USER GUIDE

Data Exchange

Release 9.2-MP3 (October 25, 2019)

StiboSystems

MASTER DATA MANAGEMENT

Table of Contents

Table of Contents **2**
Data Exchange **24**
Inbound Tools 24
Data Format 24
Receiver Method 24
Outbound Tools 24
Data Format 24
Delivery Method 25
Gateway Tool 25
Web Service Tools 25
Data Formats **26**
Advanced STEPXML Format **29**
Schema 29
Format Availability 29
Mapping 29
Inbound Data 30
Outbound Data 31
Alphabetical Index - XML Format **34**
Format Availability 34
Mapping 34
Outbound Data 34
Ariba CIF Format **36**
Format Availability 36
Mapping 36
Outbound Data 36
Ariba CIF 3.0 Format **38**
Format Availability 38
Mapping 38
Outbound Data 38
BMEcat Format **40**
Format Availability 40
Mapping 40
Inbound Data 40
Outbound Data 42
BMEcat 2005 Format **45**
Format Availability 45
Mapping 45
Inbound Data 45
Outbound Data 47
BMEcat 2005 for ETIM Data Configuration **51**
CSV Format **52**
Format Availability 52
Mapping 52
Inbound Data 52
Outbound Data 55
Exporting Data via JDBC with CSV Format **58**
Install Required Drivers 58
Configure Properties 58
Select Format 59
Map Data 59
Select Delivery Method 61
Analytics using JDBC Example **63**

Map Data 63
Select Delivery Method 65
cXML Format 67
Format Availability 67
Mapping 67
Outbound Data 67
eCl@ss Format 69
Format Availability 70
Mapping 70
Multiple Versions 70
Finding an Attribute 71
Inbound Data 72
eCl@ss Classification Import 74
Map eCl@ss Attributes to STEP Attributes 78
ETIM Format 81
Format Availability 81
Mapping 81
ETIM Configuration 81
ETIM Versions 81
ETIM Classification 82
ETIM Feature Types 84
Valid Values are Determined by ETIM Version 84
Valid Units are Determined by Classification to Attribute Link 85
Handling Import Errors 86
ETIM and ETIM v2 Format 87
Format Availability 87
Mapping 87
Inbound Data 87
Inbound Parameters 87
Import Manager 88
IIEP 89
ETIM IXF Format 90
Format Availability 90
Mapping 90
Inbound Data 90
Create Product to ETIM Classification Link 94
Set Validity on ETIM Attributes for Products 97
Manually Import ETIM Attribute Values 102
Excel Format 113
Format Availability 113
Mapping 114
Inbound Data 114
Outbound Data 117
Date and Number Formatting with Excel 119
Importing Multi-Valued References or Links with Excel 121
Managing Attribute Parameters with Excel 130
Excel Custom Template 134
Format Availability 134
Mapping 134
Output 134
Excel Custom Template Configuration 136
Configuration 136

Image conversion configurations with Excel Custom Template 144

Excel Smartsheet Format 146

Format Availability 146

Setup Requirements 146

Additional Information 147

Initiating Products Using a Smartsheet 148

Prerequisites 148

Product Initiation 148

Exporting a Template with a Supplier Selector 151

Maintaining Products Using a Smartsheet 154

Prerequisites 154

Product Maintenance 154

Smartsheet and LOVs 158

Limitations 158

Smartsheet Data and Template Configurations 159

Export Manager Wizard 159

Select Objects Step 159

Select Format Step 160

Map Data Step 165

Advanced Step 168

Select Delivery Method Step 168

Saving and Modifying the Configuration 168

Additional Configuration Options 168

Smartsheet Import Configurations 170

Creating an Import Configuration 170

Select Format Step 170

Advanced Settings Step 170

Saving and Modifying the Configuration 172

Using a Smartsheet 173

Entering Values for Multi-Valued LOVs and References 173

Validating the Sheet 175

Copying Data in a Smartsheet Template 176

Smartsheet Types 177

Multi Level Smartsheets 177

Multiple Product Object Type Smartsheets 178

Open Format Smartsheet 179

Export Manager Configuration 179

Default Cell Formats for Open Format Smartsheets 180

Working with Formulas in Open Format Smartsheets 181

Validation of Formulas 181

Additional Considerations for Open Format Smartsheets 181

Handling of Cells with Number Formats 182

Read-Only Cells 183

Background Shading and Text Formatting 183

Excel Conditional Formatting 184

Copy and Paste Behavior for Rows and Columns 184

Additional Information About Smartsheets 185

System Configuration 185

Smartsheet Cover Tab 185

Smartsheets Special Attribute Handling 186

Conditional Attributes 186
Workflow State Mandatory Attributes 187
Smartsheet Background Processes and Queues 188
Using Smartsheet Import Background Process 188
Using Smartsheet Export Background Process 189
Smartsheet Multi-Supplier View 191
Smartsheet System Properties 192
Limit Supplier Browsing 192
Service URL Override 192
Trusted Headers Authentication 192
Increasing Default of Preformatted Template Rows 193
Smartsheet Template Product Row 194
Prerequisites 194
Configuration 195
'Example' Product 195
Smartsheet Template Product Reference 195
Example Setup 196
FAB-DIS Format 200
Format Availability 200
Mapping 200
Inbound Data 200
Outbound Data 201
FAB-DIS Export Format 203
FAB-DIS ETIM Export Mapping 204
FAB-DIS Media Export Mapping 207
FAB-DIS Import Format 210
FAB-DIS Import Workbench Configuration 213
Prerequisites 213
Asset Importer Configuration 213
IIEP Configuration 215
FAB-DIS Import Mapping 219
IIEP Mapping 219
Automatic Mapping 222
Media Mapping via Asset Importer Configuration 223
FAB-DIS Import Web UI Configuration 224
FAB-DIS Import Background Processes 225
Troubleshooting 225
FAB-DIS Import Example 227
ETIM Tab 227
MediaTab 229
FixedWidth Format 233
Format Availability 233
Mapping 233
Inbound Data 233
FixedWidth Format Configuration Examples 236
Example Data 236
End Positions (Zero Based) Example 236
End Positions (One Based) Example 238
Width Example 240
Flatplan Excel Format 242
Format Availability 242

