenericXML' for those formats, or the Event ID for event-based STEPXML. Using the default file name template, a comma-separated value file would be named 'csv-1490808860412.csv' and 'GenericXML-1490810861593.xml' would indicate that Generic XML was used.
- **$timestamp**: This variable is replaced with the time in year-day-month-hour-minutes-seconds-milliseconds, excluding the dashes. For example, 20161511094709625 is used for an export on 15 NOV 2016 at 9:47:09:625 AM.

  Adding a definition for the arrangement of the date / time elements provides a way to generate a more readable date. Static characters can be included to separate the timestamp elements and individual elements can be ordered as required, as shown below.

  Note: To revert the **$timestamp** variable so that it uses the legacy definition (milliseconds between January 1, 1970, and the time when the file is uploaded), contact Stibo Systems Support for more information.

  $timestamp(ddMMyyyy): The timestamp is modified to display day-month-year (excluding dashes) and is output as '15112016' for 15 NOV 2016.

  $timestamp(yyyy-MM-dd_HH.mm.ss): The timestamp is modified to display year-month-day_ hour.minutes.seconds (including the dashes, underscore, and periods) and is output as '2016-11-15_ 09.32.43' for 2016 NOV 15 at 9:32.43 AM.

- **$extension** This variable is replaced with the extension of the output file based on the selected format in the Output Templates flipper. For XML-based formats, the output file is set with the .XML extension, while Excel files use .XLS, and comma-separated values use .CSV.

Note: The File Name Template does not support conversions of file formats and can only be used to deliver files in the format specified in the configuration.

7. In **Zip export file**, select an option from the dropdown to specify if the output file should be delivered in a .ZIP (compressed) file format.
   - **Yes** uses the File Name Template and the extension .ZIP. When a file is zipped, the File Name Template is used for the .ZIP file. The $filename template is replaced with 'result_0', and the selected file type is compressed with the name 'exported.' For example, a zipped STEPXML output with the default File Name Template exported on 15 NOV 2016 results in an output .ZIP file named 'result_0-1479230247017.zip.'
   - **No** uses the File Name Template for the file name along with the appropriate extension for the selected data format.
8. In **Export Process Template**, leave the default setting, which is required to start the background process for the export.
9. Click the **Finish** button to complete the wizard and begin the export as described in **Running a Data Export**.

JDBC Delivery Method

The JDBC delivery method allows STEP data to be delivered directly to tables in Relational database management systems (RDBMS) like Oracle, MySQL, MS SQL Server, and PostgreSQL. Though the ways the JDBC plugin can be deployed are various, one of its prime uses is to send STEP data to data analytics tools for the purpose of displaying STEP data in a data analytics dashboard. This delivery method requires CSV format and is available in both the Export Manager and in outbound integration endpoints (OIEPs).

To use this delivery method in an OIEP refer to **JDBC Delivery Method** topic.

Prerequisites

Important: For complete setup requirements, refer to the **Exporting Data via JDBC with CSV Format** topic in the **Data Format** documentation.

Install the required drivers

JDBC specification 4.1-compliant drivers should be placed in a directory accessible from all application servers. These drivers can then be made available for the delivery plugin via the dynamic properties JDBCDeliveryPlugin.DriverPath.[n] and JDBCDeliveryPlugin.DriverClass.[n]. For more information regarding applicable Java drivers, review the RDBMS vendor's homepage on the web.

Configure data for the dropdown parameters

1. Prior to configuration, clicking the **Driver Location** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **JDBCDeliveryPlugin.DriverPath.[n]** property. As an example:

   ```
   JDBCDeliveryPlugin.DriverPath.1 = L:/shared/mysql-connector-java-5.1.42-bin.jar
   ```

   In this example, the drivers are stored on the application server's L:/shared drive.

2. Prior to configuration, clicking the **Driver Class** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **JDBCDeliveryPlugin.DriverClass.[n]** property. As an example:

   ```
   JDBCDeliveryPlugin.DriverClass.1 = com.mysql.jdbc.Driver
   ```

   In this example, the drivers class used is 'com.mysql.jdbc.Driver.'

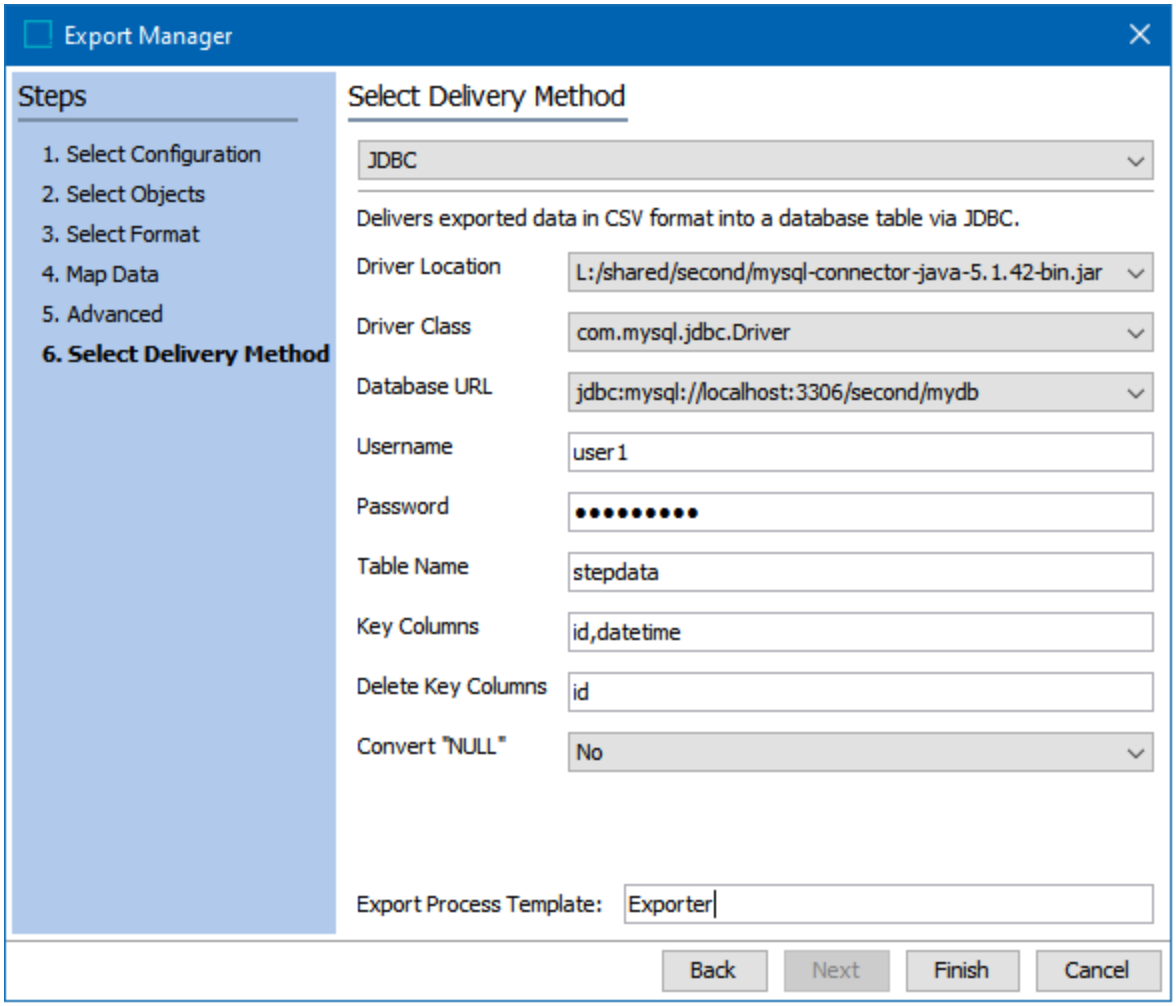
3. Prior to configuration, clicking the **Database URL** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **JDBCDeliveryPlugin.URL.[n]** property.

   ```
   JDBCDeliveryPlugin.URL.1 = jdbc:mysql://localhost:3306/mydb
   ```

   In this example, the database URL used is 'jdbc:mysql://localhost:3306/mydb.'

Configuration

For information on a parameter, hover over the parameter label to display help text.



1. In **Select Delivery Method**, choose **JDBC** from the dropdown.
2. In **Driver Location**, select from the dropdown one of the paths to the relevant .jar file.
3. In **Driver Class**, select from the dropdown one of the pre-configured driver class.
4. In **Database URL**, select from the dropdown one of the pre-configured URLs to the destination database.
5. In **Username**, enter the username required to access the destination database.
6. In **Password**, enter the password required to access the destination database.
7. In **Table Name**, enter the name of the table in the destination database to which STEP will be publishing data
8. In **Key Columns**, list the names of the columns appearing on both the exported CSV file and the table in the destination database, separated by commas (and no spaces), into which STEP will publish data

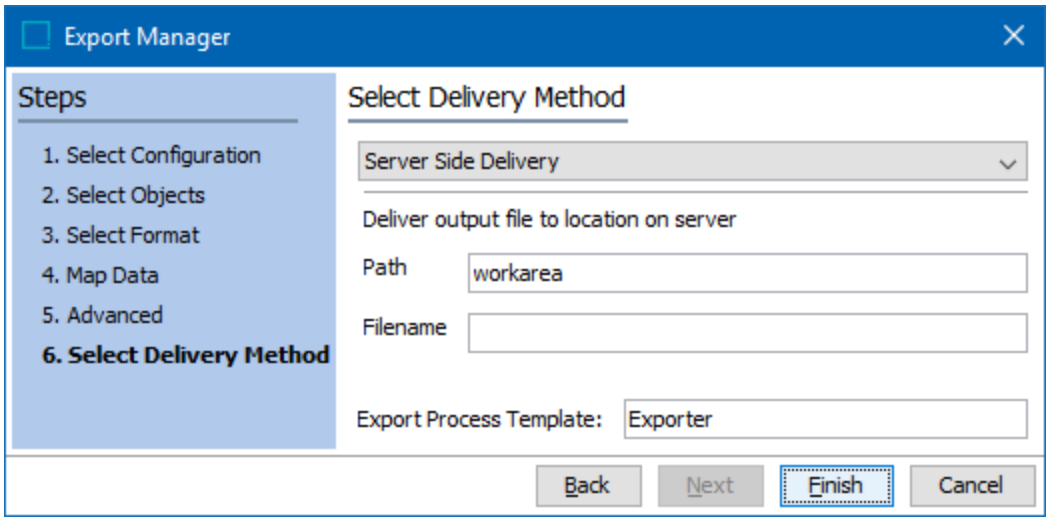
9. In **Delete Key Columns**, list the names of the columns appearing on both the exported CSV file and the table in the destination database, separated by commas (and no spaces), from which STEP will delete data. The headers contained in this field can differ from the headers in the 'Key Columns' field, but they must also be part of the upsert key definition.
10. In **Convert "NULL"**, choose Yes if the string “NULL” should be converted to the value null. This may, for instance, be used for clearing a value in a column in the target database. This parameter defaults to No.
11. In **Export Process Template**, leave the default setting, which is required to start the background process for the export.
12. Click the **Finish** button to complete the wizard and begin the export as described in **Running a Data Export**.

Server Side Delivery Method

The server side delivery option is only available in Export Manager and saves the output file on the STEP application server via a background process. This delivery method is typically used to export a large file to a specified location, while an external system (consumer of the data) is looking for an input file in this specified location.

Configuration

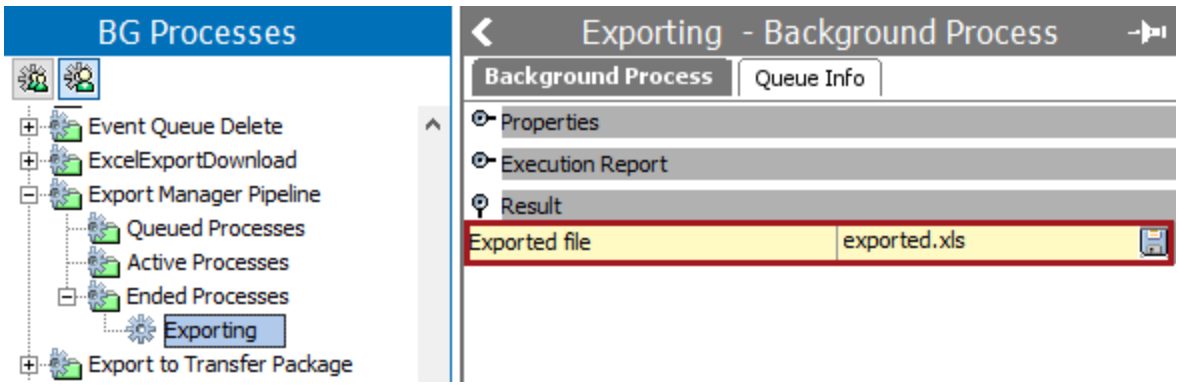
For information on a parameter, hover over the parameter label to display help text.



1. Select **Server Side Delivery** from the dropdown.
2. For **Path**, specify the application server directory where the file should be delivered. The location of the delivery is included in the background process execution report. In the image above, 'workarea' would indicate that the output would be saved at <root>\workarea.
3. For the **Export Process Template** parameter, leave the default setting, which is required to start the background process for the export.
4. Click **Finish** to display the Save Export Configuration window as defined in **Running a Data Export**.

Output

By default, the output is saved on the Background Processes tab, under the Export Manager Pipeline node, which can be easily accessed as described in **Monitoring a Data Export**.


BG Processes
Event Queue Delete
ExcelExportDownload
Export Manager Pipeline
Queued Processes
Active Processes
Ended Processes
Exporting
Export to Transfer Package
Exporting - Background Process
Background Process
Queue Info
Properties
Execution Report
Result
Exported file
exported.xls

SFTP Delivery Method

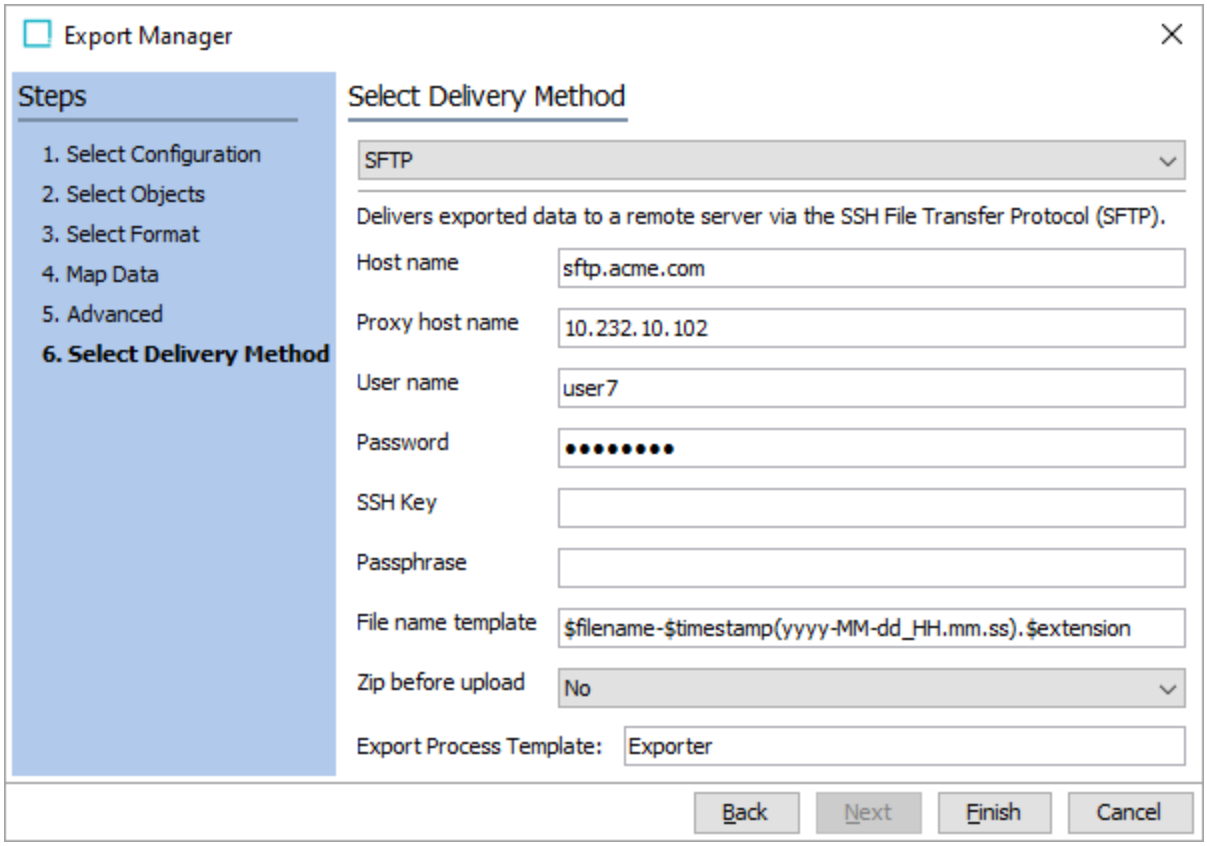
The preferred method for file transfer protocol (FTP) delivery is the SFTP secure delivery method. The SFTP delivery method allows an exported file to be delivered to an external system and is often used when the output files are large or when a different or remote system is in use.

For information on the FTP delivery method, refer to **FTP Delivery Method**.

To use this method in an OIEP refer to the **SFTP Delivery Method** topic.

Configuration

For information on a parameter, hover over the parameter label to display help text.



1. In **Select Delivery Method**, choose **SFTP** from the dropdown.
2. In **Host name**, enter the host name of the SFTP server to be used for the delivery.
3. In **Proxy host name**, add the host name to be used for the server proxy. This field is optional.
4. In **User name**, enter the user name that has access to log on to the FTP server.
5. In **Password**, enter the password that will be used to log on to the FTP server. If using the SSH Key, then leave this field blank.

6. In **SSH Key**, enter the full path to the Secure Shell (SSH) key file, if using. If you filled in the Password field in step 5, then leave this field blank.

   Note: This SSH key must be an **RSA** private key in the **OpenSSH PEM** format. Other SSH key types or those generated in the new OpenSSH format (the default used in OpenSSH 7.8+) are not currently supported and will result in an **SFTPException: invalid privatekey** error.

7. In **Passphrase**, enter the passphrase that accompanies the SSH key entered, if applicable. If the SSH key does not have a passphrase or you are not using an SSH key, then leave this field blank.
8. In **File name template**, to create the file name used for the output, enter template variables (identified with a $), static text, or use a combination of both. The default template is $filename-$timestamp(yyyy-MM-dd_HH.mm.ss).$extension. Depending on the settings of the configuration, the default template outputs a file name similar to 'GenericXML-2020-07-30_14.07.44.xml.'it The hyphens (-), underscore (_), and the periods (.) are actual characters that are used to build the complete name of the exported file. Each variable is described below:
   - **$filename** This variable is replaced with text to indicate the format of the delivered file as specified in the Output Templates flipper. For example, the output file name would include the text 'excel' or 'GenericXML' for those formats, or the Event ID for event-based STEPXML. Using the default file name template, a comma-separated value file would be named 'csv-1490808860412.csv' and 'GenericXML-1490810861593.xml' would indicate that Generic XML was used.
   - **$timestamp**: This variable is replaced with the time in year-day-month-hour-minutes-seconds-milliseconds, excluding the dashes. For example, 20161511094709625 is used for an export on 15 NOV 2016 at 9:47:09:625 AM.

     Adding a definition for the arrangement of the date / time elements provides a way to generate a more readable date. Static characters can be included to separate the timestamp elements and individual elements can be ordered as required, as shown below.

     Note: To revert the **$timestamp** variable so that it uses the legacy definition (milliseconds between January 1, 1970, and the time when the file is uploaded), contact Stibo Systems Support for more information.

     $timestamp(ddMMyyyy): The timestamp is modified to display day-month-year (excluding dashes) and is output as '15112016' for 15 NOV 2016.

     $timestamp(yyyy-MM-dd_HH.mm.ss): The timestamp is modified to display year-month-day_hour.minutes.seconds (including the dashes, underscore, and periods) and is output as '2016-11-15_09.32.43' for 2016 NOV 15 at 9:32.43 AM.

   - **$extension** This variable is replaced with the extension of the output file based on the selected format in the Output Templates flipper. For XML-based formats, the output file is set with the .XML extension, while Excel files use .XLS, and comma-separated values use .CSV.

   Note: The File Name Template does not support conversions of file formats and can only be used to deliver files in the format specified in the configuration.

Important: When a file needs to be sent to a specific directory the full file path must be added to the front of the file name template, e.g., /upload/hotfolders/GDSN Receiver Inbound/In/$filename-$timestamp (yyyy-MM-dd_HH.mm.ss).$extension.

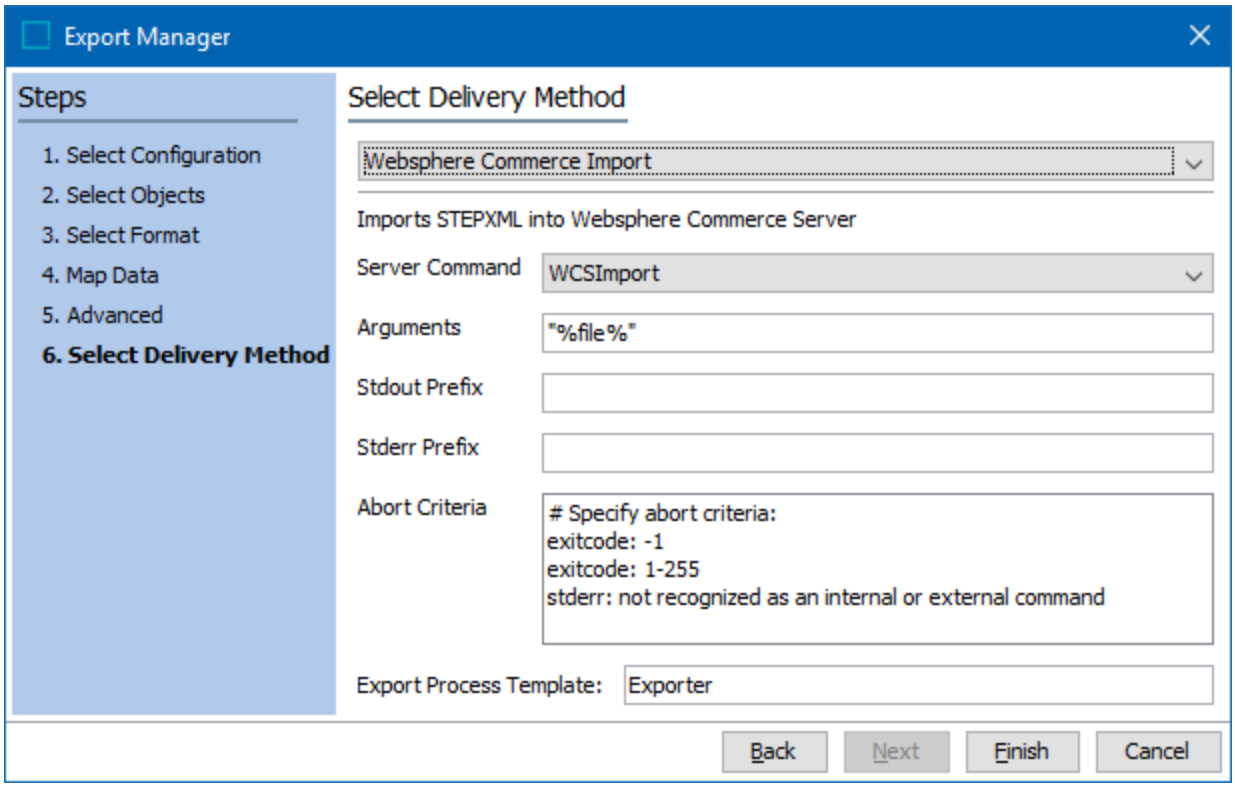
9. In **Zip before upload**, select an option from the dropdown to specify if the output file should be delivered in a .ZIP (compressed) file format.
   - **Yes** uses the File Name Template and the extension .ZIP. When a file is zipped, the File Name Template is used for the .ZIP file. The $filename template is replaced with 'result_0', and the selected file type is compressed with the name 'exported.' For example, a zipped STEPXML output with the default File Name Template exported on 15 NOV 2016 results in an output .ZIP file named 'result_0-1479230247017.zip.'
   - **No** uses the File Name Template for the file name along with the appropriate extension for the selected data format.
10. In **Export Process Template**, leave the default setting, which is required to start the background process for the export.
11. Click the **Finish** button to complete the wizard and begin the export as described in **Running a Data Export**.

WebSphere Commerce Import Delivery Method

The file delivery option is only available in Export Manager. It delivers a STEPXML file to a WebSphere Commerce server.

Configuration

For information on a parameter, hover over the parameter label to display help text.



1. Select **Websphere Commerce Import** from the dropdown.
2. For each of the available parameters, add the required information.
3. For the **Export Process Template** parameter, leave the default setting, which is required to start the background process for the export.
4. Click **Finish** to display the Save Export Configuration window as defined in **Running a Data Export**.

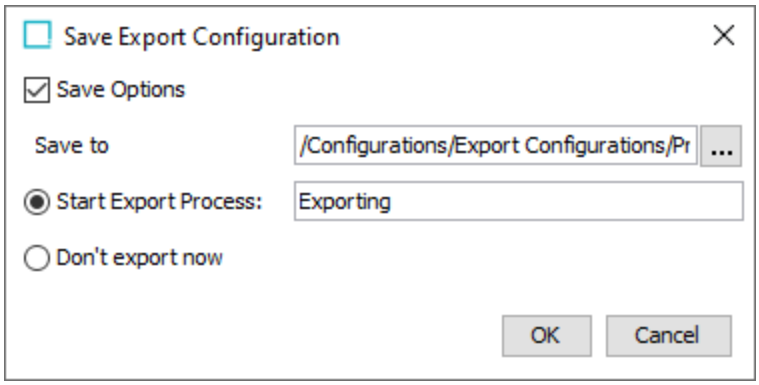
Running a Data Export

The Save Export Configuration dialog appears when you click the Finish button in the Export Manager in workbench. A saved configuration provides a way to keep all the parameter selections set while using the export wizard. Saving a configuration allows you to perform the same export again, without having to go through each step of the wizard. A saved configuration is required when scheduling exports in workbench and also when executing an export in Web UI. Whether you save your configuration or not, this dialog also allows you to start the export process.

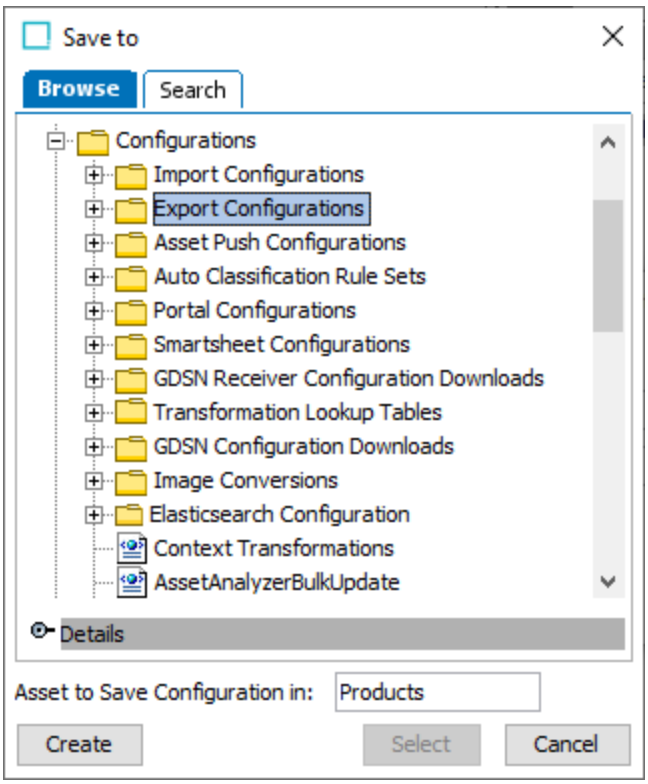
Run an Export Configuration

Follow these steps to save and/or run an export configuration.

1. Set the **Save Options** checkbox:
   - Checked (default) is required to save changes to an existing or new configuration
   - Uncheck to run or close this configuration without saving it.



2. If the Save Options checkbox is checked, click the ellipsis button (...) on the Save to parameter to display the **Save to** dialog.



3. On the **Save to** dialog, choose the option for your situation.
   - To save a new configuration: use the Browse or Search option to select the folder for the new configuration. Type the name of the new configuration in the **Asset to Save Configuration in** parameter, and click the **Create** button that is now enabled.
   - To update an existing configuration: use the Browse tab or the Search tab to select the configuration folder (to hold a newly created configuration) or a file (to overwrite an existing configuration) and click the **Select** button which becomes enabled.
   - To close this dialog without changes: click **Cancel**.

4. Back on the Save Export Configuration dialog, choose a radio button:
   - Select **Start Export Process** and add a name in the field to make it easier to locate the specific process later.
   - Select **Don't export now** to only save changes to the configuration.

5. Click **OK** to complete the wizard. If you started an export, the **Starting Process** dialog appears. For more information, refer to the **Monitoring a Data Export** topic.

Maintaining a Saved Export Configuration

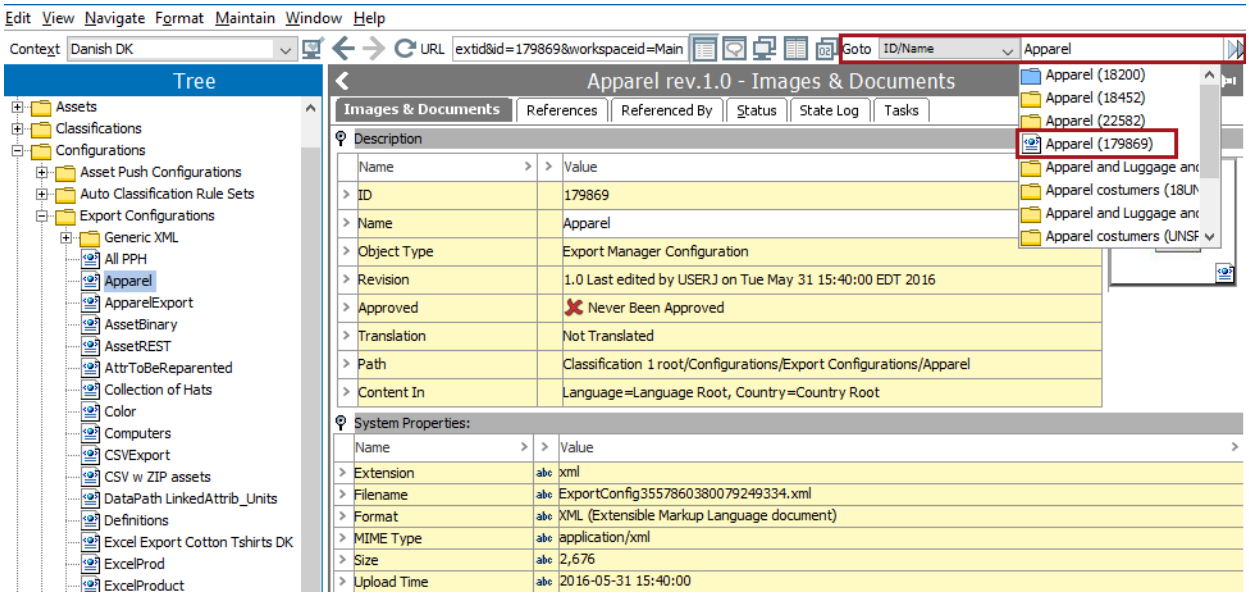
A saved export configuration allows you to repeatedly export a group of objects using the same wizard parameter settings. Completing the Export Manager wizard in workbench includes an option to save an export configuration (refer to **Running a Data Export**). Viewing a saved configuration allows you to confirm the current settings. Editing a saved configuration affects future exports that use the configuration, and can be a valuable step in the process of creating and testing an export to ensure the correct data is output in the proper manner.

Note: An export configuration definition can be exported as comments and submitted to an external source control system for comparison purposes. For details, refer to the **Configuration Management** documentation.

View a Saved Export Configuration

After clicking the Finish button on the Export Manager wizard, the option to save the configuration is displayed. Although you can save an export configuration to any classification folder, using a specific folder for export configurations makes it easier to locate them for review and running.

Note: If you know the name of the configuration, use the **Goto** search option to find it anywhere in the Tree, using the icon to distinguish the configuration from other objects.



The object type of a saved export configuration is Export Manager Configuration. The metadata for the configuration displayed in the System Properties flipper cannot be edited.

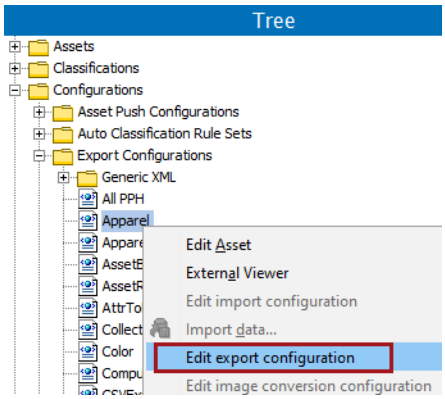
Edit a Saved Export Configuration

Use the following steps to apply updates to the configuration selected for editing.

- To change only the name of the configuration, edit the Name field in the configuration editor.
- To edit an existing configuration and save with a new name, refer to the steps in the **Running a Data Export** section.
- To change the settings of the existing configuration parameters, use the edit option or edit via XML. Both methods are defined below.

Edit export configuration via Export Manager

1. Select an export configuration, right-click and select **Edit export configuration** option.



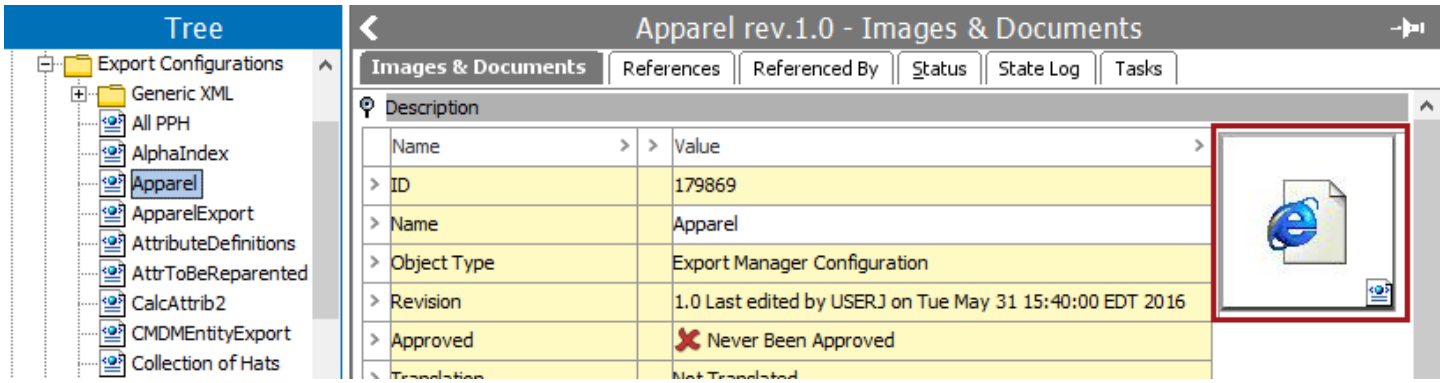
2. The Export Manager wizard opens on the Select Objects step.
3. Make all changes in each step and click the **Finish** button to update the selected configuration.

Edit export configuration via XML

This method should be attempted only if user is well versed with STEPXML. Even a minor error may corrupt the configuration file.

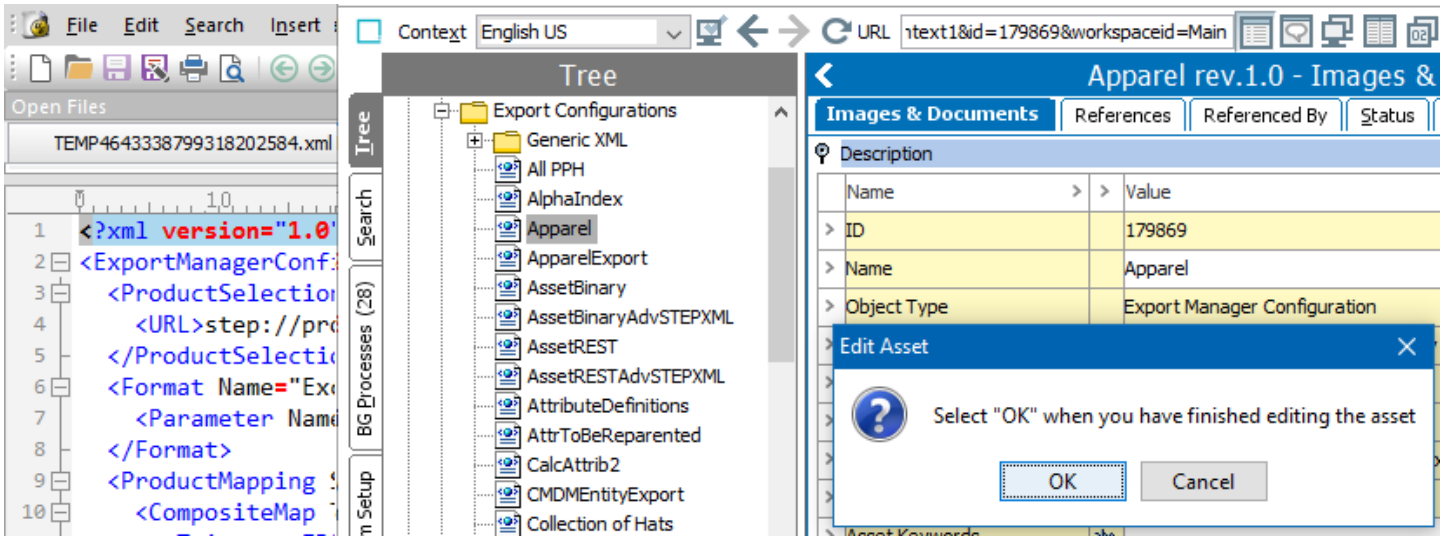
Note: Changing the name of the export configuration via XML file is not allowed.

1. Select an export configuration, double-click the XML file displayed on the Images & Documents tab.



The XML opens in an editor, along with an Edit Asset dialog.

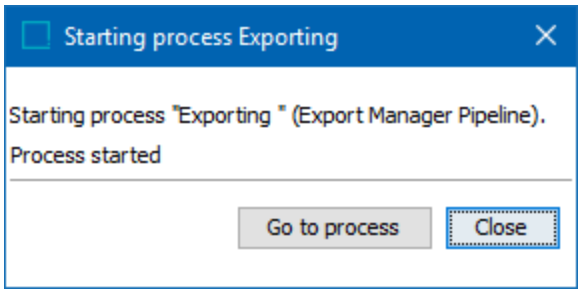
Important: Do not dismiss the Edit Asset dialog until updates to the XML file are complete.



2. Make necessary changes in the XML file editor, save, and close the file.
3. In the Edit Asset dialog, click **OK** to save the changes, or **Cancel** to discard them.

Monitoring a Data Export

When you launch a data export process (as described in the **Running a Data Export** topic), the system starts a background process. It assigns a process number to that particular data export, and gives it the name assigned on the Save Export Configuration dialog. The starting process dialog is displayed:



Monitoring a data export allows users to view the process' activity and status, take action based on the process' result, analyze the execution report for details, and truncate the execution report to eliminate unneeded information.

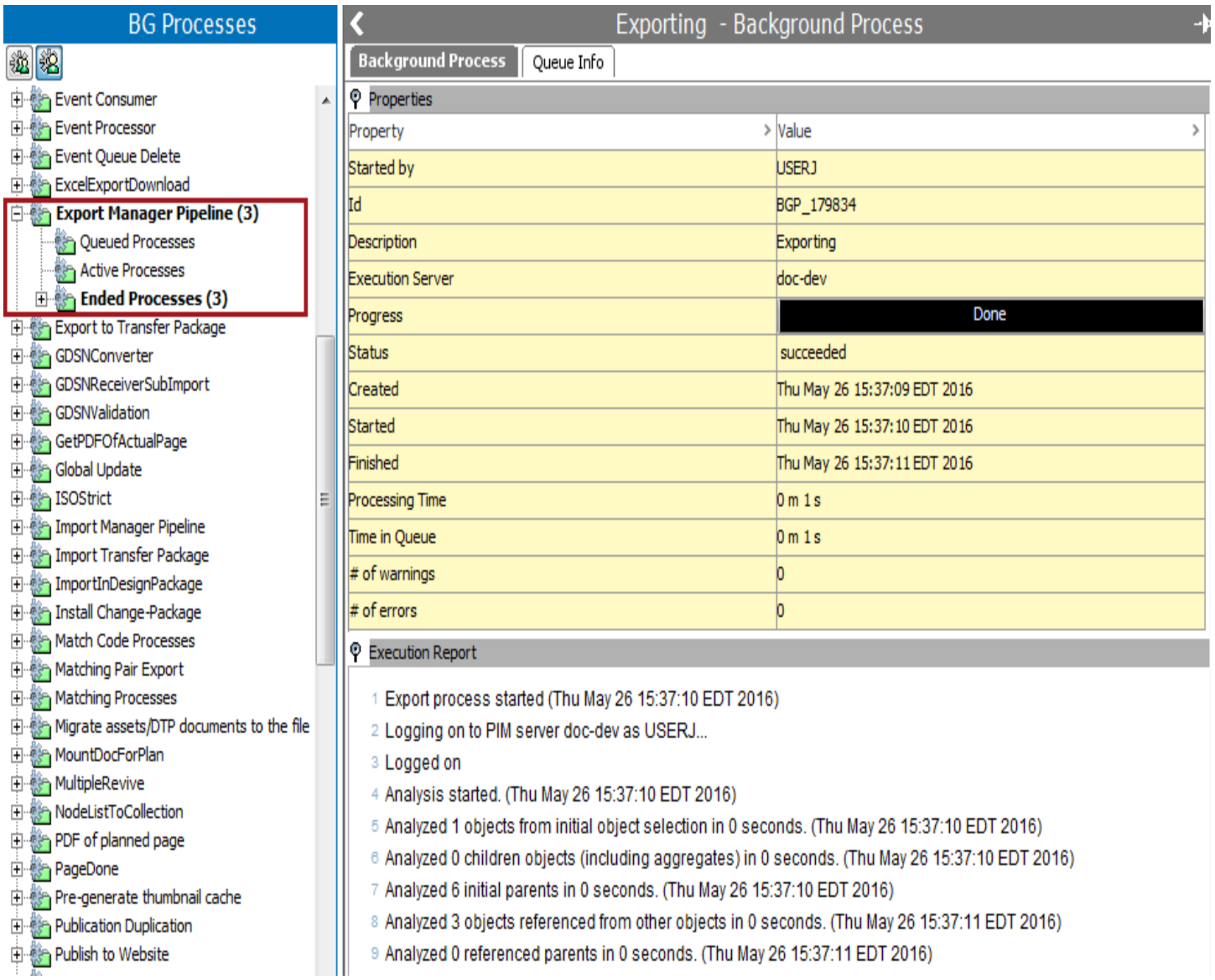
For information on initiating and monitoring BGPs in Web UI, refer to the **Background Process List Screen** topic in the **Web User Interfaces** documentation.

Monitoring the Export Process

Choose a method to monitor a particular data export process:

- In the **Starting Process Exporting** window (that appears when you launch the export), click the **Go To Process** button.
- On the **BG Processes** tab open the **Export Manager Pipeline** node. For details on the states of a background process, refer to the **BG Processes States** topic in the **System Setup** documentation.

The Properties flipper includes details about the export process.



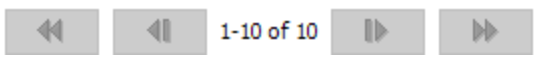
Execution Report Flipper

The execution information includes the status of the process and the number of warnings and errors.

Typically, you do not monitor the process as it is being carried out. However, if the process fails and you need to investigate why or if you need to contact Stibo Systems Help Desk, the information can be useful.

Navigation

After the first entry, the Execution Report displays the following navigation buttons:

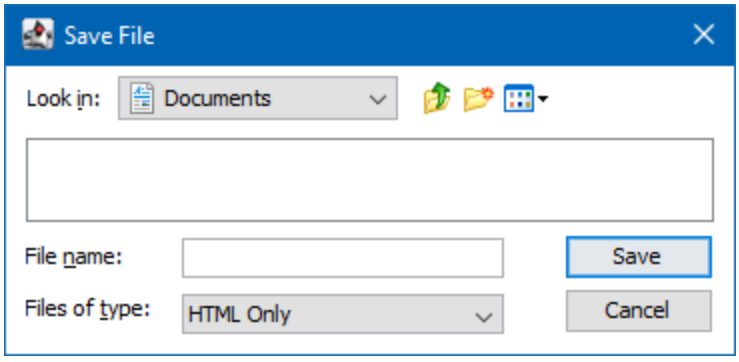


The buttons are enabled when until the log reaches over 100 lines.

- Selecting the button will navigate back one page.
- Selecting the button will navigate to the beginning of the log.
- Selecting the button will navigate forward one page.
- Selecting the button will navigate to the end of the log.

Save

1. Click the Save button (Save...) to save the Execution Report as an 'HTML Only' file type.



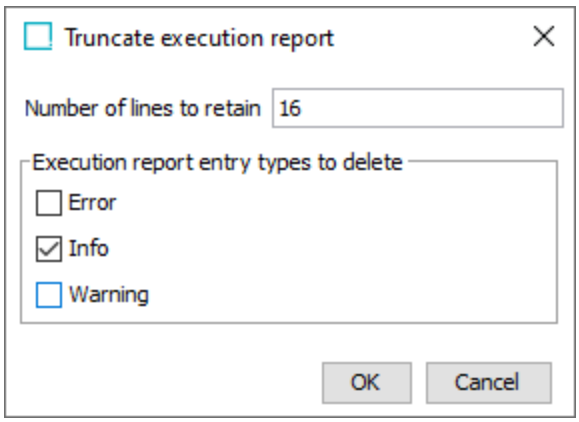
Note: Changing the default format manually may result in an error.

2. Enter a name for the log report and click **Save**.

Truncate

When the Execution Report become cluttered or only specific information is needed, use the Truncate option to permanently eliminate selected rows.

1. Click the Truncate button (Truncate) to display the Truncate execution report dialog:



2. In **Number of lines to retain**, the current number of log entries will display. Type the number of lines to keep starting from the end of the log, and permanently deleting the oldest rows, based on the selected entry types below.
3. In **Execution report entry types to delete**, select the types that should be permanently removed.

> **Note:** The Properties flipper includes a count of the Warnings and Errors included in the Execution Report.

4. Click **OK** to permanently remove the selected number and types of execution report entries.

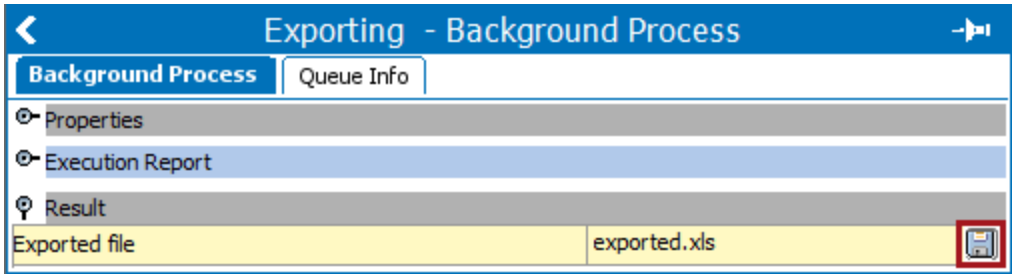
Result Flipper

The selected delivery option determines if you can download the file from the Background Process tab. When available, the file is stored on the STEP application server, and can typically be found at workarea\background-processarea\Exporter, in a folder named for the background process ID (for example, BGP_179834 in the image above).

Depending on your system setup, you may also be able to edit the file from within STEP.

Download an Export File

1. Click the save icon to the right of the file name.



2. In the **Save** window, find the location where to save the file, and click **Save**.

You can now open and edit the file.

View an Export File

Double-click the name of the exported file to open it using an available editor for the output format. In the image above, the file will be opened in Excel.

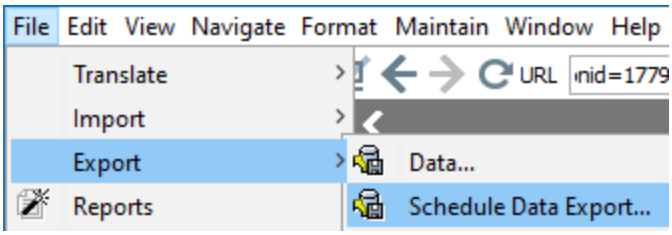
Scheduling a Data Export

Scheduling a data export provides additional functionality for the Export Manager, allowing users to repeat an export with regularity. For example, sending the same data to the same location or recipient on a regular basis, to be consumed by a downstream system. Scheduling requires that an export configuration is saved under any of the classification folders. A saved configuration eliminates the need to set the export parameters each time the data should be output.

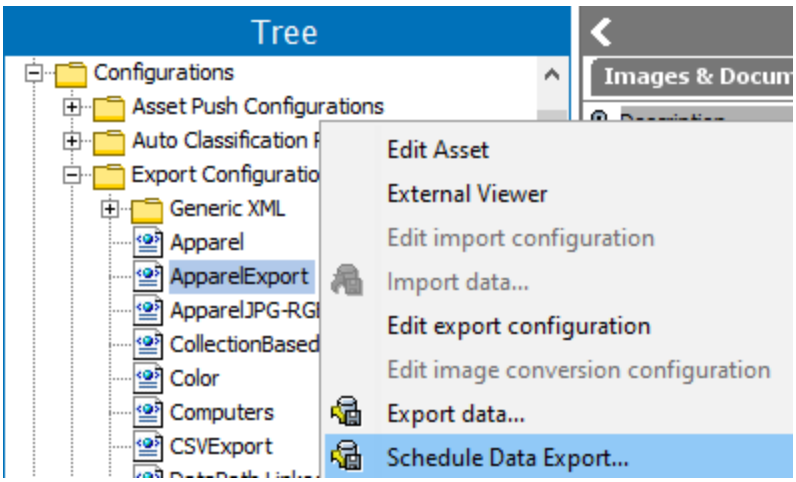
Important: Consider the time zone of the application server compared to that of the workbench (the client) where the schedule is created or viewed. When scheduling a job, the local time zone is displayed in the workbench, but the time zone of the server is used to run the background process. Although displayed, the time zone of the client is not included in the instruction to the server to run the job. This can cause confusion about when the job will run since the scheduled time is not automatically converted to accommodate potential differences in time zones.

Create a Scheduled Export

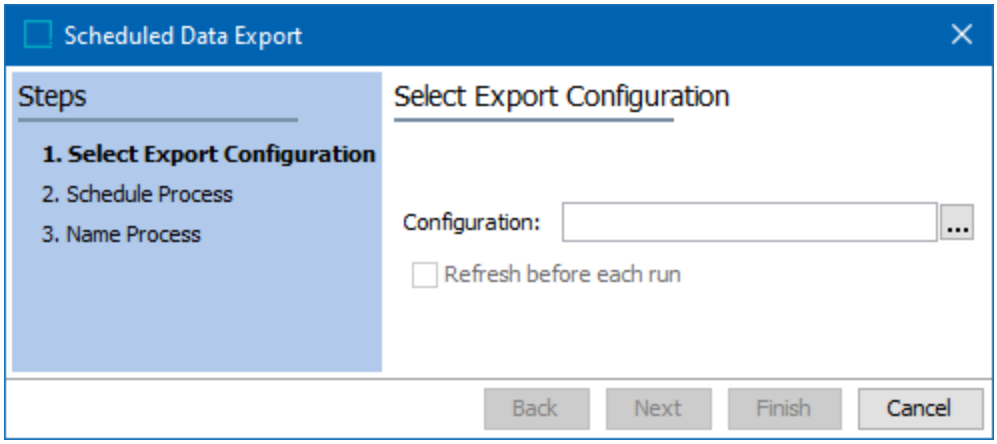
1. Select the appropriate context in STEP.
2. Verify the required Export Configuration already exists for the export you want to schedule. To create a new one, use the steps described in the **Creating a Data Export** documentation and then save the configuration as described in the **Running a Data Export** section.
3. Open the Schedule Data Export wizard using one of these methods:
   - From the File menu > Export > **Schedule Data Export**.



   - In the Tree, select an existing export configuration, right-click, and select **Schedule Data Export**.



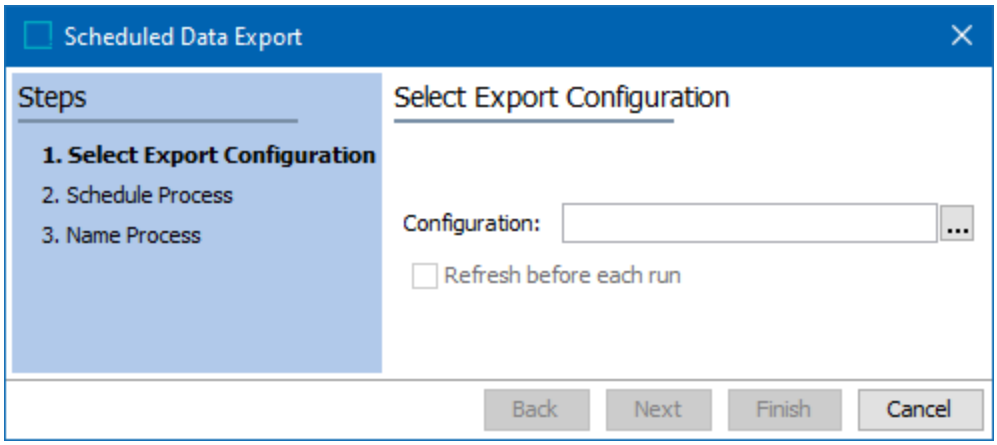
4. After selecting one of the above options to start the Schedule Data Export wizard, the following dialog is displayed and includes these steps:



- **Select Export Configuration** allows you to select the relevant export configuration.
- **Schedule Process** allows you to schedule the export.
- **Name Process** allows you to name the process so that you can locate it on the BG Processes tab.

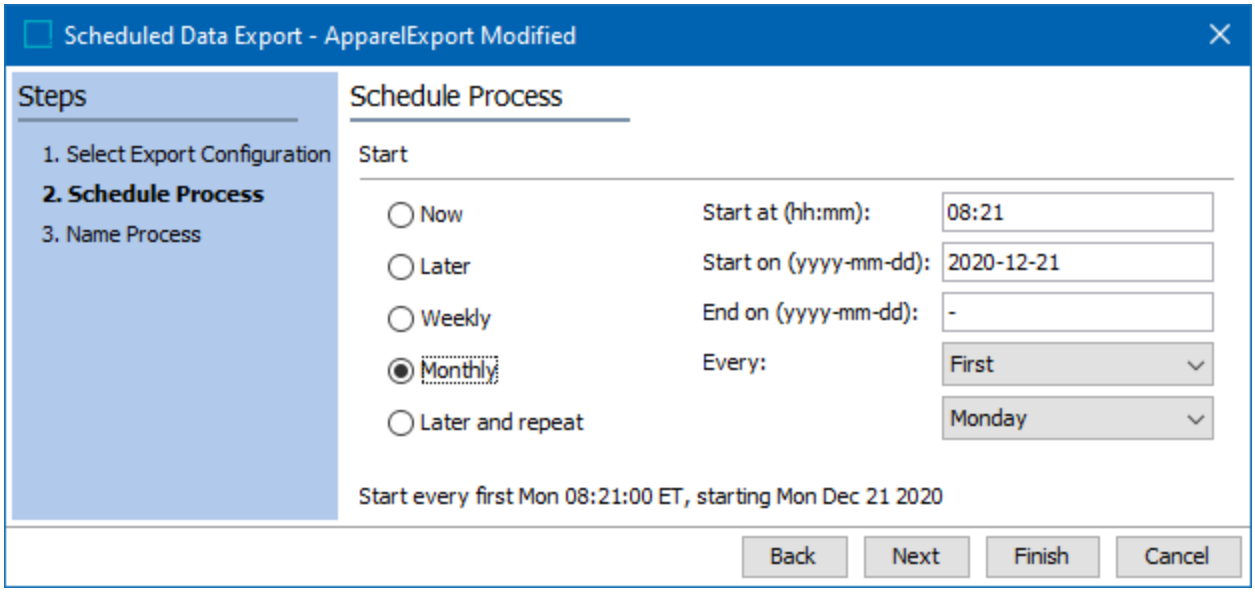
5. Manage scheduled data exports using the following information:
   - **Deleting a Scheduled Background Process** within the **System Setup** documentation, removes a scheduled export process, which is required to run current parameters when the original configuration is modified.
   - **Scheduled Process Properties** within the **System Setup** documentation, displays the status of a scheduled process from queued, to active, and finally when it has ended.

Scheduled Data Export - Select Export Configuration



1. For **Configuration**, use a method to select the required configuration:
   - Type the configuration name or ID into the text box and select it from the matches displayed in the dropdown.
   - Click the ellipsis button (...). The Select Export Configuration window displays. Browse or search for the configuration, and click **Select**.
2. If the configuration is based on a Collection, the **Refresh before each run** checkbox is enabled. For more on collections, refer to the **Collections** documentation.
   - Check the box to update the collection before each scheduled export.
   - Leave the box unchecked if you want to only export the objects currently in the collection at the time of export.
3. Click **Next** to display **Scheduled Data Export - Schedule Process**.

Scheduled Data Export - Schedule Process



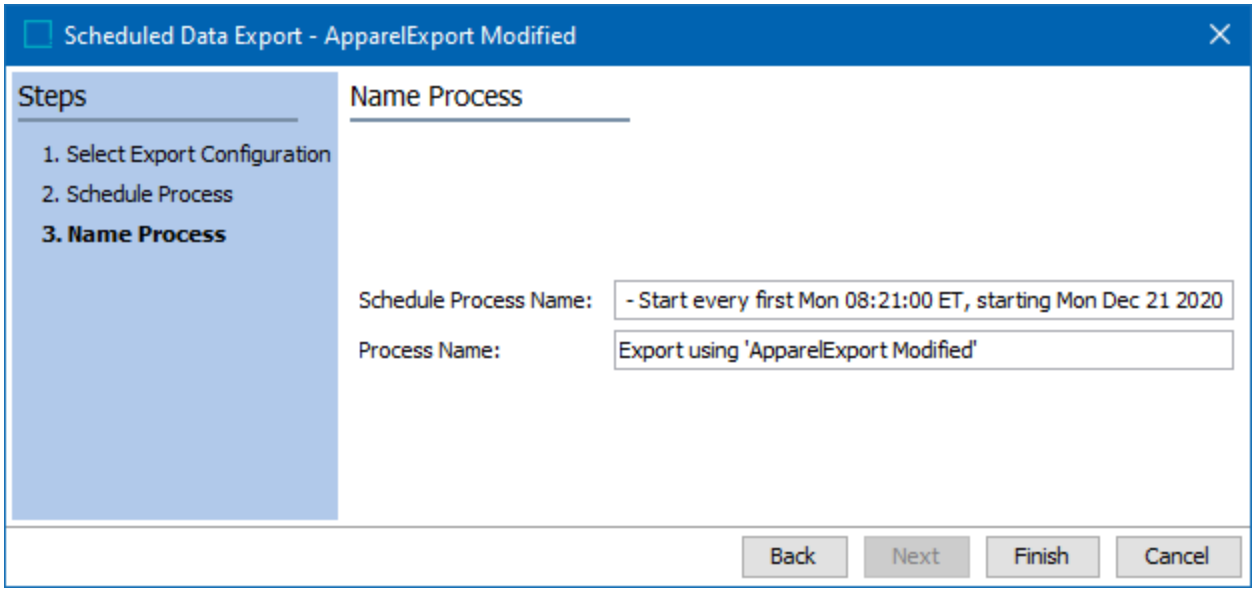
Important: Consider the time zone of the application server compared to that of the workbench (the client) where the schedule is created or viewed. When scheduling a job, the local time zone is displayed in the workbench, but the time zone of the server is used to run the background process. Although displayed, the time zone of the client is not included in the instruction to the server to run the job. This can cause confusion about when the job will run since the scheduled time is not automatically converted to accommodate potential differences in time zones.

1. Select one of the following options.
   - **Now** runs the process only once when you click **Finish**.
   - **Later** runs the process only once at the specified date and time.
   - **Weekly** runs the process on the days select in the specified period until the process is deleted.
   - **Monthly** runs the process every month at the specified day and time until the process is deleted.
   - **Later and repeat** runs the process at the specified start date and time, and repeatedly after a specified number of minutes between attempts to export until the process is deleted.

   Note: Since a scheduled export process continues to use the configuration it was started on, if you change the parameters of the export configuration, you should delete the relevant scheduled processes. Refer to **Deleting a Scheduled Background Process** within the **System Setup** documentation.

2. Click **Next** to display **Scheduled Data Export - Name Process**.

Scheduled Data Export - Name Process



1. For **Schedule Process Name**, enter a name for the scheduled process. This name is displayed on the process on the BG Processes tab > Scheduled Processes node for all schedule options except 'Now.' The default value includes the name of the export configuration and scheduling details.
2. In the **Process Name** field, enter a name for the exporting process. This name is displayed on the BG Processes tab > Export Manager Pipeline. The default value includes the name of the export configuration.
3. Click **Finish**.

To monitor scheduled processes, refer to the **Scheduled Process Properties** section within the **System Setup** documentation.

Gateway Integration Endpoints

Gateway integration endpoints enable STEP to access external systems through business rules that make calls to fetch data or update status. For example, to create objects in STEP using an ID from an external Enterprise Resource Planning (ERP) system, STEP can use a gateway endpoint to access and retrieve the external system ID.

The gateway endpoint controls all access to a given system. It holds the information that is required to access the external system including the user name, password, and the server URL. This means the information does not have to be included in the JavaScript business action and that the duration and frequency of calls can be monitored and logged.

The business rule uses synchronous REST calls from JavaScript. For more information about REST and the REST API, access the **STEP API Documentation**, available at [system]/sdk or from the Start Page.

Note: Keystore is not supported.

Setup Requirements

Setting up and using a gateway integration endpoint involves the following steps:

1. Create a setup group to hold the endpoint as described in **Initial Setup for a Gateway Integration Endpoint**.
2. Create a gateway endpoint as described in **Creating a Gateway Integration Endpoint**.
3. Configure the endpoint as described in **Configuring a Gateway Integration Endpoint**.
4. Create a business rule to access the gateway endpoint as defined in **Gateway Integration Endpoint Bind** in the **Resource Materials** online help.
5. Enable the endpoint and test connectivity as described in **Running a Gateway Integration Endpoint**.

Additional Information

The following information is useful once a gateway integration endpoint is set up:

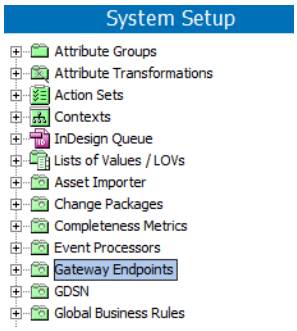
1. Maintain or modify the endpoint as described in **Maintaining a Gateway Integration Endpoint**.
2. Monitor the endpoint as described in **Monitoring a Gateway Integration Endpoint**.
3. Export a gateway integration endpoint definition as comments and submit to an external source control system for comparison purposes as described in **Configuration Management** documentation.

Initial Setup for a Gateway Integration Endpoint

Before creating a gateway integration endpoint, one or more setup groups must be created that are allowed to hold gateway integration endpoints. You must also specify the setup group(s) in which gateway integration endpoints can be created. This setup only needs to be performed once, and most systems will already have it completed.

Only users with the relevant privileges can view or maintain gateway integration endpoints. For detailed information, refer to the **Action Sets** section and the **Users and Groups** section in the **System Setup** documentation.

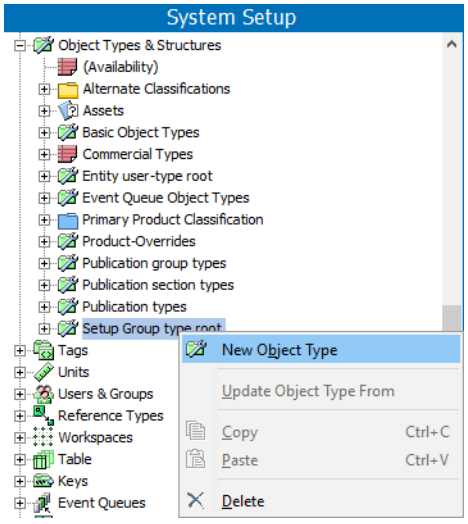
Review your System Setup tab to determine if one or more gateway integration endpoint nodes already exist. The name of the node on your system should include the word 'gateway' but is not required to match the one in the image below.



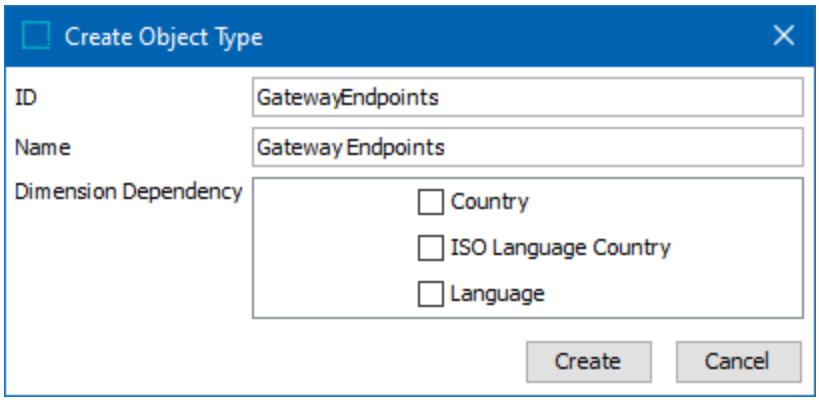
Once the setup has been completed, the steps in this section are only needed if additional levels of organization are desired.

Create the Gateway Setup Group

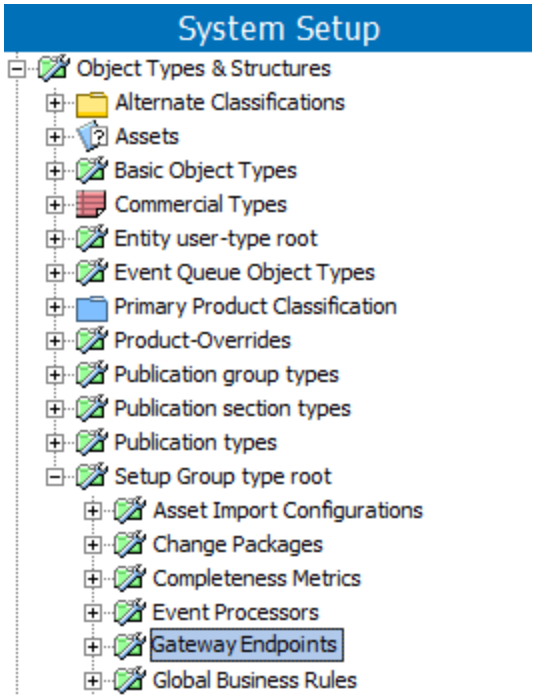
1. In System Setup, expand **Object Types & Structures**.
2. Right-click 'Setup Group type root', and choose **New Object Type**.



3. Enter an **ID** and a **Name**, select any required Dimension Dependencies, and click **Create**.

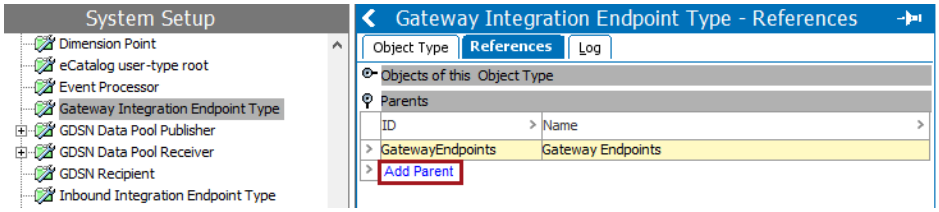


4. The new setup group appears in System Setup under 'Object Types & Structures' as a child in the **Setup Group type root**.

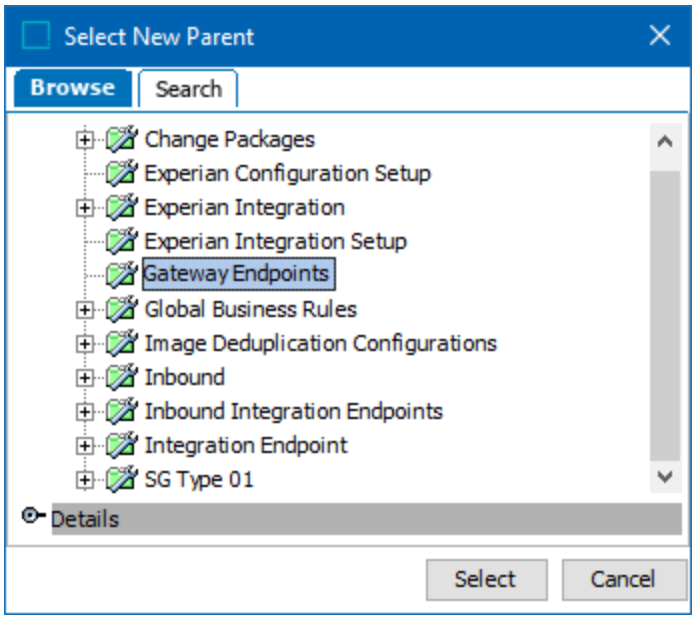


Link the Gateway Object Type to Setup Group

1. In Object Types & Structures > expand Basic Object Types > select **Gateway Integration Endpoint Type**.
2. On the References tab > Parents flipper > click the **Add Parent** link.

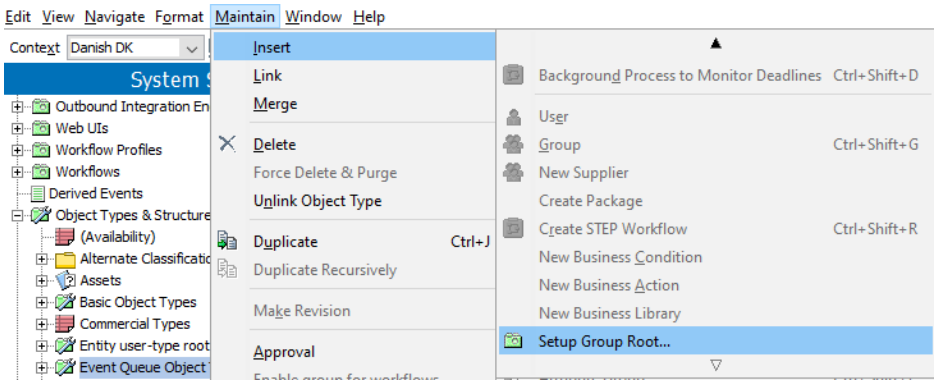


3. In the Select New Parent dialog, select the setup group you created, and click **Select** to make it a valid parent.

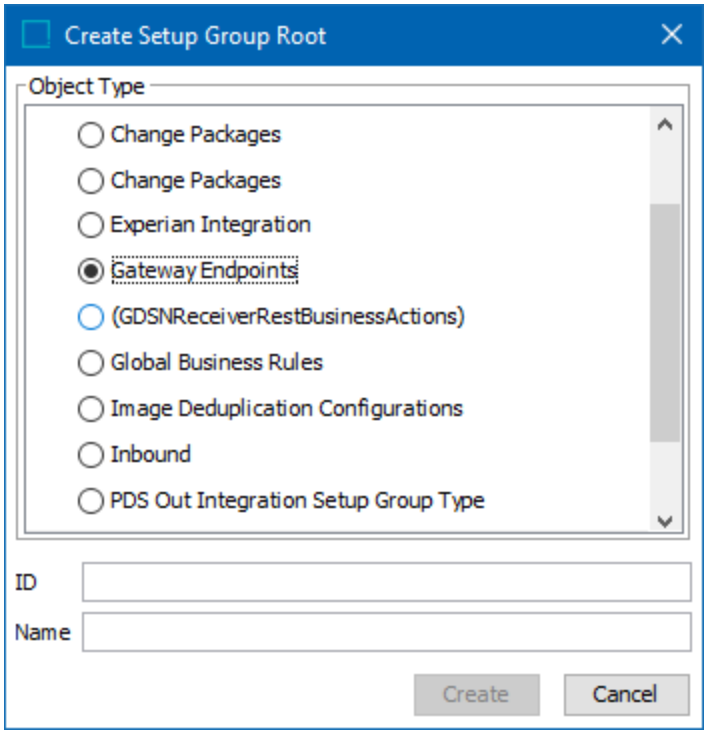


Create an Instance of the Gateway Endpoint Object

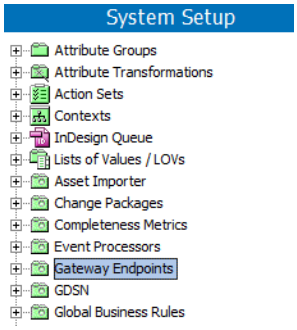
1. On the System Setup tab, select any object in the **System Setup** hierarchy to activate the following Maintain menu selection.
2. Click the Maintain menu, point to Insert, and select **Setup Group Root**.



3. In the Create Setup Group Root dialog, select the gateway object type, enter an **ID** and a **Name**, and click **Create**.



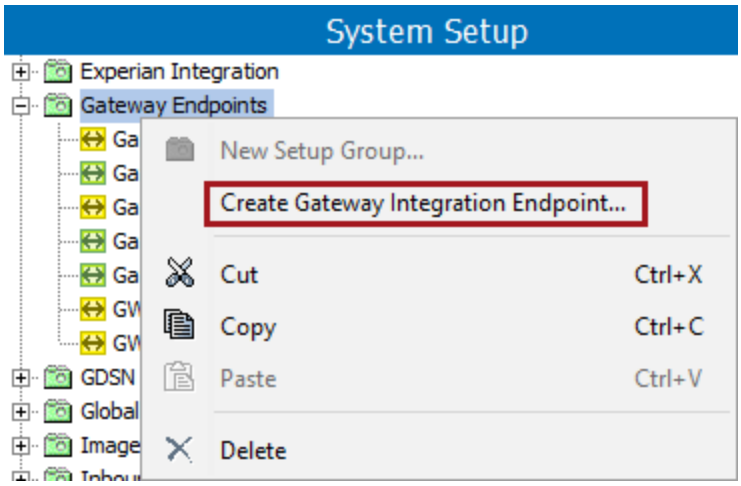
A setup group is created as a node in the System Setup hierarchy. Gateway integration endpoints can now be created under this new node.



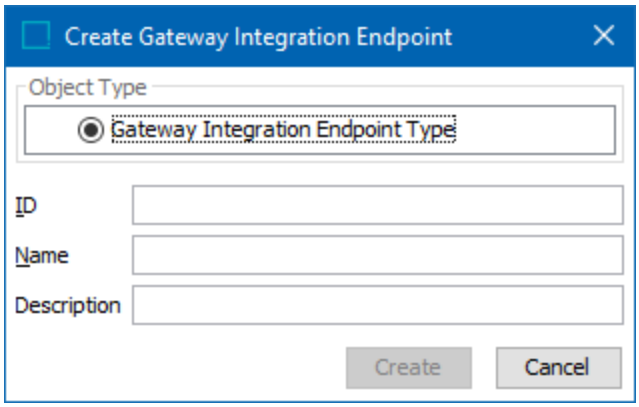
Creating a Gateway Integration Endpoint

After creating a setup group for gateway integration endpoints, create a gateway integration endpoint to access data on an external server.

1. In System Setup, right-click the Gateway Endpoints setup group, and click **Create Gateway Integration Endpoint**.



2. Enter an **ID** and a **Name** for the integration endpoint. The Description parameter is optional.

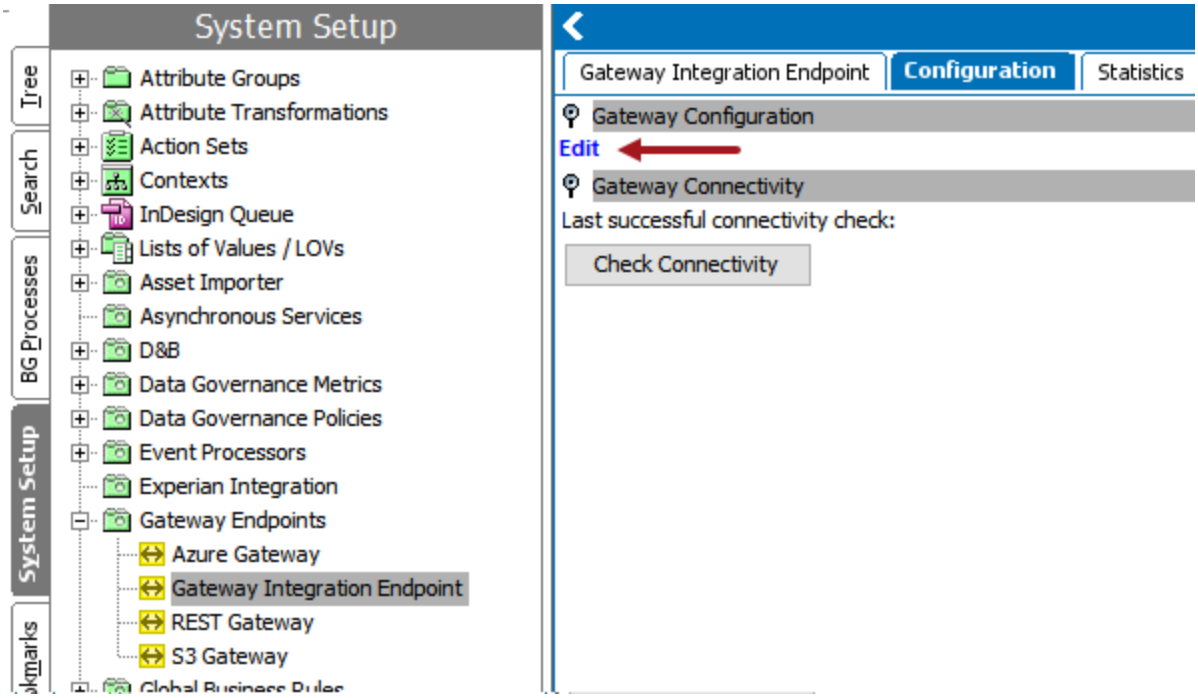


3. Click **Create** and then configure the endpoint. For more information, refer to the **Configuring a Gateway Integration Endpoint** documentation.

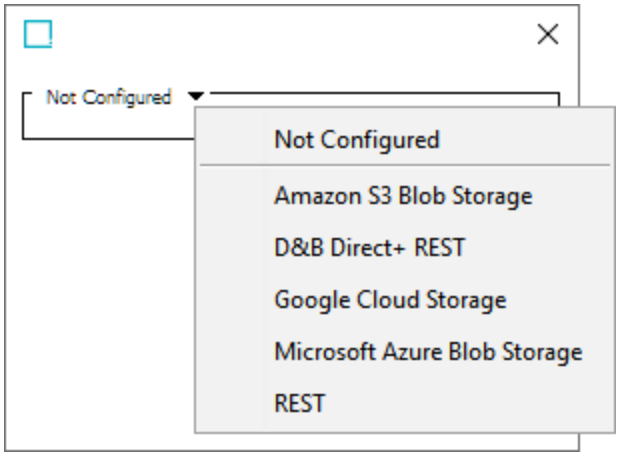
Configuring a Gateway Integration Endpoint

Once a gateway integration endpoint is created, the configuration settings allow you to identify the server housing the required data.

1. In System Setup, expand the gateway endpoints setup group, and select the relevant gateway integration endpoint.
2. On the Configuration tab, click the **Edit** link to display the Gateway Integration Endpoint Configuration dialog.



3. On the Gateway Integration Endpoint Configuration dialog, choose which external system the gateway integration endpoint will access.



For more information on how to configure a gateway integration endpoint for specific external systems, refer to:

- **Configuring a Gateway Integration Endpoint - Amazon S3 Blob Storage**
- **Configuring a Gateway Integration Endpoint - Google Cloud Storage**
- **Configuring a Gateway Integration Endpoint - Microsoft Azure Blob Storage**
- **Configuring a Gateway Integration Endpoint - REST**

Configuring a Gateway Integration Endpoint - Amazon S3 Blob Storage

A gateway integration endpoint (GIEP) allows STEP to communicate with an external storage system. Once a GIEP has been created and Amazon S3 Blob Storage is selected, the configuration settings allow you to identify the location of the required data.

Prerequisites

To use the Gateway Integration Endpoint Configuration dialog for Amazon S3 Blob Storage, the following case sensitive properties must be set up first in the sharedconfig.properties file on the STEP application server.

Note: Sensitive configuration values that should be filtered from view are denoted with 'Secret.' This means that the actual values are not visible to users or to Stibo Systems, for example, via Admin Portal configuration lists and remote diagnostics.

Changes to the properties file, outlined below, are implemented when the server is restarted.

- **S3Connection**

  The syntax for specifying the connection details is:

  ```
  BlobStorage.S3.Secret.Connection.1=<connection-string-
  alias>,<AWSAccessKeyID>,<AWSSecretAccessKeyPassword>
  BlobStorage.S3.Secret.Connection.2=<connection-string-
  alias2>,<AWSAccessKeyID2>,<AWSSecretAccessKeyPassword2>
  ```

  The <connection-string-alias> and <connection-string-alias2> are displayed in the 'S3 Connection' dropdown menu for the Gateway Integration Endpoint (explained in the next section). This shows the alias instead of the actual S3 required connection values (aws- access-key, aws-secret-access-key-password, aws region).

- **BucketName**

  The syntax for specifying the BucketName is:

  ```
  BlobStorage.S3.Secret.BucketName.1=<my-bucket1>
  BlobStorage.S3.Secret.BucketName.2=<my-bucket2>
  ```

  The <my-bucket1> and <my-bucket2> are displayed in the 'BucketName' dropdown menu for the Gateway Integration Endpoint (explained in the next section).

- **Region**

  The syntax for specifying the AWS Region Code is:

```
BlobStorage.S3.Region.1=<my-region-code>
BlobStorage.S3.Region.2=<my-region-code2>
```

The <my-region-code> and <my-region-code2> are displayed in the 'Region' dropdown menu for the Gateway Integration Endpoint (explained in the next section).

For available region codes to use for AWS Region, refer to: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-regions-availability-zones.html#concepts-available-regions

As an example, a valid value would be eu-west-1 if using this region.

Note: While only two of each property is shown above; you can have additional properties by using integer at the end of the property name (e.g., .1, .2, .3, etc.). When duplicates exist, only the last value is displayed in the dialog.

- **Proxy config**

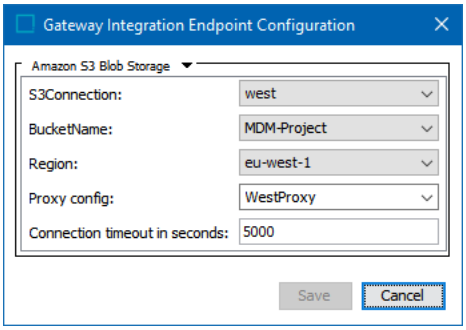
  Configuration is needed for on-prem systems if the delivery connection must first pass through a proxy server with its own login requirement.

  The syntax for specifying the proxy is defined in the **HTTP Proxy Configurations** topic.

Configuring the Gateway Integration Endpoint

Once the properties as described above are entered into the sharedconfig.properties file, the options outlined within these properties display in the dropdowns in the dialog. If the dropdowns are empty, then the properties are not set up or are set up incorrectly.

1. On the Gateway Integration Endpoint Configuration dialog, use the following parameters to specify which external system the gateway integration endpoint will access.

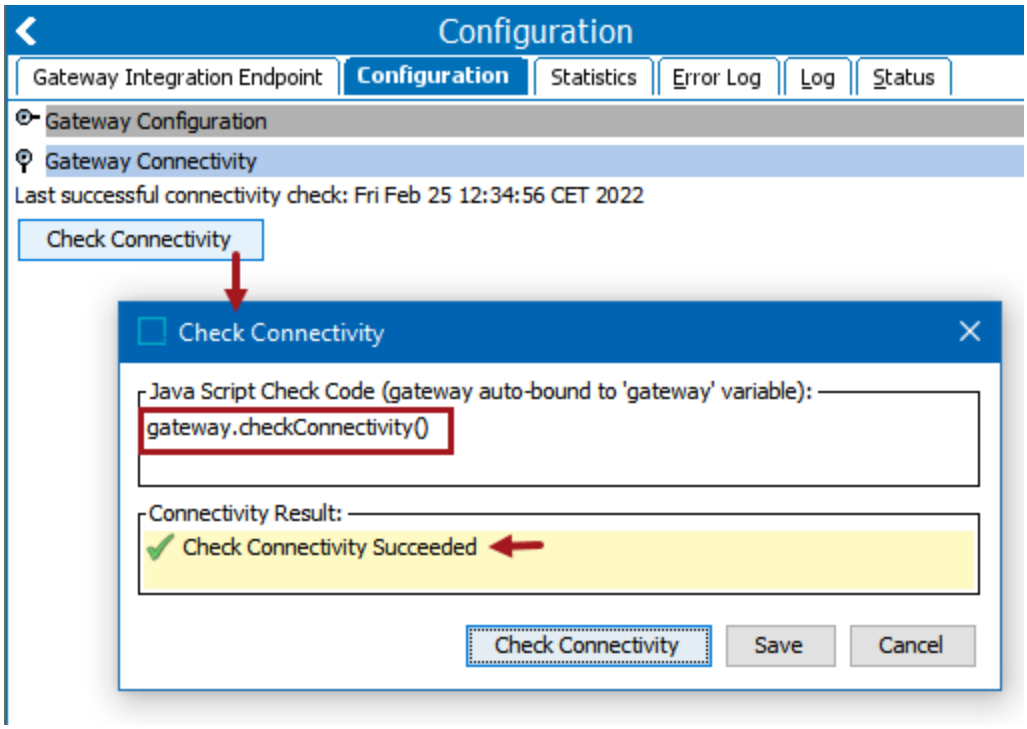


   - **S3Connection** - The desired S3 connection.
   - **BucketName** - The S3 bucket name to use. This name must already be established on the Amazon account.
   - **Region** - The AWS Region Code to use.

- **Proxy config** - Select the desired HTTP proxy configuration if the delivery connection must first pass through a proxy server with its own login requirement. Intended for use by on-prem systems where the AWS S3 Storage is hosted externally.
- **Connection timeout in seconds** - The connection timeout on the request in seconds. If left blank, the driver default is used.

2. Click **Save** to complete the configuration.
3. Enable the endpoint as defined in the **Running a Gateway Integration Endpoint** topic.
4. Test the connection from the gateway as follows:
   - On the Gateway Connectivity flipper, click the **Check Connectivity** button.
   - In the Check Connectivity dialog, in the Java Script Check Code section, add:

```
gateway.checkConnectivity()
```



   - Click the **Check Connectivity** button and verify success or make the necessary corrections to connect.

Using the Gateway Integration Endpoint

Configuration of a GIEP is required to set up:

- **Asset Publisher Event Processor**

  For more information, refer to the **Asset Publisher Processing Plugin Parameters and Triggers** topic in the **Event Processors** section of **System Setup** documentation.

- **Cloud Blob Storage Delivery Method**

  For more information, refer to the **Cloud Blob Storage Delivery Method** topics in the **OIEP Delivery Methods** and the **Export Manager Delivery Methods** sections of the **Data Exchange** documentation.

Configuring a Gateway Integration Endpoint - Google Cloud Storage

A gateway integration endpoint (GIEP) allows STEP to communicate with an external storage system. Once a GIEP has been created and Google Cloud Storage has been selected, the configuration settings allow you to identify the location of the required data.

Prerequisites

To use the Gateway Integration Endpoint Configuration dialog for Google Cloud Storage, grant the following Google Cloud Storage permissions:

- storage.buckets.get
- storage.objects.create
- storage.objects.delete
- storage.objects.get
- storage.objects.getIamPolicy
- storage.objects.list
- storage.objects.update

Additionally, add the following case sensitive properties in the sharedconfig.properties file on the STEP application server.

Note: Sensitive configuration values that will be filtered from view are denoted with 'Secret.' This means that the actual values will not be visible to users or to Stibo Systems. For example, via Admin Portal configuration lists and remote diagnostics.

- **Connection**

  The syntax for specifying the connection is:

  ```
  BlobStorage.Google.Secret.ConnectionKeyPath.<#>=<connection-string-alias>,<key-file-location>
  ```

  For example:

  ```
  BlobStorage.Google.Secret.ConnectionKeyPath.1=Website_GCS_Storage1,/shared/customer-config/GCS/key1.json
  BlobStorage.Google.Secret.ConnectionKeyPath.2=Website_GCS_Storage2,/shared/customer-config/GCS/key2.json
  ```

The connection string alias value supplied for the property is displayed in the 'ConnectionKeyPath' dropdown menu for the Gateway Integration Endpoint Configuration dialog (explained in the next section). Use a comma to separate the connection-string-alias and the key-file-location values. Enter as many lines as necessary, using one for each available connection key path and using a unique integer. When duplicates exist, only the last value is displayed in the dialog.

The connection with Google Cloud Storage uses Application Default Credentials and requires that a JSON file holds the key to be created. Refer to the Google Cloud console documentation (https://cloud.google.com/iam/docs/keys-create-delete) to create a JSON key type file.

Download the JSON key file and place it in a location that is accessible by the STEP application. On SaaS environments, for example, the file can be uploaded over an sFTP connection to the customer-config folder.

- **Bucket**

  The syntax for specifying the bucket is:

  ```
  BlobStorage.Google.BucketName.<#>=<my-bucket>
  ```

  For example:

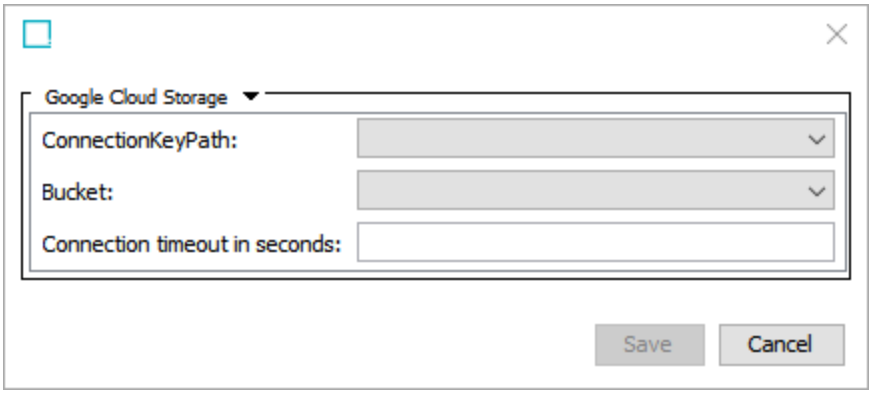
  ```
  BlobStorage.Google.BucketName.1=ProductImages
  BlobStorage.Google.BucketName.2=Illustrations
  ```

  The value supplied for the property is displayed in the 'Bucket' dropdown menu for the Gateway Integration Endpoint Configuration dialog (explained in the next section). Enter as many lines as necessary, using one for each available bucket and using a unique integer. When duplicates exist, only the last value is displayed in the dialog.

Configuring the Gateway Integration Endpoint

After setting the Google Cloud Storage permissions and adding the sharedconfig.properties file entries to supply values on the dialog, configure the GIEP. If the dropdowns are empty, revisit the sharedconfig.properties file to correct the error.

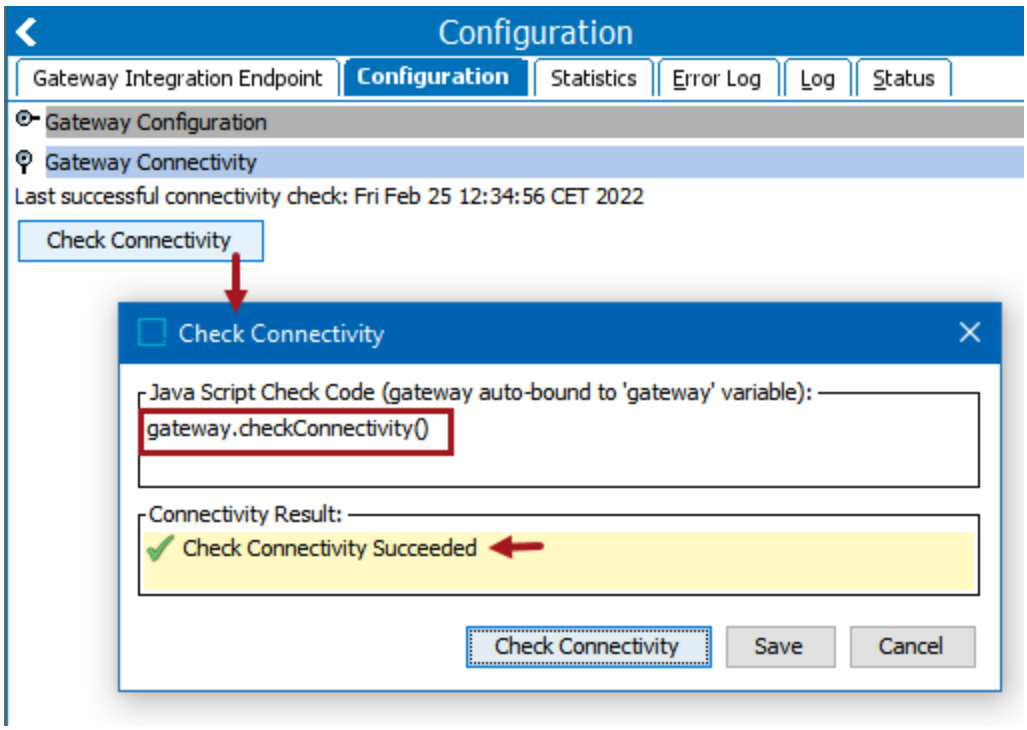
1. On the Gateway Integration Endpoint Configuration dialog, use the following parameters to specify which external system the GIEP will access.



- **ConnectionKeyPath** - Select the desired Google Cloud Storage connection.
- **Bucket** - Select the Google Cloud Storage bucket to use.
- **Connection timeout in seconds** - Add number of seconds until timeout. If left blank, the driver default is used.

2. Click **Save** to complete the configuration.
3. Enable the endpoint as defined in the **Running a Gateway Integration Endpoint** topic.
4. Test the connection from the gateway as follows:
   - On the Gateway Connectivity flipper, click the **Check Connectivity** button.
   - In the Check Connectivity dialog, in the Java Script Check Code section, add:

```
gateway.checkConnectivity()
```



   - Click the **Check Connectivity** button and verify success or make the necessary corrections to connect.

Using the Gateway Integration Endpoint

Configuration of a GIEP is required to set up:

- **Asset Publisher Event Processor**

  For more information, refer to the **Asset Publisher Processing Plugin Parameters and Triggers** topic in the **Event Processors** section of **System Setup** documentation.

- **Cloud Blob Storage Delivery Method**

  For more information, refer to the **Cloud Blob Storage Delivery Method** topics in the **OIEP Delivery Methods** and the **Export Manager Delivery Methods** sections of the **Data Exchange** documentation.

Configuring a Gateway Integration Endpoint - Microsoft Azure Blob Storage

A gateway integration endpoint (GIEP) allows STEP to communicate with an external storage system. Once a GIEP has been created and Microsoft Azure Blob Storage is selected, the configuration settings allow you to identify the location of the required data.

Prerequisites

To use the Gateway Integration Endpoint Configuration dialog for Microsoft Azure Blob Storage, the following case sensitive properties must be set up first in the sharedconfig.properties file on the STEP application server.

Note: Sensitive configuration values that will be filtered from view are denoted with 'Secret.' This means that the actual values will not be visible to users or to Stibo Systems, for example, via Admin Portal configuration lists and remote diagnostics.

Changes to the properties file, outlined below, are implemented when the server is restarted.

- **Connection Info**

  The syntax for specifying the connection details is:

  ```
  BlobStorage.Azure.Secret.Connection.1=<connection-string-alias>,<azure-connection-string>
  BlobStorage.Azure.Secret.Connection.2=<connection-string-alias2>,<azure-connection-string2>
  ```

  The <connection-string-alias> and <connection-string-alias2> will be displayed in the 'Connection Info' dropdown menu for the Gateway Integration Endpoint (explained in the next section), showing the alias instead of the actual connection string value.

  More information about connection string values and how to find or create them can be found online at https://portal.azure.com (go to Storage Accounts >> Your-Storage- Account >> Access Key >> Connection String).

- **Container Name**

  The syntax for specifying the container name is:

  ```
  BlobStorage.Azure.Secret.ContainerName.1=<my-container-name>
  BlobStorage.Azure.Secret.ContainerName.2=<my-other-container-name>
  ```

  The <my-container-name> and <my-other-container-name> will be displayed in the 'Container Name' dropdown menu for the Gateway Integration Endpoint (explained in the next section).

Note: Two of each property are shown above; however, you can have numerous config properties by increasing the integer at the end (e.g., .1, .2, .3, etc.).

Example configuration:

```
BlobStorage.Azure.Secret.Connection.1=AzureConn,DefaultEndpointsProtocol=https;Accou
ntName=pimuser;AccountKey=t/mGLRrqkX64WM0yBdIp4qEMHELl0Y933zVDuKHz+/vBC20wRud3GXstYa
xQqk1yF9C3D9d/AZbJ+AStTVkmlA==;EndpointSuffix=core.windows.net
BlobStorage.Azure.Secret.ContainerName.1=myProducts
```

It is also possible to use a Shared Access Signature (SAS) credential for the ConnectionString.

The SAS token must be created directly on the blob storage account itself (and not the corresponding container); and as a minimum, it must have **Service**, **Container**, and **Object** specified as its 'Allowed resource types' as well as **Read**, **Write**, and **List** for its 'Allowed permissions' to grant the proper access rights to STEP.

These resource types and permissions are required to allow STEP to perform all the needed operations to deliver the content (blobs) to the specified Azure Blob Storage account's container.

Important: If the SAS token has insufficient privileges, the delivery will result in an error message similar to this one:

If you are using a SAS token, and the server returned the error message 'Signature did not match', you can compare the string to sign with the one generated by the SDK. To log the string to sign, pass in the context key value pair 'Azure-Storage-Log-String-To-Sign': true to the appropriate generateSas method call. Before going to production, disable 'Azure-Storage-Log-String-To-Sign' as this string can potentially contain PII.

Status code 403, "<?xml version="1.0" encoding="utf-8"?><Error><Code>AuthorizationResourceTypeMismatch</Code><Message>This request is not authorized to perform this operation using this resource type. RequestId:836910b1-801e-001a-4da2-900fc9000000 Time:2022-07-05T19:11:07.4796215Z</Message></Error>"

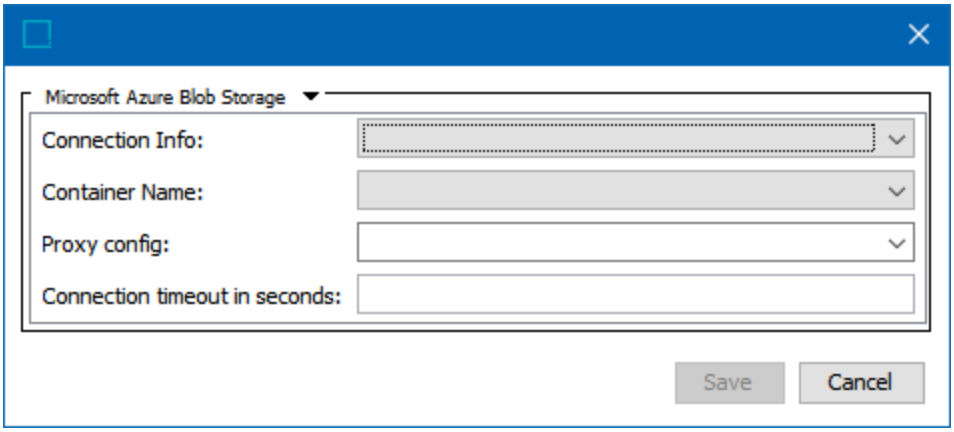
An example using a SAS configuration is below:

```
BlobStorage.Azure.Secret.Connection.2=AzureConnSAS,BlobEndpoint=https://pimuser.blob
.core.windows.net/;SharedAccessSignature=sv=2021-06-08&ss=b&srt=sco&sp=rwlx&se=2023-
01-01T05:00:00Z&st=2022-07-
05T18:58:41Z&spr=https&sig=N5X7J9tCMscbOTYioQ4sb30H7B%2B0j8dk74MSCQ6Gxsw%3D
BlobStorage.Azure.Secret.ContainerName.2=myProducts
```

Configuring the Gateway Integration Endpoint

Once the properties as described above are entered into the sharedconfig.properties file, the options outlined within these properties will display in the dropdowns in the configuration dialog. If the dropdowns are empty, then the properties are not set up or set up incorrectly.

1. On the Gateway Integration Endpoint Configuration dialog, use the following parameters to specify which external system the gateway integration endpoint will access.

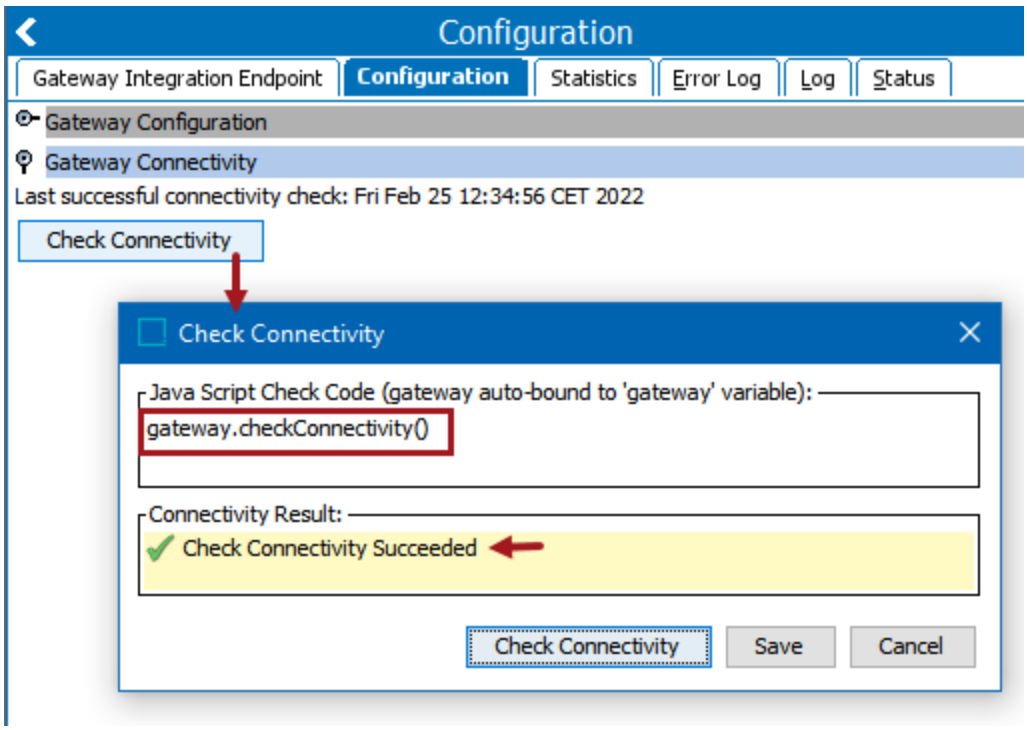


   - **Connection Info** - The desired Azure connection.
   - **Container Name** - The Azure container name to use.
   - **Proxy config** - Select the desired HTTP proxy configuration if the delivery connection must first pass through a proxy server with its own login requirement.
   - **Connection timeout in seconds** - The connection timeout on the request in seconds. If left blank, the driver default will be used.

     Note: For more information regarding proxy configurations, refer to the topic **HTTP Proxy Configurations** in the **Data Exchange** documentation.

2. Click **Save** to complete the configuration.
3. Enable the endpoint as defined in the **Running a Gateway Integration Endpoint** topic.
4. Test the connection from the gateway as follows:
   - On the Gateway Connectivity flipper, click the **Check Connectivity** button.
   - In the Check Connectivity dialog, in the Java Script Check Code section, add:

```
gateway.checkConnectivity()
```



- Click the **Check Connectivity** button and verify success or make the necessary corrections to connect.

Using the Gateway Integration Endpoint

Configuration of a GIEP is required to set up:

- **Asset Publisher Event Processor**

  For more information, refer to the **Asset Publisher Processing Plugin Parameters and Triggers** topic in the **Event Processors** section of **System Setup** documentation.

- **Cloud Blob Storage Delivery Method**

  For more information, refer to the **Cloud Blob Storage Delivery Method** topics in the **OIEP Delivery Methods** and the **Export Manager Delivery Methods** sections of the **Data Exchange** documentation.

Configuring a Gateway Integration Endpoint - REST

Once a gateway integration endpoint has been created and the external system has been selected, the configuration settings allow you to identify the server housing the required data.

To use mTLS or TLS with this type of endpoint, refer to the **Mutual Transport Layer Security** topic.

Prerequisites

All dropdowns are system-dependent and require the following sharedconfig.properties file properties to display dropdown options.

1. To populate the system-dependent **Server URLs** dropdown, edit the sharedconfig.properties file using the case -sensitive `RESTGateway.ServerURL` property. If necessary, use a comma to separate multiple URLs. The following is an example of a complete property entry for two systems named 'qa' and 'stage,' as well as their URLs 'http://step-qa' and 'http://step-stage':

   ```
   RESTGateway.ServerURL=qa=http://step-qa,stage=http://step-stage
   ```

   When configuring the **Address Typeahead** functionality (in the **Web User Interfaces** documentation), use the URL for the applicable address search service:

   - Loqate - https://api.addressy.com/
   - Google Places - https://maps.googleapis.com/maps/api/place/
   - For other address service options, refer to the service documentation for the relevant URL.

2. To populate the **SSL trust store location** dropdown, edit the sharedconfig.properties file using either the legacy `RESTGateway.SSLTrustStoreLocation` property or the global trust store `SSL.Default.Truststore.Location` property, as defined in the **Mutual Transport Layer Security** topic.

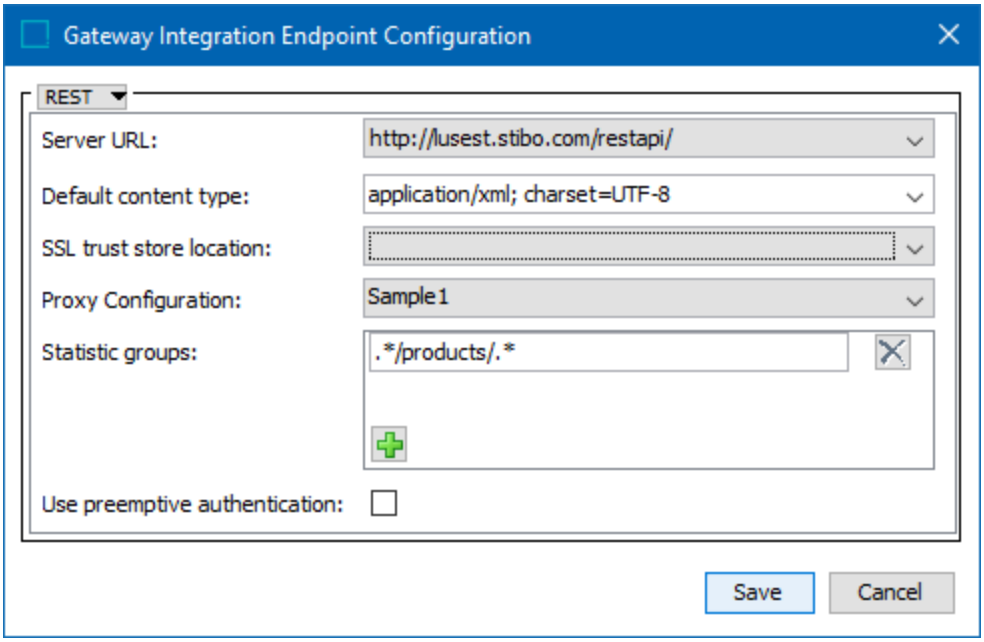
   Note: The `RESTGateway.SSLTrustStoreLocation` property does not work in conjunction with an outbound mTLS or TLS certificate configured in the `SSL.Default.Keystore.Location` the `SSL.<Alias>.Keystore.Location` properties. If you need a trust store in conjunction with mTLS, it must be configured using the `SSL.Default.Truststore.Location` property.

3. To populate the **Proxy Configuration** dropdown, edit the sharedconfig.properties file using the `RESTGateway.ProxyConfiguration` property. Refer to the **Proxy Configuration Setup** section below for details.

Configuration

The following steps are for REST configuration.

1. On the Gateway Integration Endpoint Configuration dialog, use the following parameters to specify the external system the gateway integration endpoint will access.



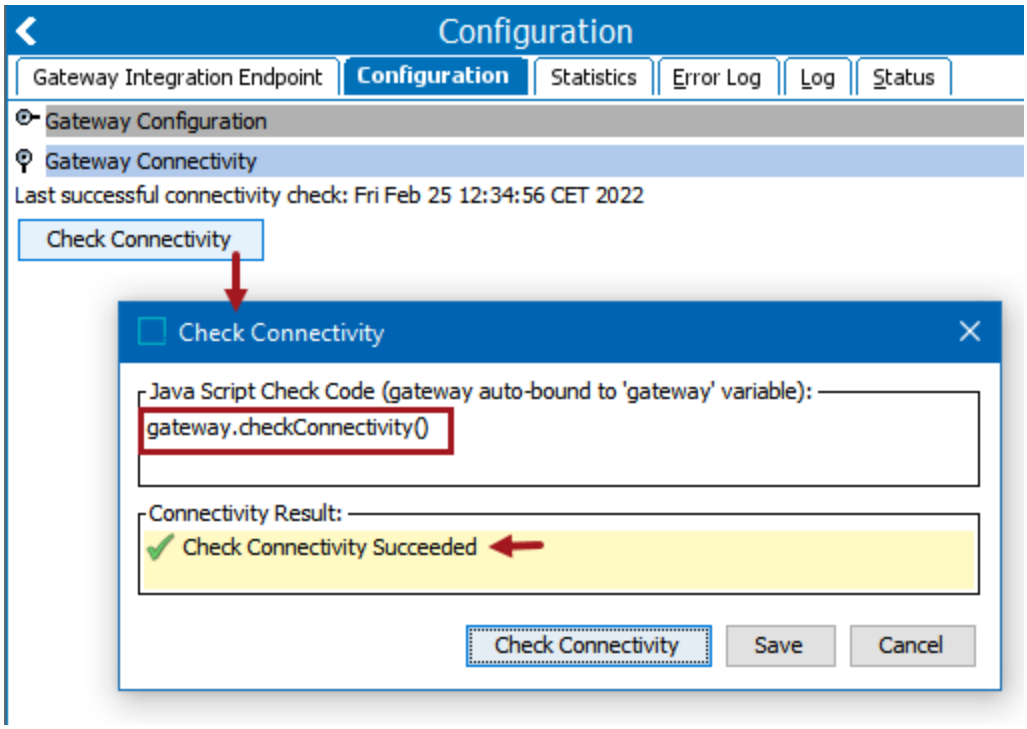
- **Server URL** - The URL of the server to be accessed from the dropdown. Dropdown options are provided as defined in the Prerequisites section above.
- **Default content type** - Data format or content type to be used for the endpoint. Type your own or select from the dropdown list of the most commonly used options.
- **SSL trust store location** - The trust store holds certificates that verify that the system can be trusted.

  The gateway connection is not configured to use SSL encryption when no global trust store is configured and/or if no trust store location (set in the legacy trust store property) is selected. In this scenario, the gateway connection is established using a certificate signed by a recognized Certification Authority, such as Verisign or Thawte.
- **Proxy Configuration** - Select from this dropdown if the gateway connection must first pass through a proxy server with its own login requirement.
- **Statistic groups** - Optional. Regular expression can be used to group executed REST methods on the endpoint's Statistics tab. The regular expression must match the entire URL, for example, `.*/products/.*`. Use the regular expression syntax available in Java in the **java.util.regex.Pattern** class. For more information, refer to the **Gateway Integration Endpoint Statistics** section or the **Regular Expression** topic in the **Resource Materials** online help.

2. Click **Save** to complete the configuration.
3. Enable the endpoint as defined in the **Running a Gateway Integration Endpoint** topic.
4. Test the connection from the gateway as follows:

- On the Gateway Connectivity flipper, click the **Check Connectivity** button.
- In the Check Connectivity dialog, in the Java Script Check Code section, add:

```
gateway.checkConnectivity()
```



- Click the **Check Connectivity** button and verify success or make the necessary corrections to connect.

5. Create the following as needed:
   - JavaScript business action to access the gateway endpoint, as defined in the **Gateway Integration Endpoint Bind** topic of the **Resource Materials** online help.
   - Address Typeahead to access an address search service, as defined in the **Address Typeahead** topic of the **Web User Interfaces** documentation.

Authentication

The gateway integration endpoint REST plugin supports both basic authentication and token-based authentication.

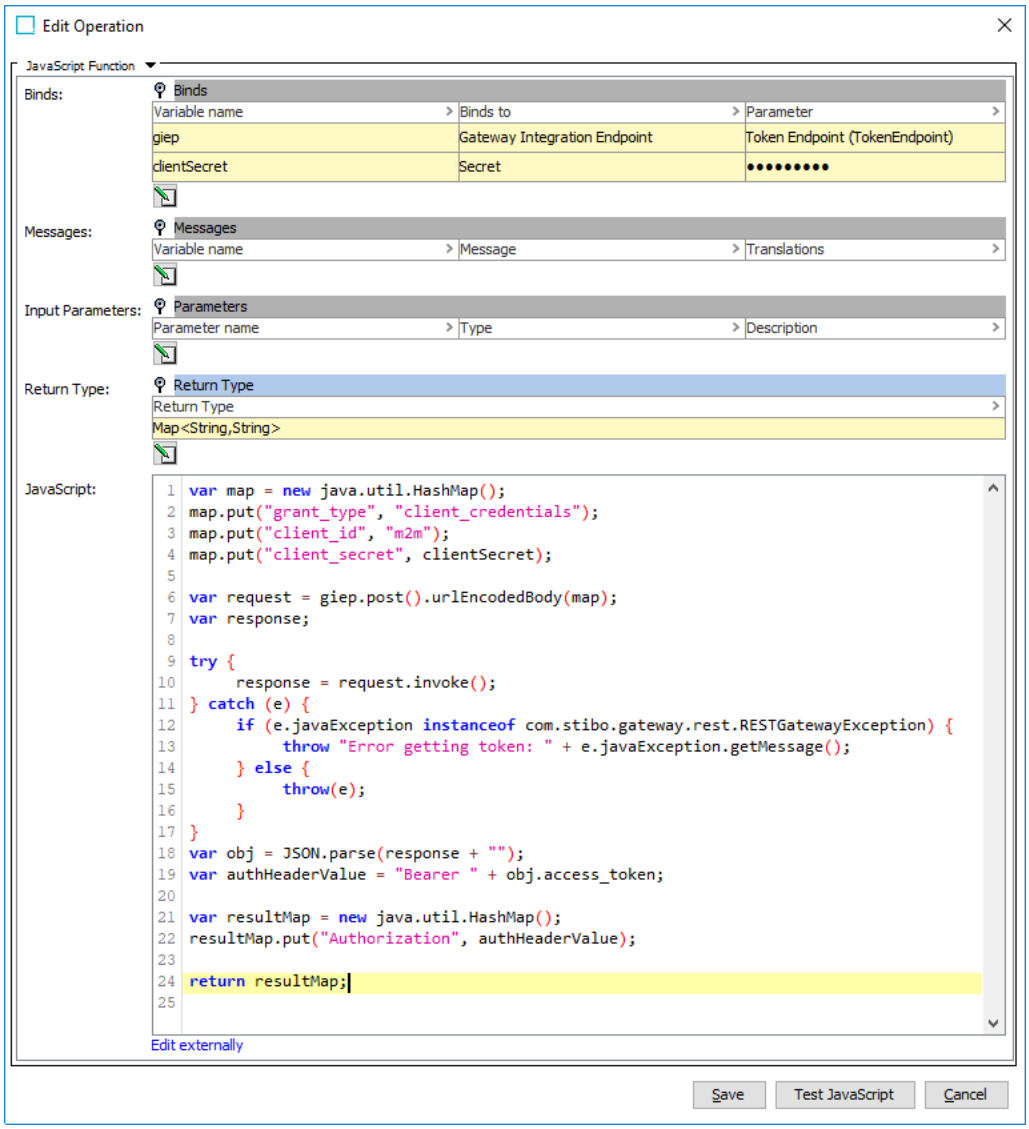
Basic Authentication

With preemptive authentication enabled, the basic authentication value for the Authorization header is sent with the first request to the external service instead of only sending the value after having received a basic authentication challenge from the service.

Token-Based Authentication

With the token-based authentication option, the responsibility for producing any required request headers is delegated to a business function. The business function must be configured to not expect any input and must produce a Map<String, String>, each map entry being a header to be sent with the request to the service.

The screenshot below shows the editor for a compatible business function that retrieves a token using the OAuth 2.0 client credentials flow and passes back to the gateway integration endpoint REST plugin as a value for the Authorization header.



The JavaScript shown above can be copied below:

```
var map = new java.util.HashMap();
map.put("grant_type", "client_credentials");
map.put("client_id", "m2m");
map.put("client_secret", clientSecret);

var request = giep.post().urlEncodedBody(map);
var response;

try {
    response = request.invoke();
} catch (e) {
    if (e.javaException instanceof com.stibo.gateway.rest.RESTGatewayException) {
        throw "Error getting token: " + e.javaException.getMessage();
    } else {
        throw (e);
    }
}
var obj = JSON.parse(response + "");
var authHeaderValue = "Bearer " + obj.access_token;

var resultMap = new java.util.HashMap();
resultMap.put("Authorization", authHeaderValue);

return resultMap;
```

The parameter for selecting the business function is the 'Auth Header Value Function.' The REST plugin automatically calls the business function when a new token is required.

Note: It is strongly discouraged to configure both basic authentication and token-based authentication since this configuration is not supported.

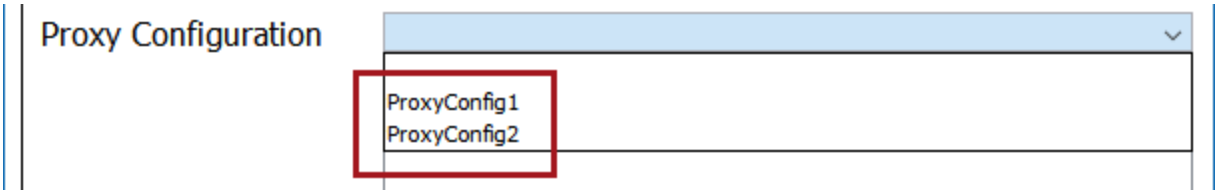
Proxy Configuration Setup

The following are two examples of how the dynamic `RESTGateway.ProxyConfiguration` property can be configured:

```
RESTGateway.ProxyConfiguration.1=ProxyConfig1,10.64.8.253,808,username,password
RESTGateway.ProxyConfiguration.2=ProxyConfig2,10.64.9.253,64,user10,password1
```

To break the first of these two properties into its component parts (each element separated by commas), refer to the screenshot and the numbered list below:

1. Required text to enable the REST Gateway Proxy configuration property
2. The name of the configuration. This is what appears in the dropdown as a selectable option, for example:

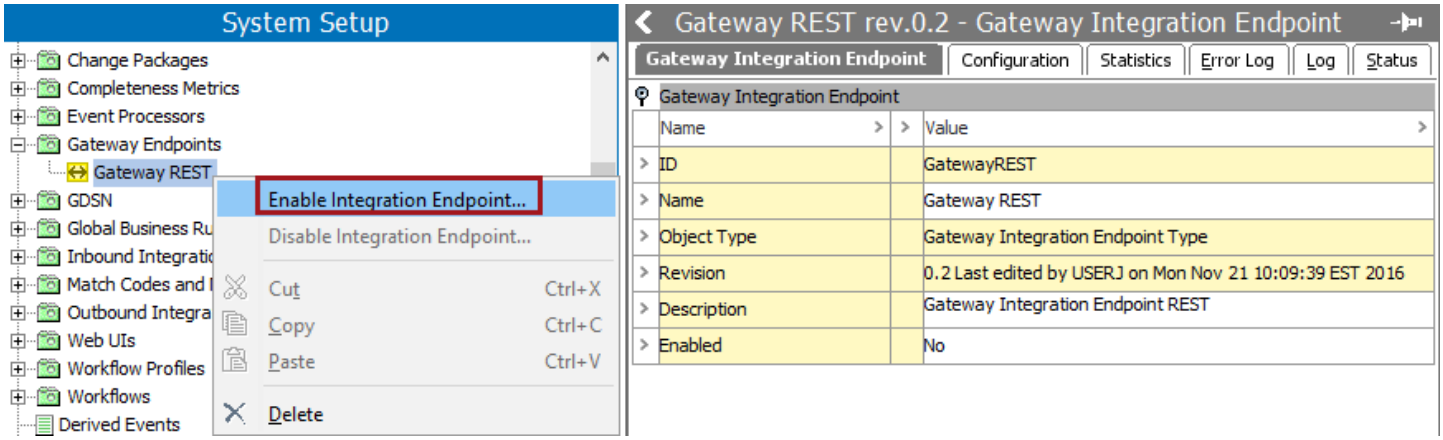


3. The IP of the proxy server being accessed
4. The port of the proxy server being accessed
5. The username for the proxy server being accessed (optional)
6. The password for the proxy server being accessed (optional)

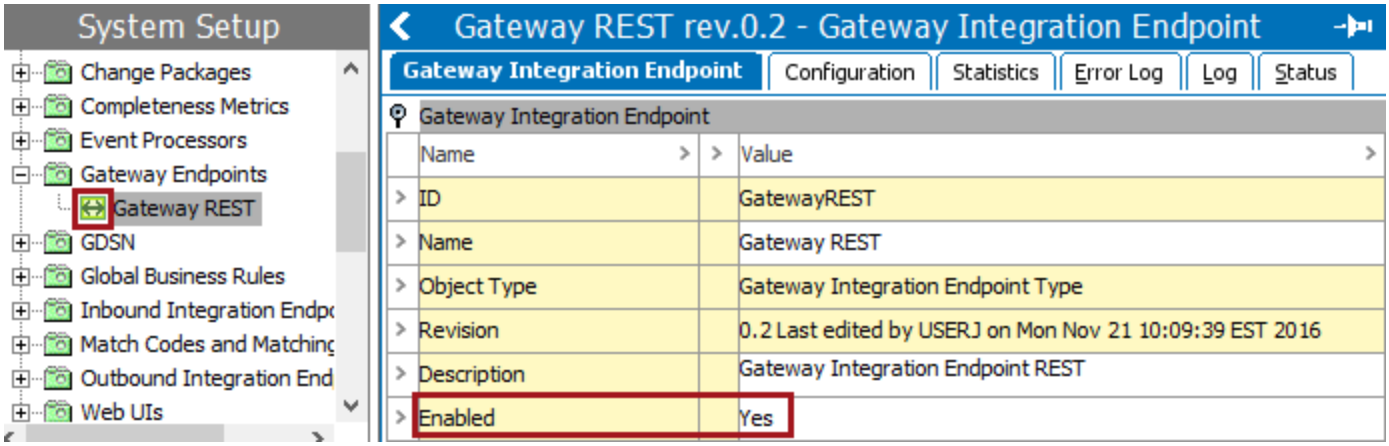
Running a Gateway Integration Endpoint

After configuring a gateway endpoint, it must be enabled and connected before it can run.

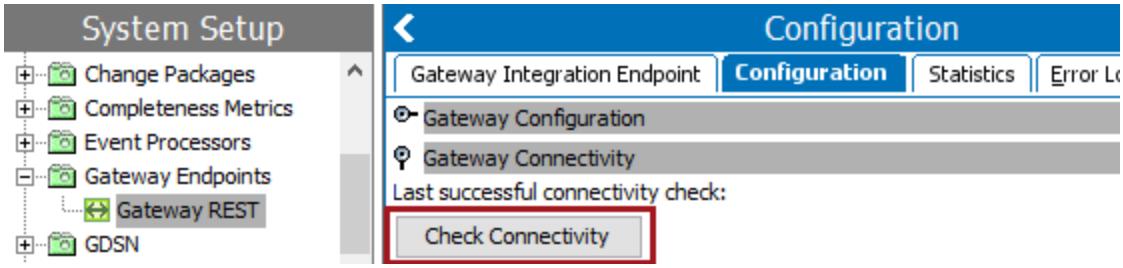
1. On System Setup expand the Gateway Endpoints node to display all existing gateway endpoints.
2. Select a gateway endpoint that it has been disabled by user request(indicated by the yellow gateway icon), right-click, and select **Enable Integration Endpoint** from the menu.



The tree navigator icon is now green ( ), indicating it is active and ready to use. The Gateway Integration Endpoint flipper 'Enabled' parameter now displays 'Yes.'

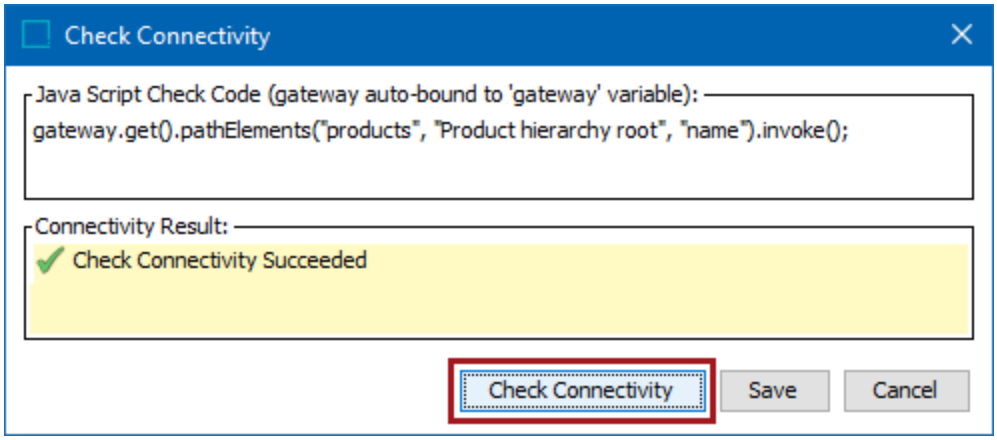


3. On the Configuration tab under the Gateway Connectivity flipper, click the **Check Connectivity** button.



4. In the **Check Connectivity** dialog, add JavaScript to access the external system.

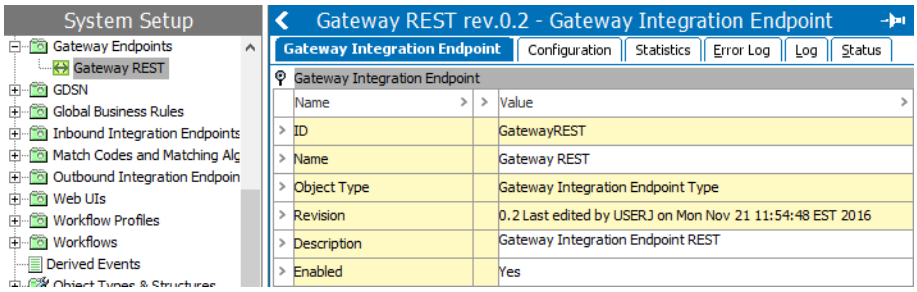
   In the following example, the JavaScript gets the name of the "Product Hierarchy root." For information about the JavaScript syntax, access the **STEP API Documentation**, available at [system]/sdk or from the Start Page.



5. Click the **Check Connectivity** button to display the connectivity result.

Maintaining a Gateway Integration Endpoint

Gateway integration endpoint settings can be viewed and edited from the Gateway Integration Endpoint editor.



The editor has the following tabs:

- **Gateway Integration Endpoint**: holds basic information like ID, Name, and Enable status.
- **Configuration**: holds information about the external server, user credentials and allows you to test the connection to the external server. Also allows editing of this information via the Edit link in the Gateway Configuration flipper. For more information, refer to the **Configuring a Gateway Integration Endpoint** section.
- **Statistics**: provides statistics about frequency and timings of external calls using the gateway. For more information, refer to the **Gateway Integration Endpoint Statistics** section.
- **Error Log**: shows errors reported from external calls using the gateway. For more information, refer to the **Monitoring a Gateway Integration Endpoint** section.
- **Log**: provides information about changes to the gateway configuration.
- **Status**: provides information about revisions, hidden values and diagnostics.

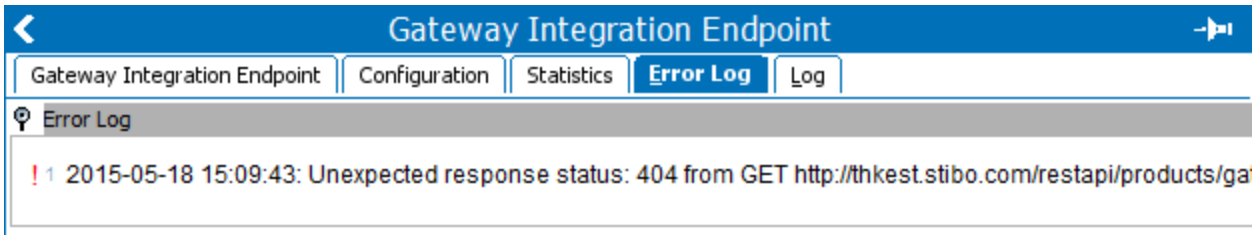
Monitoring a Gateway Integration Endpoint

Errors may occur when business actions are used to access gateway endpoints. Typical errors include not being able to access the external system or the external system not containing the data being requested in the business action. Errors can be monitored from within STEP or by directly accessing the endpoint via a URL.

Monitor from STEP

Any unexpected response status is received and logged in the Error Log tab on the gateway integration endpoint. This provides an overview of all logged errors on the gateway integration endpoint.

In the following example, an unexpected response status is logged when a method has been executed to retrieve data on an object which did not exist on the external system.



Monitor Outside of STEP

The following URLs allow you to get an overview of the status and any errors on processes without logging onto STEP.

View all endpoints

Use the following URL, replacing <appserver> with your own application server name.

```
http://<appserver>/admin/monitoring
```

For example:

```
http://myappserver/admin/monitoring
```

Endpoints are listed by name in the Sensor column of the list displayed.

View a specific gateway endpoint

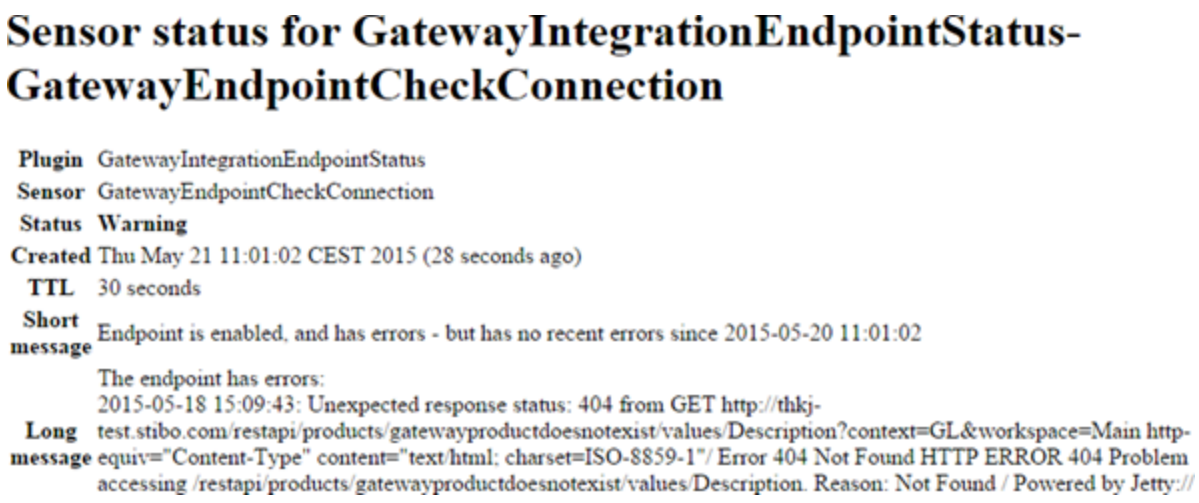
Use the following URL, replacing <appserver> with your own application server name, and replacing <Gateway-ID> with the STEP ID of the desired endpoint. The ID is displayed in the gateway integration endpoint editor. For more information, refer to **Maintaining a Gateway Integration Endpoint**.

```
http://<appserver>/admin/monitoring/GatewayIntegrationEndpointStatus-<Gateway-ID>
```

For example:

```
http://myappserver/admin/monitoring/GatewayIntegrationEndpointStatus-
GatewayEndpointCheckConnection
```

A report similar to the one below is displayed for the identified gateway endpoint with the ID 'GatewayEndpointCheckConnection.'

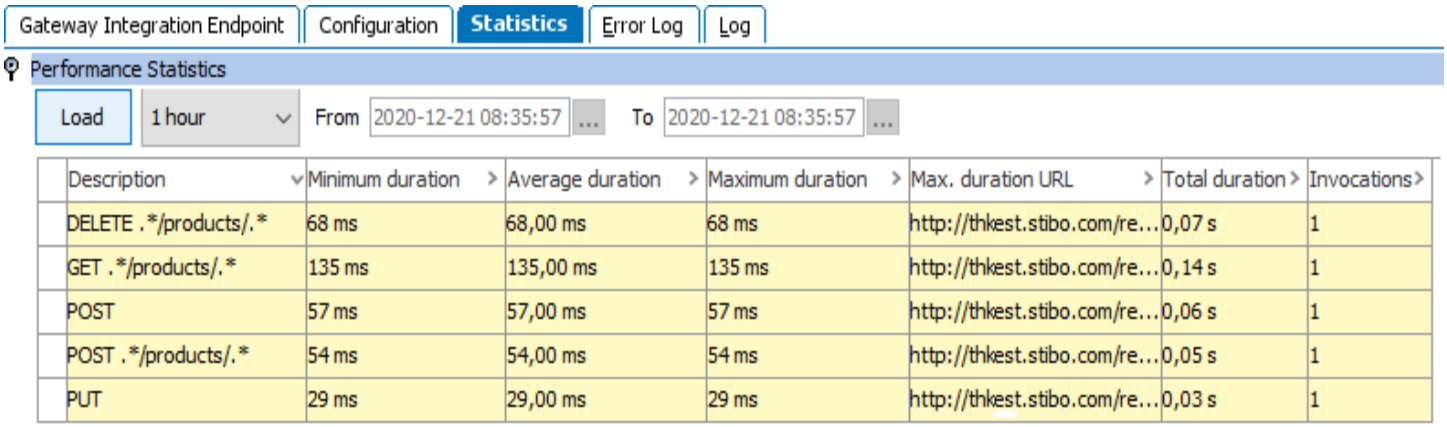


Sensor status for GatewayIntegrationEndpointStatus-GatewayEndpointCheckConnection

Plugin	GatewayIntegrationEndpointStatus
Sensor	GatewayEndpointCheckConnection
Status	**Warning**
Created	Thu May 21 11:01:02 CEST 2015 (28 seconds ago)
TTL	30 seconds
Short message	Endpoint is enabled, and has errors - but has no recent errors since 2015-05-20 11:01:02
Long message	The endpoint has errors: 2015-05-18 15:09:43: Unexpected response status: 404 from GET http://thkj-test.stibo.com/restapi/products/gatewayproductdoesnotexist/values/Description?context=GL&workspace=Main http-equiv="Content-Type" content="text/html; charset=ISO-8859-1"/ Error 404 Not Found HTTP ERROR 404 Problem accessing /restapi/products/gatewayproductdoesnotexist/values/Description. Reason: Not Found / Powered by Jetty://

Gateway Integration Endpoint Statistics

The Statistics tab includes information for executed REST methods grouped by the HTTP request types (GET, POST, PUT, and so on).



Description	Minimum duration	Average duration	Maximum duration	Max. duration URL	Total duration	Invocations
DELETE .*/products/.*	68 ms	68,00 ms	68 ms	http://thkest.stibo.com/re...	0,07 s	1
GET .*/products/.*	135 ms	135,00 ms	135 ms	http://thkest.stibo.com/re...	0,14 s	1
POST	57 ms	57,00 ms	57 ms	http://thkest.stibo.com/re...	0,06 s	1
POST .*/products/.*	54 ms	54,00 ms	54 ms	http://thkest.stibo.com/re...	0,05 s	1
PUT	29 ms	29,00 ms	29 ms	http://thkest.stibo.com/re...	0,03 s	1

For more granular groupings, regular expression can be used to display individual REST calls. This is set up on the Configuration tab > Gateway Configuration flipper > 'Statistic groups' parameter. For more information, refer to the **Statistic groups** section of the **Configuring a Gateway Integration Endpoint** documentation or the **Regular Expression** topic in **Resource Materials** online help.

Import Manager

The Import Manager allows you to import data on demand and to save or modify an import configuration. The final step of the wizard allows you to save the import configuration. This is useful for frequently-used imports where the parameters and options are always, or nearly always, the same.

Importing formats that require mapping gives access to a number of data quality features using transformations. The mapping process allows you to identify errors in the data before importing, to trim white space, or otherwise manipulate data to meet your specifications. Checking the data prior to import can prevent background process failures due to invalid data formatting.

Setup Requirements

Setting up and using the Import Manager involves the following steps:

1. Based on the data being imported, determine how to start the Import Manager wizard as described in **Creating a Data Import**.
2. In the wizard, choose a previously saved configuration or start a new import as described in **Import Manager - Select Configuration**.
3. In the wizard, select the data file to be imported as described in **Import Manager - Select Data Source**.
4. In the wizard, select the format of the data file, which determines the import method as described in **Import Manager - Select Format**.
5. In the wizard, when required, select which pieces of information are imported as described in **Import Manager - Map Data**.
6. In the wizard, when required, specify if imported data is new or if it matches data already in STEP as described in **Import Manager - Identify Objects**.
7. In the wizard, specify how and where information is to be imported as described in **Import Manager - Identify Destination**.
8. In the wizard, specify business conditions and/or actions that affect the import as described in **Import Manager - Select Business Rules**.
9. In the wizard, when required, select a context and specify if imported data should be modified as described in **Import Manager - Advanced Settings**.
10. In the wizard, save the configuration (if necessary), and run the import as described in **Running a Data Import**.
11. Monitor the import and download the output file as described in **Monitoring a Data Import**.

Additional Information

The following information is useful once an import configuration is saved:

1. Maintain a saved configuration as described in **Maintaining a Saved Import Configuration**.

Creating a Data Import

The Import Manager wizard allows you to import data on demand and to create or modify an import configuration. The final step of the wizard allows you to save the import configuration. This is useful for frequently-used imports where the parameters and options are always, or nearly always, the same.

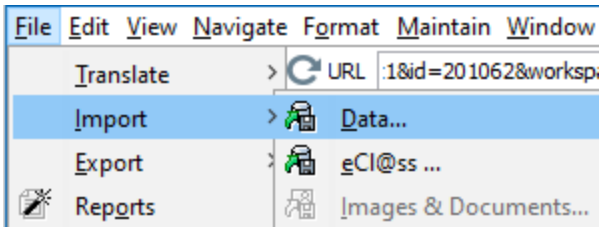
The user running the import should have the following setup privilege action sets: Create domain, Create value in hard domain, Create value in medium domain, Delete domain, Merge entire domain, and View domain. For more information, refer to **Action Sets** in the **System Setup** documentation.

Important: Excel binary files, those with an XLSB extension, are not supported during import.

To import data using the Import Manager:

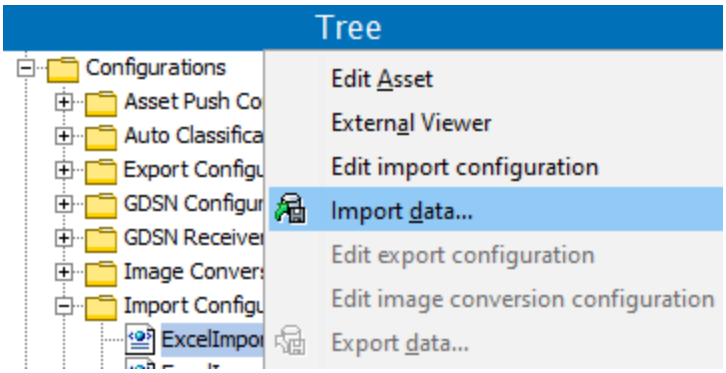
1. Verify that all attributes and reference / link types being imported already exist in STEP.
2. Use one of the following methods to launch the Import Manager wizard, noting that using an import configuration pre-loads the import file and settings.
   - **From the File Menu**: Click the File menu > Import > **Data**.

     The wizard opens in Step 1 and allows you to select an existing configuration or create a new one.

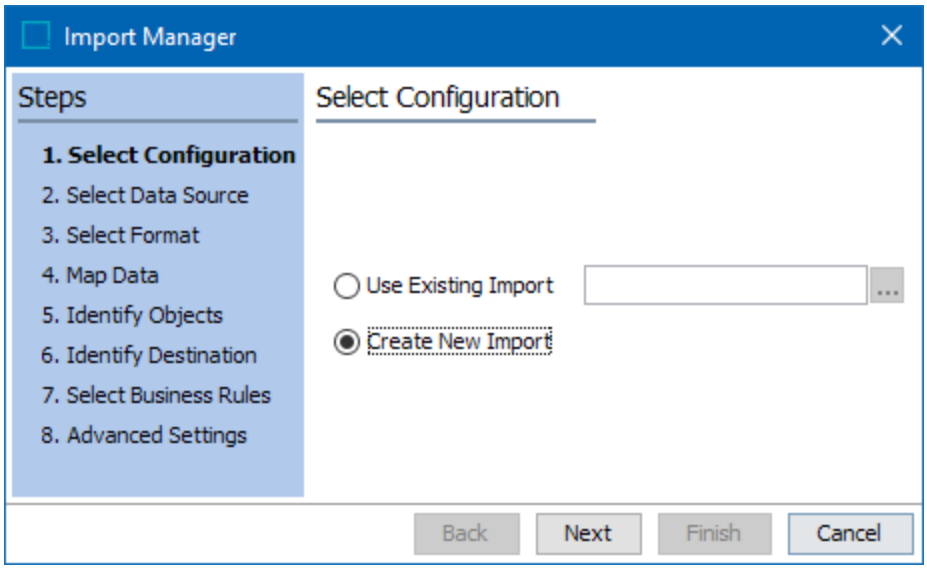
     

   - **From an Import Configuration**: On the Tree, select an import configuration, right-click, and then select **Import Data**.

     The wizard opens in Step 2 with the configuration and objects included in the configuration already selected.

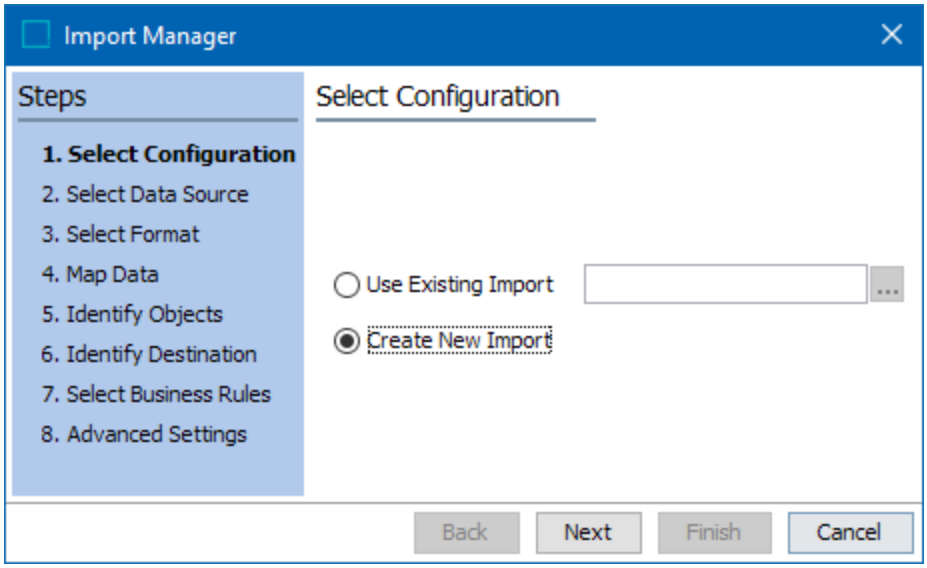
3. The Import Manager wizard displays and can involve the following steps:



- **Select Configuration** allows you to specify an existing import configuration or create a new one. If you frequently load data files manually, and they have the same data in the same columns, you can save a configuration with all necessary settings.
- **Select Data Sources** allows you to select the source of the data. If the source is a file, open a browser and select the file that you want to use. You can also choose an FTP connection.
- **Select Format** allows you to select the type of file to import and the basic parameters of the file.
- **Map Data** when required, allows you to map the columns of data in the import file to the equivalent fields in STEP.
- **Identify Objects** allows you to confirm if objects in the input file are new or existing and specify to locate objects by attribute value (when the file does not contain STEP IDs), by the object ID, or by name.
- **Identify Destination** allows you to specify a default object type and parent for new objects.
- **Select Business Rules** allows you to select business conditions and business actions to test / execute in relation to the data being imported. Data to be imported is processed based on the selected business rules.
- **Advanced Settings** allows you to set infrequently used operations, such as removing attribute values that are not listed in the input file or setting the approval status of imported objects.

4. Follow the steps defined in **Running a Data Import** to start the import background process.

Import Manager - Select Configuration



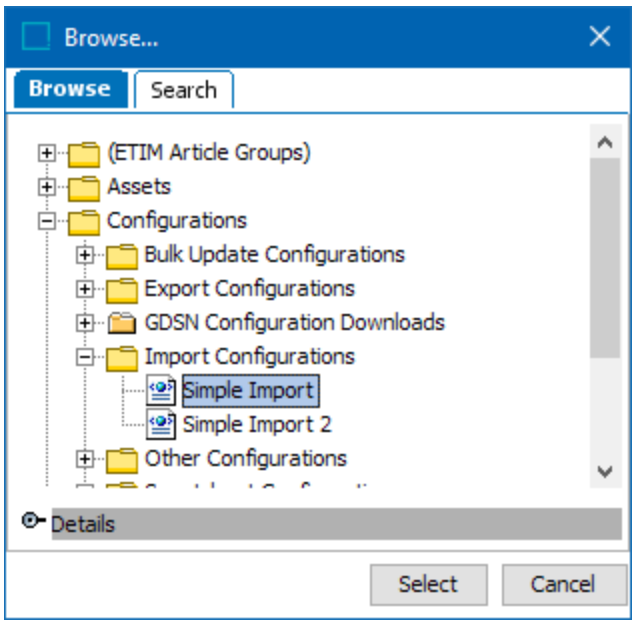
Once created, an import configuration can be saved as an asset in STEP for use with other import files. A configuration file contains an XML representation of the parameter settings in Import Manager.

1. Select a configuration option:
   - **Use Existing Import** - allows you to select a previously saved configuration. Click the ellipsis button (...) to browse or search for the configuration in STEP.
   - To create a new configuration, Select **Create New Import**.

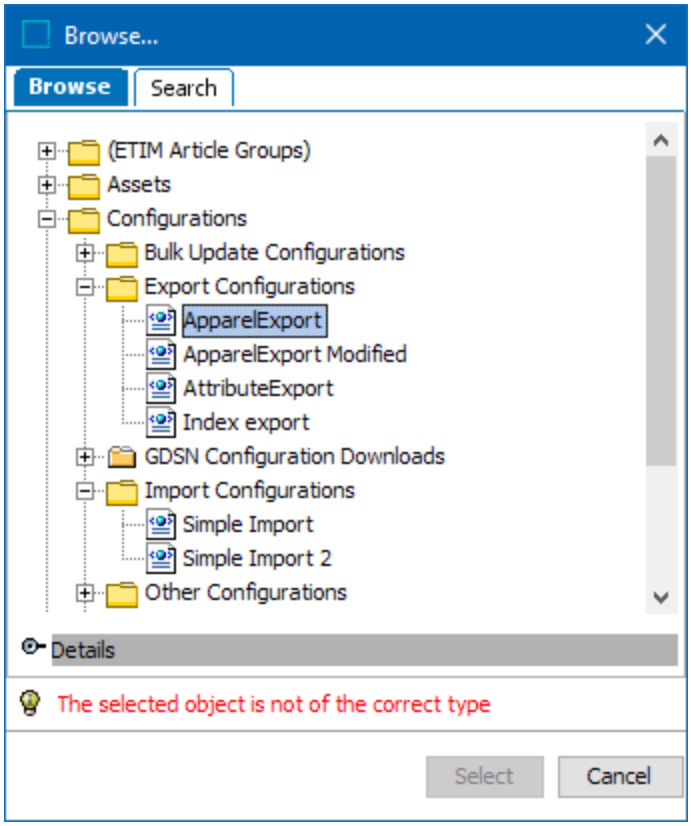
2. Click **Next** to display **Import Manager - Select Data Source** or when using an existing configuration, if no additional changes are necessary, click **Finish** to display Save Import Configuration dialog as defined in **Running a Data Import**.

Using a Saved Import Configuration

For example, if a user frequently loads data files that always have the same attributes in the same columns, they can store a configuration that has all the import options already set, including the mappings of the columns etc. This way, all the user has to do is select that configuration and (if necessary) point to the file they want to load.



Note: Selecting an incompatible configuration, such as a bulk update or export configuration, displays a message and the user must make a new selection.

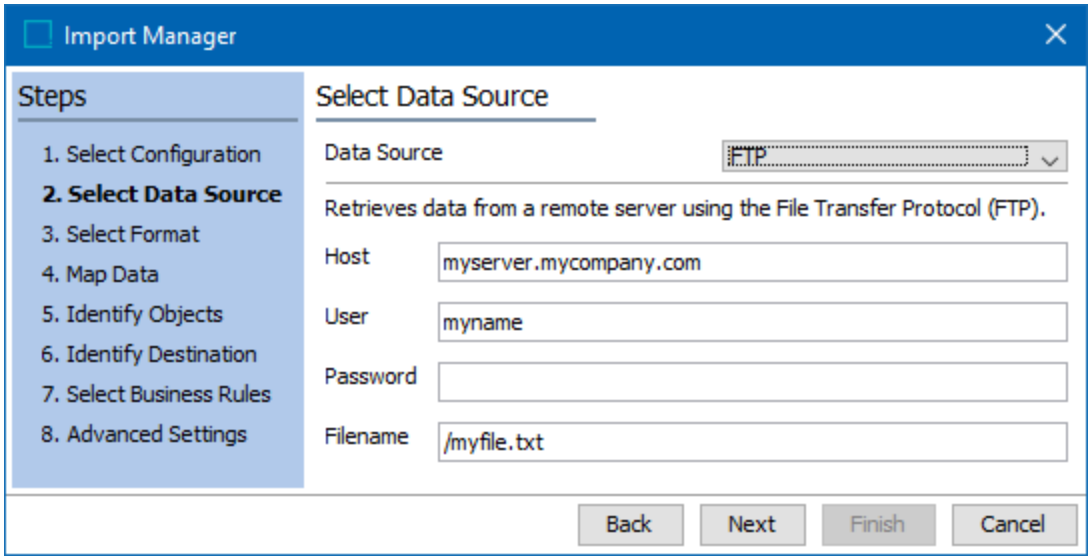


Import Manager - Select Data Source

If you selected an import configuration in the previous step, the saved data is displayed. When creating a new import configuration, you must select the source from the following options. The steps for each are described below.

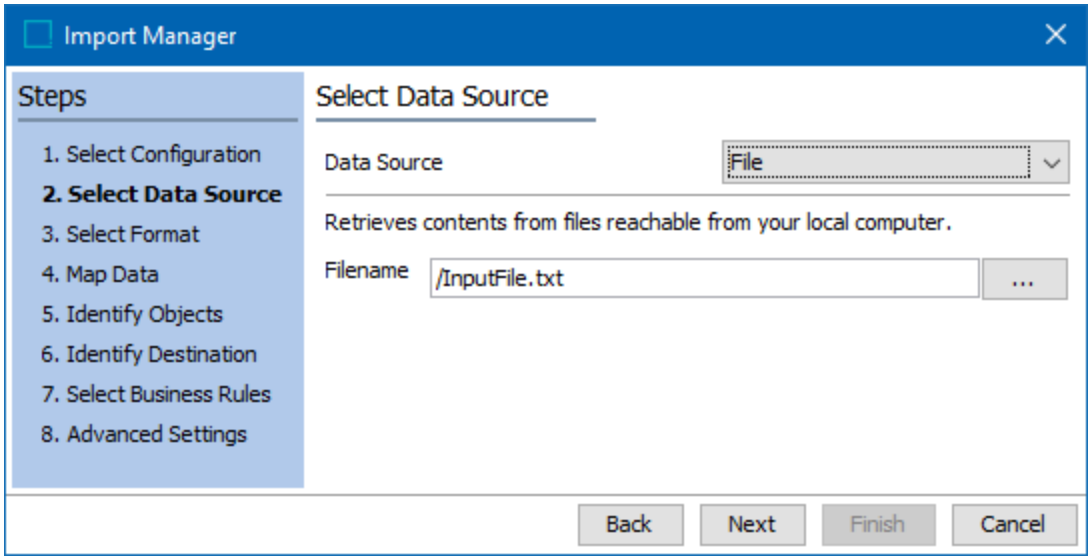
- **FTP** allows you to select a file from an FTP site.
- **File** allows you to select a file from your local hard drive or from your network.

Using FTP



1. Select **FTP** from the Data Source list.
2. Enter the **Host** name, the **User** name, and the **Password** (if any).
3. Enter the **Filename** including the path, and then click **Next** to display **Import Manager - Select Format**.

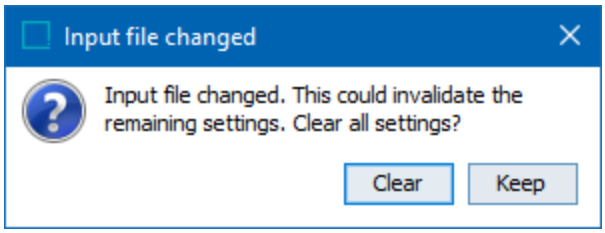
Using File



It is common practice to copy very large files to your local drive prior to importing.

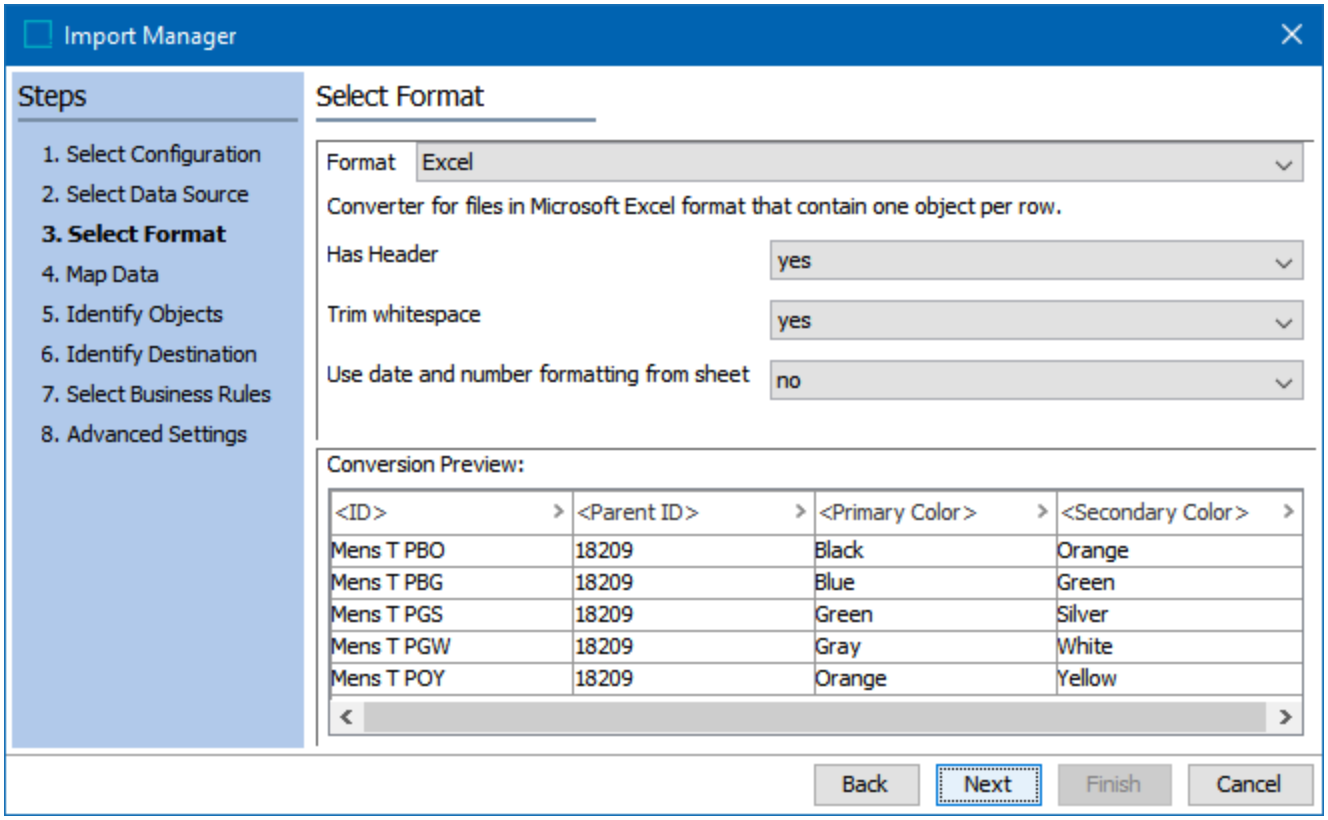
1. Select **File** from the Data Source list and click the ellipsis button (**...**) on the Filename field.
2. Locate the relevant file, and then click **Select File**, and then click **Next** to display **Import Manager - Select Format**.

   When using a previously saved configuration (described in **Running a Data Import**), selecting a file that is different from the one saved in the configuration can result in invalidating the configuration settings. The following dialog allows the user to proceed with the saved settings, or clear them.



   - Click **Clear** to remove the configuration settings and then reset them manually. Use this option, for example, when the configuration expects a Fixed Width file but the newly selected file is an Excel file.
   - Click **Keep** to apply the saved configuration settings to the new file. Use this option, for example, when the new file selected is in the layout expected by the configuration but the data is different.

Import Manager - Select Format



Select the format of the file manually if it is not automatically detected by the Import Manager. Depending on the format selected, a number of additional settings become available. Before proceeding, some formats populate the Conversion Preview section to allow for visual inspection of the data.

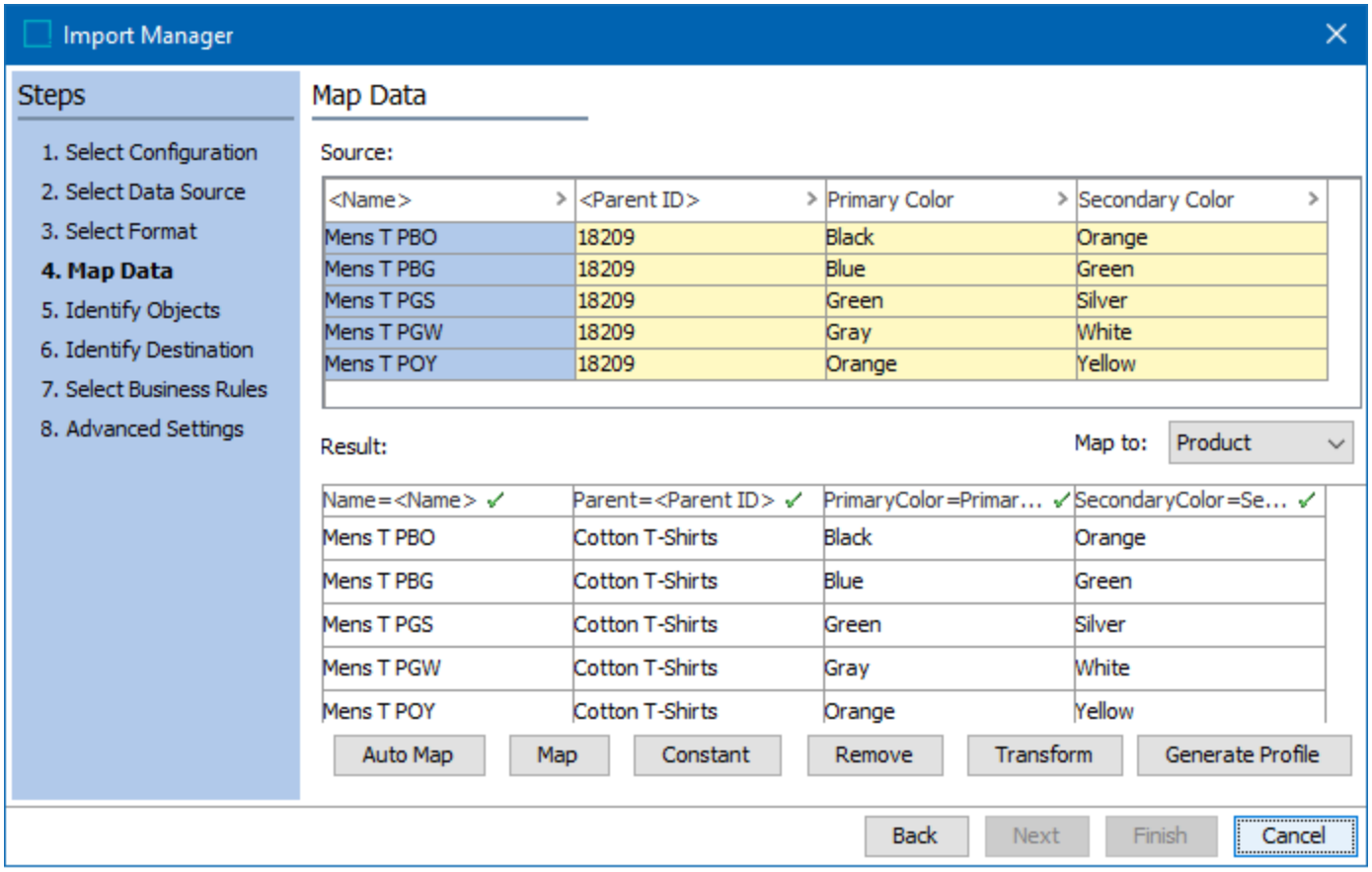
The following formats are available:

- Advanced STEPXML Format
- BMEcat Format
- BMEcat 2005 Format
- CSV Format
- ECLASS Format
- ETIM Format
- ETIM 6 Format
- ETIM v2 Format
- Excel Format
- Excel Smartsheet Format
- FixedWidth Format

- Flatplanner Format
- Generic XML Format
- IDoc MATMAS 05 Format
- Publication Excel Format
- STEPXML Format
- UNSPSC Format

Click the **Next** button to display **Import Manager - Map Data** when required, or **Import Manager - Identify Objects** if mapping is not needed.

Import Manager - Map Data



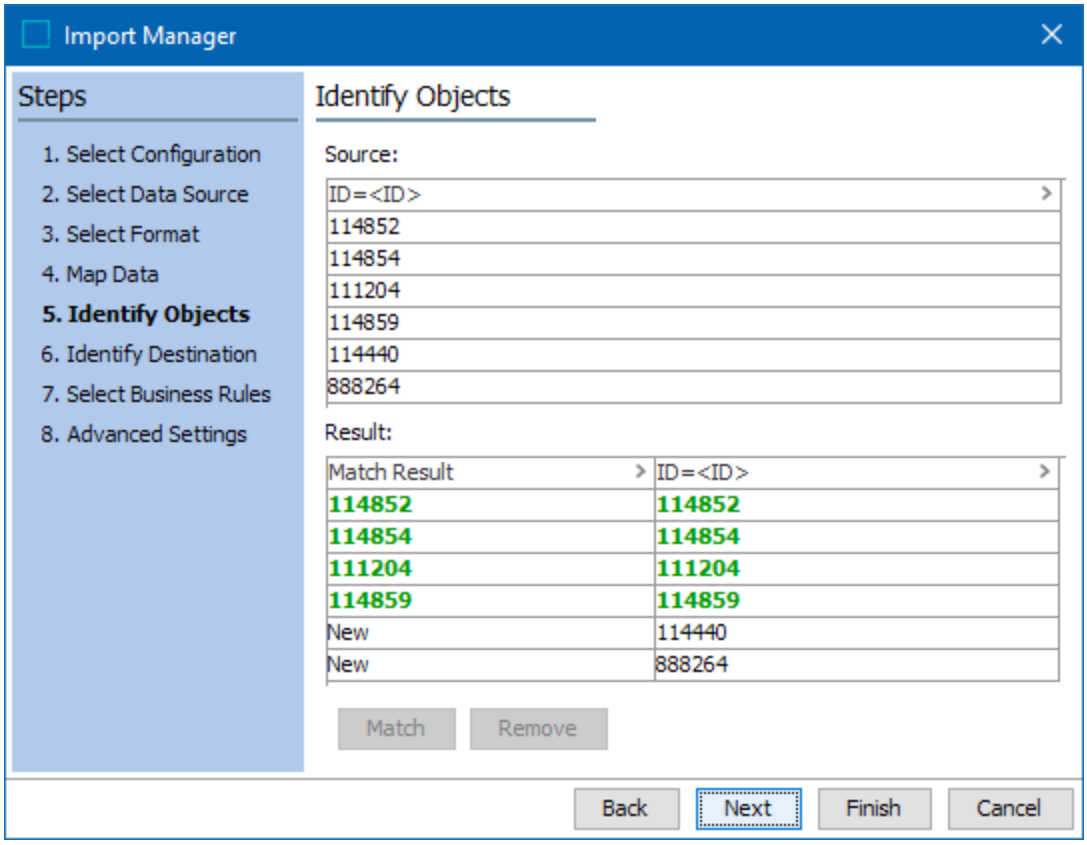
Mapping is required when importing some file types. Mapping relates the import data to the appropriate element in the STEP data model and specifies the desired object type.

The mapping step for Import Manager and IIEPs include the same options. In addition to mapping data, additional functionality is available that allows you to modify data and generate a profile to verify the quality of the inbound data.

For information about the formats that require mapping and all of the available options, refer to the **Inbound Map Data Options** section.

Click the Next button to display **Import Manager - Identify Objects**.

Import Manager - Identify Objects



Identify Objects allows you to Identify Objects allows you to verify if loaded objects are existing data in STEP or will be created as new objects. Additionally, you can specify to locate existing objects by attribute value rather than by the ID or name.

Importing With Object IDs

When the ID has been mapped, the Result panel displays the first 20 results:

- Objects found in STEP are displayed in green
- New objects display 'New' in the Match Result column

You cannot introduce new products by using Identify Objects (in this step). When importing with object ID, all products must exist already in STEP. If there is no match of the attribute value to a Object ID, the record is ignored.

You can select any attribute that you have already mapped in the prior Map Data screen, but you can only match on one attribute. For example, if you have several attribute columns that are 'alternate part numbers' or 'replacement part numbers', you cannot select multiple attributes at one time in order to find the actual Object ID in STEP.

Matching can only be done on one attribute at one time. However, you can try several columns on the same import by using the Remove button and by selecting another attribute column in the top panel.

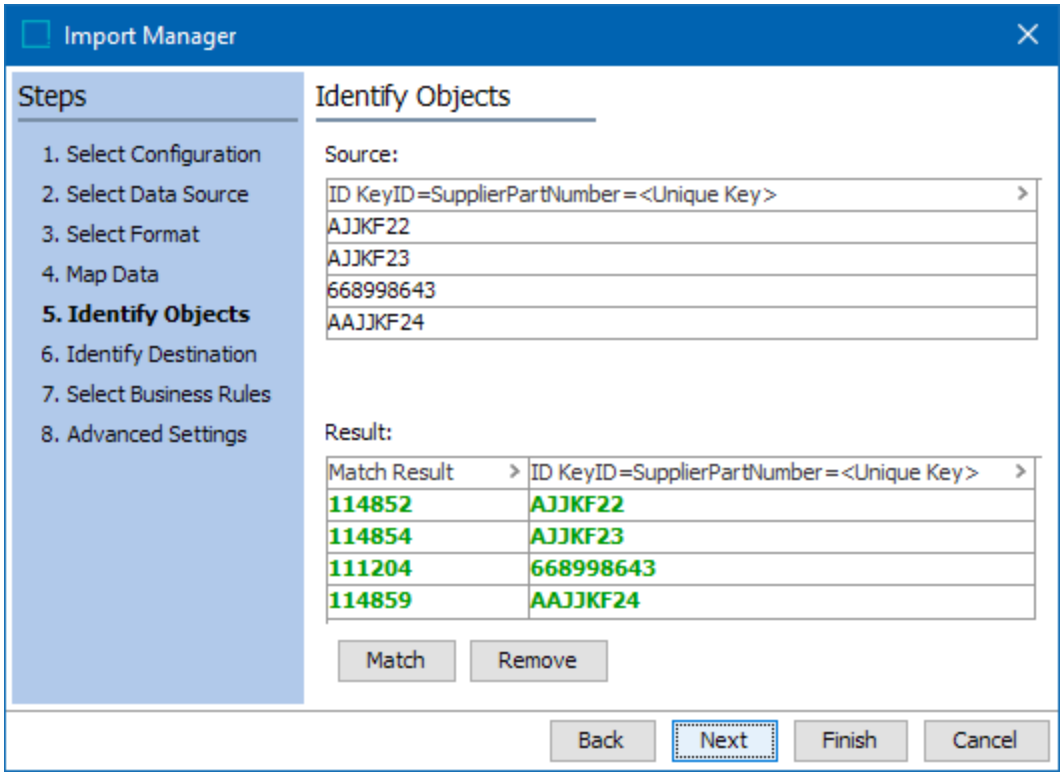
Click the Next button to display **Import Manager - Identify Destination**.

Importing with Unique Key

Note: Unique Key must be activated before the import process is started.

When the key has been mapped, the Result panel displays the first 20 results:

- The Result pane shows a column respective IDs for the unique key.
- Objects found in STEP are displayed in green.



Importing Without Object IDs

If new objects without IDs are to be imported, an ID Pattern must be set for the target object type. Otherwise the objects cannot be created.

You can select any column that you have already mapped in the prior screen, the Map Data screen, but you can only match on one column. So if you have several columns that are 'alternate part numbers' or 'replacement part numbers', etc. you cannot select multiple columns at one time in order to find the actual Object ID in STEP. Matching can only be done on a column at once. However, you can try several columns by using the 'Remove' button and by selecting another column in the top panel.

Identifying Objects by Attribute Values

When matching on an attribute other than STEP ID, prior to running live imports, test typical data to determine if performance is negatively affected.

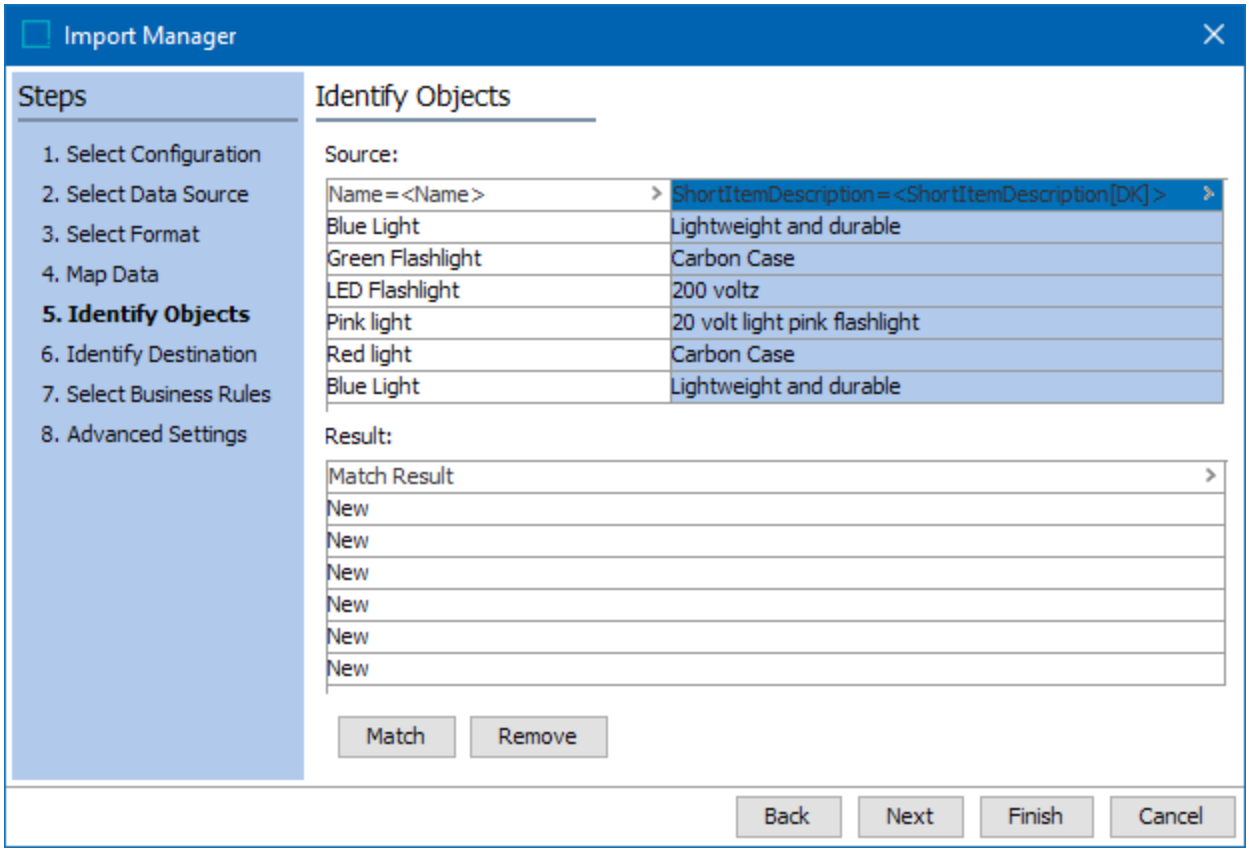
The following example describes how to identify products by attribute values. The product data is loaded from a file with no product IDs listed but contains the manufacturer's own part numbers.

Important: New products are not loaded to STEP when they are located by attribute values. Even if you have set up the system to auto-generate an ID for new products, and even if you allow new products to be created, the system rejects any new products in this file when the products are located by attribute values.

Normally, the manufacturer part number would be an attribute of your own product. If the manufacturer's part number is unique, you can then load the file by having the system locate the products by an attribute value – the manufacturer's part number - instead of mapping a column of data to the product ID as you would normally do.

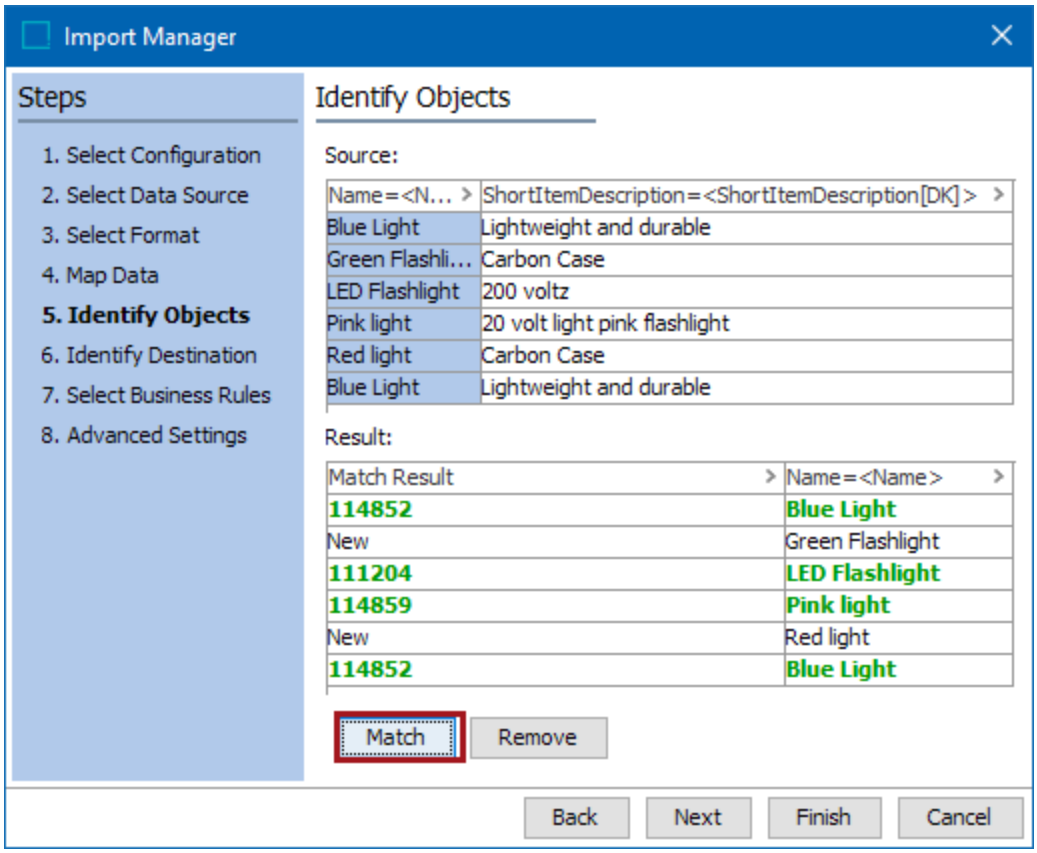
A file with two columns of data is imported to STEP. Both columns are mapped to attributes. The first one is mapped to the attribute 'Manufacturer Part Number', and the second is mapped to 'Description.'

1. Select a column in the top panel, and click **Match**. The system attempts to locate products by the values in that attribute.



Note: The Match button is only enabled if you have not mapped a column to a product ID in the Map Data step. STEP assumes that if you defined a column of data as the ID, no matching of attribute values to Objects IDs is required.

If the match is successful, the result is displayed in green in the **Results** pane.

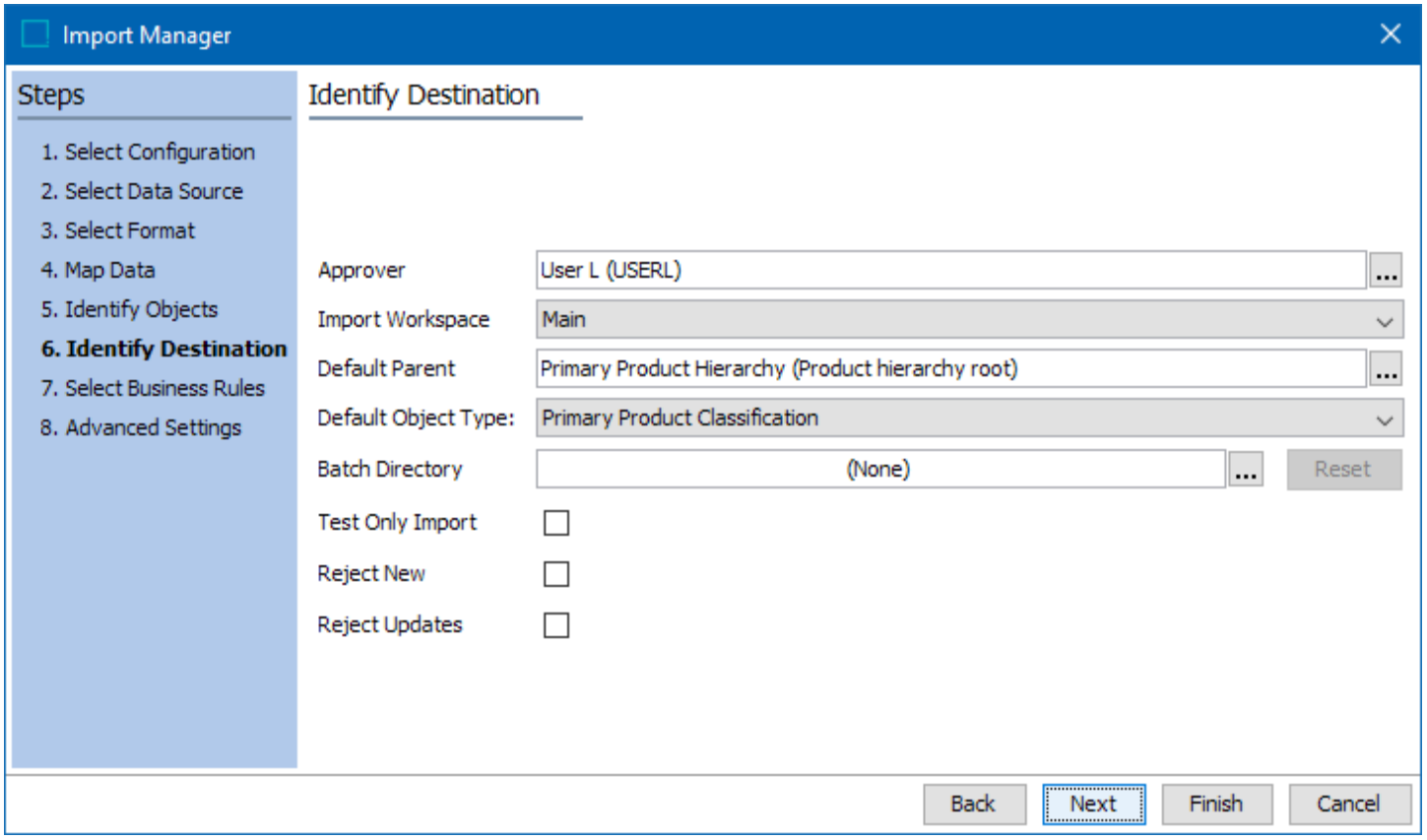


The first column is the ID of the product in STEP. The second column is the attribute value that matches that product.

Match only checks the first 20 products in the file. If it is unable to find a match it the result in the first column is 'New.'

2. Click the Next button to display **Import Manager - Identify Destination** or click **Finish** if you have made all required selections and want to launch the import process as defined in **Running a Data Import**.

Import Manager - Identify Destination



- **Approver** is used to specify the user name of the person that is allowed to approve the products after the import process is complete. When you click the ellipsis button (...), you can search or browse for the user.

  You are the default Approver. Normally, this is used in conjunction with a customized workflow. Refer to the 'Approve Import Changes' option on the Advanced Settings step in this wizard.

- **Import Workspace** is used to specify which workspace the data is loaded into. You can use this feature if you have more than one maintenance type of workspace in your system, such as 'Staging', created prior to the 9.2 release. The default is Main. You can never import data into the Approved workspace.

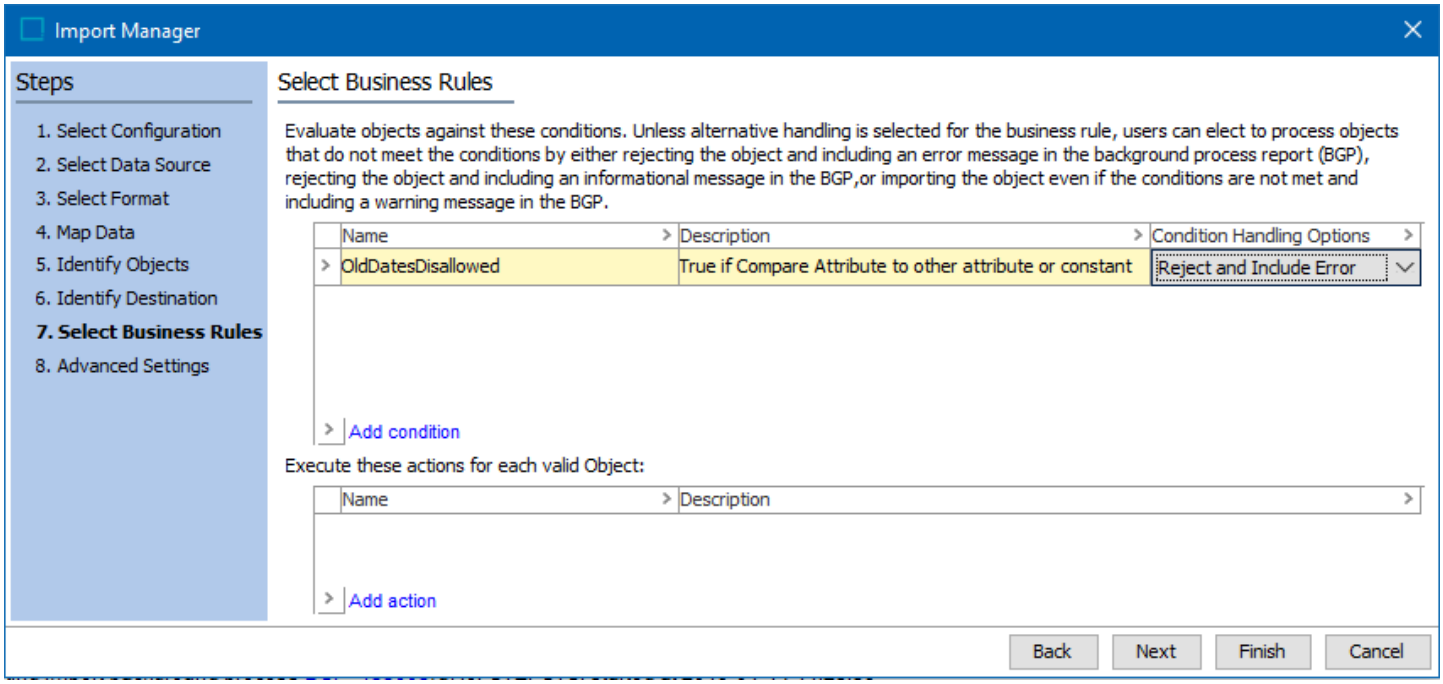
- **Default Parent** is used to specify where new products should be placed in STEP by default, if no Parent ID is provided. Normally, a parent product ID is required when creating new products in STEP. However, sometimes you may not know who the parent product is or will be. You may thus place the new products into a holding area or default area for later processing. Use this Default Parent parameter if no column has been mapped to the Parent ID, or if the entry in the Parent ID column is empty. If any records in your load file are new products, and they do not have a Parent ID specified, without this Default Parent entry they will not be created in STEP.

For all object types other than Products, the Parent ID must always be present in the load file, as there is no Default Parent functionality. When parent data is missing in the import file, it can sometimes be derived via transformations or by mapping a constant value in the Map Data step.

- **Default Object Type** is used to specify which object type to use when there are new products in the load file. **An object type is required to create new objects, entities, assets, and classifications in STEP.** Valid object types are based on the selected default parent product. It is not unusual to have different object types for categories, subcategories, families, or individual items. It is also not unusual for objects to have several possible object types, depending on your system setup. Common setup is to use this feature when all products in the load file are of the same object type.
- **Batch Directory** is an obsolete option and selecting a directory has no effect on the import. Instead of using this option to link products to a classification folder, use the Map Data step to link a product to one or more classifications with the specified Reference / Link Type. For more information on importing to classifications, refer to the **Map Product Classification Links** section of the **Inbound Map Data - Map** documentation.
- **Test Import Only** is used to load a small file into STEP without saving or committing any changes. No data is imported into STEP when this option is checked. The Import Manager wizard goes through all the steps of importing the file, but does not perform the final step of confirming the changes in STEP. This allows you to review the execution report so that you address any warnings or errors encountered when loading the file. You can the make appropriate changes to the load file and do the load again, until all errors have been resolved.
- **Reject New** is used to specify a reject to any new products that are listed in the load file. If selected, new products are not loaded or created. The execution report lists the products that have been rejected as a result of this selection. This is useful, for example, when loading price files.
- **Reject Updates** is used to specify whether only new products should be imported from a load file that contains both new and existing products. This way you avoid updating existing products in STEP. Existing products are left untouched.

Click the **Next** button to display **Import Manager - Select Business Rules**.

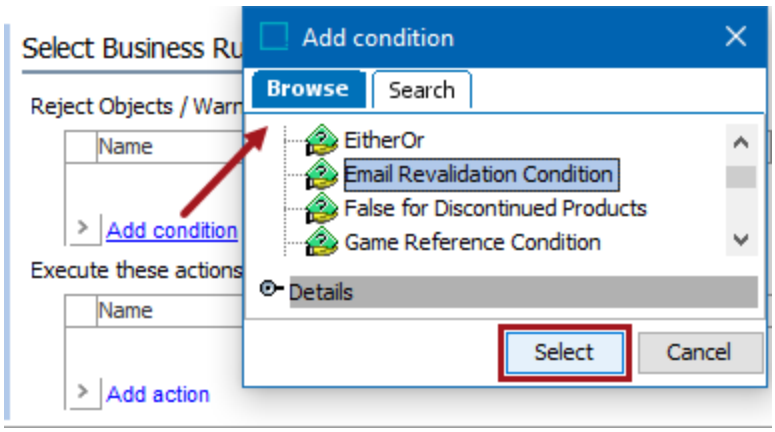
Import Manager - Select Business Rules



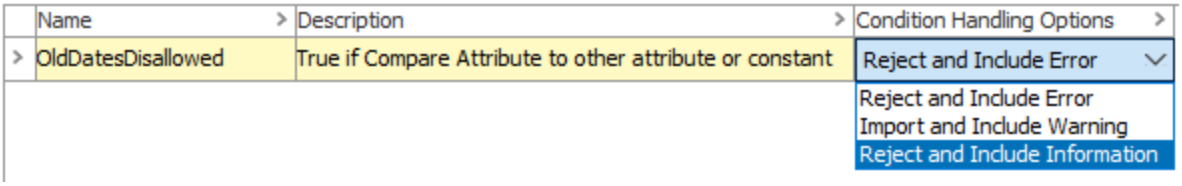
Import data is processed based on selected business rules. All global business rules configured on a given system can be selected in the 'Select Business Rules' step. First, selected business conditions are validated against the objects being imported. Depending on the severity selected per business condition, failing conditions can be reported as either errors or warnings. Second, selected business actions are executed on the valid objects. If a business action results in an error, the import process skips the object and logs the error information within the Import Execution Report. For details on validation order, refer to the **Business Rule Limitations** section below.

Important: The combination of importing references, running validation business rules, and the 'Approve Import Changes' parameter in the Advanced Settings step can have unexpected outcomes. For details, refer to the **Business Rule Limitations** section below and the **Import Manager - Advanced Settings** topic.

1. If necessary, click the **Add condition** link to add conditions. Then, use Browse or Search tabs to locate the condition, and click the **Select** button. Repeat until all the relevant conditions are added.



Once a condition is added, users may select the severity level for objects rejects based on the business condition.

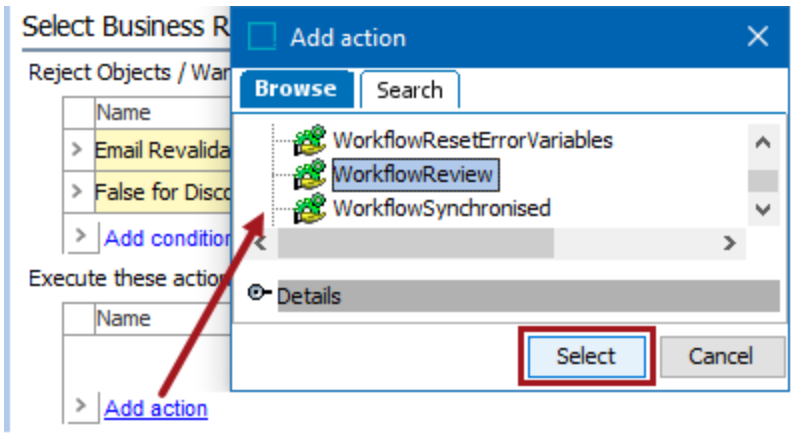


The three available options are:

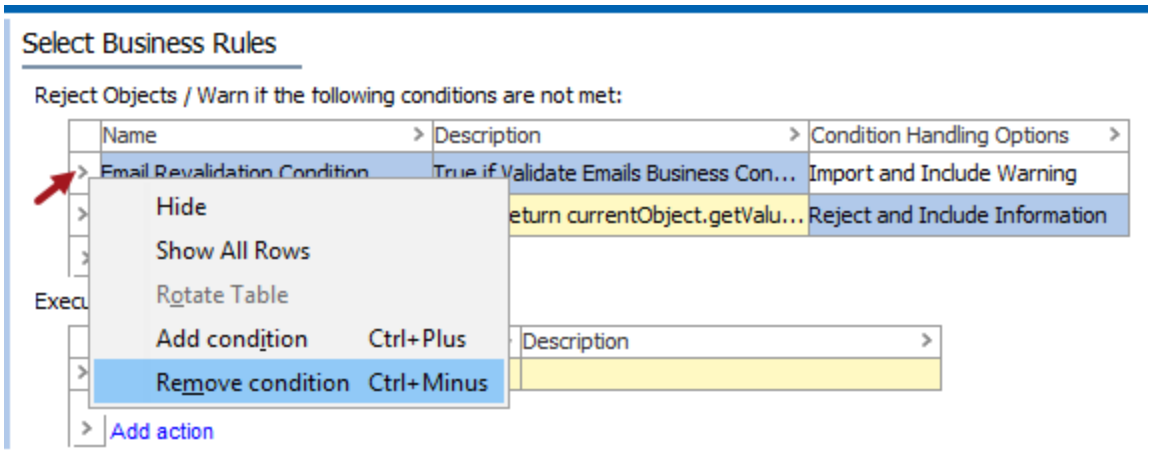
- **Reject and Include Error** - when selected, objects rejected on import will be listed on the execution report as errors, and the import's background process (BGP) may finish with a status of 'Completed with Errors.' This setting is the default for each condition in the 'Condition Handling Options' column.
- **Import and Include Warning** - when selected, allows objects to be imported even if they do not meet the conditions set in the business condition, but will include a warning for each of those objects in the BGP's execution report.
- **Reject and Include Information** - when selected, the items that do not meet the conditions set by the business condition will be rejected and information about that rejection will be included in the BGP's execution report. The information listed in the execution report will not be labeled as either a warning or an error, and the BGP will not end with a 'Completed with Errors' status.

For more information on how the severity settings affect the execution report, refer to the **Business Rule Limitations** section below.

2. If necessary, click the **Add action** link to add actions. Then, use the Browse or Search tabs to locate the action, and click the **Select** button. Repeat until all the relevant actions are added.



3. If a **condition** must be removed, select it, right-click, and then select **Remove condition**.



4. If an **action** must be removed, select it, right-click, and then select **Remove action**.
5. Click the **Next** button to display **Import Manager - Advanced Settings** or click the **Finish** button to launch the import process as defined in **Running a Data Import**.

Business Rule Limitations

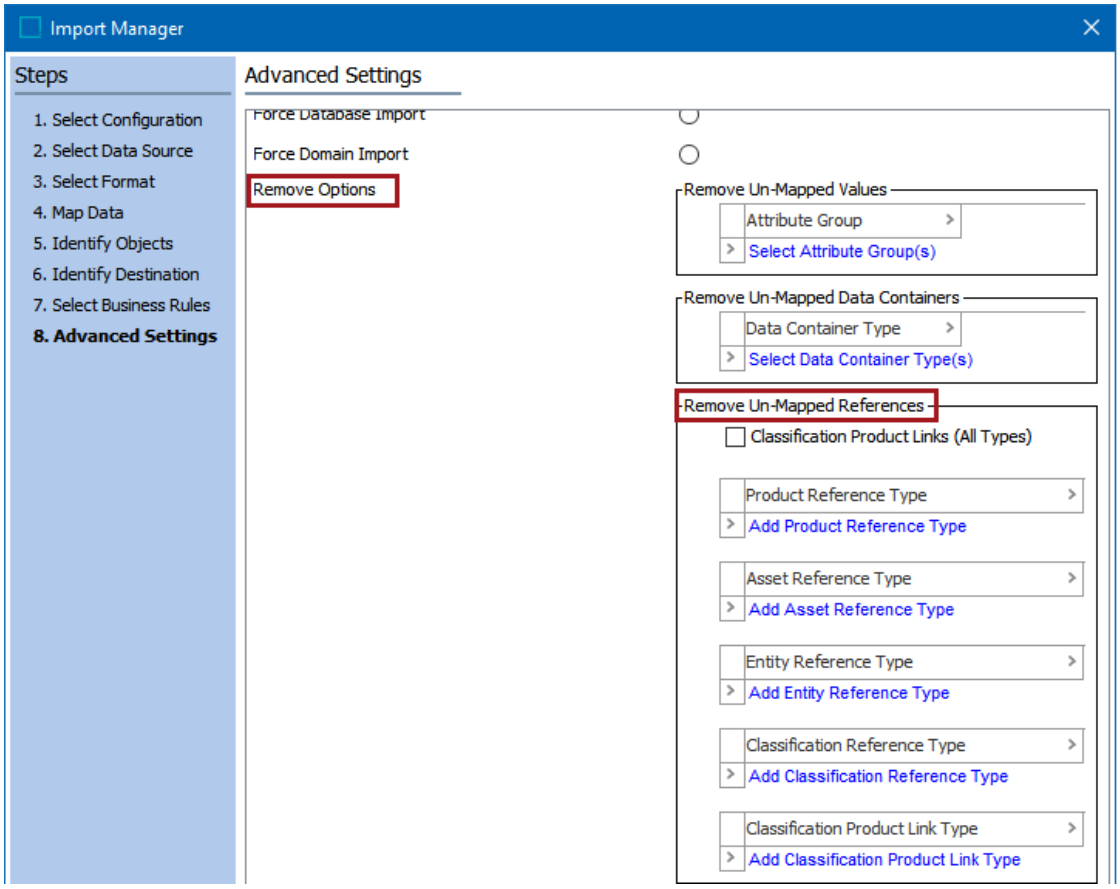
- Objects must be imported one at a time for the validation mechanism to work. It is therefore not possible to import nested STEPXML documents when you are using business rules.
- If one or more business rules are selected, the Import Manager uses domain mode.
- Based on the Condition Handling Option selected for each business condition, business conditions will run in the following sequence: those with 'Reject and Include Error' selected, those with 'Import and Include Warning' selected, and those with 'Reject and Include Information' selected.
- Changes are not detected for references and links. Objects with references in the import are therefore always reported as changed.
- Sometimes imported references are deferred if they depend on objects that have not been imported yet. As a consequence, the deferred parts of an imported object are not present while it is being validated by business

rules. (The references are deferred until after the business rules have run.) Therefore, use caution when using business conditions to validate references.

- When actions are used, the 'Approve Import Changes' on Advanced step is disabled - importer cannot automatically auto approve imported data because side effects from business actions are unknown. However it is still possible to approve imported objects via a business action.
- While the 'Reject and Include Error' option can be used for onboarding / enriching objects from suppliers via the Excel Smartsheet format, business conditions set with the 'Reject and Include Error' option will report rejected objects as 'hard errors' when they fail validation in the Smartsheet. Although the Smartsheet validation failed, the data is still imported and further validation can be performed within STEP using established enrichment procedures. For more information on Excel Smartsheets, refer to the **Excel Smartsheet Format** topic in this guide.

For more information on business rules, refer to the **Business Rules** documentation.

Import Manager - Advanced Settings



The Advanced Settings screen provides a series of options that are not required on a day-to-day basis for data imports. Options are displayed as disabled, or are hidden completely, when the user running the import does not have valid permissions to run the option.

Important: Advanced Settings options are generally used to clean up data within STEP. However, if you are not careful, you can cause data integrity issues. Double-check your load file and your System Setup before using any of these options. Common setup is to perform a test on a test server first, and to start testing with small files to verify the results before changing large amounts of data.

After setting the options as defined below, click the **Finish** button to display the Save Import Configuration dialog as described in **Running a Data Import**.

General Advanced Settings

- **Context** - If you are creating a new configuration, the context shown is the one that is currently selected in STEP. That is usually the safest way to import data. You can, however, select a context different than the currently select one. If you selected an existing configuration, the context that was set up in that configuration

is shown. Dimension Point specific mappings will overrule this setting. Consider the context setting when creating an Import Configuration that will be used repeatedly. You can only import to one context at a time, so if you have several languages, you will have to do several imports.

Note: If there are transformations applied in the Map Data step, to import the data in different contexts, the same will still be applicable. Additionally, when importing ETIM6, you selected a Language in the Select Format step. Verify that you choose the Context that should be updated with the selected language.

- **Import Process Template** - This is used to specify a workflow process. The default is Importer, which is the standard workflow.
- **Match Units By** - When the system loads data, it has the capability of trying to separate actual values from their units in the load file. Use this option to indicate if units in the data file should be matched on the unit's ID or the unit's name (default).
- **Approve Import Changes** - This option is used to approve the changes that the import causes. When checked, the import mode option 'Force Domain Import' is automatically selected and cannot be changed. If business rules are applied from the previous step this option cannot be selected - instead it is possible to approve the entire object from business rules.

Important: If business rules are not applied in the Select Business Rules step, imported reference approvals can get special treatment. If the reference target has not yet been imported, the reference import does not fail but is deferred. Once all objects are imported, the reference is attempted again, and if Approve Import Changes is checked, the approval is attempted again for each source of the imported, deferred references.