Mapping 242
Inbound Data 242
Outbound Data 243
Generic XML Format 245
Limitations 245
Format Availability 245
Mapping 245
Inbound Data 245
Outbound Data 247
Generic XML Inbound Processing Instructions 250
DimensionPointID Processing Instructions in Generic XML 251
MultiSource Processing Instructions in Generic XML 253
Record Processing Instructions in Generic XML 255
Repeated Processing Instructions in Generic XML 256
Source Processing Instructions in Generic XML 259
Preventing Duplicate Field Names 259
SourceID Processing Instructions in Generic XML 260
Generic XML Import Simple Example 263
Input Document 263
Template 264
Conversion Preview 265
Generic XML Import Advanced Example 266
Input Document 266
Creating a Template 267
Remove Redundant Elements 267
Add Record and Source Instructions 268
Add Repeated Instruction 269
Add Repeated Instruction With a Filter 270
Add MultiSource Instruction 271
Conversion Preview 272
Generic XML Outbound Processing Instructions 274
Allow Empty Tags Parameter in Generic XML 275
Attributes Processing Instructions in Generic XML 278
Attribute Aspects in Generic XML 282
Attribute Links and LOVs Processing Instructions in Generic XML 285
Mapping Attribute Links, LOVs, and LOV Values in Generic XML 289
Attribute Links and Units Processing Instructions in Generic XML 295
Mapping Attribute Links, Units, and Unit Base in Generic XML 299
DoNotEmit Processing Instructions in Generic XML 302
List of Values Object Type in Generic XML 305
Multi-Valued Objects in Generic XML 309
Multi-Cardinality Tags in Generic XML 311
Repeated Targets 312
MultiTarget 313
Composite Mapping Targets 314
Nestable Data Paths in Generic XML 316

Data Path Concept 316
Nesting Concept 316
Valid Path Step Types 317
Data Sources Available for Data Paths 317
Nestable Data Paths in Generic XML Example 319
Data Path Configuration 320
Mapping for Nestable Data Paths in Generic XML Example 323
RemoveIfEmpty Processing Instructions in Generic XML 331
Reporting Deleted Assets with a Generic XML OIEP 334
Object Type and Event Type Selection 334
Template 335
Mapping 335
Results 335
Reporting Deleted Classifications with a Generic XML OIEP 337
Object Type and Event Type Selection 337
Template 338
Mapping 339
Result 339
Reporting Deleted Entities with a Generic XML OIEP 340
Object Type and Event Type Selection 340
Template 341
Mapping 342
Results 342
Reporting Deleted Product References with a Generic XML OIEP 343
Product 343
Object Type and Event Type Selection 343
Template 344
Mapping 345
Results 345
Mapping to Report Deleted Product References with a Generic XML OIEP 347
Single-Cardinality Tags in Generic XML 356
Record 357
Target Default 358
Target Override Default 358
Units with a Generic XML OIEP 359
Object Type and Event Type Selection 359
Template 360
Mapping 360
Results 361
Mapping Unit Aspects with a Generic XML OIEP 362
Generic XML Sample Export 364
IDoc MATMAS 05 Format 366
Format Availability 366
Mapping 366
Inbound Data 366
Outbound Data 368
Publication Excel Format 371
Format Availability 371
Mapping 371

Inbound Data 371
Outbound Data 372
SmartLabel Format 374
Format Availability 374
Mapping 374
Outbound Data 374
Configuring the SmartLabel Template 377
SmartLabel Attributes 377
Modifying the SmartLabel Template 378
STEPXML Format 381
Limitations 381
Format Availability 381
Mapping 382
Inbound Data 382
Outbound Data 383
STEPXML Outbound Parameters 385
Global Settings 385
Data Objects 386
Configuration 390
Publishing 398
STEPXML Tags and Examples 400
AssetContent Tag in STEPXML 401
Asset Content Templates 401
Asset Content Output 403
Assets Tag in STEPXML 405
Name 405
Values 405
AttributeLink Tag in STEPXML 408
IncludeInherited 408
AutoApprove in STEPXML 410
Backward Compatibility in STEPXML 411
Classifications Tag in STEPXML 413
IncludeParent 413
Additional STEPXML Tags 413
Name 413
Values 413
ClassificationReference 413
MetaData 413
Product to Classification Links Owned by the Classification 414
Create Objects in STEPXML 415
Data Containers in STEPXML 417
Setting Up Data Containers 417
Adding and Maintaining Data Containers 418
Delete Objects in STEPXML 421
Event-Based OIEP Triggered by Deleting Products, Classifications, and Assets 421
Delete Product Reference in STEPXML 423
Delete Values in STEPXML 426
Export Inheritance in STEPXML Example 427
Filter Data Containers in STEPXML 429
Known Limitation 430
Filter Products in STEPXML 431
Filter References in STEPXML 435
Asset References 435

Classification References 435
Product References 435
Results 436
Filter Values in STEPXML 438
IncludeKey Tag in STEPXML 439
Inherit References in STEPXML 441
Inherit Values in STEPXML 444
IncludeInherited 444
IncludeInheritedAttributes 444
Minimum, Referenced, and Selected in STEPXML 447
Hierarchy Export Including Objects Referenced Example 448
Product Export Including Definitions, LOVs, and Units for Attributes Example 450
Multi-Context Data in STEPXML 452
Products Tag in STEPXML 454
ExportSize 454
IncludeParent 454
Additional STEPXML Tags 454
Name 454
Values 455
AssetCrossReference 455
ProductCrossReference 455
ClassificationReference 455
Product Nesting Example 455
Nested structure 455
Non-nested structure 456
Product with Units, References, and Multi-Values Attributes Example 456
Product to Classification Links Owned by the Classification 457
Referenced and Embedded XML Attributes in STEPXML 459
ReplacementRules Tag in STEPXML 465
Using the ReplacementRules Tag 465
SequenceProduct Tag in STEPXML 471
SingleUpdateMode in STEPXML 475
STEP-ProductInformation Tag in STEPXML 476
DefinitionsAsComments 476
EventsAsComments 476
ExportDeletedData 476
ExportDerivedAttrs 477
FollowOverrideSubProducts 477
ResolveInlineRefs 478
Validation 478
UNSPSC Format 479
Format Availability 479
Mapping 479
Inbound Data 479
xCBL Format 483
Format Availability 483
Mapping 483
Outbound Data 483
Data Mapping 486
Inbound Data Mapping 486
Outbound Data Mapping 486