- **Auto-Initiate STEP Workflows on Item Creation** - If a workflow is valid for either products, classifications, assets, or entities, it is possible to have an instance of a given workflow created whenever an object of the valid object type(s) is created. That is, a workflow is automatically started when the object is created. This functionality is available for workflows set to Auto Initiate on Object Creation. Users with the privilege Disable STEP Workflow Auto-initiation in Imports are allowed to disable auto-initiation in the Import Manager for tabular imports, or directly in the XML for XML imports by setting AutoInitiate="N" in the STEP-ProductInformation tag.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<STEP-ProductInformation ContextID="Context1" WorkspaceID="Main"AutoInitiate="N"
```

- **Trigger STEP Workflow import events on Item updates** - This option controls whether or not the import triggers events. In XML, the functionality is enabled / disabled using the STEP-ProductInformation tag attribute 'STEPWorkflowImportEvent' that can be set to "Y" or "N." The option is typically used in imports based on pre-configured Import Configurations. When checked, the import mode option 'Force Domain Import' is automatically selected and cannot be changed.
- **Suppress Re-Translations** - Checking Suppress Re-Translation, prevents products, classifications, entities or assets from getting the status **Re-Translation Needed**. This option only has effect if the data import is set to be approved too.
- **Import Mode options** - If the previous two options 'Approve Import Changes' or 'Trigger STEP Workflow import events on Item Updates' are selected, these options are disabled and cannot be modified.

Important: The following options are displayed and enabled based two factors: the privileges of the user running the import and the content of the import. When importing products, the user must have all user privileges to the product hierarchy root (without any restrictions). When importing entities, the user must have all user privileges to the entity hierarchy root (without any restrictions). Restrictions could include some attribute groups, or dimension points, or having limited privileges assigned to any of the object type being imported.

Auto Select Import Mode - Choose this option to allow STEP to analyze the import file and its size and run either Domain or Database import mode, as appropriate. **Users should be aware that if the Database import mode is selected after the STEP analysis, then events are not generated for changes to externally maintained values during an import. If generating events is necessary for your operations, then using Force Domain Import (explained below) is recommended.**

Force Database Import - This option should only be used when the inbound data is of high quality. Since no additional privilege checks are made, this mode can allow for a faster import and is useful for very large imports, data migration, and when working with externally maintained data or commercial data. This option does not allow creating, editing, or deleting global revisable objects. Reference creation speed is the same for both Database and Domain import modes. There are further limitations for event generation (defined in the **Core Events** topic of **System Setup** documentation) and for business rule invocation (defined in the **Import Manager - Select Business Rules** topic). Database mode is not available when running In-Memory (defined in the **In-Memory Database Component for STEP** topic within the **Resource Materials** documentation in online help).

Force Domain Import - All user privileges are validated during the import, which can impede performance in this mode. Generally, this mode delivers a more detailed execution report. If access is not allowed, the import fails and the error "Not privileged to change value ..." is included in the import execution report. Reference creation speed is the same for both Database and Domain import modes.

Remove Options

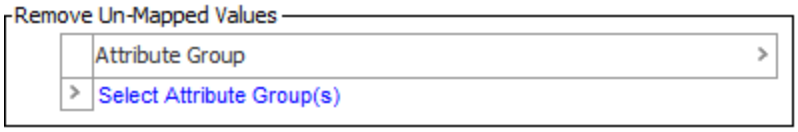
Setting any of the Remove Options means that any existing data for the selected option type will be removed on the objects included in the import file, and **only** the data of the selected option type for the objects in the import file will exist in STEP after the import. Essentially, the remove options allow you to remove existing data from STEP and replace it with the data in the inbound file for the objects being imported.

Leave the Remove Options set as default to add the imported data to the existing data in STEP.

Important: The Remove Options can both add and remove data from STEP. Prior to using them, it is a good idea to first create an export file of the data being modified. This provides a way to restore the unmodified data, in the event that the import does not have the desired effect.

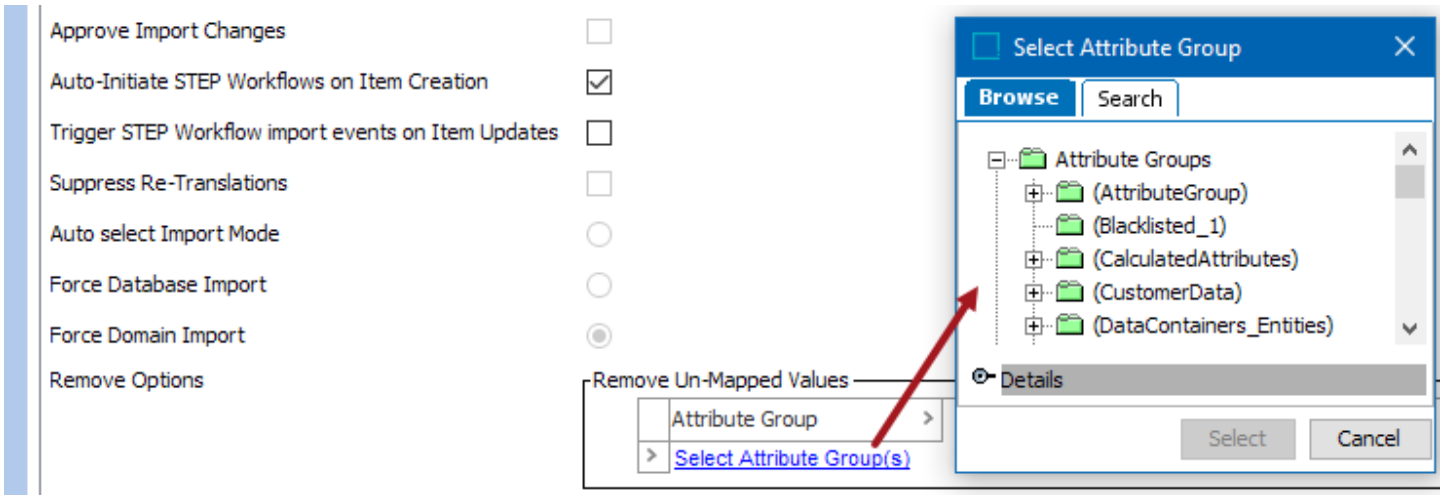
Remove Un-Mapped Values

This option enables you to remove attribute values from one or more selected attribute groups and replace them with the values specified in the load file, leave them blank, or return them to their inherited value.



Important: Use this option with care as it can both add and remove data from STEP.

Click the 'Select Attribute Group(s)' link. In the Select Attribute Group dialog, select the Attribute groups available in the STEP that should be modified by the import. Only values included in the import file will exist for the selected groups after the successful import.

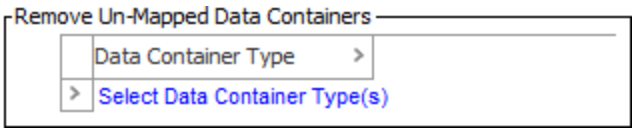


For example, you have 200 products listed in your load file, and they are all gloves of one type or another. You use eight product attributes to define these gloves. Now you want to use just four attributes. Furthermore, all of the original eight attributes are in their own attribute group. You load the 200 products using just the four attributes that you want to use, and you then select the attribute group under the 'Remove Un-Mapped Values' option. STEP removes all attribute values in that attribute group for those 200 products only, and then loads the attribute values as specified in the load file. Thus, it has removed all eight old attribute values for all 200 products, and it will show only the four attribute values present in the load file.

Important: When importing values for a multi-valued attribute, the STEP importer will not append, but instead replace, existing multi-valued attribute values with the new values that are being imported (e.g., if a multi-valued attribute in STEP has a value of 'x' and a value of 'y' is imported for that multi-valued attribute, the new value for that multi-valued attribute will now be 'y', not 'x' and 'y'). For more information on multi-valued attributes, refer to the **Single and Multi-valued Attributes** topic in the **System Setup** documentation.

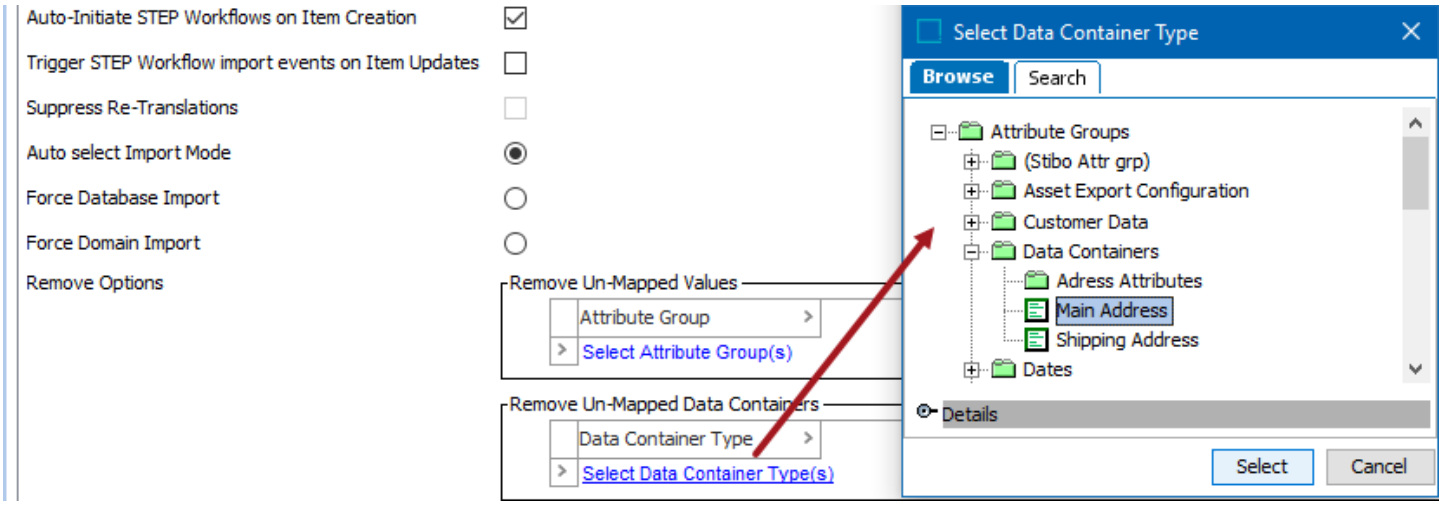
Remove Un-Mapped Data Containers

Data container types can be removed from the entities being imported into STEP via the Data Containers type option. A dialog allows you to confirm that you want to remove all unmapped types for the imported objects.



Important: Use this option with care as it can both add and remove data from STEP.

Click the 'Select Data Container Type(s)' link. In the Select Data Container Type dialog, select the Data Container type available in the STEP that should be modified by the import. After a successful import, only values for the selected Data Container Types included in the import file will exist on the imported objects.

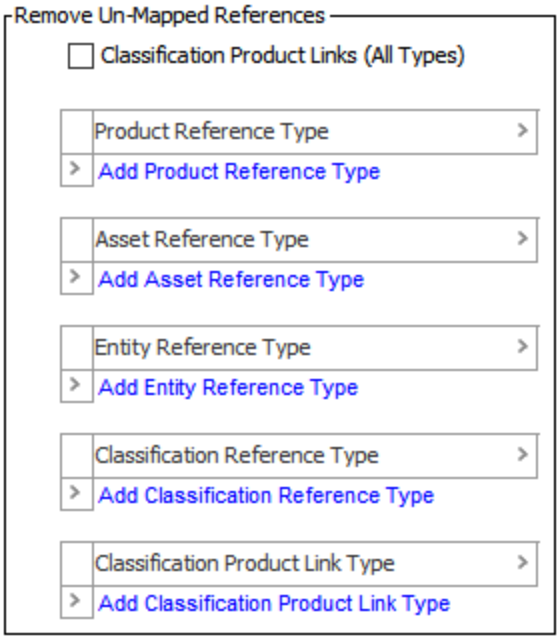


For example, you have a data container type that allows for multiple data containers (e.g., Shipping Address). Your warehouse usually ships to all the locations in Shipping Address, but now your warehouse has been designated to ship to addresses in a different data container type named Shipping Addresses - West. Instead of adding all new addresses to Shipping Addresses, you can do an import and un-map the original data container type and import only the applicable data container type and the associated data containers.

Remove Un-Mapped References

Selecting a reference option allows a user to specify which reference type they would like to remove. Information and examples for each reference type follows. For each type, a dialog allows you to confirm that you want to remove all unmapped types for the imported objects.

For more information, refer to **Reference Types** in the **System Setup** documentation.

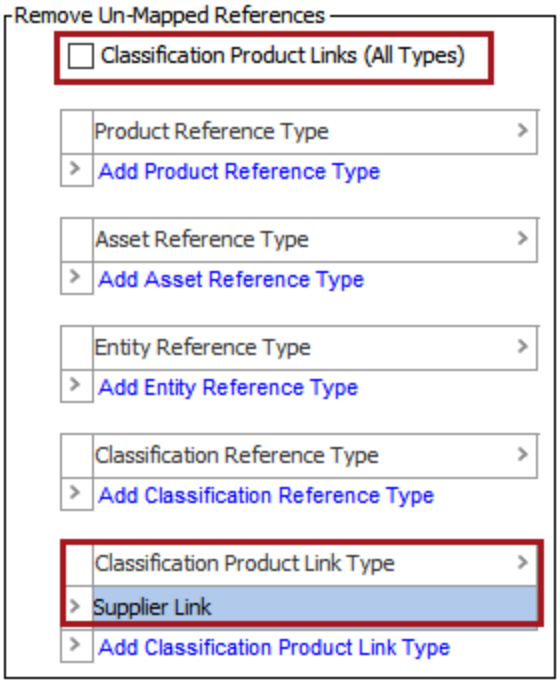


For details on how to add or remove references types, refer to the **Adding and Removing Un-Mapped References** section below.

Important: Use these options with care as they can both add and remove data from STEP.

- **Classification Product Links (All Types)** - *Recommended practice is to leave this box unchecked. If you do use it, note that there is no easy way to undo the changes.* This is a legacy option and checking this box affects all 'Product to Classification Link Types' (explained in the **Product to Classification Link Types** topic), not just those selected in the Classification Product Link Type option below. This checkbox has no effect on product-to-classification references (explained in the **Reference Types** topic). The option exists because, at one time, Types were not available for 'Product to Classification Links', and product-to-classification references were not allowed.

  For example, in the image below, using any import format (including a Smartsheet), a single Classification Product Link Type of 'Supplier Link' has been selected.



- If the 'Classification Product Links (All Types)' checkbox is not checked (as shown above), only the 'Supplier Link' type is affected by the removal of unmapped references.
- However, if the 'Classification Product Links (All Types)' checkbox is checked, all Product to Classification Link Types defined in System Setup are affected by the removal of unmapped references.

Important: The recommended option for handling Product to Classification Link Types is to use the 'Classification Product Link Type' option defined below. For Classification Reference Types, use the 'Classification Reference Type' option defined below.

- **Product Reference Type** - This option enables you to clean up product-to-product references, classification-to-product references, or entity-to-product references.

  For example, if you have products in your system that reference other products using the reference type 'Replacement Product.' Over time, products that really are replacement products have been linked to other products using other reference types such as 'Obsolete Product' or 'Similar Product.' To clean up, you list the products and their valid reference types in your load file, map them, and then select the reference type(s) that you want to remove from these products.

  The system then reads your load file and finds the source product, which is mapped to the ID and will be linked to one or more target products. All references are then removed from that product to all other products that have the reference type(s) selected in this import wizard screen. The source product is linked in the load file to the target product(s) using the reference type that was selected when you specified the target product.

  For example, Product 123 is linked to products 456, 567, 678, and 789 with the reference type of 'Obsolete.' All those links are incorrect. Therefore, you specify in your load file that 123 should link to 567 with the reference type of 'Replacement Product', and nothing else. There are no other entries for product 123 are in

the load file. In this screen, you select the reference type 'Obsolete.' After this load file has been processed, there are no links from product 123 to any product with the reference type 'Obsolete', and there is a link from 123 to 567 with the reference type of 'Replacement Product.'

> **Note:** You can only unlink the specified reference types. To unlink **all** references from a product, regardless of the type, you must select each reference type.

To make two references between two products (and all the other product ID pairs listed in the input file), you must load the file twice, specifying one reference type the first time, and the other reference type the second time. For the second import, do not use the Remove Un-Mapped References option.

- **Asset Reference Type** - This option refers to product-to-asset references, classification-to-asset references, or entity-to-asset references. It enables you to clean up references, for example, by removing some or all references between products and assets and applying only those listed in the load file.

  For example, in STEP you have products that are linked to images using the reference type 'Primary Image.' However, images that are really manufacturer logos have been linked to various products using this reference type, instead of using the 'Vendor Logo' reference type as intended. You therefore list the products and images in the load file together with the correct reference type, map the columns, and then select the reference type to be removed ('Primary Image') from the product / image. The system removes the selected reference type(s) and only the reference type(s) listed in the load file is available after import.

- **Entity Reference Type** - This option refers to any reference with a target of entity. It enables you to clean up references, for example, by removing some or all references between classifications, products, assets, entities, or publications and entities, and applying only those listed in the load file.

  For example, in STEP you have entities that are linked to entities using the reference type 'Address 1.' However, for identification purposes, they should have been linked using the 'Primary Address' reference type. You therefore list the entities and the address data (also an entity object in STEP) in the load file together with the correct reference type, map the columns, and then select the reference type to be removed ('Address 1') from the entity. The system removes the selected reference type and only the reference type listed in the load file is available after import.

- **Classification Reference Type** - When you have mapped a Classification Reference Type, this option allows you to specify the type(s) that should be removed during the import. This reference type is used when an asset / product / classification is the source, and a classification is the target.

- **Classification Product Link Type** - When you have mapped a Product to Classification Link Type, this option allows you to specify the type(s) that should be removed during the import. This is used to link a classification as the source to a product as the target, and display the product as a child of the classification. For more information on classification product link types, refer to **Product to Classification Link Types** in the **System Setup** documentation.

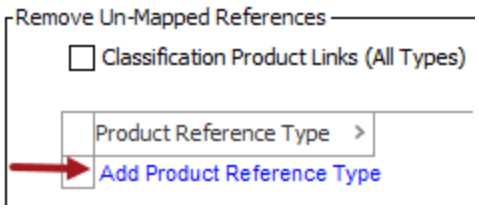
  For example, if you have products that are linked multiple times to different classifications, you can use this option to clean up product to classification links not specified by the import file. You therefore define certain products that are linked only to the classifications you list in your load file and not to any other classification.

  Do that by listing the products and classifications in the load file, mapping them, and then selecting the types that should be removed.

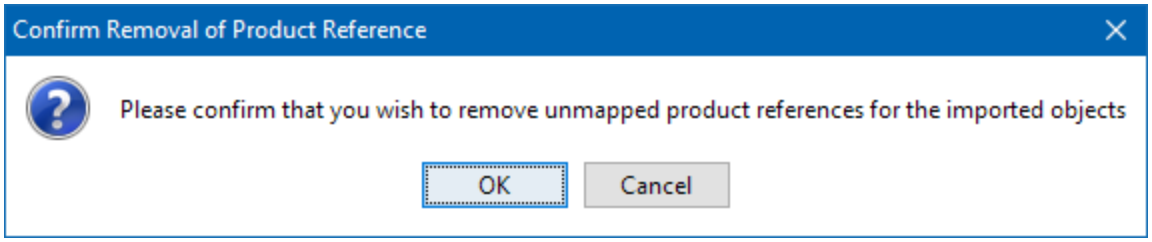
Adding and Removing Un-Mapped References

Interacting with the various **Remove Un-Mapped References** options is the same. An example of how to add and remove a reference type is explained below using the 'Product Reference Type.'

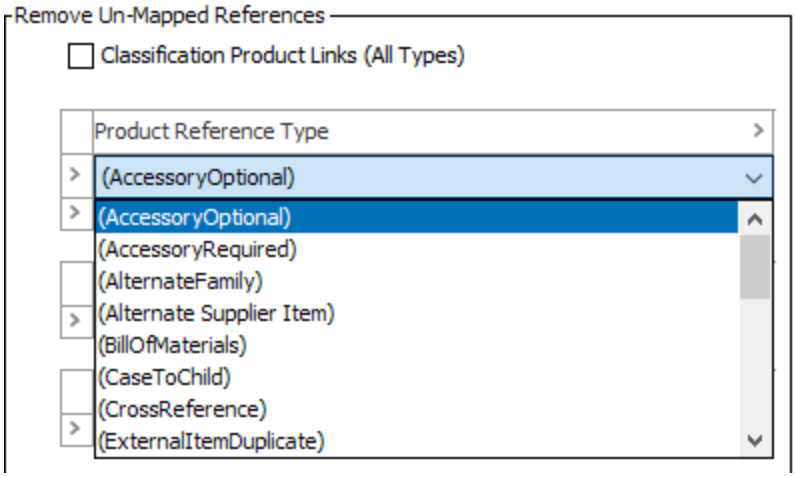
1. Click the option 'Add Product Reference Type.'



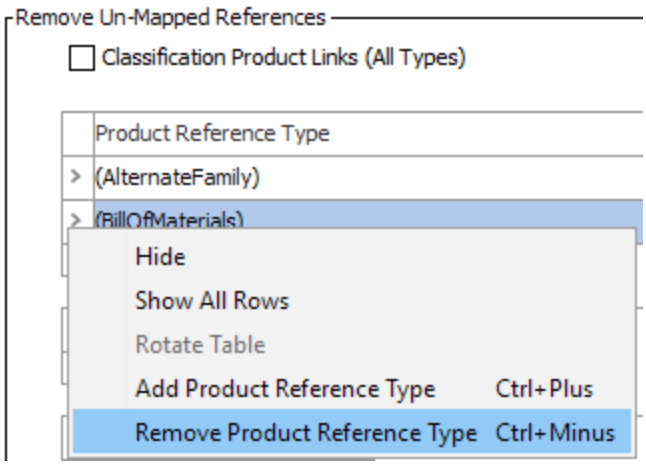
2. A dialog allows the user to confirm that they want to remove all unmapped Product references for the imported object.



3. Click OK, and then select the appropriate reference type from the dropdown list.



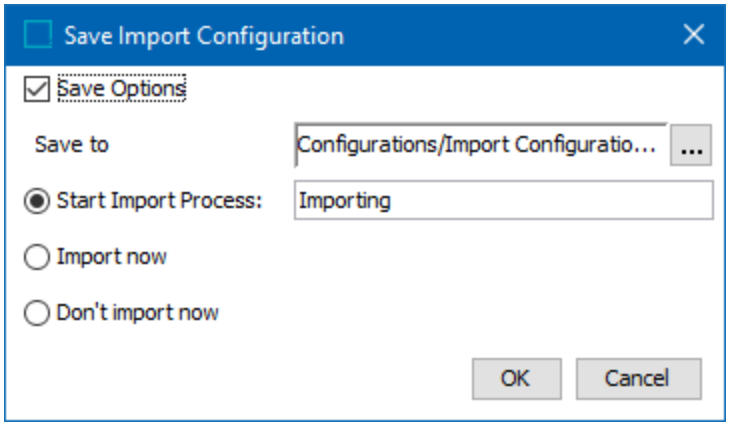
4. Right-click on the field of the Reference type will give the following options:
   - **Hide** - hides the selected reference type from the dropdown list.
   - **Show all Rows** - shows all the selected reference types.
   - **Add Reference Type** - adds an additional reference.
   - **Remove Reference Type** - removes the selected reference type.


Remove Un-Mapped References
Classification Product Links (All Types)
Product Reference Type
(AlternateFamily)
(BillOfMaterials)
Hide
Show All Rows
Rotate Table
Add Product Reference Type Ctrl+Plus
Remove Product Reference Type Ctrl+Minus

Running a Data Import

After clicking the Finish button on the Import Manager wizard in workbench, the **Save Import Configuration** window displays and allows you to save the configuration and then start the import.

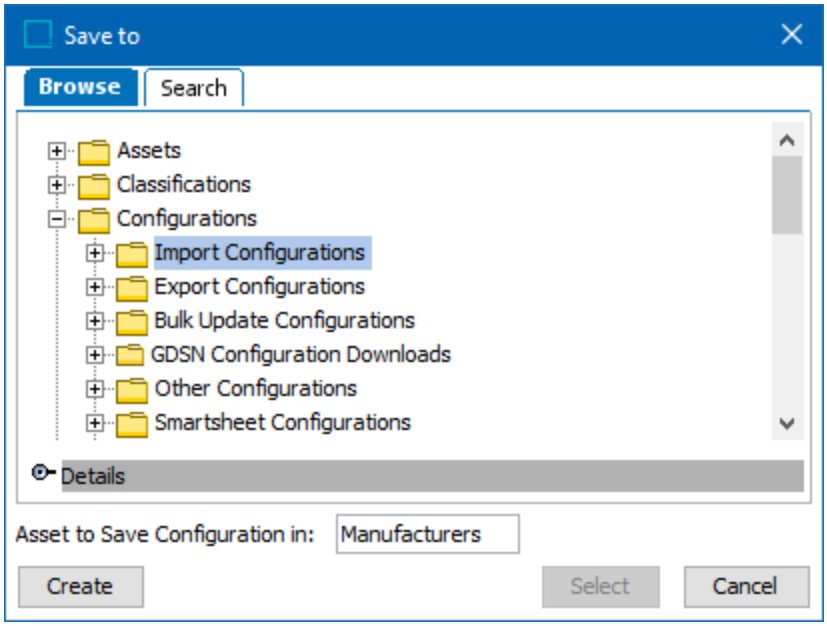
Saving a configuration allows you to modify it (instead of recreating it) during testing. Initially, use a small group of objects to test your settings. Refine the wizard parameters to resolve any issues and save the updated configuration.



1. To save the configuration so that it can be used for other imports, check **Save Options**, click the ellipsis button (...), and then use the **Browse** or **Search** tab to select a location. Configurations are saved as an special type of asset and live in the classification hierarchy.

   When updating an existing configuration, the path and name of the configuration that you selected on the first wizard screen is automatically displayed in the **Save To** box.

2. In the **Asset to Save Configuration in** field, enter a name for the configuration, and then click **Create**.



Note: Saving the import configuration is optional, you can also start the data import process without saving the configuration

3. Optionally, you can add a name for the import process in the **Start Import Process** text field. This makes finding the process on the BG Processes tab easier.
4. Determine the final action:
   - Select the **Start Import Process** radio button and click **OK** to send the job to a background process.
   - Select the **Import Now** radio button, which is available if the import file is less than 100 KB on a default system. This allows you to import the file without the use of a workflow. Click **OK** to send the job to a background process.
   - Select **Don't Import Now** to only save the settings, assuming you selected the Save Options checkbox. Click **OK** to close without importing.

   Note: No data is imported into STEP if, on the **Identify Destination** step, you checked the **Test Only Import** checkbox. Details of the test import are available on the **Import Report** dialog when selecting the Import Now option; details are available in the BG Process when selecting the Start Import Process option.
5. Determine the result of the background process as defined in the **Monitoring a Data Import** topic.

Maintaining a Saved Import Configuration

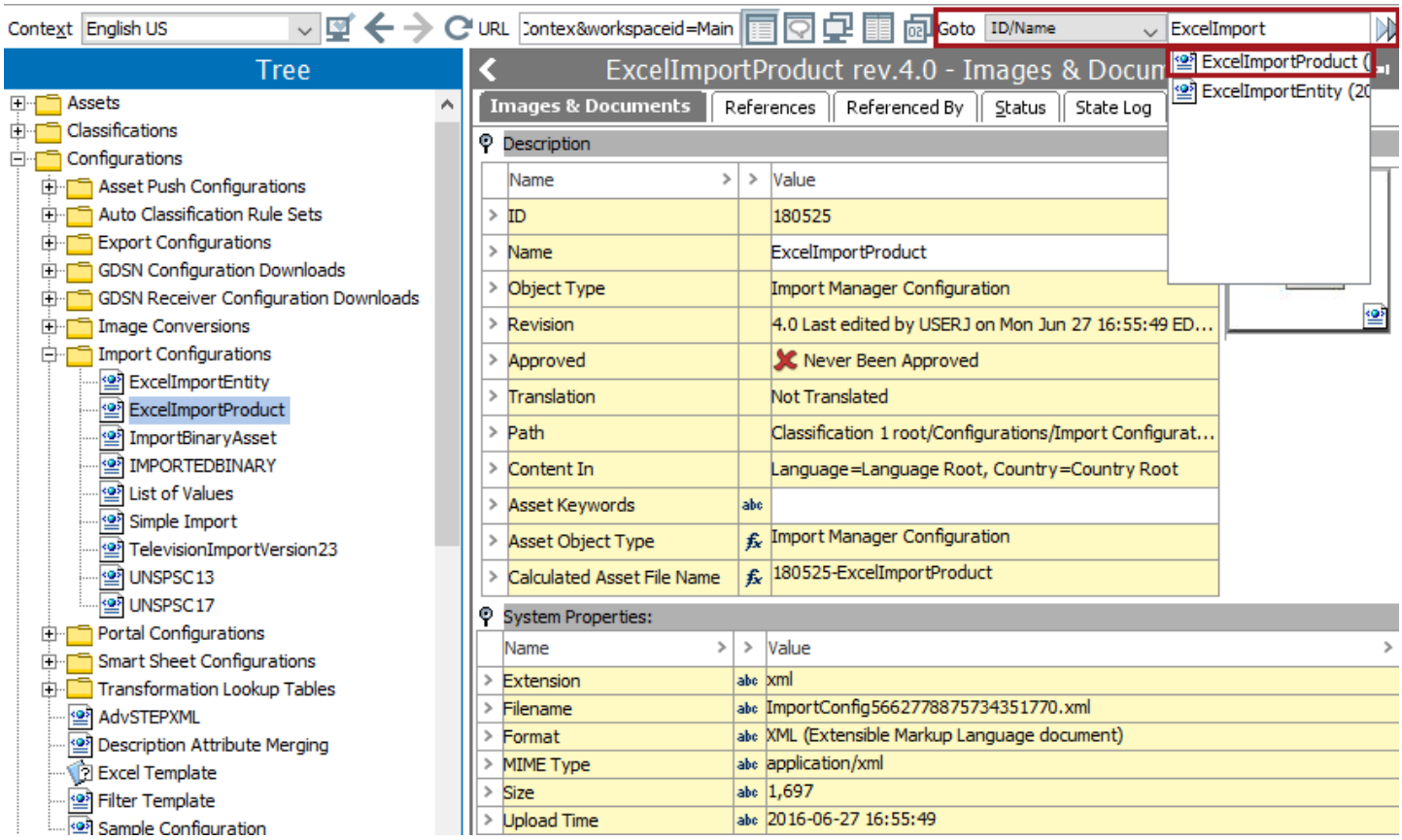
Completing the Import Manager wizard in workbench includes an option to save an import configuration (refer to **Running a Data Import**). Viewing a saved configuration allows you to confirm the current settings. Editing a saved configuration affects future imports and can be a valuable step in the process of creating and testing an import to ensure the correct data is received in the proper manner.

An Import Configuration definition can be exported as comments and submitted to an external source control system for comparison purposes as described in **Configuration Management** documentation.

View a Saved Import Configuration

After clicking the Finish button on the Import Manager wizard, the option to save the configuration is displayed. Although all classification folders are available, using a specific folder for import configurations makes it easier to locate them for review.

Note: If you know the name of the configuration, use the **Goto** search option to find it anywhere in the Tree.

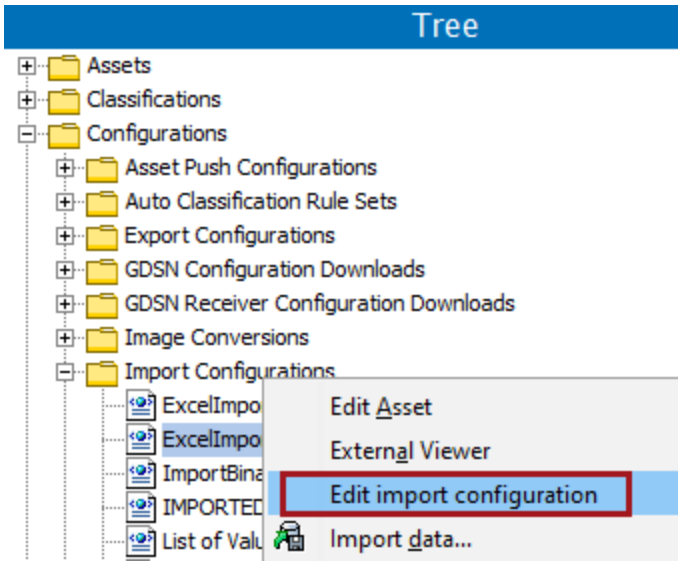


Edit a Saved Import Configuration

These editing options are available to update the selected configuration.

- To change only the name of the configuration, edit the Name field in the configuration editor.
- To edit an existing configuration and save with a new name, refer to the steps in the **Running a Data Import** section.
- To change the settings of the existing configuration parameters, follow the steps below.

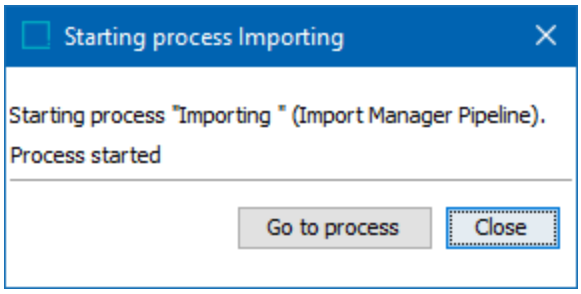
1. Select an import configuration, right-click and select **Edit import configuration** option.



2. The Import Manager wizard opens on Select Data Source.
3. Make all changes and click the **Finish** button to update the selected configuration.

Monitoring a Data Import

When you launch a data import process, STEP starts a background process. It assigns a unique process number and gives it the name assigned on the Save Import Configuration dialog. The starting process dialog is displayed:



Click the **Go To Process** button to display the current background process on the BG Processes tab.

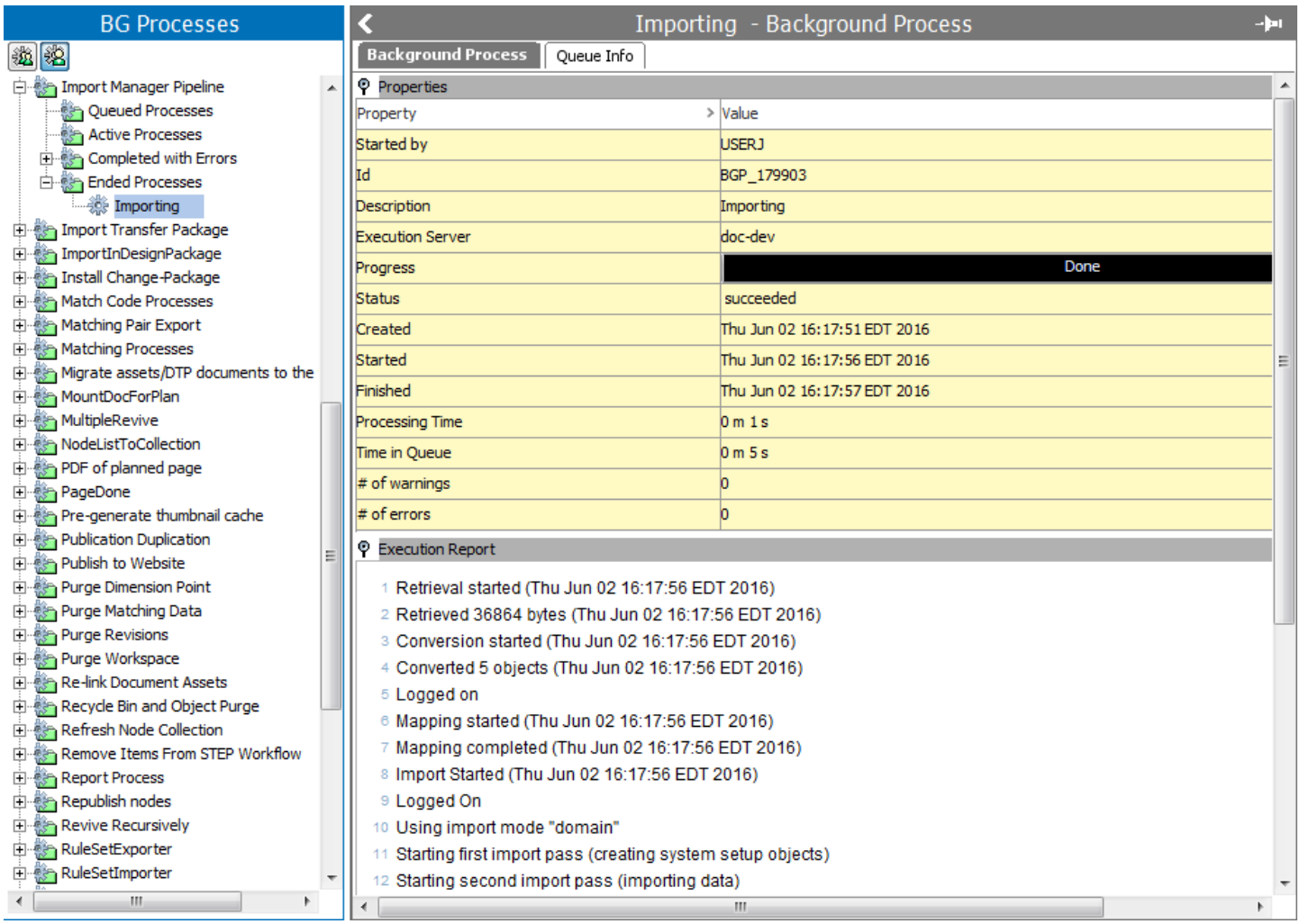
Monitoring the Import Process

On the BG Process tab, select the **Import Manager Pipeline** node. Select the process to monitor based on its status:

- For processes not yet started, open the **Queued Processes** folder.
- For processes that are currently being performed, look in **Active Processes** folder.
- For processes that have completed but had errors, look in **Completed with Errors** folder.
- For processes that have completed, look in **Ended Processes** folder.

The Properties flipper includes details about the import process.

For more information on background processes, refer to **Background Processes and Queues** in the **System Setup** documentation.



Properties Flipper

The following data is displayed in for the selected background process:

1. **Started by** displays the ID of the user who started the import.
2. **ID** is the Background Process ID which is generated sequentially and always starts with “BGP_” for each process that is started by a user.
3. **Description** is the name that is given by the user for the import process.
4. **Execution Server** is, by default, the application server where STEP is installed. However, it could also be a cluster server.
5. **Progress** displays status bar with the completion percentage of the process.
6. **Status** shows the status of the process, as defined below:
   - Completed means that the process completed successfully.
   - Completed with Errors indicates that the process was partially completed but with the errors included in the error file.

- Failed means that the process there was a fatal error and nothing was processed.
- Aborted means that the process was aborted by a user while the process was executing.

7. **Created** is the date and time when the process was created and moved to a particular background process queue.
8. **Started** is the date and time when the process started.
9. **Finished** is the date and time when the process ended.
10. **Processing Time** displays the number of hours, minutes, and seconds it took to complete the process.
11. **Time in Queue** displays the total amount of time that the process was in the queue before it started executing.
12. **# of Warnings** shows the number of warnings that were present after the process was executed. Corrections may be desired.
13. **# of Errors** shows the number of errors that were present after the process was executed. Corrections are required to complete the import successfully.

Execution Report

The execution report gives information about the data import, including validation errors found during import.

Typically, an execution report can be very long, and it can be useful to copy and paste the information into a standard text editor. You can then go through the file to locate errors and warnings.

Handling Import Errors

Some imports result in errors, for example, importing a new object (product) that is linked to another new object (classification or asset) that is also being imported. The background process fails, but the user can use the execution report to identify the error and resolve it by referring to the list of common errors.

For more information, refer to the **Import Error Messages** section and the **Import Error Message Examples** section.

When the following errors are reported, the creation of new objects is skipped entirely:

- Specified parent object does not exist
- Specified Object Type is not valid below specified parent
- No ID is provided and no ID pattern is specified for the target Object Type

When an import process completes with errors, a file is created that contains just the rows with erroneous data. The file can be saved to disk or opened directly in the Import Manager.

When a data error is encountered, the error message reports the data it was attempting to load, including the attribute name and the object ID for the value, and the type of error encountered. The line number and column number are also given, but that is not always useful since it refers to the interim XML file that is created.

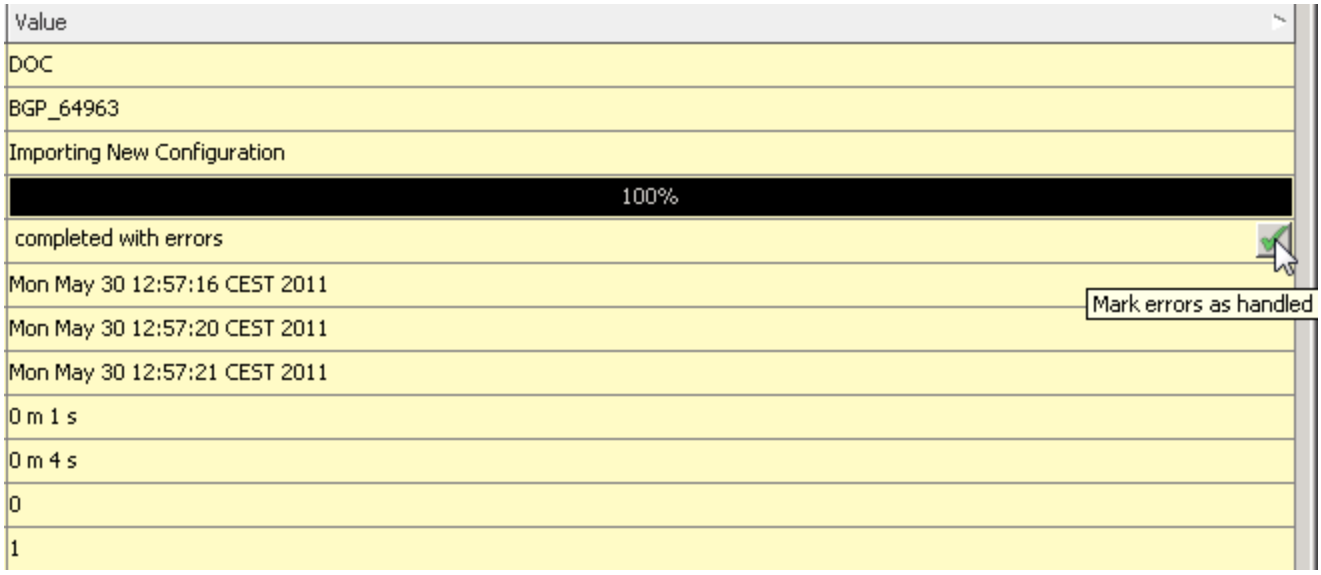
For Excel and CSV imports, the error file contains the rows from the original file with the defect data.

For XML files, the error file contains the objects with errors.

Reimporting a Corrected File

Use the error file to identify the problematic data in the import file and determine the next step:

- If you can correct the data errors with transformations, re-import the same file and create the necessary transformation. When you re-import a file, the status of the background process changes from 'completed with errors' to 'succeeded' and the process is moved to the **Ended Processes** group.
- If transformations will not resolve the issues, edit the import file offline, save it and then import it. You can make the state transition by clicking the **Mark errors as handled** button as shown below.



Note: In some cases, users may prefer a more complete list of errors than is allowed by default in the execution report. If the default maximum number of displayed errors (1,001) must be adjusted, the `Import.TotalErrorReportLimit` property can be added to the sharedconfig.properties file and set to the desired maximum. It should be noted that the full list of errors will be available via an export of the error file; the maximum in this case refers only to the number of errors that display in the execution report. It should be noted that configuring a high maximum number of displayed errors could result in performance issues.

Import Error Messages

The following table shows most of the types of error message that you are likely to encounter, with a brief explanation of the most common reason for the error. The most common errors allow users to troubleshoot and then perform the import. There are instances where issues are not related to the import file and are related to the servers, network latency, etc. For these scenarios, contact the Stibo Systems Support team for assistance.

The error messages marked with an asterisk (*) are errors that you will only encounter when you load an XML file. That is because in an XML import, the wizard does not prevent you from entering an invalid value. However, when importing CSV or XLS files, since you often select values from a list, you cannot select an invalid value.

Error Message	Description
Illegal value x for attribute y in product z	The value in the load file does not meet the validity or other constraint that is set up for the attribute.
Not in legal values list	You tried to load an attribute value into an attribute that uses an LOV, and that value does not exist in the LOV, and it is set up not to accept additional entries. This error is not displayed when an LOV is configured to 'Allow Users to Add Values = Yes.'
Attribute not legal for object type	The product's object type is not valid for the attribute. That is, the attribute is not allowed to hold a value for products with the specified object type.
*Illegal unit x for attribute y in product z	In the XML file, you have tried to assign a non-existing unit to an attribute.
Unknown Classification x in product y.	In the XML file, you have probably requested a product-to-classification link using a specified reference type, but the classification you specified is not found.
Unknown Parent product x for product y	You tried to assign a product to a parent ID, but that parent ID cannot be found in the system.
*Unknown product reference target	You attempted to link one product to another via a reference type, but the target product does not exist.
*Unknown attribute	You specified an attribute in the XML file, but that attribute does not exist.

Error Message	Description
Unknown unit	The unit encountered in the load file does not exist in STEP.
Rejected new product x	You set the option to "Reject New Products" in the wizard, and a new product was encountered in the load file.
Illegal unit for attribute	The unit exists but is not valid for the attribute you are trying to load the value into.
*Illegal Object Type for product position	You tried to create a new product via an XML load and you specified an object type that is not a valid choice, given the parent product that you specified.
*Unknown Object Type	You specified an object type (e.g., for a product) in the XML file, but that object type does not exist.
*Unknown reference Type	You specified a reference type (e.g., for a product-to-product reference) in the XML file, but that reference type does not exist.
Insufficient privileges to ...	You do not have permission to carry out one of the following actions (specified in the error message): • set value • set name • create product • move product • modify Object Type • classify product • create product reference For more information, refer to the **Action Sets** topic in the **System Setup** documentation.
Product x contains more than one reference to product y of Type z, reference data will not be imported.	You cannot make a product-to-product reference from the same source product to the same target product more than once.

Error Message	Description
Product x contains more than one reference to asset y of Type z, reference data will not be imported.	You cannot make a product-to-asset reference from the same source product to the same target asset more than once.
*Classification x contains more than one reference to Classification y of Type z, reference data will not be imported.	You cannot make a classification-to-classification reference from the same source classification to the same target classification more than once.
*Reference Type x is not valid for importing a product cross reference from a to b.	In your XML file, you have asked for a product reference that refers to a non-existent reference type.
*Reference Type x is not valid for importing an asset cross reference from a to b.	In your XML file, you have asked for an asset reference that refers to a non-existent reference type.
*Reference Type x is not valid for importing a Classification cross reference from a to b.	In your XML file, you have asked for a classification reference that refers to a non-existent reference type.
*Product x already contains a reference to Classification y but it is invisible in the specified context. The reference and any data on it will not be imported.	You have probably made a product-to-classification link dimension-dependent, and the context you are importing into cannot "view" that link, but it is still there.
Conversion Error	This run time exception error is returned when the XML file is loaded is poorly formed. Download the error file, make the indicated correction, and import the file again.
Conversion Error: Not a Valid STEPXML File (Encoding)	When any encoding type other than UTF-8 is entered in a STEPXML import file, this error is displayed in wizard on the 'Select Format.'
Unknown validator	Importing STEPXML that includes an attribute with a validator type that does not exist in STEP. Validation types with multiple words must replace the space with an underscore. For example, 'Numeric Text' must be written as 'Numeric_Text.'
Context ‘XYZ’ not found	Importing STEPXML where the Context ID is either misspelled or does not exist.

Error Message	Description
No ContextID specified in STEPXML to import	Importing STEPXML without a Context ID.
No WorkspaceID specified in STEPXML to import	Importing STEPXML without a Workspace ID. Workspace ID should always be an editable workspace, often 'Main.' Since the Approved workspace cannot be edited, it cannot receive imports.

Import Error Message Examples

The most common errors encountered during data import refer to an invalid attribute value. Sometimes there is a mismatch with the attribute's validation type, other times there is a mismatch with one of the other setups. For example, the attribute's constraints, such as masks, minimum and maximum values, maximum length, LOVs, the object's own object type, and so on. Therefore, when you encounter an error, look at the attribute setup and review the constraints.

Even if you load attribute values for a object, not all values will be automatically available in STEP. Attributes must be made valid for an object before the values can be accessed via references. STEP will always load attribute values if the object's type is valid for the attribute, and the attribute values meet the validity criteria. But, the attribute itself must be a valid attribute for that object, that is, linked somewhere in the object hierarchy or classification hierarchy where the object resides.

Use the following examples to assist in troubleshooting problems with data imports.

Illegal Unit for Attribute

The tab-delimited input file below had two columns mapped: the first column was mapped both to the Product ID and Product Name, and the second column was mapped to the attribute 'Description.'

```
T100-3526       RED 1/2" STRAIGHT WIDGET SSU05PL
T100-2625      BLUE 1/4" ANGLED GADGET BZT10GL
T100-2827      ORANGE 3/4" ELL WHATSIT PLU04HF
```

When the file was imported, the execution report showed three illegal unit errors:

- Illegal unit "/2" STRAIGHT WIDGET SSU05PL" for attribute Description in product T100-3526 (UnknownUnit). Source: Line=2, Column=Column2.
- Illegal unit "/4" ANGLED GADGET BZT10GL" for attribute Description in product T100-2625 (Unknown Unit). Source: Line=3, Column=Column2.
- Illegal unit "/4" ELL WHATSIT PLU04HF" for attribute Description in product T100-2827 (Unknown Unit). Source: Line=4, Column=Column2.

Upon inspection of the attribute with the name of 'Description', it was found that the assigned data validity check was 'Numeric_Text', and that it had a unit assigned to it: mm. Further, it was not selected as the default unit.

STEP could not handle the input text, since the attribute's validity was defined as being NumericText and had only one legal unit, and that unit could not be matched with anything in the attribute value provided by the import. In this case, either the attribute's setup is incorrect, or a different product attribute should be used that has an appropriate data validity type assigned.

Unknown Parent Product

The tab-delimited input file includes three columns. Column 1 was mapped to both Product ID and Product Name (new products), and Column 3 was mapped to the Parent ID.:

```
T100-3526       RED 1                    Hand Tools
T100-2625      1mm ANGLED GADGET        Hand Tools
T100-2827      ORANGE 3mm               Hand Tools
```

When the file was imported, the execution report showed the following three errors:

- Unknown Parent Product "Hand Tools" for product "T100-3526" (1 product(s) was rejected due to this error). Source: Line=2, Column=Column3.
- Unknown Parent Product "Hand Tools" for product "T100-2625" (1 product(s) was rejected due to this error). Source: Line=3, Column=Column3.
- Unknown Parent Product "Hand Tools" for product "T100-2827" (1 product(s) was rejected due to this error). Source: Line=4, Column=Column3.

STEP could not find the Parent IDs given in Column 3 of the input file, and therefore the new products were rejected. Even though a default Parent ID was selected in the Identify Destination screen of the import wizard, STEP accepts the value for the Parent ID in the input file as an override, and rejects the new product anyway.

A reason for this execution report error could be that one of the input file's columns was mapped to be the Parent ID, and STEP could not find it. If the product existed already, it would not be moved anywhere, it would stay with its current parent. If the product did not exist, it would not be created.

Maximum Length, Illegal Value

The input file had two columns mapped: the first column was mapped both to the Product ID and Product Name (these were new products), and the second column was mapped to the attribute "Manufacturer Part Number."

```
G100-352654     REDD-52626278-YDS-777777-1/FFGH00KK7733249-UHP
J100-267225    REDD-5262627
D100-289827    K7733249-UHP
```

The file was imported and the execution report showed the following three errors:

- Illegal value "REDD-52626278-YDS-777777-1/FFGH00KK7733249-UHP" for attribute Manufacturer Part Number in product G100-352654 (Length of the value exceeds max length of domain Is trying to insert a value with '46' characters into a domain that has a maximum length of '40' characters.). Source: Line=10, Column=Column2.
- Line= at line 9: : Length of the value exceeds max length of domain Is trying to insert a value with '46' characters into a domain that has a maximum length of '40' characters.
- Line= at line 9: : Array operation failed (1 times): Error during array operation: ORA-20291: Parent key not found (valuemap.edgeid) ORA-06512: at "STEPSYS.VALUEMAP_BEF_INS", line 18 ORA-04088: error during execution of trigger 'STEPSYS.VALUEMAP_BEF_INS' ; SQL return code=20291; SQL return code=0

These errors are reporting a single problem. The first and second errors say that one of the imported part numbers had a value in the Manufacturer Part Number column that was too long, as defined by the attribute's setup. You are given the part number, the attribute name, and the value that was trying to be imported. The third error message is useful for a programmer.

Note: This type of error does not prevent the new part number from being created. The value for the Manufacturer Part Number, however, is left blank.

Out of Range, Illegal Value

This is another example of an illegal value error:

- Illegal value "250" for attribute Focal Length (ft) in product G100-352654 (Value error: Value '250' ' is out of range). Source: Line=9, Column=Column2.
- Line= at line 8: : Value error: Value '250' ' is out of range
- Line= at line 8: : Array operation failed (1 times): Error during array operation: ORA-20291: Parent key not found (valuemap.edgeid) ORA-06512: at "STEPSYS.VALUEMAP_BEF_INS", line 18 ORA-04088: error during execution of trigger 'STEPSYS.VALUEMAP_BEF_INS' ; SQL return code=20291; SQL return code=0

This group of three error messages are reporting a single problem. The first two messages indicate that a value of 250 is too large for the maximum value allowed for the attribute.

Not a Number, Illegal Value

The following tab-delimited input file had two columns mapped: the first column was mapped to the Product ID (the products already existed in STEP), and the second column was mapped to an attribute called 'Hole Diameter.'

```
G100-352654     1-1/2
J100-267225     2-3/4
D100-289827    17-3/16
```

The file was imported and the execution report showed the following nine errors (three sets):

Set 1

- Illegal value "1-1/2" for attribute Hole Diameter in product G100-352654 (Value error: Value '1-1/2' is not a number). Source: Line=9, Column=Column2.
- Line= at line 8: : Value error: Value '1-1/2' is not a number
- Line= at line 8: : Array operation failed (1 times): Error during array operation: ORA-20291: Parent key not found (valuemap.edgeid) ORA-06512: at "STEPSYS.VALUEMAP_BEF_INS", line 18 ORA-04088: error during execution of trigger 'STEPSYS.VALUEMAP_BEF_INS' ; SQL return code=20291; SQL return code=0

Set 2

- Illegal value "2-3/4" for attribute Hole Diameter in product J100-267225 (Value error: Value '2-3/4' is not a number). Source: Line=14, Column=Column2.
- Line= at line 13: : Value error: Value '2-3/4' is not a number
- Line= at line 13: : Array operation failed (1 times): Error during array operation: ORA-20291: Parent key not found (valuemap.edgeid) ORA-06512: at "STEPSYS.VALUEMAP_BEF_INS", line 18 ORA-04088: error

during execution of trigger ‘STEPSYS.VALUEMAP_BEF_INS’ ; SQL return code=20291; SQL return code=0

Set 3

- Illegal value "17-3/16" for attribute Hole Diameter in product D100-289827 (Value error: Value ‘17-3/16’ is not a number). Source: Line=19, Column=Column2.
- Line= at line 18: : Value error: Value ‘17-3/16’ is not a number
- Line= at line 18: : Array operation failed (1 times): Error during array operation: ORA-20291: Parent key not found (valuemap.edgeid) ORA-06512: at "STEPSYS.VALUEMAP_BEF_INS", line 18 ORA-04088: error during execution of trigger ‘STEPSYS.VALUEMAP_BEF_INS’ ; SQL return code=20291; SQL return code=0

Each set of three errors addresses one of three problems. That is, different information is reported for the same error in three individual messages. The first message in each group is what is most useful to the end user. In the above series, notice that the first lines that indicate that the fractional value in the input file is not allowed for the attribute 'Hole Diameter' because the value 'is not a number.' In this case, the attribute’s validation type was set to Number, which does not allow fractional values.

This is another example:

- Illegal value "Hand tools" for attribute Focal Length (ft) in product J100-267225 (Value error: Value ‘Hand tools’ is not a number). Source: Line=14, Column=Column2.
- Line= at line 13: : Value error: Value ‘Hand tools’ is not a number
- Line= at line 13: : Array operation failed (1 times): Error during array operation: ORA-20291: Parent key not found (valuemap.edgeid) ORA-06512: at "STEPSYS.VALUEMAP_BEF_INS", line 18 ORA-04088: error during execution of trigger ‘STEPSYS.VALUEMAP_BEF_INS’ ; SQL return code=20291; SQL return code=0

The first message in the group tells you that the attribute value of 'Hand tools' in the input file is not allowed since the attribute is set up to expect a number.

And another example:

- Illegal value "50,70" for attribute Focal Length (ft) in product D100-289827 (Value error: Value ‘50,70’ is not a number). Source: Line=19, Column=Column2.
- Line= at line 18: : Value error: Value ‘50,70’ is not a number
- Line= at line 18: : Array operation failed (1 times): Error during array operation: ORA-20291: Parent key not found (valuemap.edgeid) ORA-06512: at "STEPSYS.VALUEMAP_BEF_INS", line 18 ORA-04088: error during execution of trigger ‘STEPSYS.VALUEMAP_BEF_INS’ ; SQL return code=20291; SQL return code=0

The first message in the group tells you that the attribute value of '50,70' in the input file does not match the attribute’s validation type of Number, which does not allow commas.

Object Existed with Another Object Type

The following tab-delimited input file had two columns mapped: the first column was mapped to the Product ID (the products already existed in STEP), and the second column was mapped to an attribute called "MFOB."

```
G100-352654    Yes
J100-267225    No
D100-289827    Maybe
```

The file was imported and the execution report showed the following three warnings and one error:

- Line=2: Product ‘G100-352654’ existed with another object type - supplied object type not set
- Line=3: Product ‘J100-267225’ existed with another object type - supplied object type not set
- Line=4: Product ‘D100-289827’ existed with another object type - supplied object type not set
- Illegal value "Maybe" for attribute MFOB in product D100-289827 (Not in legal values list). Source: Line=4, Column=Column2.

The first three warnings say that the products already existed in STEP with the object type of 'Product.' When the file was imported, the Identify Destination screen was skipped, and the default object type that was set in that screen was different from Product. STEP reported that it did not change the object type of the products.

The final message is an error that says that the attribute MFOB used an LOV and the value in the input file was not valid. Further investigation showed that LOV had only two valid values: Yes and No. Additionally, the LOV was set up so that no modifications could be made to the LOV by loading in data. So the value "Maybe" was rejected.

Unknown Classification

The following tab-delimited input file had two columns mapped: the first column was mapped to the product ID (the products already existed in STEP), and the second was mapped as a classification ID.

```
G100-352654    Hand tools
J100-267225    Hand tools
D100-289827    Hand tools
```

The file was imported and the execution report showed the following three errors:

- Unknown Classification "Hand tools" in product "G100-352654" (Classification ‘Hand tools’ does not exist in the import workspace). Source: Line=6, Column=Column2.
- Unknown Classification "Hand tools" in product "J100-267225" (Classification ‘Hand tools’ does not exist in the import workspace). Source: Line=10, Column=Column2.
- Unknown Classification "Hand tools" in product "D100-289827" (Classification ‘Hand tools’ does not exist in the import workspace). Source: Line=14, Column=Column2.

In this case, the import file requested that the products be linked to a classification folder, but STEP could not find a classification with the specified ID 'Hand tools.'

Not Privileged to Create Object

A file was imported and the execution report showed the following error:

- Line=2: Not privileged to Create product

Two scenarios can cause this error message:

1. Importing new products when the import user does not have the privilege to create new products. The import user must have all access required for the action to be performed during the import.
2. Importing a product with a product-to-product Reference Type by mapping one column of data to the Product ID and another column to a Reference, while the target product of the reference does not already exist in STEP. A reference can only be imported when the target already exists.

Unknown Asset ID

A file was imported and the execution report showed the following error:

- Line=2: Unknown Asset ID: AC200-627

This error is reported when attempting to link a Product ID to an Asset ID, but the asset does not exist in STEP.

Note: The Asset ID (not the Asset Name) is required in an import file.

Array Operation Failed

A file was imported and the execution report showed the following error:

- Line= at line 8: : Array operation failed (1 times): Error during array operation: ORA-20042: Value rejected: Attribute Gap not valid for this usertype ORA-06512: at "STEPSYS.VALUEMAP_BEF_INS", line 41 ORA-04088: error during execution of trigger 'STEPSYS.VALUEMAP_BEF_INS' ; SQL return code=20042; SQL return code=0

This error is reported when attempting to load an attribute value (not shown in the error) to an attribute that is not legal for the product's object type. Upon further investigation, we found that the product's object type was 'Product.' The error message is saying that the attribute that has the ID of 'Gap' is not valid for products with the object type of 'Product.' Since the value is invalid, it is ignored.

Optimistic Locking Errors

When a file was imported, the below error was displayed in the execution report:

- OptimisticVerificationException: Optimistic locking errors were detected when flushing to the data store. This indicates that some objects were concurrently modified in another transaction. Failed objects: [com.stibo.core.persistence.ProcessPO@33bf33bf: BGP_3604361] [java.util.ArrayList]

This error occurs when a user is trying to perform an import by modifying an attribute value for an existing product in STEP and at the same time there is another user who has access to the same product and is modifying the attribute value. In this scenario, the product is locked for any modification and the import fails.

The Transaction has been Rolled Back: Unique Constraint

A file was imported to create a new product and the below error was thrown:

- Caught FatalDataStoreException at Mon Jun 06 13:10:16 CEST 2016: The transaction has been rolled back. Refer to the nested exceptions for details on the errors that occurred., caused by SQLException: ORA-00001: unique constraint (STEPSYS.NODE_NODEID_UIX) violated {prepstmnt 523703057 INSERT INTO

NODE (CHECKID, NAME, NODEID, NODETYPE, USERTYPE, USERTYPEID) VALUES (?, ?, ?, ?, ?, ?) [reused=5]} [code=1, state=23000]

The above error was thrown when a new product was being created but, the ID provided to create the new product already existed in STEP. ID's for objects that are created in STEP is always Unique and if a value for an ID already exists then repeating the same again is not accepted in STEP.

Note: ID's used for one product cannot be used again for another product, however, the same ID can be used for to create a Classification folder.

Import errors are shown in the Background processes under the Execution report and the number of errors and warning are also displayed as shown below:

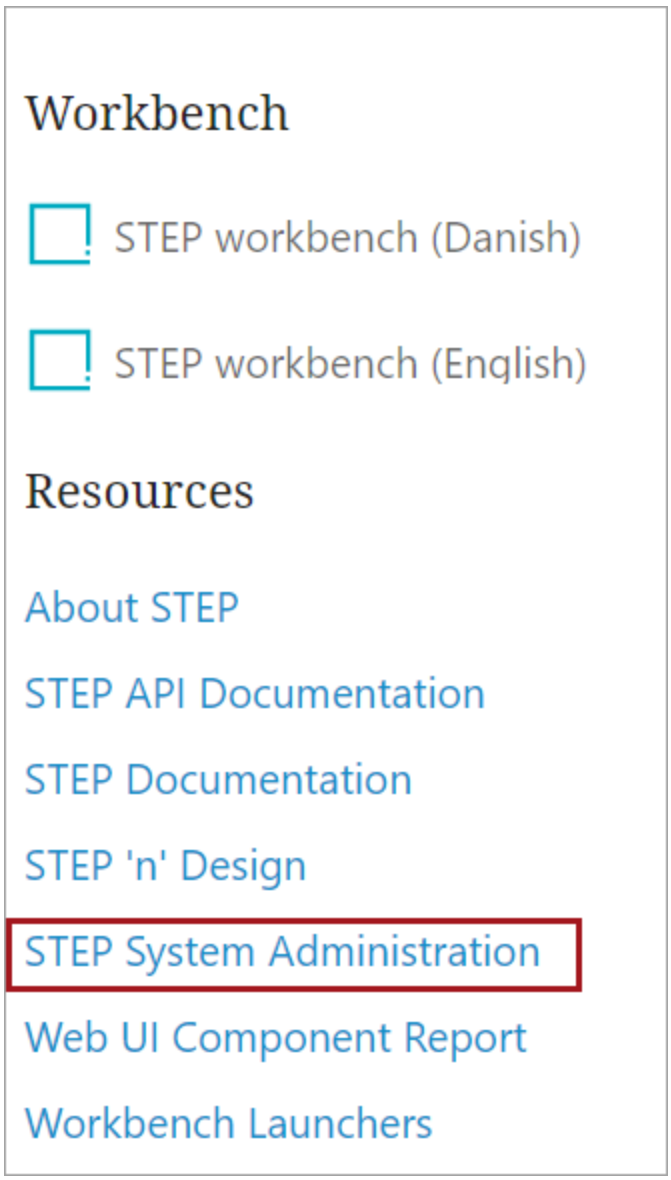
Description	Importing
Execution Server	doc-rel
Progress	100%
Status	completed with errors
Created	Sun Feb 05 06:49:19 EST 2017
Started	Sun Feb 05 06:49:20 EST 2017
Finished	NA
Processing Time	0 m 2 s
Time in Queue	0 m 1 s
# of warnings	0
# of errors	1

Execution Report

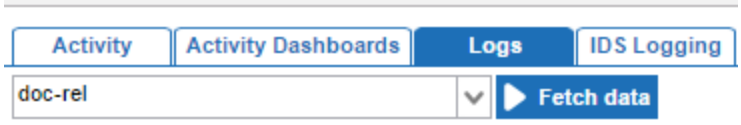
```
 1 Retrieval started (Sun Feb 05 06:49:21 EST 2017)
 2 Retrieved 8817 bytes (Sun Feb 05 06:49:21 EST 2017)
 3 Conversion started (Sun Feb 05 06:49:21 EST 2017)
 4 Converted 6 objects (Sun Feb 05 06:49:24 EST 2017)
 5 Logged on
 6 Mapping started (Sun Feb 05 06:49:24 EST 2017)
 7 Mapping completed (Sun Feb 05 06:49:24 EST 2017)
 8 Import Started (Sun Feb 05 06:49:24 EST 2017)
 9 Logged On
10 Using import mode "domain"
11 Starting first import pass (creating system setup objects)
12 Starting second import pass (importing data)
! 13 Row 6, Column : The attribute with ID 'ShortItemDescription' isn't valid for object product with ID '888264
14 Imported 2 new products, 0 new classifications, 0 new entities and 0 new assets.
15 Processed 4 existing products, 0 existing classifications, 0 existing entities and 0 existing assets.
16 Skipped 0 products, 0 classifications, 0 entities and 0 assets.
17 Deleted 0 products, 0 classifications, 0 entities and 0 assets.
18 Found 0 warnings
19 Found 1 errors
20 Import completed (Sun Feb 05 06:49:26 EST 2017)
21 Error file generation started (Sun Feb 05 06:49:26 EST 2017)
22 Error file with 1 object(s) generated (Sun Feb 05 06:49:30 EST 2017)
```

Similarly, the execution report can also be viewed in the System Administration page by following the below steps:

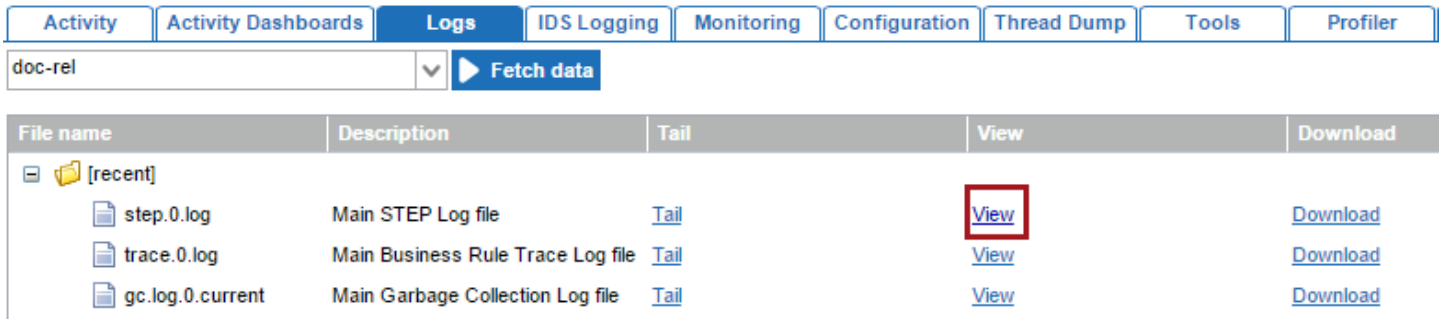
1. Launch the Start Page
2. Click on the System Administration link



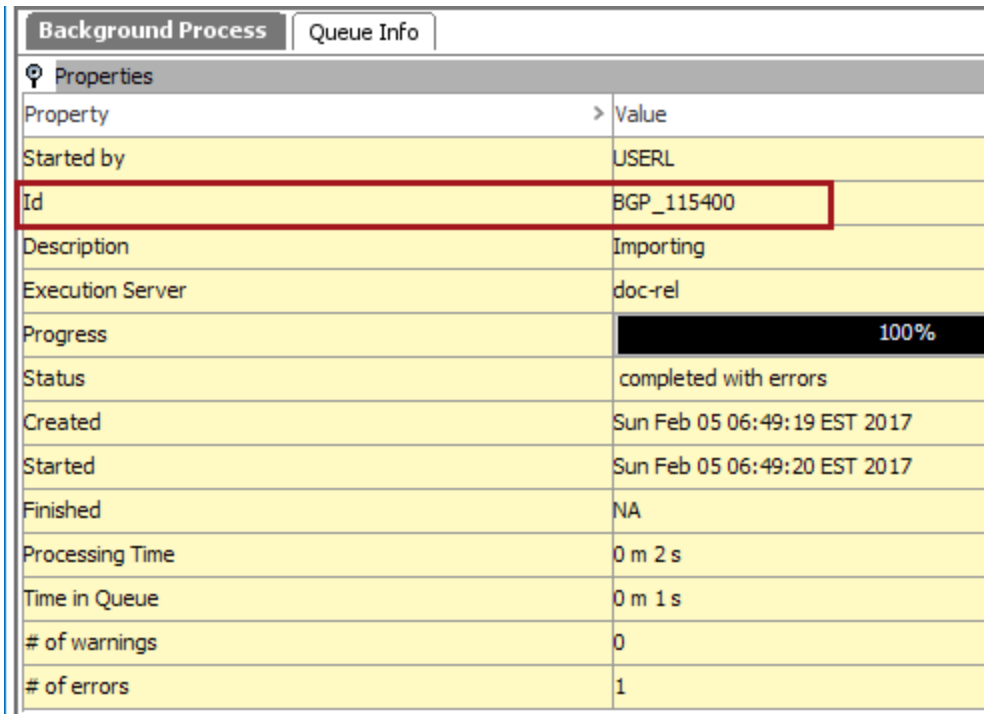
3. Once logged in, click on the Logs Tab
4. Select the Server in the dropdown list and click on 'Fetch Data'



5. Expand [recent] and then click on view option for step.0.log file



6. A new window will open with the log details shown
7. Copy the background process ID from STEP



8. Press Ctrl + F on the keyboard and search for the background process ID to find the complete log detail with more additional details.

```
Submitting bg-process (BGP_115400) with template (stibo.Importer) to queue (IN)
2017/02/05-06:49:20     2c8     com.stibo.servicemanager.beans.DefaultBackgroundProcessInvocationImpl run INFO: Start
handling of bg-process (BGP_115400) (recovery=false)
2017/02/05-06:49:21     2c8|USERL|BGP   com.stibo.util.unstable.plugin.DefaultPlugin init WARNING: Unable to find
parameter name: ValueSubstitutionAssetIDName
2017/02/05-06:49:21     2c8|USERL|BGP   com.stibo.util.unstable.plugin.DefaultPlugin init WARNING: Unable to find
parameter name: WordSubstitutionAssetIDName
2017/02/05-06:49:21     2c8|USERL|BGP   com.stibo.util.unstable.plugin.DefaultPlugin init WARNING: Unable to find
parameter name: LibraryAttributeTransformationIDName
2017/02/05-06:49:25     2c8|USERL|BGP   com.stibo.systemconfig.ConfigUtil getPropertyTypeCheck WARNING
The property Import.IsMarkNodeFilterValuesModification is not documented, check
com.stibo.core.domain.impl.importer.ValueFilterHandlerDomain.<clinit>(ValueFilterHandlerDomain.java:38) and read
http://confluence.stibo.com/display/RD/STEP+5+configuration+properties
2017/02/05-06:49:30     2c8|USERL|BGP   com.stibo.core.domain.impl.backgroundprocess.BackgroundProcessImpl updateStatus
INFO: Setting status on succeeded bg-process (BGP_115400)
2017/02/05-06:49:30     2c8|USERL|BGP   com.stibo.core.domain.impl.backgroundprocess.BackgroundProcessImpl$12
lambda$run$0 INFO: Succeeded updating status of bg-process (BGP_115400)
2017/02/05-06:49:30     2c8     com.stibo.servicemanager.beans.DefaultBackgroundProcessInvocationImpl run INFO: Finished
handling of bg-process (BGP_115400)
2017/02/05-07:00:16     4f      com.stibo.systemconfig.ConfigUtil getPropertyTypeCheck WARNING
The property Log.ConfigurationPackagesLogRoot is not documented, check
com.stibo.admin.FileUtil.getConfigurationPackagesLogFileNames(FileUtil.java:265) and read
http://confluence.stibo.com/display/RD/STEP+5+configuration+properties
2017/02/05-07:00:53     4e      com.stibo.systemconfig.ConfigUtil getPropertyTypeCheck WARNING
The property Log.ConfigurationPackagesLogRoot is not documented, check
```

As explained above, there are common errors and the execution logs can be viewed and then correction can be made accordingly and then the file can be imported. There are other errors which can occur for different reason based on the scenario. If the errors are not correctable then those report logs can be collected and then reported to the Stibo Systems Support team to have the issue fixed.

Inbound Integration Endpoints

Inbound integration endpoints (IIEPs) offer a centralized interface for monitoring and maintaining integrations with systems that send data to STEP in the form of files or messages.

IIEPs can poll different data sources for new files or messages to import on scheduled intervals–the minimum interval being one minute. As such, IIEPs offer functionality for 'near real-time' integrations. However, when the data source is REST, a POST will invoke the endpoint. For alternative real-time integrations with STEP, access the **STEP API Documentation**, available at [system]/sdk or from the Start Page, for information on SOAP and REST APIs.

IIEPs provide a way to automatically manage the functionality for the standard STEP import functionality. Without any extensions, files, or messages being imported, data is handled by the STEP Importer, and the same options for mapping and manipulation are available as when importing manually.

Setup Requirements

Setting up and using an inbound integration endpoint involves the following steps:

1. Create a setup group to hold the endpoint as described in **Initial Setup for an Inbound Integration Endpoint**.
2. Launch the inbound integrations endpoint wizard as described in **Creating an Inbound Integration Endpoint**.
3. In the wizard, add an ID, name, description, and user as described in **IIEP - Identify Endpoint**.
4. In the wizard, select the receiver of the data as described in **IIEP - Choose Receiver**.
5. In the wizard, specify processing, context, and recommended priority or legacy queue settings for the endpoint as described in **IIEP - Configure Endpoint**.
6. In the wizard, if available, configure defined pre-processors for customer-specific solutions as described in **IIEP - Configure Pre-processor**.
7. In the wizard, specify the data format and map the data, among other options as described in **IIEP - Configure Processing Engine**.
8. In the wizard, if available, configure defined post-processors for customer-specific solutions as described in **IIEP - Configure Post-processor**.
9. In the wizard, specify how often the endpoint should search for data to be processed as described in **IIEP - Schedule Endpoint**.
10. In the wizard, specify, select, and configure an error reporter that is activated if an endpoint-related background process fails as described in **IIEP - Error Handling & Reporting**.
11. Enable the endpoint and invoke it as described in **Running an Inbound Integration Endpoint**.

Additional Information

The following information is useful once an inbound integration endpoint is set up:

1. Understand each element of an IIEP as described in **Inbound Integration Endpoint Structure**.
2. Understand the effects of the transactional setting on an IIEP as described in **Integration Endpoint Transactional Settings**.

3. Maintain or modify the endpoint as described in **Maintaining an Inbound Integration Endpoint**.
4. Monitor the endpoint as described in **Monitoring an IIEP via Background Process**.
5. Monitor the endpoint as described in **Monitoring an IIEP via External System**.
6. Resolve failed background processes as described in **Handing Failed IIEP Background Processes**.
7. Export an inbound integration endpoint definition for comparison purposes in an external source control system for comparison purposes as described in the **Configuration Management** documentation.
8. Parallel imports involving multiple references on object types as described in **Reference Target Lock Policy on Object Types**.

Initial Setup for an Inbound Integration Endpoint

Before creating an inbound integration endpoint, the following items must exist:

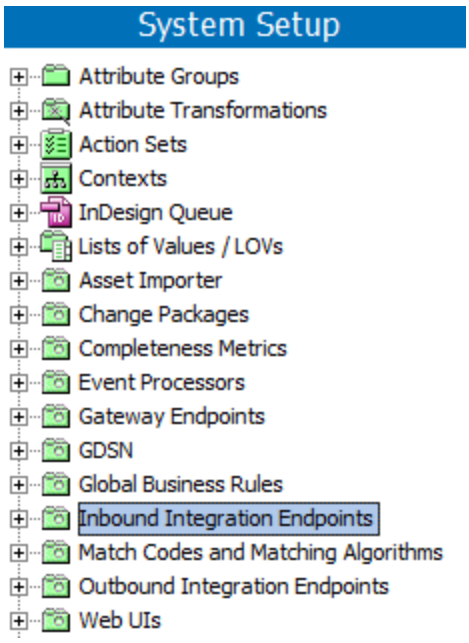
- An Inbound Integration Setup Group
- Linking between the IIEP Object Type and the Setup Group
- An instance of the IIEP Object

This setup only needs to be performed once, and most systems will already have it completed.

Only users with the relevant privileges can view or maintain inbound integration endpoints. For detailed information, refer to the **Action Sets** section and the **Users and Groups** section in the **System Setup** documentation.

Note: Attempting to delete a user who is responsible for an IIEP will result in errors. However, if a deleted user was set as an approver, the field will be emptied without notification. For more information, refer to the **Working with Users** topic in the **System Setup** documentation.

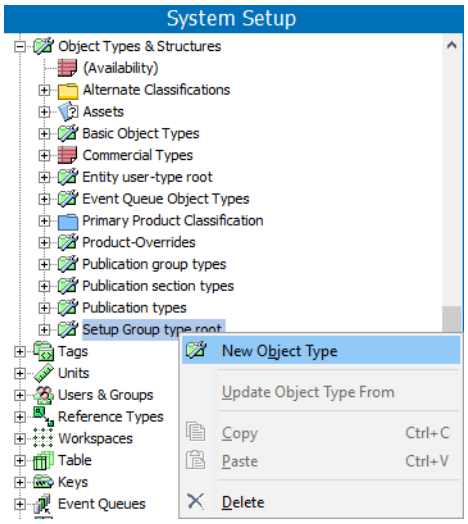
Review your System Setup tab to determine if one or more inbound integration endpoint nodes already exist. The name of the node on your system is not required to match the one in the image below.



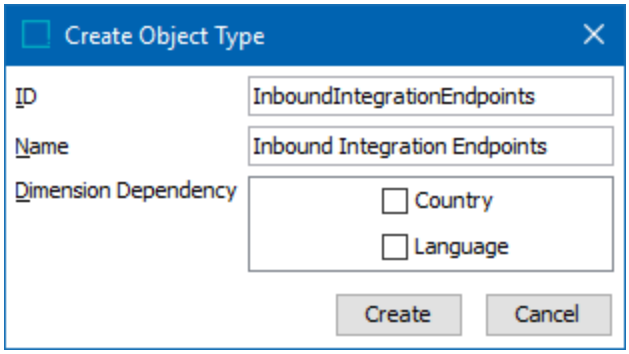
Once the setup has been completed, the steps in this section are only needed if additional levels of organization are desired.

Create the Inbound Integration Setup Group

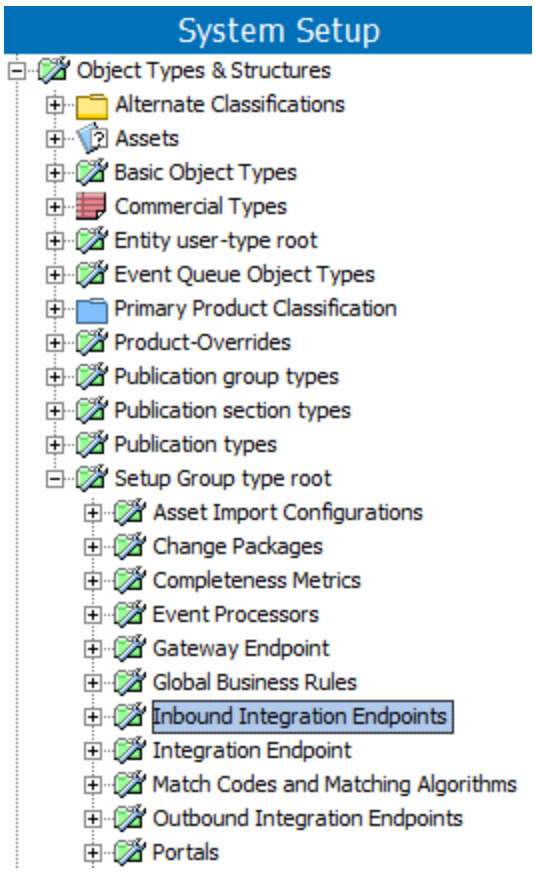
1. In System Setup, expand **Object Types & Structures**.
2. Right-click 'Setup Group type root', and choose **New Object Type**.



3. Enter an **ID** and a **Name**, select any required Dimension Dependencies based on your system, and click **Create**.

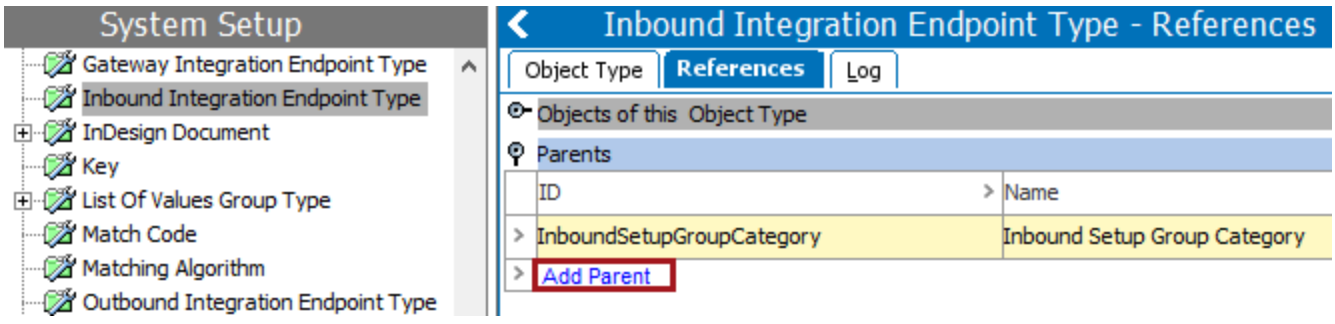


4. The new setup group appears in System Setup under 'Object Types & Structures' as a child in the **Setup Group type root**.

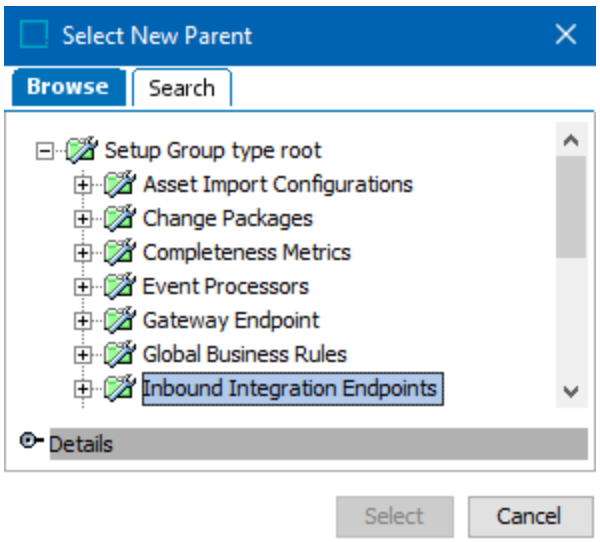


Link the IIEP Object Type to Setup Group

1. In Object Types & Structures > expand Basic Object Types > select **Inbound Integration Endpoint Type**.
2. On the References tab > Parents flipper > click the **Add Parent** link.

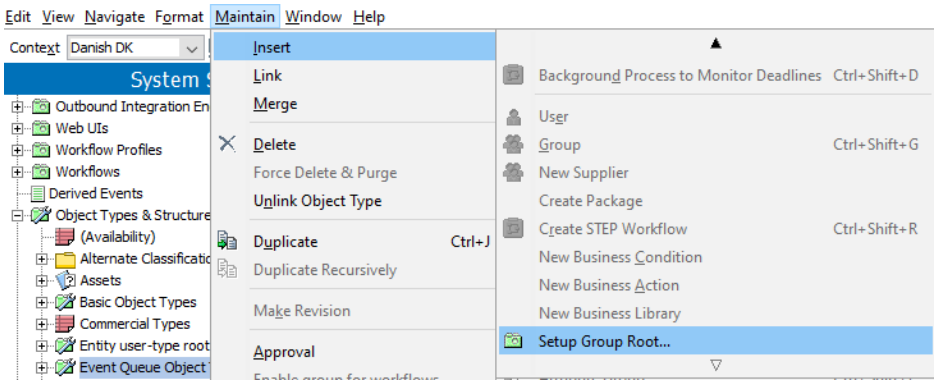


3. In the Select New Parent dialog, select the setup group you created, and click **Select** to make it a valid parent.

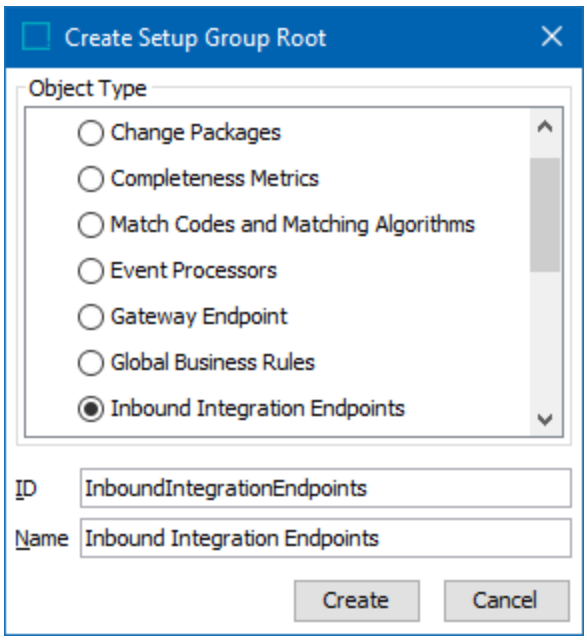


Create an Instance of the IIEP Object

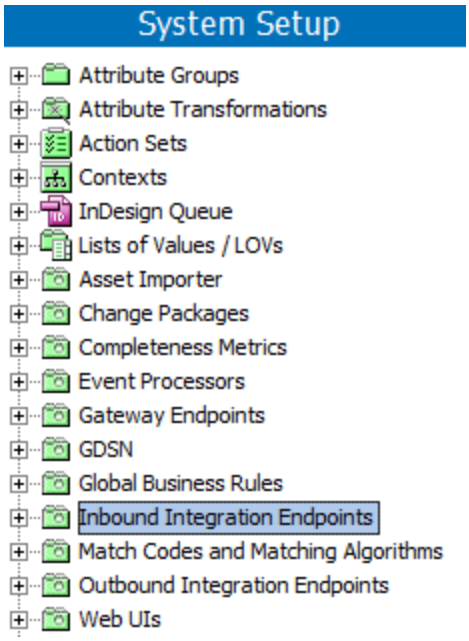
1. On the System Setup tab, select any object in the **System Setup** hierarchy to activate the following Maintain menu selection.
2. Click the Maintain menu, point to Insert, and select **Setup Group Root**.



3. In the Create Setup Group Root dialog, select the inbound integration endpoints object type, enter an **ID** and a **Name**, and click **Create**.



A setup group is created as a node in the System Setup hierarchy. Inbound integration endpoints can now be created under this new node.

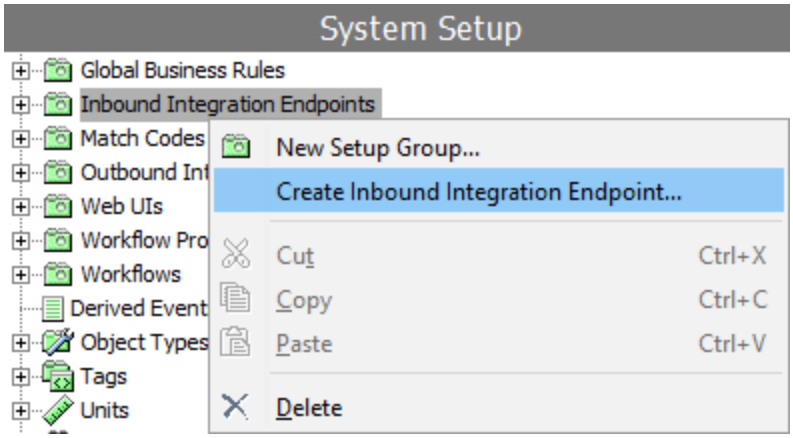


Creating an Inbound Integration Endpoint

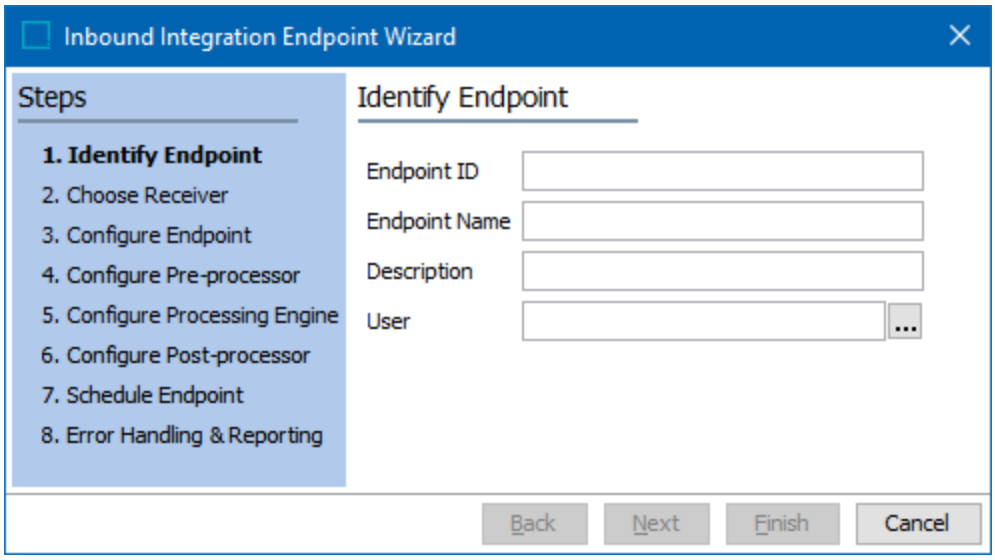
After creating a setup group for inbound integration endpoints, create an inbound integration endpoint to monitor and maintain integrations with systems that send data to STEP in the form of files or messages.

Important: Excel binary files, those with an XLSB extension, are not supported during import.

1. In System Setup, right-click the Inbound Integrations Endpoints setup group, and click **Create Inbound Integration Endpoint**.



2. The Inbound Integration Endpoint wizard displays and can involve the following steps:

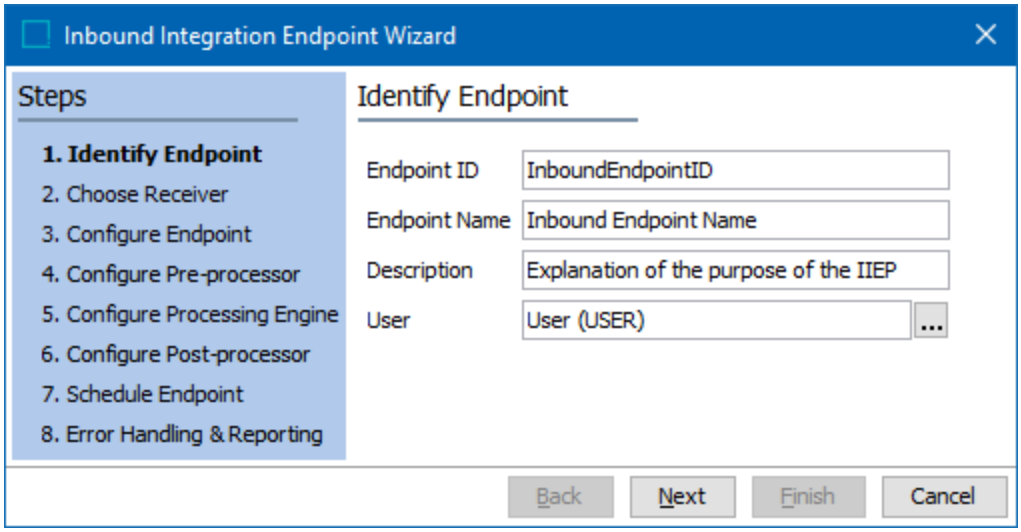


   - **Identify Endpoint** allows you to specify the name and ID of the endpoint and identify the user whose privileges are applied.
   - **Choose Receiver** allows you to specify the receiver of the data.

- **Configure Endpoint** allows you to specify processing, context and queue settings for the integration endpoint.
- **Configure Pre-processor**, when available, allows you to configure defined pre-processors for customer-specific solutions.
- **Configure Processing Engine** allows you to specify the data format and map the data, among other options.
- **Configure Post-processor**, when available, allows you to configure defined post-processes for customer-specific solutions.
- **Schedule Endpoint** allows you to specify how often the endpoint should search for data to be processed.
- **Error Handling & Reporting** allows you to configure automated retries for connection errors and an error reporter that is activated if an endpoint-related background process fails.

3. Follow the steps defined in **Running an Inbound Integration Endpoint** to enable and run the endpoint.

IIEP - Identify Endpoint



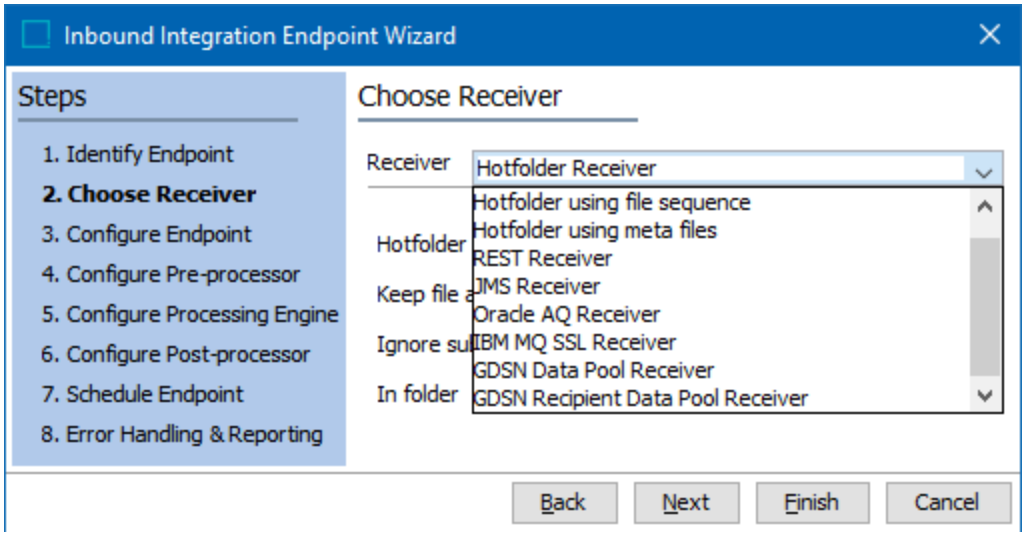
1. For **Endpoint ID**, enter an ID. Common setup is to use no spaces or punctuation.
2. For **Endpoint Name**, enter a name. Common setup is to repeat the ID with added spaces for readability.
3. For **Description**, enter an optional description for the IIEP.
4. For **User**, search or browse for a user. The privileges of the selected user determine which actions the inbound integration endpoint can perform when it processes data.

   Note: It is common setup to create a special system user account for use by each IIEP. This allows you to track changes made by the integration and also allows you to restrict (via privileges) what data the integration can change. This is especially relevant for STEPXML based IIEPs since STEPXML allows you to make significant changes to both data and configuration settings.

   Attempting to delete a user who is selected in the User parameter for an IIEP will result in errors. However, if a deleted user was set as an approver, the field will be emptied without notification. For more information, refer to the **Working with Users** topic in the **System Setup** documentation.

5. Click the **Next** button to display **IIEP - Choose Receiver**.

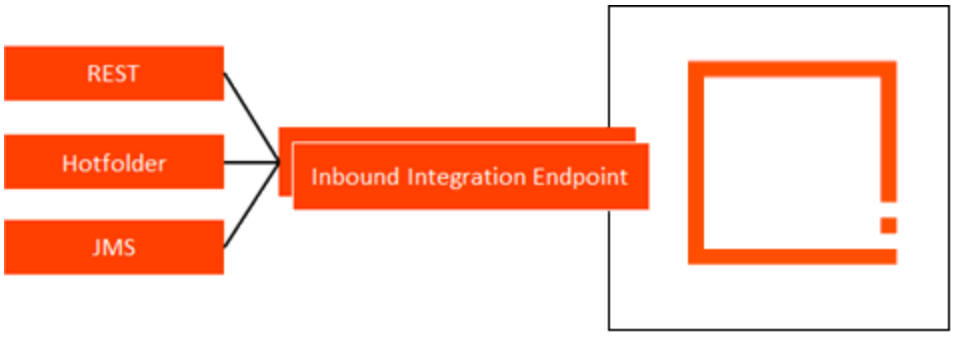
IIEP - Choose Receiver



1. Select the receiver to be used for the inbound endpoint. The available standard receivers are listed in the Receiver Methods section below.
2. Click the **Next** button to display **IIEP - Configure Endpoint**.

Receiver Methods

Data being imported into STEP via an IIEP can originate using any of the IIEP receiver methods described below. When the data is available, the IIEP processes based on its configuration, and then performs the import. Pictured below is an example of some receiver methods available.



Note: Import Manager does not require a receiver method since the user chooses a single file for import. For more on Import Manager, refer to the **Import Manager** topic.

The following receiver options are only available when importing data using an IIEP, and some are based on a license.

Method	Description
Amazon SQS	Receives messages from an Amazon SQS (Amazon Simple Queue Service). Refer to the **Amazon SQS Receiver** documentation.
Dynamic JMS Receiver	Dynamic Java Message Service (JMS) receivers are system dependent. By default, the Dynamic JMS Receiver option lets you consume and dequeue messages on defined queues. Unlike the JMS Receiver mentioned above, this receiver allows customers to supply the vendor-specific JMS libraries and JNDI configuration. Refer to the **Dynamic JMS Receiver** documentation.
GDSN Data Pool Receiver	The GDSN solution must be implemented fully before the GDSN Data Pool Receiver method is available and functional. Refer to the **GDSN Receiver Solution Enablement** documentation.
GDSN Recipient Data Pool Receiver	The GDSN solution must be implemented fully before the GDSN Recipient Data Pool Receiver method is available and functional. Refer to the **GDSN Receiver Solution Enablement** documentation.
Hotfolder Receiver	Enables setup of a standard data hotfolder, typically on the application server. Refer to the **Hotfolder Receiver** documentation.
Hotfolder using file sequence	Enables setup of a standard data hotfolder, typically on the application server, where file names include sequence ID to determine the file processing order. Refer to **Hotfolder Receiver Using File Sequence** documentation.
Hotfolder using meta files	Enables the setup of a standard data hotfolder, typically on the application server, where one or more simple .txt meta files determine the file processing order. Refer to the **Hotfolder Receiver Using Meta Files** documentation.
IBM MQ SSL	IBM MQ SSL, uses MQ series Secure Sockets Layer, and enables data exchange across IBM and non-IBM platforms. Refer to the **IBM MQ SSL** documentation.
JMS Receiver	Java Message Service (JMS) receivers are system dependent. By default, the JMS Receiver option lets you consume and dequeue messages on defined queues. Refer to the **JMS Receiver** documentation.
Kafka Receiver	Apache Kafka is an open-source distributed event-streaming data platform. The Kafka receiver enables STEP platforms integrated with Apache Kafka to take advantage of built-in options for inbound processing to a Kafka queue. Refer to the **Kafka Receiver** documentation.

Method	Description
Oracle AQ Receiver	Oracle Advanced Queuing (Oracle AQ) enables messages to be exchanged between two systems. Refer to the **Oracle AQ Receiver** documentation.
Product Data Exchange Receiver	STEP receives data from the Product Data Exchange (PDX) platform via the default PDX Inbound Integration Endpoint and the API. Refer to the **Product Data Exchange Receiver** documentation.
REST Receiver	Representational State Transfer (REST) is a software architecture style used to design web services. Refer to the **REST Receiver** documentation.
Web UI File Loading Receiver	Receiver used for the Web UI 'File Loading Widget' for non-hotfolder-based receiver types. Refer to the **Web UI File Loading Receiver** documentation.

Amazon SQS Receiver

Using the Amazon SQS Receiver allows data to be imported into STEP from an Amazon SQS queue on AWS. For Amazon S3 storage, choose FIFO or standard queues based on your integration needs. Generally, if Amazon S3 storage is used for large messages, standard queues are recommended since a message that points to the S3 location is placed on the SQS queue and the large file is placed on S3. To determine the best option for your implementation, search the Amazon SQS Developer Guide online for details on the 'FIFO queues' and 'standard queues' options.

The Amazon SQS queues require a file size of less than 256KB. Common setup is to use file compression (zip export file) to reduce the message size to meet this limitation. Amazon S3 storage account is required to handle files 256KB or larger. For more information, search Amazon S3 online.

The Amazon SQS Receiver option can work in conjunction with the Amazon SQS Delivery Method on an OIEP to move messages into and out of STEP via AWS.

To access the Amazon SQS Receiver, a Receiver.AmazonSQS component must be activated on your system in addition to the normal update procedures. Contact your Stibo Systems representative for more information.

Prerequisites

Changes to the properties file, outlined below, are implemented when the server is restarted.

1. For the 'Credentials profile' parameter, create a credentials file following the instructions at http://docs.aws.amazon.com/cli/latest/userguide/cli-config-files.html. Place the credentials file in the required path on the application server. This file is used by both the OIEP Amazon SQS Delivery Method and the IIEP Amazon SQS Receiver.
2. Prior to configuration, clicking the **Server URL** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **SQSServerUrl** property. The following is an example of a property entry for a single server:

   ```
   SQSServerUrl=1=https://sqs.us-east-2.amazonaws.com
   ```

3. Prior to configuration, clicking the **Credentials Path** dropdown parameter displays the required property name. Provide a selection, including the name of the credentials file, for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **SQSCredentialsPath** property. If necessary, use a comma and increment the number to separate multiple paths as shown in the example below.

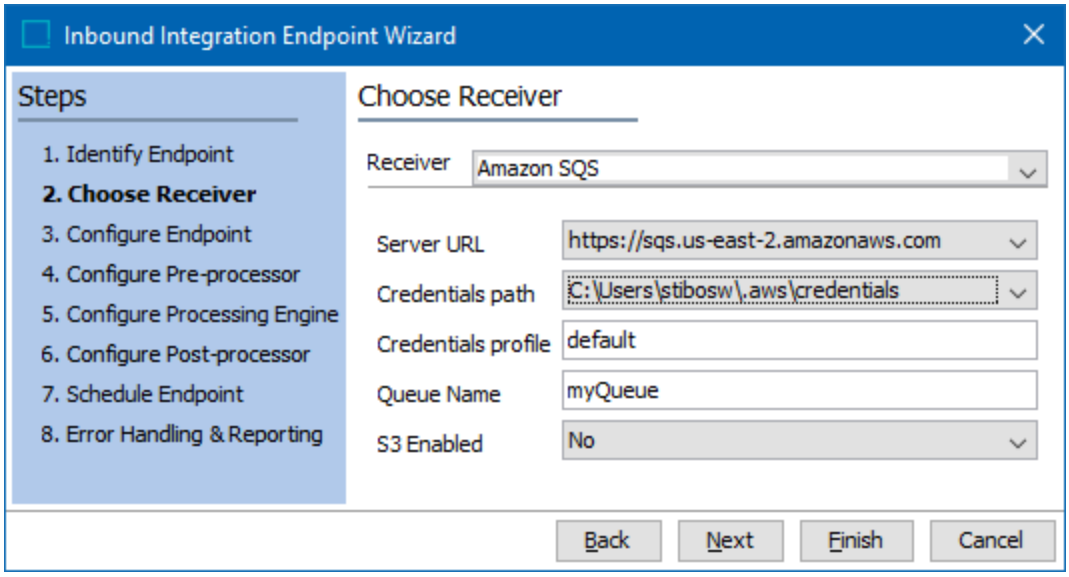
   ```
   SQSCredentialsPath=1=C:\\Users\\stibosw\\.aws\\credentials,2=C:\\Users\\stibosw\\.aws\\credentials123
   ```

4. For the 'Queue name' parameter, if necessary, create a new queue on the SQS server. Use the steps defined in the **Amazon SQS Queue Configuration** section of the **Amazon SQS Delivery Method** topic.
5. If necessary, establish the Amazon S3 storage account and create the required buckets. For details, refer to Amazon S3 on the web.

Contact your Stibo Systems representative if you need assistance with setup.

Configuration

After completing the prerequisite steps, edit the receiver of the IIEP. Use the following parameters to configure the IIEP. For information on a parameter, hover over the parameter field to display help text.



1. For **Receiver**, choose **Amazon SQS**.
2. In **Server URL**, select the URL that points to the SQS server where the data will be delivered.
3. In **Credentials path**, select the path to the desired SQS credentials file.
4. In **Credentials profile**, enter the profile name included in the SQS credentials profile.
5. In **Queue Name**, enter the Amazon queue created for this delivery. This queue is also required when Amazon S3 is being used.
6. For **S3 Enabled**, select 'Yes' if you have an Amazon S3 storage account. Setting this option to 'No' results in an error for messages that exceed the 256KB size limit.

   Note: Generally, if Amazon S3 storage is used for large messages, standard queues are recommended since a message that points to the S3 location is placed on the SQS queue and the large file is placed on S3.

7. Click the **Next** button to continue with the Configure Endpoint step, or the **Finish** button to close the wizard.

Dynamic JMS Receiver

The Dynamic JMS Receiver uses the Java Message Service (JMS) messaging standard to retrieve data from external systems. Unlike the **JMS Receiver**, the Dynamic JMS Receiver allows customers to supply the vendor-specific JMS libraries and JNDI configuration and, for example, upgrade to newer versions of these without Stibo Systems' involvement.

Important: This standard functionality only supports queues. Support for topics requires custom development via the **Extension API** (available from the Start Page using the **STEP API Documentation** link). Alternatively, topics can be supported using middleware to move the message from a queue to a topic.

The receiver has been designed to work with any Java Message Service 2.0 specification-compliant client library (specification defined by JSR 343: https://jcp.org/aboutJava/communityprocess/final/jsr343/index.html), but will also work with systems that implement older versions of the specification.

Note: The receiver has been tested with the following message brokers: Apache Active 5.15.8 and RabbitMQ 3.7.10.

Prerequisites

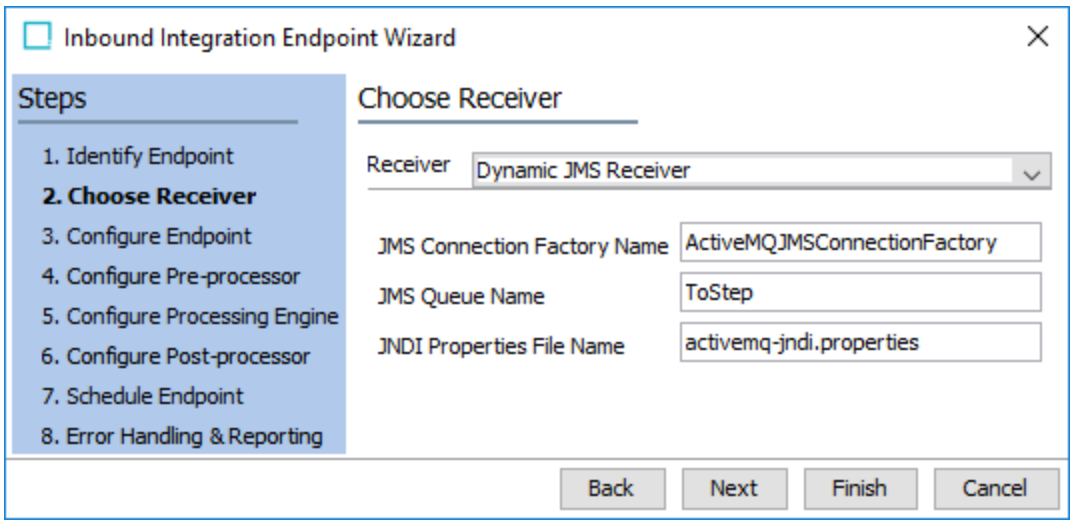
Changes to the properties file or any files found in the class path directory, as outlined below, are implemented when the server is restarted.

1. Verify the **jms-integration** add-on component is installed by reviewing the 'About STEP' option on the Start Page. Enter your credentials and click the 'Detailed version information' link. For on-premises systems, instructions for installing components can be found in the **SPOT Program** topic in the **System Administration Guide** found in **Downloadable Documentation**. For SaaS systems, contact Stibo Systems Support.
2. To enable the Dynamic JMS Receiver, the case-sensitive configuration property `JMS.ClassPath` must be set in sharedconfig.properties on the STEP application server and should point to an existing directory accessible from all application servers. Client libraries and JNDI files (Java Naming and Directory Interface) must be placed in this directory. Refer to the **Dynamic JMS Configuration Examples** section below for more information.

Once the server-side configuration is in place, the Dynamic JMS Receiver can be configured via the workbench.

Configuration

After completing the prerequisite steps, edit the receiver of the IIEP. Use the following parameters to configure the IIEP. For information on a parameter, hover over the parameter field to display help text.



1. For **Receiver**, choose **Dynamic JMS Receiver**.
2. For **JMS Connection Factory Name**, specify a JMS connection factory name. The selection must match the entry configured in the JNDI file.
3. For **JMS Queue Name**, select the physical name of the JMS Queue to be used. The selection must match the entry configured in the JNDI file.
4. For **JNDI Properties File Name**, enter the name of the JNDI file.
5. Click the **Next** button to continue with the **Configure Endpoint** step, or the **Finish** button to close the wizard.

Dynamic JMS Configuration Examples

As outlined in the following examples, in any JMS implementation, you must:

1. Put client jar files in the classpath folder.
2. Write the JNDI properties file.
3. Configure the connectionfactory and queue in the IIEP receiver.

Important: In the JNDI file, secrets (like passwords) are not encrypted.

Azure Message Bus

Place these libraries in the directory pointed to by the JMS.ClassPath configuration property:

- qpid-jms-client-0.61.0.jar
- geronimo-jms_2.0_spec-1.0-alpha-2.jar

- proton-j-0.33.10.jar
- netty-buffer-4.1.77.Final.jar
- netty-codec-4.1.77.Final.jar
- netty-common-4.1.77.Final.jar
- netty-handler-4.1.77.Final.jar
- netty-resolver-4.1.77.Final.jar
- netty-transport-4.1.77.Final.jar

Important: Although the main JMS library is qpid-jms-client-[version].jar, the other libraries (listed above) are also required as these are underlying dependencies. Failing to include all of these in the JMS classpath will result in error messages similar to this one: java.lang.NoClassDefFoundError: io/netty/util/concurrent/EventExecutorGroup

Note: Due to a change in the Qpid JMS client's implementation, version 1.6.0+ of this library is not compatible with STEP's Dynamic JMS integration.

In the JNDI properties file, add properties to set the context factory and map queues, setting connectionfactory = SBCF as shown below:

```
java.naming.factory.initial = org.apache.qpid.jms.jndi.JmsInitialContextFactory
connectionfactory.SBCF = amqps://[servicebus_
name].servicebus.windows.net?jms.username=[username]&jms.password=[password]
```

Register queues in JNDI using the form:

- queue.[servicebus_queue_name] = [servicebus_queue_name]

Example azure-jndi.properties file:

```
java.naming.factory.initial = org.apache.qpid.jms.jndi.JmsInitialContextFactory
connectionfactory.SBCF =
amqps://pimtest.servicebus.windows.net?jms.username=RootManageSharedAccessKey&jms.pa
ssword=4NzL79KOhmD8A9N8bA9QSTY3zxTHX9Hy1602xan0bqk=
queue.inbound-products = inbound-products
```

In the corresponding IIEP receiver, the JMS entries should have the following format:

- JMS Connection Factory Name SBCF
- JMS Queue Name [servicebus_queue_name]
- JNDI Properties File Name [jndi_properties_file_name]

Example receiver configuration:

- JMS Connection Factory Name = SBCF
- JMS Queue Name = inbound-products
- JNDI Properties File Name = azure-jndi.properties

For more information on the Qpid JMS client library, refer to https://github.com/MicrosoftDocs/azure-docs/blob/master/articles/service-bus-messaging/service-bus-java-how-to-use-jms-api-amqp.md

TibcoJMS

Libraries to be placed in the directory pointed to by the JMS.ClassPath configuration property:

- Tibjms.jar

The 'tibjms.jar' file can typically be found in:

```
components/shared/1.0.0/plugins/com.tibco.jms.jmsclient_[VersionNumber]
```

JNDI file content (example – file to be placed in the same directory as the libraries):

```
java.naming.provider.url=tibjmsnaming://[server1]:[port1], tibjmsnaming://[server2]:
[port2]
java.naming.factory.initial=com.tibco.tibjms.naming.TibjmsInitialContextFactory
java.naming.factory.url.pkgs=com.tibco.tibjms.naming
java.naming.security.principal=[username]
java.naming.security.credentials=[password]
```

Example receiver configuration:

- JMSConnectionFactoryName='QueueConnectionFactory'
- JMSQueueName='ToStep'
- JNDIPropertiesFileName='tibco-jndi.properties'

With this configuration, use the connection factory ‘QueueConnectionFactory’ or the factory configured in the Kaazing Gateway 'factories.conf' file. Queue names can either be the name used in Tibco, or 'queue.[*add_the_tibcoqueuename*]' (which is needed when a topic exists on the JMS server with the same name as the queue you are trying to connect to).

Apache Active MQ 5.15.8

Libraries to be placed in the directory pointed to by the JMS.ClassPath configuration property:

- activemq-client-5.15.8.jar
- hawtbuf-1.11.jar

JNDI file content (example – file to be placed in the same directory as the libraries):

```
java.naming.factory.initial = org.apache.activemq.jndi.ActiveMQInitialContextFactory
java.naming.provider.url = tcp://127.0.0.1:61616
connectionFactoryNames = ActiveMQJMSConnectionFactory
queue.FromStep = FromStep
queue.ToStep = ToStep
java.naming.security.principal=admin
java.naming.security.credentials=admin
```

Example receiver configuration:

- JMS Connection Factory Name = 'ActiveMQJMSConnectionFactory'
- JMS Queue Name = 'ToStep'
- JNDI Properties File Name = 'activemq-jndi.properties'

Apache RabbitMQ 1.11.1

In RabbitMQ, create the necessary Queue(s) and Exchange(s) to allow message publishing and error reporting. For more information, search the web.

Libraries to be placed in the directory pointed to by the JMS.ClassPath configuration property:

- amqp-client-5.5.2.jar
- fscontext-4.6-b01.jar
- rabbitmq-jms-1.11.1.jar

Note: If credentials are to be provided, add them in the JNDI file as shown below since having the credentials in the .bindings file will not work.

JNDI file content (example – file to be placed in the same directory as the libraries):

```
java.naming.factory.initial = com.sun.jndi.fscontext.RefFSContextFactory
java.naming.provider.url = file:resources/jmsclasspath/

#Credentials
java.naming.security.principal = username
java.naming.security.credentials = password
```

'.binding' file content (example - file with this specific name to be placed in the same directory as the libraries):

```
# ConnectionFactory
ConnectionFactory/ClassName=javax.jms.ConnectionFactory
ConnectionFactory/FactoryName=com.rabbitmq.jms.admin.RMQObjectFactory
ConnectionFactory/RefAddr/0/Content=jms/ConnectionFactory
ConnectionFactory/RefAddr/0/Type=name
ConnectionFactory/RefAddr/0/Encoding=String
ConnectionFactory/RefAddr/1/Content=javax.jms.ConnectionFactory
ConnectionFactory/RefAddr/1/Type=type
ConnectionFactory/RefAddr/1/Encoding=String
ConnectionFactory/RefAddr/2/Content=com.rabbitmq.jms.admin.RMQObjectFactory
ConnectionFactory/RefAddr/2/Type=factory
ConnectionFactory/RefAddr/2/Encoding=String
ConnectionFactory/RefAddr/3/Content=localhost
ConnectionFactory/RefAddr/3/Type=host
ConnectionFactory/RefAddr/3/Encoding=String
# rabbitmqQueue01 Queue
rabbitmqQueue1/ClassName=javax.jms.Queue
rabbitmqQueue1/FactoryName=com.rabbitmq.jms.admin.RMQObjectFactory
rabbitmqQueue1/RefAddr/0/Content=jms/Queue
rabbitmqQueue1/RefAddr/0/Type=name
rabbitmqQueue1/RefAddr/0/Encoding=String
rabbitmqQueue1/RefAddr/1/Content=javax.jms.Queue
rabbitmqQueue1/RefAddr/1/Type=type
rabbitmqQueue1/RefAddr/1/Encoding=String
rabbitmqQueue1/RefAddr/2/Content=com.rabbitmq.jms.admin.RMQObjectFactory
rabbitmqQueue1/RefAddr/2/Type=factory
rabbitmqQueue1/RefAddr/2/Encoding=String
# Queue Name
rabbitmqQueue1/RefAddr/3/Content=rabbitmqQueue1
rabbitmqQueue1/RefAddr/3/Type=destinationName
rabbitmqQueue1/RefAddr/3/Encoding=String
rabbitmqQueue1/RefAddr/4/Content=true
rabbitmqQueue1/RefAddr/4/Type=amqp
rabbitmqQueue1/RefAddr/4/Encoding=String
# Exchange Name
rabbitmqQueue1/RefAddr/5/Content=rabbitmqQueue1
rabbitmqQueue1/RefAddr/5/Type=amqpExchangeName
rabbitmqQueue1/RefAddr/5/Encoding=String
# Routing Key (if apply)
rabbitmqQueue1/RefAddr/6/Content=rabbitmqQueue1
rabbitmqQueue1/RefAddr/6/Type=amqpRoutingKey
rabbitmqQueue1/RefAddr/6/Encoding=String
# AMQP Queue Name
rabbitmqQueue1/RefAddr/7/Content=rabbitmqQueue1
rabbitmqQueue1/RefAddr/7/Type=amqpQueueName
rabbitmqQueue1/RefAddr/7/Encoding=String
```

Example receiver configuration:

- JMS Connection Factory Name = ConnectionFactory
- JMS Queue Name = rabbitmqQueue01
- JNDI Properties File Name = rabbitmq-jndi.properties

WebLogic JMS Connection

Libraries to be placed in the Workarea directory, which points to by the jmslasspath configuration property:

- weblogic.jar
- wlclient.jar
- wlthint3client.jar

Important: Because STEP will attempt to use some classes that are in both the `wljmsclient.jar` and the `wlthint3client.jar` libraries, conflicts will generate. To prevent this conflict, use `wlthint3client.jar` library instead of the `wljmsclient.jar` library.

The weblogic-JNDI configuration includes:

```
java.naming.factory.initial=weblogic.jndi.WLInitialContextFactory
java.naming.provider.url=t3://localhost:7001
```

This includes the queue name of:

```
jms/TestQueue=jms/TestQueue
```

Example delivery method configuration:

```
JMS Connection Factory Name = weblogic.jms.ConnectionFactory
JMS Queue Name = jms/TestQueue
JNDI Properties File Name = weblogic-jndi.properties
```

Hotfolder Receiver

This default hotfolder option enables the setup of a monitored directory as the receiver. Files are placed into this folder and are processed by the integration endpoint. Typically this is on the application server or can be reached from the application server. When the endpoint polls the hotfolder and finds more than one file, the files are handled in sequence starting with the oldest file first, based on the last modified data stamp. All formats supported by the standard import functionality are available via hotfolder import.

Note: The Asset Importer Processing Engine requires the use of the Hotfolder Receiver. For more information, refer to the **IIEP - Configure Asset Importer Processing Engine** documentation.

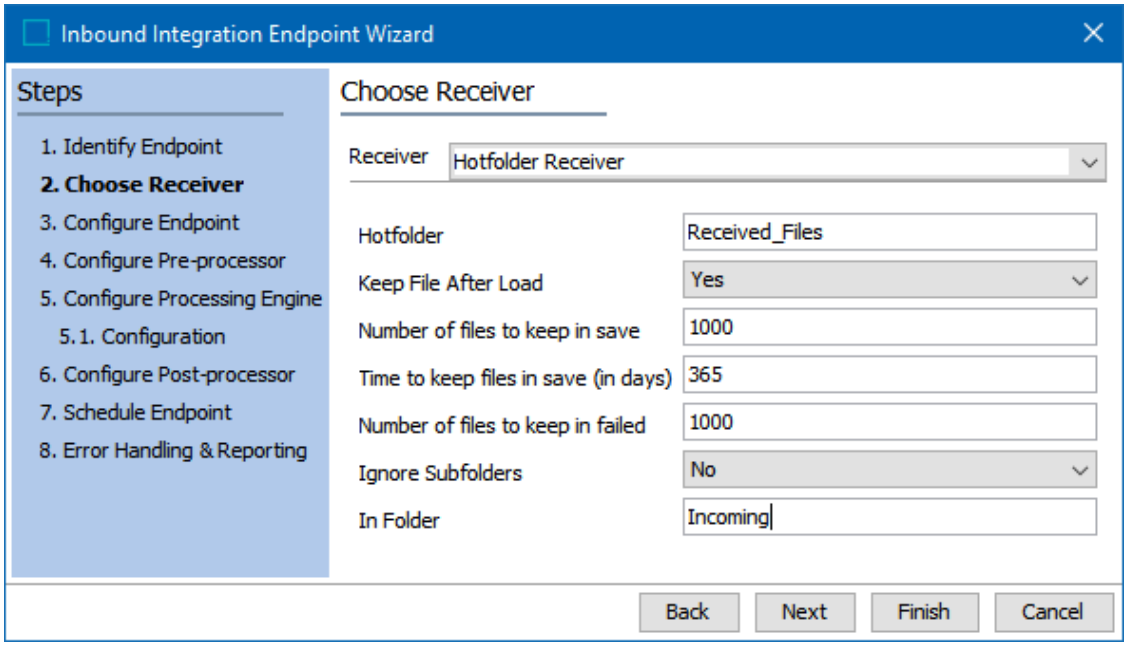
To control the order of the files to be imported, set up either a **Hotfolder Receiver Using Meta Files** or a **Hotfolder Receiver Using File Sequence** instead.

Prerequisites

The location of the 'Hotfolder' and 'In folder' parameters are determined by the sharedconfig.properties file **Install.HotfolderRoot** property. Changes to the properties file are implemented when the server is restarted.

Configuration

After completing the prerequisite steps, edit the receiver of the IIEP. Use the following parameters to configure the IIEP. For information on a parameter, hover over the parameter field to display help text.



1. For **Receiver**, choose **Hotfolder Receiver**.
2. For **Hotfolder**, enter the name used to create a folder on the application server in the 'upload' directory.
3. For **Keep file after load**, specify if files dropped in the hotfolder should be removed after processing the files.

- Yes - requires periodic manual cleanup on the server.
- No - removes the files automatically and is the common setup to prevent the need for manual cleanup.

4. For **Number of files to keep in save**, specifies the number of files to be saved in the save folder after automatic cleanup. Parameter can save up to 1,000 files.
5. For **Time to keep files in save (in days)**, specifies the number of days the files are saved in the save folder. Parameter can save files for up to 365 days.
6. For **Number of files to keep in failed**, specifies the number of files to be saved in the failed folder. Parameter can save up to 1,000 files.
7. For **Ignore sub folders**, specify if subfolders within the hotfolder (which contain files) should be ignored.
   - Yes - files in these folders will not be imported.
   - No - in order to process all files, regardless of their location within the hotfolder or a folder below it is the common setup.
8. For **In folder** is optional. Enter a name for an 'In' folder. The name entered is used to create a new folder under the directory specified above in the 'Hotfolder' parameter. If no folder is specified, files are dropped at the hotfolder top level.
9. Click the **Next** button to continue with the Configure Endpoint step, or the **Finish** button to close the wizard.

Hotfolder Receiver Using File Sequence

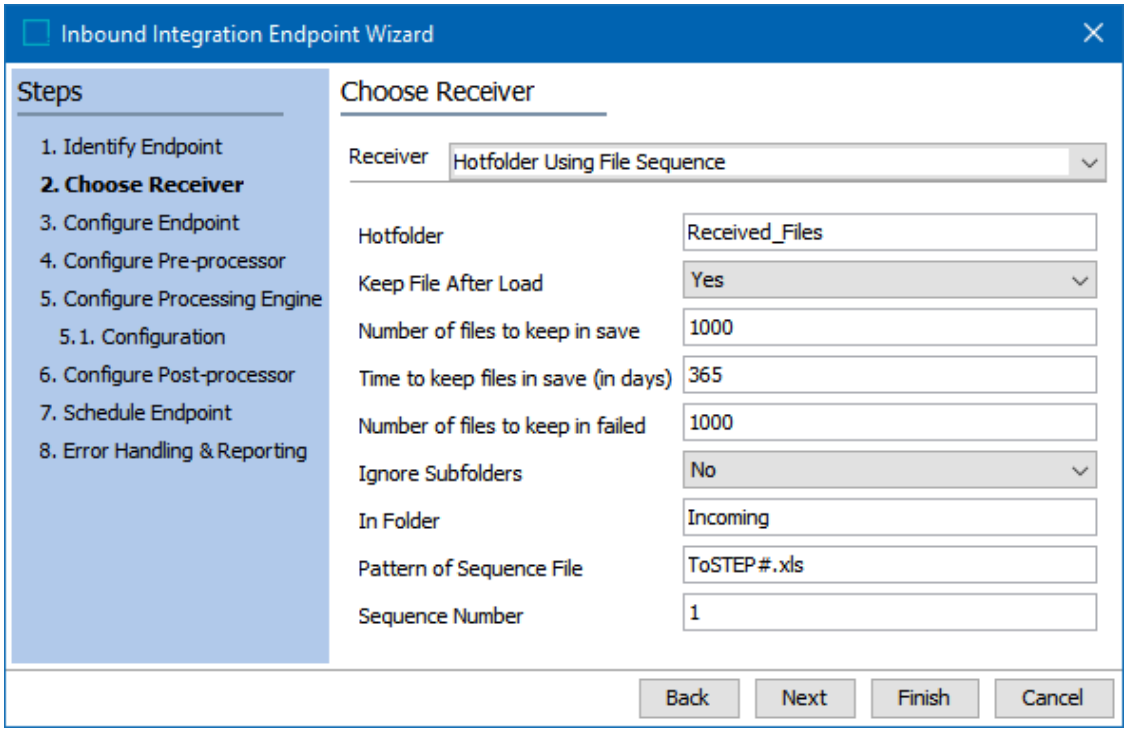
The 'Hotfolder using File Sequence' receiver enables setup of a standard data hotfolder where a sequence ID (included in the file name) determines the file processing order. Files are placed into this folder and are processed by the integration endpoint. Typically this is on the application server or can be reached from the application server. All formats supported by the standard import functionality are available via hotfolder import.

Prerequisites

The location of the 'Hotfolder' and 'In folder' parameters are determined by the sharedconfig.properties file **Install.HotfolderRoot** property. Changes to the properties file are implemented when the server is restarted.

Configuration

After completing the prerequisite steps, edit the receiver of the IIEP. Use the following parameters to configure the IIEP. For information on a parameter, hover over the parameter field to display help text.



1. For **Receiver**, choose **Hotfolder using file sequence**.
2. For **Hotfolder**, the name entered is used to create a folder on the application server in the 'upload' directory.
3. For **Keep file after load**, specify if files dropped in the hotfolder should be removed after processing the files. Selecting 'Yes' requires periodic manual cleanup on the server. Selecting 'No' removes the files automatically. Common setup is to select 'No' to prevent the need for manual cleanup.
4. For **Number of files to keep in save**, specifies the number of files to be saved in the save folder after automatic cleanup. Parameter can save up to 1,000 files.

5. For **Time to keep files in save (in days)**, specifies the number of days the files are saved in the save folder. Parameter can save files for up to 365 days.
6. For **Number of files to keep in failed**, specifies the number of files to be saved in the failed folder. Parameter can save up to 1,000 files.
7. For **Ignore sub folders**, specify if subfolders within the hotfolder (which contain files) should be ignored. If set to Yes, files in these folders will not be imported. Common setup is to select 'No' in order to process all files, regardless of their location within the hotfolder or a folder below it.
8. For **In folder** is optional. Enter a name for an 'In' folder. The name entered is used to create a new folder under the directory specified above in the 'Hotfolder' parameter. If no folder is specified, files are dropped at the hotfolder top level.
9. For **Pattern of Sequence File** is required. Enter the relevant file pattern. For example, if the hotfolder is used to parse XML files, the pattern could be: file#.xml. In this case, a file name must begin with *file* and have the extension *xml*. The # indicates a sequence number.
10. For **Sequence Number** is required. Enter the sequence number of the first file to be processed. For example, if the first number is 1, file1.xml is processed first, then file2.xml, and so on. The sequence number will automatically increment so that it is always possible to view the number for the next file. With this option, sequences can only be skipped if you manually reconfigure the 'Sequence Number.'
11. Click the **Next** button to continue with the Configure Endpoint step, or the **Finish** button to close the wizard.

Hotfolder Receiver Using Meta Files

This option enables the setup of a standard data folder where one or more meta files determine the file processing order of the data files. Files are placed into this folder and are processed by the integration endpoint. Typically this is on the application server or can be reached from the application server. All formats supported by the standard import functionality are available via hotfolder import.

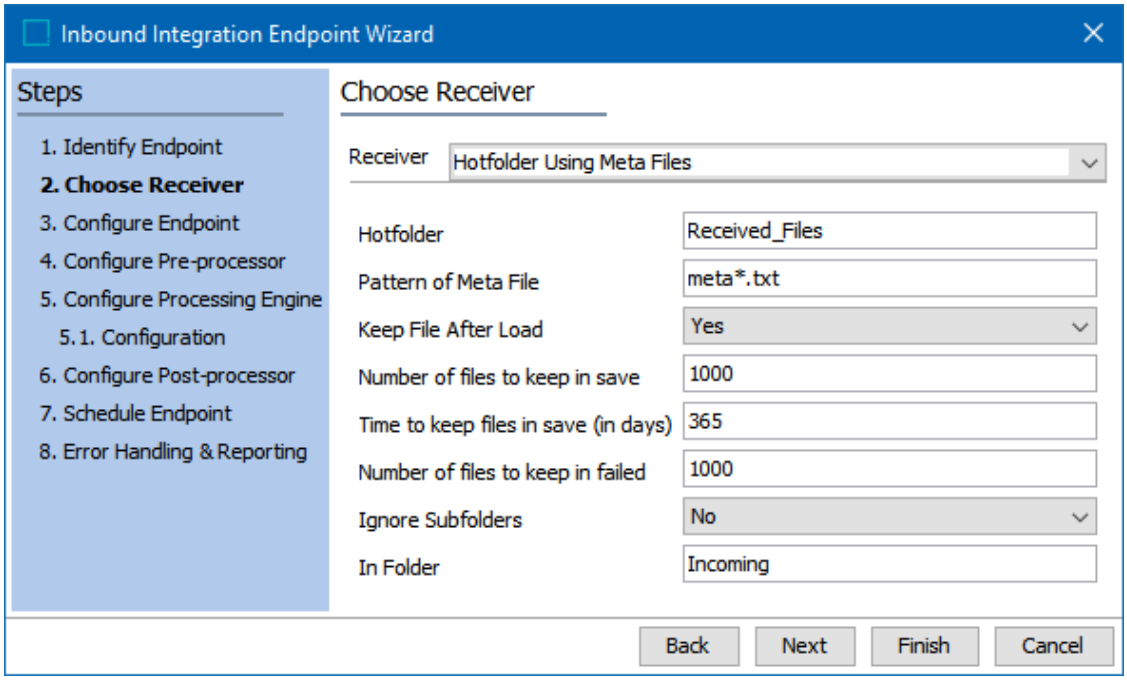
Prerequisites

The location of the 'Hotfolder' and 'In folder' parameters are determined by the sharedconfig.properties file **Install.HotfolderRoot** property. Changes to the properties file are implemented when the server is restarted.

A meta file is a simple .txt file dropped into the hotfolder together with files to be imported. The meta file must list the files to be processed, only one file is listed per line. Press the Enter key to create a new line within the meta file.

Configuration

After completing the prerequisite steps, edit the receiver of the IIEP. Use the following parameters to configure the IIEP. For information on a parameter, hover over the parameter field to display help text.



1. For **Receiver**, choose **Hotfolder using meta files**.
2. For **Hotfolder**, enter the name used to create a folder on the application server in the 'upload' directory.
3. For **Pattern of meta file** is required. Enter the name of the meta file that will be used to define the order of imported files. You can enter:

- An **exact name** of the meta file, for example, meta.txt. In this case, additional meta files cannot be dropped in the hotfolder at the same time. The hotfolder must complete the import specified in the meta file before additional meta file(s) and files can be imported.
- A **wildcard name** of the meta file, for example, sequence_*.txt. In this case, multiple meta files with names that match the wildcard name (for example, meta1.txt, meta2.txt, and so on) can exist in a hotfolder and will be processed. The hotfolder handles the meta files based on last modification date. This means that if you drop a meta1.txt and meta2.txt and the modification date in meta2.txt is older than meta1.txt, the hotfolder will process the files specified in meta2.txt before the files specified in meta1.txt.

4. For **Keep file after load**, specify if files dropped in the hotfolder should be removed after processing the files. Selecting 'Yes' requires periodic manual cleanup on the server. Selecting 'No' removes the files automatically. Common setup is to select 'No' to prevent the need for manual cleanup.
5. For **Number of files to keep in save**, specifies the number of files to be saved in the save folder after automatic cleanup. Parameter can save up to 1,000 files.
6. For **Time to keep files in save (in days)**, specifies the number of days the files are saved in the save folder. Parameter can save files for up to 365 days.
7. For **Number of files to keep in failed**, specifies the number of files to be saved in the failed folder. Parameter can save up to 1,000 files.
8. For **Ignore sub folders**, specify if subfolders within the hotfolder (which contain files) should be ignored. If set to Yes, files in these folders will not be imported. Common setup is to select 'No' in order to process all files, regardless of their location within the hotfolder or a folder below it.
9. For **In folder** is optional. Enter a name for an 'In' folder. The name entered is used to create a new folder under the directory specified above in the 'Hotfolder' parameter. If no folder is specified, files are dropped at the hotfolder top level.
10. Click the **Next** button to continue with the Configure Endpoint step, or the **Finish** button to close the wizard.

IBM MQ SSL Receiver

This IIEP-only receiver method allows connection with IBM MQ. Prior to release 2023.3, it was named 'IBM Websphere MQ SSL Receiver'. For information on connecting to IBM MQ in a non-SSL way, refer to the **JMS Receiver** topic.

Prerequisites

Changes to the properties file, outlined below, are implemented when the server is restarted.

Prior to configuration, clicking a dropdown parameter in the 'Choose Receiver' step displays the property name required to supply values that populate the parameter.

Note: In the sharedconfig.properties file, a numbered designation of an integer (**1=**, **2=**, etc.) in the value part of the property indicates that an entry should display in the UI. This allows multiple values to be stored for a single property and is required even when only a single value is required. If property values, such as passwords, should not be displayed in the UI, exclude the numbered designation, as shown in the password examples below.

The numbered designation indicates the order that the options are displayed in the dropdown. For example: `<Property name>=1=<Value 1>,2=<Value 2>,3=<Value 3>`. Using these numbered designations results in the dropdown showing values in the following order: `<Value 1>,<Value 2>,<Value 3>`. When duplicate integers exist, only the last value is displayed in the dialog.

As required, configure the following case-sensitive properties in the sharedconfig.properties on the STEP application server:

1. For **Connection URL** set the **WSMQSSLProviderURL** property.

   The elements are needed for each URL in the format [host]:[port]/[channel]:

   - [host] = hostname or IP of the IBM MQ server; in the follow example this is 10.46.88.75
   - [port] = port number for the channel; in the follow example this is 1415
   - [channel] = name of the channel; in the follow example this is BASE.CTL.SVRCONN

   ```
   WSMQSSLProviderURL=1=10.46.88.75:1415/BASE.CTL.SVRCONN
   ```

2. For **Queue Manager** set **WSMQSSLQueueManager** property.

   ```
   WSMQSSLQueueManager=1=HV088B
   ```

3. For **Queue Name** set the **WSMQSSLQueue** property.

   ```
   WSMQSSLQueue=1=LIVE.KITT
   ```

4. For **Key Store** set the **WSMQSSLKeyStoreLocation** property.

This must be in jks format, with the personal certificate for the Queue Manager. To generate, refer to the IBM MQ online help.

```
WSMQSSLKeyStoreLocation=1=file:/workarea/keystore.jks
```

5. For **Key Store Password**, set the **WSMQSSLKeyStorePassword** property.

   The password can be configured in sharedconfig.properties file or can be typed directly into the parameter. When the parameter is blank, the password from the property is used.

   To prevent the password (or any other selections) from displaying in the UI, do not include a numbered designation.

   ```
   WSMQSSLKeyStorePassword=Pa55w0rd1
   ```

6. For **Trust Store**, set the **WSMQSSLTrustStoreLocation** property.

   The Trust Store can be the same file as the Key Store. To generate, refer to IBM MQ online help.

   ```
   WSMQSSLTrustStoreLocation=1=file:/workarea/truststore.jks
   ```

7. For **Trust Store Password**, set the **WSMQSSLTrustStorePassword** property.

   The password can be configured in sharedconfig.properties file or can be typed directly into the parameter. When the parameter is blank, the password from the property is used.

   To prevent the password (or any other selections) from displaying in the UI, do not include a numbered designation.

   ```
   WSMQSSLTrustStorePassword=Pa55w0rd2
   ```

8. For **Cipher Suite**, set the **WSMQSSLCipherSuite** property.

   Set this property to the same value as SSL CipherSuite in IBM MQ.

   ```
   WSMQSSLCipherSuite=1=CTL_RSA_WITH_AES_256
   ```

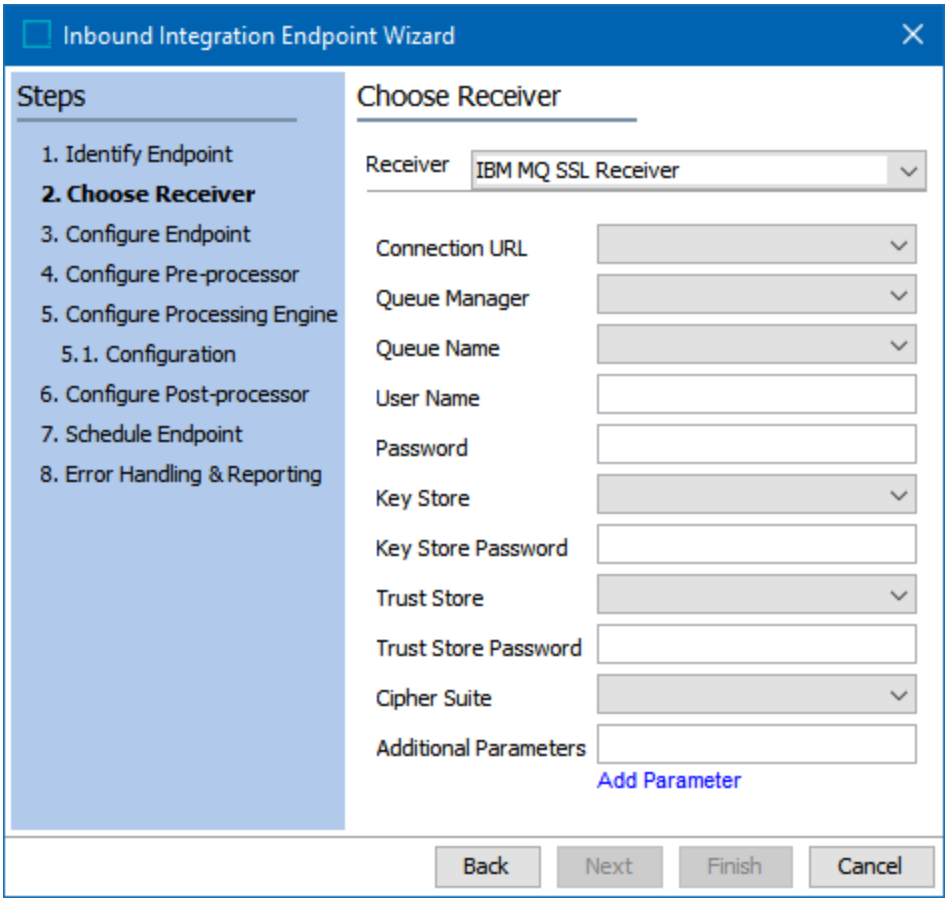
9. Contact your IT team to create a trust and key store and copy all URL certificates to the trust and key store.

Example of all property entries

```
WSMQSSLProviderURL=1=10.46.88.75:1415/BASE.CTL.SVRCONN
WSMQSSLQueueManager=1=HV088B
WSMQSSLQueue=1=LIVE.KITT
WSMQSSLKeyStoreLocation=1=/workarea/key.jks
WSMQSSLKeyStorePassword=Pa55w0rd1
WSMQSSLTrustStoreLocation=1=/workarea/key.jks
WSMQSSLTrustStorePassword=Pa55w0rd2
WSMQSSLCipherSuite=1=CTL_RSA_WITH_AES_256
```

Configuration

After completing the prerequisite steps, edit the receiver of the IIEP. Use the following parameters to configure the IIEP. For information on a parameter, hover over the parameter field to display help text.



1. For **Receiver**, choose **IBM MQ SSL Receiver**.
2. For **Connection URL**, select the URL for connection.
3. For **Queue Manager**, select the Queue Manager.
4. For **Queue Name**, select the name of the Queue for the connection.
5. For **User Name**, if required, enter the user name to be used with this integration.
6. For **Password**, if required, enter the password to be used with this integration.
7. For **Key Store**, select the key store.
8. For **Key Store Password**, leave the parameter blank to use the password from the sharedconfig.properties file, or manually type in a password.
9. For **Trust Store**, select the trust store with the CA for the Queue Manager.

10. For **Trust Store Password**, leave the parameter blank to use the password from the sharedconfig.properties file, or manually type in a password.
11. For **Cipher Suite**, since STEP is running on non-IBM jre, this must be the same value as configured in the Queue Manager.
12. For **Additional Parameters**, if required, click the **Add parameter** link, then enter the Key and Value pairs. For possible keys and values, consult the manual for IBM MQ.
13. Click the **Next** button to continue with the Configure Endpoint step, or the **Finish** button to close the wizard.

JMS Receiver

The available options for the Java Message Service (JMS) receiver are system dependent. By default, the JMS Receiver option lets you consume and dequeue messages (instead of files) on queues defined below. Supported formats include STEPXML, Generic XML, CSV (text messages). JMS is the preferred option where compatible enterprise middleware is available and there is a need for time-critical incremental updates from external systems.

The **Dynamic JMS Receiver** allows customers to supply the vendor-specific JMS libraries and JNDI configuration.

Important: This standard functionality only supports queues. Support for topics requires custom development via the **Extension API** (available from the Start Page using the **STEP API Documentation** link). Alternatively, topics can be supported using middleware to move the message from a queue to a topic.

Prerequisites

Changes to the properties file, outlined below, are implemented when the server is restarted.

1. Prior to configuration, clicking the **JMS Connection Factory Name** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **JMSConnectionFactoryName** property. For example, two factory names are displayed with the following property setting:

   ```
   JMSConnectionFactoryName=1=ConnectionFactory1,2=ConnectionFactory2
   ```

2. Prior to configuration, clicking the **JMS Provider URL** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **JMSProviderURL** property. For example, a single URL is displayed with the following property setting:

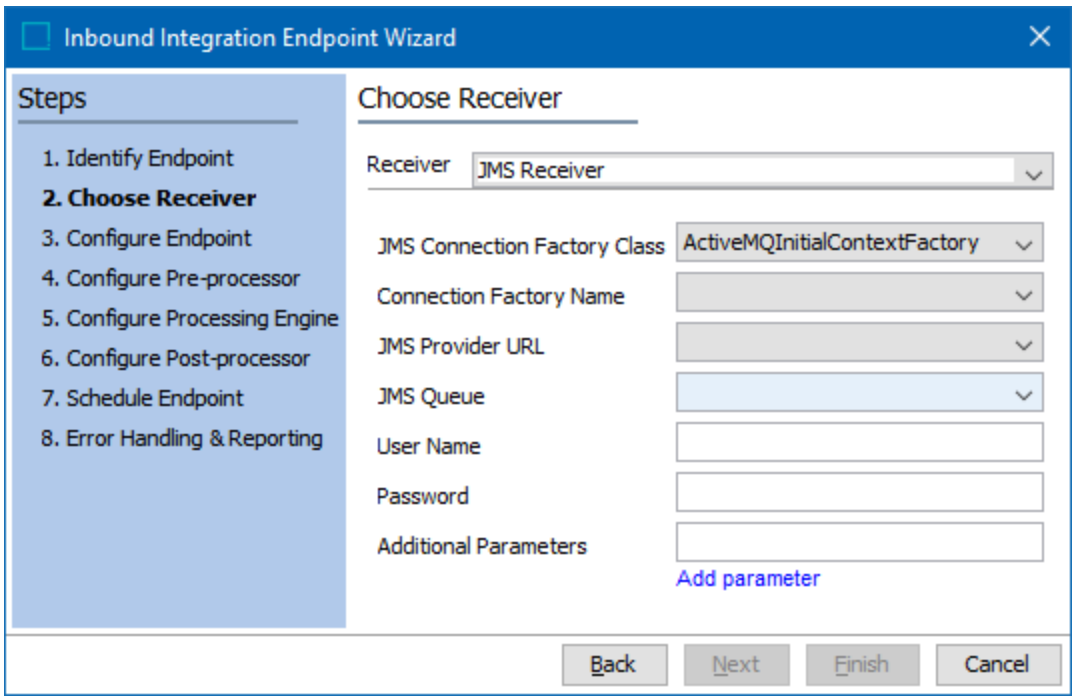
   ```
   JMSProviderURL=1=tcp://[IP address]:[Port, ActiveMQ default is 61616]
   ```

3. Prior to configuration, clicking the **JMS Queue** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **JMSQueue** property.

   ```
   JMSQueue=1=[Queue]
   ```

Configuration

After completing the prerequisite steps, edit the receiver of the IIEP. Use the following parameters to configure the IIEP. For information on a parameter, hover over the parameter field to display help text.



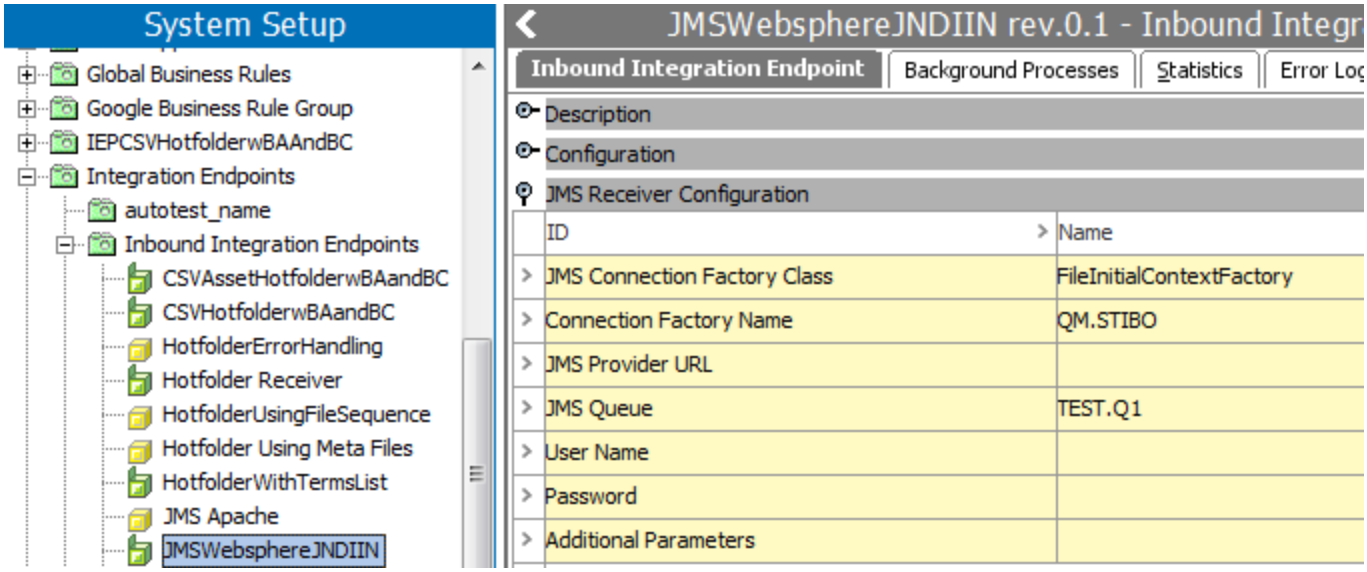
1. For **Receiver**, choose **JMS Receiver**.
2. For **JMS Connection Factory Class**, select one of the available JMS connection factory classes.
   - **ActiveMQInitialContextFactory**: Allows set up of a JMS Receiver connecting to Apache ActiveMQ. For information, search the web.
   - **FileInitialContextFactory**: Enables setting up JMS WebSphere inbound and outbound integration endpoints which reference a binding file (created from JMS WebSphere Client software). The binding file is a configuration file which includes all details of how STEP should interact with JMS WebSphere. For information about JMS WebSphere Client Software, search the web.

     Note: WMQInitialContextFactory is no longer supported, use FileInitialContextFactory instead.
3. For **Connection Factory Name**, select a connection factory name in the list.
4. For **JMS Provider URL**, select a JMS URL in the list.
5. For **JMS Queue**, select the JMS Queue to be used on Apache ActiveMQ or WebSphere MQ.
6. For **User Name** and **Password**, if required to log onto the JMS provider, enter a user name and a password.
7. For **Additional Parameters**, if required, click the **Add parameter** link, then enter the Key and Value pairs. For possible keys and values, consult the online help for IBM WebSphere MQ.
8. Click the **Next** button to continue with the **Configure Endpoint** step, or the **Finish** button to close the wizard.

Configuration Examples

The following images demonstrate the parameters and setup for a variety of JMS Receivers.

FileInitialContextFactory JMS Receiver



JMS WebSphere Delivery Using SSL

IBM Websphere MQ SSL Receiver Configuration

ID	Name
Connection URL	webspheremq-qa.stibo.com:1417/STEP.SVRCONN
Queue Manager	QM.STIBO_SSL
Queue Name	TEST.Q1
User Name	
Password	
Key Store	file:/workarea/JMSWebsphereSSLKeystore/keyStore.jks
Trust Store	file:/workarea/JMSWebsphereSSLKeystore/keyStore.jks
Cipher Suite	TLS_RSA_WITH_AES_256_CBC_SHA256
Additional Parameters	

Apache ActiveMQ JMS Receiver

Delivery Method	
JMS Delivery	
JMS Connection Factory Class	ActiveMQInitialContextFactory
Connection Factory Name	ConnectionFactory
JMS Provider URL	tcp://ATTCM3S9:61616
JMS Queue	testqueue
Binary Payload	No
User Name	
Password	
Additional Parameters	
Edit Delivery	

Kafka Receiver

Apache Kafka is an open-source distributed event-streaming data platform. The Kafka Receiver enables a STEP platform integrated with Apache Kafka to use a Kafka queue. For more information about Apache Kafka, search the web.

Prerequisites

1. Before setting up Kafka Receiver, read the **Considerations for Setting up Kafka Receiver or Delivery** topic in this documentation.
2. Prior to configuration, clicking the **Server** dropdown parameter displays the required server name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive ***Kafka.Server*** configuration property. If connecting to a cluster, use a comma-separated list.

   The following example shows two Kafka server configurations where the first server is a cluster:

   ```
   Kafka.Server.1=mybroker1a:9092,mybroker1b:9092
   Kafka.Server.2=mybroker2a:9094
   ```

3. Prior to configuration, clicking the **Topic** dropdown parameter displays the available topics. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **Kafka.Topic** configuration property. For example:

   ```
   Kafka.Topic.1=my-topic
   ```

4. OPTIONAL: Prior to configuration, clicking the **Group Id** dropdown parameter displays the available group IDs. Use this setting if you want to overwrite the auto-generated group ID. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive ***Kafka.GroupId*** configuration property. For example:

   ```
   Kafka.GroupId.1=my-group-id
   ```

5. Prior to configuration, clicking the **Keystore Location** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **Kafka.SSLKeyStoreLocation** property. For example:

   ```
   Kafka.SSLKeyStoreLocation.1=[/[path]/key_store.jks].
   ```

6. Prior to configuration, clicking the **Truststore Location** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **Kafka.SSLTrustStoreLocation** property. For example:

   ```
   Kafka.SSLTrustStoreLocation.1=[/[path]/trust_store.jks].
   ```

7. SASL / OAuth 2.0 can be configured for STEP with Kafka using the **ExtraDriverOptions** property to authenticate securely via bearer tokens. A business function allows integration with the Kafka Receiver by handling the OAuth authentication and returning a HashMap containing the Bearer Token and other details.

Configure the following case-sensitive sharedconfig.properties entries within the value of the ExtraDriverOptions as shown in the example below, which can be copied:

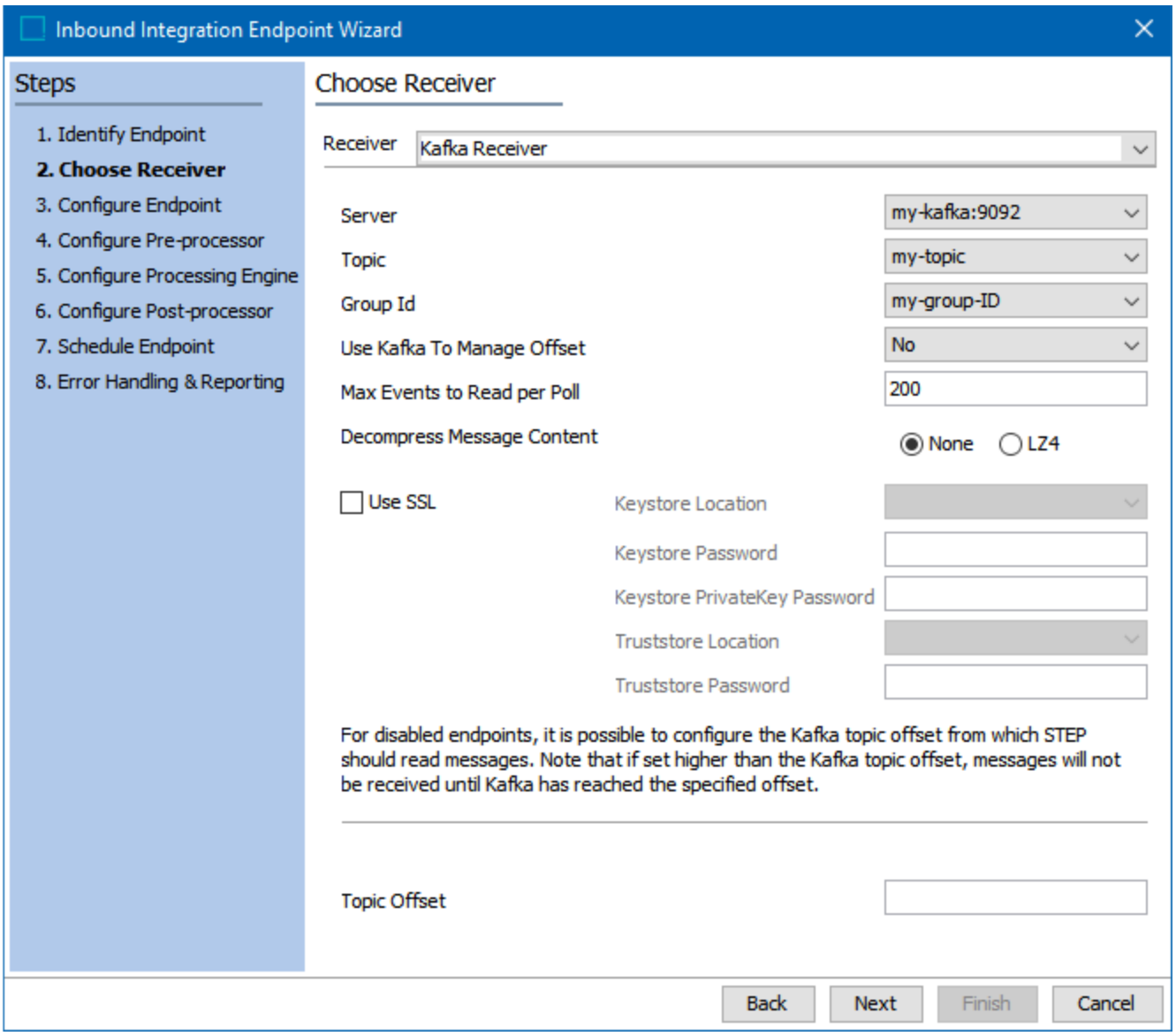
- `Kafka.Receiver.[IIEP_ID].ExtraDriverOptions` - indicate the IIEP ID.
- `stibo.authentication.function` - set the JavaScript business function ID. Refer to the **Business Function Example for Generating an OAuth Bearer Token** section in the **Kafka Delivery Method** topic.
- `security.protocol` - indicate either SASL_PLAINTEXT (while testing) or SASL_SSL (in production) for the SSL transport layer.

For example, in the online help version of this topic the property definition includes, 'MyInboundKafka' is the ID of the IIEP, 'HydraAuthFunction' is the ID of the business function, and 'SASL_PLAINTEXT' is the testing value for the security protocol and the rest of the property values are configured as displayed:

Configuration

While STEPXML is recommended for the messaging format, other formats supported for integration with STEP are allowed. The descriptions below have been written for STEPXML.

The Kafka Receiver option is available in an inbound integration endpoint (IIEP).



1. For the **Receiver**, choose **Kafka Receiver**.
2. For the **Server**, select the server(s) where the Kafka broker instances used by the endpoint are running.
3. In **Topic**, select the topic used by this endpoint.
4. In **Group Id**, select the ID to be used to overwrite the auto-generated group ID.
5. For **Use Kafka to Manage Offset**, set to 'Yes' to use the Kafka offset management mechanism instead of the STEP-owned one. This makes it easy to know when Kafka can purge messages. For more information, refer to Kafka Topic Offset on the web.

   Set to 'No' (the default) to have the offset managed completely in STEP with no commits done to Kafka.

Important: When switching from having STEP manage the offset to having Kafka do it, upon invoking the IIEP, all messages that Kafka has available for a given topic will be re-fetched. To prevent this, you can manually tell the Kafka side what the offset should be for your 'consumer group.' The offset is managed individually for each consumer group, and each Kafka IIEP in STEP is its own consumer group.

Use the `kafka-consumer-groups.sh` script to list the Kafka consumer groups and identify the one that has the name of your STEP system and the ID of your integration endpoint in its ID. Use the same script and the ID of the consumer group for your STEP IIEP to set the desired offset for your consumer group (this can only be done after the STEP IIEP has been deactivated for at least 15 minutes).

6. For **Max Events To Read Per Poll**, enter the maximum number of events to read out of Kafka per invocation.

   Note: Setting this too high can require allocating additional memory in the system.

7. In **Decompress Message Content,** optionally select a decompression option:
   - None - message content will remain compressed.
   - LZ4 - lossless data compression algorithm. Search the web for more information.
8. In **Keystore Location**, if SSL encryption is required, select an SSL encrypted connection to Kafka. Otherwise, leave this parameter blank.
9. In **Keystore Password**, enter the password for keystore if required.
10. In **Keystore PrivateKey Password,** enter the password of the private key in the keystore file, if needed.
11. In **Truststore Location**, if SSL encryption is required, select an SSL encrypted connection to the Kafka. Otherwise, leave this parameter blank.
12. In **Truststore Password**, enter the password for truststore if required.
13. In **Topic Offset**, disable the IIEP to edit the value. The Kafka offset tracks the sequential order that messages are received by Kafka topics. Enter the offset from which STEP should read messages.

    Note: When 'Use Kafka to Manage Offset' is set to 'Yes' as described in step 5 above, the Topic Offset parameter is not applicable and removed from the dialog. This is because commits happen after background processes are started for polled messages, and as these messages are consumed, STEP communicates to Kafka that the message has been consumed. Kafka then increments the offset.

14. Click the **Next** button to continue with the **IIEP - Configure Endpoint** step, or the **Finish** button to close the wizard.

Execution Report

The Kafka Receiver execution report includes the following entries:

- When the Kafka broker is running but no messages are retrieved, the number of events, the Topic, and the Topic Offset are reported.
- When the Kafka broker is not running, the IIEP fails, and a minimum number of entries are reported since no messages are being processed.

Considerations for Setting Up Kafka Receiver or Delivery

There are a number of considerations before starting the setup for Kafka integration.

- Message size limitation
- Event ordering
- Event message serialization
- Reference handling for dangling references
- SASL authentication, if needed

Each area is explained in further detail below.

When configuring Kafka, STEPXML is recommended for the messaging format. Other formats supported for integration with STEP will also work, but the documentation has been written for STEPXML.

Compatibility

Integration with Kafka for event messaging is supported via the following versions of Apache Kafka:

- 3.5.1
- 3.4x
- 3.2x
- 3.0x
- 2.6x
- 2.5x
- 2.4x

Message Size Limitation

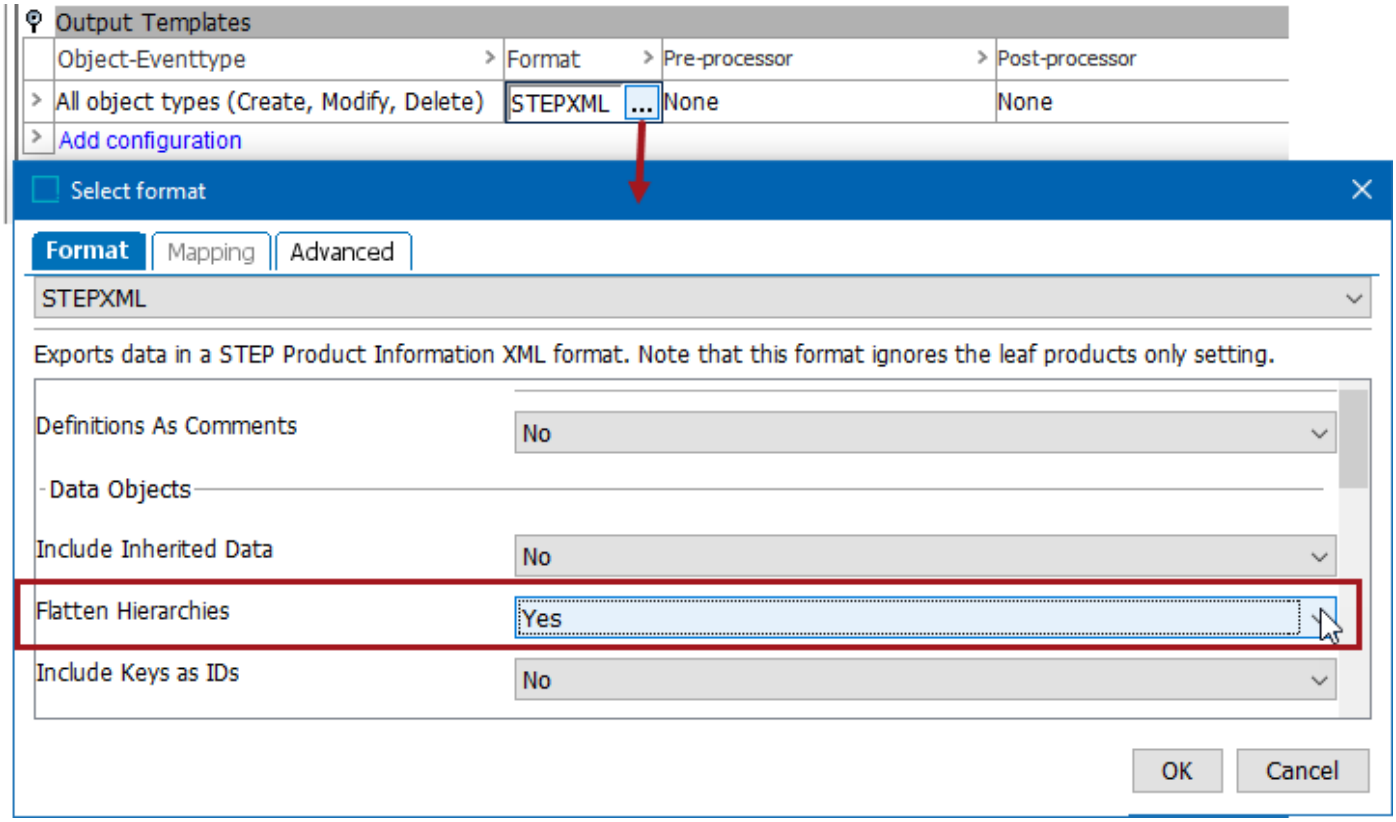
Kafka imposes a default restriction on message sizing to a max of 1 MB. To address this, there are a number of options, with Option 1 being the preferred and recommended method.

Option 1

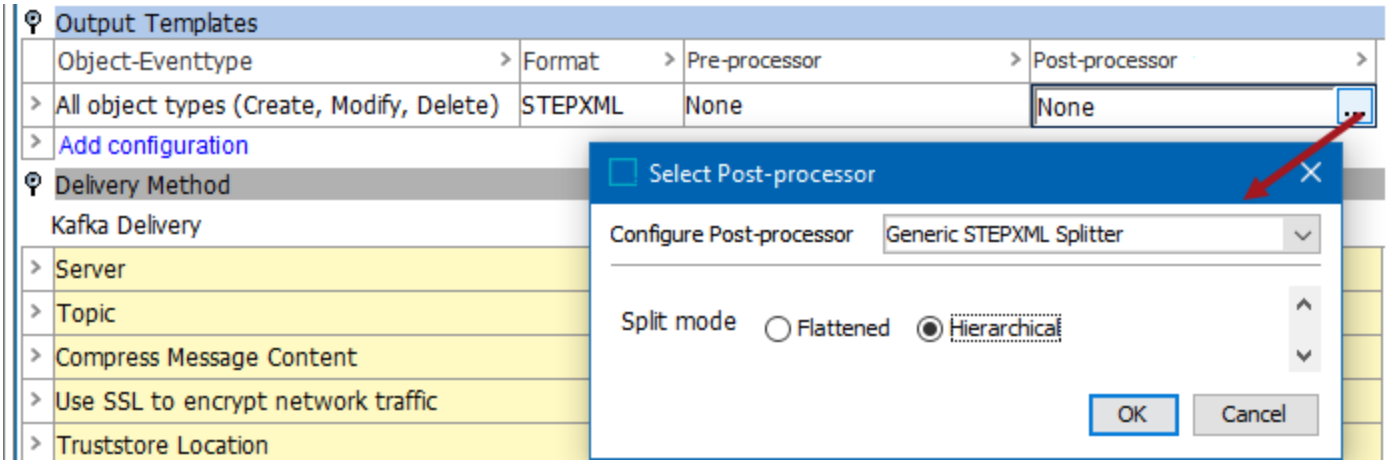
Important: This is the preferred and recommended method.

For Outbound Integration Endpoints: STEPXML format allows using inherited values.

1. On the endpoint, in the Format parameter, select STEPXML and set the 'Flatten Hierarchies' parameter to '**Yes**.'



2. For the Post-processor parameter, select **Generic STEPXML Splitter** and set the 'Split mode' to **Hierarchical**. The Generic STEPXML Splitter splits up STEPXML messages to multiple STEPXML valid fragments containing one single node per STEPXML file.

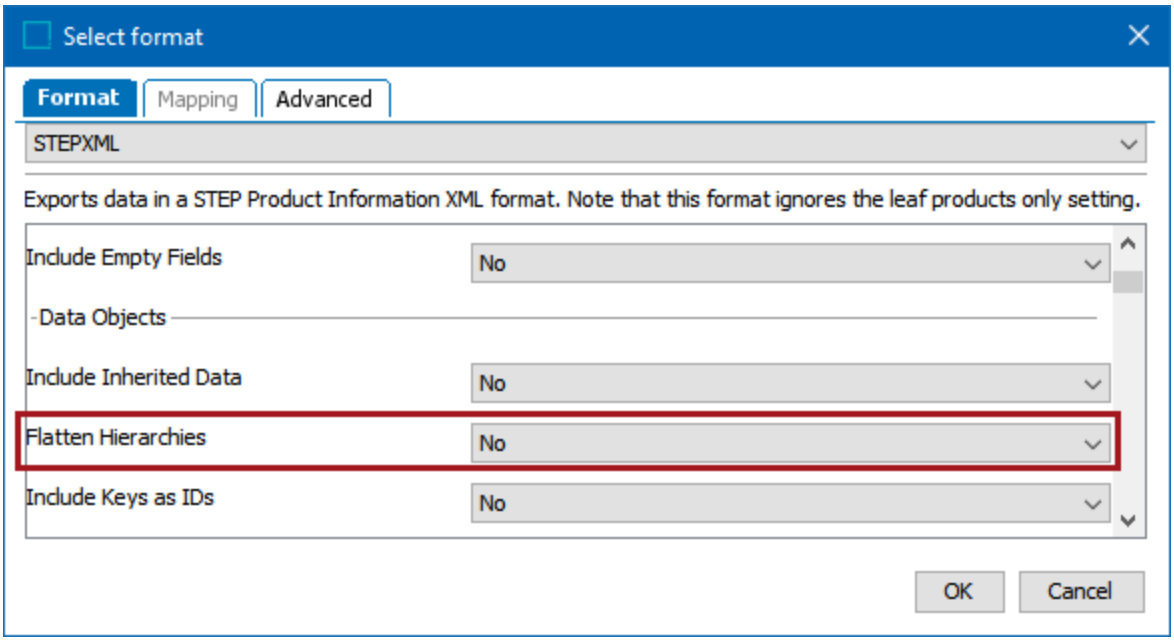


Option 2

Split up STEPXML messages to events for a single node per message and reduce the event XML size.

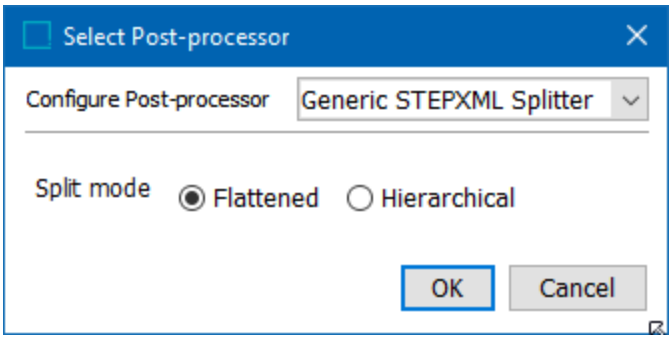
Note: Option 1 is the recommended and preferred method.

1. On the endpoint, in the Format parameter, select STEPXML and set the 'Flatten Hierarchies' parameter to '**No**.' It is not needed for this, as it will be flattened in the post-processor.



2. On the endpoint, select the **Generic STEPXML Splitter** post-processor. The Generic STEPXML Splitter splits up STEPXML messages to multiple STEPXML valid fragments containing one single node per STEPXML file.

3. Use the **Flattened** option to further reduce the size of the individual event messages. Flattened denormalizes any child hierarchies in STEPXML to be written out as a single root hierarchy.



Note: If using inherited values in the export, it is recommended to flatten the values in STEPXML format as demonstrated in Option 1 above.

For more information on post-processors, refer to the **Configure the Pre-processor and Post-processor** section of the following topics:

- **OIEP - Select Objects - Output Templates Flipper**
- **OIEP - Event-Based - Output Templates Flipper**

Option 3

Configure a compression of the event content.

While Kafka does support compression out of the box, it does not help guard against the 1 MB limit, as will be measured on uncompressed message. This option is offered as a method to further address the max message size. LZ4 is currently the only supported compression format. If configured to be used on the integration endpoint when integrating with custom developed Kafka, any producers or consumers must be able to handle compression / decompression of LZ4 event content.

Note: For performance reasons, it is not recommend to use Kafka compression if configuring this explicitly on the endpoint, as this will have limited effect in terms of size.

Important: Ensure all events are processed before changing compression settings for an existing endpoint. Existing events on the Kafka topic are not changed as a result of changing the compression setting.

Topic Partitions

STEP can only read from single-partition topics (topics created with '--partitions 1') but is capable of delivering to multi-partition topics. To have messages distributed across multiple partitions, they must have different keys. By default, the key for all messages published from STEP is the ID of the OIEP responsible. This behavior can be changed by entering a new value for the 'Kafka Message Key Template' field in the Kafka Delivery Method configuration.

The following variables can be used to produce message keys:

- **$endpointId** - ID of the OIEP publishing the message.
- **$nodeType** - The type of node being published (e.g., product, classification, etc).
- **$nodeId** - ID of the STEP node being published.

Note: Because STEP IDs are not unique, $nodeType can be combined with $nodeId to create a unique identifier.

For more information, refer to the **Kafka Delivery Method** topic in the **Creating an Outbound Integration Endpoint** section of the **Outbound Integration Endpoints** documentation.

Important: In order to publish to multi-partition topics and use the $nodeType and $nodeId variables, ensure that messages only contain data for a single STEP object. If exporting STEPXML the Generic STEPXML Splitter must be used. If using the Generic JSON format option, the OIEP must be configured to run with a batch size of '1.'

Event Message Serialization

This is relevant if implementing your own Kafka producers / consumers for integration with STEP.

1. A event produced by STEP has the outbound integration endpoint ID set as a key. When implementing your own producer, this must be the same key set per producer because the key is used for partitioning purposes.
2. The key is passed in as a string, which is serialized with the standard Kafka: org.apache.kafka.common.serialization.StringSerializer.
3. The message content itself is a bytearray that is serialized with: org.apache.kafka.common.serialization.ByteArraySerializer.
4. On the consumer side, deserialize with the opposing deserializer for key and message content. Implementations are namely: org.apache.kafka.common.serialization.StringDeserializer and org.apache.kafka.common.serialization.ByteArrayDeserializer

Reference Handling for Dangling References

The Kafka connector and the Generic STEPXML Splitter occasionally have known issues when processing references where the referred to node has not yet been created in the target system. This creates a 'dangling reference.'

Often this dangling reference can be solved by being consistent with data handling, however with an inbound receiver that is having issues with referential integrity, the following can help.

Note: This is relevant only when dealing with referential integrity issues on an inbound receiver.

Example 1

In the following example, two outbound events contain XML that is generated in the events queue.

Event 1: circularref1

```
<?xml version="1.0" encoding="UTF-8"?>
<STEP-ProductInformation ExportTime="2020-09-28 14:38:18" ExportContext="GL"
ContextID="GL" WorkspaceID="Approved" UseContextLocale="false">
<Product ID="circularref1" UserTypeID="Product" ParentID="Product hierarchy root"
Changed="true">
<Name Changed="true">circularref1</Name>
```

```
<ProductCrossReference ProductID="circularref2" Type="prodToProd" Changed="true">
</ProductCrossReference>
</Product>
</STEP-ProductInformation>
```

Event 2: circularref2

```
<?xml version="1.0" encoding="UTF-8"?>
<STEP-ProductInformation ExportTime="2020-09-28 14:38:18" ExportContext="GL"
ContextID="GL" WorkspaceID="Approved" UseContextLocale="false">
<Product ID="circularref2" UserTypeID="Product" ParentID="Product hierarchy root"
Changed="true">
<Name Changed="true">circularref2</Name>
<ProductCrossReference ProductID="circularref1" Type="prodToProd" Changed="true">
</ProductCrossReference>
</Product>
</STEP-ProductInformation>
```

For 'circularref1' and ' circularref2' to both be successfully approved upon import, partial approval is given to 'circularref1' without the reference. Then 'circularref2' is fully approved with its reference to 'circularref1'. A second approval is then given to 'circularref1' so that the reference from 'circularrelf2' can be added.

In this case, the two XMLevents would look like the following, and not cause issues with referential integrity:

```
<?xml version="1.0" encoding="UTF-8"?>
<STEP-ProductInformation ExportTime="2020-09-28 14:38:18" ExportContext="GL"
ContextID="GL" WorkspaceID="Approved" UseContextLocale="false">
<Products>
<Product ID="circularref2" UserTypeID="Product" ParentID="Product hierarchy root"
Changed="true" Referenced="true">
<ProductCrossReference ProductID="circularref1" Type="prodToProd" Changed="true">
<Product ID="circularref1" UserTypeID="Product" ParentID="Product hierarchy root"
Changed="true"></Product>
</ProductCrossReference>
</Product>
</Products>
</STEP-ProductInformation>
```

Example 2

Consider that both 'circularref1' and 'circularref2' are new products to be imported into STEP or a downstream system.

Product 1: circularref1

```
<?xml version="1.0" encoding="UTF-8"?>
<STEP-ProductInformation ExportTime="2020-09-28 14:38:18" ExportContext="GL"
ContextID="GL" WorkspaceID="Approved" UseContextLocale="false">
<Product ID="circularref1" UserTypeID="Product" ParentID="Product hierarchy root"
Changed="true">
```

```
<Name Changed="true">circularref1</Name>
<ProductCrossReference ProductID="circularref2" Type="prodToProd" Changed="true">
</ProductCrossReference>
</Product>
</STEP-ProductInformation>
```

Product 2: circularref2

```
<?xml version="1.0" encoding="UTF-8"?>
<STEP-ProductInformation ExportTime="2020-09-28 14:38:18" ExportContext="GL"
ContextID="GL" WorkspaceID="Approved" UseContextLocale="false">
<Product ID="circularref2" UserTypeID="Product" ParentID="Product hierarchy root"
Changed="true">
<Name Changed="true">circularref2</Name>
<ProductCrossReference ProductID="circularref1" Type="prodToProd" Changed="true">
</ProductCrossReference>
</Product>
</STEP-ProductInformation>
```

Notice that 'circularref1' references 'circularref2' and 'circularref2' references 'circularref1.' When using the Generic STEPXML Splitter during an inbound integration endpoint (IIEP), it would fail because the product 'circularref2' would not exist yet on the system for circularref1' to reference. The referential integrity is not being kept.

To fix this problem you can:

1. Fix this manually by creating 'circularref2' in the system, and then rerun the IIEP.
2. Remove the events from the Kafka topic using **kafka-delete-records.sh** or a similar command
3. Instead of using the Generic STEPXML Splitter, limit the batch size on the endpoint to be able to deliver more nodes in one XML event. Use LZ4 compression to reduce the message size as much as possible. In most cases, the 1 MB limit affords a larger number of events to be processed, but this is a system-specific consideration. If needed, the 1 MB limit can be raised on the Kafka cluster to allow for larger batches to be processed. For any downstream systems that may be receiving these events, make them aware of these changes. A form of eventual consistency with regards to references may need to be implemented.
4. Use advanced STEPXML and then employ embedding for the references. This option could produce more verbose XML and could exceed the 1 MB message size limit if used on a system with many references. Make sure to reconfigure the Generic STEPXML Splitter to 'Hierarchical' rather than 'Flattened' for this option. An example template that embeds the references into the event message is the following:

```
<STEP-ProductInformation ResolveInlineRefs="true" FollowOverrideSubProducts="true">
<UserTypes ExportSize="Selected"/>
<EdgeTypes/>
<CrossReferenceTypes ExportSize="Selected"/>
<Products ExportSize="Minimum">
<Product>
<Name/>
<ProductCrossReference Type="prodToProd">
```

```
<Product Referenced="true" Embedded="true">
<empty/>
</Product>
</ProductCrossReference>
</Product>
</Products>
</STEP-ProductInformation>
```

Kafka Topic Offset

To achieve transactional integrity in STEP its important to note that the Kafka topic offset is managed in STEP rather than stored in Apache Zookeeper. This way if some message processing fails in the importer, it is possible to rollback the transaction alongside the topic offset increment. In doing this, it achieves transactional integrity when reading from the Kafka topic. The offset is associated with the inbound integration endpoint (IIEP) itself. For this reason, switching an IIEP to read from another topic is not allowed, as this would inherit the offset. Instead, a new IIEP should be created from scratch and configured to read from the other Kafka topic.

In workbench, if the topic offset of an IIEP is larger than 0, topic changes are not allowed. The topic in the configuration wizard is rendered as a read-only option.

Note: If you export / import inbound integration endpoints between systems, the imported endpoint will always start from offset 0. If you restore a STEP database backup the offset for this will also be restored.

Static Membership

To avoid issues with Kafka rebalance, and to generally make reconnecting an endpoint to Kafka faster, a static membership is employed. As a result, the connection info for the IIEP on Kafka is cached for a default of up to 30 minutes. If the IIEP is scheduled to run before the 30 minutes elapse, the reconnects are a bit faster. For this reason the session timeout in the Kafka driver is set to be 30 minutes which is the default maximum on a Kafka installation. If this is configured differently, set this timeout manually from the default setting, which is 1800000 for 30 minutes in milliseconds. To set manually, navigate to the **Kafka.Receiver.<ENDPOINT-ID>.SessionTimoutInMs** configuration property.

SASL Authentication

Support for Simple Authentication and Security Layer (SASL) authentication (both PLAIN and SCRAM) is supported for the Kafka receiver and delivery plugins. Using SASL gives you more data security options and allows for alternatives to the other array of Kafka connector authentication functionality support, which includes support for AWS MSK, Heroku, and Aiven (with TLS Client Certificate Authentication).

The properties need to be added to the sharedconfig.properties file. Below is an example config of PLAIN username/password authentication:

```
Kafka.Receiver.YOURENDPOINTID.ExtraDriverOptions=sasl.mechanism=PLAIN,security.proto
col=SASL_
PLAINTEXT,sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule
required username="admin" password="admin-secret";
```

For SASL_SSL with PLAIN username/password authentication, the Keystore configuration in the SSL part of the Kafka Receiver / Delivery plugin can be omitted. If there is no requirement that the Kafka server has to trust our SSL certificate, then none is needed. A Truststore Location / Password is required to indicate that your system trusts the Kafka Servers Certificate.

Below is a sample config for SCRAM authentication:

```
Kafka.Receiver.YOURENDPOINTID.ExtraDriverOptions=sasl.mechanism=SCRAM-SHA-
256,security.protocol=SASL_
SSL,sasl.jaas.config=org.apache.kafka.common.security.scram.ScramLoginModule
required username="admin" password="admin-secret";
```

All of the configuration options are taken from confluent.io documentation on how to configure SASL authentication found in this link: https://docs.confluent.io/platform/current/kafka/overview-authentication-methods.html.

Oracle AQ Receiver

Oracle Advanced Queuing (Oracle AQ) enables messages to be exchanged between two systems.

Prerequisites

Changes to the properties file, outlined below, are implemented when the server is restarted.

1. Prior to configuration, clicking the **Connection URL** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **OracleAQReceiverConnectionURL** property. For example, two URLs are displayed with the following property setting:

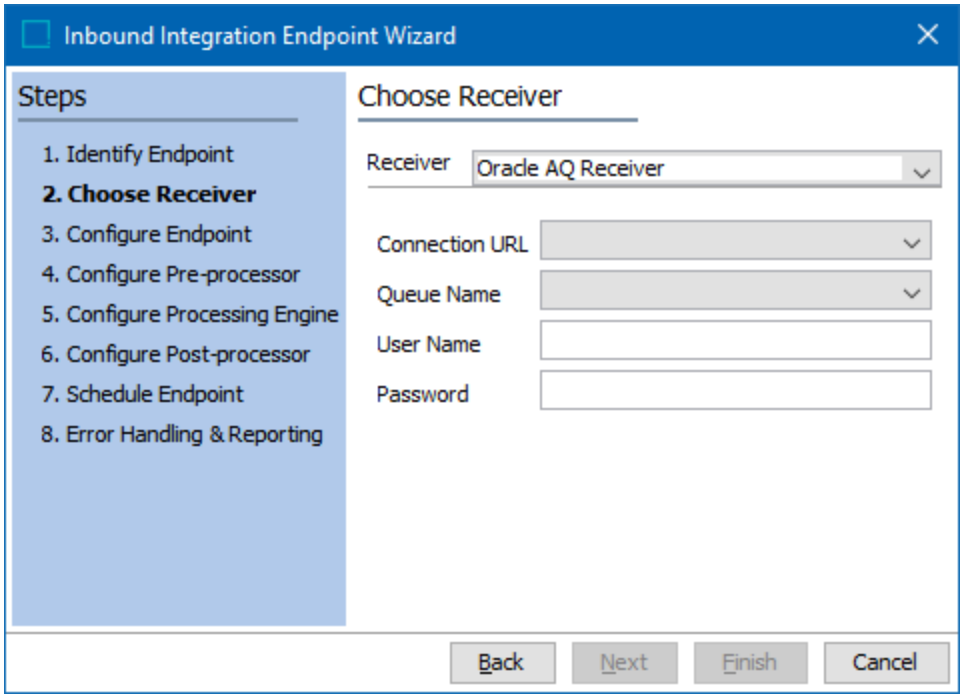
   ```
   OracleAQReceiverConnectionURL=1=ConnectionURL1,2=ConnectionURL2
   ```

2. Prior to configuration, clicking the **Queue Name** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **OracleAQReceiverQueueName** property. For example, a single URL is displayed with the following property setting:

   ```
   OracleAQReceiverQueueName=1=QueueName
   ```

Configuration

After completing the prerequisite steps, edit the receiver of the IIEP. Use the following parameters to configure the IIEP. For information on a parameter, hover over the parameter field to display help text.



1. For **Receiver**, choose **Oracle AQ Receiver**.
2. For **Connection URL**, select a URL pointing to Oracle AQ.
3. For **Queue name**, select an Oracle AQ queue name.
4. For **User Name**, enter a user name to be used to log on Oracle.
5. For **Password**, enter a password to be used to log on Oracle.
6. Click the **Next** button to continue with the Configure Endpoint step, or the **Finish** button to close the wizard.

Product Data Exchange Receiver

The Product Data Exchange Receiver option is only available in IIEPs. STEP receives data from the Product Data Exchange (PDX) platform via the default PDX Inbound Integration Endpoint and the API.

For additional information on PDX, refer to the **Product Data Exchange** topic within the **Data Integration** documentation, or contact your Stibo Systems representative.

Prerequisites

Changes to the properties file, outlined below, are implemented when the server is restarted.

1. Provide a selection for the **Server URL** parameter (image shown in the Configuration section) via the sharedconfig.properties file on the STEP application server using the case-sensitive **PDS.Url** property.

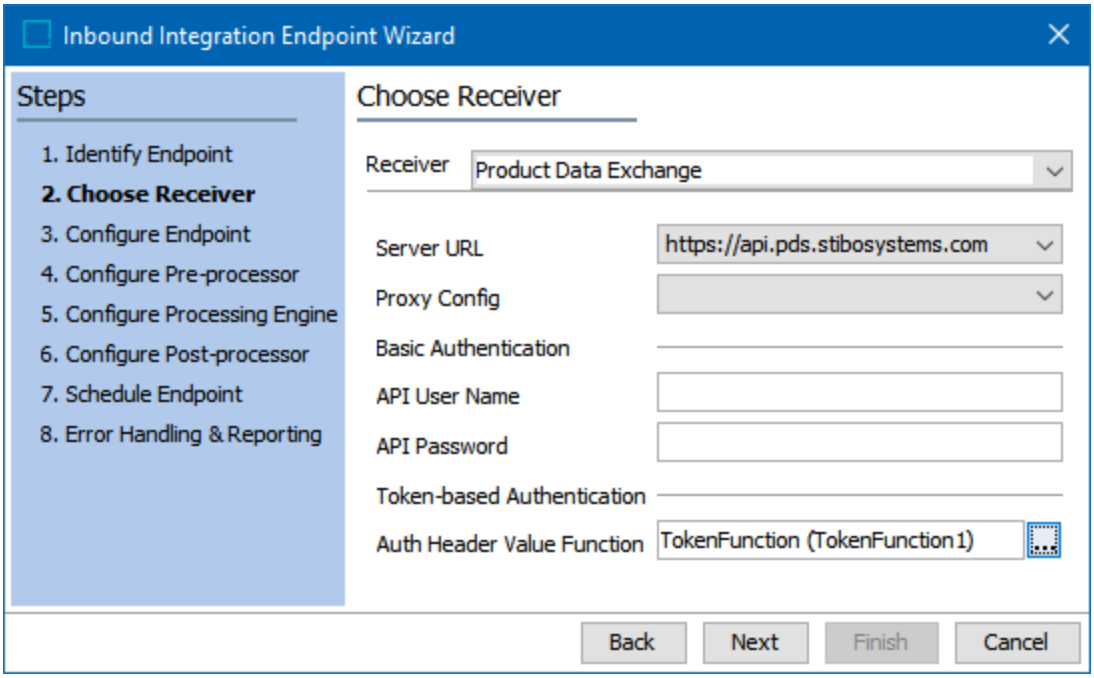
```
PDS.Url=1=https://api.pds.stibosystems.com
```

2. Determine authentication method.
3. Perform additional configuration required in the IIEP as described in the **PDX Inbound Integration Endpoint Configuration** topic.

Configuration

After completing the prerequisite steps, edit the receiver of the IIEP. Use the following parameters to configure the IIEP. For information on a parameter, hover over the parameter field to display help text.

1. For the **Receiver** parameter, choose **Product Data Exchange**.



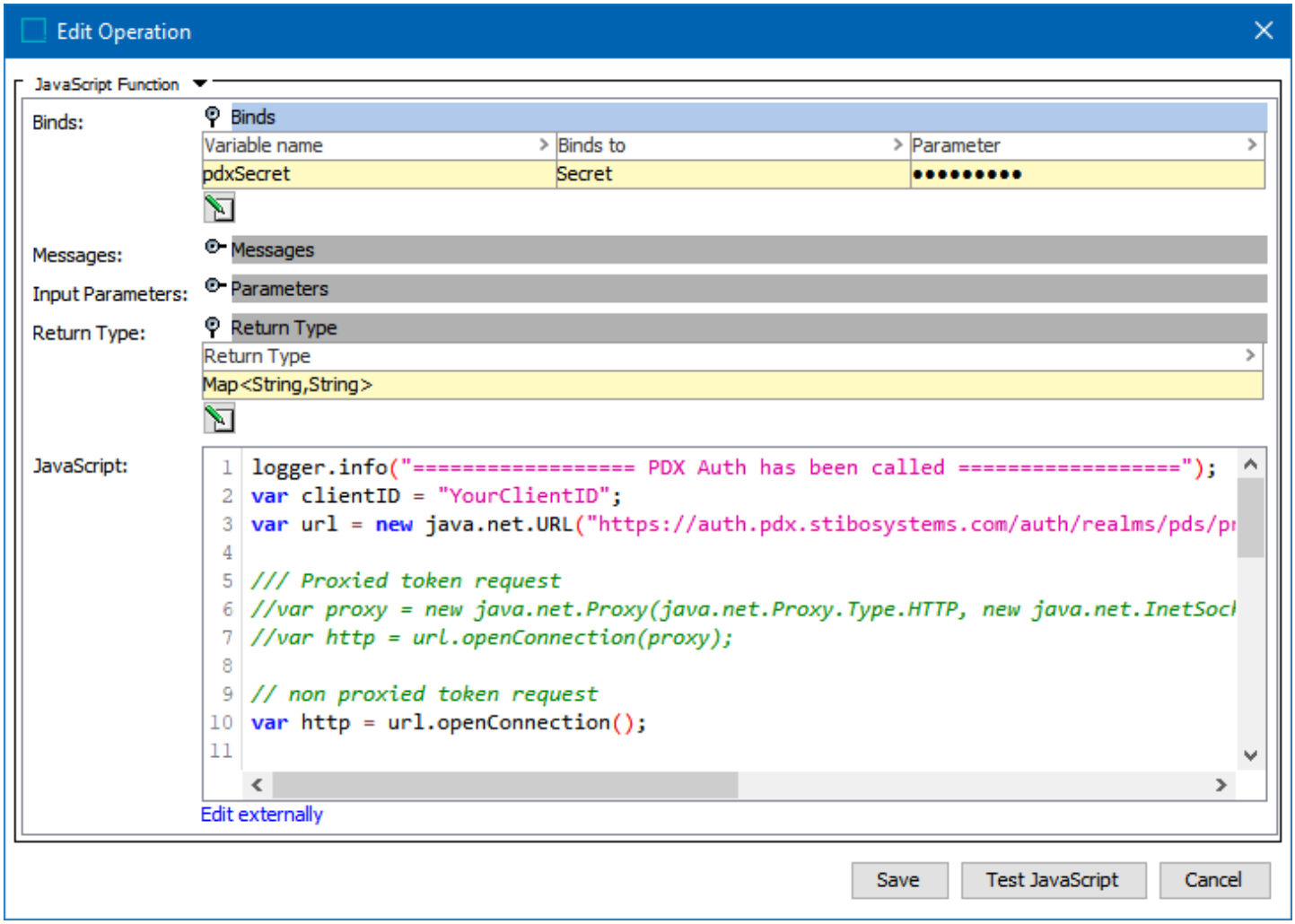
2. For the **Server URL** parameter, select the URL from the dropdown.
3. For the **Proxy Config** parameter, select the desired HTTP proxy configuration if the delivery connection must first pass through a proxy server with its own login requirement.

   Note: For more information regarding HTTP proxy configurations, refer to the topic **HTTP Proxy Configurations** in the **Data Exchange** documentation.

4. For API access via user name and password, complete the Basic Authentication section:
   - **API User Name** parameter, type the user name with access to the API.
   - For the **API Password** parameter, type the password for the user name with access to the API.
5. For token access via OAuth 2.0 authorization protocol, complete the Token-based Authentication section:
   - **Auth Header Value Function** - Select a business function that produces the required authentication headers. For general information about business functions, refer to the **Business Functions** topic in the Business Rules documentation. For examples with or without using a proxy, refer to the **Token-based Authentication Function Example** section below.
6. Click the **Next** button to continue with the Configure Endpoint step or click the **Finish** button to close the wizard.

Token-based Authentication Function Example

Use the steps below to create an example business function for token-based authentication with or without using a proxy.



1. In PDX, go to Manage team / User management / API keys and generate a key. Refer to PDX Help Center / Documentation for more information.
2. In STEP, create a JavaScript Function with:
   - Bind for secret (add variable name; Binds to = Secret; Parameter = ClientSecret (generated via PDX 'API keys' option)
   - Return Type = 'Map<String,String>'
   - JavaScript = example code below
3. Make necessary updates in the JavaScript for your system:
   - YourClientID = 'ID' of the generated API key
   - Verify the URL variable is aligned with the appropriate PDX environment (QA or PROD)

- To use a proxy, uncomment the 'Proxied token request' section and modify as required
- Add your secret Bind variable (pdxSecret is used in this example)

4. Test your integration.

```
    logger.info("================= PDX Auth has been called =================");
    var clientID = "YourClientID";
    var url = new java.net.URL
    ("https://auth.pdx.stibosystems.com/auth/realms/pds/protocol/openid-connect/token");

    /// Proxied token request
    //var proxy = new java.net.Proxy(java.net.Proxy.Type.HTTP, new java.net.InetSocketAddress
    (java.net.InetAddress.getByName("proxy.host.name"), proxyPortNumber));
    //var http = url.openConnection(proxy);

    // non proxied token request
    var http = url.openConnection();



    http.setRequestMethod("POST");

    http.setDoOutput(true);
    http.setRequestProperty("Content-Type", "application/x-www-form-urlencoded; charset=UTF-
    8");
    http.connect();

    var os = http.getOutputStream()

    try {
            os.write(new java.lang.String("grant_type=client_credentials&client_
    id="+clientID+"&client_secret="+pdxSecret).getBytes());
    } finally {

        os.close();

    }
```

```
    var input= http.getInputStream();
    try {
    var reader = new java.io.BufferedReader(new java.io.InputStreamReader(input));
    var string ="";
    while (reader.ready()){
            string=string+reader.readLine();
        }
    var json = JSON.parse(string)
    var map = new java.util.HashMap();
        map.put("Authorization",json.token_type + " " + json.access_token);
        logger.info("Authorization "+json.token_type + " " + json.access_token);
         return map;
    } finally {
        reader.close();
    }
```

REST Receiver

REST stands for Representational State Transfer. It is a software architecture style that you can use to design web services. REST as an architecture style does not require processing and is naturally more flexible. Unlike SOAP, REST does not have to use XML to provide the response. You can find REST-based Web services that output the data in Comma Separated Value (CSV), JavaScript Object Notation (JSON), and Really Simple Syndication (RSS). Supported formats include XML files using STEPXML or Generic XML.

To learn more about REST, search for resources on the web.

For information about how to use the STEP REST API, access the **STEP API Documentation**, available at [system]/sdk or from the Start Page.

Prerequisites

The path used for the file upload is system specific, and is automatically used when new REST receivers are created. In the configured path, a folder named REST is created and files that are uploaded using REST API appear in this folder. Each REST IIEP has its own 'failed' and 'save' folder.

When an IIEP has been configured to use the REST Receiver, XML files can be posted the IIEP via HTTP POST to the following URL:

```
[Host]/restapi/integrationendpoints/[Endpoint ID]/upload?context=[Existing Context]&workspace=Main
```

The request header 'Content-Type' must have the value 'text/xml' and basic authorization must be used. For example, the 'Authorization' header must have the value 'Basic ' combined with a 64 bit encoding of [Username]:[Password]. For example, 'Basic c3RlcHN5czpzdGVwc3lz.'

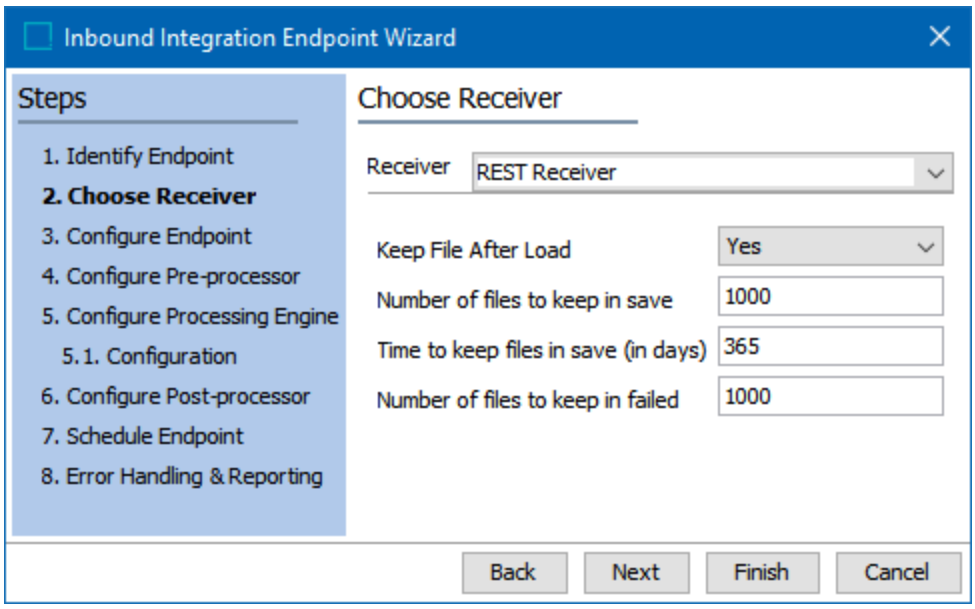
In the URL shown above, the context and workspace qualifiers have no significance, but they are required for the REST API.

The POST will in itself invoke the endpoint, so there is no reason to schedule a REST-based endpoint. The resource for the REST POST invoke request is:

```
[Host]/restapi/integrationendpoints/[Endpoint ID]/invoke
```

Configuration

After completing the prerequisite steps, edit the receiver of the IIEP. Use the following parameters to configure the IIEP. For information on a parameter, hover over the parameter field to display help text.



1. For **Receiver**, choose **REST Receiver**.
2. For **Keep file after load**, specify if files dropped in the hotfolder should be removed after processing the files. This option is available because files posted via REST are temporarily stored in a directory on the application server (under Hotfolder root).
   - **Yes** requires periodic manual cleanup on the server.
   - **No** removes the files automatically. Common setup is to select 'No' to prevent the need for manual cleanup.
3. For **Number of files to keep in save**, specifies the number of files to be saved in the save folder after automatic cleanup. Parameter can save up to 1,000 files.
4. For **Time to keep files in save (in days)**, specifies the number of days the files are saved in the save folder. Parameter can save files for up to 365 days.
5. For **Number of files to keep in failed**, specifies the number of files to be saved in the failed folder. Parameter can save up to 1,000 files.
6. Click the **Next** button to continue with the **Configure Endpoint** step, or the **Finish** button to close the wizard.

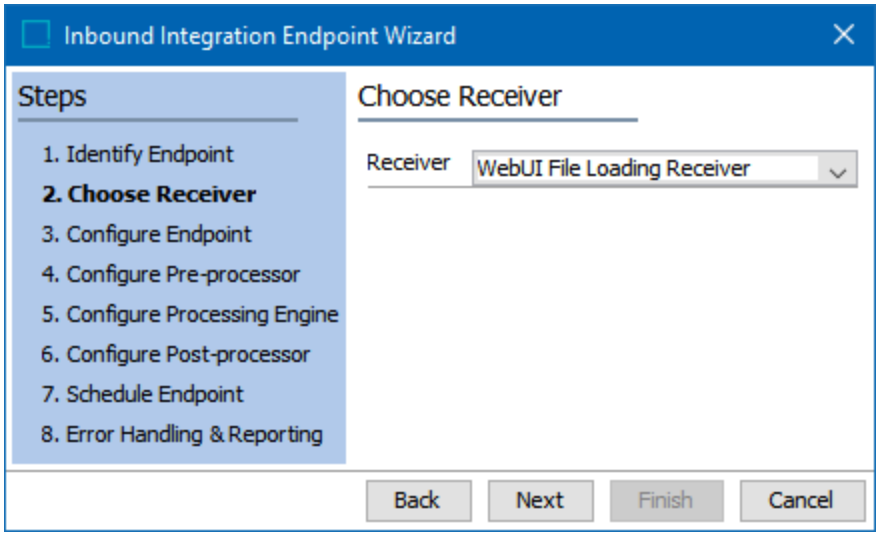
Web UI File Loading Receiver

The 'Web UI File Loading Receiver' option is used in conjunction with the Web UI 'File Loading Widget' and allows users to process files via an inbound integration endpoint that uses a receiver that is not a hotfolder.

After configuring the IIEP, log in to Web UI to configure the File Loading Widget, as defined in the **File Loading Widget** topic in the **Web User Interfaces** documentation.

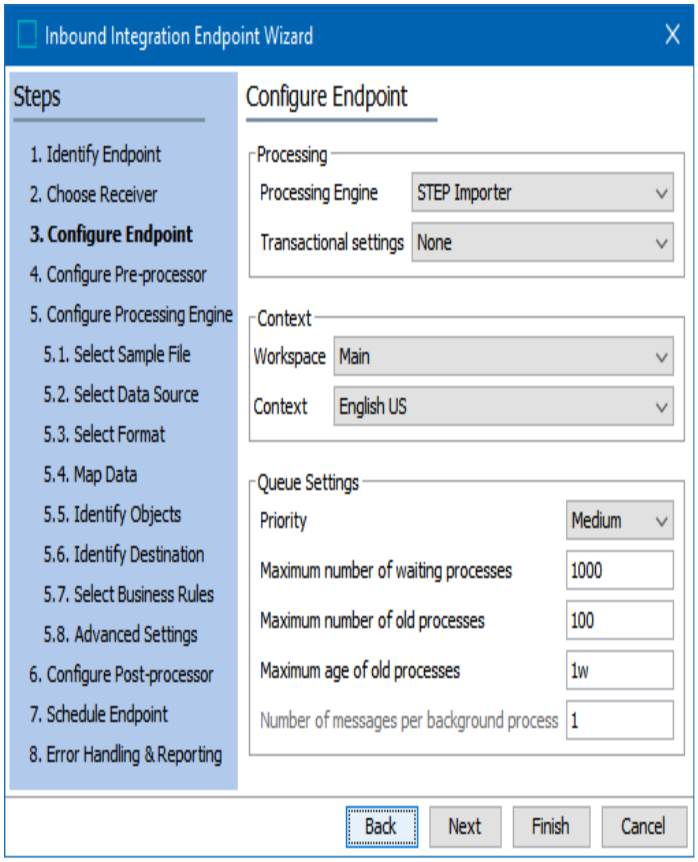
Configuration

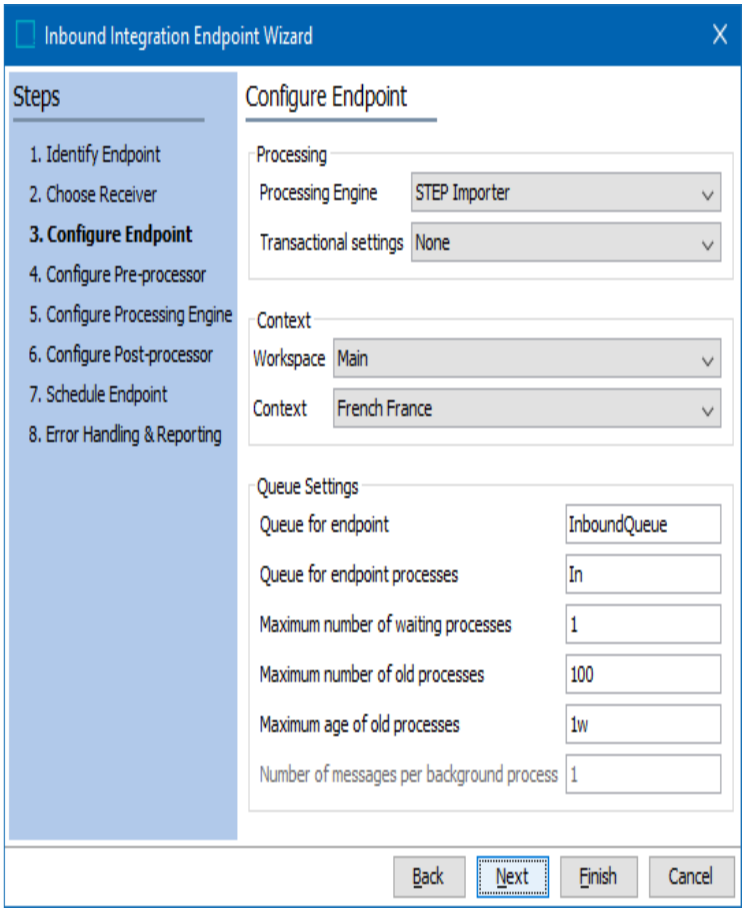
Use the following parameters to configure the IIEP.



1. For **Receiver**, choose **Web UI File Loading Receiver**.
2. Click the **Next** button to continue with the Configure Endpoint step, or the **Finish** button to close the wizard.

IIEP - Configure Endpoint





1. For **Processing Engine** on a standard STEP system, select an available option:
   - **STEP Importer** - enables you to use the same import functionality as in the Import Manager. Additional setup is required on the upcoming Configure Processing Engine step.
   - **Asset Importer** - allows you to use the same import functionality as Asset Importer. Additional setup is required on the upcoming Configure Processing Engine step.
   - **Business Rule Based Message Processor** - available when the 'business-action-processor' component is activated. Additional setup is required on the upcoming Configure Processing Engine step.
   - **FAB-DIS Importer** - available when the 'fabdis' component is activated. Additional setup is required on the upcoming Configure Processing Engine step.
   - **GDSN Inbound message processor** - available when the GDSN solution is implemented.
   - **GDSN Receiver Inbound message processor** - available when the GDSN solution is implemented.
   - **Product Data Exchange** - available when the 'productdatasyndication-integration' component is activated. No further configuration is required. For information, refer to the **Product Data Exchange** topic.
   - **STEP Match and Merge Importer** - available when the Matching and Merging deduplication solution is implemented, and requires additional configuration.

Note: If this IIEP will be used with the Web UI File Loading Widget, refer to the **File Loading Widget** topic in the **Web User Interfaces** documentation.

2. For **Transactional Settings** specify if the processes will be chain, strict, or none. For details on the effects of each option, refer to **Integration Endpoint Transactional Settings** documentation.

Note: When configuring the IIEP for the GDSN Receiver, it is recommended to set Transaction Settings to 'Strict.'

3. The queue size of a background process with strict or chain transactional setting must be 1. Only a transactional setting of None allows multiple background processes to work in parallel / concurrently. For more information, refer to **Integration Endpoint Transactional Settings** documentation.
4. For the **Context** options, select the **Workspace** and **Context** that will receive the imported data. Common setup is to use Main since it allows objects that require approval to be displayed in the Main workspace before being approved and moved to the Approved workspace. For externally maintained attributes or global maintained objects, importing to the Main workspace also adds the data to the Approved workspace, just as it does when adding data manually.
   - The selected context and workspace affects import formats where an import context and workspace is not included in the import file.
   - If the endpoint imports STEPXML files, the context and workspace specified in the STEPXML file override the context and workspace selected in this option.
5. For **Queue Settings**, set up the background processes to be used for the endpoint.
   - **Priority:** When the recommended priority-based BGP execution mechanism is configured, waiting BGPs are prioritized for execution based on the priority of the BGP and the created time. Refer to the **Priority** section of the **BG Processes Execution Management** topic in the **System Setup** documentation. The legacy 'Queue for endpoint' and legacy 'Queue for endpoint processes' parameters are not available.
   - For **Queue for endpoint**, enter a name for the queue that is used by the IIEP Background Process to poll the endpoint. This legacy option is not available when the recommended priority-based background process (BGP) execution mechanism is configured. The first time you activate the endpoint, a queue with the specified name is created if it does not already exist. If in doubt about how to populate this parameter, create a new queue for the IIEP, for example with a name that includes the IIEP ID.
   - For **Queue for endpoint processes**, enter a name for the queue that is used by the background processes started by the endpoint to handle the actual import. This legacy option is not available when the recommended priority-based background process (BGP) execution mechanism is configured. The queue is automatically created on the system if it does not already exist. High priority integrations or integrations with long-running processes should typically have their own queue, for example with a name that includes the IIEP ID.
   - For **Maximum number of waiting processes**, specify how many background processes the endpoint is allowed to start. A background process can be in the state 'waiting' or 'active.' This field is disabled if the 'Transactional settings' parameter (discussed above) is Strict.
   - For **Maximum number of old processes**, specify the maximum number of ended background processes the system is allowed to retain. When the set limitation has been exceeded, this auto-cleanup function deletes succeeded and ended background processes that were started when the IIEP was

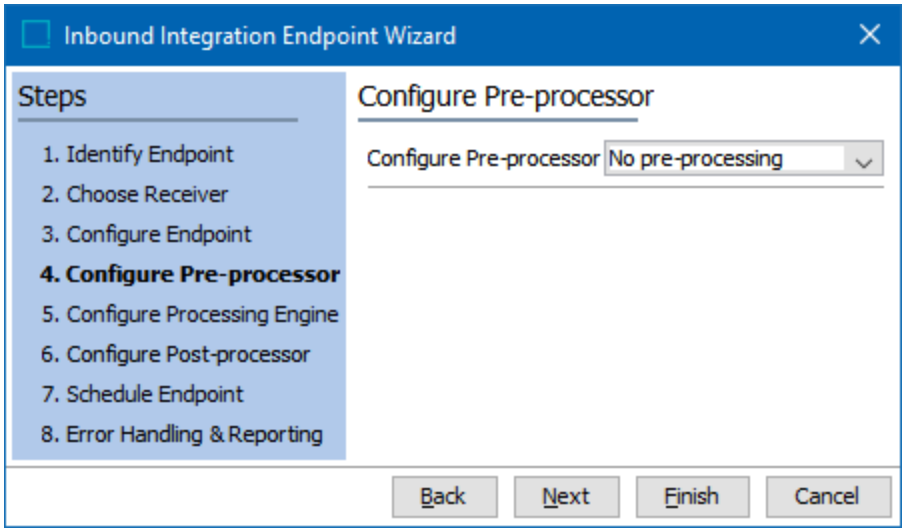
invoked. The oldest processes are deleted first. The default for this setting is 100. All background processes, regardless of their ending state (CompletedWithErrors, Aborted, Failed, Succeeded), are subject to the auto-delete functionality. Those that are in a 'Suspended' state are exempt.

- For **Maximum age of old processes**, specify the maximum age of ended background processes the system is allowed to retain. To populate this value, the following case-sensitive notations should be used: y = years, M = months, w = weeks, d = days, h=hours, m = minutes, and s = seconds. The default for this setting is 1wk. All background processes, regardless of their ending state (CompletedWithErrors, Aborted, Failed, Succeeded), are subject to the auto-delete functionality. Those that are in a 'Suspended' state are exempt.
- For **Number of messages per background process**, specify the number of messages that will be handled in one background process before a new background process is generated. This is useful if the endpoint handles many small messages that would otherwise generate a large number of background processes. Common setup is to let several messages / files be handled per background process. This field is only enabled if the transactional setting is strict.

Note: When configuring the IIEP for the GDSN Receiver, it is recommended to set the number of messages per background processes to a minimum of 100.

6. Click the **Next** button to display **IIEP - Configure Pre-processor** when available, or display **IIEP - Configure Processing Engine**.

IIEP - Configure Pre-processor



A pre-processor has access to the files / messages delivered by the receiver and can manipulate or discard them. This allows you to create extensions to STEP in order to handle your specific inbound data processing needs. For assistance in creating a pre-processor, contact Stibo Systems Support.

1. If necessary, choose a pre-processor and follow the steps in the linked topics for setup:

   - **No pre-processing** is selected by default.
   - **XML Normalizer** pre-processor is available to make minor modifications to the layout of an inbound XML file so that STEP can process it using the Generic XML format. It allows a specified tag to be copied from one place within the file to another place within the same tag. For details, refer to the **IIEP - Configure XML Normalizer Pre-processor** documentation.
   - **Transformation by XSLT Pre-processor** allows an XSLT 2.0-compliant stylesheet to be used during import to transform valid inbound XML into STEPXML, or into non-STEPXML that can then be imported using the Generic XML format option. For details on the **XSLT** pre-processor, refer to **IIEP - Configure Transformation by XSLT Pre-processor** documentation.
   - **Transformation by Import Configuration** can be used to transform Excel and CSV files to STEPXML on import.
   - **STEPXML Joiner** and STEPXML Joiner for Change is used for exporting configurations and settings via the outbound functionality. It is accessible to systems with version control system integration have access. For details, refer to the **Integration Endpoint Options for VCS Integration** topic in the **Configuration Management** documentation.

2. Click the **Next** button to display **IIEP - Configure Processing Engine**

IIEP - Configure Transformation by XSLT Pre-processor

The Transformation by XSLT pre-processor allows an XSLT 2.0-compliant stylesheet to be used during import to transform valid inbound XML into STEPXML, or into non-STEPXML that can then be imported using the Generic XML format option.

This allows the XML file to be parsed via an XSLT. The data is extracted and represented in a tabular format that can be handled with the standard Import Manager functionality.

Note: To use this pre-processor successfully, the incoming file or message size must be 50 MB or smaller.

Prerequisites

Before configuring the Transformation by XSLT Pre-processor, you must first:

1. Perform the **Initial Setup for XSLT Stylesheets** - this one-time setup is described below.
2. **Create an XSLT Stylesheet Object** - these steps must be performed for each separate stylesheet needed and is described below.

Once both of the prerequisites have been completed, proceed to the **Configure the Transformation by XSLT Pre-processor** section below.

These two configuration steps also apply to the Transformation by XSLT post-processor. For more information, refer to **OIEP - Post-processor - Transformation by XSLT** section of the **Data Exchange** documentation.

Note: Saxon is an XSLT and XQuery processor. Information can be found via links on the **Open Source Components** page at the bottom of the list of online help topics in the left navigation panel.

Initial Setup for XSLT Stylesheets

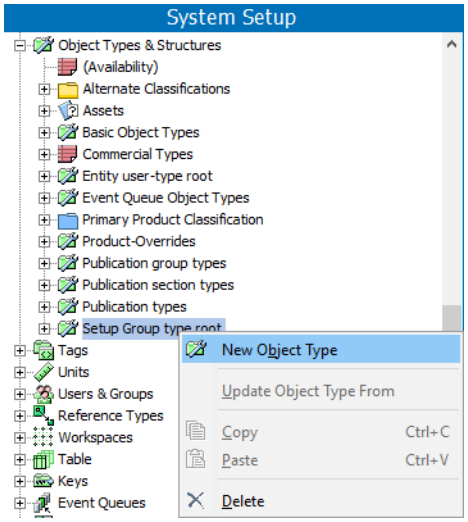
Creating an XSLT stylesheet setup group must only be performed once, unless additional levels of organization are desired.

Navigate to System Setup > Object Types & Structures > Setup Group type root, and determine if an entry exists for the XSLT Stylesheet.

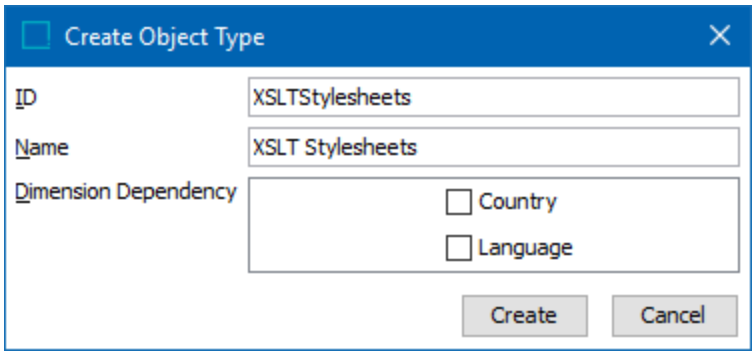
- If an XSLT Stylesheet setup group does not exist, continue with the steps below to create a setup group, link the XSLT object type, and create an instance of the XSLT object.
- If an XSLT Stylesheet setup group exists, but has no children, proceed to the **Link the XSLT Stylesheets object type to setup group** section below.
- If an XSLT Stylesheet setup group exists, and has an XSLT Stylesheet Object as a child, proceed directly to the **Creating an XSLT Stylesheet** section below.

Create the XSLT stylesheet setup group

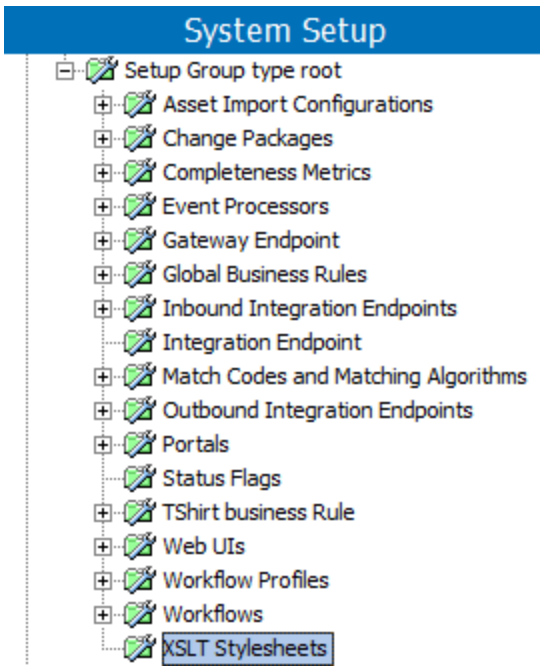
1. In System Setup, expand **Object Types & Structures**.
2. Right-click 'Setup Group type root', and choose **New Object Type**.



3. Enter an **ID** and a **Name**, select any required Dimension Dependencies, and click **Create**.

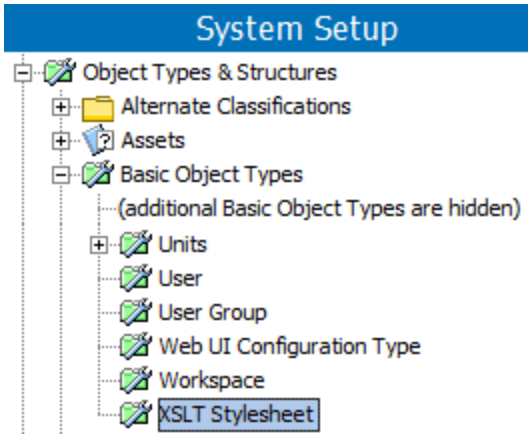


4. The new setup group appears in System Setup under 'Object Types & Structures' as a child in the **Setup Group type root**.

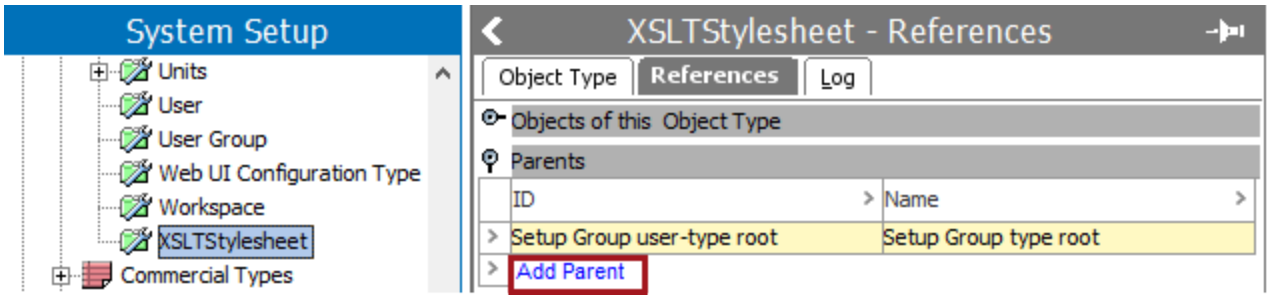


Link the XSLT stylesheet object type to the XSLT setup group

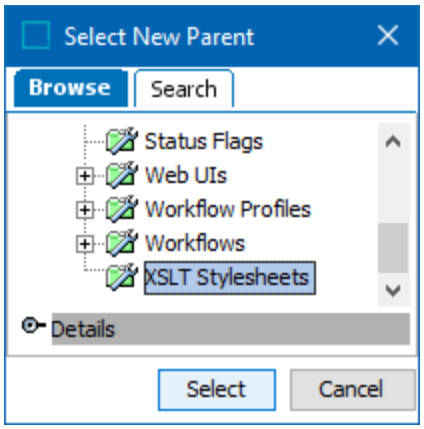
1. In Object Types & Structures > expand Basic Object Types > select **XSLT Stylesheet**.



2. On the References tab > Parents flipper > click the **Add Parent** link.



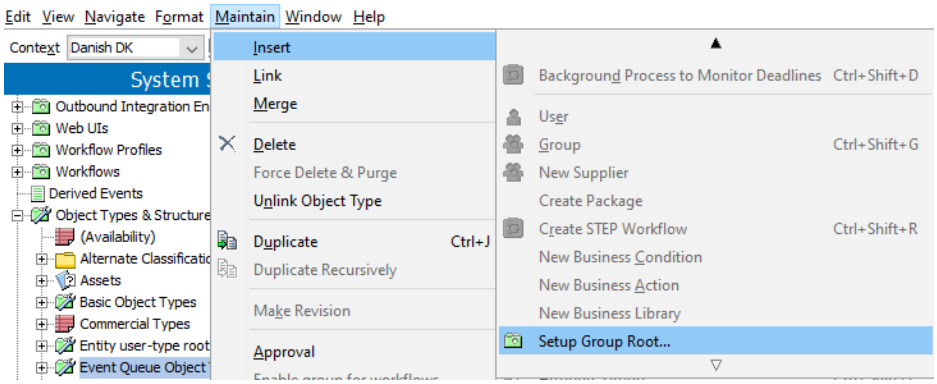
3. In the Select New Parent dialog, select the setup group type root you created, and click **Select** to make it a valid parent.



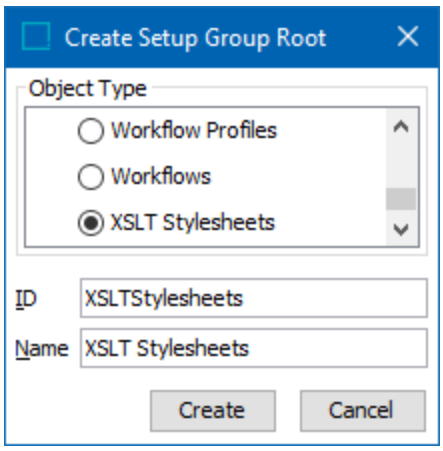
The Setup Group type root node now displays XSLT Stylesheets with a child of XSLT Stylesheet.

Create an instance of the XSLT Stylesheet object

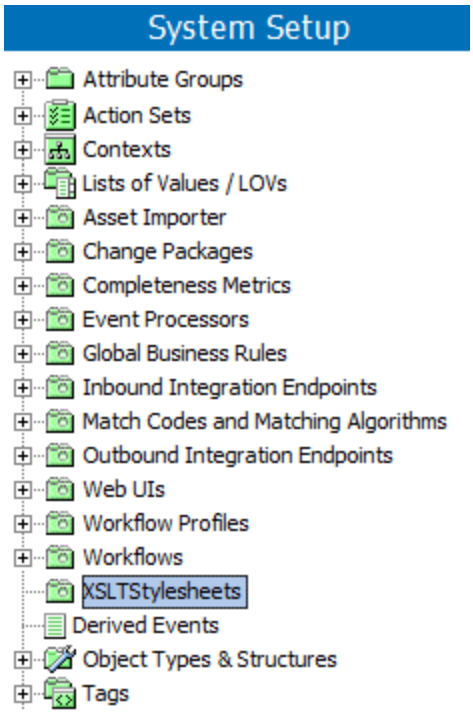
1. On the System Setup tab, select any object in the **System Setup** hierarchy to activate the following **Maintain** menu selection.
2. Click the Maintain menu, point to Insert, and select **Setup Group Root**.



3. In the Create Setup Group Root dialog, select the XSLT Stylesheets object type, enter an **ID** and a **Name**, and click **Create**.



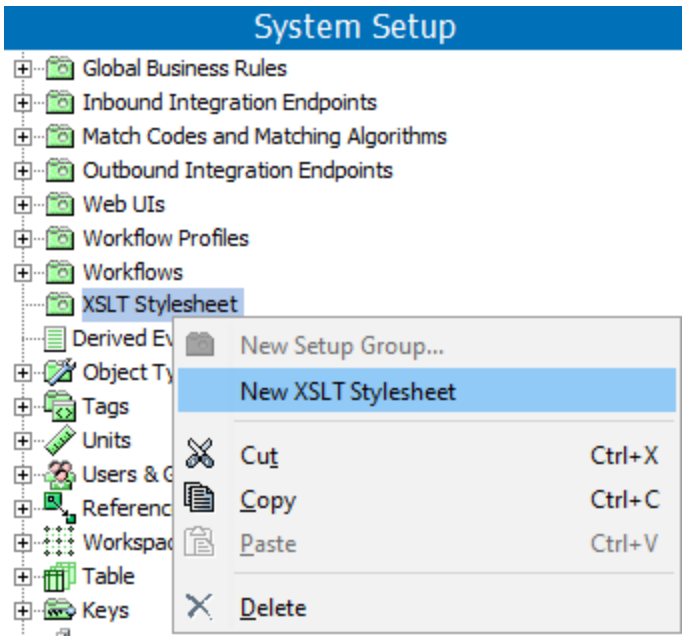
A setup group is created as a node in the System Setup hierarchy. XSLT Stylesheets can now be created under this new node.



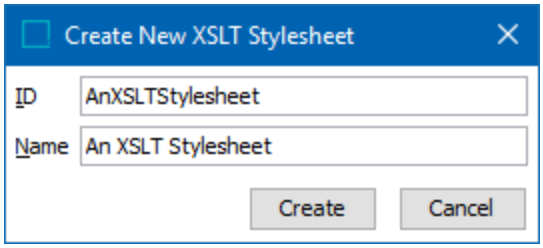
Create an XSLT Stylesheet Object

Once a setup group node has been created to hold XSLT Stylesheets, you can add your own stylesheet, which will be available for selection in the IIEP pre-processor.

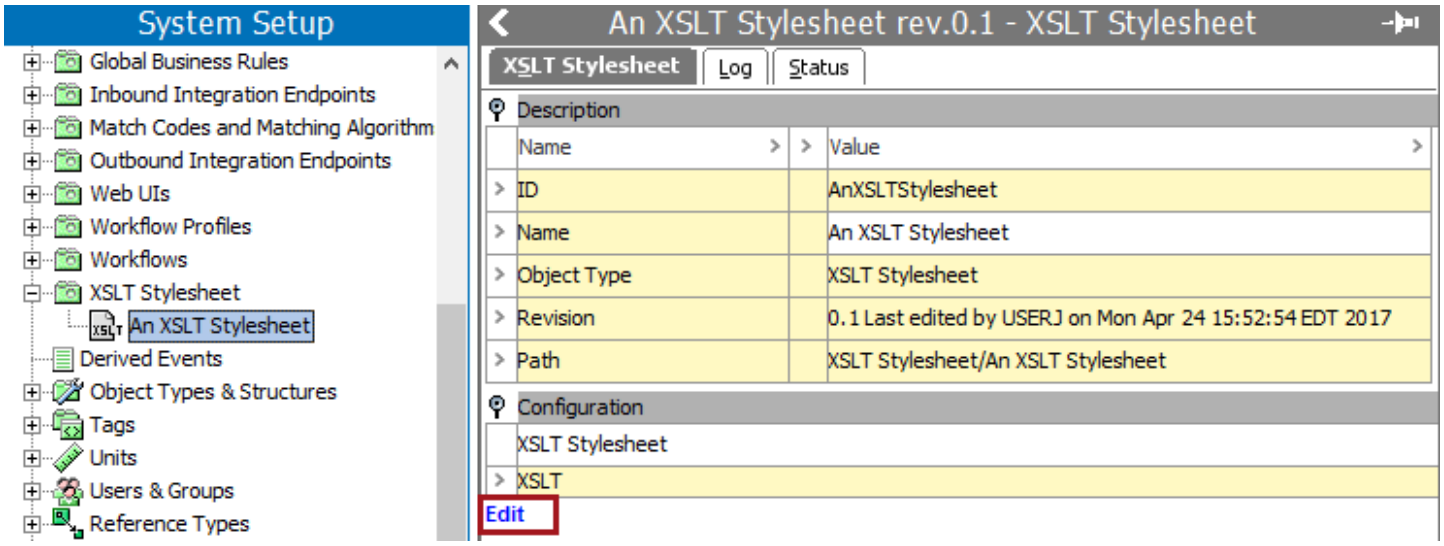
1. In System Setup, select your XSLT Stylesheet node, right-click, and choose **New XSLT Stylesheet**.



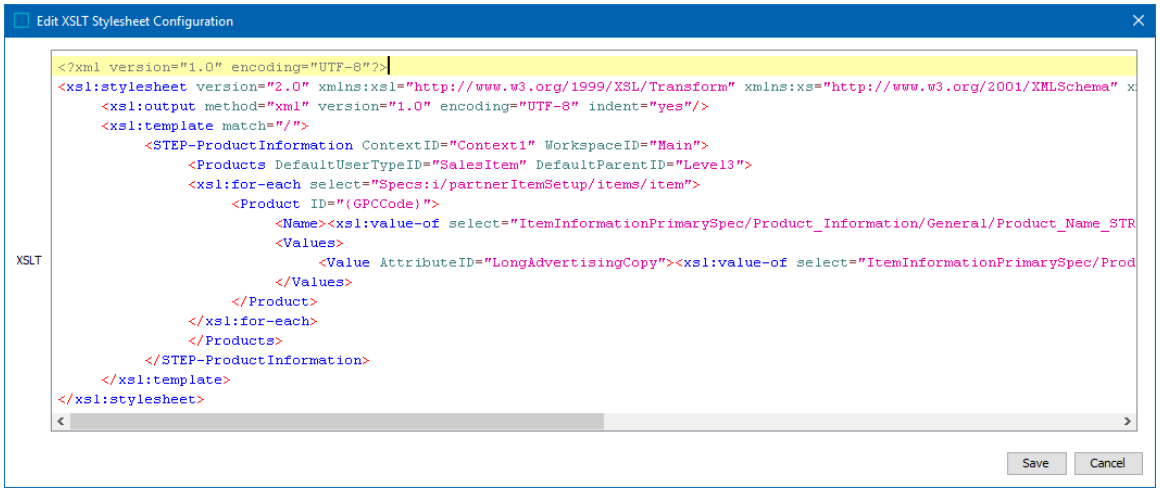
2. Enter an **ID** and a **Name**, and click **Create**.



3. Select the new stylesheet object that displays below the XSLT node and click the **Edit** link.



4. Type or copy your XSLT 2.0-compliant stylesheet into the text box, and click the **Save** button.



The XSLT stylesheet is now ready for selection in the IIEP wizard.

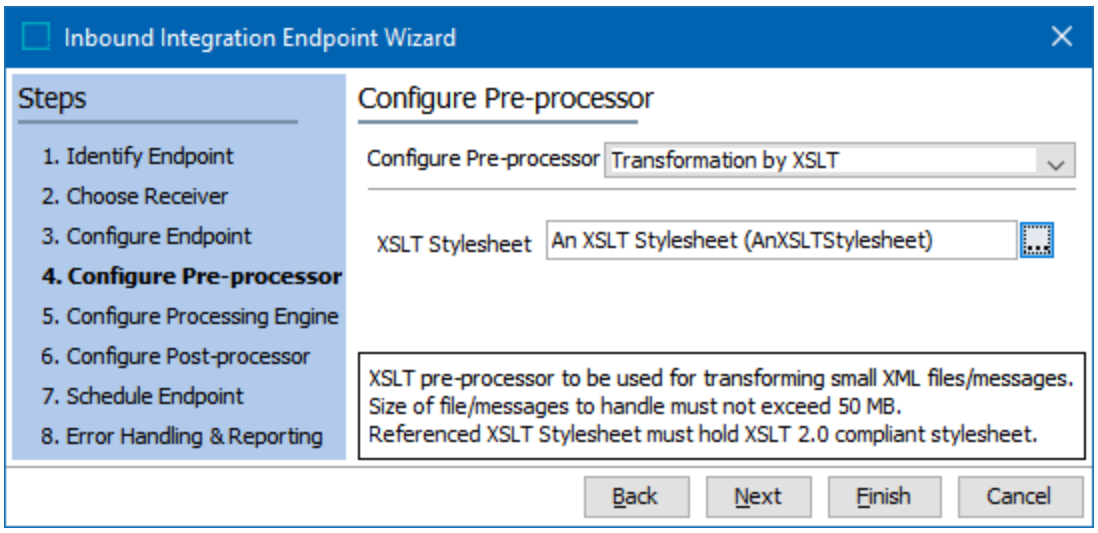
If creating an XSLT stylesheet intended to enable a 'Proof View Action' button, be sure the XSLT file uses `/webui/proofview/assets` rather than `/restapi/assets` when referencing STEP data. For example:

`"/webui/proofview/assets/{$imageURL}/thumbnail">`

Doing so prevents the system from requiring an additional login to display the requested proof view. This applies to both assets and product data fetched from either a REST API or from a proofview API.

Configure the Transformation by XSLT Pre-processor

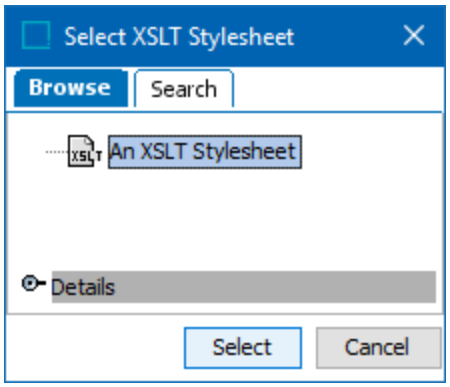
Once the prerequisite tasks have been completed, you are ready to configure the pre-processor in the IIEP wizard.



1. In the **Configure Pre-processor** dropdown, select **Transformation by XSLT**.
2. In the **XSLT Stylesheet** parameter, click the ellipsis button (...) to display the Select XSLT Stylesheet

dialog.

3. Select your stylesheet and click the **Select** button.



The importer will use the selected XSLT stylesheet to transform inbound XML file, so that STEP can process it using STEPXML or Generic XML format, based on the selection in the Configure Processing Engine step.

4. On the wizard dialog, click the **Next** button to display **IIEP - Configure Processing Engine**.

IIEP - Configure XML Normalizer Pre-processor

While this pre-processor can be used for any inbound XML file, it has been useful for GDSN solutions when processing the following scenarios:

- The GDSN BMS Receiver requires a flattened version of the recursive XML structures that are in the 'tradeItem' tags in the XML when dealing with child nodes in the packaging hierarchy.
- A GDSN CIC is sent for the top node in the packaging hierarchy and contains the CIC status for all nodes in the hierarchy.

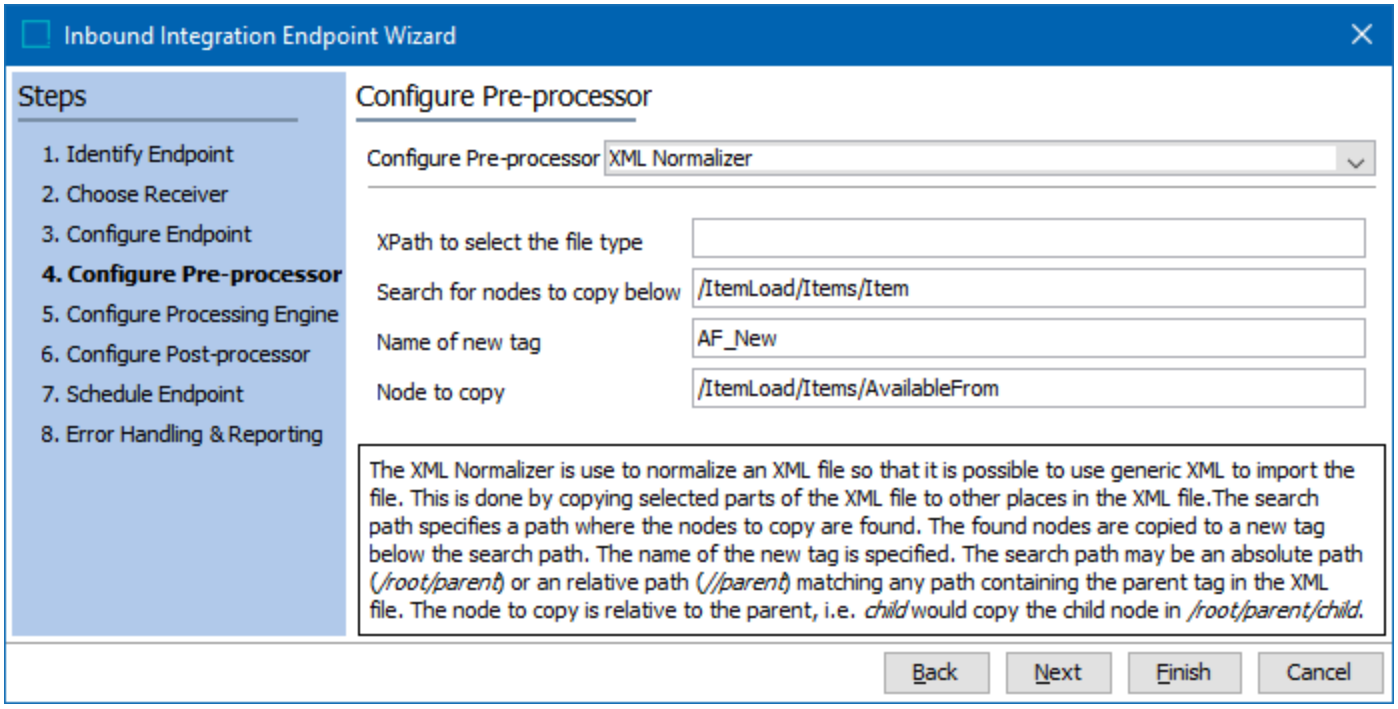
Important: For a successful import, verify that the node data is valid to exist in the new location.

For a simple example, as shown in the following XML input file (which could be available to the IIEP via the selected receiver method), although the 'AvailableFrom' tag is present in the file, it is required by STEP to be included at a lower level. The import file includes it at the 'Items' level, but it should be imported into STEP as an attribute on each of the two products within each 'Item' tag.