Inbound Map Data Options **487**
Formats 487
Map To Object Type 488
Mapping Data 488
Modifying Data 488
Profiling Data 489
Tips for Mapping Data 489
Inbound Map Data - Auto Map **490**
Predefined Header Text 490
Auto Map Example 491
Inbound Map Data - Constant **493**
Mapping a Constant 493
Inbound Map Data - Generate Profile **497**
Profiling Setup 497
Profiling Data 497
On the Fly Data Validation 500
Inbound Map Data - Map **502**
Mapping the Same Column Multiple Times 502
Mapping Options 503
Add Child to Override - Map Inbound **504**
Asset Reference - Map Inbound **507**
Attribute - Map Inbound **514**
Attribute Info - Map Inbound **519**
Attribute Info File Structure Requirements 521
Attribute Validation Info - Map Inbound **524**
Attribute Validation Info File Structure Requirements 526
Classification Reference - Map Inbound **529**
Data Container Attribute - Map Inbound **532**
Delete Values - Map Inbound **536**
Entity Reference - Map Inbound **538**
Entity Reference via Source Record ID - Map Inbound **543**
Prerequisites 543
Map Entity References 543
ID or Key - Map Inbound **547**
Mapping ID 547
Mapping Key 550
Multivalued Variable - Map Inbound **552**
Name - Map Inbound **556**
Object Type - Map Inbound **558**
Map to Create New Items with Object Type 559
Map to Change the Existing Object Type 560
Overrides Product - Map Inbound **563**
Parent - Map Inbound **566**
Product Classification Links - Map Inbound **568**
Product Reference - Map Inbound **572**
Reference Meta-Data - Map Inbound **575**
Variable - Map Inbound **580**
Inbound Map Data - Map To **584**
Mapping Based on Object Type 584
Asset Object Type - Map To Inbound **585**
Attribute Object Type - Map To Inbound **586**
Classification Object Type - Map To Inbound **587**
Entity Object Type - Map To Inbound **589**

Product Object Type - Map To Inbound **591**

Inbound Map Data - Remove **593**

Remove a Mapped Column 593

Inbound Map Data - Transform **595**

Transforming Data 595

Dimension Dependent Data 596

Outbound Map Data Options **597**

Formats 597

Mapping in Outbound Tools 597

Mapping 599

Transformations 599

Outbound Map Data - Data Source **600**

Available Data Source Options 600

Asset References - Data Source Outbound **602**

Attribute Information - Data Source Outbound **606**

Attributes (and Data Containers) - Data Source Outbound **609**

Attributes Inherited from Classification - Data Source Outbound **613**

Business Functions - Data Source Outbound **617**

Classification References – Data Source Outbound **626**

Classifications - Data Source Outbound .. **628**

Constant Value - Data Source Outbound .. **630**

Custom Attributes - Data Source Outbound **634**

Data Path - Data Source Outbound **635**

Entity References – Data Source Outbound **636**

ID or Key - Data Source Outbound **638**

Index Words – Data Source Outbound **641**

Mapping Index Words Linked to Products Using Word 642

Mapping Index Words Linked to Products Using Sort Word 643

Mapping Index Words from Classification 644

Inherit Data and References - Data Source Outbound **646**

Insert Referenced Objects - Data Source Outbound **649**

Is Deleted - Data Source Outbound **652**

Multi Level Parent Attributes - Data Source Outbound **653**

Multi Level References - Data Source Outbound **657**

Name - Data Source Outbound **666**

Object Type Name - Data Source Outbound **668**

Page Number – Data Source Outbound ... **670**

Parent ID - Data Source Outbound **671**

Product Classification Links – Data Source Outbound **673**

Product-Override Child ID – Data Source Outbound **676**

Product References - Data Source Outbound **678**

STEP Workflow Task Info - Data Source Outbound **681**

System Setup - Data Source Outbound **687**

Outbound Map Data - Mapping Targets 688
Tabular Format Targets 688
XML Format Targets 689
Outbound Map Data - Transform 691
Transforming Data 691
Data Container Separator - Transform Outbound 693
Multi-Value Separator - Transform Outbound 696
Aspect - Transform Outbound 699
Attribute Meta Data Aspect 705
Attribute Type Aspect 707
Base Unit ID, Name, and Value Aspects ...709
Closest Attribute Link Meta Data Aspect ..713
Closest Attribute Link Meta Data Aspect and Display Sequence 715
Dimension Dependencies Aspects 719
Is Deleted Aspect 722
Key Aspect 726
LOV ID Aspect 729
LOV Value-ID Aspect 732
LOV Value-ID or Value Aspect 735
Unit Aspect 740
Unit ID Aspect 743
Value and Unit Aspect 746
Value and Unit Meta Data Aspect 748
Value Aspect 751
Transformation Examples 753
Concatenate Column with Unit Data Example .....
Concatenate Data Using Variables Example 758
Concatenate into Single-Valued Data Example 763
Dimension Point for Single Dimension Data Example 766
LOV Value-ID or Value Aspect 770
Extract Characters at Position Example ...775
Extract Characters Between and Extract Characters After Example 778
Format Number Example 783
Header and Value Aspects Example 786
If Equals Source Example 791
Insert Text Example 792
Math Operation Example 797
Merge Attributes and Constant Example ..801
Merge Data for Multi-Valued Data Example 804
Replace Substrings of the Value Example 814
Split and Extract Data Example 817
Replace the Whole Value Example 824
Transform Date by Locale or by Pattern ...830
Export Transformations 831
Import Transformations 834
Transform Using Attribute Transformation Example 838
Transformation Lookup Table Example ... 840
Export Manager 844
Setup Requirements 844

Additional Information 844
Creating a Data Export 845
Considerations for Exporting Products 849
Export Manager - Select Configuration 850
Export Manager - Select Objects 852
Export Manager - Select Format 855
Export Manager - Map Data 857
Export Manager - Advanced 858
Export Manager - Select Delivery Method 861
Email Delivery Method 862
File Delivery Method 864
FTP Delivery Method 865
JDBC Delivery Method 867
Server Side Delivery Method 870
SFTP Delivery Method 872
Websphere Commerce Import Delivery Method 874
Running a Data Export 875
Maintaining a Saved Export Configuration 877
View a Saved Export Configuration 877
Edit a Saved Export Configuration 877
Monitoring a Data Export 880
Monitoring the Export Process 880
Execution Report 881
Result 883
Scheduling a Data Export 884
Scheduled Data Export - Select Export Configuration 886
Scheduled Data Export - Schedule Process 887
Scheduled Data Export - Name Process 888
Gateway Integration Endpoints 889
Setup Requirements 889
Additional Information 889
Initial Setup for a Gateway Integration Endpoint 890
Create the Gateway Setup Group 890
Link the Gateway Object Type to Setup Group 891
Create an Instance of the Gateway Endpoint Object 892
Creating a Gateway Integration Endpoint 894
Configuring a Gateway Integration Endpoint 895
Proxy Configuration Setup 897
Running a Gateway Integration Endpoint 898
Maintaining a Gateway Integration Endpoint 900
Monitoring a Gateway Integration Endpoint 901
Monitor from STEP 901
Monitor Outside of STEP 901
View all endpoints 901
View a specific gateway endpoint 901
Gateway Integration Endpoint Statistics 903
Exporting Gateway Integration Endpoint Definitions as Comments 904
Import Manager 905
Setup Requirements 905

Additional Information 905
Creating a Data Import 906
Import Manager - Select Configuration 908
Import Manager - Select Data Source 910
Using FTP 910
Using File 911
Import Manager - Select Format 912
Import Manager - Map Data 914
Import Manager - Identify Objects 915
Importing With Object IDs 915
Importing with Unique Key 916
Importing Without Object IDs 916
Import Manager - Identify Destination 919
Import Manager - Select Business Rules .. 921
Business Rule Limitations 923
Import Manager - Advanced Settings 924
General Advanced Settings 925
Remove Options 926
Adding and Removing Un-Mapped References 931
Running a Data Import 933
Maintaining a Saved Import Configuration 935
View a Saved Import Configuration 935
Edit a Saved Import Configuration 935
Monitoring a Data Import 937
Monitoring the Import Process 937
Properties Flipper 938
Execution Report 939
Reimporting a Corrected File 940
Import Error Messages 941
Import Error Message Examples 945
Illegal Unit for Attribute 945
Unknown Parent Product 945
Maximum Length, Illegal Value 946
Out of Range, Illegal Value 947
Not a Number, Illegal Value 947
Object Existed with Another Object Type 948
Unknown Classification 949
Not Privileged to Create Object 949
Unknown Asset ID 950
Array Operation Failed 950
Optimistic Locking Errors 950
The Transaction has been Rolled Back: Unique Constraint 950
Inbound Integration Endpoints 955
Setup Requirements 955
Additional Information 955
Initial Setup for an Inbound Integration Endpoint 957
Create the Inbound Integration Setup Group 958
Link the IIEP Object Type to Setup Group ... 959
Create an Instance of the IIEP Object 960
Creating an Inbound Integration Endpoint 962
IIEP - Identify Endpoint 964
IIEP - Choose Receiver 965
IIEP Receiver Methods 966

Amazon SQS Receiver **968**

Hotfolder Receiver **970**

Hotfolder Receiver Using File Sequence ..**972**

Hotfolder Receiver Using Meta Files **974**

IBM WebSphere MQ SSL Receiver **976**

JMS Receiver **978**

Configuration Examples 979

FileInitialContextFactory JMS Receiver ...980

JMS Websphere Delivery Using SSL 980

Apache ActiveMQ JMS Receiver 981

Dynamic JMS Receiver **982**

Dynamic JMS Configuration Examples 983

Azure Message Bus 983

TibcoJMS 983

Apache Active MQ 5.15.8 984

Apache RabbitMQ 1.11.1 984

WebLogic JMS Connection 986

Oracle AQ Receiver **987**

Product Data Syndication Receiver **989**

REST Receiver **991**

Web UI File Loading Receiver **993**

IIEP - Configure Endpoint **994**

IIEP - Configure PreProcessor **997**

IIEP - Configure Transformation by XSLT PreProcessor **998**

Initial Setup for XSLT Stylesheets 998

Create an XSLT Stylesheet Object 1002

Configure the Transformation by XSLT Preprocessor 1004

IIEP - Configure XML Normalizer PreProcessor **1006**

IIEP - Configure Processing Engine **1012**

IIEP - Configure Asset Importer Processing Engine **1013**

Metadata File Format 1018

IIEP - Configure Business Rule Based Message Processor Processing Engine . **1019**