```
<?xml version="1.0" encoding="utf-8"?>
<ItemLoad>
  <Items>
  <AvailableFrom>2015-01-01 00:00:00</AvailableFrom>
    <Item>
      <EAN>9900524977488</EAN>
      <ProductName>AC-UZ444</ProductName>
    </Item>
    <Item>
      <EAN>9900524977490</EAN>
      <ProductName>AC-UZ311</ProductName>
    </Item>
  </Items>
</ItemLoad>
```

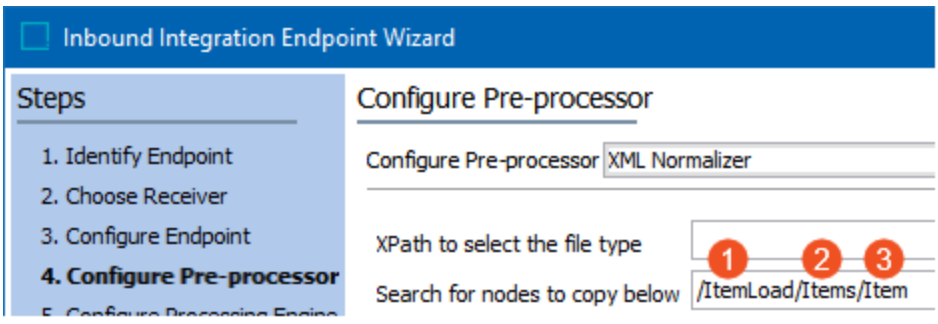
Configure and Use the XML Normalizer

1. In the **Configure Pre-processor** dropdown, select **XML Normalizer**.
2. Use the following parameters to identify the text to be copied and the new location for it to be copied.



- In **XPath to select the file type**, add an XPath if needed. This parameter is optional. For more information, search the web.
- In **Search for nodes to copy below**, identify the path of the parent tag where the copied text should be inserted. This can be an absolute path or a relative path matching any path containing the parent tag in the XML file.

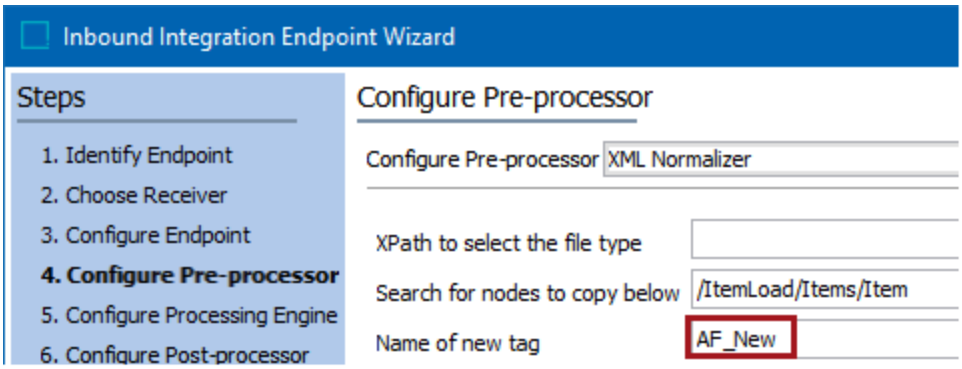
  Using the original example input file above, the path /ItemLoad/Items/Item indicates that the copied node will be placed within the 'Item' node.




```
<?xml version="1.0" encoding="utf-8"?>
<ItemLoad>
  <Items>
    <AvailableFrom>2015-01-01 00:00:00</AvailableFrom></AF_New>
    <Item>
      <EAN>9900524977488</EAN>
```

- In **Name of new tag**, type the new tag identifier. You must use different text than is currently used to additionally identify the tag that is copied. This tag must appear in the Sample file supplied in the 'Select Sample File' sub-step of the Configure Processing Engine wizard step. For more information on the sample file, refer to **IIEP - Configure STEP Importer Processing Engine** documentation.

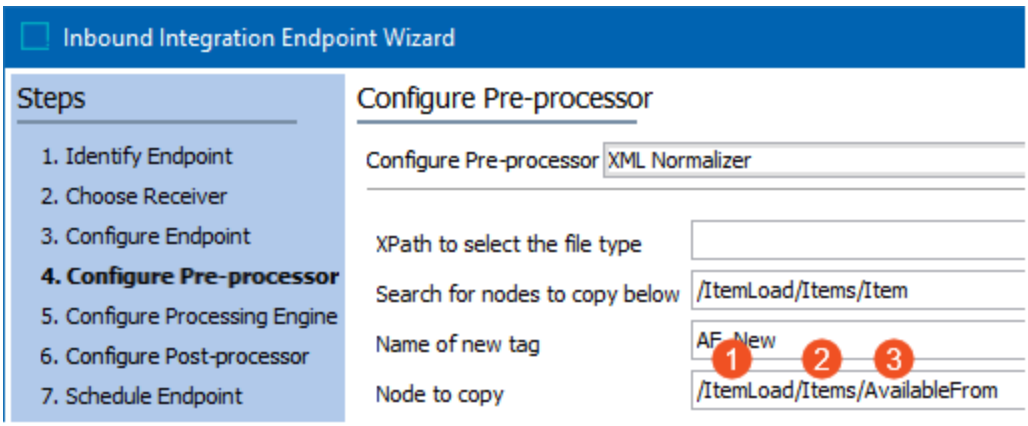
  In the wizard image, 'AF_New' is the tag that will be added as a prefix to the information being copied. The normalized XML file below shows the newly copied tag with the '<AF_New>' wrapper.




```
<?xml version="1.0" encoding="utf-8"?>
<ItemLoad>
  <Items>
    <Item>
      <EAN>2700524977488</EAN>
      <ProductName>AC-UZ444</ProductName>
      <AF_New><AvailableFrom>2015-01-02 00:00:00</AvailableFrom></AF_New>
```

- In **Node to copy**, identify the path and the tag to be copied.

  Using the original example input file, the node to copy (AvailableFrom) is currently located at /ItemLoad/Items/AvailableFrom.




```
<?xml version="1.0" encoding="utf-8"?>
<ItemLoad>
  <Items>
    <AvailableFrom>2015-01-01 00:00:00</AvailableFrom>
      <Item>
        <EAN>9900524977488</EAN>
```

3. Include the new tag in the expected location, within the sample file supplied on the **Select Sample File** sub-step. For details, refer to the **IIEP - Configure STEP Importer Processing Engine** documentation.

   Using the original example input file, you could create the following Sample File which includes the new tag in the required location (as determined by the XML Normalizer parameters) :

```
<?xml version="1.0" encoding="utf-8"?>
<ItemLoad>
  <Items>
    <Item>
      <EAN>2700524977488</EAN>
      <ProductName>AC-UZ444</ProductName>
      <AF_New><AvailableFrom>2015-01-02 00:00:00</AvailableFrom></AF_New>
    </Item>
  </Items>
</ItemLoad>
```

4. Include the new tag in the expected location, within the parsing template supplied on the **Select Format** sub-step. For details, refer to the **IIEP - Configure STEP Importer Processing Engine** documentation.

   Using with the original example, you could create the following Template which includes the instruction to retrieve data for the new tag:

```
<?xml version="1.0" encoding="utf-8"?>
<ItemLoad>
  <Items>
    <Item>
    <?Record?>
      <EAN><?Source?></EAN>
      <ProductName><?Source?></ProductName>
<AF_New><AvailableFrom><?Source?></AvailableFrom></AF_New>
    </Item>
  </Items>
</ItemLoad>
```

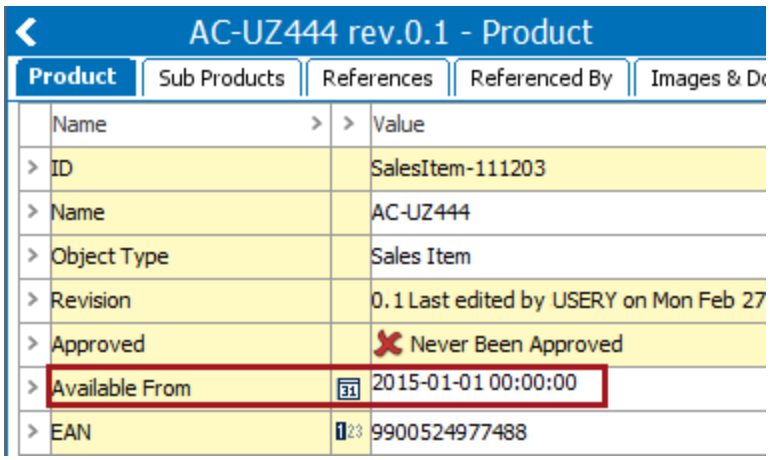
5. Click the **Next** button to display **IIEP - Configure Processing Engine**.

Pre-processor Results

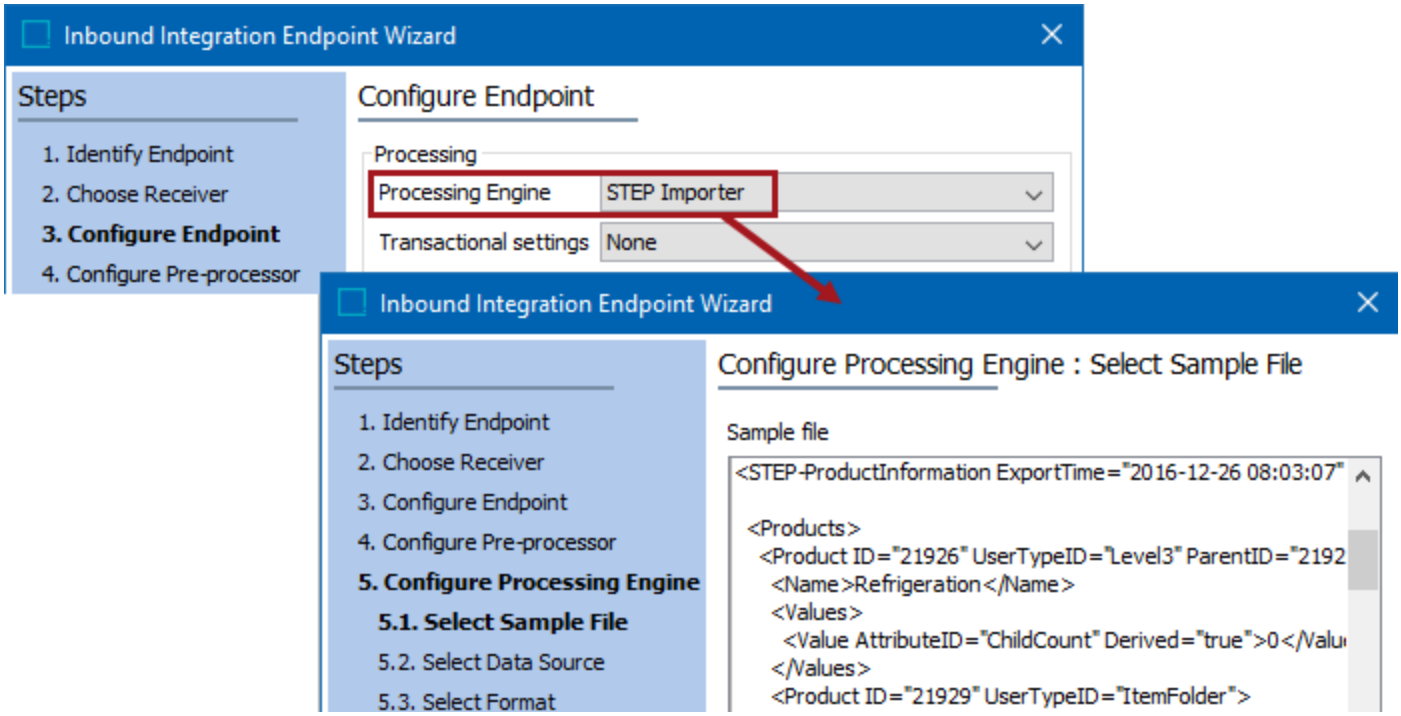
Continuing the original example, the input file is normalized by the pre-processor as follows:

```
<ItemLoad>
  <Items>
  <AvailableFrom>2015-01-01 00:00:00</AvailableFrom>
    <Item>
      <EAN>9900524977488</EAN>
      <ProductName>AC-UZ444</ProductName>
      <AF_New><AvailableFrom>2015-01-01 00:00:00</AvailableFrom></AF_New>
    </Item>
    <Item>
      <EAN>9900524977490</EAN>
      <ProductName>AC-UZ311</ProductName>
      <AF_New><AvailableFrom>2015-01-01 00:00:00</AvailableFrom></AF_New>
    </Item>
  </Items>
</ItemLoad>
```

And ultimately, the STEP attributes are updated based on the normalized Generic XML, as shown in the first product from the input file.


AC-UZ444 rev.0.1 - Product
Product
Sub Products
References
Referenced By
Images & D
Name
Value
ID
SalesItem-111203
Name
AC-UZ444
Object Type
Sales Item
Revision
0.1 Last edited by USERY on Mon Feb 27
Approved
Never Been Approved
Available From
2015-01-01 00:00:00
EAN
9900524977488

IIEP - Configure Processing Engine

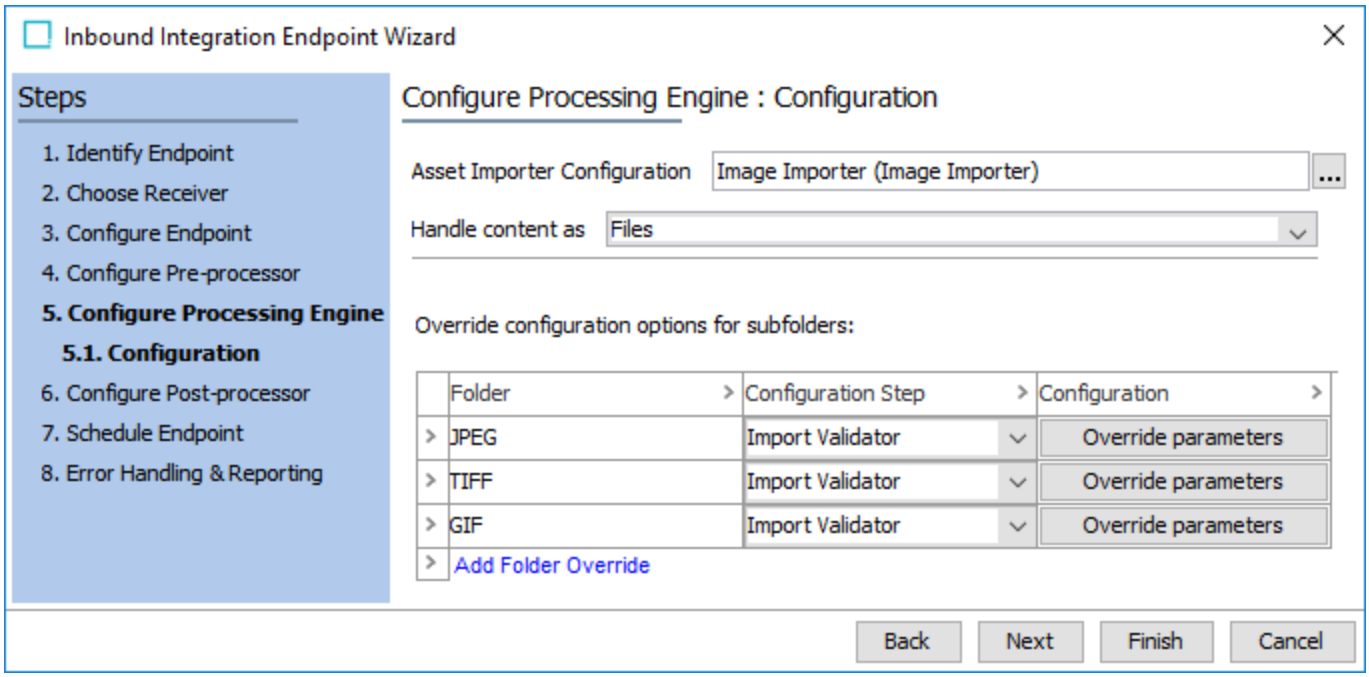


The Processing Engine is responsible for performing the actual import, and can be replaced via custom extension, for example, if some special format not supported in the standard STEP Importer is to be handled.

Important: The prior Configure Endpoint step determines the options available for the Configure Processing Engine step.

1. Select the processing engine and use the following topics to configure the necessary sub-steps. Options may include:
   - Asset Importer
   - Business Rule Based Message Processor
   - STEP Importer
   - FAB-DIS Importer
   - GDSN inbound message processor - only available when the GDSN solution is implemented
   - GDSN Receiver Inbound message processor - only available when the GDSN solution is implemented
   - Product Data Exchange - no further configuration is required. For information, refer to the **Product Data Exchange** topic.
   - STEP Match and Merge Importer - only available when the Matching & Merging deduplication solution is implemented. For information, refer to the **IIEP - Configure Match and Merge Importer** topic.
2. Click the **Next** button to display **IIEP - Configure Post-processor**when available, or display **IIEP - Schedule Endpoint**.

IIEP - Configure Asset Importer Processing Engine



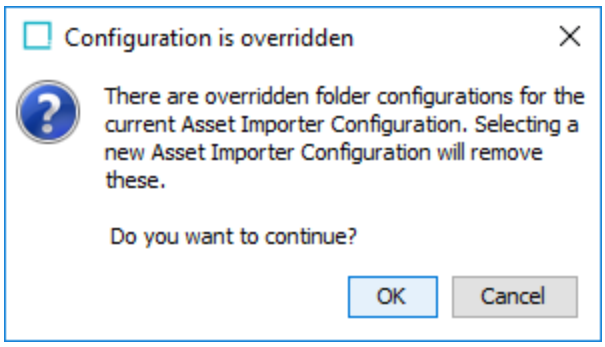
When using an Asset Importer endpoint, it is necessary to specify an asset importer configuration and define how the imported content will be received.

Prerequisites

The following settings are important when using this processing engine:

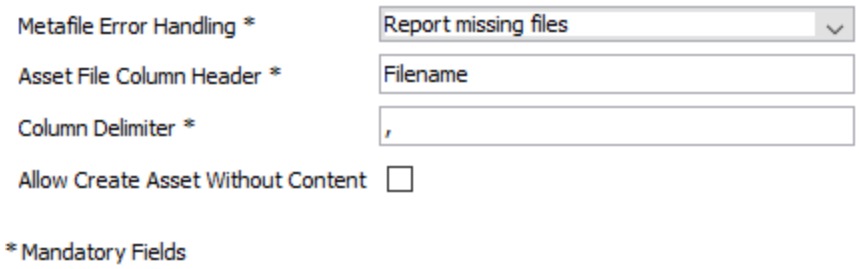
- On the Choose Receiver step, the 'Hotfolder Receiver' option must be selected as defined in the **IIEP - Choose Receiver** topic.
- If desired, folder overrides can also be configured, allowing users to replace certain aspects of the asset import configuration for sub folders within the top-level hotfolder. For required configuration, refer to the **Asset Importer Configuration** section of the **Asset Importer** documentation.

> **Note:** Note that if you attempt to change the asset importer configuration *after* configuring overrides, a prompt will appear warning you that the overrides will be removed.



Configuration

1. In the **Asset Importer Configuration** parameter, click the ellipsis button (...) and select an asset importer configuration from the node list. Click **Select**.
2. In the **Handle content as** parameter, define how files will be delivered to the asset importer by selecting the appropriate receiver option from the dropdown. The options include:
   - **Files -** This option should be used when importing single asset files into STEP. No additional parameters are required.
   - **Metafile -** This option should be used when a metadata file is necessary to supply information related to the assets, which may be the case if the assets and meta file reside in different locations. By configuring this parameter, users can point imported metadata files to related assets that are about to be imported or updated (via a path accessible by the STEP application server process). Refer to the **Metadata File Format** section below.



While importing the assets using the metafile, note the following required locations for the files:

- Assets are placed in the default hotfolder root (not the configured asset hotfolder). For example, while adding assets on the application server, the path will be `\[hotfolder for Windows system]` or `/upload/[hotfolder for Linux system]`.
- Metafile (the .txt file) is placed in the `[configured asset folder]/IN` directory.

The **Metafile Error Handling** parameter determines how the errors are handled. Options include: 'Don't report inconsistencies,' 'Report missing files,' and 'Report all inconsistencies.' The default selection, 'Report missing files', should be selected if errors should be reported when the metafile references assets which cannot be found. 'Don't report inconsistencies' should be selected if inconsistencies between the

metafile references and the assets found should not be reported as errors. This means that 'Report all inconsistencies' should be selected if any and all inconsistencies between the metafile references and the assets should be reported as errors.

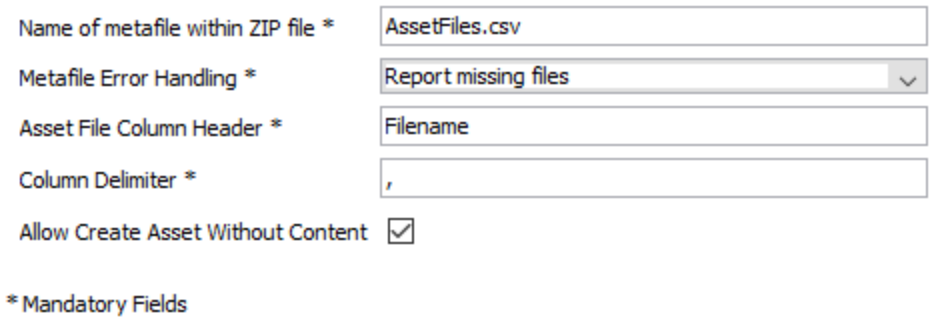
In the **Asset File Column Header** parameter, specify the column heading of the metafile that contains the asset information. This includes an asset name; an external URL; or a relative path if the corresponding asset is in a subfolder below the root hotfolder directory. For example, 'Supplier1/Asset.jpg' corresponds to the asset name 'Asset.jpg' found within the 'Supplier1' subfolder.

Note: Importing assets from URLs may increase the import time. Only freely accessible assets can be imported from URLs, entering a password is not allowed.

In the **Column Delimiter** parameter, specify which character is used to separate the columns in the metadata file. The default value is a comma (,) due to backward compatibility, but it is recommended to use either a semicolon or a pipe character, which is less likely to be included in any metadata fields.

Check the **Allow Create Asset Without Content** parameter to create an *asset placeholder* when no content is available to the importer. This can happen when STEP cannot access the asset (i.e., due to an incorrect file name or blocked access to the URL). The asset is created and referenced as configured, the thumbnail shows 'No Content' instead of an image. Update the image via subsequent imports of the same asset using the 'Asset Content Replace' checkbox in the **Content Importer**.

- **Zip File -** This option should be used when importing zip files that only contain assets. The file is unpacked during the import process and all containing asset files are imported according to the remaining configuration options. No additional parameters are required.
- **ZIP Metafile -** This option should be used when importing a .ZIP file that contains both the metadata file and the corresponding asset files. When importing with this configuration, asset placeholders are created from the metafile, and assets are subsequently loaded into STEP to update the placeholder content. Refer to the **Metadata File Format** section below.



In the **Name of the metafile within ZIP file** parameter, indicate the file name of the metafile so the system can distinguish between it and the asset files. Valid file formats are .CSV and .txt.

The **Metafile Error Handling** parameter determines how the errors are handled. Options include: 'Don't report inconsistencies', 'Report missing files', and 'Report all inconsistencies.' The default selection, 'Report missing files', should be selected if errors should be reported when the metafile references assets which cannot be found. 'Don't report inconsistencies' should be selected if inconsistencies between the metafile references and the assets found should not be reported as errors. Naturally this means that 'Report all inconsistencies' should be selected if any and all inconsistencies between the metafile references and the assets should be reported as errors.

In the **Asset File Column Header** parameter, specify the column heading of the metafile that contains the asset information. This includes an asset name; an external URL; or a relative path if the corresponding asset is in a subfolder below the root hotfolder directory. For example, 'Supplier1/Asset.jpg' corresponds to the asset name 'Asset.jpg' found within the 'Supplier1' subfolder.

> **Note:** Importing assets from URLs may increase the import time. Only freely accessible assets can be imported from URLs, entering a password is not allowed.

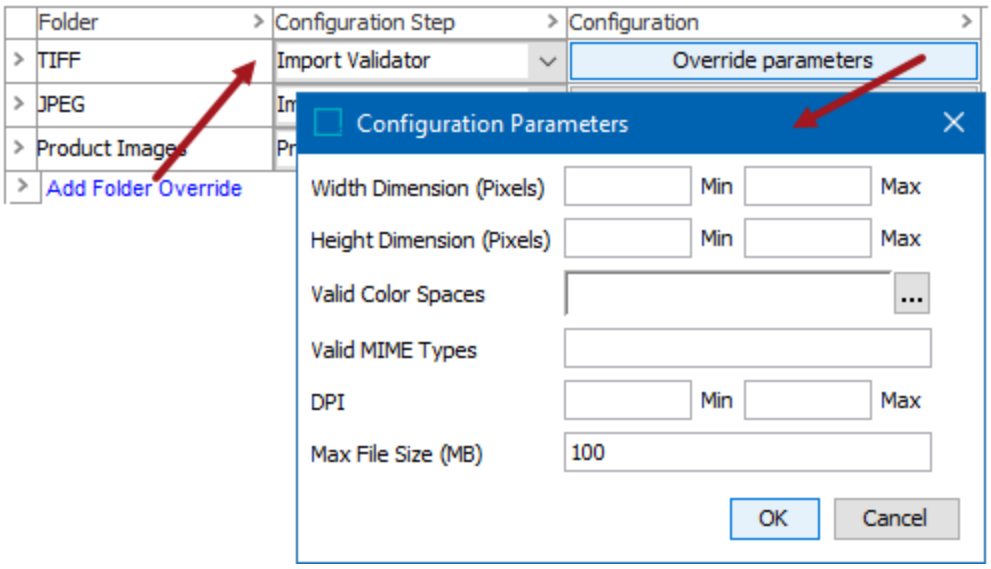
In the **Column Delimiter** parameter, specify which character should be used to separate each of the columns in the metadata file. The default value is a comma (,) due to backward compatibility, but it is recommended to use either a semicolon or a pipe character, which is less likely to be included in any metadata fields.

Check the **Allow Create Asset Without Content** parameter to create an *asset placeholder* when no content is available to the importer. This can happen when STEP cannot access the asset (i.e., due to an incorrect file name or blocked access to the URL). The asset is created and referenced as configured, the thumbnail shows 'No Content' instead of an image.

3. In the **Override configuration options for subfolders** area of the wizard, click the **Add Folder Override** link to add an override to the existing configuration. This will add a row to the override screen, in which an individual configuration aspect can be specified.

> **Note:** This option is not available when using the FAB-DIS Importer processing engine.

Override configuration options for subfolders:



- In the **Folder** field, specify a subfolder to override. If only one configuration difference is required between the parent and child folder, only a single configuration (row) is needed for each subfolder. However, if more than one configuration difference exists between the parent and child, each difference requires an individual row to be added (with the Folder name shared between multiple rows).
- In the **Configuration Step** dropdown, choose which aspect of the configuration you want to override. Almost every step in the Asset Importer configuration wizard is available for override, including: Content Importer, Import Validator, Hierarchy Builder, Asset Matcher, Product Linker, Approver, Business Rules, Metadata Importer, and Workflow Handler. The Configuration Step selection determines the dialog displayed by the Override parameters button (below).
- In the **Configuration** field, click the **Override parameters** button and make any of the necessary changes in the screen that appears and click **OK**. The parameters presented are based on the Configuration Step selection.
- Additional options are available by clicking the arrow in the first column of the override configuration options table. This displays a menu with options including Add Folder Override and Remove Folder Override.

By properly utilizing override subfolders, a single asset importer configuration can be used to control an entire hierarchy of hotfolders. All subfolders will inherit the asset importer configuration as normal, but those configured in this way can override specific aspects of the base configuration, allowing for flexibility within a single asset importer configuration.

For example, different subfolders may be associated with different MIME Types (such as JPEG, TIFF, etc.). As part of the override configuration pictured above, this use case is addressed by overriding the 'Import Validator' step of the importer configuration and changing the 'MIME Type' parameter to '/jpeg.'

4. Click the **Next** button to display **IIEP - Schedule Endpoint**.

Metadata File Format

By configuring an IIEP with the Metafile or ZIP Metafile receiver options, the asset importer is able to process user-defined metadata files. To import such a file, a header row must be included in the imported file or the metadata file to be imported, and values must be provided for each column that references each asset.

The standard metafile configuration options described in the above section allow for a maximum of three data fields to be read from the file. However, if the 'Metadata Importer' is configured, the asset importer configuration can support additional columns, each representing an asset metadata attribute value to be imported. Information included in the file but that is not mapped in the metadata importer is ignored.

For more information on the Metadata Importer, refer to the **Metadata Importer** section of the **Asset Importer** documentation.

Each of the standard fields is described below:

- **File Name Field**: Used in the Metafile and ZIP Metafile receiver selections to identify the file name of the asset. This field is required in each metafile as it correlates the actual asset files to the content in the metafile.
- **Match Metadata Field**: Used in the Metadata Product Linker configuration to specify the value for the product match criteria.
- **Reference Type Metadata Field**: Used in the Metadata Product Linker configuration to specify the STEP ID of the reference that will be used to link the asset and the product.

For more information on the Product Linker, refer to the **Product Linker** section of the **Asset Importer** documentation

Examples

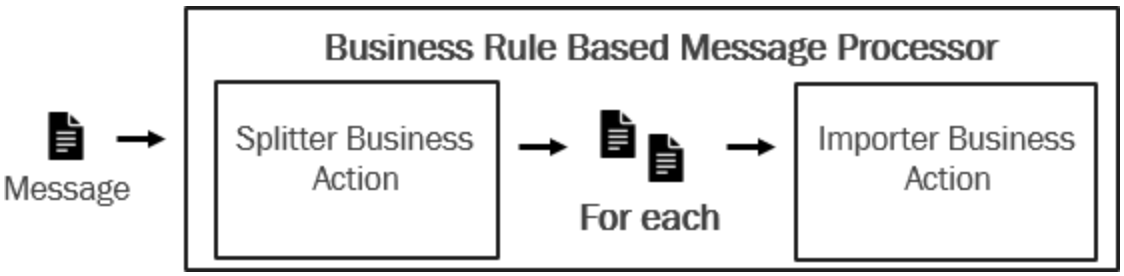
1. Including metadata info to import (Metadata Importer):
   - **Filename;Product;Reference;Title;Photographer;PhotoCategory**
   - 1234.jpg;Item-12345;PrimaryImage;Glacier Bay Alaska;Peter Parker;Landscapes
2. Using a relative path (Metafile Receiver)
   - **Filename;Product;Reference**
   - Supplier1/1234.jpg;Item-12345;PrimaryImage
3. Multiple lines of the same asset to create multiple product references
   - **Filename;Product;Reference**
   - 1234.jpg;Item-12345;PrimaryImage
   - 1234.jpg;Item-67890;PrimaryImage

IIEP - Configure Business Rule Based Message Processor Processing Engine

The Inbound Business Rule Based Message Processor allows you to process inbound messages using either JavaScript-based business actions or Java business actions developed via the Extension API.

For more information on JavaScript business actions, refer to the **Business Action Execute JavaScript** topic in the **Business Rules** documentation. For information on the Extension API, click the **STEP API Documentation** button on the STEP start page.

As illustrated below, the processor can use an optional 'splitter' business action to split the original inbound message into multiple parts that are to be handled individually, and invokes the 'importer' business action for each part, or for the original message when no splitter is selected.



The importer business action is responsible for updating STEP based on information in the provided messages via the API functionality exposed in the Scripting / Extension API. A new transaction is started for each importer invocation and, via the configuration, it is possible to specify if processing of subsequent messages should be performed when an exception is thrown from the importer business action code. If desired, the business action code author can design the error handling strategy so that some errors are treated as warnings, while others cause the transaction to be rolled back and the processing potentially stopped.

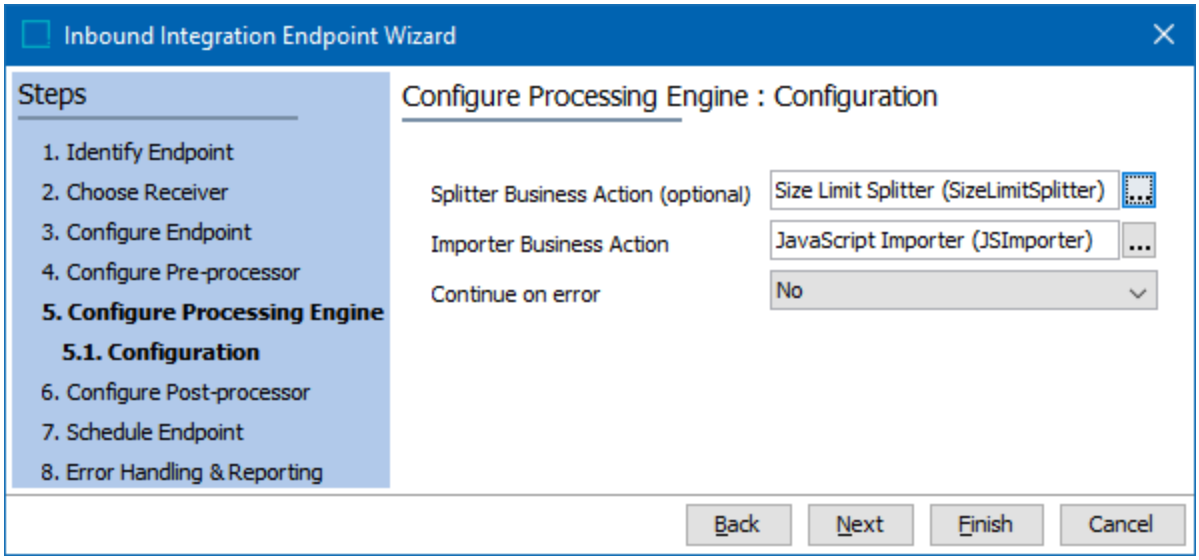
Important: Given the flexibility of this functionality, it is easy to write importer business action code that will not perform well. Generally, avoid long-running queries, extensive traversal of hierarchies, and business actions holding large data structures in memory. Imports via the Business Rule Based Message Processor functionality cannot be expected to perform as well as the standard STEP Importer processing engine. For high load imports of formats not supported by the STEP Importer, an alternative is to transform incoming files / messages to a supported format in an IIEP pre-processor.

Prerequisites

1. Contact your Stibo Systems account manager or partner manager to activate the **Business Action Processor** commercial license. This enables the event-based importer and exporter to process inbound messages in custom / generic formats using JavaScript-based business rules, and enables exports to custom / generic formats including JSON using JavaScript business rules.
2. Business actions used by this processor should be configured to be valid for all object types.
3. Review the **Splitter Business Action Details and Examples** and **Importer Business Action Details and Examples** sections below for more information about the business actions expected for this processor.

Configuration

To display the configuration for this processor, you must first select the Business Rule Based Message Processor on the previous Configure Endpoint step (defined in the **IIEP - Configure Endpoint** topic).



Set the following parameters as necessary.

1. **Splitter Business Action (optional)** - This optional parameter defines a business action to split the incoming file / message into separate messages for individual import. For details, refer to the **Splitter Business Action Details and Examples** section below.

   Click the ellipsis button (...) and select a business action from the list. Click **Select**.

2. **Importer Business Action** - This required parameter defines a business action used to update STEP based on either the original incoming file / message (if no splitter is configured) or based on messages produced by the selected splitter business action. For details, refer to the **Importer Business Action Details and Examples** section below.

   A new transaction is started for each invocation. If an exception is thrown during the processing of a message, all changes in STEP performed by the action during the specific invocation are reverted.

   Click the ellipsis button (...) and select a business action from the list. Click **Select**.

3. **Continue on error** - Determines if subsequent (split) messages should be processed when an exception is thrown from the importer business action during processing of a message.

   Select 'Yes' or 'No' from the dropdown.

4. Click the **Next** button to display **IIEP - Configure Post-processor** when available, or **IIEP - Schedule Endpoint**.

Splitter Business Action Details and Examples

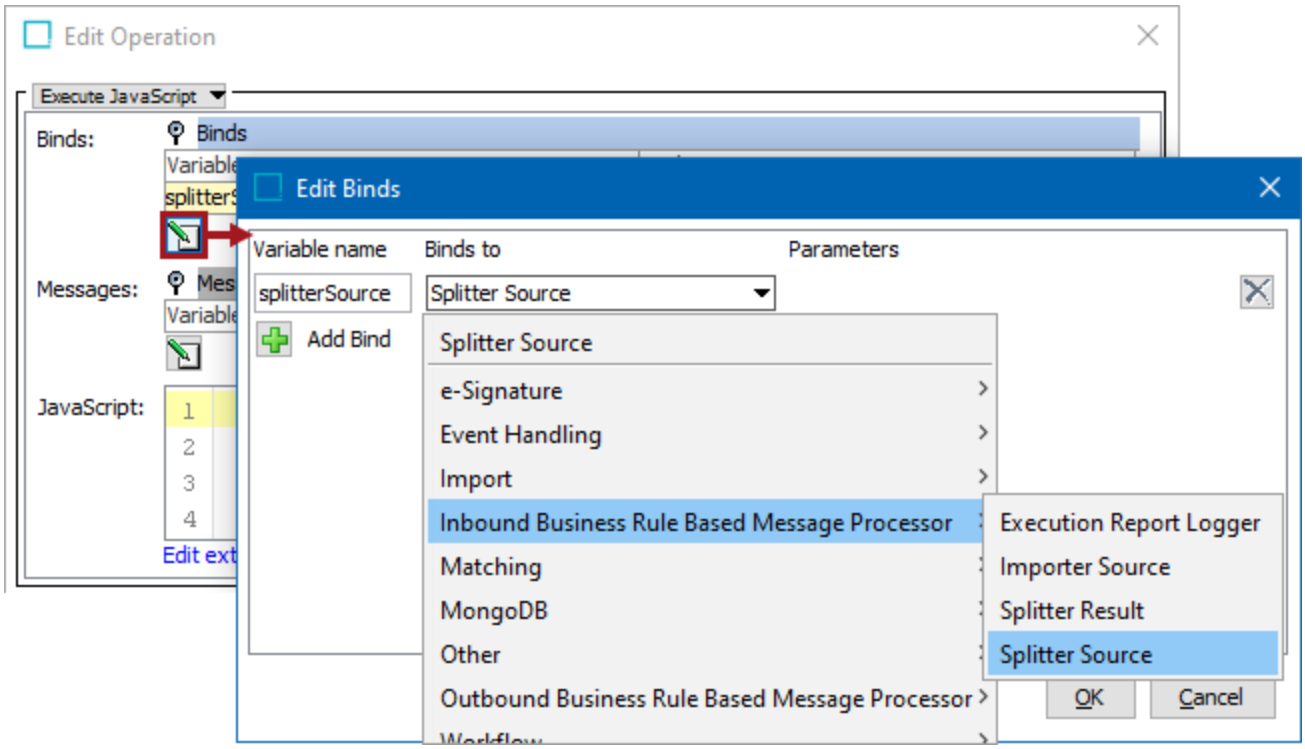
The **InboundBusinessProcessorSplitterSource** interface (for JavaScript business actions, available via the Inbound Splitter Source bind option) provides access to the inbound file / message for the splitter business action.

The InboundBusinessProcessorSplitterSource interface offers these methods for reading the file / message:

- `InboundBusinessProcessorSplitterSource.lines()` reads the file line by line and no size restrictions are enforced.
- `InboundBusinessProcessorSplitterSource.getContentAsString()` reads the entire content as a string and messages may not exceed 20MB.

The action delivers split messages via the **InboundBusinessProcessorSplitterResult** interface `addMessage(String)` method that can be called multiple times (for JavaScript business actions, the interface is available via the Inbound Splitter Result bind option).

In addition, the importer business action has access to the standard Logger for logging to the main step log and the **InboundBusinessProcessorExecutionReportLogger** interface for logging messages to the background process execution report. Notice that the splitter is not executed in a transaction, meaning that it is not possible to modify STEP data from the splitter.



JSON Message Splitter with JavaScript Business Action Example

This example shows how a simple JSON message in JavaScript can be read into memory, parsed to a JavaScript object, and split into multiple messages.

Sample file / message content

```
{
  "products": [
    {
      "ean": "4905004972222",
      "shortDescription": "Fender AM 60 Jazzmaster",
      "manufacturerName": "Fender",
      "color": "Green",
      "category": "AlternativeDesignGuitars"
    },
    {
      "ean": "4905004972234",
      "shortDescription": "Gibson Les Paul Custom EB GH",
      "manufacturerName": "Gibson",
      "color": "Black",
      "category": "SingleCutGuitars"
    }
  ]
}
```

Sample JavaScript

```
// Splitter Source bound to splitterSource
// Splitter Result bound to splitterResult

var mesg = JSON.parse(splitterSource.getContentAsString());

if (mesg.products) {
  mesg.products.forEach(function(element) {
    splitterResult.addMessage(JSON.stringify(element));
  });
}
```

CSV Message Splitter with JavaScript Business Action Example

This example shows how a CSV message can be split into one message per line. As opposed to the previous example, the entire message will not be read into memory.

Sample file / message content

```
4905004972222|Fender AM 60 Jazzmaster|Fender|Green|AlternativeDesignGuitars
4905004972234|Gibson Les Paul Custom EB GH|Gibson|Black|SingleCutGuitars
```

Sample JavaScript

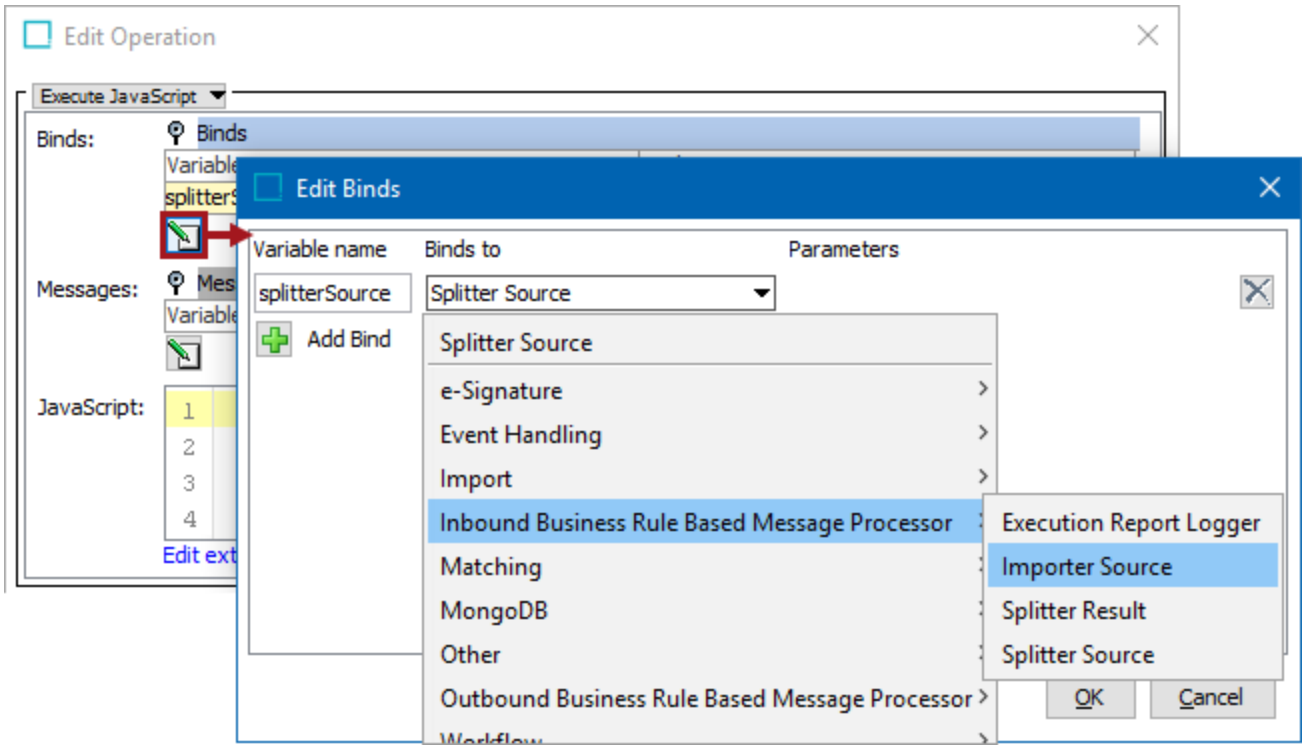
```
// Splitter source bound to splitterSource
// Splitter Result bound to splitterResult

var ite = splitterSource.lines();
while (ite.hasNext()) {
```

```
  splitterResult.addMessage(ite.next());

}
```

Importer Business Action Details and Examples

The importer business action is invoked for each message produced by the splitter or once for the inbound message in case no splitter is selected. The importer business action has access to the message to process via the **InboundBusinessProcessorImporterSource** interface getMessage(): String method (for JavaScript business actions, the interface is available via the Inbound Importer Source bind option). Further, the action has access to a standard Logger, a Manager, an **InboundBusinessProcessorExecutionReportLogger** and for JavaScript business actions, 'static binds' to attributes, reference types, etc., can be used.

Important: Take care to ensure that messages passed to the importer action do not exceed 1MB. An exception is thrown if `getMessage()` is called for larger messages.



JSON Message Importer with JavaScript Business Action Example

This example shows how STEP can be updated based on an inbound message. Notice how some errors cause an exception to be thrown (and the transaction to be rolled back) while others only cause a warning to be logged in the background process execution report.

Note: This script uses keys to identify products in STEP. Lookup by ID or key is always preferred over queries.

Sample message content

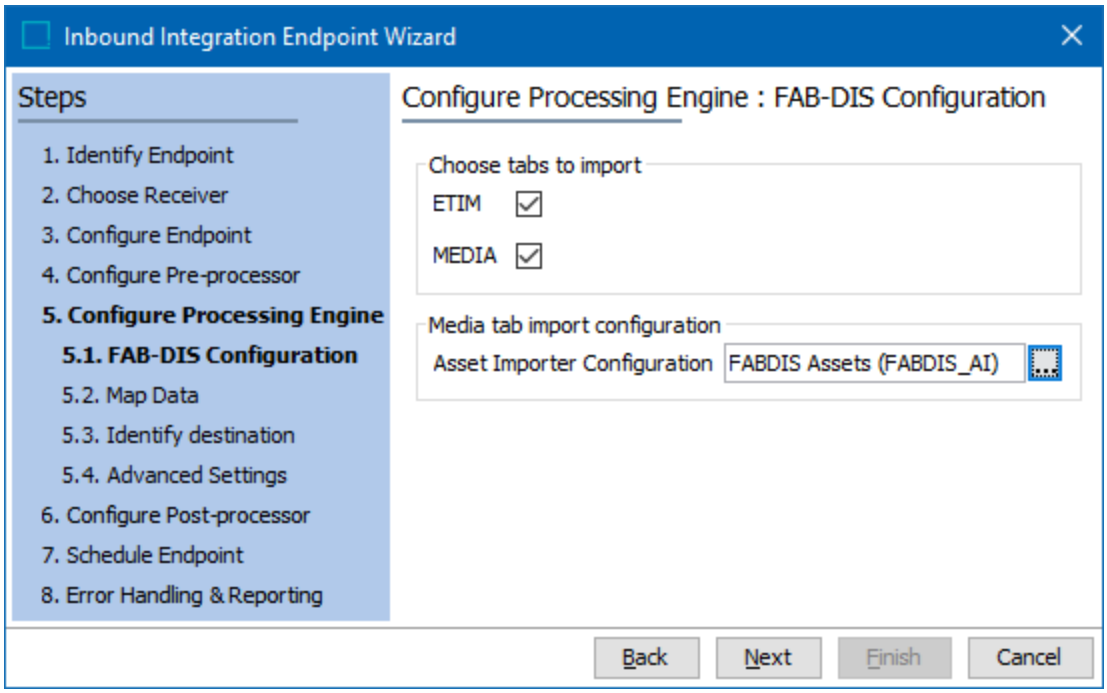
```
{
  "ean": "4905004972222",
  "shortDescription": "Fender AM 60 Jazzmaster",
  "manufacturerName": "Fender",
  "color": "Green",
  "category": "AlternativeDesignGuitars"
}
```

Sample script

```
// InboundMessage bound to inboundMessage
// Manager bound to manager
// ExecutionReportLogger bound to executionReportLogger

function getCategory(category) {
  if (!category) {
    throw new java.lang.RuntimeException("No Category specified. This is a required
field.");
  }
  var categoryProduct = manager.getProductHome().getProductByID(category);
  if (!categoryProduct) {
    throw new java.lang.RuntimeException("No Category with ID '" + category + "'.");
  }
  return categoryProduct;
}

function setValue(product, attributeId, value, isMandatory) {
  try {
    product.getValue(attributeId).setSimpleValue(value);
  } catch (e) {
    if (e.javaException instanceof com.stibo.core.domain.ValidatorException) {
      var message = "Could not set value '" + value + "' for attribute with ID '" +
attributeId + "' on product with ID '" + product.getID() + "': " +
e.javaException.getMessage();
      if (isMandatory) {
        throw new java.lang.RuntimeException(message);
      } else {
        executionReportLogger.logWarning(message);
      }
    } else {
      throw(e);
    }
```

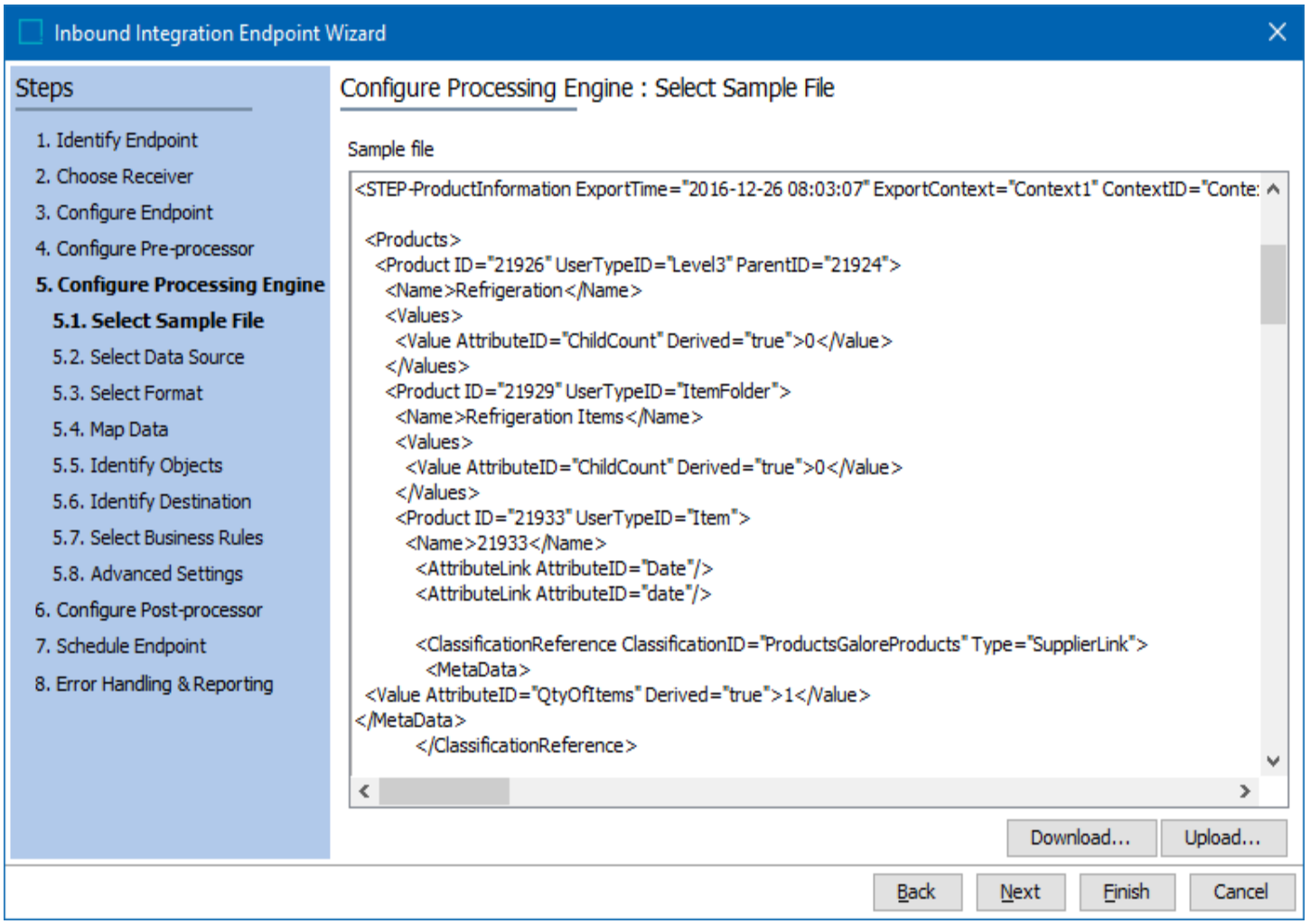
```
  }
}

var prodMessage = JSON.parse(inboundMessage.getMessage());
var ean = prodMessage.ean;

if (!ean) {
  throw new java.lang.RuntimeException("No EAN specified. This is a required
field.");
}
var prod = manager.getNodeHome().getObjectByKey("EAN", ean);
var category = getCategory(prodMessage.category);

if (!prod) {
  try {
    prod = category.createProduct(null, "BuyItem");
  } catch (e) {
    if (e.javaException instanceof
com.stibo.core.domain.ObjectTypeConstraintException || e.javaException instanceof
com.stibo.core.domain.NodeIdUniqueConstraintException) {
      throw new java.lang.RuntimeException("Could not create new product: " +
e.javaException.getMessage());
    }
    throw(e);
  }
  setValue(prod, "EAN", ean, true);
}
if (!prod.getParent().equals(category)) {
  try {
    prod.setParent(category);
  } catch (e) {
    if (e.javaException instanceof
com.stibo.core.domain.ObjectTypeConstraintException || e.javaException instanceof
com.stibo.core.domain.CycleConstraintException) {
      throw new java.lang.RuntimeException("Could not move product: " +
e.javaException.getMessage());
    }
    throw(e);
  }
}
setValue(prod, "ShortDescription", prodMessage.shortDescription, false);
setValue(prod, "ManufacturerName", prodMessage.manufacturerName, false);
setValue(prod, "Color", prodMessage.color, false);
```

IIEP - Configure FAB-DIS Importer Processing Engine



For details on the full configuration required by the FAB-DIS Importer, refer to the **FAB-DIS Import Format** topic.

IIEP - Configure STEP Importer Processing Engine



When using the STEP Importer processor, this step and its potential sub-steps, will typically be identical to the steps in the Import Manager. Additionally, the same options are available for mapping and transforming data, and for referencing business rules to be run during import.

1. Create a sample file that contains the basic structure of the data, as shown in the image above.
2. In **5.1 Select Sample File**, click the **Upload** button to upload the sample (representative) file for the files or messages the IIEP will process. This file provides data for mapping (when required) and configures how to import data, and is stored in the database, but is not imported into STEP. This step is not available in the Import Manager since it allows you to only import a single file at a time.

   If you want to modify a sample file that has already been uploaded, click the **Download** button and save the sample file to your computer. Make your changes and then click the **Upload** button to upload the modified sample file to the endpoint.

Note: When a pre-processor transforms the file to import, use a sample file that matches whatever format is produced by the pre-processor. If the sample file is Excel, no preview is shown.

- When using the **Transformation by XSLT** pre-processor, use a sample file that matches your selected format (STEPXML or Generic XML) otherwise an error is returned that the format is not valid. For more information, refer to **IIEP - Configure Transformation by XSLT Pre-processor** documentation.
- When using the **XML Normalizer** pre-processor, verify that the sample file includes both the new tag, and the copied tag data in the correct location. For more information, refer to **IIEP - Configure XML Normalizer Pre-processor** documentation.
- When importing **ETIM6**, modify the import file to include only the <IXF >, <Header>, and <Groups> sections and save it as your sample.

3. In **5.2 Select Data Source** file, if the IIEP is used to update commercial data, also called Terms Lists, select the Termslist Targets. For more information, refer to topics in the **Commercial Data** section of the **Publisher (Adobe InDesign Integration)** documentation.
4. In **5.3 Select Format** choose from the following options to indicate the type and basic parameters of the file you are importing.

   When using the XML Normalizer pre-processor, verify that the template file includes both the new tag, and the copied tag data in the correct location. For more information, refer to **IIEP - Configure Pre-processor** documentation.

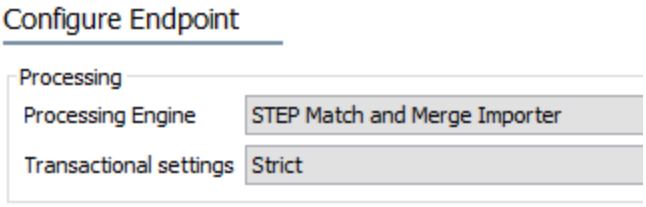
   - Advanced STEPXML Format
   - BMEcat Format
   - BMEcat 2005 Format
   - CSV Format
   - ETIM Format
   - ETIM IXF Format
   - ETIM v2 Format
   - Excel Format
   - Excel Smartsheet Format
   - FixedWidth Format
   - Flatplanner Format
   - Generic XML Format
   - IDoc MATMAS 05 Format
   - Publication Excel Format
   - STEPXML Format
   - UNSPSC Format

5. In **5.4 – Map Data** match the columns of data in the import file to the appropriate STEP equivalents, and include any transformations. For information about all of the available mapping options, refer to the **Inbound Map Data Options** section.

6. In **5.5 – Identify Objects** allows you to verify if loaded objects are existing data in STEP or will be created as new objects. Additionally, you can specify to locate existing objects by attribute value rather than by the ID, key, or name. For more information, refer to **Import Manager - Identify Objects**.
7. In **5.6 – Identify Destination** specify settings such as the user who is responsible for approving the objects after import, where to place new objects in STEP, and the object type of new objects. Be cautious if setting default parent and object types. For more information, refer to **Import Manager - Identify Destination**.
8. In **5.7 – Select Business Rules** specify which business rule(s) to apply when data is imported. For more information, refer to **Import Manager - Select Business Rules**.
9. In **5.8 – Advanced Settings** are not required on a day-to-day basis for data imports. Generally these settings are used to clean up data within STEP. Common setup is to perform a test on the test server first, and start testing with small files since these options can cause data integrity issues. For more information, refer to **Import Manager - Advanced Settings**.
10. Click the **Next** button to display **IIEP - Configure Post-processor** when available, or display **IIEP - Schedule Endpoint**.

IIEP - Configure Match and Merge Importer Processing Engine

The STEP Match and Merge Importer plays an integral role in the Matching and Merging deduplication solution, and is responsible for importing, processing, and standardizing data, and ultimately matching that data with existing golden records. Post-processing is not supported by the Match and Merge Importer.



Match and Merge Importing Process and Logic

Data can flow into STEP via an asynchronous inbound integration endpoint (IIEP). The IIEP is designed to receive large batches of source records from any of a number of Receiver plugins.

The incoming source data is translated into STEPXML import files. These input files are typically handled one at a time in sequence, according to the parallel settings of the IIEP queue, as defined in the **IIEP - Configure Endpoint** topic of the **Data Exchange** documentation. The result of the import operation is logged in workbench on the IIEP configuration's Background Processes tab and on the background process execution log.

Any failed records are stored on the BGP in a separate error file which allows the failed updates to be reattempted when errors have been corrected.

For details on configuring a data exchange method, either asynchronous or synchronous, for match and merge, refer to the **Configuring the Match Data Exchange Method** topic in the **Matching, Linking, and Merging** documentation.

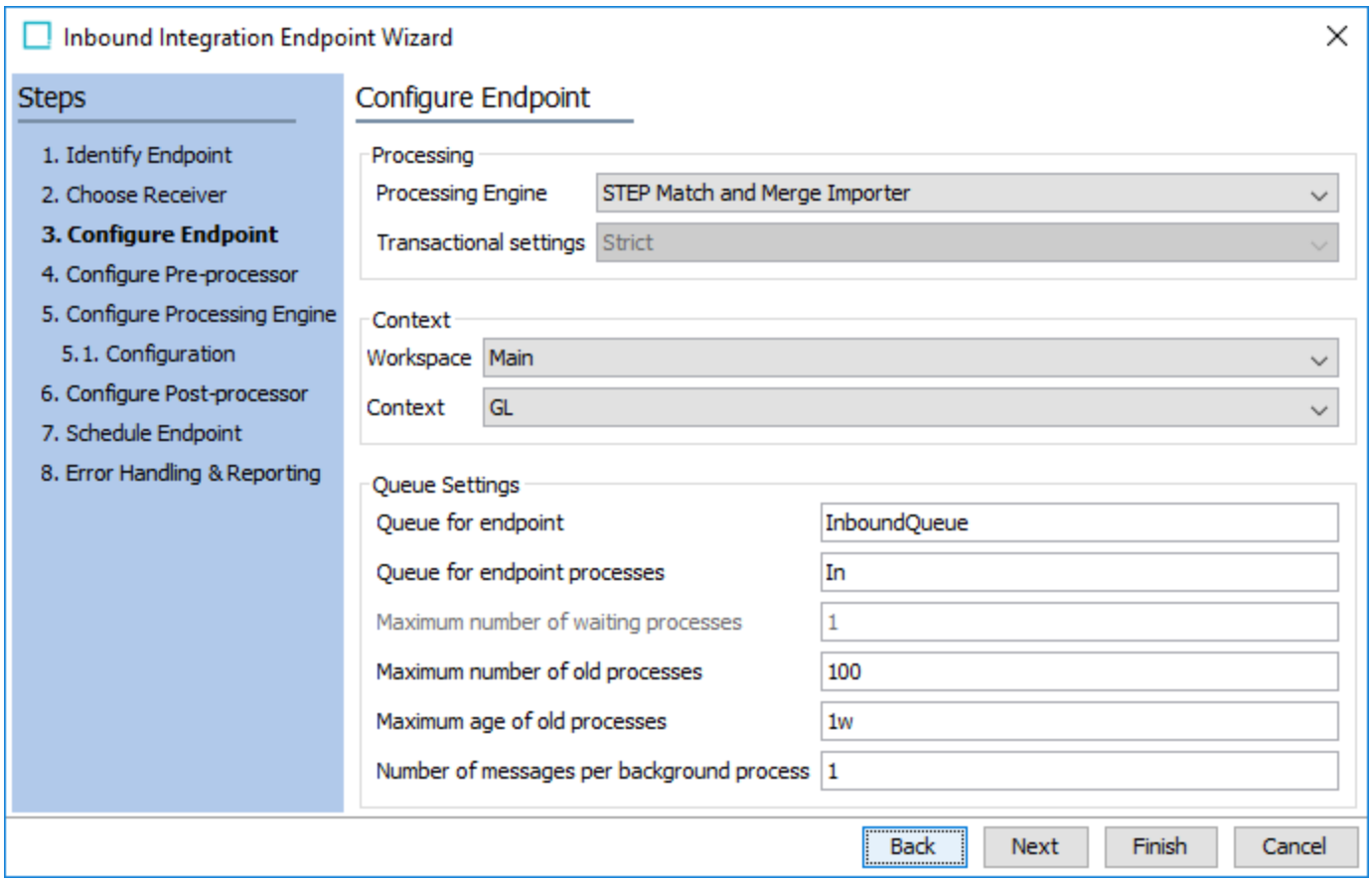
Match and Merge IIEP Configuration

Configuration of the STEP Match and Merge Importer begins on the 'Configure Endpoint' step of the Inbound Integration Endpoint wizard.

1. On the **Configure Endpoint** step, click the 'Processing Engine' dropdown and select 'STEP Match and Merge Importer.'

   Note: For this processing engine, the **Transactional settings** and **Maximum number of waiting processes** parameters are pre-configured and cannot be changed.

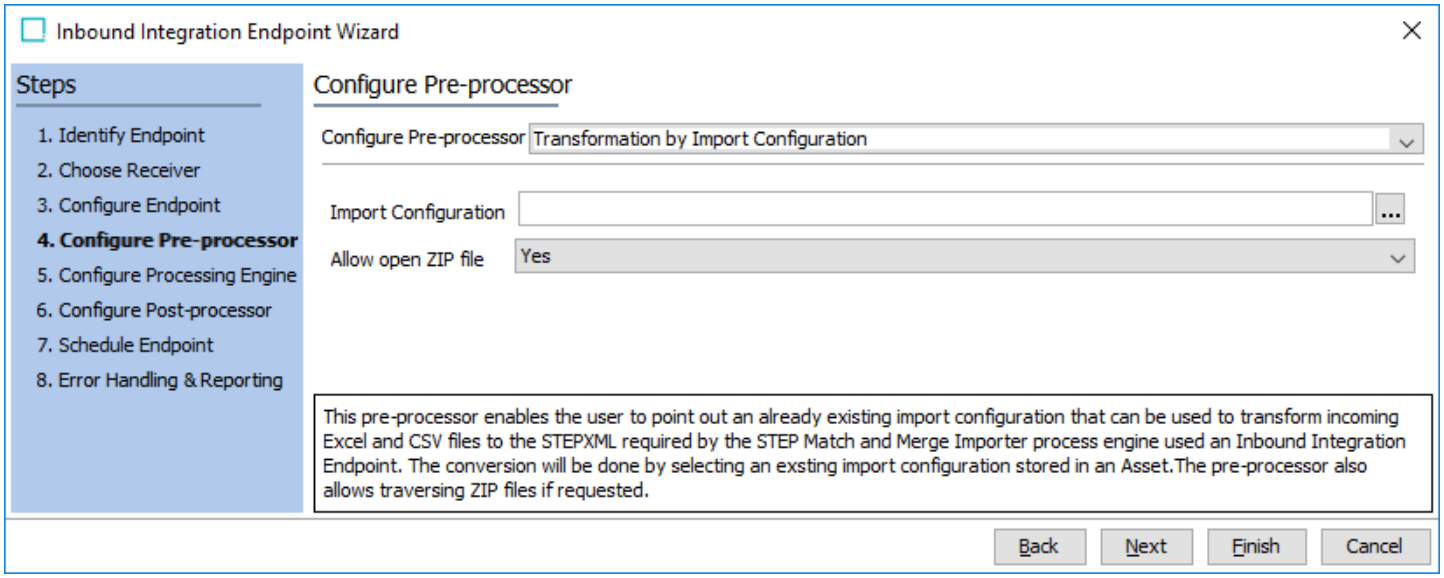
   For details on how the other parameters in this step may be configured, refer to the **IIEP - Configure Endpoint** documentation.



2. On the **Configure Pre-processor** step, an existing import configuration can be used to transform Excel / CSV files into STEPXML, so that the processing engine can read those files.
   - For the 'Configure Pre-processor' dropdown, select 'Transformation by Import Configuration.'

     When using this option, the relevant import configuration may contain Excel / CSV mappings to STEPXML. If these files contain columns for Source System and Source Record ID, then the Reference Meta-Data option must be used to map the Source System to the relevant References Type, and the Source Record ID must be mapped to the relevant meta-data attribute on the Reference Type in question.

   - For the 'Import Configuration' parameter, click the ellipsis button (...) and browse or search for an import configuration to use for transforming incoming Excel / CSV files.
   - For the 'Allow open ZIP file' parameter, select 'Yes' from the dropdown to allow the importer to open ZIP files and convert the files within.



If STEP IDs are not typically available in your import files, it is recommended to use an import configuration that can create entity-to-entity references via Source Record ID.

For more information on Excel / CSV imports, refer to the **Entity Reference via Source Record ID - Map Inbound** section of the **Data Exchange** documentation.

For more information on XML imports, refer to the **Source Record ID in STEPXML** section below.

Note: If the background process with this pre-processor fails and you want to restart it, the number of items mapped by import configuration can increase according to the previous run. To avoid this, disable the inbound integration endpoint, enable it again, and restart the import from the configured receiver.

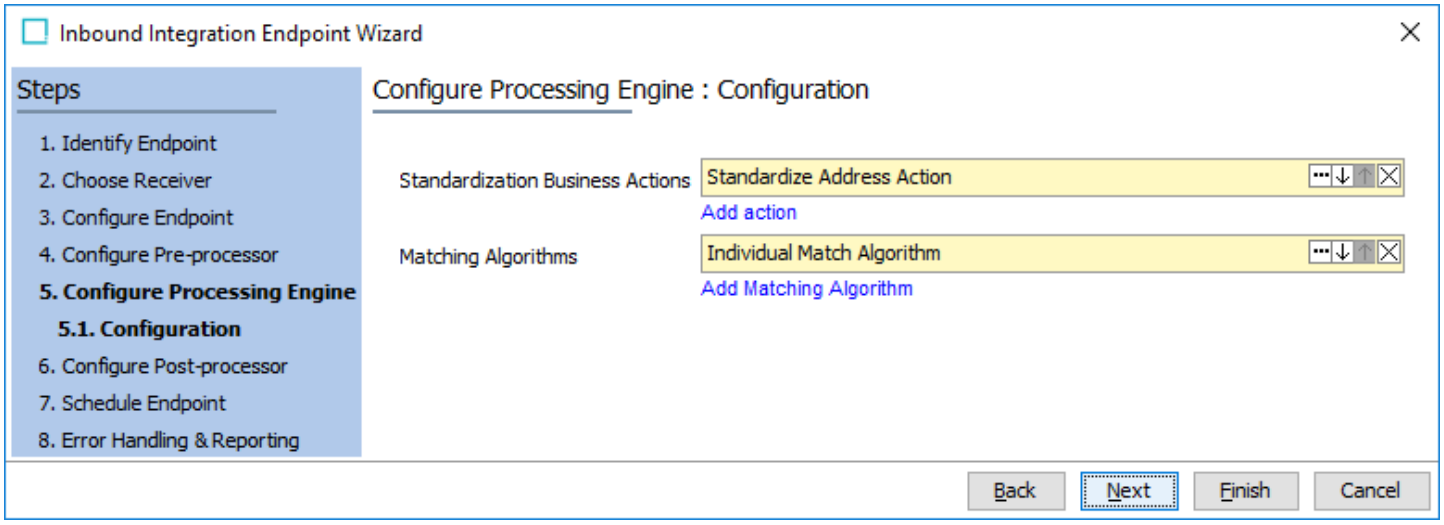
3. On the **Configure Processing Engine** step, business actions can be configured to process the temporary STEP source objects in order to standardize the data, which allows for more accurate matching.

   For both of the parameters, click the link again to add items. Execution is performed in the order displayed. Click the ellipsis button (...) to change an item, click the arrows to change the order, and click the **X** button to remove an item.

   - If a business action is required, click the **Add action** link and browse or search for the relevant business action.

     Important: The selected business action is called on the incoming data in a non-persistent form. Many of the operations available in the API are not applicable to non-persistent objects and will fail. You can successfully use operations related to reading and modifying attribute values, references, and data containers, however, approval and workflow-related operations cannot be used successfully.

- In the 'Matching Algorithms' parameter, click the **Add Matching Algorithm** link and browse or search for a matching algorithm. The matching algorithms configured are used to match the inbound source records against existing golden records and invoke survivorship rules.



Source Record ID in STEPXML

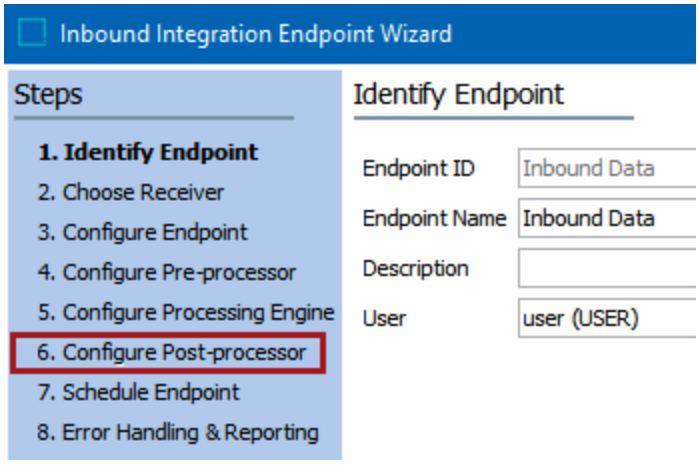
The Source Record ID of a given record is stored in the Source Record ID Attribute of that record. This attribute is a metadata attribute placed on the record's Source Relation Reference, which is a reference between the source record and the Source System in question.

The following is a STEPXML example of the relevant tags:

```
    <Entity>
      <Name>Carl Lucas</Name>
        <Values>
          <Value AttributeID="FirstName">Carl</Value>
          <Value AttributeID="MiddleName"></Value>
          <Value AttributeID="LastName">Lucas</Value>
          <Value AttributeID="Email">l.carl@email.com</Value>
          <Value AttributeID="PhoneNo"></Value>
        </Values>
    <EntityCrossReference EntityID="SourceSystem_Test" Type="MergeSourceRelation">
          <MetaData>
            <Value AttributeID="SourceRecordID">SST_00001</Value>
          </MetaData>
```

```
14      </EntityCrossReference>
15      <EntityCrossReference Type="CustomerToOrganization">
16            <Entity>
17              <EntityCrossReference EntityID="SAP_Test" Type="MergeSourceRelation">
18                    <MetaData>
19                      <Value AttributeID="SourceRecordID">SSB_00002</Value>
20                    </MetaData>
21                  </EntityCrossReference>
22      </Entity>
23          </EntityCrossReference>
24      </Entity>
```

IIEP - Configure Post-processor



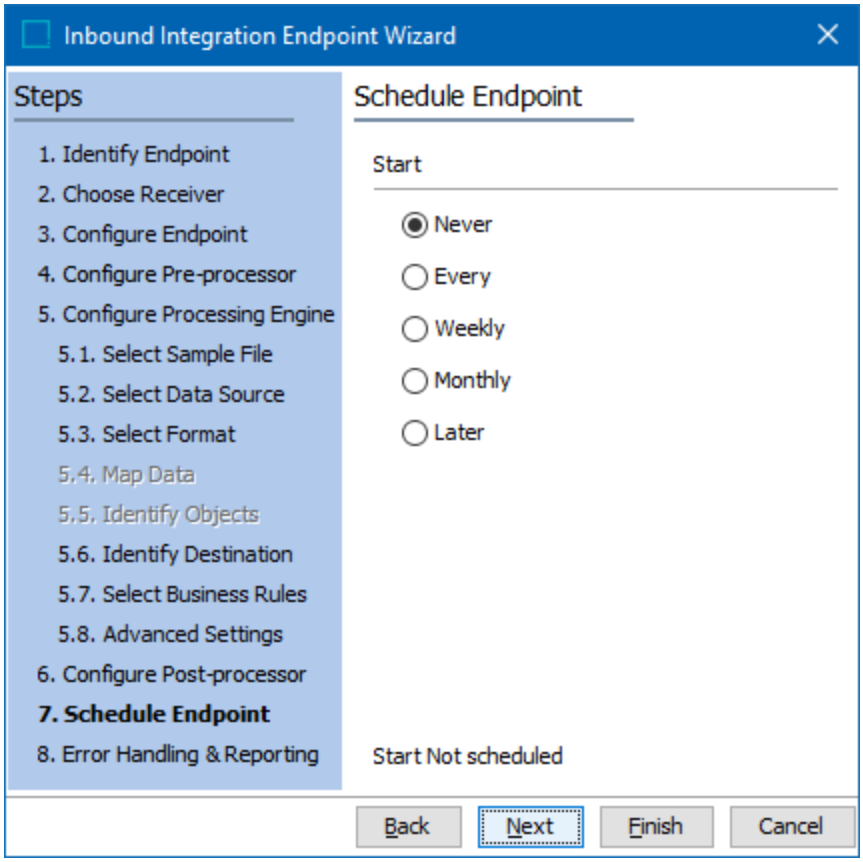
A post-processor has access to import events (information about what has changed) and, based on the events, can trigger any required system change. Typically, post-processor logic is implemented via business rules referenced from the processing engine configuration. For assistance in creating a custom post-processor, contact Stibo Systems Support.

The 'Invoke OIEP' post-processor option is available for use with an external Version Control System (VCS). For more information on VCS integration, refer to the **Version Control System Integration** topic in the **Configuration Management** documentation.

Note: For on-premises systems, to enable the wizard step and access the 'Invoke OIEP' post-processor, you must apply the configuration-management add-on component.

Click the **Next** button to display **IIEP - Schedule Endpoint**.

IIEP - Schedule Endpoint



Use Schedule Endpoint to specify how often the endpoint should search for data to be processed considering the following:

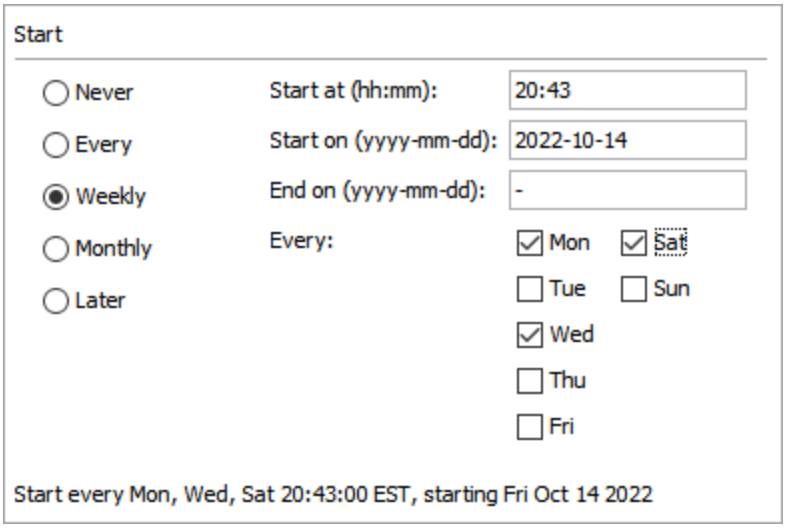
- IIEPs that do not use the REST Receiver method should typically be scheduled to poll the data source for new files / messages with regular intervals.
- IIEPs that do use the REST Receiver method can be invoked via a REST POST request. With this functionality, for example, a system delivering data to STEP could potentially invoke the IIEP after having uploaded a file to a Hotfolder. For more information, refer to the **REST Receiver** documentation.

Important: Consider the time zone of the application server compared to that of the workbench (the client) where the schedule is created or viewed. When scheduling a job, the local time zone is displayed in the workbench, but the time zone of the server is used to run the background process. Although displayed, the time zone of the client is not included in the instruction to the server to run the job. This can cause confusion about when the job will run since the scheduled time is not automatically converted to accommodate potential differences in time zones.

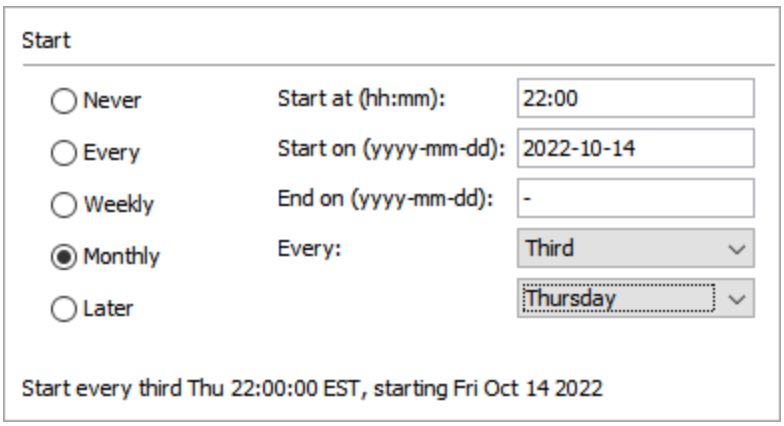
1. Choose a radio button to select the necessary schedule:
   - **Never** - invoke the endpoint manually, no additional parameters are required, and no schedule is applied. This is the default setting and should be used while testing your endpoint.

- **Every** - automatically run the endpoint repeatedly, every selected number of minutes. One (1) minute is the shortest interval allowed and is closest to real time. Enter the number of minutes in the text box. The selection is summarized at the bottom of the dialog.

- **Weekly** - automatically run the endpoint repeatedly, based on the selected time, start and end dates, and days of the week. Use this option if a daily schedule is needed. The 'Start at' parameter determines the time of day that the endpoint will run. The 'Start on' parameter determines the date the endpoint will first run, while the 'End on' parameter determines the date of the endpoint's final run. The 'Every' checkboxes determine the days of the week when the endpoint will run. The selections are summarized at the bottom of the dialog.



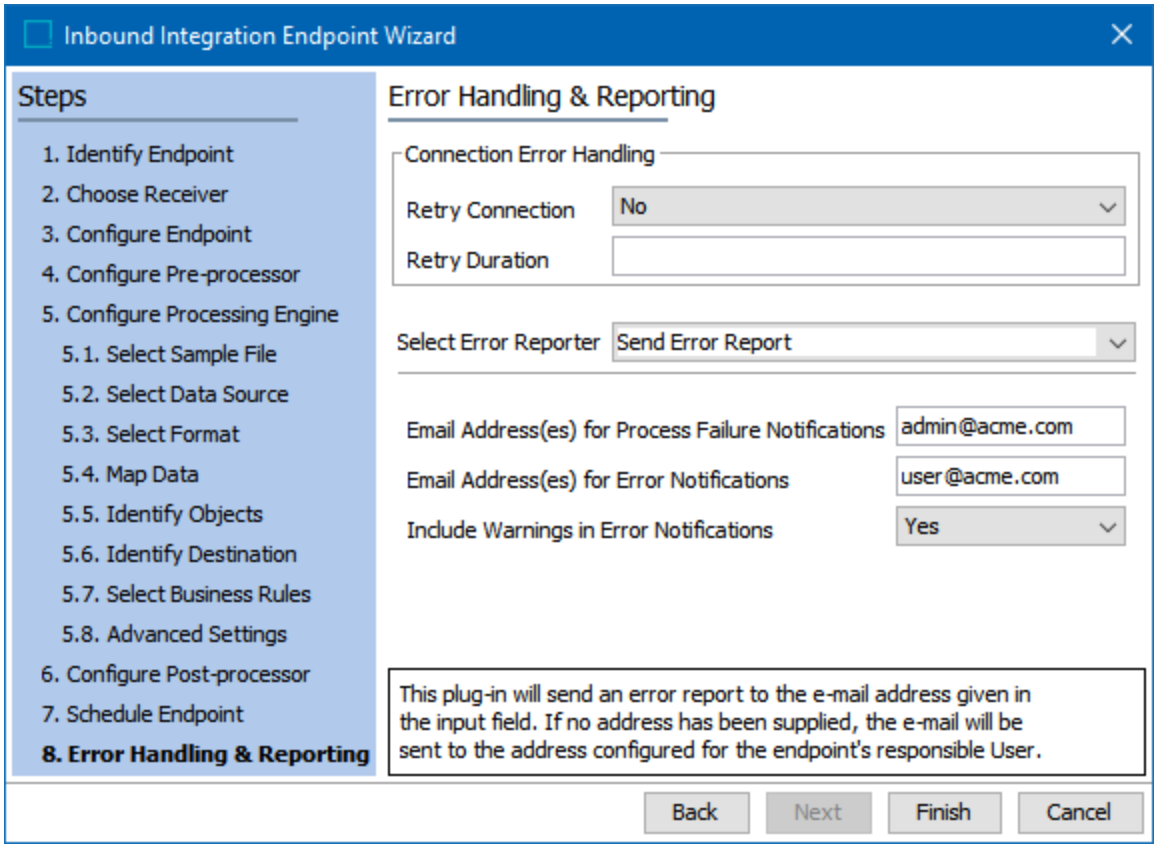
- **Monthly** - automatically run the endpoint repeatedly, once a month, based on the selected time, start and end dates, week of the month, and day of the week. The 'Start at' parameter determines the time of day that the endpoint will run. The 'Start on' parameter determines the date the endpoint will first run, while the 'End on' parameter determines the date of the endpoint's final run. The 'Every' dropdown parameter selections for the week of the month and the day of the week determine when the endpoint will run. The selections are summarized at the bottom of the dialog.



- **Later** - automatically run the endpoint only once, at the time and date specified. The selections are summarized at the bottom of the dialog.

2. Click the **Next** button to display **IIEP - Error Handling & Reporting**.

IIEP - Error Handling & Reporting



For additional information on IIEP errors, refer to the **Resolving a Failed Background Process** section of the **Handling Failed IIEP Background Processes** topic.

1. For **Connection Error Handling**, the default is no connection error handling. The Retry Duration parameter is ignored when 'No' is set for the Retry Connection parameter. Enabling Connection Error Handling allows automated reconnection attempts when the external system is unavailable. When connection retries begin, a warning is logged to the Execution Report; if a connection cannot be established after the Retry Duration expires, an error message is logged to the Execution Report. Refer to the **Automated Retries** section below for more information.

    - On the **Retry Connection** parameter, set 'Yes' to automate reconnection attempts for HTTP-based receivers including Amazon SQS and Kafka. A 'No' setting requires manually restarting the IIEP if the connection fails.

      Note: Authentication-related connection errors are not retried and the IIEP fails immediately.

    - On the **Retry Duration** parameter, when blank, the default of 30 days is applied. Valid settings are an integer plus one of the following time indicators: s (seconds), m (minutes), h (hours), and d (days). For example, 45m indicates that connection retries will continue for 45 minutes, during which time, the IIEP

shows a 'Failed (retrying)' state. Multiple integers with time units in order of size may also be used. For example, 1h45m30s indicates that connection retries will continue for 1 hour, 45 minutes and 30 seconds. After the 'Retry Duration' expires, if the connection cannot be re-established with the last retry, the IIEP fails, and email alerts are distributed based on the 'Select Error Reporter' settings. For more information, refer to the **Running an Inbound Integration Endpoint** topic.

2. On **Select Error Reporter**, select an option to be activated if an endpoint-related background process has warnings or fails.

   - **Send Error Report**: Use the following parameters to add a relevant email address (or use a semi-colon to add more than one email address) or group email that can be used to receive error and warning reports. When emailing multiple accounts, consider creating a group email and use that email address in the parameters instead of adding multiple email addresses for individuals. With this configuration, a single email group can be updated as the group members change while the 'Select Error Reporter' parameter remains unchanged.

     Note: To send email, the SMTP server must be configured on the application server. For information on configuring email from STEP, refer to the **Email from STEP** topic in the **Resource Materials** of online help.

     The email alert contains information about the failed endpoint, including the server's name, the background process, failed file, failing process step, cause of the error, and a copy of the file that triggered the error.

     If no email address is entered, the alert(s) are sent to the email address of the user who created the integration endpoint. If no email is defined for this user, or if no mail server is defined in the configuration, the error report will be written to the failed Background Process Execution Log.

     When Connection Error Handling is enabled and warning notifications are included with error emails using the parameter below, a warning is written to the Execution Report when retries begin. An email is sent to the address(es) in the Process Failure Notifications parameter after the retry duration expires and the endpoint fails.

     - **Email Address(es) for Process Failure Notifications**: Sends an alert email to the listed address (es) when a fatal processing error occurs that puts the IIEP in a 'Failed' or state.
     - **Email Address(es) for Error Notifications**: Sends an alert email to the listed address(es) when data warnings and/or errors occur that do not cause the process to stop.
     - **Include Warnings in Error Notifications**: A 'Yes' setting means alert emails include information on both warnings and errors. A 'No' setting means only error information is addressed in Error Notification emails. To receive an email when the IIEP enters the 'Failed (retrying)' state, set this option to 'Yes.'

   - **Do Nothing** disables the error reporter and no notifications by email are sent.

Note: If the error report is larger than the default maximum, 10 MB, the report will not be attached to the auto-generated email sent to the configured email address(es). If the default maximum is not suitable, an admin can set the following case-sensitive property in the sharedconfig.properties file on the application server to adjust the maximum file size:

```
Integration.Endpoint.ErrorFileSizeLimit={MB size}
```

Replace the '{MB size}' element with the maximum MB file size allowed for error report attachments. Error reports larger than this configured maximum are not attached to the email.

3. Click **Finish**. The endpoint is created and appears in the setup group.
4. The endpoint must be enabled before it can start processing data. For more information, refer to the **Running an Inbound Integration Endpoint** documentation.

Running an Inbound Integration Endpoint

After configuring an inbound integration endpoint (IIEP), it must be enabled before it can run on schedule or be invoked manually.

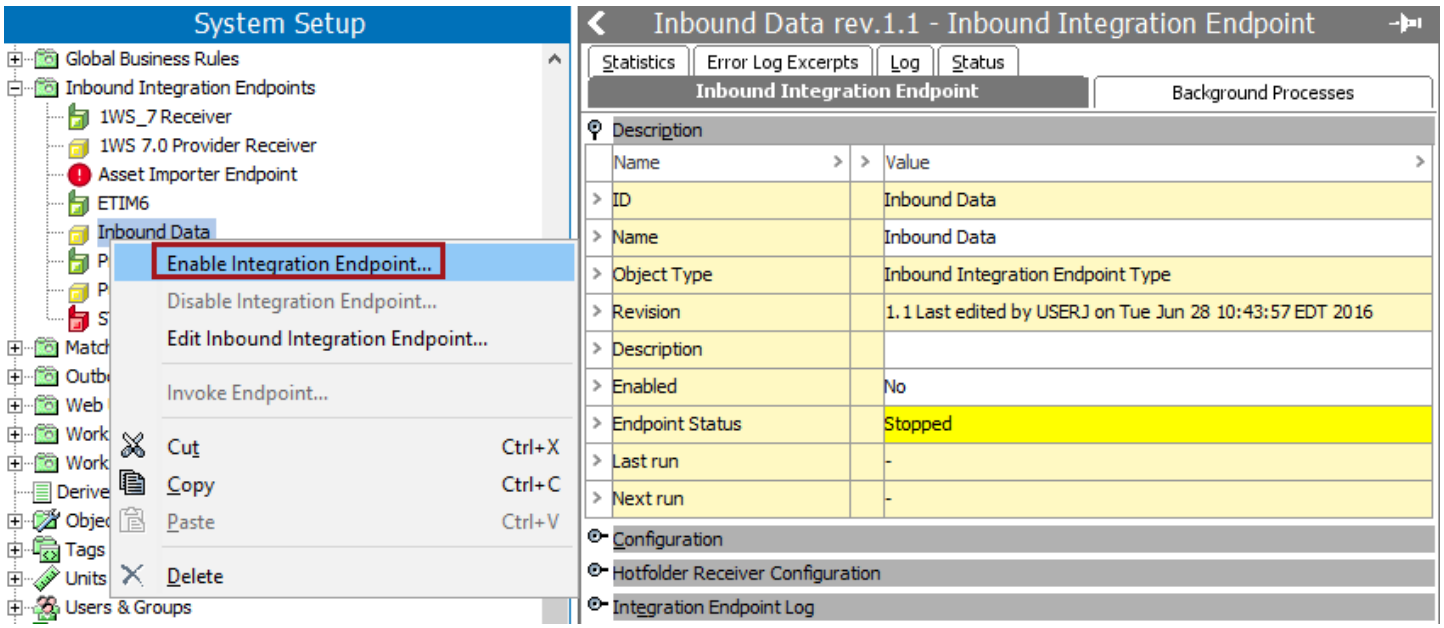
The following icons indicate the status of the IIEP. The endpoint status is displayed in the System Setup hierarchy and on the endpoint editor in the Description flipper.

Icon	Endpoint Status	Description
	Enabled	The integration endpoint is active, connected, and running.
	Disabled	The integration endpoint has stopped, and no data is being imported. Newly created endpoints, and endpoints that have been manually disabled, will have this status.
	Failed (retrying)	The IIEP is running and is attempting to recover from a connection error. After retrying in the original background process (BGP) for approximately one minute without success, the wait time begins and the process goes into a Waiting state before the next attempt to retry processing. When the wait time expires, the BGP state changes to running and connection is reattempted.
	Failed	An error occurred during processing that caused the IIEP's BGP to fail. No data will be imported until the endpoint has been manually resumed / reactivated. For more information, refer to **Handling Failed IIEP Background Processes** documentation.

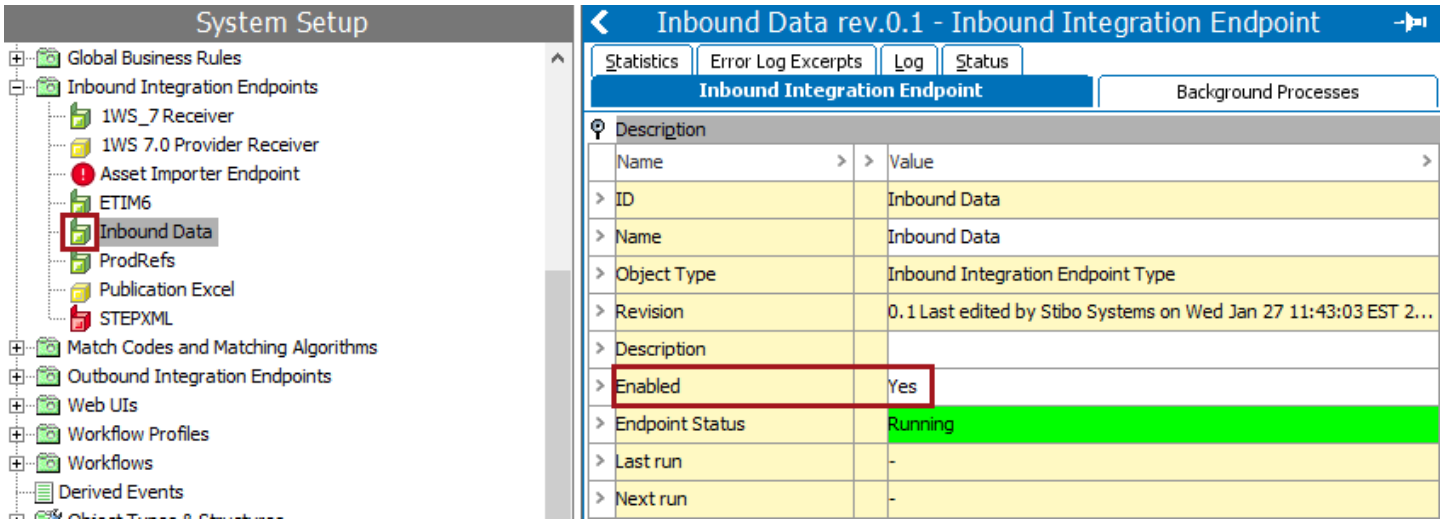
Enable and Invoke an Inbound Integration Endpoint

Once configured, use the following steps to run the endpoint.

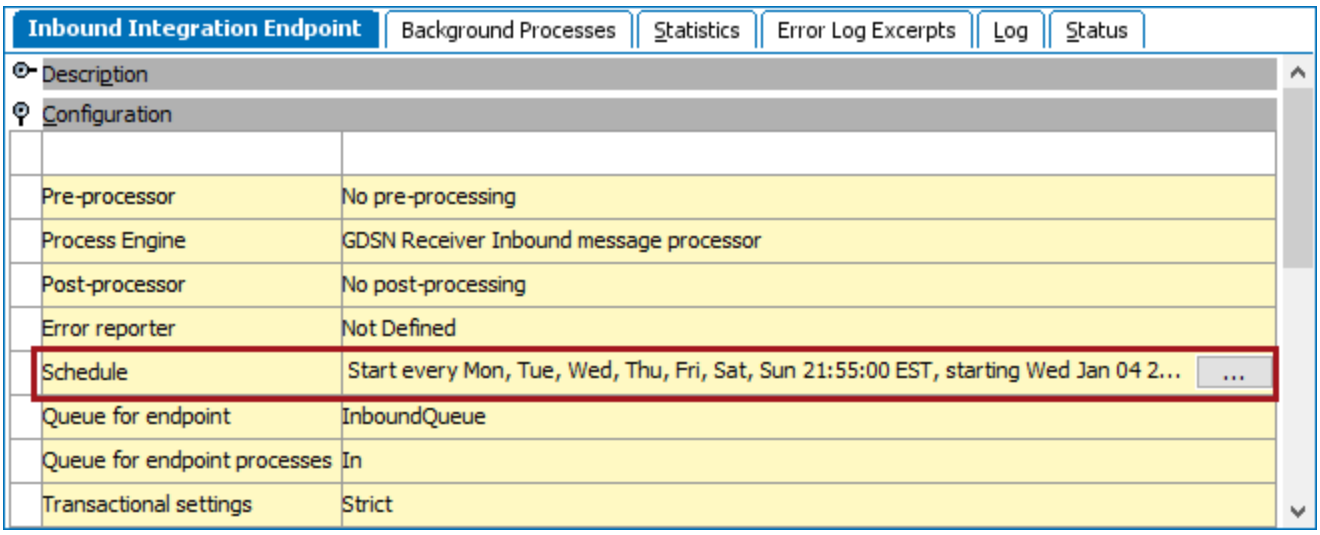
1. On System Setup expand the Inbound Integrations Endpoints node to display all existing inbound endpoints.
2. Select a disabled inbound endpoint ( ) or a failed endpoint ( ), right-click, and select **Enable Integration Endpoint** from the menu.



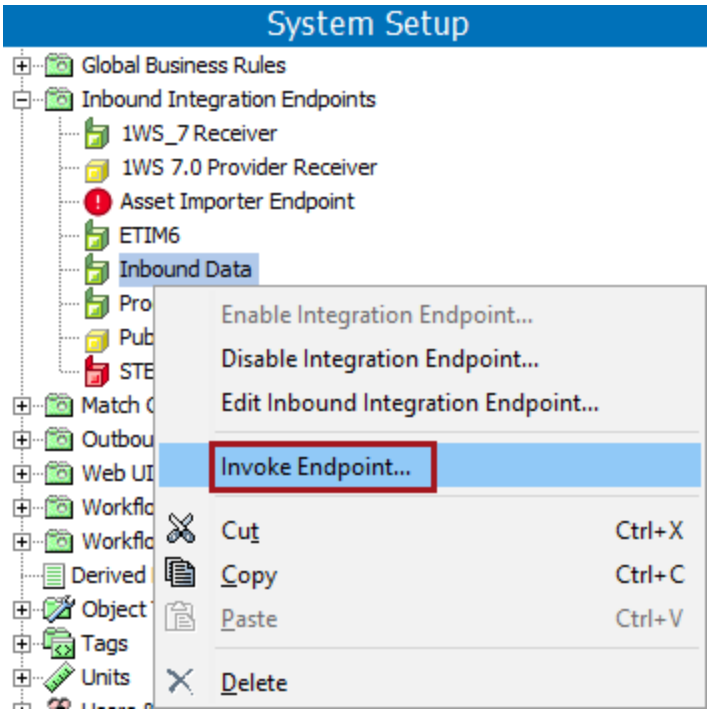
The tree navigator icon now shows enabled. On the Inbound Integration Endpoint tab, the Description flipper includes the 'Enabled' parameter, which now displays 'Yes.'



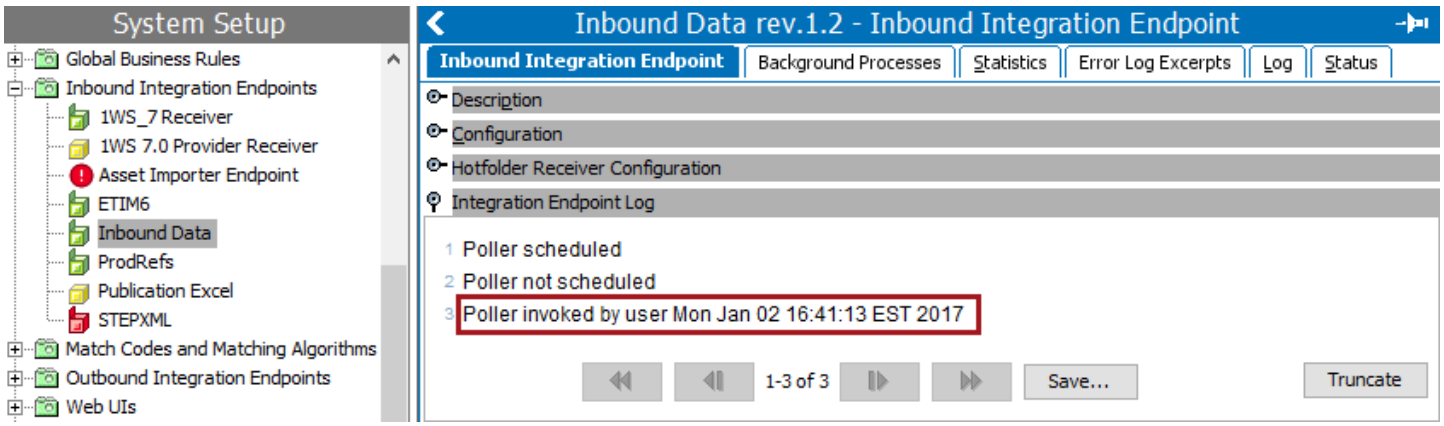
3. Open the Configuration flipper to view the Schedule parameter.
   - If scheduled, the Schedule parameter shows when the IIEP will be invoked automatically.



- If the endpoint is not scheduled, or if the IIEP needs to be invoked on demand, right-click the enabled IIEP and click **Invoke Endpoint**.

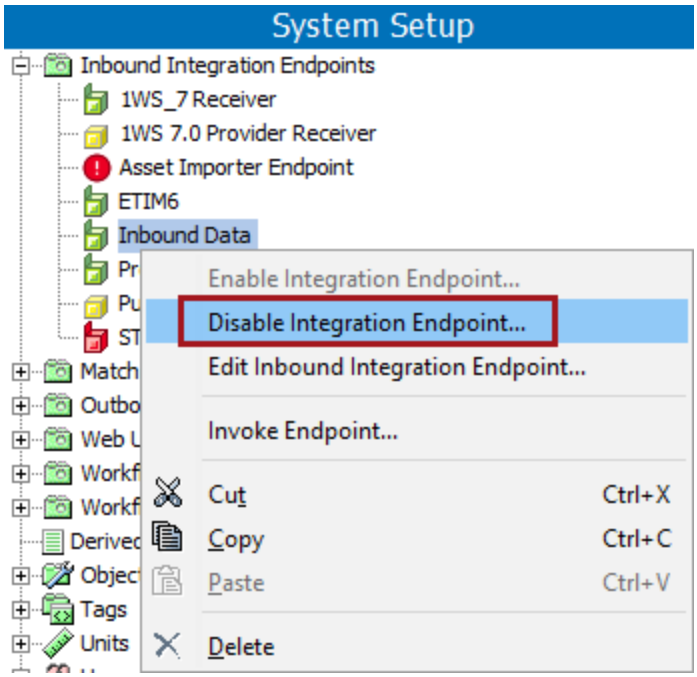


4. When invoked on schedule and manually, files or messages available in the location identified by the IIEP receiver configuration are processed. The Integration Endpoint Log flipper displays the time the endpoint was invoked. For scheduled IIEPs, the log includes a link to the BGP that was started. For more information, refer to **Maintaining an Inbound Integration Endpoint**.



Disable an Inbound Integration Endpoint

When an IIEP displaying the enabled icon ( ) should be stopped from processing data, you can disable it. The configuration remains unchanged, and no further data is processed. After selecting the IIEP, right-click and select **Disable Integration Endpoint** to set the IIEP to disabled and display the disabled icon ( ).

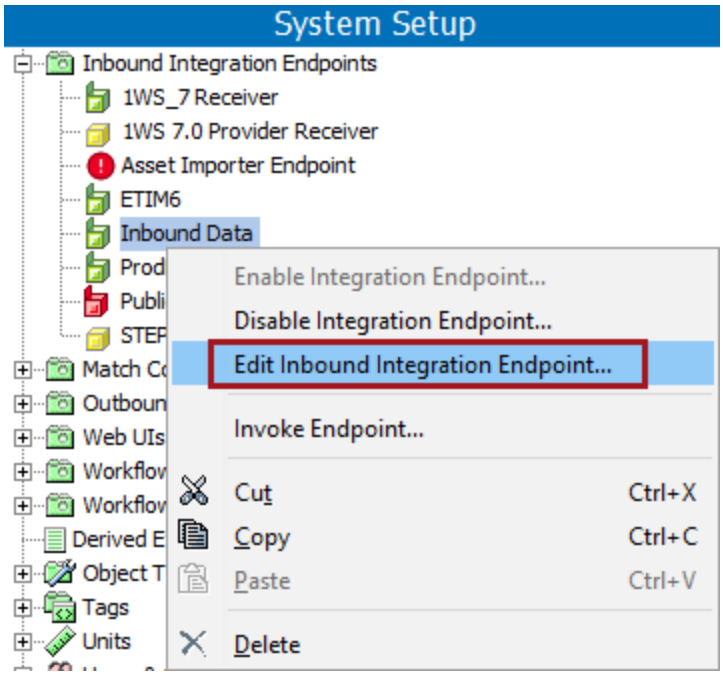


Once disabled, the IIEP can be restarted using the Enable Integration Endpoint step defined above.

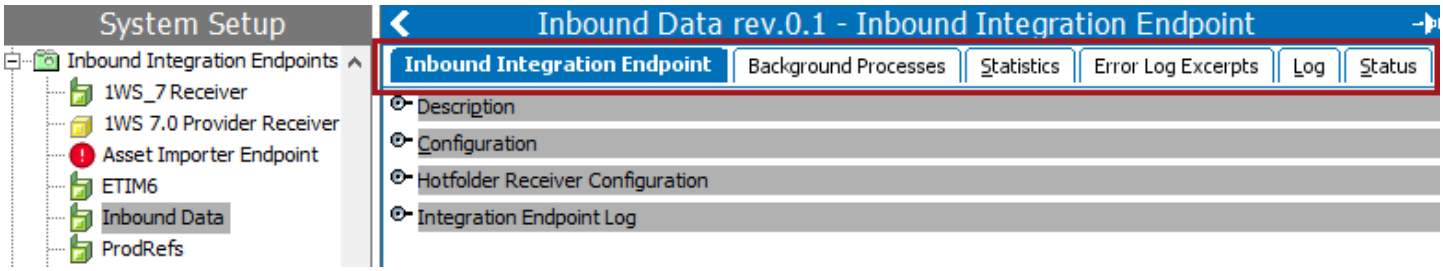
Additional information about restarting a failed endpoint, indicated by the failed icon ( ) can be found in **Handling Failed IIEP Background Processes** documentation.

Maintaining an Inbound Integration Endpoint

Inbound integration endpoint settings can be viewed in the Inbound Integration Endpoint editor. To edit the settings, reopen the wizard using the hyperlinks defined below, or with the **Edit Inbound Integration Endpoint** option on the IIEP right-click menu.



Each tab on the editor is defined below and includes parameters to maintain the IIEP.



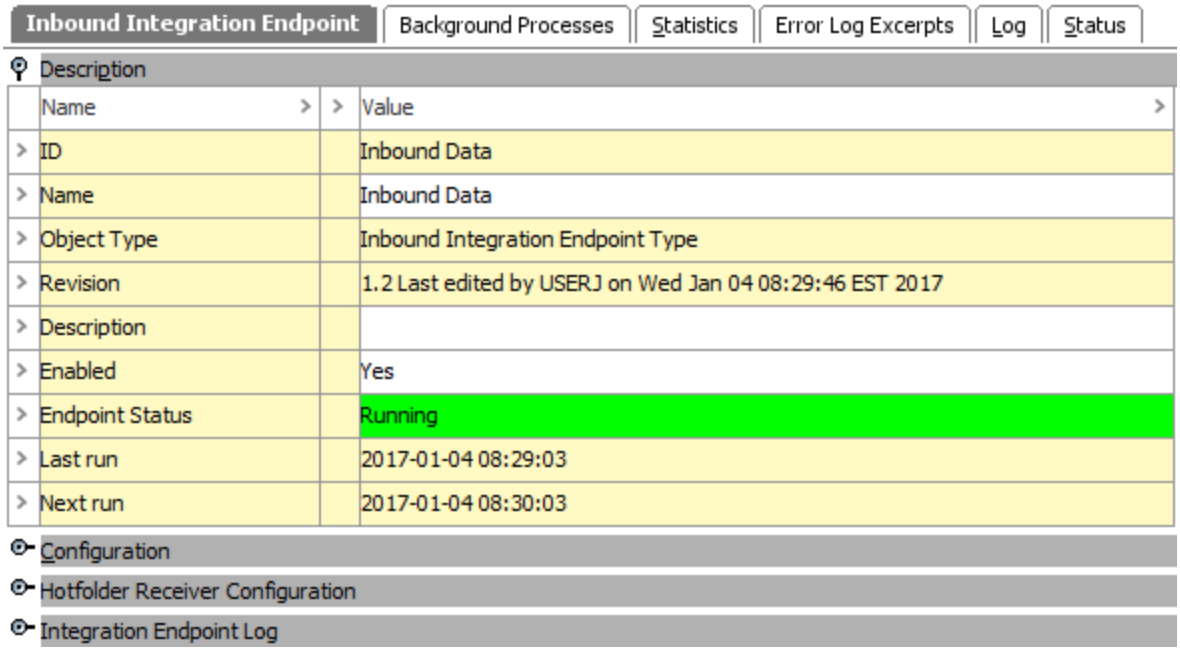
Inbound Integration Endpoint Tab

This tab holds basic information within the flippers defined below.

Description Flipper

The Description flipper includes basic information to identify the IIEP. Only the name, description, and enabled parameters can be edited. This data is originally set up in the IIEP wizard, refer to **IIEP - Identify Endpoint** documentation.

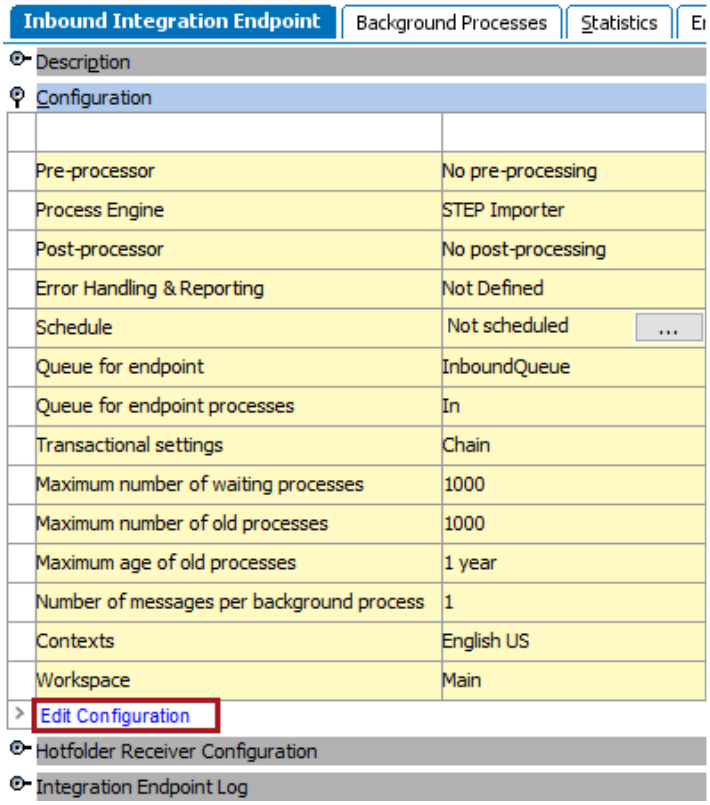
For information on the Endpoint Status, refer to **Running an Inbound Integration Endpoint** documentation.

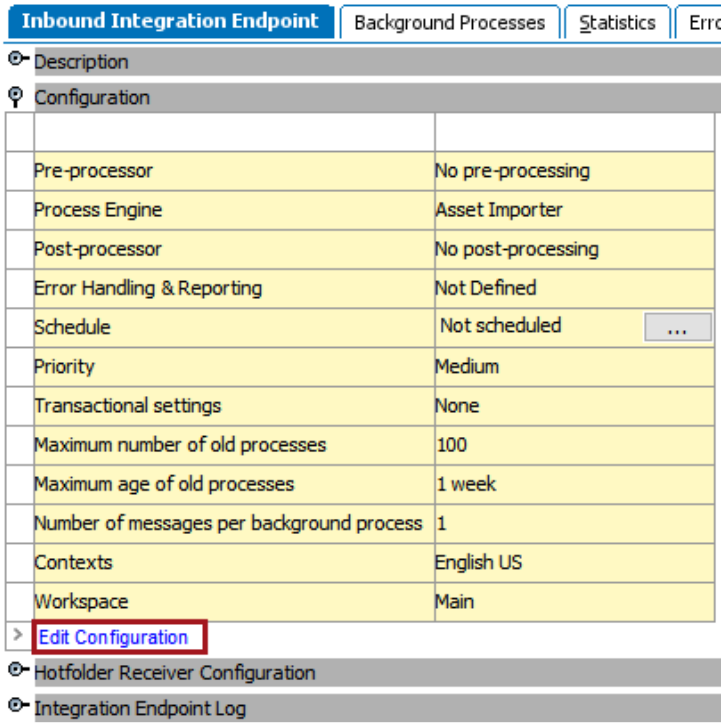


Configuration Flipper

The Configuration flipper displays many of the settings determined through the wizard.

- The **Edit Configuration** link opens the wizard and allows you to modify the following parameters:
- For information on the Pre-processor, refer to **IIEP - Configure Pre-processor**.
- For information on the Process Engine, refer to **IIEP - Configure Processing Engine**.
- For information on the Post-processor, refer to **IIEP - Configure Post-processor**.
- For information on the Error reporter, refer to **IIEP - Error Handling & Reporting**.
- Changes to the schedule can be made easily using the Schedule parameter ellipsis button (...). This opens the wizard on the Schedule Endpoint step, refer to **IIEP - Schedule Endpoint**.
- For information on the Queue for endpoint (legacy), Queue for endpoint process (legacy), Transactional settings, Maximum number of waiting processes, Maximum number of old processes, Maximum age of old processes, Number of messages per background processes, Contexts, and Workspace parameters, refer to **IIEP - Configure Endpoint**.
- For information on the Priority parameter, refer to the **BG Processes Execution Management** topic in the **System Setup** documentation.

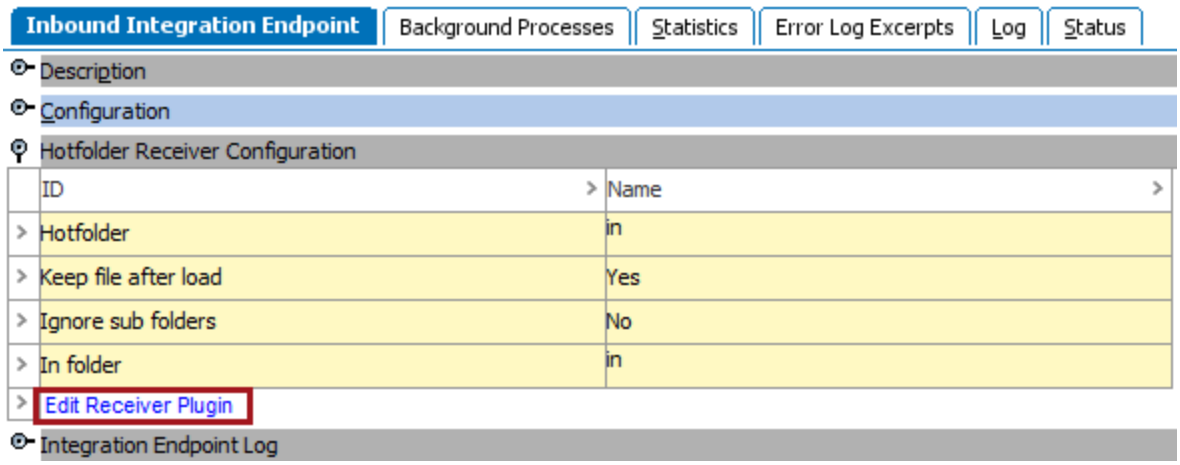




Receiver Configuration Flipper

The Receiver Configuration flipper displays details about the receiver method selected for the IIEP. For more information about receiver methods, refer to the **Receiver Methods** section of the **IIEP - Choose Receiver** topic.

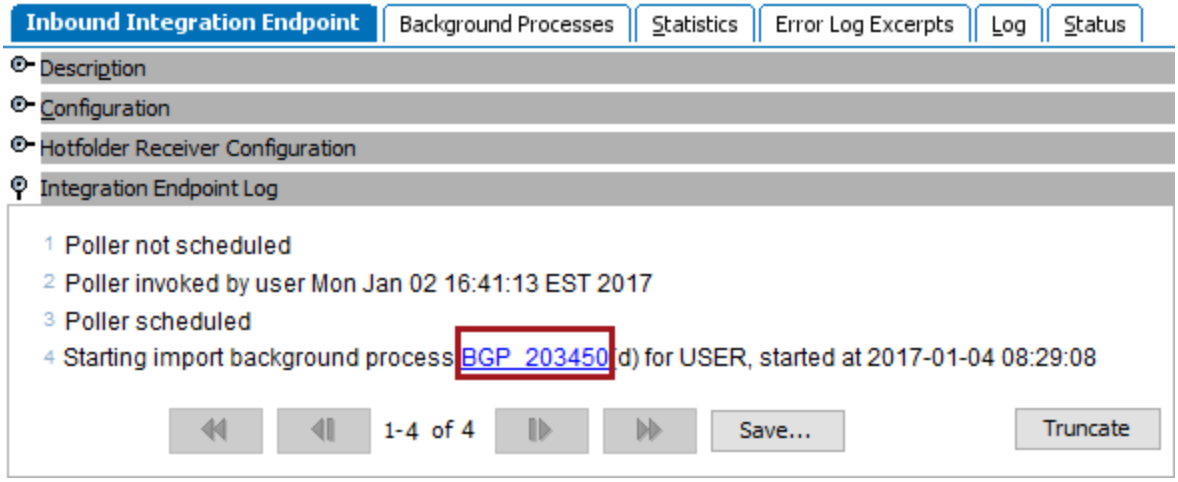
- The **Edit Receiver Plugin** link opens the wizard on the Choose Receiver step and allows you to modify the settings. For more information, refer to the **Receiver Methods** section of the **IIEP - Choose Receiver** topic.



Integration Endpoint Log Flipper

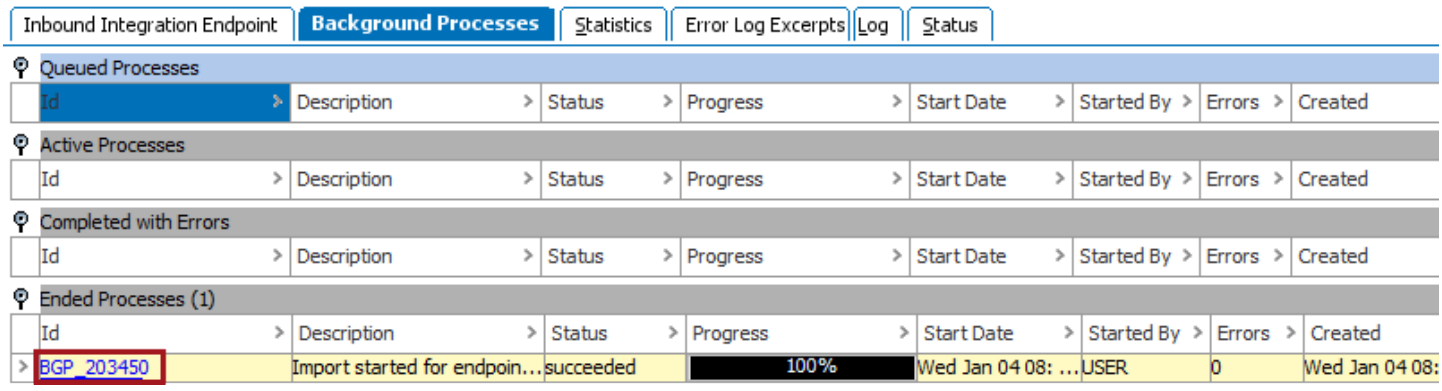
The Integration Endpoint Log flipper contains information about the endpoint background process. An entry is generated each time the endpoint is invoked, when background processes generated by the endpoint are started, and when processing errors occur.

- The background process (**BGP**) link opens the individual Importer Background Processes.



Background Processes Tab

This tab holds information about any related processes. Hover over an ID to display the background process (**BGP**) link, click it to open the individual inbound background processes.



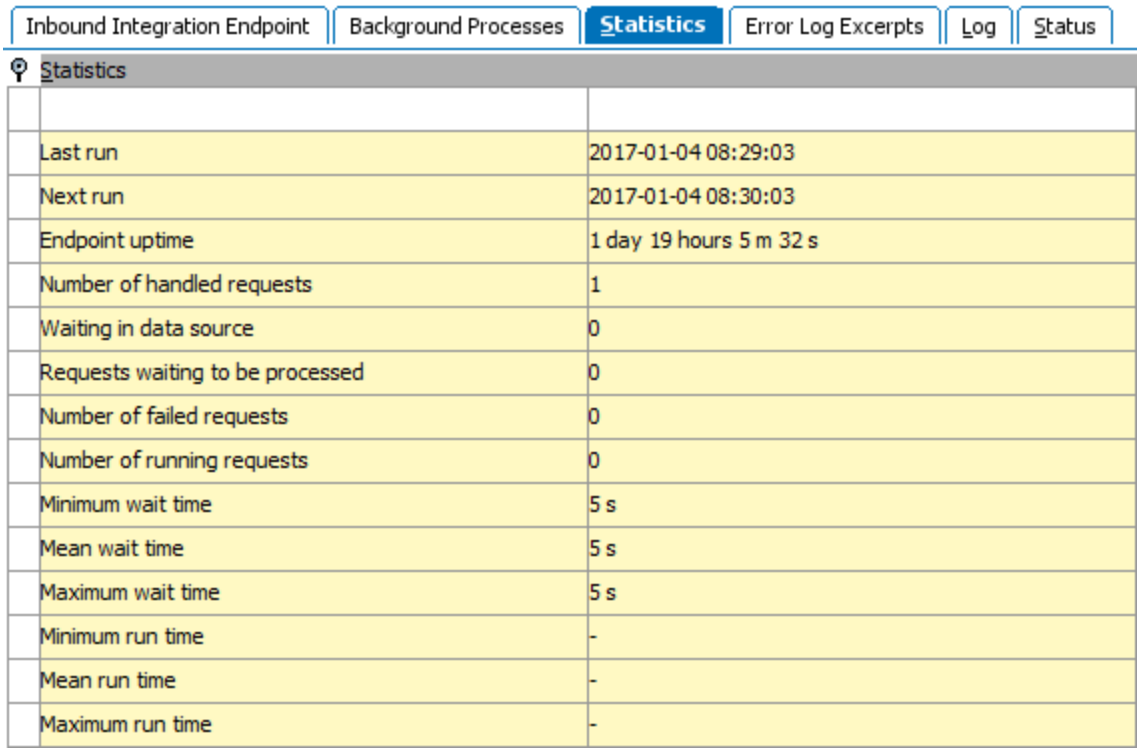
For details on the states of a background process, refer to the **BG Processes States** topic in the **System Setup** documentation.

For more information, refer to the **Monitoring an IIEP via Background Process** section.

Statistics Tab

This tab provides statistics about the number of requests handled, run times, and wait times. Key measures are Endpoint up-time, Waiting in data source, Number of failed requests (Background Processes with errors), and Mean run time (average processing time per import). This data cannot be exported.

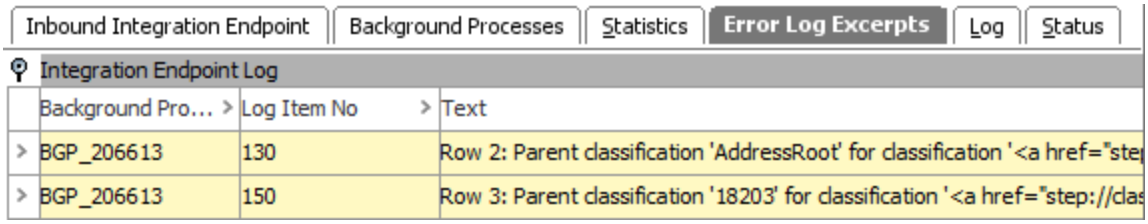
Note: Receivers can poll the data source to check for waiting files or messages more frequently than the IIEPs are invoked.



Last run	2017-01-04 08:29:03
Next run	2017-01-04 08:30:03
Endpoint uptime	1 day 19 hours 5 m 32 s
Number of handled requests	1
Waiting in data source	0
Requests waiting to be processed	0
Number of failed requests	0
Number of running requests	0
Minimum wait time	5 s
Mean wait time	5 s
Maximum wait time	5 s
Minimum run time	-
Mean run time	-
Maximum run time	-

Error Log Excerpts Tab

This tab shows data from the main Java log file related to failed background processes with Log Level > 'Info.' Click a hyperlink to a failed background process and correct the cause of the failed background process. For more information, refer to **Handling Failed IIEP Background Processes**.

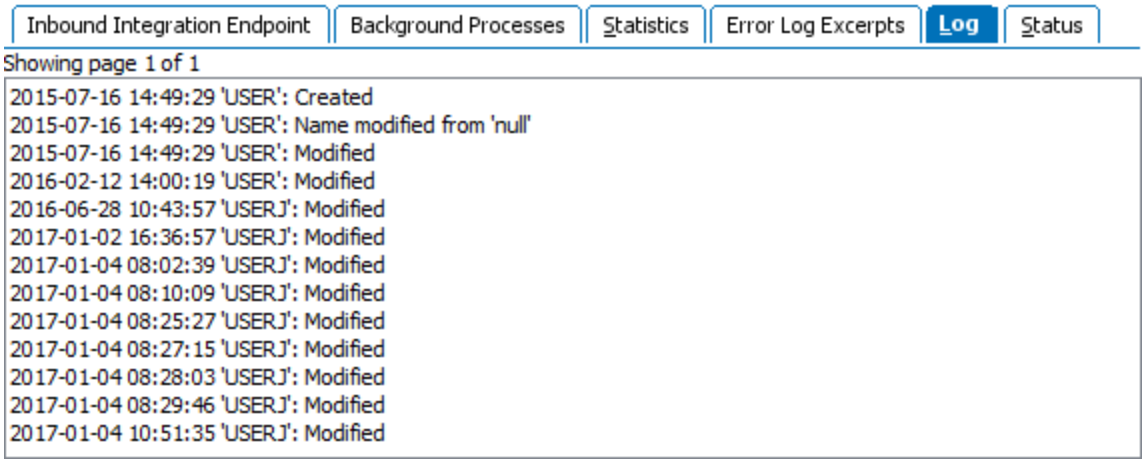


Background Pro...	Log Item No	Text
BGP_206613	130	Row 2: Parent classification 'AddressRoot' for classification '<a href="ste
BGP_206613	150	Row 3: Parent classification '18203' for classification '<a href="step://cla

The number of errors that display on the 'Error Log Excerpts' tab per background process is, by default, 10. This maximum can be adjusted adding the `IntegrationEndpoint.ErrorLogExcerptsLimit` property to the sharedconfig.properties file, and setting it to the desired maximum. It should be noted that configuring a high maximum number of displayed errors for this tab could result in performance issues.

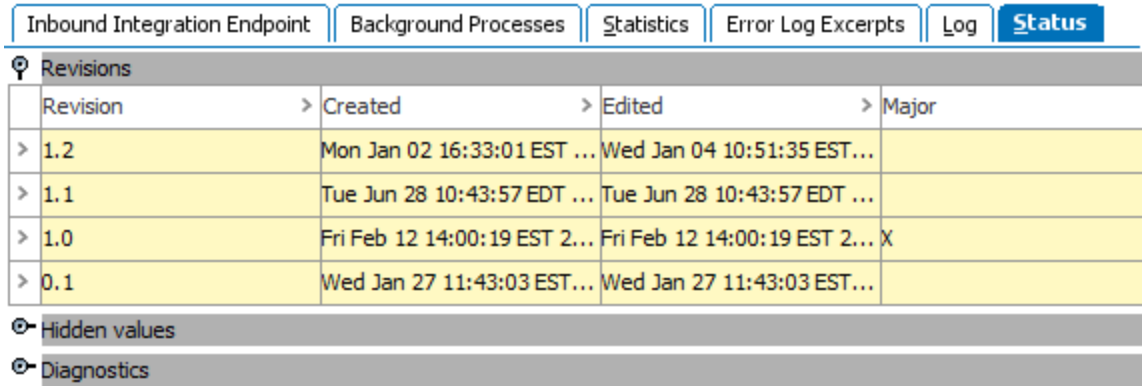
Log Tab

This tab provides information about changes to the IIEP configuration.



Status Tab

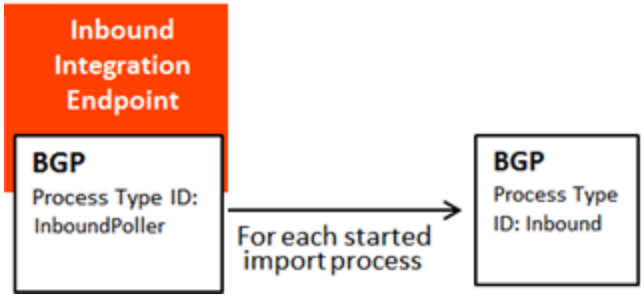
This tab provides information about revisions, hidden values and diagnostics.



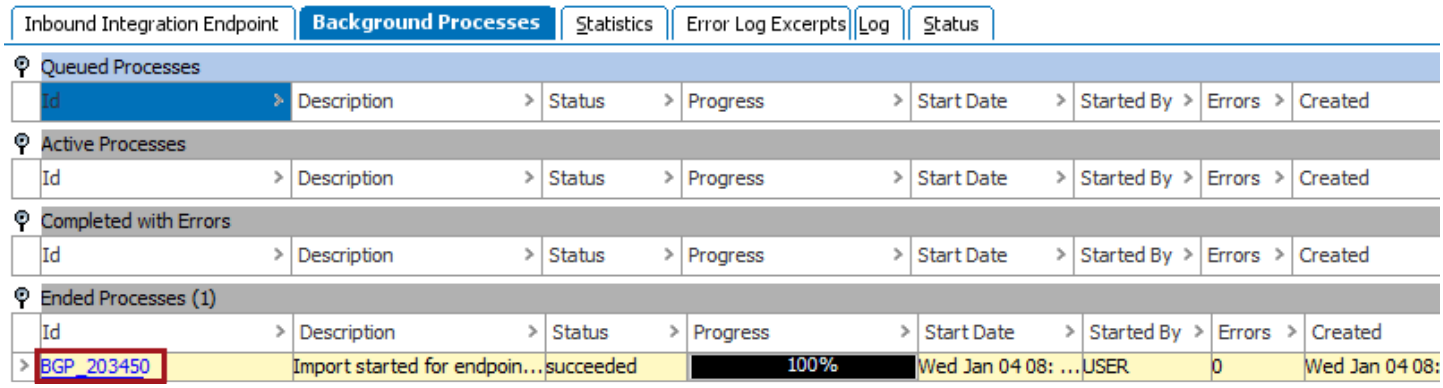
Monitoring an IIEP via Background Process

You can monitor an IIEP from within STEP by way of a background process.

An active inbound integration endpoint uses an associated background process to handle the scheduled invocation of the endpoint. The execution report for this background process is integrated in the workbench endpoint editor and includes invocation information as well as messages logged from the receiver. When using the standard STEP Importer processing engine, the actual import is handled in separate background processes started by the endpoint, as illustrated below.



By default, the started integration endpoint background processes are displayed within the IIEP editor on the Background Processes tab, and are not visible on the BG Processes tab.



For more information, refer to **Maintaining an Inbound Integration Endpoint**.

Background processes are run based on the execution mechanism configured and can be either the priority-based execution or queue setting execution.

Background Process Priority Settings

When the recommended priority-based BGP execution mechanism is configured, waiting BGPs are prioritized for execution based on the priority of the BGP and the created time. The legacy 'Queue for Endpoint' and legacy 'Queue for Endpoint Processes' parameters are not available. Refer to the **Priority** section of the **BG Processes Execution Management** topic in the **System Setup** documentation.

Background Legacy Process Queue Settings

When configuring an inbound integration endpoint that uses the STEP Importer processing engine, you will specify a background process queue for each of the following types of background processes:

- Queue for endpoint
- Queue for endpoint processes

Typically, all background processes of the same type (same process type ID) will use the same queue. However, integration endpoint processes can be tied to different integration endpoints and use different queues.

Legacy Background Process Queue

The main background process uses the default **InboundQueue** background process queue for IIEPs.

However, when you create an integration endpoint, you can specify that you want to use a different IIEP background process queue for the main IIEP background process queue. For more information about this setting, refer to the **IIEP - Configure Endpoint** documentation.

Legacy Background Process Queue Size

The queue size setting can have great impact on processing. The background process queue size property determines the number of background processes can be executed concurrently on the queue. If the size is 1, only one background process can run at a time. If the size is 2, two processes can run concurrently, and so forth.

Important: Although the integration endpoint configuration wizards suggest the number of queues to use, beware that if you create multiple endpoints and do not change the default queue suggestions, your setup could include endpoints that block each other since the queues have a default size of 1.

This is not a big concern for the endpoint processes that, most of the time, are idle and only take up a slot on the queue briefly when the endpoint is invoked (such as when polling a hotfolder for new files). However, it can be an issue for the generated background processes that perform the actual export processing. The decision as to whether different endpoints should use the same or different processing queues and the size of the Queues should therefore be an informed one.

Overwriting default values for a queue causes a new queue to be created with the queue size of 1. Use the following sharedconfig.properties entry to modify the default queue size value:

```
BackgroundProcess.Queue.[name of queue].Size = [number of allowed concurrent processes]
```

Monitoring an IIEP via External System

Integration endpoints can be monitored from external systems using Monitoring Sensors or the REST API.

Performance Data

When using a sensor with details, the 'Performance Data' section of the status page includes the information outlined below.

Note: All of the background processes (BGPs) of the IEP are collected and grouped by the Status, Wait Time, and Run Time in the database.

- **Max Run Time** - largest value in the Run Time column; determines the longest Run Time among all IEP BGPs.
- **Mean Run Time** - average value in the Run Time column; determines the average Run Time among all IEP BGPs.
- **Min Run Time** - smallest value in the Run Time column; determines the shortest Run Time among all IEP BGPs.
- **Total Run Time** - sum of all values in the Run Time column; determines how much time all the IEP BGPs were running.
- **Max Wait Time** - largest value in the Wait Time column; determines the longest Wait Time (in the queue) among all IEP BGPs.
- **Mean Wait Time** - average value in the Wait Time column; determines the average Wait Time (in the queue) among all IEP BGPs.
- **Min Wait Time** - smallest value in the Wait Time column; determines the shortest Wait Time (in the queue) among all IEP BGPs.
- **Number of Failed BGPs** - count of the BGPs in 'completedwitherrors' and 'failed' states.
- **Number of handled requests** - count of the BGPs NOT in the 'running' and 'waiting' states.
- **Number of running BGPs** - count of the BGPs in 'completedwitherrors,' 'failed,' 'aborted,' and 'succeeded' states.
- **Number of waiting BGPs** - count of the BGPs in the 'waiting' state.
- **Number Waiting In Receiver** - IIEP specific; returns the number of messages waiting in the Receiver for handling. Returns -1 if the Receiver is not able to return the number of waiting messages.
- **Total number of BGPs** - count of the BGPs in all possible BGP states ('completedwitherrors,' 'failed,' 'waiting,' 'running,' 'aborted,' and 'succeeded').

Monitoring Sensors

For each integration endpoint created in STEP, a Monitoring Sensor is automatically created. Monitoring Sensors allow external systems to query the status of individual endpoints via HTTP, without authentication. In the following sample URLs, substitute your own system URL and endpoint ID to access your own endpoints.

A Monitoring Sensor can return the following information:

1. **Simple traffic light response** returns OK, WARNING, CRITICAL, or UNKNOWN for inbound endpoints via:

   ```
   http://[System URL]/admin/monitoring/InboundIntegrationEndpointStatus-[Endpoint ID]/status
   ```

2. **Nagios friendly response** returns detailed statistics in Nagios friendly format for inbound endpoints via:

   ```
   http://[System URL]/admin/monitoring/InboundIntegrationEndpointStatus-[Endpoint ID]/nagios
   ```

3. **Full detailed XML response** returns XML with detailed statistics for inbound endpoints via:

   ```
   http://[System URL]/admin/monitoring/InboundIntegrationEndpointStatus-[Endpoint ID]/xml
   ```

REST API Monitoring

An alternative to Monitoring Sensors is to use the REST API, which requires authentication.

1. **Overview of all** configured integration endpoints via:

   ```
   [Host]/restapi/integrationendpoints?context=[Context]&workspace=[Main]
   ```

2. **Detailed statistics** for specific endpoint via:

   ```
   [Host]/restapi/integrationendpoints/[Endpoint ID]?context=[Context]&workspace=[Main]
   ```

3. **Execution report** for specific endpoint via:

   ```
   [Host]/restapi/integrationendpoints/[Endpoint ID]/log?context=[Context]&workspace=[Main]
   ```

4. **Java log entries related to errors in Background Processes** generated by specific endpoint via:

   ```
   [Host]/restapi/integrationendpoints/[Endpoint ID]/errorexcerpts?context=[Context]&workspace=[Main]
   ```

5. **Overview of Background Processes** generated by specific endpoint via:

   ```
   [Host]/restapi/integrationendpoints/[Endpoint ID]/backgroundprocesses?context=[Context]&workspace=[Main]
   ```

Import Error Messages

The following table shows most of the types of error message that you are likely to encounter, with a brief explanation of the most common reason for the error. The most common errors allow users to troubleshoot and then perform the import. There are instances where issues are not related to the import file and are related to the servers, network latency, etc. For these scenarios, contact the Stibo Systems Support team for assistance.

The error messages marked with an asterisk (*) are errors that you will only encounter when you load an XML file. That is because in an XML import, the wizard does not prevent you from entering an invalid value. However, when importing CSV or XLS files, since you often select values from a list, you cannot select an invalid value.

Error Message	Description
Illegal value x for attribute y in product z	The value in the load file does not meet the validity or other constraint that is set up for the attribute.
Not in legal values list	You tried to load an attribute value into an attribute that uses an LOV, and that value does not exist in the LOV, and it is set up not to accept additional entries. This error is not displayed when an LOV is configured to 'Allow Users to Add Values = Yes.'
Attribute not legal for object type	The product's object type is not valid for the attribute. That is, the attribute is not allowed to hold a value for products with the specified object type.
*Illegal unit x for attribute y in product z	In the XML file, you have tried to assign a non-existing unit to an attribute.
Unknown Classification x in product y.	In the XML file, you have probably requested a product-to-classification link using a specified reference type, but the classification you specified is not found.
Unknown Parent product x for product y	You tried to assign a product to a parent ID, but that parent ID cannot be found in the system.
*Unknown product reference target	You attempted to link one product to another via a reference type, but the target product does not exist.
*Unknown attribute	You specified an attribute in the XML file, but that attribute does not exist.

Error Message	Description
Unknown unit	The unit encountered in the load file does not exist in STEP.
Rejected new product x	You set the option to "Reject New Products" in the wizard, and a new product was encountered in the load file.
Illegal unit for attribute	The unit exists but is not valid for the attribute you are trying to load the value into.
*Illegal Object Type for product position	You tried to create a new product via an XML load and you specified an object type that is not a valid choice, given the parent product that you specified.
*Unknown Object Type	You specified an object type (e.g., for a product) in the XML file, but that object type does not exist.
*Unknown reference Type	You specified a reference type (e.g., for a product-to-product reference) in the XML file, but that reference type does not exist.
Insufficient privileges to ...	You do not have permission to carry out one of the following actions (specified in the error message): • set value • set name • create product • move product • modify Object Type • classify product • create product reference For more information, refer to the **Action Sets** topic in the **System Setup** documentation.
Product x contains more than one reference to product y of Type z, reference data will not be imported.	You cannot make a product-to-product reference from the same source product to the same target product more than once.

Error Message	Description
Product x contains more than one reference to asset y of Type z, reference data will not be imported.	You cannot make a product-to-asset reference from the same source product to the same target asset more than once.
*Classification x contains more than one reference to Classification y of Type z, reference data will not be imported.	You cannot make a classification-to-classification reference from the same source classification to the same target classification more than once.
*Reference Type x is not valid for importing a product cross reference from a to b.	In your XML file, you have asked for a product reference that refers to a non-existent reference type.
*Reference Type x is not valid for importing an asset cross reference from a to b.	In your XML file, you have asked for an asset reference that refers to a non-existent reference type.
*Reference Type x is not valid for importing a Classification cross reference from a to b.	In your XML file, you have asked for a classification reference that refers to a non-existent reference type.
*Product x already contains a reference to Classification y but it is invisible in the specified context. The reference and any data on it will not be imported.	You have probably made a product-to-classification link dimension-dependent, and the context you are importing into cannot "view" that link, but it is still there.
Conversion Error	This run time exception error is returned when the XML file is loaded is poorly formed. Download the error file, make the indicated correction, and import the file again.
Conversion Error: Not a Valid STEPXML File (Encoding)	When any encoding type other than UTF-8 is entered in a STEPXML import file, this error is displayed in wizard on the 'Select Format.'
Unknown validator	Importing STEPXML that includes an attribute with a validator type that does not exist in STEP. Validation types with multiple words must replace the space with an underscore. For example, 'Numeric Text' must be written as 'Numeric_Text.'
Context ‘XYZ’ not found	Importing STEPXML where the Context ID is either misspelled or does not exist.

Error Message	Description
No ContextID specified in STEPXML to import	Importing STEPXML without a Context ID.
No WorkspaceID specified in STEPXML to import	Importing STEPXML without a Workspace ID. Workspace ID should always be an editable workspace, often 'Main.' Since the Approved workspace cannot be edited, it cannot receive imports.

Import Error Message Examples

The most common errors encountered during data import refer to an invalid attribute value. Sometimes there is a mismatch with the attribute's validation type, other times there is a mismatch with one of the other setups. For example, the attribute's constraints, such as masks, minimum and maximum values, maximum length, LOVs, the object's own object type, and so on. Therefore, when you encounter an error, look at the attribute setup and review the constraints.

Even if you load attribute values for a object, not all values will be automatically available in STEP. Attributes must be made valid for an object before the values can be accessed via references. STEP will always load attribute values if the object's type is valid for the attribute, and the attribute values meet the validity criteria. But, the attribute itself must be a valid attribute for that object, that is, linked somewhere in the object hierarchy or classification hierarchy where the object resides.

Use the following examples to assist in troubleshooting problems with data imports.

Illegal Unit for Attribute

The tab-delimited input file below had two columns mapped: the first column was mapped both to the Product ID and Product Name, and the second column was mapped to the attribute 'Description.'

```
T100-3526       RED 1/2" STRAIGHT WIDGET SSU05PL
T100-2625      BLUE 1/4" ANGLED GADGET BZT10GL
T100-2827      ORANGE 3/4" ELL WHATSIT PLU04HF
```

When the file was imported, the execution report showed three illegal unit errors:

- Illegal unit "/2" STRAIGHT WIDGET SSU05PL" for attribute Description in product T100-3526 (UnknownUnit). Source: Line=2, Column=Column2.
- Illegal unit "/4" ANGLED GADGET BZT10GL" for attribute Description in product T100-2625 (Unknown Unit). Source: Line=3, Column=Column2.
- Illegal unit "/4" ELL WHATSIT PLU04HF" for attribute Description in product T100-2827 (Unknown Unit). Source: Line=4, Column=Column2.

Upon inspection of the attribute with the name of 'Description', it was found that the assigned data validity check was 'Numeric_Text', and that it had a unit assigned to it: mm. Further, it was not selected as the default unit.

STEP could not handle the input text, since the attribute's validity was defined as being NumericText and had only one legal unit, and that unit could not be matched with anything in the attribute value provided by the import. In this case, either the attribute's setup is incorrect, or a different product attribute should be used that has an appropriate data validity type assigned.

Unknown Parent Product

The tab-delimited input file includes three columns. Column 1 was mapped to both Product ID and Product Name (new products), and Column 3 was mapped to the Parent ID.:

```
T100-3526       RED 1                    Hand Tools
T100-2625      1mm ANGLED GADGET        Hand Tools
T100-2827      ORANGE 3mm               Hand Tools
```

When the file was imported, the execution report showed the following three errors:

- Unknown Parent Product "Hand Tools" for product "T100-3526" (1 product(s) was rejected due to this error). Source: Line=2, Column=Column3.
- Unknown Parent Product "Hand Tools" for product "T100-2625" (1 product(s) was rejected due to this error). Source: Line=3, Column=Column3.
- Unknown Parent Product "Hand Tools" for product "T100-2827" (1 product(s) was rejected due to this error). Source: Line=4, Column=Column3.

STEP could not find the Parent IDs given in Column 3 of the input file, and therefore the new products were rejected. Even though a default Parent ID was selected in the Identify Destination screen of the import wizard, STEP accepts the value for the Parent ID in the input file as an override, and rejects the new product anyway.

A reason for this execution report error could be that one of the input file's columns was mapped to be the Parent ID, and STEP could not find it. If the product existed already, it would not be moved anywhere, it would stay with its current parent. If the product did not exist, it would not be created.

Maximum Length, Illegal Value

The input file had two columns mapped: the first column was mapped both to the Product ID and Product Name (these were new products), and the second column was mapped to the attribute "Manufacturer Part Number."

```
G100-352654     REDD-52626278-YDS-777777-1/FFGH00KK7733249-UHP
J100-267225    REDD-5262627
D100-289827    K7733249-UHP
```

The file was imported and the execution report showed the following three errors:

- Illegal value "REDD-52626278-YDS-777777-1/FFGH00KK7733249-UHP" for attribute Manufacturer Part Number in product G100-352654 (Length of the value exceeds max length of domain Is trying to insert a value with '46' characters into a domain that has a maximum length of '40' characters.). Source: Line=10, Column=Column2.
- Line= at line 9: : Length of the value exceeds max length of domain Is trying to insert a value with '46' characters into a domain that has a maximum length of '40' characters.
- Line= at line 9: : Array operation failed (1 times): Error during array operation: ORA-20291: Parent key not found (valuemap.edgeid) ORA-06512: at "STEPSYS.VALUEMAP_BEF_INS", line 18 ORA-04088: error during execution of trigger 'STEPSYS.VALUEMAP_BEF_INS' ; SQL return code=20291; SQL return code=0

These errors are reporting a single problem. The first and second errors say that one of the imported part numbers had a value in the Manufacturer Part Number column that was too long, as defined by the attribute's setup. You are given the part number, the attribute name, and the value that was trying to be imported. The third error message is useful for a programmer.

Note: This type of error does not prevent the new part number from being created. The value for the Manufacturer Part Number, however, is left blank.

Out of Range, Illegal Value

This is another example of an illegal value error:

- Illegal value "250" for attribute Focal Length (ft) in product G100-352654 (Value error: Value '250' ' is out of range). Source: Line=9, Column=Column2.
- Line= at line 8: : Value error: Value '250' ' is out of range
- Line= at line 8: : Array operation failed (1 times): Error during array operation: ORA-20291: Parent key not found (valuemap.edgeid) ORA-06512: at "STEPSYS.VALUEMAP_BEF_INS", line 18 ORA-04088: error during execution of trigger 'STEPSYS.VALUEMAP_BEF_INS' ; SQL return code=20291; SQL return code=0

This group of three error messages are reporting a single problem. The first two messages indicate that a value of 250 is too large for the maximum value allowed for the attribute.

Not a Number, Illegal Value

The following tab-delimited input file had two columns mapped: the first column was mapped to the Product ID (the products already existed in STEP), and the second column was mapped to an attribute called 'Hole Diameter.'

```
G100-352654     1-1/2
J100-267225     2-3/4
D100-289827    17-3/16
```

The file was imported and the execution report showed the following nine errors (three sets):

Set 1

- Illegal value "1-1/2" for attribute Hole Diameter in product G100-352654 (Value error: Value '1-1/2' is not a number). Source: Line=9, Column=Column2.
- Line= at line 8: : Value error: Value '1-1/2' is not a number
- Line= at line 8: : Array operation failed (1 times): Error during array operation: ORA-20291: Parent key not found (valuemap.edgeid) ORA-06512: at "STEPSYS.VALUEMAP_BEF_INS", line 18 ORA-04088: error during execution of trigger 'STEPSYS.VALUEMAP_BEF_INS' ; SQL return code=20291; SQL return code=0

Set 2

- Illegal value "2-3/4" for attribute Hole Diameter in product J100-267225 (Value error: Value '2-3/4' is not a number). Source: Line=14, Column=Column2.
- Line= at line 13: : Value error: Value '2-3/4' is not a number
- Line= at line 13: : Array operation failed (1 times): Error during array operation: ORA-20291: Parent key not found (valuemap.edgeid) ORA-06512: at "STEPSYS.VALUEMAP_BEF_INS", line 18 ORA-04088: error

during execution of trigger ‘STEPSYS.VALUEMAP_BEF_INS’ ; SQL return code=20291; SQL return code=0

Set 3

- Illegal value "17-3/16" for attribute Hole Diameter in product D100-289827 (Value error: Value ‘17-3/16’ is not a number). Source: Line=19, Column=Column2.
- Line= at line 18: : Value error: Value ‘17-3/16’ is not a number
- Line= at line 18: : Array operation failed (1 times): Error during array operation: ORA-20291: Parent key not found (valuemap.edgeid) ORA-06512: at "STEPSYS.VALUEMAP_BEF_INS", line 18 ORA-04088: error during execution of trigger ‘STEPSYS.VALUEMAP_BEF_INS’ ; SQL return code=20291; SQL return code=0

Each set of three errors addresses one of three problems. That is, different information is reported for the same error in three individual messages. The first message in each group is what is most useful to the end user. In the above series, notice that the first lines that indicate that the fractional value in the input file is not allowed for the attribute 'Hole Diameter' because the value 'is not a number.' In this case, the attribute’s validation type was set to Number, which does not allow fractional values.

This is another example:

- Illegal value "Hand tools" for attribute Focal Length (ft) in product J100-267225 (Value error: Value ‘Hand tools’ is not a number). Source: Line=14, Column=Column2.
- Line= at line 13: : Value error: Value ‘Hand tools’ is not a number
- Line= at line 13: : Array operation failed (1 times): Error during array operation: ORA-20291: Parent key not found (valuemap.edgeid) ORA-06512: at "STEPSYS.VALUEMAP_BEF_INS", line 18 ORA-04088: error during execution of trigger ‘STEPSYS.VALUEMAP_BEF_INS’ ; SQL return code=20291; SQL return code=0

The first message in the group tells you that the attribute value of 'Hand tools' in the input file is not allowed since the attribute is set up to expect a number.

And another example:

- Illegal value "50,70" for attribute Focal Length (ft) in product D100-289827 (Value error: Value ‘50,70’ is not a number). Source: Line=19, Column=Column2.
- Line= at line 18: : Value error: Value ‘50,70’ is not a number
- Line= at line 18: : Array operation failed (1 times): Error during array operation: ORA-20291: Parent key not found (valuemap.edgeid) ORA-06512: at "STEPSYS.VALUEMAP_BEF_INS", line 18 ORA-04088: error during execution of trigger ‘STEPSYS.VALUEMAP_BEF_INS’ ; SQL return code=20291; SQL return code=0

The first message in the group tells you that the attribute value of '50,70' in the input file does not match the attribute’s validation type of Number, which does not allow commas.

Object Existed with Another Object Type

The following tab-delimited input file had two columns mapped: the first column was mapped to the Product ID (the products already existed in STEP), and the second column was mapped to an attribute called "MFOB."

```
G100-352654    Yes
J100-267225    No
D100-289827    Maybe
```

The file was imported and the execution report showed the following three warnings and one error:

- Line=2: Product ‘G100-352654’ existed with another object type - supplied object type not set
- Line=3: Product ‘J100-267225’ existed with another object type - supplied object type not set
- Line=4: Product ‘D100-289827’ existed with another object type - supplied object type not set
- Illegal value "Maybe" for attribute MFOB in product D100-289827 (Not in legal values list). Source: Line=4, Column=Column2.

The first three warnings say that the products already existed in STEP with the object type of 'Product.' When the file was imported, the Identify Destination screen was skipped, and the default object type that was set in that screen was different from Product. STEP reported that it did not change the object type of the products.

The final message is an error that says that the attribute MFOB used an LOV and the value in the input file was not valid. Further investigation showed that LOV had only two valid values: Yes and No. Additionally, the LOV was set up so that no modifications could be made to the LOV by loading in data. So the value "Maybe" was rejected.

Unknown Classification

The following tab-delimited input file had two columns mapped: the first column was mapped to the product ID (the products already existed in STEP), and the second was mapped as a classification ID.

```
G100-352654    Hand tools
J100-267225    Hand tools
D100-289827    Hand tools
```

The file was imported and the execution report showed the following three errors:

- Unknown Classification "Hand tools" in product "G100-352654" (Classification ‘Hand tools’ does not exist in the import workspace). Source: Line=6, Column=Column2.
- Unknown Classification "Hand tools" in product "J100-267225" (Classification ‘Hand tools’ does not exist in the import workspace). Source: Line=10, Column=Column2.
- Unknown Classification "Hand tools" in product "D100-289827" (Classification ‘Hand tools’ does not exist in the import workspace). Source: Line=14, Column=Column2.

In this case, the import file requested that the products be linked to a classification folder, but STEP could not find a classification with the specified ID 'Hand tools.'

Not Privileged to Create Object

A file was imported and the execution report showed the following error:

- Line=2: Not privileged to Create product

Two scenarios can cause this error message:

1. Importing new products when the import user does not have the privilege to create new products. The import user must have all access required for the action to be performed during the import.
2. Importing a product with a product-to-product Reference Type by mapping one column of data to the Product ID and another column to a Reference, while the target product of the reference does not already exist in STEP. A reference can only be imported when the target already exists.

Unknown Asset ID

A file was imported and the execution report showed the following error:

- Line=2: Unknown Asset ID: AC200-627

This error is reported when attempting to link a Product ID to an Asset ID, but the asset does not exist in STEP.

Note: The Asset ID (not the Asset Name) is required in an import file.

Array Operation Failed

A file was imported and the execution report showed the following error:

- Line= at line 8: : Array operation failed (1 times): Error during array operation: ORA-20042: Value rejected: Attribute Gap not valid for this usertype ORA-06512: at "STEPSYS.VALUEMAP_BEF_INS", line 41 ORA-04088: error during execution of trigger 'STEPSYS.VALUEMAP_BEF_INS' ; SQL return code=20042; SQL return code=0

This error is reported when attempting to load an attribute value (not shown in the error) to an attribute that is not legal for the product's object type. Upon further investigation, we found that the product's object type was 'Product.' The error message is saying that the attribute that has the ID of 'Gap' is not valid for products with the object type of 'Product.' Since the value is invalid, it is ignored.

Optimistic Locking Errors

When a file was imported, the below error was displayed in the execution report:

- OptimisticVerificationException: Optimistic locking errors were detected when flushing to the data store. This indicates that some objects were concurrently modified in another transaction. Failed objects: [com.stibo.core.persistence.ProcessPO@33bf33bf: BGP_3604361] [java.util.ArrayList]

This error occurs when a user is trying to perform an import by modifying an attribute value for an existing product in STEP and at the same time there is another user who has access to the same product and is modifying the attribute value. In this scenario, the product is locked for any modification and the import fails.

The Transaction has been Rolled Back: Unique Constraint

A file was imported to create a new product and the below error was thrown:

- Caught FatalDataStoreException at Mon Jun 06 13:10:16 CEST 2016: The transaction has been rolled back. Refer to the nested exceptions for details on the errors that occurred., caused by SQLException: ORA-00001: unique constraint (STEPSYS.NODE_NODEID_UIX) violated {prepstmnt 523703057 INSERT INTO

NODE (CHECKID, NAME, NODEID, NODETYPE, USERTYPE, USERTYPEID) VALUES (?, ?, ?, ?, ?, ?) [reused=5]} [code=1, state=23000]

The above error was thrown when a new product was being created but, the ID provided to create the new product already existed in STEP. ID's for objects that are created in STEP is always Unique and if a value for an ID already exists then repeating the same again is not accepted in STEP.

Note: ID's used for one product cannot be used again for another product, however, the same ID can be used for to create a Classification folder.

Import errors are shown in the Background processes under the Execution report and the number of errors and warning are also displayed as shown below:

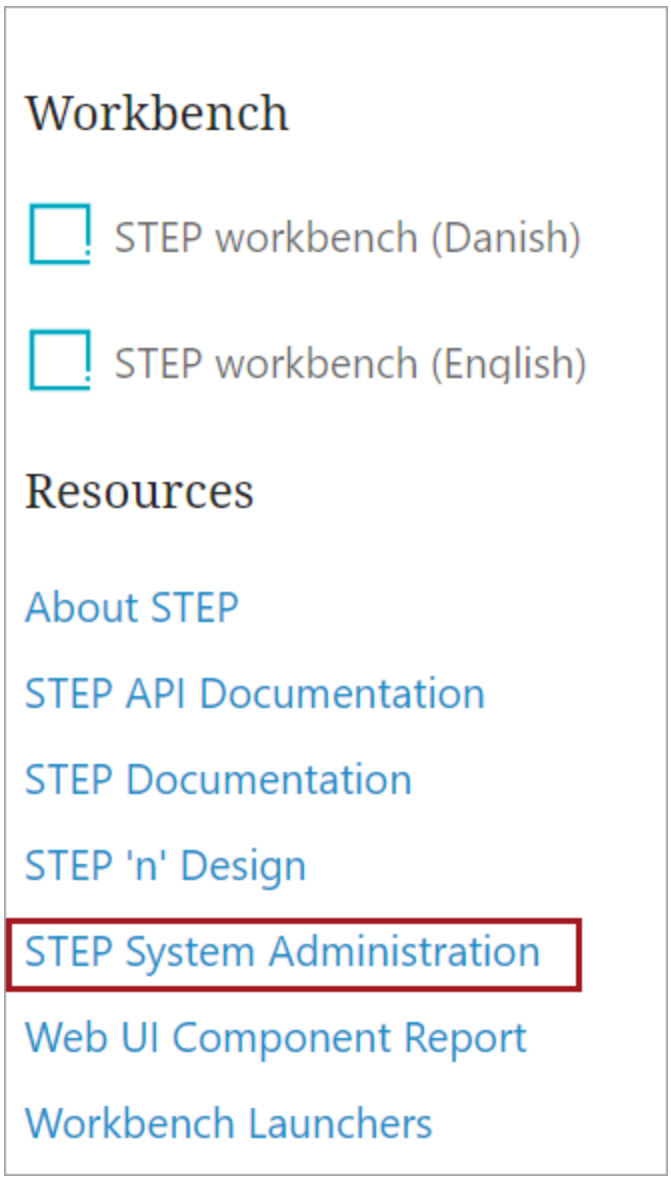
Description	Importing
Execution Server	doc-rel
Progress	100%
Status	completed with errors
Created	Sun Feb 05 06:49:19 EST 2017
Started	Sun Feb 05 06:49:20 EST 2017
Finished	NA
Processing Time	0 m 2 s
Time in Queue	0 m 1 s
# of warnings	0
# of errors	1

Execution Report

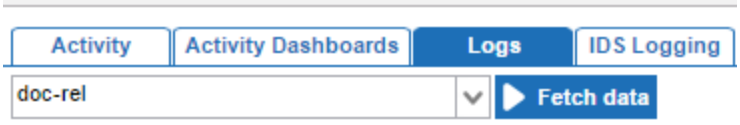
```
 1 Retrieval started (Sun Feb 05 06:49:21 EST 2017)
 2 Retrieved 8817 bytes (Sun Feb 05 06:49:21 EST 2017)
 3 Conversion started (Sun Feb 05 06:49:21 EST 2017)
 4 Converted 6 objects (Sun Feb 05 06:49:24 EST 2017)
 5 Logged on
 6 Mapping started (Sun Feb 05 06:49:24 EST 2017)
 7 Mapping completed (Sun Feb 05 06:49:24 EST 2017)
 8 Import Started (Sun Feb 05 06:49:24 EST 2017)
 9 Logged On
10 Using import mode "domain"
11 Starting first import pass (creating system setup objects)
12 Starting second import pass (importing data)
! 13 Row 6, Column : The attribute with ID 'ShortItemDescription' isn't valid for object product with ID '888264
14 Imported 2 new products, 0 new classifications, 0 new entities and 0 new assets.
15 Processed 4 existing products, 0 existing classifications, 0 existing entities and 0 existing assets.
16 Skipped 0 products, 0 classifications, 0 entities and 0 assets.
17 Deleted 0 products, 0 classifications, 0 entities and 0 assets.
18 Found 0 warnings
19 Found 1 errors
20 Import completed (Sun Feb 05 06:49:26 EST 2017)
21 Error file generation started (Sun Feb 05 06:49:26 EST 2017)
22 Error file with 1 object(s) generated (Sun Feb 05 06:49:30 EST 2017)
```

Similarly, the execution report can also be viewed in the System Administration page by following the below steps:

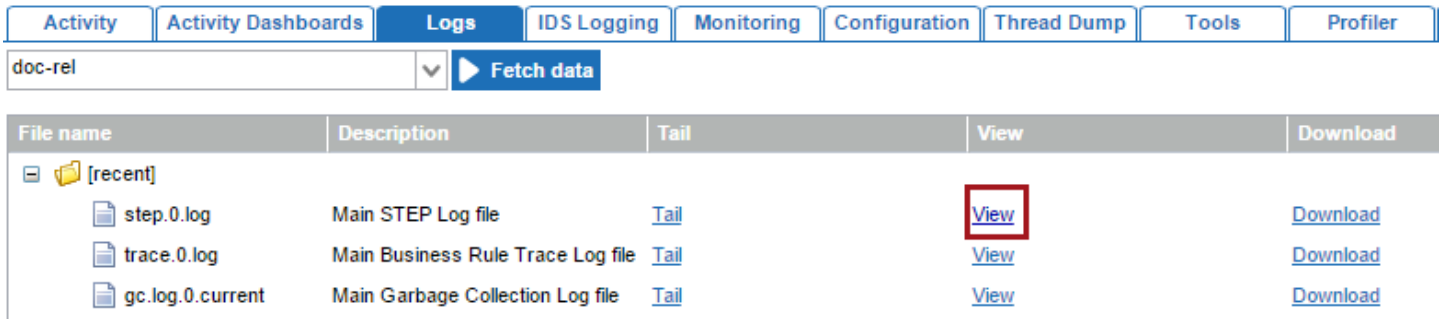
1. Launch the Start Page
2. Click on the System Administration link



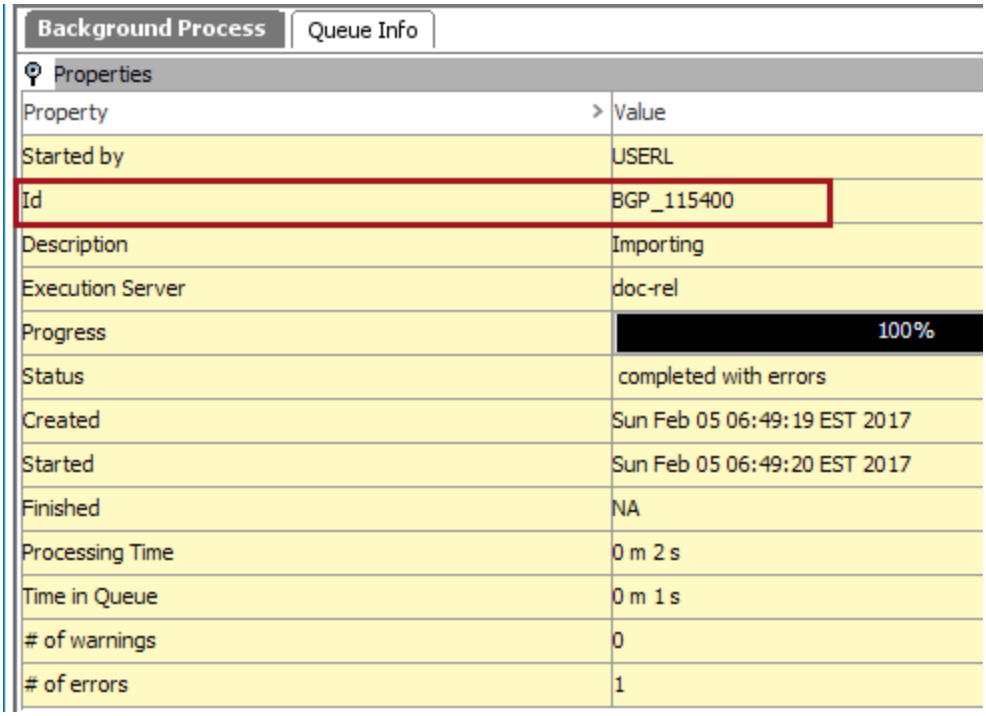
3. Once logged in, click on the Logs Tab
4. Select the Server in the dropdown list and click on 'Fetch Data'



5. Expand [recent] and then click on view option for step.0.log file



6. A new window will open with the log details shown
7. Copy the background process ID from STEP



8. Press Ctrl + F on the keyboard and search for the background process ID to find the complete log detail with more additional details.

```
Submitting bg-process (BGP_115400) with template (stibo.Importer) to queue (IN)
2017/02/05-06:49:20     2c8     com.stibo.servicemanager.beans.DefaultBackgroundProcessInvocationImpl run INFO: Start
handling of bg-process (BGP_115400) (recovery=false)
2017/02/05-06:49:21     2c8|USERL|BGP   com.stibo.util.unstable.plugin.DefaultPlugin init WARNING: Unable to find
parameter name: ValueSubstitutionAssetIDName
2017/02/05-06:49:21     2c8|USERL|BGP   com.stibo.util.unstable.plugin.DefaultPlugin init WARNING: Unable to find
parameter name: WordSubstitutionAssetIDName
2017/02/05-06:49:21     2c8|USERL|BGP   com.stibo.util.unstable.plugin.DefaultPlugin init WARNING: Unable to find
parameter name: LibraryAttributeTransformationIDName
2017/02/05-06:49:25     2c8|USERL|BGP   com.stibo.systemconfig.ConfigUtil getPropertyTypeCheck WARNING
The property Import.IsMarkNodeFilterValuesModification is not documented, check
com.stibo.core.domain.impl.importer.ValueFilterHandlerDomain.<clinit>(ValueFilterHandlerDomain.java:38) and read
http://confluence.stibo.com/display/RD/STEP+5+configuration+properties
2017/02/05-06:49:30     2c8|USERL|BGP   com.stibo.core.domain.impl.backgroundprocess.BackgroundProcessImpl updateStatus
INFO: Setting status on succeeded bg-process (BGP_115400)
2017/02/05-06:49:30     2c8|USERL|BGP   com.stibo.core.domain.impl.backgroundprocess.BackgroundProcessImpl$12
lambda$run$0 INFO: Succeeded updating status of bg-process (BGP_115400)
2017/02/05-06:49:30     2c8     com.stibo.servicemanager.beans.DefaultBackgroundProcessInvocationImpl run INFO: Finished
handling of bg-process (BGP_115400)
2017/02/05-07:00:16     4f      com.stibo.systemconfig.ConfigUtil getPropertyTypeCheck WARNING
The property Log.ConfigurationPackagesLogRoot is not documented, check
com.stibo.admin.FileUtil.getConfigurationPackagesLogFileNames(FileUtil.java:265) and read
http://confluence.stibo.com/display/RD/STEP+5+configuration+properties
2017/02/05-07:00:53     4e      com.stibo.systemconfig.ConfigUtil getPropertyTypeCheck WARNING
The property Log.ConfigurationPackagesLogRoot is not documented, check
```

As explained above, there are common errors and the execution logs can be viewed and then correction can be made accordingly and then the file can be imported. There are other errors which can occur for different reason based on the scenario. If the errors are not correctable then those report logs can be collected and then reported to the Stibo Systems Support team to have the issue fixed.

Additional Information About Inbound Integration Endpoints

This information is relevant to understanding additional functionality available for implementing and using an inbound integration endpoint. Included are details on the following features:

- Export inbound integration endpoint definition for comparison purposes in an external source control system for comparison purposes as described in the **Configuration Management** documentation.
- Resolve failed background processes as described in **Handling Failed IIEP Background Processes**.
- Understand each element of an IIEP as described in **Inbound Integration Endpoint Structure**.
- Understand how background processes will be generated, how the messages will be processed, and what will happen upon failure as described in **Integration Endpoint Transactional Settings**.
- Parallel imports involving multiple references on object types as described in **Reference Target Lock Policy on Object Types**.

Handling Failed IIEP Background Processes

Errors can cause an IIEP to go into a 'Failed' state ( ) or into a 'Failed (retrying)' state ( ), based on the type of error encountered and the settings on the Error Handling and Reporting step of the IIEP wizard.

For details on automatically handling connection errors which result in a 'Failed (retrying)' state ( ), refer to the **IIEP - Error Handling & Reporting** topic and the **Running an Inbound Integration Endpoint** topic.

When an IIEP is in a Failed state ( ), you can reactivate it using Disable / Enable or Resume.

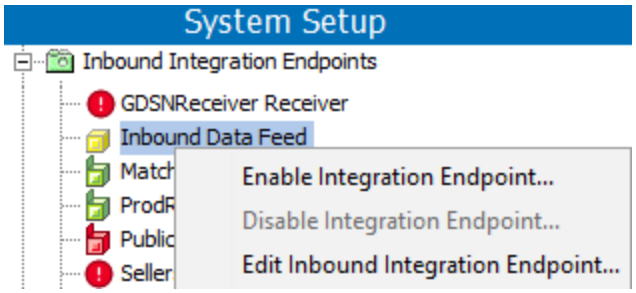
- If a background process fails and the processing of data is set to **strict**, you must restart the endpoint before the endpoint is able to start another background process.
- If a background process fails and the processing of data is set to **chain**, the remaining background processes that belongs to the chain will also fail. You must correct the failed background process before the endpoint is able to start another background process that belong to the same chain. The endpoint will, however, continue to process data from other chained background processes.

For more information about the strict and chain settings, refer to **Integration Endpoint Transactional Settings** documentation.

Disable and Enable a Background Process

Setting an integration endpoint to **Disable** and then **Enable** *clears* the log file.

- Right-click the endpoint and select **Disable Integration Endpoint**.
- Right-click the endpoint again and select **Enable Integration Endpoint**.

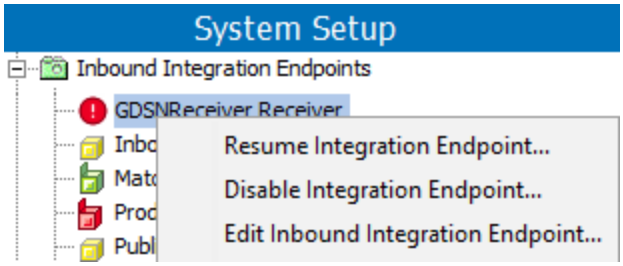


Resume a Background Process

Resuming an integration endpoint *does not clear* the log file

Note: Whenever the STEP system is patched or restarted, the main background process is automatically restarted. The integration endpoint log still exists, and continues to show an overview of the started background processes.

- Right-click the endpoint and select **Resume Integration Endpoint**.



Resolving a Failed Background Process

Errors can happen before the background process starts, and also after it is running. Error handling and reporting can be configured when creating an IIEP, as described in the **IIEP - Error Handling & Reporting** topic.

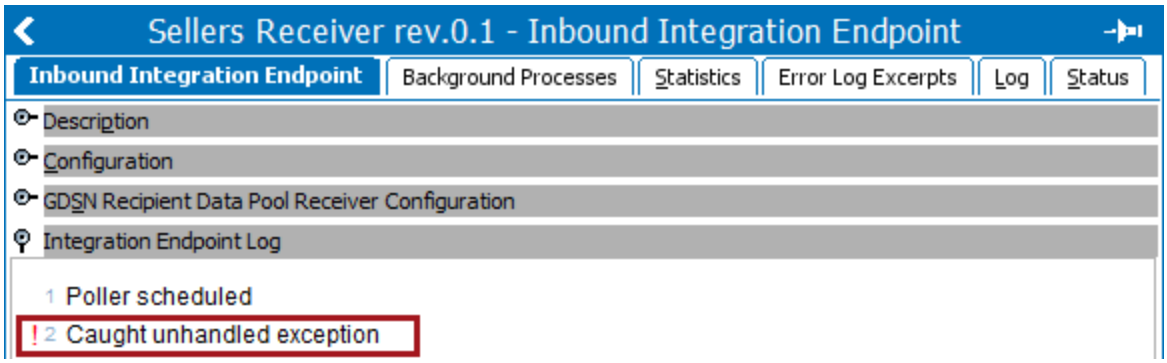
The levels of failure reported in STEP are errors and warnings.

- Warnings are minor errors that do not stop the import. A warning status allows the process to complete and data validation to be performed elsewhere in STEP. For example, specifying a parent that does not exist, or an invalid value for a validated attribute.
- Errors stop the import and prevent the endpoint from processing more data until it is restarted once the error is resolved. Errors are used in cases such as when a malformed XML, file is received, or when an endpoint expects XML but an Excel file is provided.

User friendly error logging is included on the Execution Report of the background process (accessible from the Background Processes tab of the IIEP), and more technical detailed logging is in the step.0.log (accessible from the system Start Page via the STEP System Administration button).

IIEP Fails Before the BGP Starts

In cases where the IIEP fails before the BGP is initiated, a report is available in Integration Endpoint Log on the Inbound Integration Endpoint tab. Refer to this log for information to make needed corrections.



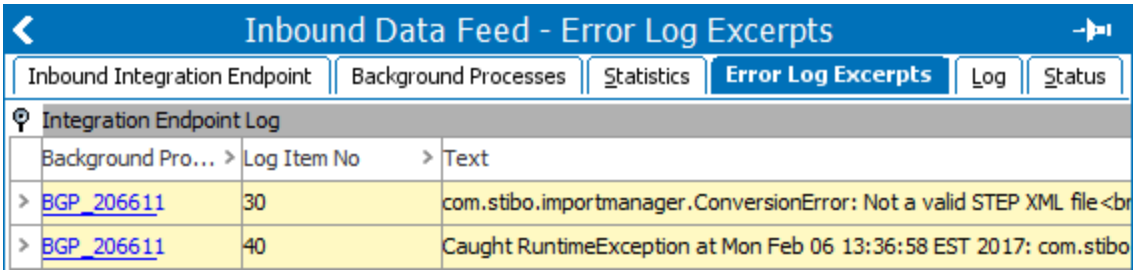
IIEP Fails After the BGP Starts

The Error Log Excerpts tab includes details about failures within the background process.

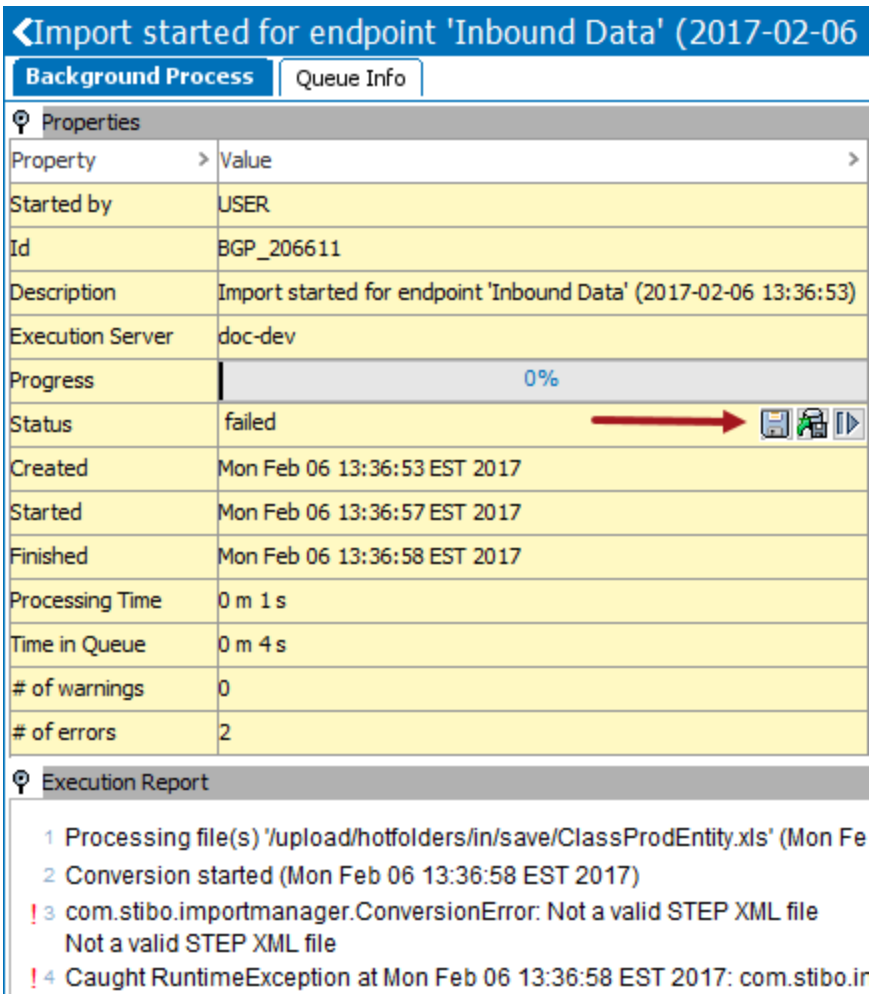
1. In **System Setup** select the relevant integration endpoint, and then click the **Error Log Excerpts** tab to display a list of failed background processes and the description of the failure.

   You can also verify if a process has been suspended because it is dependent on a failed process.

2. In the **Background Process** column, hover over the background process to correct and click the **BGP** link.



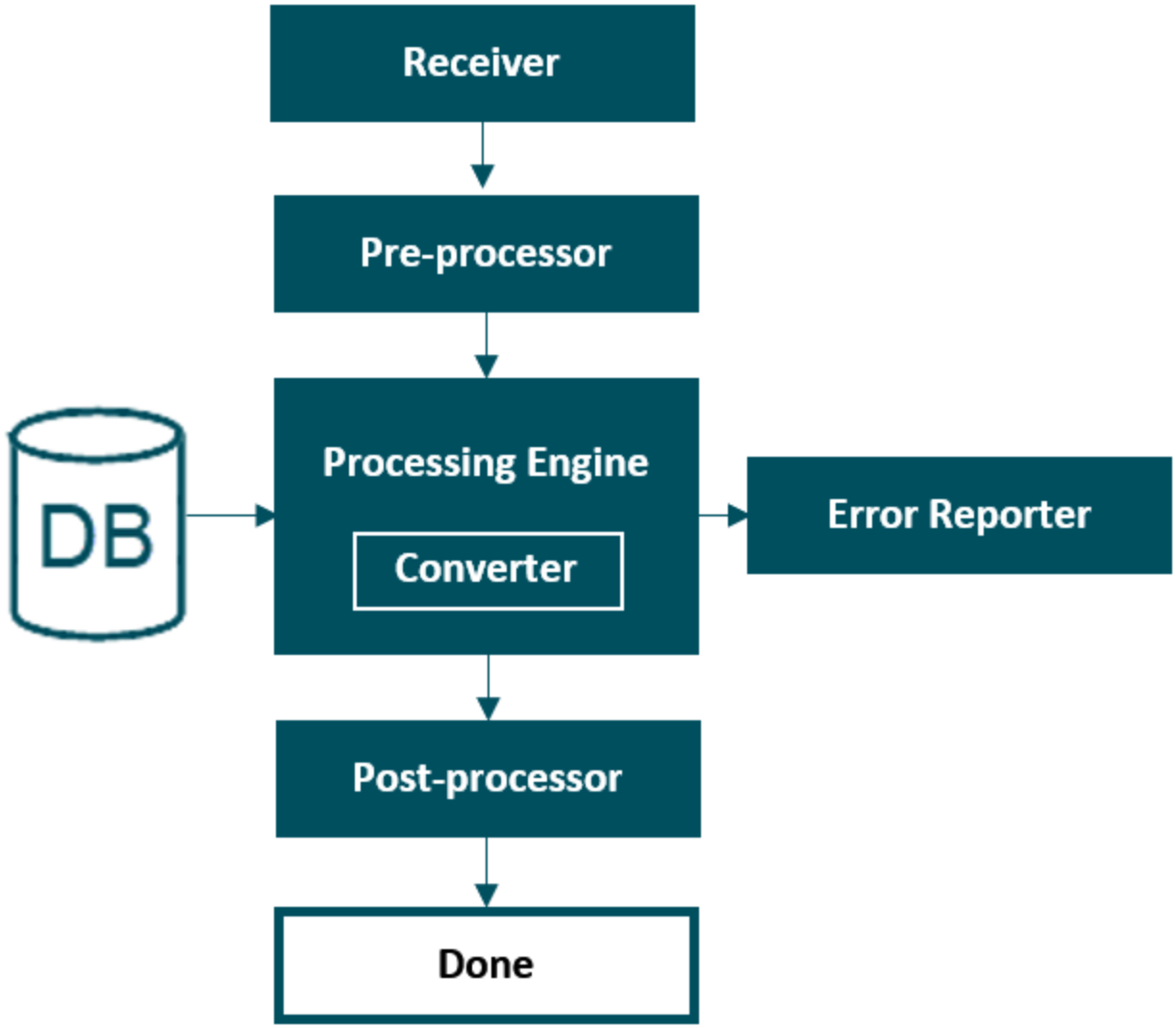
   The Background Process editor displays.



3. In the Background Process editor **Execution Report**, view the progress of the background process, and where it failed.
4. In the Properties area **Value** column, click the Save button ( ) near the failed status and save the file locally.
5. Open the file in a relevant editor and make the required changes.
6. After completing the changes, upload the file back to the same background process. In the Background Process editor on the Properties flipper, review the Status parameter Value column and click the Upload button ( ) near the failed status.
7. Restart the process in the Properties area via the Status parameter Value column. Click the Restart button ( ) near to the failed status. The background process is restarted.
8. If the background process is part of a batch of chained processes, restart the other processes that are part of the chain.

Inbound Integration Endpoint Structure

The IIEP functionality has been created to allow for easy extensions and customizations, and is a plugin framework with interchangeable parts. The graphic below illustrates the structure and the colored boxes represent the interchangeable parts, which are described below.



Receiver

The receiver is responsible for retrieving files or messages and passing them on to the succeeding plugins. The available receiver options are described in the **Receiver Methods** section of the **IIEP - Choose Receiver** topic.

Pre-processor

A pre-processor (for example, an XML normalizer) has access to the file or message delivered by the receiver and can manipulate or discard it. STEP does not include any predefined pre-processors. For more information, refer to the **IIEP - Configure Pre-processor** documentation.

Processing Engine / Converter

The processing engine is responsible for performing the actual data import. The STEP Importer and Asset Importer engines are available with core functionality. If a format not supported in the standard STEP Importer is required, the processing engine can be replaced via an extension. Most often, however, creating a converter for the STEP Importer is sufficient for such cases. This approach also has the benefit that the same converter is available for manual imports. For more information, refer to the **IIEP - Configure Processing Engine** documentation.

Error Reporter

The Send Error Report option sends an alert email to specified addresses if errors or warnings occur when files or messages are handled by the processing engine. For more information, refer to the **IIEP - Error Handling & Reporting** documentation.

Post-processor

A customized post-processor has access to import events (information about what has changed) and, based on the events, can trigger any required system change. For more information on events, refer to the **Events** documentation in the **System Setup**.

By default, STEP does not include any pre-configured post-processor, since post-processor logic is also typically implemented via business rules referenced from the processing engine configuration.

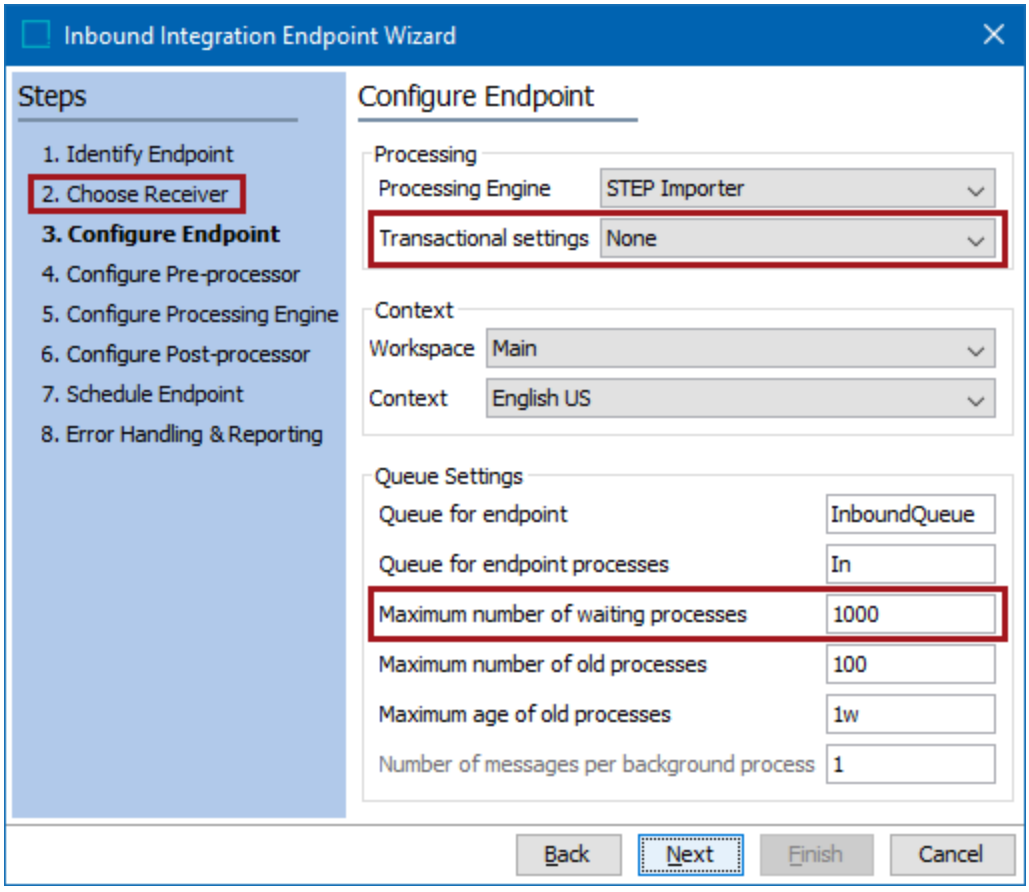
Integration Endpoint Transactional Settings

For both inbound and outbound integration endpoints, 'transactional settings' specify how background processes will be generated, how the messages will be processed, and what will happen upon failure.

After configuring transactional settings on the integration endpoint, configuration for queue size and parallel are set in the sharedconfig.properties file as defined in the **Background Processes and Queues** topic of the **System Setup** documentation.

Inbound Integration Endpoints

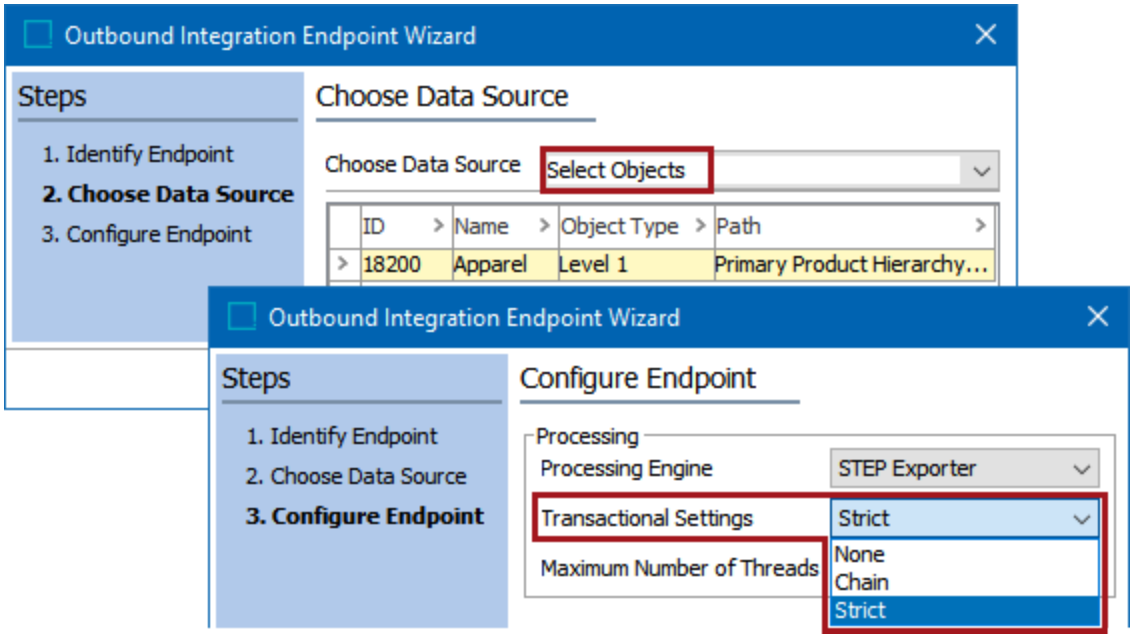
In the IIEP wizard, background process and error handling are managed by three (3) elements: the selected Receiver (on the Choose Receiver step), the 'Transactional settings' parameter, and the 'Maximum number of waiting processes' parameter (both on the Configure Endpoint step). These elements also work together to determine what additional options are available in the wizard, as defined below.



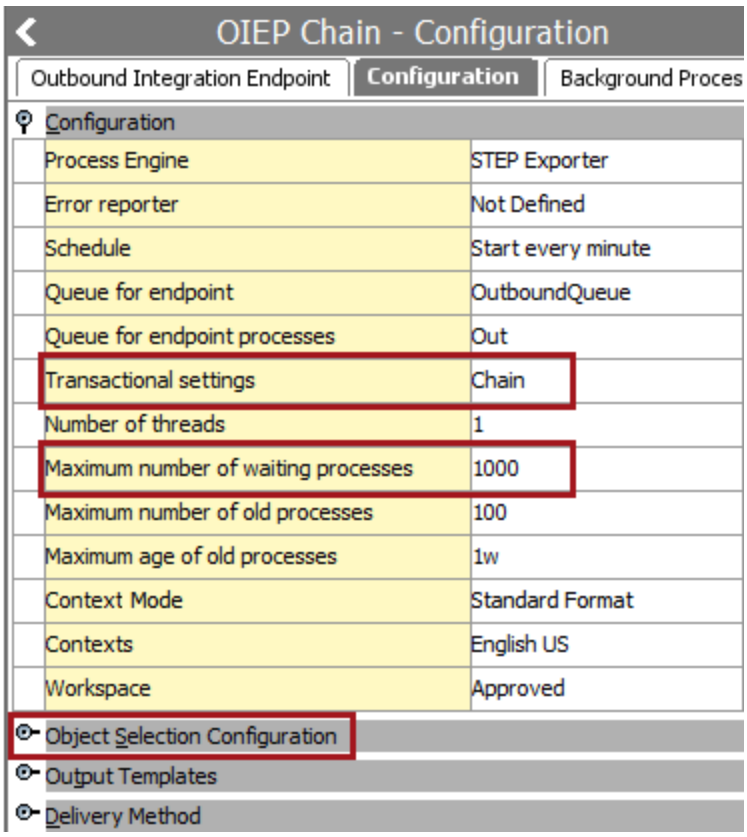
For more information, refer to the **IIEP - Configure Endpoint** documentation.

Outbound Integration Endpoints

While creating an outbound endpoint using the wizard, the Choose Data Source step determines if the transactional settings can be set in the wizard. As defined below, selecting a Select Objects data source provides several options for transactional settings; selecting an Event Queue data source requires the 'Strict' setting.



For Select Objects OIEPs, the transactional settings can be modified in the OIEP editor, on the Configuration tab. The 'Transactional settings' parameter and the 'Maximum number of waiting processes' parameter determine how background processes and errors are managed.



For more information on select objects OIEPs, refer to the **OIEP - Select Objects - Configure Endpoint** topic.

For details on the 'Transactional settings' and 'Maximum number of waiting processes' parameters, refer to the **OIEP - Configuration Flipper** topic.

For event-based OIEPs, only the 'Strict' transactional setting is allowed. 'Strict' ensures that an event in the STEP event queue is not removed until the endpoint has successfully delivered the result of the event to the external system. If event processing fails (i.e., the external system is not running and therefore does not accept any data), the endpoint is disabled but events are not removed from STEP. Once the error state is fixed by a user, the endpoint can be resumed; events waiting to be sent to the external system are processed, and the data history is retained.

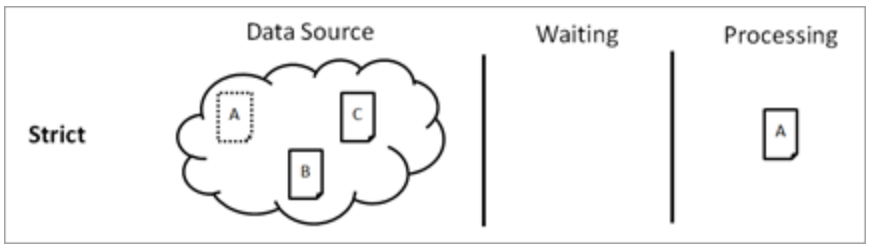
For more information on event-based OIEPs, refer to the **OIEP - Event-Based - Configure Endpoint** topic.

Transactional Settings Options

When available, the possible transactional settings are: Strict, Chain, and None.

Strict

Using the 'strict' transactional setting processes data in a strict order, using one background process at a time. When one background process completes successfully, the endpoint starts the next background process. If a background process fails with an error, or if it cannot deliver a file, the next process is not started, and the 'Failed' status is set. This is the most common setup, unless there is a reason to use Chain or None (for assets).



The queue size of a background process with strict transactional settings must be one (1). A strict transactional setting does not allow multiple background processes to work in parallel / concurrently regardless of the queue size. By definition, no new background processes can be generated while one is active.

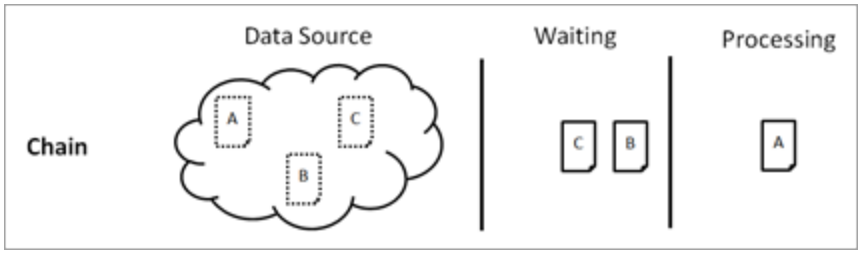
Strict is allowed for all inbound receiver methods and for outbound select data endpoints but is required for outbound event-based endpoints.

Chain

Using the legacy 'chain' transactional setting processes data in batches of chained background processes. With this setting, a batch is generated each time the endpoint polls for data and finds more than one message or file that has not yet been processed.

Note: Generally, 'chain' is not recommended due to slower processing speeds compared to 'strict' with similar features for enforcing order of import files.

The **Maximum number of waiting processes** parameter allows you to specify the maximum number of background processes with the status 'Waiting' that are allowed. If a background process in a batch fails, the remaining background processes in the batch will also fail. However, the endpoint remains active with the status 'Running' and continues to process data in the next batch of chained background processes. A chain endpoint stops if a file cannot be delivered.

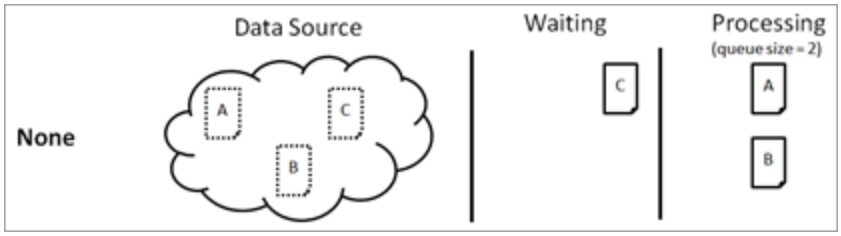


The queue size of a background process with chain transactional settings must be one (1). A chain transactional setting does not allow multiple background process to work in parallel / concurrently regardless of the queue size. By definition, no new background processes can be generated while a background process is active. The default 'Maximum number of waiting processes' is 1,000.

Although chain is allowed for all inbound receiver methods and for outbound select data endpoints, it is not recommended as it results in slower processing times.

None

Using the 'none' transactional setting processes data concurrently, without any transactional restrictions or data dependencies. This is useful, for example, when processing assets. The default queue size of a background process with transactional settings of 'none' is one (1). Data is not processed in a strict order and if one background process fails, the endpoint continues to process data in the next background process in the queue.



None is not allowed for the inbound 'Hotfolder Using File Sequence Receiver' or 'Hotfolder Using Meta Files Receiver' methods but can be set for other inbound endpoint receivers. It can also be set for outbound select objects endpoints.

Important: Switching the 'Transactional setting' parameter after an OIEP has been created will also change the 'Maximum number of waiting processes' parameter. If changing from Strict to Chain or None, the 'Maximum number of waiting processes' will stay at one (1). Users will need to go in and change that number to higher than one (1) in order to attain Chain or None processing. If the OIEP was set up with the Chain or None settings, then the 'Maximum number of waiting processes' will continue to display 1000 when changed to Strict until the workbench is refreshed.

Reference Target Lock Policy on Object Types

The 'Reference Target Lock Policy' parameter for an asset, entity, classification, and product object type manages how objects are locked while they are being referenced.

When long-running transactions cannot be simplified easily, consider setting the 'Reference Target Lock Policy' parameter to 'Relaxed' on the object types to which the long transaction applies.

Strict or Relaxed

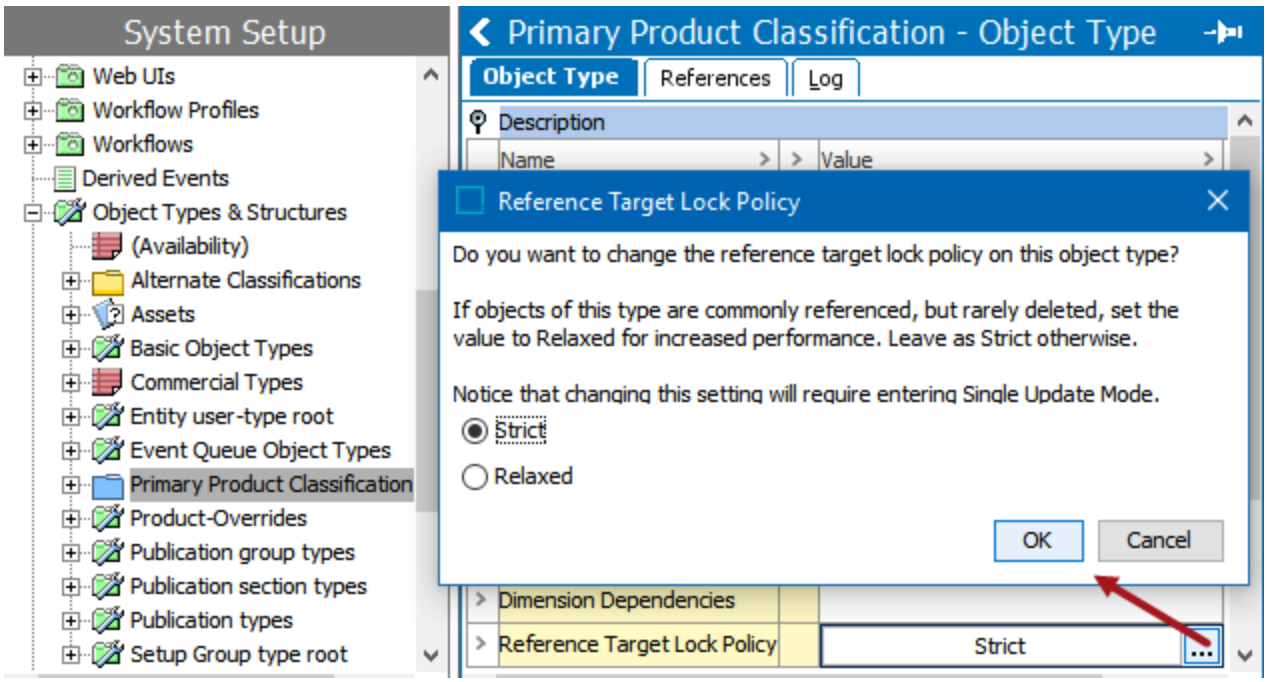
The 'Strict' setting is default for the 'Reference Target Lock Policy' parameter. When a reference between two objects is being created, the target object is locked to ensure that it is not being deleted while the reference is being created. This means that only one process or user can edit it at a time. Often, deletion of the object is not a deletion from the database, but a deletion due to revision control when the insert of a new history entry occurs.

When using the Optimistic Locking option, a 'Strict' setting can result in an issue running parallel imports where the first import locks the object type being referenced, and the second import eventually stops running because it cannot access the locked object. Since STEP continues to retry the import, this can negatively impact inbound feeds.

The 'Relaxed' setting uses a less restrictive lock (a shared 'user lock') on the reference target objects being edited so that concurrent updates by multiple processes and/or users are allowed. A full lock is used only when a deletion is attempted. The Relaxed setting allows faster parallel creation of references to the same target object. This setting, when used with object types that are frequently referenced but rarely deleted, improves the performance and stability of parallel inbound imports, bulk updates, and users concurrently creating references to the same objects.

Important: Do not use the 'Relaxed' setting on an object type for objects that are often deleted as this can result in poor performance, and risks locks and deadlocks.

For information on locking options, refer to the **Optimistic and Pessimistic Locking** Recommendations topic in the **System Administration** documentation.



The Relaxed / Strict setting can be changed as needed, but Oracle databases require the system to enter Single Update Mode for only as long as it takes to update the setting. For more information, refer to the **Single-Update Mode** topic.

Note: Changing this setting during peak production hours can have a negative impact on performance.

Outbound Integration Endpoints

Outbound integration endpoint (OIEP) functionality can, in many ways, be thought of as a way to manage the standard STEP export functionality. Two types of OIEPs are available to meet the data export needs. As defined below, when creating either type of OIEP, you will use a wizard for the initial configuration, and then use the OIEP editor to manually further define your required OIEP.

Event-Based Outbound Integration Endpoint

OIEPs can be used for incremental output with an **Event-Based OIEP** which publishes data based on events occurring in STEP. An event, for example, could be generated when an object in STEP is approved. Multiple OIEPs can be configured to listen for such an event, and each will use the recommended simplified priority background queuing mechanism or will have an associated legacy event queue for registered events. The OIEP can then, with scheduled intervals (minimum interval being once every minute), check the event queue for new events and publish data accordingly. For details on events, refer to the **Events** section of the **System Setup** documentation. For information about setting up an event processor, refer to **Event Processors** in the **System Setup**.

Setup Requirements for Event-Based OIEPs

Setting up and using a select object outbound integration endpoint involves the steps described in Creating an Event-Based OIEP:

1. Create a setup group to hold the endpoint as described in **Initial Setup for an Outbound Integration Endpoint**.
2. Launch the outbound integrations endpoint wizard as described in **Creating an Event-Based Outbound Integration Endpoint**.
3. In the wizard, add an ID, name, description, and user as described in **OIEP - Event-Based - Identify Endpoint**.
4. In the wizard, choose Select Objects for the data source as described in **OIEP - Event-Based - Choose Data Source**.
5. In the wizard, specify processing and context settings for the endpoint as described in **OIEP - Event-Based - Configure Endpoint**.
6. In the OIEP editor, configure to send an email if an endpoint-related background process fails as described in the **Error Handling & Reporting** section of the **OIEP - Configuration Flipper** documentation.
7. In the OIEP editor, specify determine when the OIEP runs as described in the **Schedule** section of the **OIEP - Configuration Flipper** documentation.
8. In the OIEP editor, specify the objects to be output, the format and pre- or post-processors (if any) as described in **OIEP - Event-Based - Output Templates Flipper**.
9. In the OIEP editor, determine how data is delivered as described in **OIEP - Delivery Method Flipper**.
10. In the OIEP editor, define what data changes will cause the OIEP to run as described in the **OIEP - Event-Based - Event Triggering Definitions Tab**.
11. In the OIEP editor, set the queue status as described in the **Event-Based OIEP Status and Queue Status**.

12. Enable the endpoint and invoke it as described in **Running an Outbound Integration Endpoint**, paying particular attention to the **Prerequisites for Event-Based OIEPs** section.

Additional Information for Event-Based OIEPs

The following information is useful once an inbound integration endpoint is set up:

1. Review use cases for an event-based OIEP as described in **Event-Based OIEP Examples**.
2. Maintain or modify the endpoint from within STEP as described in **Monitoring an OIEP via Background Process**.
3. Maintain or modify the endpoint outside of STEP as described in **Monitoring an OIEP via External Systems**.
4. Understand the effects of batching on an OIEP as described in **Event-Based OIEP Event Batching**.
5. Understand the effects of manipulating events on an OIEP as described in **Event-Based OIEP Event Actions**.
6. Understand the multithreading functionality on an OIEP as described in **Event-Based OIEP Multithreading Support**.
7. Understand queued events on an OIEP as described in **Event-Based OIEP Queued Events**.
8. Understand the relationship between OIEP Status and Queue Status as described in **Event-Based OIEP Status and Queue Status**.
9. Understand each element of an OIEP as described in **Outbound Integration Endpoint Structure**.
10. Export an outbound integration endpoint definition and submit to an external source control system for comparison purposes as described in the **Configuration Management** documentation.

Select Objects Outbound Integration Endpoint

In the simplest form, a **Select Object OIEP** uses a static set of data and works exactly like a scheduled export (created in Export Manager). This type of OIEP publishes data non-incrementally from selected hierarchies to external systems on scheduled intervals. The only difference between a Select Objects OIEP and a scheduled export in Export Manager is that the OIEP has more standard delivery options than Export Manager and offers extended monitoring capabilities.

Setup Requirements for Select Objects OIEPs

Setting up and using a select object outbound integration endpoint involves the following steps:

1. Create a setup group to hold the endpoint as described in **Initial Setup for an Outbound Integration Endpoint**.
2. Launch the outbound integration endpoint wizard as described in **Creating a Select Objects Outbound Integration Endpoint**.
3. In the wizard, add an ID, name, description, and user as described in **OIEP - Select Objects - Identify Endpoint**.
4. In the wizard, choose Select Objects for the data source as described in **OIEP - Select Objects - Choose Data Source**.

5. In the wizard, specify processing and context settings for the endpoint as described in **OIEP - Select Objects - Configure Endpoint**.
6. In the OIEP editor, configure to send an email if an endpoint-related background process fails as described in the **Error Handling & Reporting** section of the **OIEP - Configuration Flipper** documentation.
7. In the OIEP editor, determine when the OIEP runs as described in the **Schedule** section of the **OIEP - Configuration Flipper** documentation.
8. In the OIEP editor, specify the objects to be output, the format, and pre- or post-processors (if any) as described in **OIEP - Select Objects - Output Templates Flipper**.
9. In the OIEP editor, determine how data is delivered as described in **OIEP - Delivery Method Flipper**.
10. Enable the endpoint and invoke it as described in **Running an Outbound Integration Endpoint**.

Additional Information for Select Objects OIEPs

The following information is useful once a select objects outbound integration endpoint is set up:

1. Determine the progress of the endpoint from within STEP as described in **Monitoring an OIEP via Background Process**.
2. Determine the progress of the endpoint outside of STEP as described in **Monitoring an OIEP via External Systems**.
3. Maintain or modify the endpoint as described in **Maintaining an Outbound Integration Endpoint**.
4. Understand each element of an OIEP as described in **Outbound Integration Endpoint Structure**.
5. Export an outbound integration endpoint definition and submit to an external source control system for comparison purposes as described in the **Configuration Management** documentation.

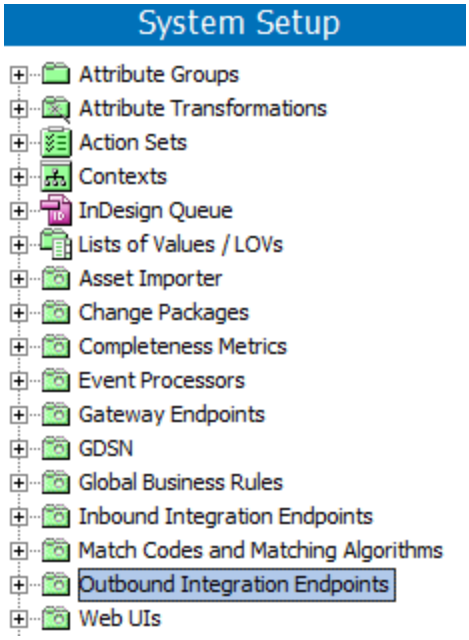
Initial Setup for an Outbound Integration Endpoint

Before creating an outbound integration endpoint, one or more setup groups must be created that are allowed to hold outbound integration endpoints. You must also specify the setup group(s) in which outbound integration endpoints can be created. This setup only needs to be performed once, and most systems will already have it completed.

Only users with the relevant privileges can view or maintain outbound integration endpoints. For detailed information, refer to the **Action Sets** section and the **Users and Groups** section in the **System Setup** documentation.

Note: Attempting to delete a user who is responsible for an OIEP will result in errors. For more information, refer to the **Working with Users** topic in the **System Setup** documentation.

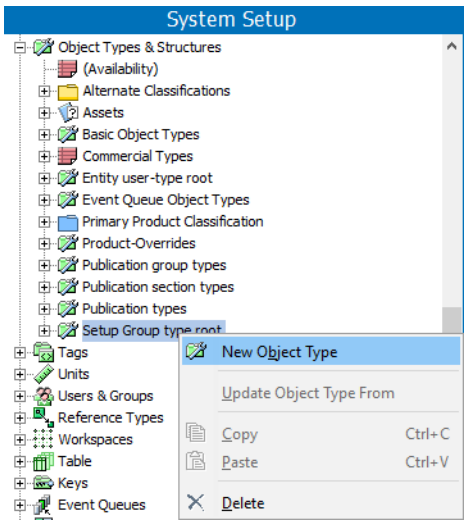
Review your System Setup tab to determine if one or more outbound integration endpoint nodes already exist. The name of the node on your system is not required to match the one in the image below.



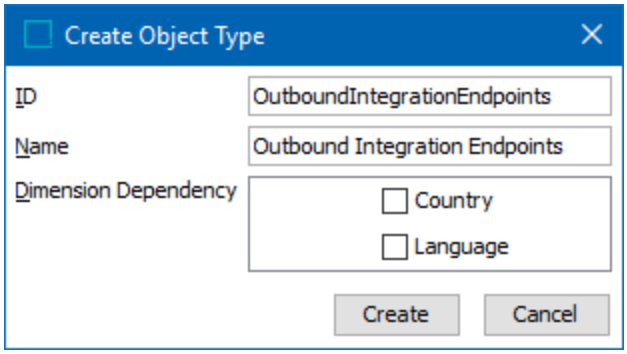
Once the setup has been completed, the steps in this section are only needed if additional levels of organization are desired.

Create the Outbound Integration Setup Group

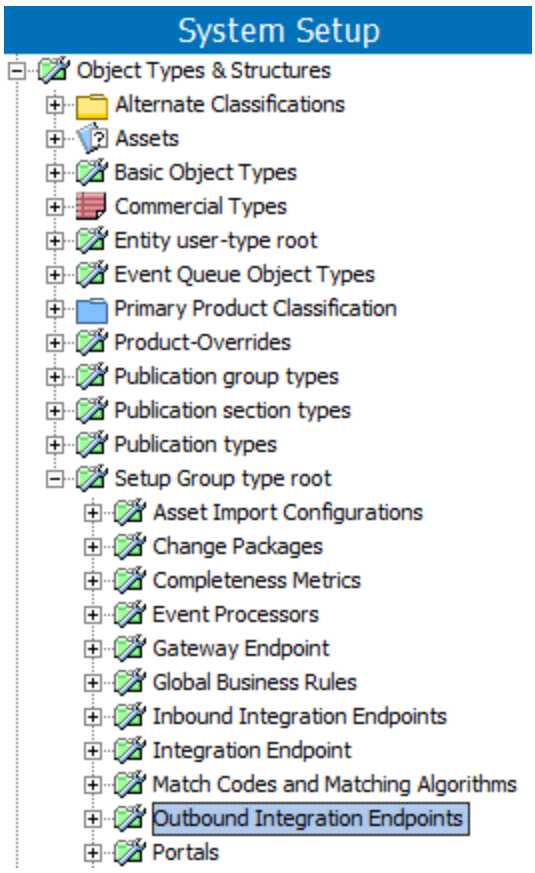
1. In System Setup, expand **Object Types & Structures**.
2. Right-click 'Setup Group type root', and choose **New Object Type**.



3. Enter an **ID** and a **Name**, select any required **Dimension Dependencies**, and click **Create**.

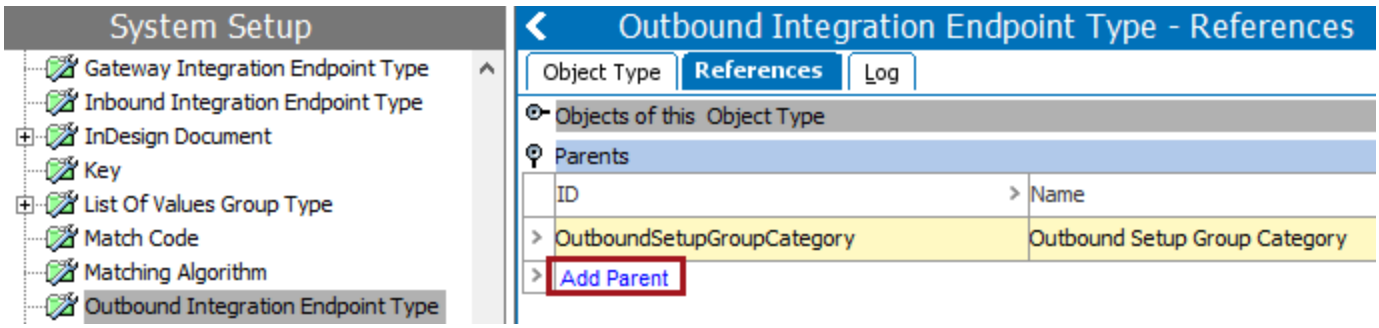


4. The new setup group appears in System Setup under 'Object Types & Structures' as a child in the **Setup Group type root**.

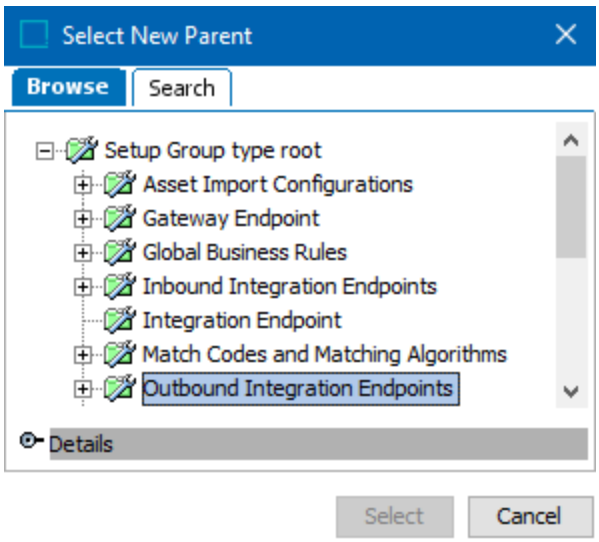


Link the OIEP Object Type to Setup Group

1. In Object Types & Structures node, expand Basic Object Types and select **Outbound Integration Endpoint Type**.
2. On the References tab, open the Parents flipper and click the **Add Parent** link.

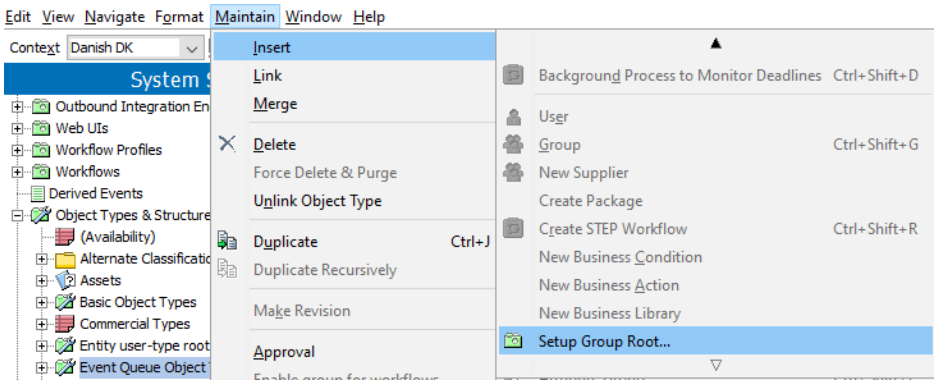


3. In the Select New Parent dialog, select the setup group you created, and click **Select** to make it a valid parent.

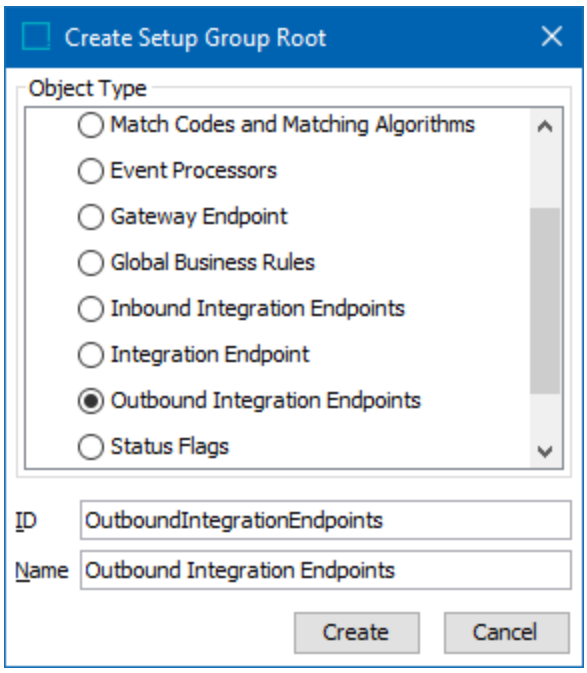


Create an Instance of the OIEP Object

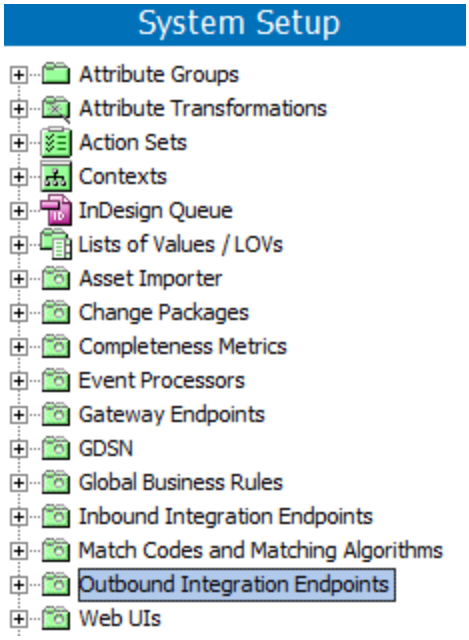
1. On the System Setup tab, select any object in the **System Setup** hierarchy to activate the following Maintain menu selection.
2. Click the Maintain menu, point to Insert, and select **Setup Group Root**.



3. In the Create Setup Group Root dialog, select the outbound integration endpoints object type, enter an **ID** and a **Name**, and click **Create**.



A setup group is created as a node in the System Setup hierarchy. Outbound integration endpoints can now be created under this new node.



Creating an Outbound Integration Endpoint

After creating a setup group for outbound integration endpoints, use the following topics to create the type of OIEP to meet your export needs:

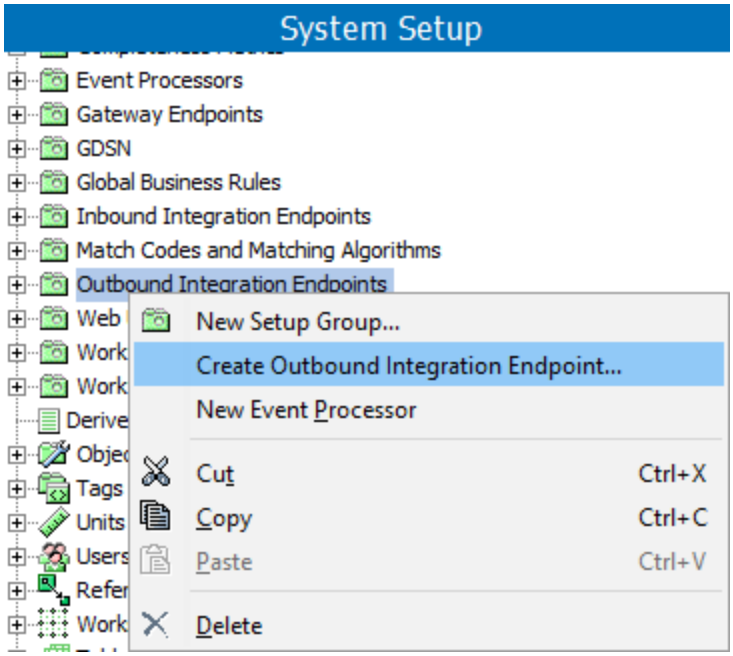
- To create an event-based OIEP that can monitor STEP for changes and export as needed, refer to **Creating an Event-Based Outbound Integration Endpoint**.
- To create an OIEP to send a static set of STEP data, refer to **Creating a Select Objects Outbound Integration Endpoint**.

Creating an Event-Based Outbound Integration Endpoint

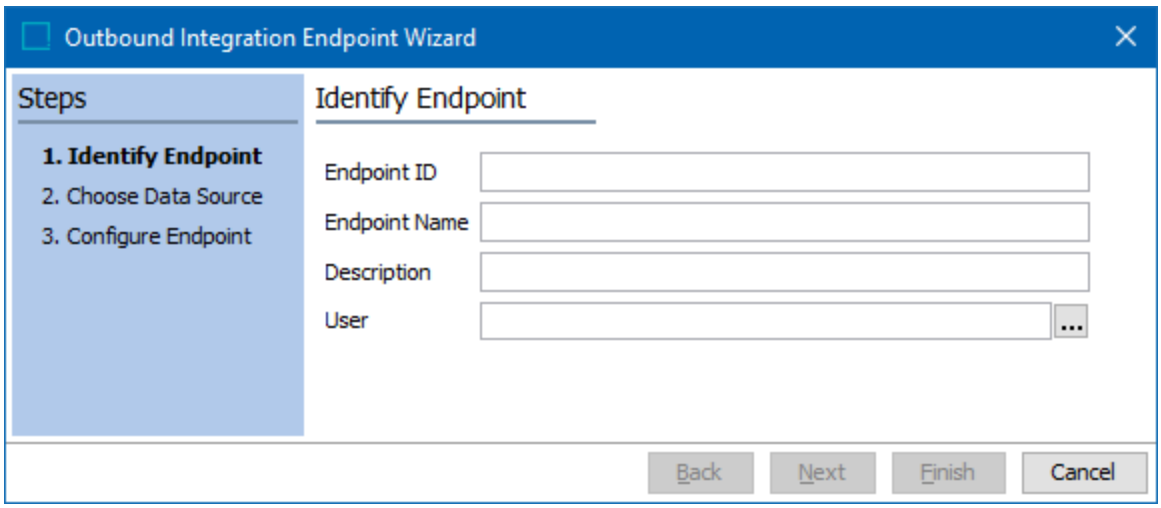
An event-based OIEP is used to monitor for approved data changes in STEP (based on the revisability setting of the changed data), and to send the changed data to one or more external systems.

After creating a setup group for outbound integration endpoints, create an event-based outbound integration endpoint to monitor changes in STEP and output data. For information on a select objects OIEP, that can send a static set of data from STEP, refer to **Creating a Select Objects Outbound Integration Endpoint**.

1. In System Setup, right-click the Outbound Integrations Endpoints setup group, and click **Create Outbound Integration Endpoint**.

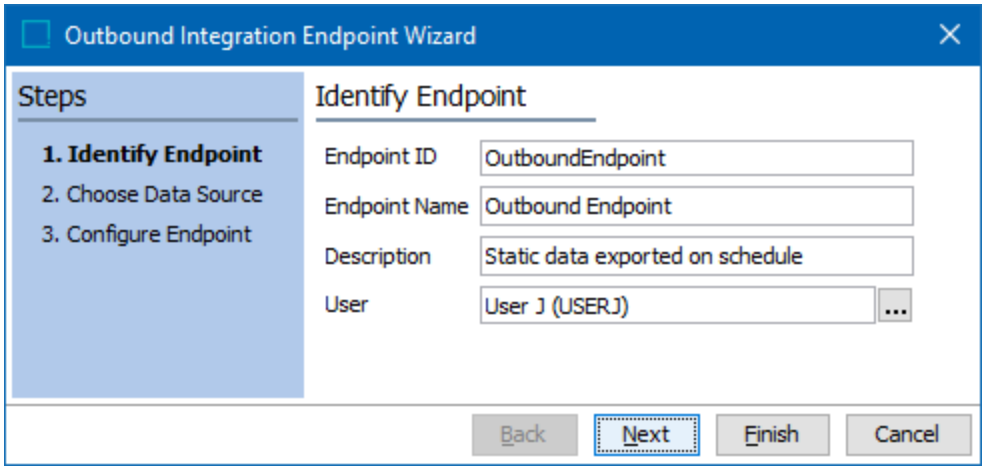


2. The Outbound Integration Endpoint wizard displays and involves the following steps:



3. Complete setup require providing data in the wizard as well as some manual configuration. All setup includes the following sections:
   - In the wizard, **OIEP - Event-Based - Identify Endpoint** allows you to specify the name and ID of the endpoint and identify the user whose privileges are applied.
   - In the wizard, **OIEP - Event-Based - Choose Data Source** allows you to specify a static set of data or an event queue.
   - In the wizard, **OIEP - Event-Based - Configure Endpoint** allows you to specify processing and context settings for the integration endpoint.
   - In the OIEP editor, **OIEP - Event-Based - Manual Configuration** directs you to complete the required manual setup once the OIEP wizard steps are performed.

OIEP - Event Based - Identify Endpoint



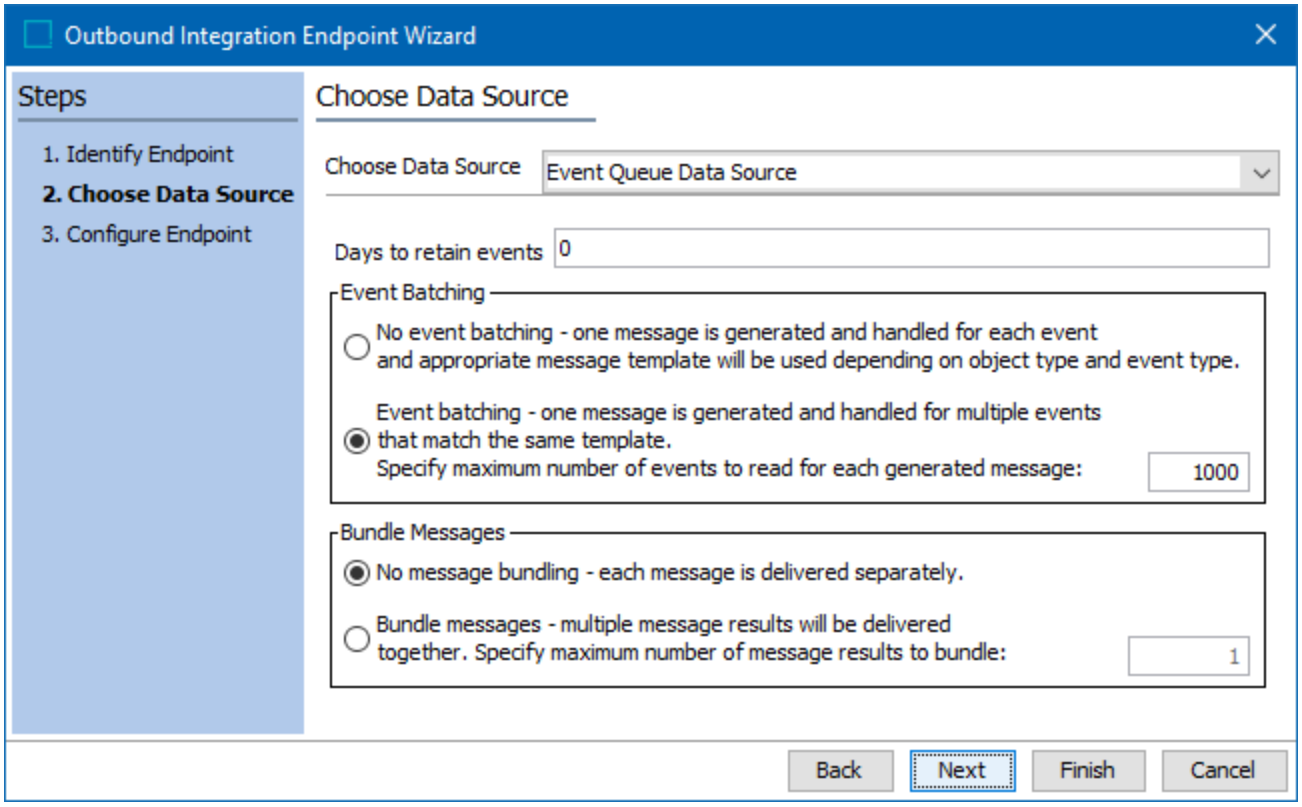
An event-based OIEP allows you to export data to be published as incremental changes are made and approved, based on the object's revisability.

1. Enter an **ID**. Common setup is to use no spaces or punctuation.
2. Enter a **Name**. Common setup is to repeat the ID with added spaces for readability.
3. Enter an optional **Description**. It is good practice to add description for future reference.
4. Use the search or browse for a **User**. The privileges of the selected user determine which actions the integration endpoint can perform and what data can be processed. A common setup is to create a special system user for this purpose so that the effects of the endpoint are easily identified and are not dependent on a particular STEP user.

Note: Attempting to delete a user who is selected in the User parameter for an OIEP will result in errors. For more information, refer to the **Working with Users** topic in the **System Setup** documentation.

5. Click **Next** to display **OIEP - Event-Based - Choose Data Source**.

OIEP - Event-Based - Choose Data Source

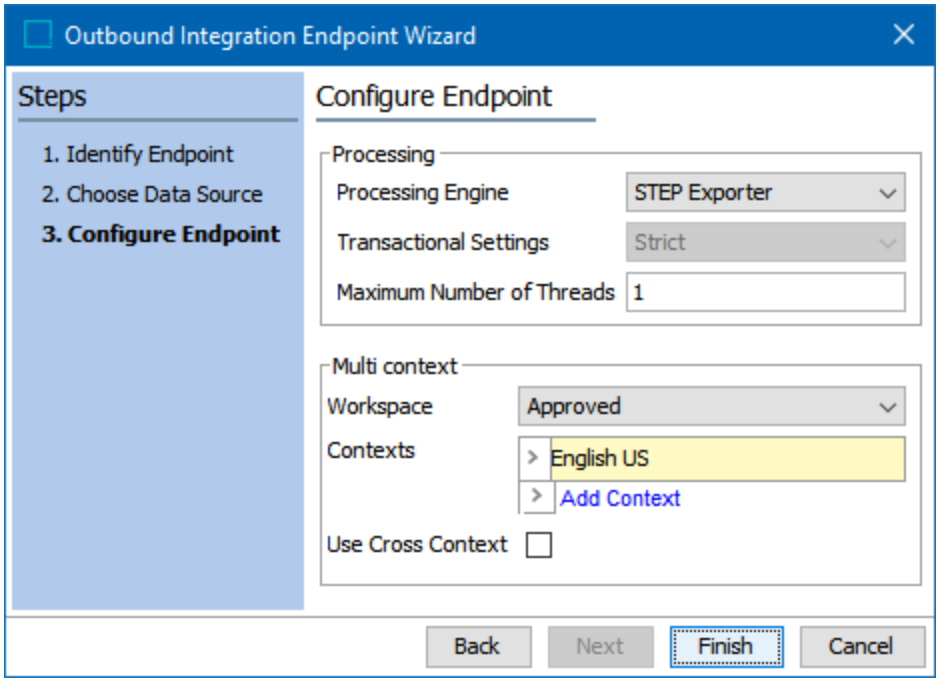


1. Choose the data source **Event Queue Data Source** for incremental exports where data should be published based on events. Creating an event-based OIEP automatically creates an event queue to process events.

   Note: A **Select Objects** data source requires a separate setup, for detailed steps, refer to the **Creating an Outbound Integration Endpoint** documentation.

2. For the **Days to Retain Events** field, enter the number of days for the endpoint to retain events once they are read. Setting this number > 0 means that read events will stay on the queue for the specified number of days. This can be used as a security measure. If something goes wrong with the receiver, retaining read events will allow you to republish data for the retained events again later. Common setup is to use between 5 and 10 days.
3. For the **Event Batching** field, select an option to determine how many events should be included in each exported file. For more information, refer to **Event-Based OIEP Event Batching**.
4. For the **Bundle Messages** field, select an option to determine if multiple files generated by the same background process should be delivered together in the same message.
5. Click **Next** to display **OIEP - Event-Based - Configure Endpoint**.

OIEP - Event Based - Configure Endpoint



1. For the **Processing Engine** parameter, select an option from the dropdown:
   - **STEP Exporter** uses the same functionality as the Export Manager. This is the only processing engine option on a standard STEP system and performs the actual data export.
   - **Business Rule Based Message Processor** allows you to export STEP data using either JavaScript-based business actions, or Java business actions developed via the Extension API. For more information, refer to the **OIEP - Configuration Flipper for Business Rule Based Message Processor** topic or click the **STEP API Documentation** button on the Start Page.
   - If you need a customized processing engine, contact your Stibo Systems account manager or partner manager. For more information, refer to **Outbound Integration Endpoint Structure** documentation.
2. The **Transactional Settings** option must be set to 'Strict' for event-based OIEPs. For more information, refer to the **Integration Endpoint Transactional Settings** documentation.
3. The **Maximum Number of Threads** option is only valid for an Event Queue Data Source endpoint. The default thread setting is one (1), in which case the endpoint produces a single message at a time, with all events in the batch processed serially. Increasing the thread number results in each batch size being divided by the thread number so that the contents of a batch can be processed in parallel. When increasing the maximum number of threads, you must also consider hardware limitations. For more information, refer to **Event-Based OIEP Multithreading Support**.
4. Set the **Multi context** options that the endpoint should use to export data:
   - For **Workspace**, use the dropdown to select the desired workspace. A common setup is to use the Approved workspace, except when you need to generate events for objects before they are approved, for example, during import or from a workflow.
   - For **Contexts**, when exporting data in the STEPXML format or when using a custom cross-context enabled format, one file can contain data from multiple contexts. For other formats, the standard 'Context

splitter Post-processor' should be used to generate separate files for the selected contexts. Click the **Add Context** link and choose one or more contexts.

- For **Use Cross Context**, enable this option if adding more than one context or if there is any chance that you will later add more than one context. Using cross context mode even with only one context selected means that it is not necessary to change downstream systems should more contexts be added later.

> **Note:** Changing from standard format to cross-context STEPXML can cause issues to downstream systems that are parsing the delivered files / messages.

5. Click **Finish** to complete the wizard.
6. Perform the manual configuration described in **OIEP - Event-Based - Manual Configuration**.

OIEP - Event-Based - Manual Configuration

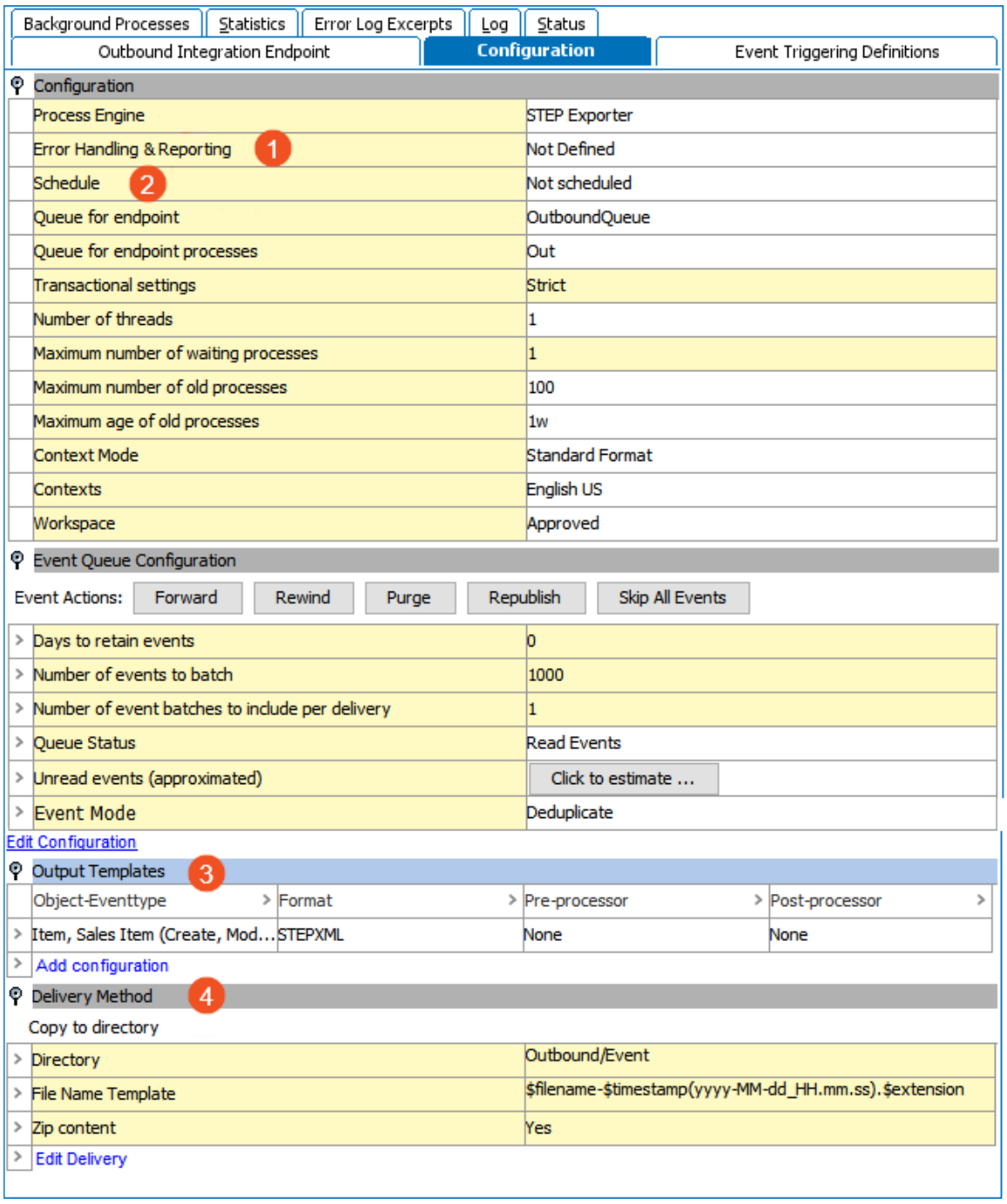
After completing the Outbound Integration Endpoint wizard for an event-based OIEP using the STEP Exporter process engine, the OIEP editor includes additional parameters that must be set manually before the OIEP can run.

Note: In addition to the set up in this topic, additional manual configuration is required when using the Business Rule Based Message Processor, as defined in the **OIEP - Configuration Flipper for Business Rule Based Message Processor** topic.

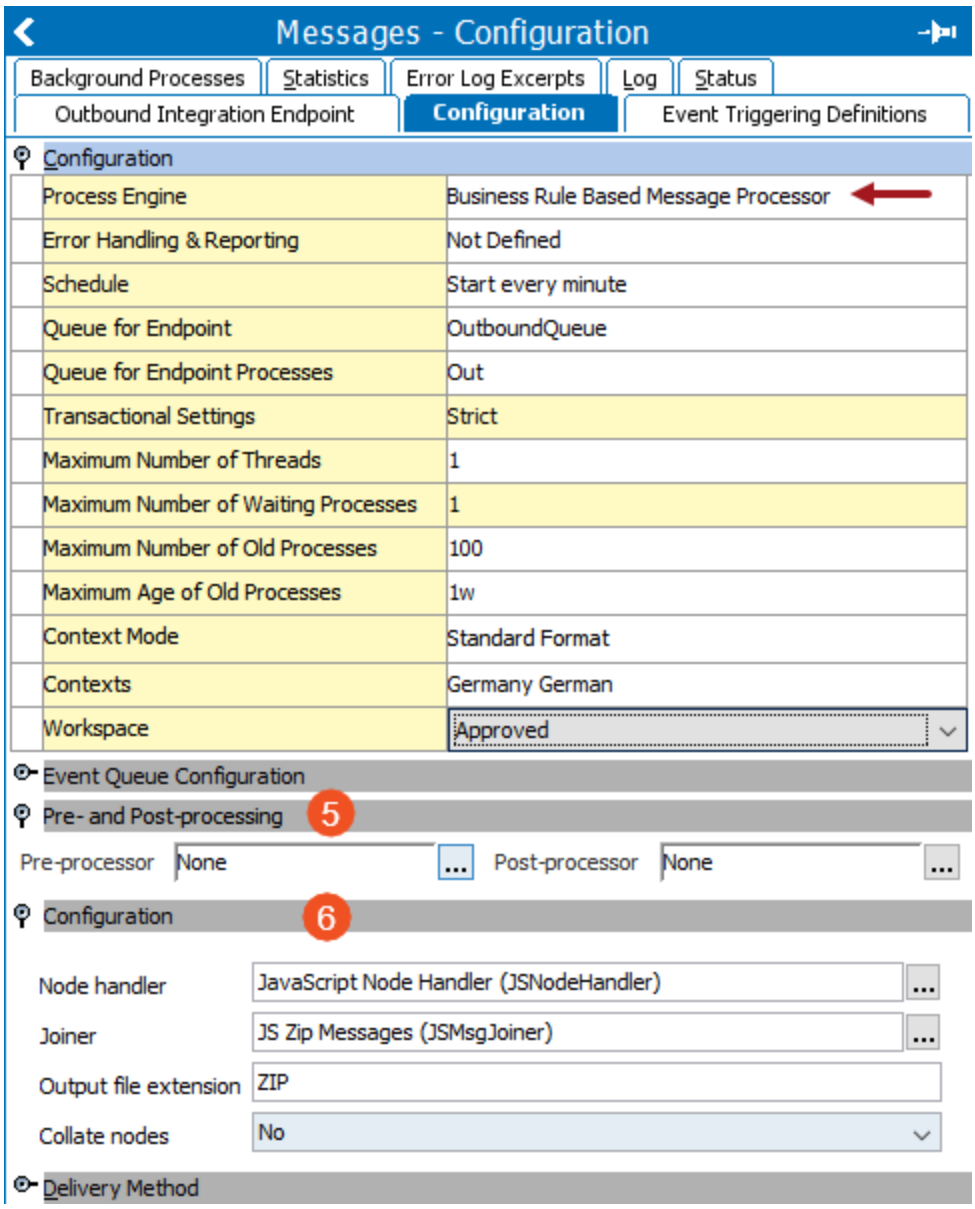
The configuration set from the wizard can also be modified manually using the Configuration tab.

Follow these steps to complete the manual configuration and set the event-based OIEP to run:

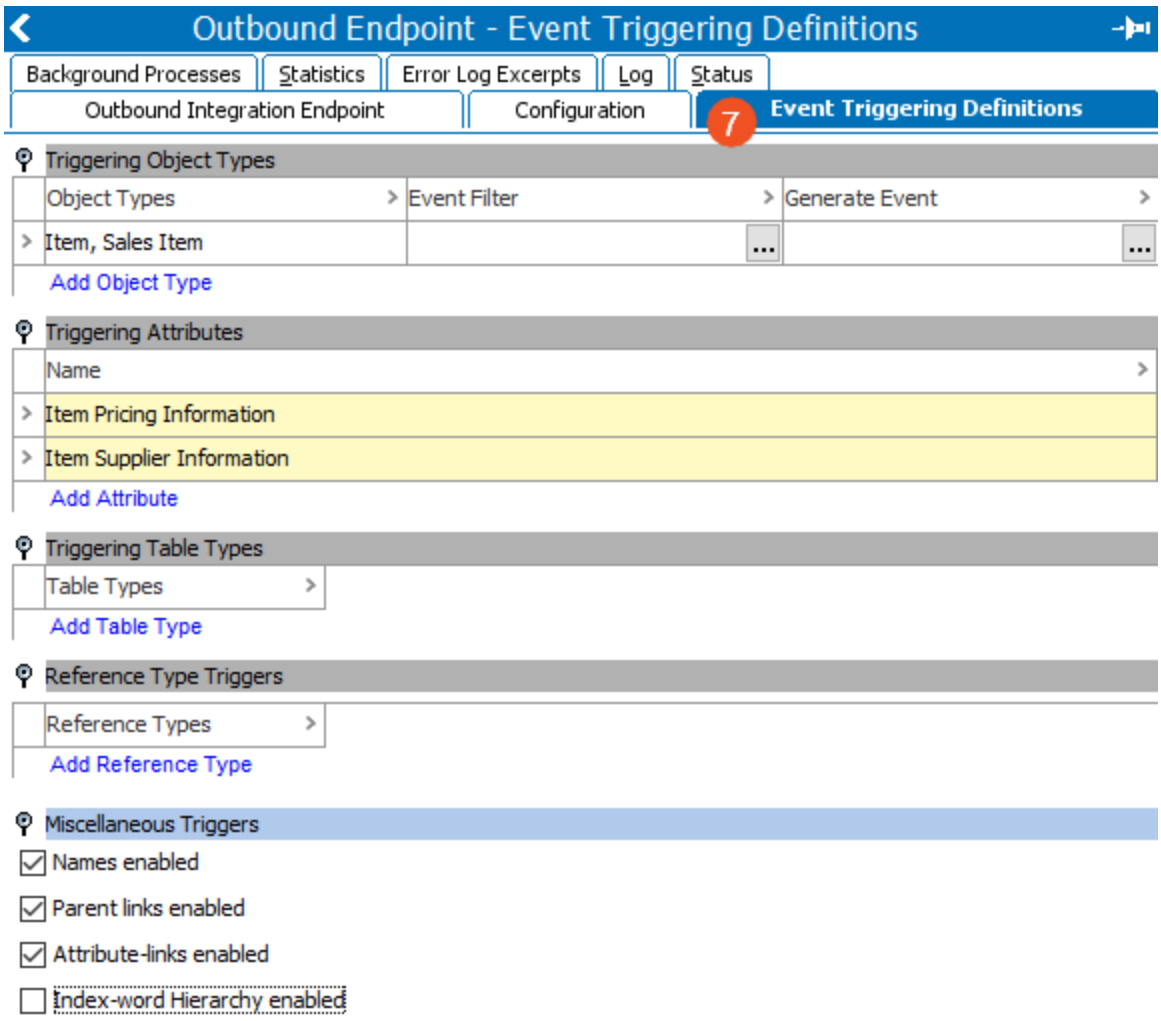
1. In the OIEP editor, configure to send an email if an endpoint-related background process fails as described in the **Error Handling & Reporting** section of **OIEP - Configuration Flipper**.
2. In the OIEP editor, determine when the OIEP runs as described in the **Schedule** section of **OIEP - Configuration Flipper**.
3. In the OIEP editor, when using the STEP Exporter process engine, specify the objects to be output, the format, and pre- or post-processors (if any) as described in **OIEP - Event-Based - Output Templates Flipper**.
4. In the OIEP editor, determine how data is delivered as described in **OIEP - Delivery Method Flipper**.



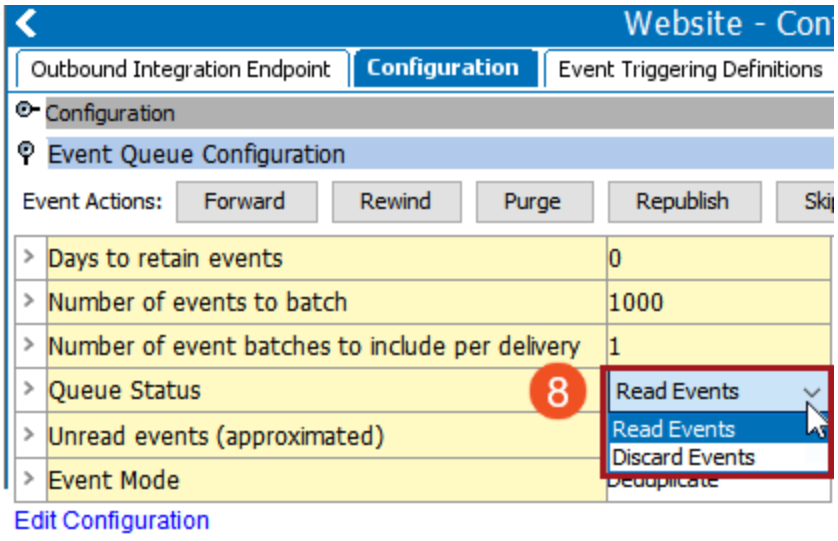
5. In the OIEP editor, when using the Business Rule Based Message Processor process engine, specify the pre- or post-processors (if any) as described in **OIEP - Pre- and Post-processing Flipper**.
6. In the OIEP editor, when using the Business Rule Based Message Processor process engine, specify the business actions and settings as described in **OIEP - Configuration Flipper for Business Rule Based Message Processor**.



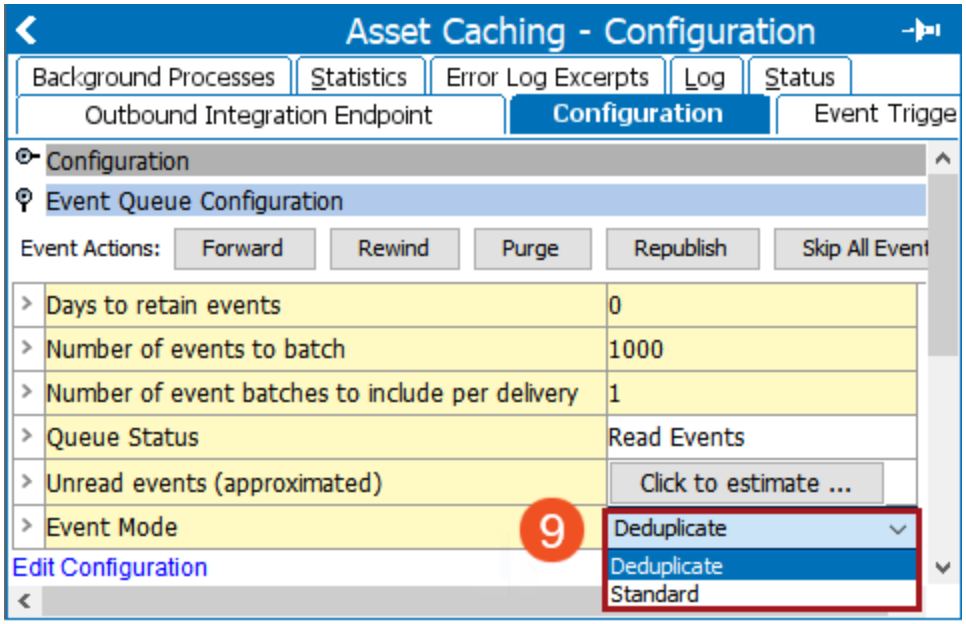
7. In the OIEP editor, define what data changes will cause the OIEP to run as described in the **OIEP - Event-Based - Event Triggering Definitions Tab**.



8. Set the Queue Status as described in **Event-Based OIEP Status and Queue Status**.



9. Determine the appropriate Event Mode setting.



- **Standard** - The default for new configurations. Standard mode adds all events to a given queue for an object without further processing, which can include more events than necessary, resulting in duplication of processing and exports with the same outcome. Standard mode allows for backward compatibility and is recommended for these event queue configuration scenarios:
  - unique object-event type combinations (a single derived or republish event instead of multiple event triggers on attributes and references, etc.)
  - business rules that complete quickly
  - typically larger queue sizes (> 100k events)
  - large batch sizes (approximately 10k events)
  - exports that complete quickly without pre-processing or post-processing

  In these scenarios, the Standard mode performs better than the Deduplicate mode since Deduplicate includes the overhead required to remove redundant events.

- **Deduplicate** - Deduplicate mode removes events from a queue for object-event type combinations that already exist on the queue. This reduces redundant business rule processing and exports and prevents downstream systems from receiving the same data when the same objects are present in multiple event batches, which also reduces processing required on downstream systems. The required overhead to remove redundant events saves overall processing time and improves performance in the following scenarios. Deduplicate is the recommended mode for these event queue configuration scenarios:

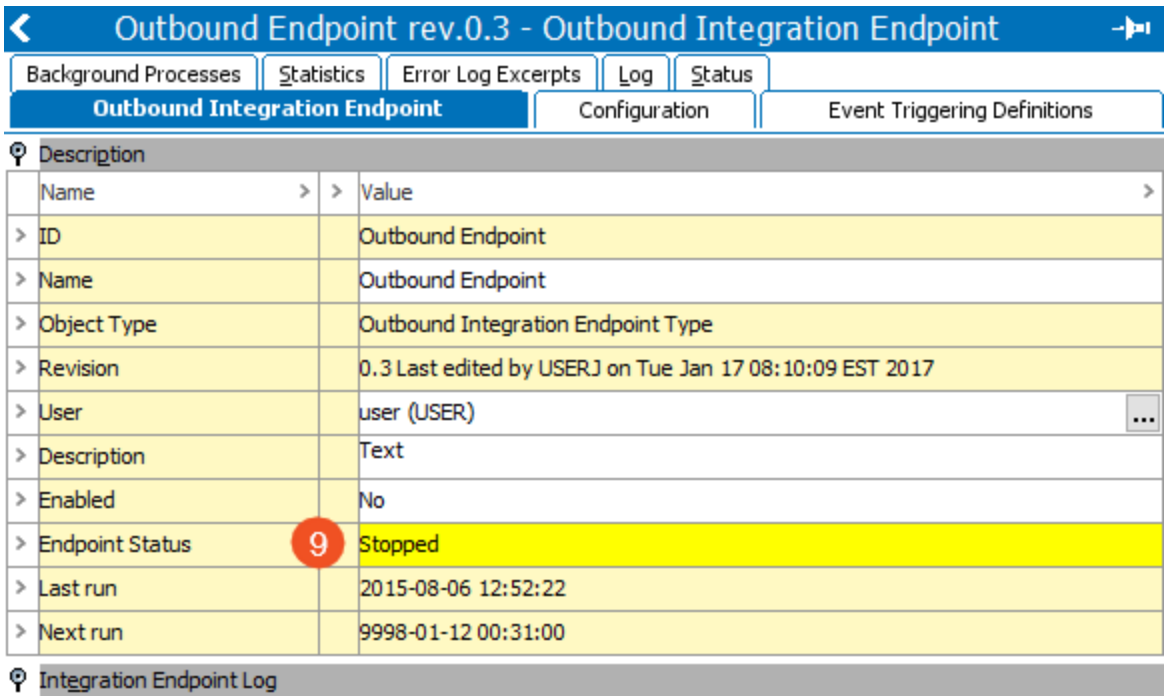
- publishing to PDX using the STEPXML template
- high duplication of events waiting in the queue for processing
- complex business rules / exports that take a significant amount of time to complete for a batch
- downstream systems that have trouble keeping up with message processing
- typically smaller queue sizes (< 10k events)
- small batch sizes (< 10 events)

In the Performance Analysis Tool Activity tree, logging has been added for the time spent deduplicating events when an outbound integration endpoint (OIEP) is configured with the Deduplicate Event Mode. Using this information, customers can make data-driven decisions regarding which is the best mode to use (Deduplicate or Standard), as the decision is highly dependent on the underlying use case and overall expected throughput. In some cases the time it takes to complete the deduplication of events outweighs the potential performance gains in processing, while in other cases the processing itself is the bottleneck. Therefore, being able to compare the time spent deduplicating against the overall run time supports administrators in selecting the setting that will lead to the most optimal performance for their particular scenario.

Note: Deduplicate mode may not run as fast as Standard mode due to the analysis and removal of events when each batch is processed. When high volumes of unprocessed events are in queue (> 500k), possibly from a temporary stoppage where the events cannot be discarded, processing speed decreases as the event count grows. Consider using Standard mode temporarily until the queue has a more manageable backlog.

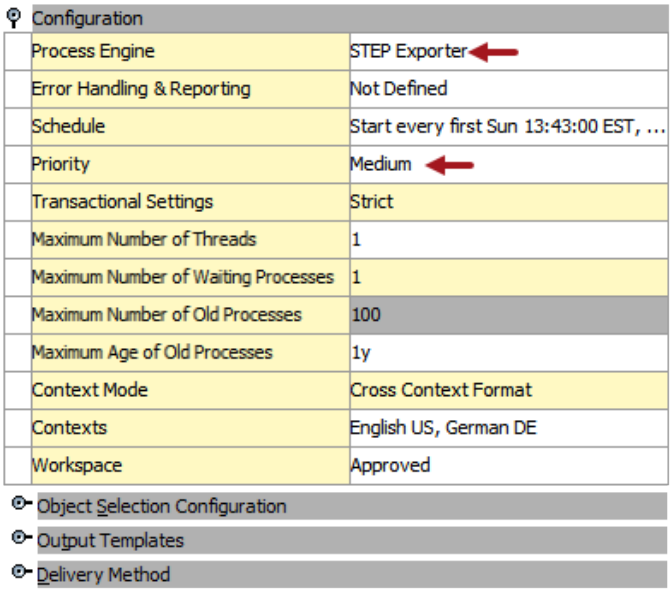
If the OIEP is unlikely to have multiple similar events for objects, the typical volume is low, the export and business rules are simple (batches complete very quickly), and/or the batch sizes are large, then the overhead from attempting to remove redundant events may result in slower execution than that of the Standard mode.

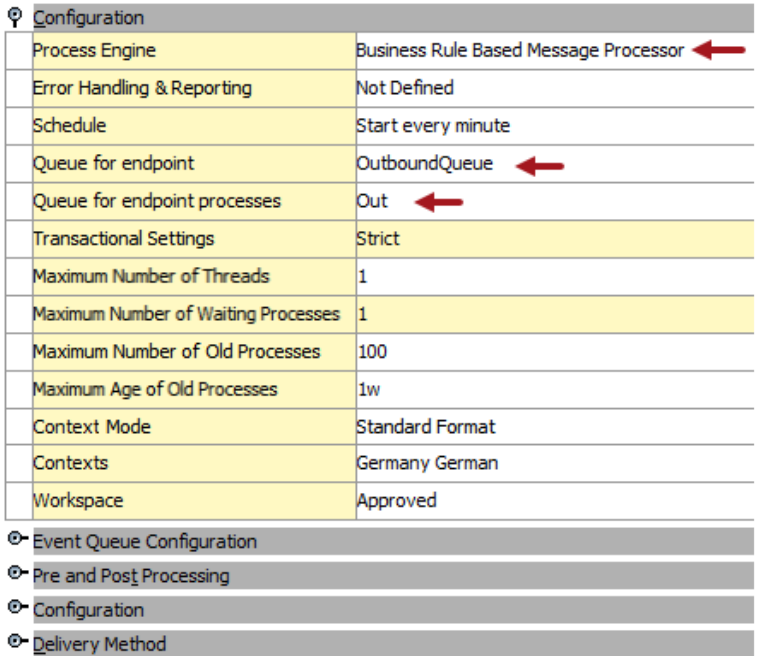
10. Enable the endpoint and invoke it as described in **Running an Outbound Integration Endpoint**, paying particular attention to the **Prerequisites for Event-Based OIEPs** section.


Outbound Endpoint rev.0.3 - Outbound Integration Endpoint
Background Processes
Statistics
Error Log Excerpts
Log
Status
Outbound Integration Endpoint
Configuration
Event Triggering Definitions
Description
Name
Value
ID
Outbound Endpoint
Name
Outbound Endpoint
Object Type
Outbound Integration Endpoint Type
Revision
0.3 Last edited by USERJ on Tue Jan 17 08:10:09 EST 2017
User
user (USER)
Description
Text
Enabled
No
Endpoint Status
9
Stopped
Last run
2015-08-06 12:52:22
Next run
9998-01-12 00:31:00
Integration Endpoint Log

OIEP - Configuration Flipper

The following flippers are available when the STEP Exporter processor or the Business Rule Based Message processor is selected for an OIEP. For both processors, the Configuration flipper includes the same parameters for both Event-Based and Select Objects endpoints. Each parameter is described below.





Note: When a GDSN processor or Business Rule Based Message Processor is selected for the Process Engine parameter, different or additional flippers are displayed.

Process Engine

STEP Exporter is the only processing engine option on a standard STEP system. If you need a customized processing engine, contact your Stibo Systems account manager or partner manager.

The processing engine is originally selected in the wizard when configuring the endpoint. For more information, refer to **OIEP - Event-Based - Configure Endpoint** documentation or **OIEP - Select Objects - Configure Endpoint** documentation.

Error Handling & Reporting

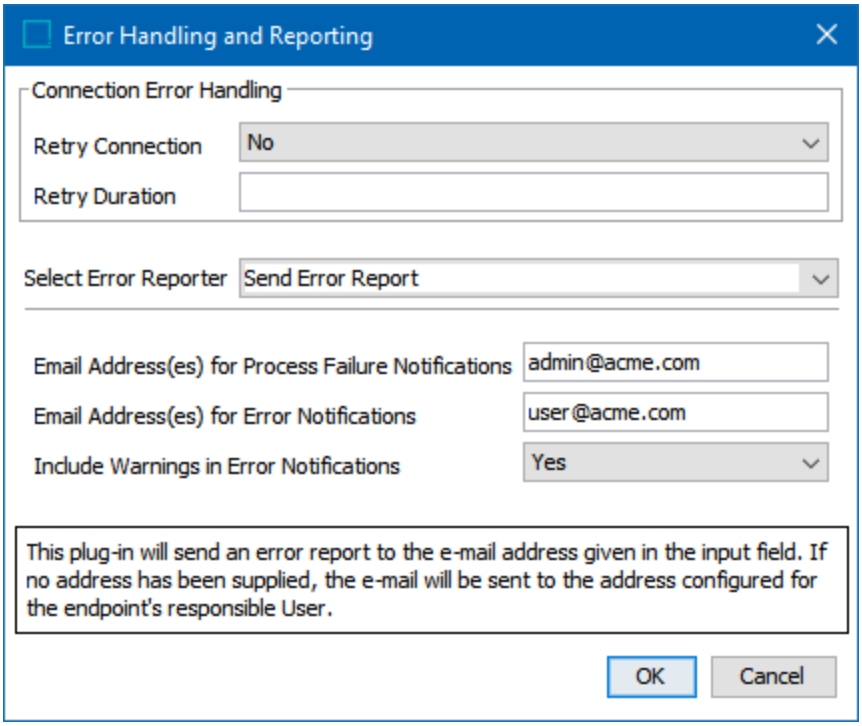
The 'Error Handling & Reporting' parameter is not included in the wizard, but it can be configured from the OIEP editor to send an error report to the specified email addresses if a background process fails or has error or warnings. The email alert contains information about the failed endpoint, including the server name, the background process, failed file, failing process step, and the cause of the error. A standard STEP system (without custom extension) includes only the 'Send Error Report' option.

For the Error reporter plugin to send email, the SMTP server must be configured on the application server. For information on configuring email from STEP, refer to the **Email from STEP** topic in the **Resource Materials** of online help.

Configure Error Handling and Reporting

1. Open the relevant OIEP on the Configuration tab, open the Configuration flipper and click into the **Error Handling & Reporting** parameter to display the ellipsis button (...).

2. Click the ellipsis button (...) to display the **Error Handling and Reporting** dialog.



3. For **Connection Error Handling**, the default is no connection error handling. The Retry Duration parameter is ignored when 'No' is set for the Retry Connection parameter. Enabling Connection Error Handling allows automated reconnection attempts when the external system is unavailable. When connection retries begin, a warning is logged to the Execution Report; if a connection cannot be established after the Retry Duration expires, an error message is logged to the Execution Report.

   - On the **Retry Connection** parameter, set 'Yes' to automate reconnection attempts for HTTP-based delivery methods including Amazon SQS, Cloud Blob Storage, Git, Kafka, REST, REST Direct, and SFTP. A 'No' setting requires manually restarting the OIEP if the connection fails.

     Note: Authentication-related connection errors are not retried and the OIEP fails immediately.

- On the **Retry Duration** parameter, when blank, the default or 30 days is applied. Valid settings are an integer plus one of the following time indicators: s (seconds), m (minutes), h (hours), and d (days). For example, 45m indicates that connection retries will continue for 45 minutes, during which time, the OIEP shows a 'Failed (retrying)' state. Multiple integers with time units in order of size may also be used. For example, 1h45m30s indicates that connection retries will continue for 1 hour, 45 minutes and 30 seconds. After the 'Retry Duration' expires, if the connection cannot be re-established with the last retry, the OIEP fails, and email alerts are distributed based on the 'Select Error Reporter' settings. For more information, refer to the **Running an Outbound Integration Endpoint** topic.

4. For **Select Error Reporter**, select an option:

- **Send Error Report**: Use the following parameters to add a relevant email address (or use a semi-colon to add more than one email address) or group email that can be used to receive error and warning reports. When emailing multiple accounts, consider creating a group email and use that email address in the parameters instead of adding multiple email addresses for individuals. With this configuration, a single email group can be updated as the group members change while the 'Select Error Reporter' parameter remains unchanged.

  Note: To send email, the SMTP server must be configured on the application server. For information on configuring email from STEP, refer to the **Email from STEP** topic in the **Resource Materials** of online help.

  The email alert contains information about the failed endpoint, including the server's name, the background process, failed file, failing process step, cause of the error, and a copy of the file that triggered the error.

  If no email address is entered, the alert(s) are sent to the email address of the user who created the integration endpoint. If no email is defined for this user, or if no mail server is defined in the configuration, the error report will be written to the failed Background Process Execution Log.

  - **Email Address(es) for Process Failure Notifications**: Sends an alert email to the listed address (es) when a fatal processing error occurs that puts the IIEP in a 'Failed' state.
  - **Email Address(es) for Error Notifications**: Sends an alert email to the listed address(es) when data warnings and/or errors occur that do not cause the process to stop.
  - **Include Warnings in Error Notifications**: A 'Yes' setting means alert emails include information on both warnings and errors. A 'No' setting means only error information is addressed in Error Notification emails. To receive an email when the IIEP enters the 'Failed (retrying)' state, set this option to 'Yes.'

- **No Error Report** disables the error reporter and no notifications by email are sent.

Note: If the error report is larger than the default maximum, 10 MB, the report will not be attached to the auto-generated email sent to the configured email address(es). If the default maximum is not suitable, an admin can set the following case-sensitive property in the sharedconfig.properties file on the application server to adjust the maximum file size:

`Integration.Endpoint.ErrorFileSizeLimit={MB size}`

Replace the '{MB size}' element with the maximum MB file size allowed for error report attachments. Error reports larger than this configured maximum are not attached to the email.

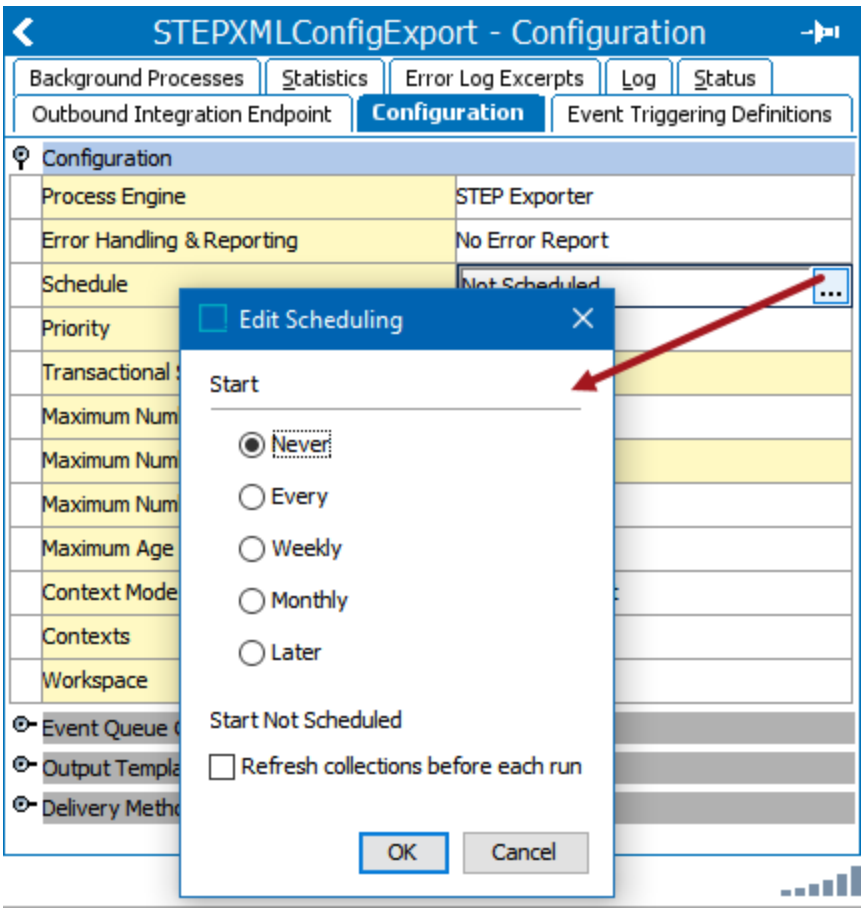
Schedule

The schedule parameter is not included in the wizard, but it can be configured to repeatedly run the OIEP. For Select Objects OIEPs, the background processes runs as scheduled, regardless of changes to the data. For Event-Based OIEPs, the background process runs as scheduled but only when events are available for processing.

Important: Consider the time zone of the application server compared to that of the workbench (the client) where the schedule is created or viewed. When scheduling a job, the local time zone is displayed in the workbench, but the time zone of the server is used to run the background process. Although displayed, the time zone of the client is not included in the instruction to the server to run the job. This can cause confusion about when the job will run since the scheduled time is not automatically converted to accommodate potential differences in time zones.

1. On the relevant OIEP, click the Configuration tab and open the Configuration flipper. Click into the **Schedule** parameter to display the ellipsis button (...).

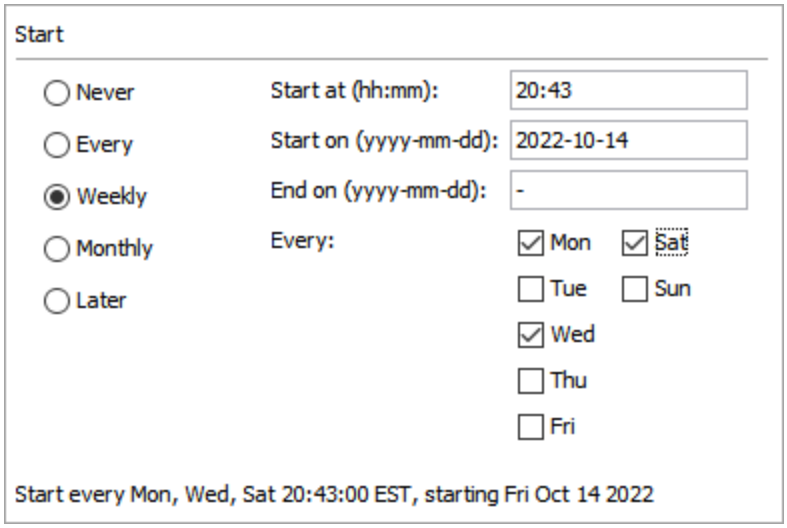
2. Click the ellipsis button (...) and then select one of the following options:



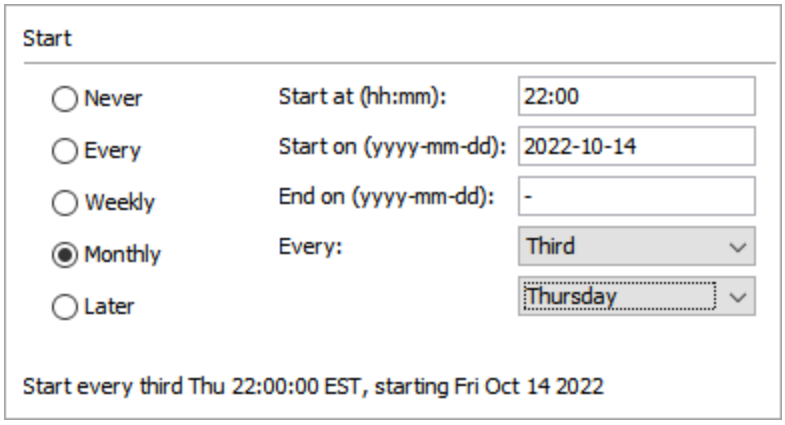
- **Never** - invoke the endpoint manually, no additional parameters are required, and no schedule is applied. This is the default setting and should be used while testing your endpoint.
- **Every** - automatically run the endpoint repeatedly, every selected number of minutes. One (1) minute is the shortest interval allowed and is closest to real time. Enter the number of minutes in the text box. The selection is summarized at the bottom of the dialog.

- **Weekly** - automatically run the endpoint repeatedly, based on the selected time, start and end dates, and days of the week. Use this option if a daily schedule is needed. The 'Start at' parameter determines the time of day that the endpoint will run. The 'Start on' parameter determines the date the endpoint will first run, while the 'End on' parameter determines the date of the endpoint's final run. The 'Every' checkboxes

determine the days of the week when the endpoint will run. The selections are summarized at the bottom of the dialog.



- **Monthly** - automatically run the endpoint repeatedly, once a month, based on the selected time, start and end dates, week of the month, and day of the week. The 'Start at' parameter determines the time of day that the endpoint will run. The 'Start on' parameter determines the date the endpoint will first run, while the 'End on' parameter determines the date of the endpoint's final run. The 'Every' dropdown parameter selections for the week of the month and the day of the week determine when the endpoint will run. The selections are summarized at the bottom of the dialog.



- **Later** - automatically run the endpoint only once, at the time and date specified. The selections are summarized at the bottom of the dialog.

3. Select **Refresh collections before each run** if the outbound integration endpoint contains collections that should be automatically refreshed before each run.

Priority

When the recommended priority-based BGP execution mechanism is configured, waiting BGPs are prioritized for execution based on the priority of the BGP and the created time. The legacy 'Queue for Endpoint' and legacy 'Queue for Endpoint Processes' parameters are not available. Refer to the **Priority** section of the **BG Processes Execution Management** topic in the **System Setup** documentation.

Legacy Queue for Endpoint

This legacy option is not available when the recommended priority-based background process (BGP) execution mechanism is configured.

In a legacy implementation, 'Queue for endpoint' is the name for the queue that is used by the OIEP Background Process to poll the endpoint. The background process handles the actual export. The first time you activate the endpoint, a queue with the specified name is created if it does not already exist. Typically, high-priority integrations and integrations with long-running processes should have their own queue for endpoint processes.

If in doubt about how to populate this parameter, create a new queue for the OIEP, for example, including the OIEP ID in the name.

Legacy Queue for Endpoint Processes

This legacy option is not available when the recommended priority-based background process (BGP) execution mechanism is configured.

STEP allows you to define multiple queues. 'Queue for endpoint processes' is the name for the queue that is used by the background processes started by the endpoint to handle the actual export. The queue is automatically created on the system if it does not already exist. High priority integrations or integrations with long-running processes should typically have their own queue, for example, including the ID in the queue name.

Transactional Settings

'Transactional Settings' can be used for 'Select Objects' OIEPs, but the 'Strict' mode is required for 'Event-Based' OIEPs. For details, refer to the **Integration Endpoint Transactional Settings** topic.

Maximum Number of Threads

Although 'Maximum Number of Threads' is available on the Configuration tab for both static and event-based OIEPs, the setting only affects event-based processes. The default thread setting is one (1) which causes the endpoint to produce a single message at a time, with all events in the batch processed serially. Increasing the thread number results in each batch size being divided by the thread number so that the contents of a batch can be processed in parallel. For more information, refer to **Event-Based OIEP Multithreading Support** topic.

Maximum Number of Waiting Processes

'Maximum Number of Waiting Processes' specifies the maximum number background processes an endpoint can start to handle messages.

By default, the maximum number of waiting processes is set to 1000 for 'Transactional Settings' of Chain or None. When the 'Transactional Settings' parameter is Strict, this value must be 1. Changing an endpoint to have a 'Transactional Settings' of Strict updates this parameter to 1.

Maximum Number of Old Processes

'Maximum Number of Old Processes' specifies the maximum number of ended processes the system retains. This auto-cleanup option deletes succeeded and ended processes started by the OIEP when the limitation is exceeded. The oldest processes are deleted first.

By default, the maximum number of old processes is set to 1000.

Maximum Age of Old Processes

'Maximum Age of Old Processes' specifies the maximum age of succeeded and ended processes. Use the following case-sensitive notation: y = years, M = months, w = weeks, d = days, h=hours, m = minutes, and s = seconds.

By default, the maximum age of old processes is set to one (1) year.

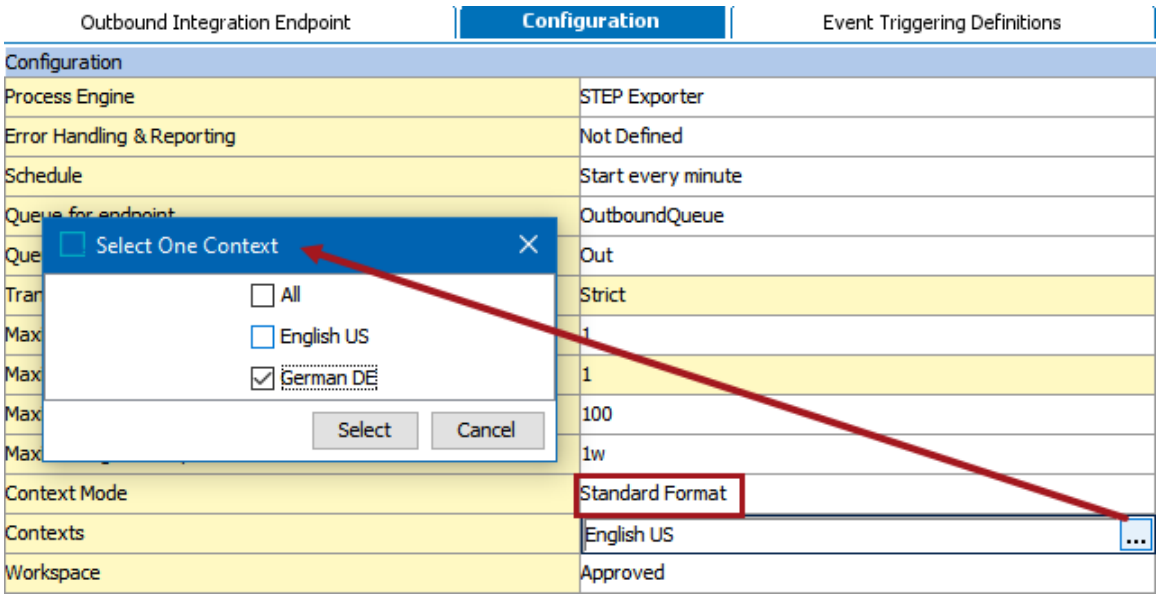
Context Mode and Contexts

When exporting data in the STEPXML format or when using a custom cross-context enabled format, one file can contain data from multiple contexts. For other formats, the standard context splitter post-processor should be used. Using this option, separate files are generated for the selected contexts. For more information, refer to the **OIEP - Post-processor - Context Splitter** documentation.

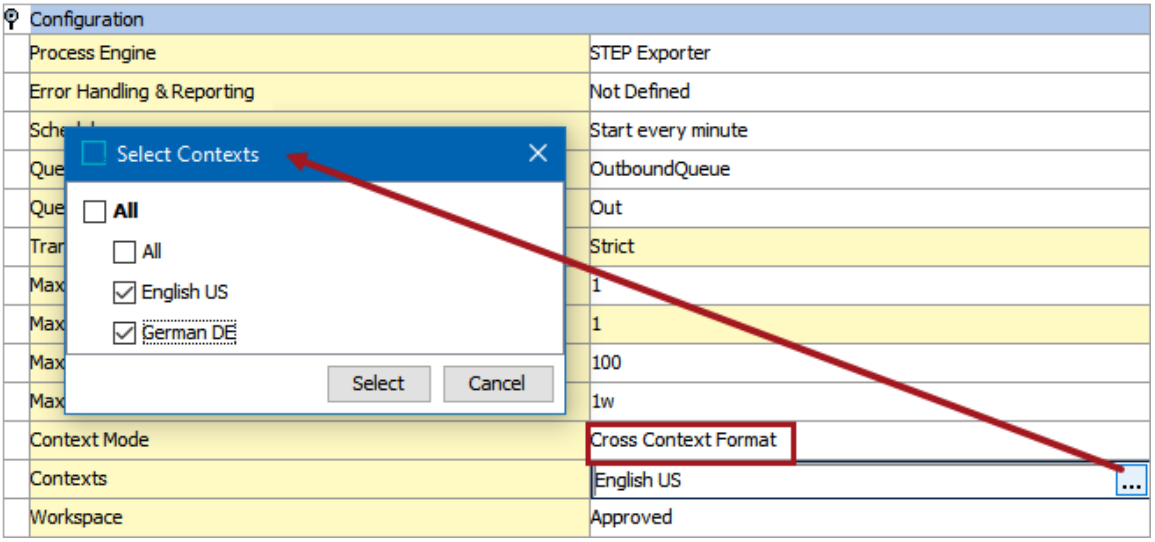
By default, the context is set to the one which is currently being used while creating the OIEP.

Changing Selected Contexts

The following image shows the Standard Format where a single context can be selected.



This image shows the Cross Context Format where a multiple contexts can be selected.



An explicit choice on whether the STEP Exporter should run in 'Standard Format' or 'Cross Context Format' mode is made when setting up the OIEP. It is possible to use cross context mode for endpoints with only a single context selected. This means that it will not be necessary to change downstream systems should more contexts be added later. For endpoints in 'Standard Format' mode (with just one context configured), it will not be possible to add more contexts without explicitly changing the export mode first.

1. Click the Contexts field to display an ellipsis button (...).
2. Click the ellipsis button (...) to display the 'Select Contexts' dialog.
3. Choose the desired contexts and click the **Select** button to save.

The configuration variants are shown in the examples below:

Single context, 'Standard Format':

Context Mode	Standard Format
Contexts	English US

Single context, 'Cross Context Format':

Context Mode	Cross Context Format
Contexts	English US

Multiple contexts, 'Cross Context Format':

Context Mode	Cross Context Format
Contexts	Danish Denmark, English US

When more than one context is selected, Context Mode becomes read-only and set to Cross Context Format (as shown above). The only way to change back to Standard Format is to deselect contexts until only one remains.

Workspace

Workspace defines which data is used for the export. Common setup is to use the Approved workspace, except when you need to generate events for objects before they are approved, for example, during import or from a workflow.

By default, the Approved workspace is selected.

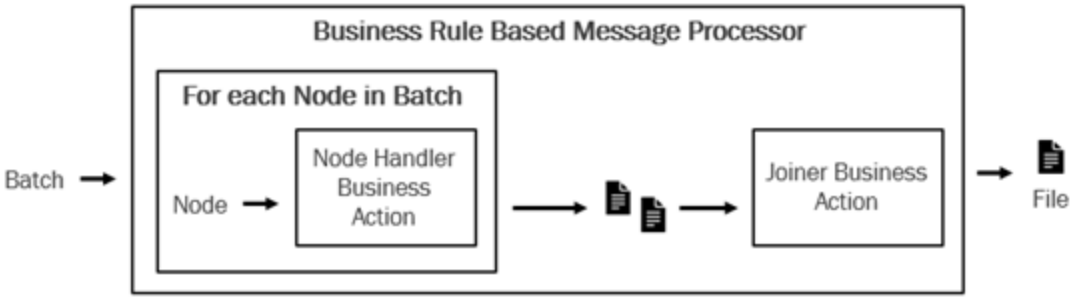
OIEP - Configuration Flipper for Business Rule Based Message Processor

The outbound Business Rule Based Message Processor allows you to compose the representation of STEP data to be exported using either JavaScript-based business actions or Java business actions developed via the Extension API.

For more information on JavaScript business actions, refer to the **Business Action Execute JavaScript** topic in the **Business Rules** documentation. For information on the Extension API, click the **STEP API Documentation** button on the STEP start page.

As illustrated below, the Business Rule Based Message Processor in an OIEP uses two business actions:

- A **node handler** is invoked once for each batch element to be exported and is responsible for producing the export representation of the node / event.
- A **joiner** has access to all representations produced by the node handler and is responsible for joining these into a single message to be delivered from STEP.



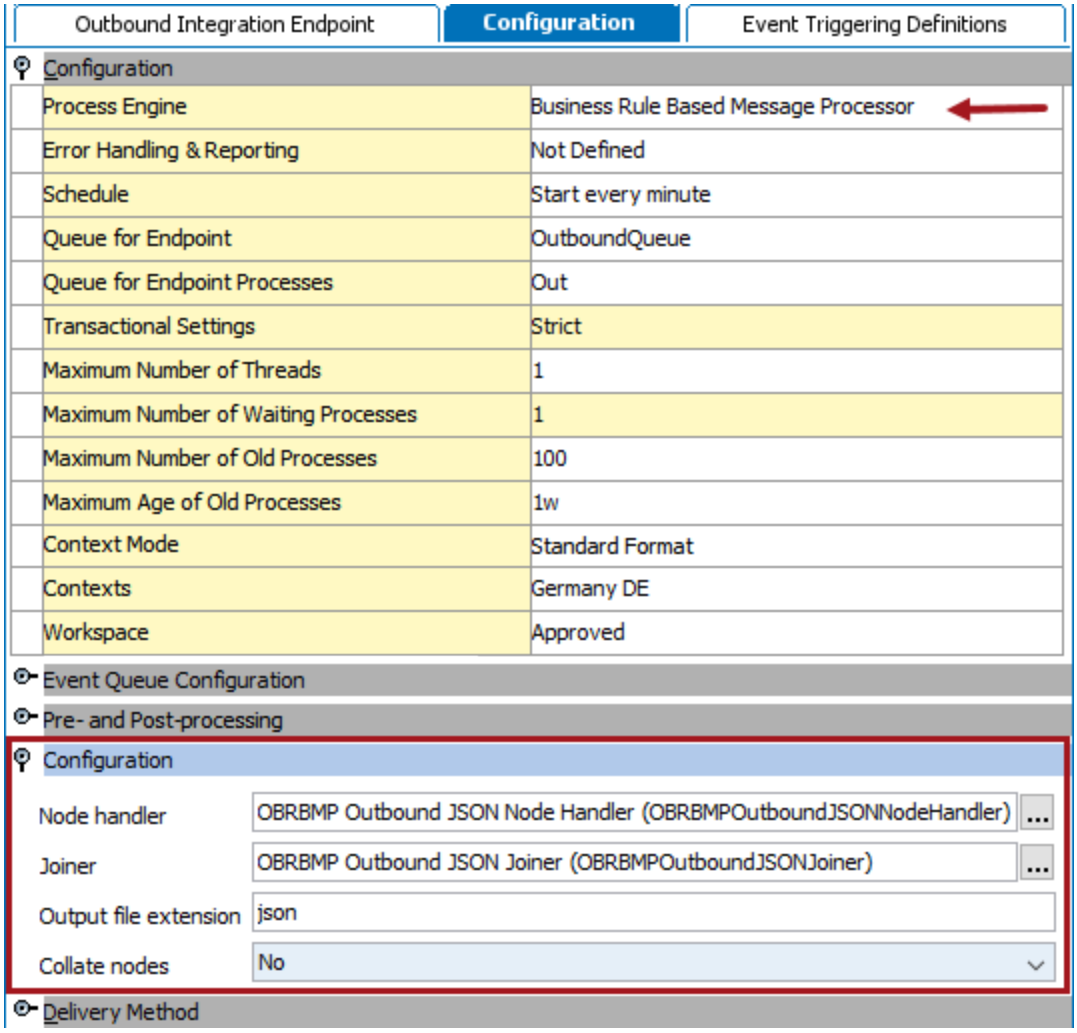
Prerequisites

1. Contact your Stibo Systems account manager or partner manager to activate the **Business Action Processor** commercial license. This enables the event-based importer and exporter to process inbound messages in custom / generic formats using JavaScript-based business rules, and enables exports to custom / generic formats including JSON using JavaScript business rules.
2. Business actions used by this processor should be configured to be valid for all object types.
3. Review the **Node Handler Business Action Details and Examples** and the **Joiner Business Action Details and Examples** sections below for more information about the business actions expected for this processor.

Configuration

A second Configuration flipper is available when the Business Rule Based Message Processor is selected for an OIEP. For information on selecting a process engine, refer to the **OIEP - Event-Based - Configure Endpoint** topic or the **OIEP - Select Objects - Configure Endpoint** topic.

This flipper includes the same parameters for both Event-Based and Select Objects endpoints. Each parameter is described below.



Set up the Business Rule Based Message Processor in the Configuration flipper:

1. For the **Node Handler** parameter, click the ellipsis button (...) to select the business action responsible for producing the representation of each node / event in the batch to be exported.
2. For the **Joiner** parameter, click the ellipsis button (...) to select the business action responsible for combining messages produced by the node handler into one message.
3. For the **Output file extension** parameter, enter the extension for the files to be exported.
4. The **Collate nodes** parameter is only applicable to event-based OIEPs, where a batch can contain multiple events for the same node. Select an option from the dropdown:
   - **Yes** - pass nodes to the node handler business action once per batch. Access is not available to information about the type of event that caused the node / event to be queued.
   - **No** - pass nodes /events as they occur, while having access to information about the type of event that caused the node / event to be queued.

Node Handler Business Action Details and Examples

The node handler business action is invoked once per node / event in the batch to be processed and is responsible for producing a textual representation of the data.

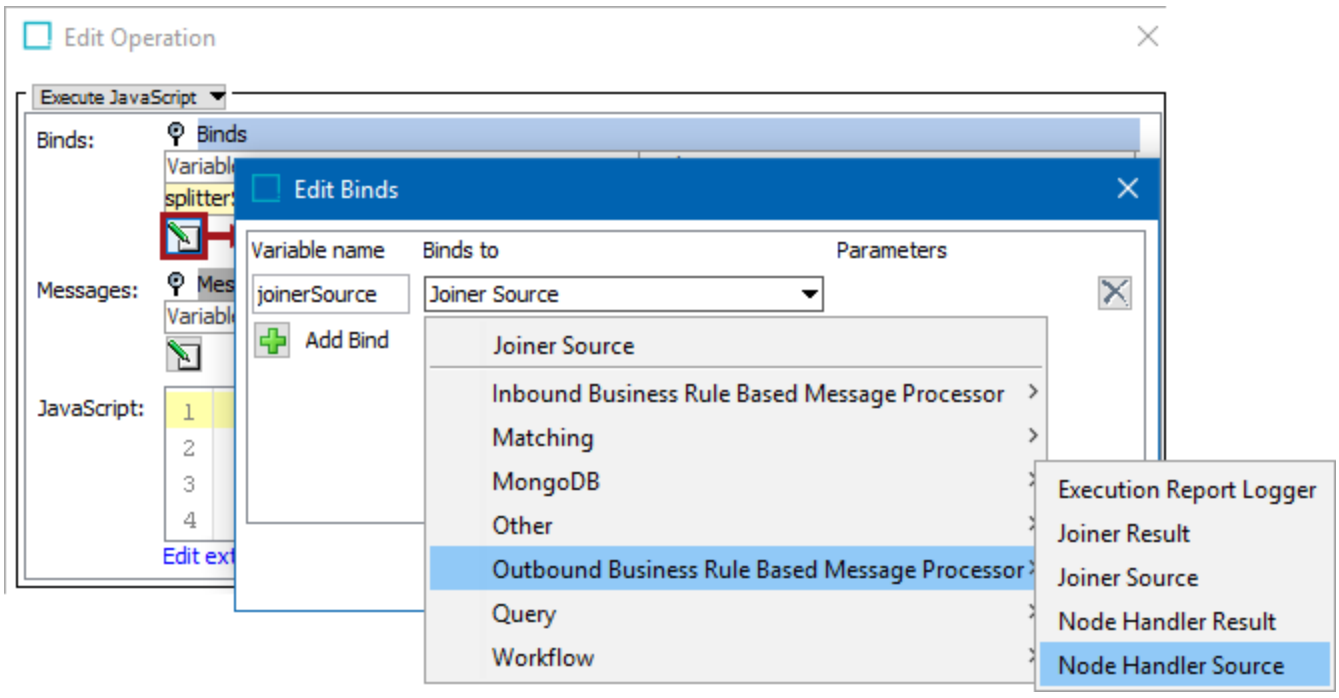
The **OutboundBusinessProcessorNodeHandlerSource** interface (for JavaScript business actions, the interface is available via the Outbound Node Handler Source bind option) provides the business action with access to the batch element. The interface has the following methods for accessing node / event information:

- `getNode(): Node:` The node to export / the node related to the event to be processed. This will always be the node as it looks now in the system. That is to say, the node will not for event-based publishing be a historical version representing the data as it looked when the event was generated. This method can return null if the node has been purged (`isPurged()` returns true).
- `getSimpleEventType(): SimpleEventType:` The event type. This is only available if the outbound integration endpoint is event-based and the 'Collate nodes' parameter is set to 'No.'
- `isDeleted(): boolean:` Returns true if the node related to the batch element being processed has been deleted. This covers the case where the node has been deleted but still exists in the recycle bin, and also the case where the node has been purged from the system.
- `isPurged(): boolean:` Returns true if the node no longer exists in the system.
- `getPurgedNodeID(): String:` If the node has been purged (`isPurged()` returns true), it is possible to get the ID of the purged node via this method. It is not possible to get the 'type' of the purged node, meaning that this method primarily makes sense for setups where only nodes of a single super type are published (since, for example, a product and a classification can have the same ID).

The **OutboundBusinessProcessorNodeHandlerResult** interface (for JavaScript business actions, the interface is available via the Outbound Node Handler Result bind option) is used by the node handler business action to pass the output via the following methods:

- `addMessage(String messageGroup, String message)` allows for messages to be added to what could be thought of as a named bucket. As the joiner business action (described below) can read messages from such named buckets. This allows, for example, all product 'upsert' messages (messages that 'update' or 'insert' as necessary) to be put into one bucket and all product 'delete' messages to be put into another, and the joiner can then add these messages to specific sections of the final combined message.
- `addMessage(String message)` adds a message to a generic anonymous group. Use this option if all message should be treated the same way by the joiner.

Further, the action has access to a standard Logger, a Manager and an **OutboundBusinessProcessorExecutionReportLogger**.



Notice that the node handler is not run in a transaction meaning that it is not possible to modify STEP data from the node handler.

JSON Message Node Handler JavaScript Business Action Example

This example shows how a simple JSON messages can be constructed and messages for 'upserts' and 'deletions' added to different message groups.

Note: `JSON.stringify()` cannot handle Java Strings and these therefore are converted to JavaScript strings when values are set for 'mesg' object properties.

```
// Node Handler Source bound to nodeHandlerSource
// Node Handler Result bound to nodeHandlerResult
// ExecutionReportLogger bound to executionReportLogger

var simpleEventType = nodeHandlerSource.getSimpleEventType();
if (simpleEventType == null) {
  executionReportLogger.logInfo("No event information available in node handler");
} else {
  executionReportLogger.logInfo("Event with ID '" + simpleEventType.getID()+ "'
passed to node handler");
}
var node = nodeHandlerSource.getNode();
if (node != null && node instanceof com.stibo.core.domain.Product) {
  executionReportLogger.logInfo("Node handler handling product with URL: " +
node.getURL());
```

```
  var mesg = {};
  mesg.stepid = node.getID() + "";
  mesg.ean = node.getValue("EAN").getSimpleValue() + "";
  if (nodeHandlerSource.isDeleted()) {
    nodeHandlerResult.addMessage("delete", JSON.stringify(mesg));
  } else {
    mesg.category = node.getParent() == null ? null : node.getParent().getID() + "";
    mesg.shortDescription = node.getValue("ShortDescription").getSimpleValue() + "";
    mesg.manufacturerName = node.getValue("ManufacturerName").getSimpleValue()+ "";
    mesg.color = node.getValue("Color").getSimpleValue()+ "";
    nodeHandlerResult.addMessage("upsert", JSON.stringify(mesg));
  }
}
```

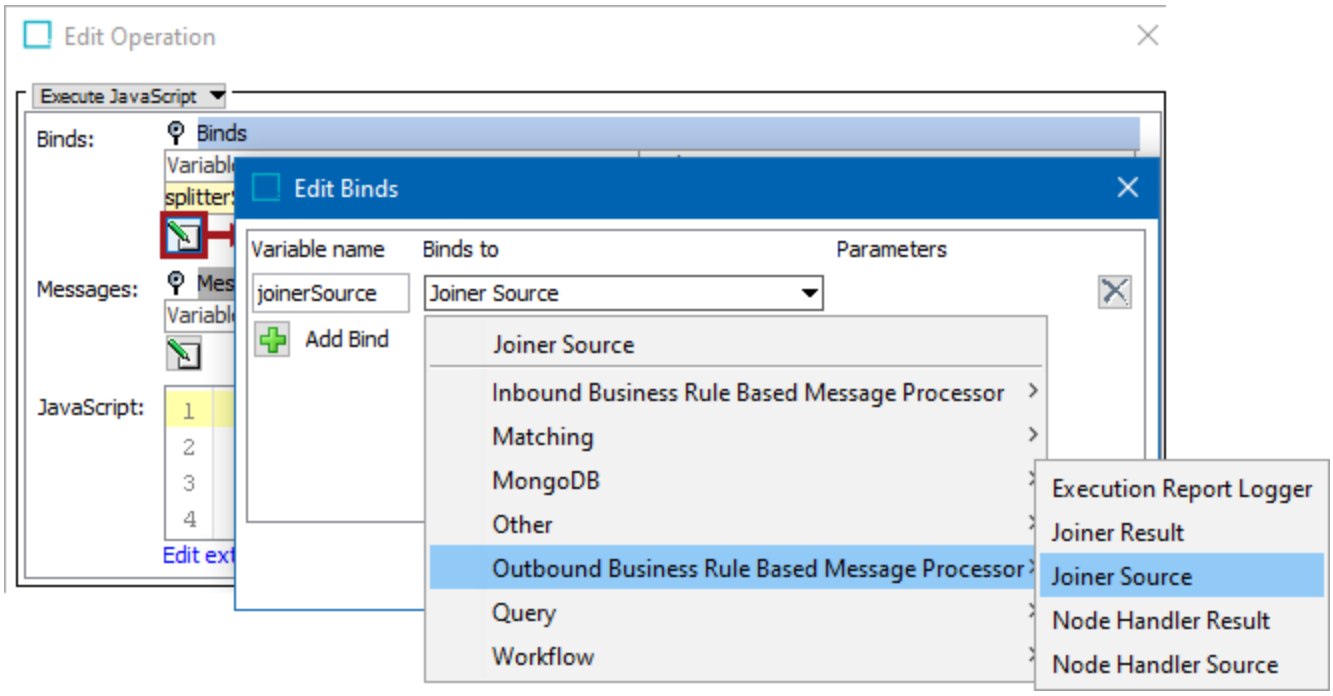
Joiner Business Action Details and Examples

The joiner business action is responsible for combining the messages produced by the node handler into a single message that can be delivered to downstream systems.

The **OutboundBusinessProcessorJoinerSource** interface (for JavaScript business actions, the interface is available via the Outbound Joiner Source bind option) provides the business action access to the messages.

The **OutboundBusinessProcessorJoinerResult** interface `appendToMessage(String)` method (for JavaScript business actions, this interface is available via the Outbound Joiner Result bind option) allows the business action to deliver its output. The `appendToMessage(String)` does not automatically apply any new lines or formatting.

The business action also has access to a standard Logger and an **OutboundBusinessProcessorExecutionReportLogger**.



Notice that the joiner is not run in a transaction meaning that it is not possible to modify STEP data from the node joiner.

JSON Message Joiner JavaScript Business Action Example

This example shows how messages from different message groups can be combined to a single message and further shows how duplicates can be avoided (there can be several elements in an event batch representing the same node).

```
// Joiner Source bound joinerSource
// Joiner Result bound to joinerResult

function appendFromGroup(messageGroup) {
  var seen = [];
  var first = true;
  while(joinerSource.hasNext(messageGroup)) {
    var messageString = joinerSource.getNextMessage(messageGroup);
    var hash = messageString.hashCode();
    if (seen.indexOf(hash) == -1) {
      seen.push(hash);
      if (first) {
        first = false;
      } else {
        joinerResult.appendToMessage(",");
      }
      joinerResult.appendToMessage(messageString);
    }
  }
}

joinerResult.appendToMessage("{\"products\":{\"upsert\":[");
appendFromGroup("upsert");
joinerResult.appendToMessage("],\"delete\":[");
appendFromGroup("delete");
joinerResult.appendToMessage("]}}");
```

OIEP - Event-Based - Output Templates Flipper

An output template defines the data to be exported and connects it to the desired format. This enables you to use one output format for a specific object type, combined with a specific event type, and another output format for a different object type, combined with a different specific event type. Without an output template, even if events are triggered (based on the Event Triggering Definitions parameters), nothing is exported.

At least one output is required, but for event-based OIEPs, using multiple output templates can allow different messages to be generated based on the object type / event type combinations. Keep in mind that batching is restricted to a single template, so only use multiple templates when required. Refer to more information below under the **Using Multiple OIEPs or Multiple Output Templates** section.

Configuring an output template involves the following steps:

- Configure the object types and event types
- Specify the format
- Configure pre-processor and post-processor, based on availability on your system

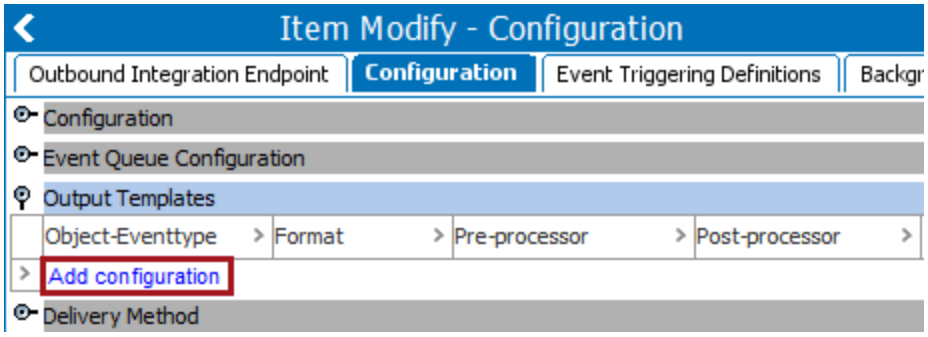
Important: For an event-based outbound integration endpoint, an output template must be created for each event trigger, otherwise the trigger will not output any data. For more information, refer to **OIEP - Event-Based - Event Triggering Definitions Tab**.

Configure the Object Types and Event Types

The object type and event type selections determine the STEP data that will be exported. Different output templates (export configurations) can be used to export data based on the event type (core events or derived events) and the object type for which the event is generated.

Important: When exporting delete events, generally events are processed in the same order as they occurred. However, in some circumstances, the order is not reliable. For more information, refer to the **Event-Based OIEP Order of Delete Events** topic.

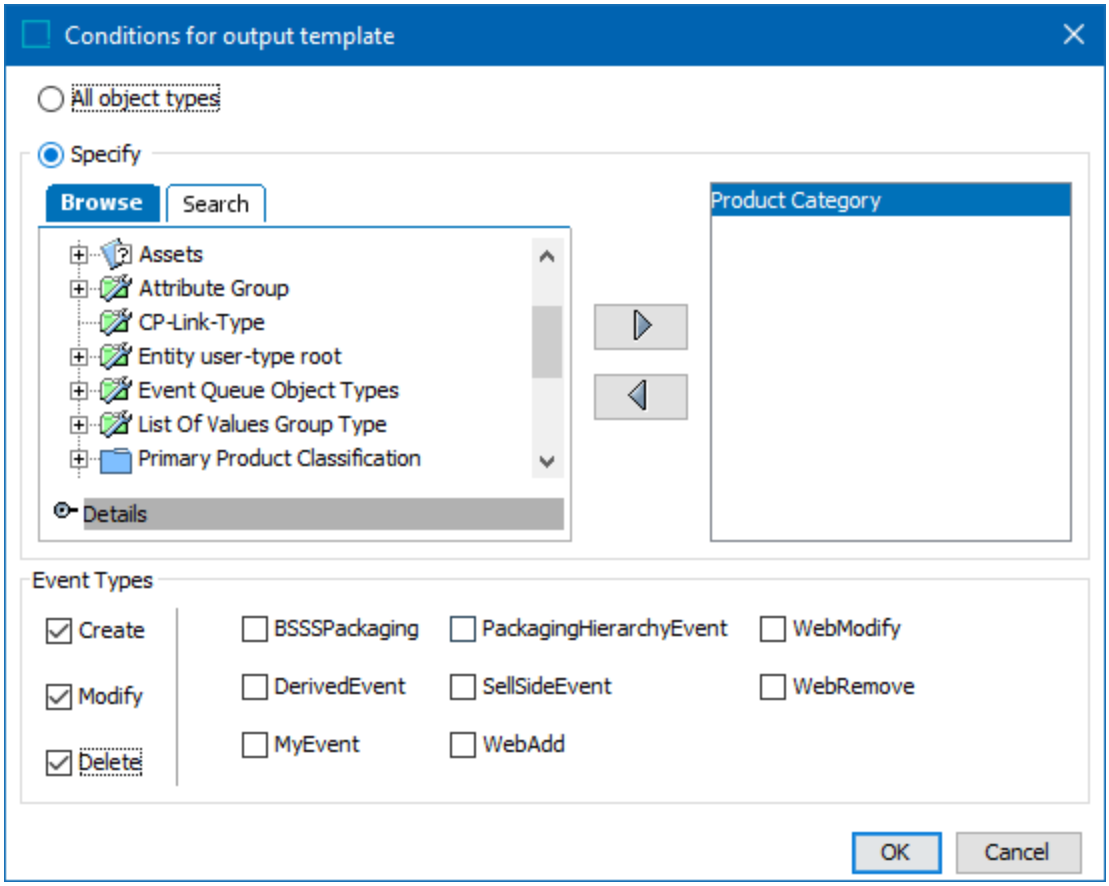
1. In System Setup, select the relevant OIEP > Configuration tab > Output Templates flipper > click the **Add configuration** link to display the 'Conditions for output template' dialog.



2. In the 'Conditions for output template' dialog:
   - For the **Object Types** area at the top, click 'All object types' or 'Specify' to select the object type(s) using the Browse or Search options. Click the right arrow button ( ) to move the object type to the right side of the dialog. To remove an object type, click the left arrow button ( ).
   - For the **Event Types** area at the bottom, select the event type. Core event types (create, modify, and delete) display on the left side and available derived event types display on the right side. Unlike core events, derived events must be created before they will display. For more information, refer to the **Derived Events** section of the **System Setup** documentation.

     When using the Mongo delivery method, specific settings are required to deliver delete events. For details, refer to the **Delivering Delete Events** section of the **Mongo Delivery Method** topic in this guide.

     For delete events in particular, the order of processing can vary based on the scenario. For details, refer to the **Event-Based OIEP Order of Delete Events** topic in this guide.



Note: Selecting an object and event type defines the object types for export, but does not necessarily define the objects that are included in the actual message, since the format of the template may dictate that parent, child, or referenced objects are included. Additionally, the advanced settings for format determine if the selected objects, their children, or both should be considered for the export.

3. Repeat the process until all required formats are displayed, keeping in mind the limitations of multiple output templates defined in the next section.

   Note: Only one (1) output template is allowed for each Object type and Event type combination.

   In the image below, all of the available system event types (create, modify, delete) were selected for each template created:

Output Templates

Object-Eventtype	Format	Pre-processor	Post-processor
Product Category (Create, Modify, Delete)	Generic XML (1 mappings)	None	None
Product Family (Create, Modify, Delete)	Generic XML (1 mappings)	None	None
Product (Create, Modify, Delete)	Generic XML (1 mappings)	None	None
Add configuration			

4. If necessary, edit existing Object-Event type settings by clicking into the desired field to display an ellipsis button (...). Click the ellipsis button (...) to display the 'Conditions for output template' dialog discussed above.

Using Multiple OIEPs or Multiple Output Templates

Generating output for multiple recipients can be accomplished in the following ways, which should be evaluated for best outcome.

Multiple OIEPs

- Triggering events for event-based endpoints are global. For example, you may want a change in STEP name to trigger an event on some objects, but not others. This scenario requires a different configuration of triggers on multiple endpoints.
- Delivery method is per endpoint, not per output format. If you need a different output to go to each external system, you must use multiple endpoints.
- Messages generated via events and those generated via object selection cannot be combined in a single endpoint. Even when the format is the same, this scenario requires separate message generation and therefore requires separate endpoints.

Multiple Output Templates

Multiple output templates can be created to allow data output based on a combination of specific object types and event types.

Important: Be aware that events are processed sequentially and are added to batches, as allowed, by the required output templates. Each batch also involves overhead processing operations. Depending on the data being exported, multiple output templates can result in many batches with relatively few events in each. To optimize processing, it is better to have fewer large batches instead of many small ones.

Using multiple output templates prevents the same data from being exported across all chosen object types and event types. Multiple output templates also help to reduce data output, which increases performance both for STEP and for downstream systems. In addition, the use of multiple output templates should reduce the need for customizations.

For example, consider an event-based OIEP that is configured so that when a change is made to a 'Product Category' object, the export *should not* include child objects, but when a change is made to a 'Product Family' object, the export *should* include child products. This can be achieved by configuring multiple XML output templates, and by selecting the specific object and event type.

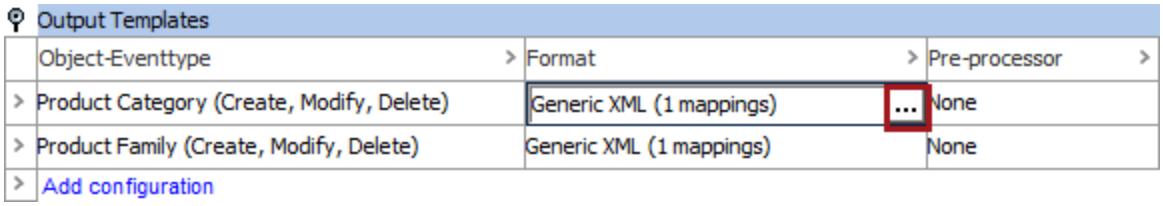
Limitations of Multiple Output Templates

- Output files are only delivered to one destination, regardless of the number of output templates.
- Output templates encompassing all object types are only available if all object types are chosen when making the configuration. This prevents excess output from the endpoints.
- Batching occurs per template type.
- Different output formats cannot be defined for the same object type. Common setup is to avoid combining different output formats (e.g., STEPXML and CSV) in the same endpoint.

Configure the Format

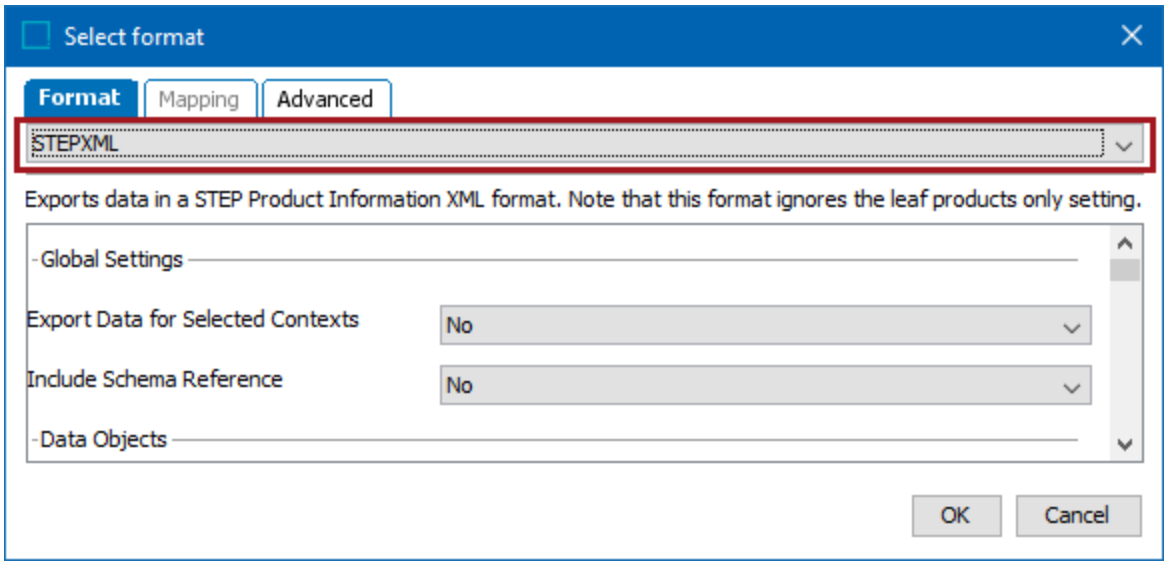
The format selection determines how the STEP data is formatted (XML or tabular).

1. Click into the format column to display the ellipsis button (...). Click the ellipsis button (...) to display the 'Select format' dialog.



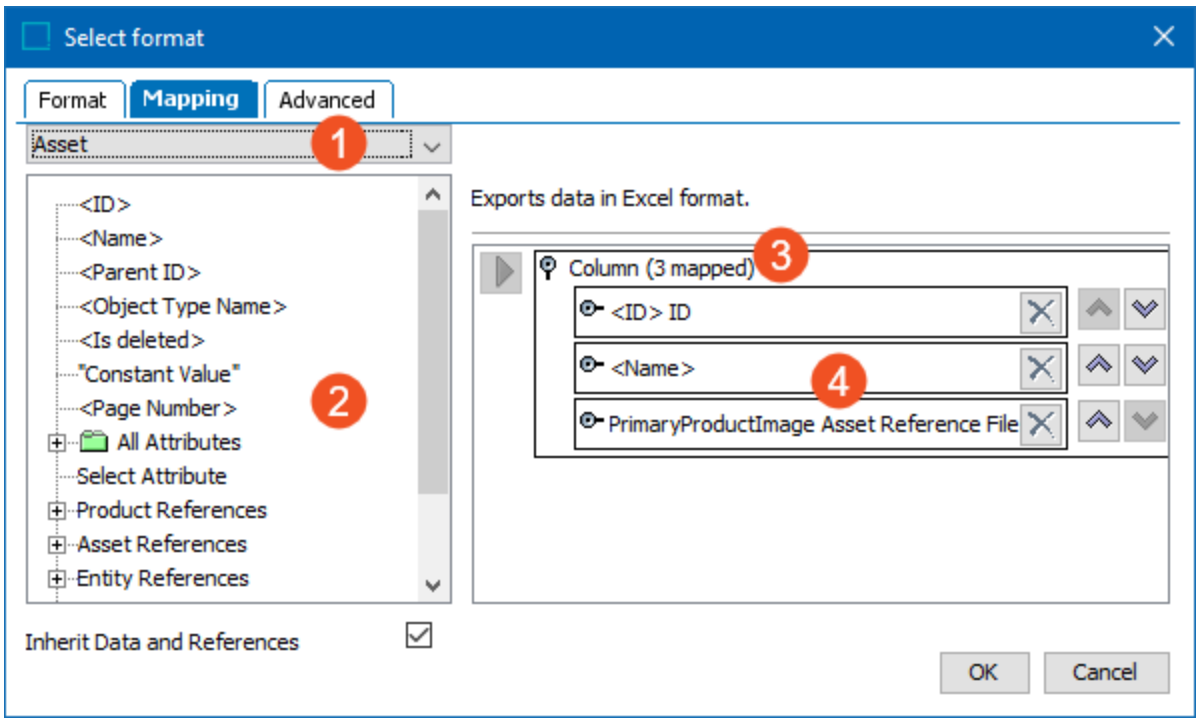
2. In the **Select format** window, on the **Format** tab, select the desired format from the dropdown. The options are the same as in the Export Manager, based on your system licenses. Use the following links for additional information:
   - Advanced STEPXML Format
   - Alphabetical Index - XML Format
   - Ariba CIF Format
   - Ariba CIF 3.0 Format
   - BMEcat Format
   - BMEcat 2005 Format
   - CSV Format
   - cXML Format

- Excel Format
- Excel Smartsheet Format
- Flatplanner Format
- Generic XML Format
- Generic JSON Format
- IDoc MATMAS05 Format
- Publication Excel Format
- STEPXML Format
- STEPXML Configuration Export Format
- xCBL Format



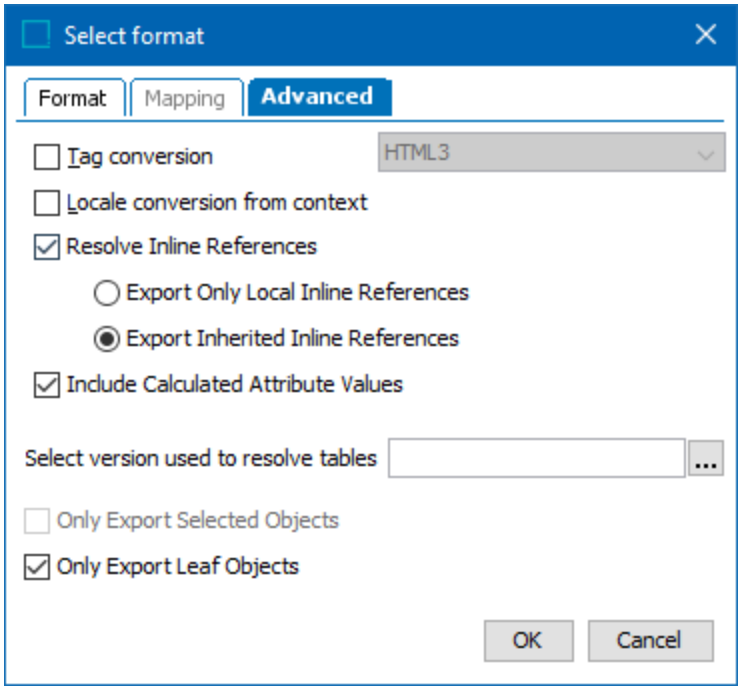
3. When available, specify how to map data to the export format, including the elements identified in the image below: object super type dropdown, data source, mapping target, and mapping rules.

   For more information about the object super type dropdown, refer to the **Export Manager - Select Objects** topic. For more information about the other elements, refer to the **Export Manager - Map Data** topic.



Note: The **Mapping** tab is enabled when the format selected requires mapping. Selected formats, including STEPXML and Advanced STEPXML, are automatically mapped so the Mapping tab is disabled.

4. On the **Advanced** tab, a number of advanced export options can be specified.



- **Tag conversion**: Converts tags to match the selected output format. This setting is optional.

  If this box is left unchecked, any formatting tags that are included in attribute values will not be converted in the outbound message. For example, if an attribute value contains bold text (which must be made bold with a STEP Style Tag, e.g., <bold>/</bold>), this tag will not be converted to its corresponding HTML output format (e.g., <B>/</B>) in the export. For more information on STEP Tags, refer to the **Tags** topic in **System Setup** documentation.

- **Locale conversion from context**: Converts numbers into the numeric format that corresponds to the selected locale. By default, this option is not checked. **Note**: If Smartsheets are used for format, this must remain unchecked.
- **Resolve Inline References**: Resolves inline references when they are exported. This box is checked by default, along with the option 'Export Inherited Inline References.'

  If unchecked, inline references will be exported with the inline reference tagging instead of the actual content pulled in by the inline reference. For example, if an attribute called 'Product Number' has an inline reference to pull in the STEP ID of the object (e.g., 12345), the attribute value will not contain 12345. Instead, it will contain tagging similar to the following:

  ```
  <ref attrid="" equalsign="=" includeattrname="false" resolveto="objid"
  separator=";"/>
  ```

  This setting would be valid if the user planned to re-import the file at a later time and not overwrite the inline reference with a static value. For more information on inline references, refer to **Inline References in Attribute Values** in the **Getting Started** documentation.

- **Include Calculated Attribute Values**: Resolves calculated attribute values when they are exported. By default, this option is checked. If not checked, calculated attribute values are exported with empty values, unless the Force Calculation option has been set. The Force Calculation option on the Transformations dialog is available when mapping a calculated attribute using a transformation. For more information, refer to **Outbound Map Data - Transform**.

  The value template is exported for each selected context, including a qualifier ID, which makes it possible to import the same data back into STEP. For information about calculated attribute values, refer to the **Calculated Attributes** topic in **System Setup** documentation.

Important: If many complex calculated attribute values (traversing hierarchies and/or references) are used, consider if they should be exported, since they can impact performance. If required, consider scheduling for non-peak times. Simple calculations are not detrimental to an export, regardless of the quantity.

- **Select version used to resolve tables**: This setting is available for STEPXML if 'Include Tables' is set to Yes and for Advanced STEPXML if the template includes version-dependent content. A publication version should be selected if the tables contain content relevant to a particular publication (such as column or row types that are only valid for certain publication types) or a publication version (such as commercial data). For more information, refer to the **Exporting Resolved Tables** topic in the **Tables** documentation.
- **Only Export Selected Objects**: Specifies that only objects from the output template are exported. No children are exported. This setting is unchecked by default and is only available if 'Only Export Leaf Objects' is unchecked. This setting does not apply to STEPXML.

- **Only Export Leaf Objects**: Specifies that only the leaf objects (lowest level of the export object for both events and selected objects) of the selected top hierarchy are included in the export. Selected objects and triggering objects are only included if they have no children. This setting applies to the CSV or Excel format.

Note: Uncheck both 'Only Export Selected Objects' and 'Only Export Leaf Objects' if all objects (both for triggering events and object selection) and their children should be included in the export.

For more information, refer to **Export Manager - Advanced** documentation.

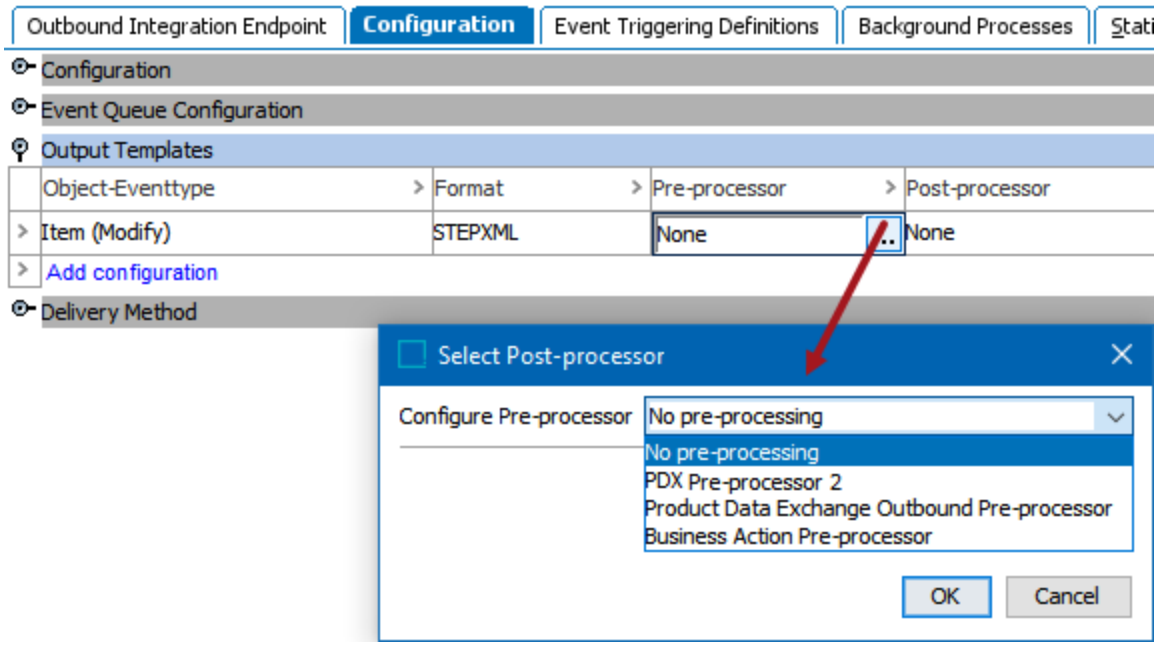
Configure the Pre-processor and Post-processor

Pre-processor and post-processor options are available depending on your system configuration.

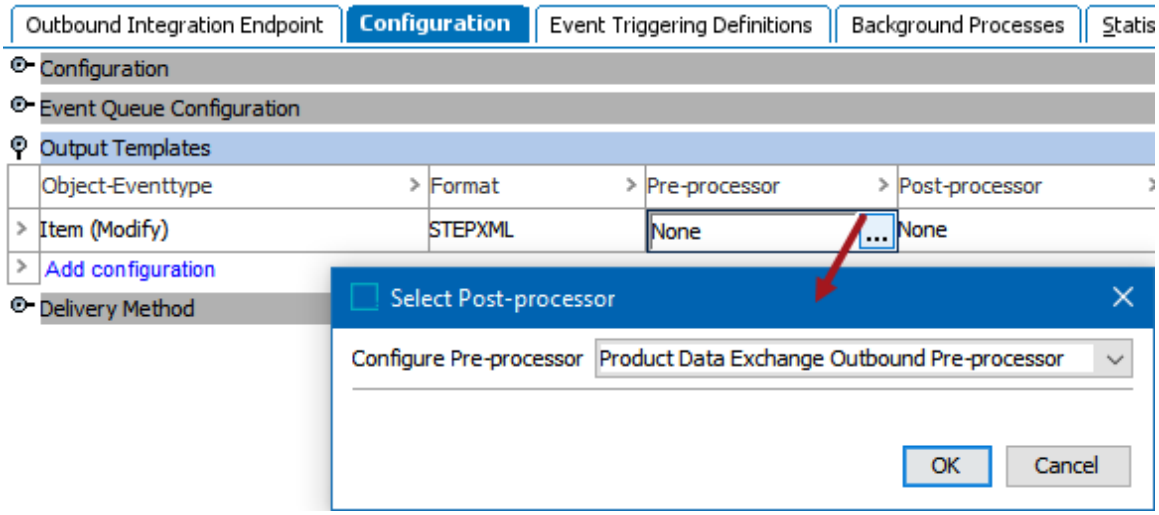
Pre-processor

- **Business Action Pre-processor** can be applied on event-based OIEPs before exporting events from a batch in an event queue. Applying the pre-processor means a node can be added to a current batch, or an event can be removed from the current batch before a message is created for an export. You can configure a pre-processor for each of the output templates. For more information, refer to **OIEP - Pre-processor - Business Action**.

  A Triggering Object Type event filter or Generate Event option can be used in place of a pre-processor. For more information, refer to the **OIEP - Event-Based - Event Triggering Definitions Tab** documentation.



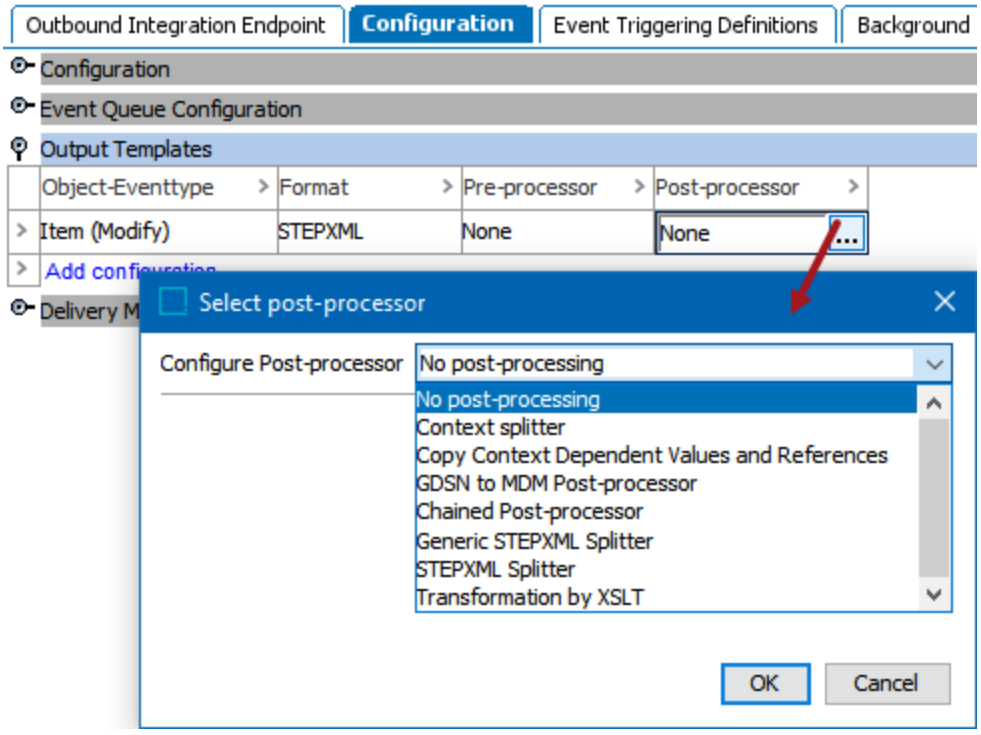
Product Data Exchange Outbound Pre-processor can be used to handle change events on reference assets, and forwards this information to the PDX product. For more information, refer to the Setting Up the PDX OIEP topic in the Product Data Syndication section of the Data Integration documentation.



Post-processor

Standard post-processing options should be evaluated if multiple contexts are included in the output. You can configure a post-processor for each of the output templates. The Triggering Object Type generate event option can be used in place of a post-processor. For more information, refer to the **OIEP - Event-Based - Event Triggering Definitions Tab** documentation.

1. Click into the Post-processor column to display the ellipsis button (...). Click the ellipsis button (...) to display the 'Select Post-processor' dialog.



2. In the **Configure Post-processor**, select from the following options:

- **No post-processing** exports files using the standard export functionality.
- **Context splitter** generates an export file for each configured context. Each exported file will only contain context specific data. For more information, refer to the **OIEP - Post-processor - Context Splitter** topic.

  For the **Copy inherited product values** parameter, select **Yes** to copy and save inherited product values to the child product; or select **No** to use inherited product values in the export, but leave the child product unmodified.

  When using Excel or CSV format, and multiple contexts are configured for export, you must select Context Splitter, or a single context will be included in the export.
- **Copy Context Dependent Values and References** to export context-dependent values and references and add the corresponding ContextID attribute to the value or reference in question. The endpoint generates one file containing values and references for each context specified. For more information, refer to the **OIEP Post-processor - Copy Context Dependent Values and References** section.

  For the **Copy inherited product values** parameter, select **Yes** to copy and save inherited product values to the child product; or select **No** to use inherited product values in the export, but leave the child product unmodified.

  If you use the **Copy Context Dependent Values and References** post-processor to add ContextIDs to a cross-context export (exporting data for selected contexts), the downstream system can use the <ContextID> tag to identify specific contexts. Additionally, this facilitates re-importing the exported data into the correct contexts in STEP after processing by a third-party service or application.
- **Chained Post-processor** as defined in the **OIEP - Post-processor - Chained Post-processor** topic.
- **Generic STEPXML Splitter** splits up STEPXML messages to multiple STEPXML valid fragments containing one single node per STEPXML file.
- **STEPXML Splitter** produces one file per STEP object as defined in the **'STEPXML Splitter Post-processor for OIEP'** section of the **Integration Endpoint Options for VCS Integrations** topic in the **Configuration Management** documentation.
- **Transformation by XSLT** transforms data via an XSLT stylesheet before exporting. Under the XSLT-Stylesheet parameter, click the ellipsis button (...) and browse or search for the stylesheet to use for the transformation. For more information, refer to **OIEP - Post-processor - Transformation by XSLT** section of the **Data Exchange** documentation.
- **GDSN to MDM Post-processor** handles an exported STEPXML file that may include many packaging hierarchies and many objects within each packaging hierarchy. The post-processor will split these files up by STEP context and by trade item before applying transformations. The transformed files will have a hash code generated and compared to a previous hash code to verify if there have been changes. Optional business actions can also be executed to affect workflows or other purposes. For more information, consult your Stibo Systems account manager or partner manager.

  Note: This post-processor is only available when running the GDSN to Receiver component.

OIEP - Pre-processor - Business Action

By default, no pre-processor is included in an OIEP output template. Applying the Business Action Pre-processor means a business rule can be run against the data, prior to the export. For example, a node can be added to a current batch, or an event can be removed from the current batch before a message is created for an export.

Using this pre-processor requires:

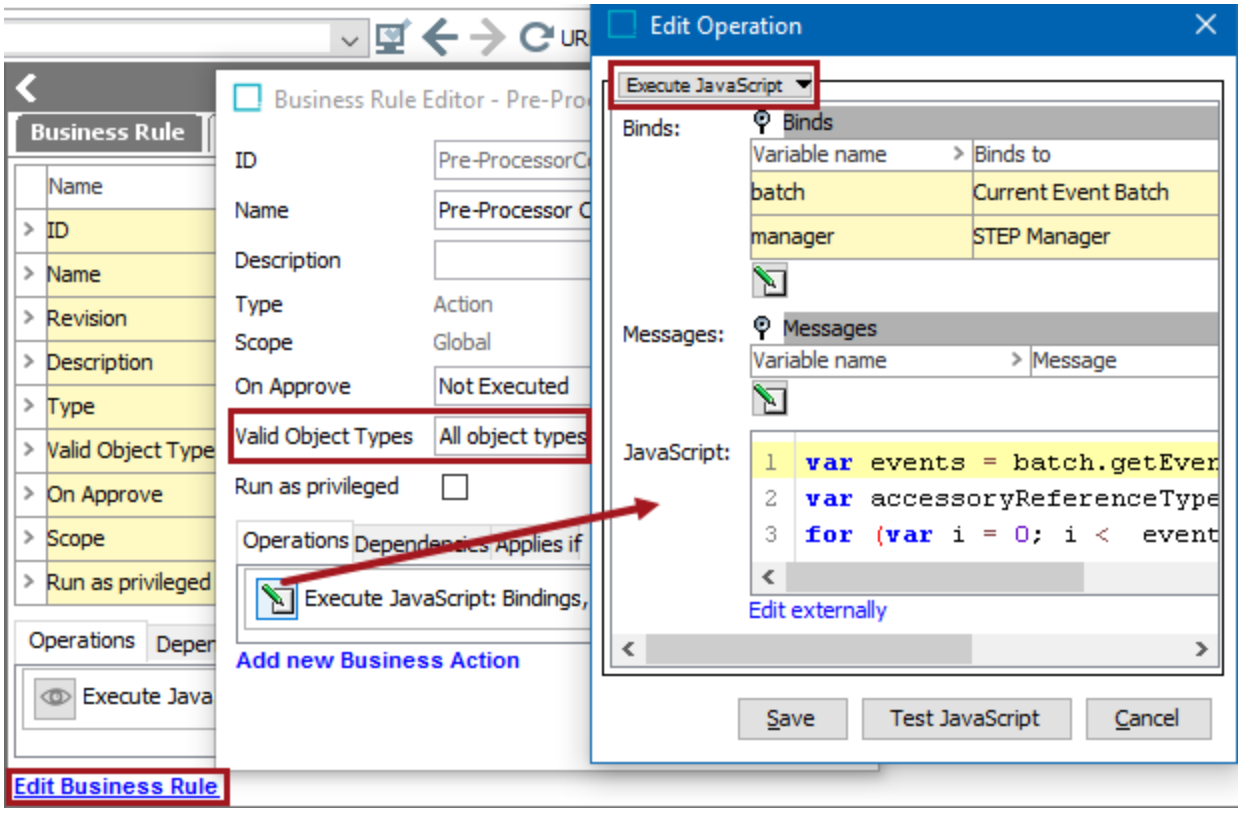
1. A business action - in this example, it will add referenced products to the products that are contained in the message.
2. An event-based OIEP configured with the business action - in this example, it is configured to deliver messages that inform a downstream system about changes to a product.

Business Action Configuration

Create a business action that is valid for 'All Object Types.' For more information on creating business rules, refer to the **Specifying a Business Action Operation** topic.

For a full JavaScript code example, refer to the online version of this topic.

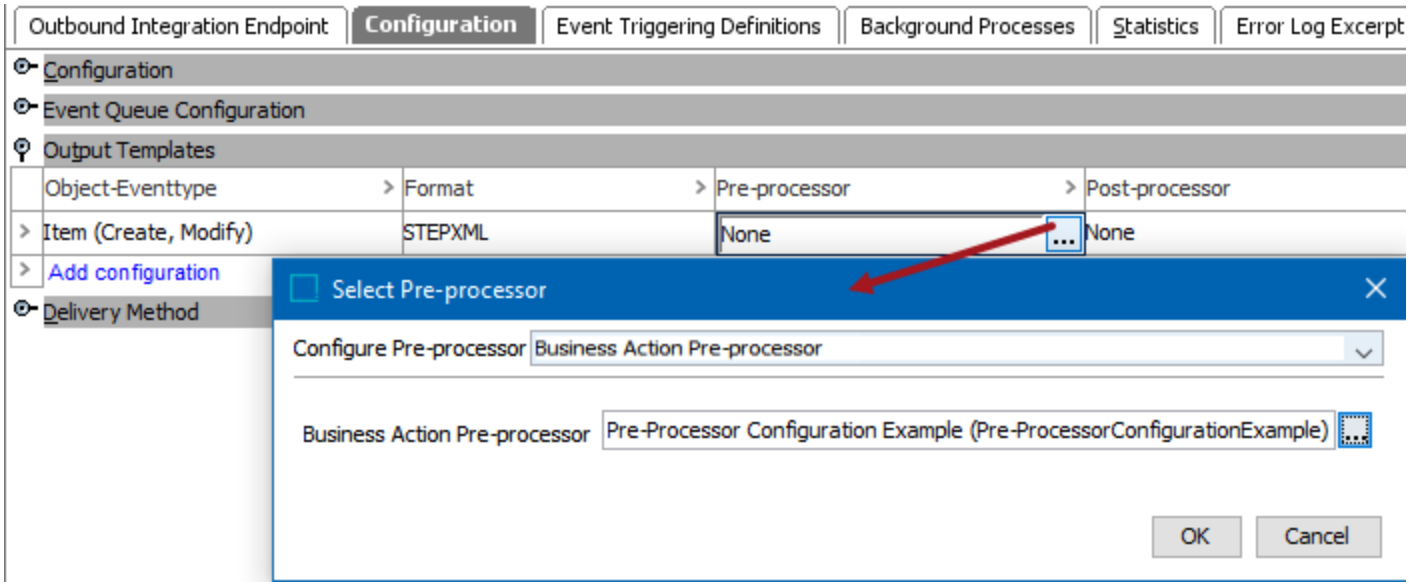
The business rule is configured in the System Setup tab under Global Business Rules as displayed below:



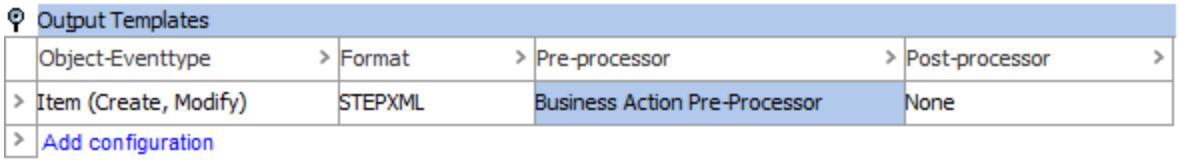
OIEP Configuration

Apply the business action as a pre-processor for any of the output templates on an OIEP.

1. Create an event-based OIEP. For details, refer to the **Creating an Event-Based Outbound Integration Endpoint** documentation.
2. On the **Configuration** tab, open the **Output Templates** flipper and locate the output template that requires the business action.



3. In the **Pre-processor** column, click the ellipsis button (...) to display the Select Pre-processor dialog.
4. In **Configure Pre-processor**, select the **Business Action Pre-processor** option.
5. In **Business Action Pre-processor**, select the business action created earlier. The output template is updated to show the pre-processor will run.



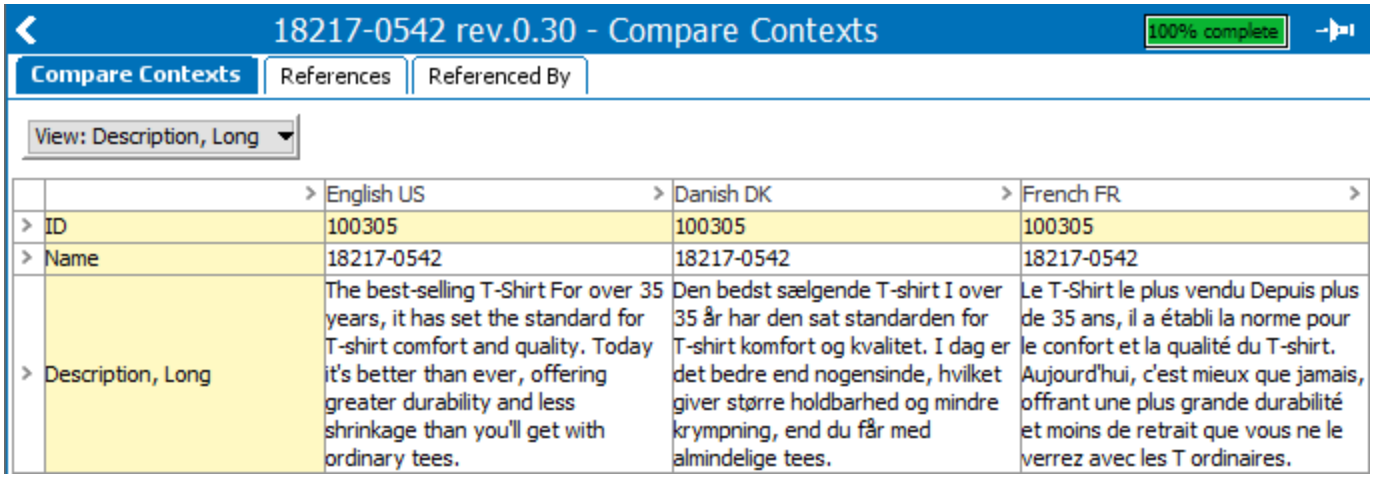
OIEP - Post-processor - Context Splitter

The Context Splitter post-processor splits the export file into individual files based on the contexts on which the data is dependent. This can be used when the data from different contexts should be sent to the external systems separately. It is also commonly used for reporting purposes for individual contexts.

The context splitter adds the <ContextID> without a <Qualifier ID>. This makes the file similar to an export of local values done from a single context. The <ContextID> tag specifies the ID of the context in which the object is exported, but does not provide the specific dimension point details which can only be supplied by the <QualifierID> tag. Using the context splitter assumes that the external system does not have any details of dependency points from STEP.

The following example illustrates using an outbound integration endpoint with the Context Splitter post-processor. It includes attribute values, but the post-processor also works with inherited references such as classification or asset references.

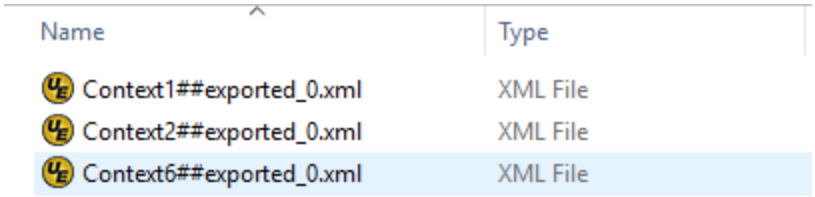
The product has a language-dependent attribute, 'Description, Long', with values defined in three contexts: English US, Danish DK, and French FR.



	English US	Danish DK	French FR
ID	100305	100305	100305
Name	18217-0542	18217-0542	18217-0542
Description, Long	The best-selling T-Shirt For over 35 years, it has set the standard for T-shirt comfort and quality. Today it's better than ever, offering greater durability and less shrinkage than you'll get with ordinary tees.	Den bedst sælgende T-shirt I over 35 år har den sat standarden for T-shirt komfort og kvalitet. I dag er det bedre end nogensinde, hvilket giver større holdbarhed og mindre krympning, end du får med almindelige tees.	Le T-Shirt le plus vendu Depuis plus de 35 ans, il a établi la norme pour le confort et la qualité du T-shirt. Aujourd'hui, c'est mieux que jamais, offrant une plus grande durabilité et moins de retrait que vous ne le verrez avec les T ordinaires.

XML Output Example – Context Splitter

Using the context splitter, a separate file is generated for each context. As shown below, the names of the files output identify the Context IDs from which the objects were exported.



The <ContextID> tag has been added to the export. The files can be distinguished by the Context ID and there is no <Qualifier ID> or any information about the dimension points. The XML output for each of the files is:

```
<STEP-ProductInformation ExportTime="2017-08-07 11:51:18" ExportContext="Context6" ContextID="Context1" WorkspaceID="Main" Us
  <Products>
    <Product ID="100305" UserTypeID="SalesItem" ParentID="18206">
      <Name>18217-0542</Name>
      <Values>
        <Value AttributeID="DescriptionLong">The best-selling T-Shirt For over 35 years, it has set the standard for T-shirt
```

```
<STEP-ProductInformation ExportTime="2017-08-07 11:51:18" ExportContext="Context6" ContextID="Context2" WorkspaceID="Main" Us
  <Products>
    <Product ID="100305" UserTypeID="SalesItem" ParentID="18206">
      <Name>18217-0542</Name>
      <Values>
        <Value AttributeID="DescriptionLong">Le T-Shirt le plus vendu Depuis plus de 35 ans, il a établi la norme pour le con
```

```
<STEP-ProductInformation ExportTime="2017-08-07 11:51:18" ExportContext="Context6" ContextID="Context6" WorkspaceID="Main" Us
  <Products>
    <Product ID="100305" UserTypeID="SalesItem" ParentID="18206">
      <Name>18217-0542</Name>
      <Values>
        <Value AttributeID="DescriptionLong">Den bedst sælgende T-shirt I over 35 år har den sat standarden for T-shirt komfo
```

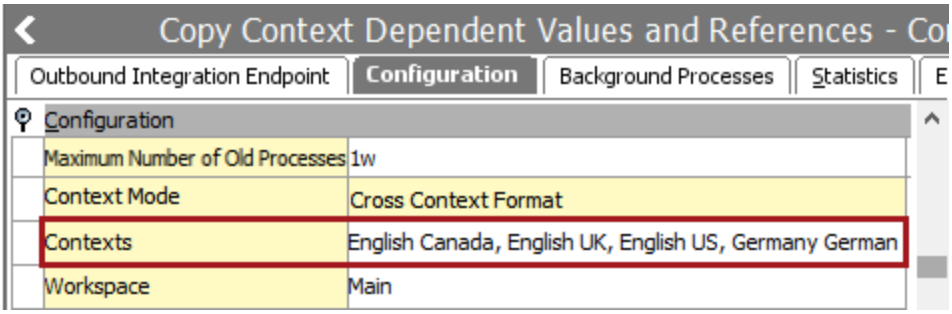
XML Output Example - Standard Cross-Context Export

Using a standard cross-context export (exporting data for selected contexts) without the context splitter, the single output file includes only the attribute values specific to particular dimension points (indicated by QualifierIDs). The XML output is:

```
    <ValueGroup AttributeID="DescriptionLong">
      <Value QualifierID="en-US">The best-selling T-Shirt For over 35 years, it has set the stan
      <Value QualifierID="fr">Le T-Shirt le plus vendu Depuis plus de 35 ans, il a établi la nor
      <Value QualifierID="Danish">Den bedst sælgende T-shirt I over 35 år har den sat standarden
    </ValueGroup>
```

OIEP - Post-processor - Copy Context Dependent Values and References

When multiple contexts are selected in the Configuration flipper of an OIEP, this post-processor reduces the complexity of inheritance of values in cross-context exports, and allows downstream systems to receive a multi-context export in a single file.



Important: Using the 'Copy Context Dependent Values and References' post-processor requires a strong understanding of the STEP Qualifier concept that defines in which dimensions and at which inheritance levels values are present.

If you want to further process the information in a form similar to a typical single-context export, we strongly recommend using the context-splitter post-processor instead. For more information, refer to the **OIEP - Post-processor - Context Splitter** topic.

ContextID and QualifierID XML Tags

The 'Copy Context Dependent Values and References' post-processor adds the <ContextID> and <QualifierID> XML tags to the export. A standard cross-context export (exporting data for multiple contexts) only adds the <QualifierID>. The <QualifierID> tag specifies the ID of the Dimension Point in which the attribute value exists but does not provide the specific context. For example, the <QualifierID> of 'en-US' could be the ID of the 'English' dimension point, but without the <ContextID> tag, the additional contexts that an attribute value has inherited down to cannot be determined.

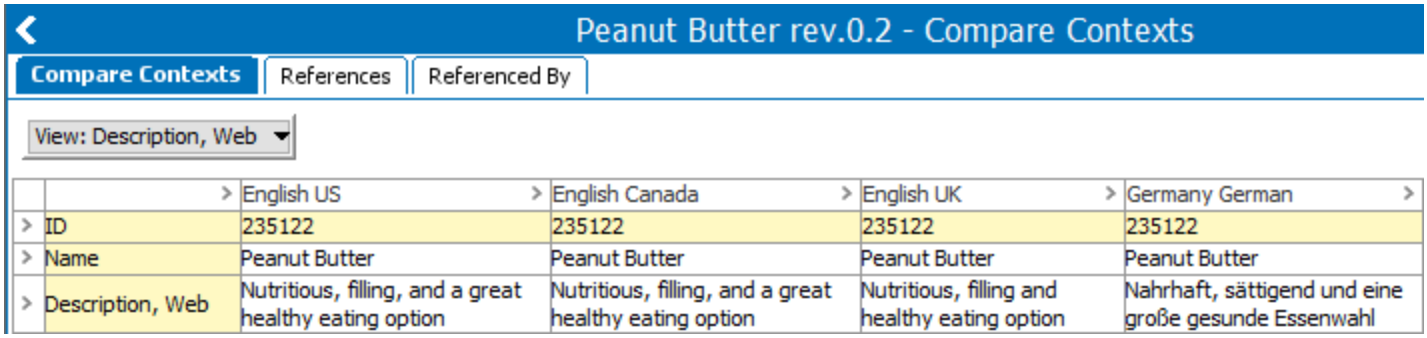
When the <ContextID> XML tag is used in combination with the <QualifierID> XML tag, it is possible for the downstream system to determine the inheritance relationship between a value (or reference) and a context. The <QualifierID> XML tag is useful when a service tries to re-import processed data into STEP. Because the downstream system knows which context the data must be imported into, it is easier to target specific contexts within STEP for import.

The following example illustrates the differences between using the Export Manager for a standard cross-context export (exporting data for multiple contexts) and using an outbound integration endpoint with the **Copy Context Dependent Values and References** post-processor enabled.

Although attribute values are used, the post-processor also works with inherited references such as classification or asset references.

The product with ID '235122' has a language-dependent attribute, 'Description, Web.' The following contexts will be considered:

- 'English US' (Context 1) has a value defined and is the master context (determined by its selection in the Context dropdown
- 'English Canada' (Context3) does not have a value defined but inherits values from the English US context
- 'English UK' (Context7) has a value defined (notice the slightly different wording)
- 'Germany German' (Context5) has a value defined



	English US	English Canada	English UK	Germany German
ID	235122	235122	235122	235122
Name	Peanut Butter	Peanut Butter	Peanut Butter	Peanut Butter
Description, Web	Nutritious, filling, and a great healthy eating option	Nutritious, filling, and a great healthy eating option	Nutritious, filling and healthy eating option	Nahrhaft, sättigend und eine große gesunde Essenwahl

XML Output Example - Standard Cross-Context Export

Using a standard cross-context export (exporting data for selected contexts), the following XML is generated:

```
<?xml version="1.0" encoding="utf-8"?>
<STEP-ProductInformation ExportTime="2018-04-16 05:24:26" ExportContext="Context1" ContextID="Cont
 <Products>
   <Product ID="235122" UserTypeID="SalesItem" ParentID="235115">
     <Name>Peanut Butter</Name>
     <ProductCrossReference ProductID="249236" QualifierID="en-US" Type="OIEP_POSTPROCESSOR"/>
     <ProductCrossReference ProductID="157618" QualifierID="UK English" Type="OIEP_POSTPROCESSOR"/
     <ProductCrossReference ProductID="157618" QualifierID="en-US" Type="OIEP_POSTPROCESSOR"/>
     <Values>
       <ValueGroup AttributeID="DescriptionWeb">
         <Value QualifierID="en-US">Nutritious, filling, and a great healthy eating option</Value>
         <Value QualifierID="German">Nahrhaft, sättigend und eine große gesunde Essenwahl</Value>
         <Value QualifierID="UK English">Nutritious, filling and healthy eating option</Value>
       </ValueGroup>
      </Values>
   </Product>
 </Products>
</STEP-ProductInformation>
```

Only the attribute values specific to particular dimension points—indicated by QualifierIDs—are exported, and only the attribute values that are different from the master context. Master values are taken from the 'en-US' and the English UK values are taken from 'UK English.'

Note: Standard export methods collapse duplicate values, and only the top value of the inheritance tree is exported.

XML Output Example - with Copy Context Dependent Values and References post-processor

The following shows an XML output of an OIEP cross-context export (exporting data for selected contexts) with the **Copy Context Dependent Values and References** post-processor enabled:

```
<?xml version="1.0" encoding="utf-8"?>
<STEP-ProductInformation ExportTime="2018-04-13 06:24:04" ExportContext="Context1" ContextID="Context1" WorkspaceID="M
 <Products>
  <Product ID="235122" UserTypeID="SalesItem" ParentID="235115">
    <Name>Peanut Butter</Name>
    <ProductCrossReference ProductID="249236" ContextID="Context3" QualifierID="en-US" Type="copyContextDepReference"/
    <ProductCrossReference ProductID="157618" ContextID="Context1" QualifierID="en-US" Type="copyContextDepReference"/
    <ProductCrossReference ProductID="157618" ContextID="Context7" QualifierID="UK English" Type="copyContextDepRefere
    <Values>
     <Value AttributeID="DescriptionWeb" ContextID="Context1" QualifierID="en-US">Nutritious, filling, and a great hea
     <Value AttributeID="DescriptionWeb" ContextID="Context7" QualifierID="UK English">Nutritious, filling and healthy
     <Value AttributeID="DescriptionWeb" ContextID="Context3" QualifierID="en-US">Nutritious, filling, and a great hea
     <Value AttributeID="DescriptionWeb" ContextID="Context5" QualifierID="German">Nahrhaft, sättigend und eine große
    </Values>
  </Product>
 </Products>
</STEP-ProductInformation>
```

The <ContextID> tag has been added to the export along with the <QualifierID> tag, and each value—including inherited values—is explicitly exported. This means that the 'English Canada' (Context3) values are included in the export, even though it inherits from the master context. If the value is overwritten in the English Canada context, as shown below, the <QualifierID> changes to the ID of the 'Canada English' language dimension point.

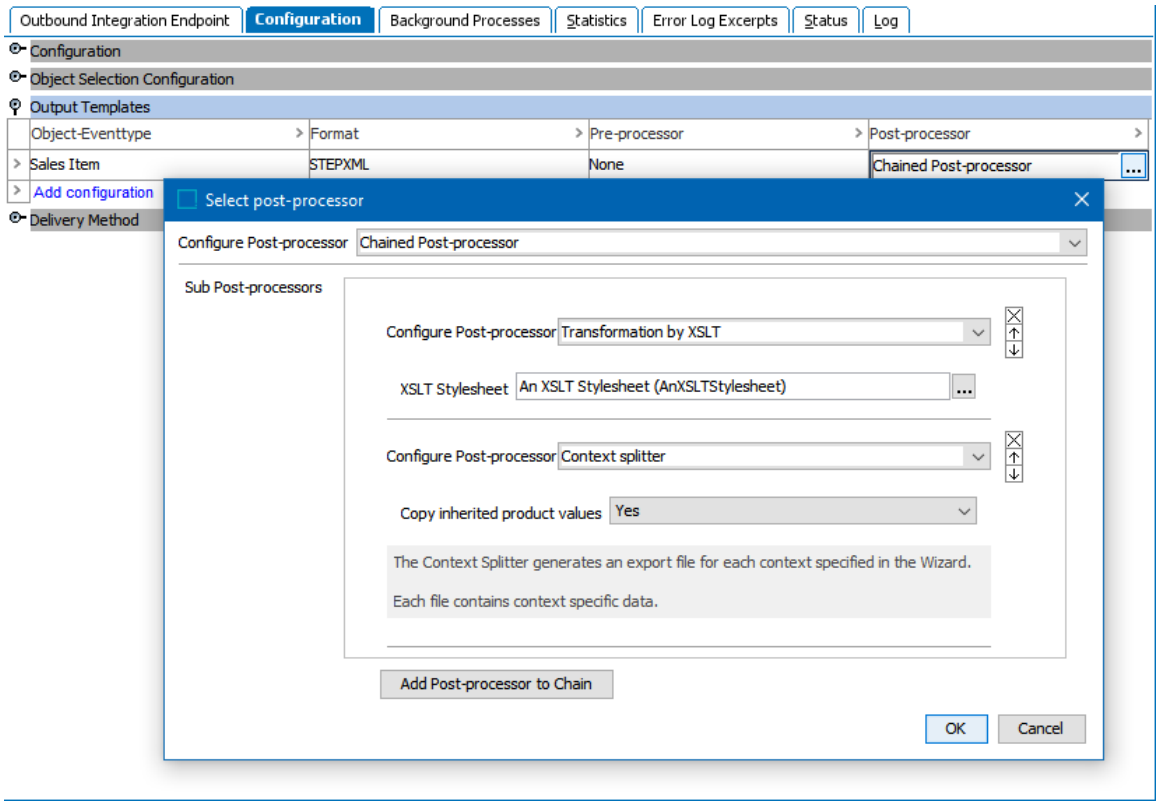
```
<Value AttributeID="DescriptionWeb" ContextID="Context3" QualifierID="Canada English">Filling and a great healthy eat
```

The default filename of the delivered XML is 'CopyContextDependent##exported_0.'However, the filename can be changed based on the selected delivery method, as defined in the **OIEP - Delivery Method Flipper** topic.

OIEP - Post-Processor - Chained Post-processor

The Chained Post-processor is a unique operation that allows users to stack multiple post-processors.

Configure the Chained Post-processor



1. On an OIEP, navigate to the 'Output Templates' section of the 'Configuration' tab.
2. Select the 'Post-processor' field, and then click the ellipsis button (...).
3. On the 'Select Post-processor' dialog, select the 'Chained Post-processor' option from the dropdown.

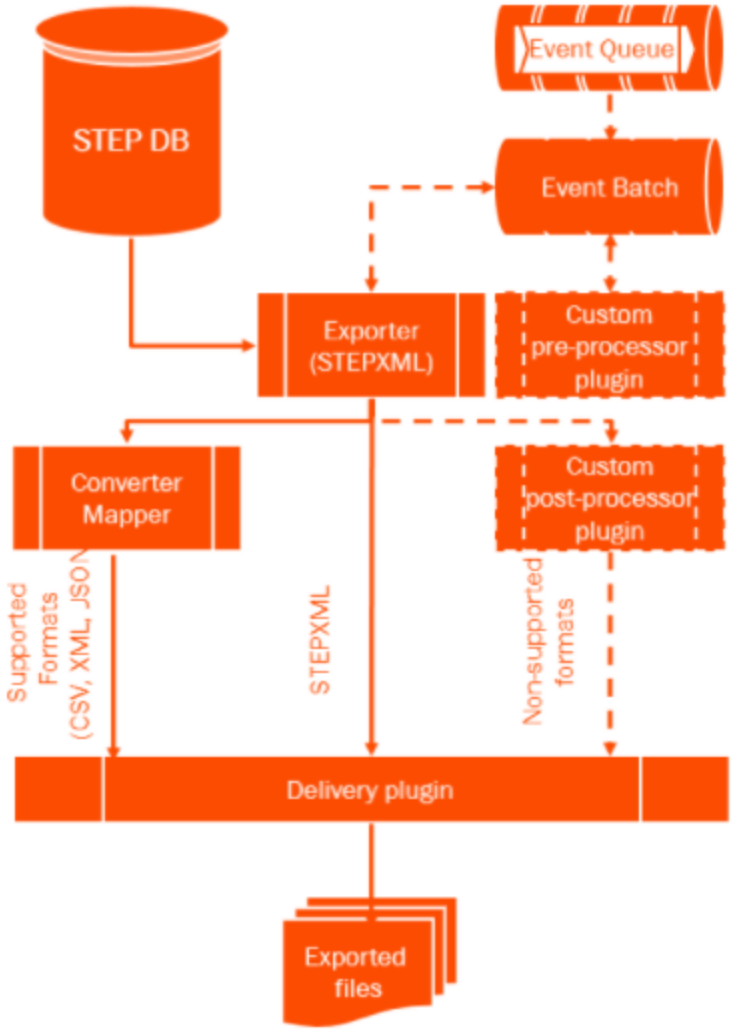
To configure the Chain Post-processor, users will then need to configure the additional post-processors independently. For more information, refer to the following post-processor topics:

- OIEP - Post-processor - Context Splitter
- OIEP - Post-processor - Copy Context Dependent Values and References
- OIEP - Post-processor - Transformation by XSLT

OIEP - Post-processor - Transformation by XSLT

The Transformation by XSLT post-processor allows an XSLT 2.0-compliant stylesheet to be used during export to transform outbound STEPXML files.

Note: To use this post-processor successfully, the outbound file or message size must be 50 MB or smaller.



Considerations

- The XSLT Post-processor requires STEP XML or Advanced STEPXML as output. To reduce the size of the output, configure Advanced STEPXML to limit the exported attributes.
- Generic XML output for XSLT is not supported. To resolve the error message 'Post-processor is not declared compatible with format converters, skipping format conversion', set the output as STEPXML or Advanced STEPXML.

Prerequisites

Before configuring the Transformation by XSLT post-processor, you must first:

1. Perform the **Initial Setup for XSLT Stylesheets**.
2. **Create an XSLT Stylesheet Object** - these steps must be performed for each separate stylesheet needed.

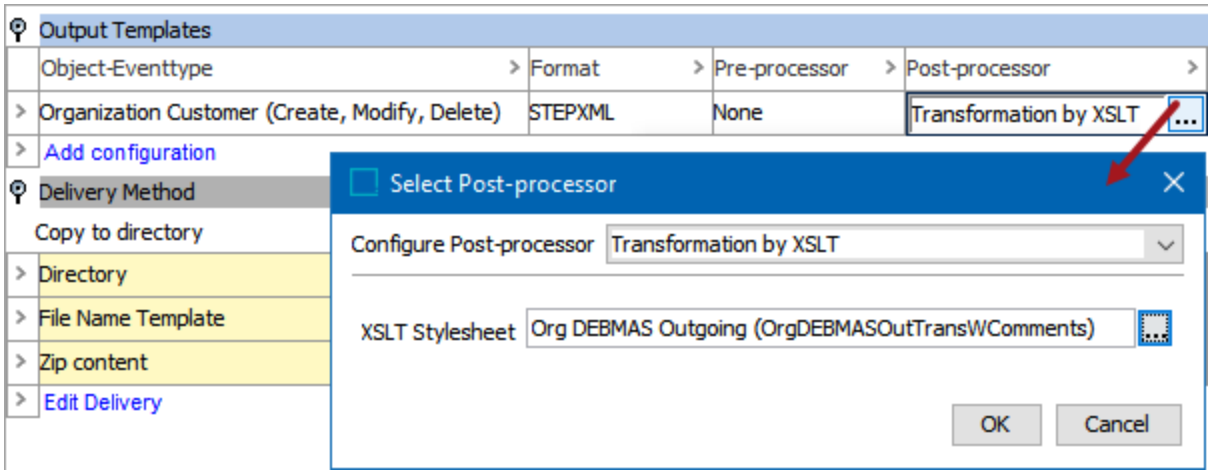
   For more information on these two steps, refer to the **IIEP - Configure Transformation by XSLT Pre-processor** section of the **Data Exchange** documentation.

Note: Saxon is an XSLT and XQuery processor. Information can be found via links on the **Open Source Components** page that can be found at the bottom of the list of topics in the left navigation panel of online help.

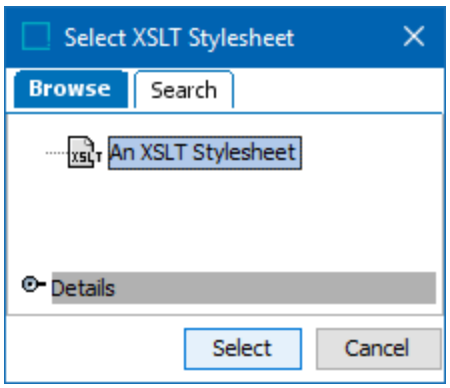
Configure the Transformation by XSLT Post-processor

Once the prerequisite tasks have been completed, you are ready to configure the post-processor in the OIEP configuration tab.

Note: The `<xsl:result-document>` tag is not supported. It is not possible to create multiple output documents from a single input document.



1. In the **Configure Post-processor** dropdown, select **Transformation by XSLT**.
2. In the **XSLT Stylesheet** parameter, click the ellipsis button (...) to display the Select XSLT Stylesheet dialog.
3. Select your stylesheet and click the **Select** button.

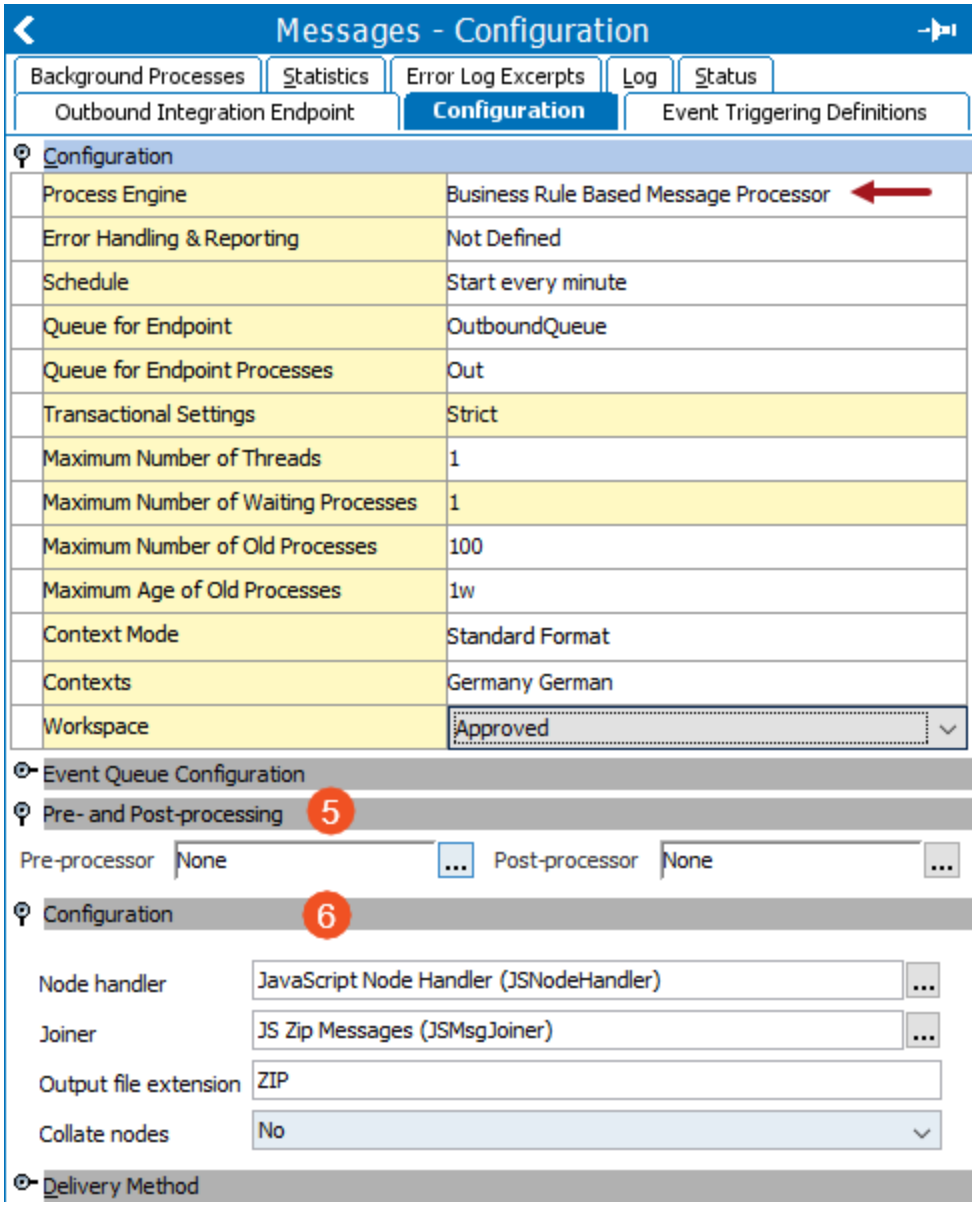


The exporter will use the selected XSLT stylesheet to transform the outbound file.

4. Click **OK** once the stylesheet has been selected.

OIEP - Event-Based - Pre- and Post- processing Flipper

The Pre and Post-processing flipper is available when the Business Rule Based Message processor is selected for an OIEP. This Configuration flipper includes the same parameters for both Event-Based and Select Objects endpoints. Each parameter is described below.



These are the same parameters that are available in the Output Templates Flipper when using the STEP Exporter process engine.

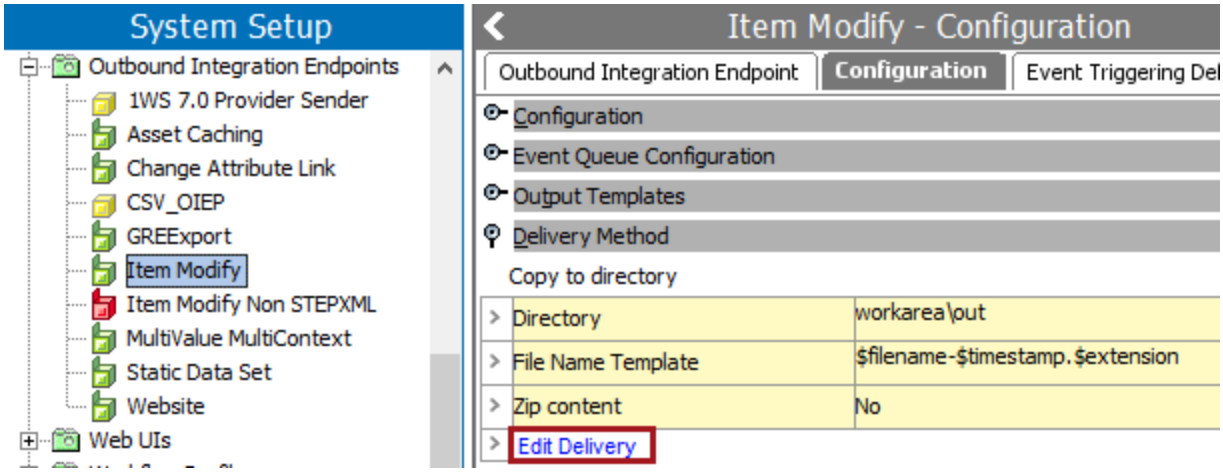
For details, refer to the **Configure the Pre-processor and Post-processor** section of the following topics:

- **OIEP - Select Objects - Output Templates Flipper**
- **OIEP - Event-Based - Output Templates Flipper**

OIEP - Delivery Method Flipper

The Delivery Method flipper in an OIEP includes the same parameters for both Event-Based and Select Objects endpoints. The delivery method controls how the output is delivered to an external system.

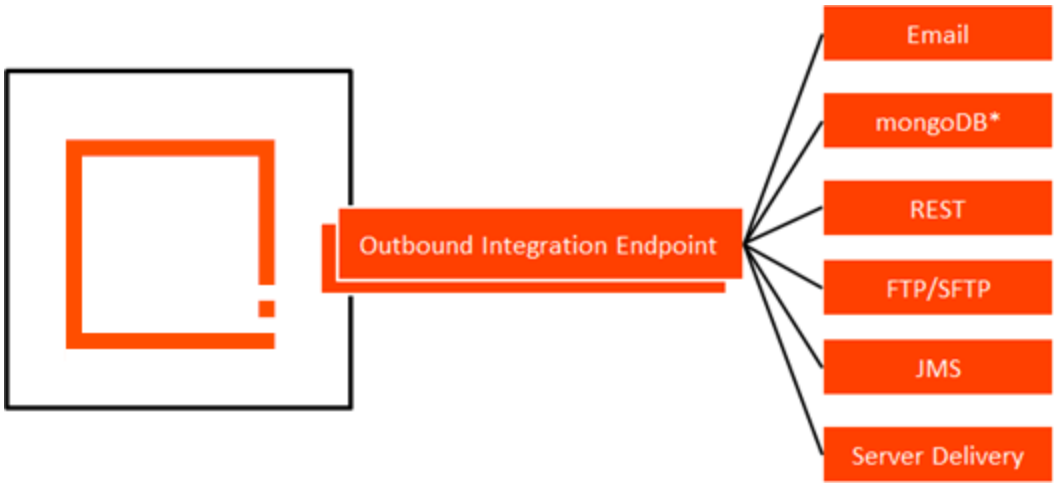
Open the Delivery Method flipper to display the selected delivery method. As shown below, the parameters vary based on the selected delivery method.



Delivery Methods

Delivery methods are used by both outbound data tools (Export Manager and OIEP) but the available options vary. For information on the delivery methods available in Export Manager, refer to **Export Manager - Select Delivery Method**.

An OIEP takes the data determined by its configuration and delivers it using the selected delivery method to the downstream system or on a directory on the application server.



The following delivery method options are available (although some are based on a license) when exporting using an OIEP.

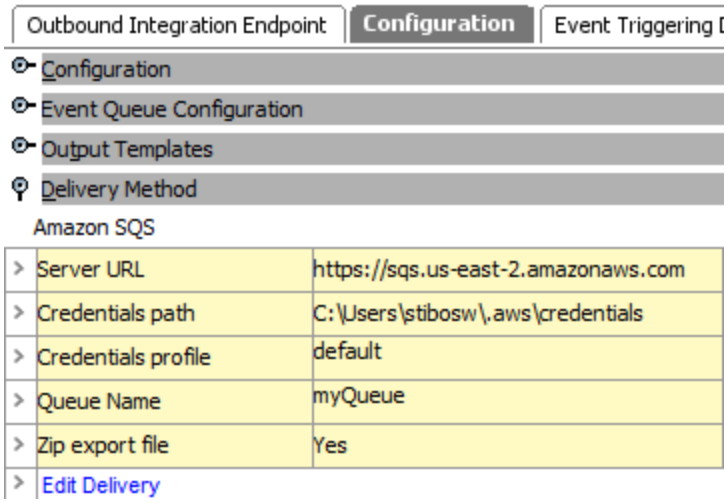
Method	Description
Amazon SQS	Delivers messages to the Amazon SQS (Amazon Simple Queue Service). Refer to **Amazon SQS Delivery Method** documentation.
Azure Blob Storage Delivery	Delivers files to Azure Blob Storage. Refer to **Azure Blob Storage Delivery Method** documentation.
Change Package Git Delivery	Delivers change packages to a Git repository. Refer to the **Integration Endpoint Options for VCS Integration** topic in the **Configuration Management** documentation.
Cloud Blob Storage Delivery	Delivers exported files to cloud storage. Amazon S3, Google Cloud Storage, and Microsoft Azure are supported. Refer to **Cloud Blob Storage Delivery Method** documentation.
Copy to Directory	Default selection for new OIEPs delivers files to a directory on the application server and allows the delivery to be zipped. Refer to **Copy to Directory Delivery Method** documentation.
Deploy	Delivers files to a directory on the application server. Does not allow the delivery to be zipped. Refer to **Deploy Delivery Method** documentation.
Dynamic JMS	The standard Dynamic JMS delivery options (Apache Active MQ, IBM MQ, and Oracle AQ) can be used to deliver messages to a Dynamic JMS Receiver without further customizations. Unlike the JMS delivery method mentioned above, this delivery method allows customers to supply the vendor-specific JMS libraries and JNDI configuration. Refer to the **Dynamic JMS Delivery Method** documentation.
Email	Delivers files as an email attachment and allows a zipped delivery. Refer to **Email Delivery Method** documentation.
FTP	Delivers files using file transfer protocol (FTP). Refer to **FTP Delivery Method** documentation.
Git Delivery	Delivers files produced by the OIEP processing engine or a configured post-processor to a branch in a remote Git repository. Refer to the **Integration Endpoint Options for VCS Integration** topic in the **Configuration Management** documentation.
GDSN	The GDSN solution must be implemented fully before the GDSN Data Pool method is available

Method	Description
Datapool	and functional. Refer to **GDSN Receiver Solution Enablement** documentation.
GDSN Receiver Datapool	The GDSN solution must be implemented fully before the GDSN Receiver Data Pool method is available and functional. Refer to **GDSN Receiver Solution Enablement** documentation.
IBM MQ SSL	IBM MQ SSL, uses MQ series Secure Sockets Layer, and enables data exchange across IBM and non-IBM platforms. Refer to **IBM MQ SSL Delivery Method** documentation.
JDBC	The JDBC delivery option enables delivery of data to RDBMS-type databases like Oracle, MySQL, MS SQL Server, PostgreSQL, etc. Refer to **JDBC Delivery Method** documentation.
JMS	The standard JMS delivery options (Apache Active MQ, IBM MQ, and Oracle AQ) can be used to deliver messages to a JMS Receiver without further customizations. Refer to **JMS Delivery Method** documentation.
Kafka Delivery	Apache Kafka is an open-source distributed event-streaming data platform. The Kafka delivery enables the STEP platform integrated with Apache Kafka to take advantage of built-in options for outbound processing to a Kafka queue. Refer to the **Kafka Delivery Method** documentation.
Mongo	Receives data from a STEP event queue and loads it into a MongoDB database. The MongoDB is often used for website back-end, reporting, and high performance feeds to other back-end systems. Refer to **Mongo Delivery Method** documentation.
Oracle AQ	Oracle Advanced Queuing (Oracle AQ) enables messages to be exchanged between two systems. Refer to **Oracle AQ Delivery Method** documentation.
Product Data Exchange 2	Product Data Exchange sends data to the Product Data Exchange (PDX) platform via the default PDX Outbound Integration Endpoint and the API. Refer to **Product Data Exchange 2 Delivery Method** documentation.
REST	The REST Delivery Method delivers a call-back URL to the REST service and does not include actual STEP data. Refer to **REST Delivery Method** documentation.
REST Direct	The REST Direct delivery method differs from the standard REST delivery method in that the data is delivered directly to the REST service and no call-back URL is required. Refer to **REST Direct Delivery Method** documentation.

Method	Description
SFTP	Delivers a file using the Secure File Transfer Protocol (SFTP). Refer to **SFTP Delivery Method** documentation.
Wiki	Delivers metadata to the metadata wiki platform. Refer to **Wiki Delivery Method** documentation.

Amazon SQS Delivery Method

The Amazon SQS Delivery Method delivers messages to the Amazon Simple Queue Service (Amazon SQS). This delivery option is only available in OIEPs. When multiple XML files are available to be delivered at the same time, they are sent individually (not concatenated into a single file).



To access the Amazon SQS Delivery method, a Delivery.AmazonSQS component must be activated on your system in addition to the normal update procedures. Contact your Stibo Systems representative for more information.

The Amazon SQS queues require a file size of less than 256KB. Common setup is to use file compression (zip export file) to reduce the message size to meet this limitation. Amazon S3 storage account is required to handle files 256KB or larger. For more information, search Amazon S3 online.

Prerequisites

Changes to the properties file, outlined below, are implemented when the server is restarted.

1. For the 'Credentials profile' parameter, create a credentials file following the instructions at http://docs.aws.amazon.com/cli/latest/userguide/cli-config-files.html. Place the credentials file (named 'credentials', without any file extension) in the required path on the application server. This file is used by both the OIEP Amazon SQS Delivery Method and the IIEP Amazon SQS Receiver.
2. Prior to configuration, clicking the **Server URL** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **SQSServerUrl** property. The following is an example of a property entry for a single server:

   ```
   SQSServerUrl=1=https://sqs.us-east-2.amazonaws.com
   ```

3. Prior to configuration, clicking the **Credentials Path** dropdown parameter displays the required property name. Provide a selection, including the name of the credentials file, for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **SQSCredentialsPath**

property. If necessary, use a comma and increment the number to separate multiple paths as shown in the example below.

```
SQSCredentialsPath=1=C:\\Users\\stibosw\\.aws\\credentials,2=C:\\Users\\stibosw\\.aws\\credentials123
```

4. Prior to configuration, clicking the **S3 Bucket Name** dropdown parameter displays the required property name. If Amazon S3 is being used, provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **SQSDeliveryS3BucketName** property. The following is an example of a property entry for a two buckets where a comma separates multiple buckets:

```
SQSDeliveryS3BucketName=1=AmazonBucketName1,2=AmazonBucketName2
```

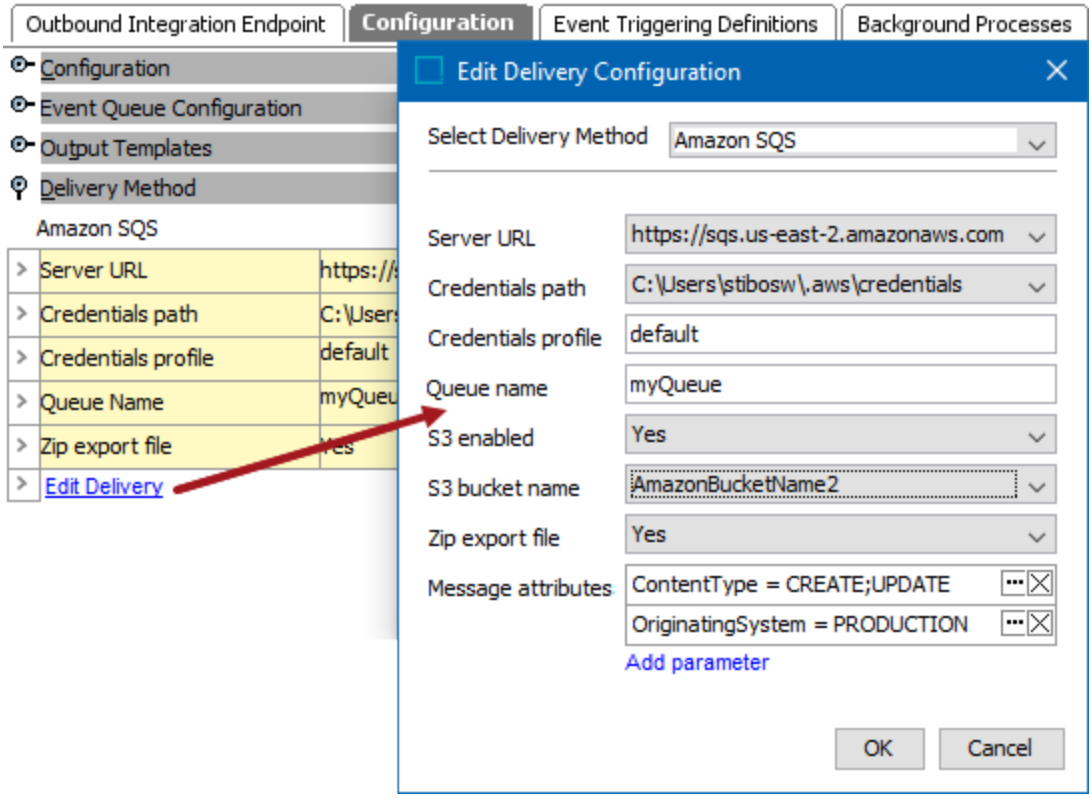
5. For the 'Queue name' parameter, if necessary, create a new queue on the SQS server. Use the steps defined in the **Amazon SQS Queue Configuration** section below.
6. If necessary, establish the Amazon S3 storage account and create the required buckets. For details, refer to Amazon S3 on the web.
7. If message attributes are required, create up to seven (7) string Message Attributes on the Amazon website. The message attribute names in Amazon correspond to the Key parameters created in the delivery method configuration below.

Contact your Stibo Systems representative if you need assistance with setup.

Configuration

After completing the prerequisite steps, edit the delivery method of the OIEP. Use the following steps to configure the OIEP to use the Amazon SQS delivery option.

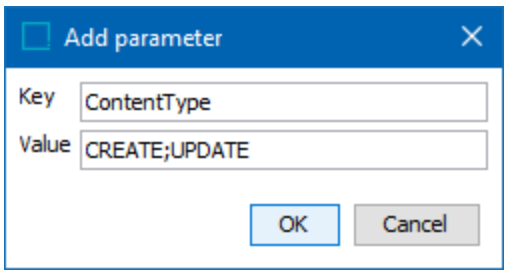
For information on a parameter, hover over the parameter field to display help text.



1. For **Select Delivery Method**, choose **Amazon SQS**.
2. In **Server URL**, select the URL that points to the SQS server where the data will be delivered.
3. In **Credentials path**, select the path to the desired SQS credentials file.
4. In **Credentials profile**, enter the profile name included in the SQS credentials profile.
5. In **Queue Name**, enter the Amazon queue created for this delivery. This queue is also required when Amazon S3 is being used.
6. For **S3 Enabled**, select 'Yes' if you have an Amazon S3 storage account. Setting this option to 'No' results in an error for messages that exceed the 256KB size limit.

   Note: When 'S3 Enabled' = Yes, the Amazon framework sends messages that are smaller than 256KB continue to be processed without S3 using Amazon SQS.

7. In **S3 Bucket Name**, select the Amazon bucket name that should be used for this delivery. This name must already be established on the Amazon account.
8. In **Zip export file**, specify whether to zip the contents before delivery.
9. **Message Attributes** provides user-defined data that is visible within Amazon SQS (refer to the **Amazon SQS View Message Attribute** section below) and is specific to the OIEP. Click the **Add parameter** link to create a message attribute, then enter a text value for the Key and the Value parameter. In the image above, the Key = ContentType, and the Value = CREATE;UPDATE, which could be used to indicate that the OIEP is processing both create and update events.

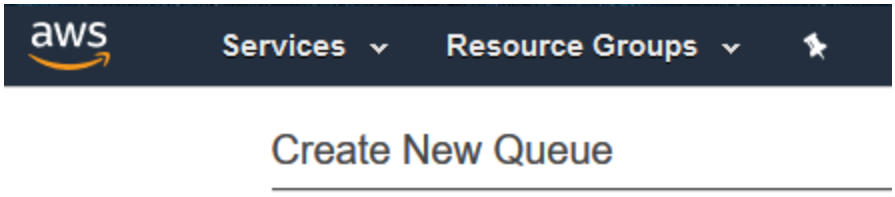


10. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

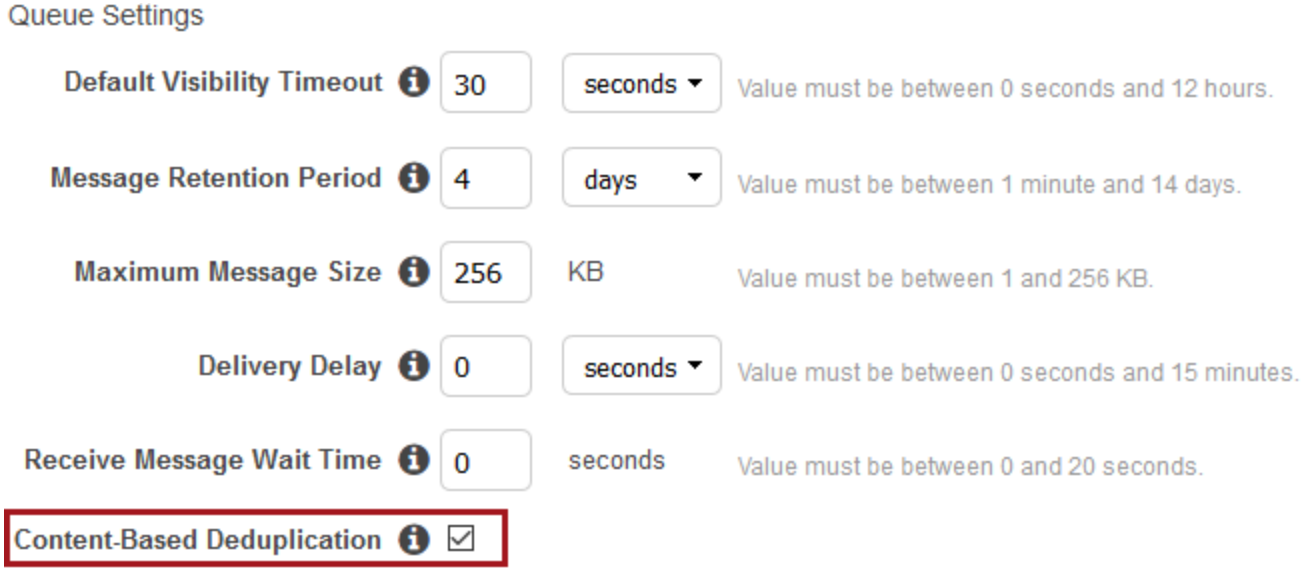
Amazon SQS Queue Configuration

When required, use the following steps to create a new Amazon SQS queue.

1. Log in to the Amazon server based on your region.
2. Search for **Simple Queue Service** and select it to display the queue list page.
3. Click the **Create New Queue** button to display the Create New Queue page.



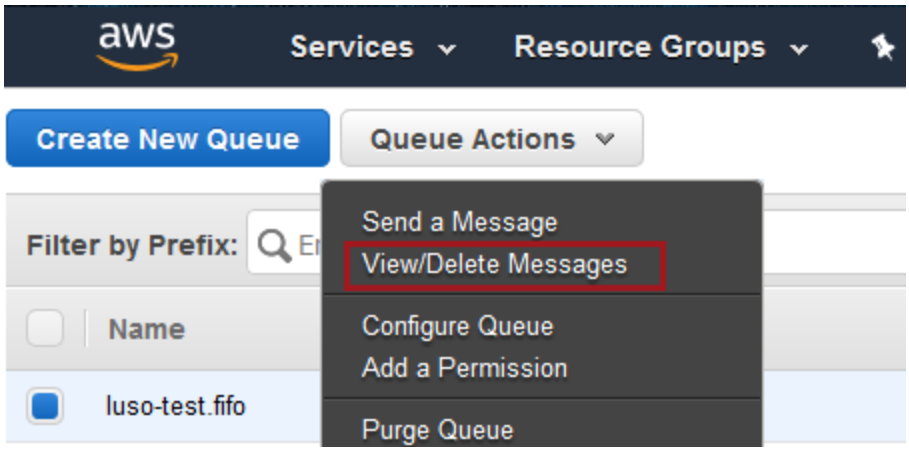
4. For **Queue Name**, type a queue name including the '**.fifo**' extension.
5. For the **type of queue** section, select **FIFO Queue**.
6. Click the **Configure Queue** button at the bottom of the page to display the default Queue Attributes parameters.
7. Check the **Content-Based Deduplication** parameter box to enable the queue.



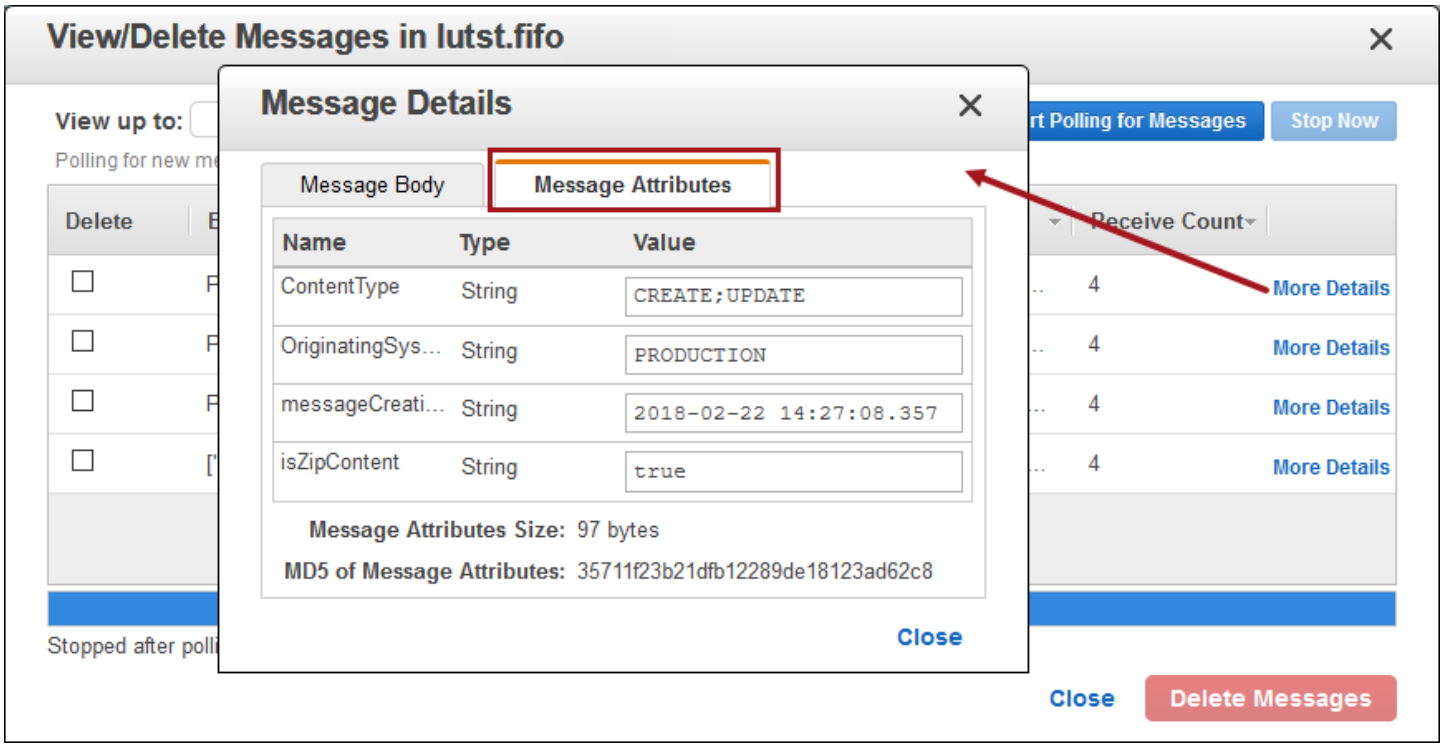
8. Click the **Create Queue** button. The new queue is displayed in the list.

Amazon SQS View Message Attributes

1. Log in to Amazon SQS and display your queue(s).
2. Right-click the desired queue to display the menu (or click the Queue Actions dropdown button) and click the **View/Delete Messages** option.



3. Click the **More Details** link to display the Message Details dialog.



4. On the Message Details dialog, click the **Message Attributes** tab to display the standard and user-defined message attributes.

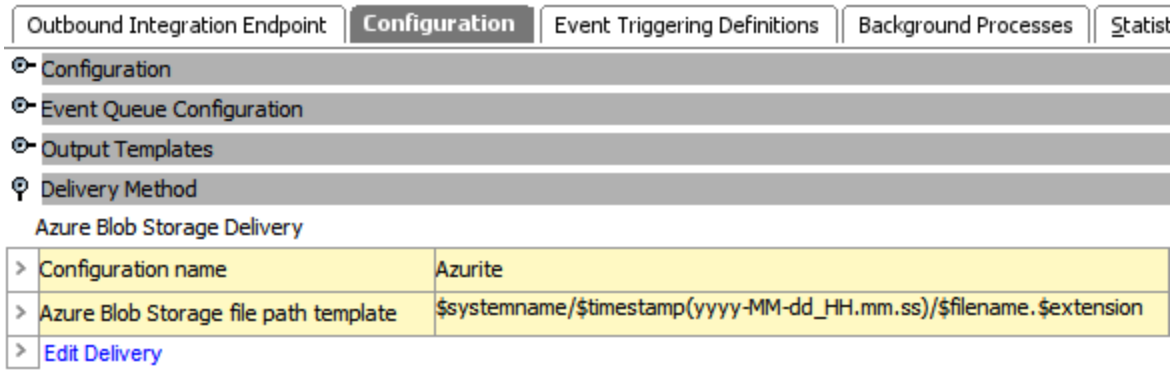
Note: In addition to the user-defined message attributes, two standard message attributes are included: **messageCreationTime** displays the delivery datetime, and **isZipContent** indicates the status of the 'Zip export file' parameter.

Azure Blob Storage Delivery Method

The Azure Blob Storage Delivery plugin for outbound integration endpoints makes it possible to deliver files to Azure Blob Storage. This delivery method is available in OIEPs and Export Manager.

Note: To deliver OIEP files to Azure blob storage, the Cloud Blob Storage Delivery Method can also be used. The differentiation between the setup and functionalities is that the Cloud Blob Storage Delivery plugin is part of the STEP baseline, can also be used for Amazon S3, and the Export Manager and OIEP configurations use the gateway integration endpoints versus having separate delivery method integration properties.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



To use this delivery method in Export Manager, refer to the **Azure Blob Storage Delivery Method** topic.

Prerequisites

The Azure Blob Storage Delivery plugin is part of the 'cloudstorage-azure' component that must be installed in addition to the STEP baseline. No additional licenses are required.

Prior to configuration, click the **Configuration name** dropdown parameter to display the required configuration to be used. You will provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the following configuration properties:

```
AzureBlobStorageDeliveryPlugin.ConfigurationNames
AzureBlobStorageDeliveryPlugin.ConnectionString.[Configuration Name]
AzureBlobStorageDeliveryPlugin.ContainerName.[Configuration Name]
```

The value for 'AzureBlobStorageDeliveryPlugin.ConfigurationNames' is a comma-separated list of user-defined names for the desired Azure Blob Storage configurations. For each name, corresponding 'AzureBlobStorageDeliveryPlugin.ConnectionString.[Configuration Name]' and 'AzureBlobStorageDeliveryPlugin.ContainerName.[Configuration Name]' properties must be set with the values being the connection string for the configuration and the desired blob container name, respectively.

An example configuration using the storage account access key (*AccountName* / *AccountKey*) method is below:

```
AzureBlobStorageDeliveryPlugin.ConfigurationNames=Azurite

AzureBlobStorageDeliveryPlugin.ConnectionString.Azurite=DefaultEndpointsProtocol=http;AccountName=devstoreaccount1;AccountKey=Eby8vdM02xNOcqFlqUwJPLlmEtlCDXJ1OUzFT50uSRZ6IFsuFq2UVErCz4I6tq/K1SZFPTOtr/KBHBeksoGMGw==;BlobEndpoint=http://127.0.0.1:10000/devstoreaccount1;
AzureBlobStorageDeliveryPlugin.ContainerName.Azurite=productData
```

It is also possible to use a Shared Access Signature (SAS) credential for the ConnectionString.

The SAS token must be created directly on the blob storage account itself (and not the corresponding container); and as a minimum, it must have **Service**, **Container**, and **Object** specified as its 'Allowed resource types' as well as **Read**, **Write**, and **List** for its 'Allowed permissions' to grant the proper access rights to STEP.

These resource types and permissions are required to allow STEP to perform all the needed operations to deliver the content (blobs) to the specified Azure Blob Storage account's container.

Important: If the SAS token has insufficient privileges, the delivery will result in an error message similar to this one:

If you are using a SAS token, and the server returned an error message that says 'Signature did not match', you can compare the string to sign with the one generated by the SDK. To log the string to sign, pass in the context key value pair 'Azure-Storage-Log-String-To-Sign': true to the appropriate generateSas method call. Remember to disable 'Azure-Storage-Log-String-To-Sign' before going to production as this string can potentially contain PII.

Status code 403, "<?xml version="1.0" encoding="utf-8"?><Error><Code>AuthorizationResourceTypeMismatch</Code><Message>This request is not authorized to perform this operation using this resource type. RequestId:836910b1-801e-001a-4da2-900fc9000000 Time:2022-07-05T19:11:07.4796215Z</Message></Error>"

An example using a SAS configuration is below:

```
AzureBlobStorageDeliveryPlugin.ConfigurationNames=bestProducts
AzureBlobStorageDeliveryPlugin.ConnectionString.bestProducts=
BlobEndpoint=https://pimtest.blob.core.windows.net/;SharedAccessSignature=sv=2021-06-08&ss=b&srt=sco&sp=rwlx&se=2023-01-01T05:00:00Z&st=2022-07-05T18:58:41Z&spr=https&sig=N5X7J9tCMscbOTYioR4sb30H7B%2B0j8dk74MSCQ6Gxsw%3D
AzureBlobStorageDeliveryPlugin.ContainerName.bestProducts=productData
```

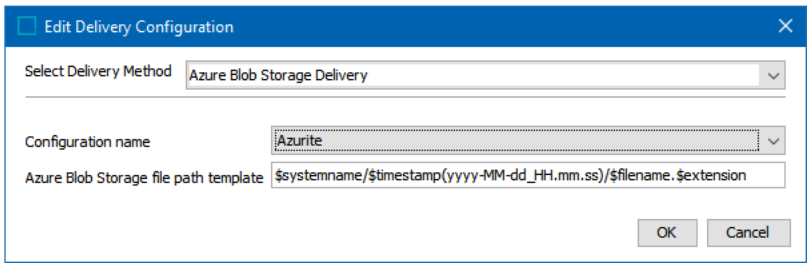
Once the property configuration is in place and the system has been restarted, the configuration name(s) will appear as selectable options in the STEP Workbench.

Configuration

1. On the **Configuration** tab, in the **Delivery Method** flipper, click **Edit Delivery**.
2. In **Select Delivery Method**, choose **Azure Blob Storage Delivery**.

3. The **Configuration name** comes from the properties you set above. If you configured the properties to use multiple configuration names within a comma-separated list, then you will choose from the dropdown.

   Example shown below:



4. Configure the Azure Blob Storage file path template.

   In addition to selecting the appropriate connection string and container, it is possible to make use of Azure Blob Storage virtual directories. Four variables are available:

   - $systemname – The STEP system name. Useful when several STEP systems are delivering to the same blob container.
   - $timestamp([YMDHMS]) – Delivery timestamp. Desired format to be supplied in Java SimpleDateFormat compatible format.
   - $filename – Name of the file produced by the outbound integration endpoint.
   - $extension – The extension of the file produced by the outbound integration endpoint.

If the blob containers and virtual directories do not already exist, the plugin will create them.

Important: Existing files with the same virtual directory path and file name will be overwritten.

5. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

Change Package Git Delivery Method

The Change Package Git Delivery method allows integration with popular repositories, supporting the HTTPS (token-based) or the SSH (file-based) access methods for GitHub, GitLab, and Bitbucket.

The Change Package Git Delivery method delivers files produced by the OIEP processing engine using an integrated STEPXML Splitter to deliver multiple files to a branch in a remote Git repository. Refer to https://git-scm.com for more information about Git.

The change package is represented below the specified branch within a configurable directory structure. At the end of the directory structure, change packages grouping files by the Primary and Secondary flippers, then by type of object with XML or JSON files named by object type and the ID of the object.

Important: For on-premises systems, this feature requires the configuration-management component.

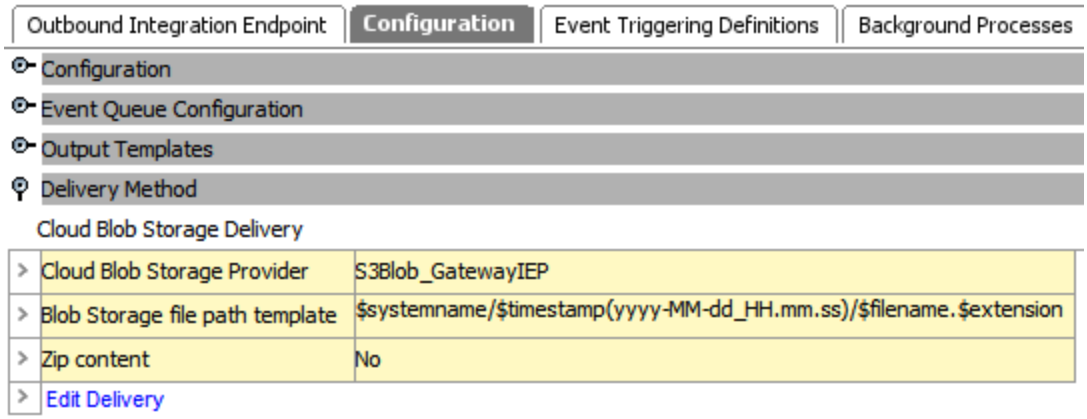
For details, refer to the **Integration Endpoint Options for VCS Integration** topic in the **Configuration Management** documentation.

Cloud Blob Storage Delivery Method

The Cloud Blob Storage Delivery method delivers exported files to cloud storage. Amazon S3, Google Cloud Storage, and Microsoft Azure are supported. This delivery method is available in Export Manager and OIEPs.

Note: The Cloud Blob Storage Delivery Method is similar to the legacy Azure Blob Storage Delivery Method. The difference between the setup and functionalities is that the Cloud Blob Storage Delivery plugin is part of the STEP baseline, can also be used for Amazon S3 and Google Cloud Storage, and the Export Manager and OIEP configurations use the gateway integration endpoints versus having separate delivery method integration properties.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



To use this delivery method in Export Manager, refer to the **Cloud Blob Storage Delivery Method** topic.

Prerequisites

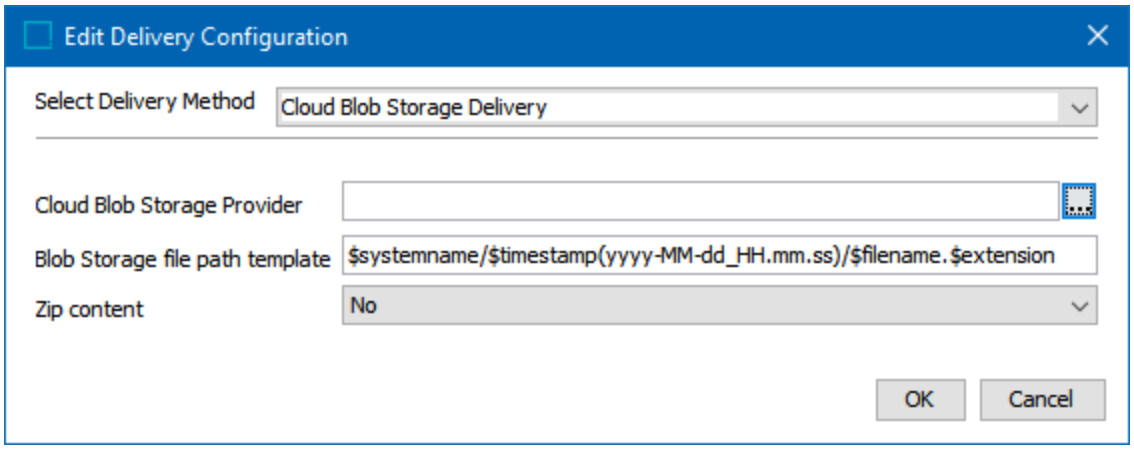
A blob storage gateway integration endpoint must be configured before moving ahead with the Export Manager or OIEP setup.

- For Amazon S3, directions for setting up this endpoint are in the **Configuring a Gateway Integration Endpoint - Amazon S3 Blob Storage** topic in the **Data Exchange** documentation.
- For Google Cloud Storage (GCS), directions for setting up this endpoint can be found in the **Configuring a Gateway Integration Endpoint - Google Cloud Storage** topic in the **Data Exchange** documentation.
- For Microsoft Azure (ABS), directions for setting up this endpoint can be found in the **Configuring a Gateway Integration Endpoint - Microsoft Azure Blob Storage** topic in the **Data Exchange** documentation.

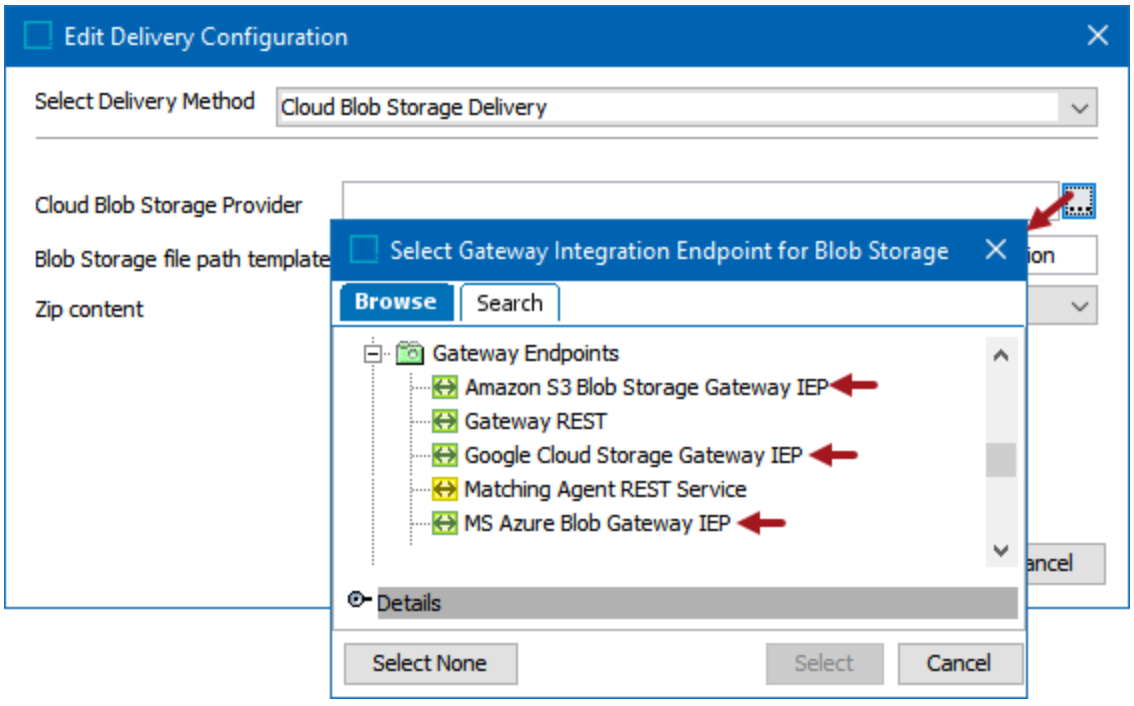
Also, buckets (S3 and GCS) and containers (ABS) must be created. If they do not exist in advance, you will get an exception error during the delivery process.

Configuring the OIEP Delivery

For information on a parameter, hover over the parameter label to display help text.

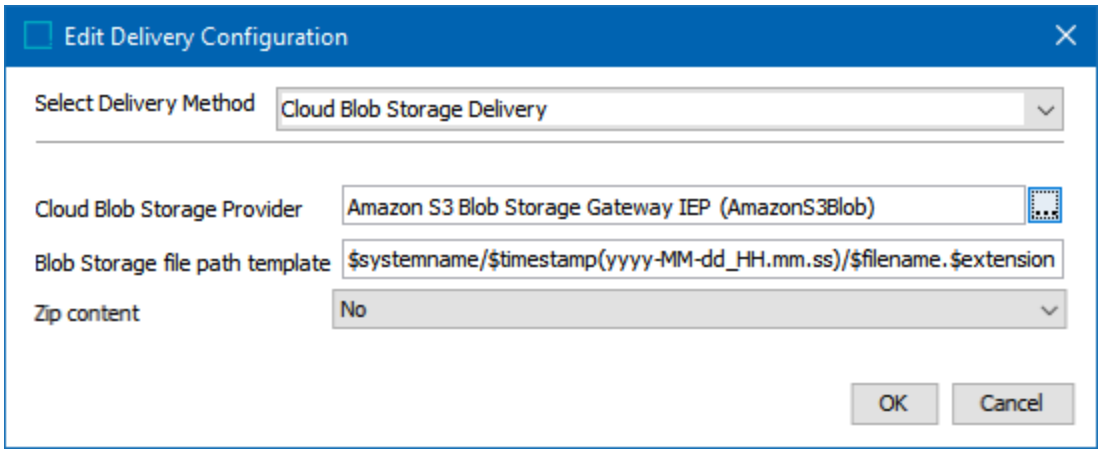


1. From the **Select Delivery Method** parameter dropdown, choose **Cloud Blob Storage Delivery**.
2. Click the ellipsis button (...) to the right of the **Cloud Blob Storage Provider** parameter, select a gateway endpoint configured for blob storage, and click the **Select** button. The name and ID of the selection display in the Edit Delivery Configuration dialog.



3. Configure the **Blob Storage file path template** using the available variables:

   - $systemname – The STEP system name. Useful when several STEP systems are delivering to the same blob container.
   - $timestamp([YMDHMS Format]) – Delivery timestamp. Desired format to be supplied in Java SimpleDateFormat compatible format.
   - $filename – Name of the file produced.
   - $extension – The extension of the file produced.



   In addition to selecting the appropriate connection string and container, you can make use of Azure Blob Storage virtual directories. If the virtual directories do not already exist, the plugin creates them. As mentioned in the **Prerequisites** section, buckets (S3 and GCS) and containers (ABS) must exist in advance, or you will get an exception error during the delivery.

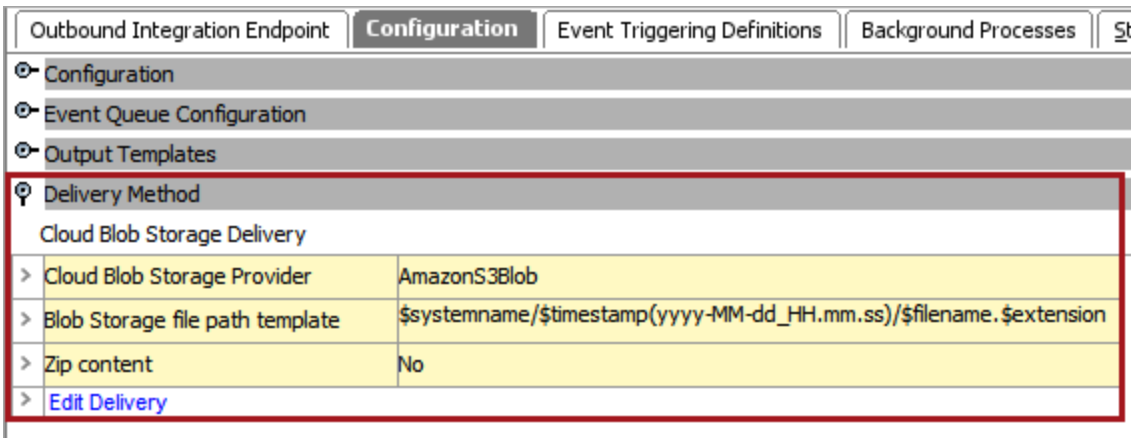
   Important: Existing files with the same virtual directory path and file name are overwritten.

4. In **Zip content**, select 'yes' or 'no' from the dropdown to specify if the output file should be delivered in a .ZIP (compressed) file format.

   - **Yes** uses 'export-' before the timestamp variable, and then the extension ZIP. For example, a zipped STEPXML output with the default File Name Template exported on 15 NOV 2016 results in an output .ZIP file named 'export-1479230247017.zip.' The contents of the ZIP file follow the $filename variable applied in the 'File name template' parameter.
   - **No** uses the file type for the file name along with the appropriate extension for the selected data format.

5. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

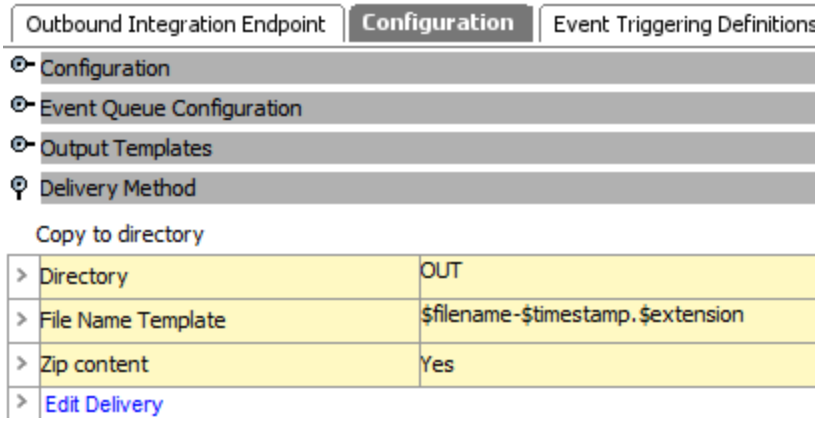
   The configuration is displayed in the OIEP editor.


Outbound Integration Endpoint
Configuration
Event Triggering Definitions
Background Processes
Configuration
Event Queue Configuration
Output Templates
Delivery Method
Cloud Blob Storage Delivery
Cloud Blob Storage Provider
AmazonS3Blob
Blob Storage file path template
$systemname/$timestamp(yyyy-MM-dd_HH.mm.ss)/$filename.$extension
Zip content
No
Edit Delivery

Copy to Directory Delivery Method

For OIEPs, there is no single delivery option for server-side delivery, but the 'Deploy' and 'Copy to directory' options provide the same function in that they both deliver files to a directory on the application server. Common setup is to use 'Copy to directory' since it is more flexible because it allows the delivery to be zipped. Neither of these options is available in Export Manager.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



For more information, refer to the **Deploy Delivery Method** documentation.

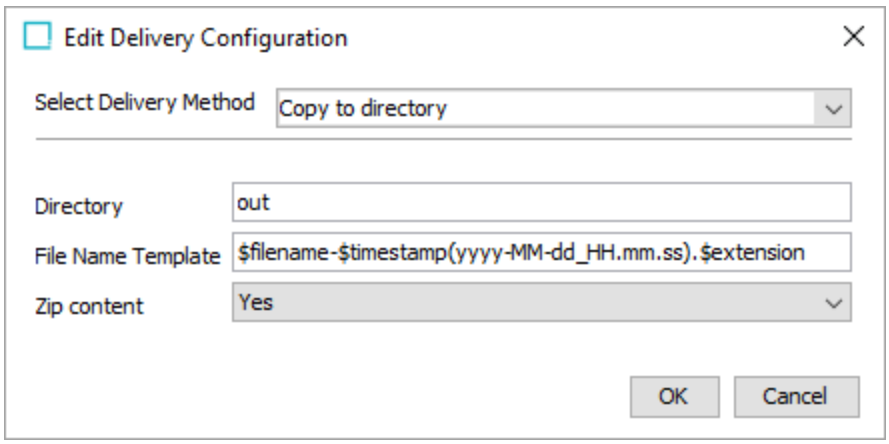
Prerequisites

The default delivery path is **opt/stibo/step** on the application server. This can be modified via the sharedconfig.properties file on the STEP application server using the case-sensitive **DirectoryDeliveryPlugin.RootDir** property. Changes to the properties file are implemented when the server is restarted.

Configuration

For information on a parameter, hover over the parameter field to display help text.

1. On the **Configuration** tab, navigate to the **Delivery Method** flipper, then click **Edit Delivery**.
2. Click the **Select Delivery Method** dropdown and choose **Copy to Directory**.



3. In **Directory**, specify the directory on the application server that will house the exported files. Do not start this text with a slash (/). Text entered will be appended to the path as defined in the Prerequisites section.

   In the image above, assuming the default path is being used, the file will be delivered to 'opt/stibo/step/out.'

4. In **File Name Template**, to create the file name used for the output, enter template variables (identified with a $), static text, or use a combination of both. The default template is $filename-$timestamp(yyyy-MM-dd_HH.mm.ss).$extension. Depending on the settings of the configuration, the default template outputs a file name similar to 'GenericXML-2020-07-30_14.07.44.xml.' Each variable is described below:

   Note: The 'Zip context' parameter also has an impact on the file name.

   - **$filename** For event-based OIEPs, this variable is replaced with text to indicate the format of the delivered file as specified in the Output Templates flipper, except for STEPXML when the first and last Event IDs are used. For example, the output file name could be 'csv-2020-07-30_14.09.40.csv' or '1804038-1804038.xml' to indicate that STEPXML was used for a single event. When exporting multiple contexts, '##' is used as a separator within the filename to distinguish the context name for each output.
   - **$timestamp**: This variable is replaced with the time in year-day-month-hour-minutes-seconds-milliseconds, excluding the dashes. For example, 20161511094709625 is used for an export on 15 NOV 2016 at 9:47:09:625 AM.

     Adding a definition for the arrangement of the date / time elements provides a way to generate a more readable date. Static characters can be included to separate the timestamp elements and individual elements can be ordered as required, as shown below.

     Note: To revert the **$timestamp** variable so that it uses the legacy definition (milliseconds between January 1, 1970, and the time when the file is uploaded), contact Stibo Systems Support for more information.

     $timestamp(ddMMyyyy): The timestamp is modified to display day-month-year (excluding dashes) and is output as '15112016' for 15 NOV 2016.

$timestamp(yyyy-MM-dd_HH.mm.ss): The timestamp is modified to display year-month-day_ hour.minutes.seconds (including the dashes, underscore, and periods) and is output as '2016-11-15_ 09.32.43' for 2016 NOV 15 at 9:32.43 AM.

- **$extension** This variable is replaced with the extension of the selected output file format.

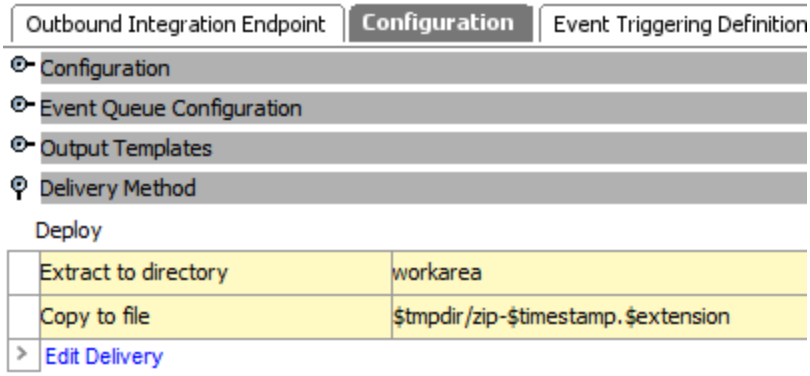
Note: The File Name Template does not support the conversion of file formats and can only be used to deliver files in the format specified on the Configuration tab of the outbound integration endpoint.

5. For **Zip content**, select an option from the dropdown to specify if the output file should be delivered in a .ZIP (compressed) file format.
   - **Yes** uses 'result_0' before the timestamp variable, and then the extension ZIP. For example, a zipped STEPXML output with the default File Name Template exported on 7 JUL 2020 at 2:07 p.m. results in an output .ZIP file named 'result_0-2020-07-30_14.07.44.zip.' The contents of the ZIP file follow the $filename variable applied in the 'File name template' parameter.
   - **No** uses the 'File Name Template' parameter for the file name along with the appropriate extension for the selected data format.
6. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

Deploy Delivery Method

For OIEPs, there is no single delivery option for server side delivery, but the 'Deploy' and 'Copy to directory' options provide the same functionality in that they both deliver files to a directory on the application server. Common setup is to use 'Copy to directory' since it is more flexible because it allows the delivery to be zipped. Neither of these options is available in Export Manager.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



For more information, refer to the **Copy to Directory Delivery Method** documentation.

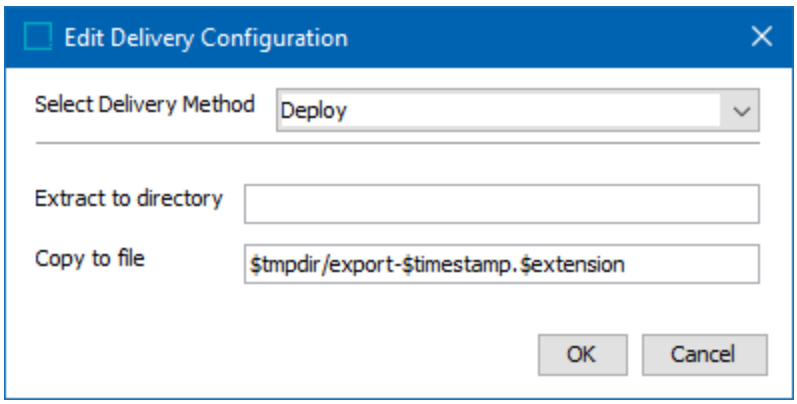
Prerequisites

The default delivery path is **opt/stibo/step** on the application server. This can be modified via the sharedconfig.properties file on the STEP application server using the case-sensitive **DirectoryDeliveryPlugin.RootDir** property. Changes to the properties file are implemented when the server is restarted.

Configuration

For information on a parameter, hover over the parameter field to display help text.

1. On the **Configuration** tab, navigate to the **Delivery Method** flipper, then click **Edit Delivery**.
2. Click the **Select Delivery Method** field to display the dropdown and choose **Deploy**.



3. In **Extract to Directory**, specify a path on the application server where the exported .ZIP file should be unzipped. The field will only have an effect if the exported file is a .ZIP file, otherwise this field will be ignored.
4. In **Copy to file**, specify a location and file name template to define a path where exported file should be delivered. Static text is also allowed in addition to, or instead of, the template options (which are indicated by the initial $). The elements in the example template shown in the image are described below:
   - **$tmpdir** This variable points to the directory path defined by the **ApplicationServer.TmpDir** entry in the sharedconfig.properties file. Remove '$tempdir' to enforce a location relative to the **DirectoryDeliveryPlugin.RootDir** delivery path in the sharedconfig.properties file.
   - **export** This represents and can be replaced with any static text required in the file name. For example, the Copy to File field could read 'out/EmergencySKUExport-$timestamp.$extension' instead, and would deliver the output to the 'out' directory below the path indicated in the properties file, and the file name would start with the text 'EmergencySKUExport-' before adding the timestamp, a period, and the file extension.
   - **$timestamp**: This variable is replaced with the time in year-day-month-hour-minutes-seconds-milliseconds, excluding the dashes. For example, 20161511094709625 is used for an export on 15 NOV 2016 at 9:47:09:625 AM.

     Adding a definition for the arrangement of the date / time elements provides a way to generate a more readable date. Static characters can be included to separate the timestamp elements and individual elements can be ordered as required, as shown below.

     Note: To revert the **$timestamp** variable so that it uses the legacy definition (milliseconds between January 1, 1970, and the time when the file is uploaded), contact Stibo Systems Support for more information.

     $timestamp(ddMMyyyy): The timestamp is modified to display day-month-year (excluding dashes) and is output as '15112016' for 15 NOV 2016.

     $timestamp(yyyy-MM-dd_HH.mm.ss): The timestamp is modified to display year-month-day_ hour.minutes.seconds (including the dashes, underscore, and periods) and is output as '2016-11-15_ 09.32.43' for 2016 NOV 15 at 9:32.43 AM.

   - **$extension** This variable is replaced with the extension of the selected output file format.

Note: The 'Deploy' template does not support conversions of file formats and can only be used to deliver files in the format specified on the Configuration tab of the outbound integration endpoint.

5. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

Dynamic JMS Delivery Method

The Dynamic JMS Delivery method uses the Java Message Service (JMS) messaging standard to deliver data to external systems. Unlike the JMS Delivery method, the Dynamic JMS Delivery method allows customers to supply the vendor-specific JMS libraries and JNDI configuration and, for example, upgrade to newer versions of these without Stibo Systems' involvement.

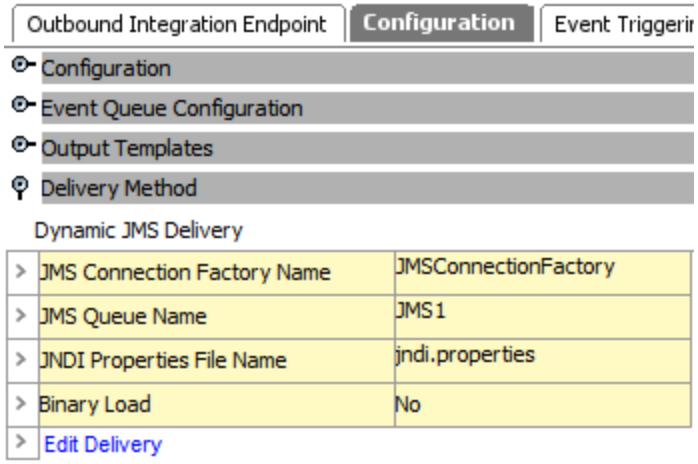
Important: This standard functionality only supports queues. Support for topics requires custom development via the **Extension API** (available from the Start Page using the **STEP API Documentation** link). Alternatively, topics can be supported using middleware to move the message from a queue to a topic.

The delivery method has been designed to work with any Java Message Service 2.0 specification-compliant client library (specification defined by JSR 343: https://jcp.org/aboutJava/communityprocess/final/jsr343/index.html), but will also work with systems that implement older versions of the specification.

Note: The delivery method has been tested with the following message brokers: Apache Active 5.15.8 and RabbitMQ 3.7.10.

This delivery option is only available in OIEPs.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



Prerequisites

Changes to the properties file or any files found in the class path directory, as outlined below, are implemented when the server is restarted.

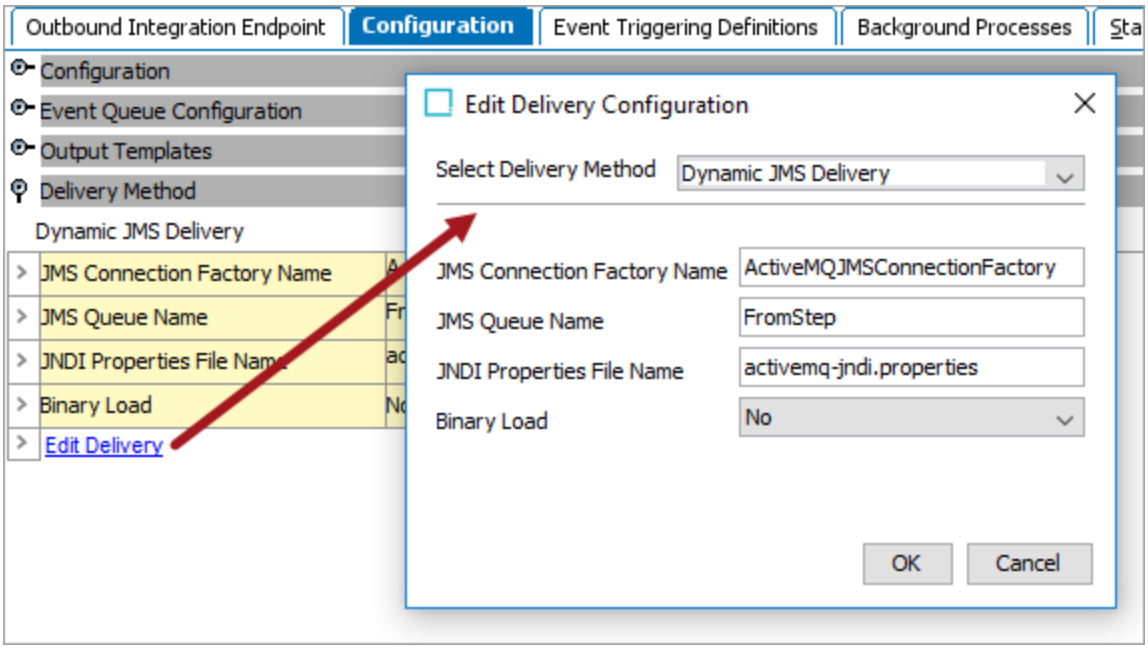
1. Verify the **jms-integration** add-on component is installed by reviewing the 'About STEP' option on the Start Page. Enter your credentials and click the 'Detailed version information' link. For on-premises systems, instructions for installing components can be found in the **SPOT Program** topic in the **System Administration Guide** found in **Downloadable Documentation**. For SaaS systems, contact Stibo Systems Support.
2. To enable the Dynamic JMS Delivery option, the case-sensitive configuration property 'JMS.ClassPath' must be set in sharedconfig.properties on the STEP application server and should point to an existing directory accessible from all application servers. Client libraries and JNDI files (Java Naming and Directory Interface) must be placed in this directory. Refer to the **Dynamic JMS Configuration Examples** section below for more information.

Once the server-side configuration is in place, the Dynamic JMS Delivery method can be configured via the workbench.

Configuration

For information on a parameter, hover over the parameter field to display help text.

1. On the **Configuration** tab, in the **Delivery Method** flipper, click **Edit Delivery**.



2. In **Select Delivery Method**, choose **Dynamic JMS Delivery**.
3. In **JMS Connection Factory Name**, specify a JMS connection factory name. The selection must match the entry configured in the JNDI file.
4. In **JMS Queue Name**, select the physical name of the JMS Queue to be used. The selection must match the entry configured in the JNDI file.

5. In **JNDI Properties File Name**, enter the name of the JNDI file.
6. In **Binary Load**, select 'Yes' if the message contents will be in a binary format (such as Excel).
7. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

Dynamic JMS Configuration Examples

As outlined in the following examples, in any JMS implementation, you must:

1. Put client jar files in the classpath folder.
2. Write the JNDI properties file.
3. Configure the connection factory and queue in the OIEP delivery method.

Azure Message Bus

Place these libraries in the directory pointed to by the `JMS.ClassPath` configuration property:

- geronimo-jms_1.1_spec-1.1.1.jar
- jakarta.jms-api-2.0.3.jar
- netty-buffer-4.1.82.Final.jar
- netty-codec-4.1.82.Final.jar
- netty-codec-http-4.1.82.Final.jar
- netty-common-4.1.82.Final.jar
- netty-handler-4.1.82.Final.jar
- netty-resolver-4.1.82.Final.jar
- netty-transport-4.1.82.Final.jar
- netty-transport-classes-epoll-4.1.82.Final.jar
- netty-transport-classes-kqueue-4.1.82.Final.jar
- netty-transport-native-epoll-4.1.82.Final-linux-x86_64.jar
- netty-transport-native-kqueue-4.1.82.Final-osx-x86_64.jar
- netty-transport-native-unix-common-4.1.82.Final.jar
- proton-j-0.34.0.jar
- qpid-jms-client-1.7.0.jar
- qpid-jms-discovery-1.7.0.jar

Important: Although the main JMS library is qpid-jms-client-[version].jar, the other libraries (listed above) are also required as these are underlying dependencies. Failing to include all of these in the JMS classpath will result in error messages similar to this one: `java.lang.NoClassDefFoundError: io/netty/util/concurrent/EventExecutorGroup`

Note: Due to a change in the Qpid JMS client's implementation, version 1.6.0+ of this library is not compatible with STEP's Dynamic JMS integration.

In the JNDI properties file, add properties to set the context factory and map queues, setting `connectionfactory = SBCF` as shown below:

```
java.naming.factory.initial = org.apache.qpid.jms.jndi.JmsInitialContextFactory
connectionfactory.SBCF = amqps://[servicebus_
name].servicebus.windows.net?jms.username=[username]&jms.password=[password]
```

Register queues in JNDI using the form:

- queue.[servicebus_queue_name] = [servicebus_queue_name]

Example azure-jndi.properties file:

```
java.naming.factory.initial = org.apache.qpid.jms.jndi.JmsInitialContextFactory
connectionfactory.SBCF =
amqps://pimtest.servicebus.windows.net?jms.username=RootManageSharedAccessKey&jms.password=4NzL79KOhmD8A9N8bA9QSTY3zxTHX9Hy1602xan0bqk=
queue.inbound-products = inbound-products
```

In the corresponding IIEP receiver, the JMS entries should have the following format:

- JMS Connection Factory Name SBCF
- JMS Queue Name [servicebus_queue_name]
- JNDI Properties File Name [jndi_properties_file_name]

Example receiver configuration:

- `JMS Connection Factory Name = SBCF`
- `JMS Queue Name = inbound-products`
- `JNDI Properties File Name = azure-jndi.properties`

For more information on the Qpid JMS client library, refer to https://github.com/MicrosoftDocs/azure-docs/blob/master/articles/service-bus-messaging/service-bus-java-how-to-use-jms-api-amqp.md

TibcoJMS

Libraries to be placed in the directory pointed to by the JMS.ClassPath configuration property:

- Tibjms.jar

The 'tibjms.jar' file can typically be found in:

```
components/shared/1.0.0/plugins/com.tibco.jms.jmsclient_[VersionNumber]
```

JNDI file content (example – file to be placed in the same directory as the libraries):

```
java.naming.provider.url=tibjmsnaming://[server1]:[port1], tibjmsnaming://[server2]:
[port2]
java.naming.factory.initial=com.tibco.tibjms.naming.TibjmsInitialContextFactory
java.naming.factory.url.pkgs=com.tibco.tibjms.naming
java.naming.security.principal=[username]
java.naming.security.credentials=[password]
```

Example delivery method configuration:

- `JMSConnectionFactoryName='QueueConnectionFactory'`
- `JMSQueueName='ToStep'`
- `JNDIPropertiesFileName='tibco-jndi.properties'`

With this configuration, use the connection factory ‘QueueConnectionFactory’ or the factory configured in the Kaazing Gateway 'factories.conf' file. Queue names can either be the name used in Tibco, or 'queue.[*add_the_tibcoqueuename*]' (which is needed when a topic exists on the JMS server with the same name as the queue you are trying to connect to).

Apache Active MQ 5.15.8

Libraries to be placed in the directory pointed to by the JMS.ClassPath configuration property:

- activemq-client-5.15.8.jar
- hawtbuf-1.11.jar

JNDI file content (example – file to be placed in the same directory as the libraries):

```
java.naming.factory.initial = org.apache.activemq.jndi.ActiveMQInitialContextFactory
java.naming.provider.url = tcp://127.0.0.1:61616
connectionFactoryNames = ActiveMQJMSConnectionFactory
queue.FromStep = FromStep
queue.ToStep = ToStep
java.naming.security.principal=admin
java.naming.security.credentials=admin
```

Example delivery method configuration:

- `JMS Connection Factory Name = 'ActiveMQJMSConnectionFactory'`
- `JMS Queue Name = 'ToStep'`
- `JNDI Properties File Name = 'activemq-jndi.properties'`

RabbitMQ

In RabbitMQ, create the necessary Queue(s) and Exchange(s) to allow message publishing and error reporting. For more information, search the web.

Libraries to be placed in the directory pointed to by the JMS.ClassPath configuration property. For each JAR file, use the latest updated version, indicated by the *{latest}* text in the following list:

- amqp-client-*{latest}*.jar
- fscontext-{latest}.jar
- rabbitmq-jms-{latest}.jar

Note: If credentials are to be provided, add them in the JNDI file as shown below since having the credentials in the .bindings file will not work.

JNDI file content (example – file to be placed in the same directory as the libraries):

```
java.naming.factory.initial = com.sun.jndi.fscontext.RefFSContextFactory
java.naming.provider.url = file:///opt/stibo/resources/jmsclasspath/

#Credentials
java.naming.security.principal = username
java.naming.security.credentials = password
```

'.bindings' file content must be in the directory identified by the `java.naming.provider.url` property above (example - file with this specific name to be placed in the same directory as the libraries):

```
# ConnectionFactory
ConnectionFactory/ClassName=javax.jms.ConnectionFactory
ConnectionFactory/FactoryName=com.rabbitmq.jms.admin.RMQObjectFactory
ConnectionFactory/RefAddr/0/Content=jms/ConnectionFactory
ConnectionFactory/RefAddr/0/Type=name
ConnectionFactory/RefAddr/0/Encoding=String
ConnectionFactory/RefAddr/1/Content=javax.jms.ConnectionFactory
ConnectionFactory/RefAddr/1/Type=type
ConnectionFactory/RefAddr/1/Encoding=String
ConnectionFactory/RefAddr/2/Content=com.rabbitmq.jms.admin.RMQObjectFactory
```

```
ConnectionFactory/RefAddr/2/Type=factory
ConnectionFactory/RefAddr/2/Encoding=String
ConnectionFactory/RefAddr/3/Content=superhost
ConnectionFactory/RefAddr/3/Type=host
ConnectionFactory/RefAddr/3/Encoding=String
ConnectionFactory/RefAddr/4/Content=STEP-frva
ConnectionFactory/RefAddr/4/Type=virtualHost
ConnectionFactory/RefAddr/4/Encoding=String
ConnectionFactory/RefAddr/5/Content=5672
ConnectionFactory/RefAddr/5/Type=port
ConnectionFactory/RefAddr/5/Encoding=String
ConnectionFactory/RefAddr/6/Content=false
ConnectionFactory/RefAddr/6/Type=ssl
ConnectionFactory/RefAddr/6/Encoding=String

# Product-Synchro Queue
Product-Synchro/ClassName=javax.jms.Queue
Product-Synchro/FactoryName=com.rabbitmq.jms.admin.RMQObjectFactory
Product-Synchro/RefAddr/0/Content=jms/Queue
Product-Synchro/RefAddr/0/Type=name
Product-Synchro/RefAddr/0/Encoding=String
Product-Synchro/RefAddr/1/Content=javax.jms.Queue
Product-Synchro/RefAddr/1/Type=type
Product-Synchro/RefAddr/1/Encoding=String
Product-Synchro/RefAddr/2/Content=com.rabbitmq.jms.admin.RMQObjectFactory
Product-Synchro/RefAddr/2/Type=factory
Product-Synchro/RefAddr/2/Encoding=String
Product-Synchro/RefAddr/3/Content=Product-Synchro
Product-Synchro/RefAddr/3/Type=destinationName
Product-Synchro/RefAddr/3/Encoding=String
Product-Synchro/RefAddr/4/Content=true
Product-Synchro/RefAddr/4/Type=amqp
Product-Synchro/RefAddr/4/Encoding=String
Product-Synchro/RefAddr/5/Content=amq.fanout
Product-Synchro/RefAddr/5/Type=amqpExchangeName
Product-Synchro/RefAddr/5/Encoding=String
Product-Synchro/RefAddr/6/Content=Product-Synchro
Product-Synchro/RefAddr/6/Type=amqpRoutingKey
Product-Synchro/RefAddr/6/Encoding=String
Product-Synchro/RefAddr/7/Content=Product-Synchro
Product-Synchro/RefAddr/7/Type=amqpQueueName
Product-Synchro/RefAddr/7/Encoding=String
```

Example delivery method configuration:

- `JMS Connection Factory Name = ConnectionFactory`
- `JMS Queue Name = rabbitmqQueue01`
- `JNDI Properties File Name = rabbitmq-jndi.properties`

WebLogic JMS Connection

Libraries to be placed in the Workarea directory, which points to by the jmslasspath configuration property:

- weblogic.jar
- wlclient.jar
- wlthint3client.jar

Important: Because STEP will attempt to use some classes that are in both the `wljmsclient.jar` and the `wlthint3client.jar` libraries, conflicts will generate. To prevent this conflict, use `wlthint3client.jar` library instead of the `wljmsclient.jar` library.

The weblogic-JNDI configuration includes:

```
java.naming.factory.initial=weblogic.jndi.WLInitialContextFactory
java.naming.provider.url=t3://localhost:7001
```

This includes the queue name of:

```
jms/TestQueue=jms/TestQueue
```

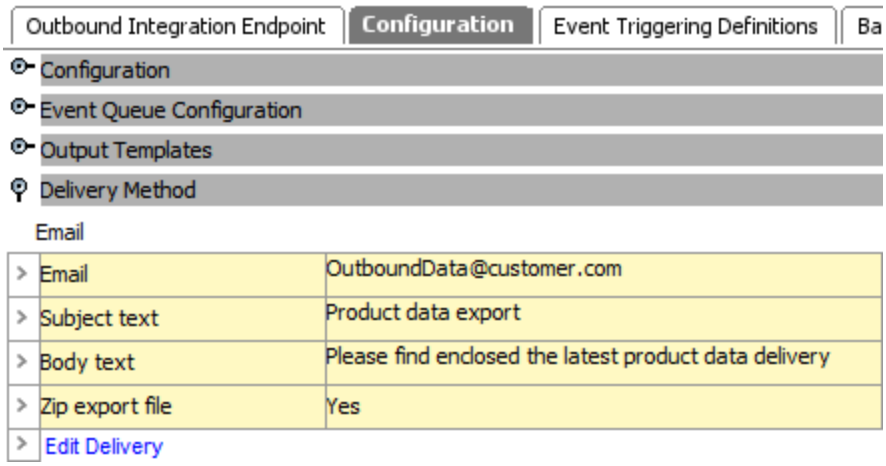
Example delivery method configuration:

```
JMS Connection Factory Name = weblogic.jms.ConnectionFactory
JMS Queue Name = jms/TestQueue
JNDI Properties File Name = weblogic-jndi.properties
```

Email Delivery Method

Similar to the 'Copy to directory' delivery method, the email delivery option allows the delivery to be zipped. This method is typically used to test the export output, or to deliver a small export to one or more persons.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



This delivery method is also available in Export Manager as defined in the **Email Delivery Method** topic.

Prerequisites

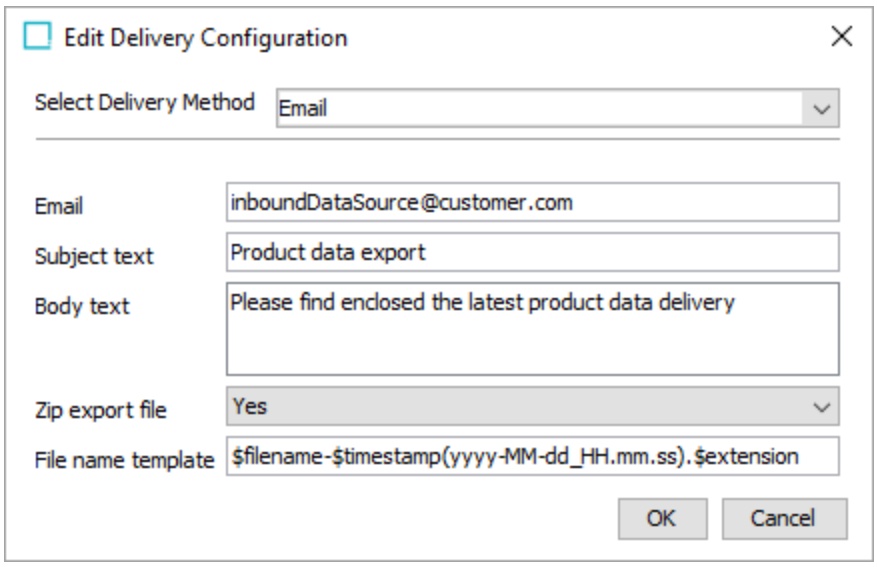
To deliver an exported file via email, the STEP server must have access to the SMTP server. Be aware of email file-size limits because, depending on the export's contents, the file could be very large.

For information on configuring email from STEP, refer to the **Email from STEP** topic in the **Resource Materials** of online help.

Configuration

For information on a parameter, hover over the parameter field to display help text.

1. Click the **Select Delivery Method** parameter to display the dropdown and choose **Email**.
2. For the **Email** parameter, enter an email address. For multiple recipients, enter the email addresses separated by a semicolon.
3. For the **Subject text** parameter, enter the text that will display for the email subject line. The server name is prepended to the subject text on the email.
4. For the **Body text** parameter, enter the text that will display for the email body.



5. In **Zip export file**, select 'yes' or 'no' from the dropdown to specify if the output file should be delivered as an email attachment in a .ZIP (compressed) file format.

   Note: For the Email delivery method, the files named in the ZIP file do not respect the $filename variable.

   - **Yes** uses 'export-' before the timestamp variable, and then the extension ZIP. For example, a zipped STEPXML output with the default File Name Template exported on 15 NOV 2016 results in an output .ZIP file named 'export-1479230247017.zip.' The contents of the ZIP file follow the $filename variable applied in the 'File name template' parameter.
   - **No** uses the file type for the file name along with the appropriate extension for the selected data format.

6. In **File name template**, to create the file name used for the output, enter template variables (identified with a $), static text, or use a combination of both. The default template is $filename-$timestamp(yyyy-MM-dd_HH.mm.ss).$extension. Depending on the settings of the configuration, the default template outputs a file name similar to 'GenericXML-2020-07-30_14.07.44.xml.' Each variable is described below:

   Note: The 'Zip export file' parameter also has an impact on the file name.

   - **$filename:** This variable is replaced with text to indicate the format of the delivered file as specified in the Output Templates flipper. For example, the output file name would include the text 'excel' or 'GenericXML' for those formats, or the Event ID for event-based STEPXML. Using the default file name template, a comma-separated value file would be named 'csv--2020-07-30_14.07.44.csv' and 'GenericXML--2020-07-30_14.07.44.xml' would indicate that Generic XML was used.
   - **$timestamp**: This variable is replaced with the time in year-day-month-hour-minutes-seconds-milliseconds, excluding the dashes. For example, 20161511094709625 is used for an export on 15 NOV 2016 at 9:47:09:625 AM.

Adding a definition for the arrangement of the date / time elements provides a way to generate a more readable date. Static characters can be included to separate the timestamp elements and individual elements can be ordered as required, as shown below.

Note: To revert the **$timestamp** variable so that it uses the legacy definition (milliseconds between January 1, 1970, and the time when the file is uploaded), contact Stibo Systems Support for more information.

$timestamp(ddMMyyyy): The timestamp is modified to display day-month-year (excluding dashes) and is output as '15112016' for 15 NOV 2016.

$timestamp(yyyy-MM-dd_HH.mm.ss): The timestamp is modified to display year-month-day_ hour.minutes.seconds (including the dashes, underscore, and periods) and is output as '2016-11-15_ 09.32.43' for 2016 NOV 15 at 9:32.43 AM.

- **$extension:** This variable is replaced with the extension of the output file based on the selected format in the Output Templates flipper. For XML-based formats, the output file is set with the .XML extension, while Excel files use .XLS, and comma-separated values use .CSV.

Note: The File Name Template does not support conversions of file formats and can only be used to deliver files in the format specified in the configuration.

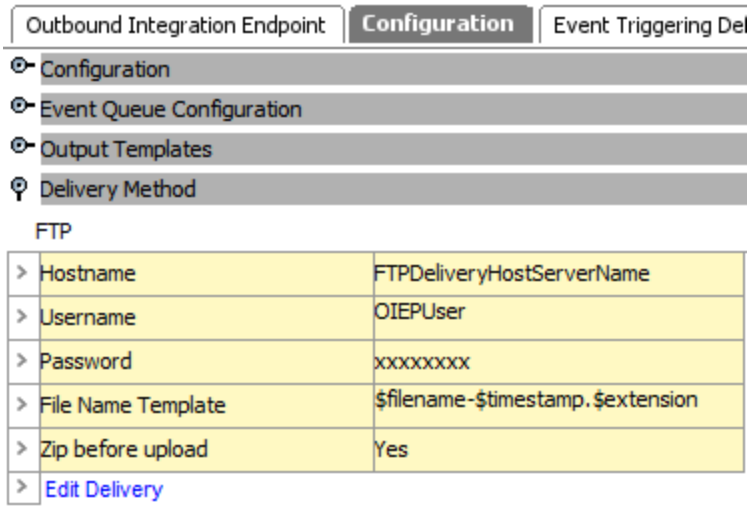
7. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

FTP Delivery Method

The FTP delivery method allows an exported file to be delivered to an external system and is often used when the output files are large, or when a different or remote system is in use. The preferred method for FTP delivery is the SFTP secure delivery method.

For information on the OIEP SFTP delivery method, refer to the **SFTP Delivery Method** topic.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



To use this delivery method with Export Manager, refer to the **FTP Delivery Method** topic.

Prerequisites

Changes to the properties file, outlined below, are implemented when the server is restarted.

Prior to configuration, clicking the **HostName** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **FTPDeliveryHostName** property. The required format of the property is (square brackets not included):

```
SFTPDeliveryHostName=1=[host1],2=[host2]
```

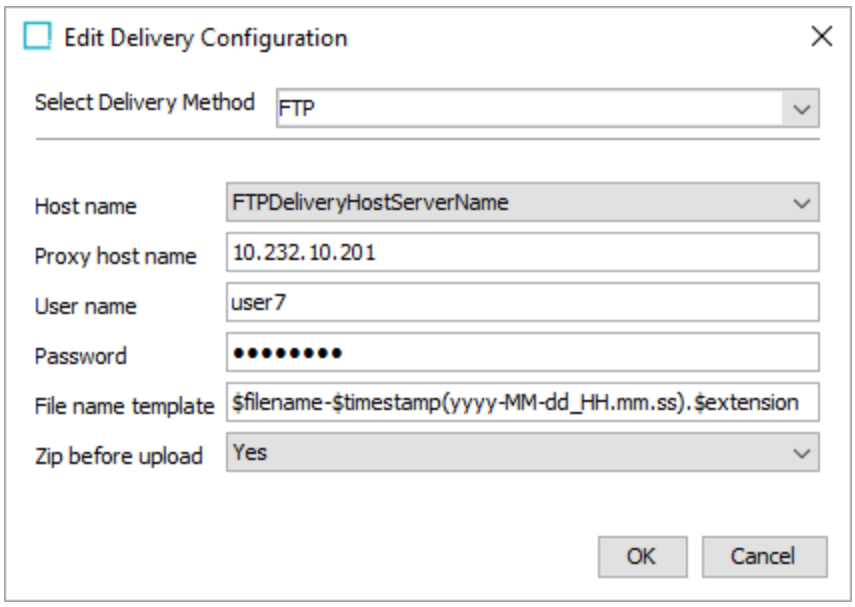
Using the host name shown in the image below, the entry in the properties file would be:

```
FTPDeliveryHostName=1=FTPDeliveryHostServerName
```

Configuration

For information on a parameter, hover over the parameter field to display help text.

1. In **Select Delivery Method**, choose **FTP**.



2. In **Host name**, select the host name or IP address of the FTP server to be used for the delivery.
3. In **Proxy host name**, add the host name to be used for the server proxy. This field is optional.
4. In **User name**, enter the user name that has access to log on to the FTP server.
5. In **Password**, enter the password that will be used to log on to the FTP server.
6. In **File name template**, to create the file name used for the output, enter template variables (identified with a $), static text, or use a combination of both. The default template is $filename-$timestamp(yyyy-MM-dd_HH.mm.ss).$extension. Depending on the settings of the configuration, the default template outputs a file name similar to 'GenericXML-2020-07-30_14.07.44.xml.' Each variable is described below:
   - **$filename** For event-based OIEPs, this variable is replaced with text to indicate the format of the delivered file as specified in the Output Templates flipper, except for STEPXML when the first and last Event IDs are used. For example, the output file name could be 'csv-2020-07-30_14.09.40.csv' or '1804038-1804038.xml' to indicate that STEPXML was used for a single event.

     Note: The 'Zip before upload' parameter also has an impact on the file name.

   - **$timestamp**: This variable is replaced with the time in year-day-month-hour-minutes-seconds-milliseconds, excluding the dashes. For example, 20161511094709625 is used for an export on 15 NOV 2016 at 9:47:09:625 AM.

     Adding a definition for the arrangement of the date / time elements provides a way to generate a more readable date. Static characters can be included to separate the timestamp elements and individual elements can be ordered as required, as shown below.

     Note: To revert the **$timestamp** variable so that it uses the legacy definition (milliseconds between January 1, 1970, and the time when the file is uploaded), contact Stibo Systems Support for more information.

$timestamp(ddMMyyyy): The timestamp is modified to display day-month-year (excluding dashes) and is output as '15112016' for 15 NOV 2016.

$timestamp(yyyy-MM-dd_HH.mm.ss): The timestamp is modified to display year-month-day_ hour.minutes.seconds (including the dashes, underscore, and periods) and is output as '2016-11-15_ 09.32.43' for 2016 NOV 15 at 9:32.43 AM.

- **$extension** This variable is replaced with the extension of the output file based on the selected format in the Output Templates flipper. For XML-based formats, the output file is set with the .XML extension, while Excel files use .XLS, and comma-separated values use .CSV.

Note: The File Name Template does not support conversions of file formats and can only be used to deliver files in the format specified on the Configuration tab > Output Templates flipper of the outbound integration endpoint.

7. In **Zip before upload**, select an option from the dropdown to specify if the output file should be delivered in a .ZIP (compressed) file format.
   - **Yes** uses 'result_0' before the timestamp variable, and then the extension ZIP. The File Name Template is used for the contents of the ZIP file. For example, a zipped STEPXML output with the default File Name Template exported on 15 NOV 2016 results in an output .ZIP file named 'result_0-2020-07-30_ 14.07.44.zip.' The contents of the ZIP file follow the $filename variable applied in the 'File name template' parameter.
   - **No** uses the File Name Template for the file name along with the appropriate extension for the selected data format.
8. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

Git Delivery Method

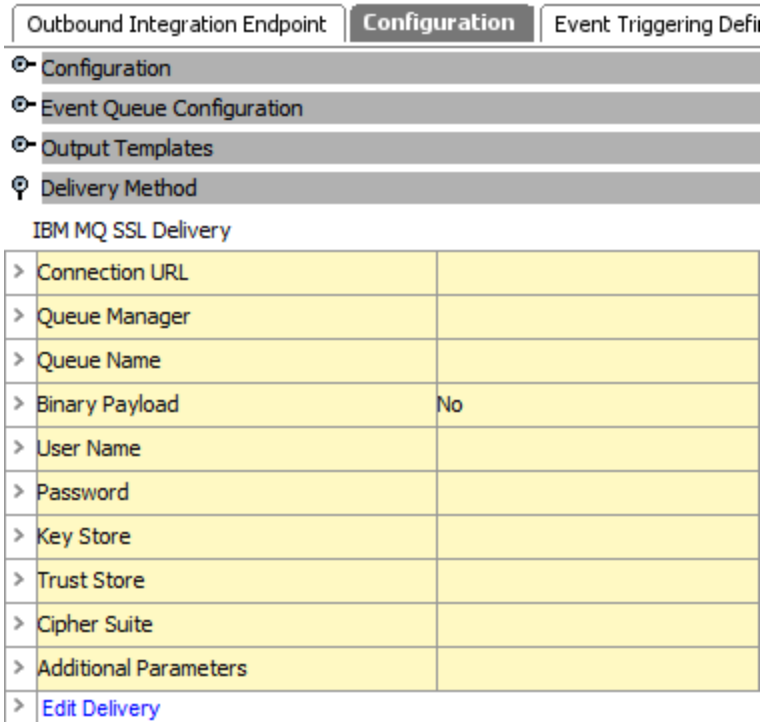
The Git Delivery method delivers files produced by the outbound integration endpoint (OIEP) processing engine or a configured post-processor to a branch in a remote Git repository. Refer to https://git-scm.com for more information about Git.

For details, refer to the **Integration Endpoint Options for VCS Integration** topic in the **Configuration Management** documentation.

IBM MQ SSL Delivery Method

This delivery method allows connection with IBM MQ. Prior to release 2023.3, it was named 'IBM Websphere MQ SSL Delivery'. For information on connecting to IBM MQ in a non-SSL way, refer to **JMS Delivery Method**. IBM MQ SQL delivery is only available in an OIEP.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



Prerequisites

Changes to the properties file, outlined below, are implemented when the server is restarted.

Prior to configuration, clicking a dropdown parameter in the 'Edit Delivery Configuration' dialog displays the property name required to supply values that populate the parameter.

Note: In the sharedconfig.properties file, a numbered designation of an integer (**1=**, **2=**, etc.) in the value part of the property indicates that an entry should display in the UI. This allows multiple values to be stored for a single property and is required even when only a single value is required. If property values, such as passwords, should not be displayed in the UI, exclude the numbered designation, as shown in the password examples below.

The numbered designation indicates the order that the options are displayed in the dropdown. For example: `<Property name>=1=<Value 1>,2=<Value 2>,3=<Value 3>`. Using these numbered designations results in the dropdown showing values in the following order: `<Value 1>,<Value 2>,<Value 3>`. When duplicate integers exist, only the last value is displayed in the dialog.

As required, configure the following case-sensitive properties in the sharedconfig.properties on the STEP application server:

1. For **Connection URL**, set the **WSMQSSLProviderURL** property, as shown below.

   Use the [host]:[port]/[channel] format as defined:

   - [host] = hostname or IP of the MQ server; in the follow example this is 10.46.88.75
   - [port] = port number for the channel; in the follow example this is 1415
   - [channel] = name of the channel; in the follow example this is BASE.CTL.SVRCONN

   ```
   WSMQSSLProviderURL=1=10.46.88.75:1415/BASE.CTL.SVRCONN
   ```

2. For **Queue Manager**, set the **WSMQSSLQueueManager** property, as shown below.

   ```
   WSMQSSLQueueManager=1=HV088B
   ```

3. For **Queue Name**, set the **WSMQSSLQueue** property, as shown below.

   ```
   WSMQSSLQueue=1=LIVE.KITT
   ```

4. For **Key Store**, set the **WSMQSSLKeyStoreLocation** property, as shown below.

   Use the jks format with the personal certificate for the Queue Manager. To generate this, refer to the IBM MQ online help. To indicate a URL and not a path, prefix the Key Store with 'file:'.

   ```
   WSMQSSLKeyStoreLocation=1=file:/workarea/keystore.jks
   ```

5. For **Key Store Password**, set the **WSMQSSLKeyStorePassword** property, as shown below.

   The password can be configured in sharedconfig.properties file or can be typed directly into the parameter. When the parameter is blank, the password from the property is used.

   To prevent the password (or any other selections) from displaying in the UI, do not include a numbered designation.

   ```
   WSMQSSLKeyStorePassword=Pa55w0rd1
   ```

6. For **Trust Store**, set the **WSMQSSLTrustStoreLocation** property, as shown below.

   This can be the same file as key store. To generate this, refer to the IBM MQ online help. As shown below, prefix the Trust Store with 'file:' to indicate a URL, not a path.

   ```
   WSMQSSLTrustStoreLocation=1=file:/workarea/truststore.jks
   ```

7. For **Trust Store Password**, set the **WSMQSSLTrustStorePassword** property, as shown below.

   The password can be configured in sharedconfig.properties file or can be typed directly into the parameter. When the parameter is blank, the password from the property is used.

To prevent the password (or any other selections) from displaying in the UI, do not include a numbered designation.