Splitter Business Action Details and Examples 1020

JSON Message Splitter with JavaScript Business Action Example 1021

CSV Message Splitter with JavaScript Business Action Example 1022

Importer Business Action Details and Examples 1022

JSON Message Importer with JavaScript Business Action Example 1023

IIEP - Configure FAB-DIS Importer Processing Engine **1026**

IIEP - Configure STEP Importer Processing Engine **1027**

IIEP - Configure Match and Merge Importer **1030**

Match and Merge Importing Process and Logic 1030

Configuration Wizard Steps 1031

Source Record ID in STEPXML 1034

IIEP - Configure PostProcessor **1036**

IIEP - Schedule Endpoint **1037**

IIEP - Configure Error Reporter **1040**

Running an Inbound Integration Endpoint 1042

Enable and Invoke an Inbound Integration Endpoint 1042

Disable an Inbound Integration Endpoint ... 1044

Maintaining an Inbound Integration Endpoint 1046

Inbound Integration Endpoint Tab 1046

Description Flipper 1046

Configuration Flipper 1047

Receiver Configuration Flipper 1048

Integration Endpoint Log Flipper 1049

Background Processes Tab 1049

Statistics Tab 1050

Error Log Excerpts Tab 1050

Log Tab 1051

Status Tab 1051

Monitoring an IIEP via Background Process 1052

Selecting the Background Process Queue . 1052

Background Process Queue Size Property 1053

Monitoring an IIEP via External System .. 1054

Performance Data 1054

Monitoring Sensors 1054

REST API Monitoring 1055

Import Error Messages 1056

Import Error Message Examples 1060

Illegal Unit for Attribute 1060

Unknown Parent Product 1060

Maximum Length, Illegal Value 1061

Out of Range, Illegal Value 1062

Not a Number, Illegal Value 1062

Object Existed with Another Object Type ... 1063

Unknown Classification 1064

Not Privileged to Create Object 1064

Unknown Asset ID 1065

Array Operation Failed 1065

Optimistic Locking Errors 1065

The Transaction has been Rolled Back: Unique Constraint 1065

Additional Information About Inbound Integration Endpoints 1070

Exporting Inbound Integration Endpoint Definitions as Comments 1071

Handling Failed IIEP Background Processes 1073

Disable and Enable a Background Process 1073

Resume a Background Process 1073

Resolving a Failed Background Process ... 1074

Inbound Integration Endpoint Structure . 1077

Receiver 1077

Preprocessor 1077

Processing Engine / Converter 1077

Error Reporter 1078

Postprocessor 1078

Integration Endpoint Transactional Settings 1079

Transactional Settings Options 1081

Reference Target Lock Policy on Object Types **1083**

Outbound Integration Endpoints **1084**

Event-Based Outbound Integration Endpoint 1084

Setup Requirements for Event-Based OIEPs 1084

Additional Information for Event-Based OIEPs 1085

Select Objects Outbound Integration Endpoint 1085

Setup Requirements for Select Objects OIEPs 1085

Additional Information for Select Objects OIEPs 1086

Initial Setup for an Outbound Integration Endpoint **1087**

Create the Outbound Integration Setup Group 1087

Link the OIEP Object Type to Setup Group 1089

Create an Instance of the OIEP Object 1090

Creating an Outbound Integration Endpoint **1092**

Creating an Event-Based Outbound Integration Endpoint **1093**

OIEP - Event Based - Identify Endpoint .. **1095**

OIEP - Event-Based - Choose Data Source **1096**

OIEP - Event Based - Configure Endpoint **1097**

OIEP - Event-Based - Manual Configuration **1099**

OIEP - Configuration Flipper **1104**

Process Engine 1104

Error Reporter 1104

Schedule 1105

Queue for Endpoint 1108

Queue for Endpoint Processes 1108

Transactional Settings 1108

Number of Threads 1108

Maximum Number of Waiting Processes ... 1108

Maximum Number of Old Processes 1109

Maximum Age of Old Processes 1109

Contexts 1109

Workspace 1110

OIEP - Configuration Flipper for Business Rule Based Message Processor **1111**

Node Handler Business Action Details and Examples 1113

JSON Message Node Handler JavaScript Business Action Example 1114

Joiner Business Action Details and Examples 1115

JSON Message Joiner JavaScript Business Action Example 1116

OIEP - Event-Based - Output Templates Flipper **1117**

Configure the Object Types and Event Types 1117

Using Multiple OIEPs or Multiple Output Templates 1119

Multiple OIEPs 1119

Multiple Output Templates 1119

Limitations of Multiple Output Templates 1120
Configure the Format 1120
Configure the Pre-processor and Post-processor 1123
Pre-processor 1123
Post-processor 1124
OIEP - Pre-Processor - Business Action 1126
Business Action Configuration 1126
OIEP Configuration 1127
OIEP - Post-Processor - Context Splitter 1128
OIEP - Post-Processor - Copy Context Dependent Values and References 1130
OIEP - Event-Based - Pre and Post Processing Flipper 1133
OIEP - Delivery Method Flipper 1135
Amazon SQS 1135
Copy to Directory 1135
Deploy 1135
Email 1136
FTP 1136
IBM Websphere MQ SSL Delivery 1136
JDBC 1137
JMS 1137
Dynamic JMS 1138
Mongo 1138
Oracle AQ 1138
Product Data Syndication 1139
REST 1139
REST Direct 1139
SFTP 1140
Wiki 1140
OIEP Delivery Methods 1141
Amazon SQS Delivery Method 1144
Copy to Directory Delivery Method 1149
Deploy Delivery Method 1151
Email Delivery Method 1153
FTP Delivery Method 1154
IBM Websphere MQ SSL Delivery Method 1156
JDBC Delivery Method 1158
JMS Delivery Method 1160
Sharedconfig.properties File Examples 1162
Azure Message Bus 1162
Dynamic JMS Delivery Method 1165
Azure Message Bus 1166
TibcoJMS 1167
Apache Active MQ 5.15.8 1167
Apache RabbitMQ 1.11.1 1168
WebLogic JMS Connection 1169
Mongo Delivery Method 1171
Mongo Delivery Method Elements 1175
Default Databases and Collections 1177
STEP JSON and Mongo JSON 1178
JavaScript Triggers 1178
Bound Variables 1179
Example JavaScript to Maintain a Collection in a Classification 1179

Prerequisites for Configuring the MongoDB Adapter **1181**
Configure the OIEP 1181
Configuring the JavaScript Triggers 1183
MongoDB Adapter Setup Quick Guides **1184**
STEP JSON Schema 1184
Configuring Mongo Authentication Quick Guide 1184
SSL Configuration Quick Guide 1185
Mongo Delivery Method Conversion Example **1187**
Values 1187
References and Links 1189
Attribute Types 1192
Data Containers 1192
Reference Types 1194
Asset Push Locations 1194
Tables 1194
Table Formatting 1195
Attribute Links 1197
Data Container Type Links 1197
Data Containers Types 1198
Unit 1198
List Of Values 1199
Oracle AQ Delivery Method **1200**
Product Data Syndication Delivery Method **1201**
REST Delivery Method **1203**
REST Direct Delivery Method **1205**
Additional Request Information 1208
SFTP Delivery Method **1210**
Wiki Delivery Method **1212**
OIEP - Event-Based - Event Triggering Definitions Tab **1214**
Triggering Object Types Flipper 1215
Triggering Attributes Flipper 1220
Triggering Table Types Flipper 1221
Reference Type Triggers Flipper 1221
Triggering Data Container Types 1222
Miscellaneous Triggers Flipper 1223
OIEP - Event-Based - Event Queue Configuration Flipper **1225**
Creating a Select Objects Outbound Integration Endpoint **1227**
OIEP - Select Objects - Identify Endpoint 1229
OIEP - Select Objects - Choose Data Source **1230**
OIEP - Select Objects - Configure Endpoint **1232**
OIEP - Select Objects - Manual Configuration **1234**
OIEP - Configuration Flipper **1238**
Process Engine 1238
Error Reporter 1238
Schedule 1239
Queue for Endpoint 1242
Queue for Endpoint Processes 1242
Transactional Settings 1242
Number of Threads 1242

Maximum Number of Waiting Processes ...1242
Maximum Number of Old Processes 1243
Maximum Age of Old Processes 1243
Contexts 1243
Workspace 1244
OIEP - Configuration Flipper for Business Rule Based Message Processor 1245
Node Handler Business Action Details and Examples 1247
JSON Message Node Handler JavaScript Business Action Example 1248
Joiner Business Action Details and Examples 1249
JSON Message Joiner JavaScript Business Action Example 1250
OIEP - Select Objects - Object Selection Configuration Flipper 1251
OIEP - Select Objects - Output Templates Flipper 1254
Configure the Object Types 1254
Using Multiple OIEPs or Multiple Output Templates 1256
Limitations of Multiple Output Templates 1256
Configure the Format 1256
Configure the Pre-processor and Post-processor 1260
Pre-processor 1260
Post-processor 1261
OIEP - Pre-Processor - Business Action .1263
Business Action Configuration 1263
OIEP Configuration 1264
OIEP - Post-Processor - Context Splitter 1265
OIEP - Post-Processor - Copy Context Dependent Values and References 1267
OIEP - Event-Based - Pre and Post Processing Flipper 1270
OIEP - Delivery Method Flipper 1272
Amazon SQS 1272
Copy to Directory 1272
Deploy 1272
Email 1273
FTP 1273
IBM Websphere MQ SSL Delivery 1273
JDBC 1274
JMS 1274
Dynamic JMS 1275
Mongo 1275
Oracle AQ 1275
Product Data Syndication 1276
REST 1276
REST Direct 1276
SFTP 1277
Wiki 1277
OIEP Delivery Methods 1278
Amazon SQS Delivery Method 1281
Copy to Directory Delivery Method 1286
Deploy Delivery Method 1288
Email Delivery Method 1290
FTP Delivery Method 1291

IBM Websphere MQ SSL Delivery Method 1293

JDBC Delivery Method 1295

JMS Delivery Method 1297

Sharedconfig.properties File Examples 1299

Azure Message Bus 1299

Dynamic JMS Delivery Method 1302

Azure Message Bus 1303

TibcoJMS 1304

Apache Active MQ 5.15.8 1304

Apache RabbitMQ 1.11.1 1305

WebLogic JMS Connection 1306

Mongo Delivery Method 1308

Mongo Delivery Method Elements 1312

Default Databases and Collections 1314

STEP JSON and Mongo JSON 1315

JavaScript Triggers 1315

Bound Variables 1316

Example JavaScript to Maintain a Collection in a Classification 1316

Prerequisites for Configuring the MongoDB Adapter 1318

Configure the OIEP 1318

Configuring the JavaScript Triggers 1320

MongoDB Adapter Setup Quick Guides .. 1321

STEP JSON Schema 1321

Configuring Mongo Authentication Quick Guide 1321

SSL Configuration Quick Guide 1322

Mongo Delivery Method Conversion Example 1324

Values 1324

References and Links 1326

Attribute Types 1329

Data Containers 1329

Reference Types 1331

Asset Push Locations 1331

Tables 1331

Table Formatting 1332

Attribute Links 1334

Data Container Type Links 1334

Data Containers Types 1335

Unit 1335

List Of Values 1336

Oracle AQ Delivery Method 1337

Product Data Syndication Delivery Method 1338

REST Delivery Method 1340

REST Direct Delivery Method 1342

Additional Request Information 1345

SFTP Delivery Method 1347

Wiki Delivery Method 1349

Running an Outbound Integration Endpoint 1351

Enable and Invoke an Outbound Integration Endpoint 1351

Disable an Outbound Integration Endpoint .1354

Maintaining an Outbound Integration Endpoint 1355

Outbound Integration Endpoint Tab 1355
Description Flipper 1355
Integration Endpoint Log Flipper 1355
Configuration Tab 1357
Configuration Flipper 1357
Event Queue Configuration Flipper 1359
Object Selection Configuration Flipper ...1360
Output Templates Flipper 1360
Delivery Method Flipper 1361
Background Processes Tab 1361
Statistics Tab 1362
Error Log Excerpts Tab 1363
Log Tab 1363
Status Tab 1363
Monitoring an OIEP via Background Process 1365
Selecting the Background Process Queue .1366
Background Process Queue Size Property 1366
Monitoring an OIEP via External Systems 1367
Performance Data 1367
Monitoring Sensors 1367
REST API Monitoring 1368
Additional Information About All Outbound Integration Endpoints 1369
Exporting Outbound Integration Endpoint Definitions as Comments 1370
Handling Failed OIEP Background Processes 1372
Disable and Enable a Background Process 1372
Resume a Background Process 1372
Resolving a Failed Background Process ... 1373
Integration Endpoint Transactional Settings 1375
Transactional Settings Options 1377
Outbound Integration Endpoint Structure 1379
Business Rules Event Processing 1379
Processing Engine / Converter 1379
Error Reporter 1380
Post-Processor 1380
Delivery 1380
Additional Information About Event-Based OIEPs 1381
Event-Based OIEP Event Batching 1382
Event-Based OIEP Examples 1383
Event-Based Example Business Rules for Derived Events 1384
Event-Based Example Leaf Products with Inherited Data 1389
Event-Based Example Products Linked to Classification 1392
Event-Based Example Upward Inheritance 1396
Event-Based OIEP Forward, Rewind, Purge, and Republish 1399
Forward Events 1400
Rewind Events 1401
Purge Events 1401
Republish Events 1402
Event-Based OIEP Multithreading 1405

Support .....

Multithreading Setup 1405

Additional Considerations 1406

Increasing the Batch Size 1406

Serialized and Parallelized 1406

Event-Based OIEP Queued Events **1407**

Estimate Unread Events 1407

View First Event Batch 1407

Event-Based OIEP Status and Queue Status **1410**

Set the Queue Status 1410

Web Service Endpoint **1412**

Prerequisites 1412

Data Exchange

Data stored in STEP can be exchanged and integrated with external systems. For example, this allows STEP 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, see **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, see **Assets and Content with STEPXML**. For exchanging assets independent of data, see 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 (see **Data Formats**).

Receiver Method

For an IIEP, the selected receiver method determines how STEP accesses the inbound message (see **IIEP Receiver Methods**). 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 (see **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 (see **OIEP Delivery Methods** and the **Export Manager - Select Delivery Method**).

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 only Web Service supported by STEP is a Matching and Merging Web Service Endpoint. For more information, see the **Web Service Endpoint** topic in this documentation.

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.

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. See **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. See **Alphabetical Index - XML Format** documentation.	No	Yes
Ariba CIF	CIF is a simple, comma-delimited list of catalog items and their attributes. See **Ariba CIF Format** documentation.	No	Yes
Ariba CIF 3.0	CIF is a simple, comma-delimited list of catalog items and their attributes. See **Ariba CIF 3.0 Format** documentation.	No	Yes
BMEcat	XML-based standard for electronic data transfer by electronic catalogs. See **BMEcat Format** documentation.	Yes	Yes
BMEcat 2005	XML-based standard for electronic data transfer by electronic catalogs. See **BMEcat 2005 Format** documentation.	Yes	Yes
CSV	Comma-separated values (CSV) file. See **CSV Format** documentation.	Yes	Yes
cXML	XML 'commerce eXtensible Markup Language' (cXML). See **cXML Format** documentation.	No	Yes
eCl@ss	eCl@ss® is a hierarchical classification and product description system for grouping materials, products, and services. See **eCl@ss Format** documentation. *This import is only available via the eCl@ss Classification Import	Yes*	No

Format	Description	Inbound	Outbound
	wizard.		
ETIM	CSV Electro-Technical Information Model (ETIM) classification structure where each file contains a single language. See **ETIM and ETIM v2 Format** documentation.	Yes	No
ETIM IXF	XML Electro-Technical Information Model (ETIM) classification structure that includes multiple languages. See **ETIM IXF Format** documentation.	Yes	No
ETIM v2	CSV Electro-Technical Information Model (ETIM) classification structure where each file contains a single language. See **ETIM and ETIM v2 Format** documentation.	Yes	No
Excel	Microsoft Excel spreadsheet file. See **Excel Format** documentation.	Yes	Yes
Excel Custom Template	Microsoft Excel spreadsheet file. See **Excel Custom Template** documentation.	No	Yes
Excel Smartsheet	Smartsheet spreadsheet file. See **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. See **FAB-DIS Format** documentation.	Yes	Yes
FixedWidth	Text file with rows and columns of data, where each column has a fixed width. See **FixedWidth Format** documentation.	Yes	No
Flatplan Excel	Excel file for publications. See **Flatplan Excel Format**. *This export is only available via Export Manager.	Yes	Yes*
Generic XML	XML with a variety of accessible formats. See **Generic XML Format** documentation.	Yes	Yes
IDoc MATMAS	XML format used when exchanging information with SAP. See **IDoc MATMAS 05 Format** documentation.	Yes	Yes