```
WSMQSSLTrustStorePassword=Pa55w0rd2
```

8. For **Cipher Suite**, set the **WSMQSSLCipherSuite** property, as shown below.

   STEP is running on non-IBM JRE, so this must be the same value as configured in the Queue Manager.

   ```
   WSMQSSLCipherSuite=1=CTL_RSA_WITH_AES_256
   ```

9. Contact your IT team to create a trust and key store and copy all URL certificates to the trust and key store.

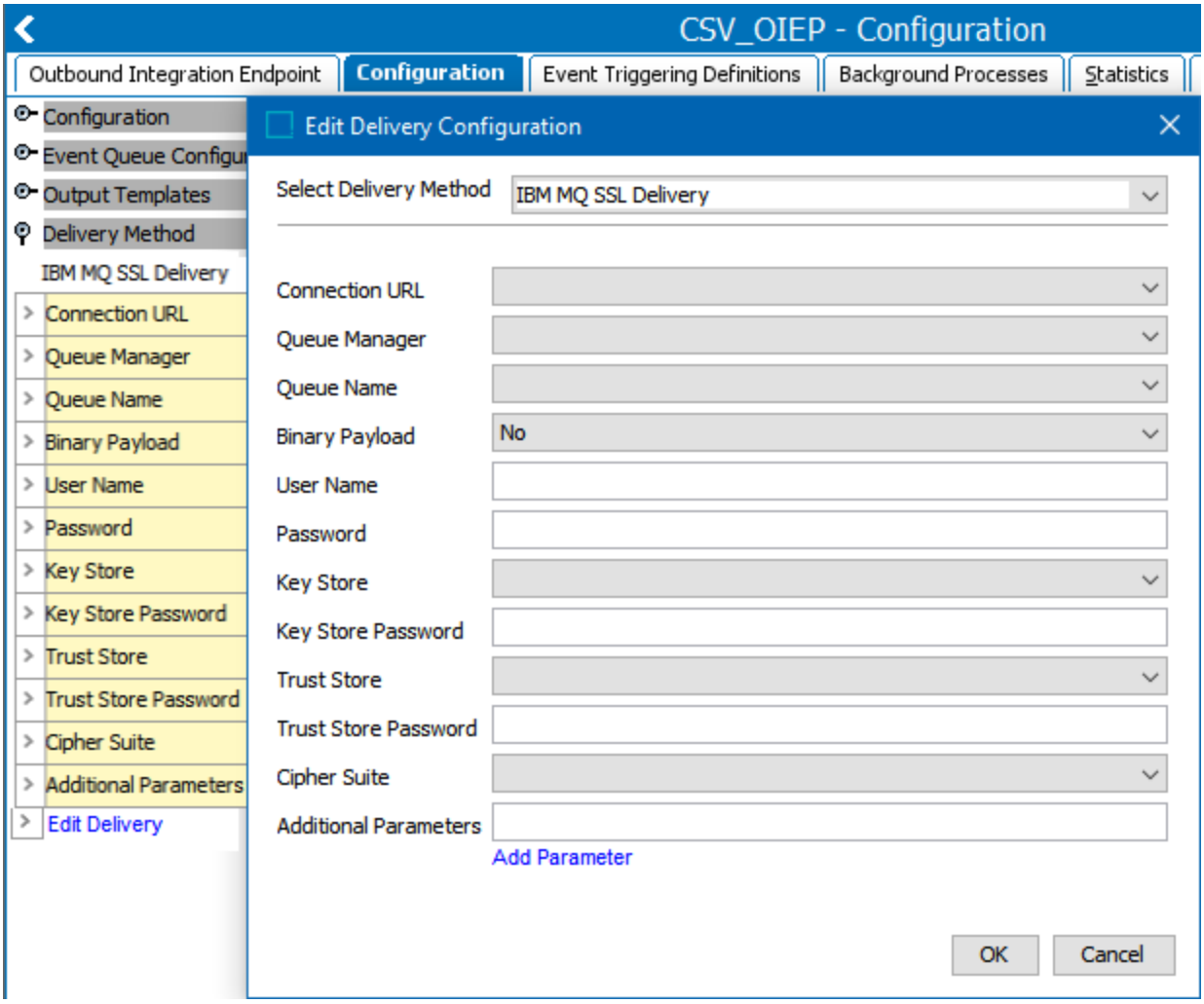
Example of all property entries

```
WSMQSSLProviderURL=1=10.46.88.75:1415/BASE.CTL.SVRCONN
WSMQSSLQueueManager=1=HV088B
WSMQSSLQueue=1=LIVE.KITT
WSMQSSLKeyStoreLocation=1=/workarea/key.jks
WSMQSSLKeyStorePassword=Pa55w0rd1
WSMQSSLTrustStoreLocation=1=/workarea/key.jks
WSMQSSLTrustStorePassword=Pa55w0rd2
WSMQSSLCipherSuite=1=CTL_RSA_WITH_AES_256
```

Configuration

For information on a parameter, hover over the parameter field to display help text.

1. In the **Select Delivery Method** field, choose **IBM MQ SSL Delivery**.



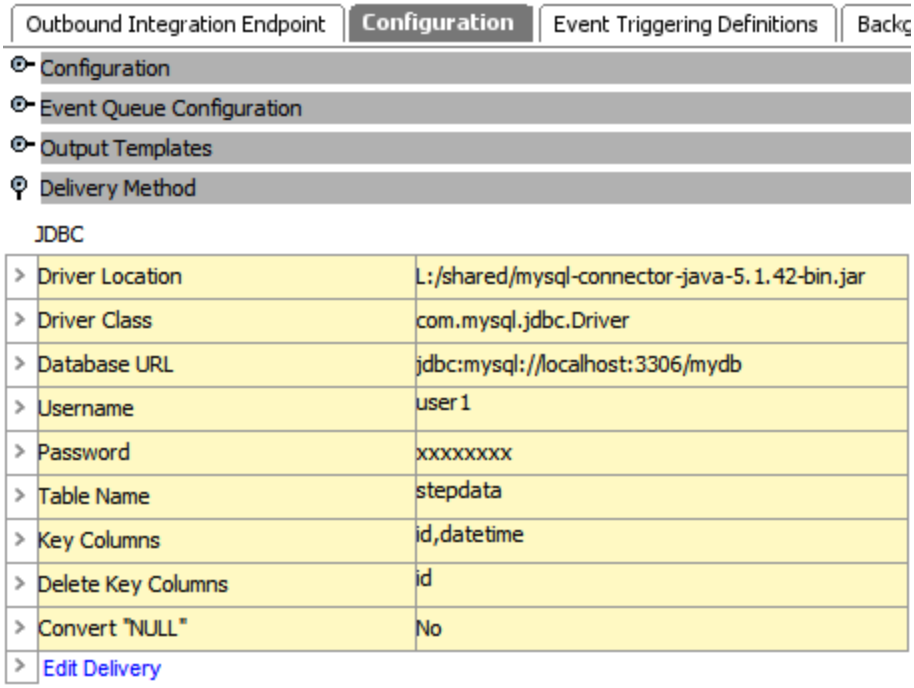
2. In **Connection URL**, select the URL for the connection.
3. In **Queue Manager**, select the name of the Queue Manager.
4. In **Queue Name**, select the name of the Queue for the connection.
5. In **Binary Payload**, select 'Yes' if the message contents will be in a binary format (such as Excel).
6. In **User Name**, if required, enter the user name to be used with this integration.
7. In **Password**, if required, enter the password to be used with this integration.
8. In **Key Store**, select the keystore in jks format, with the personal certificate for the Queue Manager.
9. In **Key Store Password**, leave the parameter blank to use the password from the sharedconfig.properties file, or manually type in a password.
10. In **Trust Store**, select the trust store with the CA for the Queue Manager.

11. In **Trust Store Password**, leave the parameter blank to use the password from the sharedconfig.properties file, or manually type in a password.
12. In **Cipher Suite**, set to the same value as SSL CipherSuite in IBM MQ.
13. If **Additional Parameters** are required, click the **Add parameter** link and enter the Key and the Value. For valid keys and values, refer to the IBM MQ online help.
14. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

JDBC Delivery Method

The JDBC delivery method allows STEP data to be delivered directly to tables in Relational database management systems (RDBMS) like Oracle, MySQL, MS SQL Server, and PostgreSQL. Though the ways the JDBC plugin can be deployed are various, one of its prime uses is to send STEP data to data analytics tools for the purpose of displaying STEP data in a data analytics dashboard. This delivery method requires CSV format and is available in both the Export Manager and in outbound integration endpoints (OIEPs).

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



For more information on the Export Manager option, refer to **JDBC Delivery Method** in the **Export Manager** documentation.

Prerequisites

Important: For complete setup requirements, refer to the **Exporting Data via JDBC with CSV Format** topic in the **Data Format** documentation.

Install the required drivers

JDBC specification 4.1-compliant drivers should be placed in a directory accessible from all application servers. These drivers can then be made available for the delivery plugin via the dynamic properties JDBCDeliveryPlugin.DriverPath.[n] and JDBCDeliveryPlugin.DriverClass.[n]. For more information regarding applicable Java drivers, review the RDBMS vendor's homepage on the web.

Configure data for the dropdown parameters

Changes to the properties file, outlined below, are implemented when the server is restarted.

1. Prior to configuration, clicking the **Driver Location** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **JDBCDeliveryPlugin.DriverPath.[n]** property. As an example:

   ```
   JDBCDeliveryPlugin.DriverPath.1 = L:/shared/mysql-connector-java-5.1.42-bin.jar
   ```

   In this example, the drivers are stored on the application server's L:/shared drive.

2. Prior to configuration, clicking the **Driver Class** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **JDBCDeliveryPlugin.DriverClass.[n]** property. As an example:

   ```
   JDBCDeliveryPlugin.DriverClass.1 = com.mysql.jdbc.Driver
   ```

   In this example, the drivers class used is 'com.mysql.jdbc.Driver.'

3. Prior to configuration, clicking the **Database URL** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **JDBCDeliveryPlugin.URL.[n]** property.

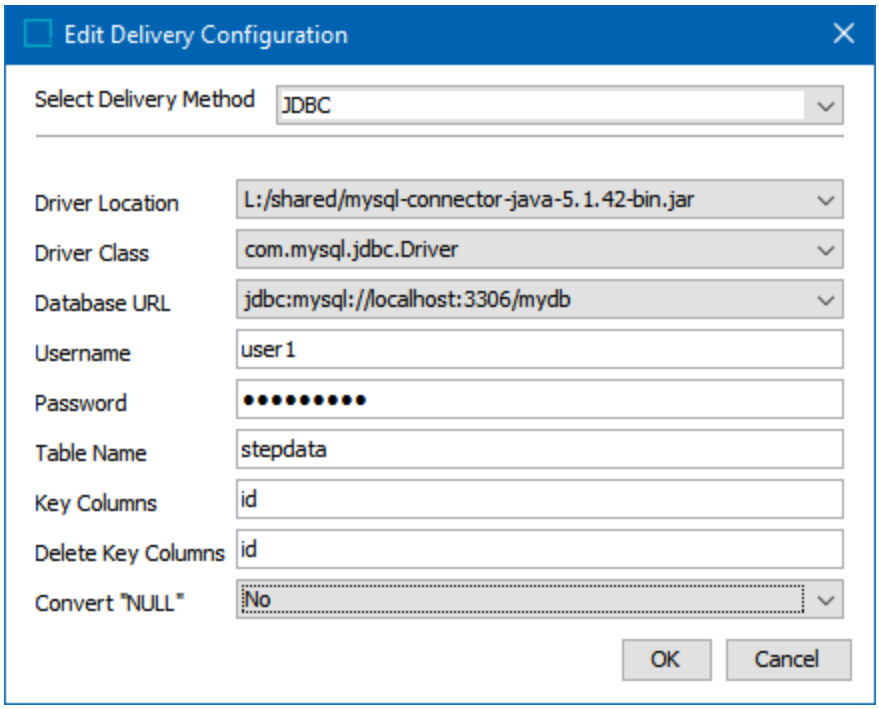
   ```
   JDBCDeliveryPlugin.URL.1 = jdbc:mysql://localhost:3306/mydb
   ```

   In this example, the database URL used is 'jdbc:mysql://localhost:3306/mydb.'

Configuration

For information on a parameter, hover over the parameter field to display help text.

In the OIEP editor on the **Configuration** tab, navigate to the **Delivery Method** flipper, then click **Edit Delivery**.



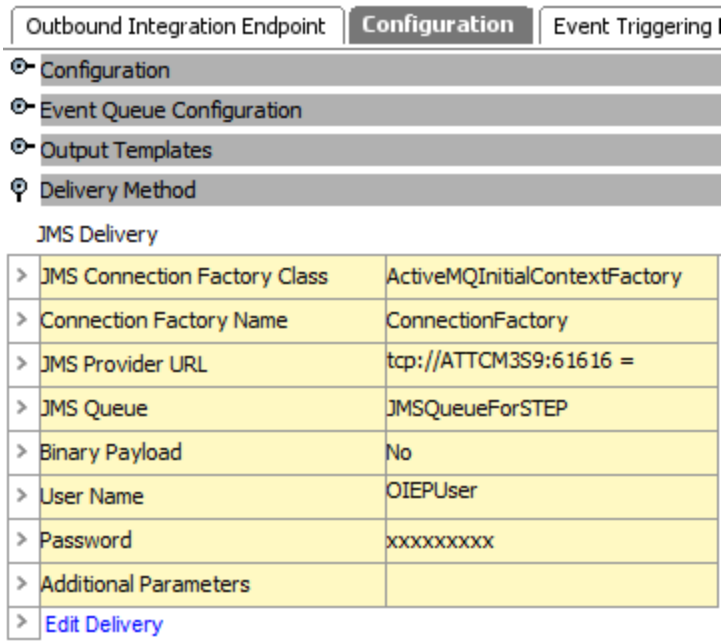
1. In **Select Delivery Method**, choose **JDBC** from the dropdown.
2. In **Driver Location**, select from the dropdown one of the paths to the relevant .jar file.
3. In **Driver Class**, select from the dropdown one of the pre-configured driver class.
4. In **Database URL**, select from the dropdown one of the pre-configured URLs to the destination database.
5. In **Username**, enter the username required to access the destination database.
6. In **Password**, enter the password required to access the destination database.
7. In **Table Name**, enter the name of the table in the destination database to which STEP will be publishing data
8. In **Key Columns**, list the names of the columns appearing on both the exported CSV file and the table in the destination database, separated by commas (and no spaces), into which STEP will publish data
9. In **Delete Key Columns**, list the names of the columns appearing on both the exported CSV file and the table in the destination database, separated by commas (and no spaces), from which STEP will delete data. The headers contained in this field can differ from the headers in the 'Key Columns' field, but they must also be part of the upsert key definition.
10. In **Convert "NULL"**, choose Yes if the string “NULL” should be converted to the value null. This may, for instance, be used for clearing a value in a column in the target database. This parameter defaults to No.
11. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

JMS Delivery Method

The available options for the Java Message Service (JMS) delivery method are system dependent. STEP has the following standard JMS delivery options: Apache Active MQ, WebSphere MQ, and Oracle AQ. Each of these can be used to deliver messages to a JMS Receiver without any further customizations. To deliver messages to other JMS providers, contact your Stibo Systems account manager for further customizations. This delivery option is only available in OIEPs.

Important: This standard functionality only supports queues. Support for topics requires custom development via the **Extension API** (available from the Start Page using the **STEP API Documentation** link). Alternatively, topics can be supported using middleware to move the message from a queue to a topic.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



Prerequisites

Changes to the properties file, outlined below, are implemented when the server is restarted.

Refer to the **Sharedconfig.properties File Examples** section below for more information.

1. Prior to configuration, clicking the **Connection Factory Name** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive `JMSConnectionFactoryName` property.

2. Prior to configuration, clicking the **JMS Provider URL** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive `JMSProviderURL` property. If necessary, use a comma to separate multiple URLs.

   The following is an example of a complete property entry for two systems named 'qa' and 'stage,' as well as their URLs 'http://step-qa' and 'http://step-stage':

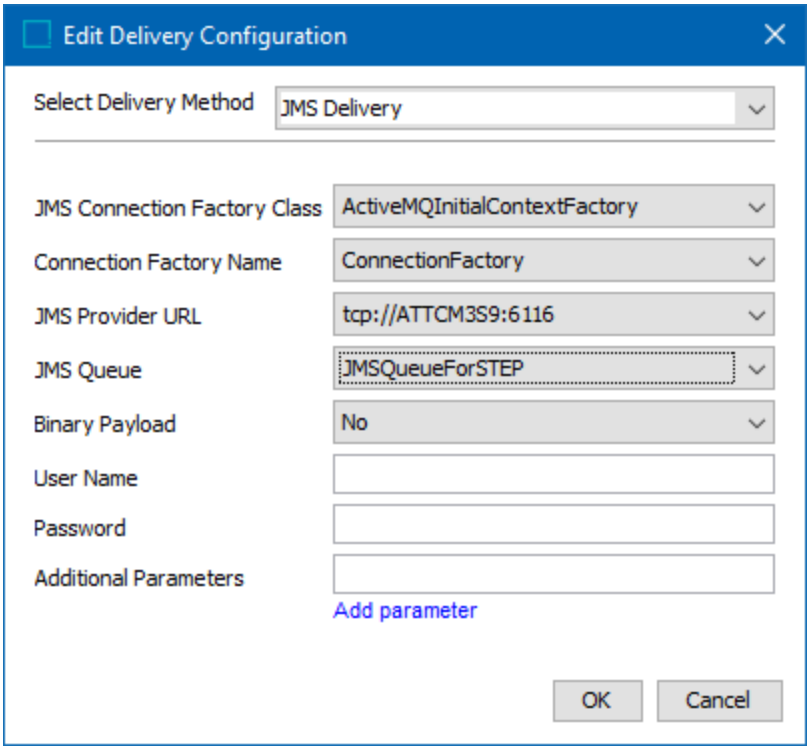
   ```
   JMSProviderURL=qa=http://step-qa,stage=http://step-stage
   ```

3. Prior to configuration, clicking the **JMS Queue** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive `JMSQueue` property.

Configuration

For information on a parameter, hover over the parameter field to display help text.

1. On the **Configuration** tab, in the **Delivery Method** area, click **Edit Delivery**.
2. In **Select Delivery Method**, choose **JMS Delivery**.



3. In **JMS Connection Factory Class**, choose the JMS connection factory class that corresponds to the driver class from the JMS provider vendor. On a standard system, the following options are available:

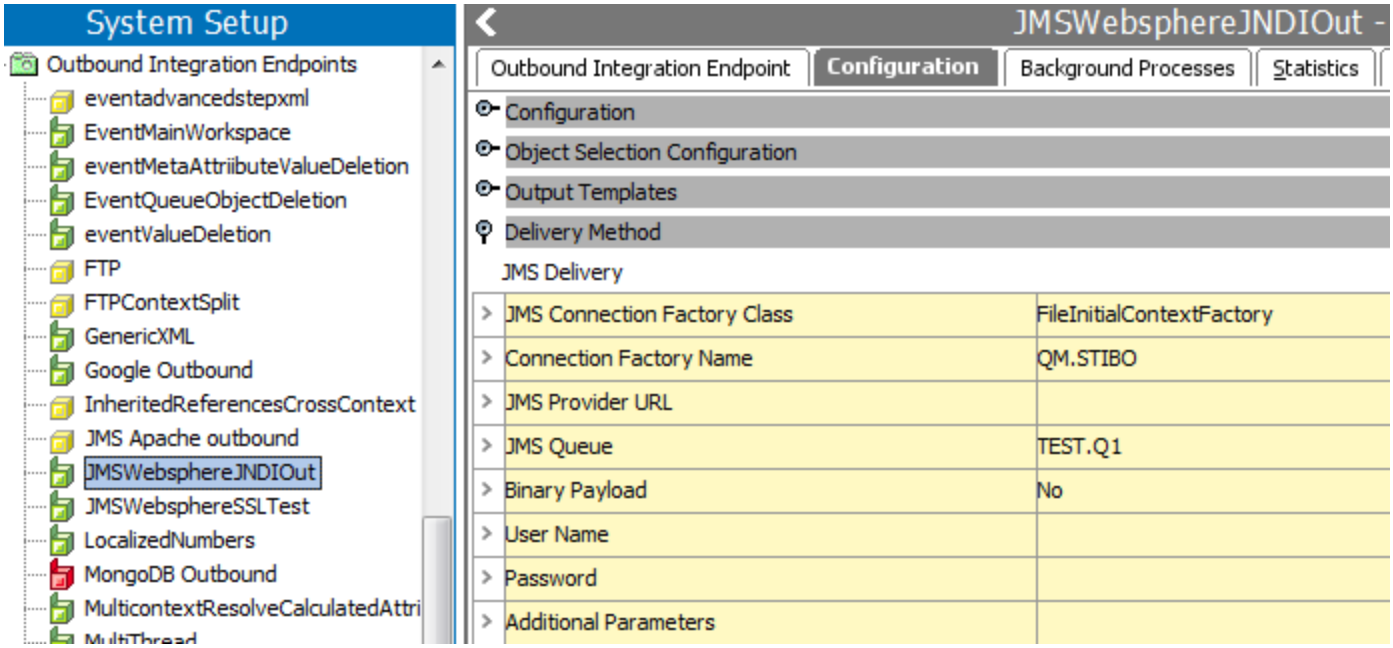
- **ActiveMQInitialConnectionFactory**: Connects the JMS delivery to Apache Active MQ. For information, search the web.

  Note: WMQInitialContextFactory is not supported, use FileInitialContextFactory instead.

- **FileInitialContextFactory**: Enables setting up JMS WebSphere inbound and outbound integration endpoints which reference a binding file (created from JMS WebSphere Client software). The binding file is a configuration file which includes all details of how STEP should interact with JMS WebSphere. For information about JMS WebSphere Client Software, search the web.

4. In **Connection Factory Name**, select a connection factory name from the list.
5. In **JMS Provider URL**, select a JMS Provider URL from the list.
6. In **JMS Queue**, select the physical name of the JMS Queue to be used on Apache Active MQ or WebSphere MQ.
7. In **Binary Payload**, select 'Yes' if the message contents will be in a binary format (such as Excel).
8. In **User Name**, if required, enter the user name that will be used to log onto the JMS provider.
9. In **Password**, if required, enter the password that will be used to log onto the JMS provider.
10. If additional parameters are required, click **Add parameter**, then enter the Key and Value.
11. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

FileInitialContextFactory JMS Delivery Configuration Example



JMS WebSphere Delivery Using SSL Configuration Example

Delivery Method	
IBM Websphere MQ SSL Delivery	
Connection URL	webspheremq-qa.stibo.com:1417/STEP.SVRCONN
Queue Manager	QM.STIBO_SSL
Queue Name	TEST.Q1
Binary Payload	Yes
User Name	
Password	
Key Store	file:/workarea/JMSWebsphereSSLKeystore/keyStore.jks
Trust Store	file:/workarea/JMSWebsphereSSLKeystore/keyStore.jks
Cipher Suite	TLS_RSA_WITH_AES_256_CBC_SHA256
Additional Parameters	

Apache Active MQ JMS Delivery

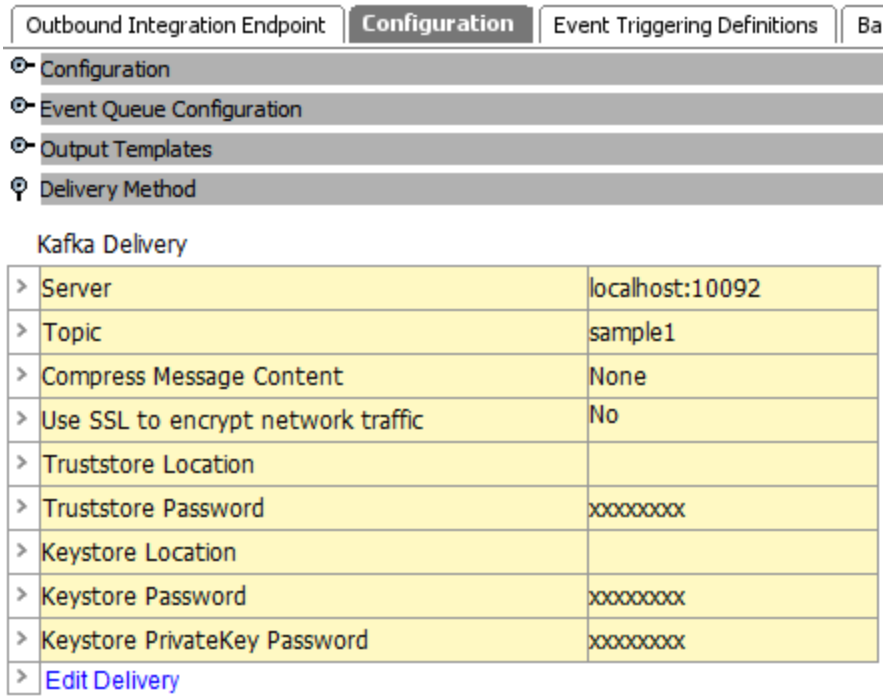
Delivery Method	
JMS Delivery	
JMS Connection Factory Class	ActiveMQInitialContextFactory
Connection Factory Name	ConnectionFactory
JMS Provider URL	tcp://ATTCM3S9:61616
JMS Queue	testqueue
Binary Payload	No
User Name	
Password	
Additional Parameters	
Edit Delivery	

Kafka Delivery Method

The Kafka Delivery method enables a STEP platform integrated with Apache Kafka, which is an open-source distributed event-streaming data platform, to use a Kafka queue. For more information about Apache Kafka, search the web.

This delivery method is only available in OIEPs.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



Prerequisites

1. Before setting up a Kafka Delivery, read the **Considerations for Setting up Kafka Receiver or Delivery** topic in this documentation.
2. Prior to configuration, clicking the **Server** dropdown parameter displays the required server name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive ***Kafka.Server*** configuration property. If connecting to a cluster, use a comma-separated list.

   The following example shows two Kafka server configurations where the first server is a cluster:

   ```
   Kafka.Server.1=mybroker1a:9094,mybroker1b:9094
   Kafka.Server.2=mybroker2a:9094
   ```

3. Prior to configuration, clicking the **Topic** dropdown parameter displays the available topics. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive ***Kafka.Topic*** configuration property. For example:

```
Kafka.Topic.1 = sample4
```

4. Prior to configuration, clicking the **KeystoreLocation** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **Kafka.SSLKeyStoreLocation** property. For example:

```
Kafka.SSLKeyStoreLocation.1=[/[path]/key_store.jks].
```

5. Prior to configuration, clicking the **Truststore Location** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **Kafka.SSLTrustStoreLocation** property. For example:

```
Kafka.SSLTrustStoreLocation.1=[/[path]/trust_store.jks].
```

6. SASL / OAuth 2.0 can be configured for STEP with Kafka using the **ExtraDriverOptions** property to authenticate securely via bearer tokens. A business function allows integration with the Kafka Delivery method by handling the OAuth authentication and returning a HashMap containing the Bearer Token and other details.

   Configure the following case-sensitive sharedconfig.properties entries within the value of the ExtraDriverOptions as shown in the example below, which can be copied:

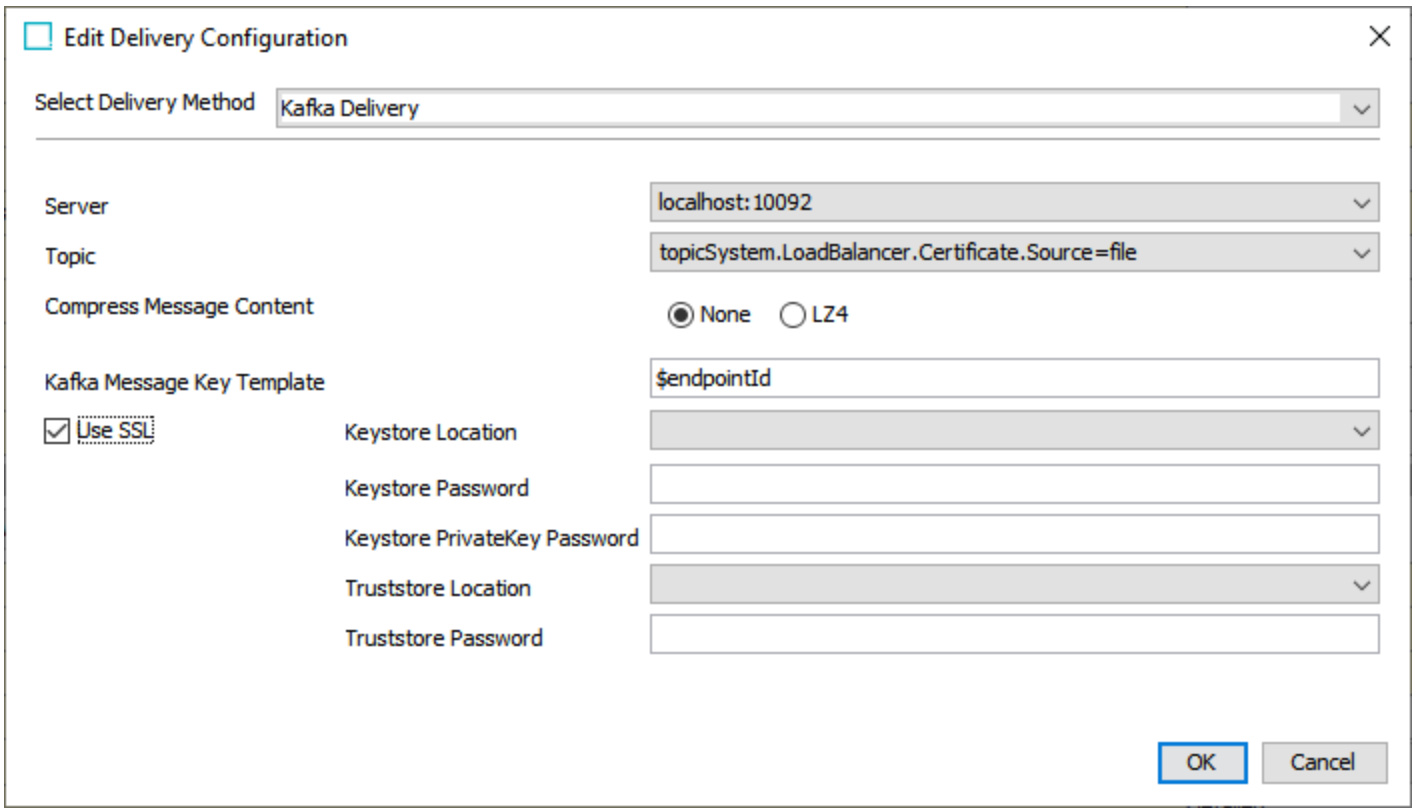
   - `Kafka.Delivery.[OIEP_ID].ExtraDriverOptions` - indicate the OIEP ID.
   - `stibo.authentication.function` - set the JavaScript business function ID. Refer to the Business Function Example for Generating an OAuth Bearer Token section below.
   - `security.protocol` - indicate either SASL_PLAINTEXT (while testing) or SASL_SSL (in production) for the SSL transport layer.

   For example, in the online help version of this topic the property definition includes, ‘MyKafka’ is the ID of the OIEP, ‘HydraAuthFunction’ is the ID of the business function, and ‘SASL_PLAINTEXT’ is the testing value for the security protocol and the rest of the property values are configured as displayed:

Configuration

The Kafka Delivery option is available when configuring an outbound integration endpoint (OIEP).

On the OIEP select 'Edit Delivery,' and then provide the following information:



1. For **Delivery**, choose **Kafka Delivery**.
2. For the **Server**, select the server(s) where the Kafka broker instances used by the endpoint are running.
3. In **Topic**, select the topic used by this endpoint.
4. For **Max Events To Read Per Poll**, enter the maximum number of events to read out of Kafka per invocation.

   Note: Setting this too high can require allocating additional memory in the system.

5. In **Decompress Message Content**, optionally select a decompression option:
   - None - message content will remain compressed.
   - LZ4 - lossless data compression algorithm. Search the web for more information.
6. In **Kafka Message Key Template**, enter the desired key template. Viable templates include: ${endpointId}, ${nodeType}, and ${nodeId}.
7. In **Keystore Location**, if SSL encryption is required, select an SSL encrypted connection to Kafka. Otherwise, leave this parameter blank.
8. In **Keystore Password**, enter the password for keystore if required.
9. In **Keystore PrivateKey Password**, enter the password of the private key in the keystore file, if needed.

10. In **Truststore Location**, if SSL encryption is required, select an SSL encrypted connection to the Kafka. Otherwise, leave this parameter blank.
11. In **Truststore Password**, enter the password for truststore if required.
12. Click the **Next** button to continue with the **IIEP - Configure Endpoint** step, or the **Finish** button to close the wizard.

Business Function Example for Generating an OAuth Bearer Token

In the online help version of this topic the code example includesa framework for developing a business function to be used with Kafka Delivery and Receiver properties when SASL / OAuth 2.0 Bearer Token authentication is configured.

Considerations for Setting Up Kafka Receiver or Delivery

There are a number of considerations before starting the setup for Kafka integration.

- Message size limitation
- Event ordering
- Event message serialization
- Reference handling for dangling references
- SASL authentication, if needed

Each area is explained in further detail below.

When configuring Kafka, STEPXML is recommended for the messaging format. Other formats supported for integration with STEP will also work, but the documentation has been written for STEPXML.

Compatibility

Integration with Kafka for event messaging is supported via the following versions of Apache Kafka:

- 3.5.1
- 3.4x
- 3.2x
- 3.0x
- 2.6x
- 2.5x
- 2.4x

Message Size Limitation

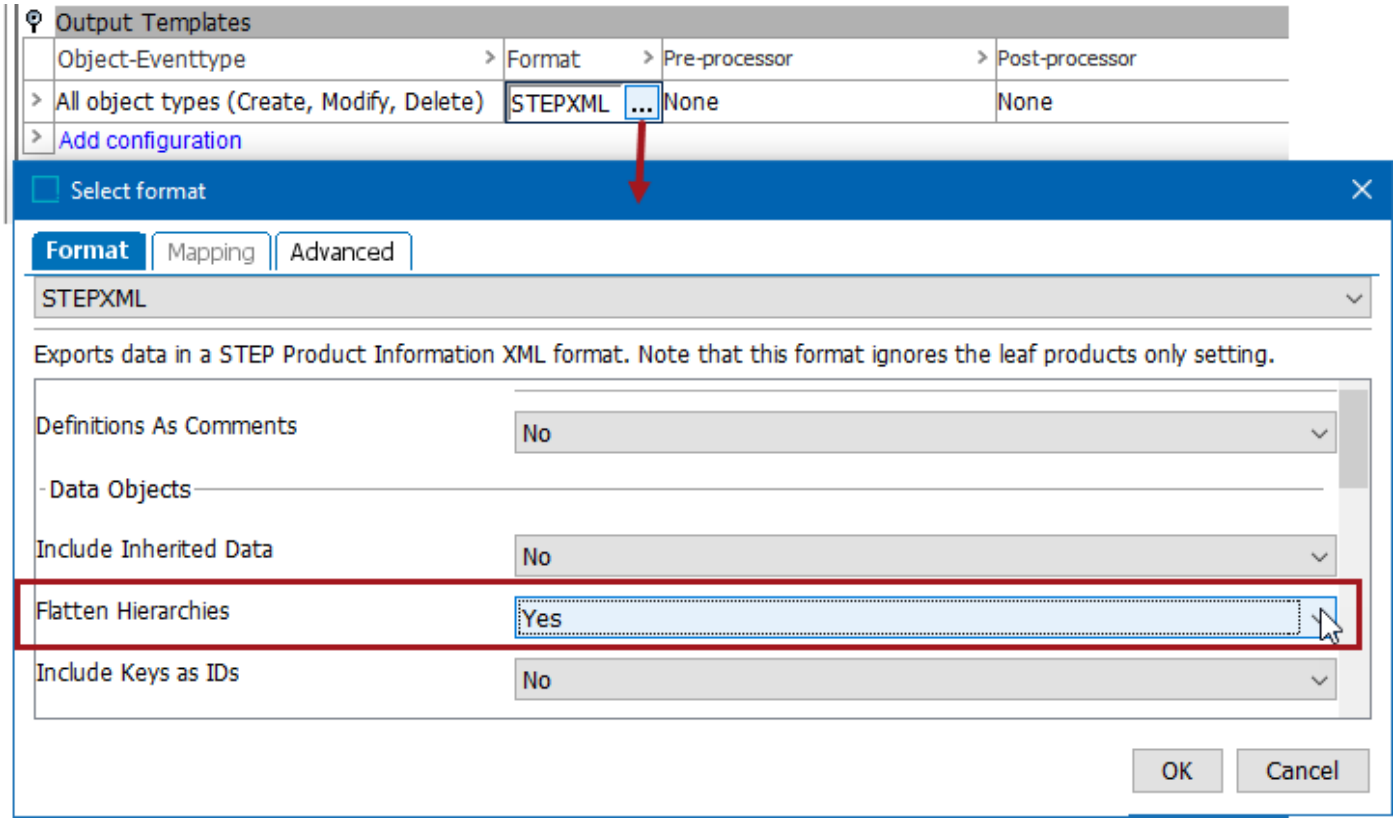
Kafka imposes a default restriction on message sizing to a max of 1 MB. To address this, there are a number of options, with Option 1 being the preferred and recommended method.

Option 1

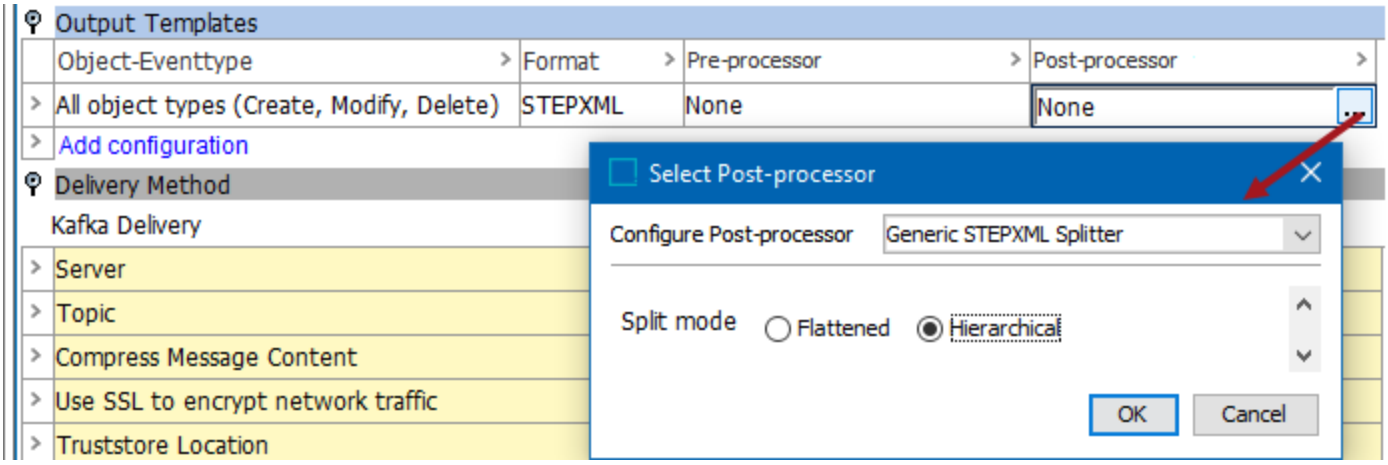
Important: This is the preferred and recommended method.

For Outbound Integration Endpoints: STEPXML format allows using inherited values.

1. On the endpoint, in the Format parameter, select STEPXML and set the 'Flatten Hierarchies' parameter to '**Yes**.'



2. For the Post-processor parameter, select **Generic STEPXML Splitter** and set the 'Split mode' to **Hierarchical**. The Generic STEPXML Splitter splits up STEPXML messages to multiple STEPXML valid fragments containing one single node per STEPXML file.

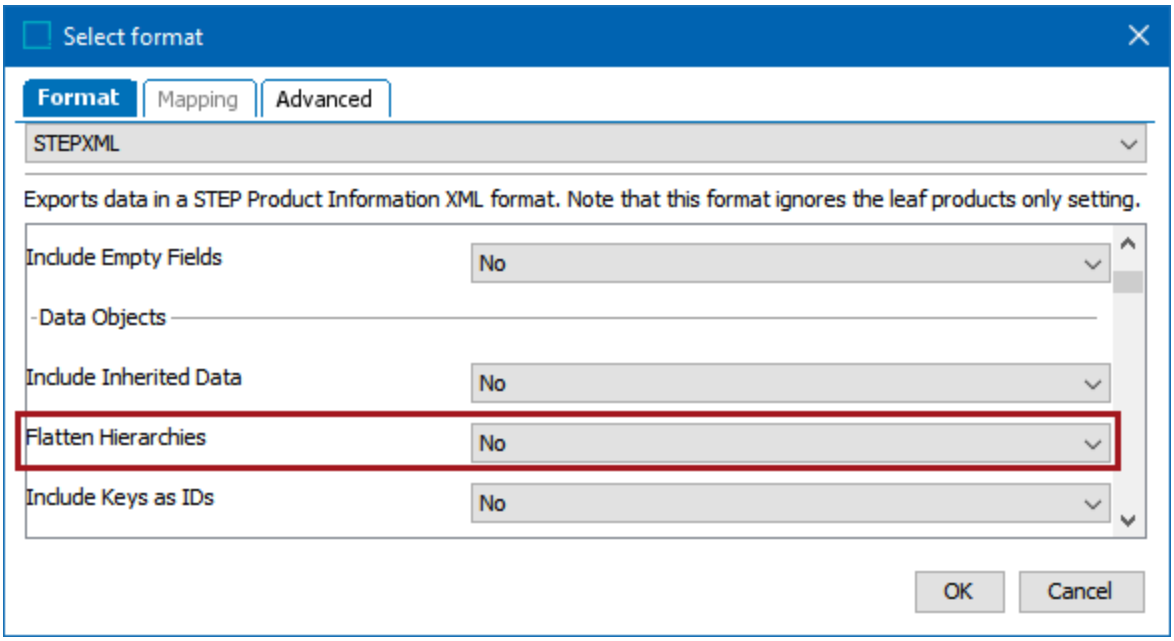


Option 2

Split up STEPXML messages to events for a single node per message and reduce the event XML size.

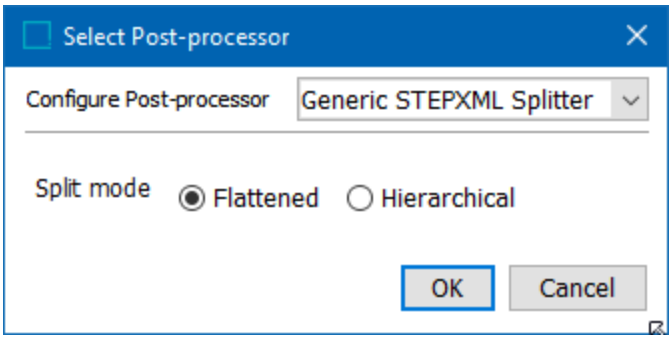
Note: Option 1 is the recommended and preferred method.

1. On the endpoint, in the Format parameter, select STEPXML and set the 'Flatten Hierarchies' parameter to '**No**.' It is not needed for this, as it will be flattened in the post-processor.



2. On the endpoint, select the **Generic STEPXML Splitter** post-processor. The Generic STEPXML Splitter splits up STEPXML messages to multiple STEPXML valid fragments containing one single node per STEPXML file.

3. Use the **Flattened** option to further reduce the size of the individual event messages. Flattened denormalizes any child hierarchies in STEPXML to be written out as a single root hierarchy.



Note: If using inherited values in the export, it is recommended to flatten the values in STEPXML format as demonstrated in Option 1 above.

For more information on post-processors, refer to the **Configure the Pre-processor and Post-processor** section of the following topics:

- **OIEP - Select Objects - Output Templates Flipper**
- **OIEP - Event-Based - Output Templates Flipper**

Option 3

Configure a compression of the event content.

While Kafka does support compression out of the box, it does not help guard against the 1 MB limit, as will be measured on uncompressed message. This option is offered as a method to further address the max message size. LZ4 is currently the only supported compression format. If configured to be used on the integration endpoint when integrating with custom developed Kafka, any producers or consumers must be able to handle compression / decompression of LZ4 event content.

Note: For performance reasons, it is not recommend to use Kafka compression if configuring this explicitly on the endpoint, as this will have limited effect in terms of size.

Important: Ensure all events are processed before changing compression settings for an existing endpoint. Existing events on the Kafka topic are not changed as a result of changing the compression setting.

Topic Partitions

STEP can only read from single-partition topics (topics created with '--partitions 1') but is capable of delivering to multi-partition topics. To have messages distributed across multiple partitions, they must have different keys. By default, the key for all messages published from STEP is the ID of the OIEP responsible. This behavior can be changed by entering a new value for the 'Kafka Message Key Template' field in the Kafka Delivery Method configuration.

The following variables can be used to produce message keys:

- **$endpointId** - ID of the OIEP publishing the message.
- **$nodeType** - The type of node being published (e.g., product, classification, etc).
- **$nodeId** - ID of the STEP node being published.

Note: Because STEP IDs are not unique, $nodeType can be combined with $nodeId to create a unique identifier.

For more information, refer to the **Kafka Delivery Method** topic in the **Creating an Outbound Integration Endpoint** section of the **Outbound Integration Endpoints** documentation.

Important: In order to publish to multi-partition topics and use the $nodeType and $nodeId variables, ensure that messages only contain data for a single STEP object. If exporting STEPXML the Generic STEPXML Splitter must be used. If using the Generic JSON format option, the OIEP must be configured to run with a batch size of '1.'

Event Message Serialization

This is relevant if implementing your own Kafka producers / consumers for integration with STEP.

1. A event produced by STEP has the outbound integration endpoint ID set as a key. When implementing your own producer, this must be the same key set per producer because the key is used for partitioning purposes.
2. The key is passed in as a string, which is serialized with the standard Kafka: org.apache.kafka.common.serialization.StringSerializer.
3. The message content itself is a bytearray that is serialized with: org.apache.kafka.common.serialization.ByteArraySerializer.
4. On the consumer side, deserialize with the opposing deserializer for key and message content. Implementations are namely: org.apache.kafka.common.serialization.StringDeserializer and org.apache.kafka.common.serialization.ByteArrayDeserializer

Reference Handling for Dangling References

The Kafka connector and the Generic STEPXML Splitter occasionally have known issues when processing references where the referred to node has not yet been created in the target system. This creates a 'dangling reference.'

Often this dangling reference can be solved by being consistent with data handling, however with an inbound receiver that is having issues with referential integrity, the following can help.

Note: This is relevant only when dealing with referential integrity issues on an inbound receiver.

Example 1

In the following example, two outbound events contain XML that is generated in the events queue.

Event 1: circularref1

```
<?xml version="1.0" encoding="UTF-8"?>
<STEP-ProductInformation ExportTime="2020-09-28 14:38:18" ExportContext="GL"
ContextID="GL" WorkspaceID="Approved" UseContextLocale="false">
<Product ID="circularref1" UserTypeID="Product" ParentID="Product hierarchy root"
Changed="true">
<Name Changed="true">circularref1</Name>
```

```
<ProductCrossReference ProductID="circularref2" Type="prodToProd" Changed="true">
</ProductCrossReference>
</Product>
</STEP-ProductInformation>
```

Event 2: circularref2

```
<?xml version="1.0" encoding="UTF-8"?>
<STEP-ProductInformation ExportTime="2020-09-28 14:38:18" ExportContext="GL"
ContextID="GL" WorkspaceID="Approved" UseContextLocale="false">
<Product ID="circularref2" UserTypeID="Product" ParentID="Product hierarchy root"
Changed="true">
<Name Changed="true">circularref2</Name>
<ProductCrossReference ProductID="circularref1" Type="prodToProd" Changed="true">
</ProductCrossReference>
</Product>
</STEP-ProductInformation>
```

For 'circularref1' and ' circularref2' to both be successfully approved upon import, partial approval is given to 'circularref1' without the reference. Then 'circularref2' is fully approved with its reference to 'circularref1'. A second approval is then given to 'circularref1' so that the reference from 'circularrelf2' can be added.

In this case, the two XMLevents would look like the following, and not cause issues with referential integrity:

```
<?xml version="1.0" encoding="UTF-8"?>
<STEP-ProductInformation ExportTime="2020-09-28 14:38:18" ExportContext="GL"
ContextID="GL" WorkspaceID="Approved" UseContextLocale="false">
<Products>
<Product ID="circularref2" UserTypeID="Product" ParentID="Product hierarchy root"
Changed="true" Referenced="true">
<ProductCrossReference ProductID="circularref1" Type="prodToProd" Changed="true">
<Product ID="circularref1" UserTypeID="Product" ParentID="Product hierarchy root"
Changed="true"></Product>
</ProductCrossReference>
</Product>
</Products>
</STEP-ProductInformation>
```

Example 2

Consider that both 'circularref1' and 'circularref2' are new products to be imported into STEP or a downstream system.

Product 1: circularref1

```
<?xml version="1.0" encoding="UTF-8"?>
<STEP-ProductInformation ExportTime="2020-09-28 14:38:18" ExportContext="GL"
ContextID="GL" WorkspaceID="Approved" UseContextLocale="false">
<Product ID="circularref1" UserTypeID="Product" ParentID="Product hierarchy root"
Changed="true">
```

```
<Name Changed="true">circularref1</Name>
<ProductCrossReference ProductID="circularref2" Type="prodToProd" Changed="true">
</ProductCrossReference>
</Product>
</STEP-ProductInformation>
```

Product 2: circularref2

```
<?xml version="1.0" encoding="UTF-8"?>
<STEP-ProductInformation ExportTime="2020-09-28 14:38:18" ExportContext="GL"
ContextID="GL" WorkspaceID="Approved" UseContextLocale="false">
<Product ID="circularref2" UserTypeID="Product" ParentID="Product hierarchy root"
Changed="true">
<Name Changed="true">circularref2</Name>
<ProductCrossReference ProductID="circularref1" Type="prodToProd" Changed="true">
</ProductCrossReference>
</Product>
</STEP-ProductInformation>
```

Notice that 'circularref1' references 'circularref2' and 'circularref2' references 'circularref1.' When using the Generic STEPXML Splitter during an inbound integration endpoint (IIEP), it would fail because the product 'circularref2' would not exist yet on the system for circularref1' to reference. The referential integrity is not being kept.

To fix this problem you can:

1. Fix this manually by creating 'circularref2' in the system, and then rerun the IIEP.
2. Remove the events from the Kafka topic using **kafka-delete-records.sh** or a similar command
3. Instead of using the Generic STEPXML Splitter, limit the batch size on the endpoint to be able to deliver more nodes in one XML event. Use LZ4 compression to reduce the message size as much as possible. In most cases, the 1 MB limit affords a larger number of events to be processed, but this is a system-specific consideration. If needed, the 1 MB limit can be raised on the Kafka cluster to allow for larger batches to be processed. For any downstream systems that may be receiving these events, make them aware of these changes. A form of eventual consistency with regards to references may need to be implemented.
4. Use advanced STEPXML and then employ embedding for the references. This option could produce more verbose XML and could exceed the 1 MB message size limit if used on a system with many references. Make sure to reconfigure the Generic STEPXML Splitter to 'Hierarchical' rather than 'Flattened' for this option. An example template that embeds the references into the event message is the following:

```
<STEP-ProductInformation ResolveInlineRefs="true" FollowOverrideSubProducts="true">
<UserTypes ExportSize="Selected"/>
<EdgeTypes/>
<CrossReferenceTypes ExportSize="Selected"/>
<Products ExportSize="Minimum">
<Product>
<Name/>
<ProductCrossReference Type="prodToProd">
```

```
<Product Referenced="true" Embedded="true">
<empty/>
</Product>
</ProductCrossReference>
</Product>
</Products>
</STEP-ProductInformation>
```

Kafka Topic Offset

To achieve transactional integrity in STEP its important to note that the Kafka topic offset is managed in STEP rather than stored in Apache Zookeeper. This way if some message processing fails in the importer, it is possible to rollback the transaction alongside the topic offset increment. In doing this, it achieves transactional integrity when reading from the Kafka topic. The offset is associated with the inbound integration endpoint (IIEP) itself. For this reason, switching an IIEP to read from another topic is not allowed, as this would inherit the offset. Instead, a new IIEP should be created from scratch and configured to read from the other Kafka topic.

In workbench, if the topic offset of an IIEP is larger than 0, topic changes are not allowed. The topic in the configuration wizard is rendered as a read-only option.

Note: If you export / import inbound integration endpoints between systems, the imported endpoint will always start from offset 0. If you restore a STEP database backup the offset for this will also be restored.

Static Membership

To avoid issues with Kafka rebalance, and to generally make reconnecting an endpoint to Kafka faster, a static membership is employed. As a result, the connection info for the IIEP on Kafka is cached for a default of up to 30 minutes. If the IIEP is scheduled to run before the 30 minutes elapse, the reconnects are a bit faster. For this reason the session timeout in the Kafka driver is set to be 30 minutes which is the default maximum on a Kafka installation. If this is configured differently, set this timeout manually from the default setting, which is 1800000 for 30 minutes in milliseconds. To set manually, navigate to the **Kafka.Receiver.<ENDPOINT-ID>.SessionTimoutInMs** configuration property.

SASL Authentication

Support for Simple Authentication and Security Layer (SASL) authentication (both PLAIN and SCRAM) is supported for the Kafka receiver and delivery plugins. Using SASL gives you more data security options and allows for alternatives to the other array of Kafka connector authentication functionality support, which includes support for AWS MSK, Heroku, and Aiven (with TLS Client Certificate Authentication).

The properties need to be added to the sharedconfig.properties file. Below is an example config of PLAIN username/password authentication:

```
Kafka.Receiver.YOURENDPOINTID.ExtraDriverOptions=sasl.mechanism=PLAIN,security.proto
col=SASL_
PLAINTEXT,sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule
required username="admin" password="admin-secret";
```

For SASL_SSL with PLAIN username/password authentication, the Keystore configuration in the SSL part of the Kafka Receiver / Delivery plugin can be omitted. If there is no requirement that the Kafka server has to trust our SSL certificate, then none is needed. A Truststore Location / Password is required to indicate that your system trusts the Kafka Servers Certificate.

Below is a sample config for SCRAM authentication:

```
Kafka.Receiver.YOURENDPOINTID.ExtraDriverOptions=sasl.mechanism=SCRAM-SHA-
256,security.protocol=SASL_
SSL,sasl.jaas.config=org.apache.kafka.common.security.scram.ScramLoginModule
required username="admin" password="admin-secret";
```

All of the configuration options are taken from confluent.io documentation on how to configure SASL authentication found in this link: https://docs.confluent.io/platform/current/kafka/overview-authentication-methods.html.

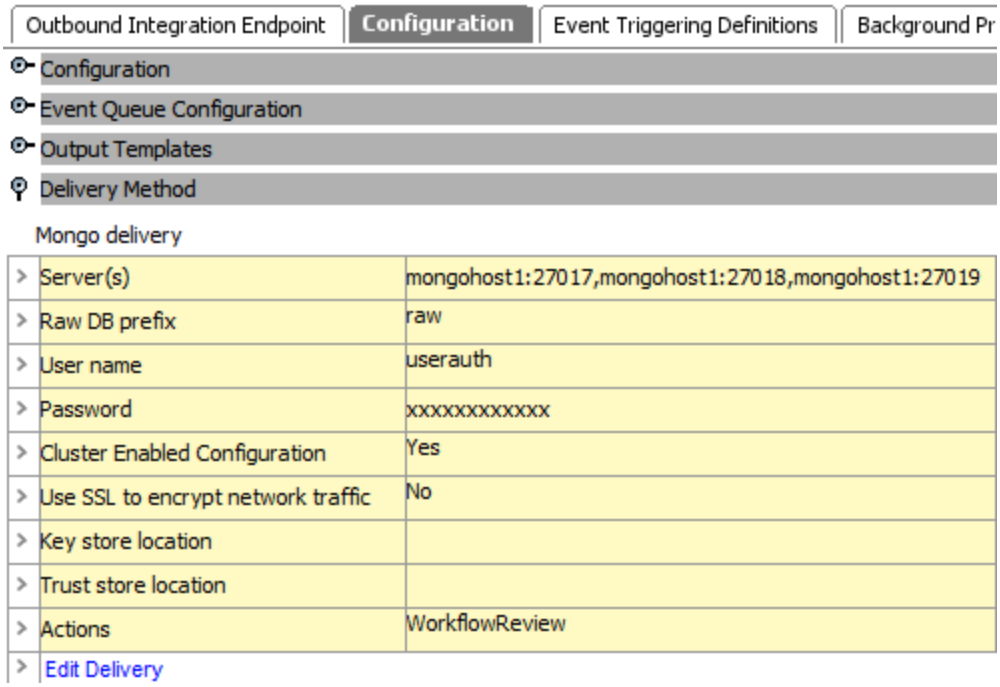
Mongo Delivery Method

Prior to selecting the Mongo Delivery Method, ensure your setup meets the prerequisites explained in **Prerequisites for Configuring the MongoDB Adapter** documentation. This delivery option is only available in OIEPs.

For more information, refer to https://docs.mongodb.com/manual/ or the **MongoDB Adapter Setup Quick Guides** documentation.

For more information on the Mongo Delivery Method functionality, refer to the **Mongo Delivery Method Elements** documentation.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



Prerequisites

Changes to the properties file, outlined below, are implemented when the server is restarted.

1. If your system displays radio button options for the server selection, refer to the **Legacy Configuration** section below for more details.

2. Prior to configuration, clicking the **Server(s)** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive dynamic **MongoDB.Server.Configuration.[n]** property, where the [n] is used for differentiation. The configuration entries must be unique (MongoDB.Server.Configuration.1, MongoDB.Server.Configuration.2, MongoDB.Server.Configuration.3, etc.). When duplicates exist, only the last value is displayed in the dialog. Each host is separated from the respective port using a colon (:), and additional servers are added using a comma-separated list. For example:

   ```
   MongoDB.Server.Configuration.1=host4:40004
   ```

   indicates an independent MongoDB instance. A second and third configuration could be supplied for servers being used in a cluster using the format:

   ```
   MongoDB.Server.Configuration.2=host1:10001,host2:20002,host3:30003
   MongoDB.Server.Configuration.3=host4:40004,host5:50005,host6:60006,host7:70007
   ```

3. The **Key store location** is only required if the Mongo DB requires the clients to use SSL to encrypt connections.

   The password for the key store is configured in sharedconfig.properties on the STEP application server, using the case-sensitive **MongoDB.SSLKeyStorePassword** property. For example:

```
MongoDB.SSLKeyStorePassword = keystore_password
```

4. In **Trust store location**, is only required if the Mongo DB requires the clients to use SSL to encrypt connections.

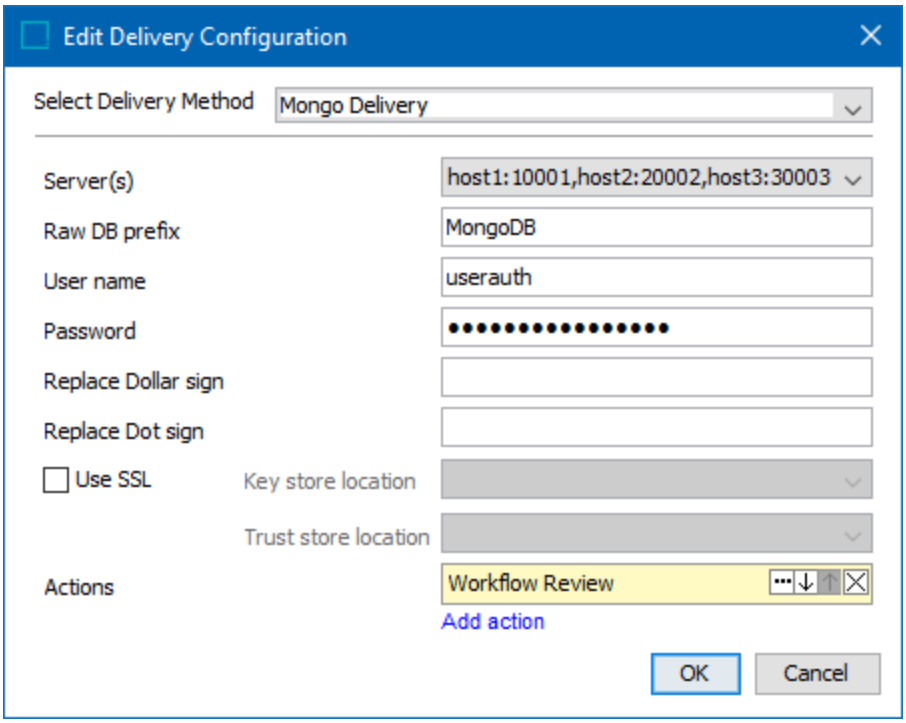
   The password for the key store is configured in sharedconfig.properties on the STEP application server, using the case-sensitive **MongoDB.SSLTrustStorePassword** property. For example:

   ```
   MongoDB.SSLTrustStorePassword = truststore_password
   ```

Configuration

For information on a parameter, hover over the parameter field to display help text.

1. In **Select Delivery Method**, choose **Mongo Delivery**.



Note: For MongoDB 4.2 and 4.4, choose **Mongo Delivery (from release 4.2)** from the **Select Delivery Method** dropdown menu.

2. In **Server(s)** dropdown, select the desired MongoDB server configuration, either an independent MongoDB instance or a MongoDB cluster setup.

3. In **Raw DB prefix**, if required, enter a prefix for all raw databases the adapter creates. The output is [prefix] + [context], resulting in a database name of MongoDBContext1 using the data in the image above, and exporting for Context1. Optional.

4. In **User name**, if MongoDB is configured to use authentication, enter the MongoDB user. Optional.
5. In **Password**, if MongoDB is configured to use authentication, enter the MongoDB password. Optional.
6. **Replace Dollar sign** performs replacements for JSON document key names. Text entered replaces any dollar sign ($) during delivery. If blank, all dollar signs in key names are replaced automatically by the JSON converter with Unicode 'uFF04.' This allows overriding the default Unicode replacement with the specified character(s), for example, an underscore (_).
7. **Replace Dot sign** performs replacements for JSON document key names. Text entered replaces any dot sign (.) during delivery. If blank, all periods in key names are replaced automatically by the JSON converter with Unicode 'uFF0E.' This allows overriding the default Unicode replacement with the specified character (s), for example, an underscore (_).
8. Check **Use SSL** if MongoDB requires the client to use SSL encryption. Optional. If no further parameters are configured, it is assumed that the MongoDB uses a SSL certificate that is issued by a trusted certificate authority (like Verisign or Thawte). If this is not the case, or if more security of the connection is requested, you can configure an SSL key store and an SSL trust store.
    - In **Key store location**, enter a key store location, for example, file://[path]/key_store.jks. The Key store holds the STEP Mongo Adapter private key and SSL certificate in a keystore file as generated by the Java utility 'keytool' (in jks format). The password for the key store is configured as defined in the **Prerequisites** section.
    - In **Trust store location**, enter a trust store location, for example, file://[path]/trust_store.jks. The trust store holds the Mongo DB public key and certificate in a keystore file as generated by the Java utility 'keytool' (in jks format). The trust store configuration is only needed if the certificate is self-signed and not issued by a trusted certificate authority. The password for the key store is configured as defined in the **Prerequisites** section.
9. In **Actions**, click the **Add action** link to enter one or more STEP business actions used to create and populate aggregated collections. These actions are run after the raw collections have been populated.
10. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

Delivering Delete Events

The Mongo delivery method requires the following specific OIEP settings to successfully deliver delete events:

1. The OIEP is event-based, verified by viewing that the Configuration tab includes the Event Queue Configuration flipper. For more information, refer to the **OIEP - Event-Based - Event Queue Configuration Flipper** topic.
2. In the Event Queue Configuration flipper, the Queue Status parameter is set to Read Events.
3. In the Output Templates flipper, the Object-Eventtype column includes 'Delete' as an event type. For more information, refer to the **OIEP - Event-Based - Output Templates Flipper** topic.
4. Also in the Output Templates flipper:

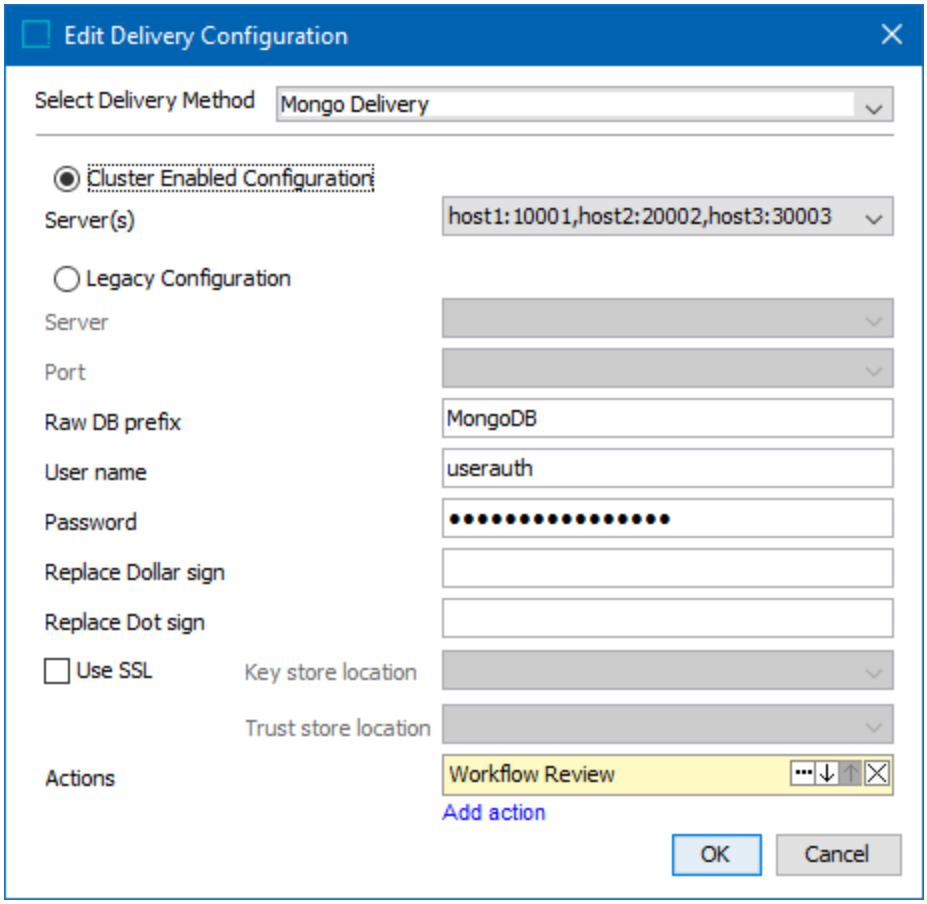
- The **Format** column displays 'Advanced STEPXML'.
- The **Template** includes the appropriate <DeleteProducts/> tags.

  Advanced STEPXML is the only format that allows delete events to be included in the XML. For more information, refer to the **Event-Based OIEP Triggered by Deleting Products, Classifications, and Assets** section of the **Delete Objects in STEPXML** topic.

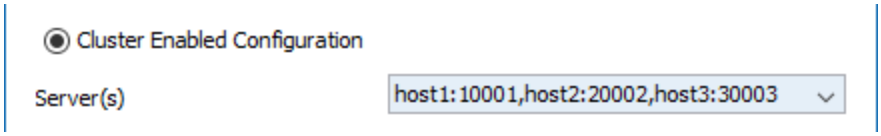
Legacy Configuration

When the sharedconfig.properties file includes the MongoDB.Server property and the MongoDB.Port property, the dialog displays both a cluster option and a legacy option. Typically, this happens when a STEP system had a MongoDB host / port configured prior to the update that introduced the clustered configuration functionality.

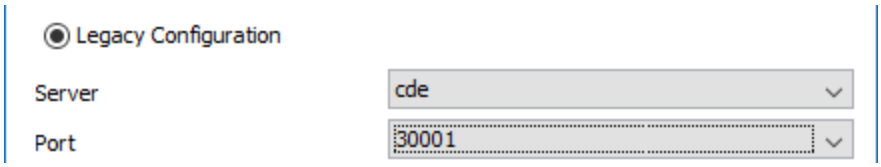
Note: Upgrading the sharedconfig.properties file to modify all configurations using the legacy properties to the cluster enabled property, and then removing the legacy properties completely, results in the radio button options being removed from the dialog, as shown in the image at the top of this topic. Additionally, the cluster option and the legacy option are not included in the **Mongo Delivery (from release 4.2)** delivery method.



When the **Cluster Enabled Configuration** radio button is selected, the **Server(s)** dropdown displays entries for the dynamic MongoDB.Server.Configuration.[n] property.



When the **Legacy Configuration** radio button is selected, the **Server** and **Port** dropdowns display entries for the MongoDB.Server and MongoDB.Port properties.



Multiple independent servers or ports can be listed using the following format:

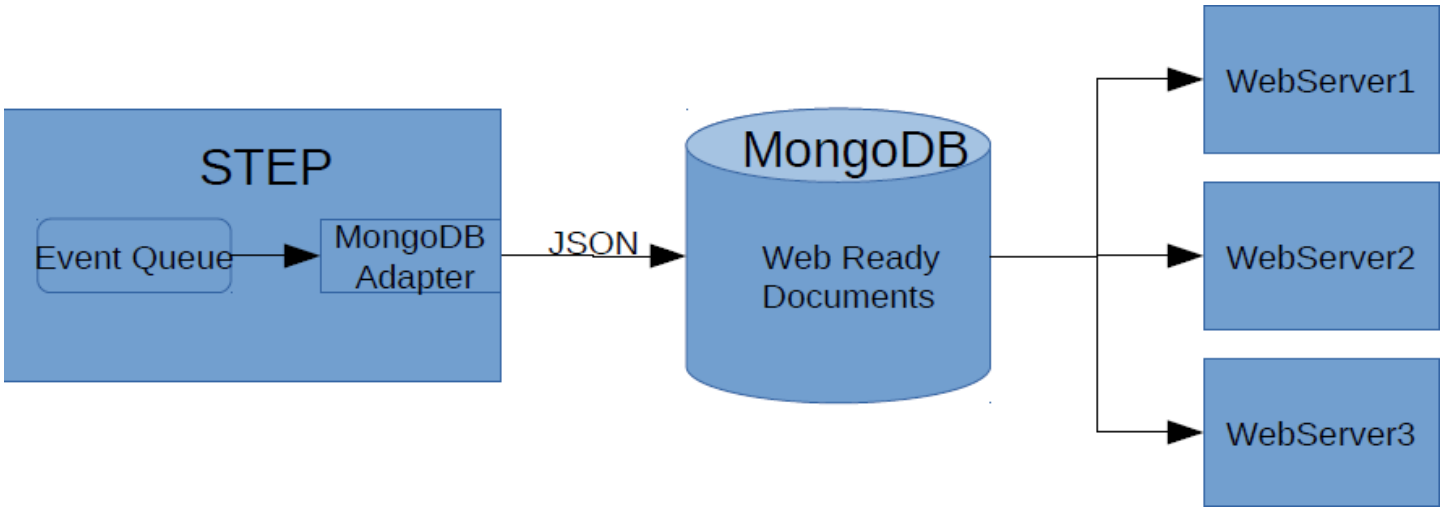
```
MongoDB.Server=1=vps124887.vps.ovh.ca,2=cde,3=rty
MongoDB.Port=1=30001,2=30002,3=30003
```

Mongo Delivery Method Elements

The STEP Mongo Adapter is an outbound integration endpoint delivery option that receives data from a STEP event queue and loads it into a MongoDB database. The MongoDB is often used for website back-end, reporting, and high performance feeds to other back-end systems.

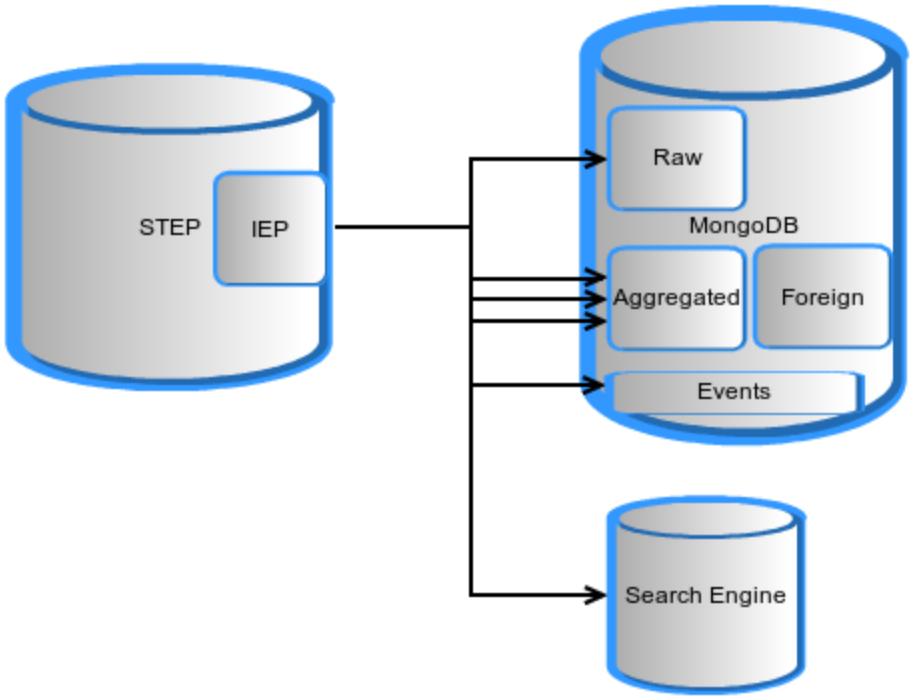
The Mongo adapter converts STEPXML to JSON and keeps a set of 'raw' collections in MongoDB that are in sync with STEP. Additionally, it allows you to add JavaScript triggers to maintain aggregated collections.

The following illustration shows how the Mongo Adapter can be used to feed a Mongo database that is used as the back-end to a website:



By keeping collections of web-ready JSON documents available in MongoDB, the JSON can be passed directly to the browser's JavaScript. This results in the web server having little processing to do, which means that it needs to pass the right collection result directly back to the browser.

The following drawing illustrates the collections involved:



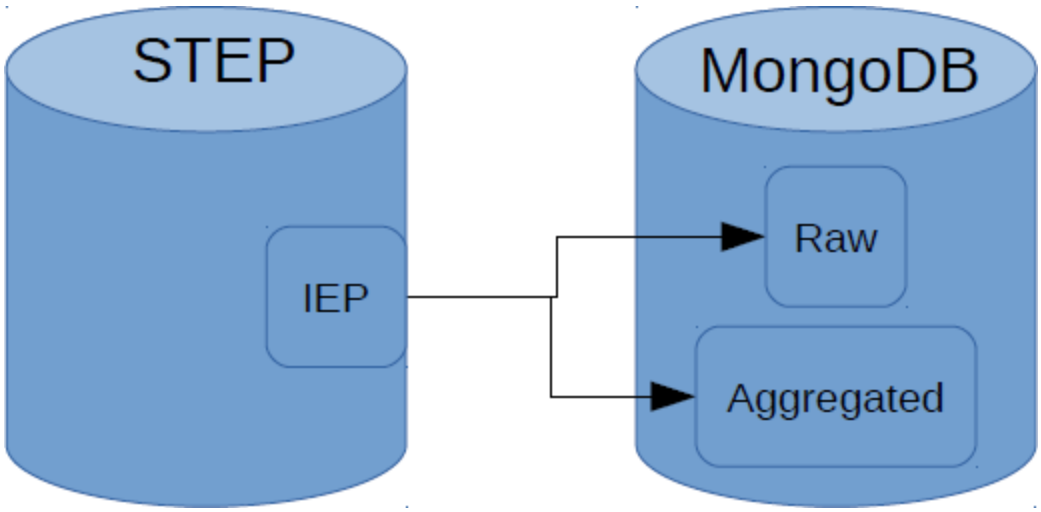
The STEP Mongo Adapter ensures that the raw collections are a one-to-one mirror of data in STEP. The adapter does not embed nodes found through references or child associations. These nodes are stored separately in the raw collections.

It is possible to configure the adapter so that it creates one or more customized, aggregated MongoDBs (databases or collections) where the documents can be tailored to meet the exact usage requirements, like reporting, analysis, and so on.

The JavaScript triggers allow maintaining aggregated collections, for example:

- List of children, including title and other data necessary in a tree-like view.
- Information from the target end of references that is needed in a given view.
- Mapping to a JSON structure that is closer to the website data model.

The following image shows a MongoDB database containing a raw database that is maintained directly by the STEP Mongo Adapter and an aggregated database maintained by JavaScript triggers.



A JavaScript trigger is executed under certain conditions, for example, only for a specific object type. This reduces the amount of checking necessary in the JavaScript code. Once the raw collection is updated and the JavaScript triggers (if configured) have completed, an event is inserted into the events collection. Also, an update is sent to the search engine.

Note: Aggregated collections and raw collections do not need to be in the same MongoDB instance. However, if required, different collections in the aggregate can be combined into their own MongoDB instance.

Default Databases and Collections

The STEP Mongo Adapter maintains a number of MongoDBs, referred to as 'raw databases', and a number of collections in the raw databases, referred to as 'raw collections.' A raw database is created for each context that is exported and the database is named after the context. For example, if the two contexts EN and FR are exported, two raw databases are created, one named EN and one named FR.

It is possible to configure a string that prepend the raw database name when the STEP Mongo Adapter is configured. When data is exported from multiple contexts, a JSON document is created per context, per exported object. Continuing the previous example, two JSON documents are created for each object: one for the object in the EN context and one for the object in the FR context.

A raw collection is only created when the particular STEP data type is exported. So, if no assets are exported from STEP, no asset collection is created. Each raw database contains all or some of the following raw collections:

Raw Collection	Purpose
product	Exported products

Raw Collection	Purpose
asset	Exported assets
classification	Exported classifications
entity	Exported entities
attribute	Exported attribute definitions
data containers	Exported data container definitions
referenceType	Exported reference types
listofvalues	Exported list of values (LOVs)
unit	Exported unit definitions

Inherited values, calculated values, and unit names on values are handled already in the raw collections. Since changes to these affect replicated information, STEP ensures that appropriate updates are made to sync the replicated information.

STEP JSON and Mongo JSON

The STEP Mongo Adapter converts STEPXML input to STEP JSON (JavaScript Object Notation) documents, and then stores the JSON documents in the MongoDB. The STEP JSON schema can be downloaded from your STEP server at: http:///*[enter step-server]*/files/StepSchema.json

Formally, MongoDB documents are BSON documents, which is a binary representation of a JSON document. For more information, refer to the MongoDB documentation on the web.

For an expanded example of the basic conversion using STEP Mongo Adapter, refer to the **Mongo Delivery Method Conversion Example** documentation.

JavaScript Triggers

STEP business action triggers with JavaScript populate the aggregated MongoDB databases and collections. You can configure one or more business action triggers for the STEP Mongo Adapter. The business action triggers are fired after the STEP Mongo Adapter has populated the raw collections.

During configuration of the STEP Mongo Adapter, the configuration wizard allows you to select multiple JavaScript to run under different conditions by adding preconditions to the business actions. A precondition is, for example, a JavaScript trigger that is only fired for updates to a given type of objects. For more information, refer to the **Applies If Tab** section of the **Editing a Business Rule or Function** documentation.

Due to the nature of inheritance and calculated attributes, the trigger may be called even if there are no changes to the underlying raw objects.

The business action triggers are executed once per JSON document sent to the MongoDB database. If two objects are exported from STEP, two JSON documents are sent to the MongoDB database and the business action triggers are executed twice. If the objects are exported in multiple contexts, one JSON document is generated per context, per object. This means that if two products in two contexts are exported, the business action triggers are executed four times.

Business action triggers are executed on the STEP server. Business actions triggers need access to the MongoDB database to fetch data from the raw collections and write or read data to / from the aggregated collections. For the best performance, network latency between the STEP server and the MongoDB server should be minimal.

Important: Place aggregated collections in their own database so that the raw database contains only collections.

Bound Variables

The JavaScript trigger have accesses to information in the Mongo Adapter environment by binding script variables when the JavaScript trigger is defined.

Type	Gives access to
MongoDBContext	• The current STEP context. • The name of the raw MongoDB database. • The MongoDB database via the com.mongodb.Mongo object. For more information, refer to: http://api.mongodb.org/java/current/com/mongodb/Mongo.html. The actual Mongo object that is passed can be a wrapper, but it adheres to the interface
JSON context	The JSON object with the data that has been persisted in MongoDB.

For more information about executing JavaScript in business rules, refer to the **Business Action: Execute JavaScript** section of the **Business Rules** documentation.

Example JavaScript to Maintain a Collection in a Classification

The following code-snippet illustrates an example JavaScript that maintains a collection containing the list of products (ID and Name) in a classification.

The structure of the collection is:

```
{_id: "<classification-id>", products : { { id: "<product-id1>", name: "<product-name1>"}, { id: "<product-id2", name: "<product-name2>"}, ... }
```

```
function getClassificationIDs(classrefs) {
    var result = new Array();
    if (classrefs != null) {
        for (i = 0; i < classrefs.targets.size(); ++i) {
            result.push(new String(classrefs.targets[i].targetID));
        }
    }
    return result;
}

var db = mongoContext.getMongo().getDB('extra');
var collection = db.getCollection('classificationproducts');

var classrefs = mongoData.references ? mongoData.references["Web Classifications"] :
null;

var classificationIDs = getClassificationIDs(classrefs);

var deletequery = {"products.id" : mongoData._id, "_id" : { $nin :
getClassificationIDs(classrefs) }};
var adelete = { $pull : { products : { id : mongoData._id } } };

collection.update(deletequery, adelete, false, true);

var query = {"products.id" : mongoData._id };
var name = mongoData.name;
var update = { $set : { "products.$.name" : name } };

collection.update(query, update, false, true);

classificationIDs.map(function(item) {
    var query = { _id : item };
    var update = { $addToSet : { products : { id : mongoData._id, name : name } } };
    collection.update(query, update, true, false);
});
```

Prerequisites for Configuring the MongoDB Adapter

The following are prerequisites for setting up the MongoDB adapter:

1. A MongoDB instance with suitable amounts of disk space.
2. The STEP MongoDB Adapter uses a network port to access the MongoDB database on the server where the MongoDB is running. The network port must be open and accessible from the STEP servers.
3. Update sharedconfig.properties or config.properties with the **name of the MongoDB server**, using the MongoDB.Server configuration property and a list of number-to-value pairs, separated by commas, as follows:

   ```
   MongoDB.Server = 1=MongoDBServer1, 2=MongoDBServer2, 3=MongoDBServer3
   ```

4. Update sharedconfig.properties or config.properties with the **MongoDB port** using the MongoDB.Port configuration property, and a list of number-to-value pairs, separated by commas, as follows:

   ```
   MongoDB.Port = 1=10001, 2=2002, 3=3003
   ```

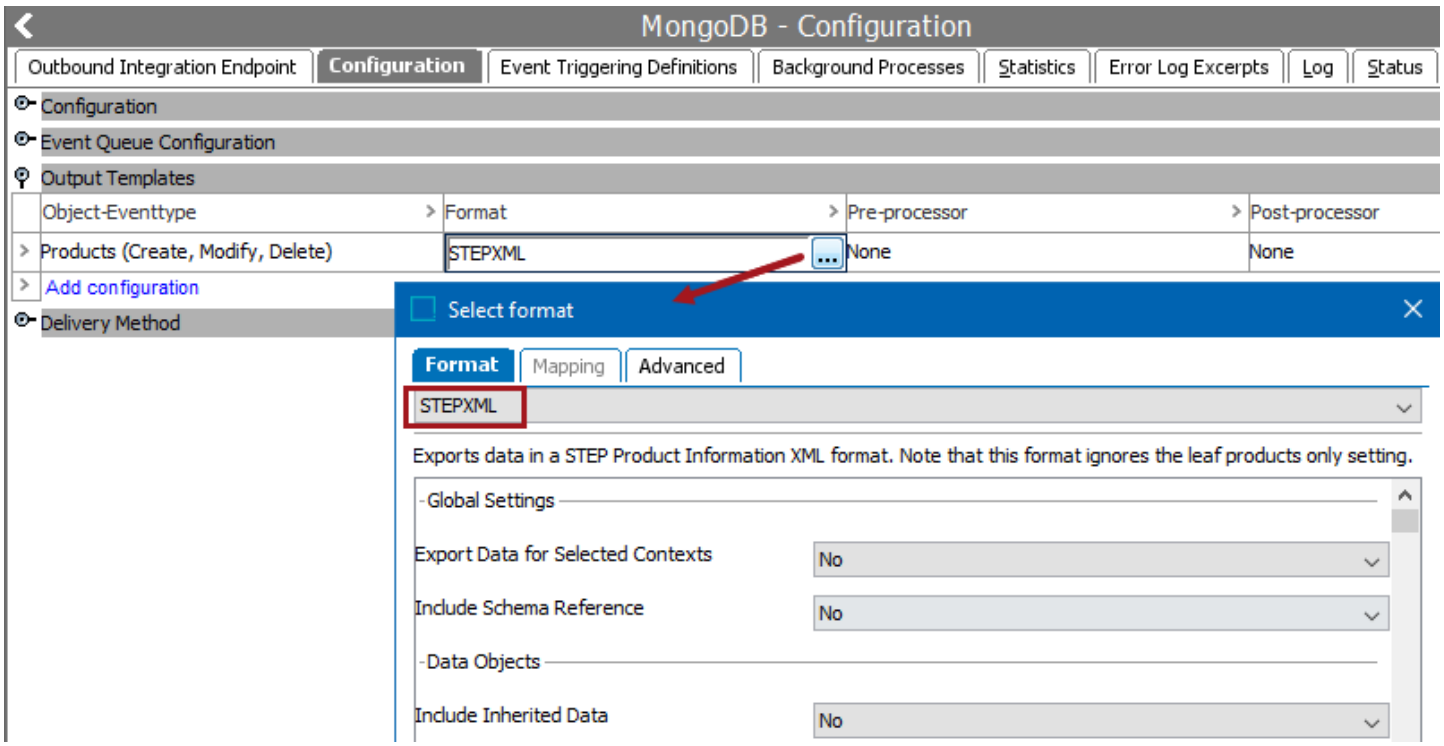
   For example, to configure the adapter to use a MongoDB instance running on either 'Server1' or 'Server2' using port '27712' on both servers, the properties would be:

   ```
   MongoDB.Server = 1=Server1, 2=Server2
   MongoDB.Port = 1=27712
   ```

Configure the OIEP

The STEP MongoDB Adapter is run within an outbound integration endpoint (OIEP) and is configured as the delivery method for the endpoint.

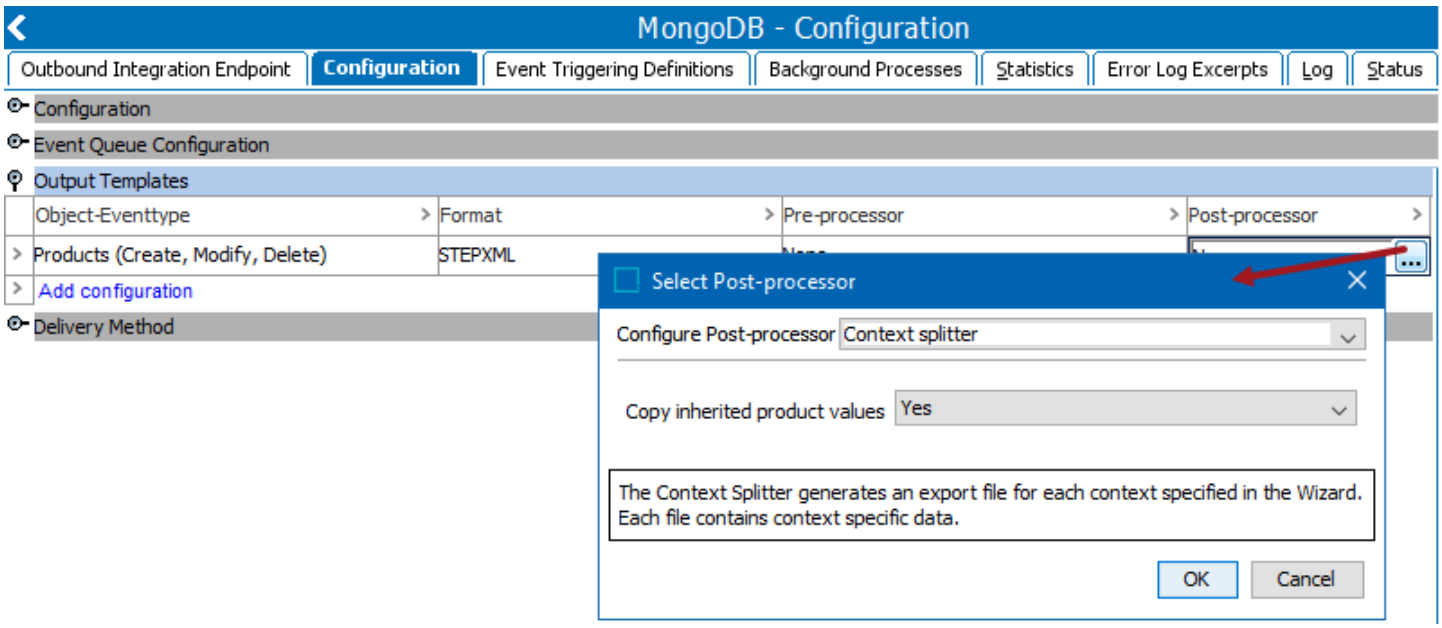
1. In System Setup, create an OIEP, selecting STEP Exporter as the Process Engine. For details on creating a new OIEP, refer to the **Creating an Event-Based Outbound Integration Endpoint** topic.
2. For the new OIEP, open Configuration > Output Templates > Add configuration > Format > click the ellipsis button (...).



3. On the Select Format dialog, leave objects as default values except for the following parameters:
   - Select the **STEPXML** export format in the dropdown at the top of the dialog
   - Data Objects > **Flatten Hierarchies** = Yes
   - Data Objects > **Include Entities** = Selected
   - Data Objects > **Include Data Containers** = Yes
   - Data Objects > **Include Products** = Selected
   - Data Objects > **Include Classifications** = Selected
   - Data Objects > **Include Assets** = Selected
   - Configuration > **Include Attribute Groups** = Minimum
   - Configuration > **Include Data Container Definitions** = Minimum
   - Configuration > **Include Link, Reference and Object Types** = Minimum
   - Configuration > **Include List of Values** = Minimum
   - Configuration > **Include Units** = Minimum

   For more information on these STEPXML outbound parameters, refer to the **STEPXML Outbound Parameters** topic of the **Data Exchange** documentation.

4. In the Post-processor field, click the ellipsis button (...) and select **Context splitter**, then click OK.



5. For Delivery Method, if MongoDB requires user authentication, configure a MongoDB user and password. The MongoDB user must be configured in the MongoDB admin database and have permissions to create new MongoDB databases (the MongoDB role "readWriteAnyDatabase").

Configuring the JavaScript Triggers

JavaScript triggers are fired for all updates to the raw collections. The purpose of the triggers is to maintain the aggregated collections. The JavaScript triggers are configured as business actions in STEP and then applied when configuring the MongoDB Adapter.

Continue with setting up the Mongo Delivery Method options, by following the **Mongo Delivery Method** documentation.

MongoDB Adapter Setup Quick Guides

The following information is available to assist in the setup of Mongo Delivery Method.

STEP JSON Schema

The STEP JSON schema can be downloaded from your STEP server at: http:///*[enter step-server]*/files/StepSchema.json

Configuring Mongo Authentication Quick Guide

The following quick guide describes how to configure a Mongo database to use authentication. Details may be found in the Mongo documentation on the web.

The guide assumes that the setup is used on a Mongo database installation running as a single server installation (i.e., the Mongo database is running as a standalone server).

- Start the Mongo database without authentication.
- Log onto the database using the Mongo client.
- Create a system administrator user.

```
> use admin
switched to db admin
> db.createUser(
   {
     user: "admin",
     pwd: "password",
     roles:
     [
       {
        role: "userAdminAnyDatabase",
        db: "admin"
       }
     ]
   }
)

Successfully added user: {
  "user" : "admin",
  "roles" : [
   {
    "role" : "userAdminAnyDatabase",
    "db" : "admin"
   }
  ]
}
```

4. Create the user that will be used when the MongoDB adapter logs onto the Mongo database. This user should be created in the admin database with roles to read and write any databases in Mongo:

```
> use admin
switched to db admin
> db.createUser( {"user" : "stepsys", "pwd" : "stepsys", "roles" : [
"readWriteAnyDatabase" ] })
Successfully added user: { "user" : "stepsys", "roles" : [ "readWriteAnyDatabase" ]
}
```

5. Stop the Mongo database and enable authentication. This may be done in the Mongo configuration file:

```
auth = true
```

   or from the command line:

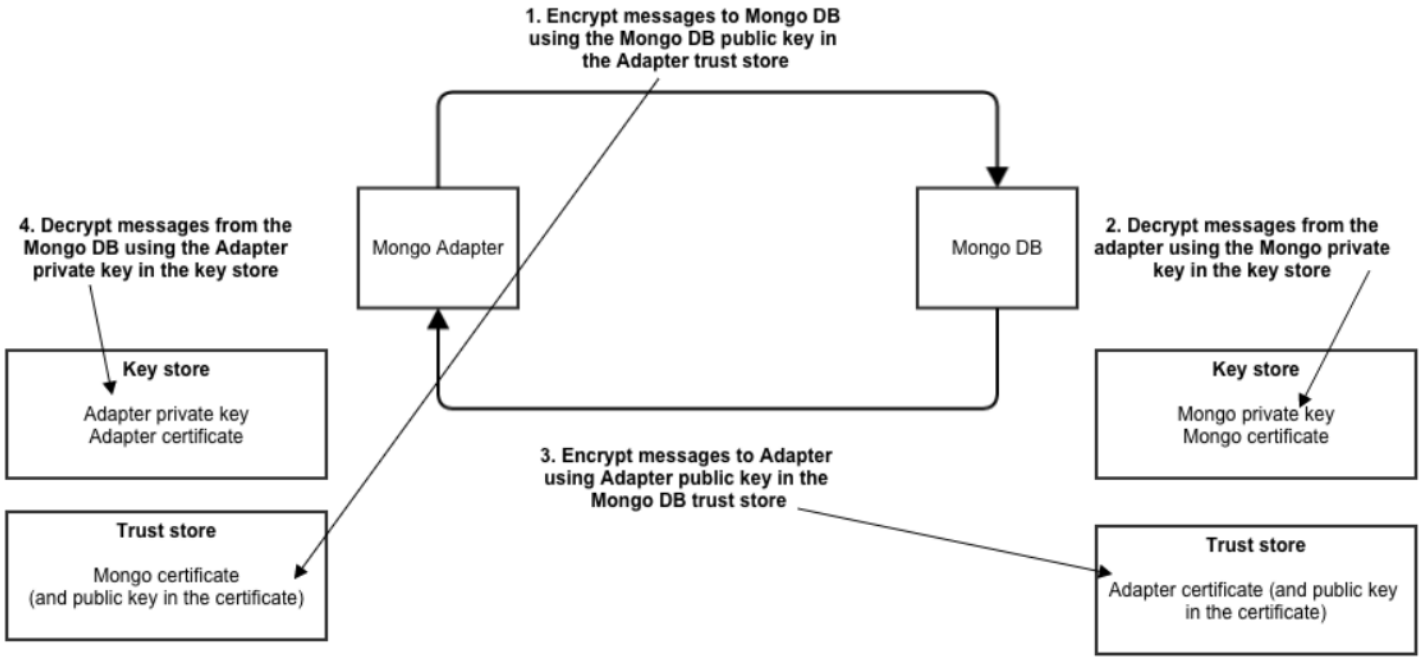
```
mongod --auth --config
```

SSL Configuration Quick Guide

When the Mongo DB adapter is configured for SSL, a key store and a trust store must be configured.

- The key store is a key store file in jks format containing the Mongo Adapter (STEP) private key and certificate.
- The trust store is a trust store file in jks format containing the Mongo DB public key and certificate.

Key store files may be created from the public and private key files and certificates using utilities like the java keytool utility. The diagram below illustrates the uses of the key store and trust store.



To configure SSL encryption on the STEP-to-Mongo database connection:

- Configure the Mongo database to use SSL: http://docs.mongodb.org/manual/tutorial/configure-ssl/
- Configure the Mongo client to connect to a Mongo database using SSL: http://docs.mongodb.org/manual/tutorial/configure-ssl-clients/

The Mongo database available as a free download does not support SSL. (SSL requires the Enterprise edition of Mongo.) Instead, download the appropriate source and compile your own version of the Mongo database from the web: https://www.mongodb.com/docs/manual/installation/

Mongo Delivery Method Conversion Example

The following examples use this product with type = 'Item', ID 'XYZ', and Name = 'XYZ Name.' The product is located in the family 'FamilyXYZ2.'

The STEP Mongo Adapter transforms this to the STEP JSON document:

```
{
_id : "EXA-5002-1001",
objectTypeID : "Chair",
parentID : "Chairs",
name : "Arm chair",
type : "product"
}
```

The id of the product is used as unique ObjectId in the MongoDB database, and therefore the STEPXML id field is transformed to _id. The UserTypeID XML attribute is converted to the objectTypeID field. The type field describes the kind of STEP object that is exported. For Basic STEP objects, the types are 'product', 'asset', 'classification', and 'entity.' Other STEP objects are described in the following topics.

Values

In this example, a product has two attributes:

- A single value attribute Color with the value 'Brown'
- A multi value attribute Height with the value = '43', the unit = 'inches', and the value = '120', with the unit = 'cm.'

When this is transformed to a STEP JSON document by the STEP MongoDB adapter, it results in the following:

```
{
  _id : "EXA-5002-1001",
  objectTypeID : "Chair",
  parentID : "Chairs",
  name : "Arm chair",
  type : "product"
  values : {
    Color : "Brown",
    Height : [ "43 inches", "120 cm"]
  }
}
```

The data displayed includes:

- Values of the product are stored as a subdocument with the name 'values.'
- The value of the single-valued attribute 'Color' is represented as a field-and-value pair: the ID of the attribute is the field, and the value is the value of the attribute represented as a string.
- The multi-valued attribute 'Height' is represented as a field-to-arrays of strings. Again, the ID of the attribute is the field, and the string array contains a string for each value.
- The name of the unit is appended to the attribute value with a space for all values with a unit. In the example above, '120 cm' is the concatenation of the value '120' and 'cm' from the unit named 'unece.unit.CM.'

Values of specification attributes that are inherited to the product are included in the JSON document of the product. For example, if the parent contains an attribute called Brand with the value 'Office chairs', the JSON document looks like the following:

```
{
  _id : "EXA-5002-1001",
  objectTypeID : "Chair",
  parentID : "Chairs",
  name : "Arm chair",
  type : "product"
  values : {
    color : "Brown",
    Height : [ "43 inches", "120 cm"],
    Brand: "Office chairs"
    }
}
```

LOV Value IDs

When values are included from an LOV that uses Value IDs (via the parameter 'Use Ids on values' = Yes on the LOV), the Value and the ValueID are stored as a subdocument with the name 'extValues.' For backward compatibility, the LOV value is also included in the 'value' subdocument, meaning this data is available twice, and can be extracted based on the need.

Note: If the LOV does not use Value IDs, the extValues subdocument is not written.

In this example, the LOV value of '4001' has a Value ID of '4001_CALYPSO.'

```
{
  _id : "EXA-5002-1001",
  objectTypeID : "Chair",
  parentID : "Chairs",
  name : "Arm chair",
  type : "product"
  values : {
    color : "Brown",
    Height : [ "43 inches", "120 cm"],
    Brand: "Office chairs"
    attribute_id: "4001",
},
    extValues: {
    attribute_id: {
        value: "4001",
        valueID: "4001_CALYPSO"
    }
}
```

The data displayed includes:

- The selected LOV value is stored within the subdocument with the name 'values' in the attribute_id field and again with the 'extValues' subdocument in the 'attribute_id: {value:' field.
- The value ID of the selected LOV value is represented once with the 'extValues' subdocument in the 'attribute_id: {valueID:' field.

References and Links

References and links are converted to STEP JSON in a subdocument called 'references.' This subdocument contains a field-and-value pair, where the field is the ID of the reference type, and the value is either a subdocument that defines the target of a reference, or an array of subdocuments that each define the target of a reference.

If the reference can only reference one target for the same source, the value is a subdocument. If the reference can reference more targets for the same source, the value is an array.

In this example, the product uses a 'Primary Image' asset reference to reference the image for the product. The asset has the ID 'Image_EXA-5002-1001.' The STEP JSON then exports as:

```
{
  _id : "EXA-5002-1001",
  objectTypeID : "Chair",
  parentID : "Chairs",
  name : "Arm chair",
  type : "product"
  values : {
    color : "Brown",
```

```
    Height : [ "43 inches", "120 cm"],
  Brand: "Office chairs"
  }
  references : {
    Primary Image : { targetID : "Image-EXA-5002-1001" }
  }
}
```

To find the asset for the reference, it is necessary to find the reference type 'Primary Image', which is stored in the raw collection 'referenceType.' The target type of the reference comes from the reference type.

The JSON of the reference target is found by searching the raw collection given by the reference type of an object with 'targetID.'

In the example above, the target type of the 'Primary Image' reference type is 'asset.' By looking in the 'asset' raw collection, the asset given by Image-EXE-5002-1001 is found.

Adding the reference 'Secondary Image', which can reference more targets such as ('Image1' and 'Image2') to the product, results in the following STEP JSON:

```
{
  _id : "EXA-5002-1001",
  objectTypeID : "Chair",
  parentID : "Chairs",
  name : "Arm chair",
  type : "product"
  values : {
    color : "Brown",
    Height : [ "43 inches", "120 cm"],
  Brand: "Office chairs"
}
references : {
  Primary Image : { targetID : "Image-EXA-5002-1001" },
    Secondary Image : [ { targetID : "Image1" }, { targetID : "Image2" } ]
  }
}
```

If the reference contains a meta data attribute, the values of the meta data attribute are added as a subdocument to the reference subdocument that contains the value of the attribute:

```
{
  _id : "EXA-5002-1001",
  objectTypeID : "Chair",
  parentID : "Chairs",
  name : "Arm chair",
  type : "product"
  values : {
    color : "Brown",
    Height : [ "43 inches", "120 cm"],
```

```
  Brand: "Office chairs"
}
  references : {
    Primary Image : { targetID : "Image-EXA-5002-1001",
                      values: { ShowOnWeb : "true" },
                     },
      Secondary Image : [ { targetID : "Image1" }, { targetID : "Image2" } ]
  }
}
```

The reference type 'Primary Image' has a meta data attribute called 'ShowOnWeb.' The attribute is set to 'true' for the reference from the product to the asset 'PrimaryAsset.'

Classification to product links are special because they are owned by either the product or the classification. This will always be the same for a specific classification to product link type. The JSON document only contains the classification-to-product links owned by the object. That is, the JSON document for a product only contains classification-to-product links owned by the product.

Attribute Types

Exporting an attribute type with the ID 'AttributeTypeID', and the name 'Name' linked in two parent attribute groups 'Parent1' and 'Parent2' results in the following JSON:

```
{
_id : "AttributeTypeID",
parentID : [ "Parent1", "Parent2"],
name : "Name",
listOfValuesID : "ListOfValuesID",
validUnitIDs : ["unece.unit.MMT", "unece.unit.CMT"],
type : "attribute"
}
```

Reference types are stored in the raw collection 'attribute.' To export reference types, verify that 'Include Attribute Definitions' is set to at least 'Selected' in the Process Engine configuration.

Data Containers

For both entities and products, data containers associated with these objects will be included, if included in the export configuration. A 'dataContainers' property will contain all of the associated attributes in this data container:

```
{
"_id":"CommonPlaceBook",
"objectTypeID":"Product user-type root",
"dataContainers":{
  "DC-2127822":{
    "values":{
      "Color":"Red",
```

```
      "Height":"12 in",
      "Width":"9 in"
    },
    "extValues":{
      "Color":{
        "value":"Red"
      },
      "Height":{
        "unitID":"unece.unit.INH",
        "value":"12"
      },
      "Width":{
        "unitID":"unece.unit.INH",
        "value":"9"
      }
    }
  }
},
"name":"Robert Black's Commonplace Book",
"type":"product",
"parentID":"TopNode"
}
```

In this example, the attributes 'Color,' 'Height,' and 'Width' in the single valued Data Container DC-2127822 are all exported to JSON.

In case of a multivalued data container, the content will always be exported as an array:

```
"ContEmailDataContainer": [
  {
    "extValues": {
      "CalcContEmailDataContainer": {
        "value": "313bobby@stibosystems.net"
      },
      "ContEmail": {
        "value": "313bobby@stibosystems.net"
      }
    },
    "values": {
      "CalcContEmailDataContainer": "313bobby@stibosystems.net",
      "ContEmail": "313bobby@stibosystems.net"
    }
  }
],
```

The values in the data containers are within the square brackets.

Reference Types

Exporting a product-to-product reference type with the ID 'ReferenceTypeID' and the name 'ReferenceName' generates the following JSON:

```
{
_id : "ReferenceTypeID",
name : "ReferenceName",
type : "referenceType",
targetType : "product"
}
```

In this example, 'targetType' is the target type of the reference. Possible target types are 'product','asset', 'classification', and 'entity.'

Reference types are stored in the raw collection 'referenceType.' When you export reference types, verify that Configuration > Include Link, Reference and Object Types is set to at least 'Minimum' in the Process Engine configuration.

Asset Push Locations

Asset Push Locations are stored in the assetPushLocations subdocument, in the root of the Asset documents in the asset collection. It provides the relative path to different versions of the asset. The assetPushLocations subdocument contains field-and-value pairs, where the field name is the assetpush configuration id, and the value is the relative path that the image file has been pushed to. The relative path is relative to the AssetPushClients root folder, so to get the full path of the image, you have to prepend the path to the assetpush client root directory, as shown in the following example:

```
assetPushLocations : {
    AllAssets-approved : "AllAssets-approved/std.lang.all/73/15/7315.pdf",
    AllWebsiteImages-approved : "AllWebsiteImages-
approved/std.lang.all/73/15/7315.pdf",
    small : "small/std.lang.all/73/15/7315.jpg",
    large : "large/std.lang.all/73/15/7315.pdf"
}
```

Tables

Tables are defined on classifications or products. The table subdocument is stored in the root of the product and classification document, in the product and classification collection. The table subdocument contains field-and-value pairs, where the field name is the table Type name of the table on the product or classification.

A table is exported from STEP in a resolved and transformed state. That is, any transformations defined in STEP have already been applied to the table and the row and column layout has therefore been resolved.

Formatting is not applied to the table before it is exported. So, a web application is not bound by the STEP table formatting. However, if any formatting has been defined for the table in STEP, it is included in the export so that the website application can be configured to render the formatting on the website.

The table contains an array of column elements that contain the formatting meta data for each column and an array of rows. The row contains the formatting information for each row, plus an array of cells. Each cell has a number of formatting meta data attributes and a text field that contains the content of the table cell.

Table Formatting

Formatting meta data can be stored on table, column, row, and cell level, and is inherited downwards from table level to column level to row level to cell level.

If a column is specified to use an underlined text style, and a row is specified to be bold text style, the formatting information is accumulated so that the cell at the intersection of the row and column is rendered as both underlined and bold.

Formatting meta data defined at a lower level overrides meta data at a higher level. For example, if a table has a gray background color, and a column has a red background color, then cells in that row are rendered with a red background.

A simple way of formatting a table in HTML is to map the tableType, columnType, and rowType to element classes in HTML, and then use a CSS style sheet to define the table formatting, as shown in the following example.

```
tables : {
  Description Table : {
    columns :
      [ { columnType : "Description" }, { columnType : "Description" }, { columnType
: "Description" } ],
    rows : [
      { rowType : "Header",
        cells : [
          { ruleRight : "0.5 pt",
            text : "Header 1",
            verticalAlignment : "top",
            cellStoryDirection : "horizontal",
            backgroundColor : "Light Blue",
            column : "0",
            ruleBelow : "0.5 pt",
            ruleAbove : "0.5 pt",
          textStyle : "TableHeader—Description",
            ruleLeft : "0.5 pt" },
          { ruleRight : "0.5 pt",
            text : "Header 2",
            verticalAlignment : "top",
            cellStoryDirection : "horizontal",
            backgroundColor : "Light Blue",
            column : "1",
            ruleBelow : "0.5 pt",
            ruleAbove : "0.5 pt",
            textStyle : "TableHeader—Description",
```

```
            ruleLeft : "0.5 pt" }] },
      { rowType : "Header",
          cells : [
          { ruleRight : "0.5 pt",
            text : "Value 1",
            verticalAlignment : "top",
            cellStoryDirection : "horizontal",
            backgroundColor : "Light Blue",
            column : "0",
            ruleBelow : "0.5 pt",
            ruleAbove : "0.5 pt",
            textStyle : "TableHeader—Description",
            ruleLeft : "0.5 pt" },
          { ruleRight : "0.5 pt",
            text : "Value 2",
            verticalAlignment : "top",
            cellStoryDirection : "horizontal",
            backgroundColor : "Light Blue",
            column : "1",
            ruleBelow : "0.5 pt",
            ruleAbove : "0.5 pt",
            textStyle : "TableHeader—Description",
            ruleLeft : "0.5 pt" }] }
] } }
```

Attribute Links

AttributeLinks define the validity of attributes for products linked into the product and classification hierarchies. The attributeLinks subdocument is stored at the root of the product and classification document structure. The attribute links can themselves contain a values subdocument that defines the meta data values attached to the attributeLink.

The attributeLinks subdocument contains field value pairs, where the field name is the attributeID, and the value is a subdocument that contains any meta data values related to the attribute link. The following is an example of an attributeLinks subdocument:

```
attributeLinks : { Voltage range : { }
                   Rotary address switches : { },
                   Default address : { },
                   Power consumption : { }
                   }
```

Data Container Type Links

The validity for Data Container Types for products can be restricted to products in certain product and/or classification hierarchies just like attributes can.

If the classification or product root to which a data container is restricted is exported, then the following JSON subdocument is returned:

```
"dataContainerTypeLinks": [
  "ProdDC"
]
```

The following example is an export of the product root node called 'TopNodeWDC.' In this case, both the 'SpecificationAttribute' attribute is restricted to this root node, and the 'ProdDC' data container is restricted to this root node. This export returns an attributeLinks subdocument (as described in the 'Attribute Links' section) and a new dataContainerTypeLinks subdocument:

```
{
  "_id": "TopNodeWDC",
  "attributeLinks": {
    "SpecificationAttribute": {}
  },
  "dataContainerTypeLinks": [
    "ProdDC"
  ],
  ...
}
```

Data Containers Types

Data Container types collection contains the definition of the data containers. These collections may include metadata of the data container. The JSON will look similar to the following:

```
{
  "_id":"DC-2127822",
  "name":"Product Description",
  "type":"datacontainertype",
  "multiValued":"false"
}
{
  "_id": "ContPhoneDataContainer",
  "extValues": {
    "DC Long Description": {
      "value": "This DC is used for Phone numbers"
    }
  },
  "multiValued": "true",
  "name": "Phone",
  "type": "datacontainertype",
  "values": {
    "DC Long Description": "This DC is used for Phone numbers"
  }
}
```

Unit

Exporting an unit with the ID 'unece.unit.CMT', and the name 'cm' results in the following JSON:

```
{
"_id" : "unece.unit.CMT",
"values" : {
"4941" : "centimeter"
},
"conversionToBase" : {
    "factor" : "100",
    "unitID" : "unece.unit.MTR",
    "offset" : "0"
},
"name" : "cm",
"type" : "unit"
}
```

The unit document contains the 'conversionToBase ' object that can be used for converting values from a unit to its base unit. The 'conversionToBase ' object of the above example shows how to convert values of the unit centimeters to meters.

Units are stored in the raw collection 'unit.' To export units, verify that Configuration > Include Units is set to at least 'Minimum' in the Process Engine configuration.

List Of Values

Exporting a ListOfValues object with the ID 'LOVID', and the name 'List of values' results in the following JSON:

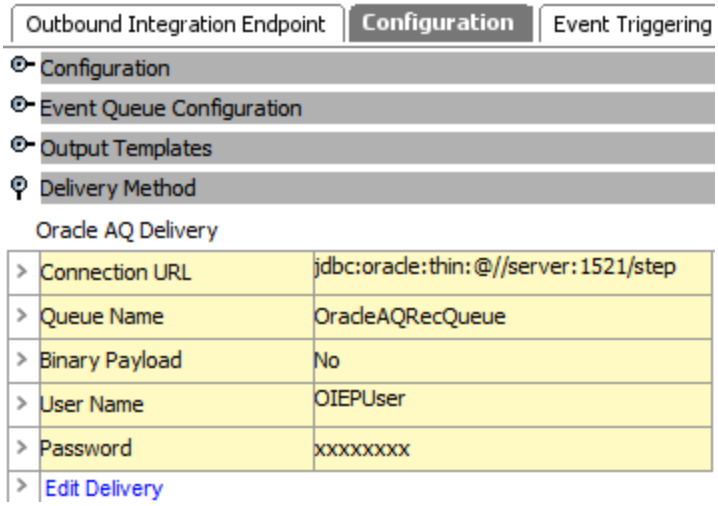
```
{
"_id" : "LOVID",
"values" : {
"6823" : "This is a metadata value of the lov'"
},
"name" : "List of values",
"validUnitIDs" : [
"unece.unit.CMT",
"unece.unit.MTR"
],
"type" : "listofvalues"
}
```

List Of Values are stored in the raw collection 'listofvalues.' To export List Of Values, verify that 'Include List Of Values Definitions' is set to at least 'Minimum' in the Process Engine configuration.

Oracle AQ Delivery Method

This delivery option is only available in OIEPs.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



Prerequisites

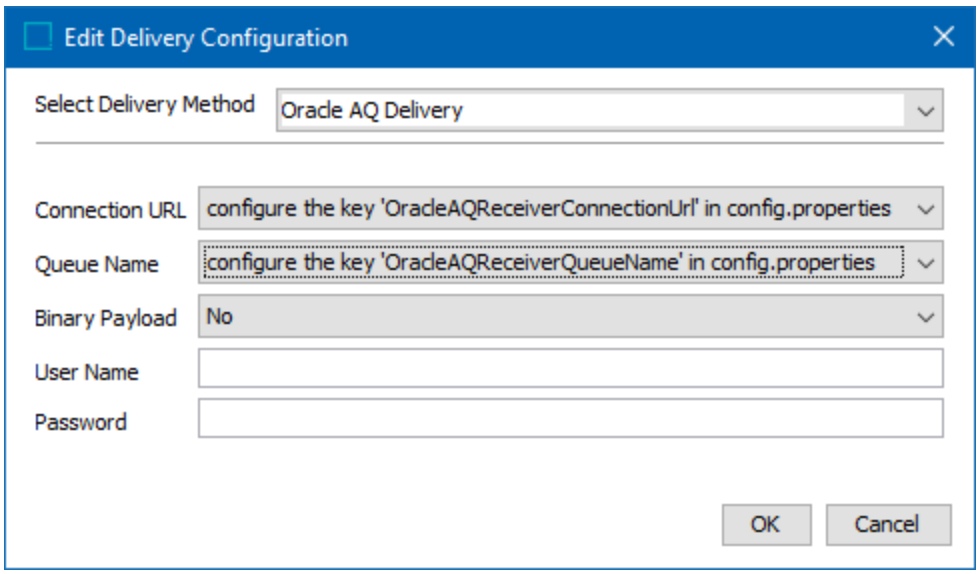
Changes to the properties file, outlined below, are implemented when the server is restarted.

1. Prior to configuration, clicking the **Connection URL** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **OracleAQReceiverConnectionURL** property.
2. Prior to configuration, clicking the **Queue name** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **OracleAQReceiverQueueName** property.

Configuration

For information on a parameter, hover over the parameter field to display help text.

1. In the **Select Delivery Method** list, choose **Oracle AQ Delivery**.



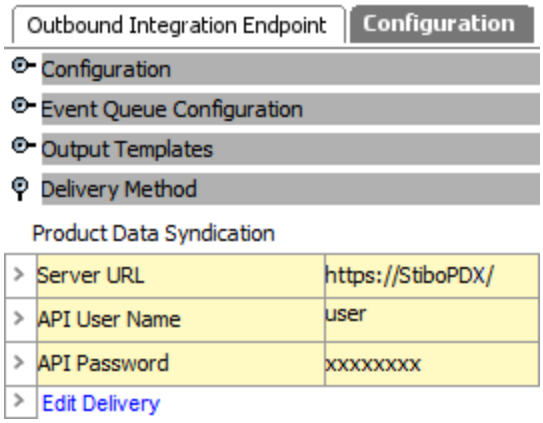
2. In the **Connection URL** list, select a URL that points to Oracle AQ.
3. In the **Queue name** list, select an Oracle AQ queue name.
4. For **Binary Payload**, select Yes or No. 'Yes' is selected for non-text files.
5. In the **User Name** field, enter the user name that will be used to log on to Oracle
6. In the **Password** field, enter the password that will be used to log on to Oracle.
7. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

Product Data Exchange 2 Delivery Method

The Product Data Exchange 2 delivery method is only available in OIEPs. STEP sends data to the Product Data Exchange (PDX) platform via the default PDX Outbound Integration Endpoint and the API.

For additional information on PDX, refer to the **Product Data Exchange** topic within the **Data Integration** documentation, or contact your Stibo Systems representative.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.

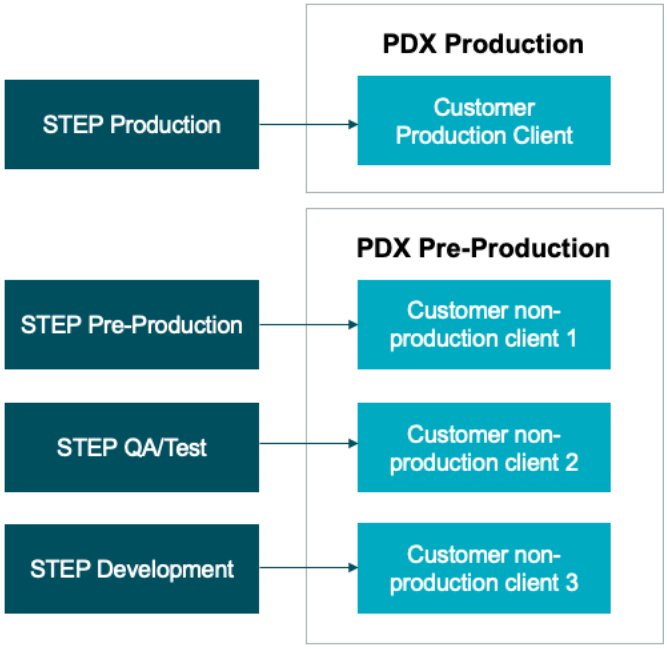


Prerequisites

The server-side setting of the sharedconfig.properties PDX.Url determines the PDX environments eligible in the OIEP configuration of the delivery method. Two PDX environments are relevant to this setting:

- PDX Production (https://api.pdx.stibosystems.com)
- PDX Pre-production (https://api.pdx-preprod.stibosystems.com)

General setup will include one account on the Production environment, which processes and syndicates live data, while multiple non-production accounts / clients may exist on the Pre-production environment. In the image below, STEP environments are in the column on the left, and those on the right represent PDX environments with PDX accounts.



Note: Changes to the properties file, outlined below, are implemented when the server is restarted.

The sharedconfig.properties on the STEP Production environment specify the PDX Production environment (containing the customer's production account / client) as the valid target of the PDX OIEP configured, as seen in the example below:

```
PDX.Url=1=https://api.pdx.stibosystems.com,2=https://api.pdx-preprod.stibosystems.com
```

To ensure product information maintained in all the contexts selected for syndication in the PDX OIEP setup can be transferred, the following property should be set to 'false':

```
PDXDelivery2.LocaleChecking=false
```

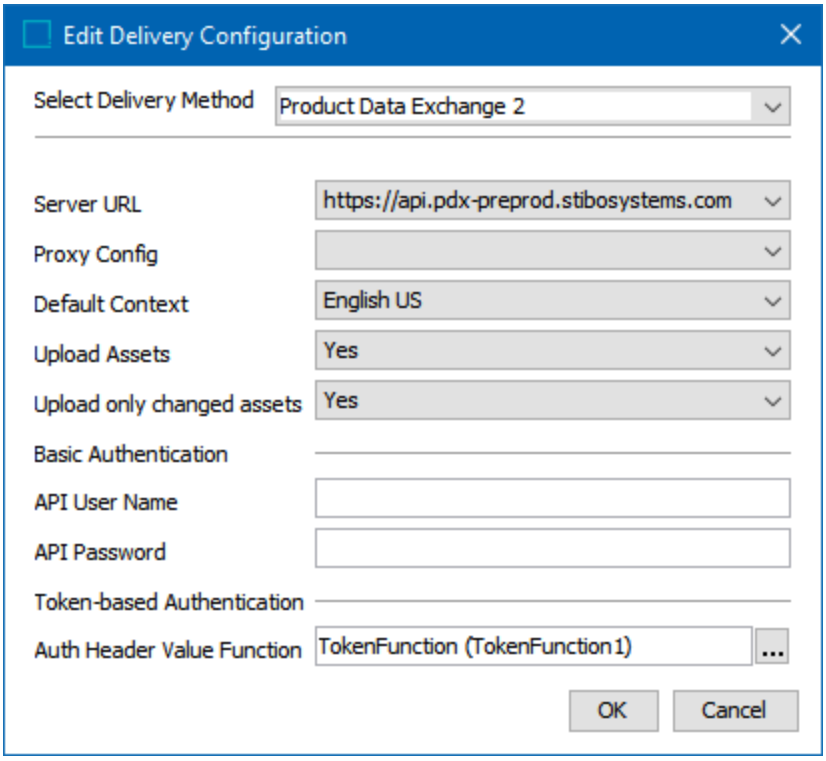
1. Determine the API user name and password. This will be required on the Edit Delivery Configuration dialog.
2. If required, provide a selection for the **Proxy Config** parameter. The configurations that appear in this dropdown are populated from properties added into the sharedconfig.properties file using the dynamic property 'Http.ProxyConfiguration.[name],' which has its own property for Host, Port, User, and Password.
3. Perform additional configuration required in the OIEP as laid out in the **Setting Up the PDX OIEP** topic in the **Data Integration** documentation.

Configuration

For information on a parameter, hover over the parameter field to display help text.

1. On the **Configuration** tab of the OIEP, in the **Delivery Method** flipper, click the **Edit Delivery** link to display the 'Edit Delivery Configuration' dialog.

2. For the **Select Delivery Method** parameter, choose **Product Data Exchange 2**.



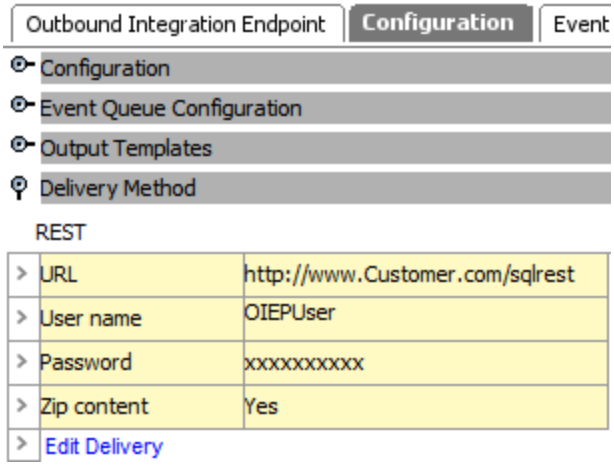
3. **Server URL** - Select the URL from the dropdown, if not preselected.
4. **API User Name** - Provide the user name for the user that has access to the API.
5. **API Password** - Provide the password for the user that has access to the API.
6. **Proxy Config** - Select the desired HTTP proxy configuration if the delivery connection must first pass through a proxy server with its own login requirement.
7. **Default Context** - Select the context from the dropdown menu that will serve as the default context. The data in this context will be used to populate the default language layer in PDX.
8. **Upload only changed assets**- Set the 'Upload only changed assets' setting to 'Yes,' if you only want to include referenced asset objects where either attribute values or the asset content has changed.
9. Click the **OK** button to save the delivery method.

Note: For more information regarding HTTP proxy configurations, refer to the topic **HTTP Proxy Configurations** in the **Data Exchange** documentation.

REST Delivery Method

The REST Delivery Method delivers a call-back URL to the REST service and does not include STEP data. The data can be fetched from the call-back URL by the receiving REST service. This delivery option is only available in OIEPs.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



Prerequisites

Changes to the properties file, outlined below, are implemented when the server is restarted.

Prior to configuration, clicking the **URL** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **RESTDeliveryURL** property. If necessary, use a comma to separate multiple URLs.

This should be a URL to a REST POST method, for example, http://server/rest_URL. A URL where the result of the endpoint can be fetched is returned in the POST call.

The following is an example of a complete property entry for two systems named 'qa' and 'stage,' as well as their URLs 'http://step-qa' and 'http://step-stage':

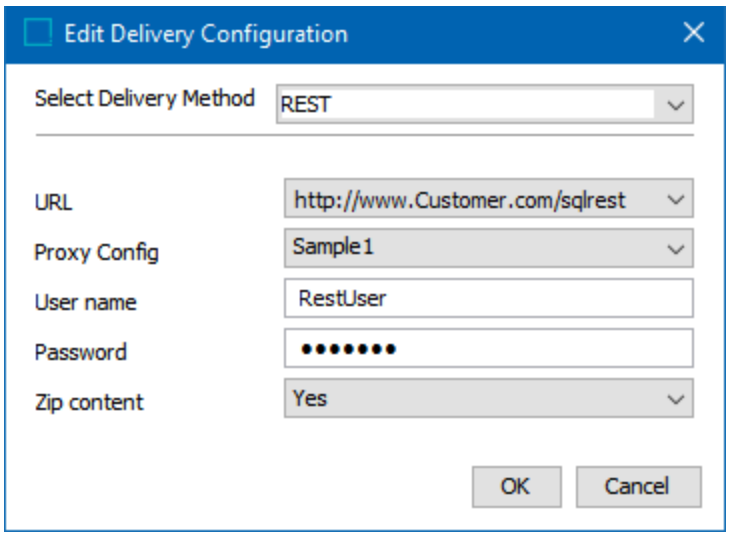
```
RESTDeliveryURL=qa=http://step-qa,stage=http://step-stage
```

Important: For information about how to use the REST API to upload files to REST, access the **STEP API Documentation**, available at [system]/sdk or from the Start Page.

Configuration

For information on a parameter, hover over the parameter field to display help text.

1. For **Select Delivery Method**, choose **REST**.



2. In **URL**, select the URL that points to the REST endpoint where you would like to receive the delivered data.
3. In **Proxy Config**, select the desired HTTP proxy configuration if the delivery connection must first pass through a proxy server with its own login requirement.
4. In **User Name**, enter the user name that will be used to log on to the REST endpoint.
5. In **Password**, enter the password that will be used to log on to the REST endpoint.
6. In **Zip content**, specify whether to zip the contents before upload.
7. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

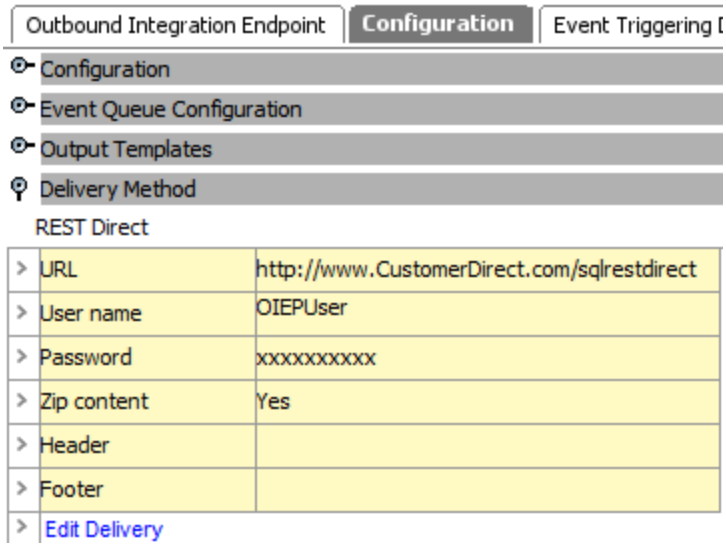
Note: For more information regarding HTTP proxy configurations, refer to the topic **HTTP Proxy Configurations** in the **Data Exchange** documentation.

REST Direct Delivery Method

The REST Direct delivery method differs from the standard REST delivery method in that the data is delivered directly to the REST service and no call-back URL is required.

This delivery option is only available in OIEPs and if the delivery fails the OIEP is disabled. There is no resilience handling.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



To use mTLS or TLS with this type of endpoint, refer to the **Mutual Transport Layer Security** topic.

Prerequisites

Changes to the properties file, outlined below, are implemented when the server is restarted.

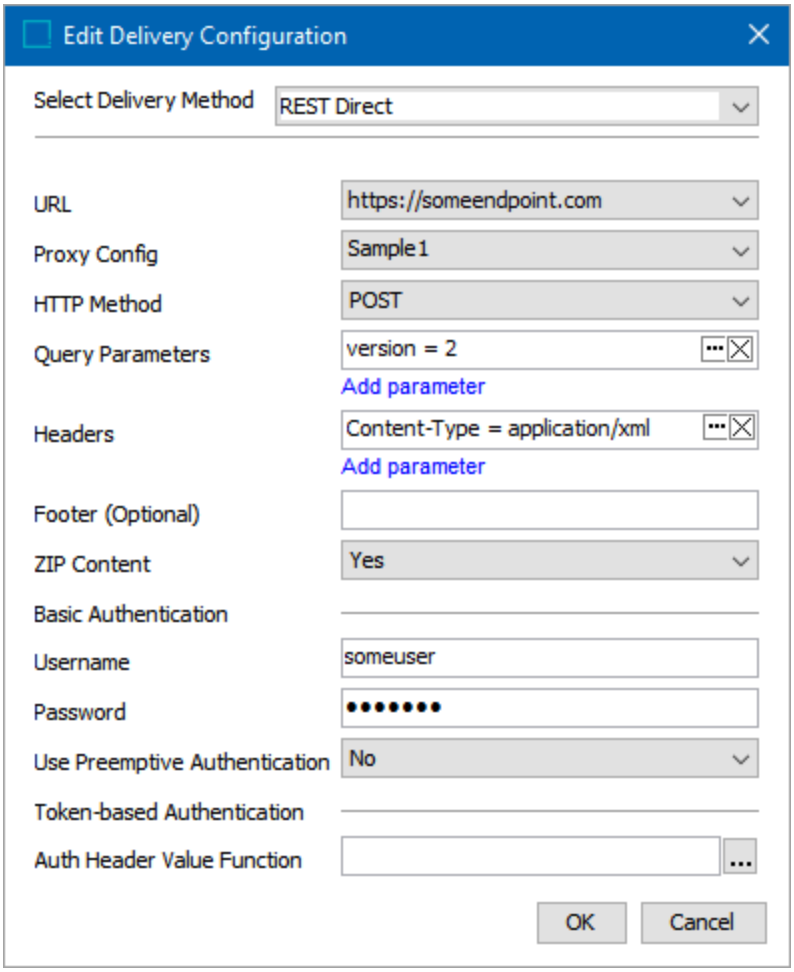
1. The REST Direct delivery method reads the outgoing file into memory before sending. To handle the size of the outgoing file and prevent time-outs and rejections you must scale the heap size.
2. Prior to configuration, clicking the **URL** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **RestDirectDeliveryURL** property. If necessary, use a comma to separate multiple URLs. The image below shows the result of using the following example for a property entry involving two systems:

```
RestDirectDeliveryURL = 1=http://myfirstendpoint, 2=http://mysecondendpoint
```

Configuration

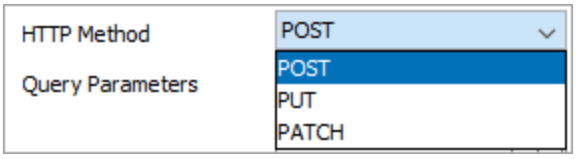
For information on a parameter, hover over the parameter field to display help text.

1. For **Select Delivery Method**, choose **REST Direct**.

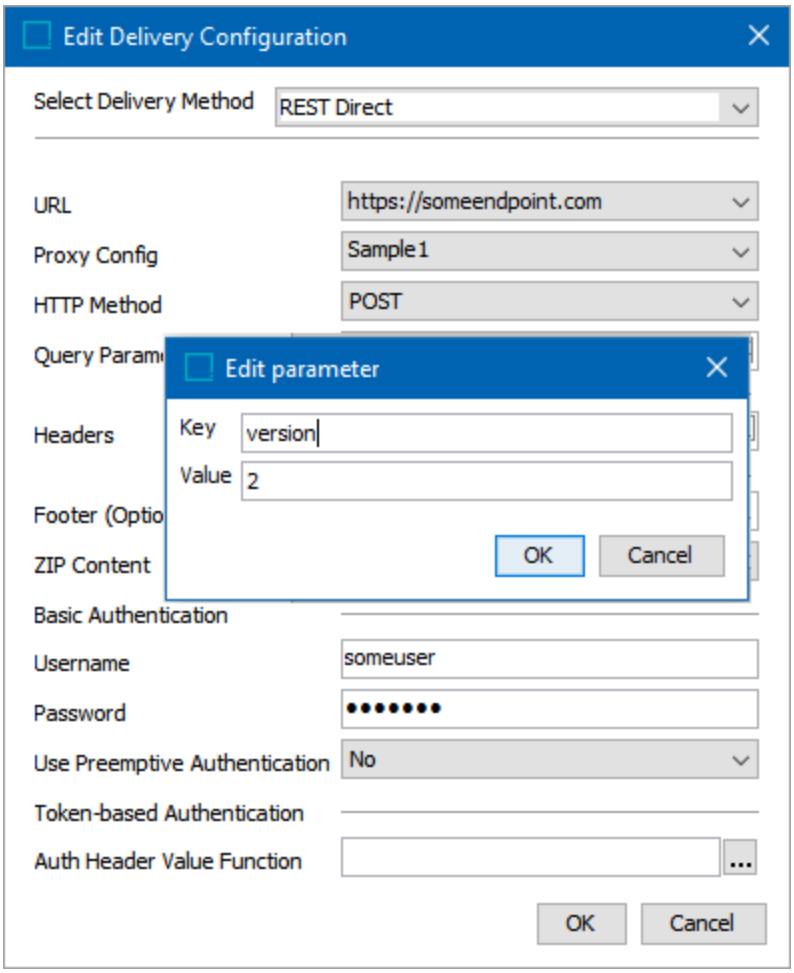


2. In **URL**, select the URL that points to the REST endpoint where the delivered data should be received.
3. In **Proxy Config**, select the desired HTTP proxy configuration if the delivery connection must first pass through a proxy server with its own login requirement.

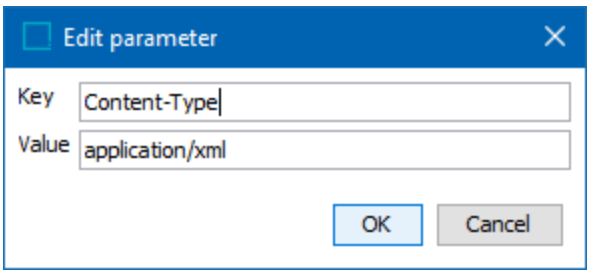
4. In the **HTTP Method** option, use the dropdown to select a POST, PUT, or PATCH method.



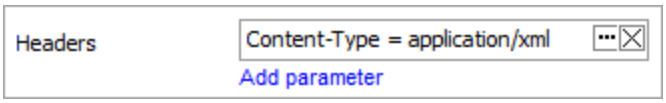
If required, in the **Query Parameters** field add any query parameters that are required for a successful REST transaction. Click on the **Add parameter** link to add the first query parameter. Once a query parameter is added, click the ellipsis button (...) to add additional query parameters. To remove query parameters, click on the 'X.'



5. For **Headers**, click the **Add Parameter** link and add a key and a value. This is part of the HTTP network protocol.



6. On the Add parameter dialog, click the **OK** button and the Header is displayed in the field. If multiple headers are needed, use the **Add Parameter** link to add each additional key and value pair.

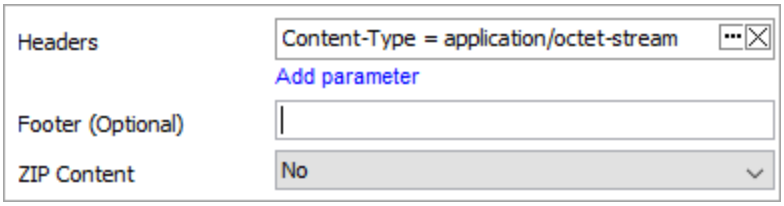


Note: Once a header is displayed, click the ellipsis button (...) to edit it or the X button to remove it.

For more information about headers, refer to the 'Additional Request Information' section at the end of this topic.

7. In **Footer**, add data required for the recipient to verify that the full message was received. This is part of the HTTP network protocol but not required.

   The footer can be used to mark the end of a multi-part REST call (that is a REST call containing the payload split in more packages). The footer could also contain a checksum that the receiver can use to detect if the payload in a multi-part message has been changed by the middleware.



8. In **Zip Content**, specify whether to zip the contents before upload.
9. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

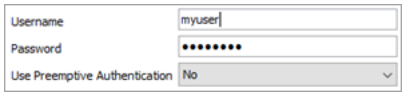
Note: For more information regarding HTTP proxy configurations, refer to the topic **HTTP Proxy Configurations** in the **Data Exchange** documentation.

Authentication

The REST Direct delivery plugin supports both basic authentication and token-based authentication.

Basic Authentication

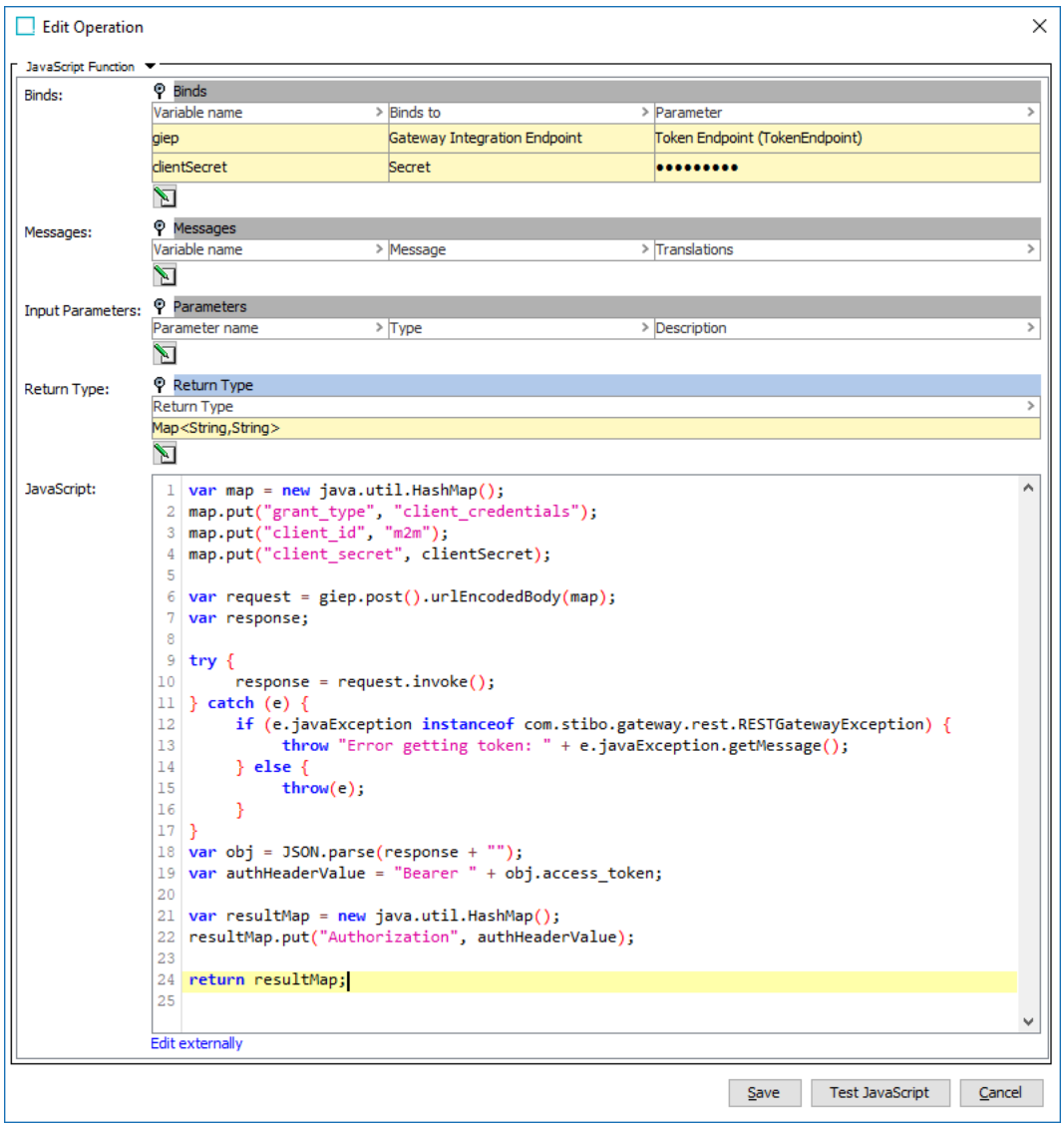
For basic authentication, enter the username and password and specify whether or not the plugin should use preemptive authentication. With preemptive authentication enabled, the basic authentication value for the Authorization header will be sent with the first request to the external service, instead of only sending the value after having received a basic authentication challenge from the service.



Token-Based Authentication

With the token-based authentication option, the responsibility for producing any required request headers is delegated to a business function. The business function must be configured to not expect any input and must produce a Map<String, String>. Each map entry is a header that will be sent with the request to the service.

The following screenshot shows the editor for a compatible business function that retrieves a token using the OAuth 2.0 client credentials flow and passes the token back to the delivery plugin as a value for the Authorization header.



The REST Direct delivery plugin will automatically call the business function when a new token is required.

Note: It is strongly discouraged to configure both basic authentication and token-based authentication. This configuration combination is not supported.

Additional Request Information

By default, the REST Direct delivery option sends multipart/form-data POST requests with the exported file available in the part named 'file' with Content-Type application/octet-stream. The example below shows the properties for such a request:

Headers

Accept-Encoding = gzip,deflate

Connection = close

Content-Length = 1354

Content-Type = multipart/form-data; boundary=JN_qruUpDaHqm9BgW_b4-adAHDftQbjuvDI

Body (Metadata)

Content-Disposition: form-data; name="file"; filename="result.zip"

Content-Type: application/octet-stream

Via the UI it is possible to change the HTTP method and also to overwrite the Content-Type header. For instance, this header could be set to 'application/xml' for a STEPXML export (unzipped) thereby overwriting the default 'multipart/form-data' value making the request properties be:

Headers

Accept-Encoding: gzip,deflate

Connection: close

Content-Length: 1354

Content-Type: application/xml

Body <The exported XML>Additional Request Information

By default, the REST Direct delivery option sends multipart/form-data POST requests with the exported file available in the part named 'file' with Content-Type application/octet-stream. The example below shows the properties for such a request:

Headers

Accept-Encoding = gzip,deflate

Connection = close

Content-Length = 1354

Content-Type = multipart/form-data; boundary=JN_qruUpDaHqm9BgW_b4-adAHDftQbjuvDI

Body (Metadata)

Content-Disposition: form-data; name="file"; filename="result.zip"

Content-Type: application/octet-stream

Via the UI it is possible to change the HTTP method and also to overwrite the Content-Type header. For instance, this header could be set to 'application/xml' for a STEPXML export (unzipped) thereby overwriting the default 'multipart/form-data' value making the request properties be:

Headers

Accept-Encoding: gzip,deflate

Connection: close

Content-Length: 1354

Content-Type: application/xml

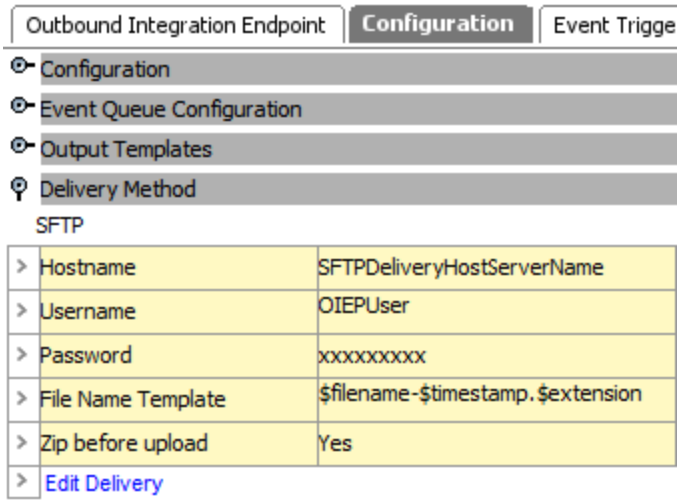
Body <The exported XML>

SFTP Delivery Method

The preferred method for file transfer protocol (FTP) delivery is the SFTP secure delivery method. The SFTP delivery method allows an exported file to be delivered to an external system and is often used when the output files are large or when a different or remote system is in use.

For information on the OIEP FTP delivery method, refer to the **FTP Delivery Method** topic.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



This delivery method is also available in Export Manager as defined in the **SFTP Delivery Method** topic.

Prerequisites

Changes to the properties file, outlined below, are implemented when the server is restarted.

Prior to configuration, clicking the **Host name** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **FTPDeliveryHostName** property. The required format of the property is (square brackets not included): FTPDeliveryHostName=1=[host1],2=[host2].

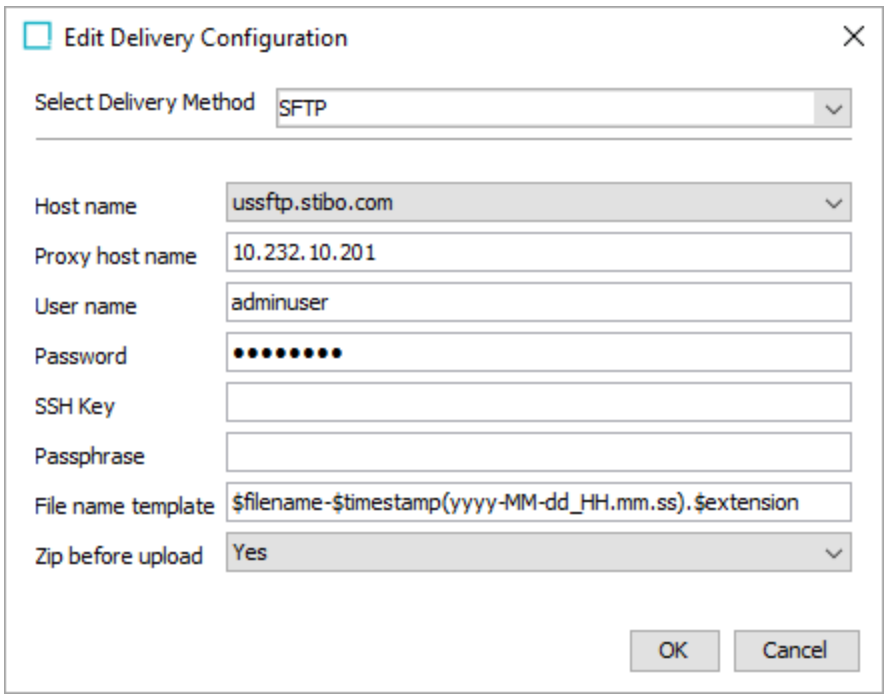
Using the host name shown in the image, the entry in the properties file would be:

```
FTPDeliveryHostName=1=ussftp.stibo.com
```

Configuration

For information on a parameter, hover over the parameter field to display help text.

1. Click the **Select Delivery Method** list to display the dropdown and select **SFTP**.



2. In **Host name**, select the host name of the SFTP server to be used for the delivery.
3. In **Proxy host name**, add the host name to be used for the server proxy. This field is optional.
4. In **User name**, enter the user name that has access to log on to the FTP server.
5. In **Password**, enter the password that will be used to log on to the FTP server. If using the SSH Key, then leave this field blank.
6. In **SSH Key**, enter the full path to the Secure Shell (SSH) key file, if using. If you filled in the Password field in step 5, then leave this field blank.

> **Note:** This SSH key must be an **RSA** private key in the **OpenSSH PEM** format. Other SSH key types or those generated in the new OpenSSH format (the default used in OpenSSH 7.8+) are not currently supported and will result in an **SFTPException: invalid privatekey** error.

7. In **Passphrase**, enter the passphrase that accompanies the SSH key entered, if applicable. If the SSH key does not have a passphrase or you are not using an SSH key, then leave this field blank.
8. In **File name template**, to create the file name used for the output, enter template variables (identified with a $), static text, or use a combination of both. The default template is $filename-$timestamp(yyyy-MM-dd_HH.mm.ss).$extension. Depending on the settings of the configuration, the default template outputs a file name similar to 'GenericXML-2020-07-30_14.07.44.xml.' Each variable is described below:

> **Note:** The 'Zip before upload' parameter also has an impact on the file name.

- **$filename** For event-based OIEPs, this variable is replaced with text to indicate the format of the delivered file as specified in the Output Templates flipper, except for STEPXML when the first and last Event IDs are used. For example, the output file name could be 'csv-2020-07-30_14.09.40.csv' or '1804038-1804038.xml' to indicate that STEPXML was used for a single event.
- **$timestamp**: This variable is replaced with the time in year-day-month-hour-minutes-seconds-milliseconds, excluding the dashes. For example, 20161511094709625 is used for an export on 15 NOV 2016 at 9:47:09:625 AM.

  Adding a definition for the arrangement of the date / time elements provides a way to generate a more readable date. Static characters can be included to separate the timestamp elements and individual elements can be ordered as required, as shown below.

  Note: To revert the **$timestamp** variable so that it uses the legacy definition (milliseconds between January 1, 1970, and the time when the file is uploaded), contact Stibo Systems Support for more information.

  $timestamp(ddMMyyyy): The timestamp is modified to display day-month-year (excluding dashes) and is output as '15112016' for 15 NOV 2016.

  $timestamp(yyyy-MM-dd_HH.mm.ss): The timestamp is modified to display year-month-day_hour.minutes.seconds (including the dashes, underscore, and periods) and is output as '2016-11-15_09.32.43' for 2016 NOV 15 at 9:32.43 AM.

- **$extension** This variable is replaced with the extension of the output file based on the selected format in the Output Templates flipper. For XML-based formats, the output file is set with the .XML extension, while Excel files use .XLS, and comma-separated values use .CSV.

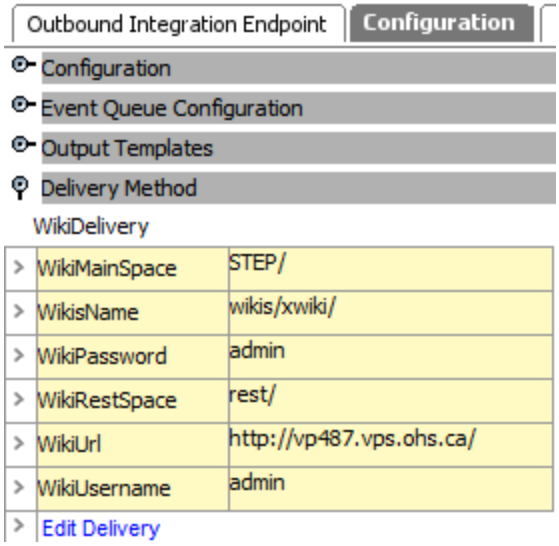
Note: The File Name Template does not support conversions of file formats and can only be used to deliver files in the format specified on the Configuration tab > Output Templates flipper of the outbound integration endpoint.

9. In **Zip before upload**, select an option from the dropdown to specify if the output file should be delivered in a .ZIP (compressed) file format.
   - **Yes** uses 'result_0' before the timestamp variable, and then the extension ZIP. For example, a zipped STEPXML output with the default File Name Template exported on 30 JUL 2020 results in an output .ZIP file named 'result_0-2020-07-30_14.07.44.zip.' The contents of the ZIP file would follow the $filename variable applied in the 'File name template' parameter.
   - **No** uses the 'File name template' for the file name along with the appropriate extension for the selected data format.
10. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

Wiki Delivery Method

The Wiki Delivery Method works with an OIEP configured to deliver metadata to a specified XWiki advanced open source enterprise wiki system. The contents of the wiki can be accessed by clicking a 'Go to Wiki' link from either Web UI or workbench.

The Wiki Metadata functionality is a licensed component and requires a separate installation recipe. For details and the latest recipe, contact your Stibo Systems representative.



Prerequisites

Changes to the properties file, outlined below, are implemented when the server is restarted.

1. Review the **Wiki Metadata** topic for details on the functionality and setup.
2. Prior to configuration, clicking the **Base URL** dropdown parameter displays the required property name. Provide a Base URL selection for the dropdown parameter via the sharedconfig.properties file using the case-sensitive **Wiki.WikiUrl** property. A protocol (http:// or https://) and an ending slash (/) are mandatory. The following is an example of a property entry for two wikis.

   ```
   Wiki.WikiUrl=1=http://vp487.vps.ohs.ca/,2=https://127.0.0.1:8080/
   ```

   If only one URL is required, in the example above, eliminate the ',2=https://127.0.0.1:8080/' text.

   Note: The format for multiple wikis is 1=url1,2=url2,3=url3, where each entry is incremented and unique.

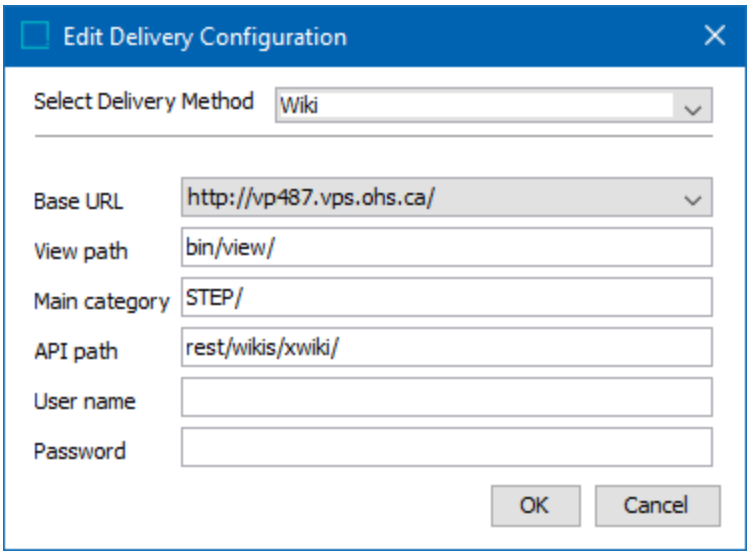
Configuration

For information on a parameter, hover over the parameter field to display help text.

After completing the prerequisite steps, edit the delivery method of the OIEP created during the installation. Use the following steps to configure the OIEP to use the Wiki delivery option.

Important: For the OIEP created automatically upon restarting the server after installing the STEP wiki metadata recipe, most of the following parameters are pre-populated and the default settings should not be changed. Review the parameter details below for more information.

1. For **Select Delivery Method**, choose **Wiki**.



2. In **Base URL**, select the URL that points to the wiki that will be used for this STEP system. Refer to the Prerequisites section for information on setup.
3. In **View Path**, enter the URL elements between the Base URL and the wiki pages, or main category if defined. This is used for the presentation of wiki pages and allows each user to watch pages they select. The slash at the end is mandatory. As shown in the image above, 'bin/view/' is the view path for the following URL: http://vp487.vps.ohs.ca/**bin/view/**STEP/attribute/
4. In **Main category**, enter a name for the main category path. This allows you to organize the metadata pages and can be multiple levels by adding a slash, '/,' between each one. The slash at the end is mandatory. As shown in the image above, 'STEP/' is the main category for the following URL: http://vp487.vps.ohs.ca/bin/view/**STEP/**attribute/
5. In **API path**, specify the path for the wiki REST API, which is used to make modifications to the wiki pages. The slash at the end is mandatory.
6. In **User name**, enter the user with permission to call the REST API and who can make modifications to it.
7. In **Password**, enter the password for the user name.
8. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

OIEP - Event-Based - Event Triggering Definitions Tab

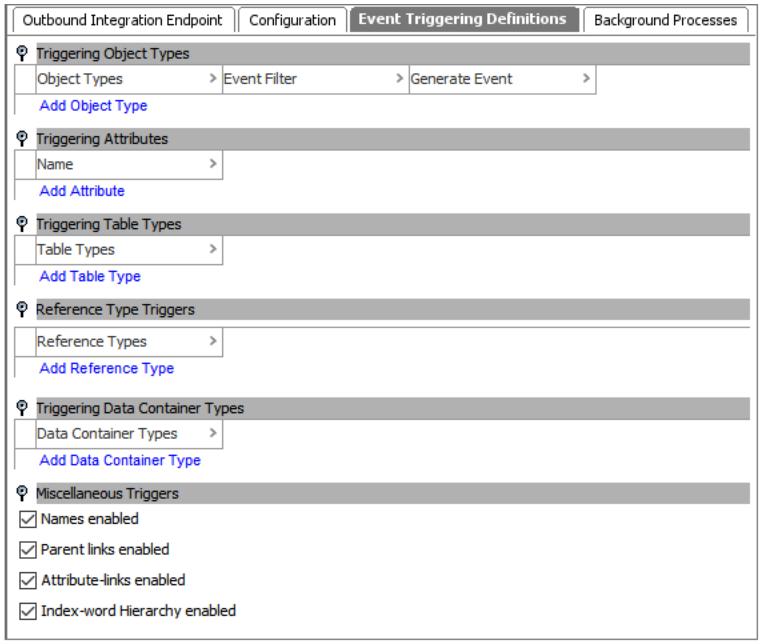
When Event Queue has been selected as the data source in an outbound integration endpoint (OIEP), or when setting up an event processor (EP), you can specify if core events (modify, delete, or create events) should be collected by the OIEP or EP for specific object types.

- Workspace revisable objects, such as a product, generate an event when data changes are approved.
- Global revisable objects, such as an attribute or an entity storing customer data, generate an event as soon as the data changes because approval is not available or required.

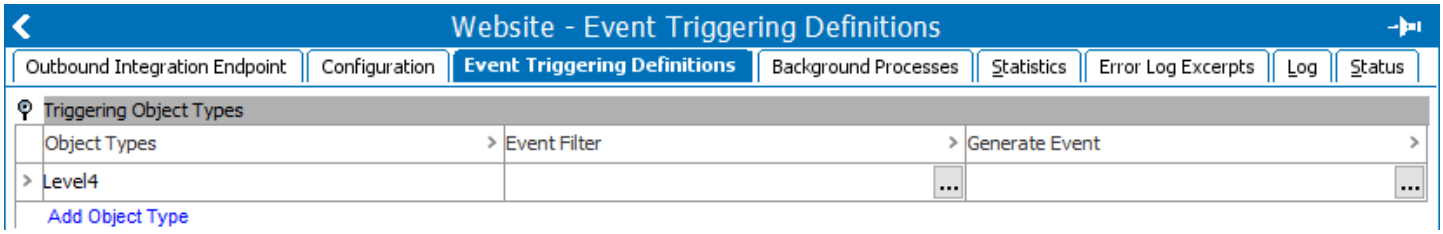
Event triggering definitions define what changes (events) are important and should be monitored by the OIEP or EP. For an OIEP, changes to any element defined on this tab are then formatted based on the output template and ultimately queued for export. For an EP, changes to any element defined on this tab are then handled by the selected core processor. For details about event generation, refer to the **Events** section of the **System Setup** documentation.

For event-based OIEPs or EPs, the Event Triggering Definitions tab allows you to add or edit the available triggers. At a minimum, you must set an Object Type trigger before the OIEP or the EP can be enabled to queue events. Additional triggers can be added to further limit the data monitored by the OIEP. For more information, refer to the flipper section for each trigger below.

Important: For an OIEP, an output template must be created for each event trigger, otherwise the triggered data will not be exported. For more information, refer to **OIEP - Event-Based - Output Templates Flipper**.



Triggering Object Types Flipper



Triggering object types are used to filter the data being monitored by the OIEP. For example, if you are only interested in exporting data when a change is made on a specific object type, there is no reason for the OIEP to listen to events happening on other object types. Setting a trigger on only the required object type reduces the load on the OIEP.

Consider the following during configuration:

- If functionality should vary by object type, a unique endpoint should be created for each object type.
- Common setup is to include at least one attribute or attribute group unless the integration endpoint handles System Setup data only or Create / Delete events only.
- If standard object types (product, entity, classification) are selected as triggers and the output template includes a Modify event, at least one attribute or attribute group must be selected to trigger the Modify event.

Object Types

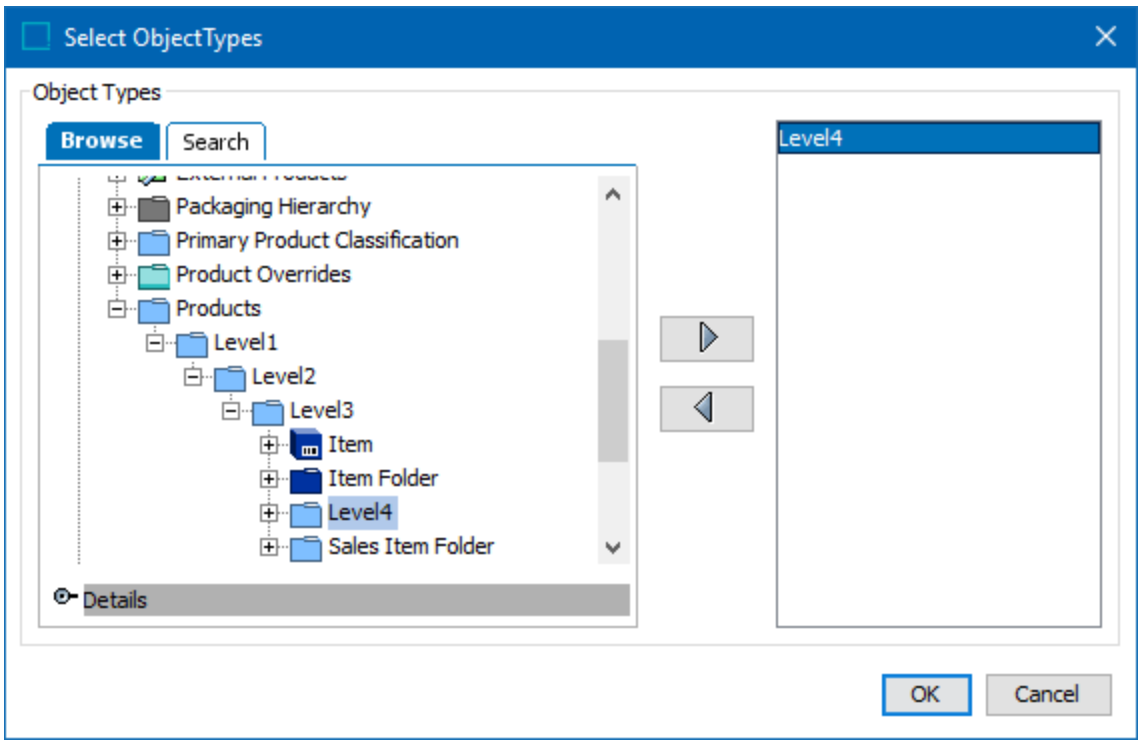
The Triggering Object Types section allows you to identify object type(s) that are important for this OIEP. The OIEP will listen for events on these object types to generate the desired output as defined by the output template configuration. When listening for events on changes to objects like attributes, the object type for these must be selected as well. All object types can be added in a single row unless business rules are used to filter and process events (functionality described below).

Keep in mind that events are triggered based on the revisability setting of the object type. Global revisable events are triggered on a change transaction. Workspace revisable events are triggered on approval. For more information, refer to the **Events** section of the **System Setup** documentation.

Important: The OIEP cannot be enabled or queue events until at least one object type is set as a trigger.

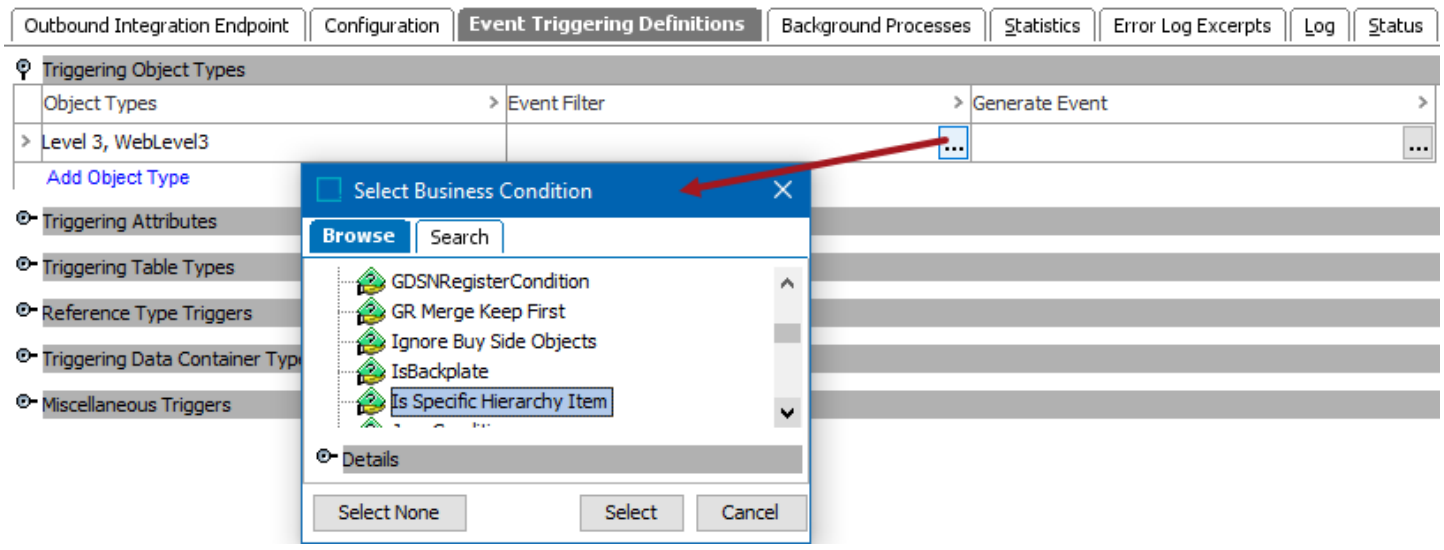
To select the object types:

- Click the **Add Object Type** link, then browse or search for the relevant object type.
- While the most commonly used object types are displayed in the browsing area, use Search to find other object types.
- Any System Setup data (references, attributes, units, etc.) can be selected.
- At least one object type **must** be selected.



Event Filter

Event filters are business conditions that can be added for each row of the Object Type Triggering section and can be used to apply additional event filtering. Event filters are run after any required approval, if applicable, and can be used in place of a pre-processor. Despite being created on an OIEP, an event filter runs at the time of event generation. If the condition evaluates to true, the event is written and registered on the event queue. When the condition evaluates to false, the event is not written at all unless a corresponding derived event is generated.



For example, when a change happens on a parent object, but only the children of this parent need to be output, a business condition can be created to filter out the parent. Then, a business action (refer to the Generate Event section below) can trigger an export of the children.

Conditions used as event filters are typically written in JavaScript since the built-in condition options, in most cases, will not provide the required flexibility.

For JavaScript based event filter conditions, the following available binds are of special relevance:

- **Event Type** binds in a derived event so you can check if the event currently being handed is of a specific derived event type.
- **Current Event Type** tests if the event currently being handed is a BasicEventType or a DerivedEventType. Assuming that 'Current Event Type' is bound to the variable 'currentEventType':

```
if (currentEventType instanceof
com.stibo.core.domain.eventqueue.BasicEventType) {
// It is a core event
}
if (currentEventType.equals
(com.stibo.core.domain.eventqueue.BasicEventType.Modify)) {
// It is a core Modify event
}
```

  You can also test derived events and if you have created an 'Event Type' binding, test if the current event is of a specific type. Assuming that 'Current Event Type' is bound to the variable 'currentEventType' and the derived event 'webModify' is bound to the variable 'webModify':

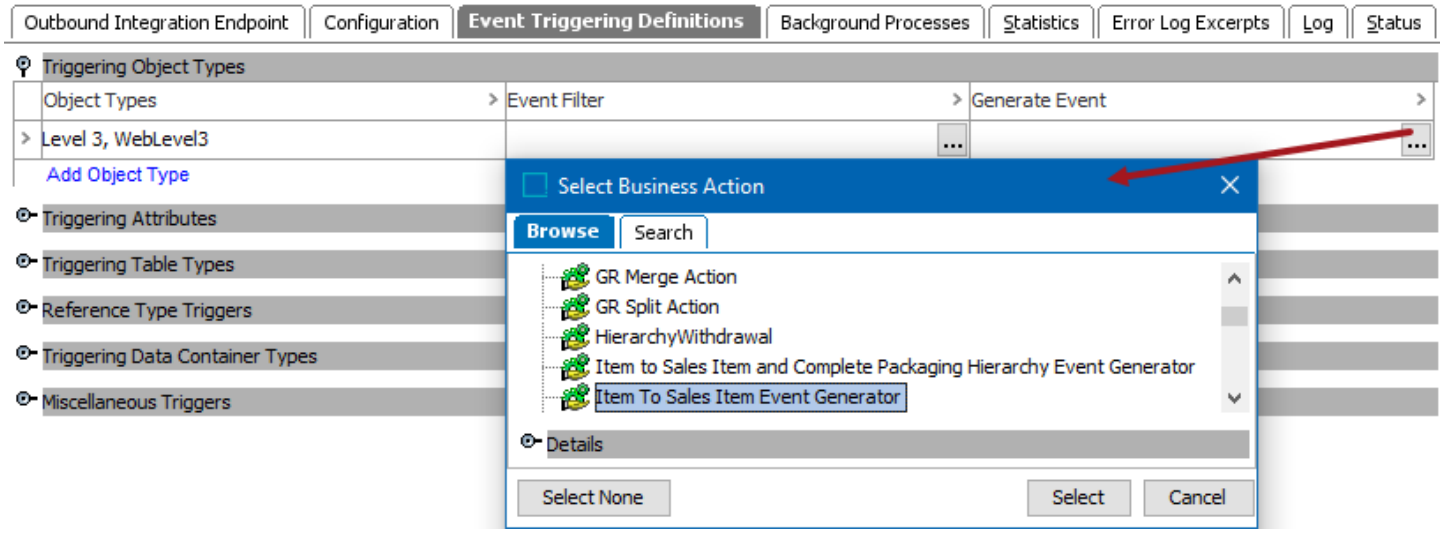
```
if (currentEventType instanceof
com.stibo.core.domain.eventqueue.DerivedEventType) {
// It is a derived event
}
if (currentEventType.equals(webModify)) {
// It is a derived WebModify event
}
```

- **Approve Context** lets you compare an object in Main and Approved before the actual synchronization is carried out. It is important to understand that event filter conditions are evaluated immediately when events are generated. If the event is generated based on an approval, in the event filter condition, you will have access to the Approved Context. Refer to the **Approve Context Bind** section of the **Resource Materials** online help or the 'ApprovePlugin.ApproveContext' interface in the Public Java API JavaDoc.

For more information, refer to **Event-Based OIEP Example Business Rules for Derived Events**.

Generate Event

Event generators are business actions that run after approval (when approval is required) and can be added for each row of the object type triggering section. The purpose of an event generator is to produce derived events based on other events (most often core events). New core events cannot be generated. For more information, refer to **Derived Events** of the **System Setup** documentation.



Event generators have access to the current event regardless of the chosen event filter, and an event filter is not required. This means derived events can be generated from events discarded by the filter. As with event filters, the event generator logic is also executed immediately when an event is generated.

Generated actions must typically be formulated in JavaScript. In addition to the binds mentioned in the event filter section above, the following are of special relevance for event generators.

- **Current Event Queue** lets you bind in the current event queue so that you can put new derived events on the queue (after filtering). Assuming that 'Current Event Queue' is bound to 'currentEventQueue', the derived event type 'webModify' is bound to the variable 'webModify', and 'Current Object' is bound to 'currentObject', an event can be generated as follows:

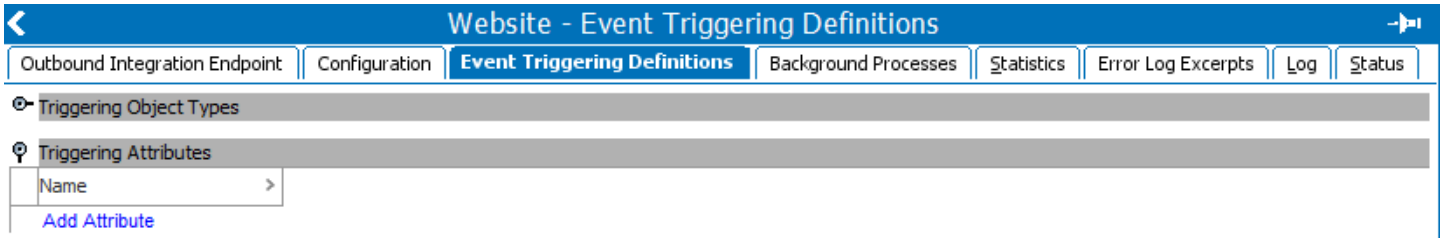
```
currentEventQueue.queueDerivedEvent(webModify, currentObject);
```

- **Event Queue** lets you place an event on a queue different from current. Use this bind if you produce events from actions not hooked in as generators.

Important: Avoid creating an endless loop by considering that any derived event generated by an event generator for the current event queue will pass through the object type filter and any event filters and event generators again.

For more information, refer to **Event-Based OIEP Example Business Rules for Derived Events**.

Triggering Attributes Flipper

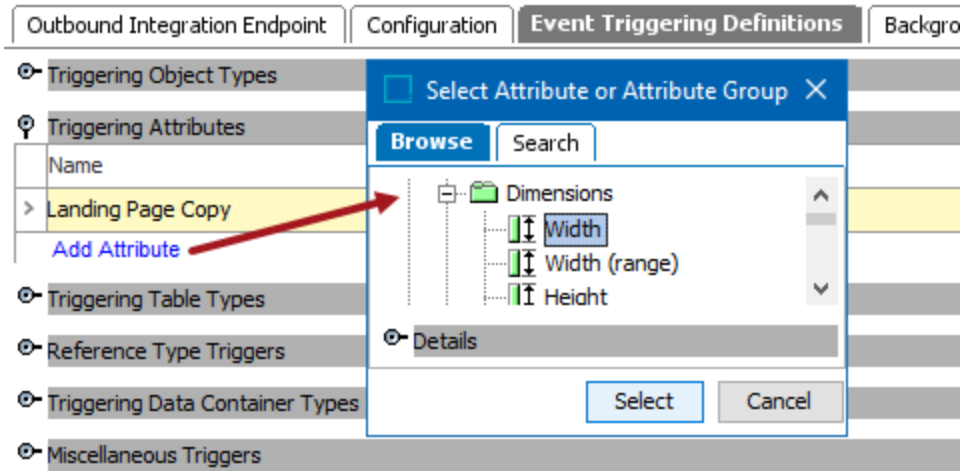


Triggering Attributes are used to filter the data being monitored by the OIEP in conjunction with the Triggering Object Types section. When a Triggering Attribute is defined, and that attribute's value changes for objects of the object types selected in the Triggering Object Types section, events are registered. The 'Triggering Attributes' section is a positive list, so if no attributes are selected, events will not be registered when, for example, a value changes and the object holding the value is approved.

Be aware that you can select both revisable and externally maintained attributes, and events are generated as follows:

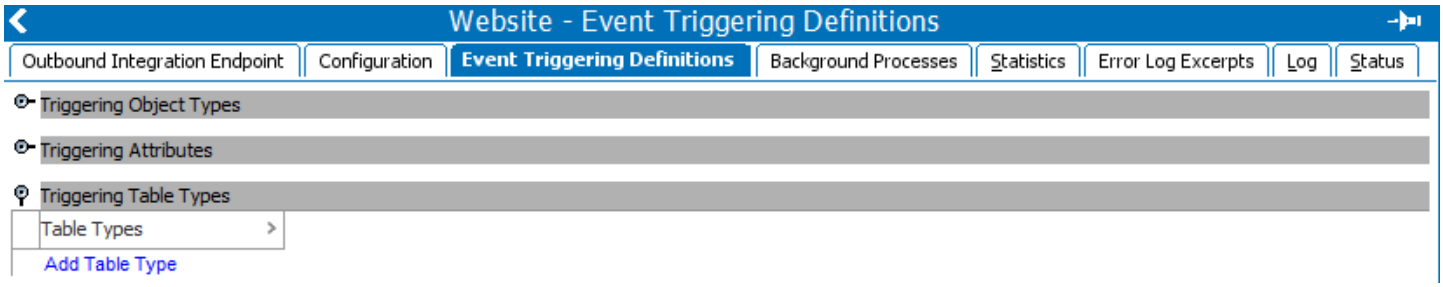
- Changes to values for externally maintained attributes and global revisable entities cause an event to be generated immediately.
- Changes to values for revisable attributes cause an event to be generated when the object holding the values is approved.

For more information, refer to the **Events** section of the **System Setup** documentation.

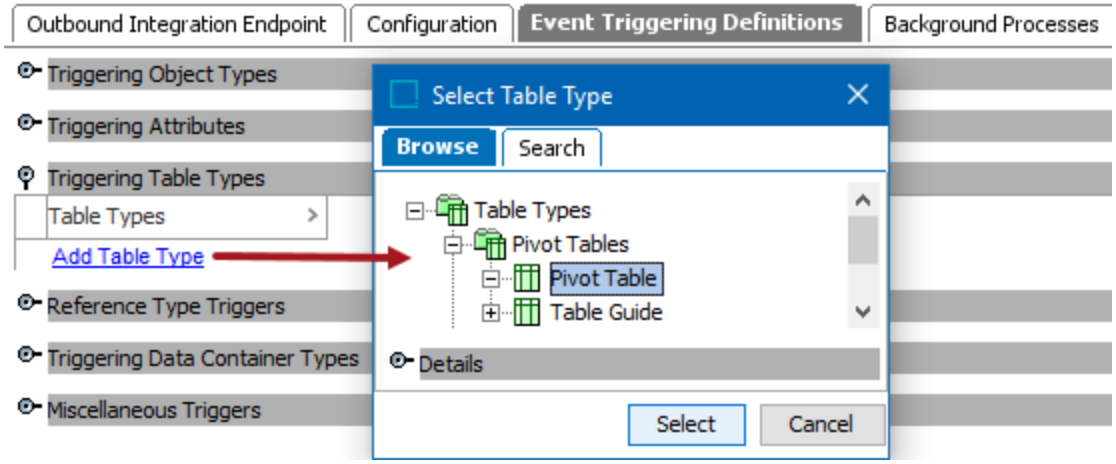


- Click **Add Attribute**, then browse or search for the relevant attribute(s) and/or attribute group(s).
- At least one attribute or attribute group must be selected, unless the integration handles ONLY System Setup data OR Create / Delete events.
- When an attribute or attribute group is selected as a trigger, an output template must be configured with a Modify event or the endpoint will skip the event since it cannot find a corresponding template. For more information, refer to the **OIEP - Event-Based - Output Templates Flipper**.

Triggering Table Types Flipper

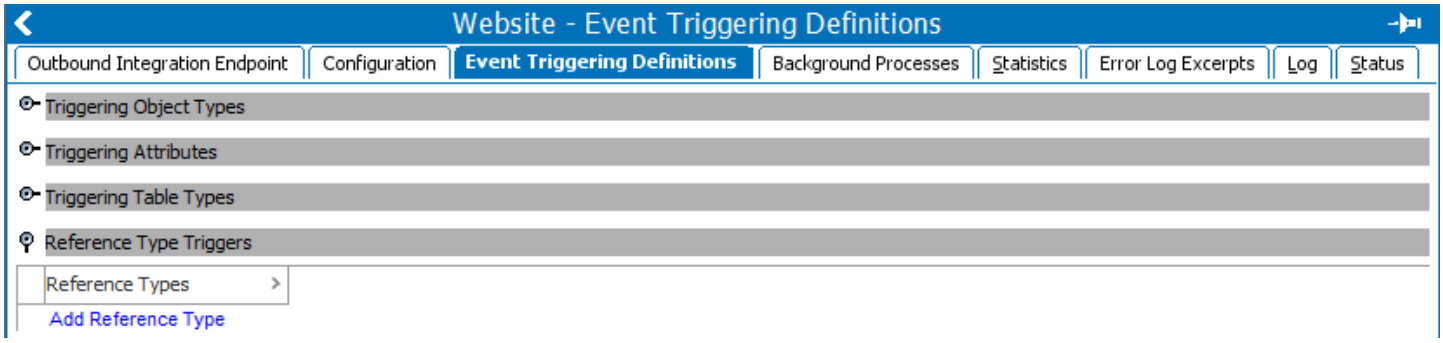


The Trigger Table Types flipper allows you to specify (filter) table types for which you want events registered when tables owned by objects of the type are listed in the Triggering Object Types section change. Common setup is to use tables, and this setting serves only for print output.



- Click **Add Table Type**, then browse or search for the relevant table type. Modifications to the table configuration on the product. For example, adding, removing, or modifying the table rows or columns; or changing the data on a free text cell followed by an approval will trigger an event for the endpoint.

Reference Type Triggers Flipper

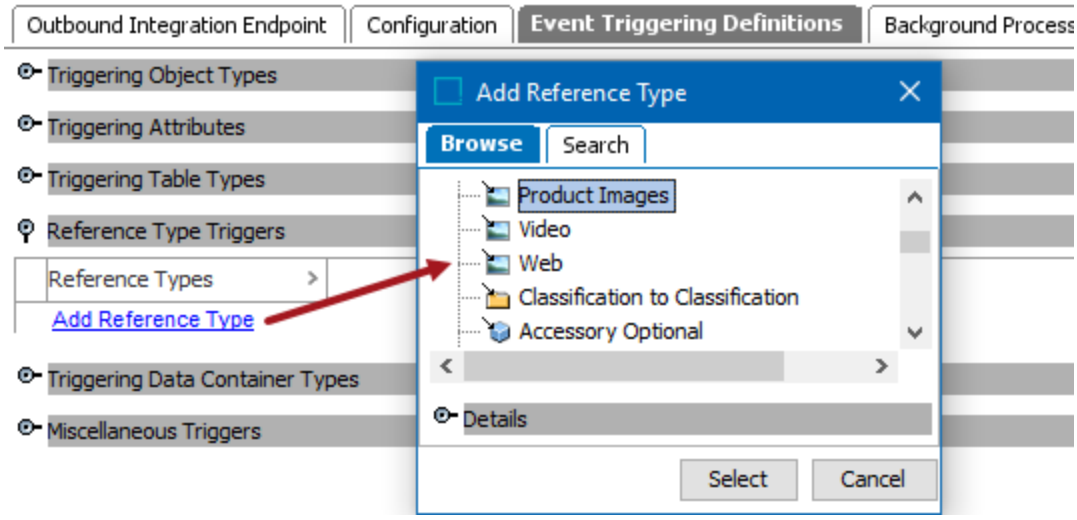


Use reference / link types to define (filter) which reference / link types to listen to for changes. This is a positive list so only the items listed are triggers. If functionality should vary by object type, a unique endpoint should be created for each object type.

Be aware that you can select both revisable and externally maintained reference / link types and events are generated as follows:

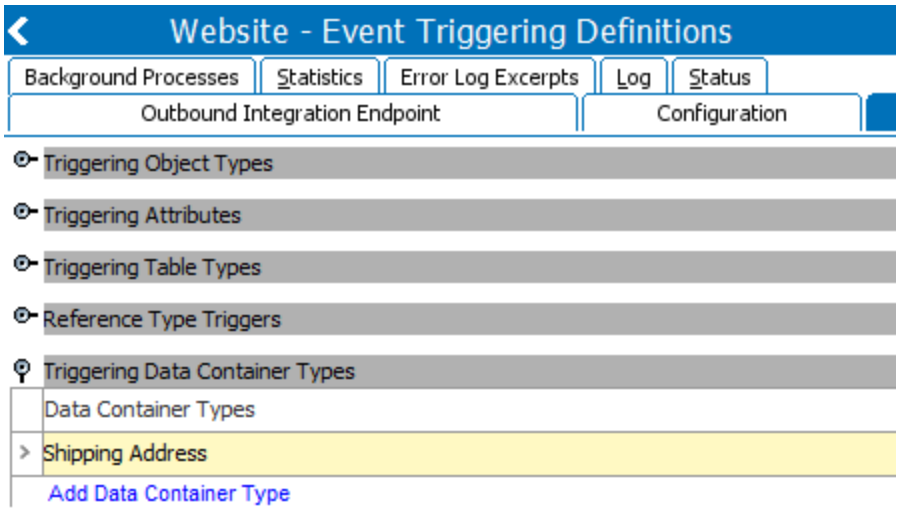
- Changes to values for externally maintained reference / link types cause an event to be generated immediately if an object type on which the reference is valid is selected as a triggering object type.
- Changes to values for revisable reference / link types cause an event to be generated when the object holding the values is approved if an object type on which the reference is valid is selected as a triggering object type.

For more information, refer to the **Events** section of the **System Setup** documentation.

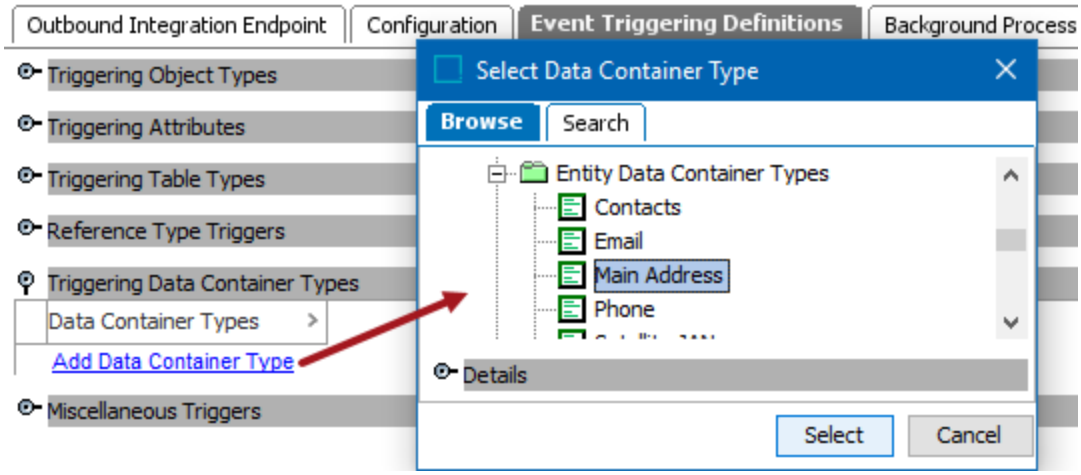


- Click **Add Reference Type**, then browse or search for the relevant reference type.

Triggering Data Container Types

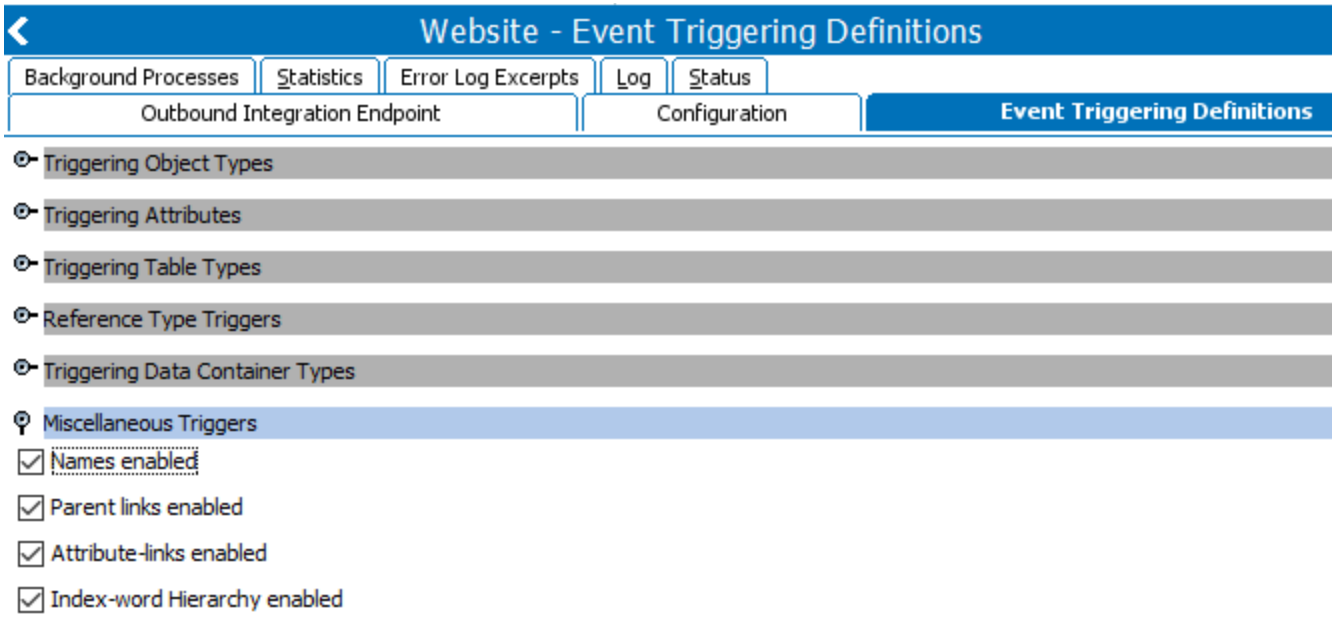


When monitoring changes on entity objects, specific data container types can be configured to trigger events upon creation / deletion of corresponding data containers. Any data container types omitted from the triggering definition will not trigger events.



- Click **Add Data Container Type**, then browse or search for the relevant data container type. When a data container of this type is created / deleted, it will trigger an event for the endpoint.

Miscellaneous Triggers Flipper



Miscellaneous triggers complement the triggers set in the other flippers. By default, the following miscellaneous triggers are set to additionally filter the data being monitored by the OIEP:

- **Names enabled**: Triggers an event when a change is made to the object name of triggering object types for a global revisable object, or when a workspace revisable object is approved. An output template must be configured with a Modify event or the endpoint will skip the event since it cannot find a corresponding template.
- **Parent links enabled**: Triggers an event when changes are made to the parentage of triggering object types for a global revisable object, or when a workspace revisable object is approved. For example, when a product

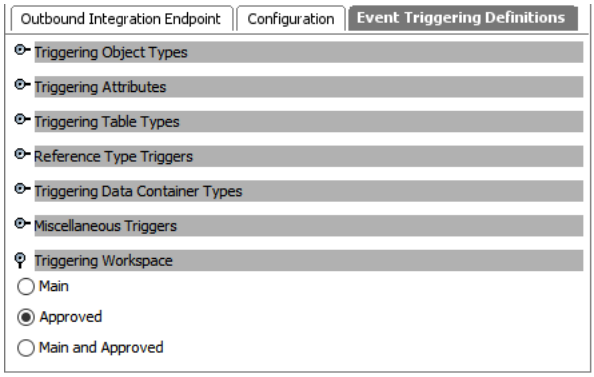
or asset is moved into another parent folder. An output template must be configured with a Modify event or the endpoint will skip the event since it cannot find a corresponding template.

- **Attribute-links enabled**: Triggers an event when attribute hierarchy link is changed on an object of the types for a global revisable object, or when a workspace revisable object is approved. For example, when an attribute is linked to a product or classification.
- **Index-word hierarchy enabled**: Triggers an event when an index word changes, or an index hierarchy link is added on a triggering object type. Common setup is to use indexes, and this option for print output.

Triggering Workspace Flipper

This flipper is displayed in rare cases, specifically, only if either the 'Main' or the 'Main and Approved' option was previously selected in a release where this option was available. Changing the selection to the default 'Approved' radio button removes the flipper from the OIEP without the ability to reconfigure triggering events from either the 'Main' or the 'Main and Approved' since triggering events from Main is not advised when using OIEPs.

By default, events are triggered on the Approved workspace. Derived event functionality is available for triggering events prior to approval, as defined in the **Derived Events** topic of the **System Setup** documentation.



Important: Triggering from the 'Main' or 'Main and Approved' workspace can negatively impact performance and is not generally recommended for OIEPs. An alternate approach to triggering events from the main workspace is to use Derived Events to trigger objects after a group of changes are made by a user or integration / import.

For OIEPs created prior to release 11.0, once the 'Approved' radio button is selected, the flipper is no longer available since 'Approved' is the recommended setting.

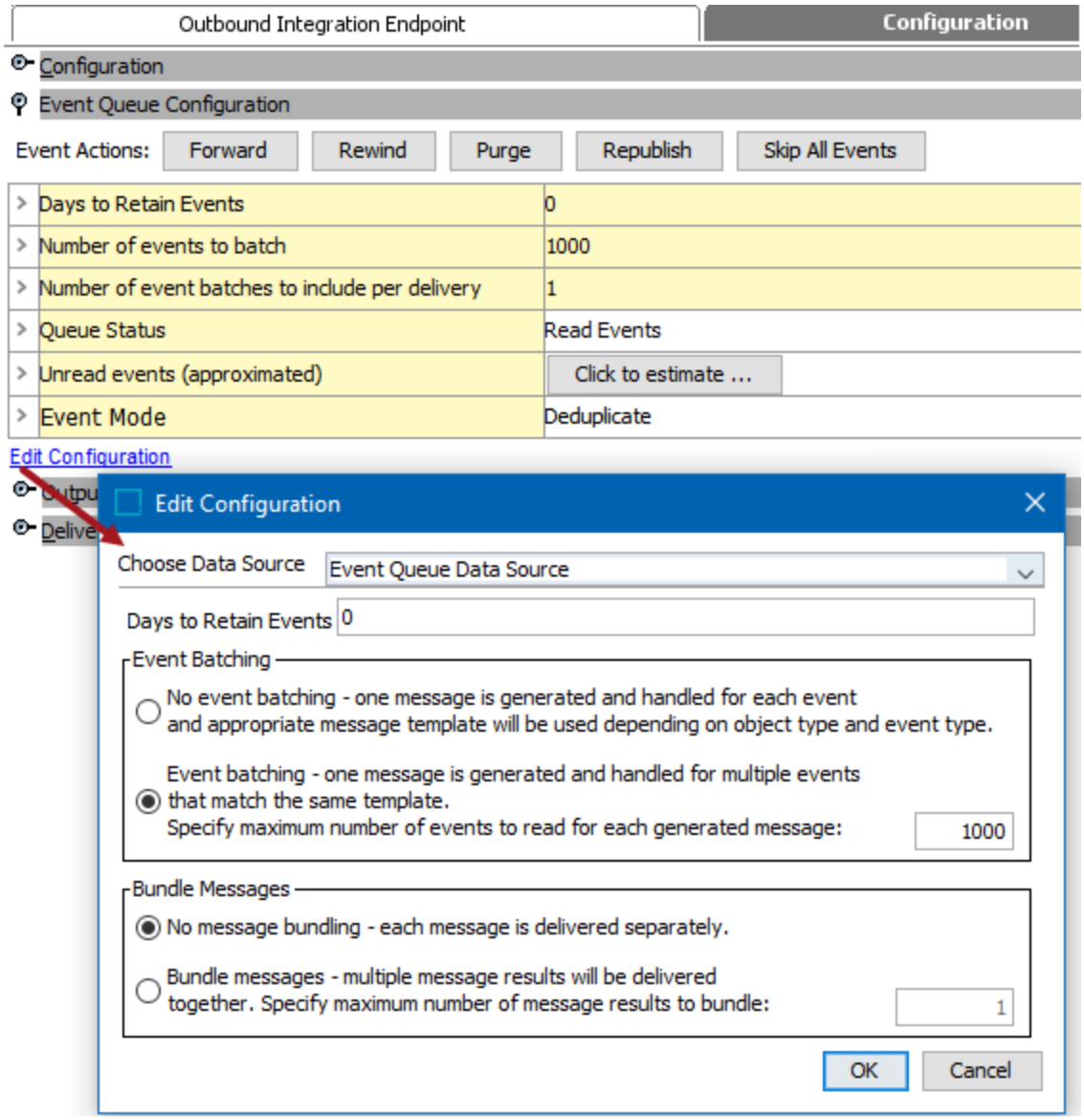
- The recommended option 'Approved' creates events when an action causes an update to the Approved workspace, generally as a result of approving an object.
- The 'Main' option creates an event when a change is made in the Main (maintenance) workspace, prior to approval, and can have a negative performance impact.
- The 'Main and Approved' option results in events both before and after approval and can have a negative performance impact.

Note: To activate a change to the Triggering Workspace parameter, you must invoke the event processor with at least one event on the queue.

OIEP - Event-Based - Event Queue Configuration Flipper

On the OIEP Configuration tab, the Event Queue Configuration flipper is only displayed when the selected OIEP uses the Event Queue data source, as defined on the Choose Data Source step of the wizard. For more information, refer to **OIEP - Event-Based - Choose Data Source**.

Click the 'Edit Configuration' link to open the **Edit Configuration** dialog.



- For information on the **Event Actions** buttons and the **Days to Retain Events** parameter, refer to **Event-Based OIEP Event Actions**.

- For information on the **Number of events to batch** and **Number of event batches to include per delivery** (which are controlled by the Event Batching and Bundle Messages parameters on the Edit Configuration dialog), refer to **Event-Based OIEP Event Batching**.
- For information on the **Queue Status** parameter, refer to **Event-Based OIEP Status and Queue Status**.
- For information on the **Unread events (approximated)** parameter, refer to **Event-Based OIEP Queued Events**.

Creating a Select Objects Outbound Integration Endpoint

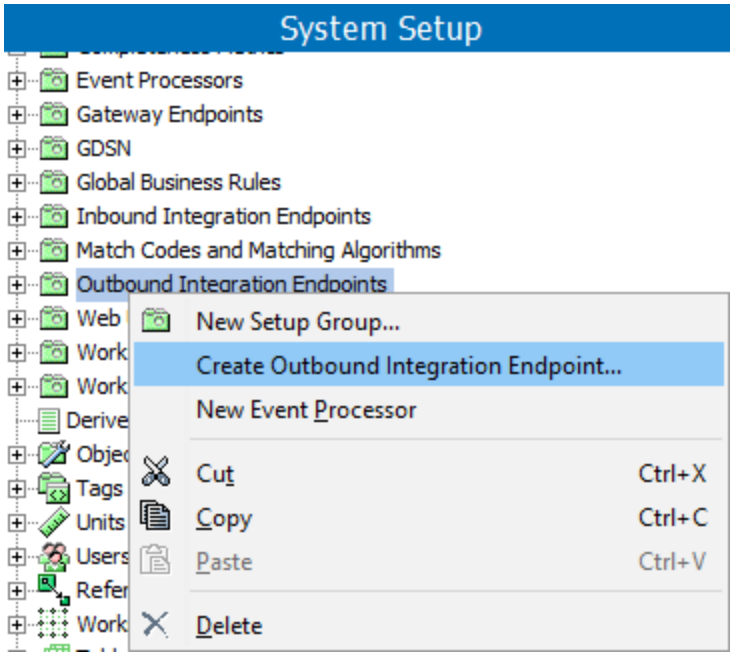
A Select Object OIEP publishes data non-incrementally from selected hierarchies (one or more root nodes) to external systems on scheduled intervals. This type of OIEP uses a static set of data and works exactly like a scheduled export created in Export Manager, with the differences being that the OIEP has more standard delivery options and offers extended monitoring capabilities.

For example, a Select Object OIEP can be used to send a specific set of data to an external system on regular intervals. Configuration of the output templates allow the same data to be delivered at the same time, using multiple formats, a pre-processor, or post-processor.

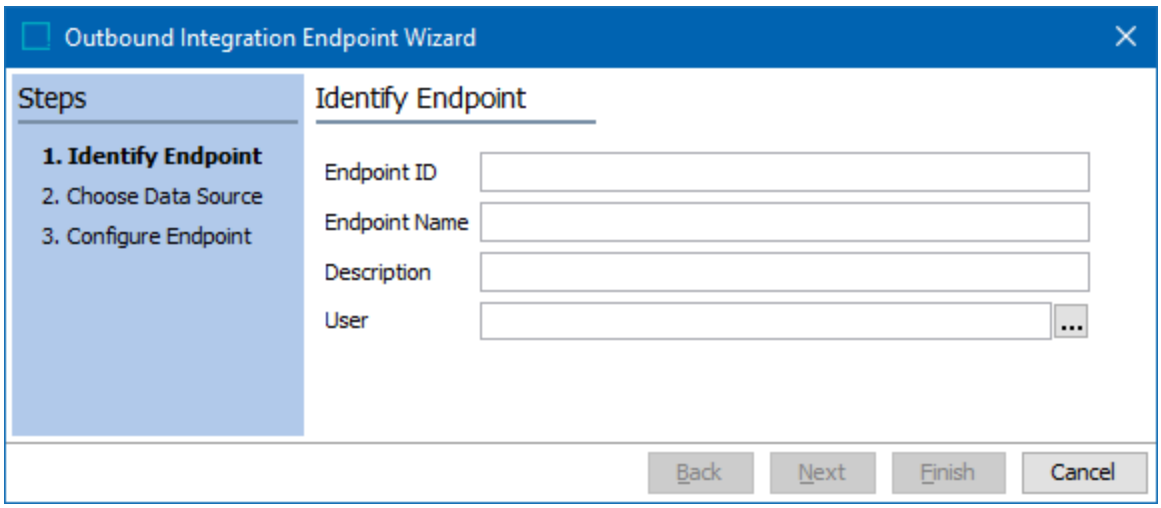
Select Object OIEP Setup

After creating a setup group for outbound integration endpoints, create a Select Object outbound integration endpoint to send a static set of data from STEP. For information on an event-based OIEP, that can monitor STEP for changes and export as needed, refer to **Creating an Event-Based Outbound Integration Endpoint**.

1. In System Setup, right-click the Outbound Integrations Endpoints setup group, and click **Create Outbound Integration Endpoint**.

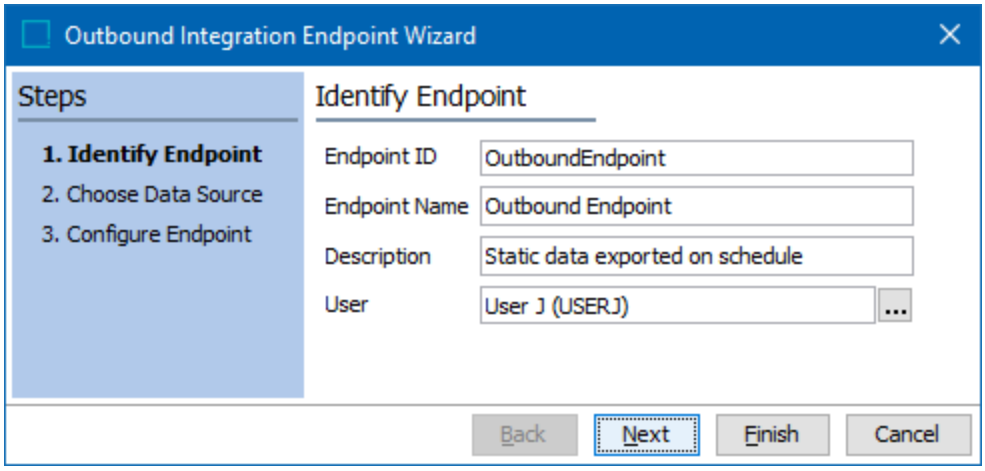


2. The Outbound Integration Endpoint wizard displays and involves the following steps:



3. Complete setup requires providing data in the wizard as well as some manual configuration. All setup includes the following sections:
   - In the wizard, **OIEP - Select Objects - Identify Endpoint** allows you to specify the name and ID of the endpoint and identify the user whose privileges are applied.
   - In the wizard, **OIEP - Select Objects - Choose Data Source** allows you to specify a static set of data or an event queue.
   - In the wizard, **OIEP - Select Objects - Configure Endpoint** allows you to specify processing and context settings for the integration endpoint.
   - In the OIEP editor, **OIEP - Select Objects - Manual Configuration** directs you to complete the required manual setup.

OIEP - Select Objects - Identify Endpoint



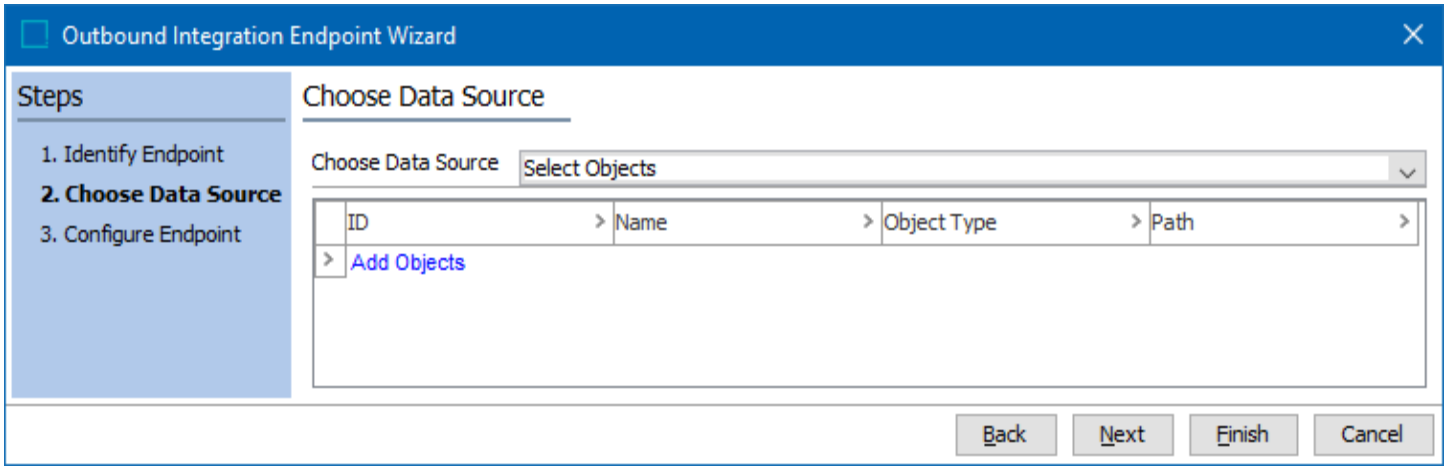
1. For **Endpoint ID**, enter an ID. Common setup is to use no spaces or punctuation.
2. For **Endpoint Name**, enter a **Name**. Common setup is to repeat the ID with added spaces for readability.
3. For **Description**, enter an optional **Description**. Common setup is to include a description for future reference.
4. For **User**, type a user name into the field, or click the ellipsis button (...) to search or browse for a user. The privileges of the selected user determine which actions the integration endpoint can perform and what data can be processed. Common setup is to create a special system user for this purpose so that the effects of the endpoint are easily identified.

   Once the Endpoint ID, Endpoint Name, and User are supplied, the Next button is enabled.

Note: Attempting to delete a user who is selected in the User parameter for an OIEP will result in errors. For more information, refer to the **Working with Users** topic in the **System Setup** documentation.

5. Click **Next** to display **OIEP - Select Objects - Choose Data Source**.

OIEP - Select Objects - Choose Data Source

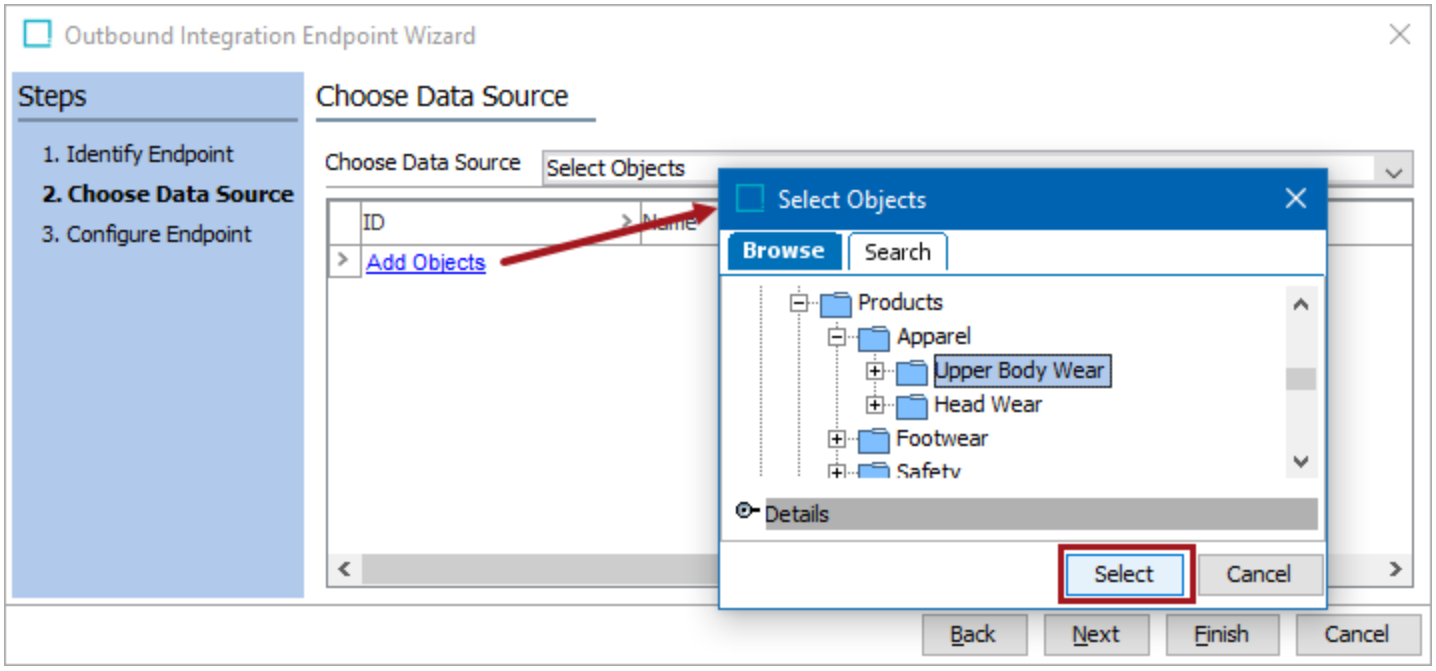


1. Choose the data source **Select Objects** for non-incremental exports where you want to publish data for specific hierarchies every time the endpoint is invoked.

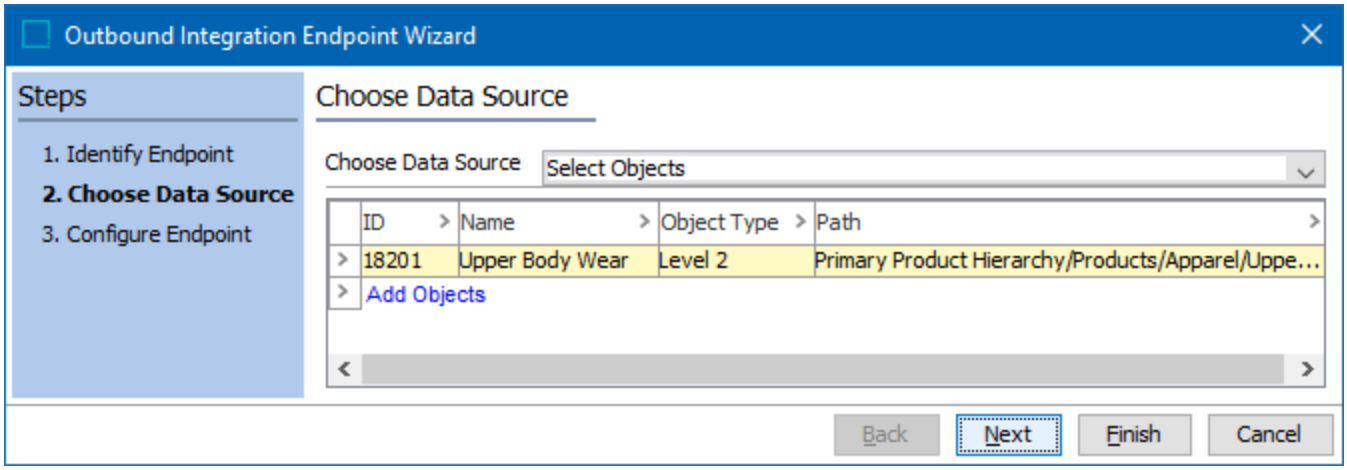
   Note: An **Event Queue Data Source** requires a separate setup. For detailed steps, refer to the **Creating an Outbound Integration Endpoint** documentation.

2. Click the **Add Objects** link to display the Select Objects dialog. Select one or more root node objects from the Tree hierarchies to be exported and click the **Select** button.

   Note: Selecting a Tree hierarchy object is not required if the endpoint should export system-specific data, such as attributes, LOVs, and units that are not used in any specific product, classification, or entity hierarchy.

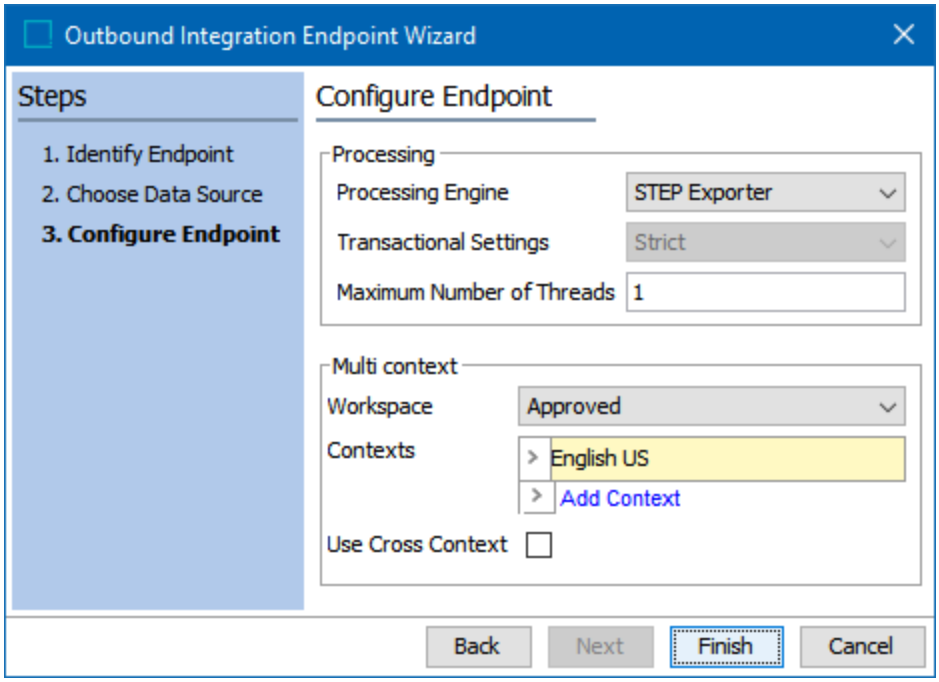


The selection(s) are displayed as a data source.

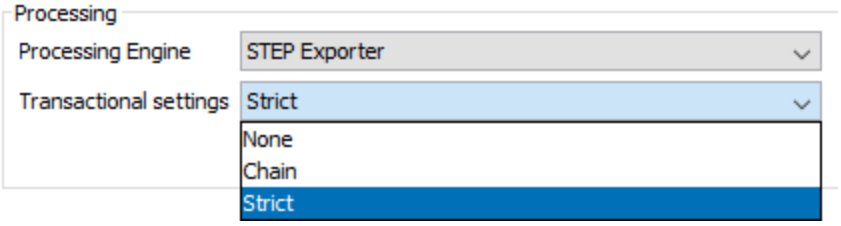


3. Click **Next** to display **OIEP - Select Objects - Configure Endpoint**.

OIEP - Select Objects - Configure Endpoint



1. For the **Processing Engine** parameter, select an option from the dropdown:
   - **STEP Exporter** uses the same functionality as the Export Manager. This is the only processing engine option on a standard STEP system and performs the actual data export.
   - **Business Rule Based Message Processor** allows you to export STEP data using either JavaScript-based business actions, or Java business actions developed via the Extension API. For more information, refer to the **OIEP - Configuration Flipper for Business Rule Based Message Processor** topic or click the **STEP API Documentation** button on the Start Page.
   - If you need a customized processing engine, contact your Stibo Systems account manager or partner manager. For more information, refer to **Outbound Integration Endpoint Structure** documentation.
2. The **Transactional Settings** option can be set for select objects OIEPs. For more information about Transactions Settings, refer to the **Integration Endpoint Transactional Settings** documentation.

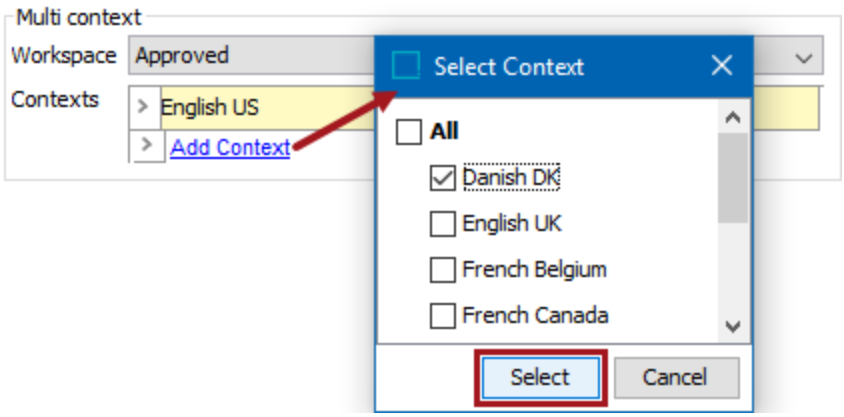


3. The **Maximum Number of Threads** option is only valid for an Event Queue Data Source endpoint.
4. Set the **Multi context** options that the endpoint should use to export data. By default, the Workspace and Context selected is the one in which the outbound integration endpoint has been created.

- For **Workspace**, use the dropdown to select the desired workspace. Common setup is to use the Approved workspace, except when you need to generate events for objects before they are approved, for example, during import or from a workflow.
- For **Contexts**, when exporting data in the STEPXML format or when using a custom cross-context enabled format, one file can contain data from multiple contexts. For other formats, the standard 'Context splitter Post-processor' should be used to generate separate files for the selected contexts.

  Click the **Add Context** link and choose one or more contexts. You can only pick more than one context if you have selected / enabled the **Use Cross Context** option. It is possible to also use cross context mode for endpoints with only a single context selected. This means that it will not be necessary to change downstream systems should more contexts be added later. If 'Use Cross Context' is not picked during setup, it will not be possible to add more contexts without explicitly changing the export mode first in the Configuration tab of the OIEP editor.

  Click the **Select** button to add them to the Contexts list.



5. Click **Finish** to complete the wizard.
6. Perform the manual configuration described in **OIEP - Select Objects - Manual Configuration**.

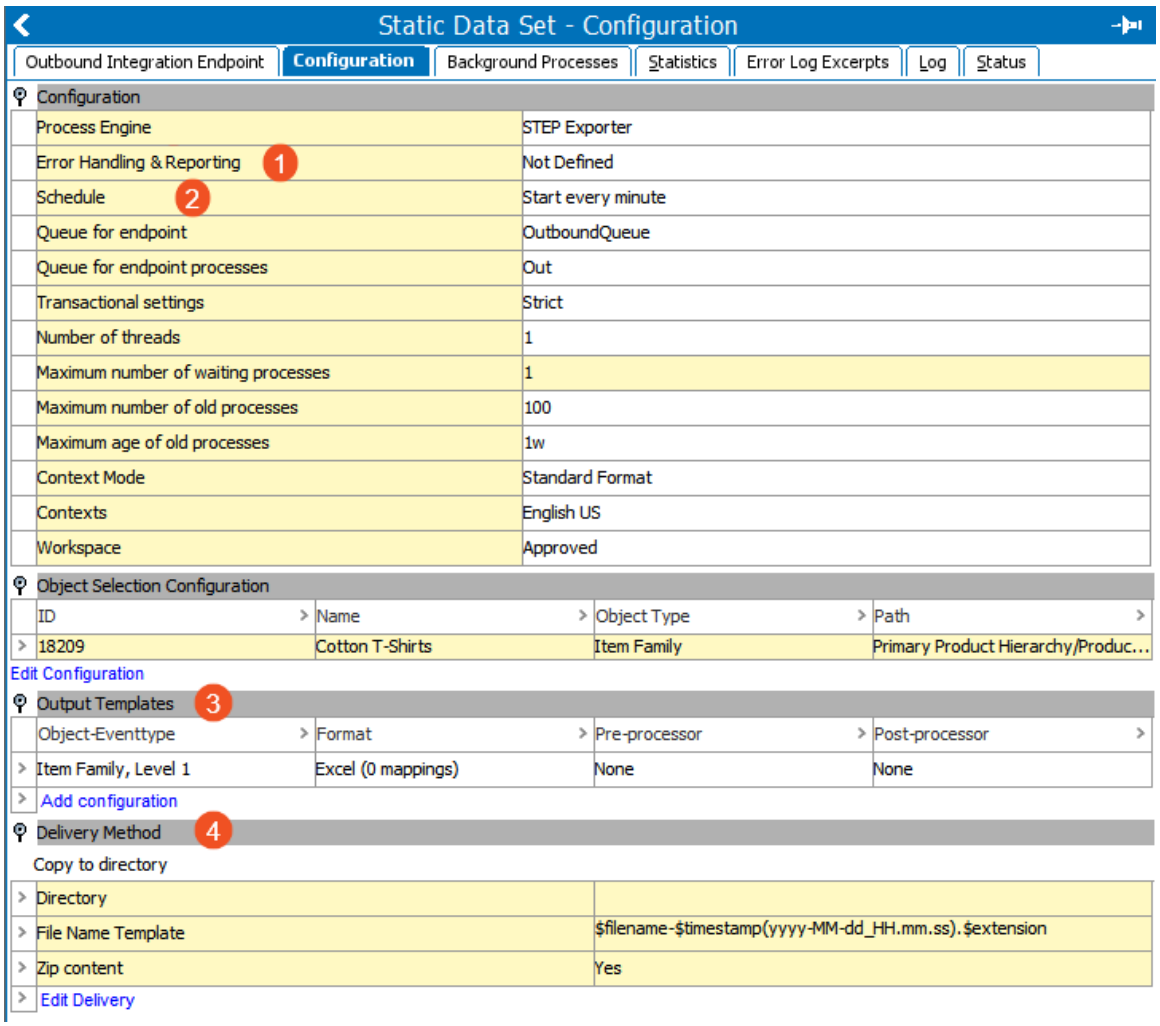
OIEP - Select Objects - Manual Configuration

After completing the Outbound Integration Endpoint wizard, for a Select Objects OIEP using the STEP Exporter process engine, the OIEP editor includes additional parameters that must be set manually before the OIEP can run.

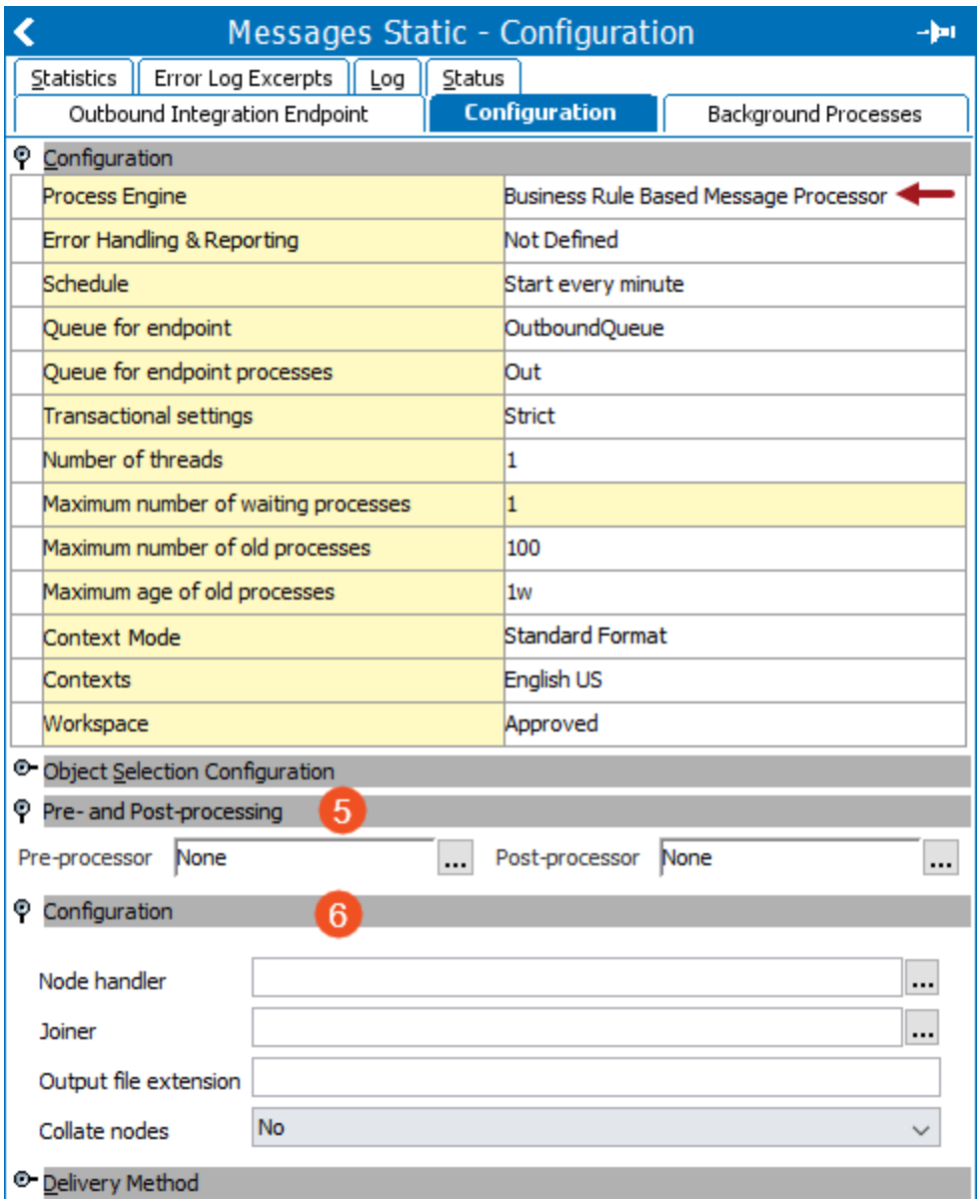
> **Note:** In addition to the set up in this topic, additional manual configuration is required when using the Business Rule Based Message Processor, as defined in the **OIEP - Configuration Flipper for Business Rule Based Message Processor** topic.

To manually edit or review other parameter settings on an OIEP, refer to the **Maintaining an Outbound Integration Endpoint** documentation.

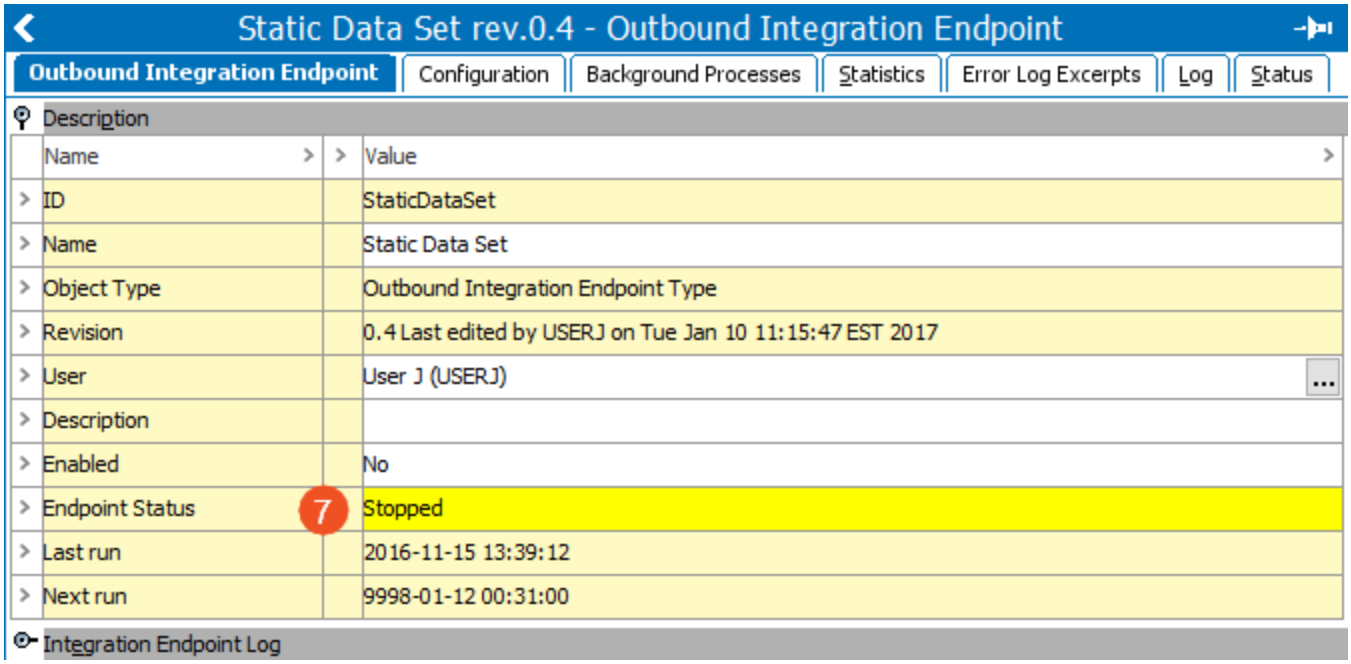
1. In the OIEP editor, configure to send an email if an endpoint-related background process fails as described in the **Error Handling & Reporting** section of the **OIEP - Configuration Flipper** documentation.
2. In the OIEP editor, determine when the OIEP runs as described in the **Schedule** section of the **OIEP - Configuration Flipper** documentation.
3. In the OIEP editor, when using the STEP Exporter process engine, specify the objects to be output, the format, and pre- or post-processors (if any) as described in **OIEP - Select Objects - Output Templates Flipper**.
4. In the OIEP editor, determine how data is delivered as described in **OIEP - Delivery Method Flipper**.



5. In the OIEP editor, when using the Business Rule Based Message Processor process engine, specify the pre- or post-processors (if any) as described in **OIEP - Pre and Post-processing Flipper**.
6. In the OIEP editor, when using the Business Rule Based Message Processor process engine, specify the business actions and settings as described in **OIEP - Configuration Flipper for Business Rule Based Message Processor**.

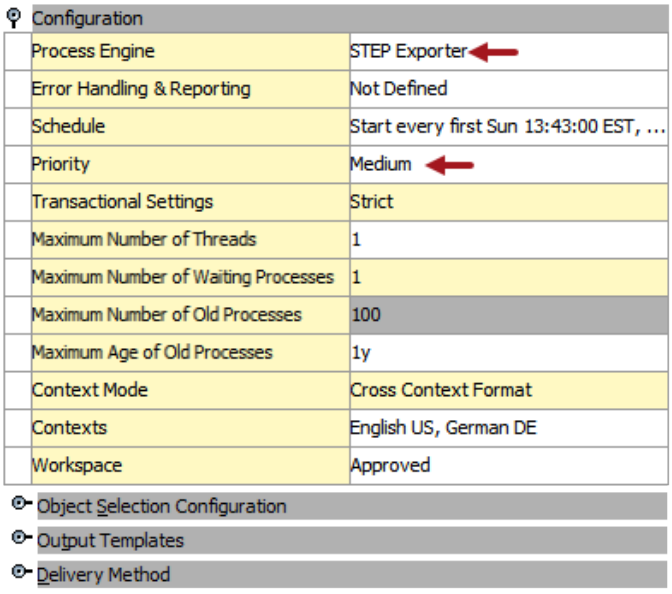


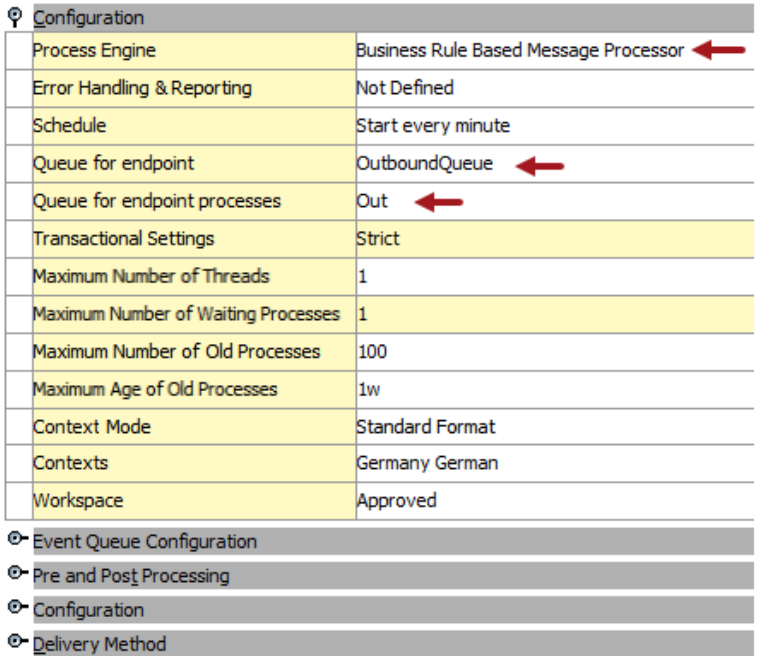
7. Enable the endpoint and invoke it as described in **Running an Outbound Integration Endpoint**.


Static Data Set rev.0.4 - Outbound Integration Endpoint
Outbound Integration Endpoint
Configuration
Background Processes
Statistics
Error Log Excerpts
Log
Status
Description
Name
Value
ID
StaticDataSet
Name
Static Data Set
Object Type
Outbound Integration Endpoint Type
Revision
0.4 Last edited by USERJ on Tue Jan 10 11:15:47 EST 2017
User
User J (USERJ)
Description
Enabled
No
Endpoint Status
7
Stopped
Last run
2016-11-15 13:39:12
Next run
9998-01-12 00:31:00
Integration Endpoint Log

OIEP - Configuration Flipper

The following flippers are available when the STEP Exporter processor or the Business Rule Based Message processor is selected for an OIEP. For both processors, the Configuration flipper includes the same parameters for both Event-Based and Select Objects endpoints. Each parameter is described below.





Note: When a GDSN processor or Business Rule Based Message Processor is selected for the Process Engine parameter, different or additional flippers are displayed.

Process Engine

STEP Exporter is the only processing engine option on a standard STEP system. If you need a customized processing engine, contact your Stibo Systems account manager or partner manager.

The processing engine is originally selected in the wizard when configuring the endpoint. For more information, refer to **OIEP - Event-Based - Configure Endpoint** documentation or **OIEP - Select Objects - Configure Endpoint** documentation.

Error Handling & Reporting

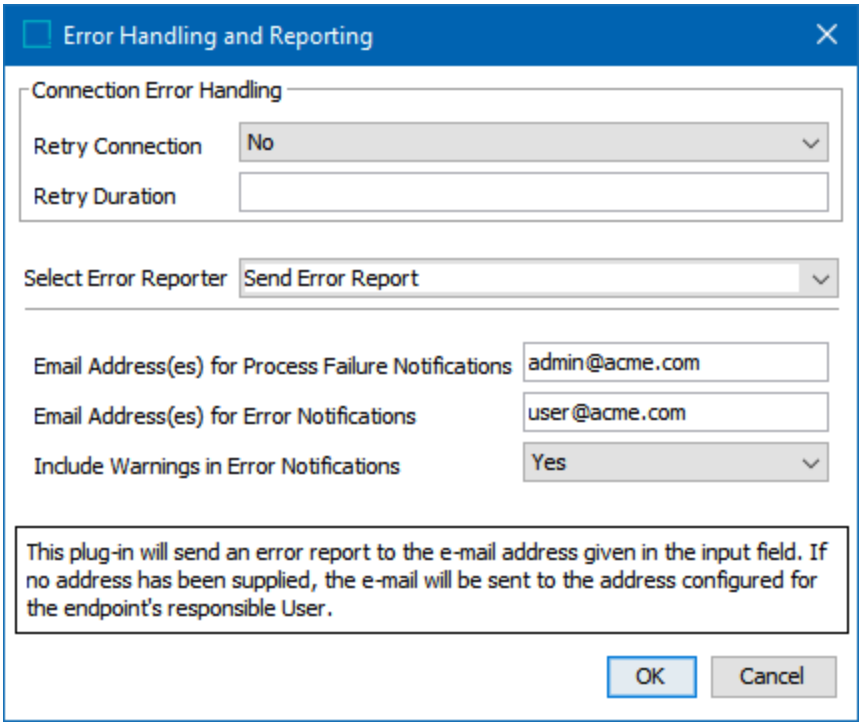
The 'Error Handling & Reporting' parameter is not included in the wizard, but it can be configured from the OIEP editor to send an error report to the specified email addresses if a background process fails or has error or warnings. The email alert contains information about the failed endpoint, including the server name, the background process, failed file, failing process step, and the cause of the error. A standard STEP system (without custom extension) includes only the 'Send Error Report' option.

For the Error reporter plugin to send email, the SMTP server must be configured on the application server. For information on configuring email from STEP, refer to the **Email from STEP** topic in the **Resource Materials** of online help.

Configure Error Handling and Reporting

1. Open the relevant OIEP on the Configuration tab, open the Configuration flipper and click into the **Error Handling & Reporting** parameter to display the ellipsis button (...).

2. Click the ellipsis button (...) to display the **Error Handling and Reporting** dialog.



3. For **Connection Error Handling**, the default is no connection error handling. The Retry Duration parameter is ignored when 'No' is set for the Retry Connection parameter. Enabling Connection Error Handling allows automated reconnection attempts when the external system is unavailable. When connection retries begin, a warning is logged to the Execution Report; if a connection cannot be established after the Retry Duration expires, an error message is logged to the Execution Report.

   - On the **Retry Connection** parameter, set 'Yes' to automate reconnection attempts for HTTP-based delivery methods including Amazon SQS, Cloud Blob Storage, Git, Kafka, REST, REST Direct, and SFTP. A 'No' setting requires manually restarting the OIEP if the connection fails.

     Note: Authentication-related connection errors are not retried and the OIEP fails immediately.

- On the **Retry Duration** parameter, when blank, the default or 30 days is applied. Valid settings are an integer plus one of the following time indicators: s (seconds), m (minutes), h (hours), and d (days). For example, 45m indicates that connection retries will continue for 45 minutes, during which time, the OIEP shows a 'Failed (retrying)' state. Multiple integers with time units in order of size may also be used. For example, 1h45m30s indicates that connection retries will continue for 1 hour, 45 minutes and 30 seconds. After the 'Retry Duration' expires, if the connection cannot be re-established with the last retry, the OIEP fails, and email alerts are distributed based on the 'Select Error Reporter' settings. For more information, refer to the **Running an Outbound Integration Endpoint** topic.

4. For **Select Error Reporter**, select an option:

- **Send Error Report**: Use the following parameters to add a relevant email address (or use a semi-colon to add more than one email address) or group email that can be used to receive error and warning reports. When emailing multiple accounts, consider creating a group email and use that email address in the parameters instead of adding multiple email addresses for individuals. With this configuration, a single email group can be updated as the group members change while the 'Select Error Reporter' parameter remains unchanged.

  Note: To send email, the SMTP server must be configured on the application server. For information on configuring email from STEP, refer to the **Email from STEP** topic in the **Resource Materials** of online help.

  The email alert contains information about the failed endpoint, including the server's name, the background process, failed file, failing process step, cause of the error, and a copy of the file that triggered the error.

  If no email address is entered, the alert(s) are sent to the email address of the user who created the integration endpoint. If no email is defined for this user, or if no mail server is defined in the configuration, the error report will be written to the failed Background Process Execution Log.

  - **Email Address(es) for Process Failure Notifications**: Sends an alert email to the listed address (es) when a fatal processing error occurs that puts the IIEP in a 'Failed' state.
  - **Email Address(es) for Error Notifications**: Sends an alert email to the listed address(es) when data warnings and/or errors occur that do not cause the process to stop.
  - **Include Warnings in Error Notifications**: A 'Yes' setting means alert emails include information on both warnings and errors. A 'No' setting means only error information is addressed in Error Notification emails. To receive an email when the IIEP enters the 'Failed (retrying)' state, set this option to 'Yes.'

- **No Error Report** disables the error reporter and no notifications by email are sent.

Note: If the error report is larger than the default maximum, 10 MB, the report will not be attached to the auto-generated email sent to the configured email address(es). If the default maximum is not suitable, an admin can set the following case-sensitive property in the sharedconfig.properties file on the application server to adjust the maximum file size:

`Integration.Endpoint.ErrorFileSizeLimit={MB size}`

Replace the '{MB size}' element with the maximum MB file size allowed for error report attachments. Error reports larger than this configured maximum are not attached to the email.

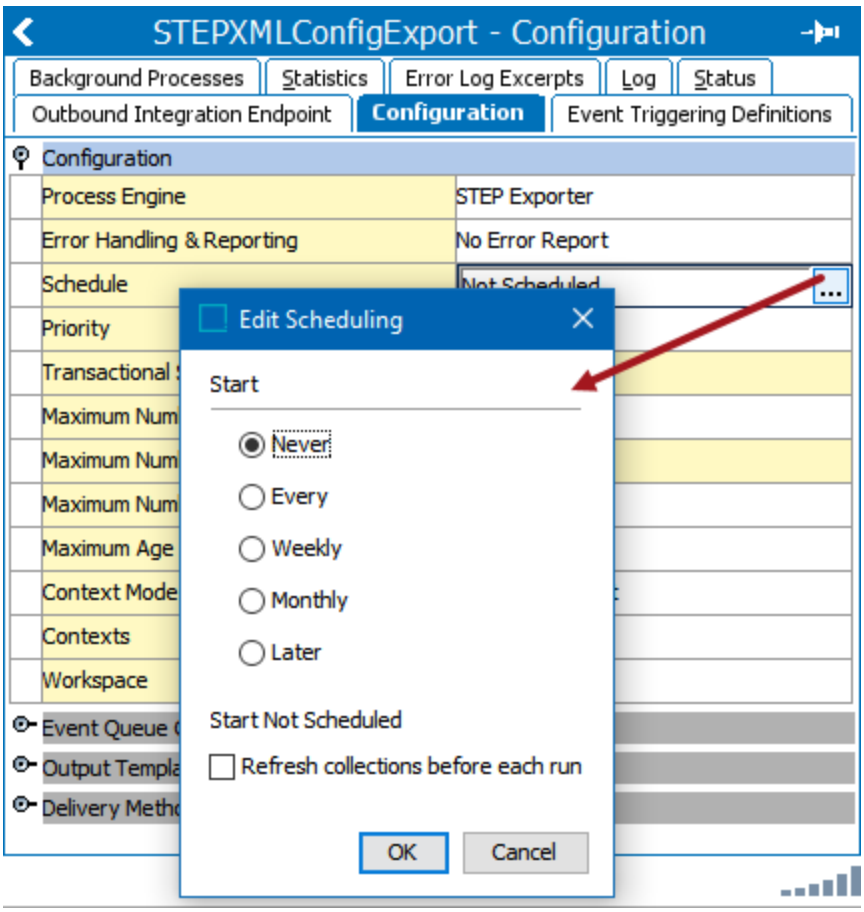
Schedule

The schedule parameter is not included in the wizard, but it can be configured to repeatedly run the OIEP. For Select Objects OIEPs, the background processes runs as scheduled, regardless of changes to the data. For Event-Based OIEPs, the background process runs as scheduled but only when events are available for processing.

Important: Consider the time zone of the application server compared to that of the workbench (the client) where the schedule is created or viewed. When scheduling a job, the local time zone is displayed in the workbench, but the time zone of the server is used to run the background process. Although displayed, the time zone of the client is not included in the instruction to the server to run the job. This can cause confusion about when the job will run since the scheduled time is not automatically converted to accommodate potential differences in time zones.

1. On the relevant OIEP, click the Configuration tab and open the Configuration flipper. Click into the **Schedule** parameter to display the ellipsis button (...).

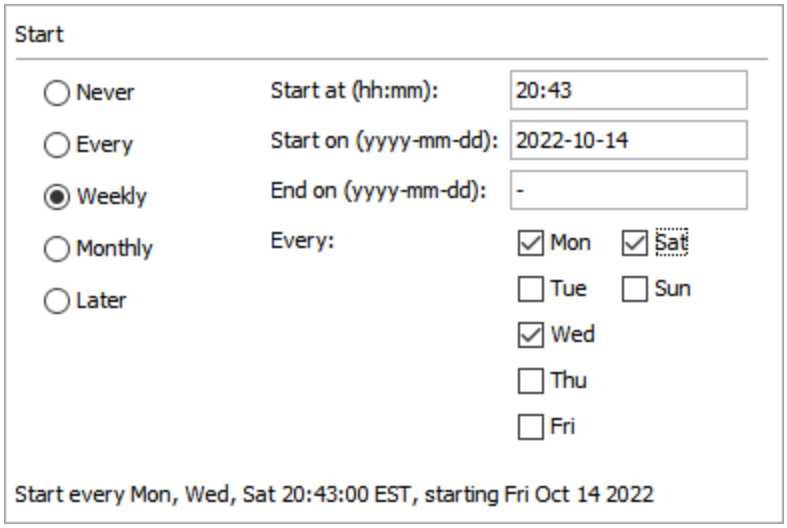
2. Click the ellipsis button (...) and then select one of the following options:



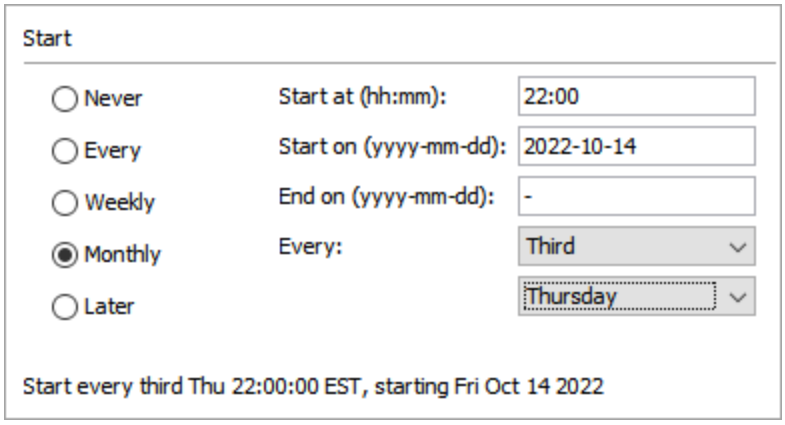
- **Never** - invoke the endpoint manually, no additional parameters are required, and no schedule is applied. This is the default setting and should be used while testing your endpoint.
- **Every** - automatically run the endpoint repeatedly, every selected number of minutes. One (1) minute is the shortest interval allowed and is closest to real time. Enter the number of minutes in the text box. The selection is summarized at the bottom of the dialog.

- **Weekly** - automatically run the endpoint repeatedly, based on the selected time, start and end dates, and days of the week. Use this option if a daily schedule is needed. The 'Start at' parameter determines the time of day that the endpoint will run. The 'Start on' parameter determines the date the endpoint will first run, while the 'End on' parameter determines the date of the endpoint's final run. The 'Every' checkboxes

determine the days of the week when the endpoint will run. The selections are summarized at the bottom of the dialog.



- **Monthly** - automatically run the endpoint repeatedly, once a month, based on the selected time, start and end dates, week of the month, and day of the week. The 'Start at' parameter determines the time of day that the endpoint will run. The 'Start on' parameter determines the date the endpoint will first run, while the 'End on' parameter determines the date of the endpoint's final run. The 'Every' dropdown parameter selections for the week of the month and the day of the week determine when the endpoint will run. The selections are summarized at the bottom of the dialog.



- **Later** - automatically run the endpoint only once, at the time and date specified. The selections are summarized at the bottom of the dialog.

3. Select **Refresh collections before each run** if the outbound integration endpoint contains collections that should be automatically refreshed before each run.

Priority

When the recommended priority-based BGP execution mechanism is configured, waiting BGPs are prioritized for execution based on the priority of the BGP and the created time. The legacy 'Queue for Endpoint' and legacy 'Queue for Endpoint Processes' parameters are not available. Refer to the **Priority** section of the **BG Processes Execution Management** topic in the **System Setup** documentation.

Legacy Queue for Endpoint

This legacy option is not available when the recommended priority-based background process (BGP) execution mechanism is configured.

In a legacy implementation, 'Queue for endpoint' is the name for the queue that is used by the OIEP Background Process to poll the endpoint. The background process handles the actual export. The first time you activate the endpoint, a queue with the specified name is created if it does not already exist. Typically, high-priority integrations and integrations with long-running processes should have their own queue for endpoint processes.

If in doubt about how to populate this parameter, create a new queue for the OIEP, for example, including the OIEP ID in the name.

Legacy Queue for Endpoint Processes

This legacy option is not available when the recommended priority-based background process (BGP) execution mechanism is configured.

STEP allows you to define multiple queues. 'Queue for endpoint processes' is the name for the queue that is used by the background processes started by the endpoint to handle the actual export. The queue is automatically created on the system if it does not already exist. High priority integrations or integrations with long-running processes should typically have their own queue, for example, including the ID in the queue name.

Transactional Settings

'Transactional Settings' can be used for 'Select Objects' OIEPs, but the 'Strict' mode is required for 'Event-Based' OIEPs. For details, refer to the **Integration Endpoint Transactional Settings** topic.

Maximum Number of Threads

Although 'Maximum Number of Threads' is available on the Configuration tab for both static and event-based OIEPs, the setting only affects event-based processes. The default thread setting is one (1) which causes the endpoint to produce a single message at a time, with all events in the batch processed serially. Increasing the thread number results in each batch size being divided by the thread number so that the contents of a batch can be processed in parallel. For more information, refer to **Event-Based OIEP Multithreading Support** topic.

Maximum Number of Waiting Processes

'Maximum Number of Waiting Processes' specifies the maximum number background processes an endpoint can start to handle messages.

By default, the maximum number of waiting processes is set to 1000 for 'Transactional Settings' of Chain or None. When the 'Transactional Settings' parameter is Strict, this value must be 1. Changing an endpoint to have a 'Transactional Settings' of Strict updates this parameter to 1.

Maximum Number of Old Processes

'Maximum Number of Old Processes' specifies the maximum number of ended processes the system retains. This auto-cleanup option deletes succeeded and ended processes started by the OIEP when the limitation is exceeded. The oldest processes are deleted first.

By default, the maximum number of old processes is set to 1000.

Maximum Age of Old Processes

'Maximum Age of Old Processes' specifies the maximum age of succeeded and ended processes. Use the following case-sensitive notation: y = years, M = months, w = weeks, d = days, h=hours, m = minutes, and s = seconds.

By default, the maximum age of old processes is set to one (1) year.

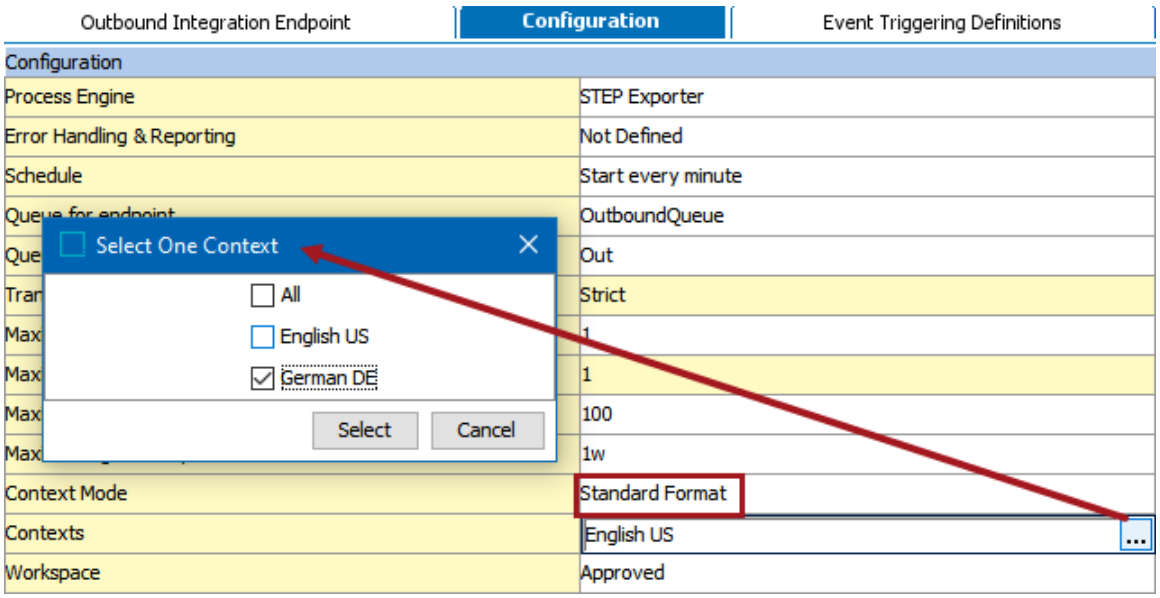
Context Mode and Contexts

When exporting data in the STEPXML format or when using a custom cross-context enabled format, one file can contain data from multiple contexts. For other formats, the standard context splitter post-processor should be used. Using this option, separate files are generated for the selected contexts. For more information, refer to the **OIEP - Post-processor - Context Splitter** documentation.

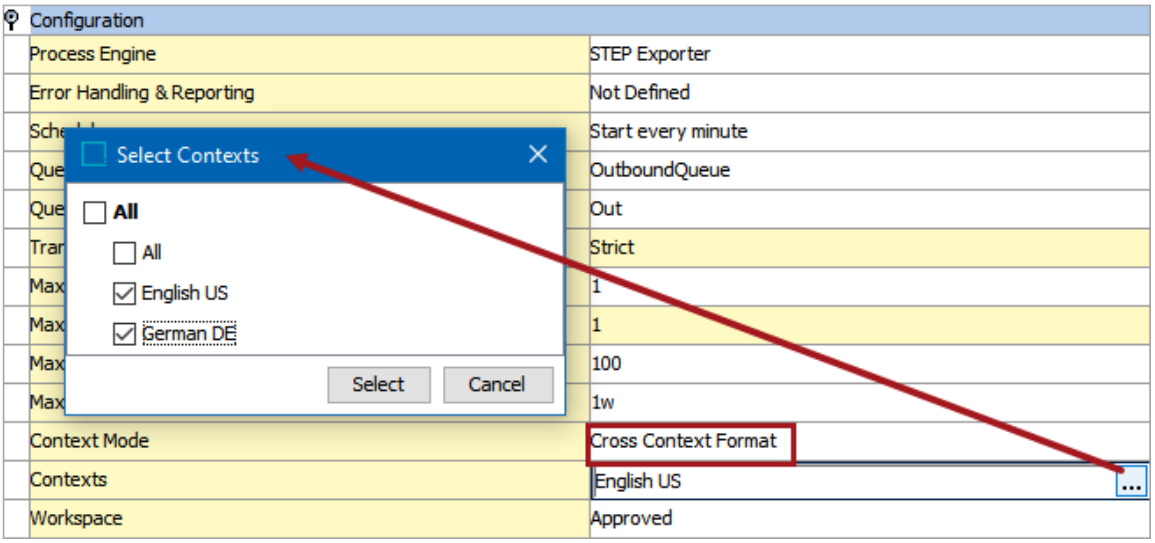
By default, the context is set to the one which is currently being used while creating the OIEP.

Changing Selected Contexts

The following image shows the Standard Format where a single context can be selected.



This image shows the Cross Context Format where a multiple contexts can be selected.



An explicit choice on whether the STEP Exporter should run in 'Standard Format' or 'Cross Context Format' mode is made when setting up the OIEP. It is possible to use cross context mode for endpoints with only a single context selected. This means that it will not be necessary to change downstream systems should more contexts be added later. For endpoints in 'Standard Format' mode (with just one context configured), it will not be possible to add more contexts without explicitly changing the export mode first.

1. Click the Contexts field to display an ellipsis button (...).
2. Click the ellipsis button (...) to display the 'Select Contexts' dialog.
3. Choose the desired contexts and click the **Select** button to save.

The configuration variants are shown in the examples below:

Single context, 'Standard Format':

Context Mode	Standard Format
Contexts	English US

Single context, 'Cross Context Format':

Context Mode	Cross Context Format
Contexts	English US

Multiple contexts, 'Cross Context Format':

Context Mode	Cross Context Format
Contexts	Danish Denmark, English US

When more than one context is selected, Context Mode becomes read-only and set to Cross Context Format (as shown above). The only way to change back to Standard Format is to deselect contexts until only one remains.

Workspace

Workspace defines which data is used for the export. Common setup is to use the Approved workspace, except when you need to generate events for objects before they are approved, for example, during import or from a workflow.

By default, the Approved workspace is selected.

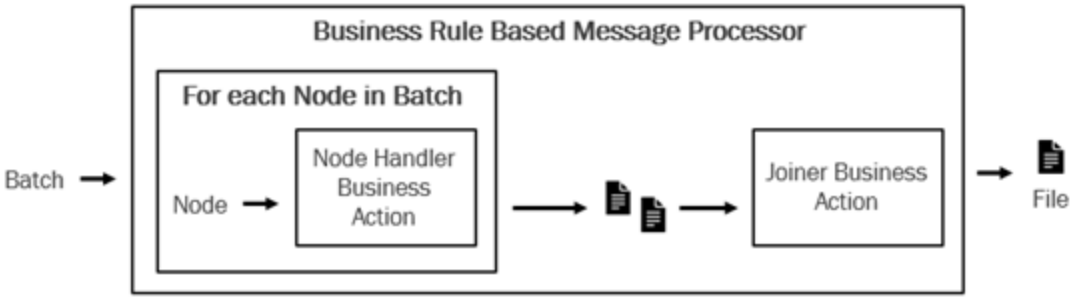
OIEP - Configuration Flipper for Business Rule Based Message Processor

The outbound Business Rule Based Message Processor allows you to compose the representation of STEP data to be exported using either JavaScript-based business actions or Java business actions developed via the Extension API.

For more information on JavaScript business actions, refer to the **Business Action Execute JavaScript** topic in the **Business Rules** documentation. For information on the Extension API, click the **STEP API Documentation** button on the STEP start page.

As illustrated below, the Business Rule Based Message Processor in an OIEP uses two business actions:

- A **node handler** is invoked once for each batch element to be exported and is responsible for producing the export representation of the node / event.
- A **joiner** has access to all representations produced by the node handler and is responsible for joining these into a single message to be delivered from STEP.



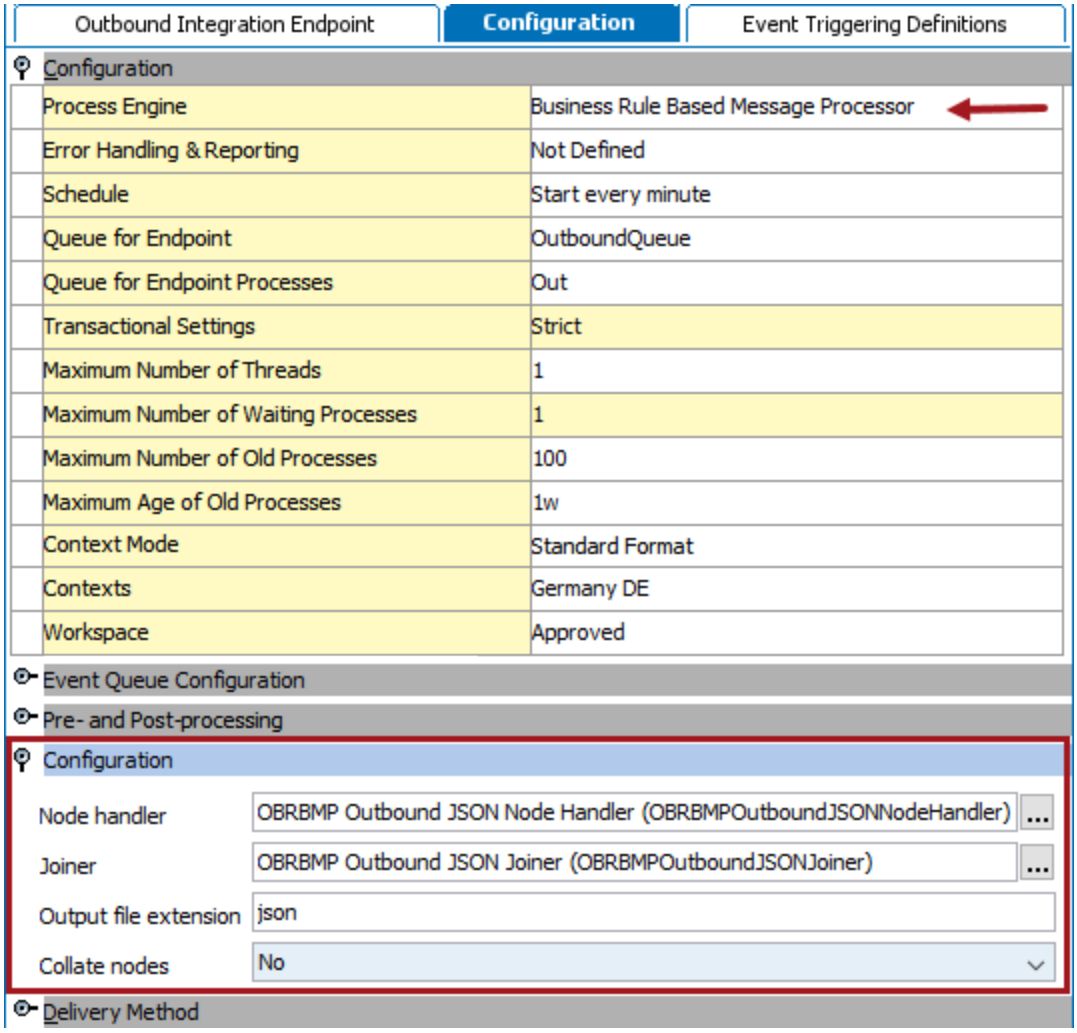
Prerequisites

1. Contact your Stibo Systems account manager or partner manager to activate the **Business Action Processor** commercial license. This enables the event-based importer and exporter to process inbound messages in custom / generic formats using JavaScript-based business rules, and enables exports to custom / generic formats including JSON using JavaScript business rules.
2. Business actions used by this processor should be configured to be valid for all object types.
3. Review the **Node Handler Business Action Details and Examples** and the **Joiner Business Action Details and Examples** sections below for more information about the business actions expected for this processor.

Configuration

A second Configuration flipper is available when the Business Rule Based Message Processor is selected for an OIEP. For information on selecting a process engine, refer to the **OIEP - Event-Based - Configure Endpoint** topic or the **OIEP - Select Objects - Configure Endpoint** topic.

This flipper includes the same parameters for both Event-Based and Select Objects endpoints. Each parameter is described below.



Set up the Business Rule Based Message Processor in the Configuration flipper:

1. For the **Node Handler** parameter, click the ellipsis button (...) to select the business action responsible for producing the representation of each node / event in the batch to be exported.
2. For the **Joiner** parameter, click the ellipsis button (...) to select the business action responsible for combining messages produced by the node handler into one message.
3. For the **Output file extension** parameter, enter the extension for the files to be exported.
4. The **Collate nodes** parameter is only applicable to event-based OIEPs, where a batch can contain multiple events for the same node. Select an option from the dropdown:
   - **Yes** - pass nodes to the node handler business action once per batch. Access is not available to information about the type of event that caused the node / event to be queued.
   - **No** - pass nodes /events as they occur, while having access to information about the type of event that caused the node / event to be queued.

Node Handler Business Action Details and Examples

The node handler business action is invoked once per node / event in the batch to be processed and is responsible for producing a textual representation of the data.

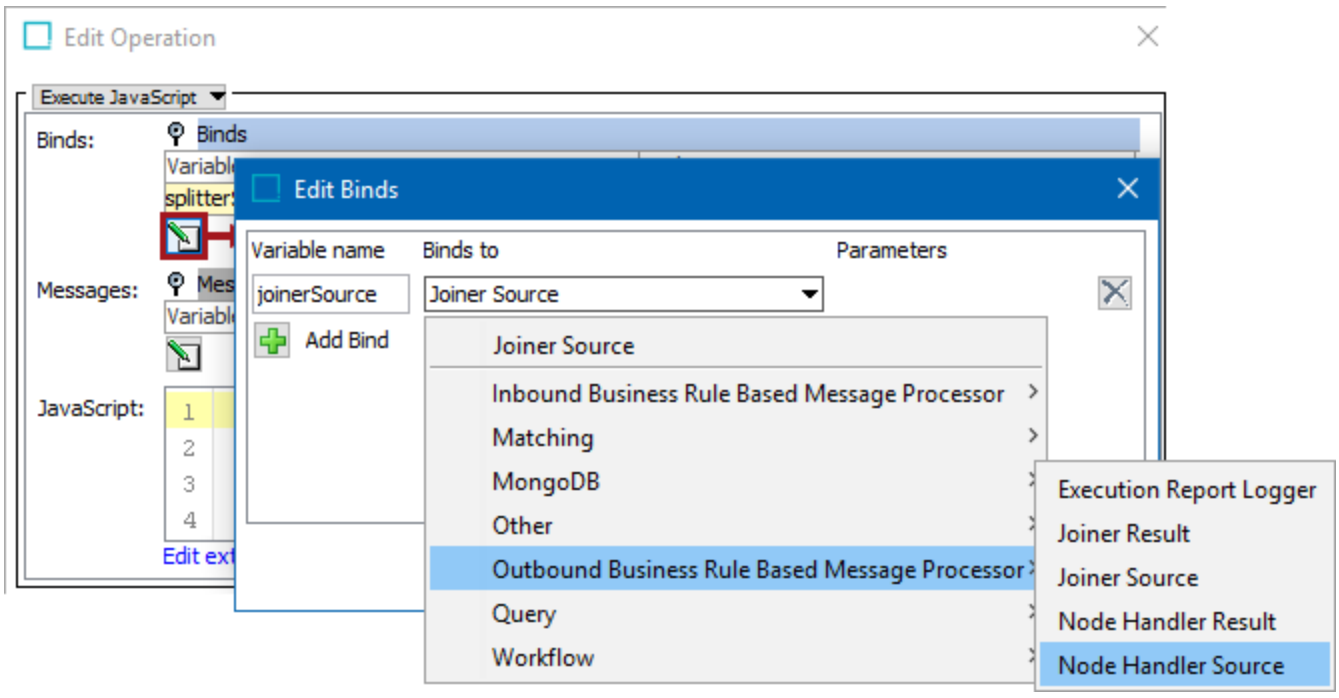
The **OutboundBusinessProcessorNodeHandlerSource** interface (for JavaScript business actions, the interface is available via the Outbound Node Handler Source bind option) provides the business action with access to the batch element. The interface has the following methods for accessing node / event information:

- `getNode(): Node:` The node to export / the node related to the event to be processed. This will always be the node as it looks now in the system. That is to say, the node will not for event-based publishing be a historical version representing the data as it looked when the event was generated. This method can return null if the node has been purged (`isPurged()` returns true).
- `getSimpleEventType(): SimpleEventType:` The event type. This is only available if the outbound integration endpoint is event-based and the 'Collate nodes' parameter is set to 'No.'
- `isDeleted(): boolean:` Returns true if the node related to the batch element being processed has been deleted. This covers the case where the node has been deleted but still exists in the recycle bin, and also the case where the node has been purged from the system.
- `isPurged(): boolean:` Returns true if the node no longer exists in the system.
- `getPurgedNodeID(): String:` If the node has been purged (`isPurged()` returns true), it is possible to get the ID of the purged node via this method. It is not possible to get the 'type' of the purged node, meaning that this method primarily makes sense for setups where only nodes of a single super type are published (since, for example, a product and a classification can have the same ID).

The **OutboundBusinessProcessorNodeHandlerResult** interface (for JavaScript business actions, the interface is available via the Outbound Node Handler Result bind option) is used by the node handler business action to pass the output via the following methods:

- `addMessage(String messageGroup, String message)` allows for messages to be added to what could be thought of as a named bucket. As the joiner business action (described below) can read messages from such named buckets. This allows, for example, all product 'upsert' messages (messages that 'update' or 'insert' as necessary) to be put into one bucket and all product 'delete' messages to be put into another, and the joiner can then add these messages to specific sections of the final combined message.
- `addMessage(String message)` adds a message to a generic anonymous group. Use this option if all message should be treated the same way by the joiner.

Further, the action has access to a standard Logger, a Manager and an **OutboundBusinessProcessorExecutionReportLogger**.



Notice that the node handler is not run in a transaction meaning that it is not possible to modify STEP data from the node handler.

JSON Message Node Handler JavaScript Business Action Example

This example shows how a simple JSON messages can be constructed and messages for 'upserts' and 'deletions' added to different message groups.

Note: `JSON.stringify()` cannot handle Java Strings and these therefore are converted to JavaScript strings when values are set for 'mesg' object properties.

```
// Node Handler Source bound to nodeHandlerSource
// Node Handler Result bound to nodeHandlerResult
// ExecutionReportLogger bound to executionReportLogger

var simpleEventType = nodeHandlerSource.getSimpleEventType();
if (simpleEventType == null) {
  executionReportLogger.logInfo("No event information available in node handler");
} else {
  executionReportLogger.logInfo("Event with ID '" + simpleEventType.getID()+ "'
passed to node handler");
}
var node = nodeHandlerSource.getNode();
if (node != null && node instanceof com.stibo.core.domain.Product) {
  executionReportLogger.logInfo("Node handler handling product with URL: " +
node.getURL());
```

```
  var mesg = {};
  mesg.stepid = node.getID() + "";
  mesg.ean = node.getValue("EAN").getSimpleValue() + "";
  if (nodeHandlerSource.isDeleted()) {
    nodeHandlerResult.addMessage("delete", JSON.stringify(mesg));
  } else {
    mesg.category = node.getParent() == null ? null : node.getParent().getID() + "";
    mesg.shortDescription = node.getValue("ShortDescription").getSimpleValue() + "";
    mesg.manufacturerName = node.getValue("ManufacturerName").getSimpleValue()+ "";
    mesg.color = node.getValue("Color").getSimpleValue()+ "";
    nodeHandlerResult.addMessage("upsert", JSON.stringify(mesg));
  }
}
```

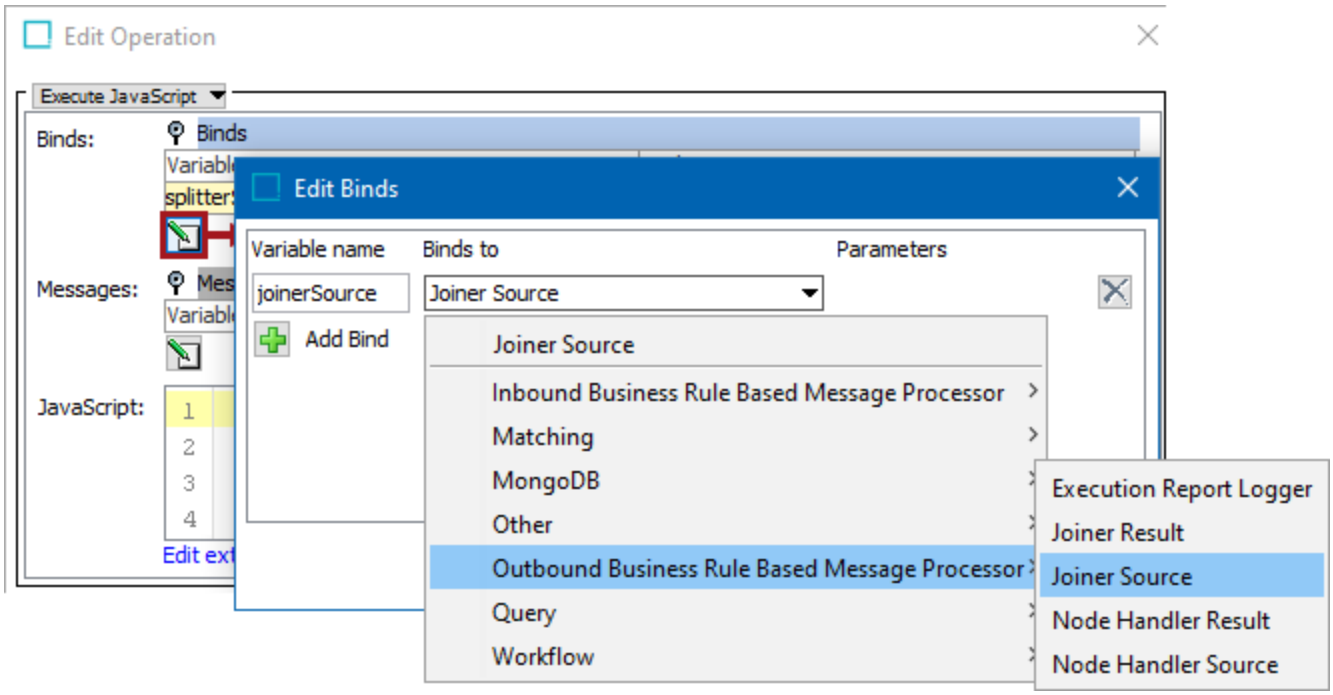
Joiner Business Action Details and Examples

The joiner business action is responsible for combining the messages produced by the node handler into a single message that can be delivered to downstream systems.

The **OutboundBusinessProcessorJoinerSource** interface (for JavaScript business actions, the interface is available via the Outbound Joiner Source bind option) provides the business action access to the messages.

The **OutboundBusinessProcessorJoinerResult** interface `appendToMessage(String)` method (for JavaScript business actions, this interface is available via the Outbound Joiner Result bind option) allows the business action to deliver its output. The `appendToMessage(String)` does not automatically apply any new lines or formatting.

The business action also has access to a standard Logger and an **OutboundBusinessProcessorExecutionReportLogger**.



Notice that the joiner is not run in a transaction meaning that it is not possible to modify STEP data from the node joiner.

JSON Message Joiner JavaScript Business Action Example

This example shows how messages from different message groups can be combined to a single message and further shows how duplicates can be avoided (there can be several elements in an event batch representing the same node).

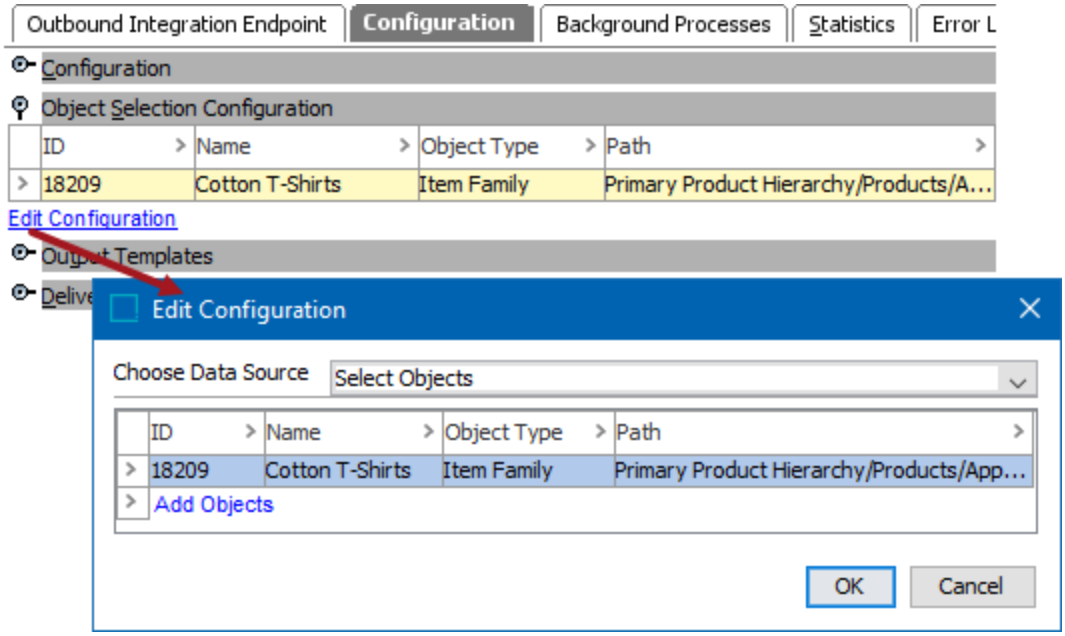
```
// Joiner Source bound joinerSource
// Joiner Result bound to joinerResult

function appendFromGroup(messageGroup) {
  var seen = [];
  var first = true;
  while(joinerSource.hasNext(messageGroup)) {
    var messageString = joinerSource.getNextMessage(messageGroup);
    var hash = messageString.hashCode();
    if (seen.indexOf(hash) == -1) {
      seen.push(hash);
      if (first) {
        first = false;
      } else {
        joinerResult.appendToMessage(",");
      }
      joinerResult.appendToMessage(messageString);
    }
  }
}

joinerResult.appendToMessage("{\"products\":{\"upsert\":[");
appendFromGroup("upsert");
joinerResult.appendToMessage("],\"delete\":[");
appendFromGroup("delete");
joinerResult.appendToMessage("]}}");
```

OIEP - Select Objects - Object Selection Configuration Flipper

On the OIEP Configuration tab, the Object Selection Configuration flipper is only displayed when the selected OIEP uses the Select Objects data source, as defined on the Choose Data Source step of the wizard. For more information, refer to **OIEP - Select Objects - Choose Data Source**.

Note: Selecting a Tree hierarchy object is not required if the endpoint should export system-specific data, such as attributes, LOVs, and units that are not used in any specific product, classification, or entity hierarchy.

1. Click the Edit Configuration link to open the Edit Configuration dialog.

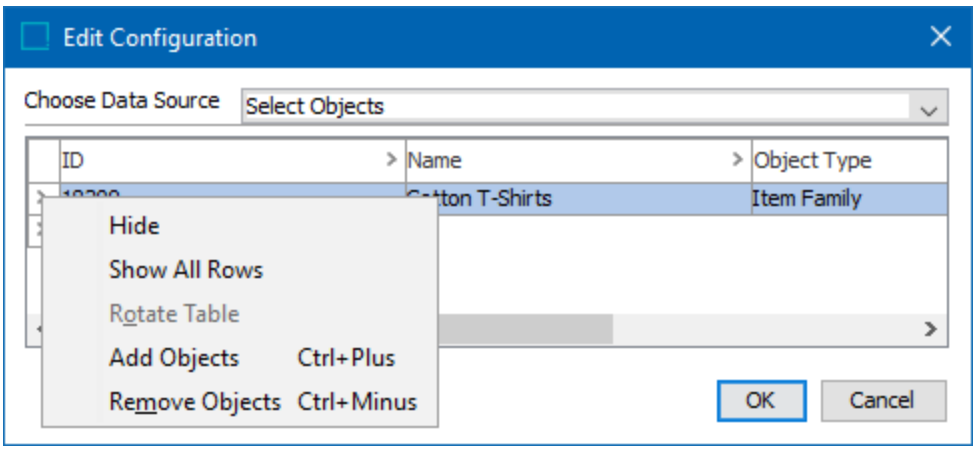


- Select an existing row and right-click to display additional options.

  Hide and **Remove Objects** only affect the selected row.

  Show All Rows and **Rotate Table** affect all rows.

  Add Objects is the same option as accessed with the Add Objects link discussed below.



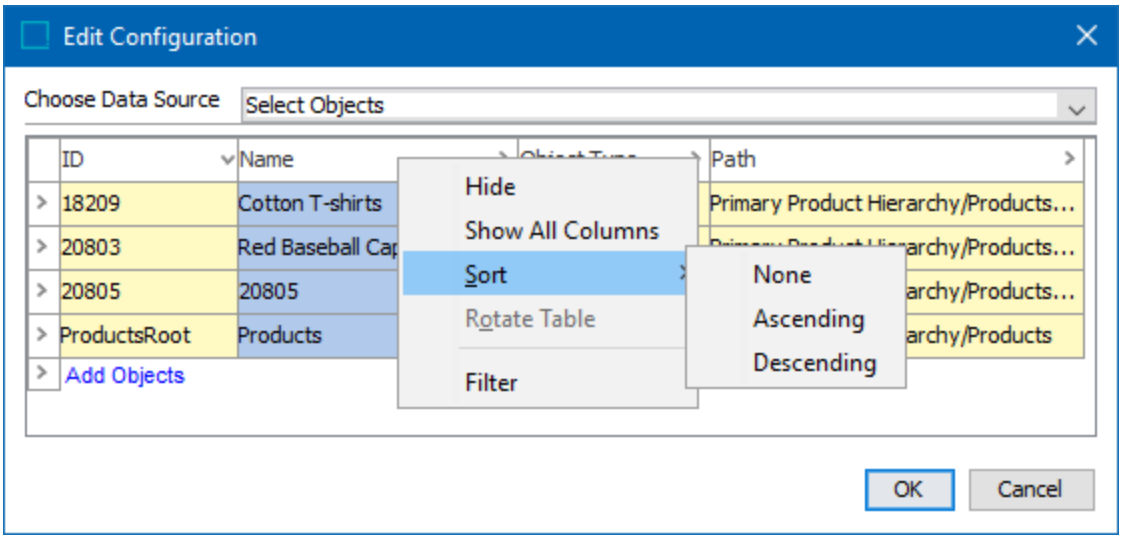
- Select an existing column and right-click to display additional options.

  Hide only affects the selected column.

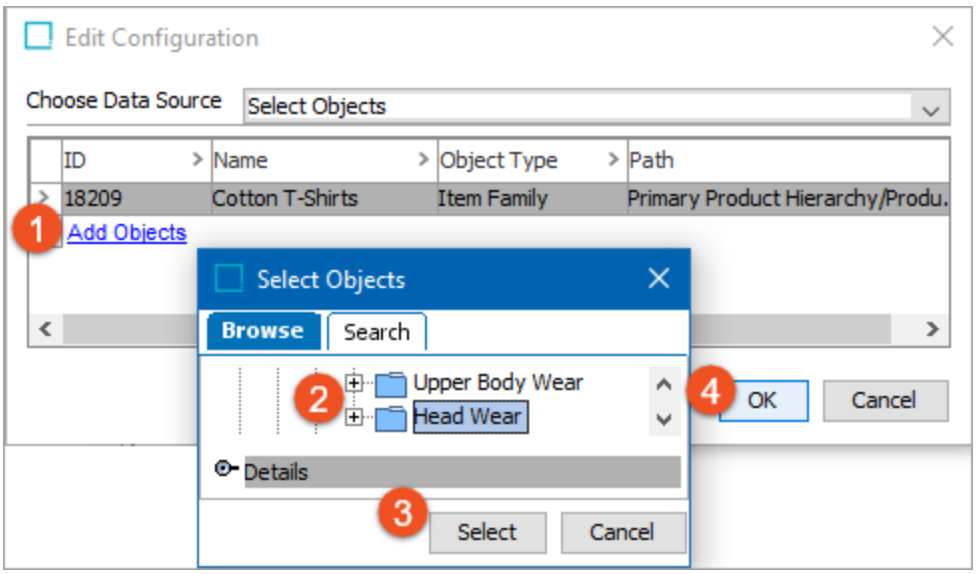
  Show All Columns affects all columns.

  Sort affects all rows and sorts based on the selected column.

  Filter adds a filter option to each column.



- Click the **Add Objects** link to open the Select Objects dialog. Choose a new node and click the Select button. On the Edit Configuration dialog, click OK to include the newly selected data in the output.


Edit Configuration
Choose Data Source
Select Objects
ID
Name
Object Type
Path
18209
Cotton T-Shirts
Item Family
Primary Product Hierarchy/Produ.
1
Add Objects
Select Objects
Browse
Search
2
Upper Body Wear
Head Wear
4
OK
Cancel
Details
3
Select
Cancel

OIEP - Select Objects - Output Templates Flipper

An output template defines the data to be exported and connects it to the desired format. This feature enables you to use a different output format for each selected object type if required.

At least one output template is required, but common setup is to use a single output template for a Select Objects OIEP since the selected data determines the template. Keep in mind that batching is restricted to a single template, so only use multiple templates when required.

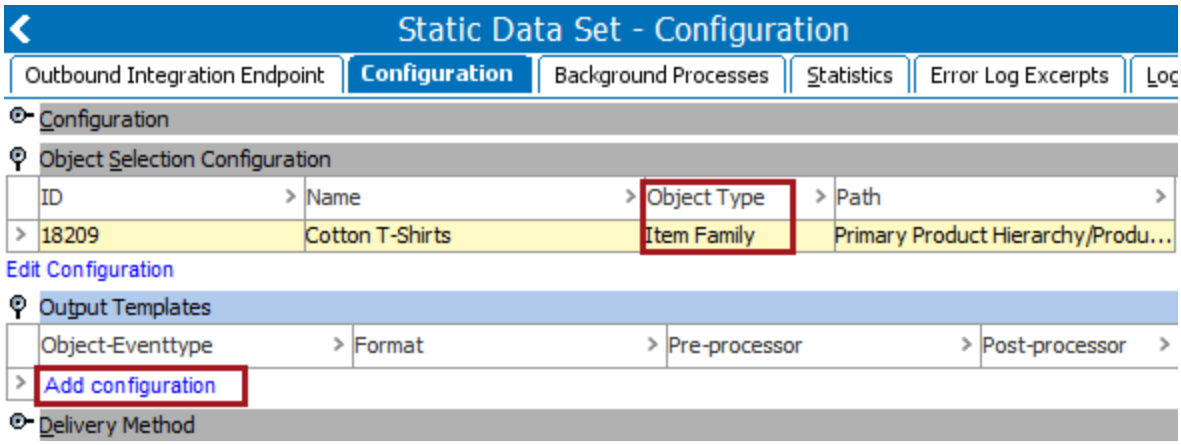
Configuring a select objects output template involves the following steps:

- Configure the object types
- Specify the format
- Configure pre-processor and post-processor, based on availability on your system.

Configure the Object Types

The object type selection determines the STEP data that will be exported.

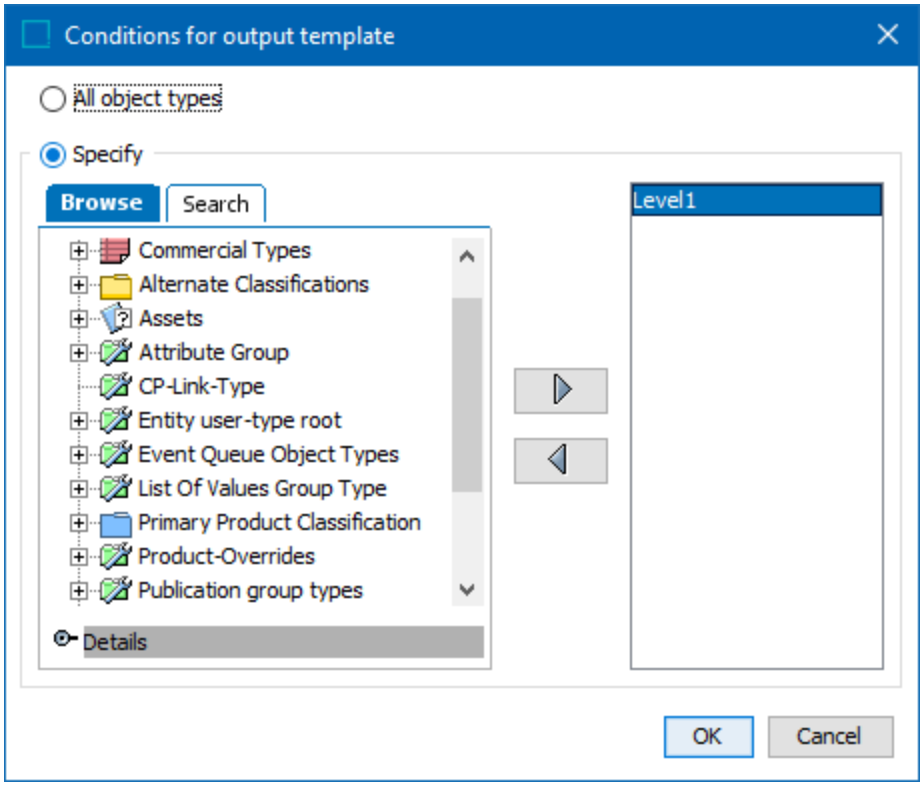
1. In System Setup, select the relevant OIEP > Configuration tab > Output Templates flipper > click the **Add configuration** link to display the 'Conditions for output template' dialog.



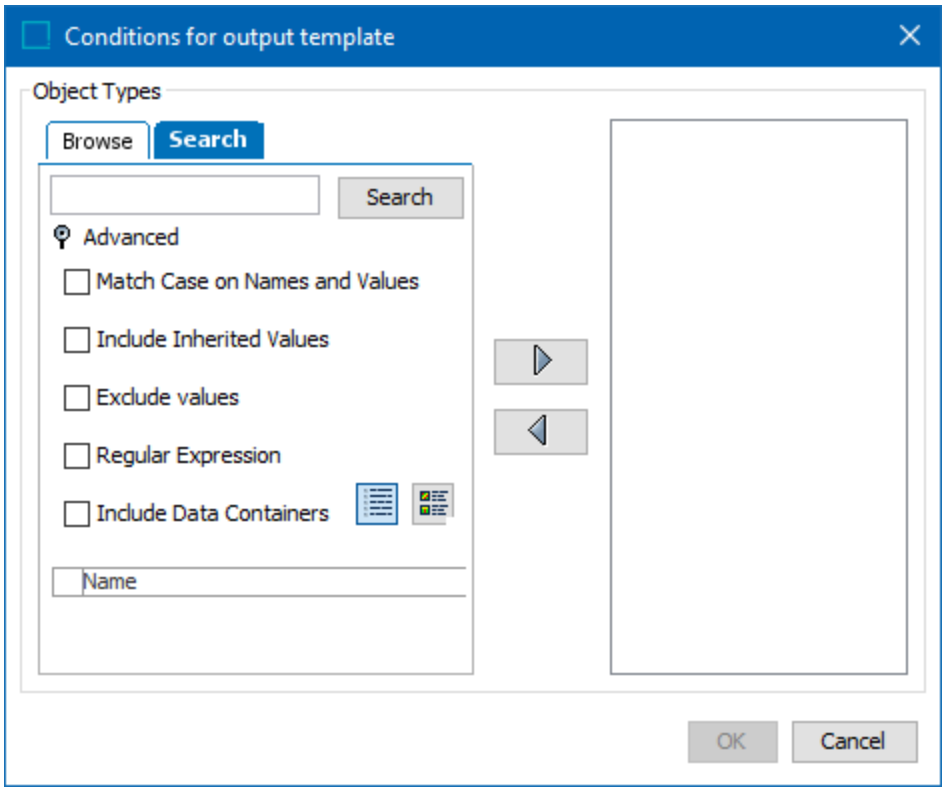
2. In the 'Conditions for output template' dialog, select 'All object types' or 'Specify' and choose the object types that correspond at the root level to the objects selected in the wizard **OIEP - Select Objects - Choose Data Source** step, which is displayed in the Object Selection Configuration flipper under the Object Type column.

   If Collections were selected in the wizard, in the 'Conditions for output template' dialog, add the relevant object types of the objects which are part of the collection.

> **Note:** Selecting an output template object defines only the object types for export, but does not define the objects that are included in the actual output message, since the format of the template may dictate that parent, child, or referenced objects are included. Additionally, the advanced settings for format determine if the selected objects, their children, or both should be considered for the export.

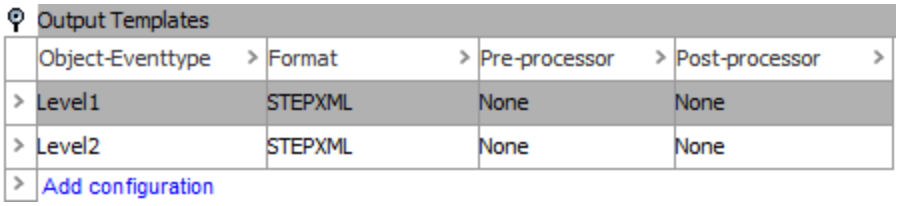


On the Search tab, open the Advanced flipper to display additional search options.



Click the **OK** button.

3. Select Objects endpoints can have only one template per object type. Repeat the previous stets until all required formats are displayed, keeping in mind the limitations of multiple output templates defined in the next section. Event type is not valid for this endpoint.



Output Templates			
Object-Eventtype	Format	Pre-processor	Post-processor
Level1	STEPXML	None	None
Level2	STEPXML	None	None
Add configuration			

4. To edit existing Object-Eventtype settings, click into the desired field to display an ellipsis button (...). Click the ellipsis button (...) to display the 'Conditions for output template' dialog discussed above.

Using Multiple OIEPs or Multiple Output Templates

Since delivery method is per endpoint, not per output format, if you need a different output to go to each external system, you must use multiple OIEPs.

Multiple output templates can be created to allow data output based on specific object types. This prevents the same data from being exported across all chosen object types. Multiple output templates also help to reduce data output, which increases performance both for STEP and for downstream systems. In addition, the use of multiple output templates should reduce the need for customizations.

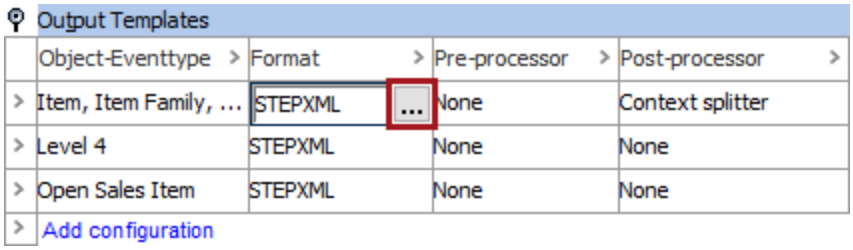
Limitations of Multiple Output Templates

- Output files are only delivered to one destination, regardless of the number of output templates.
- Output templates encompassing all object types are only available if all object types are chosen when making the configuration. This prevents excess output from the endpoints.
- Batching occurs per template type.
- Different output formats cannot be defined for the same object type. Common setup is to avoid combining different output formats (e.g., STEPXML and CSV) in the same endpoint.

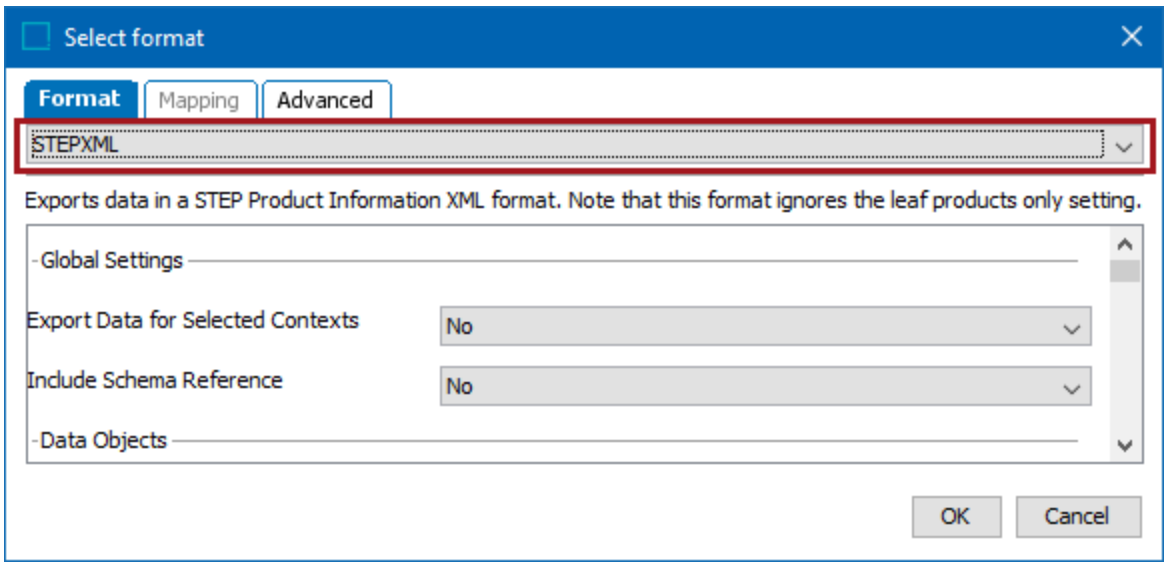
Configure the Format

The format selection determines how the STEP data is formatted (XML or tabular).

1. Click into the format column to display the ellipsis button (...). Click the ellipsis button (...) to display the 'Select format' dialog.

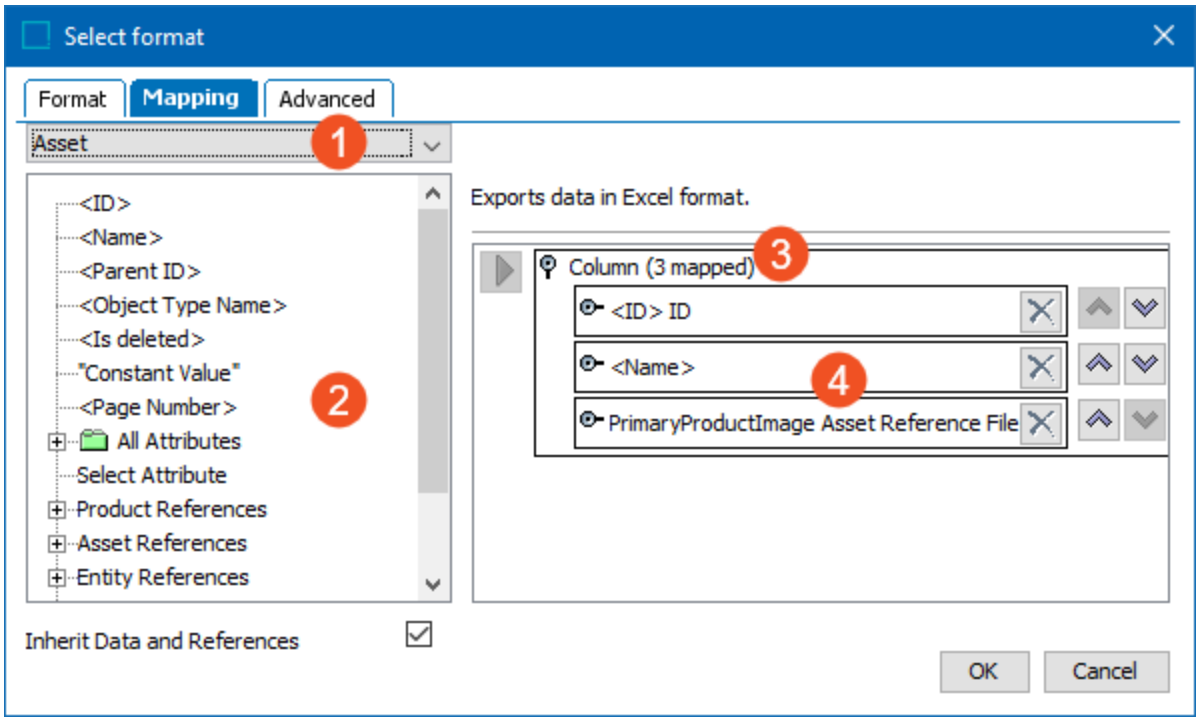


2. In the **Select format** window, on the **Format** tab, select the desired format from dropdown. These are the same options as in the Export Manager, based on your licenses. Use the following links for additional information:
   - Advanced STEPXML Format
   - Alphabetical Index - XML Format
   - Ariba CIF Format
   - Ariba CIF 3.0 Format
   - BMEcat Format
   - BMEcat 2005 Format
   - CSV Format
   - cXML Format
   - Excel Format
   - Excel Smartsheet Format
   - Flatplanner Format
   - Generic XML Format
   - Generic JSON Format
   - IDoc MATMAS05 Format
   - Publication Excel Format
   - STEPXML Format
   - STEPXML Configuration Export Format
   - xCBL Format



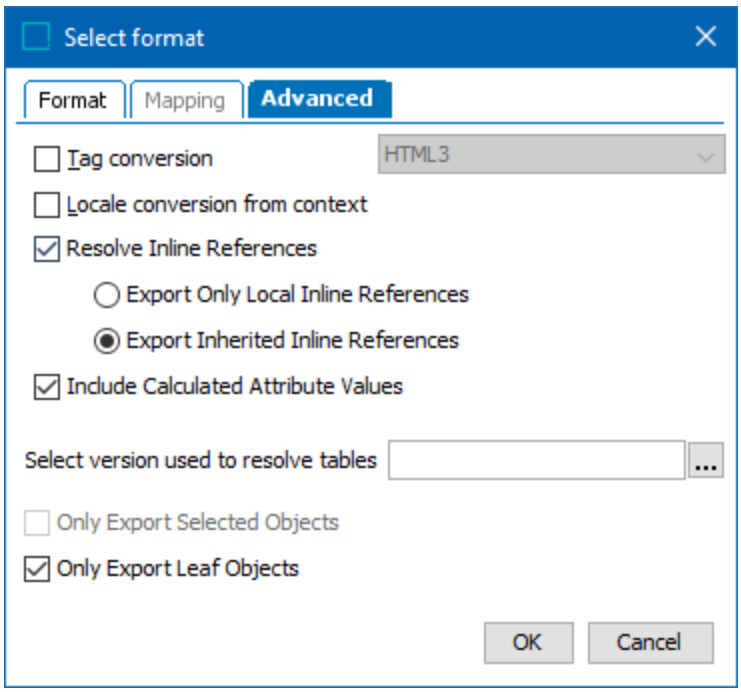
3. When available, specify how to map data to the export format, including the elements identified in the image below: object super type dropdown, data source, mapping target, and mapping rules.

   For more information about the object super type dropdown, refer to the **Export Manager - Select Objects** topic. For more information about the other elements, refer to the **Export Manager - Map Data** topic.



Note: The **Mapping** tab is enabled when the format selected requires mapping. Selected formats, including STEPXML and Advanced STEPXML, are automatically mapped so the Mapping tab is disabled.

4. On the **Advanced** tab, a number of advanced export options can be specified.



- **Tag conversion**: Converts tags to match the selected output format. This setting is optional.

  If this box is left unchecked, any formatting tags that are included in attribute values will not be converted in the outbound message. For example, if an attribute value contains bold text (which must be made bold with a STEP Style Tag, e.g., <bold>/</bold>), this tag will not be converted to its corresponding HTML output format (e.g., <B>/</B>) in the export. For more information on STEP Tags, refer to System Setup > **Tags** in the online help.

- **Locale conversion from context**: Converts numbers into the numeric format that corresponds to the selected locale. If Smartsheets are used for format, this must remain unchecked. By default, the option is not checked.
- **Resolve Inline References**: Resolves inline references when they are exported. This box is checked by default, along with the option 'Export Inherited Inline References.'

  If unchecked, inline references will be exported with the inline reference tagging instead of the actual content pulled in by the inline reference. For example, if an attribute called 'Product Number' has an inline reference to pull in the STEP ID of the object (e.g., 12345), the attribute value will not contain 12345. Instead, it will contain tagging similar to the following:

```
<ref attrid="" equalsign="=" includeattrname="false" resolveto="objid"
separator=";"/>
```

This setting would be valid if the user planned to re-import the file at a later time and not overwrite the inline reference with a static value. For more information on inline references, refer to **Inline References in Attribute Values** in the **Getting Started** documentation.

- **Include Calculated Attribute Values**: Resolves calculated attribute values when they are exported. By default, the option is checked. If not checked, calculated attribute values are exported with empty values, unless the Force Calculation option has been set. The Force Calculation option is available when mapping a calculated attribute using the Select Attribute data source, and also on the Transformations dialog when mapping with a transformation. For more information, refer to **Force Calculation** in **Attribute - Data Source Outbound** and **Outbound Map Data - Transform**.

  The value template is exported for each selected context, including a qualifier ID, which makes it possible to import the same data back into STEP. For information about calculated attribute values, refer to the **Calculated Attributes** topic in **System Setup** documentation.

Important: If many complex calculated attribute values (traversing hierarchies and/or references) are used, consider if they should be exported, since they can impact performance. If required, consider scheduling for non-peak times. Simple calculations are not detrimental to an export, regardless of the quantity.

- **Select version used to resolve tables**: This setting is available for STEPXML if 'Include Tables' is set to Yes and for Advanced STEPXML if the template includes version-dependent content. A publication version should be selected if the tables contain content relevant to a particular publication (such as column or row types that are only valid for certain publication types) or a publication version (such as commercial data). For more information, refer to the **Exporting Resolved Tables** topic in the **Tables** documentation.
- **Only Export Selected Objects**: Specifies that only objects from the output template are exported. No children are exported. This setting is unchecked by default and is only available if 'Only Export Leaf Objects' is unchecked. This setting does not apply to STEPXML.
- **Only Export Leaf Objects**: Specifies that only the leaf objects (lowest level of the export object for both events and selected objects) of the selected top hierarchy are included in the export. Selected objects and triggering objects are only included if they have no children. This setting applies to the CSV or Excel format.

Note: Uncheck both ‘Only Export Selected Objects’ and ‘Only Export Leaf Objects’ if all objects (both for triggering events and object selection) and their children should be included in the export.

For more information, refer to **Export Manager - Advanced** documentation.

Configure the Pre-processor and Post-processor

Depending on your system setup, pre-processor and post-processor options may be available.

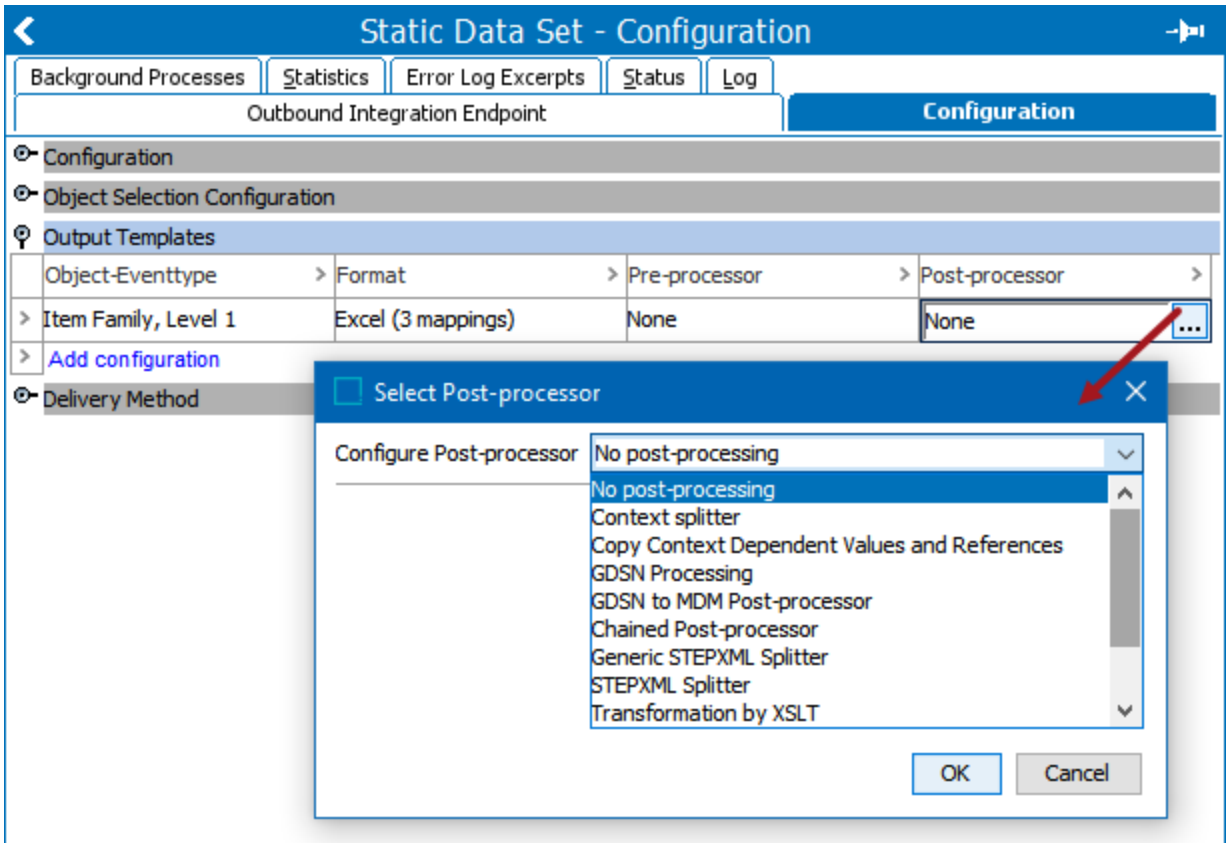
Pre-processor

Although a business action pre-processor can be set up for a select objects endpoint, it is not recognized and will not cause any changes to the data being processed.

Post-processor

Standard post-processing options should be evaluated if multiple contexts are included in the output. You can configure a post-processor for each of the output templates.

1. Click into the Post-processor column to display the ellipsis button (...). Click the ellipsis button (...) to display the 'Select Post-processor' dialog.



2. In the **Configure Post-processor** parameter, select from the following options:
   - **No post-processing** exports files using the standard export functionality.
   - **Context splitter** generates an export file for each configured context. Each exported file will only contain context specific data. For more information, refer to the **OIEP - Post-processor - Context Splitter** topic.

     For the **Copy inherited product values** parameter, select **Yes** to copy and save inherited product values to the child product; or select **No** to use inherited product values in the export, but leave the child product unmodified.

     When using Excel or CSV format, and multiple contexts are configured for export, you must select Context Splitter, or a single context will be included in the export.

- **Copy Context Dependent Values and References** to export context-dependent values and references and add the corresponding ContextID attribute to the value or reference in question. The endpoint generates one file containing values and references for each context specified. For more information, refer to the **OIEP Post-processor - Copy Context Dependent Values and References** section.

  For the **Copy inherited product values** parameter, select **Yes** to copy and save inherited product values to the child product; or select **No** to use inherited product values in the export, but leave the child product unmodified.

  If you use the **Copy Context Dependent Values and References** post-processor to add ContextIDs to a cross-context export (exporting data for selected contexts), the downstream system can use the <ContextID> tag to identify specific contexts. Additionally, this facilitates re-importing the exported data into the correct contexts in STEP after processing by a third-party service or application.

- **Chained Post-processor** as defined in the **OIEP - Post-processor - Chained Post-processor** topic.
- **Generic STEPXML Splitter** splits up STEPXML messages to multiple STEPXML valid fragments containing one single node per STEPXML file.
- **STEPXML Splitter** produces one file per STEP object as defined in the **'STEPXML Splitter Post-processor for OIEP'** section of the **Integration Endpoint Options for VCS Integrations** topic in the **Configuration Management** documentation.
- **Transformation by XSLT** transforms data via an XSLT stylesheet before exporting. Under the XSLT-Stylesheet parameter, click the ellipsis button (...) and browse or search for the stylesheet to use for the transformation. For more information, refer to **OIEP - Post-processor - Transformation by XSLT** section of the **Data Exchange** documentation.
- **GDSN to MDM Post-processor** handles an exported STEPXML file that may include many packaging hierarchies and many objects within each packaging hierarchy. The post-processor will split these files up by STEP context and by trade item before applying transformations. The transformed files will have a hash code generated and compared to a previous hash code to verify if there have been changes. Optional business actions can also be executed to affect workflows or other purposes. For more information, consult your Stibo Systems account manager or partner manager.

  Note: This post-processor is only available when running the GDSN to Receiver component.

OIEP - Pre-processor - Business Action

By default, no pre-processor is included in an OIEP output template. Applying the Business Action Pre-processor means a business rule can be run against the data, prior to the export. For example, a node can be added to a current batch, or an event can be removed from the current batch before a message is created for an export.

Using this pre-processor requires:

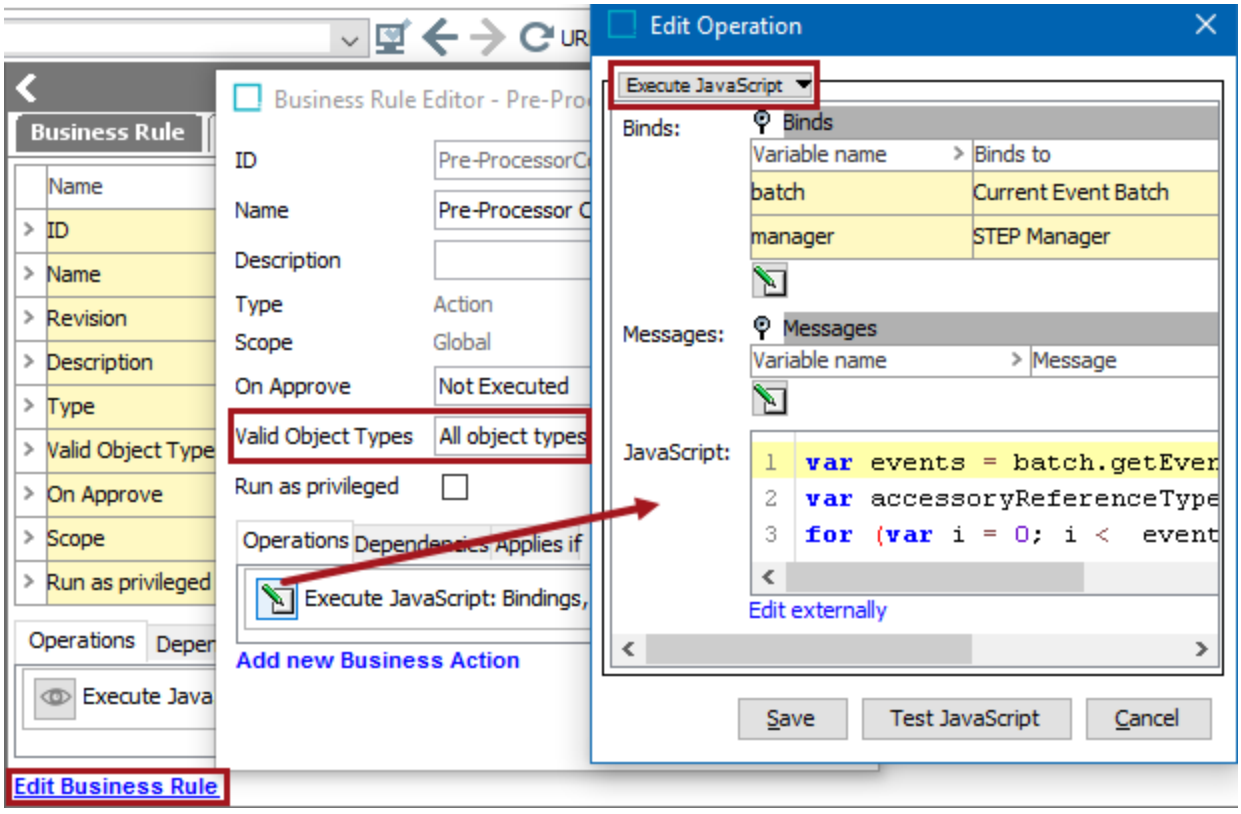
1. A business action - in this example, it will add referenced products to the products that are contained in the message.
2. An event-based OIEP configured with the business action - in this example, it is configured to deliver messages that inform a downstream system about changes to a product.

Business Action Configuration

Create a business action that is valid for 'All Object Types.' For more information on creating business rules, refer to the **Specifying a Business Action Operation** topic.

For a full JavaScript code example, refer to the online version of this topic.

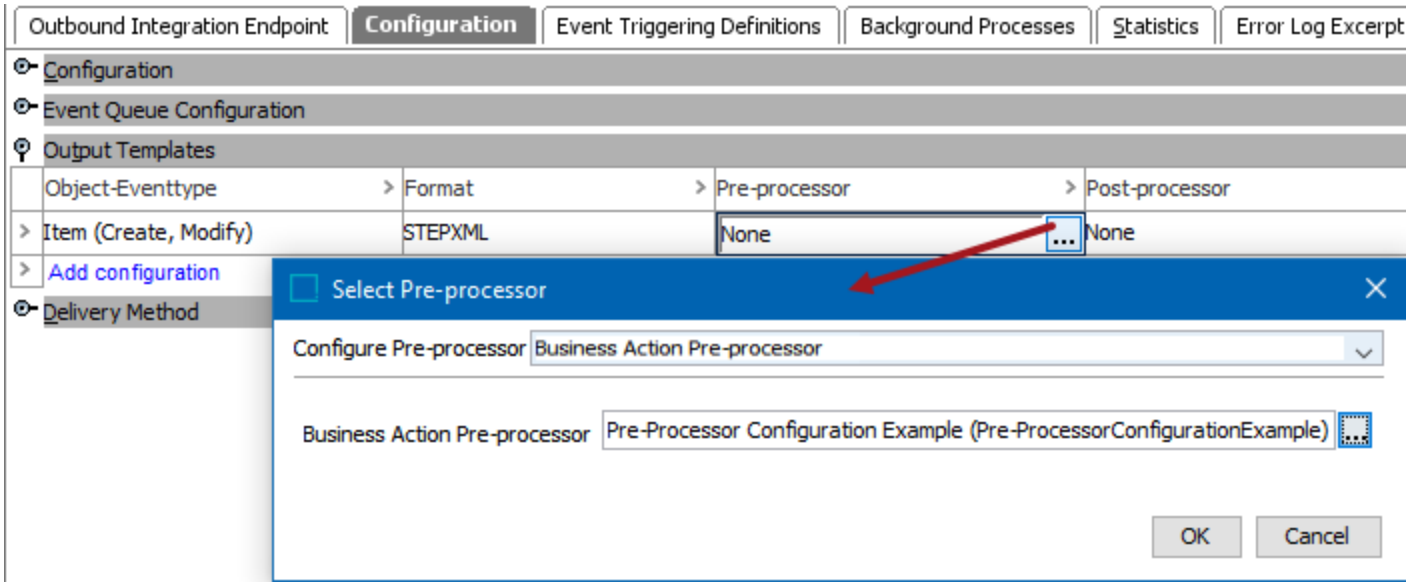
The business rule is configured in the System Setup tab under Global Business Rules as displayed below:



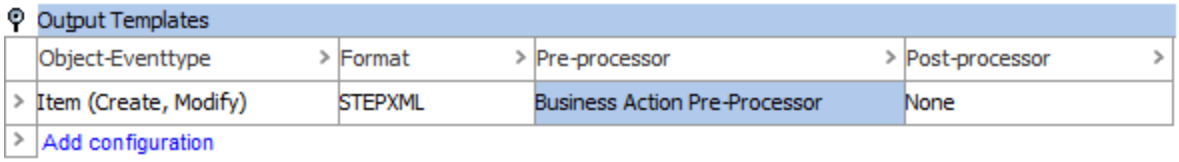
OIEP Configuration

Apply the business action as a pre-processor for any of the output templates on an OIEP.

1. Create an event-based OIEP. For details, refer to the **Creating an Event-Based Outbound Integration Endpoint** documentation.
2. On the **Configuration** tab, open the **Output Templates** flipper and locate the output template that requires the business action.



3. In the **Pre-processor** column, click the ellipsis button (...) to display the Select Pre-processor dialog.
4. In **Configure Pre-processor**, select the **Business Action Pre-processor** option.
5. In **Business Action Pre-processor**, select the business action created earlier. The output template is updated to show the pre-processor will run.



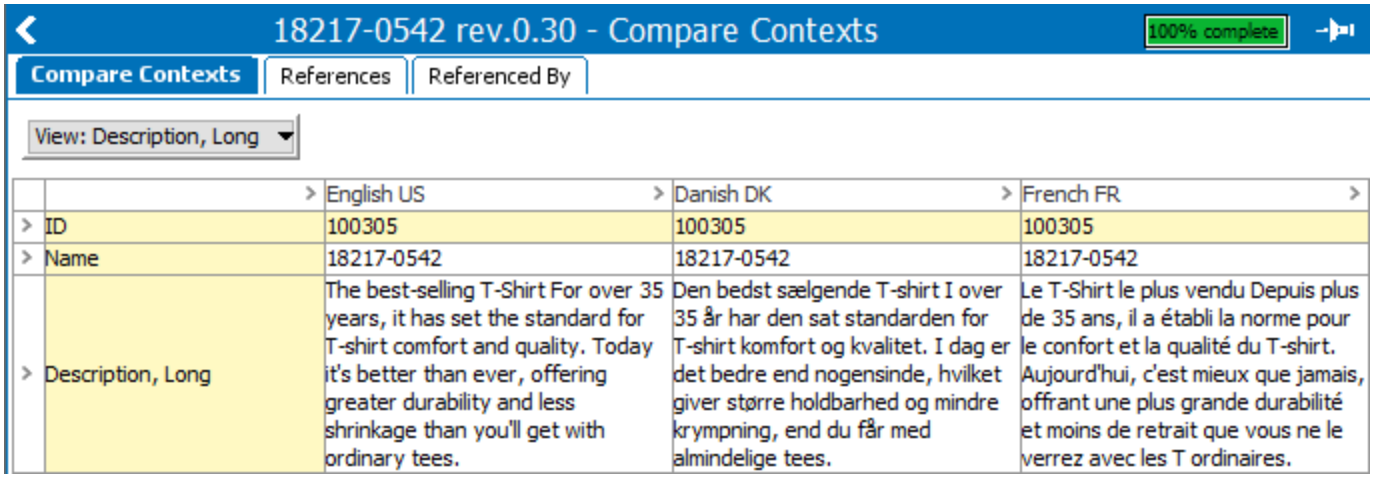
OIEP - Post-processor - Context Splitter

The Context Splitter post-processor splits the export file into individual files based on the contexts on which the data is dependent. This can be used when the data from different contexts should be sent to the external systems separately. It is also commonly used for reporting purposes for individual contexts.

The context splitter adds the <ContextID> without a <Qualifier ID>. This makes the file similar to an export of local values done from a single context. The <ContextID> tag specifies the ID of the context in which the object is exported, but does not provide the specific dimension point details which can only be supplied by the <QualifierID> tag. Using the context splitter assumes that the external system does not have any details of dependency points from STEP.

The following example illustrates using an outbound integration endpoint with the Context Splitter post-processor. It includes attribute values, but the post-processor also works with inherited references such as classification or asset references.

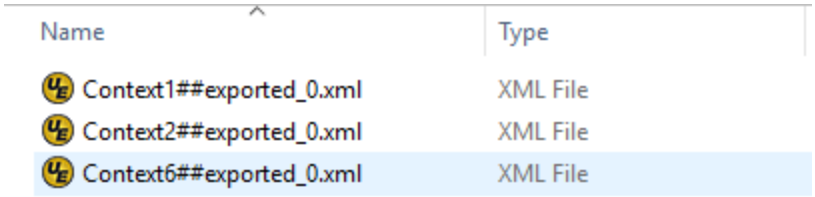
The product has a language-dependent attribute, 'Description, Long', with values defined in three contexts: English US, Danish DK, and French FR.



	English US	Danish DK	French FR
ID	100305	100305	100305
Name	18217-0542	18217-0542	18217-0542
Description, Long	The best-selling T-Shirt For over 35 years, it has set the standard for T-shirt comfort and quality. Today it's better than ever, offering greater durability and less shrinkage than you'll get with ordinary tees.	Den bedst sælgende T-shirt I over 35 år har den sat standarden for T-shirt komfort og kvalitet. I dag er det bedre end nogensinde, hvilket giver større holdbarhed og mindre krympning, end du får med almindelige tees.	Le T-Shirt le plus vendu Depuis plus de 35 ans, il a établi la norme pour le confort et la qualité du T-shirt. Aujourd'hui, c'est mieux que jamais, offrant une plus grande durabilité et moins de retrait que vous ne le verrez avec les T ordinaires.

XML Output Example – Context Splitter

Using the context splitter, a separate file is generated for each context. As shown below, the names of the files output identify the Context IDs from which the objects were exported.



The <ContextID> tag has been added to the export. The files can be distinguished by the Context ID and there is no <Qualifier ID> or any information about the dimension points. The XML output for each of the files is:

```
<STEP-ProductInformation ExportTime="2017-08-07 11:51:18" ExportContext="Context6" ContextID="Context1" WorkspaceID="Main" Us
  <Products>
    <Product ID="100305" UserTypeID="SalesItem" ParentID="18206">
      <Name>18217-0542</Name>
      <Values>
        <Value AttributeID="DescriptionLong">The best-selling T-Shirt For over 35 years, it has set the standard for T-shirt
```

```
<STEP-ProductInformation ExportTime="2017-08-07 11:51:18" ExportContext="Context6" ContextID="Context2" WorkspaceID="Main" Us
  <Products>
    <Product ID="100305" UserTypeID="SalesItem" ParentID="18206">
      <Name>18217-0542</Name>
      <Values>
        <Value AttributeID="DescriptionLong">Le T-Shirt le plus vendu Depuis plus de 35 ans, il a établi la norme pour le con
```

```
<STEP-ProductInformation ExportTime="2017-08-07 11:51:18" ExportContext="Context6" ContextID="Context6" WorkspaceID="Main" Us
  <Products>
    <Product ID="100305" UserTypeID="SalesItem" ParentID="18206">
      <Name>18217-0542</Name>
      <Values>
        <Value AttributeID="DescriptionLong">Den bedst sælgende T-shirt I over 35 år har den sat standarden for T-shirt komfo
```

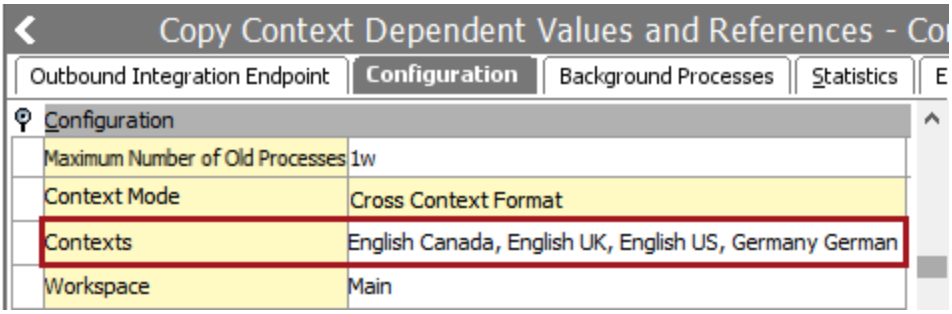
XML Output Example - Standard Cross-Context Export

Using a standard cross-context export (exporting data for selected contexts) without the context splitter, the single output file includes only the attribute values specific to particular dimension points (indicated by QualifierIDs). The XML output is:

```
    <ValueGroup AttributeID="DescriptionLong">
      <Value QualifierID="en-US">The best-selling T-Shirt For over 35 years, it has set the stan
      <Value QualifierID="fr">Le T-Shirt le plus vendu Depuis plus de 35 ans, il a établi la nor
      <Value QualifierID="Danish">Den bedst sælgende T-shirt I over 35 år har den sat standarden
    </ValueGroup>
```

OIEP - Post-processor - Copy Context Dependent Values and References

When multiple contexts are selected in the Configuration flipper of an OIEP, this post-processor reduces the complexity of inheritance of values in cross-context exports, and allows downstream systems to receive a multi-context export in a single file.



Important: Using the 'Copy Context Dependent Values and References' post-processor requires a strong understanding of the STEP Qualifier concept that defines in which dimensions and at which inheritance levels values are present.

If you want to further process the information in a form similar to a typical single-context export, we strongly recommend using the context-splitter post-processor instead. For more information, refer to the **OIEP - Post-processor - Context Splitter** topic.

ContextID and QualifierID XML Tags

The 'Copy Context Dependent Values and References' post-processor adds the <ContextID> and <QualifierID> XML tags to the export. A standard cross-context export (exporting data for multiple contexts) only adds the <QualifierID>. The <QualifierID> tag specifies the ID of the Dimension Point in which the attribute value exists but does not provide the specific context. For example, the <QualifierID> of 'en-US' could be the ID of the 'English' dimension point, but without the <ContextID> tag, the additional contexts that an attribute value has inherited down to cannot be determined.

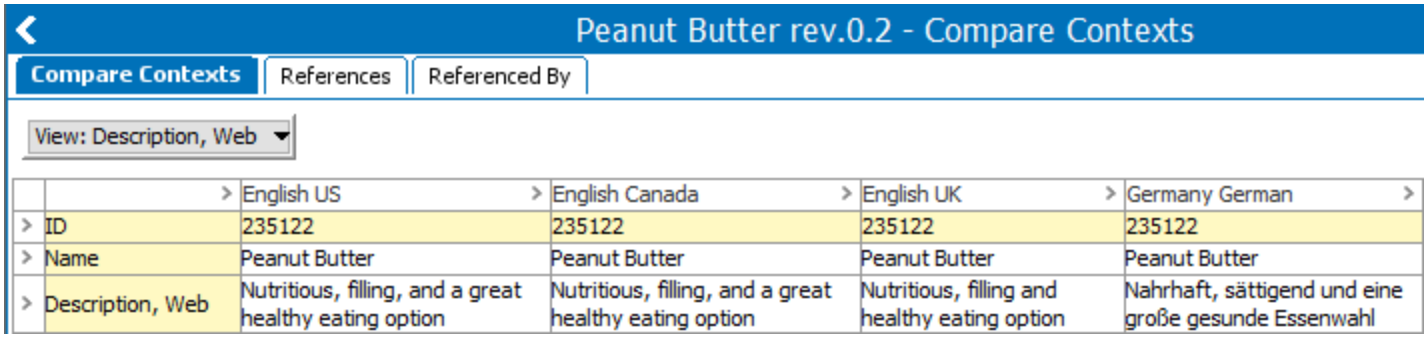
When the <ContextID> XML tag is used in combination with the <QualifierID> XML tag, it is possible for the downstream system to determine the inheritance relationship between a value (or reference) and a context. The <QualifierID> XML tag is useful when a service tries to re-import processed data into STEP. Because the downstream system knows which context the data must be imported into, it is easier to target specific contexts within STEP for import.

The following example illustrates the differences between using the Export Manager for a standard cross-context export (exporting data for multiple contexts) and using an outbound integration endpoint with the **Copy Context Dependent Values and References** post-processor enabled.

Although attribute values are used, the post-processor also works with inherited references such as classification or asset references.

The product with ID '235122' has a language-dependent attribute, 'Description, Web.' The following contexts will be considered:

- 'English US' (Context 1) has a value defined and is the master context (determined by its selection in the Context dropdown
- 'English Canada' (Context3) does not have a value defined but inherits values from the English US context
- 'English UK' (Context7) has a value defined (notice the slightly different wording)
- 'Germany German' (Context5) has a value defined



	English US	English Canada	English UK	Germany German
ID	235122	235122	235122	235122
Name	Peanut Butter	Peanut Butter	Peanut Butter	Peanut Butter
Description, Web	Nutritious, filling, and a great healthy eating option	Nutritious, filling, and a great healthy eating option	Nutritious, filling and healthy eating option	Nahrhaft, sättigend und eine große gesunde Essenwahl

XML Output Example - Standard Cross-Context Export

Using a standard cross-context export (exporting data for selected contexts), the following XML is generated:

```
<?xml version="1.0" encoding="utf-8"?>
<STEP-ProductInformation ExportTime="2018-04-16 05:24:26" ExportContext="Context1" ContextID="Cont
 <Products>
   <Product ID="235122" UserTypeID="SalesItem" ParentID="235115">
     <Name>Peanut Butter</Name>
     <ProductCrossReference ProductID="249236" QualifierID="en-US" Type="OIEP_POSTPROCESSOR"/>
     <ProductCrossReference ProductID="157618" QualifierID="UK English" Type="OIEP_POSTPROCESSOR"/
     <ProductCrossReference ProductID="157618" QualifierID="en-US" Type="OIEP_POSTPROCESSOR"/>
     <Values>
       <ValueGroup AttributeID="DescriptionWeb">
         <Value QualifierID="en-US">Nutritious, filling, and a great healthy eating option</Value>
         <Value QualifierID="German">Nahrhaft, sättigend und eine große gesunde Essenwahl</Value>
         <Value QualifierID="UK English">Nutritious, filling and healthy eating option</Value>
       </ValueGroup>
      </Values>
   </Product>
 </Products>
</STEP-ProductInformation>
```

Only the attribute values specific to particular dimension points—indicated by QualifierIDs—are exported, and only the attribute values that are different from the master context. Master values are taken from the 'en-US' and the English UK values are taken from 'UK English.'

Note: Standard export methods collapse duplicate values, and only the top value of the inheritance tree is exported.

XML Output Example - with Copy Context Dependent Values and References post-processor

The following shows an XML output of an OIEP cross-context export (exporting data for selected contexts) with the **Copy Context Dependent Values and References** post-processor enabled:

```
<?xml version="1.0" encoding="utf-8"?>
<STEP-ProductInformation ExportTime="2018-04-13 06:24:04" ExportContext="Context1" ContextID="Context1" WorkspaceID="M
 <Products>
  <Product ID="235122" UserTypeID="SalesItem" ParentID="235115">
    <Name>Peanut Butter</Name>
    <ProductCrossReference ProductID="249236" ContextID="Context3" QualifierID="en-US" Type="copyContextDepReference"/
    <ProductCrossReference ProductID="157618" ContextID="Context1" QualifierID="en-US" Type="copyContextDepReference"/
    <ProductCrossReference ProductID="157618" ContextID="Context7" QualifierID="UK English" Type="copyContextDepRefere
    <Values>
     <Value AttributeID="DescriptionWeb" ContextID="Context1" QualifierID="en-US">Nutritious, filling, and a great hea
     <Value AttributeID="DescriptionWeb" ContextID="Context7" QualifierID="UK English">Nutritious, filling and healthy
     <Value AttributeID="DescriptionWeb" ContextID="Context3" QualifierID="en-US">Nutritious, filling, and a great hea
     <Value AttributeID="DescriptionWeb" ContextID="Context5" QualifierID="German">Nahrhaft, sättigend und eine große
    </Values>
  </Product>
 </Products>
</STEP-ProductInformation>
```

The <ContextID> tag has been added to the export along with the <QualifierID> tag, and each value—including inherited values—is explicitly exported. This means that the 'English Canada' (Context3) values are included in the export, even though it inherits from the master context. If the value is overwritten in the English Canada context, as shown below, the <QualifierID> changes to the ID of the 'Canada English' language dimension point.

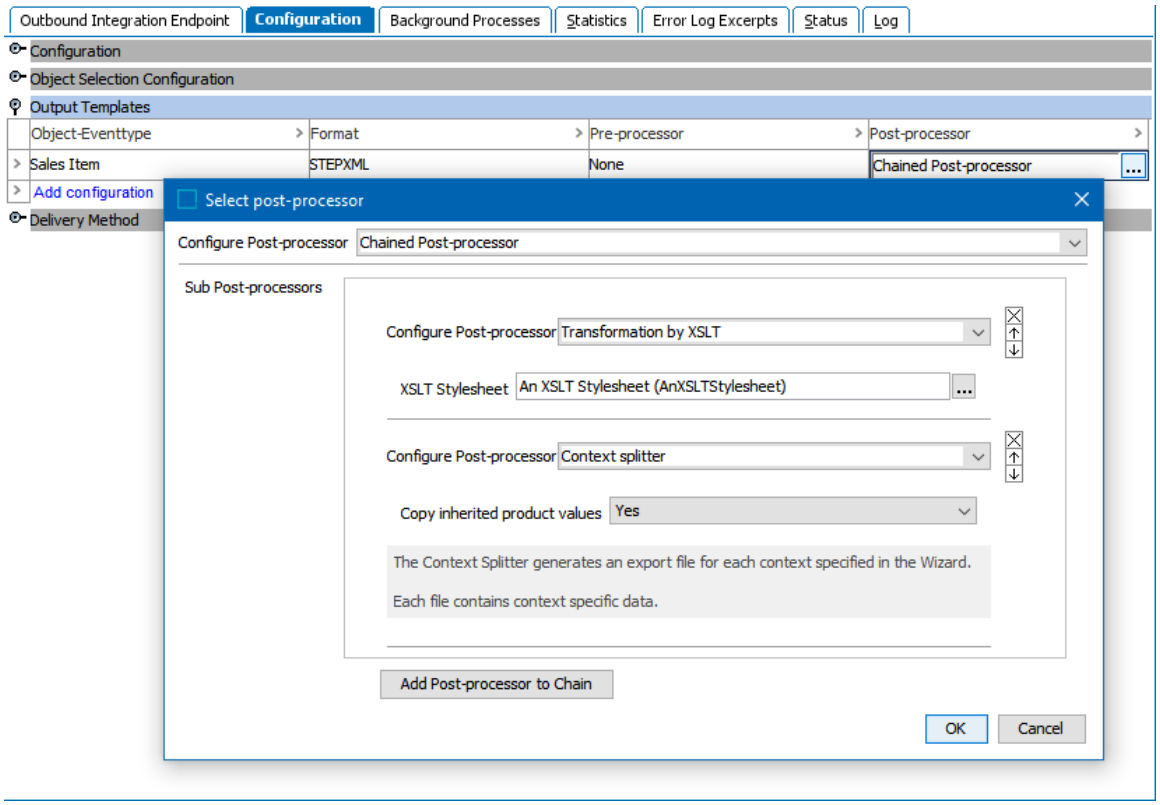
```
<Value AttributeID="DescriptionWeb" ContextID="Context3" QualifierID="Canada English">Filling and a great healthy eat
```

The default filename of the delivered XML is 'CopyContextDependent##exported_0.'However, the filename can be changed based on the selected delivery method, as defined in the **OIEP - Delivery Method Flipper** topic.

OIEP - Post-Processor - Chained Post-processor

The Chained Post-processor is a unique operation that allows users to stack multiple post-processors.

Configure the Chained Post-processor



1. On an OIEP, navigate to the 'Output Templates' section of the 'Configuration' tab.
2. Select the 'Post-processor' field, and then click the ellipsis button (...).
3. On the 'Select Post-processor' dialog, select the 'Chained Post-processor' option from the dropdown.

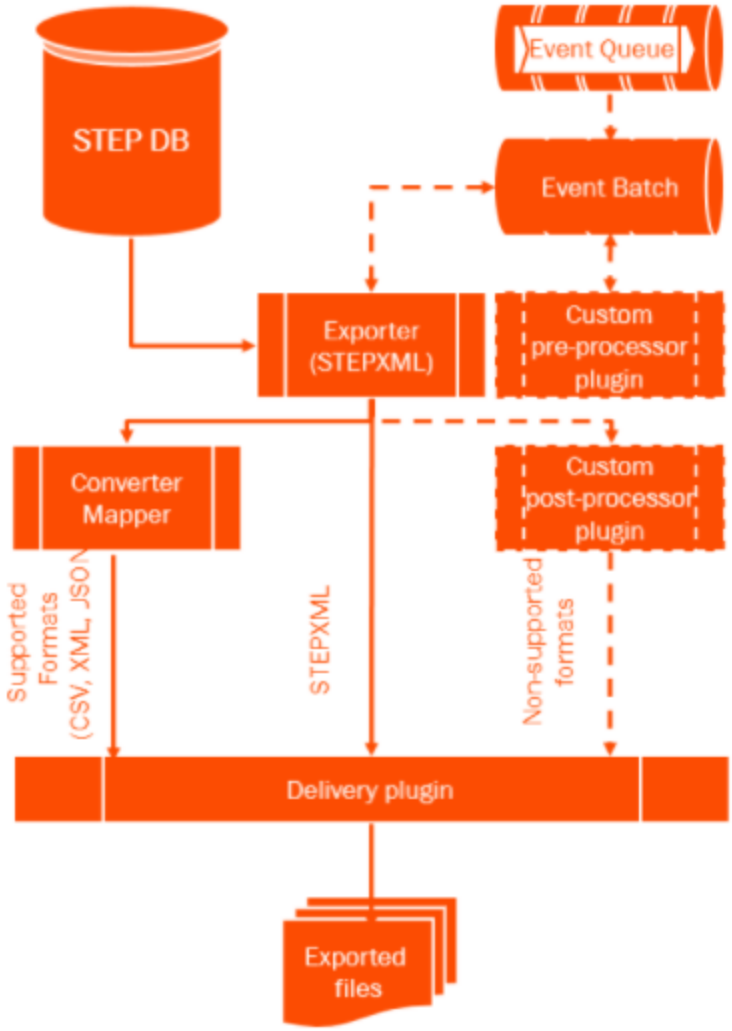
To configure the Chain Post-processor, users will then need to configure the additional post-processors independently. For more information, refer to the following post-processor topics:

- OIEP - Post-processor - Context Splitter
- OIEP - Post-processor - Copy Context Dependent Values and References
- OIEP - Post-processor - Transformation by XSLT

OIEP - Post-processor - Transformation by XSLT

The Transformation by XSLT post-processor allows an XSLT 2.0-compliant stylesheet to be used during export to transform outbound STEPXML files.

Note: To use this post-processor successfully, the outbound file or message size must be 50 MB or smaller.



Considerations

- The XSLT Post-processor requires STEP XML or Advanced STEPXML as output. To reduce the size of the output, configure Advanced STEPXML to limit the exported attributes.
- Generic XML output for XSLT is not supported. To resolve the error message 'Post-processor is not declared compatible with format converters, skipping format conversion', set the output as STEPXML or Advanced STEPXML.

Prerequisites

Before configuring the Transformation by XSLT post-processor, you must first:

1. Perform the **Initial Setup for XSLT Stylesheets**.
2. **Create an XSLT Stylesheet Object** - these steps must be performed for each separate stylesheet needed.

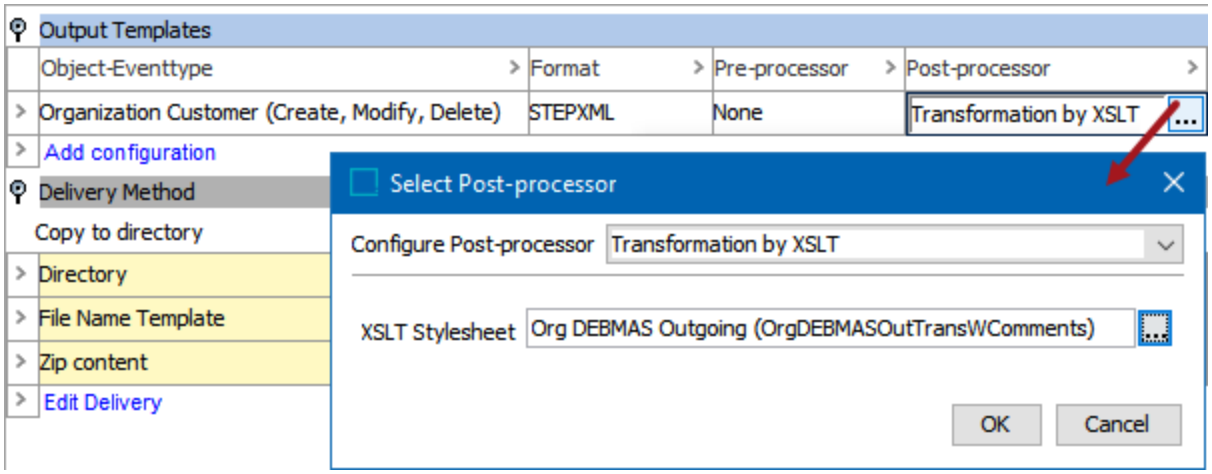
   For more information on these two steps, refer to the **IIEP - Configure Transformation by XSLT Pre-processor** section of the **Data Exchange** documentation.

Note: Saxon is an XSLT and XQuery processor. Information can be found via links on the **Open Source Components** page that can be found at the bottom of the list of topics in the left navigation panel of online help.

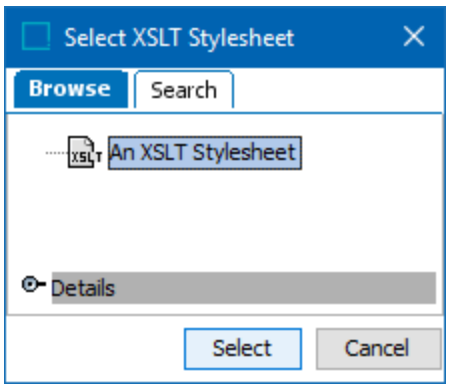
Configure the Transformation by XSLT Post-processor

Once the prerequisite tasks have been completed, you are ready to configure the post-processor in the OIEP configuration tab.

Note: The `<xsl:result-document>` tag is not supported. It is not possible to create multiple output documents from a single input document.



1. In the **Configure Post-processor** dropdown, select **Transformation by XSLT**.
2. In the **XSLT Stylesheet** parameter, click the ellipsis button (...) to display the Select XSLT Stylesheet dialog.
3. Select your stylesheet and click the **Select** button.

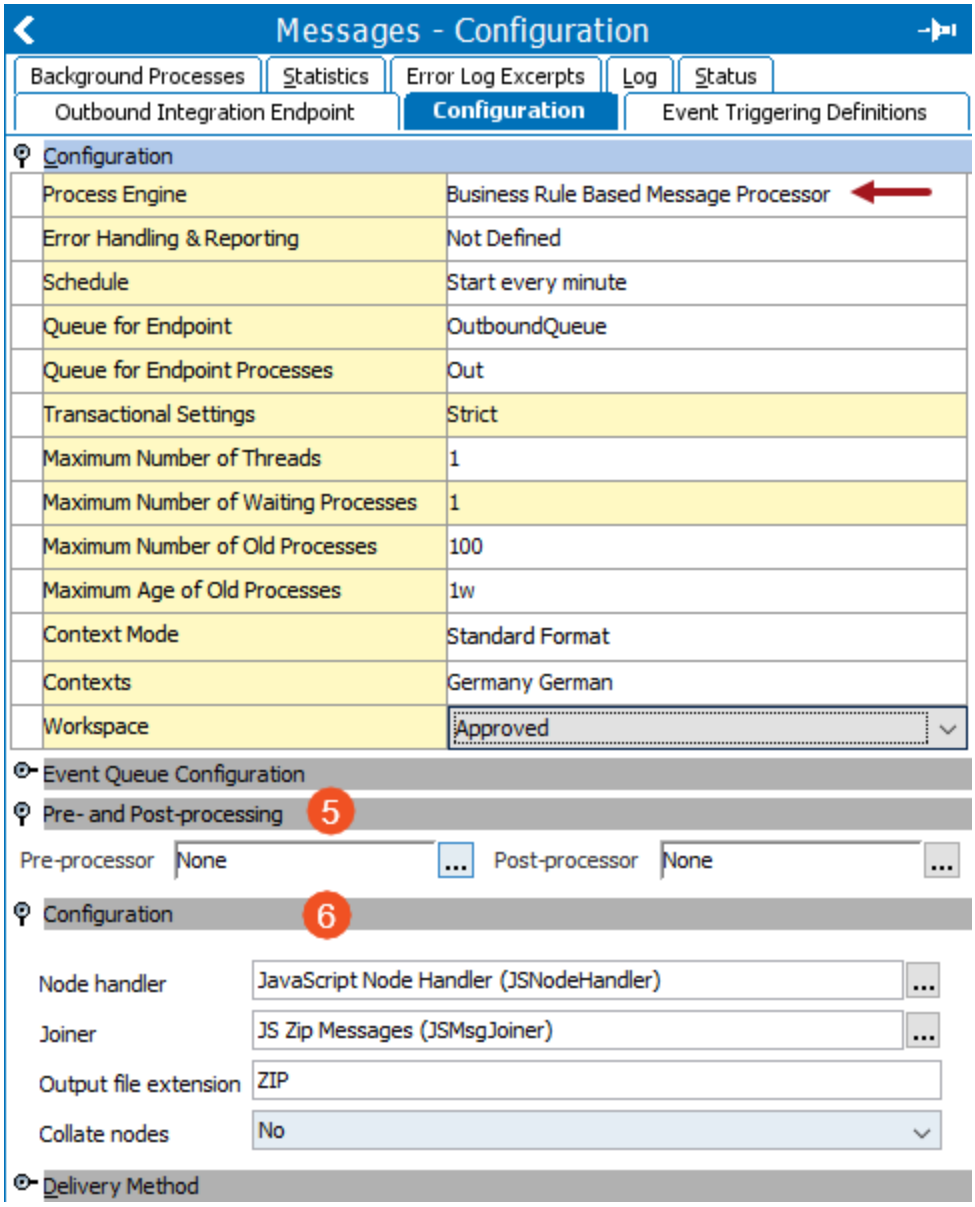


The exporter will use the selected XSLT stylesheet to transform the outbound file.

4. Click **OK** once the stylesheet has been selected.

OIEP - Event-Based - Pre- and Post- processing Flipper

The Pre and Post-processing flipper is available when the Business Rule Based Message processor is selected for an OIEP. This Configuration flipper includes the same parameters for both Event-Based and Select Objects endpoints. Each parameter is described below.



These are the same parameters that are available in the Output Templates Flipper when using the STEP Exporter process engine.

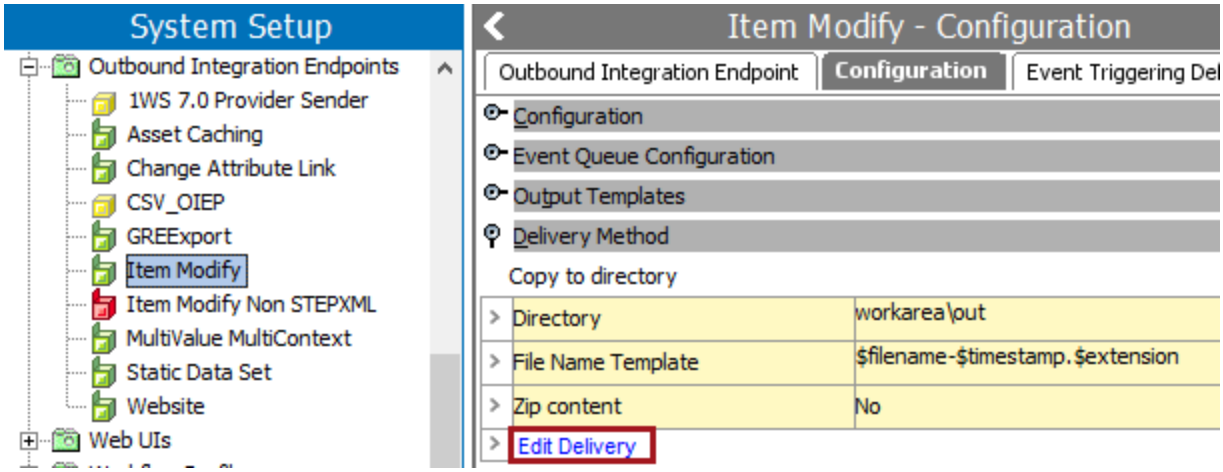
For details, refer to the **Configure the Pre-processor and Post-processor** section of the following topics:

- **OIEP - Select Objects - Output Templates Flipper**
- **OIEP - Event-Based - Output Templates Flipper**

OIEP - Delivery Method Flipper

The Delivery Method flipper in an OIEP includes the same parameters for both Event-Based and Select Objects endpoints. The delivery method controls how the output is delivered to an external system.

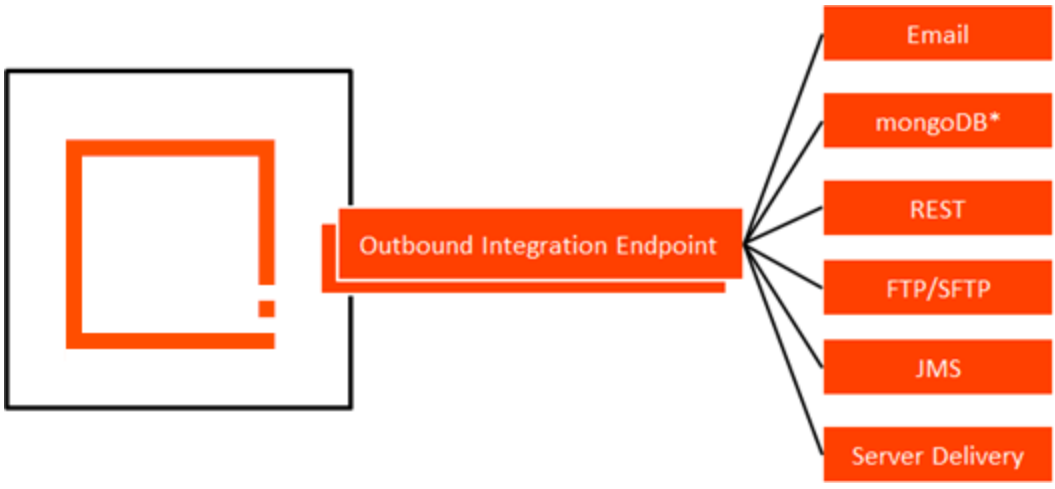
Open the Delivery Method flipper to display the selected delivery method. As shown below, the parameters vary based on the selected delivery method.



Delivery Methods

Delivery methods are used by both outbound data tools (Export Manager and OIEP) but the available options vary. For information on the delivery methods available in Export Manager, refer to **Export Manager - Select Delivery Method**.

An OIEP takes the data determined by its configuration and delivers it using the selected delivery method to the downstream system or on a directory on the application server.



The following delivery method options are available (although some are based on a license) when exporting using an OIEP.

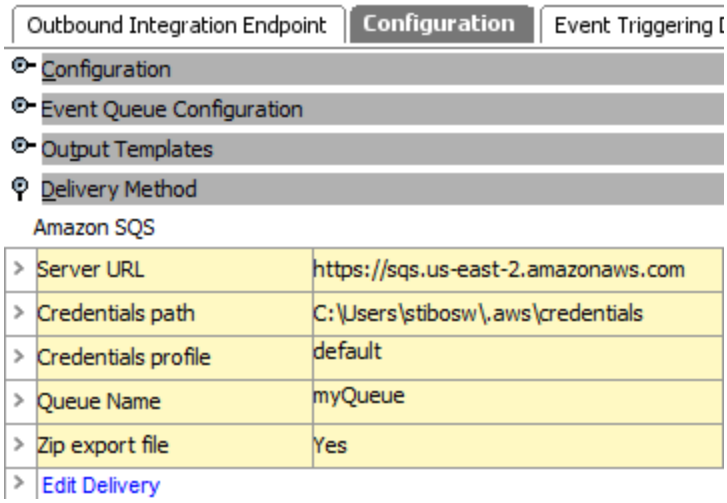
Method	Description
Amazon SQS	Delivers messages to the Amazon SQS (Amazon Simple Queue Service). Refer to **Amazon SQS Delivery Method** documentation.
Azure Blob Storage Delivery	Delivers files to Azure Blob Storage. Refer to **Azure Blob Storage Delivery Method** documentation.
Change Package Git Delivery	Delivers change packages to a Git repository. Refer to the **Integration Endpoint Options for VCS Integration** topic in the **Configuration Management** documentation.
Cloud Blob Storage Delivery	Delivers exported files to cloud storage. Amazon S3, Google Cloud Storage, and Microsoft Azure are supported. Refer to **Cloud Blob Storage Delivery Method** documentation.
Copy to Directory	Default selection for new OIEPs delivers files to a directory on the application server and allows the delivery to be zipped. Refer to **Copy to Directory Delivery Method** documentation.
Deploy	Delivers files to a directory on the application server. Does not allow the delivery to be zipped. Refer to **Deploy Delivery Method** documentation.
Dynamic JMS	The standard Dynamic JMS delivery options (Apache Active MQ, IBM MQ, and Oracle AQ) can be used to deliver messages to a Dynamic JMS Receiver without further customizations. Unlike the JMS delivery method mentioned above, this delivery method allows customers to supply the vendor-specific JMS libraries and JNDI configuration. Refer to the **Dynamic JMS Delivery Method** documentation.
Email	Delivers files as an email attachment and allows a zipped delivery. Refer to **Email Delivery Method** documentation.
FTP	Delivers files using file transfer protocol (FTP). Refer to **FTP Delivery Method** documentation.
Git Delivery	Delivers files produced by the OIEP processing engine or a configured post-processor to a branch in a remote Git repository. Refer to the **Integration Endpoint Options for VCS Integration** topic in the **Configuration Management** documentation.
GDSN	The GDSN solution must be implemented fully before the GDSN Data Pool method is available

Method	Description
Datapool	and functional. Refer to **GDSN Receiver Solution Enablement** documentation.
GDSN Receiver Datapool	The GDSN solution must be implemented fully before the GDSN Receiver Data Pool method is available and functional. Refer to **GDSN Receiver Solution Enablement** documentation.
IBM MQ SSL	IBM MQ SSL, uses MQ series Secure Sockets Layer, and enables data exchange across IBM and non-IBM platforms. Refer to **IBM MQ SSL Delivery Method** documentation.
JDBC	The JDBC delivery option enables delivery of data to RDBMS-type databases like Oracle, MySQL, MS SQL Server, PostgreSQL, etc. Refer to **JDBC Delivery Method** documentation.
JMS	The standard JMS delivery options (Apache Active MQ, IBM MQ, and Oracle AQ) can be used to deliver messages to a JMS Receiver without further customizations. Refer to **JMS Delivery Method** documentation.
Kafka Delivery	Apache Kafka is an open-source distributed event-streaming data platform. The Kafka delivery enables the STEP platform integrated with Apache Kafka to take advantage of built-in options for outbound processing to a Kafka queue. Refer to the **Kafka Delivery Method** documentation.
Mongo	Receives data from a STEP event queue and loads it into a MongoDB database. The MongoDB is often used for website back-end, reporting, and high performance feeds to other back-end systems. Refer to **Mongo Delivery Method** documentation.
Oracle AQ	Oracle Advanced Queuing (Oracle AQ) enables messages to be exchanged between two systems. Refer to **Oracle AQ Delivery Method** documentation.
Product Data Exchange 2	Product Data Exchange sends data to the Product Data Exchange (PDX) platform via the default PDX Outbound Integration Endpoint and the API. Refer to **Product Data Exchange 2 Delivery Method** documentation.
REST	The REST Delivery Method delivers a call-back URL to the REST service and does not include actual STEP data. Refer to **REST Delivery Method** documentation.
REST Direct	The REST Direct delivery method differs from the standard REST delivery method in that the data is delivered directly to the REST service and no call-back URL is required. Refer to **REST Direct Delivery Method** documentation.

Method	Description
SFTP	Delivers a file using the Secure File Transfer Protocol (SFTP). Refer to **SFTP Delivery Method** documentation.
Wiki	Delivers metadata to the metadata wiki platform. Refer to **Wiki Delivery Method** documentation.

Amazon SQS Delivery Method

The Amazon SQS Delivery Method delivers messages to the Amazon Simple Queue Service (Amazon SQS). This delivery option is only available in OIEPs. When multiple XML files are available to be delivered at the same time, they are sent individually (not concatenated into a single file).



To access the Amazon SQS Delivery method, a Delivery.AmazonSQS component must be activated on your system in addition to the normal update procedures. Contact your Stibo Systems representative for more information.

The Amazon SQS queues require a file size of less than 256KB. Common setup is to use file compression (zip export file) to reduce the message size to meet this limitation. Amazon S3 storage account is required to handle files 256KB or larger. For more information, search Amazon S3 online.

Prerequisites

Changes to the properties file, outlined below, are implemented when the server is restarted.

1. For the 'Credentials profile' parameter, create a credentials file following the instructions at http://docs.aws.amazon.com/cli/latest/userguide/cli-config-files.html. Place the credentials file (named 'credentials', without any file extension) in the required path on the application server. This file is used by both the OIEP Amazon SQS Delivery Method and the IIEP Amazon SQS Receiver.
2. Prior to configuration, clicking the **Server URL** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **SQSServerUrl** property. The following is an example of a property entry for a single server:

   ```
   SQSServerUrl=1=https://sqs.us-east-2.amazonaws.com
   ```

3. Prior to configuration, clicking the **Credentials Path** dropdown parameter displays the required property name. Provide a selection, including the name of the credentials file, for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **SQSCredentialsPath**

property. If necessary, use a comma and increment the number to separate multiple paths as shown in the example below.

```
SQSCredentialsPath=1=C:\\Users\\stibosw\\.aws\\credentials,2=C:\\Users\\stibosw\\.aws\\credentials123
```

4. Prior to configuration, clicking the **S3 Bucket Name** dropdown parameter displays the required property name. If Amazon S3 is being used, provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **SQSDeliveryS3BucketName** property. The following is an example of a property entry for a two buckets where a comma separates multiple buckets:

```
SQSDeliveryS3BucketName=1=AmazonBucketName1,2=AmazonBucketName2
```

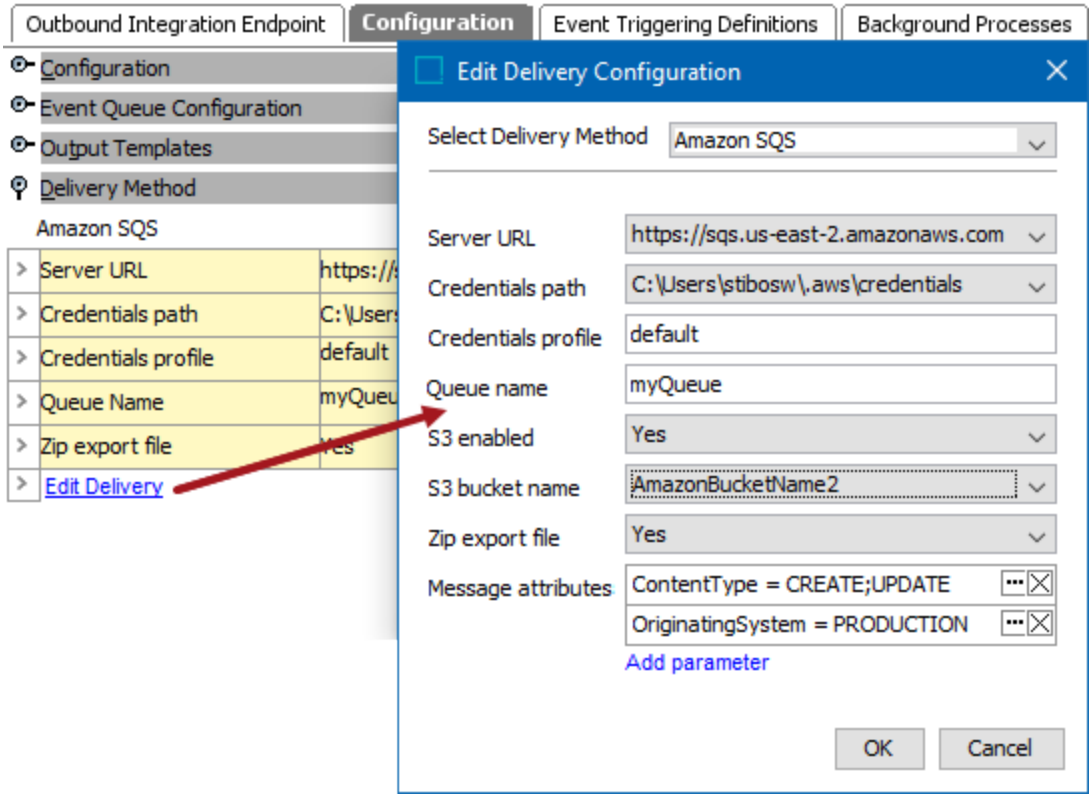
5. For the 'Queue name' parameter, if necessary, create a new queue on the SQS server. Use the steps defined in the **Amazon SQS Queue Configuration** section below.
6. If necessary, establish the Amazon S3 storage account and create the required buckets. For details, refer to Amazon S3 on the web.
7. If message attributes are required, create up to seven (7) string Message Attributes on the Amazon website. The message attribute names in Amazon correspond to the Key parameters created in the delivery method configuration below.

Contact your Stibo Systems representative if you need assistance with setup.

Configuration

After completing the prerequisite steps, edit the delivery method of the OIEP. Use the following steps to configure the OIEP to use the Amazon SQS delivery option.

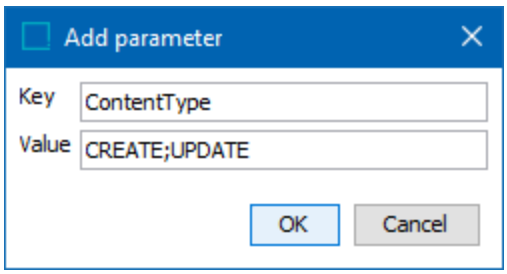
For information on a parameter, hover over the parameter field to display help text.



1. For **Select Delivery Method**, choose **Amazon SQS**.
2. In **Server URL**, select the URL that points to the SQS server where the data will be delivered.
3. In **Credentials path**, select the path to the desired SQS credentials file.
4. In **Credentials profile**, enter the profile name included in the SQS credentials profile.
5. In **Queue Name**, enter the Amazon queue created for this delivery. This queue is also required when Amazon S3 is being used.
6. For **S3 Enabled**, select 'Yes' if you have an Amazon S3 storage account. Setting this option to 'No' results in an error for messages that exceed the 256KB size limit.

   Note: When 'S3 Enabled' = Yes, the Amazon framework sends messages that are smaller than 256KB continue to be processed without S3 using Amazon SQS.

7. In **S3 Bucket Name**, select the Amazon bucket name that should be used for this delivery. This name must already be established on the Amazon account.
8. In **Zip export file**, specify whether to zip the contents before delivery.
9. **Message Attributes** provides user-defined data that is visible within Amazon SQS (refer to the **Amazon SQS View Message Attribute** section below) and is specific to the OIEP. Click the **Add parameter** link to create a message attribute, then enter a text value for the Key and the Value parameter. In the image above, the Key = ContentType, and the Value = CREATE;UPDATE, which could be used to indicate that the OIEP is processing both create and update events.

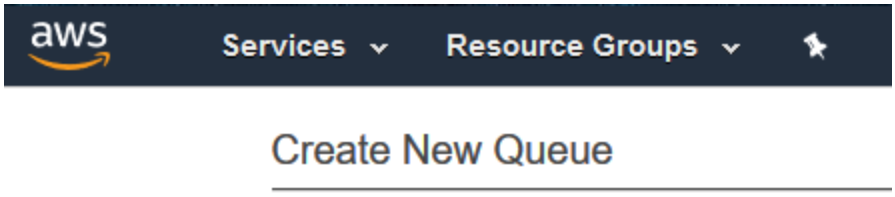


10. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

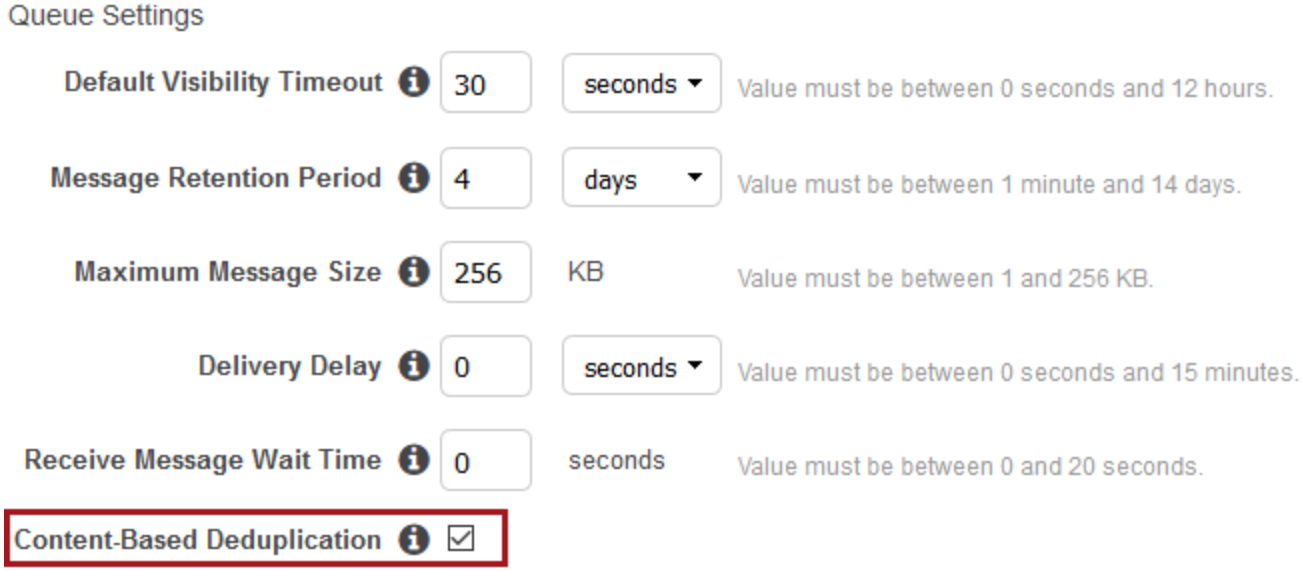
Amazon SQS Queue Configuration

When required, use the following steps to create a new Amazon SQS queue.

1. Log in to the Amazon server based on your region.
2. Search for **Simple Queue Service** and select it to display the queue list page.
3. Click the **Create New Queue** button to display the Create New Queue page.



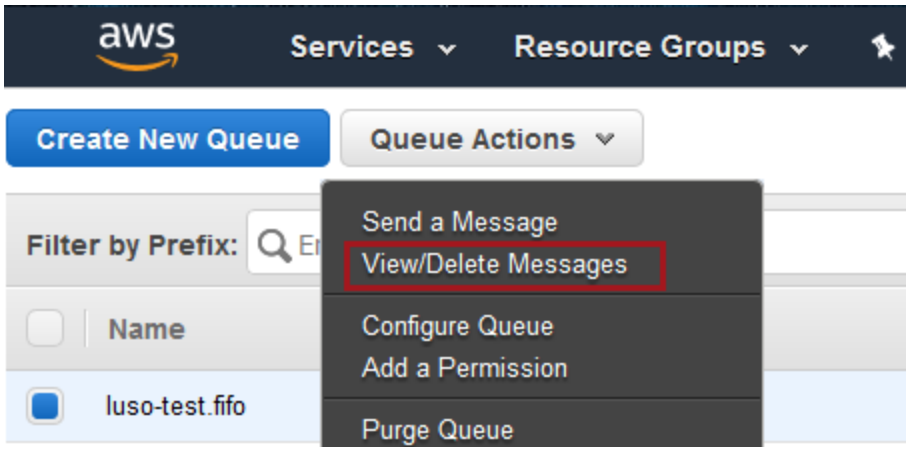
4. For **Queue Name**, type a queue name including the '**.fifo**' extension.
5. For the **type of queue** section, select **FIFO Queue**.
6. Click the **Configure Queue** button at the bottom of the page to display the default Queue Attributes parameters.
7. Check the **Content-Based Deduplication** parameter box to enable the queue.



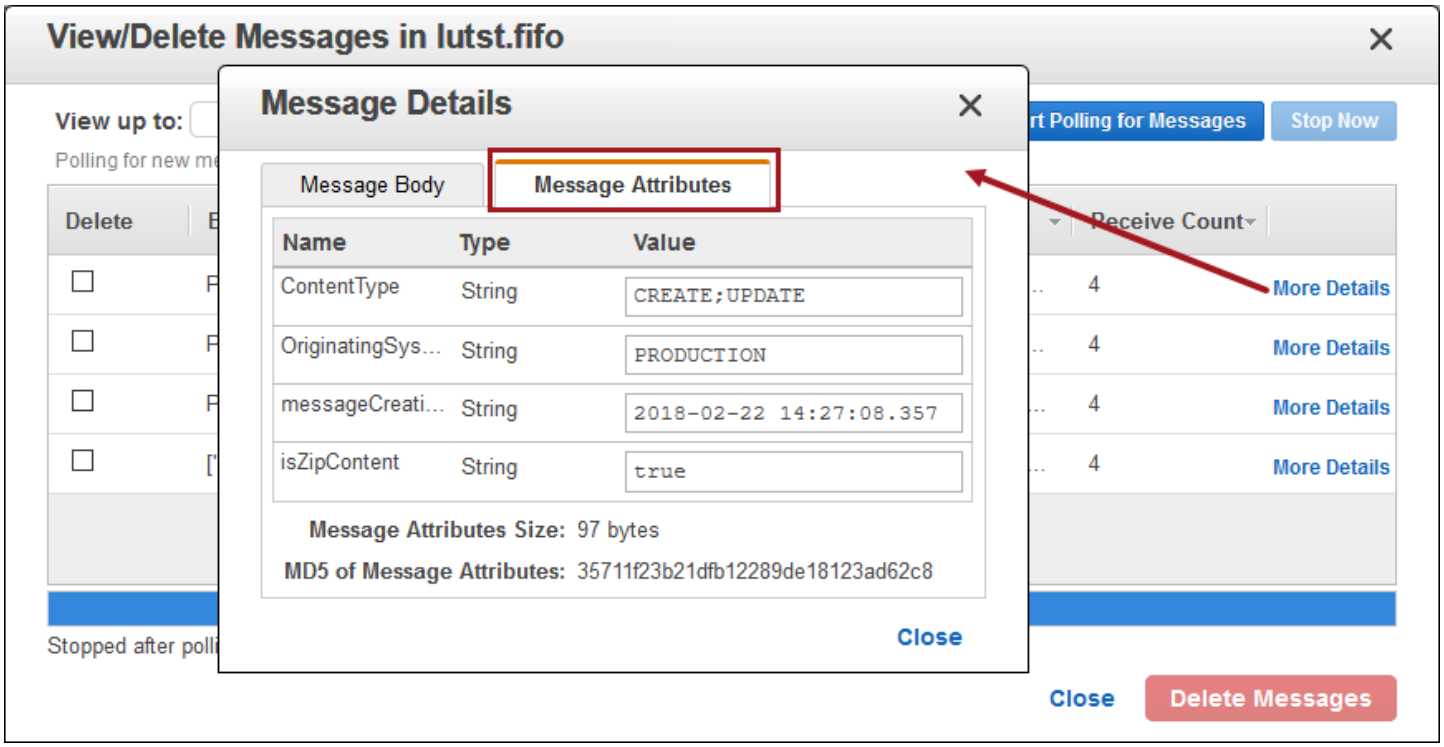
8. Click the **Create Queue** button. The new queue is displayed in the list.

Amazon SQS View Message Attributes

1. Log in to Amazon SQS and display your queue(s).
2. Right-click the desired queue to display the menu (or click the Queue Actions dropdown button) and click the **View/Delete Messages** option.



3. Click the **More Details** link to display the Message Details dialog.



4. On the Message Details dialog, click the **Message Attributes** tab to display the standard and user-defined message attributes.

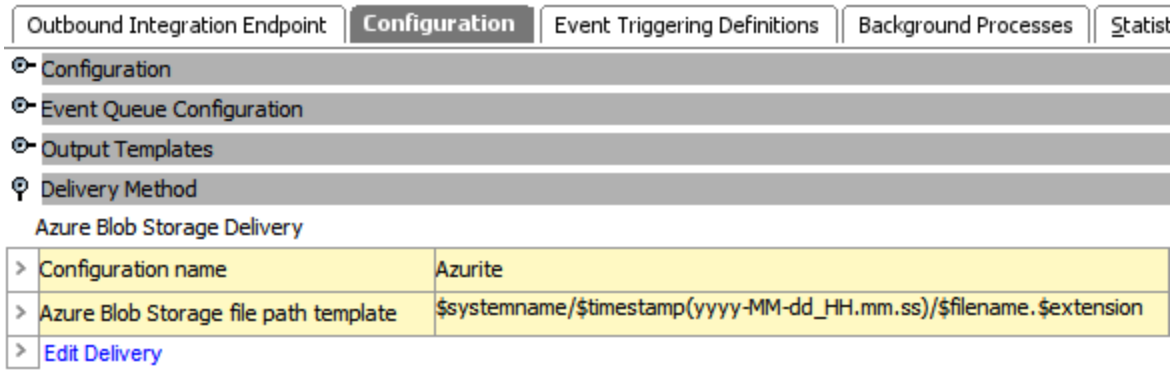
Note: In addition to the user-defined message attributes, two standard message attributes are included: **messageCreationTime** displays the delivery datetime, and **isZipContent** indicates the status of the 'Zip export file' parameter.

Azure Blob Storage Delivery Method

The Azure Blob Storage Delivery plugin for outbound integration endpoints makes it possible to deliver files to Azure Blob Storage. This delivery method is available in OIEPs and Export Manager.

Note: To deliver OIEP files to Azure blob storage, the Cloud Blob Storage Delivery Method can also be used. The differentiation between the setup and functionalities is that the Cloud Blob Storage Delivery plugin is part of the STEP baseline, can also be used for Amazon S3, and the Export Manager and OIEP configurations use the gateway integration endpoints versus having separate delivery method integration properties.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



To use this delivery method in Export Manager, refer to the **Azure Blob Storage Delivery Method** topic.

Prerequisites

The Azure Blob Storage Delivery plugin is part of the 'cloudstorage-azure' component that must be installed in addition to the STEP baseline. No additional licenses are required.

Prior to configuration, click the **Configuration name** dropdown parameter to display the required configuration to be used. You will provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the following configuration properties:

```
AzureBlobStorageDeliveryPlugin.ConfigurationNames
AzureBlobStorageDeliveryPlugin.ConnectionString.[Configuration Name]
AzureBlobStorageDeliveryPlugin.ContainerName.[Configuration Name]
```

The value for 'AzureBlobStorageDeliveryPlugin.ConfigurationNames' is a comma-separated list of user-defined names for the desired Azure Blob Storage configurations. For each name, corresponding 'AzureBlobStorageDeliveryPlugin.ConnectionString.[Configuration Name]' and 'AzureBlobStorageDeliveryPlugin.ContainerName.[Configuration Name]' properties must be set with the values being the connection string for the configuration and the desired blob container name, respectively.

An example configuration using the storage account access key (*AccountName* / *AccountKey*) method is below:

```
AzureBlobStorageDeliveryPlugin.ConfigurationNames=Azurite

AzureBlobStorageDeliveryPlugin.ConnectionString.Azurite=DefaultEndpointsProtocol=http;AccountName=devstoreaccount1;AccountKey=Eby8vdM02xNOcqFlqUwJPLlmEtlCDXJ1OUzFT50uSRZ6IFsuFq2UVErCz4I6tq/K1SZFPTOtr/KBHBeksoGMGw==;BlobEndpoint=http://127.0.0.1:10000/devstoreaccount1;
AzureBlobStorageDeliveryPlugin.ContainerName.Azurite=productData
```

It is also possible to use a Shared Access Signature (SAS) credential for the ConnectionString.

The SAS token must be created directly on the blob storage account itself (and not the corresponding container); and as a minimum, it must have **Service**, **Container**, and **Object** specified as its 'Allowed resource types' as well as **Read**, **Write**, and **List** for its 'Allowed permissions' to grant the proper access rights to STEP.

These resource types and permissions are required to allow STEP to perform all the needed operations to deliver the content (blobs) to the specified Azure Blob Storage account's container.

Important: If the SAS token has insufficient privileges, the delivery will result in an error message similar to this one:

If you are using a SAS token, and the server returned an error message that says 'Signature did not match', you can compare the string to sign with the one generated by the SDK. To log the string to sign, pass in the context key value pair 'Azure-Storage-Log-String-To-Sign': true to the appropriate generateSas method call. Remember to disable 'Azure-Storage-Log-String-To-Sign' before going to production as this string can potentially contain PII.

Status code 403, "<?xml version="1.0" encoding="utf-8"?><Error><Code>AuthorizationResourceTypeMismatch</Code><Message>This request is not authorized to perform this operation using this resource type. RequestId:836910b1-801e-001a-4da2-900fc9000000 Time:2022-07-05T19:11:07.4796215Z</Message></Error>"

An example using a SAS configuration is below:

```
AzureBlobStorageDeliveryPlugin.ConfigurationNames=bestProducts
AzureBlobStorageDeliveryPlugin.ConnectionString.bestProducts=
BlobEndpoint=https://pimtest.blob.core.windows.net/;SharedAccessSignature=sv=2021-06-08&ss=b&srt=sco&sp=rwlx&se=2023-01-01T05:00:00Z&st=2022-07-05T18:58:41Z&spr=https&sig=N5X7J9tCMscbOTYioR4sb30H7B%2B0j8dk74MSCQ6Gxsw%3D
AzureBlobStorageDeliveryPlugin.ContainerName.bestProducts=productData
```

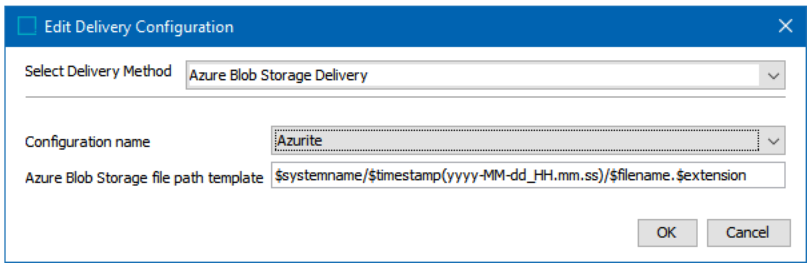
Once the property configuration is in place and the system has been restarted, the configuration name(s) will appear as selectable options in the STEP Workbench.

Configuration

1. On the **Configuration** tab, in the **Delivery Method** flipper, click **Edit Delivery**.
2. In **Select Delivery Method**, choose **Azure Blob Storage Delivery**.

3. The **Configuration name** comes from the properties you set above. If you configured the properties to use multiple configuration names within a comma-separated list, then you will choose from the dropdown.

   Example shown below:



4. Configure the Azure Blob Storage file path template.

   In addition to selecting the appropriate connection string and container, it is possible to make use of Azure Blob Storage virtual directories. Four variables are available:

   - $systemname – The STEP system name. Useful when several STEP systems are delivering to the same blob container.
   - $timestamp([YMDHMS]) – Delivery timestamp. Desired format to be supplied in Java SimpleDateFormat compatible format.
   - $filename – Name of the file produced by the outbound integration endpoint.
   - $extension – The extension of the file produced by the outbound integration endpoint.

If the blob containers and virtual directories do not already exist, the plugin will create them.

Important: Existing files with the same virtual directory path and file name will be overwritten.

5. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

Change Package Git Delivery Method

The Change Package Git Delivery method allows integration with popular repositories, supporting the HTTPS (token-based) or the SSH (file-based) access methods for GitHub, GitLab, and Bitbucket.

The Change Package Git Delivery method delivers files produced by the OIEP processing engine using an integrated STEPXML Splitter to deliver multiple files to a branch in a remote Git repository. Refer to https://git-scm.com for more information about Git.

The change package is represented below the specified branch within a configurable directory structure. At the end of the directory structure, change packages grouping files by the Primary and Secondary flippers, then by type of object with XML or JSON files named by object type and the ID of the object.

Important: For on-premises systems, this feature requires the configuration-management component.

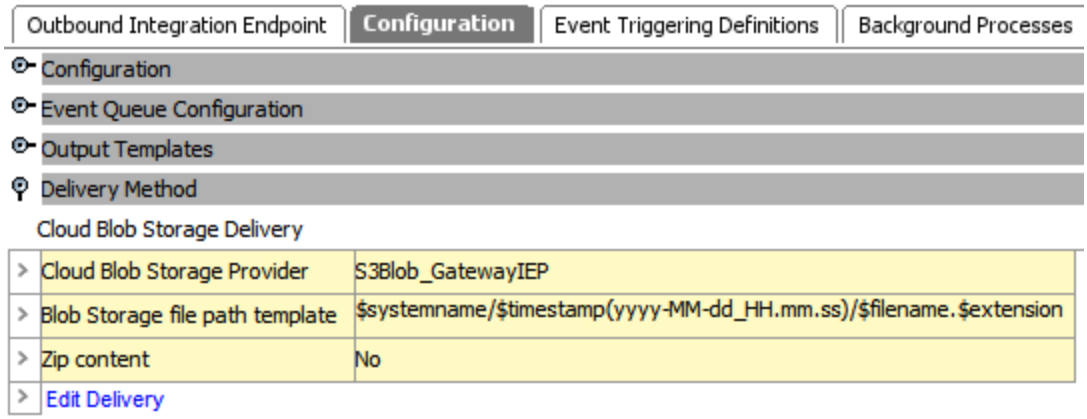
For details, refer to the **Integration Endpoint Options for VCS Integration** topic in the **Configuration Management** documentation.

Cloud Blob Storage Delivery Method

The Cloud Blob Storage Delivery method delivers exported files to cloud storage. Amazon S3, Google Cloud Storage, and Microsoft Azure are supported. This delivery method is available in Export Manager and OIEPs.

Note: The Cloud Blob Storage Delivery Method is similar to the legacy Azure Blob Storage Delivery Method. The difference between the setup and functionalities is that the Cloud Blob Storage Delivery plugin is part of the STEP baseline, can also be used for Amazon S3 and Google Cloud Storage, and the Export Manager and OIEP configurations use the gateway integration endpoints versus having separate delivery method integration properties.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



To use this delivery method in Export Manager, refer to the **Cloud Blob Storage Delivery Method** topic.

Prerequisites

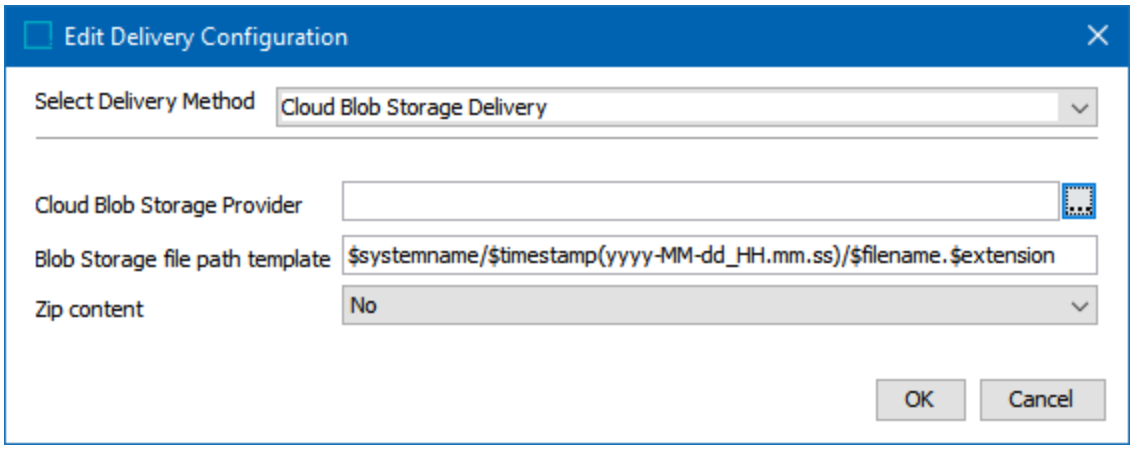
A blob storage gateway integration endpoint must be configured before moving ahead with the Export Manager or OIEP setup.

- For Amazon S3, directions for setting up this endpoint are in the **Configuring a Gateway Integration Endpoint - Amazon S3 Blob Storage** topic in the **Data Exchange** documentation.
- For Google Cloud Storage (GCS), directions for setting up this endpoint can be found in the **Configuring a Gateway Integration Endpoint - Google Cloud Storage** topic in the **Data Exchange** documentation.
- For Microsoft Azure (ABS), directions for setting up this endpoint can be found in the **Configuring a Gateway Integration Endpoint - Microsoft Azure Blob Storage** topic in the **Data Exchange** documentation.

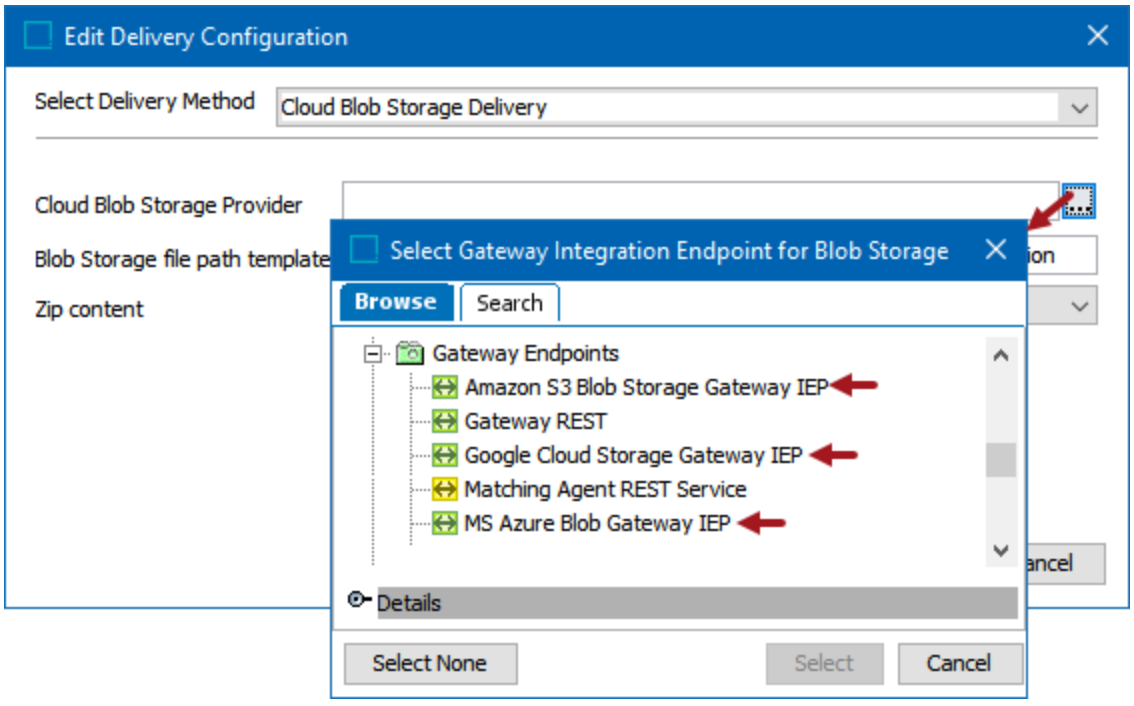
Also, buckets (S3 and GCS) and containers (ABS) must be created. If they do not exist in advance, you will get an exception error during the delivery process.

Configuring the OIEP Delivery

For information on a parameter, hover over the parameter label to display help text.

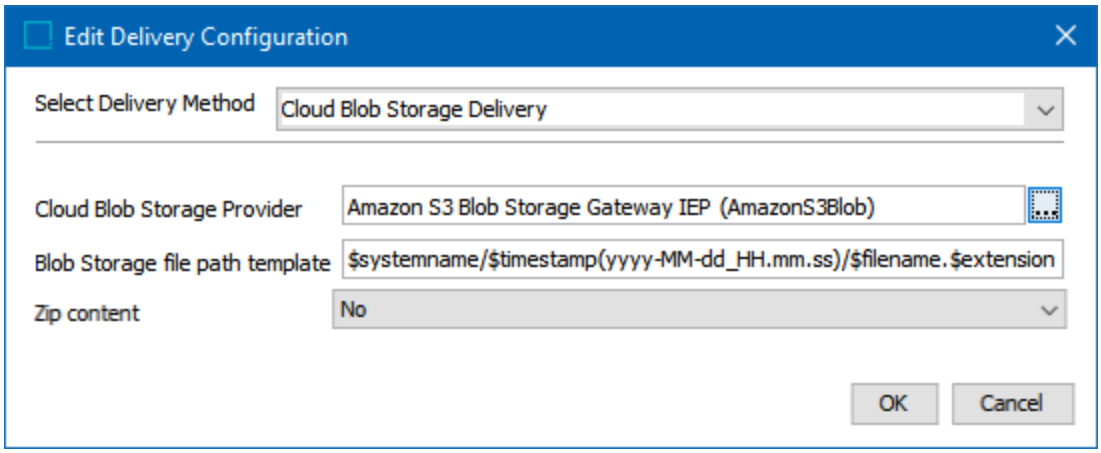


1. From the **Select Delivery Method** parameter dropdown, choose **Cloud Blob Storage Delivery**.
2. Click the ellipsis button (...) to the right of the **Cloud Blob Storage Provider** parameter, select a gateway endpoint configured for blob storage, and click the **Select** button. The name and ID of the selection display in the Edit Delivery Configuration dialog.



3. Configure the **Blob Storage file path template** using the available variables:

   - $systemname – The STEP system name. Useful when several STEP systems are delivering to the same blob container.
   - $timestamp([YMDHMS Format]) – Delivery timestamp. Desired format to be supplied in Java SimpleDateFormat compatible format.
   - $filename – Name of the file produced.
   - $extension – The extension of the file produced.



   In addition to selecting the appropriate connection string and container, you can make use of Azure Blob Storage virtual directories. If the virtual directories do not already exist, the plugin creates them. As mentioned in the **Prerequisites** section, buckets (S3 and GCS) and containers (ABS) must exist in advance, or you will get an exception error during the delivery.

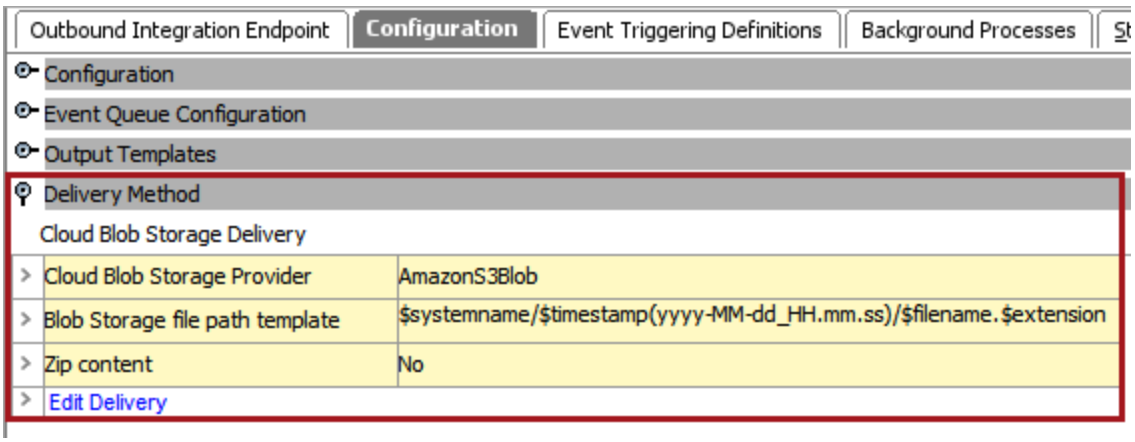
   Important: Existing files with the same virtual directory path and file name are overwritten.

4. In **Zip content**, select 'yes' or 'no' from the dropdown to specify if the output file should be delivered in a .ZIP (compressed) file format.

   - **Yes** uses 'export-' before the timestamp variable, and then the extension ZIP. For example, a zipped STEPXML output with the default File Name Template exported on 15 NOV 2016 results in an output .ZIP file named 'export-1479230247017.zip.' The contents of the ZIP file follow the $filename variable applied in the 'File name template' parameter.
   - **No** uses the file type for the file name along with the appropriate extension for the selected data format.

5. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

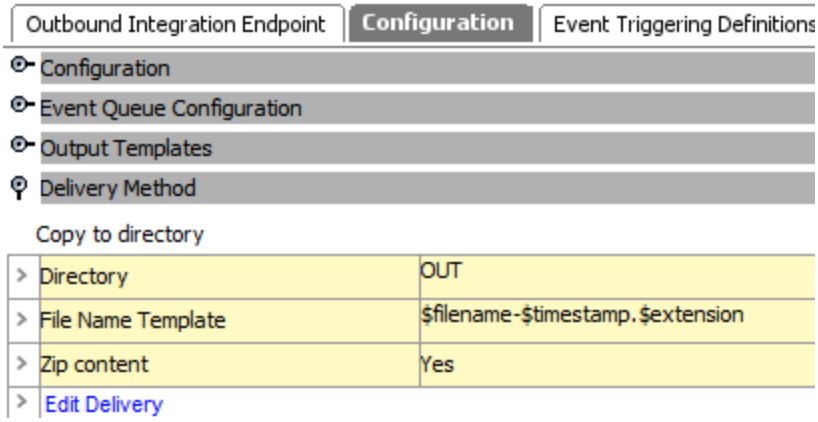
   The configuration is displayed in the OIEP editor.


Outbound Integration Endpoint
Configuration
Event Triggering Definitions
Background Processes
Configuration
Event Queue Configuration
Output Templates
Delivery Method
Cloud Blob Storage Delivery
Cloud Blob Storage Provider
AmazonS3Blob
Blob Storage file path template
$systemname/$timestamp(yyyy-MM-dd_HH.mm.ss)/$filename.$extension
Zip content
No
Edit Delivery

Copy to Directory Delivery Method

For OIEPs, there is no single delivery option for server-side delivery, but the 'Deploy' and 'Copy to directory' options provide the same function in that they both deliver files to a directory on the application server. Common setup is to use 'Copy to directory' since it is more flexible because it allows the delivery to be zipped. Neither of these options is available in Export Manager.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



For more information, refer to the **Deploy Delivery Method** documentation.

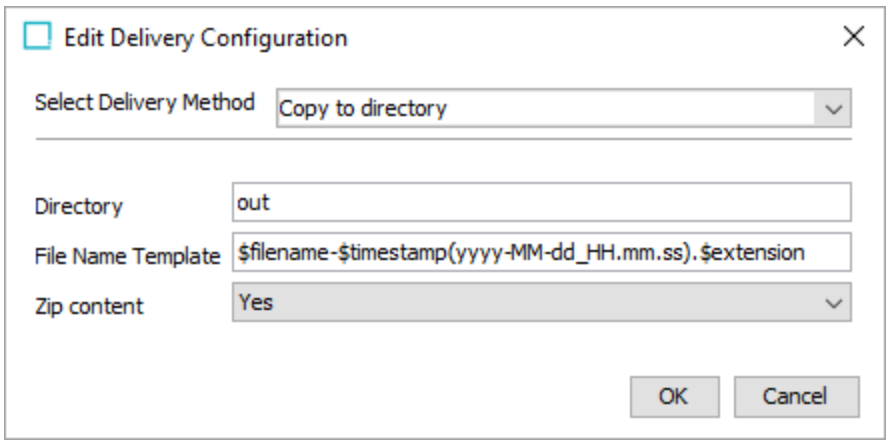
Prerequisites

The default delivery path is **opt/stibo/step** on the application server. This can be modified via the sharedconfig.properties file on the STEP application server using the case-sensitive **DirectoryDeliveryPlugin.RootDir** property. Changes to the properties file are implemented when the server is restarted.

Configuration

For information on a parameter, hover over the parameter field to display help text.

1. On the **Configuration** tab, navigate to the **Delivery Method** flipper, then click **Edit Delivery**.
2. Click the **Select Delivery Method** dropdown and choose **Copy to Directory**.



3. In **Directory**, specify the directory on the application server that will house the exported files. Do not start this text with a slash (/). Text entered will be appended to the path as defined in the Prerequisites section.

   In the image above, assuming the default path is being used, the file will be delivered to 'opt/stibo/step/out.'

4. In **File Name Template**, to create the file name used for the output, enter template variables (identified with a $), static text, or use a combination of both. The default template is $filename-$timestamp(yyyy-MM-dd_HH.mm.ss).$extension. Depending on the settings of the configuration, the default template outputs a file name similar to 'GenericXML-2020-07-30_14.07.44.xml.' Each variable is described below:

   Note: The 'Zip context' parameter also has an impact on the file name.

   - **$filename** For event-based OIEPs, this variable is replaced with text to indicate the format of the delivered file as specified in the Output Templates flipper, except for STEPXML when the first and last Event IDs are used. For example, the output file name could be 'csv-2020-07-30_14.09.40.csv' or '1804038-1804038.xml' to indicate that STEPXML was used for a single event. When exporting multiple contexts, '##' is used as a separator within the filename to distinguish the context name for each output.
   - **$timestamp**: This variable is replaced with the time in year-day-month-hour-minutes-seconds-milliseconds, excluding the dashes. For example, 20161511094709625 is used for an export on 15 NOV 2016 at 9:47:09:625 AM.

     Adding a definition for the arrangement of the date / time elements provides a way to generate a more readable date. Static characters can be included to separate the timestamp elements and individual elements can be ordered as required, as shown below.

     Note: To revert the **$timestamp** variable so that it uses the legacy definition (milliseconds between January 1, 1970, and the time when the file is uploaded), contact Stibo Systems Support for more information.

     $timestamp(ddMMyyyy): The timestamp is modified to display day-month-year (excluding dashes) and is output as '15112016' for 15 NOV 2016.

$timestamp(yyyy-MM-dd_HH.mm.ss): The timestamp is modified to display year-month-day_ hour.minutes.seconds (including the dashes, underscore, and periods) and is output as '2016-11-15_ 09.32.43' for 2016 NOV 15 at 9:32.43 AM.

- **$extension** This variable is replaced with the extension of the selected output file format.

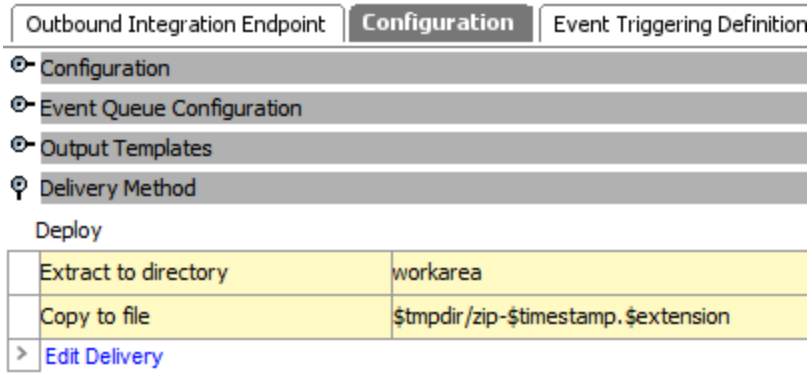
Note: The File Name Template does not support the conversion of file formats and can only be used to deliver files in the format specified on the Configuration tab of the outbound integration endpoint.

5. For **Zip content**, select an option from the dropdown to specify if the output file should be delivered in a .ZIP (compressed) file format.
   - **Yes** uses 'result_0' before the timestamp variable, and then the extension ZIP. For example, a zipped STEPXML output with the default File Name Template exported on 7 JUL 2020 at 2:07 p.m. results in an output .ZIP file named 'result_0-2020-07-30_14.07.44.zip.' The contents of the ZIP file follow the $filename variable applied in the 'File name template' parameter.
   - **No** uses the 'File Name Template' parameter for the file name along with the appropriate extension for the selected data format.
6. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

Deploy Delivery Method

For OIEPs, there is no single delivery option for server side delivery, but the 'Deploy' and 'Copy to directory' options provide the same functionality in that they both deliver files to a directory on the application server. Common setup is to use 'Copy to directory' since it is more flexible because it allows the delivery to be zipped. Neither of these options is available in Export Manager.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



For more information, refer to the **Copy to Directory Delivery Method** documentation.

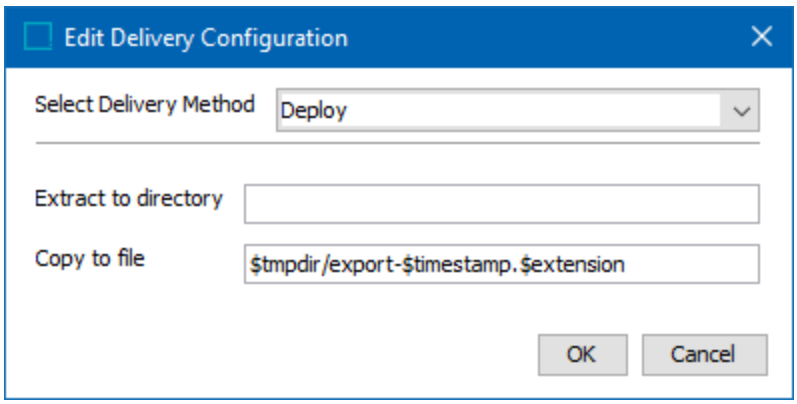
Prerequisites

The default delivery path is **opt/stibo/step** on the application server. This can be modified via the sharedconfig.properties file on the STEP application server using the case-sensitive **DirectoryDeliveryPlugin.RootDir** property. Changes to the properties file are implemented when the server is restarted.

Configuration

For information on a parameter, hover over the parameter field to display help text.

1. On the **Configuration** tab, navigate to the **Delivery Method** flipper, then click **Edit Delivery**.
2. Click the **Select Delivery Method** field to display the dropdown and choose **Deploy**.



3. In **Extract to Directory**, specify a path on the application server where the exported .ZIP file should be unzipped. The field will only have an effect if the exported file is a .ZIP file, otherwise this field will be ignored.
4. In **Copy to file**, specify a location and file name template to define a path where exported file should be delivered. Static text is also allowed in addition to, or instead of, the template options (which are indicated by the initial $). The elements in the example template shown in the image are described below:
   - **$tmpdir** This variable points to the directory path defined by the **ApplicationServer.TmpDir** entry in the sharedconfig.properties file. Remove '$tempdir' to enforce a location relative to the **DirectoryDeliveryPlugin.RootDir** delivery path in the sharedconfig.properties file.
   - **export** This represents and can be replaced with any static text required in the file name. For example, the Copy to File field could read 'out/EmergencySKUExport-$timestamp.$extension' instead, and would deliver the output to the 'out' directory below the path indicated in the properties file, and the file name would start with the text 'EmergencySKUExport-' before adding the timestamp, a period, and the file extension.
   - **$timestamp**: This variable is replaced with the time in year-day-month-hour-minutes-seconds-milliseconds, excluding the dashes. For example, 20161511094709625 is used for an export on 15 NOV 2016 at 9:47:09:625 AM.

     Adding a definition for the arrangement of the date / time elements provides a way to generate a more readable date. Static characters can be included to separate the timestamp elements and individual elements can be ordered as required, as shown below.

     Note: To revert the **$timestamp** variable so that it uses the legacy definition (milliseconds between January 1, 1970, and the time when the file is uploaded), contact Stibo Systems Support for more information.

     $timestamp(ddMMyyyy): The timestamp is modified to display day-month-year (excluding dashes) and is output as '15112016' for 15 NOV 2016.

     $timestamp(yyyy-MM-dd_HH.mm.ss): The timestamp is modified to display year-month-day_ hour.minutes.seconds (including the dashes, underscore, and periods) and is output as '2016-11-15_ 09.32.43' for 2016 NOV 15 at 9:32.43 AM.
   - **$extension** This variable is replaced with the extension of the selected output file format.

Note: The 'Deploy' template does not support conversions of file formats and can only be used to deliver files in the format specified on the Configuration tab of the outbound integration endpoint.

5. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

Dynamic JMS Delivery Method

The Dynamic JMS Delivery method uses the Java Message Service (JMS) messaging standard to deliver data to external systems. Unlike the JMS Delivery method, the Dynamic JMS Delivery method allows customers to supply the vendor-specific JMS libraries and JNDI configuration and, for example, upgrade to newer versions of these without Stibo Systems' involvement.

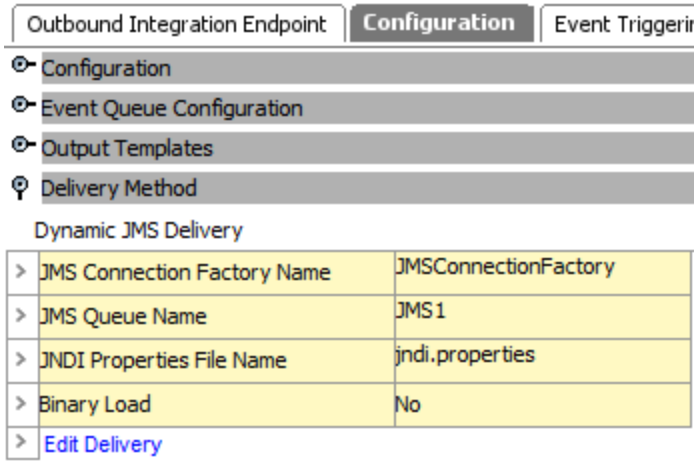
Important: This standard functionality only supports queues. Support for topics requires custom development via the **Extension API** (available from the Start Page using the **STEP API Documentation** link). Alternatively, topics can be supported using middleware to move the message from a queue to a topic.

The delivery method has been designed to work with any Java Message Service 2.0 specification-compliant client library (specification defined by JSR 343: https://jcp.org/aboutJava/communityprocess/final/jsr343/index.html), but will also work with systems that implement older versions of the specification.

Note: The delivery method has been tested with the following message brokers: Apache Active 5.15.8 and RabbitMQ 3.7.10.

This delivery option is only available in OIEPs.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



Prerequisites

Changes to the properties file or any files found in the class path directory, as outlined below, are implemented when the server is restarted.

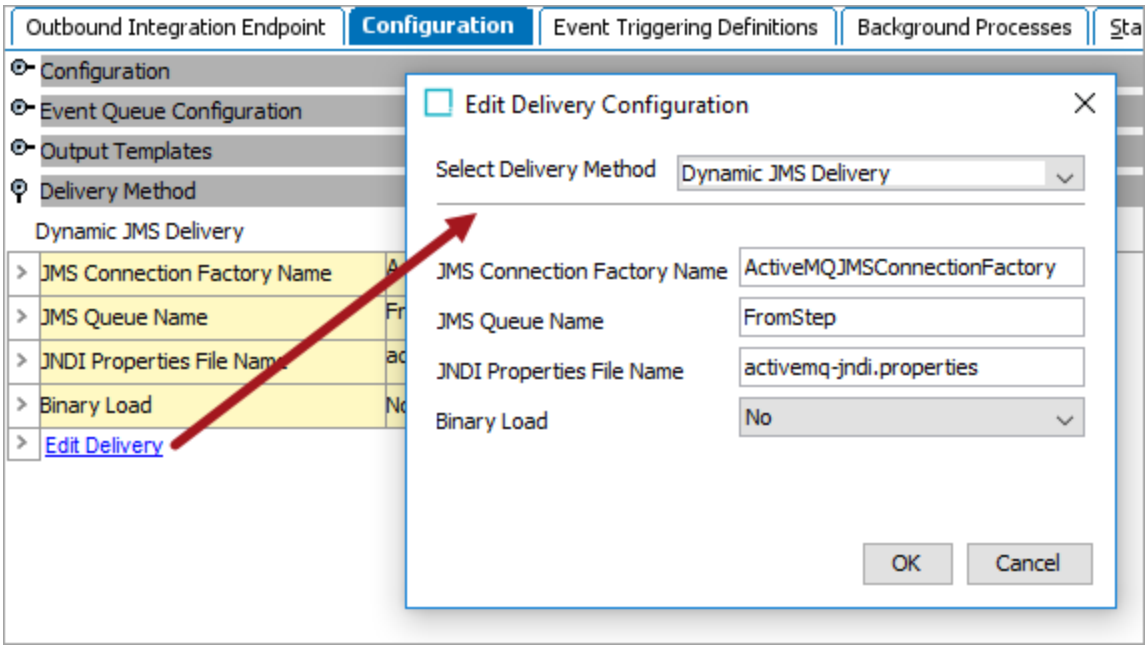
1. Verify the **jms-integration** add-on component is installed by reviewing the 'About STEP' option on the Start Page. Enter your credentials and click the 'Detailed version information' link. For on-premises systems, instructions for installing components can be found in the **SPOT Program** topic in the **System Administration Guide** found in **Downloadable Documentation**. For SaaS systems, contact Stibo Systems Support.
2. To enable the Dynamic JMS Delivery option, the case-sensitive configuration property 'JMS.ClassPath' must be set in sharedconfig.properties on the STEP application server and should point to an existing directory accessible from all application servers. Client libraries and JNDI files (Java Naming and Directory Interface) must be placed in this directory. Refer to the **Dynamic JMS Configuration Examples** section below for more information.

Once the server-side configuration is in place, the Dynamic JMS Delivery method can be configured via the workbench.

Configuration

For information on a parameter, hover over the parameter field to display help text.

1. On the **Configuration** tab, in the **Delivery Method** flipper, click **Edit Delivery**.



2. In **Select Delivery Method**, choose **Dynamic JMS Delivery**.
3. In **JMS Connection Factory Name**, specify a JMS connection factory name. The selection must match the entry configured in the JNDI file.
4. In **JMS Queue Name**, select the physical name of the JMS Queue to be used. The selection must match the entry configured in the JNDI file.

5. In **JNDI Properties File Name**, enter the name of the JNDI file.
6. In **Binary Load**, select 'Yes' if the message contents will be in a binary format (such as Excel).
7. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

Dynamic JMS Configuration Examples

As outlined in the following examples, in any JMS implementation, you must:

1. Put client jar files in the classpath folder.
2. Write the JNDI properties file.
3. Configure the connection factory and queue in the OIEP delivery method.

Azure Message Bus

Place these libraries in the directory pointed to by the `JMS.ClassPath` configuration property:

- geronimo-jms_1.1_spec-1.1.1.jar
- jakarta.jms-api-2.0.3.jar
- netty-buffer-4.1.82.Final.jar
- netty-codec-4.1.82.Final.jar
- netty-codec-http-4.1.82.Final.jar
- netty-common-4.1.82.Final.jar
- netty-handler-4.1.82.Final.jar
- netty-resolver-4.1.82.Final.jar
- netty-transport-4.1.82.Final.jar
- netty-transport-classes-epoll-4.1.82.Final.jar
- netty-transport-classes-kqueue-4.1.82.Final.jar
- netty-transport-native-epoll-4.1.82.Final-linux-x86_64.jar
- netty-transport-native-kqueue-4.1.82.Final-osx-x86_64.jar
- netty-transport-native-unix-common-4.1.82.Final.jar
- proton-j-0.34.0.jar
- qpid-jms-client-1.7.0.jar
- qpid-jms-discovery-1.7.0.jar

Important: Although the main JMS library is qpid-jms-client-[version].jar, the other libraries (listed above) are also required as these are underlying dependencies. Failing to include all of these in the JMS classpath will result in error messages similar to this one: `java.lang.NoClassDefFoundError: io/netty/util/concurrent/EventExecutorGroup`

Note: Due to a change in the Qpid JMS client's implementation, version 1.6.0+ of this library is not compatible with STEP's Dynamic JMS integration.

In the JNDI properties file, add properties to set the context factory and map queues, setting `connectionfactory = SBCF` as shown below:

```
java.naming.factory.initial = org.apache.qpid.jms.jndi.JmsInitialContextFactory
connectionfactory.SBCF = amqps://[servicebus_
name].servicebus.windows.net?jms.username=[username]&jms.password=[password]
```

Register queues in JNDI using the form:

- queue.[servicebus_queue_name] = [servicebus_queue_name]

Example azure-jndi.properties file:

```
java.naming.factory.initial = org.apache.qpid.jms.jndi.JmsInitialContextFactory
connectionfactory.SBCF =
amqps://pimtest.servicebus.windows.net?jms.username=RootManageSharedAccessKey&jms.password=4NzL79KOhmD8A9N8bA9QSTY3zxTHX9Hy1602xan0bqk=
queue.inbound-products = inbound-products
```

In the corresponding IIEP receiver, the JMS entries should have the following format:

- JMS Connection Factory Name SBCF
- JMS Queue Name [servicebus_queue_name]
- JNDI Properties File Name [jndi_properties_file_name]

Example receiver configuration:

- `JMS Connection Factory Name = SBCF`
- `JMS Queue Name = inbound-products`
- `JNDI Properties File Name = azure-jndi.properties`

For more information on the Qpid JMS client library, refer to https://github.com/MicrosoftDocs/azure-docs/blob/master/articles/service-bus-messaging/service-bus-java-how-to-use-jms-api-amqp.md

TibcoJMS

Libraries to be placed in the directory pointed to by the JMS.ClassPath configuration property:

- Tibjms.jar

The 'tibjms.jar' file can typically be found in:

```
components/shared/1.0.0/plugins/com.tibco.jms.jmsclient_[VersionNumber]
```

JNDI file content (example – file to be placed in the same directory as the libraries):

```
java.naming.provider.url=tibjmsnaming://[server1]:[port1], tibjmsnaming://[server2]:
[port2]
java.naming.factory.initial=com.tibco.tibjms.naming.TibjmsInitialContextFactory
java.naming.factory.url.pkgs=com.tibco.tibjms.naming
java.naming.security.principal=[username]
java.naming.security.credentials=[password]
```

Example delivery method configuration:

- `JMSConnectionFactoryName='QueueConnectionFactory'`
- `JMSQueueName='ToStep'`
- `JNDIPropertiesFileName='tibco-jndi.properties'`

With this configuration, use the connection factory ‘QueueConnectionFactory’ or the factory configured in the Kaazing Gateway 'factories.conf' file. Queue names can either be the name used in Tibco, or 'queue.[*add_the_tibcoqueuename*]' (which is needed when a topic exists on the JMS server with the same name as the queue you are trying to connect to).

Apache Active MQ 5.15.8

Libraries to be placed in the directory pointed to by the JMS.ClassPath configuration property:

- activemq-client-5.15.8.jar
- hawtbuf-1.11.jar

JNDI file content (example – file to be placed in the same directory as the libraries):

```
java.naming.factory.initial = org.apache.activemq.jndi.ActiveMQInitialContextFactory
java.naming.provider.url = tcp://127.0.0.1:61616
connectionFactoryNames = ActiveMQJMSConnectionFactory
queue.FromStep = FromStep
queue.ToStep = ToStep
java.naming.security.principal=admin
java.naming.security.credentials=admin
```

Example delivery method configuration:

- `JMS Connection Factory Name = 'ActiveMQJMSConnectionFactory'`
- `JMS Queue Name = 'ToStep'`
- `JNDI Properties File Name = 'activemq-jndi.properties'`

RabbitMQ

In RabbitMQ, create the necessary Queue(s) and Exchange(s) to allow message publishing and error reporting. For more information, search the web.

Libraries to be placed in the directory pointed to by the JMS.ClassPath configuration property. For each JAR file, use the latest updated version, indicated by the *{latest}* text in the following list:

- amqp-client-*{latest}*.jar
- fscontext-{latest}.jar
- rabbitmq-jms-{latest}.jar

Note: If credentials are to be provided, add them in the JNDI file as shown below since having the credentials in the .bindings file will not work.

JNDI file content (example – file to be placed in the same directory as the libraries):

```
java.naming.factory.initial = com.sun.jndi.fscontext.RefFSContextFactory
java.naming.provider.url = file:///opt/stibo/resources/jmsclasspath/

#Credentials
java.naming.security.principal = username
java.naming.security.credentials = password
```

'.bindings' file content must be in the directory identified by the `java.naming.provider.url` property above (example - file with this specific name to be placed in the same directory as the libraries):

```
# ConnectionFactory
ConnectionFactory/ClassName=javax.jms.ConnectionFactory
ConnectionFactory/FactoryName=com.rabbitmq.jms.admin.RMQObjectFactory
ConnectionFactory/RefAddr/0/Content=jms/ConnectionFactory
ConnectionFactory/RefAddr/0/Type=name
ConnectionFactory/RefAddr/0/Encoding=String
ConnectionFactory/RefAddr/1/Content=javax.jms.ConnectionFactory
ConnectionFactory/RefAddr/1/Type=type
ConnectionFactory/RefAddr/1/Encoding=String
ConnectionFactory/RefAddr/2/Content=com.rabbitmq.jms.admin.RMQObjectFactory
```

```
ConnectionFactory/RefAddr/2/Type=factory
ConnectionFactory/RefAddr/2/Encoding=String
ConnectionFactory/RefAddr/3/Content=superhost
ConnectionFactory/RefAddr/3/Type=host
ConnectionFactory/RefAddr/3/Encoding=String
ConnectionFactory/RefAddr/4/Content=STEP-frva
ConnectionFactory/RefAddr/4/Type=virtualHost
ConnectionFactory/RefAddr/4/Encoding=String
ConnectionFactory/RefAddr/5/Content=5672
ConnectionFactory/RefAddr/5/Type=port
ConnectionFactory/RefAddr/5/Encoding=String
ConnectionFactory/RefAddr/6/Content=false
ConnectionFactory/RefAddr/6/Type=ssl
ConnectionFactory/RefAddr/6/Encoding=String

# Product-Synchro Queue
Product-Synchro/ClassName=javax.jms.Queue
Product-Synchro/FactoryName=com.rabbitmq.jms.admin.RMQObjectFactory
Product-Synchro/RefAddr/0/Content=jms/Queue
Product-Synchro/RefAddr/0/Type=name
Product-Synchro/RefAddr/0/Encoding=String
Product-Synchro/RefAddr/1/Content=javax.jms.Queue
Product-Synchro/RefAddr/1/Type=type
Product-Synchro/RefAddr/1/Encoding=String
Product-Synchro/RefAddr/2/Content=com.rabbitmq.jms.admin.RMQObjectFactory
Product-Synchro/RefAddr/2/Type=factory
Product-Synchro/RefAddr/2/Encoding=String
Product-Synchro/RefAddr/3/Content=Product-Synchro
Product-Synchro/RefAddr/3/Type=destinationName
Product-Synchro/RefAddr/3/Encoding=String
Product-Synchro/RefAddr/4/Content=true
Product-Synchro/RefAddr/4/Type=amqp
Product-Synchro/RefAddr/4/Encoding=String
Product-Synchro/RefAddr/5/Content=amq.fanout
Product-Synchro/RefAddr/5/Type=amqpExchangeName
Product-Synchro/RefAddr/5/Encoding=String
Product-Synchro/RefAddr/6/Content=Product-Synchro
Product-Synchro/RefAddr/6/Type=amqpRoutingKey
Product-Synchro/RefAddr/6/Encoding=String
Product-Synchro/RefAddr/7/Content=Product-Synchro
Product-Synchro/RefAddr/7/Type=amqpQueueName
Product-Synchro/RefAddr/7/Encoding=String
```

Example delivery method configuration:

- `JMS Connection Factory Name = ConnectionFactory`
- `JMS Queue Name = rabbitmqQueue01`
- `JNDI Properties File Name = rabbitmq-jndi.properties`

WebLogic JMS Connection

Libraries to be placed in the Workarea directory, which points to by the jmslasspath configuration property:

- weblogic.jar
- wlclient.jar
- wlthint3client.jar

Important: Because STEP will attempt to use some classes that are in both the `wljmsclient.jar` and the `wlthint3client.jar` libraries, conflicts will generate. To prevent this conflict, use `wlthint3client.jar` library instead of the `wljmsclient.jar` library.

The weblogic-JNDI configuration includes:

```
java.naming.factory.initial=weblogic.jndi.WLInitialContextFactory
java.naming.provider.url=t3://localhost:7001
```

This includes the queue name of:

```
jms/TestQueue=jms/TestQueue
```

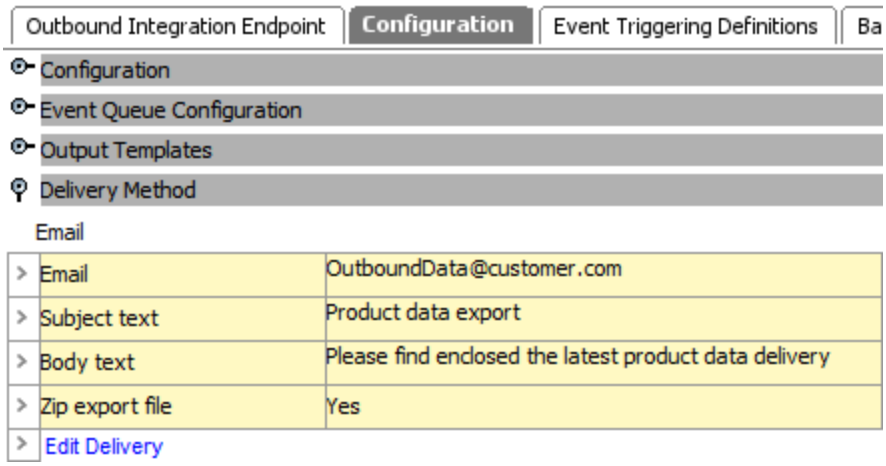
Example delivery method configuration:

```
JMS Connection Factory Name = weblogic.jms.ConnectionFactory
JMS Queue Name = jms/TestQueue
JNDI Properties File Name = weblogic-jndi.properties
```

Email Delivery Method

Similar to the 'Copy to directory' delivery method, the email delivery option allows the delivery to be zipped. This method is typically used to test the export output, or to deliver a small export to one or more persons.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



This delivery method is also available in Export Manager as defined in the **Email Delivery Method** topic.

Prerequisites

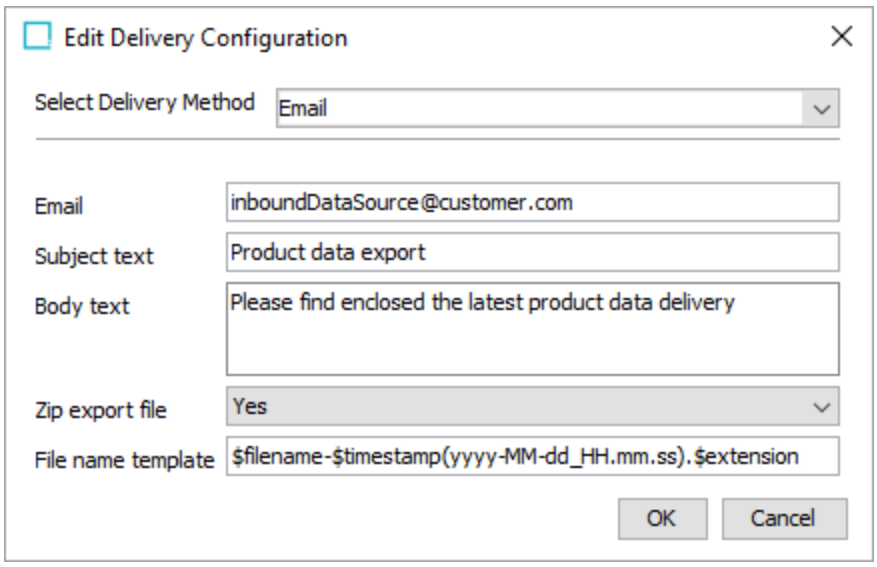
To deliver an exported file via email, the STEP server must have access to the SMTP server. Be aware of email file-size limits because, depending on the export's contents, the file could be very large.

For information on configuring email from STEP, refer to the **Email from STEP** topic in the **Resource Materials** of online help.

Configuration

For information on a parameter, hover over the parameter field to display help text.

1. Click the **Select Delivery Method** parameter to display the dropdown and choose **Email**.
2. For the **Email** parameter, enter an email address. For multiple recipients, enter the email addresses separated by a semicolon.
3. For the **Subject text** parameter, enter the text that will display for the email subject line. The server name is prepended to the subject text on the email.
4. For the **Body text** parameter, enter the text that will display for the email body.



5. In **Zip export file**, select 'yes' or 'no' from the dropdown to specify if the output file should be delivered as an email attachment in a .ZIP (compressed) file format.

   Note: For the Email delivery method, the files named in the ZIP file do not respect the $filename variable.

   - **Yes** uses 'export-' before the timestamp variable, and then the extension ZIP. For example, a zipped STEPXML output with the default File Name Template exported on 15 NOV 2016 results in an output .ZIP file named 'export-1479230247017.zip.' The contents of the ZIP file follow the $filename variable applied in the 'File name template' parameter.
   - **No** uses the file type for the file name along with the appropriate extension for the selected data format.

6. In **File name template**, to create the file name used for the output, enter template variables (identified with a $), static text, or use a combination of both. The default template is $filename-$timestamp(yyyy-MM-dd_HH.mm.ss).$extension. Depending on the settings of the configuration, the default template outputs a file name similar to 'GenericXML-2020-07-30_14.07.44.xml.' Each variable is described below:

   Note: The 'Zip export file' parameter also has an impact on the file name.

   - **$filename:** This variable is replaced with text to indicate the format of the delivered file as specified in the Output Templates flipper. For example, the output file name would include the text 'excel' or 'GenericXML' for those formats, or the Event ID for event-based STEPXML. Using the default file name template, a comma-separated value file would be named 'csv--2020-07-30_14.07.44.csv' and 'GenericXML--2020-07-30_14.07.44.xml' would indicate that Generic XML was used.
   - **$timestamp**: This variable is replaced with the time in year-day-month-hour-minutes-seconds-milliseconds, excluding the dashes. For example, 20161511094709625 is used for an export on 15 NOV 2016 at 9:47:09:625 AM.

Adding a definition for the arrangement of the date / time elements provides a way to generate a more readable date. Static characters can be included to separate the timestamp elements and individual elements can be ordered as required, as shown below.

Note: To revert the **$timestamp** variable so that it uses the legacy definition (milliseconds between January 1, 1970, and the time when the file is uploaded), contact Stibo Systems Support for more information.

$timestamp(ddMMyyyy): The timestamp is modified to display day-month-year (excluding dashes) and is output as '15112016' for 15 NOV 2016.

$timestamp(yyyy-MM-dd_HH.mm.ss): The timestamp is modified to display year-month-day_ hour.minutes.seconds (including the dashes, underscore, and periods) and is output as '2016-11-15_ 09.32.43' for 2016 NOV 15 at 9:32.43 AM.

- **$extension:** This variable is replaced with the extension of the output file based on the selected format in the Output Templates flipper. For XML-based formats, the output file is set with the .XML extension, while Excel files use .XLS, and comma-separated values use .CSV.

Note: The File Name Template does not support conversions of file formats and can only be used to deliver files in the format specified in the configuration.

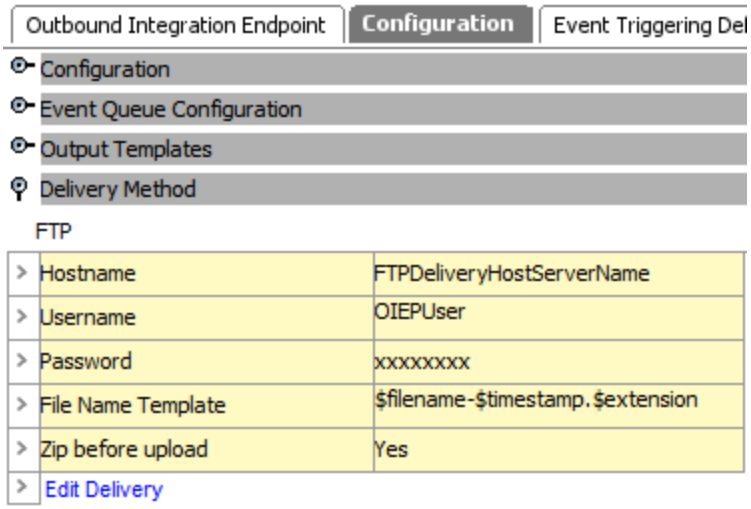
7. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

FTP Delivery Method

The FTP delivery method allows an exported file to be delivered to an external system and is often used when the output files are large, or when a different or remote system is in use. The preferred method for FTP delivery is the SFTP secure delivery method.

For information on the OIEP SFTP delivery method, refer to the **SFTP Delivery Method** topic.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



To use this delivery method with Export Manager, refer to the **FTP Delivery Method** topic.

Prerequisites

Changes to the properties file, outlined below, are implemented when the server is restarted.

Prior to configuration, clicking the **HostName** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **FTPDeliveryHostName** property. The required format of the property is (square brackets not included):

```
SFTPDeliveryHostName=1=[host1],2=[host2]
```

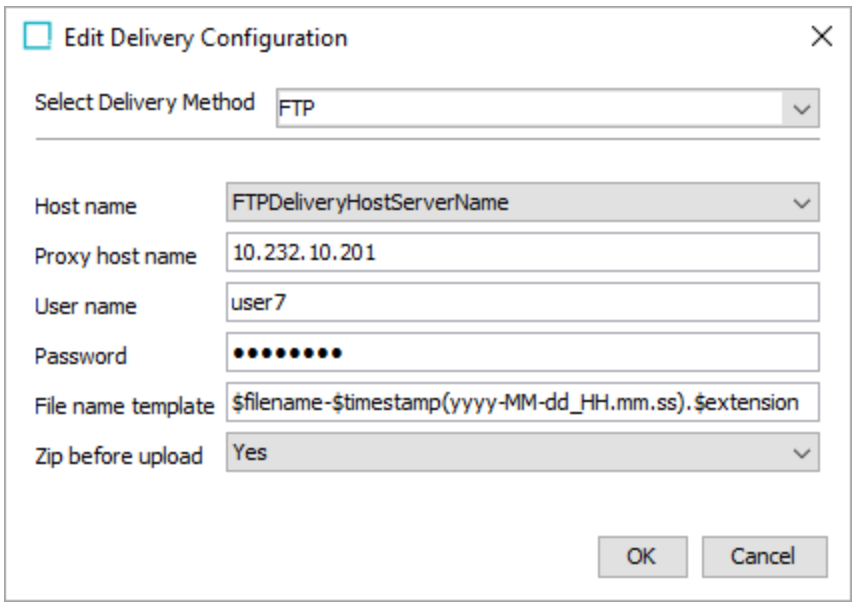
Using the host name shown in the image below, the entry in the properties file would be:

```
FTPDeliveryHostName=1=FTPDeliveryHostServerName
```

Configuration

For information on a parameter, hover over the parameter field to display help text.

1. In **Select Delivery Method**, choose **FTP**.



2. In **Host name**, select the host name or IP address of the FTP server to be used for the delivery.
3. In **Proxy host name**, add the host name to be used for the server proxy. This field is optional.
4. In **User name**, enter the user name that has access to log on to the FTP server.
5. In **Password**, enter the password that will be used to log on to the FTP server.
6. In **File name template**, to create the file name used for the output, enter template variables (identified with a $), static text, or use a combination of both. The default template is $filename-$timestamp(yyyy-MM-dd_HH.mm.ss).$extension. Depending on the settings of the configuration, the default template outputs a file name similar to 'GenericXML-2020-07-30_14.07.44.xml.' Each variable is described below:
   - **$filename** For event-based OIEPs, this variable is replaced with text to indicate the format of the delivered file as specified in the Output Templates flipper, except for STEPXML when the first and last Event IDs are used. For example, the output file name could be 'csv-2020-07-30_14.09.40.csv' or '1804038-1804038.xml' to indicate that STEPXML was used for a single event.

     Note: The 'Zip before upload' parameter also has an impact on the file name.

   - **$timestamp**: This variable is replaced with the time in year-day-month-hour-minutes-seconds-milliseconds, excluding the dashes. For example, 20161511094709625 is used for an export on 15 NOV 2016 at 9:47:09:625 AM.

     Adding a definition for the arrangement of the date / time elements provides a way to generate a more readable date. Static characters can be included to separate the timestamp elements and individual elements can be ordered as required, as shown below.

     Note: To revert the **$timestamp** variable so that it uses the legacy definition (milliseconds between January 1, 1970, and the time when the file is uploaded), contact Stibo Systems Support for more information.

$timestamp(ddMMyyyy): The timestamp is modified to display day-month-year (excluding dashes) and is output as '15112016' for 15 NOV 2016.

$timestamp(yyyy-MM-dd_HH.mm.ss): The timestamp is modified to display year-month-day_ hour.minutes.seconds (including the dashes, underscore, and periods) and is output as '2016-11-15_ 09.32.43' for 2016 NOV 15 at 9:32.43 AM.

- **$extension** This variable is replaced with the extension of the output file based on the selected format in the Output Templates flipper. For XML-based formats, the output file is set with the .XML extension, while Excel files use .XLS, and comma-separated values use .CSV.

Note: The File Name Template does not support conversions of file formats and can only be used to deliver files in the format specified on the Configuration tab > Output Templates flipper of the outbound integration endpoint.

7. In **Zip before upload**, select an option from the dropdown to specify if the output file should be delivered in a .ZIP (compressed) file format.
   - **Yes** uses 'result_0' before the timestamp variable, and then the extension ZIP. The File Name Template is used for the contents of the ZIP file. For example, a zipped STEPXML output with the default File Name Template exported on 15 NOV 2016 results in an output .ZIP file named 'result_0-2020-07-30_ 14.07.44.zip.' The contents of the ZIP file follow the $filename variable applied in the 'File name template' parameter.
   - **No** uses the File Name Template for the file name along with the appropriate extension for the selected data format.
8. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

Git Delivery Method

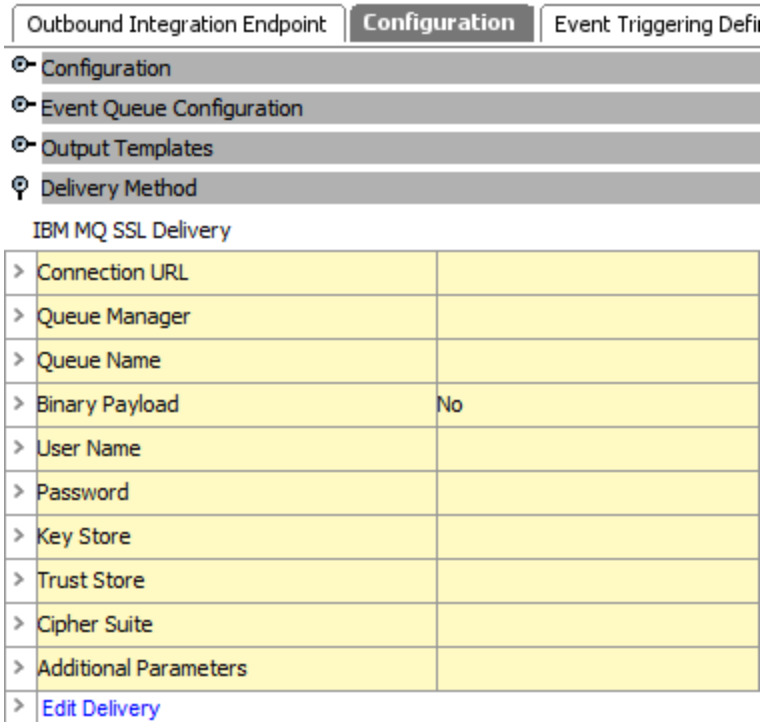
The Git Delivery method delivers files produced by the outbound integration endpoint (OIEP) processing engine or a configured post-processor to a branch in a remote Git repository. Refer to https://git-scm.com for more information about Git.

For details, refer to the **Integration Endpoint Options for VCS Integration** topic in the **Configuration Management** documentation.

IBM MQ SSL Delivery Method

This delivery method allows connection with IBM MQ. Prior to release 2023.3, it was named 'IBM Websphere MQ SSL Delivery'. For information on connecting to IBM MQ in a non-SSL way, refer to **JMS Delivery Method**. IBM MQ SQL delivery is only available in an OIEP.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



Prerequisites

Changes to the properties file, outlined below, are implemented when the server is restarted.

Prior to configuration, clicking a dropdown parameter in the 'Edit Delivery Configuration' dialog displays the property name required to supply values that populate the parameter.

Note: In the sharedconfig.properties file, a numbered designation of an integer (**1=**, **2=**, etc.) in the value part of the property indicates that an entry should display in the UI. This allows multiple values to be stored for a single property and is required even when only a single value is required. If property values, such as passwords, should not be displayed in the UI, exclude the numbered designation, as shown in the password examples below.

The numbered designation indicates the order that the options are displayed in the dropdown. For example: `<Property name>=1=<Value 1>,2=<Value 2>,3=<Value 3>`. Using these numbered designations results in the dropdown showing values in the following order: `<Value 1>,<Value 2>,<Value 3>`. When duplicate integers exist, only the last value is displayed in the dialog.

As required, configure the following case-sensitive properties in the sharedconfig.properties on the STEP application server:

1. For **Connection URL**, set the **WSMQSSLProviderURL** property, as shown below.

   Use the [host]:[port]/[channel] format as defined:

   - [host] = hostname or IP of the MQ server; in the follow example this is 10.46.88.75
   - [port] = port number for the channel; in the follow example this is 1415
   - [channel] = name of the channel; in the follow example this is BASE.CTL.SVRCONN

   ```
   WSMQSSLProviderURL=1=10.46.88.75:1415/BASE.CTL.SVRCONN
   ```

2. For **Queue Manager**, set the **WSMQSSLQueueManager** property, as shown below.

   ```
   WSMQSSLQueueManager=1=HV088B
   ```

3. For **Queue Name**, set the **WSMQSSLQueue** property, as shown below.

   ```
   WSMQSSLQueue=1=LIVE.KITT
   ```

4. For **Key Store**, set the **WSMQSSLKeyStoreLocation** property, as shown below.

   Use the jks format with the personal certificate for the Queue Manager. To generate this, refer to the IBM MQ online help. To indicate a URL and not a path, prefix the Key Store with 'file:'.

   ```
   WSMQSSLKeyStoreLocation=1=file:/workarea/keystore.jks
   ```

5. For **Key Store Password**, set the **WSMQSSLKeyStorePassword** property, as shown below.

   The password can be configured in sharedconfig.properties file or can be typed directly into the parameter. When the parameter is blank, the password from the property is used.

   To prevent the password (or any other selections) from displaying in the UI, do not include a numbered designation.

   ```
   WSMQSSLKeyStorePassword=Pa55w0rd1
   ```

6. For **Trust Store**, set the **WSMQSSLTrustStoreLocation** property, as shown below.

   This can be the same file as key store. To generate this, refer to the IBM MQ online help. As shown below, prefix the Trust Store with 'file:' to indicate a URL, not a path.

   ```
   WSMQSSLTrustStoreLocation=1=file:/workarea/truststore.jks
   ```

7. For **Trust Store Password**, set the **WSMQSSLTrustStorePassword** property, as shown below.

   The password can be configured in sharedconfig.properties file or can be typed directly into the parameter. When the parameter is blank, the password from the property is used.

To prevent the password (or any other selections) from displaying in the UI, do not include a numbered designation.