Format	Description	Inbound	Outbound
05			
Publication Excel	Excel file for publications. See **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. See **SmartLabel Format** documentation.	No	Yes
STEPXML	XML generated based on parameters displayed in a dialog. Unlike Advanced STEPXML format, no template is required. See **STEPXML Format** documentation.	Yes	Yes
UNSPSC	United Nations Standard Products and Services Code® (UNSPSC) multi-level classification folders. See **UNSPSC Format** documentation.	Yes	No
xCBL	XML Common Business Library (xCBL) for business-to-business e-commerce. See **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 (see 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, see **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 schema is extended continuously but should always be backwards compatible. For more information on legacy naming conventions, see 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 - see **Creating a Data Export**
- OIEP - see **Creating an Outbound Integration Endpoint**

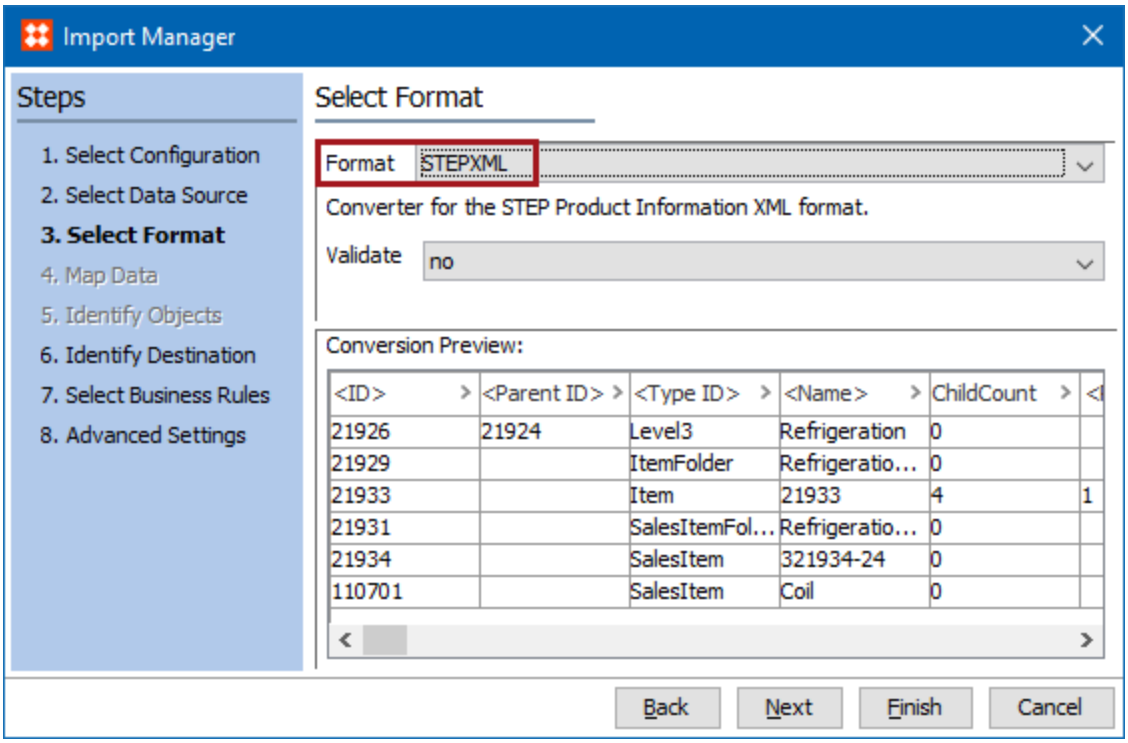