```
WSMQSSLTrustStorePassword=Pa55w0rd2
```

8. For **Cipher Suite**, set the **WSMQSSLCipherSuite** property, as shown below.

   STEP is running on non-IBM JRE, so this must be the same value as configured in the Queue Manager.

   ```
   WSMQSSLCipherSuite=1=CTL_RSA_WITH_AES_256
   ```

9. Contact your IT team to create a trust and key store and copy all URL certificates to the trust and key store.

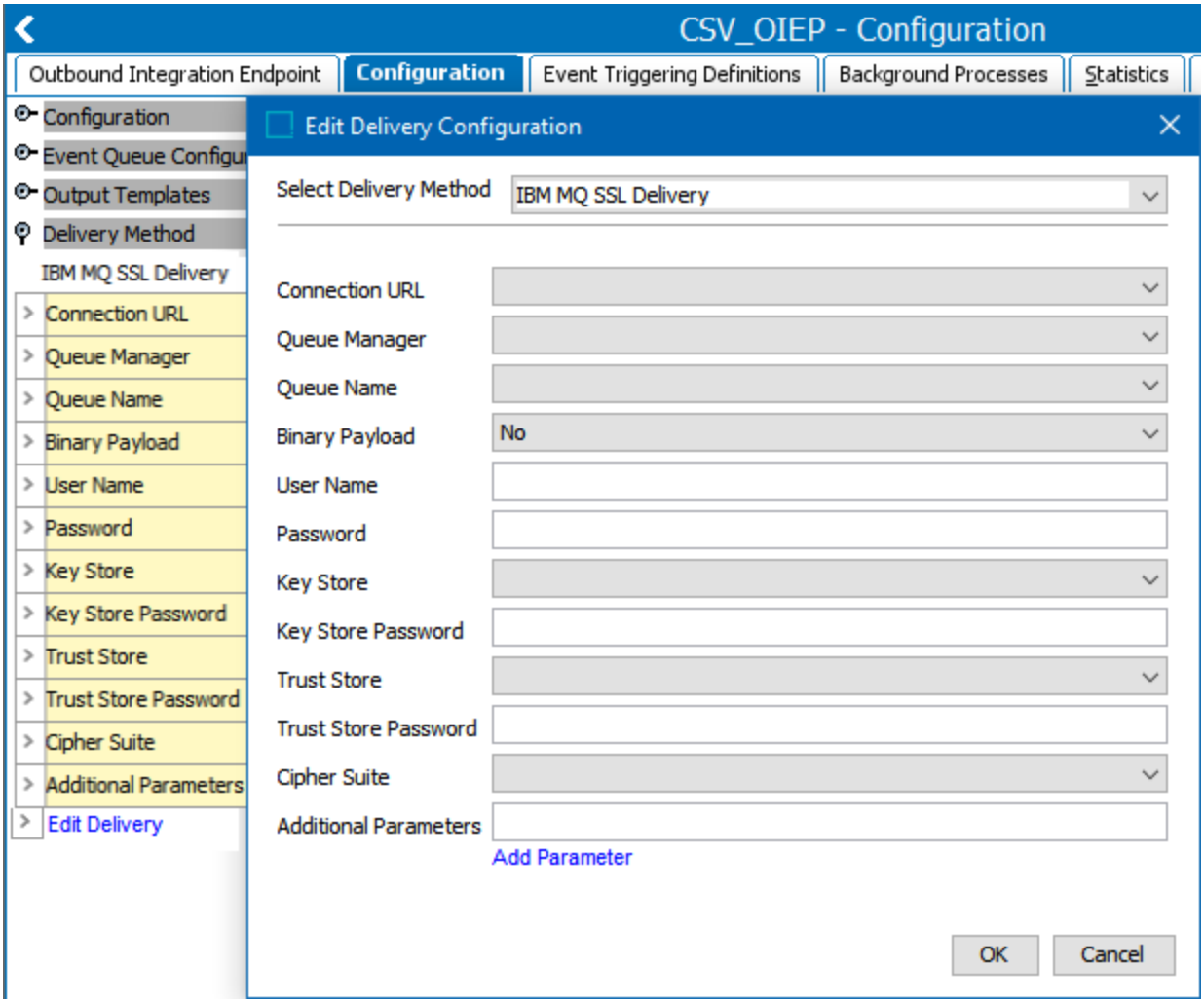
Example of all property entries

```
WSMQSSLProviderURL=1=10.46.88.75:1415/BASE.CTL.SVRCONN
WSMQSSLQueueManager=1=HV088B
WSMQSSLQueue=1=LIVE.KITT
WSMQSSLKeyStoreLocation=1=/workarea/key.jks
WSMQSSLKeyStorePassword=Pa55w0rd1
WSMQSSLTrustStoreLocation=1=/workarea/key.jks
WSMQSSLTrustStorePassword=Pa55w0rd2
WSMQSSLCipherSuite=1=CTL_RSA_WITH_AES_256
```

Configuration

For information on a parameter, hover over the parameter field to display help text.

1. In the **Select Delivery Method** field, choose **IBM MQ SSL Delivery**.



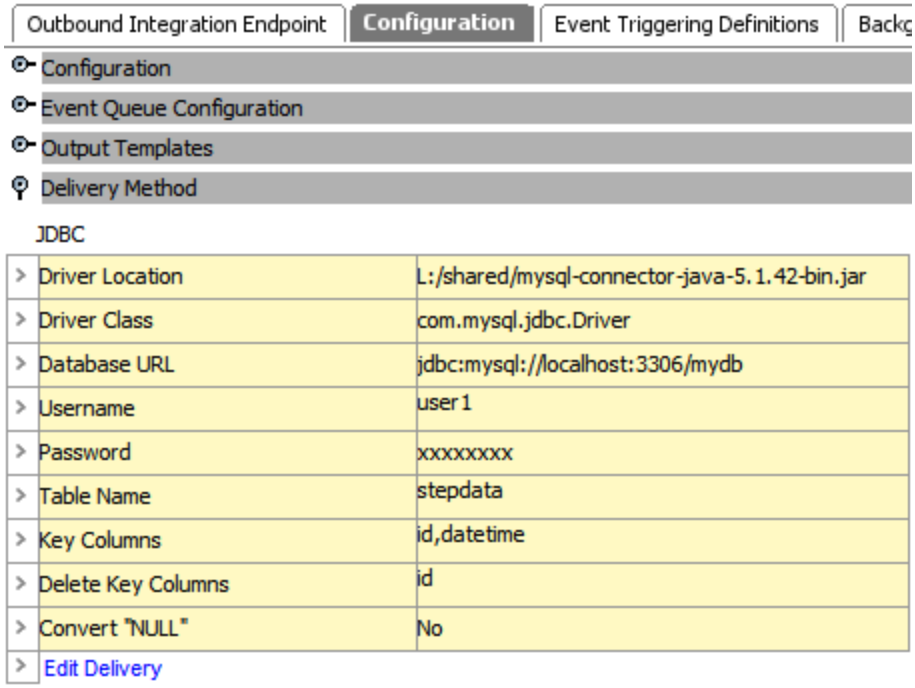
2. In **Connection URL**, select the URL for the connection.
3. In **Queue Manager**, select the name of the Queue Manager.
4. In **Queue Name**, select the name of the Queue for the connection.
5. In **Binary Payload**, select 'Yes' if the message contents will be in a binary format (such as Excel).
6. In **User Name**, if required, enter the user name to be used with this integration.
7. In **Password**, if required, enter the password to be used with this integration.
8. In **Key Store**, select the keystore in jks format, with the personal certificate for the Queue Manager.
9. In **Key Store Password**, leave the parameter blank to use the password from the sharedconfig.properties file, or manually type in a password.
10. In **Trust Store**, select the trust store with the CA for the Queue Manager.

11. In **Trust Store Password**, leave the parameter blank to use the password from the sharedconfig.properties file, or manually type in a password.
12. In **Cipher Suite**, set to the same value as SSL CipherSuite in IBM MQ.
13. If **Additional Parameters** are required, click the **Add parameter** link and enter the Key and the Value. For valid keys and values, refer to the IBM MQ online help.
14. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

JDBC Delivery Method

The JDBC delivery method allows STEP data to be delivered directly to tables in Relational database management systems (RDBMS) like Oracle, MySQL, MS SQL Server, and PostgreSQL. Though the ways the JDBC plugin can be deployed are various, one of its prime uses is to send STEP data to data analytics tools for the purpose of displaying STEP data in a data analytics dashboard. This delivery method requires CSV format and is available in both the Export Manager and in outbound integration endpoints (OIEPs).

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



For more information on the Export Manager option, refer to **JDBC Delivery Method** in the **Export Manager** documentation.

Prerequisites

Important: For complete setup requirements, refer to the **Exporting Data via JDBC with CSV Format** topic in the **Data Format** documentation.

Install the required drivers

JDBC specification 4.1-compliant drivers should be placed in a directory accessible from all application servers. These drivers can then be made available for the delivery plugin via the dynamic properties JDBCDeliveryPlugin.DriverPath.[n] and JDBCDeliveryPlugin.DriverClass.[n]. For more information regarding applicable Java drivers, review the RDBMS vendor's homepage on the web.

Configure data for the dropdown parameters

Changes to the properties file, outlined below, are implemented when the server is restarted.

1. Prior to configuration, clicking the **Driver Location** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **JDBCDeliveryPlugin.DriverPath.[n]** property. As an example:

   ```
   JDBCDeliveryPlugin.DriverPath.1 = L:/shared/mysql-connector-java-5.1.42-bin.jar
   ```

   In this example, the drivers are stored on the application server's L:/shared drive.

2. Prior to configuration, clicking the **Driver Class** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **JDBCDeliveryPlugin.DriverClass.[n]** property. As an example:

   ```
   JDBCDeliveryPlugin.DriverClass.1 = com.mysql.jdbc.Driver
   ```

   In this example, the drivers class used is 'com.mysql.jdbc.Driver.'

3. Prior to configuration, clicking the **Database URL** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **JDBCDeliveryPlugin.URL.[n]** property.

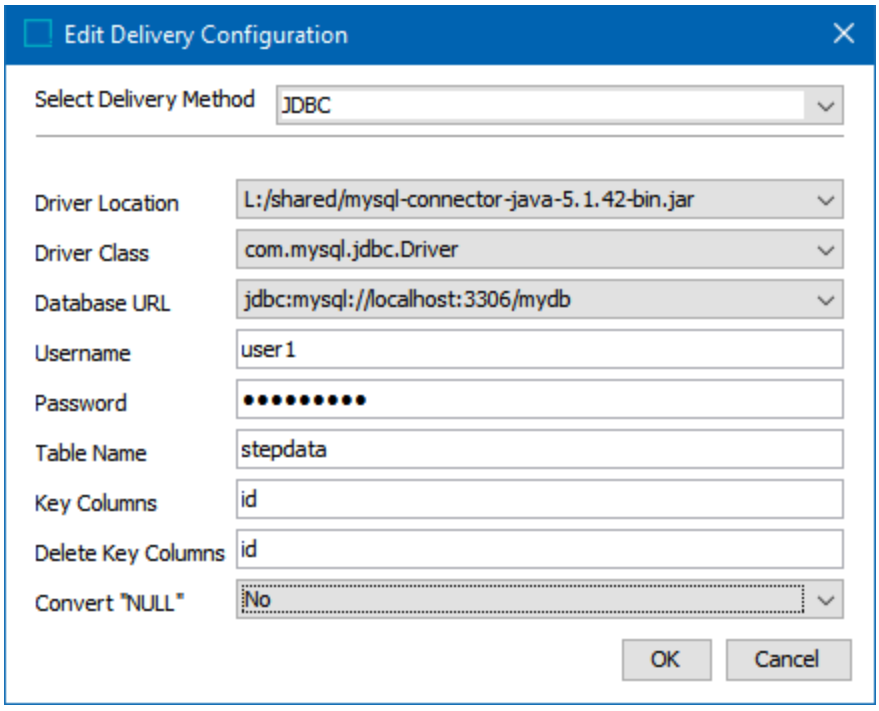
   ```
   JDBCDeliveryPlugin.URL.1 = jdbc:mysql://localhost:3306/mydb
   ```

   In this example, the database URL used is 'jdbc:mysql://localhost:3306/mydb.'

Configuration

For information on a parameter, hover over the parameter field to display help text.

In the OIEP editor on the **Configuration** tab, navigate to the **Delivery Method** flipper, then click **Edit Delivery**.



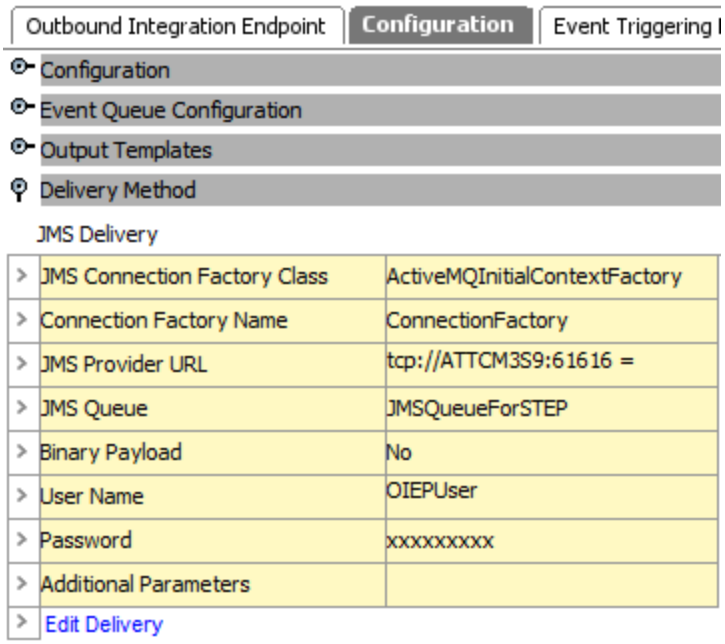
1. In **Select Delivery Method**, choose **JDBC** from the dropdown.
2. In **Driver Location**, select from the dropdown one of the paths to the relevant .jar file.
3. In **Driver Class**, select from the dropdown one of the pre-configured driver class.
4. In **Database URL**, select from the dropdown one of the pre-configured URLs to the destination database.
5. In **Username**, enter the username required to access the destination database.
6. In **Password**, enter the password required to access the destination database.
7. In **Table Name**, enter the name of the table in the destination database to which STEP will be publishing data
8. In **Key Columns**, list the names of the columns appearing on both the exported CSV file and the table in the destination database, separated by commas (and no spaces), into which STEP will publish data
9. In **Delete Key Columns**, list the names of the columns appearing on both the exported CSV file and the table in the destination database, separated by commas (and no spaces), from which STEP will delete data. The headers contained in this field can differ from the headers in the 'Key Columns' field, but they must also be part of the upsert key definition.
10. In **Convert "NULL"**, choose Yes if the string “NULL” should be converted to the value null. This may, for instance, be used for clearing a value in a column in the target database. This parameter defaults to No.
11. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

JMS Delivery Method

The available options for the Java Message Service (JMS) delivery method are system dependent. STEP has the following standard JMS delivery options: Apache Active MQ, WebSphere MQ, and Oracle AQ. Each of these can be used to deliver messages to a JMS Receiver without any further customizations. To deliver messages to other JMS providers, contact your Stibo Systems account manager for further customizations. This delivery option is only available in OIEPs.

Important: This standard functionality only supports queues. Support for topics requires custom development via the **Extension API** (available from the Start Page using the **STEP API Documentation** link). Alternatively, topics can be supported using middleware to move the message from a queue to a topic.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



Prerequisites

Changes to the properties file, outlined below, are implemented when the server is restarted.

Refer to the **Sharedconfig.properties File Examples** section below for more information.

1. Prior to configuration, clicking the **Connection Factory Name** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive `JMSConnectionFactoryName` property.

2. Prior to configuration, clicking the **JMS Provider URL** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive `JMSProviderURL` property. If necessary, use a comma to separate multiple URLs.

   The following is an example of a complete property entry for two systems named 'qa' and 'stage,' as well as their URLs 'http://step-qa' and 'http://step-stage':

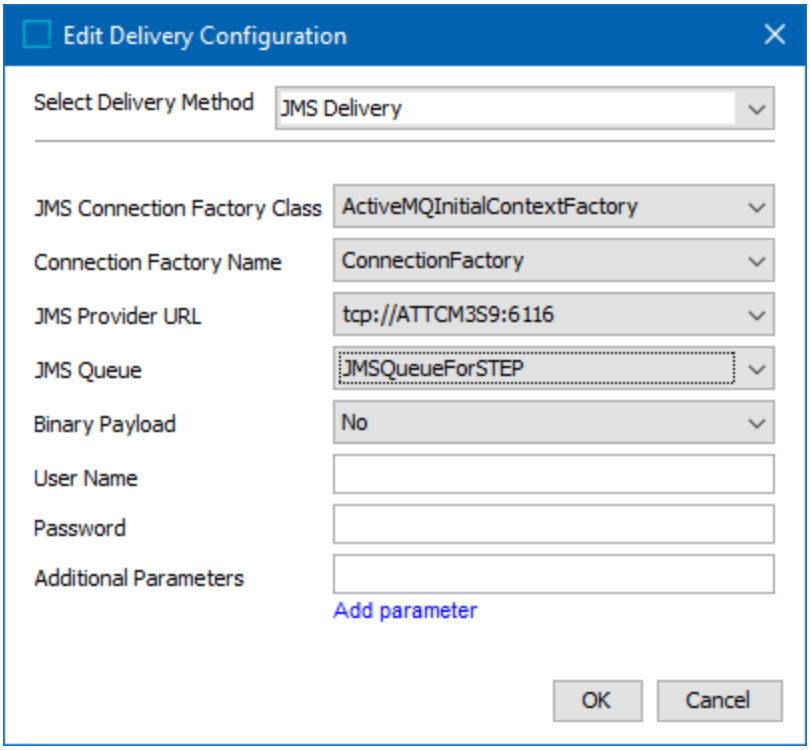
   ```
   JMSProviderURL=qa=http://step-qa,stage=http://step-stage
   ```

3. Prior to configuration, clicking the **JMS Queue** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive `JMSQueue` property.

Configuration

For information on a parameter, hover over the parameter field to display help text.

1. On the **Configuration** tab, in the **Delivery Method** area, click **Edit Delivery**.
2. In **Select Delivery Method**, choose **JMS Delivery**.



3. In **JMS Connection Factory Class**, choose the JMS connection factory class that corresponds to the driver class from the JMS provider vendor. On a standard system, the following options are available:

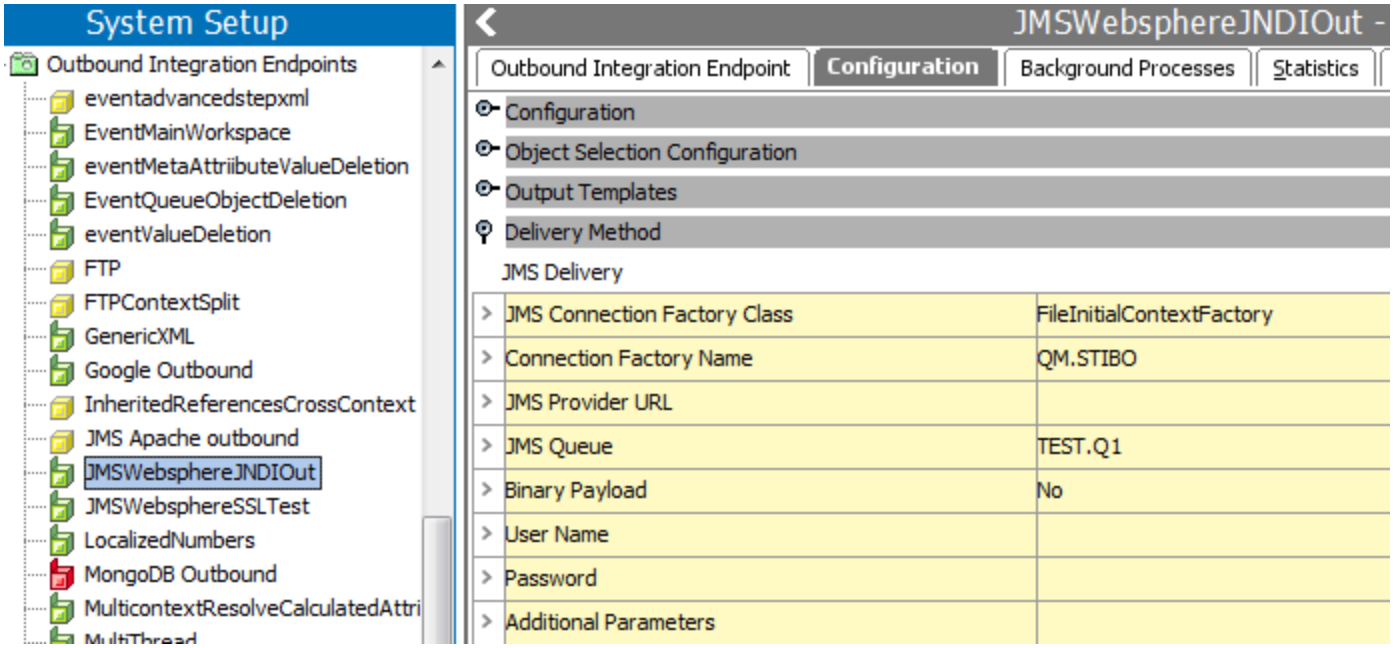
- **ActiveMQInitialConnectionFactory**: Connects the JMS delivery to Apache Active MQ. For information, search the web.

  Note: WMQInitialContextFactory is not supported, use FileInitialContextFactory instead.

- **FileInitialContextFactory**: Enables setting up JMS WebSphere inbound and outbound integration endpoints which reference a binding file (created from JMS WebSphere Client software). The binding file is a configuration file which includes all details of how STEP should interact with JMS WebSphere. For information about JMS WebSphere Client Software, search the web.

4. In **Connection Factory Name**, select a connection factory name from the list.
5. In **JMS Provider URL**, select a JMS Provider URL from the list.
6. In **JMS Queue**, select the physical name of the JMS Queue to be used on Apache Active MQ or WebSphere MQ.
7. In **Binary Payload**, select 'Yes' if the message contents will be in a binary format (such as Excel).
8. In **User Name**, if required, enter the user name that will be used to log onto the JMS provider.
9. In **Password**, if required, enter the password that will be used to log onto the JMS provider.
10. If additional parameters are required, click **Add parameter**, then enter the Key and Value.
11. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

FileInitialContextFactory JMS Delivery Configuration Example



JMS WebSphere Delivery Using SSL Configuration Example

Delivery Method	
IBM Websphere MQ SSL Delivery	
Connection URL	webspheremq-qa.stibo.com:1417/STEP.SVRCONN
Queue Manager	QM.STIBO_SSL
Queue Name	TEST.Q1
Binary Payload	Yes
User Name	
Password	
Key Store	file:/workarea/JMSWebsphereSSLKeystore/keyStore.jks
Trust Store	file:/workarea/JMSWebsphereSSLKeystore/keyStore.jks
Cipher Suite	TLS_RSA_WITH_AES_256_CBC_SHA256
Additional Parameters	

Apache Active MQ JMS Delivery

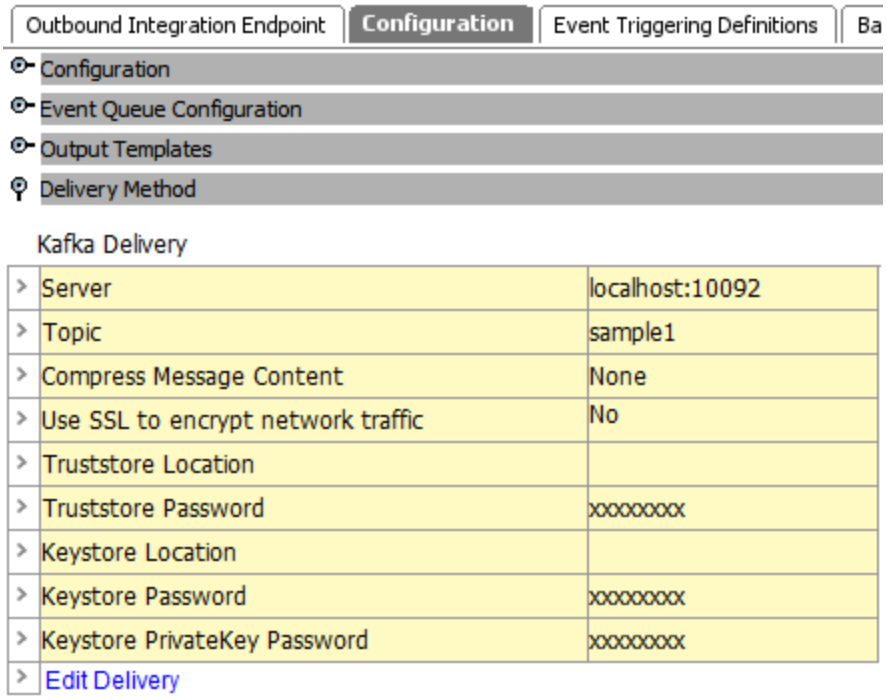
Delivery Method	
JMS Delivery	
JMS Connection Factory Class	ActiveMQInitialContextFactory
Connection Factory Name	ConnectionFactory
JMS Provider URL	tcp://ATTCM3S9:61616
JMS Queue	testqueue
Binary Payload	No
User Name	
Password	
Additional Parameters	
Edit Delivery	

Kafka Delivery Method

The Kafka Delivery method enables a STEP platform integrated with Apache Kafka, which is an open-source distributed event-streaming data platform, to use a Kafka queue. For more information about Apache Kafka, search the web.

This delivery method is only available in OIEPs.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



Prerequisites

1. Before setting up a Kafka Delivery, read the **Considerations for Setting up Kafka Receiver or Delivery** topic in this documentation.
2. Prior to configuration, clicking the **Server** dropdown parameter displays the required server name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive ***Kafka.Server*** configuration property. If connecting to a cluster, use a comma-separated list.

   The following example shows two Kafka server configurations where the first server is a cluster:

   ```
   Kafka.Server.1=mybroker1a:9094,mybroker1b:9094
   Kafka.Server.2=mybroker2a:9094
   ```

3. Prior to configuration, clicking the **Topic** dropdown parameter displays the available topics. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive ***Kafka.Topic*** configuration property. For example:

```
Kafka.Topic.1 = sample4
```

4. Prior to configuration, clicking the **KeystoreLocation** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **Kafka.SSLKeyStoreLocation** property. For example:

```
Kafka.SSLKeyStoreLocation.1=[/[path]/key_store.jks].
```

5. Prior to configuration, clicking the **Truststore Location** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **Kafka.SSLTrustStoreLocation** property. For example:

```
Kafka.SSLTrustStoreLocation.1=[/[path]/trust_store.jks].
```

6. SASL / OAuth 2.0 can be configured for STEP with Kafka using the **ExtraDriverOptions** property to authenticate securely via bearer tokens. A business function allows integration with the Kafka Delivery method by handling the OAuth authentication and returning a HashMap containing the Bearer Token and other details.

   Configure the following case-sensitive sharedconfig.properties entries within the value of the ExtraDriverOptions as shown in the example below, which can be copied:

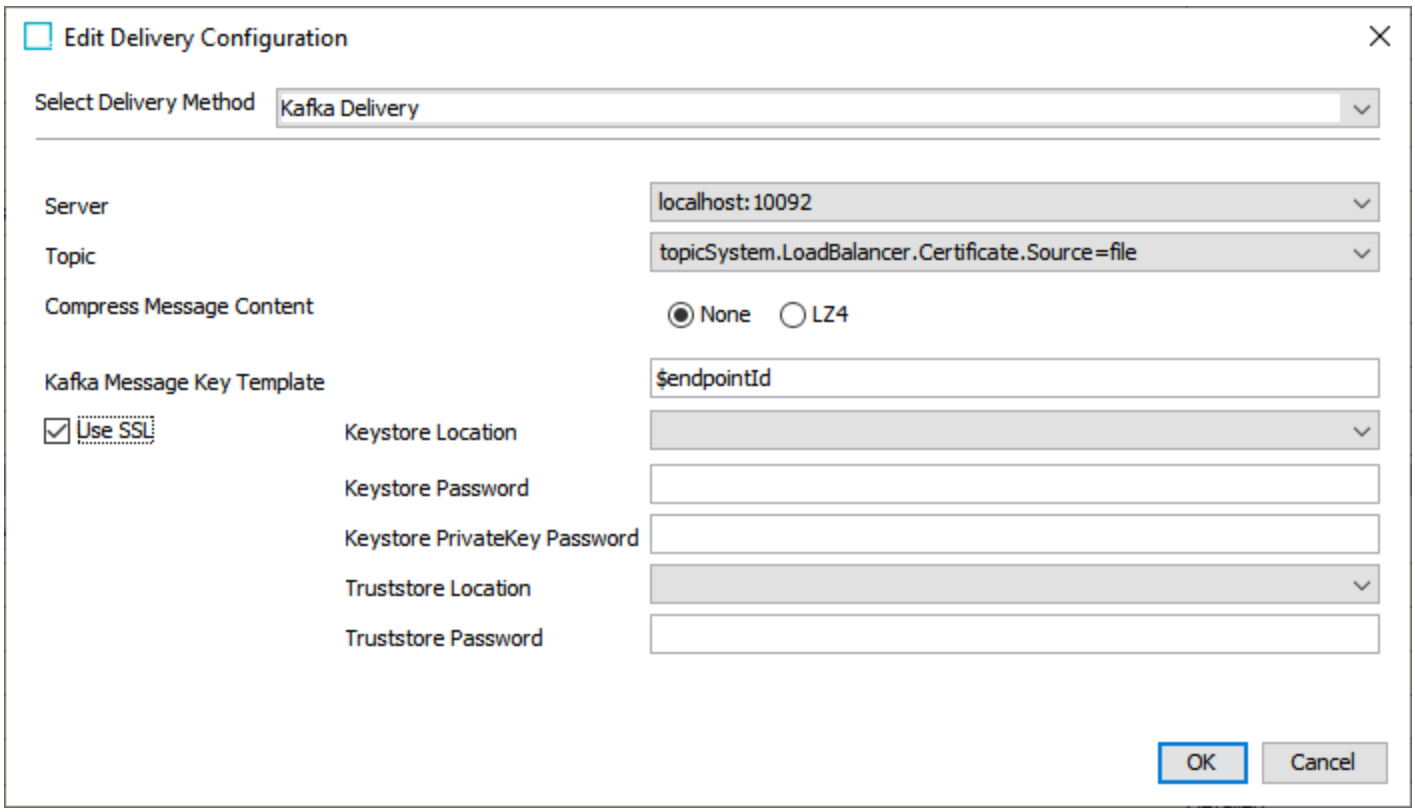
   - `Kafka.Delivery.[OIEP_ID].ExtraDriverOptions` - indicate the OIEP ID.
   - `stibo.authentication.function` - set the JavaScript business function ID. Refer to the Business Function Example for Generating an OAuth Bearer Token section below.
   - `security.protocol` - indicate either SASL_PLAINTEXT (while testing) or SASL_SSL (in production) for the SSL transport layer.

   For example, in the online help version of this topic the property definition includes, 'MyKafka' is the ID of the OIEP, 'HydraAuthFunction' is the ID of the business function, and 'SASL_PLAINTEXT' is the testing value for the security protocol and the rest of the property values are configured as displayed:

Configuration

The Kafka Delivery option is available when configuring an outbound integration endpoint (OIEP).

On the OIEP select 'Edit Delivery,' and then provide the following information:



1. For **Delivery**, choose **Kafka Delivery**.
2. For the **Server**, select the server(s) where the Kafka broker instances used by the endpoint are running.
3. In **Topic**, select the topic used by this endpoint.
4. For **Max Events To Read Per Poll**, enter the maximum number of events to read out of Kafka per invocation.

   Note: Setting this too high can require allocating additional memory in the system.

5. In **Decompress Message Content**, optionally select a decompression option:
   - None - message content will remain compressed.
   - LZ4 - lossless data compression algorithm. Search the web for more information.
6. In **Kafka Message Key Template**, enter the desired key template. Viable templates include: ${endpointId}, ${nodeType}, and ${nodeId}.
7. In **Keystore Location**, if SSL encryption is required, select an SSL encrypted connection to Kafka. Otherwise, leave this parameter blank.
8. In **Keystore Password**, enter the password for keystore if required.
9. In **Keystore PrivateKey Password**, enter the password of the private key in the keystore file, if needed.

10. In **Truststore Location**, if SSL encryption is required, select an SSL encrypted connection to the Kafka. Otherwise, leave this parameter blank.
11. In **Truststore Password**, enter the password for truststore if required.
12. Click the **Next** button to continue with the **IIEP - Configure Endpoint** step, or the **Finish** button to close the wizard.

Business Function Example for Generating an OAuth Bearer Token

In the online help version of this topic the code example includesa framework for developing a business function to be used with Kafka Delivery and Receiver properties when SASL / OAuth 2.0 Bearer Token authentication is configured.

Considerations for Setting Up Kafka Receiver or Delivery

There are a number of considerations before starting the setup for Kafka integration.

- Message size limitation
- Event ordering
- Event message serialization
- Reference handling for dangling references
- SASL authentication, if needed

Each area is explained in further detail below.

When configuring Kafka, STEPXML is recommended for the messaging format. Other formats supported for integration with STEP will also work, but the documentation has been written for STEPXML.

Compatibility

Integration with Kafka for event messaging is supported via the following versions of Apache Kafka:

- 3.5.1
- 3.4x
- 3.2x
- 3.0x
- 2.6x
- 2.5x
- 2.4x

Message Size Limitation

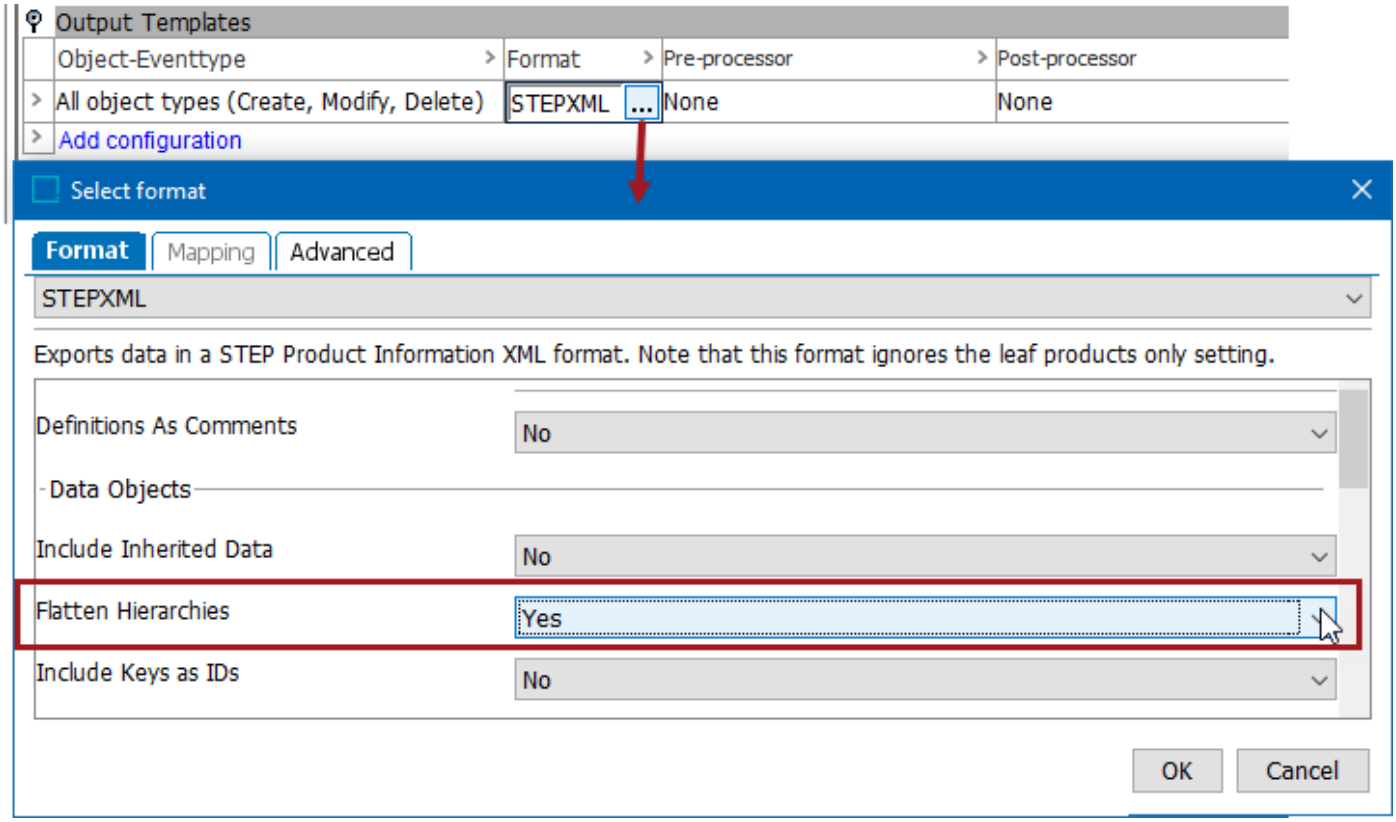
Kafka imposes a default restriction on message sizing to a max of 1 MB. To address this, there are a number of options, with Option 1 being the preferred and recommended method.

Option 1

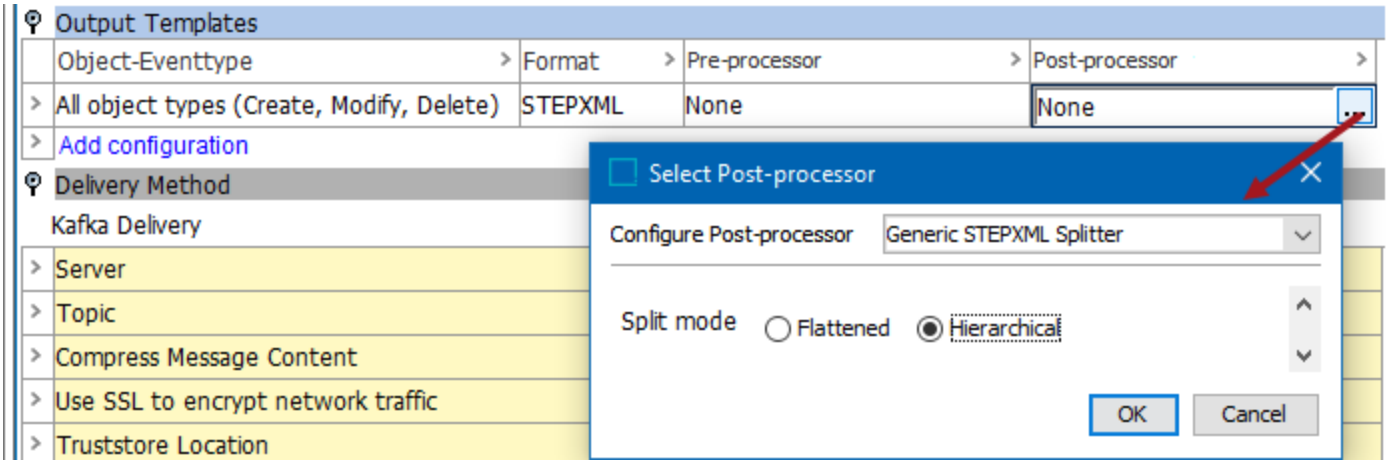
Important: This is the preferred and recommended method.

For Outbound Integration Endpoints: STEPXML format allows using inherited values.

1. On the endpoint, in the Format parameter, select STEPXML and set the 'Flatten Hierarchies' parameter to '**Yes**.'



2. For the Post-processor parameter, select **Generic STEPXML Splitter** and set the 'Split mode' to **Hierarchical**. The Generic STEPXML Splitter splits up STEPXML messages to multiple STEPXML valid fragments containing one single node per STEPXML file.

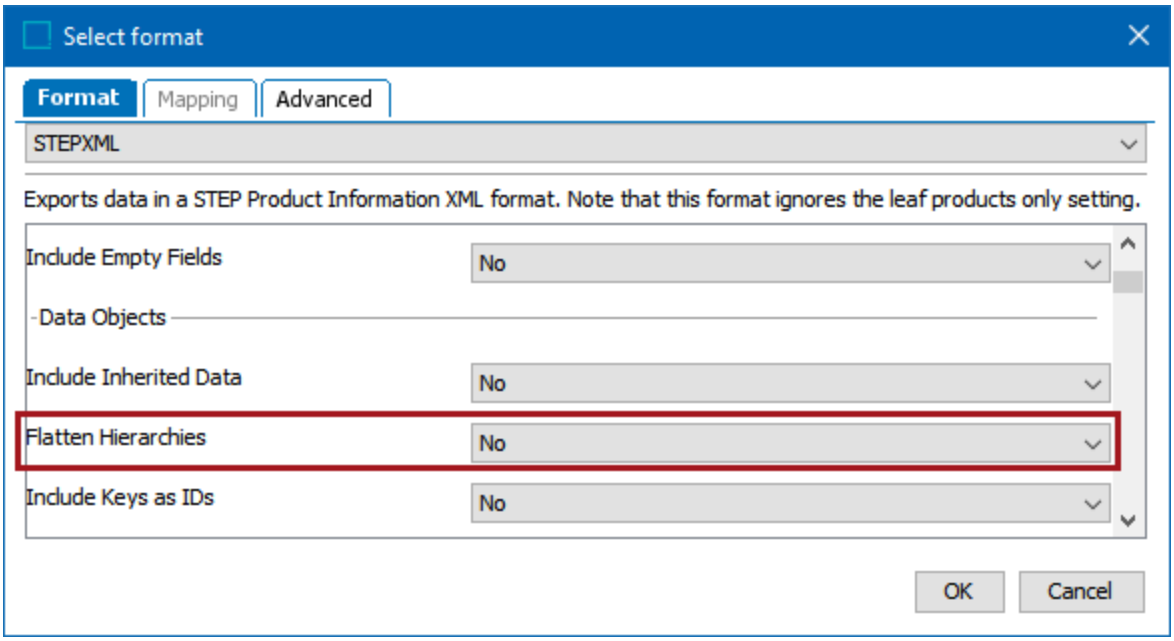


Option 2

Split up STEPXML messages to events for a single node per message and reduce the event XML size.

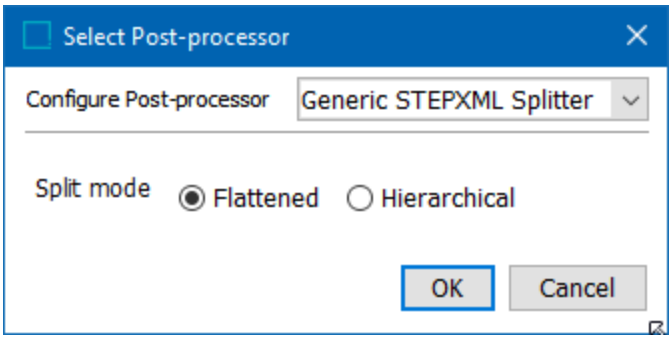
Note: Option 1 is the recommended and preferred method.

1. On the endpoint, in the Format parameter, select STEPXML and set the 'Flatten Hierarchies' parameter to '**No**.' It is not needed for this, as it will be flattened in the post-processor.



2. On the endpoint, select the **Generic STEPXML Splitter** post-processor. The Generic STEPXML Splitter splits up STEPXML messages to multiple STEPXML valid fragments containing one single node per STEPXML file.

3. Use the **Flattened** option to further reduce the size of the individual event messages. Flattened denormalizes any child hierarchies in STEPXML to be written out as a single root hierarchy.



Note: If using inherited values in the export, it is recommended to flatten the values in STEPXML format as demonstrated in Option 1 above.

For more information on post-processors, refer to the **Configure the Pre-processor and Post-processor** section of the following topics:

- **OIEP - Select Objects - Output Templates Flipper**
- **OIEP - Event-Based - Output Templates Flipper**

Option 3

Configure a compression of the event content.

While Kafka does support compression out of the box, it does not help guard against the 1 MB limit, as will be measured on uncompressed message. This option is offered as a method to further address the max message size. LZ4 is currently the only supported compression format. If configured to be used on the integration endpoint when integrating with custom developed Kafka, any producers or consumers must be able to handle compression / decompression of LZ4 event content.

Note: For performance reasons, it is not recommend to use Kafka compression if configuring this explicitly on the endpoint, as this will have limited effect in terms of size.

Important: Ensure all events are processed before changing compression settings for an existing endpoint. Existing events on the Kafka topic are not changed as a result of changing the compression setting.

Topic Partitions

STEP can only read from single-partition topics (topics created with '--partitions 1') but is capable of delivering to multi-partition topics. To have messages distributed across multiple partitions, they must have different keys. By default, the key for all messages published from STEP is the ID of the OIEP responsible. This behavior can be changed by entering a new value for the 'Kafka Message Key Template' field in the Kafka Delivery Method configuration.

The following variables can be used to produce message keys:

- **$endpointId** - ID of the OIEP publishing the message.
- **$nodeType** - The type of node being published (e.g., product, classification, etc).
- **$nodeId** - ID of the STEP node being published.

Note: Because STEP IDs are not unique, $nodeType can be combined with $nodeId to create a unique identifier.

For more information, refer to the **Kafka Delivery Method** topic in the **Creating an Outbound Integration Endpoint** section of the **Outbound Integration Endpoints** documentation.

Important: In order to publish to multi-partition topics and use the $nodeType and $nodeId variables, ensure that messages only contain data for a single STEP object. If exporting STEPXML the Generic STEPXML Splitter must be used. If using the Generic JSON format option, the OIEP must be configured to run with a batch size of '1.'

Event Message Serialization

This is relevant if implementing your own Kafka producers / consumers for integration with STEP.

1. A event produced by STEP has the outbound integration endpoint ID set as a key. When implementing your own producer, this must be the same key set per producer because the key is used for partitioning purposes.
2. The key is passed in as a string, which is serialized with the standard Kafka: org.apache.kafka.common.serialization.StringSerializer.
3. The message content itself is a bytearray that is serialized with: org.apache.kafka.common.serialization.ByteArraySerializer.
4. On the consumer side, deserialize with the opposing deserializer for key and message content. Implementations are namely: org.apache.kafka.common.serialization.StringDeserializer and org.apache.kafka.common.serialization.ByteArrayDeserializer

Reference Handling for Dangling References

The Kafka connector and the Generic STEPXML Splitter occasionally have known issues when processing references where the referred to node has not yet been created in the target system. This creates a 'dangling reference.'

Often this dangling reference can be solved by being consistent with data handling, however with an inbound receiver that is having issues with referential integrity, the following can help.

Note: This is relevant only when dealing with referential integrity issues on an inbound receiver.

Example 1

In the following example, two outbound events contain XML that is generated in the events queue.

Event 1: circularref1

```
<?xml version="1.0" encoding="UTF-8"?>
<STEP-ProductInformation ExportTime="2020-09-28 14:38:18" ExportContext="GL"
ContextID="GL" WorkspaceID="Approved" UseContextLocale="false">
<Product ID="circularref1" UserTypeID="Product" ParentID="Product hierarchy root"
Changed="true">
<Name Changed="true">circularref1</Name>
```

```
<ProductCrossReference ProductID="circularref2" Type="prodToProd" Changed="true">
</ProductCrossReference>
</Product>
</STEP-ProductInformation>
```

Event 2: circularref2

```
<?xml version="1.0" encoding="UTF-8"?>
<STEP-ProductInformation ExportTime="2020-09-28 14:38:18" ExportContext="GL"
ContextID="GL" WorkspaceID="Approved" UseContextLocale="false">
<Product ID="circularref2" UserTypeID="Product" ParentID="Product hierarchy root"
Changed="true">
<Name Changed="true">circularref2</Name>
<ProductCrossReference ProductID="circularref1" Type="prodToProd" Changed="true">
</ProductCrossReference>
</Product>
</STEP-ProductInformation>
```

For 'circularref1' and ' circularref2' to both be successfully approved upon import, partial approval is given to 'circularref1' without the reference. Then 'circularref2' is fully approved with its reference to 'circularref1'. A second approval is then given to 'circularref1' so that the reference from 'circularrelf2' can be added.

In this case, the two XMLevents would look like the following, and not cause issues with referential integrity:

```
<?xml version="1.0" encoding="UTF-8"?>
<STEP-ProductInformation ExportTime="2020-09-28 14:38:18" ExportContext="GL"
ContextID="GL" WorkspaceID="Approved" UseContextLocale="false">
<Products>
<Product ID="circularref2" UserTypeID="Product" ParentID="Product hierarchy root"
Changed="true" Referenced="true">
<ProductCrossReference ProductID="circularref1" Type="prodToProd" Changed="true">
<Product ID="circularref1" UserTypeID="Product" ParentID="Product hierarchy root"
Changed="true"></Product>
</ProductCrossReference>
</Product>
</Products>
</STEP-ProductInformation>
```

Example 2

Consider that both 'circularref1' and 'circularref2' are new products to be imported into STEP or a downstream system.

Product 1: circularref1

```
<?xml version="1.0" encoding="UTF-8"?>
<STEP-ProductInformation ExportTime="2020-09-28 14:38:18" ExportContext="GL"
ContextID="GL" WorkspaceID="Approved" UseContextLocale="false">
<Product ID="circularref1" UserTypeID="Product" ParentID="Product hierarchy root"
Changed="true">
```

```
<Name Changed="true">circularref1</Name>
<ProductCrossReference ProductID="circularref2" Type="prodToProd" Changed="true">
</ProductCrossReference>
</Product>
</STEP-ProductInformation>
```

Product 2: circularref2

```
<?xml version="1.0" encoding="UTF-8"?>
<STEP-ProductInformation ExportTime="2020-09-28 14:38:18" ExportContext="GL"
ContextID="GL" WorkspaceID="Approved" UseContextLocale="false">
<Product ID="circularref2" UserTypeID="Product" ParentID="Product hierarchy root"
Changed="true">
<Name Changed="true">circularref2</Name>
<ProductCrossReference ProductID="circularref1" Type="prodToProd" Changed="true">
</ProductCrossReference>
</Product>
</STEP-ProductInformation>
```

Notice that 'circularref1' references 'circularref2' and 'circularref2' references 'circularref1.' When using the Generic STEPXML Splitter during an inbound integration endpoint (IIEP), it would fail because the product 'circularref2' would not exist yet on the system for circularref1' to reference. The referential integrity is not being kept.

To fix this problem you can:

1. Fix this manually by creating 'circularref2' in the system, and then rerun the IIEP.
2. Remove the events from the Kafka topic using **kafka-delete-records.sh** or a similar command
3. Instead of using the Generic STEPXML Splitter, limit the batch size on the endpoint to be able to deliver more nodes in one XML event. Use LZ4 compression to reduce the message size as much as possible. In most cases, the 1 MB limit affords a larger number of events to be processed, but this is a system-specific consideration. If needed, the 1 MB limit can be raised on the Kafka cluster to allow for larger batches to be processed. For any downstream systems that may be receiving these events, make them aware of these changes. A form of eventual consistency with regards to references may need to be implemented.
4. Use advanced STEPXML and then employ embedding for the references. This option could produce more verbose XML and could exceed the 1 MB message size limit if used on a system with many references. Make sure to reconfigure the Generic STEPXML Splitter to 'Hierarchical' rather than 'Flattened' for this option. An example template that embeds the references into the event message is the following:

```
<STEP-ProductInformation ResolveInlineRefs="true" FollowOverrideSubProducts="true">
<UserTypes ExportSize="Selected"/>
<EdgeTypes/>
<CrossReferenceTypes ExportSize="Selected"/>
<Products ExportSize="Minimum">
<Product>
<Name/>
<ProductCrossReference Type="prodToProd">
```

```
<Product Referenced="true" Embedded="true">
<empty/>
</Product>
</ProductCrossReference>
</Product>
</Products>
</STEP-ProductInformation>
```

Kafka Topic Offset

To achieve transactional integrity in STEP its important to note that the Kafka topic offset is managed in STEP rather than stored in Apache Zookeeper. This way if some message processing fails in the importer, it is possible to rollback the transaction alongside the topic offset increment. In doing this, it achieves transactional integrity when reading from the Kafka topic. The offset is associated with the inbound integration endpoint (IIEP) itself. For this reason, switching an IIEP to read from another topic is not allowed, as this would inherit the offset. Instead, a new IIEP should be created from scratch and configured to read from the other Kafka topic.

In workbench, if the topic offset of an IIEP is larger than 0, topic changes are not allowed. The topic in the configuration wizard is rendered as a read-only option.

Note: If you export / import inbound integration endpoints between systems, the imported endpoint will always start from offset 0. If you restore a STEP database backup the offset for this will also be restored.

Static Membership

To avoid issues with Kafka rebalance, and to generally make reconnecting an endpoint to Kafka faster, a static membership is employed. As a result, the connection info for the IIEP on Kafka is cached for a default of up to 30 minutes. If the IIEP is scheduled to run before the 30 minutes elapse, the reconnects are a bit faster. For this reason the session timeout in the Kafka driver is set to be 30 minutes which is the default maximum on a Kafka installation. If this is configured differently, set this timeout manually from the default setting, which is 1800000 for 30 minutes in milliseconds. To set manually, navigate to the **Kafka.Receiver.<ENDPOINT-ID>.SessionTimoutInMs** configuration property.

SASL Authentication

Support for Simple Authentication and Security Layer (SASL) authentication (both PLAIN and SCRAM) is supported for the Kafka receiver and delivery plugins. Using SASL gives you more data security options and allows for alternatives to the other array of Kafka connector authentication functionality support, which includes support for AWS MSK, Heroku, and Aiven (with TLS Client Certificate Authentication).

The properties need to be added to the sharedconfig.properties file. Below is an example config of PLAIN username/password authentication:

```
Kafka.Receiver.YOURENDPOINTID.ExtraDriverOptions=sasl.mechanism=PLAIN,security.proto
col=SASL_
PLAINTEXT,sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule
required username="admin" password="admin-secret";
```

For SASL_SSL with PLAIN username/password authentication, the Keystore configuration in the SSL part of the Kafka Receiver / Delivery plugin can be omitted. If there is no requirement that the Kafka server has to trust our SSL certificate, then none is needed. A Truststore Location / Password is required to indicate that your system trusts the Kafka Servers Certificate.

Below is a sample config for SCRAM authentication:

```
Kafka.Receiver.YOURENDPOINTID.ExtraDriverOptions=sasl.mechanism=SCRAM-SHA-
256,security.protocol=SASL_
SSL,sasl.jaas.config=org.apache.kafka.common.security.scram.ScramLoginModule
required username="admin" password="admin-secret";
```

All of the configuration options are taken from confluent.io documentation on how to configure SASL authentication found in this link: https://docs.confluent.io/platform/current/kafka/overview-authentication-methods.html.

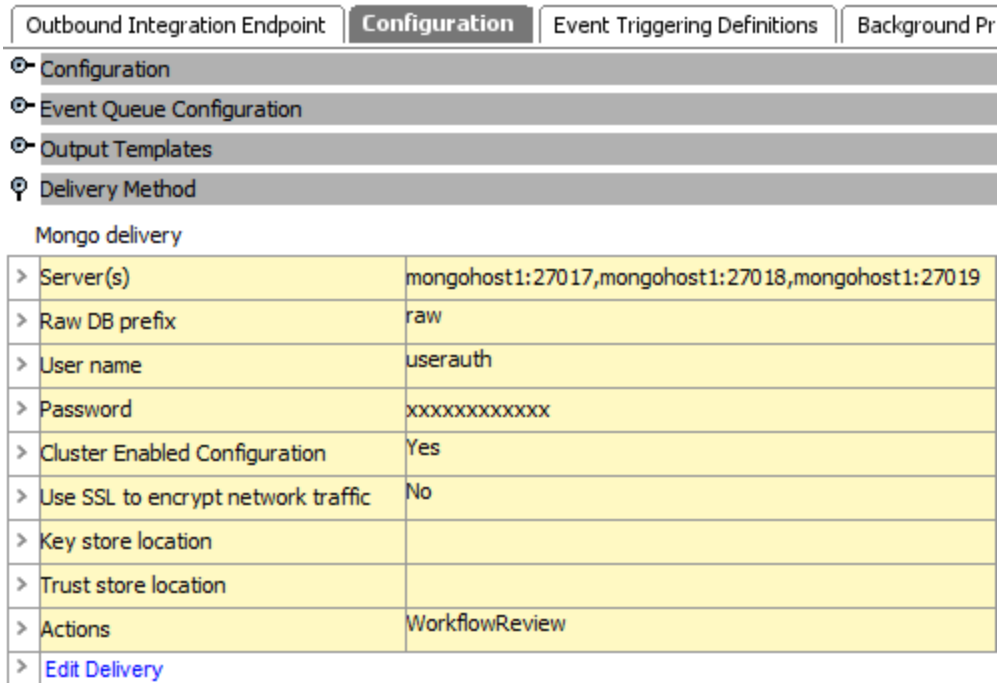
Mongo Delivery Method

Prior to selecting the Mongo Delivery Method, ensure your setup meets the prerequisites explained in **Prerequisites for Configuring the MongoDB Adapter** documentation. This delivery option is only available in OIEPs.

For more information, refer to https://docs.mongodb.com/manual/ or the **MongoDB Adapter Setup Quick Guides** documentation.

For more information on the Mongo Delivery Method functionality, refer to the **Mongo Delivery Method Elements** documentation.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



Prerequisites

Changes to the properties file, outlined below, are implemented when the server is restarted.

1. If your system displays radio button options for the server selection, refer to the **Legacy Configuration** section below for more details.

2. Prior to configuration, clicking the **Server(s)** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive dynamic **MongoDB.Server.Configuration.[n]** property, where the [n] is used for differentiation. The configuration entries must be unique (MongoDB.Server.Configuration.1, MongoDB.Server.Configuration.2, MongoDB.Server.Configuration.3, etc.). When duplicates exist, only the last value is displayed in the dialog. Each host is separated from the respective port using a colon (:), and additional servers are added using a comma-separated list. For example:

   ```
   MongoDB.Server.Configuration.1=host4:40004
   ```

   indicates an independent MongoDB instance. A second and third configuration could be supplied for servers being used in a cluster using the format:

   ```
   MongoDB.Server.Configuration.2=host1:10001,host2:20002,host3:30003
   MongoDB.Server.Configuration.3=host4:40004,host5:50005,host6:60006,host7:70007
   ```

3. The **Key store location** is only required if the Mongo DB requires the clients to use SSL to encrypt connections.

   The password for the key store is configured in sharedconfig.properties on the STEP application server, using the case-sensitive **MongoDB.SSLKeyStorePassword** property. For example:

```
MongoDB.SSLKeyStorePassword = keystore_password
```

4. In **Trust store location**, is only required if the Mongo DB requires the clients to use SSL to encrypt connections.

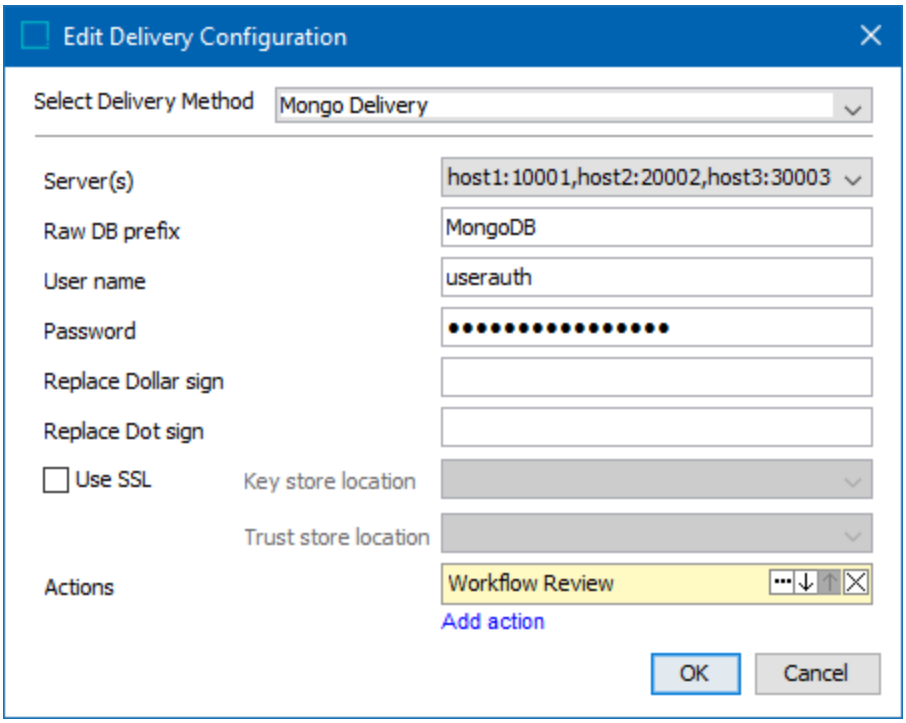
   The password for the key store is configured in sharedconfig.properties on the STEP application server, using the case-sensitive **MongoDB.SSLTrustStorePassword** property. For example:

   ```
   MongoDB.SSLTrustStorePassword = truststore_password
   ```

Configuration

For information on a parameter, hover over the parameter field to display help text.

1. In **Select Delivery Method**, choose **Mongo Delivery**.



Note: For MongoDB 4.2 and 4.4, choose **Mongo Delivery (from release 4.2)** from the **Select Delivery Method** dropdown menu.

2. In **Server(s)** dropdown, select the desired MongoDB server configuration, either an independent MongoDB instance or a MongoDB cluster setup.

3. In **Raw DB prefix**, if required, enter a prefix for all raw databases the adapter creates. The output is [prefix] + [context], resulting in a database name of MongoDBContext1 using the data in the image above, and exporting for Context1. Optional.

4. In **User name**, if MongoDB is configured to use authentication, enter the MongoDB user. Optional.
5. In **Password**, if MongoDB is configured to use authentication, enter the MongoDB password. Optional.
6. **Replace Dollar sign** performs replacements for JSON document key names. Text entered replaces any dollar sign ($) during delivery. If blank, all dollar signs in key names are replaced automatically by the JSON converter with Unicode 'uFF04.' This allows overriding the default Unicode replacement with the specified character(s), for example, an underscore (_).
7. **Replace Dot sign** performs replacements for JSON document key names. Text entered replaces any dot sign (.) during delivery. If blank, all periods in key names are replaced automatically by the JSON converter with Unicode 'uFF0E.' This allows overriding the default Unicode replacement with the specified character (s), for example, an underscore (_).
8. Check **Use SSL** if MongoDB requires the client to use SSL encryption. Optional. If no further parameters are configured, it is assumed that the MongoDB uses a SSL certificate that is issued by a trusted certificate authority (like Verisign or Thawte). If this is not the case, or if more security of the connection is requested, you can configure an SSL key store and an SSL trust store.
    - In **Key store location**, enter a key store location, for example, file://[path]/key_store.jks. The Key store holds the STEP Mongo Adapter private key and SSL certificate in a keystore file as generated by the Java utility 'keytool' (in jks format). The password for the key store is configured as defined in the **Prerequisites** section.
    - In **Trust store location**, enter a trust store location, for example, file://[path]/trust_store.jks. The trust store holds the Mongo DB public key and certificate in a keystore file as generated by the Java utility 'keytool' (in jks format). The trust store configuration is only needed if the certificate is self-signed and not issued by a trusted certificate authority. The password for the key store is configured as defined in the **Prerequisites** section.
9. In **Actions**, click the **Add action** link to enter one or more STEP business actions used to create and populate aggregated collections. These actions are run after the raw collections have been populated.
10. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

Delivering Delete Events

The Mongo delivery method requires the following specific OIEP settings to successfully deliver delete events:

1. The OIEP is event-based, verified by viewing that the Configuration tab includes the Event Queue Configuration flipper. For more information, refer to the **OIEP - Event-Based - Event Queue Configuration Flipper** topic.
2. In the Event Queue Configuration flipper, the Queue Status parameter is set to Read Events.
3. In the Output Templates flipper, the Object-Eventtype column includes 'Delete' as an event type. For more information, refer to the **OIEP - Event-Based - Output Templates Flipper** topic.
4. Also in the Output Templates flipper:

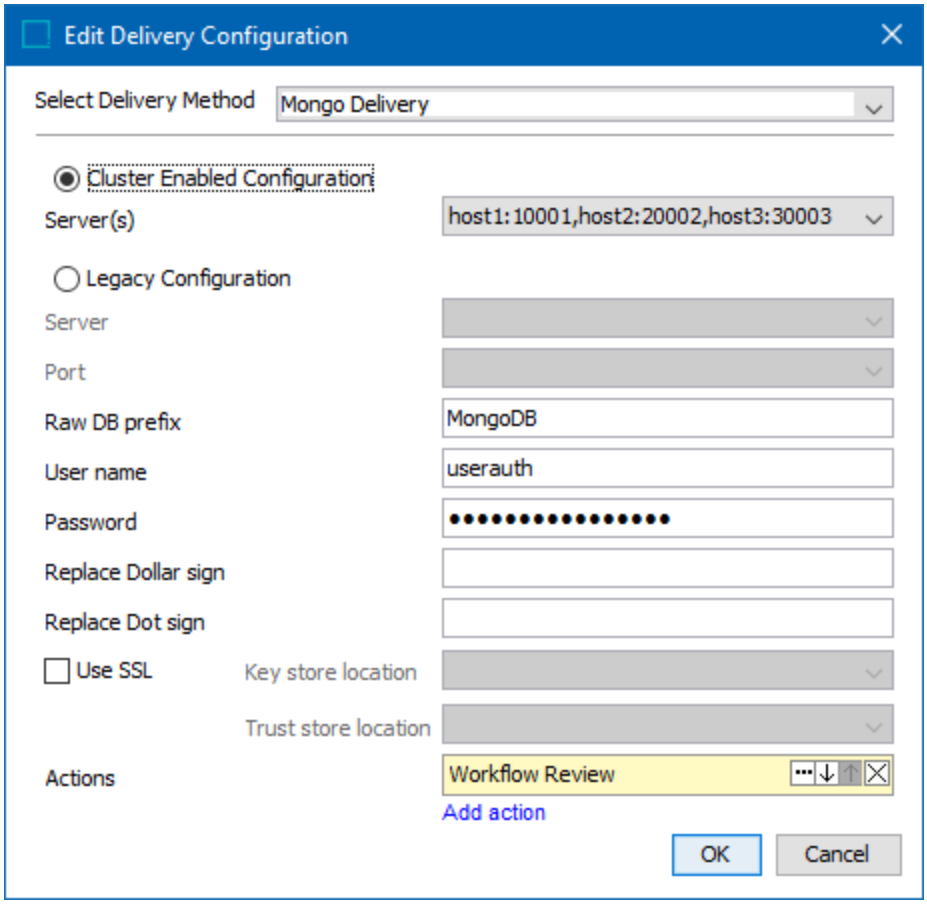
- The **Format** column displays 'Advanced STEPXML'.
- The **Template** includes the appropriate <DeleteProducts/> tags.

  Advanced STEPXML is the only format that allows delete events to be included in the XML. For more information, refer to the **Event-Based OIEP Triggered by Deleting Products, Classifications, and Assets** section of the **Delete Objects in STEPXML** topic.

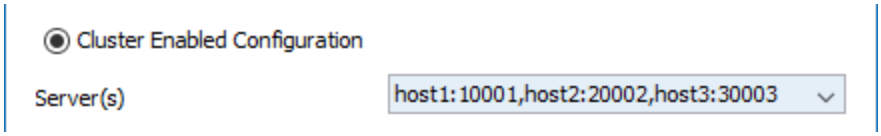
Legacy Configuration

When the sharedconfig.properties file includes the MongoDB.Server property and the MongoDB.Port property, the dialog displays both a cluster option and a legacy option. Typically, this happens when a STEP system had a MongoDB host / port configured prior to the update that introduced the clustered configuration functionality.

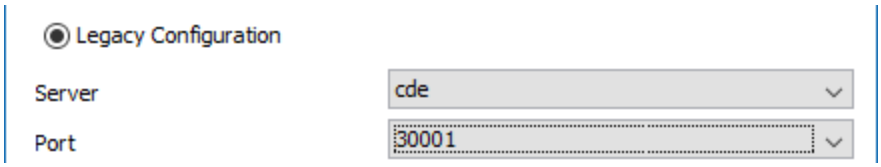
Note: Upgrading the sharedconfig.properties file to modify all configurations using the legacy properties to the cluster enabled property, and then removing the legacy properties completely, results in the radio button options being removed from the dialog, as shown in the image at the top of this topic. Additionally, the cluster option and the legacy option are not included in the **Mongo Delivery (from release 4.2)** delivery method.



When the **Cluster Enabled Configuration** radio button is selected, the **Server(s)** dropdown displays entries for the dynamic MongoDB.Server.Configuration.[n] property.



When the **Legacy Configuration** radio button is selected, the **Server** and **Port** dropdowns display entries for the MongoDB.Server and MongoDB.Port properties.



Multiple independent servers or ports can be listed using the following format:

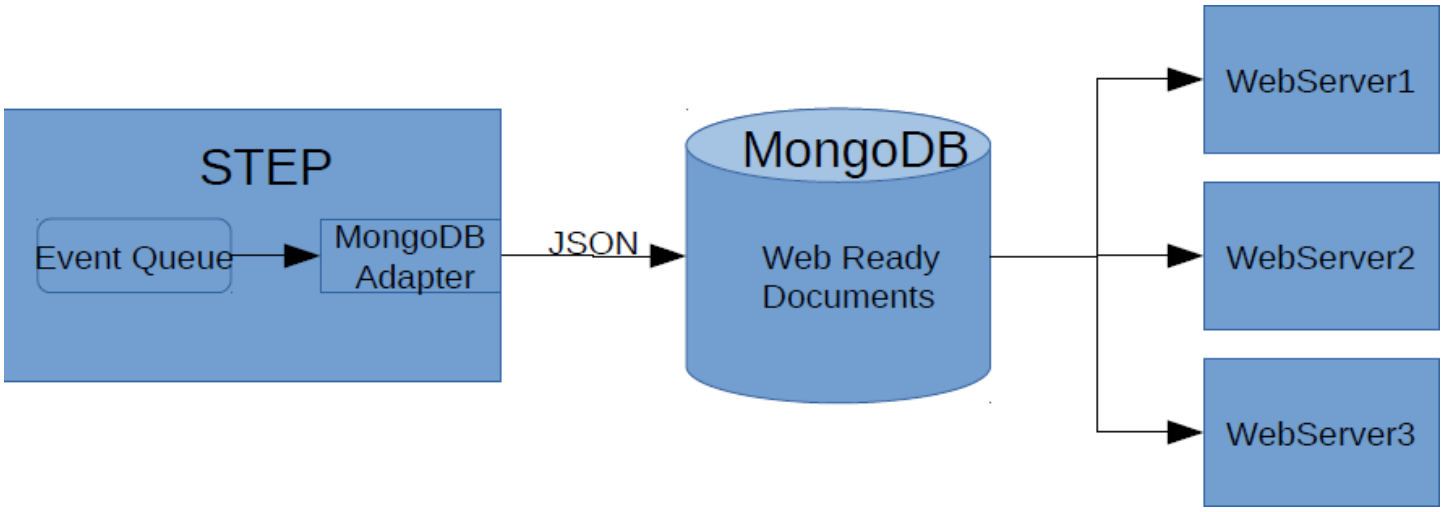
```
MongoDB.Server=1=vps124887.vps.ovh.ca,2=cde,3=rty
MongoDB.Port=1=30001,2=30002,3=30003
```

Mongo Delivery Method Elements

The STEP Mongo Adapter is an outbound integration endpoint delivery option that receives data from a STEP event queue and loads it into a MongoDB database. The MongoDB is often used for website back-end, reporting, and high performance feeds to other back-end systems.

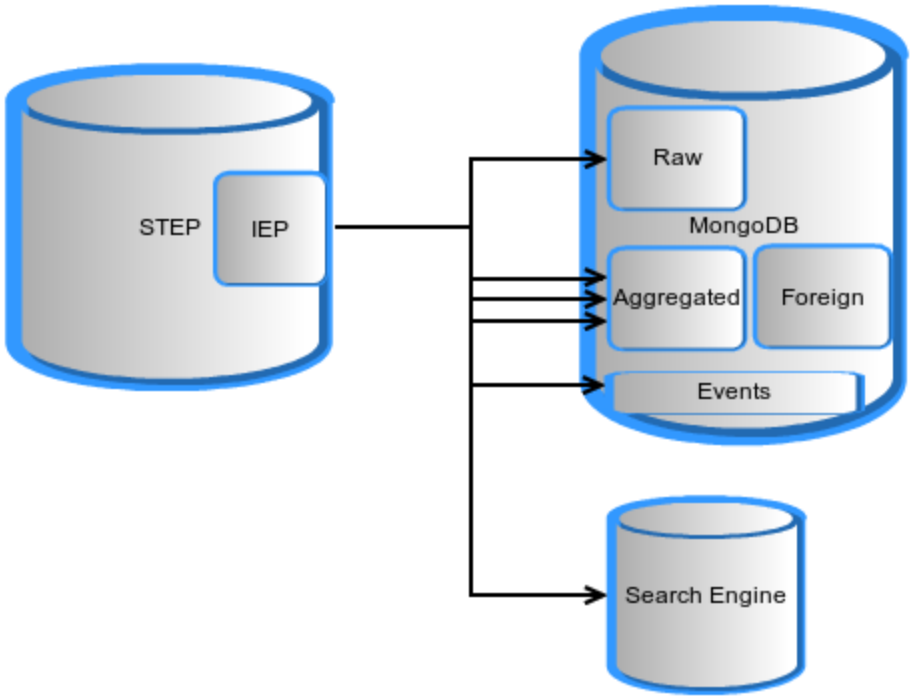
The Mongo adapter converts STEPXML to JSON and keeps a set of 'raw' collections in MongoDB that are in sync with STEP. Additionally, it allows you to add JavaScript triggers to maintain aggregated collections.

The following illustration shows how the Mongo Adapter can be used to feed a Mongo database that is used as the back-end to a website:



By keeping collections of web-ready JSON documents available in MongoDB, the JSON can be passed directly to the browser's JavaScript. This results in the web server having little processing to do, which means that it needs to pass the right collection result directly back to the browser.

The following drawing illustrates the collections involved:



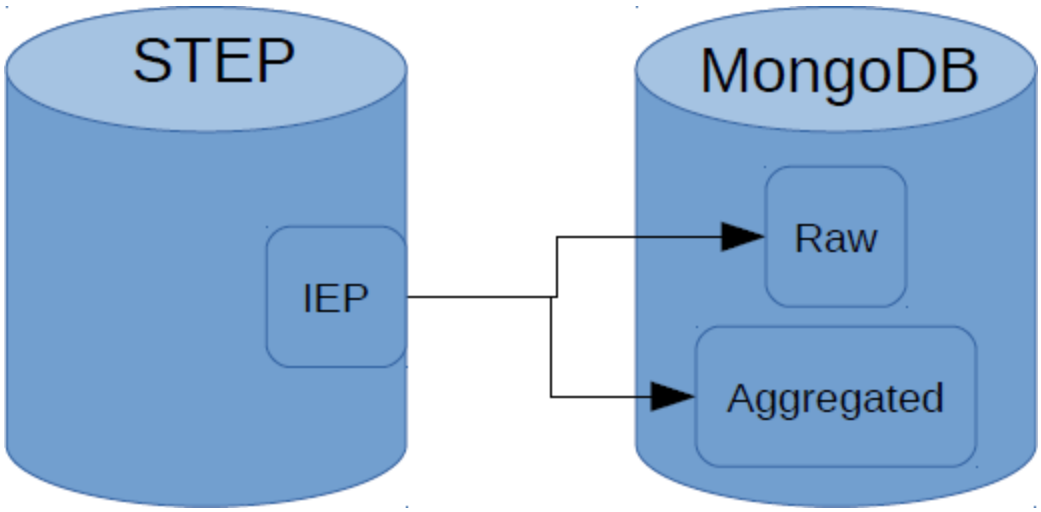
The STEP Mongo Adapter ensures that the raw collections are a one-to-one mirror of data in STEP. The adapter does not embed nodes found through references or child associations. These nodes are stored separately in the raw collections.

It is possible to configure the adapter so that it creates one or more customized, aggregated MongoDBs (databases or collections) where the documents can be tailored to meet the exact usage requirements, like reporting, analysis, and so on.

The JavaScript triggers allow maintaining aggregated collections, for example:

- List of children, including title and other data necessary in a tree-like view.
- Information from the target end of references that is needed in a given view.
- Mapping to a JSON structure that is closer to the website data model.

The following image shows a MongoDB database containing a raw database that is maintained directly by the STEP Mongo Adapter and an aggregated database maintained by JavaScript triggers.



A JavaScript trigger is executed under certain conditions, for example, only for a specific object type. This reduces the amount of checking necessary in the JavaScript code. Once the raw collection is updated and the JavaScript triggers (if configured) have completed, an event is inserted into the events collection. Also, an update is sent to the search engine.

Note: Aggregated collections and raw collections do not need to be in the same MongoDB instance. However, if required, different collections in the aggregate can be combined into their own MongoDB instance.

Default Databases and Collections

The STEP Mongo Adapter maintains a number of MongoDBs, referred to as 'raw databases', and a number of collections in the raw databases, referred to as 'raw collections.' A raw database is created for each context that is exported and the database is named after the context. For example, if the two contexts EN and FR are exported, two raw databases are created, one named EN and one named FR.

It is possible to configure a string that prepend the raw database name when the STEP Mongo Adapter is configured. When data is exported from multiple contexts, a JSON document is created per context, per exported object. Continuing the previous example, two JSON documents are created for each object: one for the object in the EN context and one for the object in the FR context.

A raw collection is only created when the particular STEP data type is exported. So, if no assets are exported from STEP, no asset collection is created. Each raw database contains all or some of the following raw collections:

Raw Collection	Purpose
product	Exported products

Raw Collection	Purpose
asset	Exported assets
classification	Exported classifications
entity	Exported entities
attribute	Exported attribute definitions
data containers	Exported data container definitions
referenceType	Exported reference types
listofvalues	Exported list of values (LOVs)
unit	Exported unit definitions

Inherited values, calculated values, and unit names on values are handled already in the raw collections. Since changes to these affect replicated information, STEP ensures that appropriate updates are made to sync the replicated information.

STEP JSON and Mongo JSON

The STEP Mongo Adapter converts STEPXML input to STEP JSON (JavaScript Object Notation) documents, and then stores the JSON documents in the MongoDB. The STEP JSON schema can be downloaded from your STEP server at: http:///*[enter step-server]*/files/StepSchema.json

Formally, MongoDB documents are BSON documents, which is a binary representation of a JSON document. For more information, refer to the MongoDB documentation on the web.

For an expanded example of the basic conversion using STEP Mongo Adapter, refer to the **Mongo Delivery Method Conversion Example** documentation.

JavaScript Triggers

STEP business action triggers with JavaScript populate the aggregated MongoDB databases and collections. You can configure one or more business action triggers for the STEP Mongo Adapter. The business action triggers are fired after the STEP Mongo Adapter has populated the raw collections.

During configuration of the STEP Mongo Adapter, the configuration wizard allows you to select multiple JavaScript to run under different conditions by adding preconditions to the business actions. A precondition is, for example, a JavaScript trigger that is only fired for updates to a given type of objects. For more information, refer to the **Applies If Tab** section of the **Editing a Business Rule or Function** documentation.

Due to the nature of inheritance and calculated attributes, the trigger may be called even if there are no changes to the underlying raw objects.

The business action triggers are executed once per JSON document sent to the MongoDB database. If two objects are exported from STEP, two JSON documents are sent to the MongoDB database and the business action triggers are executed twice. If the objects are exported in multiple contexts, one JSON document is generated per context, per object. This means that if two products in two contexts are exported, the business action triggers are executed four times.

Business action triggers are executed on the STEP server. Business actions triggers need access to the MongoDB database to fetch data from the raw collections and write or read data to / from the aggregated collections. For the best performance, network latency between the STEP server and the MongoDB server should be minimal.

Important: Place aggregated collections in their own database so that the raw database contains only collections.

Bound Variables

The JavaScript trigger have accesses to information in the Mongo Adapter environment by binding script variables when the JavaScript trigger is defined.

Type	Gives access to
MongoDBContext	• The current STEP context. • The name of the raw MongoDB database. • The MongoDB database via the com.mongodb.Mongo object. For more information, refer to: http://api.mongodb.org/java/current/com/mongodb/Mongo.html. The actual Mongo object that is passed can be a wrapper, but it adheres to the interface
JSON context	The JSON object with the data that has been persisted in MongoDB.

For more information about executing JavaScript in business rules, refer to the **Business Action: Execute JavaScript** section of the **Business Rules** documentation.

Example JavaScript to Maintain a Collection in a Classification

The following code-snippet illustrates an example JavaScript that maintains a collection containing the list of products (ID and Name) in a classification.

The structure of the collection is:

```
{_id: "<classification-id>", products : { { id: "<product-id1>", name: "<product-name1>"}, { id: "<product-id2", name: "<product-name2>"}, ... }
```

```
function getClassificationIDs(classrefs) {
    var result = new Array();
    if (classrefs != null) {
        for (i = 0; i < classrefs.targets.size(); ++i) {
            result.push(new String(classrefs.targets[i].targetID));
        }
    }
    return result;
}

var db = mongoContext.getMongo().getDB('extra');
var collection = db.getCollection('classificationproducts');

var classrefs = mongoData.references ? mongoData.references["Web Classifications"] :
null;

var classificationIDs = getClassificationIDs(classrefs);

var deletequery = {"products.id" : mongoData._id, "_id" : { $nin :
getClassificationIDs(classrefs) }};
var adelete = { $pull : { products : { id : mongoData._id } } };

collection.update(deletequery, adelete, false, true);

var query = {"products.id" : mongoData._id };
var name = mongoData.name;
var update = { $set : { "products.$.name" : name } };

collection.update(query, update, false, true);

classificationIDs.map(function(item) {
    var query = { _id : item };
    var update = { $addToSet : { products : { id : mongoData._id, name : name } } };
    collection.update(query, update, true, false);
});
```

Prerequisites for Configuring the MongoDB Adapter

The following are prerequisites for setting up the MongoDB adapter:

1. A MongoDB instance with suitable amounts of disk space.
2. The STEP MongoDB Adapter uses a network port to access the MongoDB database on the server where the MongoDB is running. The network port must be open and accessible from the STEP servers.
3. Update sharedconfig.properties or config.properties with the **name of the MongoDB server**, using the MongoDB.Server configuration property and a list of number-to-value pairs, separated by commas, as follows:

   ```
   MongoDB.Server = 1=MongoDBServer1, 2=MongoDBServer2, 3=MongoDBServer3
   ```

4. Update sharedconfig.properties or config.properties with the **MongoDB port** using the MongoDB.Port configuration property, and a list of number-to-value pairs, separated by commas, as follows:

   ```
   MongoDB.Port = 1=10001, 2=2002, 3=3003
   ```

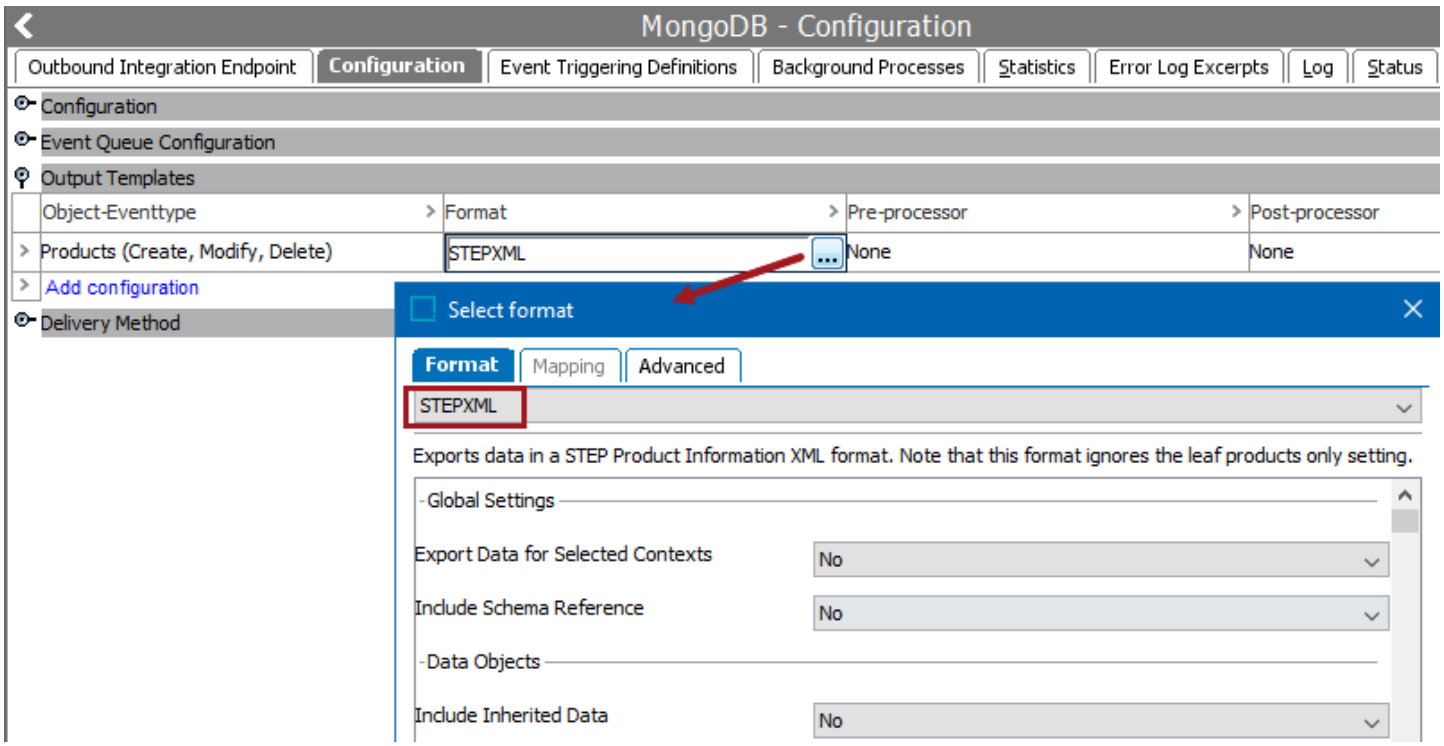
   For example, to configure the adapter to use a MongoDB instance running on either 'Server1' or 'Server2' using port '27712' on both servers, the properties would be:

   ```
   MongoDB.Server = 1=Server1, 2=Server2
   MongoDB.Port = 1=27712
   ```

Configure the OIEP

The STEP MongoDB Adapter is run within an outbound integration endpoint (OIEP) and is configured as the delivery method for the endpoint.

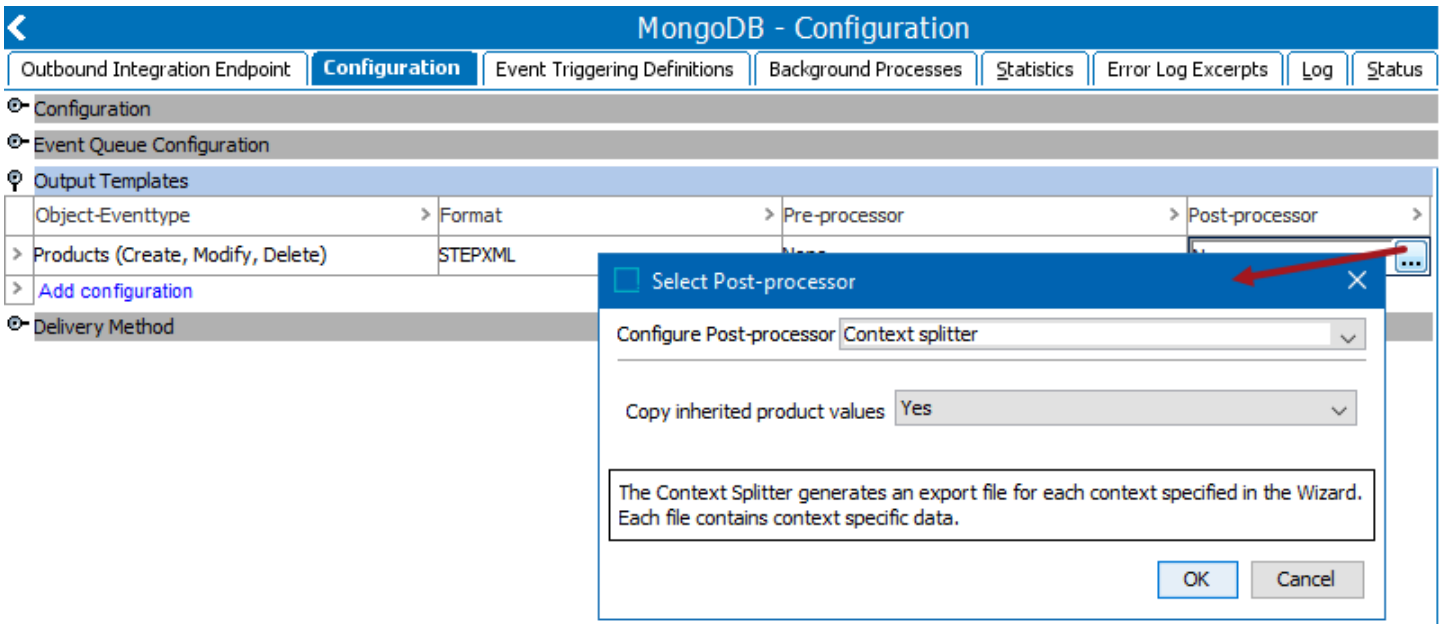
1. In System Setup, create an OIEP, selecting STEP Exporter as the Process Engine. For details on creating a new OIEP, refer to the **Creating an Event-Based Outbound Integration Endpoint** topic.
2. For the new OIEP, open Configuration > Output Templates > Add configuration > Format > click the ellipsis button (...).



3. On the Select Format dialog, leave objects as default values except for the following parameters:
   - Select the **STEPXML** export format in the dropdown at the top of the dialog
   - Data Objects > **Flatten Hierarchies** = Yes
   - Data Objects > **Include Entities** = Selected
   - Data Objects > **Include Data Containers** = Yes
   - Data Objects > **Include Products** = Selected
   - Data Objects > **Include Classifications** = Selected
   - Data Objects > **Include Assets** = Selected
   - Configuration > **Include Attribute Groups** = Minimum
   - Configuration > **Include Data Container Definitions** = Minimum
   - Configuration > **Include Link, Reference and Object Types** = Minimum
   - Configuration > **Include List of Values** = Minimum
   - Configuration > **Include Units** = Minimum

   For more information on these STEPXML outbound parameters, refer to the **STEPXML Outbound Parameters** topic of the **Data Exchange** documentation.

4. In the Post-processor field, click the ellipsis button (...) and select **Context splitter**, then click OK.



5. For Delivery Method, if MongoDB requires user authentication, configure a MongoDB user and password. The MongoDB user must be configured in the MongoDB admin database and have permissions to create new MongoDB databases (the MongoDB role "readWriteAnyDatabase").

Configuring the JavaScript Triggers

JavaScript triggers are fired for all updates to the raw collections. The purpose of the triggers is to maintain the aggregated collections. The JavaScript triggers are configured as business actions in STEP and then applied when configuring the MongoDB Adapter.

Continue with setting up the Mongo Delivery Method options, by following the **Mongo Delivery Method** documentation.

MongoDB Adapter Setup Quick Guides

The following information is available to assist in the setup of Mongo Delivery Method.

STEP JSON Schema

The STEP JSON schema can be downloaded from your STEP server at: http:///*[enter step-server]*/files/StepSchema.json

Configuring Mongo Authentication Quick Guide

The following quick guide describes how to configure a Mongo database to use authentication. Details may be found in the Mongo documentation on the web.

The guide assumes that the setup is used on a Mongo database installation running as a single server installation (i.e., the Mongo database is running as a standalone server).

- Start the Mongo database without authentication.
- Log onto the database using the Mongo client.
- Create a system administrator user.

```
> use admin
switched to db admin
> db.createUser(
   {
     user: "admin",
     pwd: "password",
     roles:
     [
       {
        role: "userAdminAnyDatabase",
        db: "admin"
       }
     ]
   }
)

Successfully added user: {
  "user" : "admin",
  "roles" : [
   {
    "role" : "userAdminAnyDatabase",
    "db" : "admin"
   }
  ]
}
```

4. Create the user that will be used when the MongoDB adapter logs onto the Mongo database. This user should be created in the admin database with roles to read and write any databases in Mongo:

```
> use admin
switched to db admin
> db.createUser( {"user" : "stepsys", "pwd" : "stepsys", "roles" : [
"readWriteAnyDatabase" ] })
Successfully added user: { "user" : "stepsys", "roles" : [ "readWriteAnyDatabase" ]
}
```

5. Stop the Mongo database and enable authentication. This may be done in the Mongo configuration file:

```
auth = true
```

or from the command line:

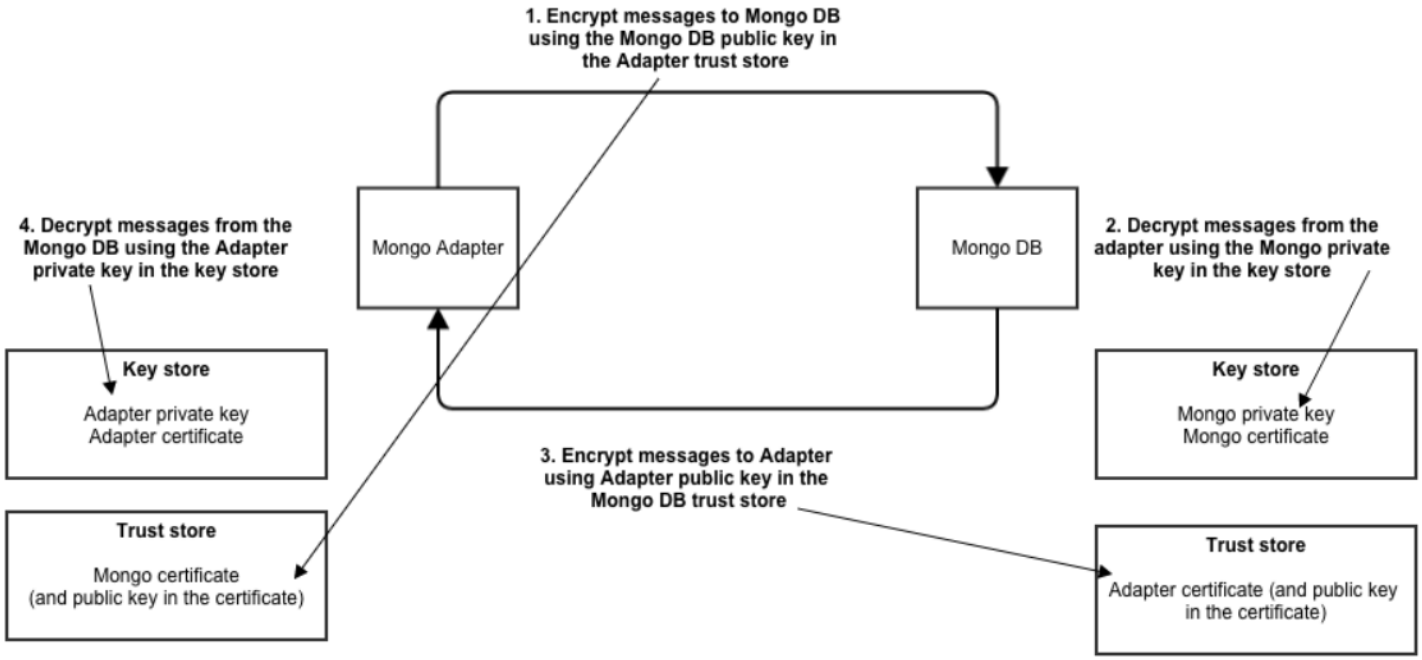
```
mongod --auth --config
```

SSL Configuration Quick Guide

When the Mongo DB adapter is configured for SSL, a key store and a trust store must be configured.

- The key store is a key store file in jks format containing the Mongo Adapter (STEP) private key and certificate.
- The trust store is a trust store file in jks format containing the Mongo DB public key and certificate.

Key store files may be created from the public and private key files and certificates using utilities like the java keytool utility. The diagram below illustrates the uses of the key store and trust store.



To configure SSL encryption on the STEP-to-Mongo database connection:

- Configure the Mongo database to use SSL: http://docs.mongodb.org/manual/tutorial/configure-ssl/
- Configure the Mongo client to connect to a Mongo database using SSL: http://docs.mongodb.org/manual/tutorial/configure-ssl-clients/

The Mongo database available as a free download does not support SSL. (SSL requires the Enterprise edition of Mongo.) Instead, download the appropriate source and compile your own version of the Mongo database from the web: https://www.mongodb.com/docs/manual/installation/

Mongo Delivery Method Conversion Example

The following examples use this product with type = 'Item', ID 'XYZ', and Name = 'XYZ Name.' The product is located in the family 'FamilyXYZ2.'

The STEP Mongo Adapter transforms this to the STEP JSON document:

```
{
_id : "EXA-5002-1001",
objectTypeID : "Chair",
parentID : "Chairs",
name : "Arm chair",
type : "product"
}
```

The id of the product is used as unique ObjectId in the MongoDB database, and therefore the STEPXML id field is transformed to _id. The UserTypeID XML attribute is converted to the objectTypeID field. The type field describes the kind of STEP object that is exported. For Basic STEP objects, the types are 'product', 'asset', 'classification', and 'entity.' Other STEP objects are described in the following topics.

Values

In this example, a product has two attributes:

- A single value attribute Color with the value 'Brown'
- A multi value attribute Height with the value = '43', the unit = 'inches', and the value = '120', with the unit = 'cm.'

When this is transformed to a STEP JSON document by the STEP MongoDB adapter, it results in the following:

```
{
  _id : "EXA-5002-1001",
  objectTypeID : "Chair",
  parentID : "Chairs",
  name : "Arm chair",
  type : "product"
  values : {
    Color : "Brown",
    Height : [ "43 inches", "120 cm"]
  }
}
```

The data displayed includes:

- Values of the product are stored as a subdocument with the name 'values.'
- The value of the single-valued attribute 'Color' is represented as a field-and-value pair: the ID of the attribute is the field, and the value is the value of the attribute represented as a string.
- The multi-valued attribute 'Height' is represented as a field-to-arrays of strings. Again, the ID of the attribute is the field, and the string array contains a string for each value.
- The name of the unit is appended to the attribute value with a space for all values with a unit. In the example above, '120 cm' is the concatenation of the value '120' and 'cm' from the unit named 'unece.unit.CM.'

Values of specification attributes that are inherited to the product are included in the JSON document of the product. For example, if the parent contains an attribute called Brand with the value 'Office chairs', the JSON document looks like the following:

```
{
  _id : "EXA-5002-1001",
  objectTypeID : "Chair",
  parentID : "Chairs",
  name : "Arm chair",
  type : "product"
  values : {
    color : "Brown",
    Height : [ "43 inches", "120 cm"],
    Brand: "Office chairs"
    }
}
```

LOV Value IDs

When values are included from an LOV that uses Value IDs (via the parameter 'Use Ids on values' = Yes on the LOV), the Value and the ValueID are stored as a subdocument with the name 'extValues.' For backward compatibility, the LOV value is also included in the 'value' subdocument, meaning this data is available twice, and can be extracted based on the need.

Note: If the LOV does not use Value IDs, the extValues subdocument is not written.

In this example, the LOV value of '4001' has a Value ID of '4001_CALYPSO.'

```
{
  _id : "EXA-5002-1001",
  objectTypeID : "Chair",
  parentID : "Chairs",
  name : "Arm chair",
  type : "product"
  values : {
    color : "Brown",
    Height : [ "43 inches", "120 cm"],
    Brand: "Office chairs"
    attribute_id: "4001",
},
    extValues: {
    attribute_id: {
        value: "4001",
        valueID: "4001_CALYPSO"
    }
}
```

The data displayed includes:

- The selected LOV value is stored within the subdocument with the name 'values' in the attribute_id field and again with the 'extValues' subdocument in the 'attribute_id: {value:' field.
- The value ID of the selected LOV value is represented once with the 'extValues' subdocument in the 'attribute_id: {valueID:' field.

References and Links

References and links are converted to STEP JSON in a subdocument called 'references.' This subdocument contains a field-and-value pair, where the field is the ID of the reference type, and the value is either a subdocument that defines the target of a reference, or an array of subdocuments that each define the target of a reference.

If the reference can only reference one target for the same source, the value is a subdocument. If the reference can reference more targets for the same source, the value is an array.

In this example, the product uses a 'Primary Image' asset reference to reference the image for the product. The asset has the ID 'Image_EXA-5002-1001.' The STEP JSON then exports as:

```
{
  _id : "EXA-5002-1001",
  objectTypeID : "Chair",
  parentID : "Chairs",
  name : "Arm chair",
  type : "product"
  values : {
    color : "Brown",
```

```
    Height : [ "43 inches", "120 cm"],
  Brand: "Office chairs"
  }
  references : {
    Primary Image : { targetID : "Image-EXA-5002-1001" }
  }
}
```

To find the asset for the reference, it is necessary to find the reference type 'Primary Image', which is stored in the raw collection 'referenceType.' The target type of the reference comes from the reference type.

The JSON of the reference target is found by searching the raw collection given by the reference type of an object with 'targetID.'

In the example above, the target type of the 'Primary Image' reference type is 'asset.' By looking in the 'asset' raw collection, the asset given by Image-EXE-5002-1001 is found.

Adding the reference 'Secondary Image', which can reference more targets such as ('Image1' and 'Image2') to the product, results in the following STEP JSON:

```
{
  _id : "EXA-5002-1001",
  objectTypeID : "Chair",
  parentID : "Chairs",
  name : "Arm chair",
  type : "product"
  values : {
    color : "Brown",
    Height : [ "43 inches", "120 cm"],
  Brand: "Office chairs"
}
references : {
  Primary Image : { targetID : "Image-EXA-5002-1001" },
    Secondary Image : [ { targetID : "Image1" }, { targetID : "Image2" } ]
  }
}
```

If the reference contains a meta data attribute, the values of the meta data attribute are added as a subdocument to the reference subdocument that contains the value of the attribute:

```
{
  _id : "EXA-5002-1001",
  objectTypeID : "Chair",
  parentID : "Chairs",
  name : "Arm chair",
  type : "product"
  values : {
    color : "Brown",
    Height : [ "43 inches", "120 cm"],
```

```
  Brand: "Office chairs"
}
  references : {
    Primary Image : { targetID : "Image-EXA-5002-1001",
                      values: { ShowOnWeb : "true" },
                     },
      Secondary Image : [ { targetID : "Image1" }, { targetID : "Image2" } ]
  }
}
```

The reference type 'Primary Image' has a meta data attribute called 'ShowOnWeb.' The attribute is set to 'true' for the reference from the product to the asset 'PrimaryAsset.'

Classification to product links are special because they are owned by either the product or the classification. This will always be the same for a specific classification to product link type. The JSON document only contains the classification-to-product links owned by the object. That is, the JSON document for a product only contains classification-to-product links owned by the product.

Attribute Types

Exporting an attribute type with the ID 'AttributeTypeID', and the name 'Name' linked in two parent attribute groups 'Parent1' and 'Parent2' results in the following JSON:

```
{
_id : "AttributeTypeID",
parentID : [ "Parent1", "Parent2"],
name : "Name",
listOfValuesID : "ListOfValuesID",
validUnitIDs : ["unece.unit.MMT", "unece.unit.CMT"],
type : "attribute"
}
```

Reference types are stored in the raw collection 'attribute.' To export reference types, verify that 'Include Attribute Definitions' is set to at least 'Selected' in the Process Engine configuration.

Data Containers

For both entities and products, data containers associated with these objects will be included, if included in the export configuration. A 'dataContainers' property will contain all of the associated attributes in this data container:

```
{
"_id":"CommonPlaceBook",
"objectTypeID":"Product user-type root",
"dataContainers":{
  "DC-2127822":{
    "values":{
      "Color":"Red",
```

```
      "Height":"12 in",
      "Width":"9 in"
    },
    "extValues":{
      "Color":{
        "value":"Red"
      },
      "Height":{
        "unitID":"unece.unit.INH",
        "value":"12"
      },
      "Width":{
        "unitID":"unece.unit.INH",
        "value":"9"
      }
    }
  }
},
"name":"Robert Black's Commonplace Book",
"type":"product",
"parentID":"TopNode"
}
```

In this example, the attributes 'Color,' 'Height,' and 'Width' in the single valued Data Container DC-2127822 are all exported to JSON.

In case of a multivalued data container, the content will always be exported as an array:

```
"ContEmailDataContainer": [
  {
    "extValues": {
      "CalcContEmailDataContainer": {
        "value": "313bobby@stibosystems.net"
      },
      "ContEmail": {
        "value": "313bobby@stibosystems.net"
      }
    },
    "values": {
      "CalcContEmailDataContainer": "313bobby@stibosystems.net",
      "ContEmail": "313bobby@stibosystems.net"
    }
  }
],
```

The values in the data containers are within the square brackets.

Reference Types

Exporting a product-to-product reference type with the ID 'ReferenceTypeID' and the name 'ReferenceName' generates the following JSON:

```
{
_id : "ReferenceTypeID",
name : "ReferenceName",
type : "referenceType",
targetType : "product"
}
```

In this example, 'targetType' is the target type of the reference. Possible target types are 'product','asset', 'classification', and 'entity.'

Reference types are stored in the raw collection 'referenceType.' When you export reference types, verify that Configuration > Include Link, Reference and Object Types is set to at least 'Minimum' in the Process Engine configuration.

Asset Push Locations

Asset Push Locations are stored in the assetPushLocations subdocument, in the root of the Asset documents in the asset collection. It provides the relative path to different versions of the asset. The assetPushLocations subdocument contains field-and-value pairs, where the field name is the assetpush configuration id, and the value is the relative path that the image file has been pushed to. The relative path is relative to the AssetPushClients root folder, so to get the full path of the image, you have to prepend the path to the assetpush client root directory, as shown in the following example:

```
assetPushLocations : {
    AllAssets-approved : "AllAssets-approved/std.lang.all/73/15/7315.pdf",
    AllWebsiteImages-approved : "AllWebsiteImages-
approved/std.lang.all/73/15/7315.pdf",
    small : "small/std.lang.all/73/15/7315.jpg",
    large : "large/std.lang.all/73/15/7315.pdf"
}
```

Tables

Tables are defined on classifications or products. The table subdocument is stored in the root of the product and classification document, in the product and classification collection. The table subdocument contains field-and-value pairs, where the field name is the table Type name of the table on the product or classification.

A table is exported from STEP in a resolved and transformed state. That is, any transformations defined in STEP have already been applied to the table and the row and column layout has therefore been resolved.

Formatting is not applied to the table before it is exported. So, a web application is not bound by the STEP table formatting. However, if any formatting has been defined for the table in STEP, it is included in the export so that the website application can be configured to render the formatting on the website.

The table contains an array of column elements that contain the formatting meta data for each column and an array of rows. The row contains the formatting information for each row, plus an array of cells. Each cell has a number of formatting meta data attributes and a text field that contains the content of the table cell.

Table Formatting

Formatting meta data can be stored on table, column, row, and cell level, and is inherited downwards from table level to column level to row level to cell level.

If a column is specified to use an underlined text style, and a row is specified to be bold text style, the formatting information is accumulated so that the cell at the intersection of the row and column is rendered as both underlined and bold.

Formatting meta data defined at a lower level overrides meta data at a higher level. For example, if a table has a gray background color, and a column has a red background color, then cells in that row are rendered with a red background.

A simple way of formatting a table in HTML is to map the tableType, columnType, and rowType to element classes in HTML, and then use a CSS style sheet to define the table formatting, as shown in the following example.

```
tables : {
  Description Table : {
    columns :
      [ { columnType : "Description" }, { columnType : "Description" }, { columnType
: "Description" } ],
    rows : [
      { rowType : "Header",
        cells : [
          { ruleRight : "0.5 pt",
            text : "Header 1",
            verticalAlignment : "top",
            cellStoryDirection : "horizontal",
            backgroundColor : "Light Blue",
            column : "0",
            ruleBelow : "0.5 pt",
            ruleAbove : "0.5 pt",
          textStyle : "TableHeader—Description",
            ruleLeft : "0.5 pt" },
          { ruleRight : "0.5 pt",
            text : "Header 2",
            verticalAlignment : "top",
            cellStoryDirection : "horizontal",
            backgroundColor : "Light Blue",
            column : "1",
            ruleBelow : "0.5 pt",
            ruleAbove : "0.5 pt",
            textStyle : "TableHeader—Description",
```

```
            ruleLeft : "0.5 pt" }] },
      { rowType : "Header",
          cells : [
          { ruleRight : "0.5 pt",
            text : "Value 1",
            verticalAlignment : "top",
            cellStoryDirection : "horizontal",
            backgroundColor : "Light Blue",
            column : "0",
            ruleBelow : "0.5 pt",
            ruleAbove : "0.5 pt",
            textStyle : "TableHeader—Description",
            ruleLeft : "0.5 pt" },
          { ruleRight : "0.5 pt",
            text : "Value 2",
            verticalAlignment : "top",
            cellStoryDirection : "horizontal",
            backgroundColor : "Light Blue",
            column : "1",
            ruleBelow : "0.5 pt",
            ruleAbove : "0.5 pt",
            textStyle : "TableHeader—Description",
            ruleLeft : "0.5 pt" }] }
] } }
```

Attribute Links

AttributeLinks define the validity of attributes for products linked into the product and classification hierarchies. The attributeLinks subdocument is stored at the root of the product and classification document structure. The attribute links can themselves contain a values subdocument that defines the meta data values attached to the attributeLink.

The attributeLinks subdocument contains field value pairs, where the field name is the attributeID, and the value is a subdocument that contains any meta data values related to the attribute link. The following is an example of an attributeLinks subdocument:

```
attributeLinks : { Voltage range : { }
                   Rotary address switches : { },
                   Default address : { },
                   Power consumption : { }
                   }
```

Data Container Type Links

The validity for Data Container Types for products can be restricted to products in certain product and/or classification hierarchies just like attributes can.

If the classification or product root to which a data container is restricted is exported, then the following JSON subdocument is returned:

```
"dataContainerTypeLinks": [
  "ProdDC"
]
```

The following example is an export of the product root node called 'TopNodeWDC.' In this case, both the 'SpecificationAttribute' attribute is restricted to this root node, and the 'ProdDC' data container is restricted to this root node. This export returns an attributeLinks subdocument (as described in the 'Attribute Links' section) and a new dataContainerTypeLinks subdocument:

```
{
  "_id": "TopNodeWDC",
  "attributeLinks": {
    "SpecificationAttribute": {}
  },
  "dataContainerTypeLinks": [
    "ProdDC"
  ],
  ...
}
```

Data Containers Types

Data Container types collection contains the definition of the data containers. These collections may include metadata of the data container. The JSON will look similar to the following:

```
{
  "_id":"DC-2127822",
  "name":"Product Description",
  "type":"datacontainertype",
  "multiValued":"false"
}
{
  "_id": "ContPhoneDataContainer",
  "extValues": {
    "DC Long Description": {
      "value": "This DC is used for Phone numbers"
    }
  },
  "multiValued": "true",
  "name": "Phone",
  "type": "datacontainertype",
  "values": {
    "DC Long Description": "This DC is used for Phone numbers"
  }
}
```

Unit

Exporting an unit with the ID 'unece.unit.CMT', and the name 'cm' results in the following JSON:

```
{
"_id" : "unece.unit.CMT",
"values" : {
"4941" : "centimeter"
},
"conversionToBase" : {
    "factor" : "100",
    "unitID" : "unece.unit.MTR",
    "offset" : "0"
},
"name" : "cm",
"type" : "unit"
}
```

The unit document contains the 'conversionToBase ' object that can be used for converting values from a unit to its base unit. The 'conversionToBase ' object of the above example shows how to convert values of the unit centimeters to meters.

Units are stored in the raw collection 'unit.' To export units, verify that Configuration > Include Units is set to at least 'Minimum' in the Process Engine configuration.

List Of Values

Exporting a ListOfValues object with the ID 'LOVID', and the name 'List of values' results in the following JSON:

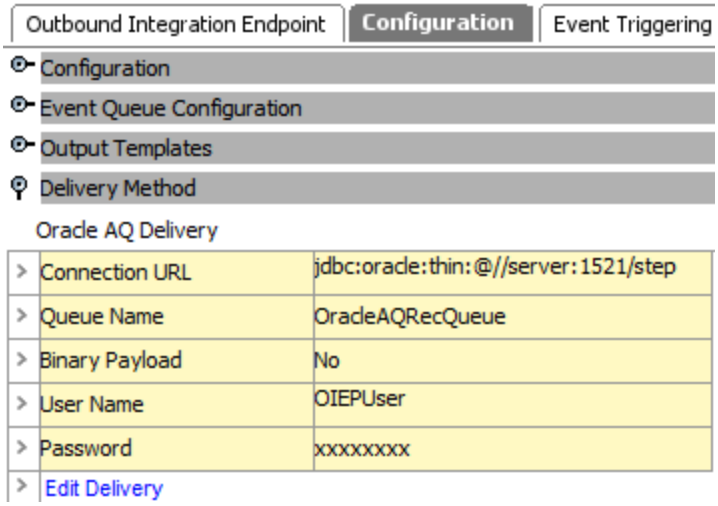
```
{
"_id" : "LOVID",
"values" : {
"6823" : "This is a metadata value of the lov'"
},
"name" : "List of values",
"validUnitIDs" : [
"unece.unit.CMT",
"unece.unit.MTR"
],
"type" : "listofvalues"
}
```

List Of Values are stored in the raw collection 'listofvalues.' To export List Of Values, verify that 'Include List Of Values Definitions' is set to at least 'Minimum' in the Process Engine configuration.

Oracle AQ Delivery Method

This delivery option is only available in OIEPs.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



Prerequisites

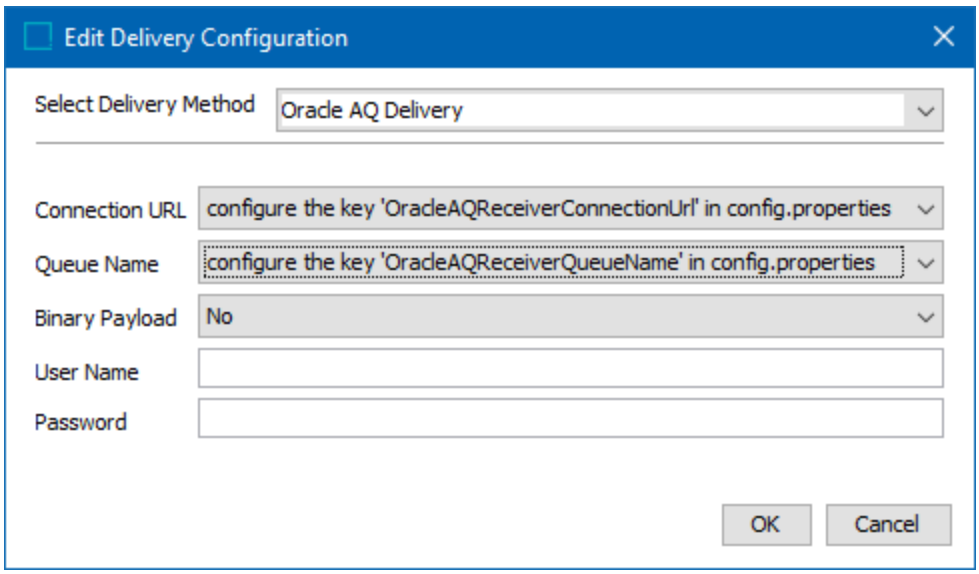
Changes to the properties file, outlined below, are implemented when the server is restarted.

1. Prior to configuration, clicking the **Connection URL** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **OracleAQReceiverConnectionURL** property.
2. Prior to configuration, clicking the **Queue name** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **OracleAQReceiverQueueName** property.

Configuration

For information on a parameter, hover over the parameter field to display help text.

1. In the **Select Delivery Method** list, choose **Oracle AQ Delivery**.



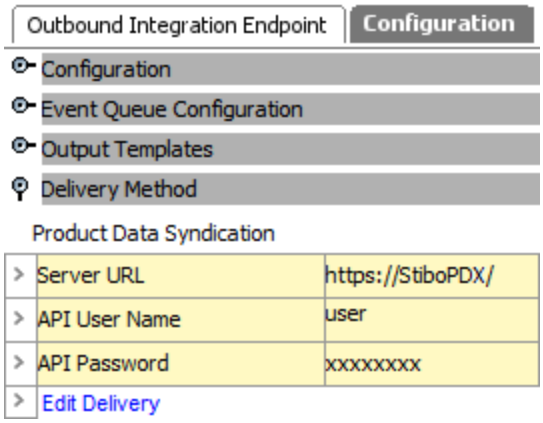
2. In the **Connection URL** list, select a URL that points to Oracle AQ.
3. In the **Queue name** list, select an Oracle AQ queue name.
4. For **Binary Payload**, select Yes or No. 'Yes' is selected for non-text files.
5. In the **User Name** field, enter the user name that will be used to log on to Oracle
6. In the **Password** field, enter the password that will be used to log on to Oracle.
7. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

Product Data Exchange 2 Delivery Method

The Product Data Exchange 2 delivery method is only available in OIEPs. STEP sends data to the Product Data Exchange (PDX) platform via the default PDX Outbound Integration Endpoint and the API.

For additional information on PDX, refer to the **Product Data Exchange** topic within the **Data Integration** documentation, or contact your Stibo Systems representative.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.

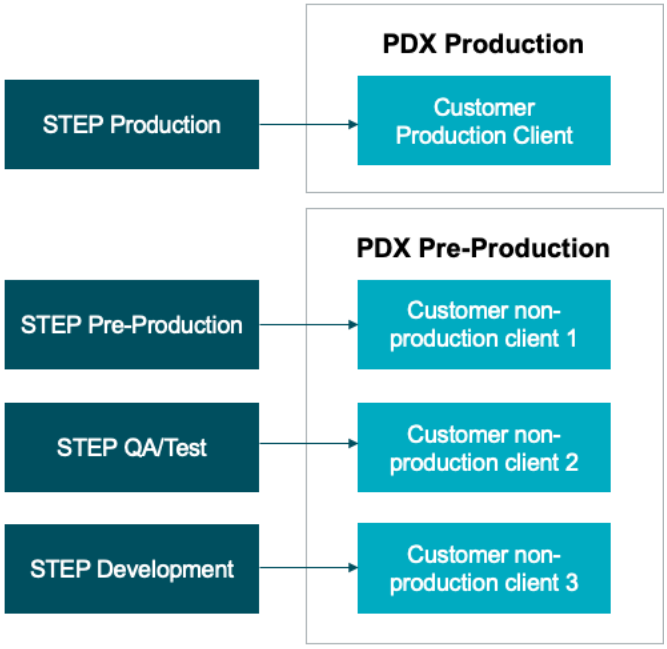


Prerequisites

The server-side setting of the sharedconfig.properties PDX.Url determines the PDX environments eligible in the OIEP configuration of the delivery method. Two PDX environments are relevant to this setting:

- PDX Production (https://api.pdx.stibosystems.com)
- PDX Pre-production (https://api.pdx-preprod.stibosystems.com)

General setup will include one account on the Production environment, which processes and syndicates live data, while multiple non-production accounts / clients may exist on the Pre-production environment. In the image below, STEP environments are in the column on the left, and those on the right represent PDX environments with PDX accounts.



Note: Changes to the properties file, outlined below, are implemented when the server is restarted.

The sharedconfig.properties on the STEP Production environment specify the PDX Production environment (containing the customer's production account / client) as the valid target of the PDX OIEP configured, as seen in the example below:

```
PDX.Url=1=https://api.pdx.stibosystems.com,2=https://api.pdx-preprod.stibosystems.com
```

To ensure product information maintained in all the contexts selected for syndication in the PDX OIEP setup can be transferred, the following property should be set to 'false':

```
PDXDelivery2.LocaleChecking=false
```

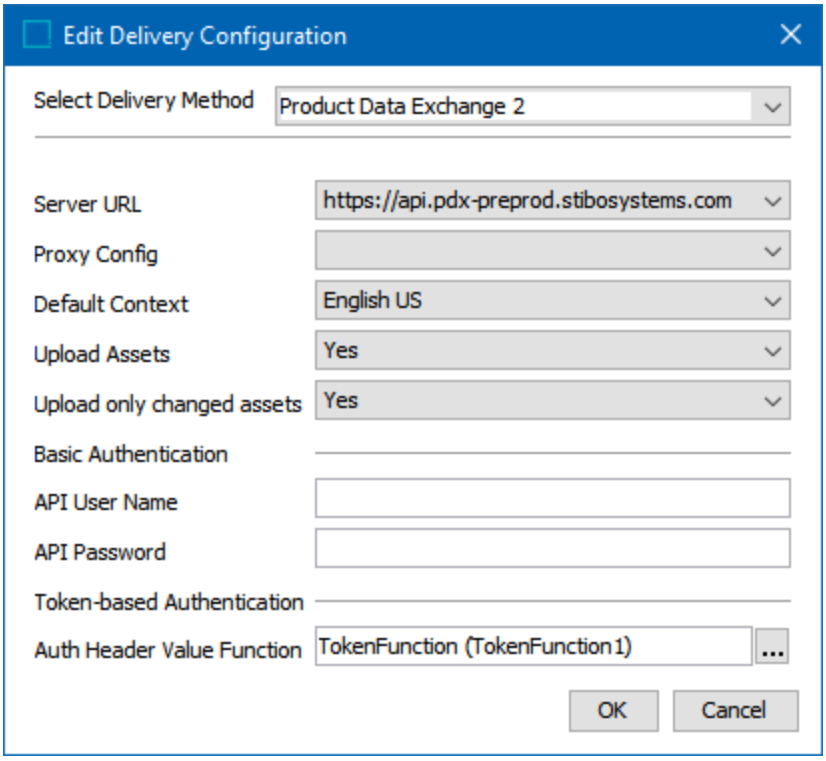
1. Determine the API user name and password. This will be required on the Edit Delivery Configuration dialog.
2. If required, provide a selection for the **Proxy Config** parameter. The configurations that appear in this dropdown are populated from properties added into the sharedconfig.properties file using the dynamic property 'Http.ProxyConfiguration.[name],' which has its own property for Host, Port, User, and Password.
3. Perform additional configuration required in the OIEP as laid out in the **Setting Up the PDX OIEP** topic in the **Data Integration** documentation.

Configuration

For information on a parameter, hover over the parameter field to display help text.

1. On the **Configuration** tab of the OIEP, in the **Delivery Method** flipper, click the **Edit Delivery** link to display the 'Edit Delivery Configuration' dialog.

2. For the **Select Delivery Method** parameter, choose **Product Data Exchange 2**.



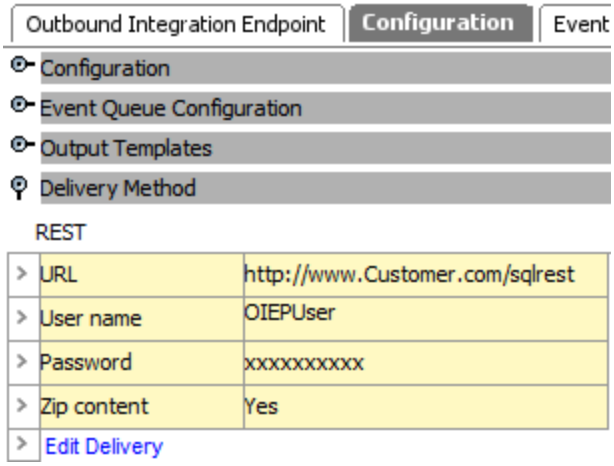
3. **Server URL** - Select the URL from the dropdown, if not preselected.
4. **API User Name** - Provide the user name for the user that has access to the API.
5. **API Password** - Provide the password for the user that has access to the API.
6. **Proxy Config** - Select the desired HTTP proxy configuration if the delivery connection must first pass through a proxy server with its own login requirement.
7. **Default Context** - Select the context from the dropdown menu that will serve as the default context. The data in this context will be used to populate the default language layer in PDX.
8. **Upload only changed assets**- Set the 'Upload only changed assets' setting to 'Yes,' if you only want to include referenced asset objects where either attribute values or the asset content has changed.
9. Click the **OK** button to save the delivery method.

Note: For more information regarding HTTP proxy configurations, refer to the topic **HTTP Proxy Configurations** in the **Data Exchange** documentation.

REST Delivery Method

The REST Delivery Method delivers a call-back URL to the REST service and does not include STEP data. The data can be fetched from the call-back URL by the receiving REST service. This delivery option is only available in OIEPs.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



Prerequisites

Changes to the properties file, outlined below, are implemented when the server is restarted.

Prior to configuration, clicking the **URL** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **RESTDeliveryURL** property. If necessary, use a comma to separate multiple URLs.

This should be a URL to a REST POST method, for example, http://server/rest_URL. A URL where the result of the endpoint can be fetched is returned in the POST call.

The following is an example of a complete property entry for two systems named 'qa' and 'stage,' as well as their URLs 'http://step-qa' and 'http://step-stage':

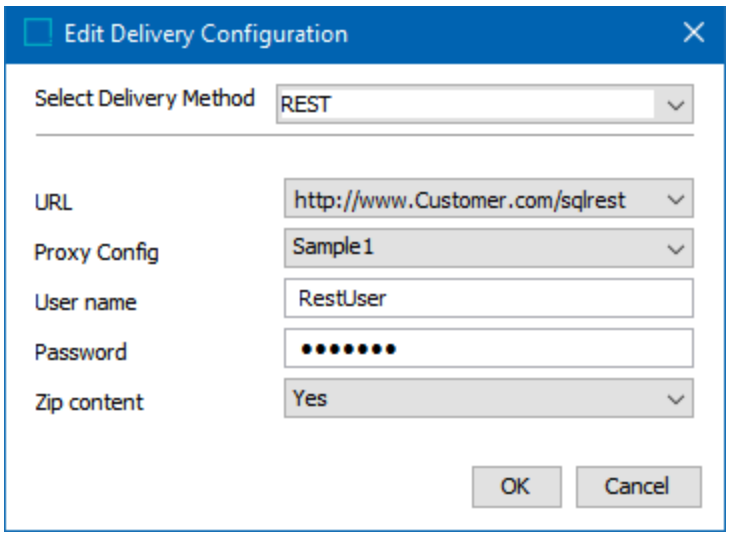
```
RESTDeliveryURL=qa=http://step-qa,stage=http://step-stage
```

Important: For information about how to use the REST API to upload files to REST, access the **STEP API Documentation**, available at [system]/sdk or from the Start Page.

Configuration

For information on a parameter, hover over the parameter field to display help text.

1. For **Select Delivery Method**, choose **REST**.



2. In **URL**, select the URL that points to the REST endpoint where you would like to receive the delivered data.
3. In **Proxy Config**, select the desired HTTP proxy configuration if the delivery connection must first pass through a proxy server with its own login requirement.
4. In **User Name**, enter the user name that will be used to log on to the REST endpoint.
5. In **Password**, enter the password that will be used to log on to the REST endpoint.
6. In **Zip content**, specify whether to zip the contents before upload.
7. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

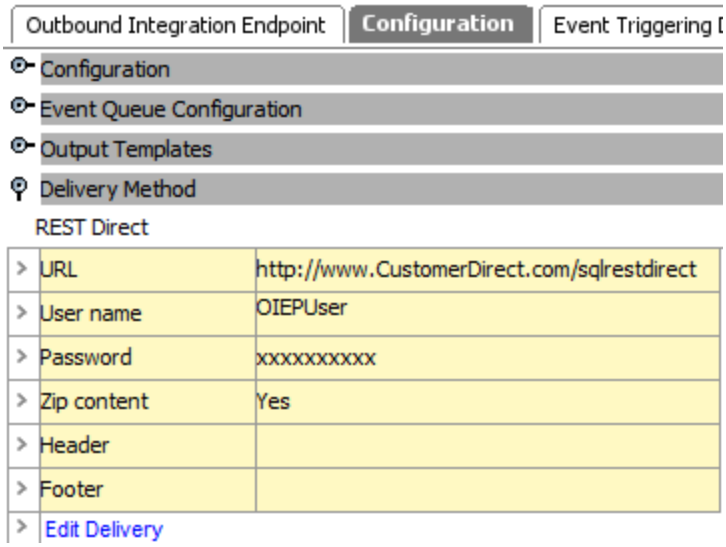
Note: For more information regarding HTTP proxy configurations, refer to the topic **HTTP Proxy Configurations** in the **Data Exchange** documentation.

REST Direct Delivery Method

The REST Direct delivery method differs from the standard REST delivery method in that the data is delivered directly to the REST service and no call-back URL is required.

This delivery option is only available in OIEPs and if the delivery fails the OIEP is disabled. There is no resilience handling.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



To use mTLS or TLS with this type of endpoint, refer to the **Mutual Transport Layer Security** topic.

Prerequisites

Changes to the properties file, outlined below, are implemented when the server is restarted.

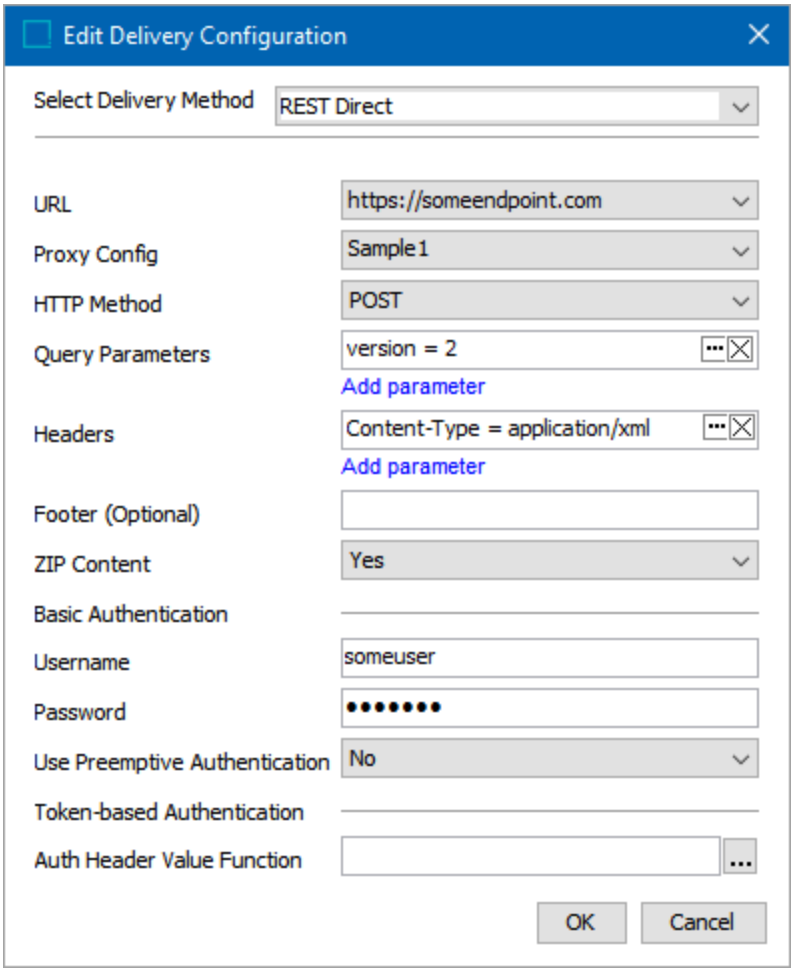
1. The REST Direct delivery method reads the outgoing file into memory before sending. To handle the size of the outgoing file and prevent time-outs and rejections you must scale the heap size.
2. Prior to configuration, clicking the **URL** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **RestDirectDeliveryURL** property. If necessary, use a comma to separate multiple URLs. The image below shows the result of using the following example for a property entry involving two systems:

```
RestDirectDeliveryURL = 1=http://myfirstendpoint, 2=http://mysecondendpoint
```

Configuration

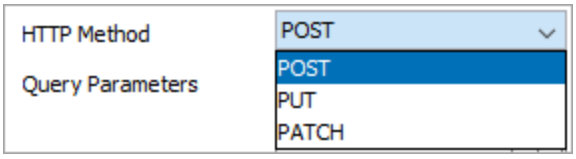
For information on a parameter, hover over the parameter field to display help text.

1. For **Select Delivery Method**, choose **REST Direct**.

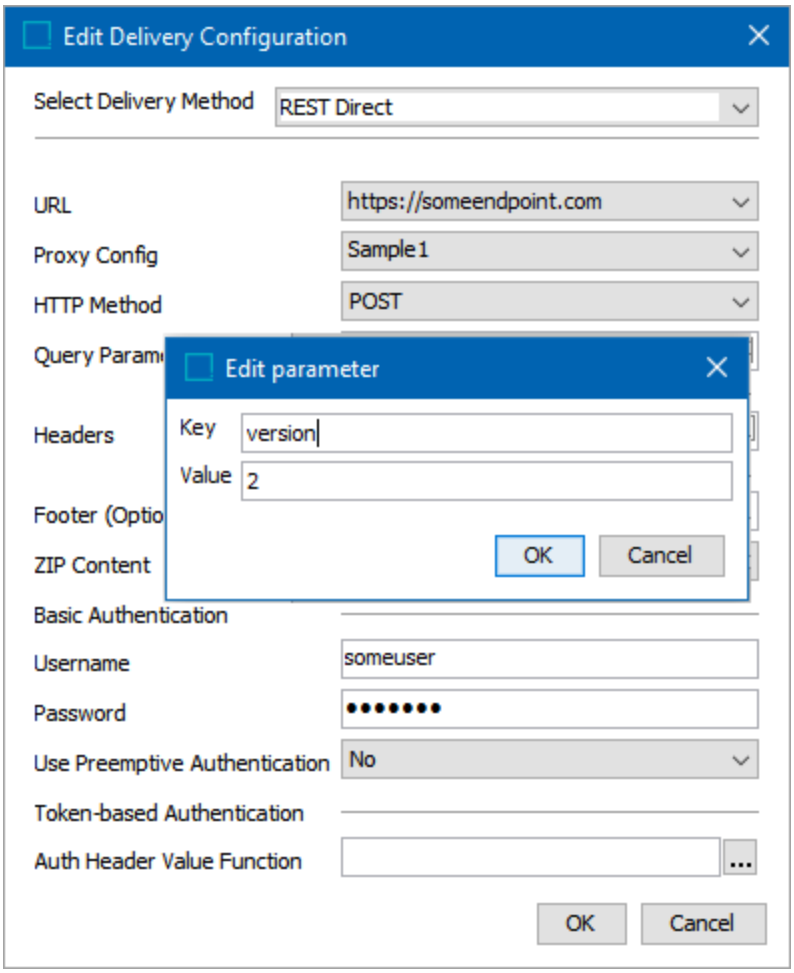


2. In **URL**, select the URL that points to the REST endpoint where the delivered data should be received.
3. In **Proxy Config**, select the desired HTTP proxy configuration if the delivery connection must first pass through a proxy server with its own login requirement.

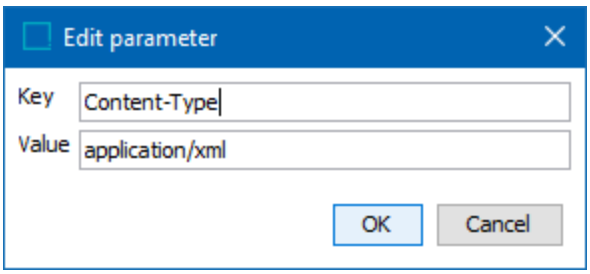
4. In the **HTTP Method** option, use the dropdown to select a POST, PUT, or PATCH method.



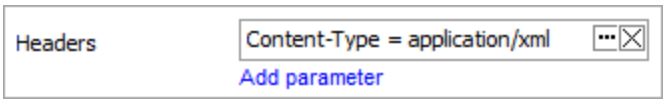
If required, in the **Query Parameters** field add any query parameters that are required for a successful REST transaction. Click on the **Add parameter** link to add the first query parameter. Once a query parameter is added, click the ellipsis button (...) to add additional query parameters. To remove query parameters, click on the 'X.'



5. For **Headers**, click the **Add Parameter** link and add a key and a value. This is part of the HTTP network protocol.



6. On the Add parameter dialog, click the **OK** button and the Header is displayed in the field. If multiple headers are needed, use the **Add Parameter** link to add each additional key and value pair.

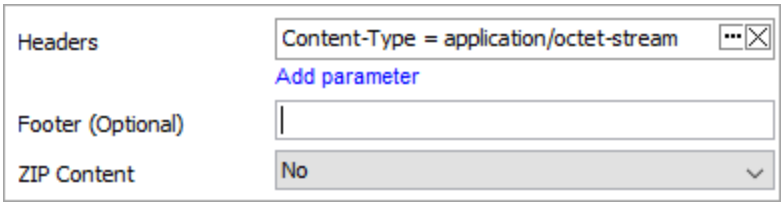


Note: Once a header is displayed, click the ellipsis button (...) to edit it or the X button to remove it.

For more information about headers, refer to the 'Additional Request Information' section at the end of this topic.

7. In **Footer**, add data required for the recipient to verify that the full message was received. This is part of the HTTP network protocol but not required.

   The footer can be used to mark the end of a multi-part REST call (that is a REST call containing the payload split in more packages). The footer could also contain a checksum that the receiver can use to detect if the payload in a multi-part message has been changed by the middleware.



8. In **Zip Content**, specify whether to zip the contents before upload.
9. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

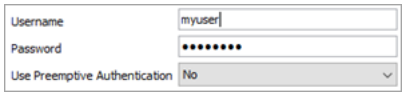
Note: For more information regarding HTTP proxy configurations, refer to the topic **HTTP Proxy Configurations** in the **Data Exchange** documentation.

Authentication

The REST Direct delivery plugin supports both basic authentication and token-based authentication.

Basic Authentication

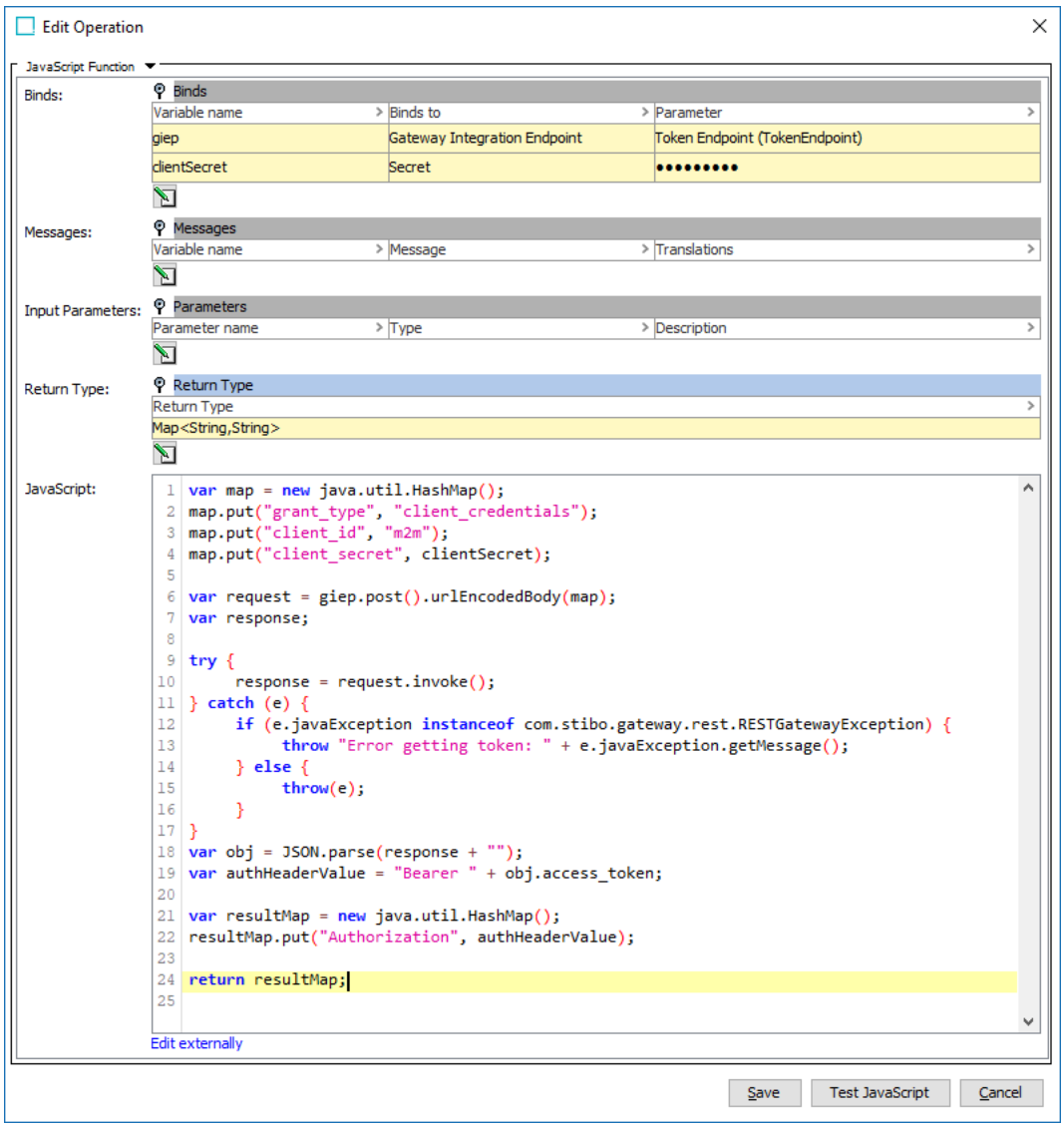
For basic authentication, enter the username and password and specify whether or not the plugin should use preemptive authentication. With preemptive authentication enabled, the basic authentication value for the Authorization header will be sent with the first request to the external service, instead of only sending the value after having received a basic authentication challenge from the service.



Token-Based Authentication

With the token-based authentication option, the responsibility for producing any required request headers is delegated to a business function. The business function must be configured to not expect any input and must produce a Map<String, String>. Each map entry is a header that will be sent with the request to the service.

The following screenshot shows the editor for a compatible business function that retrieves a token using the OAuth 2.0 client credentials flow and passes the token back to the delivery plugin as a value for the Authorization header.



The REST Direct delivery plugin will automatically call the business function when a new token is required.

Note: It is strongly discouraged to configure both basic authentication and token-based authentication. This configuration combination is not supported.

Additional Request Information

By default, the REST Direct delivery option sends multipart/form-data POST requests with the exported file available in the part named 'file' with Content-Type application/octet-stream. The example below shows the properties for such a request:

Headers

Accept-Encoding = gzip,deflate

Connection = close

Content-Length = 1354

Content-Type = multipart/form-data; boundary=JN_qruUpDaHqm9BgW_b4-adAHDftQbjuvDI

Body (Metadata)

Content-Disposition: form-data; name="file"; filename="result.zip"

Content-Type: application/octet-stream

Via the UI it is possible to change the HTTP method and also to overwrite the Content-Type header. For instance, this header could be set to 'application/xml' for a STEPXML export (unzipped) thereby overwriting the default 'multipart/form-data' value making the request properties be:

Headers

Accept-Encoding: gzip,deflate

Connection: close

Content-Length: 1354

Content-Type: application/xml

Body <The exported XML>Additional Request Information

By default, the REST Direct delivery option sends multipart/form-data POST requests with the exported file available in the part named 'file' with Content-Type application/octet-stream. The example below shows the properties for such a request:

Headers

Accept-Encoding = gzip,deflate

Connection = close

Content-Length = 1354

Content-Type = multipart/form-data; boundary=JN_qruUpDaHqm9BgW_b4-adAHDftQbjuvDI

Body (Metadata)

Content-Disposition: form-data; name="file"; filename="result.zip"

Content-Type: application/octet-stream

Via the UI it is possible to change the HTTP method and also to overwrite the Content-Type header. For instance, this header could be set to 'application/xml' for a STEPXML export (unzipped) thereby overwriting the default 'multipart/form-data' value making the request properties be:

Headers

Accept-Encoding: gzip,deflate

Connection: close

Content-Length: 1354

Content-Type: application/xml

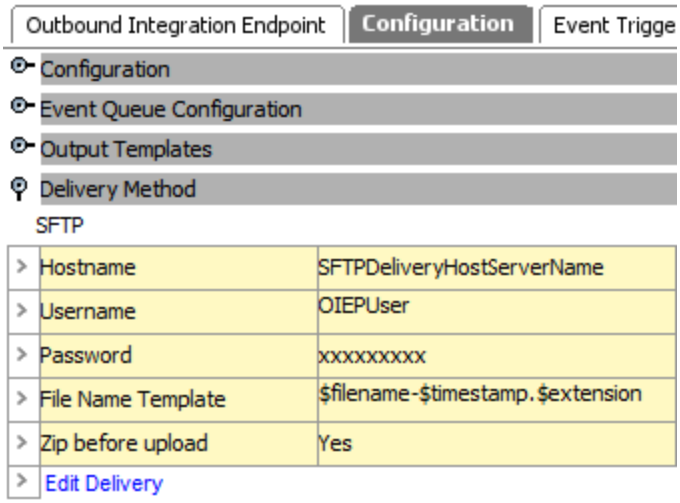
Body <The exported XML>

SFTP Delivery Method

The preferred method for file transfer protocol (FTP) delivery is the SFTP secure delivery method. The SFTP delivery method allows an exported file to be delivered to an external system and is often used when the output files are large or when a different or remote system is in use.

For information on the OIEP FTP delivery method, refer to the **FTP Delivery Method** topic.

In an OIEP, the delivery method is displayed on the Configuration tab of the editor in the Delivery Method flipper.



This delivery method is also available in Export Manager as defined in the **SFTP Delivery Method** topic.

Prerequisites

Changes to the properties file, outlined below, are implemented when the server is restarted.

Prior to configuration, clicking the **Host name** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **FTPDeliveryHostName** property. The required format of the property is (square brackets not included): FTPDeliveryHostName=1=[host1],2=[host2].

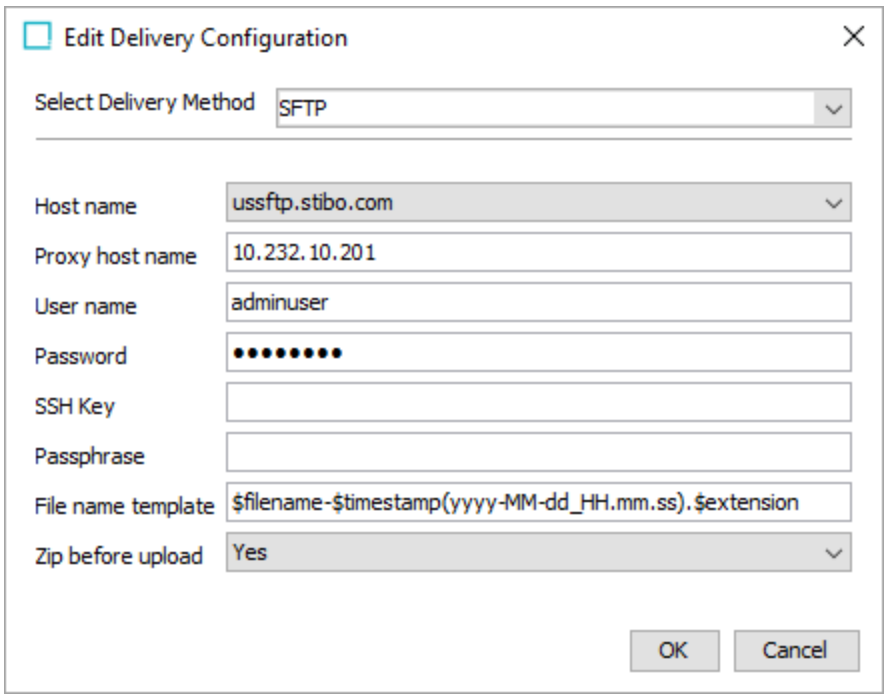
Using the host name shown in the image, the entry in the properties file would be:

```
FTPDeliveryHostName=1=ussftp.stibo.com
```

Configuration

For information on a parameter, hover over the parameter field to display help text.

1. Click the **Select Delivery Method** list to display the dropdown and select **SFTP**.



2. In **Host name**, select the host name of the SFTP server to be used for the delivery.
3. In **Proxy host name**, add the host name to be used for the server proxy. This field is optional.
4. In **User name**, enter the user name that has access to log on to the FTP server.
5. In **Password**, enter the password that will be used to log on to the FTP server. If using the SSH Key, then leave this field blank.
6. In **SSH Key**, enter the full path to the Secure Shell (SSH) key file, if using. If you filled in the Password field in step 5, then leave this field blank.

> **Note:** This SSH key must be an **RSA** private key in the **OpenSSH PEM** format. Other SSH key types or those generated in the new OpenSSH format (the default used in OpenSSH 7.8+) are not currently supported and will result in an **SFTPException: invalid privatekey** error.

7. In **Passphrase**, enter the passphrase that accompanies the SSH key entered, if applicable. If the SSH key does not have a passphrase or you are not using an SSH key, then leave this field blank.
8. In **File name template**, to create the file name used for the output, enter template variables (identified with a $), static text, or use a combination of both. The default template is $filename-$timestamp(yyyy-MM-dd_HH.mm.ss).$extension. Depending on the settings of the configuration, the default template outputs a file name similar to 'GenericXML-2020-07-30_14.07.44.xml.' Each variable is described below:

> **Note:** The 'Zip before upload' parameter also has an impact on the file name.

- **$filename** For event-based OIEPs, this variable is replaced with text to indicate the format of the delivered file as specified in the Output Templates flipper, except for STEPXML when the first and last Event IDs are used. For example, the output file name could be 'csv-2020-07-30_14.09.40.csv' or '1804038-1804038.xml' to indicate that STEPXML was used for a single event.
- **$timestamp**: This variable is replaced with the time in year-day-month-hour-minutes-seconds-milliseconds, excluding the dashes. For example, 20161511094709625 is used for an export on 15 NOV 2016 at 9:47:09:625 AM.

  Adding a definition for the arrangement of the date / time elements provides a way to generate a more readable date. Static characters can be included to separate the timestamp elements and individual elements can be ordered as required, as shown below.

  Note: To revert the **$timestamp** variable so that it uses the legacy definition (milliseconds between January 1, 1970, and the time when the file is uploaded), contact Stibo Systems Support for more information.

  $timestamp(ddMMyyyy): The timestamp is modified to display day-month-year (excluding dashes) and is output as '15112016' for 15 NOV 2016.

  $timestamp(yyyy-MM-dd_HH.mm.ss): The timestamp is modified to display year-month-day_hour.minutes.seconds (including the dashes, underscore, and periods) and is output as '2016-11-15_09.32.43' for 2016 NOV 15 at 9:32.43 AM.

- **$extension** This variable is replaced with the extension of the output file based on the selected format in the Output Templates flipper. For XML-based formats, the output file is set with the .XML extension, while Excel files use .XLS, and comma-separated values use .CSV.

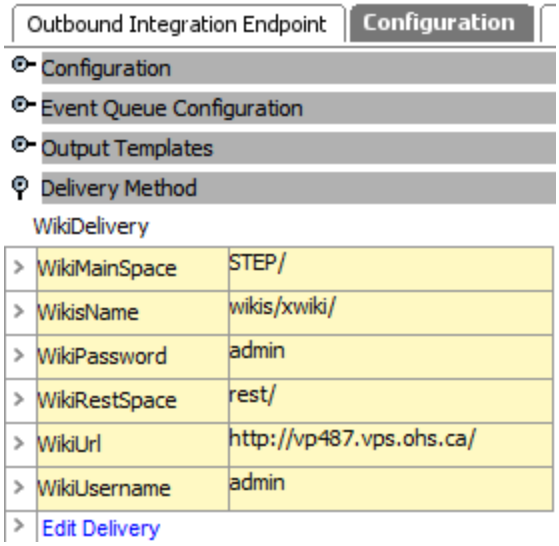
Note: The File Name Template does not support conversions of file formats and can only be used to deliver files in the format specified on the Configuration tab > Output Templates flipper of the outbound integration endpoint.

9. In **Zip before upload**, select an option from the dropdown to specify if the output file should be delivered in a .ZIP (compressed) file format.
   - **Yes** uses 'result_0' before the timestamp variable, and then the extension ZIP. For example, a zipped STEPXML output with the default File Name Template exported on 30 JUL 2020 results in an output .ZIP file named 'result_0-2020-07-30_14.07.44.zip.' The contents of the ZIP file would follow the $filename variable applied in the 'File name template' parameter.
   - **No** uses the 'File name template' for the file name along with the appropriate extension for the selected data format.
10. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

Wiki Delivery Method

The Wiki Delivery Method works with an OIEP configured to deliver metadata to a specified XWiki advanced open source enterprise wiki system. The contents of the wiki can be accessed by clicking a 'Go to Wiki' link from either Web UI or workbench.

The Wiki Metadata functionality is a licensed component and requires a separate installation recipe. For details and the latest recipe, contact your Stibo Systems representative.



Prerequisites

Changes to the properties file, outlined below, are implemented when the server is restarted.

1. Review the **Wiki Metadata** topic for details on the functionality and setup.
2. Prior to configuration, clicking the **Base URL** dropdown parameter displays the required property name. Provide a Base URL selection for the dropdown parameter via the sharedconfig.properties file using the case-sensitive **Wiki.WikiUrl** property. A protocol (http:// or https://) and an ending slash (/) are mandatory. The following is an example of a property entry for two wikis.

   ```
   Wiki.WikiUrl=1=http://vp487.vps.ohs.ca/,2=https://127.0.0.1:8080/
   ```

   If only one URL is required, in the example above, eliminate the ',2=https://127.0.0.1:8080/' text.

   Note: The format for multiple wikis is 1=url1,2=url2,3=url3, where each entry is incremented and unique.

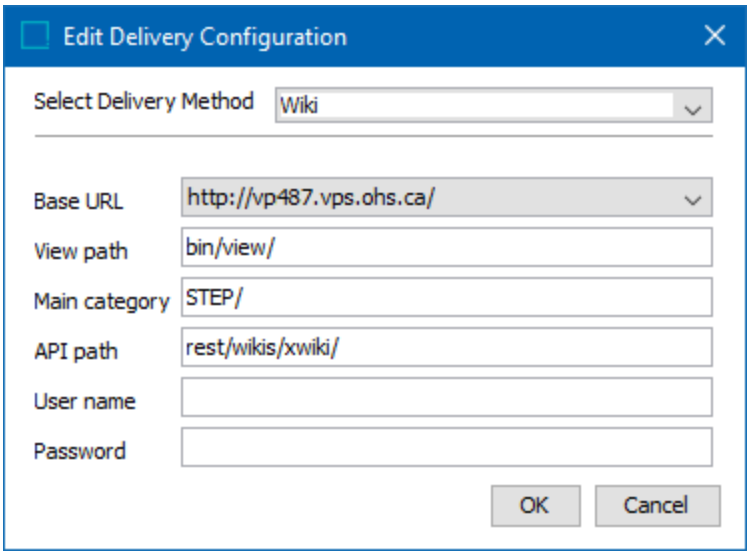
Configuration

For information on a parameter, hover over the parameter field to display help text.

After completing the prerequisite steps, edit the delivery method of the OIEP created during the installation. Use the following steps to configure the OIEP to use the Wiki delivery option.

Important: For the OIEP created automatically upon restarting the server after installing the STEP wiki metadata recipe, most of the following parameters are pre-populated and the default settings should not be changed. Review the parameter details below for more information.

1. For **Select Delivery Method**, choose **Wiki**.



2. In **Base URL**, select the URL that points to the wiki that will be used for this STEP system. Refer to the Prerequisites section for information on setup.
3. In **View Path**, enter the URL elements between the Base URL and the wiki pages, or main category if defined. This is used for the presentation of wiki pages and allows each user to watch pages they select. The slash at the end is mandatory. As shown in the image above, 'bin/view/' is the view path for the following URL: http://vp487.vps.ohs.ca/**bin/view/**STEP/attribute/
4. In **Main category**, enter a name for the main category path. This allows you to organize the metadata pages and can be multiple levels by adding a slash, '/,' between each one. The slash at the end is mandatory. As shown in the image above, 'STEP/' is the main category for the following URL: http://vp487.vps.ohs.ca/bin/view/**STEP/**attribute/
5. In **API path**, specify the path for the wiki REST API, which is used to make modifications to the wiki pages. The slash at the end is mandatory.
6. In **User name**, enter the user with permission to call the REST API and who can make modifications to it.
7. In **Password**, enter the password for the user name.
8. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

Running an Outbound Integration Endpoint

After configuring an outbound integration endpoint (OIEP), it must be enabled before it can run on schedule or be invoked manually.

The following colored icons are used to indicate the status of the integration endpoint. The endpoint status is displayed in the System Setup OIEP hierarchy and on the endpoint editor in the Description flipper on the Endpoint Status parameter.

Icon	Endpoint Status	Description
	Enabled	The integration endpoint is active, connected, and running.
	Disabled	The integration endpoint has stopped, and no data is being exported. Newly created endpoints, and endpoints that have been manually disabled, will have this status.
	Failed (retrying)	The OIEP is running and is attempting to recover from a connection error. After retrying in the original background process (BGP) for approximately one minute without success, the wait time begins, and the process goes into a Waiting state before the next attempt to retry processing. When the wait time expires, the BGP state changes to running and the connection is reattempted.
	Failed	An error occurred during processing that caused the integration endpoint background process to fail. No data will be exported until the endpoint has been manually resumed / reactivated. For more information, refer to **Handling Failed OIEP Background Processes** documentation.

Enable and Invoke an Outbound Integration Endpoint

Both event-based and select objects OIEPs must be enabled and invoked before they can run. Additionally, event-based OIEP must have the proper setting for queue status.

For example, if the external system is down and cannot accept data, you can enable the endpoint but do not schedule or invoke the endpoint until the external system is ready to consume the data. In this way, events are collected from STEP and data is then pushed to the external system when it is available.

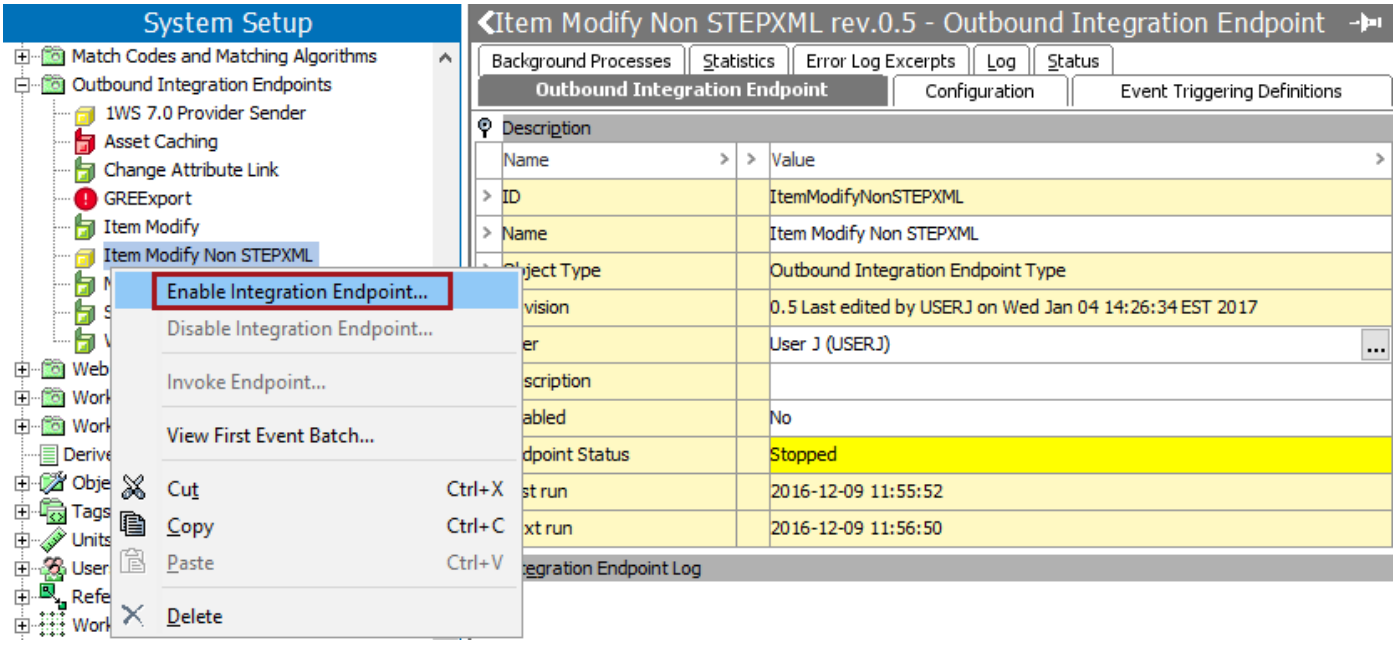
Prerequisites for Event-Based OIEPs

For an event-based OIEP, verify the Queue Status is set to Read Events or Discard Events, as appropriate. For information about how the Endpoint Status and Queue Status settings work together on event-based endpoints, refer to the **Event-Based OIEP Status and Queue Status** documentation.

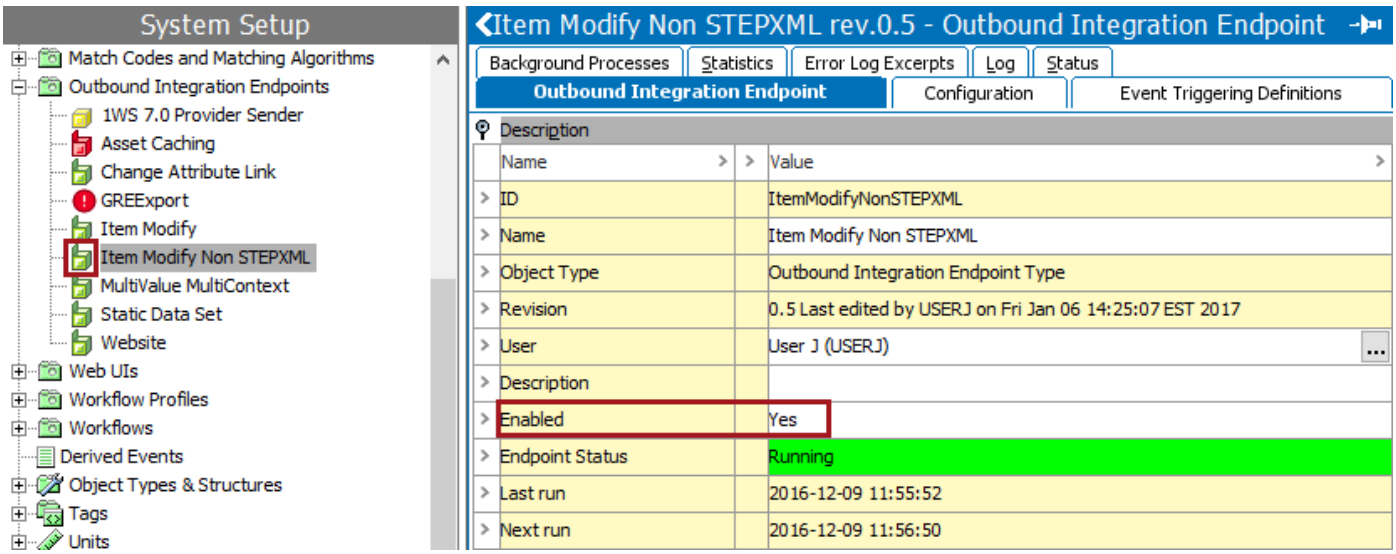
For information about setting up an event processor, refer to **Event Processors** in the **System Setup** documentation.

Required Setup for All OIEPs

1. On System Setup expand the Outbound Integrations Endpoints node to display all existing OIEPs.
2. Select a disabled endpoint ( ) or a failed endpoint ( ), right-click, and select **Enable Integration Endpoint** from the menu.

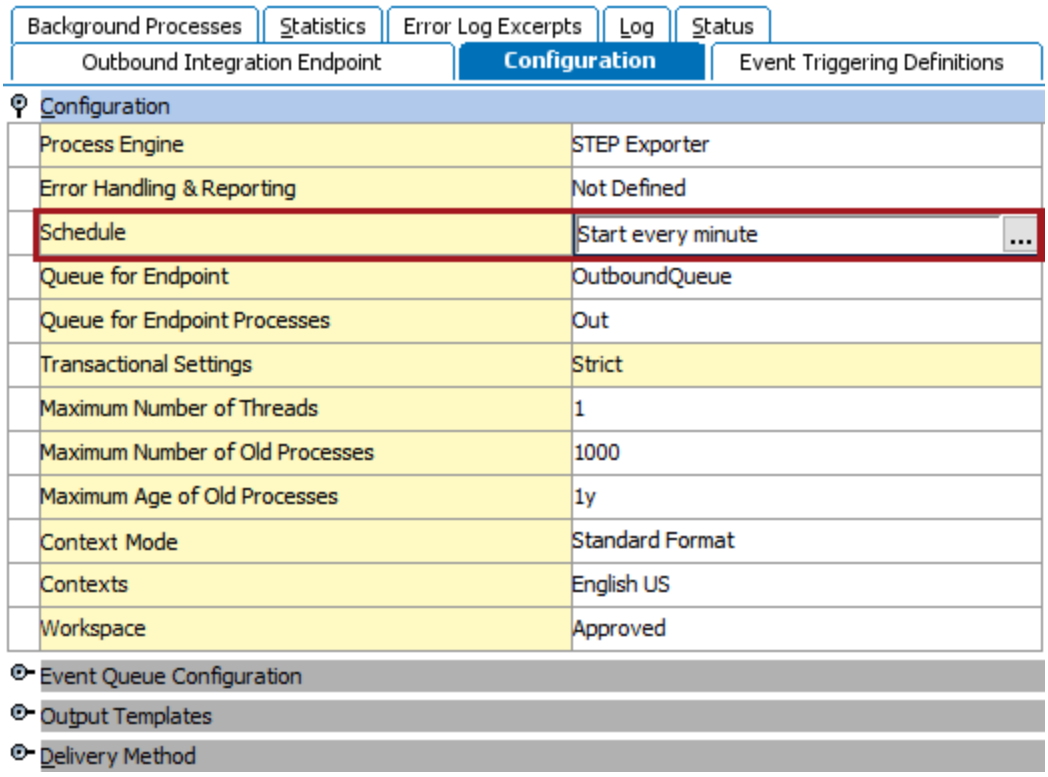


The tree navigator icon now shows enabled. On the Outbound Integration Endpoint tab, the Description flipper includes the 'Enabled' parameter, which now displays 'Yes.'



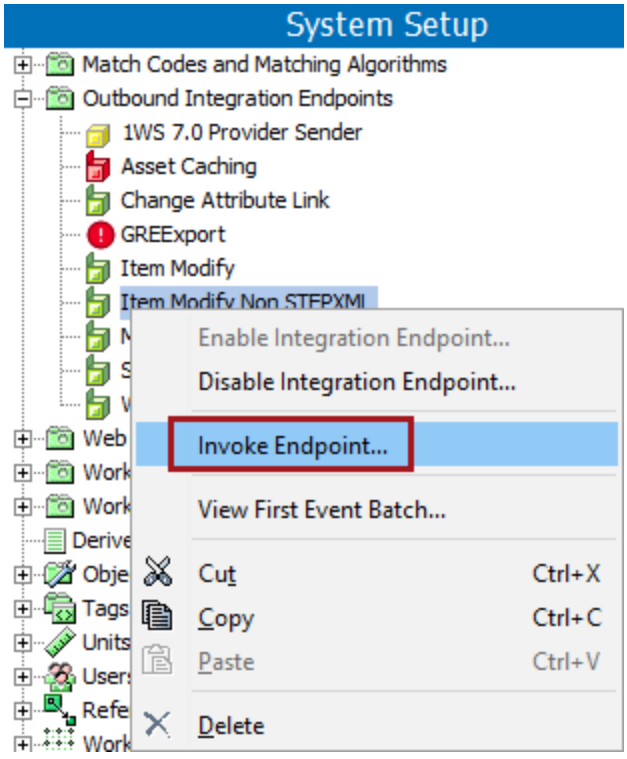
3. On the Configuration tab, open the Configuration flipper to view the Schedule parameter.

- If scheduled, the Schedule parameter shows when the OIEP will be invoked automatically.

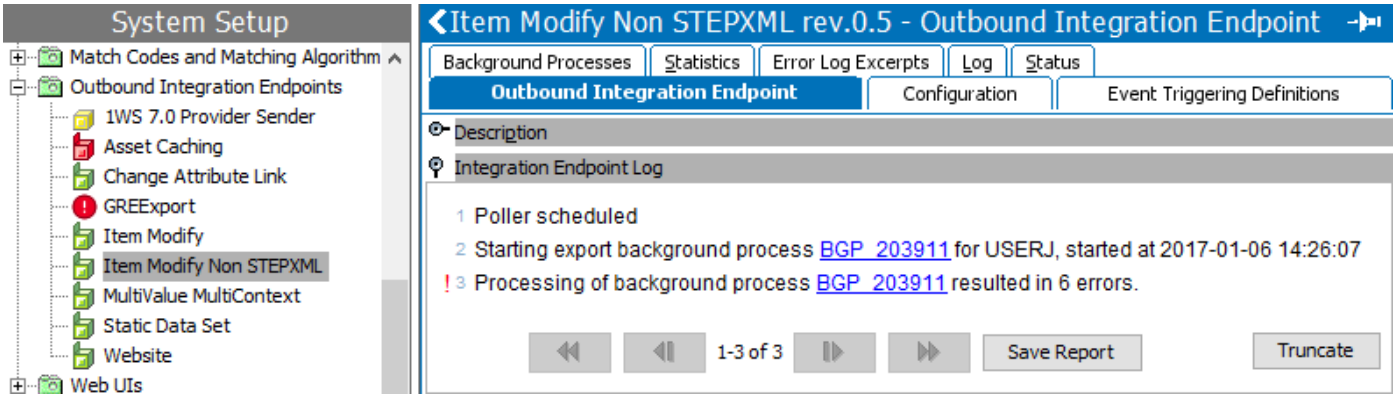


- If 'Not scheduled', or if the OIEP needs to be invoked manually. Right-click the enabled OIEP and click

Invoke Endpoint.



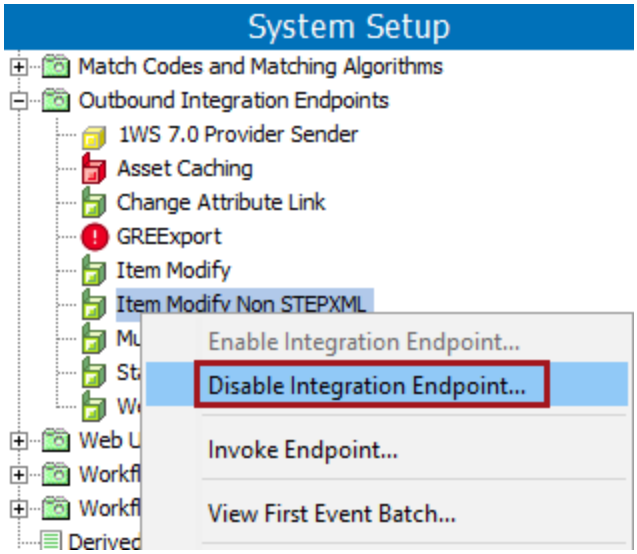
4. When invoked on schedule and manually, files or messages ready for export as identified by the OIEP configuration are processed. The Outbound Integration Endpoint tab Integration Endpoint Log flipper displays the time the endpoint was invoked. For scheduled OIEPs, the log includes a link to the background process that was started. For more information, refer to **Maintaining an Outbound Integration Endpoint**.



Disable an Outbound Integration Endpoint

When an OIEP is displaying the enabled icon ( ) should be stopped from processing data, you can disable it. The configuration remains unchanged, and no further data is processed. After selecting the OIEP, right-click and select **Disable Integration Endpoint** to set the OIEP to disabled and display the disabled icon ( ).

For example, disable an endpoint when running a data refresh activity on the system and the objects should not be published. Disabling an endpoint does not alter the configuration.

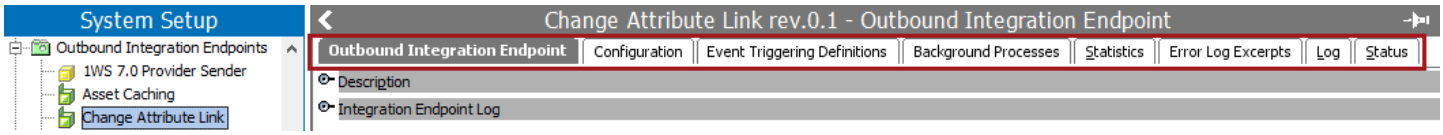


Once disabled, the OIEP can be restarted using the **Enable and Invoke an Outbound Integration Endpoint** section above. If collecting changes for future processing by the OIEP is no longer necessary, then also change the Queue Status to Discard Events to prevent excessive events from accumulating.

Additional information about restarting a failed endpoint, indicated by the failed icon ( ), can be found in the **Handling Failed OIEP Background Processes** documentation.

Maintaining an Outbound Integration Endpoint

Outbound integration endpoint settings can be viewed and edited from the outbound integration endpoint editor. For information about creating a new outbound endpoint, refer to the **Creating an Outbound Integration Endpoint** section.



Important: In order to avoid unnecessary Pollers from being generated, which can lead to a decline in system performance, endpoints should be disabled before making changes to unwanted processes. For more information regarding disabling an endpoint, refer to the **Running an Outbound Integration Endpoint** topic.

Each tab on the editor is defined below and includes parameters to maintain the OIEP.

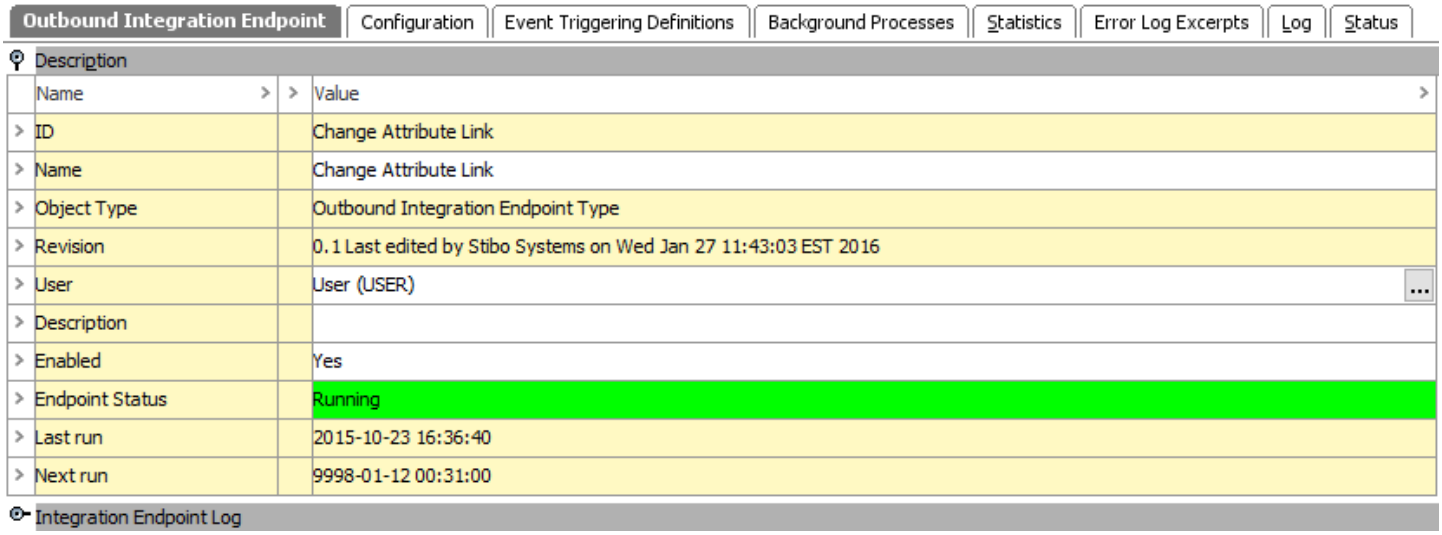
Outbound Integration Endpoint Tab

This tab holds basic information within the flippers defined below.

Description Flipper

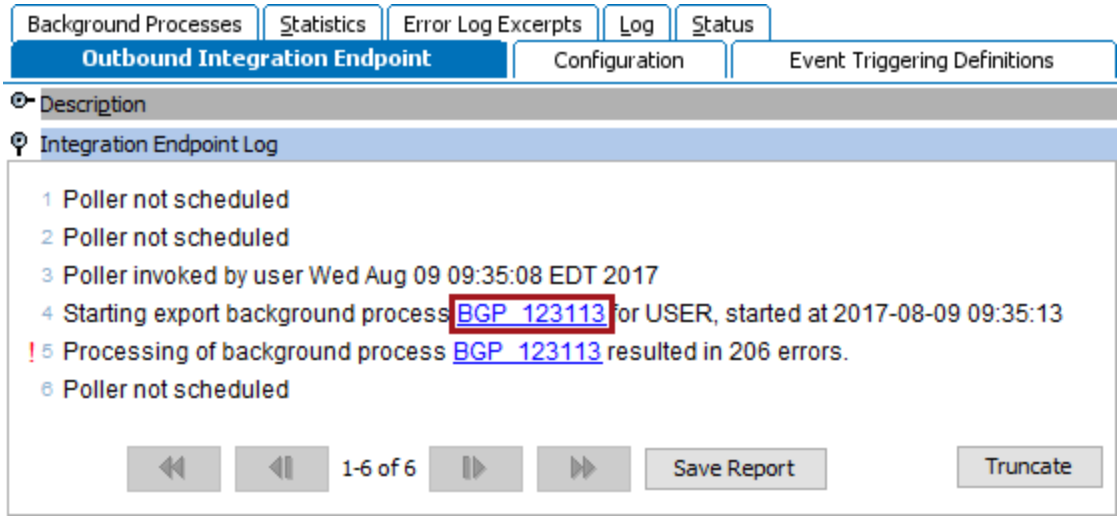
The Description flipper includes basic information to identify the OIEP. The name, user, description, and enabled parameters can be edited.

For information on the Endpoint Status, refer to **Running an Outbound Integration Endpoint** documentation.

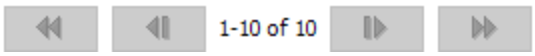


Integration Endpoint Log Flipper

The Integration Endpoint Log flipper contains information about the endpoint background process. An entry is generated each time the endpoint is invoked, when background processes generated by the endpoint are started, and when processing errors occur.

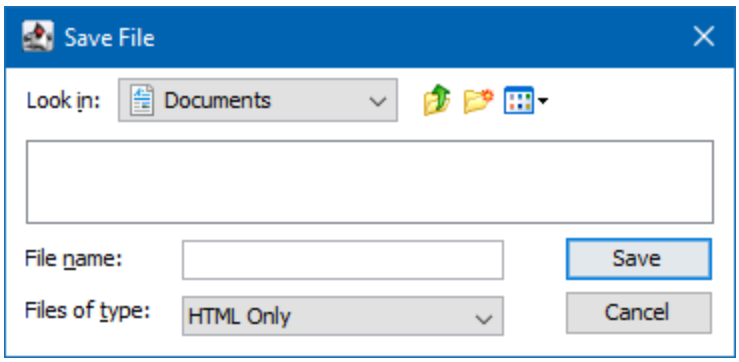


- **BGP link** - The background process opens the individual Importer Background Processes.
- **Navigation** - After the first entry, the Execution Report displays the following navigation buttons:

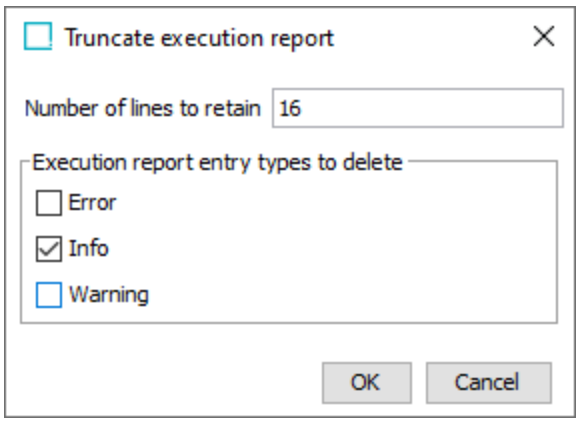
  The navigation buttons are enabled when until the log reaches over 100 lines.

  - Selecting the button will navigate back one page.
  - Selecting the button will navigate to the beginning of the log.
  - Selecting the button will navigate forward one page.
  - Selecting the button will navigate to the end of the log.
- **Save** - Click the Save button ( ) to save the Execution Report as an 'HTML Only' file type, then enter a name for the log report and click **Save**.



Note: Changing the default format manually may result in an error.

- **Truncate** - When the Execution Report becomes cluttered or only specific information is needed, use the Truncate option to permanently eliminate selected rows. Click the Truncate button (Truncate) to display the Truncate execution report dialog:



  - In **Number of lines to retain**, the current number of log entries will display. Type the number of lines to keep starting from the end of the log, and permanently deleting the oldest rows, based on the selected entry types below.
  - In **Execution report entry types to delete**, select the types that should be permanently removed.

    Note: The Properties flipper includes a count of the Warnings and Errors included in the Execution Report.

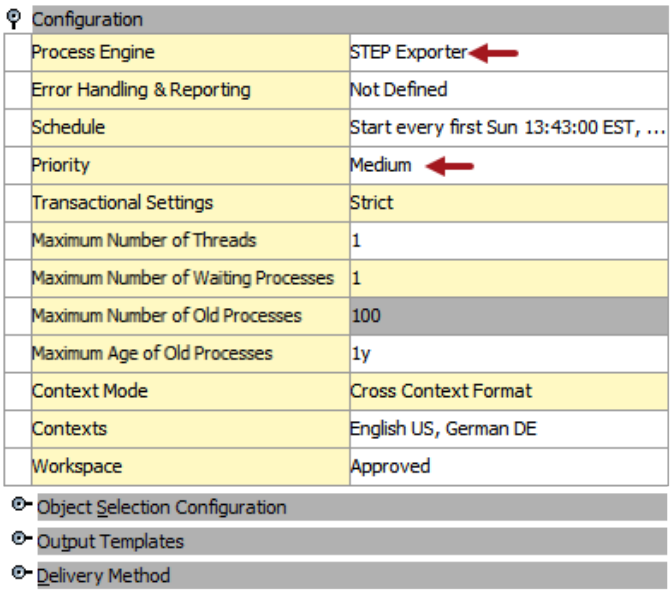
  - Click **OK** to permanently remove the selected number and types of execution report entries or click **Cancel** to leave the report unchanged.

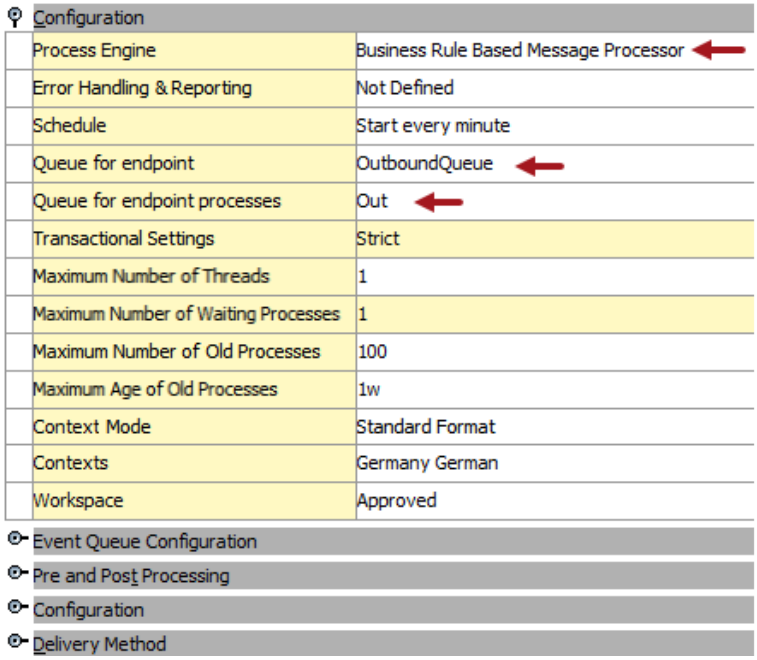
Configuration Tab

This tab contains many parameters required to successfully run an outbound integration endpoint.

Configuration Flipper

For the STEP Exporter and Business Rule Based Message Processor process engines, most of the parameters on this flipper can be edited. The type of editing available varies based on the parameter as defined below.





The **Transactional Settings** parameter cannot be edited and does not have the same significance as for IIEPs. For information, refer to **OIEP - Event-Based - Configure Endpoint** or **OIEP - Select Objects - Configure Endpoint**.

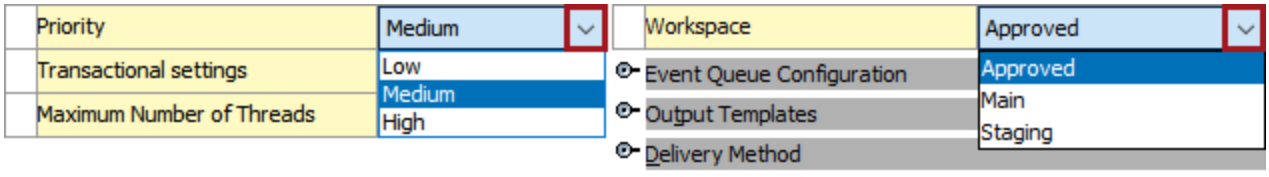
Depending on the Process Engine selected, a second Configuration flipper can be displayed. For more information, refer to the Processing Engine parameter in **OIEP - Event-Based - Configure Endpoint** or **OIEP - Select Objects - Configure Endpoint**.

The following parameters can be edited within the OIEP editor as defined:

- Click the **Process Engine**, **Priority**, or **Workspace** parameter to display a dropdown button. Selections displayed depend on your system setup.

  For information on the Process Engine or Workspace, refer to **OIEP - Event-Based - Configure Endpoint** or **OIEP - Select Objects - Configure Endpoint**.

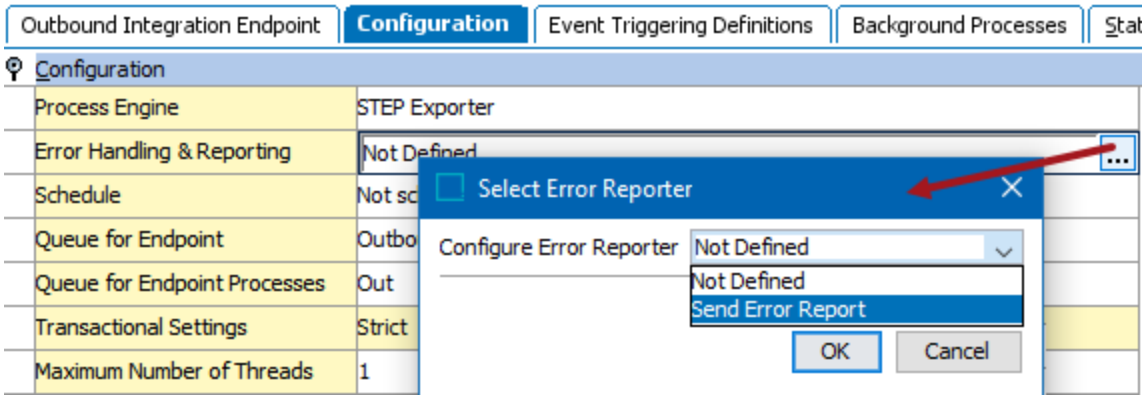
  For information on the Priority, refer to the **Priority** section of the **BG Processes Execution Management** topic in the **System Setup** documentation.



- Click the **Error Handling & Reporting**, **Schedule**, or **Contexts** parameter to display an ellipsis button (...). This opens a dialog for editing.

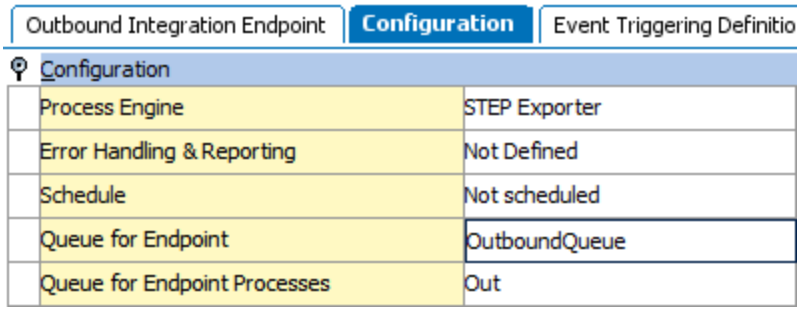
  For information on the Error Handling & Reporting or Schedule, refer to the Error Handling & Reporting section of the **OIEP - Configuration Flipper**.

  For information on Contexts, refer to **OIEP - Event-Based - Configure Endpoint** or **OIEP - Select Objects - Configure Endpoint**.



- Click the **Queue for endpoint** (legacy), **Queue for endpoint process** (legacy), **Maximum Number of Threads**, **Maximum Number of Old Processes**, or **Maximum Age of Old Processes** parameter to edit the text with the field.

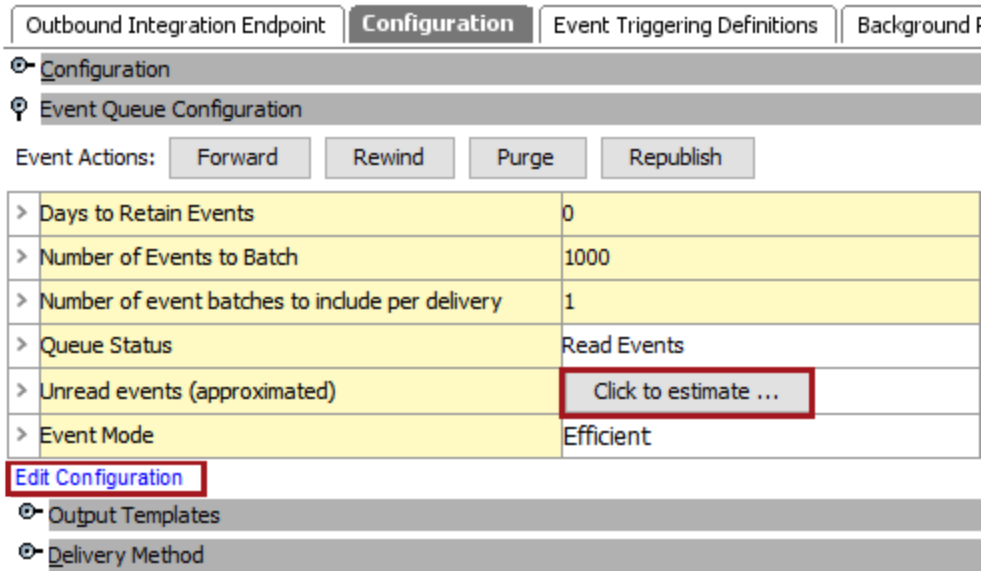
  For information on the Transactional settings, Number of threads, Contexts, and Workspace, refer to **OIEP - Event-Based - Configure Endpoint** or **OIEP - Select Objects - Configure Endpoint** or **Event-Based OIEP Multithreading Support**.



Event Queue Configuration Flipper

The additional configuration flipper displayed is determined by the type of OIEP being viewed. Event-based OIEPs include the Event Queue Configuration flipper.

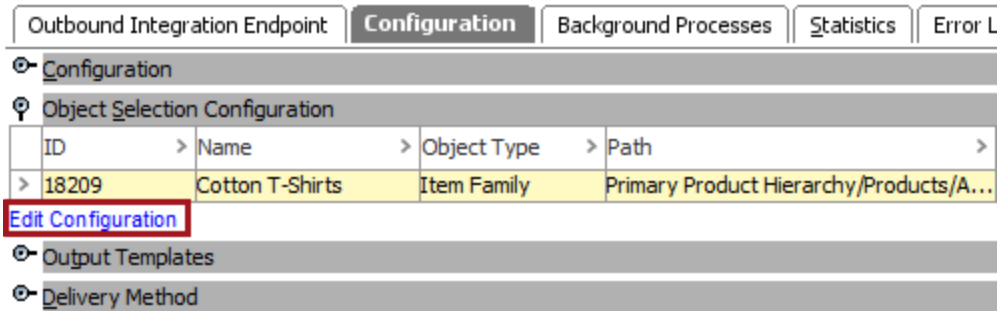
- The **Edit Configuration** link allows modification to the parameters displayed.
- The **Click to estimate** button displays a list of unread events.
- For more information on the Event Action buttons, refer to **Event-Based OIEP Event Actions**.



Object Selection Configuration Flipper

the Object Selection Configuration flipper is only displayed when the selected OIEP uses the Select Objects data source, as defined on the Choose Data Source step of the wizard. For more information, refer to **OIEP - Select Objects - Choose Data Source**.

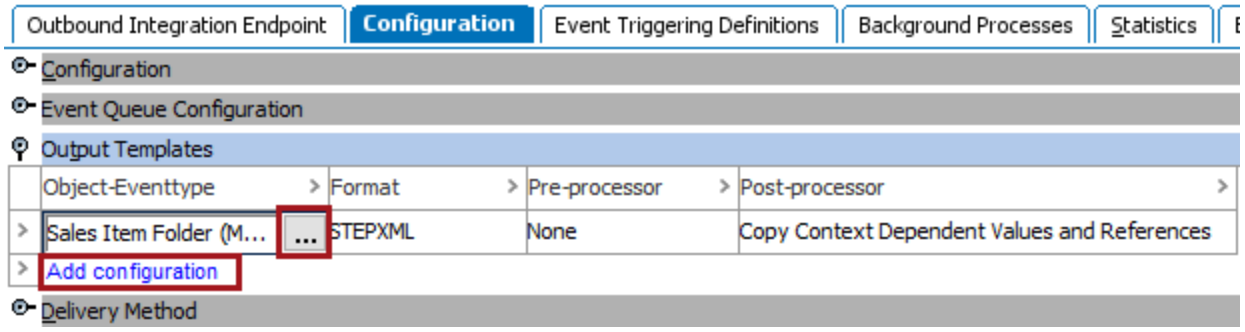
- The **Edit Configuration** link opens a dialog that allows modification to the objects selected for output. For details on the available options, refer to **OIEP - Select Objects - Object Selection Configuration Flipper**.



Output Templates Flipper

This flipper allows selection of object or event types, in addition to the format required for output. Pre-processors and post-processors can also be set and availability is based on the system configuration.

- Click into a field to display an ellipsis button (...). Click the ellipsis button (...) to display the relevant editor.
- Click the **Add configuration** link to insert a new row in the table.

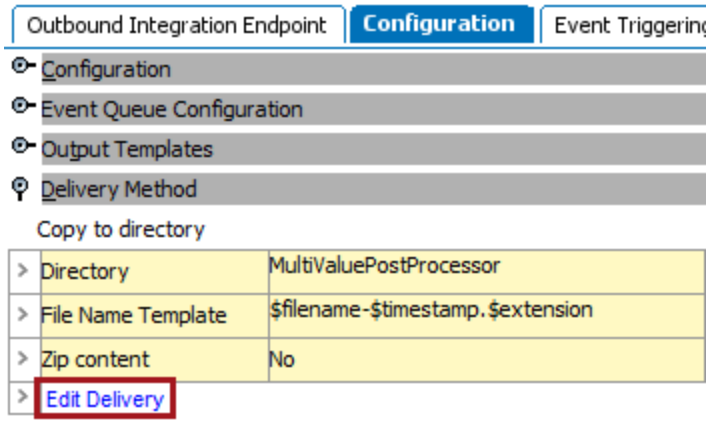


For details on the output templates, refer to **OIEP - Event-Based - Output Templates Flipper** or **OIEP - Select Objects - Output Templates Flipper**.

Delivery Method Flipper

Details about the delivery method selected are displayed.

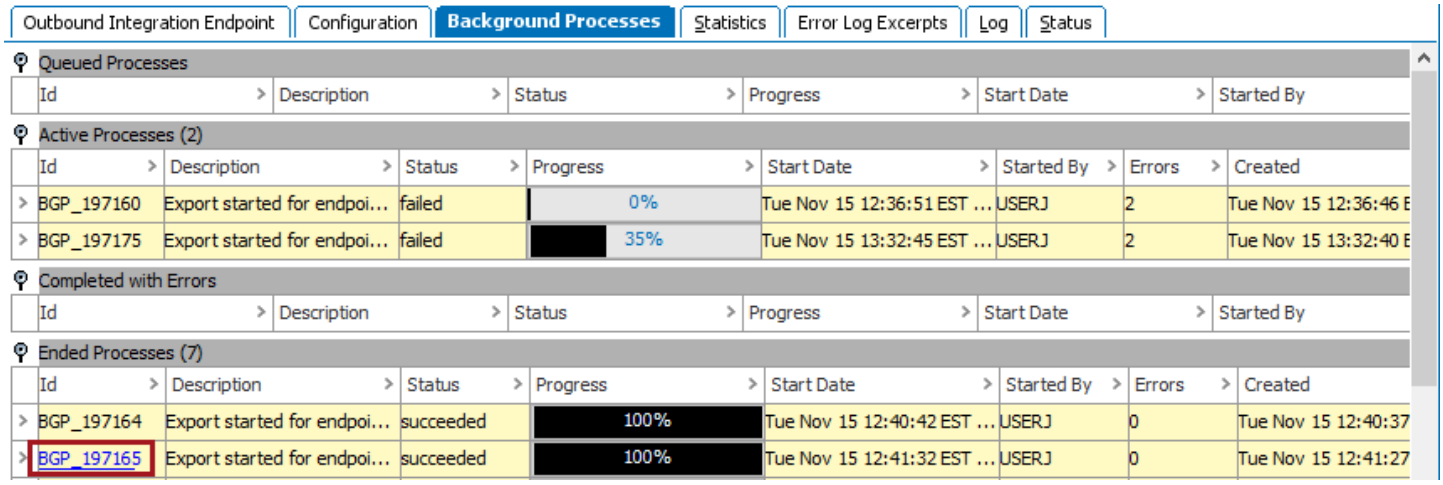
- Click the **Edit Delivery** link to modify the parameters.



For details on the available options, refer to **OIEP - Delivery Method Flipper**.

Background Processes Tab

This tab holds information about any related processes. Click the Background Process ID link to open the individual exporter background process.

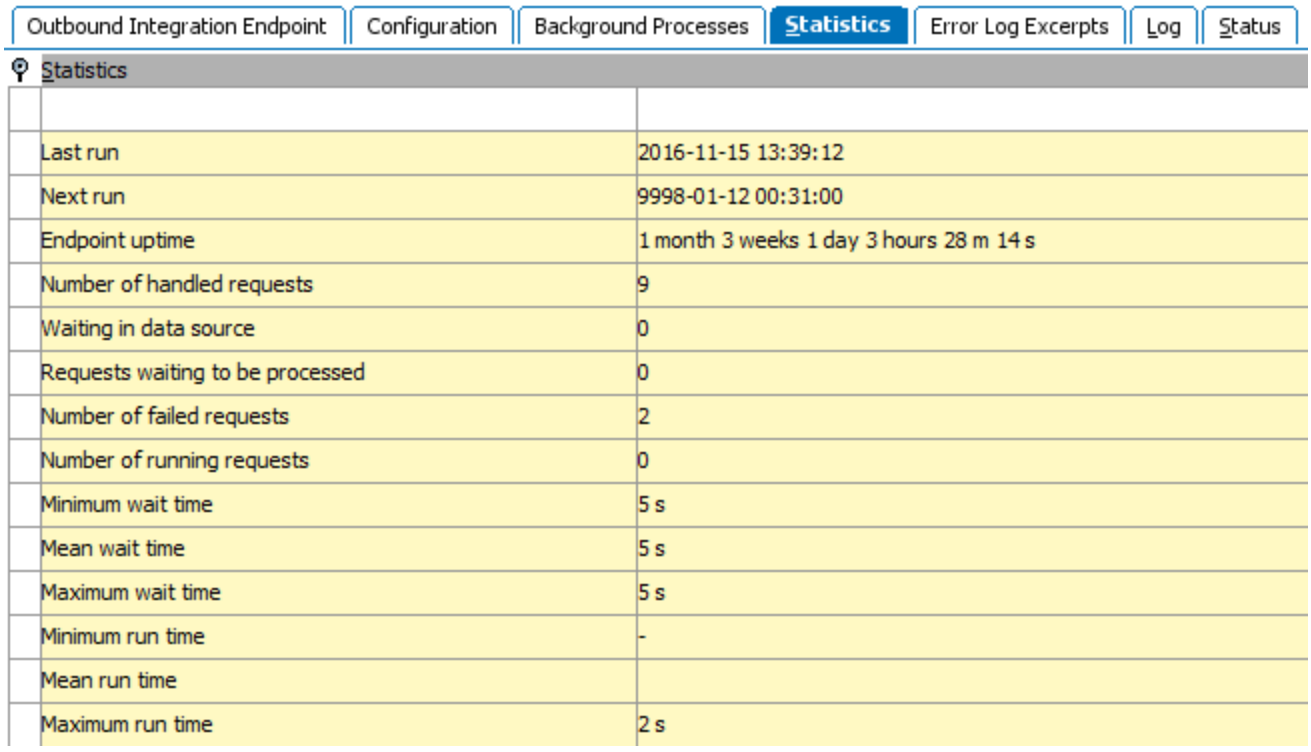


For details on the states of a background process, refer to the **BG Processes States** topic in the **System Setup** documentation.

For more information, refer to the **Monitoring an OIEP via Background Processes** section.

Statistics Tab

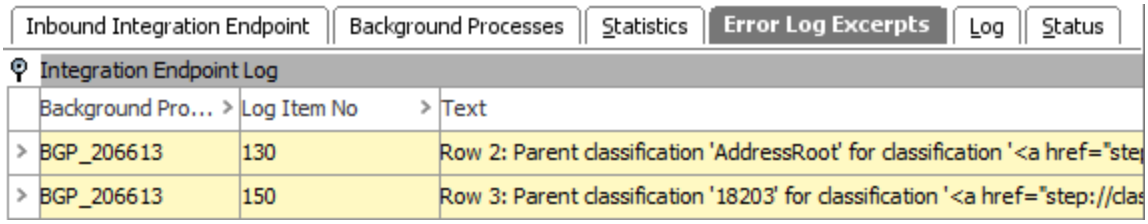
This tab provides statistics about the number of requests handled, run times, and wait times. The information cannot be edited nor exported.



Last run	2016-11-15 13:39:12
Next run	9998-01-12 00:31:00
Endpoint uptime	1 month 3 weeks 1 day 3 hours 28 m 14 s
Number of handled requests	9
Waiting in data source	0
Requests waiting to be processed	0
Number of failed requests	2
Number of running requests	0
Minimum wait time	5 s
Mean wait time	5 s
Maximum wait time	5 s
Minimum run time	-
Mean run time	
Maximum run time	2 s

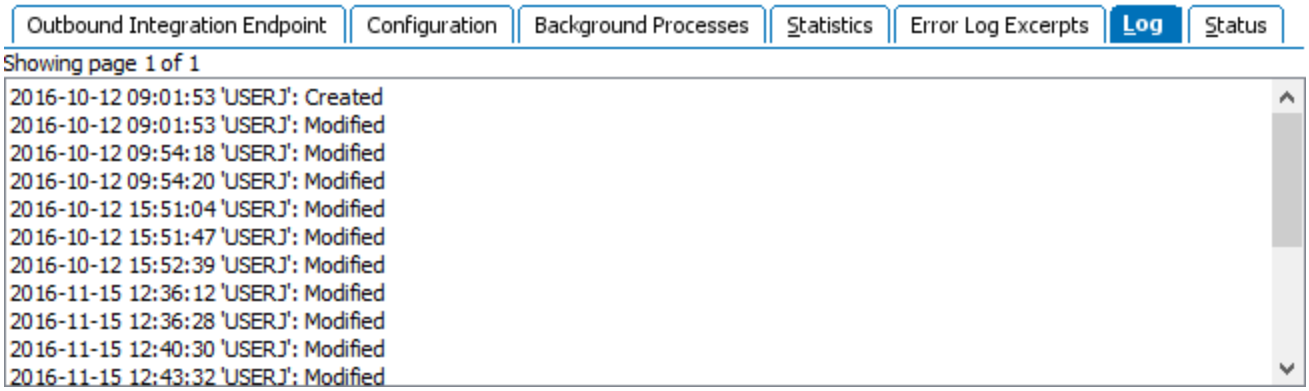
Error Log Excerpts Tab

This tab shows data from the main Java log file related to failed background processes with Log Level greater than 'Info.' Click a hyperlink to a failed background process and correct the cause of the failed background process. For more information, refer to **Handling Failed OIEP Background Processes**.



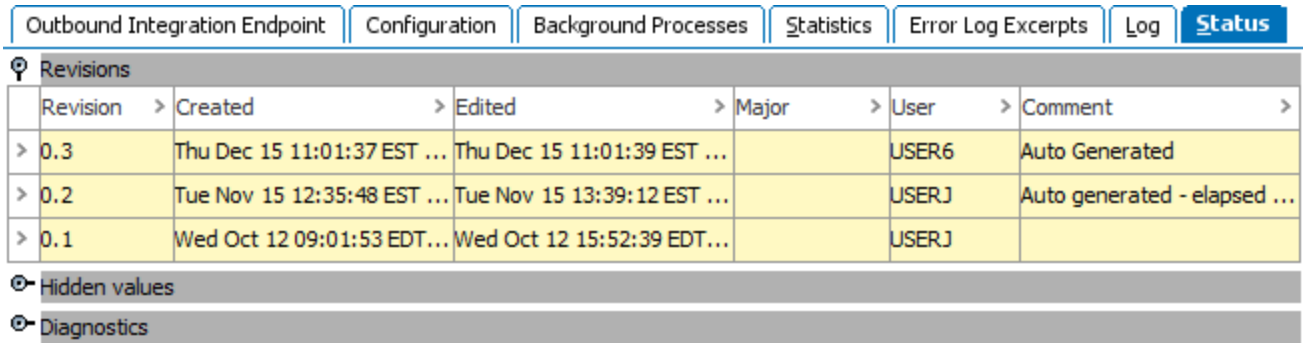
Log Tab

This tab provides information about changes to the OIEP configuration by a user, including the timestamp.

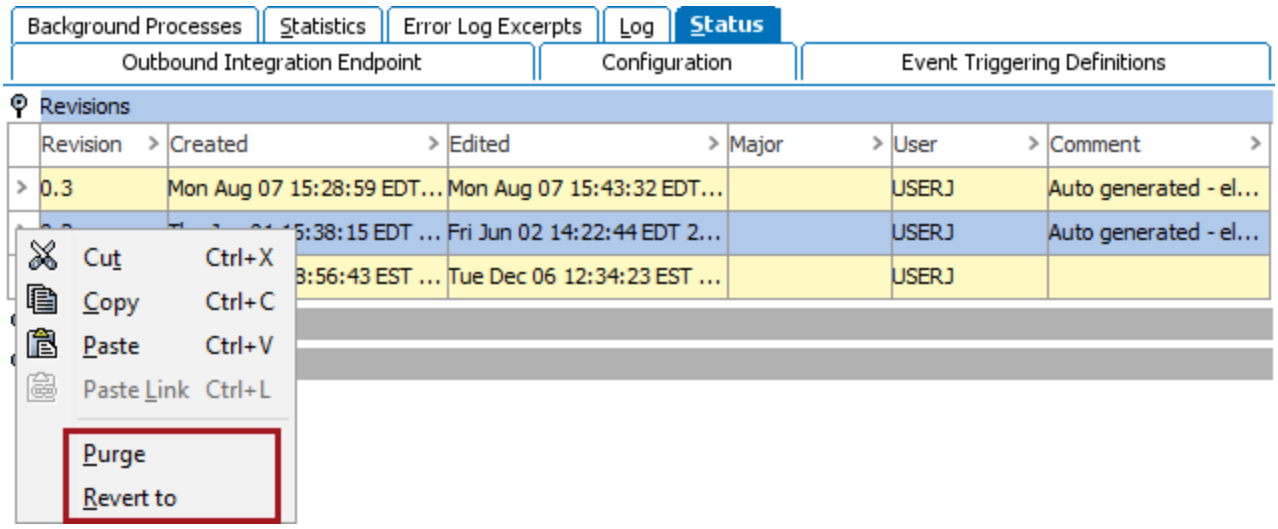


Status Tab

This tab provides information about revisions, hidden values, and diagnostics.



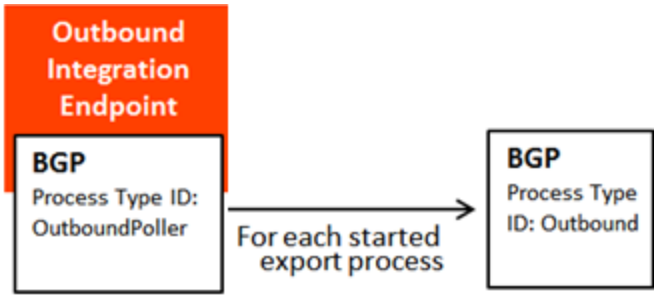
Right-click on the arrow (>) to the left of a revision row to display a menu that allows you to revert to the changes for the selected revision or delete (purge) the selected revision.



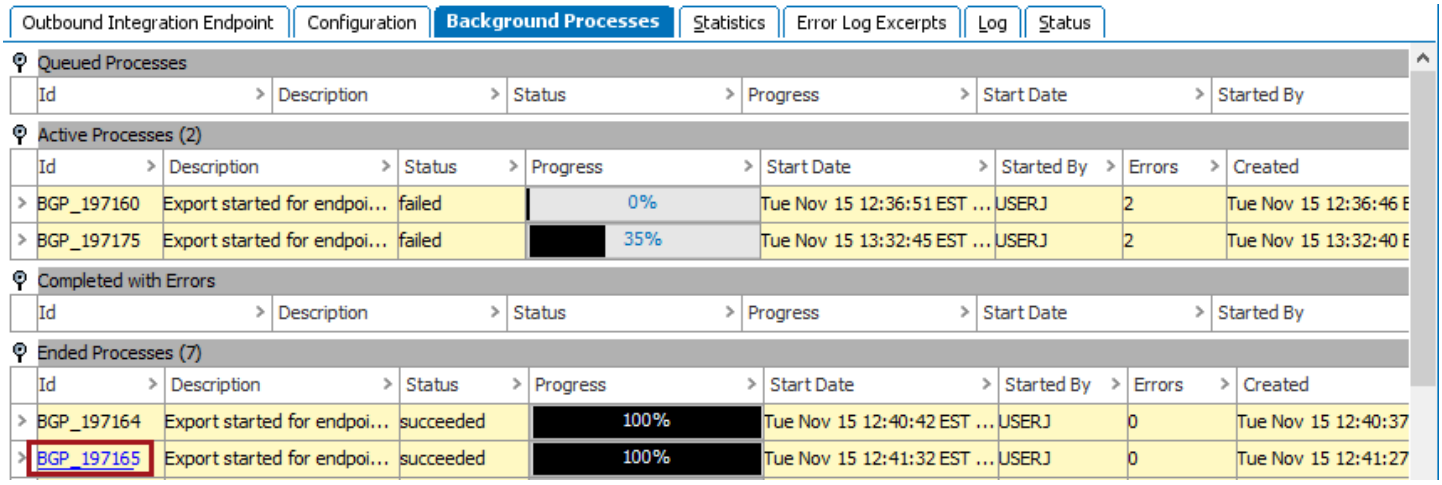
For more information on revisions, refer to the **Revisions** topic in **System Setup** documentation.

Monitoring an OIEP via Background Process

An active outbound integration endpoint uses an associated background process to handle the scheduled invocation of the endpoint. The execution report for this background process is integrated in the endpoint editor and includes invocation information as well as messages logged from the plugins. When using the standard STEP Exporter processing engine, the actual export is handled in separate background processes started by the endpoint, as illustrated below.



By default, the started integration endpoint background processes are displayed within the OIEP editor on the Background Processes tab, and are not visible on the BG Processes tab. Click the background process hyperlink to display details about the actual process. For more information, refer to **Maintaining an Outbound Integration Endpoint**.



Background processes are run based on the execution mechanism configured and can be either the recommended priority-based execution or the legacy queue setting execution.

Background Process Priority Settings

When the recommended priority-based BGP execution mechanism is configured, waiting BGPs are prioritized for execution based on the priority of the BGP and the created time. The legacy 'Queue for Endpoint' and legacy 'Queue for Endpoint Processes' parameters are not available. Refer to the **Priority** section of the **BG Processes Execution Management** topic in the **System Setup** documentation.

Legacy Background Process Queue Settings

For legacy BGP execution, when configuring an outbound integration endpoint that uses the STEP Exporter processing engine, you will specify a background process queue to use for each of the two types of background processes:

- Queue for endpoint
- Queue for endpoint processes

Typically, all background processes of the same type (same process type ID) will use the same queue. However, integration endpoint processes can be tied to different integration endpoints and use different queues.

Legacy Background Process Queue

The main background process uses the default **OutboundQueue** background process queue for OIEPs.

However, when you create an integration endpoint, you can specify that you want to use a different OIEP background process queue for the main OIEP background process queue. For more information about this setting, refer to the **OIEP - Configuration Flipper** documentation.

Legacy Background Process Queue Size

The queue size setting can have great impact on processing. The background process queue size property determines the number of background processes can be executed in parallel on the queue. If the size is 1, only one background process can run at a time. If the size is 2, two processes can run in parallel, and so forth.

Important: Although the integration endpoint configuration wizards suggest the number of queues to use, beware that if you create multiple endpoints and do not change the default queue suggestions, your setup could include endpoints that block each other since the queues have a default size of 1.

This is not a big concern for the endpoint processes that, most of the time, are idle and only take up a slot on the queue briefly when the endpoint is invoked. However, it can be an issue for the generated background processes that perform the actual export processing. The decision as to whether different endpoints should use the same or different processing queues and the size of the queues should therefore be an informed one.

Overwriting default values for a queue causes a new queue to be created with the queue size of 1. Use the following sharedconfig.properties entry to modify the default queue size value:

```
BackgroundProcess.Queue.[name of queue].Size = [number of allowed parallel processes]
```

Monitoring an OIEP via External Systems

Integration endpoints (IEPs) can be monitored from external systems using Monitoring Sensors or the REST API. Nagios is one application that can monitor the status of integration endpoints.

Performance Data

When using a sensor with details, the 'Performance Data' section of the status page includes the information outlined below.

Note: All of the background processes (BGPs) of the IEP are collected and grouped by the Status, Wait Time, and Run Time in the database.

- **Max Run Time** - largest value in the Run Time column; determines the longest Run Time among all IEP BGPs.
- **Mean Run Time** - average value in the Run Time column; determines the average Run Time among all IEP BGPs.
- **Min Run Time** - smallest value in the Run Time column; determines the shortest Run Time among all IEP BGPs.
- **Total Run Time** - sum of all values in the Run Time column; determines how much time all the IEP BGPs were running.
- **Max Wait Time** - largest value in the Wait Time column; determines the longest Wait Time (in the queue) among all IEP BGPs.
- **Mean Wait Time** - average value in the Wait Time column; determines the average Wait Time (in the queue) among all IEP BGPs.
- **Min Wait Time** - smallest value in the Wait Time column; determines the shortest Wait Time (in the queue) among all IEP BGPs.
- **Number of Failed BGPs** - count of the BGPs in 'completedwitherrors' and 'failed' states.
- **Number of handled requests** - count of the BGPs NOT in the 'running' and 'waiting' states.
- **Number of running BGPs** - count of the BGPs in 'completedwitherrors,' 'failed,' 'aborted,' and 'succeeded' states.
- **Number of waiting BGPs** - count of the BGPs in the 'waiting' state.
- **Number waiting in Data Source** - OIEP specific; returns the number of messages waiting to be output.
- **Total number of BGPs** - count of the BGPs in all possible BGP states ('completedwitherrors,' 'failed,' 'waiting,' 'running,' 'aborted,' and 'succeeded').

Monitoring Sensors

For each integration endpoint created in STEP, a Monitoring Sensor is automatically created. Monitoring Sensors allow external systems to query the status of individual endpoints via HTTP, without authentication. In the following sample URLs, substitute your own system URL and endpoint ID to access your own endpoints.

A Monitoring Sensor can return the following information:

1. **Simple traffic light response** returns OK, WARNING, CRITICAL, or UNKNOWN for outbound endpoints via:

   ```
   http://[System URL]/admin/monitoring/OutboundIntegrationEndpointStatus-[Endpoint ID]/status
   ```

2. **Nagios friendly response** returns detailed statistics in Nagios friendly format for outbound endpoints via:

   ```
   http://[System URL]/admin/monitoring/OutboundIntegrationEndpointStatus-[Endpoint ID]/nagios
   ```

3. **Full detailed XML response** returns XML with detailed statistics for outbound endpoints via:

   ```
   http://[System URL]/admin/monitoring/OutboundIntegrationEndpointStatus-[Endpoint ID]/xml
   ```

REST API Monitoring

An alternative to Monitoring Sensors is to use the REST API, which requires authentication.

1. **Overview of all** configured integration endpoints via:

   ```
   [Host]/restapi/integrationendpoints?context=[Context]&workspace=[Main]
   ```

2. **Detailed statistics** for specific endpoint via:

   ```
   [Host]/restapi/integrationendpoints/[Endpoint ID]?context=[Context]&workspace=[Main]
   ```

3. **Execution report** for specific endpoint via:

   ```
   [Host]/restapi/integrationendpoints/[Endpoint ID]/log?context=[Context]&workspace=[Main]
   ```

4. **Java log entries related to errors in Background Processes** generated by specific endpoint via:

   ```
   [Host]/restapi/integrationendpoints/[Endpoint ID]/errorexcerpts?context=[Context]&workspace=[Main]
   ```

5. **Overview of Background Processes** generated by specific endpoint via:

   ```
   [Host]/restapi/integrationendpoints/[Endpoint ID]/backgroundprocesses?context [Context]&workspace=[Main]
   ```

Additional Information About All Outbound Integration Endpoints

This information is relevant to understanding additional functionality available for implementing and using any outbound integration endpoint. Included are details on the following features:

- Export outbound integration endpoint definition for comparison purposes in an external source control system for comparison purposes as described in the **Configuration Management** documentation.
- Understand how to handle failed BGPs for outbound integration endpoints as described in **Handling Failed OIEP Background Processes**.
- Understand how background processes will be generated, how the messages will be processed, and what will happen upon failure as described in **Integration Endpoint Transactional Settings**.
- Understand each element of an OIEP as described in **Outbound Integration Endpoint Structure**.

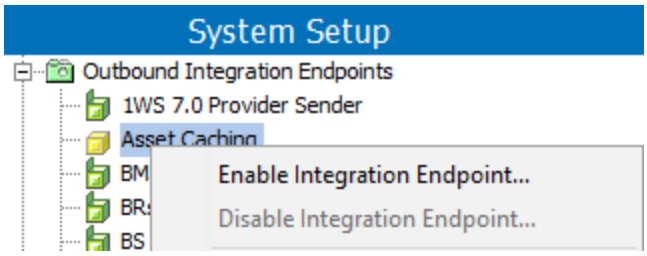
Handling Failed OIEP Background Processes

When a background process encounters an error, it is reported in the log. Errors can be reviewed in the log, resolved, and then the failed integration endpoint can be reactivated using Disable / Enable or Resume.

Disable and Enable a Background Process

Setting an integration endpoint to **Disable** and then **Enable** *does clear* the log file.

- Right-click the endpoint and select **Disable Integration Endpoint**.
- Right-click the endpoint again, and select **Enable Integration Endpoint**.

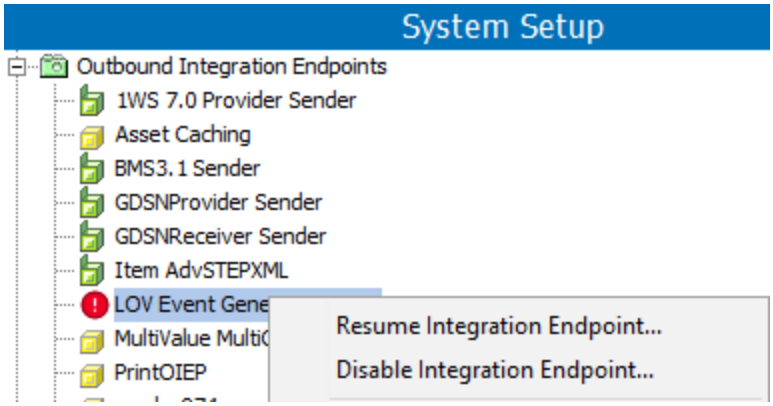


Resume a Background Process

Resuming an integration endpoint *does not clear* the log file

Note: Whenever the STEP system is patched or restarted, the main background process is automatically restarted. The integration endpoint log still exists, and continues to show an overview of the started background processes.

- Right-click the endpoint, and select **Resume Integration Endpoint**.



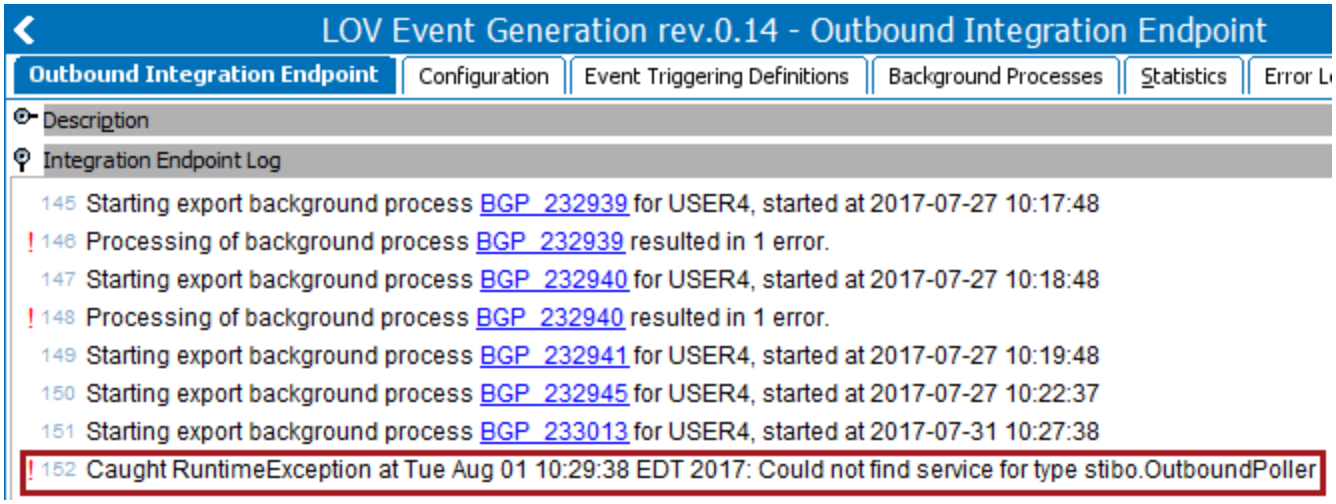
Resolving a Failed Background Process

Errors can happen before the background process starts, and also after it is running. The error reporter can be configured when creating an OIEP, as described in the **Error Handling & Reporting** section of **OIEP - Configuration Flipper** topic.

Outbound process errors can happen, for example, due to the FTP connection being unavailable, when the delivery directory is out of space, or if a network connection is down. STEP stops the OIEP until the problem can be resolved.

OIEP fails before the BGP starts

In cases where the OIEP fails before the BGP is initiated, a report is available in Integration Endpoint Log on the Outbound Integration Endpoint tab. Refer to this log for information to make needed corrections.



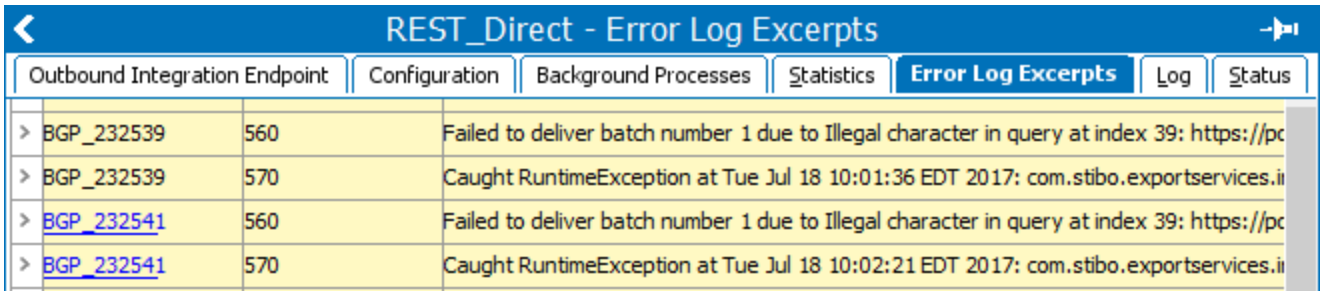
OIEP fails after the BGP starts

The Error Log Excerpts tab includes details about failures within the background process.

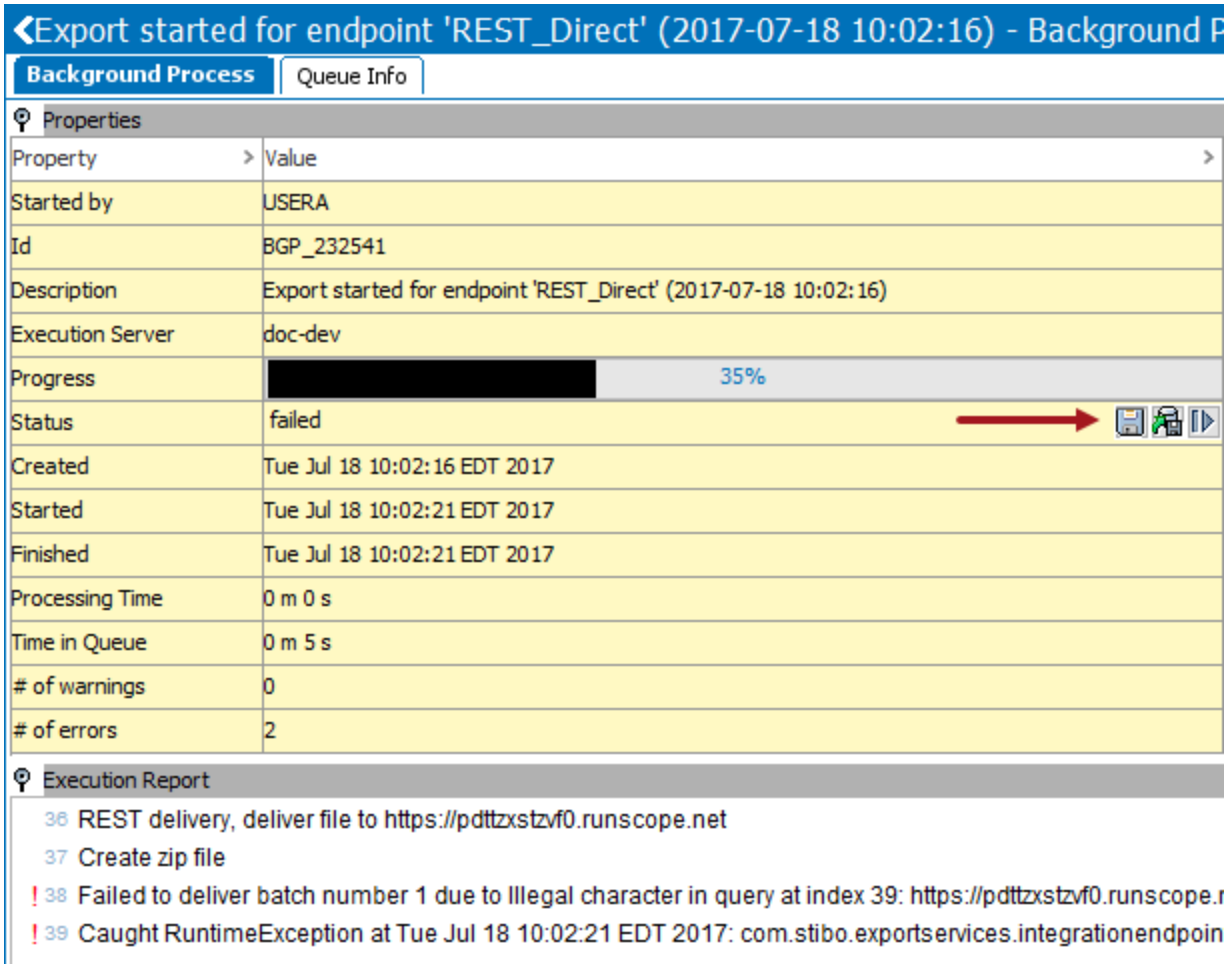
1. In **System Setup** select the relevant integration endpoint, and then click the **Error Log Excerpts** tab to display a list of failed background processes and the description of the failure.

   You can also verify if a process has been suspended because it is dependent on a failed process.

2. In the **Background Process** column, hover over the background process to correct and click the **BGP link**.



The Background Process editor displays.



3. In the Background Process editor **Execution Report**, view the progress of the background process, and where it failed.
4. In the Properties flipper, view the Status parameter, if available, click the Save button ( ) in the Value column, near the failed status, to save the file locally.
5. Open the file in a relevant editor and make the required changes.
6. After completing the changes, upload the file back to the same background process. In the Background Process editor, open the Properties flipper, view the Status parameter Value column, and click the Upload button ( ) near the failed status.
7. Restart the process in the Properties flipper on the Status parameter Value column, by clicking the Restart button ( ) near to the failed status. The background process is restarted.
8. If the background process is part of a batch of chained processes, restart the other processes that are part of the chain.

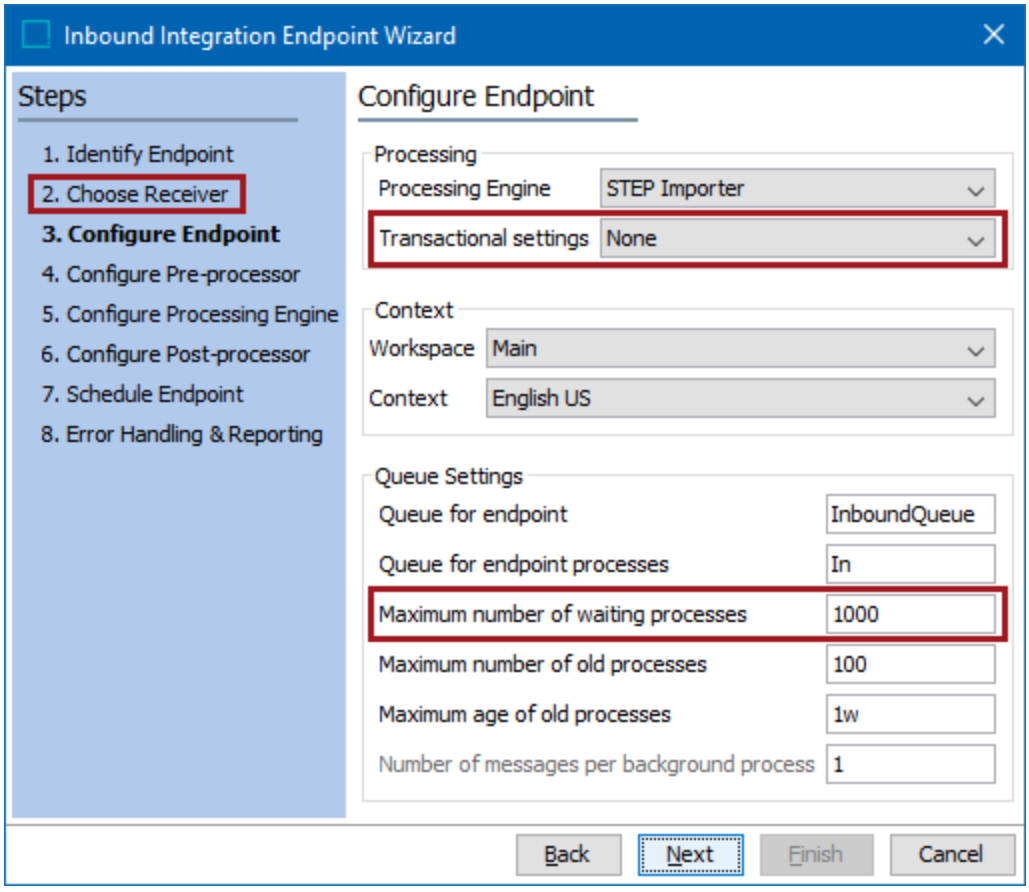
Integration Endpoint Transactional Settings

For both inbound and outbound integration endpoints, 'transactional settings' specify how background processes will be generated, how the messages will be processed, and what will happen upon failure.

After configuring transactional settings on the integration endpoint, configuration for queue size and parallel are set in the sharedconfig.properties file as defined in the **Background Processes and Queues** topic of the **System Setup** documentation.

Inbound Integration Endpoints

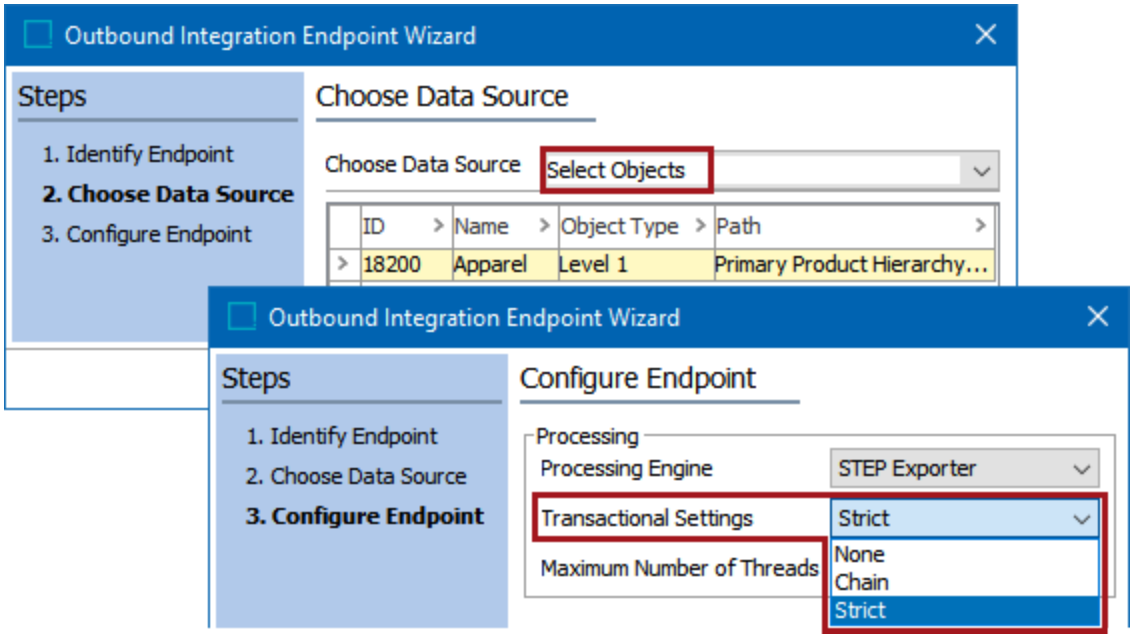
In the IIEP wizard, background process and error handling are managed by three (3) elements: the selected Receiver (on the Choose Receiver step), the 'Transactional settings' parameter, and the 'Maximum number of waiting processes' parameter (both on the Configure Endpoint step). These elements also work together to determine what additional options are available in the wizard, as defined below.



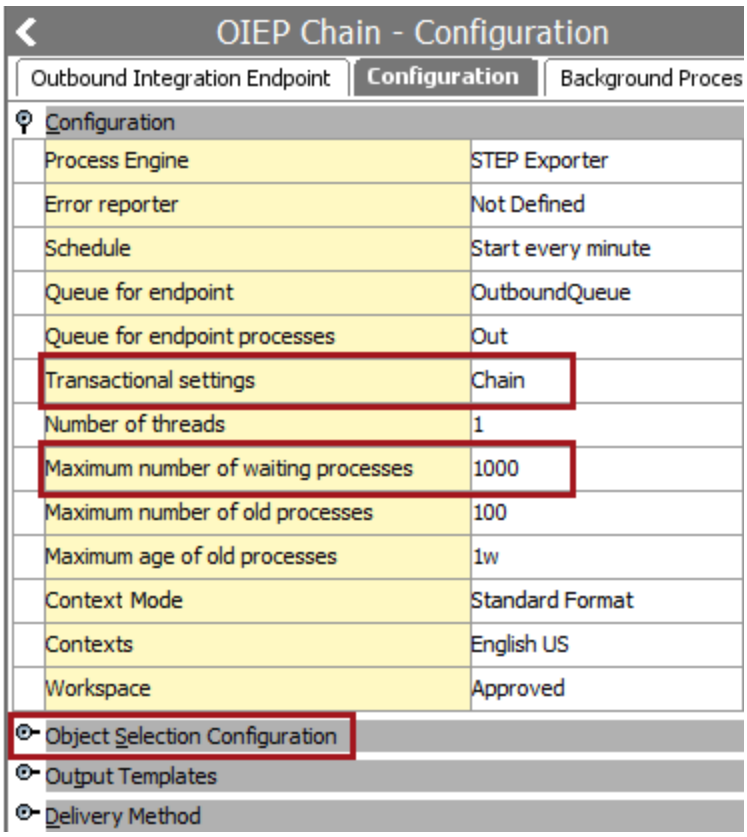
For more information, refer to the **IIEP - Configure Endpoint** documentation.

Outbound Integration Endpoints

While creating an outbound endpoint using the wizard, the Choose Data Source step determines if the transactional settings can be set in the wizard. As defined below, selecting a Select Objects data source provides several options for transactional settings; selecting an Event Queue data source requires the 'Strict' setting.



For Select Objects OIEPs, the transactional settings can be modified in the OIEP editor, on the Configuration tab. The 'Transactional settings' parameter and the 'Maximum number of waiting processes' parameter determine how background processes and errors are managed.



For more information on select objects OIEPs, refer to the **OIEP - Select Objects - Configure Endpoint** topic.

For details on the 'Transactional settings' and 'Maximum number of waiting processes' parameters, refer to the **OIEP - Configuration Flipper** topic.

For event-based OIEPs, only the 'Strict' transactional setting is allowed. 'Strict' ensures that an event in the STEP event queue is not removed until the endpoint has successfully delivered the result of the event to the external system. If event processing fails (i.e., the external system is not running and therefore does not accept any data), the endpoint is disabled but events are not removed from STEP. Once the error state is fixed by a user, the endpoint can be resumed; events waiting to be sent to the external system are processed, and the data history is retained.

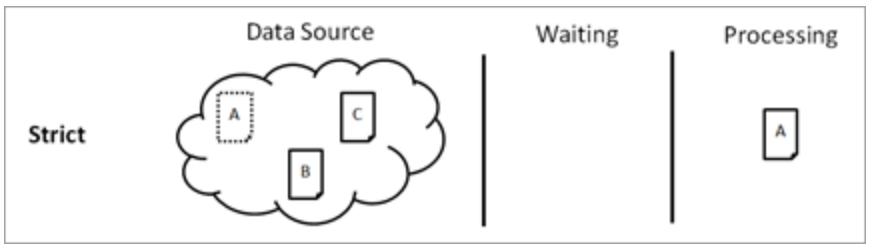
For more information on event-based OIEPs, refer to the **OIEP - Event-Based - Configure Endpoint** topic.

Transactional Settings Options

When available, the possible transactional settings are: Strict, Chain, and None.

Strict

Using the 'strict' transactional setting processes data in a strict order, using one background process at a time. When one background process completes successfully, the endpoint starts the next background process. If a background process fails with an error, or if it cannot deliver a file, the next process is not started, and the 'Failed' status is set. This is the most common setup, unless there is a reason to use Chain or None (for assets).



The queue size of a background process with strict transactional settings must be one (1). A strict transactional setting does not allow multiple background processes to work in parallel / concurrently regardless of the queue size. By definition, no new background processes can be generated while one is active.

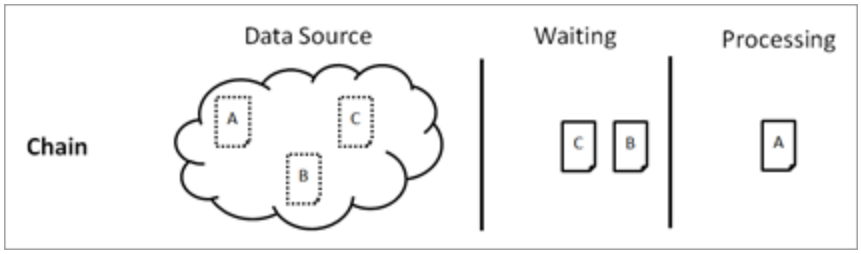
Strict is allowed for all inbound receiver methods and for outbound select data endpoints but is required for outbound event-based endpoints.

Chain

Using the legacy 'chain' transactional setting processes data in batches of chained background processes. With this setting, a batch is generated each time the endpoint polls for data and finds more than one message or file that has not yet been processed.

Note: Generally, 'chain' is not recommended due to slower processing speeds compared to 'strict' with similar features for enforcing order of import files.

The **Maximum number of waiting processes** parameter allows you to specify the maximum number of background processes with the status 'Waiting' that are allowed. If a background process in a batch fails, the remaining background processes in the batch will also fail. However, the endpoint remains active with the status 'Running' and continues to process data in the next batch of chained background processes. A chain endpoint stops if a file cannot be delivered.

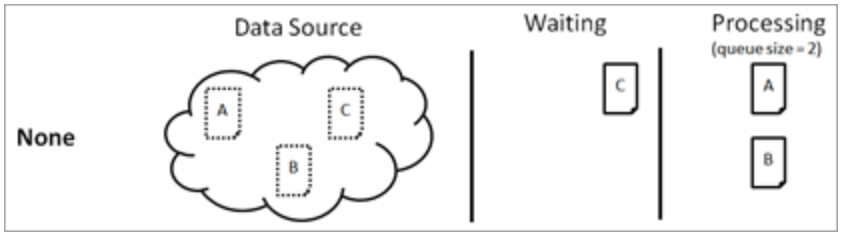


The queue size of a background process with chain transactional settings must be one (1). A chain transactional setting does not allow multiple background process to work in parallel / concurrently regardless of the queue size. By definition, no new background processes can be generated while a background process is active. The default 'Maximum number of waiting processes' is 1,000.

Although chain is allowed for all inbound receiver methods and for outbound select data endpoints, it is not recommended as it results in slower processing times.

None

Using the 'none' transactional setting processes data concurrently, without any transactional restrictions or data dependencies. This is useful, for example, when processing assets. The default queue size of a background process with transactional settings of 'none' is one (1). Data is not processed in a strict order and if one background process fails, the endpoint continues to process data in the next background process in the queue.



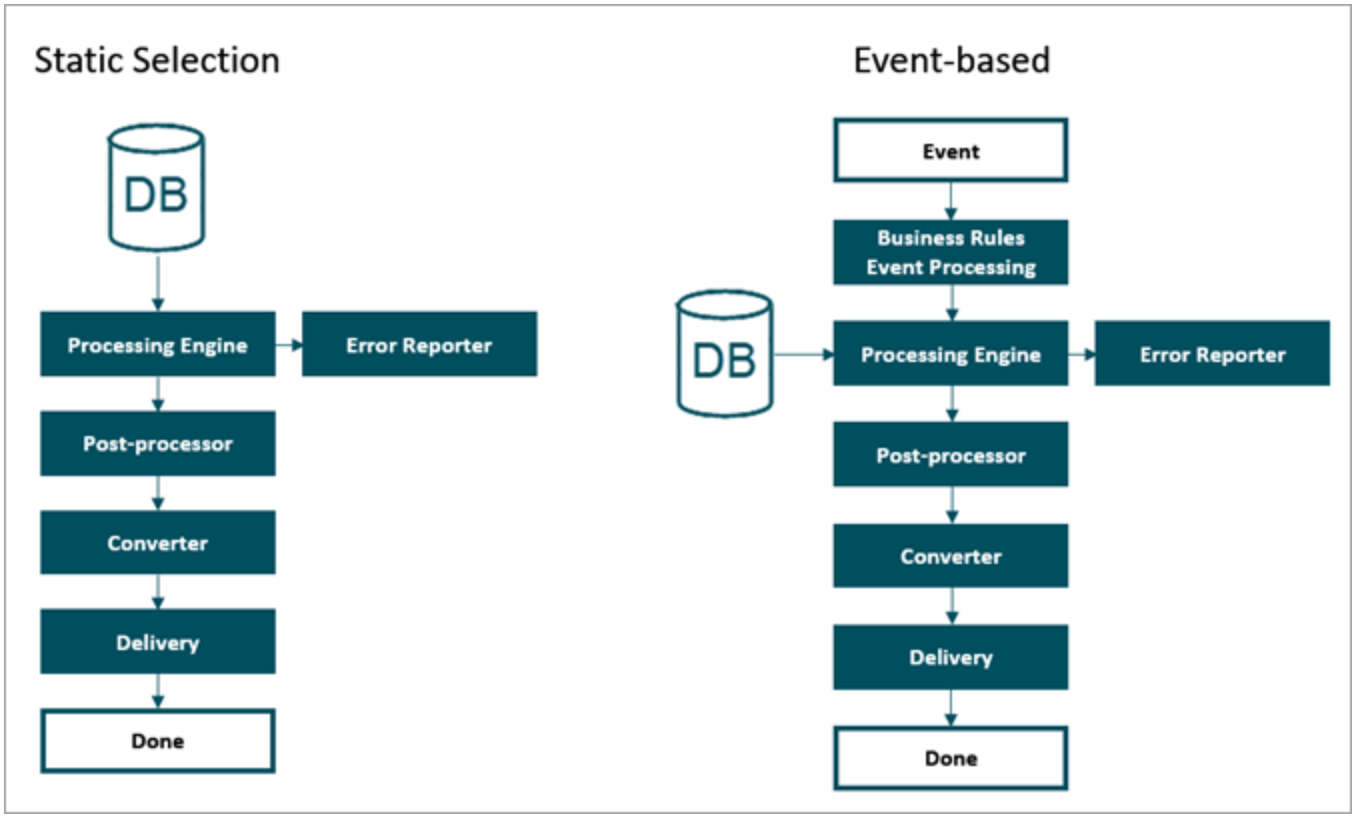
None is not allowed for the inbound 'Hotfolder Using File Sequence Receiver' or 'Hotfolder Using Meta Files Receiver' methods but can be set for other inbound endpoint receivers. It can also be set for outbound select objects endpoints.

Important: Switching the 'Transactional setting' parameter after an OIEP has been created will also change the 'Maximum number of waiting processes' parameter. If changing from Strict to Chain or None, the 'Maximum number of waiting processes' will stay at one (1). Users will need to go in and change that number to higher than one (1) in order to attain Chain or None processing. If the OIEP was set up with the Chain or None settings, then the 'Maximum number of waiting processes' will continue to display 1000 when changed to Strict until the workbench is refreshed.

Outbound Integration Endpoint Structure

The outbound integration endpoint (OIEP) functionality has been created to allow for easy extensions / customizations and can be understood as a plugin framework with interchangeable parts. The graphic below illustrates the structure for both a non-incremental endpoint based on a static selection and an event-based endpoint. The colored boxes represent the interchangeable parts, which are described below.

As the data source, an OIEP can either use a static selection of hierarchy root nodes, or events tied to selected objects in STEP. The delivery options and export formats are the same regardless of the data source selected. By default, this list will be the same as for the Export Manager.



Business Rules Event Processing

For event-based endpoints, business conditions can be used to filter events and business actions can be used to generate new events based on others. For more information, refer to the **Event Filter** and **Generate Event** sections of the **OIEP - Event-Based - Event Triggering Definitions Tab** documentation.

Processing Engine / Converter

The processing engine is responsible for performing the actual data export based either on events or a static selection. STEP Exporter is the only available option with the core functionality, and is also the standard functionality used for manual exports. If a special format not supported in the standard STEP Exporter is required, the processing engine can be replaced via an extension. Most often, however, creating a converter for

the STEP Exporter is sufficient for such cases. This approach also has the benefit that the same converter is available for manual exports. For more information, refer to the **Configure Endpoint** documentation for an **Event-Based OIEP** or a **Select Objects OIEP**.

Error Handling & Reporting

STEP includes a single Send Error Report option to report errors. An email can be sent to specified addresses if errors and/or warning occur when files or messages are handled by the processing engine. For more information, refer to the **Error Handling & Reporting** section of the **OIEP - Configuration Flipper** documentation.

Post-Processor

A post-processor has access to the file delivered by the processing engine and can manipulate it and/or split it into multiple files. STEP includes two post-processors, both used for manipulating files output by the processing engine with cross context information. For more information, refer to the **Output Templates Flipper** documentation for an **Event-Based OIEP** or a **Select Objects OIEP**.

Delivery

A delivery method has access to the file(s) output from the processing engine / post-processor. Additional delivery options can be added with the delivery plugin extension. For more information, refer to the **OIEP - Delivery Method Flipper** documentation.

Additional Information About Event-Based OIEPs

This information is relevant to understanding additional functionality available for implementing and using an event-based outbound integration endpoint. Included are details on the following features:

- Understand the effects of batching on an OIEP as described in **Event-Based OIEP Event Batching**.
- Understand the effects of manipulating events on an OIEP as described in **Event-Based OIEP Event Actions**.
- Understand the multithreading functionality on an OIEP as described in **Event-Based OIEP Multithreading Support**.
- Review the logical overview and process flow available for an OIEP as described in **Event-Based OIEP Overview and Process Flow**.
- Understand queued events on an OIEP as described in **Event-Based OIEP Queued Events**.
- Understand the relationship between OIEP Status and Queue Status as described in **Event-Based OIEP Status and Queue Status**.
- View some examples of event-based OIEPs as described in **Event-Based OIEP Examples**.

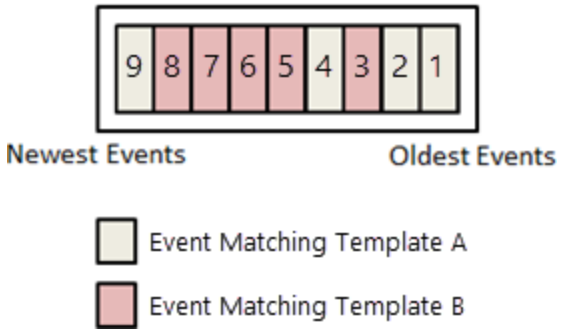
Event-Based OIEP Event Batching

The Event Batching option on an event-based OIEP does not determine the number of objects in a message, but rather, the number of events that will be processed and included in a single message. STEP does not wait for the number of events to occur before generating a message, instead, it caps the number of events in a single message so that 2+ messages are generated if more events are present than the batch number allows when the system is polled.

Unless it is a firm requirement that one file should be exported per event, common setup is to use event batching and leave the default value of 1000, or set it even higher if many events are registered and the endpoint is invoked infrequently. Appropriate batch size is typically based on the size of messages and downstream system processing capabilities, for example, use a smaller batch size for larger messages and a larger batch size for smaller messages.

STEP batches are based on the output template, which includes a combination of event type and object type. For more information, refer to **OIEP - Event-Based - Output Templates Flipper** documentation.

Consider the unread events in this illustration, which matches either 'Template A' or 'Template B' output templates:



If you work with a batch size of three (3), data related to the events will be published as follows:

- One file with data for Event 1 & 2 (Template A)
- One file with data for Event 3 (Template B)
- One file with data for Event 4 (Template A)
- One file with data for Event 5, 6 & 7 (Template B)
- One file with data for Event 8 (Template B)
- One file with data for Event 9 (Template A)

For more information, refer to the **Events** section of the **System Setup** documentation.

Event-Based OIEP Examples

This section includes event-based OIEP configuration examples for semi-advanced to advanced cases:

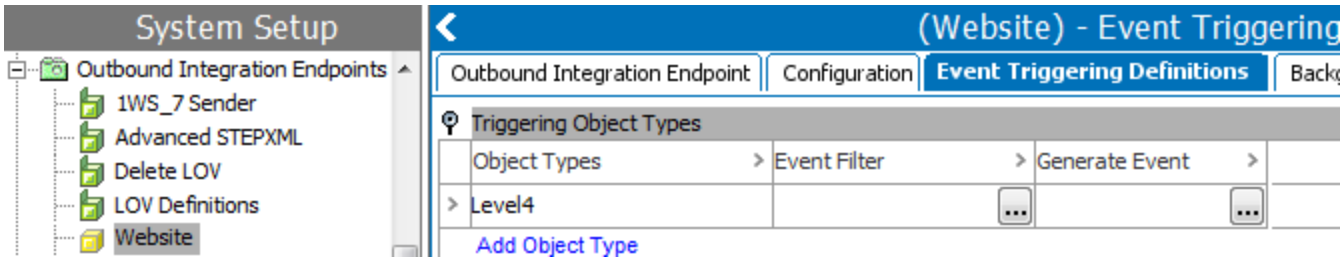
- Event-Based Example Business Rules for Derived Events
- Event-Based Example Publishing of Leaf Products with Inherited Data
- Event-Based Example Upward Inheritance
- Event-Based Example Products Linked to Classification

Event-Based Example Business Rules for Derived Events

Problem: A different output format is required for each derived event type.

Solution: Use JavaScript-based business rules in an event-based OIEP to:

- Create a business condition to filter out (discard) Republish events.
- Create a business action to generate derived events when products are linked, removed, or modified in a specific classification hierarchy and distinguishes if the products have been created, modified, or deleted in the classification structure.
- Apply the business rules to the OIEP on the Event Triggering Definitions tab, in the Triggering Object Types table.



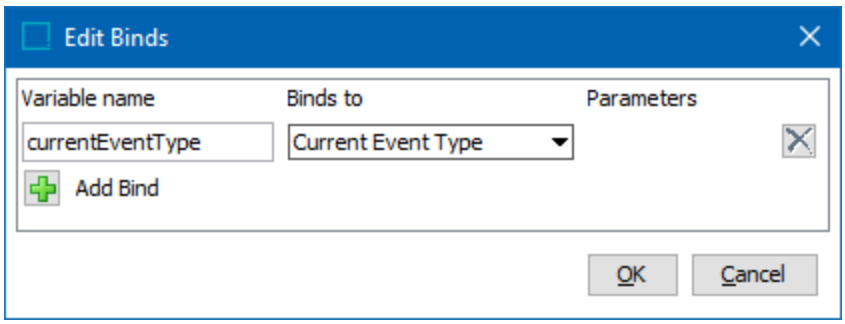
Important: The following scripts are only an example and should not be used as-is without thorough testing, including updating to match object and link types that exist on your system.

The following derived event types are used in this example:

- A product is linked into a classification and approved for the first time. In this case, a derived event with the event type 'website create' is generated.
- A product is approved and linked into a classification, but data on the product is approved with modifications. In this case, a derived event with event type 'website modify' is generated.
- A product is approved and linked into a classification, but approved with a deletion of the product to classification link. In this case, a derived event with event type 'website delete' is generated.

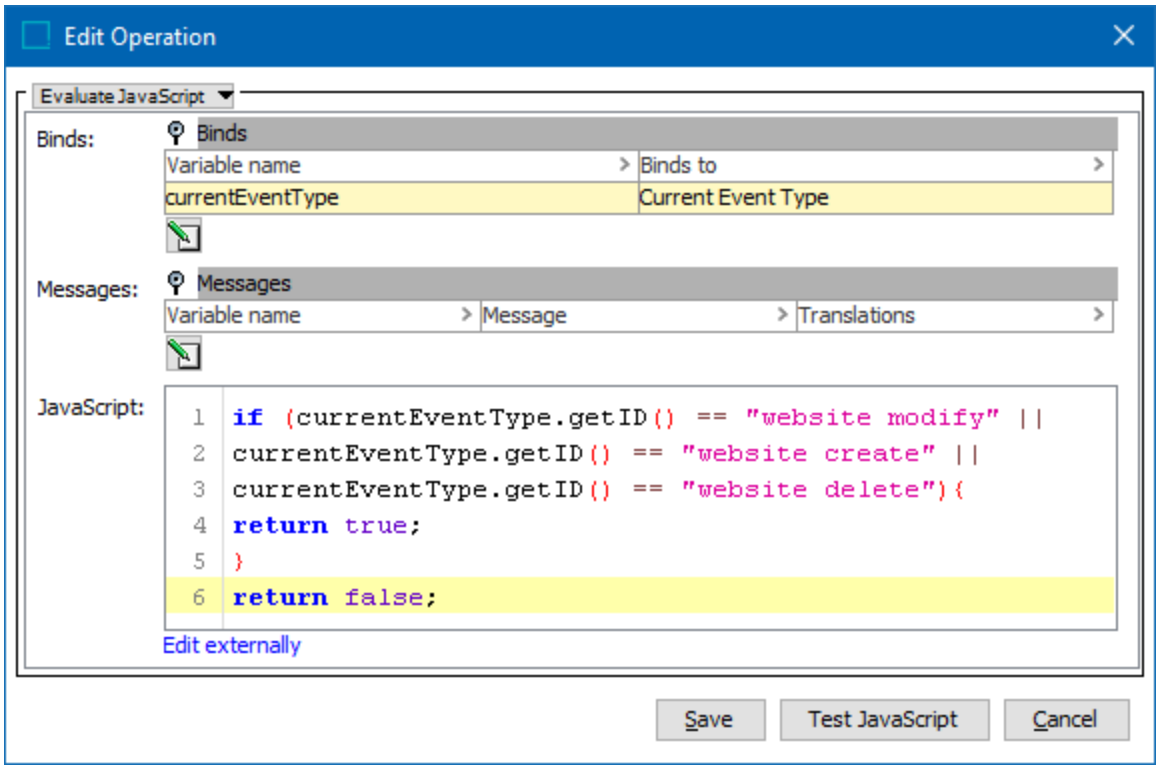
Event Filter - Using a bulk update or business action could generate Republish events, but they would be registered as core 'create' events. Instead, generate a derived event and then configure the relevant OIEP to discard all events that are not of this type.

Create a new **business condition** and make it valid for the appropriate object types. Use the **Evaluate JavaScript** operation and add a bind for the Current Event Type.



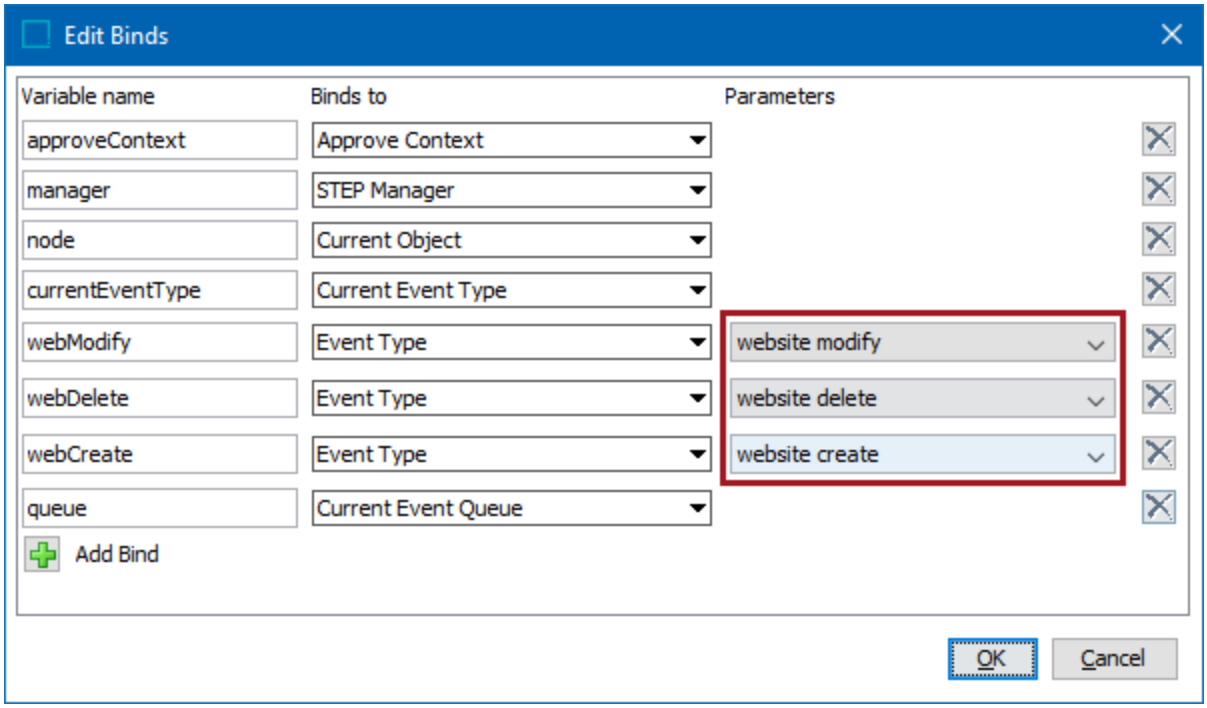
Use the following JavaScript code to identify the derived event types:

```
if (currentEventType.getID() == "website modify" ||
    currentEventType.getID() == "website create" ||
    currentEventType.getID() == "website delete"){
       return true;
}

return false;
```



Event Generator - Only standard derived event types can be used with JavaScript-based business actions. For additional information on JavaScript syntax for STEP, access the **STEP API Documentation**, available at [system]/sdk or from the Start Page.

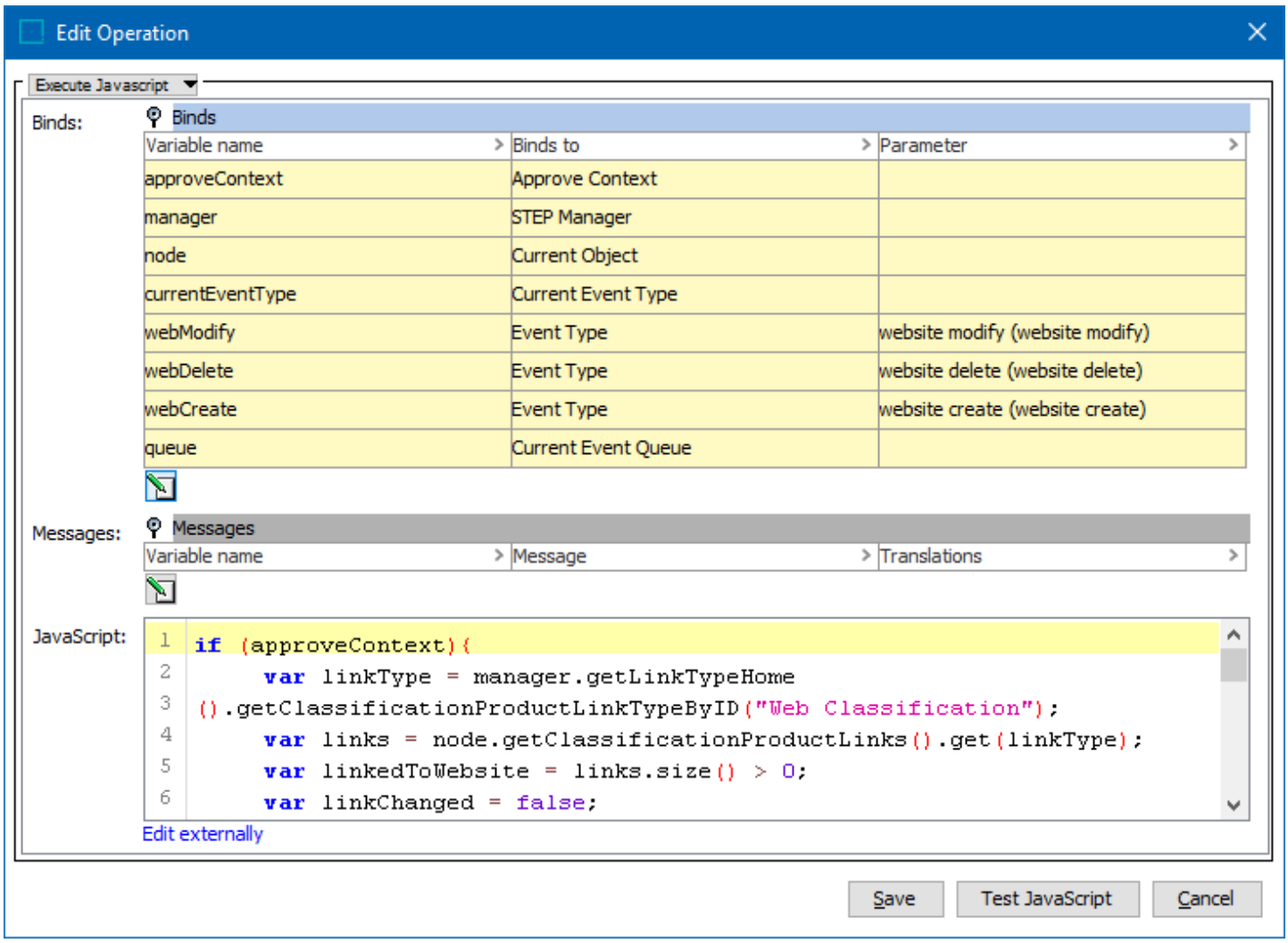
Create a new **business action** and make it valid for the appropriate object types. Use the **Execute JavaScript** operation and add the following binds, including binds for the derived events:



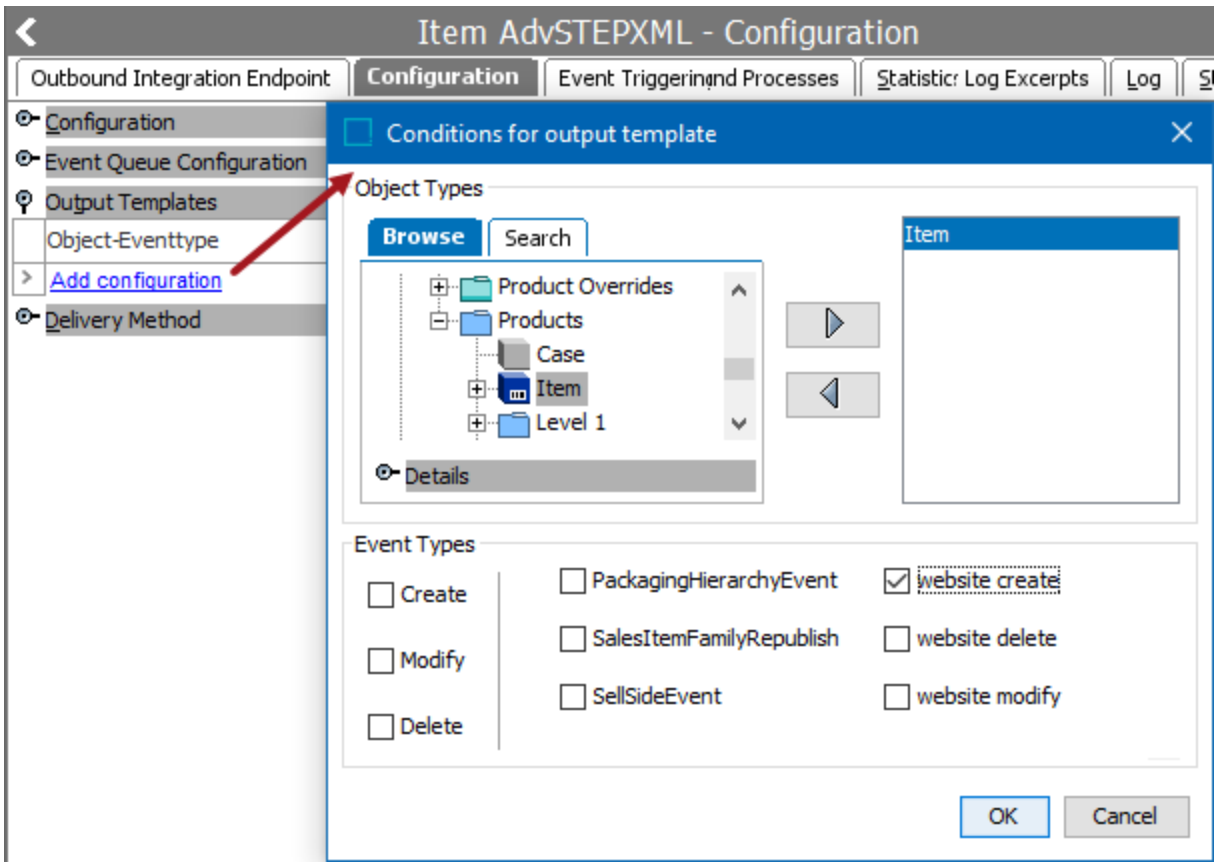
Use the following JavaScript code to create a derived event based on the approved change on the product:

```
    if (approveContext){
       var linkType = manager.getLinkTypeHome().getClassificationProductLinkTypeByID("Web Classification");

    var links = node.getClassificationProductLinks().get(linkType);
    var linkedToWebsite = links.size() > 0;
    var linkChanged = false;
    var partObjects = approveContext.getPartObjects();

           var iter = partObjects.iterator();
    while (iter.hasNext()) {

    var part = iter.next();

            if (part instanceof com.stibo.core.domain.partobject.ClassificationLinkPartObject){
```

```
            if (part.getLinkTypeID() == "Web Classification"){
                linkChanged = true;
            }
        }
        }

if (linkedToWebsite && linkChanged){
            queue.queueDerivedEvent(webCreate, node);
        } else if (!linkedToWebsite && linkChanged) {
            queue.queueDerivedEvent(webDelete, node);
        } else if (linkedToWebsite && !linkChanged){
            queue.queueDerived
Event(webModify, node);
        }
}
```



Output Template - Using the configuration described above, you will need an output template for each different type of event. This can be one or multiple lines, dependent on the need for different messages for the different events. Refer to **OIEP - Event-Based - Output Templates Flipper** topic for more information.


Item AdvSTEPXML - Configuration
Outbound Integration Endpoint
Configuration
Event Triggering and Processes
Statistic: Log Excerpts
Log
Configuration
Event Queue Configuration
Output Templates
Object-Eventtype
Add configuration
Delivery Method
Conditions for output template
Object Types
Browse
Search
Product Overrides
Products
Case
Item
Level 1
Details
Item
Event Types
Create
Modify
Delete
PackagingHierarchyEvent
SalesItemFamilyRepublish
SellSideEvent
website create
website delete
website modify
OK
Cancel

Event-Based Example Leaf Products with Inherited Data

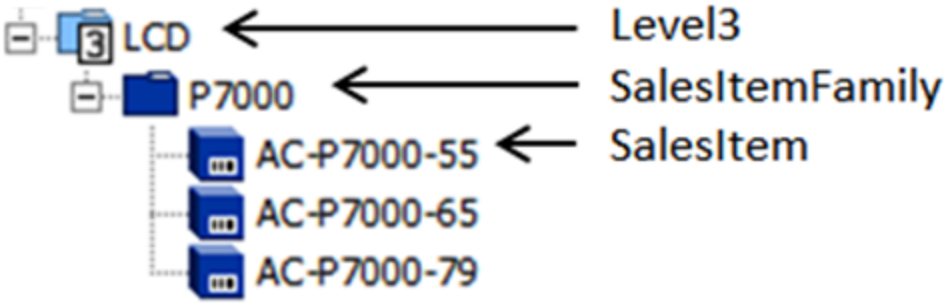
Problem: You only want to publish the leaf product level to a downstream system and the leaves inherit data from ancestors, so events will not be generated for the leaves when inherited data changes.

Solution: When values are inherited from the parent or grandparent level, use Event Filters and Event Generators in an event-based OIEP to ensure that:

- events generated for the parent or grandparent level are discarded
- derived events are generated for all descendant leaf level objects that exist in Approved mode (have been published earlier)

Important: The following scripts are only an example and should not be used as-is without thorough testing.

Using the following object types:



Use the following triggering object types setup:

Triggering Object Types

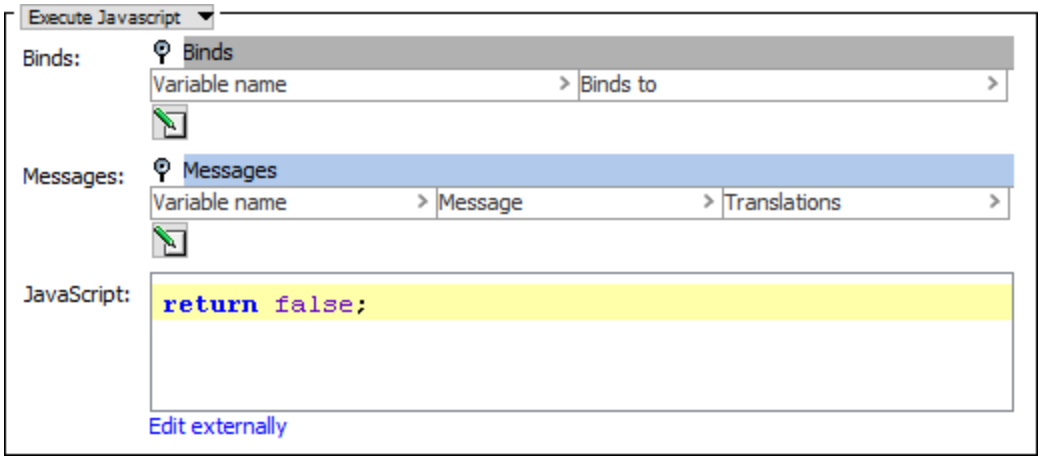
Object Types	Event Filter	Generate Event
Sales Item		
Level 3, Sales Item Family	LP Always False (LPAlwaysFalse)	LP Republish Sales Item Descendants (

Add Object Type

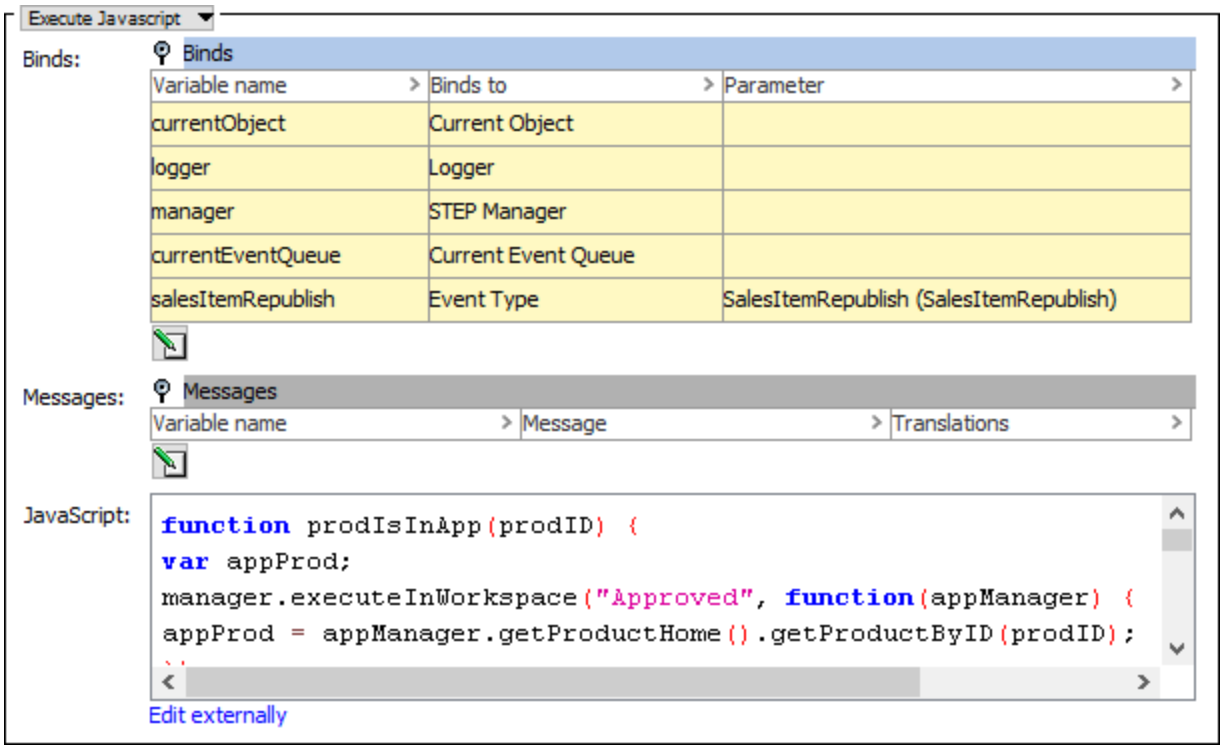
No filters or generators are used for Sales Items, so events generated for this object type always get registered in the queue.

Use the following logic for the grandparent and parent levels (Level 3 and Sales Item Family):

Event Filter - No Events are registered for objects of these object types, but we still need to catch them to be able to generate derived events for descendant Sales Items. The event filter should be a JavaScript business condition that always evaluates to false, as shown below:



Event Generator - The business action for event generation is a little more complex and uses the following binds (logger is optional):



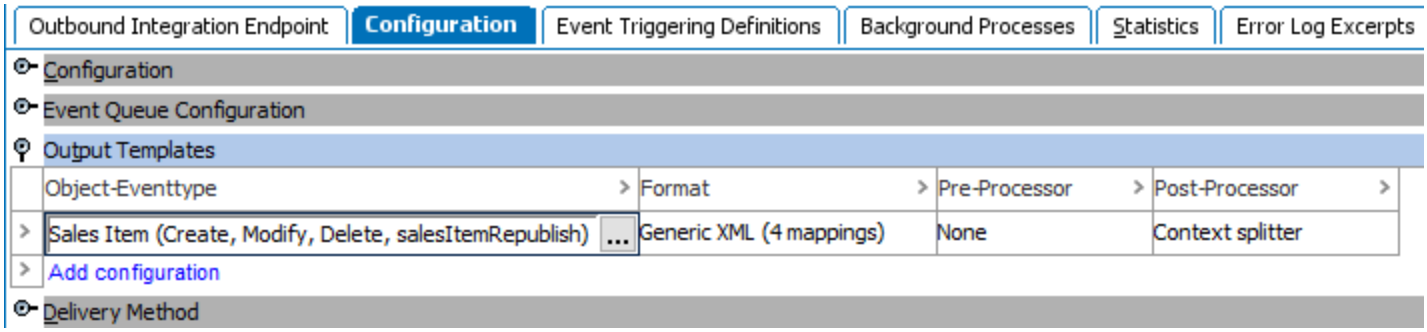
Notice the 'Event Type' bind to a derived event named 'salesItemRepublish.' This is the event that will be generated for descendant Sales Items and it must be defined in advance via the 'Derived Events' node in System Setup.

Important: The following script is only an example and should not be used as-is without thorough testing.

For a script example of the derived events approval, refer to the online version of this topic.

The script uses as recursive function 'generateEventForApprovedLeafSalesItems' to find descendent Sales items that are in the Approved workspace.

Output Template - Using the configuration described above, you will need an output template for Sales Items and the different types of possible events. This can be one or multiple lines dependent on the need for different messages for the different events.

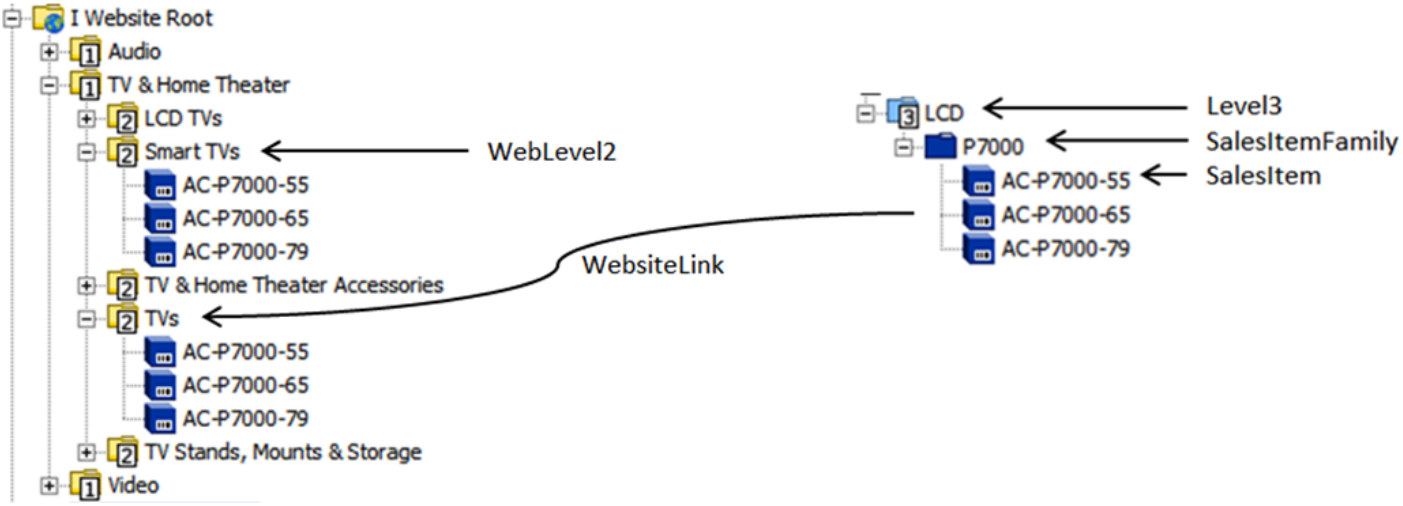


Event-Based Example Products Linked to Classification

Problem: Sometimes an external system is only interested in data about products linked into a specific classification hierarchy (e.g., a website hierarchy) and needs different types of messages depending on whether a product has just been linked into the hierarchy ('create' to the external system), is in the hierarchy and has been edited ('modify' to the external system) or has been taken out of the hierarchy ('remove / delete' to the external system).

Assumptions: For this example the setup shown below is used and a number of assumptions are made.

- Only leaf level (Sales Item) can be linked into the classification hierarchy and only leaves should be published.
- The Product to Classification Link Type ('WebsiteLink') is only used for this specific website and all Sales Item objects in Approved that have the link should be on the website.



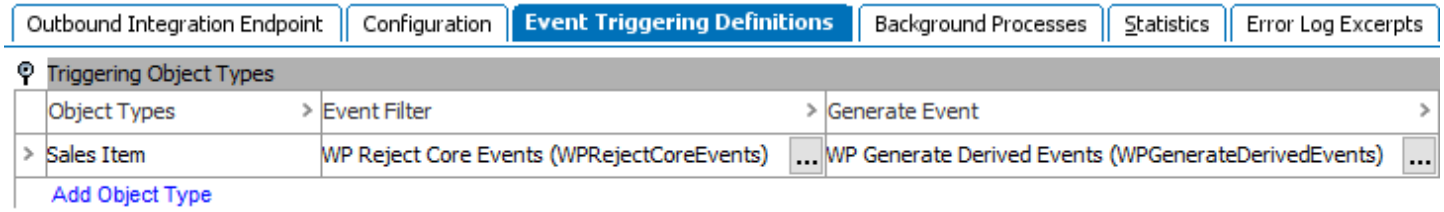
Solution: A naive approach to this problem could be to have an event filter condition for Sales Items that only let events for the objects with a WebsiteLink pass. The problem with this approach is that it would not allow you to detect the cases where a link has just been removed, so you would not be able to communicate this to the external system. Also, you would not be able to differentiate between cases where the link has just been added ('create' to an external system) and cases where a Sales Item already linked into the classification hierarchy has been modified.

A better approach is to use three derived events with IDs 'webAdd', 'webModify', and 'webRemove' in an event-based OIEP. The conditions for triggering the different events are listed below:

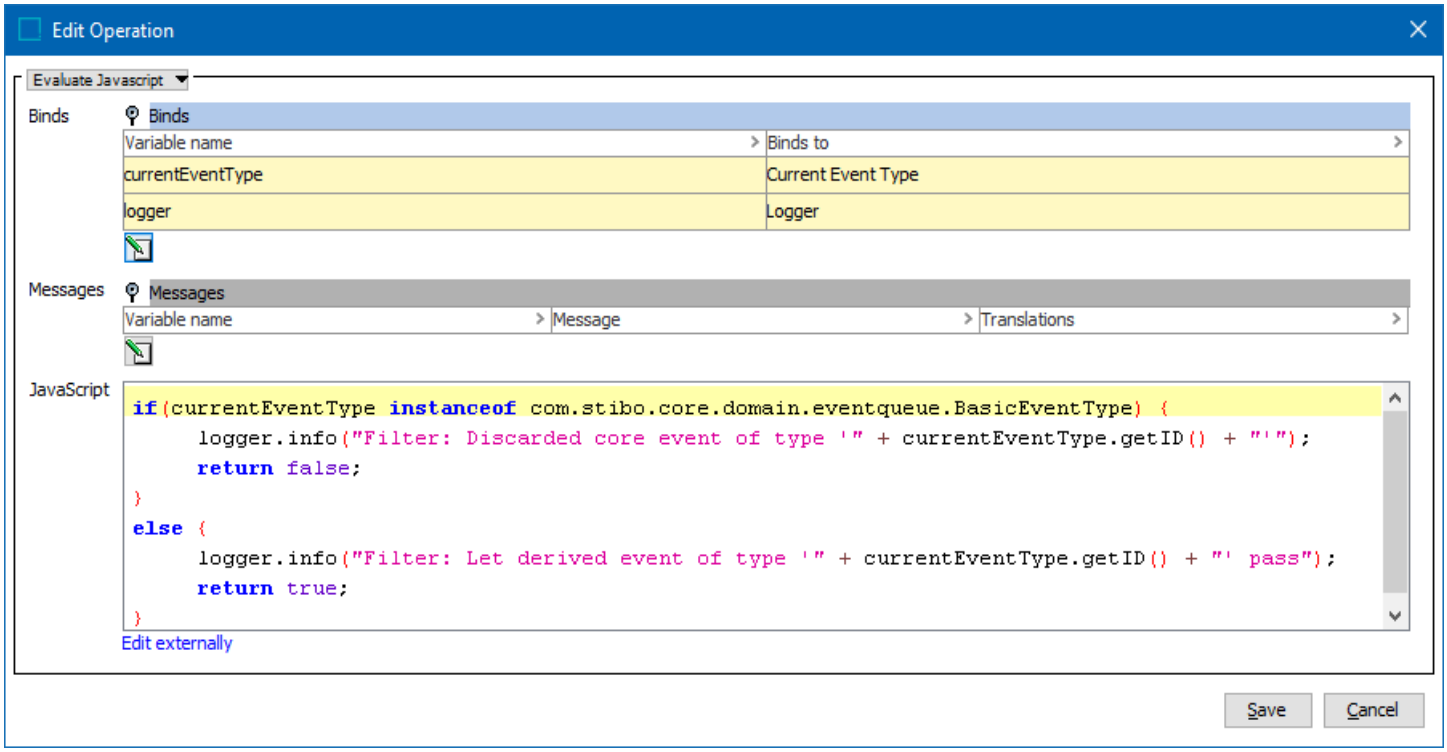
- **webAdd** - Sales Item with WebsiteLink is approved for the first time; earlier approved Sales Item where WebsiteLink has been added is approved
- **webModify** - Sales Item with WebsiteLink has been modified and is approved; externally maintained data for Sales Item with WebsiteLink in Approved is modified
- **webRemove** - Earlier approved Sales Item where WebsiteLink has been removed is approved; Sales Item with WebsiteLink is approve deleted (core Delete Event)

Important: The following scripts are only an example and should not be used as-is without thorough testing.

A setup for this case could use the following triggering object types:



Event Filter - As data will be published based on derived events alone, all core events for Sales Items should be discarded. The script below does this and also, for debugging purposes, logs to the main Java log.

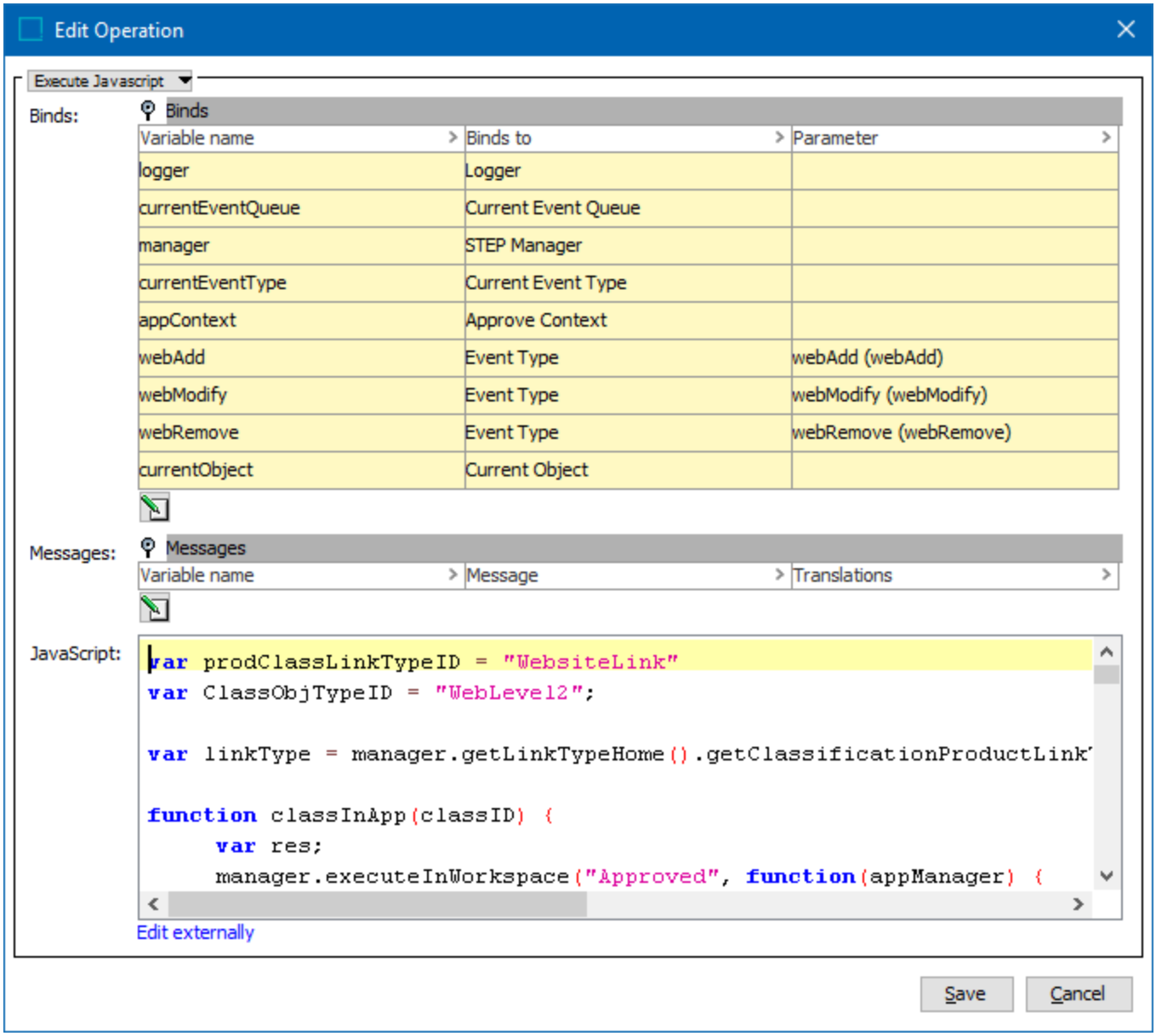


If logging is not required, the script could be:

```
return !(currentEventType instanceof
com.stibo.core.domain.eventqueue.BasicEventType);
```

Event Generator - This logic is much more complex as it will have to perform the analysis to figure out which type of derived event should be generated. An example without logging is shown below. Notice that the business action should only generate derived events based on core events. If this check is not in place, an endless loop will be created.

The script uses the following bindings:



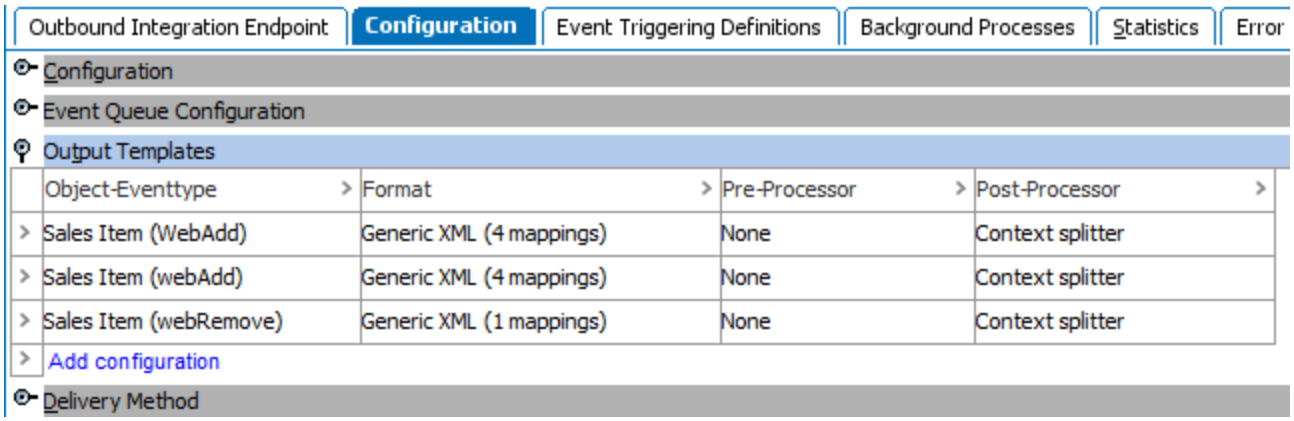
Keeping in mind that the following script is strictly to illustrate the solution, and can be difficult to read, this list outlines the intended logic:

- Check if current event is a core event, do nothing if it is a derived event.
- Check if the Approve Context is available. If so, the event is generated based on an approval. If not, the event is based on some other change, for example, externally maintained data being modified.
- If Approve Context is available, use it to get hold of current object in both Main and Approved.
- Check if current core event is a Delete.
- Check for WebsiteLinks in Main and Approved.
- If current event is a Delete, and current object has a WebsiteLink in Approved, generate a derived 'webRemove' event.

- If current object has WebsiteLink in both Main and Approved, generate a derived 'webModify' event.
- If none of the above, the Approve Context is used to figure out whether the addition or removal of a WebsiteLink is currently being approved (is a part object). The link type is not accessible directly from a ClassificationLinkPartObject and thus, the target classification object type is used to figure out if the link type is correct. Also, because there is a risk that the target classification does not exist in the Approved workspace and the link would not be synchronized even though it is in the part object set, it is checked that the classification exists there.
- If WebsiteLink exists in Main and is to be approved, generate a derived 'webAdd' event.
- If WebsiteLink exists in Approved and is to be approved, generate a derived 'webRemove' event.
- If no Approve Context is available, check whether current object exists in Approved with WebsiteLink. If it does, generate derived 'webModify' event.

Refer to the online help version of this topic for the example script.

Output Template - If different messages are to be output dependent on the selected derived event type, three output templates should be configured as shown below. This can be one or multiple lines, depending on the need for different messages for the different events.



Event-Based Example Upward Inheritance

Problem: For outbound integrations where not only the leaf level is published, sometimes data from children is used on parents, for instance, via calculated attributes. When the children change, the parents are not automatically republished.

For comparison, downward inheritance is typically handled by publishing all object descendants.

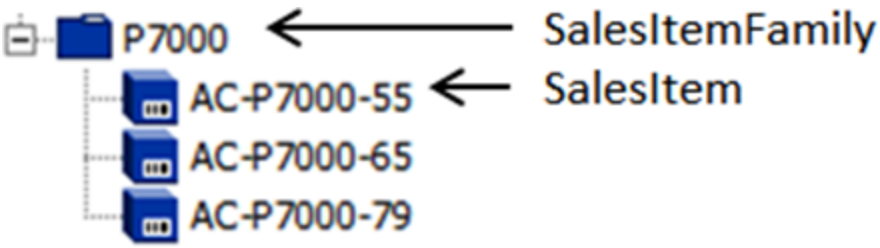
Solution:

In an event-based OIEP:

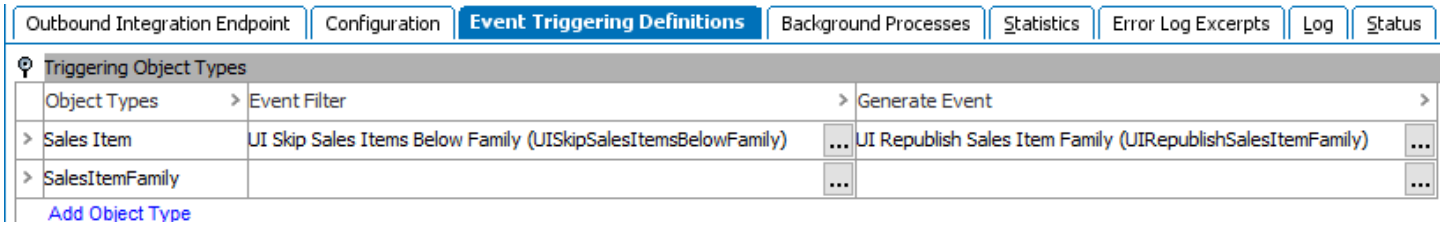
- If children get exported along with the parent, whenever an event is generated for a child with appropriate parent, catch it, discard it, and generate a derived event for the parent instead.
- If children are not published with parents, do not discard the children events.

Important: The following scripts are only an example and should not be used as-is without thorough testing.

The first option is described below, using the following object types:



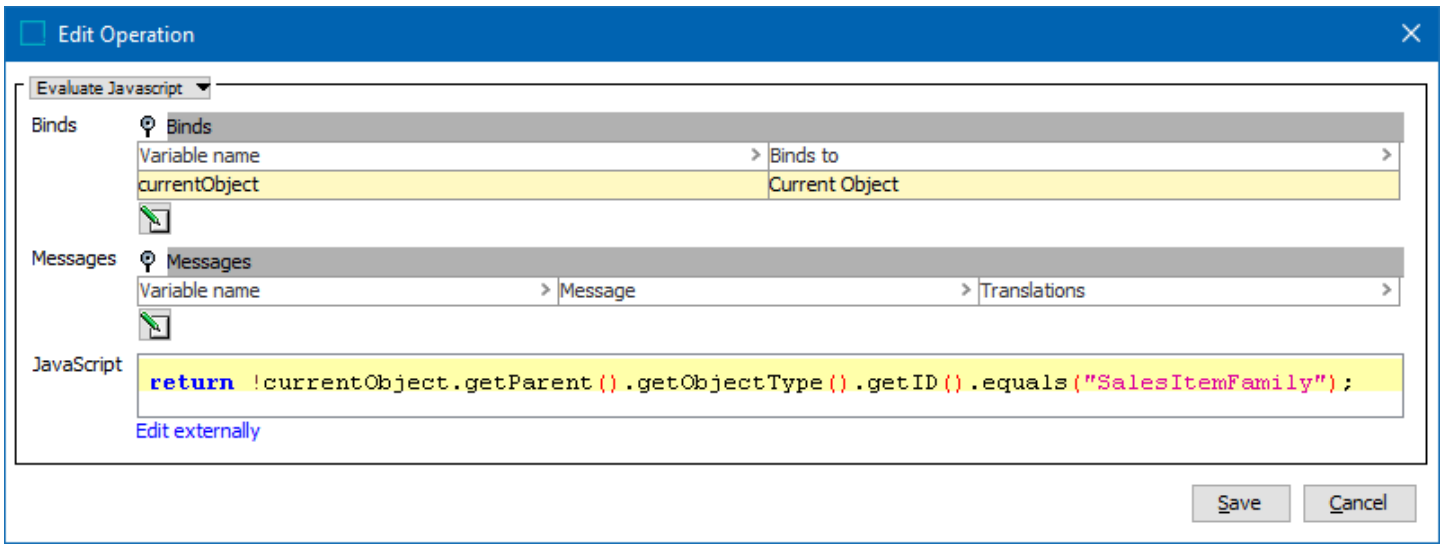
Use the following triggering object types setup:



No Filter or Generator is used for the Sales Item Family Object Type as all Events should pass for this type.

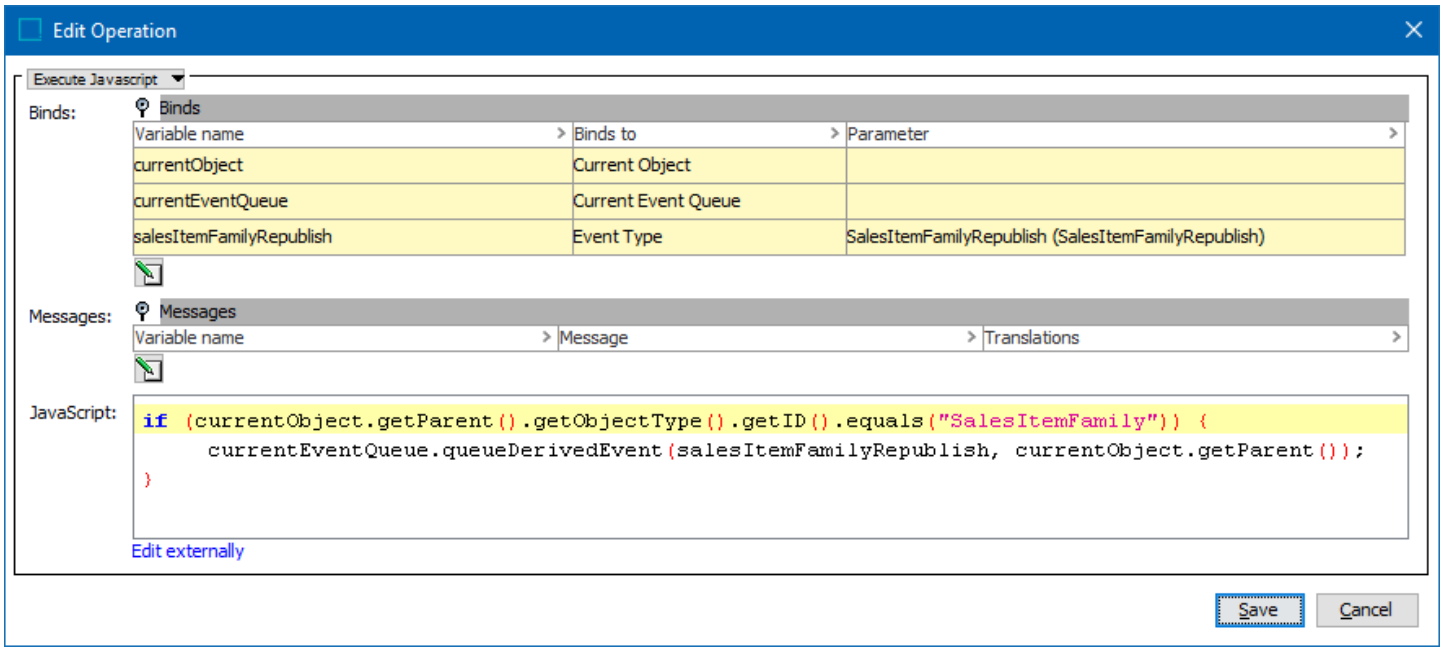
Use the following logic for the Sales Items:

Event Filter - If Sales Items can be children of both Sales Item Family and other types, a filter that checks the parent object type is required. The Script below discards events for Sales Items that are children of Sales Item Family objects:



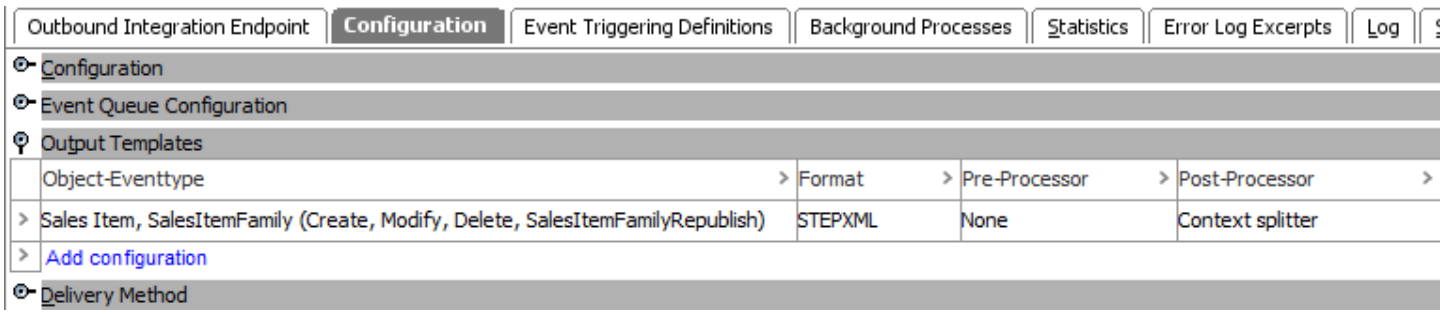
If Sales Items can only exist below Sales Item Family objects, instead an 'Always False' condition like the one used in **OIEP Example Publishing of Leaf Products with Inherited Data** can be used instead.

Event Generator - This business action script is also fairly straight forward, notice that it is assumed that a 'Derived Event' with ID 'salesItemFamilyRepublish' has already been defined in STEP.



If Sales Items can only exist below Sales Item Family objects, the 'if' statement can be left out.

Output Template - Using the configuration described above, you will need one or more output templates that cover both Sales item and Sales Item Family, and the different types of possible events must be defined. Below, a single output template is used.

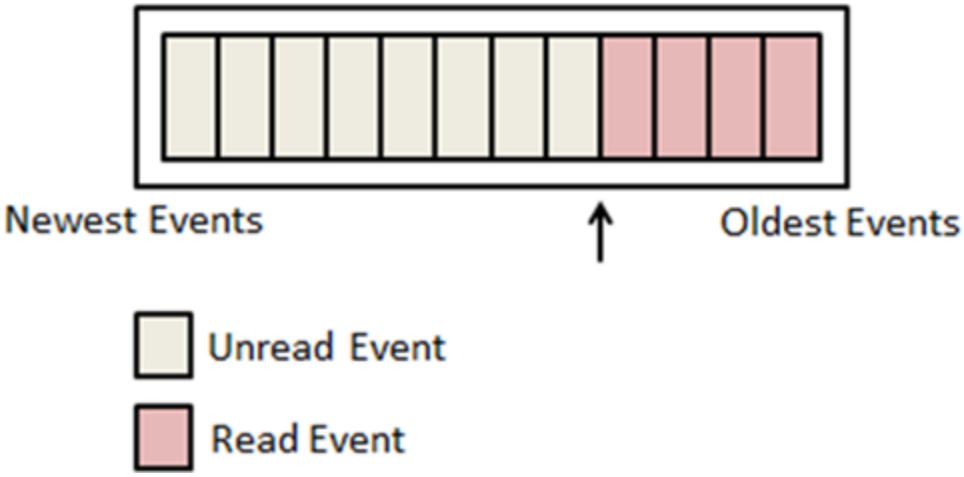

Outbound Integration Endpoint
Configuration
Event Triggering Definitions
Background Processes
Statistics
Error Log Excerpts
Log
Configuration
Event Queue Configuration
Output Templates
Object-Eventtype
Format
Pre-Processor
Post-Processor
Sales Item, SalesItemFamily (Create, Modify, Delete, SalesItemFamilyRepublish)
STEPXML
None
Context splitter
Add configuration
Delivery Method

Event-Based OIEP Event Actions

Creating an event-based OIEP automatically creates an associated event queue. After configuring the OIEP with Event Triggering Definitions, the OIEP then listens for events that match the configuration and sends those to the related queue. The OIEP polls the queue per the schedule, handles the events, marks them 'read', and publishes the data based on output templates.

The 'Days to Retain Events' parameter determines how long events are retained in the queue (when they are removed from the queue) after being read. Common setup for this parameter is between 5 and 10 days. When the setting is > 0, read events stay in the queue for the specified number of days. Retaining events allows you to republish them again should the receiving system lose messages. To do so, rewind the event queue and reprocess already processed events from a specific point in time. For information on setting the 'Days to Retain Events' parameter, refer to the **OIEP - Event-Based - Event Queue Configuration Flipper** documentation.

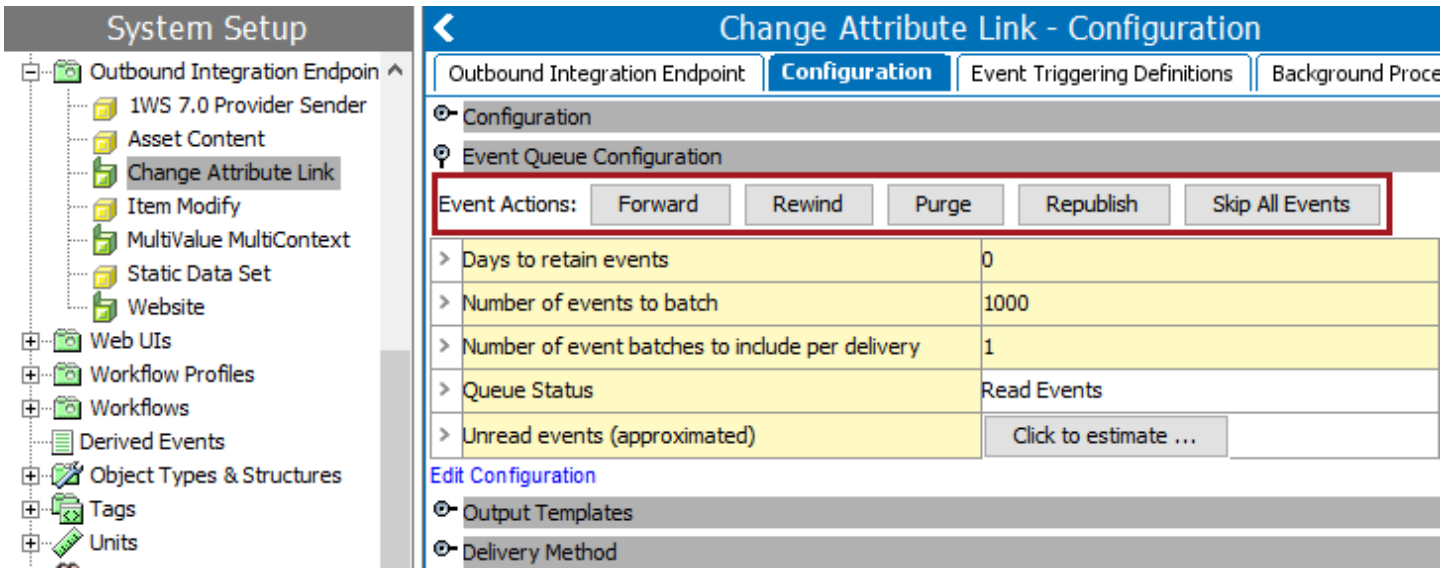
For example, if the queue is configured to retain events after they have been read, read events can be distinguished from unread events, as illustrated by the arrow in the following image.



The Forward and Rewind actions essentially lets the user move this arrow:

- Forward action marks unread Events as read (moving the pointer left)
- Rewind action marks read Events as unread (moving the pointer right)

On the OIEP Configuration tab, open the Event Queue Configuration flipper and use the buttons to modify the queue's events.



For information on the number and types of unread events, refer to the **Event-Based OIEP Queued Events** documentation.

Generating Events from Main

By default, events are triggered on the Approved workspace. Derived event functionality is available for triggering events prior to approval, as defined in the **Derived Events** topic of the **System Setup** documentation.

Forward Events

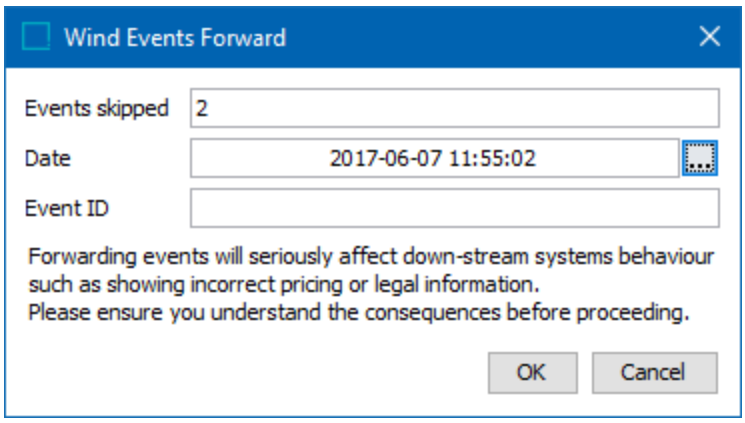
The Forward action requires that unread events exist in the queue. This option is typically used when the unread events in the queue should not be published to downstream systems. For example, consider that you performed a system refresh or data maintenance activity that generated events.

Important: Use caution when forwarding unread events as it may affect the downstream system's data, since the previous changes would be ignored by this activity.

1. In the **Wind Events Forward** dialog, click the ellipsis button (...) in the **Date** field.
2. In the **Date Picker**, select the date and time from which you want to skip events.

   When you have selected a date and time, the **Events skipped** field shows the number of events that will be skipped.

   Note: Event IDs are not shown in the UI but are read from the database, making this a troubleshooting option.



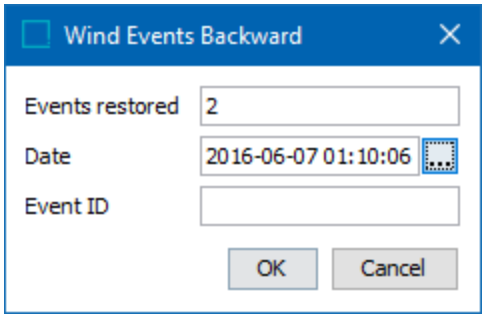
Rewind Events

The Rewind action is only available when the parameter 'Days to Retain Events' is set to a number greater than zero during creation or configuration of the OIEP. This option is typically used when a downstream system did not receive updates from the OIEP, there was a problem with file delivery, or any failure of the OIEP.

1. In **Wind Events Backward** dialog, click the ellipsis button (...) in the **Date** field.
2. In the **Date Picker**, select the date and time from which you want to restore events.

   When you have selected a date and time, the **Events restored** field shows the number of events that will be restored.

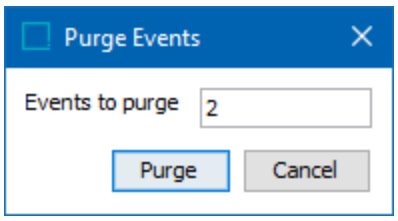
> **Note:** Event IDs are not shown in the UI but are read from the database, making this a troubleshooting option.



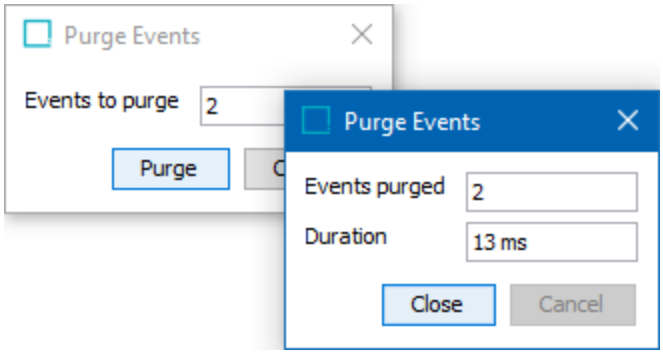
Purge Events

The Purge action is only available when the parameter 'Days to Retain Events' is set to a number greater than zero during creation or configuration of the OIEP and there are events in the queue and read events are older than the 'Days to Retain Events' parameter setting. This option is typically used when the unread events are expired, or when the events queued by the OIEP should not be published downstream.

1. In **Event Actions**, click **Purge**.
2. The Purge Events dialog, the number of **read** events that you can delete from the system is displayed.



3. Click **Purge** to display the events purged along with duration to purge those.

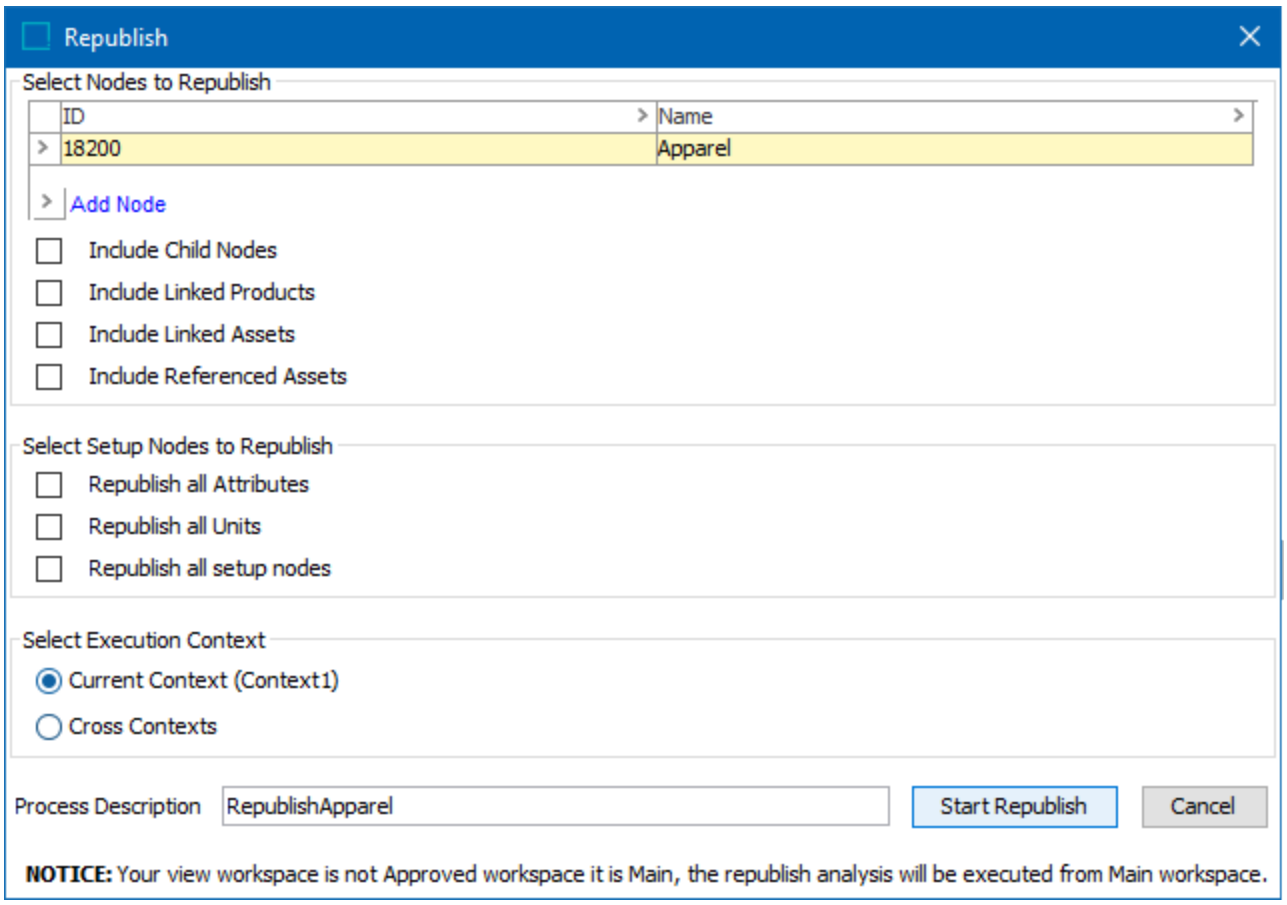


Note: When the number of read events in Purge Events shows 0, forward events to the current date and attempt to purge.

Republish Events

The Republish action generates events on products, classifications, or assets in one or more selected hierarchies. Common setup is to use this option for on-demand creation of republish events. For example, to generate events for all objects in a hierarchy that has never been published by an event queue, or events that were published but the downstream system did not accurately receive the data.

1. In **Event Actions**, click **Republish** to display the Republish dialog.



2. Click **Add node** and then browse or search for the hierarchy you want to republish.
3. Select the objects you want to republish. You have the following options:
   - **Include Child Nodes** republishes all children of the selected object.
   - **Include Linked Products** republishes all products linked to the selected object.
   - **Include Linked Assets** republishes all assets linked to the selected object.
   - **Include Referenced Assets** republishes all assets referenced by the selected object.
4. Select the setup nodes you want to republish. You have the following options:
   - **Republish All Attributes** republishes all attributes linked to the selected object.
   - **Republish all Units** republishes all units linked to the selected object.
   - **Republish all setup nodes** republishes all system setup objects.
5. Select the Execution Context. You have the following options:

- Select **Current Context** to republish in the current context. (shown in parenthesis)
- Select **Cross Contexts** to republish child nodes in all contexts.

6. In **Process Description** enter a name for the process and activate the Start Republish button. This is a mandatory field.
7. Click **Start Republish**. Events are triggered on objects that match the selection.

Republish Manually

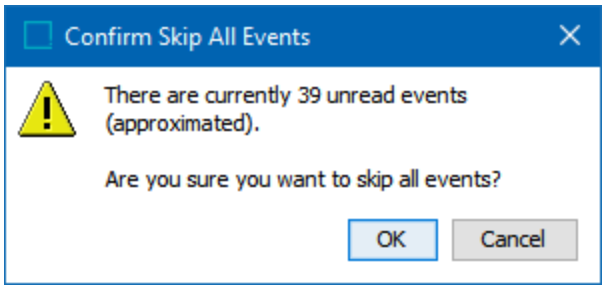
Additionally, republish events can be generated manually in STEP Workbench as follows:

- From a collection using the Republish option. For more information, refer to the **Republish** section of the **Maintaining Collections** topic in the **Getting Started** documentation.
- Via a business action using the Send Republish Event option, which is available from a business action operation. For more information, refer to the **Business Action: Send Republish Event** topic in the **Business Rules** documentation.
- Via Bulk Update using the Send Republish Event operation when the 'matching' add-on component is activated. For more information, refer to the **Send Republish Event Operation** topic in the **Bulk Updates** documentation.

Skip All Events

The Skip All Events action passes over all events without processing and removes them from the event-based outbound integration endpoints.

Click **OK** to remove all events or click **Cancel** to leave the events unchanged.



Event-Based OIEP Multithreading Support

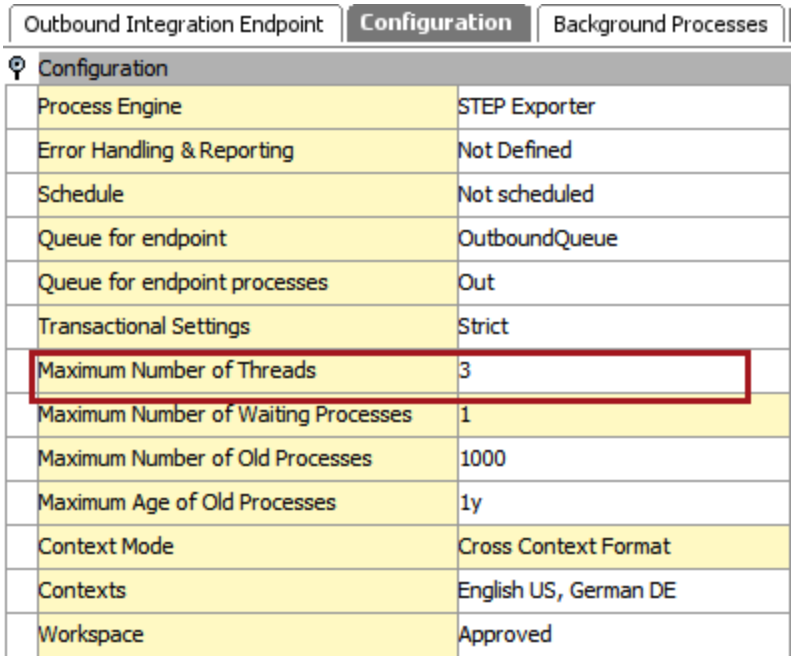
To increase performance when exporting from an event-based outbound integration endpoint (OIEP) you can increase the number of threads for a given OIEP. Common setup is to use multithreading when there is a large amount of data going to a downstream system on a regular basis, and when the downstream system is capable of handling it.

Consider the following points before increasing the number of threads to more than one:

- The STEP system hardware should have enough resources to perform with multithreading.
- The downstream (receiving) system must be able to handle parallel events.
- Multithreading can negatively impact performance when a large amount of data is involved.
- Run the settings in a test environment, starting with a small amount of data and then increasing it, before implementing it in your production system.

Multithreading Setup

Although the **Number of threads** setting is available on the Configuration tab for both static and event-based OIEPs, the setting **only** affects event-based processes.



The default thread setting is one (1), in which case the endpoint produces a single message at a time, with all events in the batch processed serially. Increasing the thread number results in each batch size being divided by the thread number so that the contents of a batch can be processed in parallel.

Since multiple event batches per message are not supported for multiple threads, when the 'Number of threads' is greater than one (1), then the 'Number of event batches to include per delivery' (shown in the image below) is automatically set to one (1). The following warning is included in the BGP execution report and also in the

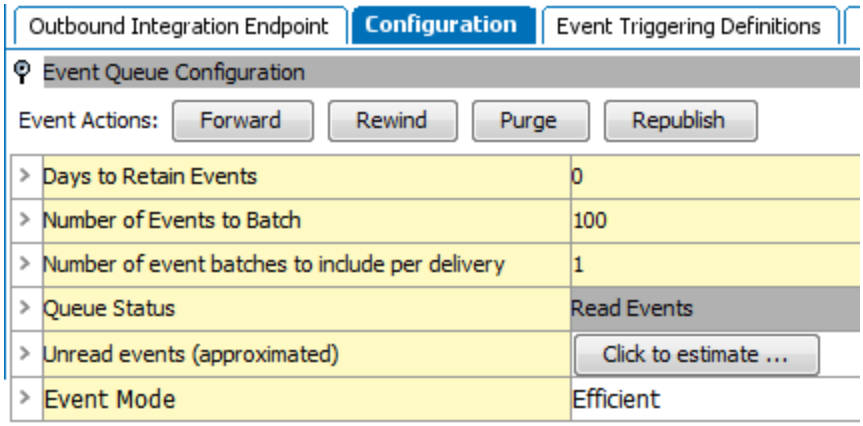
application server log: Number of threads are [x]. Number of batches per delivery will be set to 1 for integration endpoint [ID] as this is the only allowed value when multiple threads are used. Requested number of batches were [y].

When all threads are complete, the batch yields a single message, as is consistent with the 'Strict' transactional setting required by event-based OIEPs. Thread size is typically increased for particularly critical information where the speed with which the message is produced is key. However, increasing the setting beyond the capabilities of the hardware will impede the overall performance of the application server, so care must be taken in adjusting the thread size.

Additional Considerations

Increasing the Batch Size

When increasing the number of threads running, the size of the batch will decrease proportionally (e.g., a batch size of 100 using two threads will be split into two batches of 50). When adjusting the number of threads, it is therefore worthwhile to experiment with increasing the batch size, in order for each working thread to have a reasonable amount of data to process.



Serialized and Parallelized

The batch-fetching of the events is run serially and the data distributed as evenly as possible to each thread. From fetching the batch on through the delivery stage, the pre-processing, main processing, and post-processing take place in parallel. Following post-processing (if applicable), the data are handed over to the delivery stage which is again run serially. Events are therefore delivered as if everything was executed serially. With this in mind, it is important to consider the schedule of the endpoint and ensure that it is set to check for events as frequently as if the endpoint were single-threaded.

Note that when a batch is processed in parallel (via multiple threads), it is recorded in the execution report of the endpoint.

Event-Based OIEP Order of Delete Events

The order of processing delete events via an event-based OIEP is guaranteed in the following cases:

- For hierarchies (products, classifications, etc.), the deletion of a child occurs before deleting the parent.
- For references, the deletion of the reference occurs before the target can be deleted. A source owns a reference, which means triggering a delete event on the source must be completed before deleting the target.

Impact of Transactions

The method used to create transactions also determines if each transaction generates its own event, or if multiple transactions are collected to create a single event. For more information about transactions, refer to the **Database Transaction Methods** section of the **Core Events** topic in the **System Setup** documentation.

The order of events being processed is influenced by the generation and commitment of transactions.

- If the same user makes multiple deletions for the same object from the same Web UI, the deletions order is guaranteed.
- If the same user makes multiple deletions for different unrelated objects from the same Web UI, the deletion order is not guaranteed.

Unrelated Object Event Order Processing

When working with unrelated objects, processing occurs based on the order events are read from the queue. This means that the order for processing delete events for unrelated objects can vary.

For example, consider the following time line and results for products in different hierarchies and without any references between them.

Time Line

The following two users log into their own Web UI and make the following edits:

- 8:00 a.m. - User 1 logs into Web UI 1 and deletes Product A
- 8:15 a.m. - User 2 logs into Web UI 2 and deletes Product Z
- 10:00 a.m. - User 1 continues to use Web UI 1 and makes changes to Product B

Results

Since two users are making the edits, the deleted products are not related, and considering the transaction impact (defined below), the processing order of the events could be:

- If the queue is read at 9:00 a.m., the processing could start with the deletion of Product Z, and then the deletion of Product A.
- If the queue is read at 11:00 a.m., the processing could start with the deletion of Product A, then the deletion of Product Z, and finally the changes to Product B.

Event-Based OIEP Overview and Process Flow

Refer to the online help version of this topic for a logical overview and process flow chart describing key features of an event-based outbound integration endpoint (OIEP) configuration.

For more information, refer to:

- Business Actions (BA) and Business Conditions (BC), as defined in the **Business Rules** topic
- Events, Core Events, and Derived Events, as defined in the **Events** topic
- OIEP, as defined in the **Creating an Event-Based Outbound Integration Endpoint** topic
- Event Triggers, Event Generator, and Event Filter, as defined in the **OIEP - Event-Based - Event Triggering Definitions Tab** topic
- Event Queue and Event Batch, as defined in the **OIEP - Event-Based - Event Queue Configuration Flipper** topic
- Output Template, Event Types, Object Types, Pre-processor, Post-processor, as defined in the **OIEP - Event-Based - Output Templates Flipper** topic
- Message size per Output Template, as defined in the **Event-Based OIEP Event Batching** topic
- Message Processor, as defined in the **OIEP - Configuration Flipper for Business Rule Based Message Processor** topic

Event-Based OIEP Queued Events

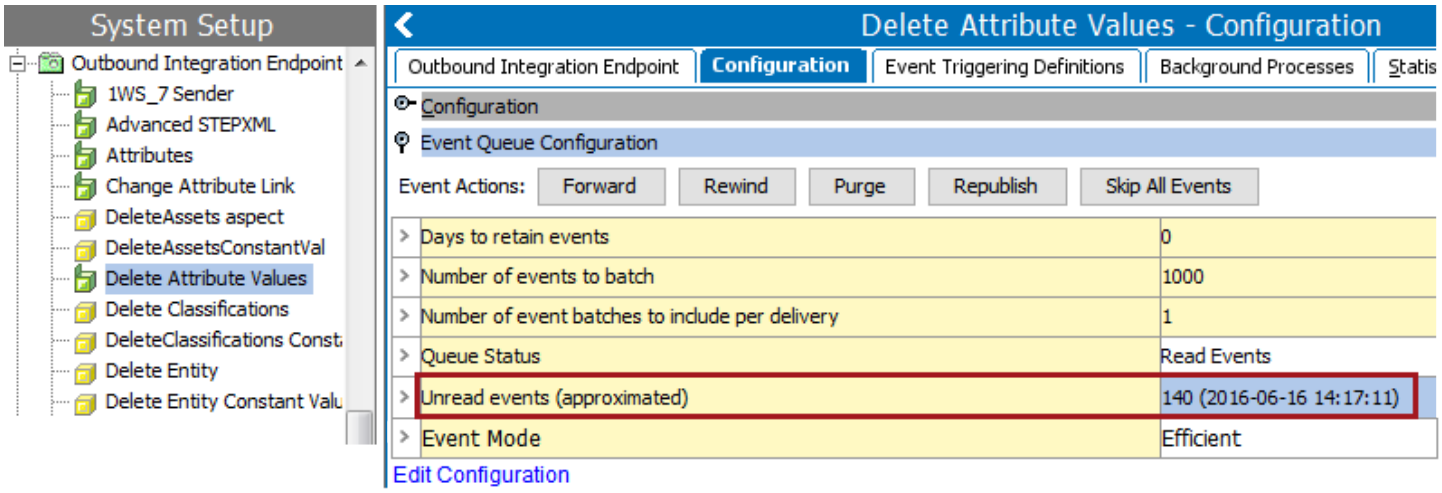
As events are found that meet the criteria set in the event-based OIEP's configuration and event triggering definitions, an estimate and details are available for review in System Setup.

Estimate Unread Events

An estimate of the current number of unread events for triggered objects in the Event Queue Configuration area 'Unread events (approximated)' parameter.

On the OIEP's Configuration tab, Click the 'Click to estimate' button to display the data.

The 'Click to estimate' button is replaced with the number of events and the time of the estimate.

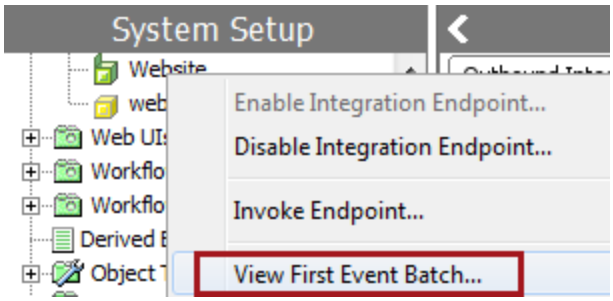


Using the forward, rewind, purge, or republish events options causes the number of unread events on the queue to change. For more information, refer to **Event-Based OIEP Event Actions**.

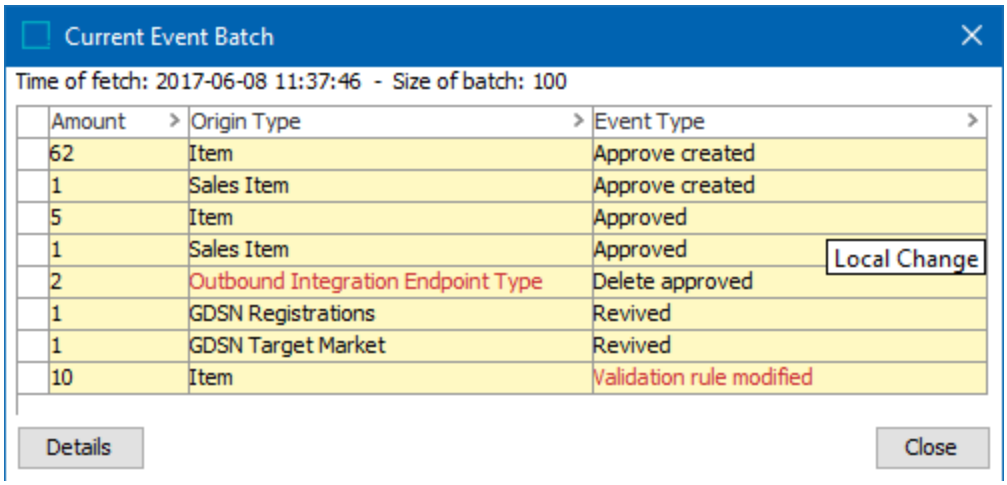
View First Event Batch

This option allows you to view more than just a count of the events.

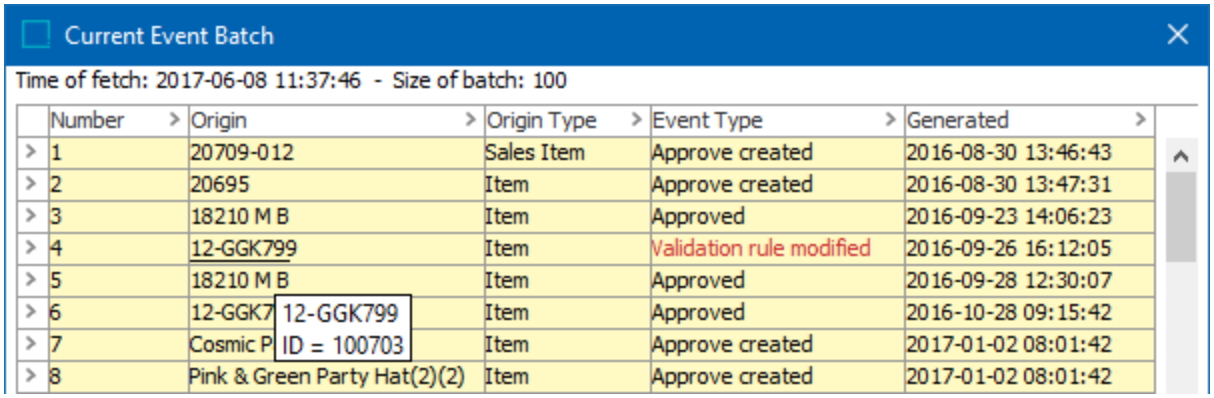
1. Right-click the OIEP and select View First Event Batch.



2. The Current Event Batch dialog displays the first 100 unread events on the queue.

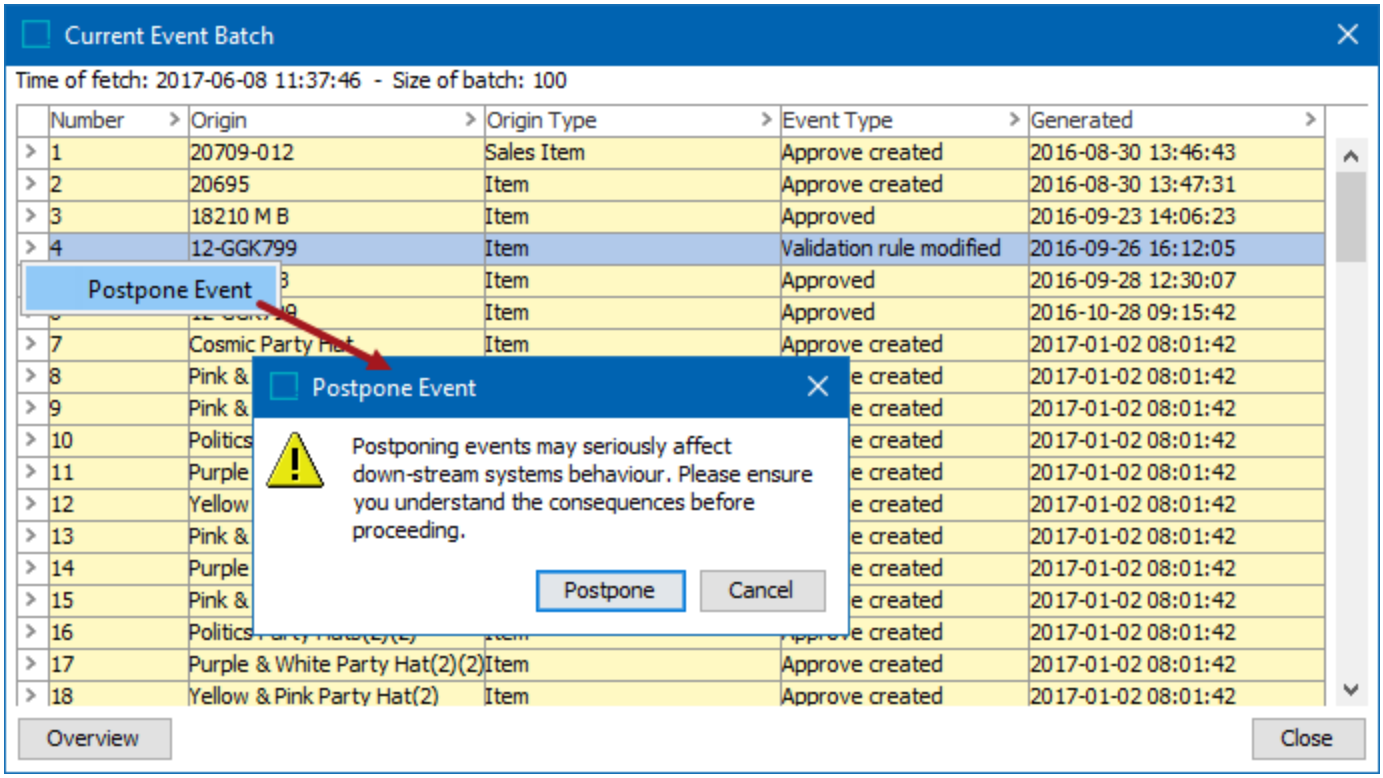


3. For more information about the events, click the **Details** button. The Origin column includes a link to the object being reported by the event.



4. To postpone an event until the end of the queue, select a row and right-click on the > column, and click **Postpone Event**.

Important: Events can rely on data included in previous messages. Reordering the sequence may cause issues in the receiving system and should be run in a test environment first when possible. Carefully consider the warning displayed after clicking the Postpone Event option allows you to cancel the action.



5. Click the **Overview** button to return to the previous view.

Event-Based OIEP Status and Queue Status

For event-based integration endpoint, the OIEP status (enabled / disabled) setting and the queue status (read events / discard events) setting are independent of each other, but work together to determine how data is processed.

This means that, for example, if a downstream system is offline but receives deliveries from an OIEP, disabling the endpoint stops publishing data, but allows the OIEP to continue registering events in STEP so that data can be published once the external system is back online.

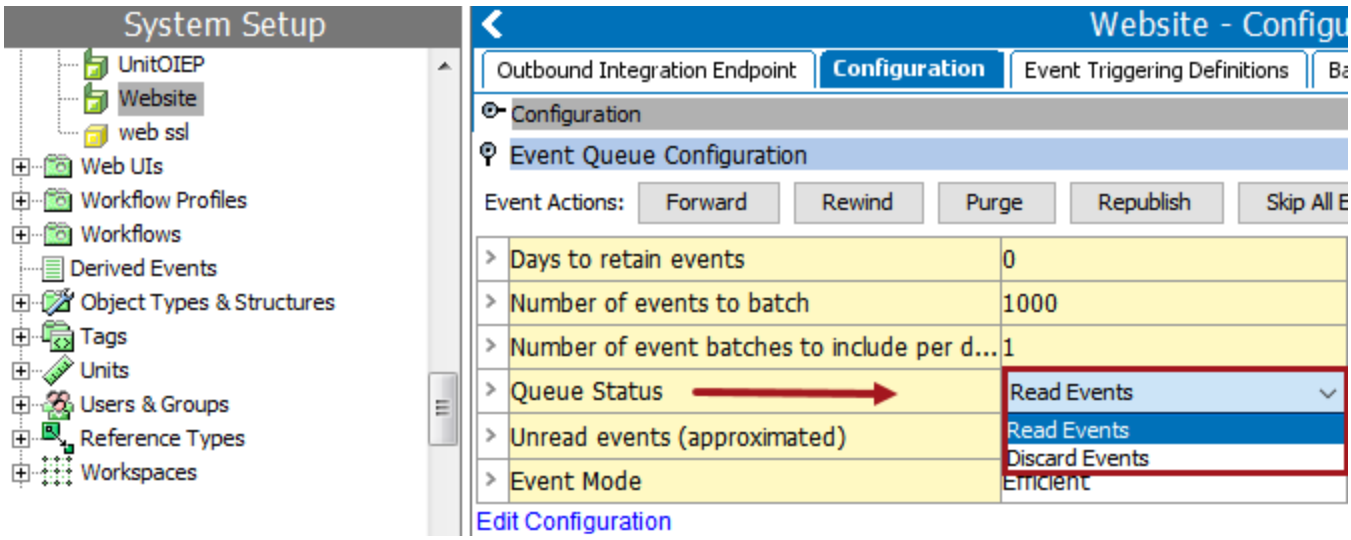
Common setup includes the following combination of queue status and endpoint status settings:

- **Enabled + Read Events = Active**. Use this setting for an active endpoint that should deliver data to downstream systems.
- **Disabled + Read Events = Paused**. Use this setting to temporarily disable the feed, while retaining access to events being generated, even while the endpoint is disabled.
- **Disabled + Discard Events = Inactive**. Use this setting when no new events should be processed (now or later) and data should not be delivered downstream.
- **Enabled + Discard Events = Transition**. Not commonly used, but can be employed when one endpoint will take over for another, or prior to running a bulk update process that should not be sent downstream. Allows the old endpoint to process queued events, but not generate any new ones as new events should be set to queue on the new endpoint (or discarded if bulk update should not be sent out).

Note: By default when a new OIEP is created and configured, the OIEP status is set to Disabled and Queue Status is set to Discard events.

Set the Queue Status

1. In System Setup, locate an event-based outbound integration endpoint and view the **Configuration** tab.
2. Open the **Event Queue Configuration** flipper.
3. In **Queue Status**, select the desired option.
   - **Read Events** allows the OIEP to register events as they occur, based on the OIEP configuration and event triggering definitions.
   - **Discard Events** ignores generated events that meet the OIEP configuration and event triggering definitions.



For information about setting the enabled / disabled parameter, refer to the **Running an Outbound Integration Endpoint**.

HTTP Proxy Configurations

Within STEP, several HTTP-based outbound delivery methods for outbound integration endpoints (OIEPs), receiver methods for inbound integration endpoints (IIEPs), and gateway integration endpoints (GIEPs), include the ability to add HTTP proxy configuration settings via a dropdown parameter. The HTTP proxy configuration option, when applicable, provides an extra layer of security and privacy for users that require it.

The endpoints that incorporate this option include:

- Microsoft Azure Blob Storage (GIEP)
- Amazon S3 Blob Storage (GIEP)
- Product Data Exchange (IIEP & OIEP)
- REST (OIEP)
- REST Direct (OIEP)
- SalesforceREST (GIEP)
- Salesforce Merge Delivery (OIEP)
- Salesforce Receiver (IIEP)
- Salesforce Update Delivery (OIEP)

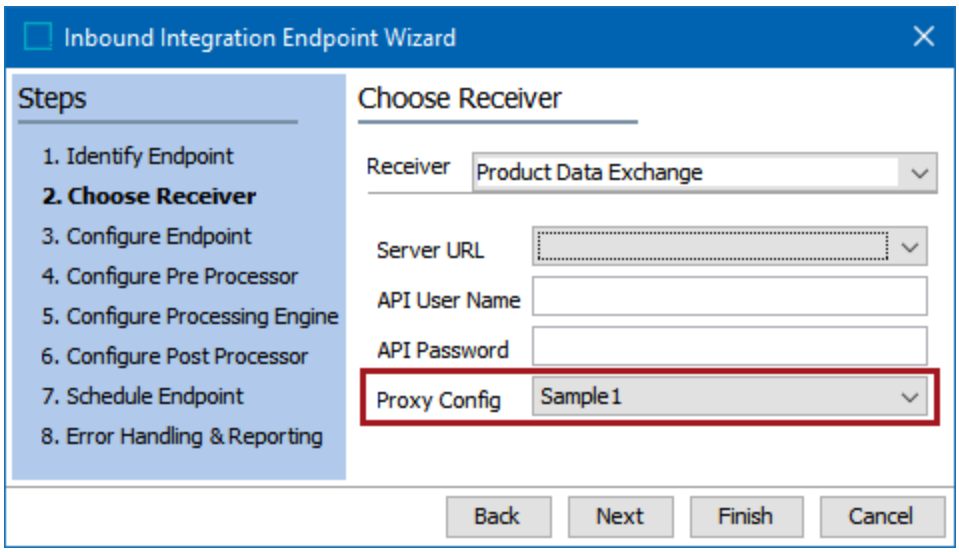
Defining HTTP Proxy Configurations

HTTP proxy configurations are defined via properties saved in the sharedconfig.properties file. Multiple HTTP proxy configurations can be made by giving them different names. Each HTTP proxy configuration has its own property for Host, Port, User, and Password.

For example, an HTTP proxy configuration with the name 'Sample1' can be defined as follows:

```
Http.ProxyConfiguration.Sample1.Host=localhost
Http.ProxyConfiguration.Sample1.Port=8080
Http.ProxyConfiguration.Sample1.User=user
Http.ProxyConfiguration.Sample1.Password=password
```

In the image below, 'Sample1' is the selected HTTP proxy configuration for the Product Data Exchange IIEP receiver.



For more information regarding:

- The sharedconfig.properties file, refer to the **Configuration** topic in the **Administration** documentation.
- OIEP delivery methods, refer to the **OIEP - Delivery Method Flipper** topic.
- IIEP receiver methods, refer to the **Receiver Methods** section of the **IIEP - Choose Receiver** topic.
- GIEPs, refer to the **Gateway Integration Endpoints** topic.

Additional information regarding HTTP proxy configurations can be found on the web.

Mutual Transport Layer Security

STEP supports Transport Layer Security (TLS) with or without mutual authentication (mTLS) for outbound HTTP traffic. Since mTLS requires trusted certificates from both the server and the client, mTLS can be used instead of basic authentication for additional security.

STEP supports inbound mTLS on the Stibo Systems SaaS environment. Inbound mTLS is possible for on premise systems but requires custom development.

Outbound mTLS

This functionality is controlled via a set of configuration properties for the key store and trust store to use across all features. These configurations must be set via case-sensitive properties in sharedconfig.properties file on the STEP application server.

mTLS functionality has been tested with the gateway integration endpoint REST plugin, the REST and REST Direct outbound integration endpoint delivery plugins, and URLConnections directly from business rule JavaScript.

Note: The properties starting with SSL.* do not work in combination with the case-sensitive legacy properties `RESTDeliverySSLKeyStoreLocation`, `RESTDeliverySSLKeyStorePassword`, `RESTDeliverySSLKeyStoreType`, and `RESTGateway.SSLTrustStoreLocation`. SSL.Default properties do not apply to endpoints configured using the legacy properties. Refer to the following section for details on using the SSL.* properties.

mTLS Configuration Properties

The following table lists the default case-sensitive configuration properties for mTLS and their descriptions. The 'SSL. Default.*' properties are the global settings that are applied in OIEP and GIEP configurations where trust store parameters are not available or cannot be edited. To use multiple certificates, refer to the **Using Multiple Certificates** section below.

Note: The case-sensitive configuration property `SSL.Default.TrustStore.Location` supersedes the case-sensitive legacy properties `RESTGateway.SSLTrustStoreLocation` and `RESTDeliverySSLKeyStoreLocation` (there is no period before SSL for this property), and also when using TLS without mutual authentication. The legacy properties were used in releases prior to 11.0.

Restart the server after making changes to the SSL.* properties.

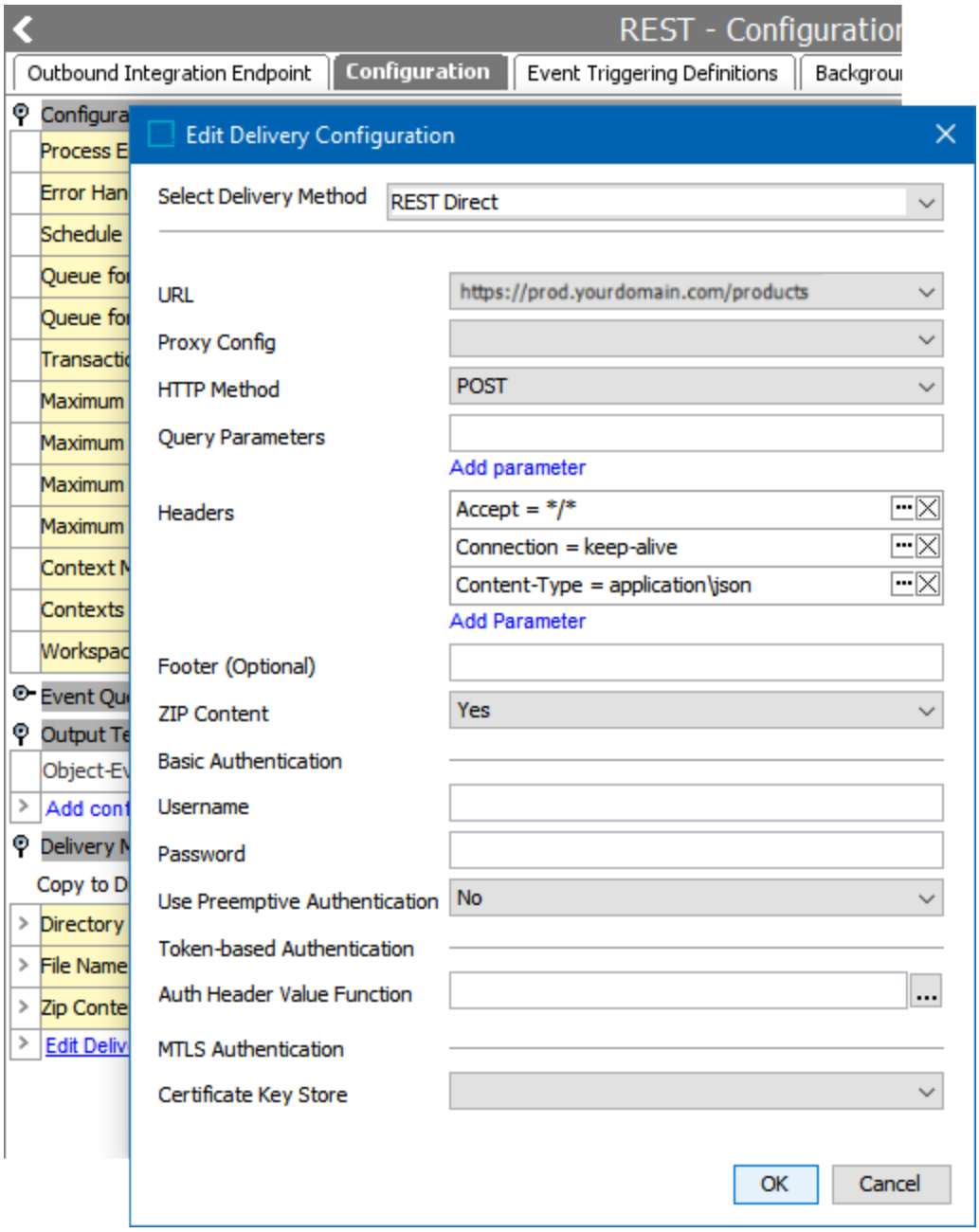
Configuration Property	Description
`SSL.Default.KeyStore.Location`	Full path to the key store file in the file system. For clusters, the key store must be in a directory accessible from all application servers.
`SSL.Default.KeyStore.Password`	Password for the keystore.
`SSL.Default.KeyStore.Type`	The type of keystore. This defaults to 'JKS.'
`SSL.Default.TrustStore.Location`	Full path to the trust store file in the file system. For clusters, the truststore must be in a directory accessible from all application servers.
`SSL.Default.TrustStore.Password`	Password for the truststore.
`SSL.Default.TrustStore.Type`	The type of truststore. This defaults to 'JKS'.

When using TLS without mutual authentication (using a self-signed certificate), no key store is required and only the trust store configuration properties must be set. If no trust store is configured, the connection is established using a certificate signed by a recognized Certification Authority, such as Verisign or Thawte.

As an example, the following properties would work together to provide options for selection in the Configuration tab of an OIEP with REST Direct delivery or a GIEP with REST:

```
SSL.Default.KeyStore.Location=/shared/customer-config/keystore/keystore.jks
SSL.Default.KeyStore.Password=password
```

In this image, below the 'MTLS Authentication' heading, the 'Certificate Key Store' parameter is empty since a default key store has been configured.



To learn more about the GIEP REST plugin, refer to the Configuring a Gateway Integration Endpoint - REST topic in the 'Gateway Integration Endpoint' section of the Data Exchange documentation.

To learn more about the OIEP REST Direct delivery method plugin, refer to the REST Direct Delivery Method topic in the 'OIEP Delivery Methods' section of the Data Exchange documentation.

Using Multiple Certificates

More than one certificate / keystore is useful when each integration requires a different certificate or integrating with multiple systems that use different CNames (the STEP system name as defined by another system). As shown in the following table, add multiple certificates / keystores in the sharedconfig.properties file on the application server after the certificates have been created. An alias, indicated by '[certificate]' in the table, is used to identify a specific certificate.

Note: A system restart is required when adding certificates or when changing property values in the sharedconfig.properties file.

Configuration Property	Description
`SSL.[certificate].KeyStore.Password`	Password for the keystore. For the [certificate] text, add a unique identifier, for example, 'Cert1'. This case-sensitive text is also required to implement the related location property, RestDirect property, and RestGateway property.
`SSL.[certificate].KeyStore.Location`	Full path to the key store file in the file system. For clusters, the key store must be in a directory accessible from all application servers. For the [certificate] text, add a unique identifier, for example, 'Cert1'. This case-sensitive text is also required to implement the related password property, RestDirect property, and RestGateway property.
`RestDirect.Mtls.CertificateKeyStores`	Keystores available for mTLS with the REST Direct delivery method.
`RestGateway.Mtls.CertificateKeyStores`	Keystores available for mTLS with REST gateway integration endpoints.

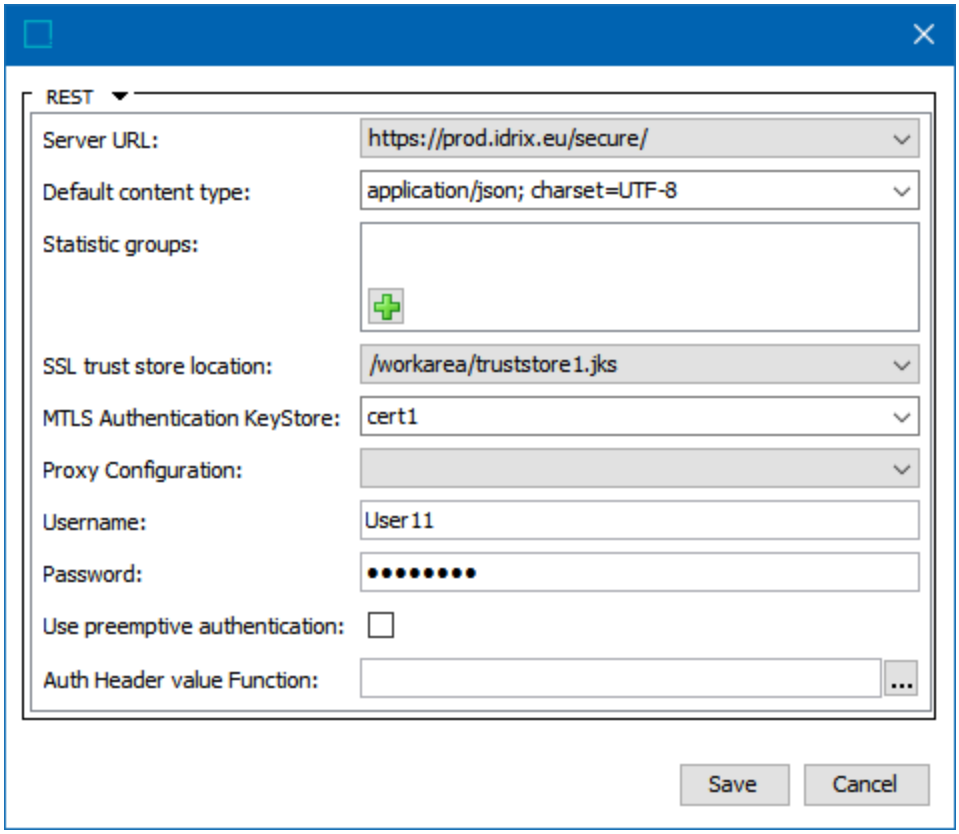
Important: The legacy properties do not apply to endpoints configured to use one of the keystores defined using the SSL.[alias].KeyStore properties.

As an example, the following properties would work together to provide options for selection in the Configuration tab of an OIEP or GIEP:

```
SSL.Default.KeyStore.Password=password
SSL.Default.KeyStore.Location=/shared/customer-config/keystore/keystore.jks

SSL.Cert1.KeyStore.Location=/shared/customer-config/keystore/Cert1.jks
SSL.Cert1.KeyStore.Password=password
SSL.Cert2.KeyStore.Location=/shared/customer-config/keystore/Cert2.jks
```

```
SSL.Cert2.KeyStore.Password=password

RestDirect.Mtls.CertificateKeyStores=,Cert1,Cert2
RestDirectDeliveryURL=1=https://prod.idrix.eu/secure/,2=https://yourDeliveryUrl.com
RestGateway.Mtls.CertificateKeyStores=,Cert1,Cert2
RESTGateway.ServerURL=1=https://prod.idrix.eu/secure/,2=https://yourDeliveryUrl.com
```

Adding a leading comma to the 'CertificateKeyStores' properties (as shown above) allows for selecting no certificate via an empty value in the parameter dropdown. When the empty value is selected in the dropdown, it indicates 'no certificate', but if a default mTLS configuration exists, that default is used. If no default is configured, and the empty value is selected in the dropdown, mTLS is not used.



Inbound mTLS for Stibo Systems SaaS

Setting up inbound mTLS on a Stibo Systems SaaS environment includes working with Stibo System Technical Support, as defined below.

Inbound mTLS is available for RESTAPI, GraphQL, and SOAP.

To use inbound mTLS on your Stibo Systems SaaS environment:

1. Create a ticket in the Support Portal:
   - Request configuration of STEP Inbound mTLS Authentication.
   - Attach **public** certificate(s) for use with mTLS.

   > **Important:** Never share **private** certificates.

2. Technical Services configures the SSL profile, attaches the provided certificates, and creates a CName for mTLS requests.

   mTLS requests must use the <environment>+'-mtls' suffix: acme-mtls.mdm.stibosystems.com

3. Technical Services configures the following sharedconfig.properties for your environment and will need to schedule a restart of your system(s) to complete the setup:
   - HTTP.Service.MTLS.CertificateSubject.Template
   - HTTP.Service.MTLS.CertificateSubject.UserMap
   - HTTP.Service.MTLS.Enabled
4. When instructed by Technical Services, verify your connection and authentication.

   Since you retain your private certificate, Technical Services cannot test your connection.

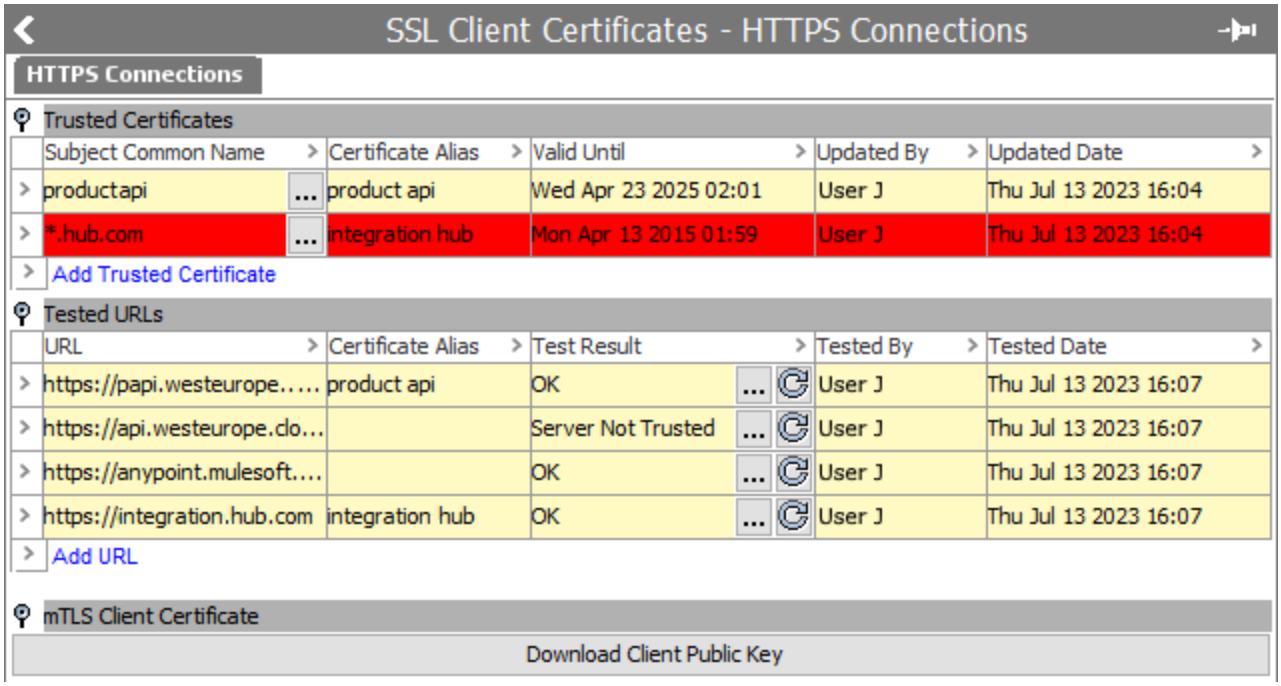
5. Update the Support Portal ticket to communicate success or issues.

SSL Client Certificates

For outbound integrations, the SSL Client Certificates node allows full management of HTTPS connection certificates for self-signed TLS or mTLS (multiple transport layer security), without any system downtime. It also provides full visibility into all configured certificates. Self-signed certificate management functionality is useful for environments where globally signed certificates are not present but the external site is trusted.

The SSL Client Certificates editor includes these flippers, which are described in the sections below:

- Trusted Certificates - manage self-signed certificates (those not globally trusted).
- Tested URLs - verify the connectivity to a remote URL and identify errors.
- mTLS Client Certificate - download the client public key for implementing outbound mTLS authentication.



Trusted Certificates

The trusted certificates flipper stores public keys that identify:

- external sites that STEP trusts
- sites that are using a self-signed certificate (not globally trusted by the Java Global Certificate Authority (CA))

Prerequisites

Required configuration varies based on your environment. Review the following entries to determine the configuration requirements for your environment:

TrustStore Entries

When a certificate is added to the SSL Client Certificates object, it is uploaded to the TrustStore configured via the entries in the sharedconfig.properties file.

If no trust store is configured on the environment, it is automatically created when the first certificate is imported. The properties defined below must be configured to specify where the keystore file is to be stored.

For Stibo Systems SaaS environments, a default TrustStore is automatically configured.

For on-premises environments, review the case-sensitive entries to identify the TrustStore:

- `SSL.Default.TrustStore.Location`

  For example:

  ```
  SSL.Default.TrustStore.Location=/shared/customer-config/truststore/truststore.jks
  ```

  Note: Provide a path and a file name for this property. When implementing this property, the path (/shared/customer-config/truststore/ in this example) must already exist and the STEP application should have write access to the path.

  The SSL Client Certificates functionality is available with release 2023.3. If this property and a trust store (with JKS) was configured before the 2023.3 release, no changes are required and the certificates are automatically visible in workbench.

- `SSL.Default.TrustStore.Password`

  For example:

  ```
  SSL.Default.TrustStore.Password=password
  ```

URL Entries

For all environments, optionally, add (or verify) the values that populate the **HTTPS URL** parameter in the New Certificate dialog. Configure the entry that corresponds to your connection method, via the appropriate case-sensitive entry:

- `RESTGateway.ServerURL` (as defined in the Configuring a Gateway Integration Endpoint - REST topic). For example:

  ```
  RESTGateway.ServerURL=qa=http://step-qa,stage=http://step-stage
  ```

- `RestDirectDeliveryURL` (as defined in the REST Direct Delivery Method topic).

  For example:

  ```
  RestDirectDeliveryURL = 1=http://myfirstendpoint, 2=http://mysecondendpoint
  ```

- `RESTDeliveryURL` (as defined in the REST Delivery Method topic).

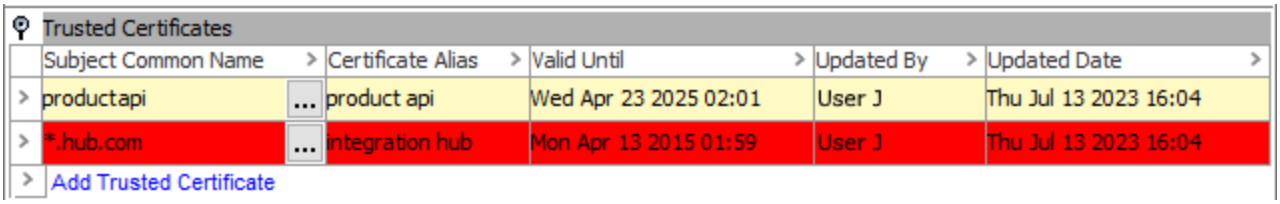
For example:

```
RESTDeliveryURL=qa=http://step-qa,stage=http://step-stage
```

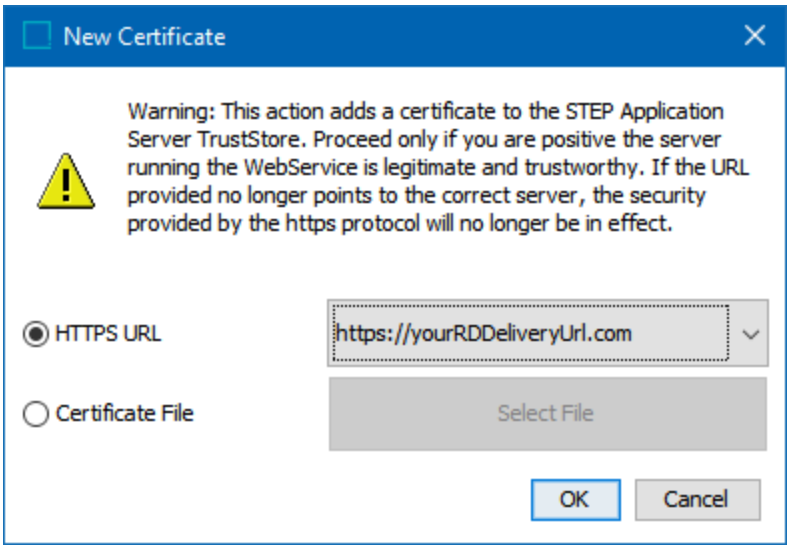
Adding a trusted certificate

To upload a trusted certificate:

1. In workbench on the System Setup tab, click the **SSL Client Certificates** node.
2. In the Trusted Certificates flipper, either click the **Add Trusted Certificate** link or right-click the > column and select Add Trusted Certificate, as shown in the Trusted Certificates editor section below.



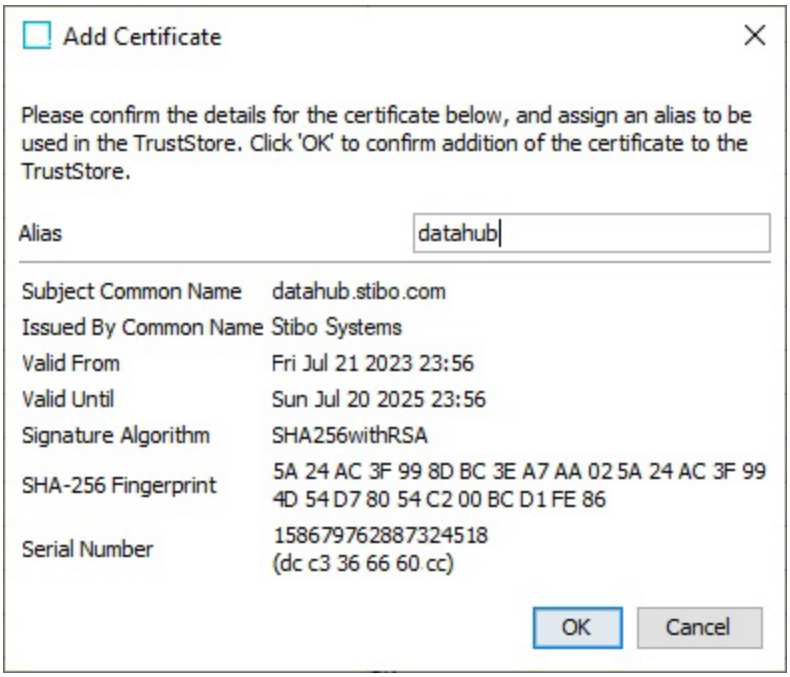
3. On the New Certificate dialog, choose a method to add a certificate:



- **HTTPS URL** - From the dropdown, select the URL for the site you want STEP to trust. The dropdown is supplied by the sharedconfig.properties URL entries defined above or URL entries that have been entered manually under the Tested URLs flipper (for which the connection test resulted in a 'Server Not Trusted' error. Click the **OK** button and the connection is verified to retrieve the certificate and import it in the truststore.
- **Certificate File** - click the **Select File** button, browse to your .crt or .cer file, and click the **Open** button. The file is verified and the certificate is imported in the truststore.

Note: No action is taken when a certificate that is already trusted is selected, for example, a CA certificate.

4. On the Add Certificate dialog:

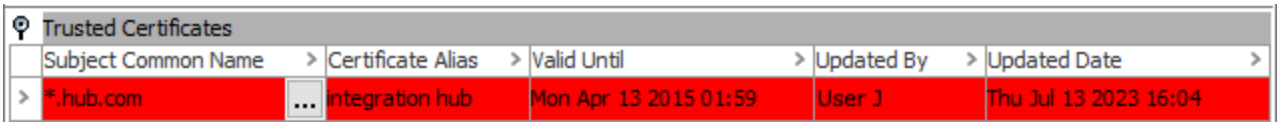


   - Review that the certificate information is as expected.
   - Add an **Alias**. This is a shortcut name for easy identification.
   - Click **OK**.
5. Verify the new information on the Trusted Certificates flipper. The certificate is listed with its Subject Common Name, Alias, and the expiration date (in the Valid Until column).
6. Test the related URL(s) under the Tested URLS flipper to confirm that a connection can now be established. After a successful test the Certificate Alias should also be visible for the URL.

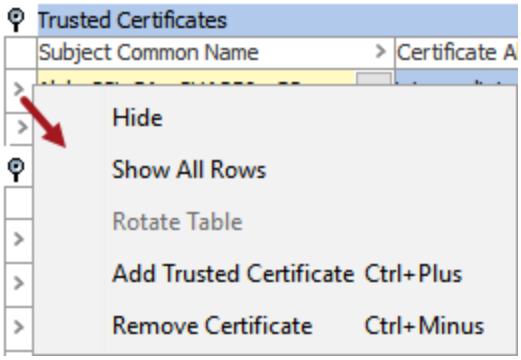
Trusted Certificates editor

The following alerts and functionality are available directly from the editor:

- When a certificate expires, the row displays with a red background.



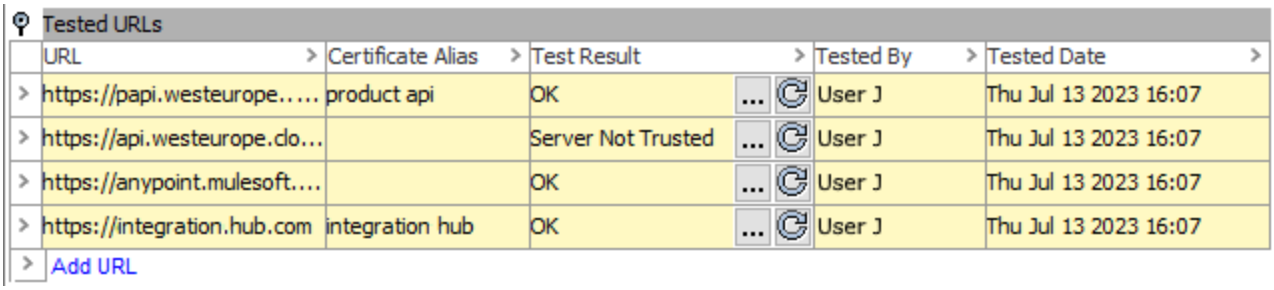
- Hide, Show All Rows, Add Trusted Certificate, and Remove Certificate options are available from the right-click menu on the > column on any row.



Tested URLs

This option allows you to verify the status of a URL and identify connection errors.

1. In workbench on the System Setup tab, click the **SSL Client Certificates** node.
2. In the Tested URLs flipper, either click the **Add URL** link or use the right-click menu as defined in the Tested URLs editor, as defined below.

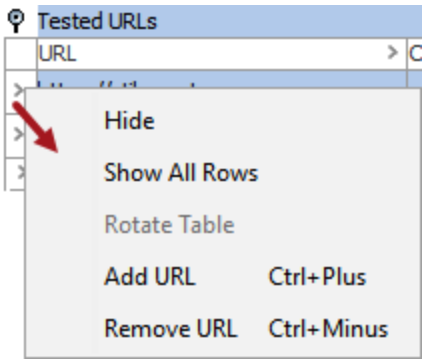


3. On the Add URL dialog, choose a method to add a **Https URL**:
   - Type a URL into the parameter. The value must begin with 'Https'.
   - Select a URL from the dropdown (supplied by the sharedconfig.properties URL entries defined above).
4. Click **OK**.
5. Review the Test Results column for the outcome of the test. Possible results include:
   - OK - displays for sites that are globally trusted or that are tested trusted by a certificate that was already imported.
   - Malformed URL - displays for incorrectly configured URLs.
   - Connection Error - displays for a variety of errors that result in a failed connection.
   - Not a HTTPS request - displays for non-HTTPS requests; only HTTPS is supported.

Tested URLs editor

The following alerts and functionality are available directly from the editor:

- The right-click menu on the > column on any row includes Hide, Show All Rows, Add URL, and Remove URL options.



- The Details button (...) displays additional information about error messages if available.
- The Retry button ( ) to run the test again.

mTLS Client Certificate

This option allows you to set up mTLS authentication with an external service by downloading the STEP public key, which can be used to authenticate requests coming from STEP by an external service. This is specifically relevant for Stibo Systems SaaS environments where customer rely on Stibo Systems to provide and configure the key. For more information on mTLS, refer to the Mutual Transport Layer Security topic.

Prerequisites

For Stibo Systems SaaS environments, a default keystore is automatically configured.

For other environments, the following configuration properties must be set:

1. Add only one public-private keypair to the .jks file.

   Note: If multiple private keys are included in the keystore, the selection is not guaranteed.

2. Add the following case-sensitive entries in the sharedconfig.properties file on the STEP application server:

   - `SSL.Default.KeyStore.Location`

     For example:

     ```
     SSL.Default.KeyStore.Location=/shared/customer-config/keystore/keystore.jks
     ```

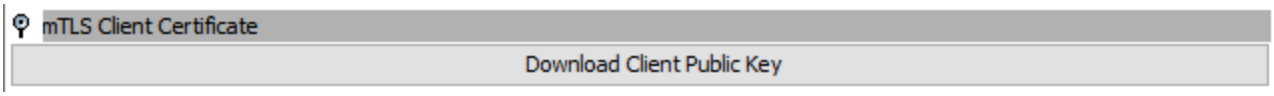
- `SSL.Default.KeyStore.Password`

  For example:

  ```
  SSL.Default.KeyStore.Password=password
  ```

Download a client public key

1. In workbench on the System Setup tab, click the SSL Client Certificates node.
2. In the mTLS Client Certificate flipper, click the **Download Client Public Key** button.



3. Save the certificate and add it to your external service to authenticate a STEP connection.

Web Service Endpoints

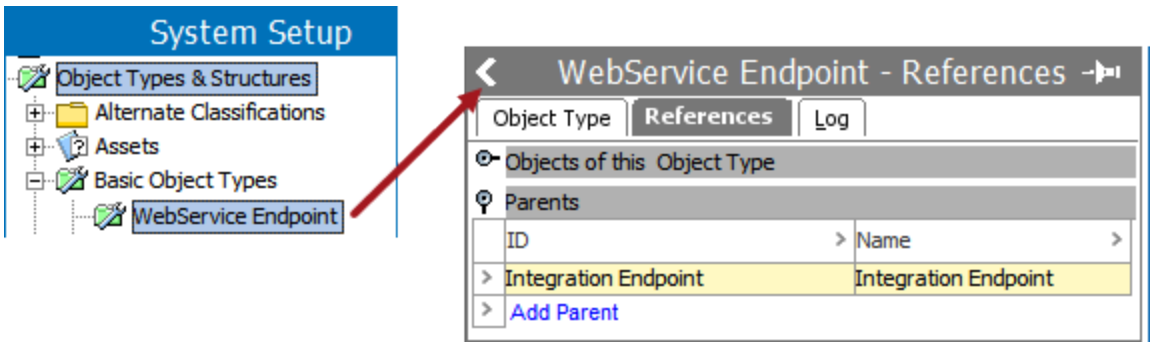
Web Services allow third-party systems to integrate with STEP data. A Web Service Endpoint is a setup entity that can be used to parameterize STEP web services built to receive request data and transmit response data. For more information on the Matching and Merging Web Service Endpoint, refer to the **Match and Merge Web Service Endpoint** section of the **Configuring Match and Merge** topic in the **Matching, Linking, and Merging** documentation.

Note: The Find Similar and Match & Merge Web Service is using an internal caching mechanism for validation rules, standardization, and matching. It can take up to five (5) minutes before this cache is updated, which is important to keep in mind when making configuration changes.

Setup entity definitions can be exported as comments and submitted to an external source control system for comparison purposes. For details, refer to the Configuration Management documentation.

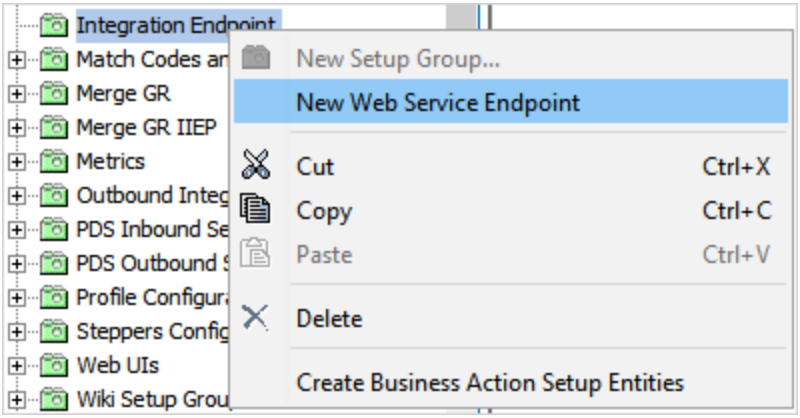
Prerequisites

A user with privileges to create system objects must create an entity-object type and associate it with a desired parent node. For more information, refer to the **Creating an Object Type** topic in the **System Setup** documentation.

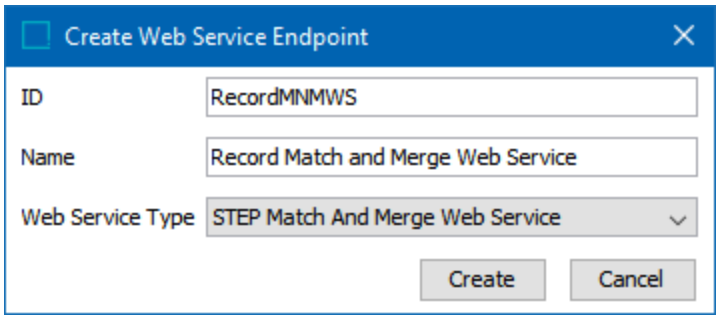


Create a Web Service Endpoint

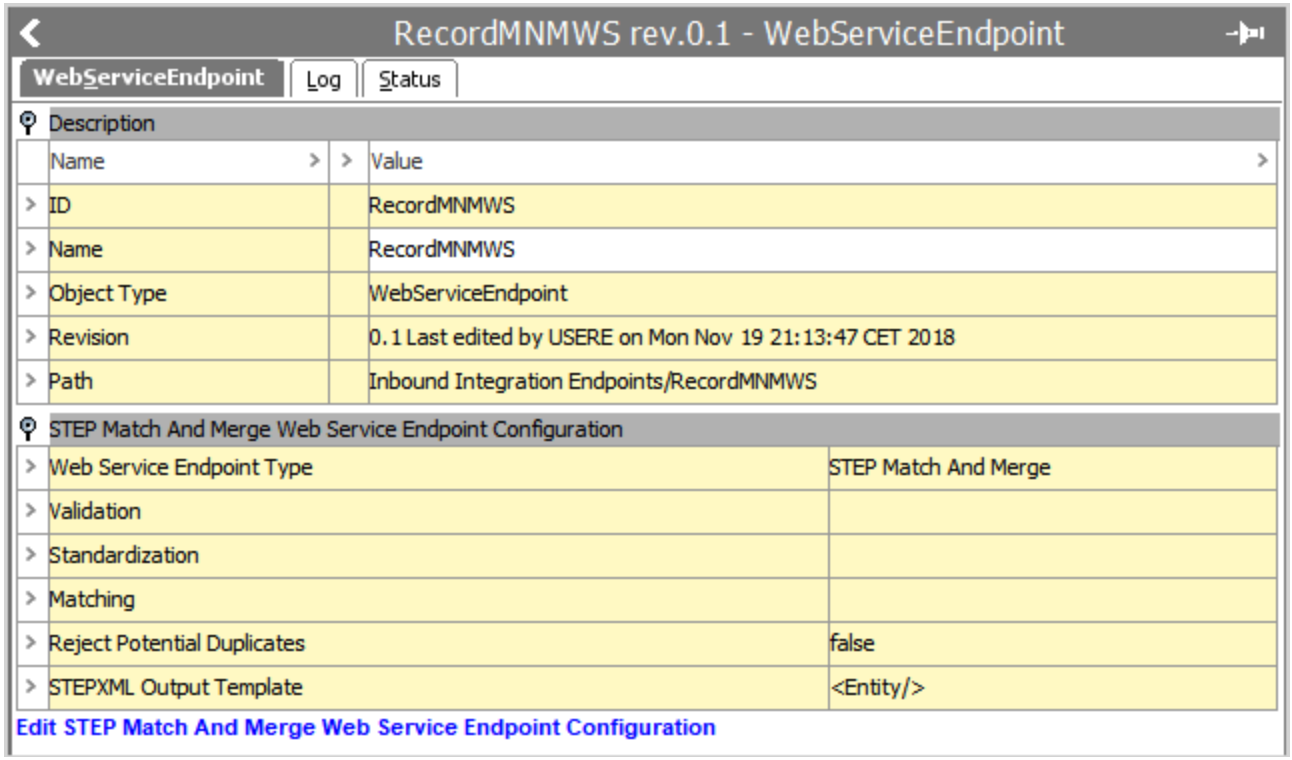
1. On the valid parent node, right-click and select 'New Web Service Endpoint.'



2. On the Create Web Service Endpoint dialog, add an ID, name, and the web service type.



An empty web service is displayed.



3. Continue with following topic for the required configuration based on the Web Service Endpoint Type selected:
   - STEP Find Similar - Web Service Endpoint Find Similar
   - STEP Match and Merge - Web Service Endpoint Match and Merge

Web Service Endpoint - Find Similar

A Web Service Endpoint receives request data and transmits response data. For more information on the Find Similar Web Service Endpoint, refer to the **Find Similar in RESTv2 API** topic of the **Find Similar** section of the **Matching, Linking, and Merging** documentation.

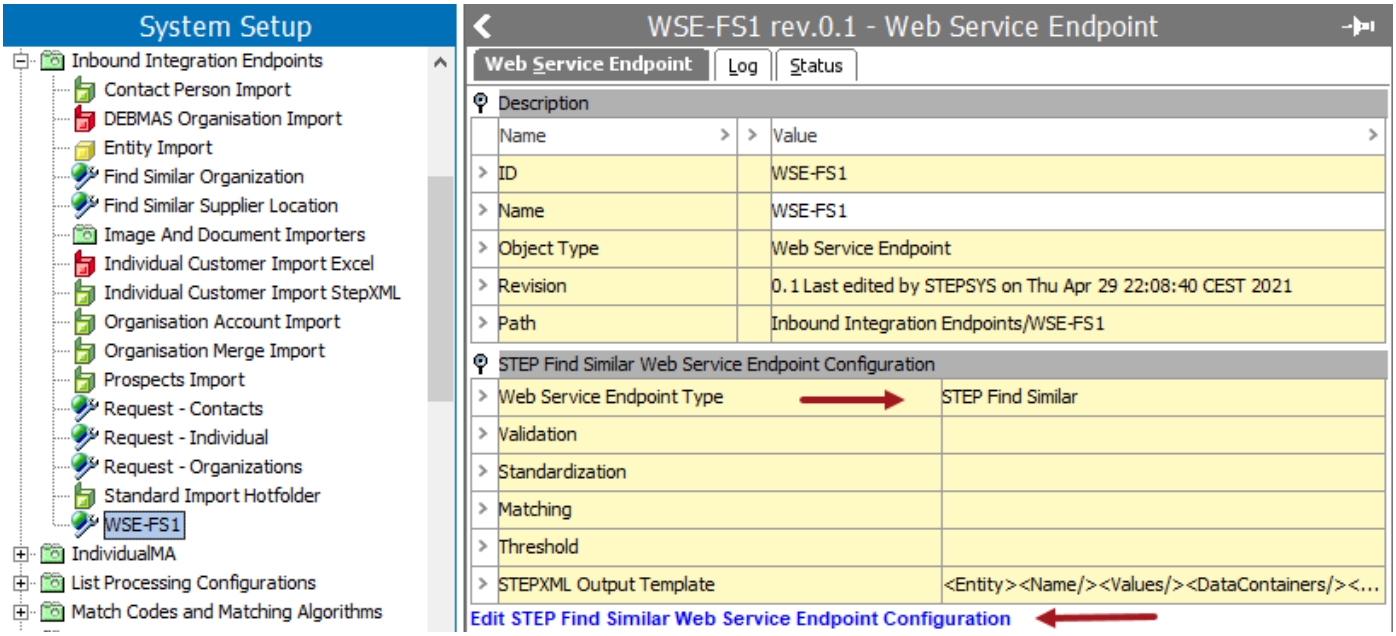
For example use cases, refer to the **Use Case Appendix** topic in the **Customer MDM Solution Enablement** documentation.

Prerequisites

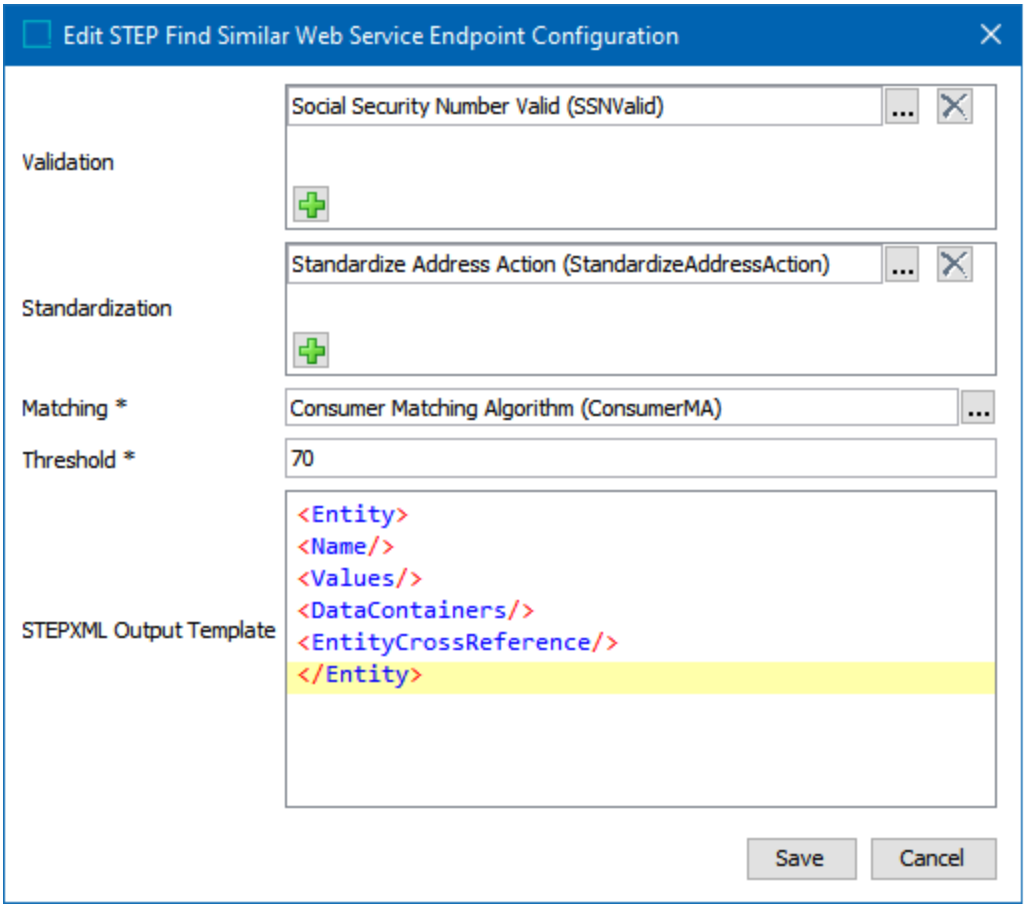
Create a web service endpoint as defined in the **Web Service Endpoints** topic.

Configure a STEP Find Similar Web Service Endpoint

1. On the Web Service Endpoint object, open the STEP Find Similar Web Service Endpoint Configuration flipper and verify the 'Web Service Endpoint Type' is set to 'STEP Find Similar'. The type cannot be changed, if necessary, create a new web service endpoint. For other types of endpoints, refer to the configuration options defined in the **Web Service Endpoint** topic.



2. Click the 'Edit STEP Find Similar Web Service Endpoint Configuration' link to configure as follows:



- **Validation** - click the plus button (+) to add a business condition. Click the ellipsis button (...) for the selector to display the available business conditions, click a condition, and click the **Select** button. Repeat this step to add as many business conditions as needed.

  A validation business condition allows the Web Service Endpoint to do early rejection of incoming requests. If the business condition fails the record, the record is never matched in the system. For example, rejecting an incoming business-to-business customer record that does not specify the customer's legal name.

- **Standardization** - click the plus button (+) to add a business action. Click the ellipsis button (...) for the selector to display the available business actions, click an action, and click the **Select** button. Repeat this step to add as many business actions as needed.

  Standardization business actions allow the Web Service Endpoint to enhance and enrich the incoming data before matching. For example, using Address Standardization before running the match algorithm.

- **Matching** - (required) click the ellipsis button (...) for the selector to display the available matching algorithms, click an algorithm, and click the **Select** button. Only one algorithm can be assigned to a Find Similar Web Service Endpoint.

Find Similar is a search based on a matching algorithm. The matching algorithm could be the one used for import, but it is often a good idea to build a separate dedicated algorithm for find similar.

- **Threshold** - (required) add the search threshold for potential duplicates. Set the match threshold for records that should be returned to the caller.

  The Auto Threshold and the Clerical Review Threshold defined in the matching algorithm do not apply to the Find Similar endpoint. If the search threshold is 70, only records that match the supplied values with a score of 70 or above will be returned in the response. For more information, refer to the **Configuring Matching Algorithms** topic in the **Matching, Linking, and Merging** documentation.

- **STEPXML Output Template** - decides how the entities returned by the web service are populated. A default template is displayed but can be edited if you require different data from STEP. For more information, refer to the **STEPXML Format** topic.

- Click the **Save** button.

3. Continue configuration and validation as defined in the **Find Similar Web Services** topic of the **Matching Linking, and Merging** documentation.

Web Service Endpoint - Match and Merge

A Web Service Endpoint receives request data and transmits response data. For more information on the Matching and Merging Web Service Endpoint, refer to the **Configuring Match and Merge** topic in the **Matching, Linking, and Merging** documentation.

Match and Merge integrates via the following calls:

- SOAP - on STEP API Documentation (available at [system]/sdk or from the Start Page), click the 'Soap API documentation' link, and click the **STEP Match And Merge Web Services SOAP API** link.
- REST - on STEP API Documentation (available at [system]/sdk or from the Start Page), click the link under the 'REST API V2' heading. Under the Entities section, click the POST button to find details and the 'Try it out' option for **/entities/match-and-merge**.

For example use cases, refer to the **Use Case Appendix** topic in the **Customer MDM Solution Enablement** documentation.

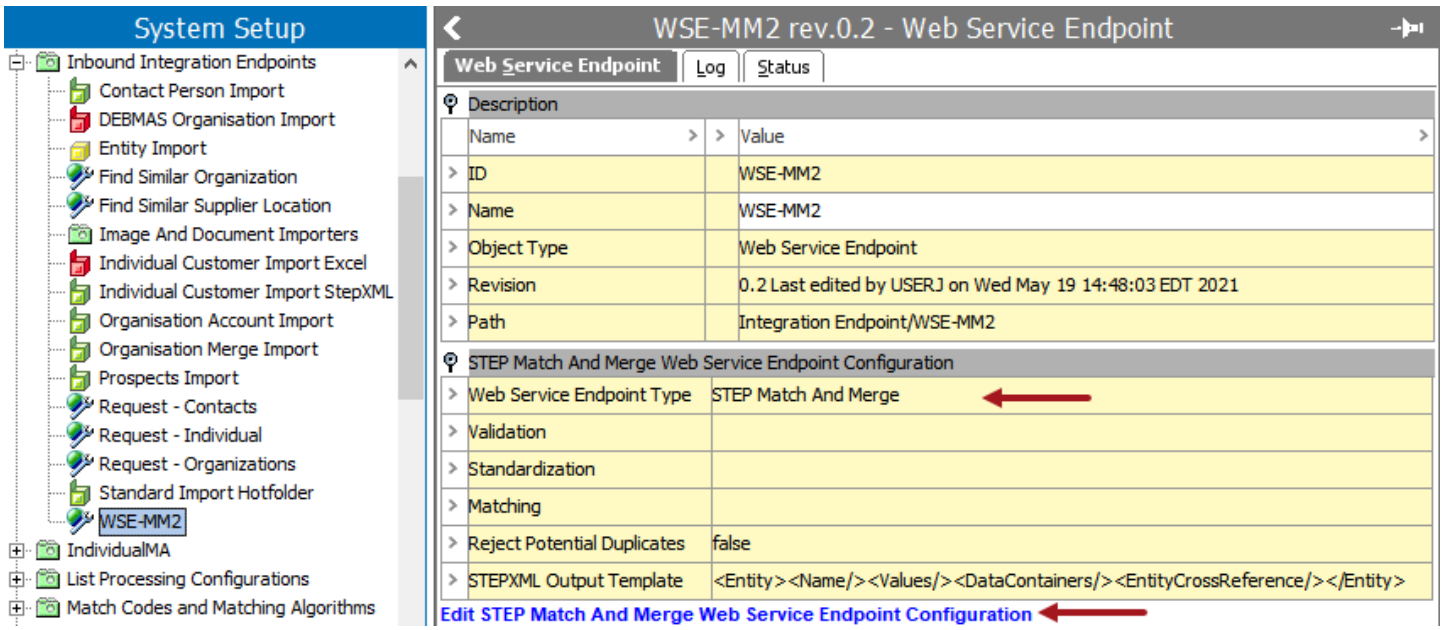
To access the Match and Merge Web Service endpoint for a given system, navigate to the following URL: [your system URL: port]/MatchingWS/matching.

Prerequisites

Create a web service endpoint as defined in the **Web Service Endpoints** topic.

Configure a STEP Match and Merge Web Service Endpoint

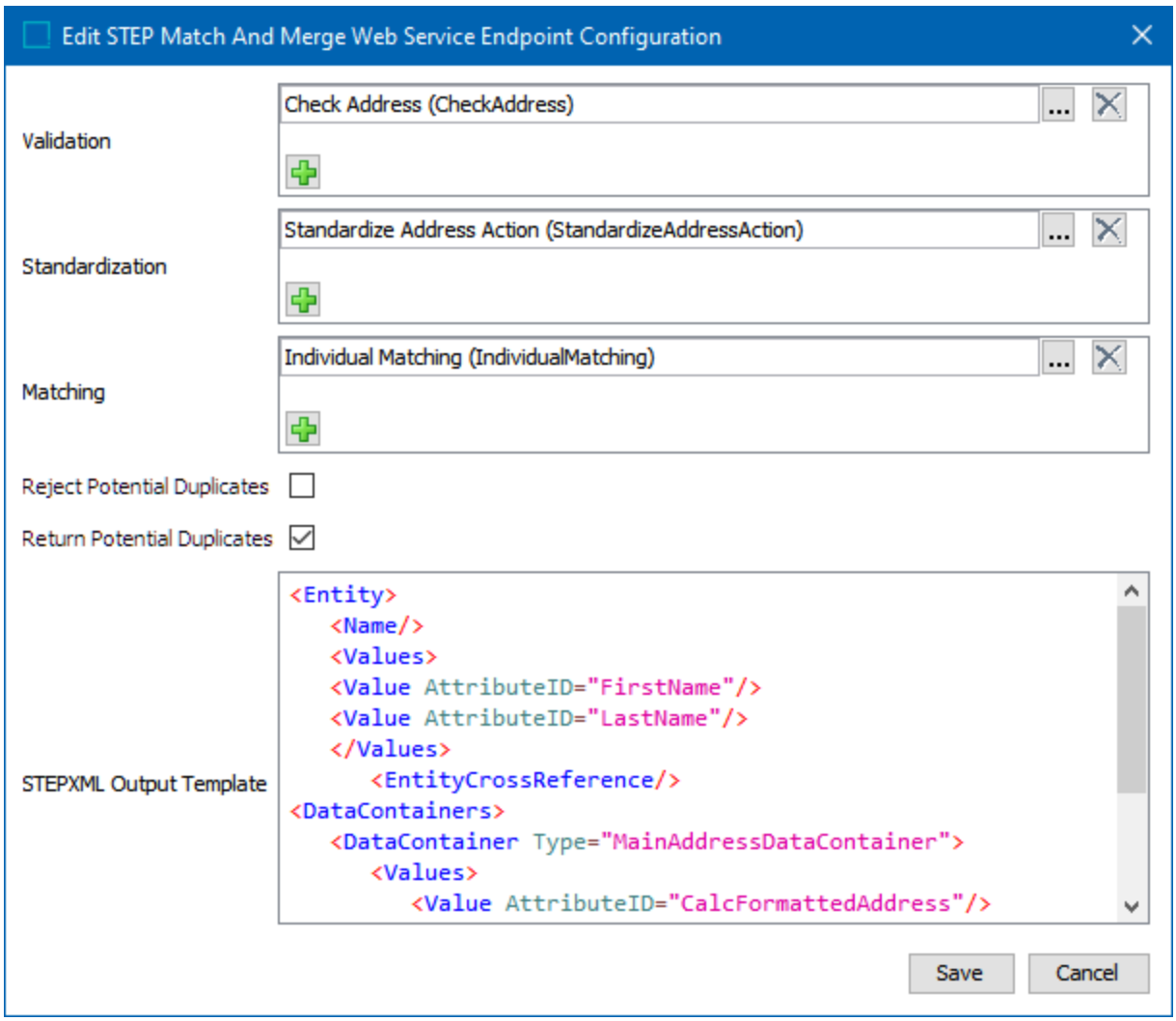
1. On the Web Service Endpoint object, open the STEP Match and Merge Web Service Endpoint Configuration flipper and verify the 'Web Service Endpoint Type' is set to 'STEP Match and Merge '. The type cannot be changed, if necessary, create a new web service endpoint. For other types of endpoints, refer to the configuration options defined in the **Web Service Endpoint** topic.



2. Click the 'Edit STEP Match and Merge Web Service Endpoint Configuration' link to configure as follows.

   To modify a parameter:

   - Click the green plus button ( ) to add a criterion ( ).
   - Click the ellipsis button ( ) on a criterion to open the relevant selection dialog box. Make a selection, and click the Select button to continue.
   - Click the X button ( ) to delete a criterion.



- **Validation** - (optional) This parameter is constrained by business conditions that limit what data is matched. If the data does not conform to any validation criteria denoted, the record is rejected. These operations result in true or false. Refer to the **Business Conditions and Contexts** section below. For more information, refer to the **Business Condition** topic in the **Business Rules** documentation.

  A validation business condition allows the Web Service Endpoint to do early rejection of incoming requests. If the business condition fails the record, the record is never matched in the system. For example, rejecting an incoming business-to-business customer record that does not specify the customer's legal name.

  Click the plus button ( ) to add a business condition. Click the ellipsis button ( ) for the selector to display the available business conditions, click a condition, and click the Select button. Repeat this step to add as many business conditions as needed.

- **Standardization** - (optional) This parameter is constrained by business actions that make the data being matched conform to a set standard format. For more information, refer to the **Business Actions** topic in the **Business Rules** documentation.

Important: Standardization business actions allow the Web Service Endpoint to enhance and enrich the incoming data before matching. For example, using Address Standardization before running the match algorithm.

Click the plus button ( ) to add a business action. Click the ellipsis button ( ) for the selector to display the available business actions, click an action, and click the Select button. Repeat this step to add as many business actions as needed.

- **Matching** - (required) This parameter is where a matching algorithm is set for the records that will be compared. In cases with different object types, there may be multiple matching algorithms. For more information, refer to the **Configuring Matching Algorithm** topic in the **Matching, Linking, and Merging** documentation.

  Match and Merge is based on a matching algorithm. The matching algorithm could be the one used for import, but it is often a good idea to build a separate dedicated algorithm for match and merge.

  Click the ellipsis button ( ) for the selector to display the available matching algorithms, click an algorithm, and click the Select button. Only one algorithm can be assigned to a Match and Merge Web Service Endpoint.

- **Reject Potential Duplicates** - (optional) When checked, this parameter automatically rejects potentially duplicated records instead of creating a clerical review task. For example, the match score on an imported record that would normally result in the creation of a clerical review task is instead automatically rejected when the matching algorithm identifies similar records.

  Incoming records that score above the auto threshold, or that are matched by STEP ID or source record ID are still accepted as updates.

  Incoming records that score below the clerical review threshold still create new records. For more information, refer to the **Match and Merge** topic in the **Matching, Linking, and Merging** documentation.

- **Return Potential Duplicates** - (optional) When checked, if a creating entity encounters a duplicate, the response XML returns the potential duplicate in the output. When unchecked, no potential duplicates are returned. This box is checked by default.

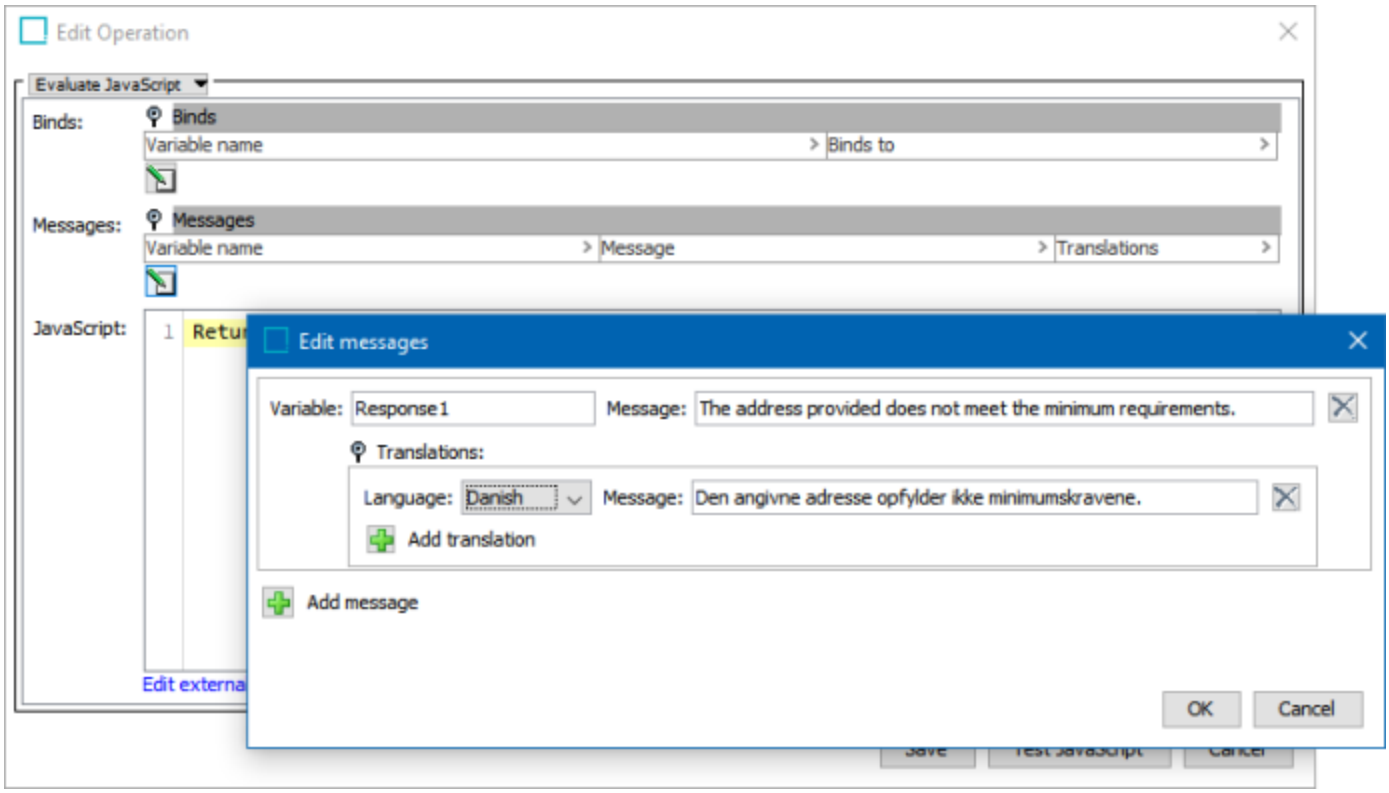
- **STEPXML Output Template** - (required) The output template is an Advanced STEPXML structure that populates and filters the STEP data in the response. The <Entity> tag must be the highest-level tag in the output template. A default template is displayed and can be edited if you require different data from STEP. Refer to the **Match Advanced STEPXML Output Template** section below.

- Click the **Save** button.

3. Continue configuration and validation as defined in the **Configuring Match and Merge** topic of the **Matching, Linking, and Merging** documentation.

Business Conditions and Contexts

When sending a web service request, the request includes the current context. This request is sent to STEP and STEP returns a value with the context language included. If the 'Evaluate JavaScript Business Condition' fails, the included context signals which translation of the business condition message to include.



For more information, refer to the **Business Condition: Evaluate JavaScript** topic in the **Business Rules** documentation.

Match Advanced STEPXML Output Template

The following in an example output template for a Match solution:

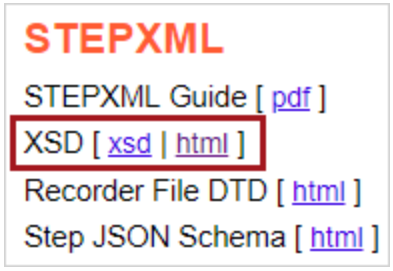
```
<Entity>
<Name/>
<Values>
<Value AttributeID="IndLastName"/>
<Value AttributeID="IndFirstName"/>
</Values>
```

```
    <DataContainers>
    <MultiDataContainer Type="ContEmailDataContainer"/>
    <DataContainer Type="AddrMainAddressDataContainer"/>
    </DataContainers>
    </Entity>
```

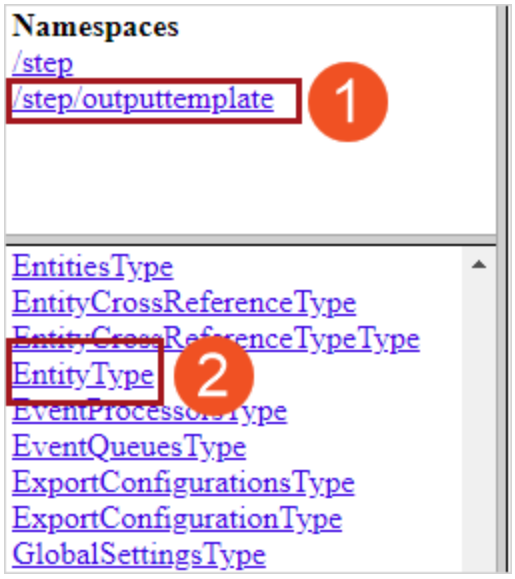
The XML template must contain a complexType 'EntityType' as defined in the STEPXML XSD.

To review the complexType:

1. Open the STEP SDK and API documentation from [server]/sdk or access the **STEP API Documentation** button on the Start Page.
2. Under the STEPXML heading, click the **html** link.



3. At the top, under the Namespaces heading, click the **/step/outputtemplate** link and scroll down to select the **EntityType** link from the complexTypes list.



Refer to the following topics for more information:

- On the Advanced STEP XML - **STEPXML Tags and Examples** topic in the **Data Exchange** documentation.
- On how to use the Advanced STEPXML output template - **Advanced STEPXML Format** topic in the **Data Exchange** documentation.

Error handling

The Web Service handles each record individually in a request. If any error occurs while processing, the individual record is skipped, and the service moves on to process the next record. An example would be:

- M&M Web Service import of record A, B, and C in same request.
- Record A processed successfully.
- Record B processing failed.
- Record C processed successfully.

In the Web Service response result for record A and C, status is ‘Processed’. Result for record B is status ‘Failed’.