Mapping

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

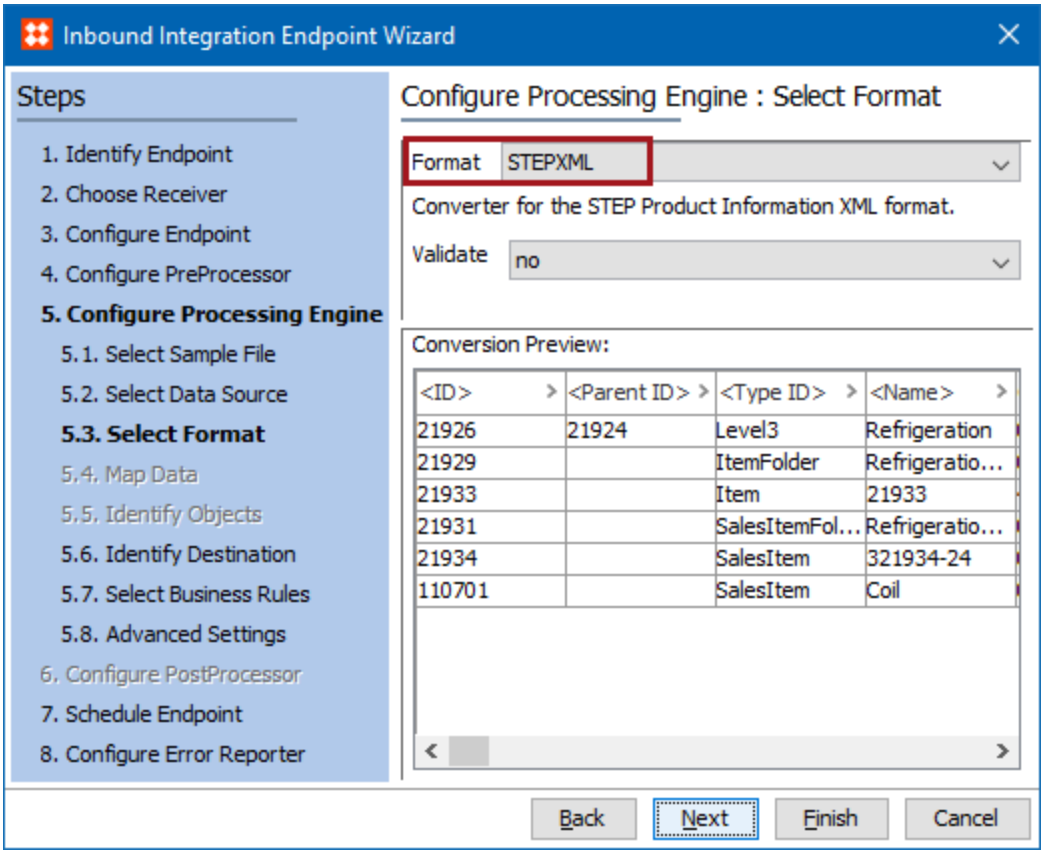
Inbound Data

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



Outbound Data

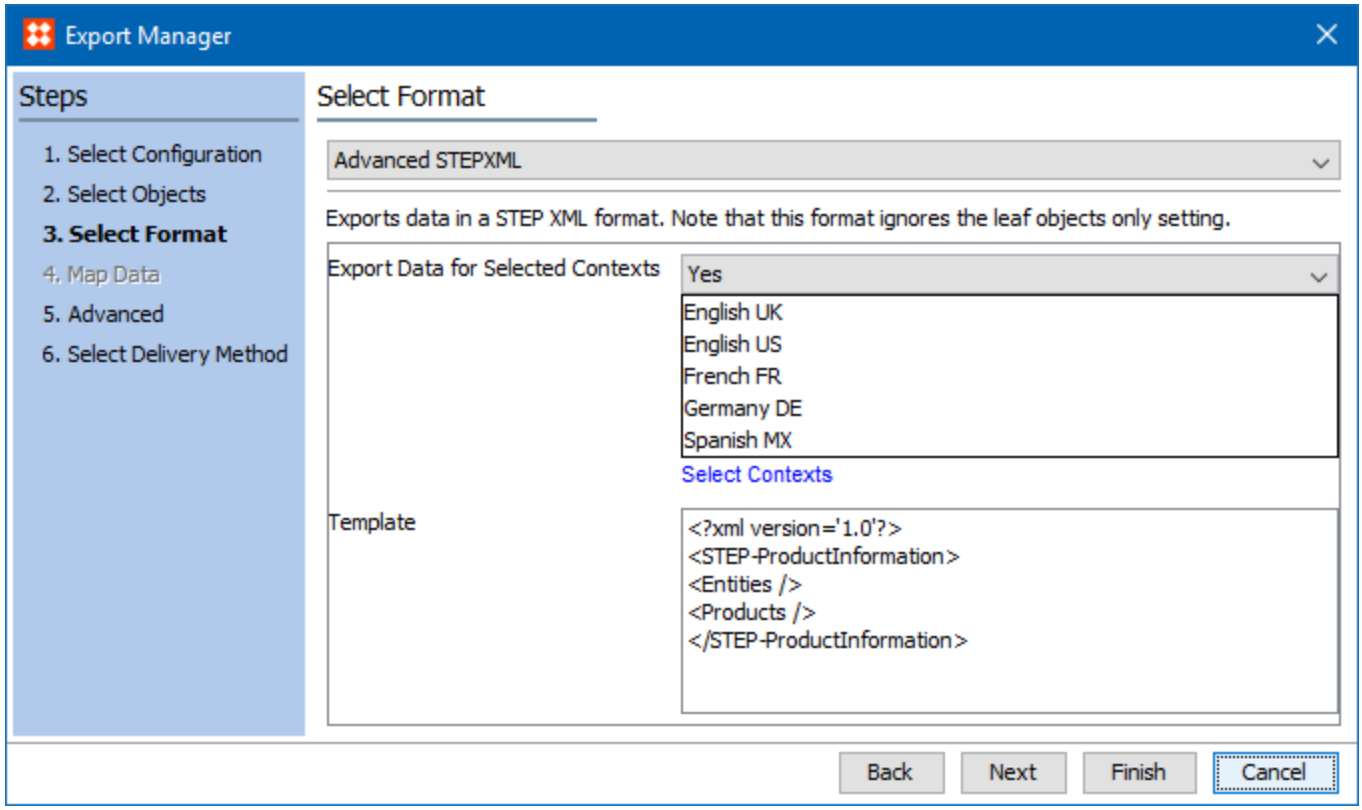
The following parameters are available for outbound data using the Export Manager and/or OIEPs, as defined below:

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.

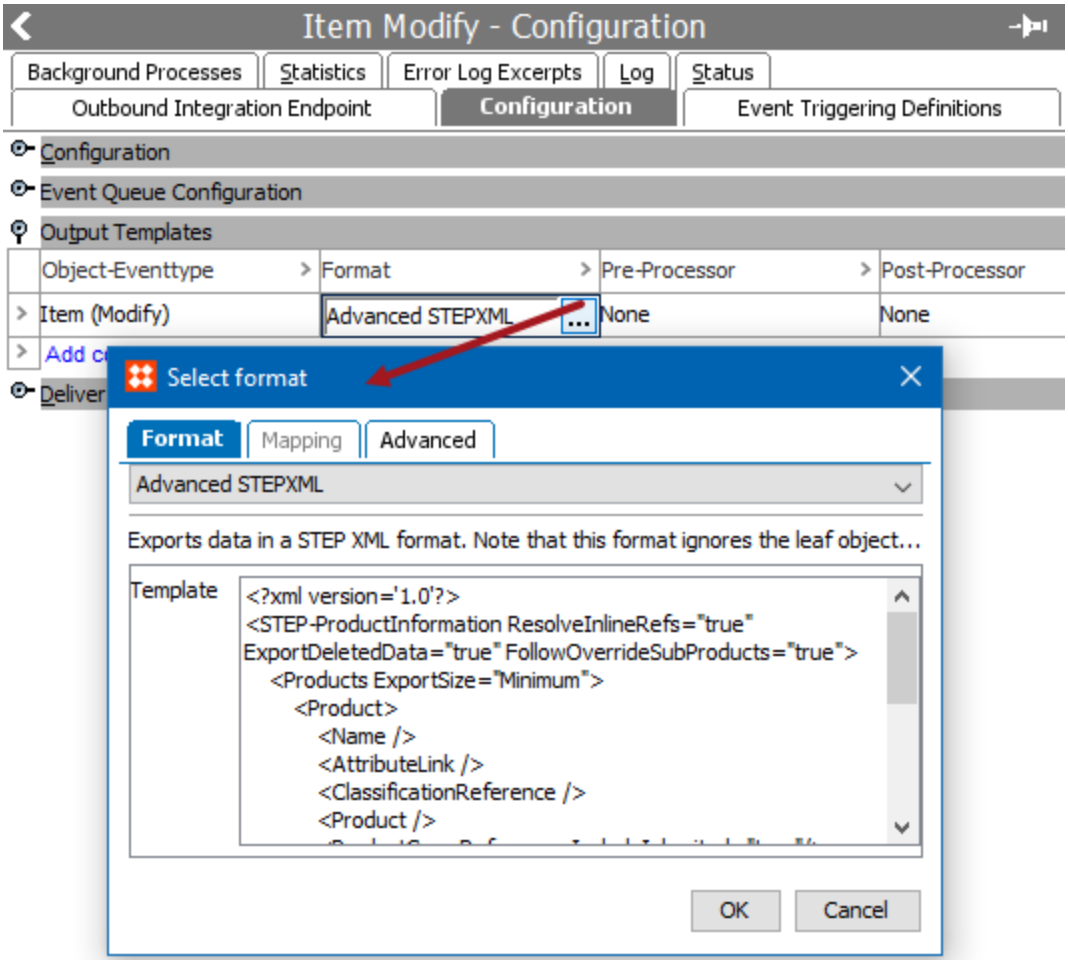
   Since no undo functionality is available in the Template field, it is good practice to use an external XML editor program for creating and editing a template. To test a template's validity for output, paste the XML into STEP.

   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 STEP 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, see the **Creating Document Indexes** topic or the **Exporting Index Data from STEP** topic in the **STEP Publisher** documentation.

Format Availability

Alphabetical Index - XML is available for selection in:

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

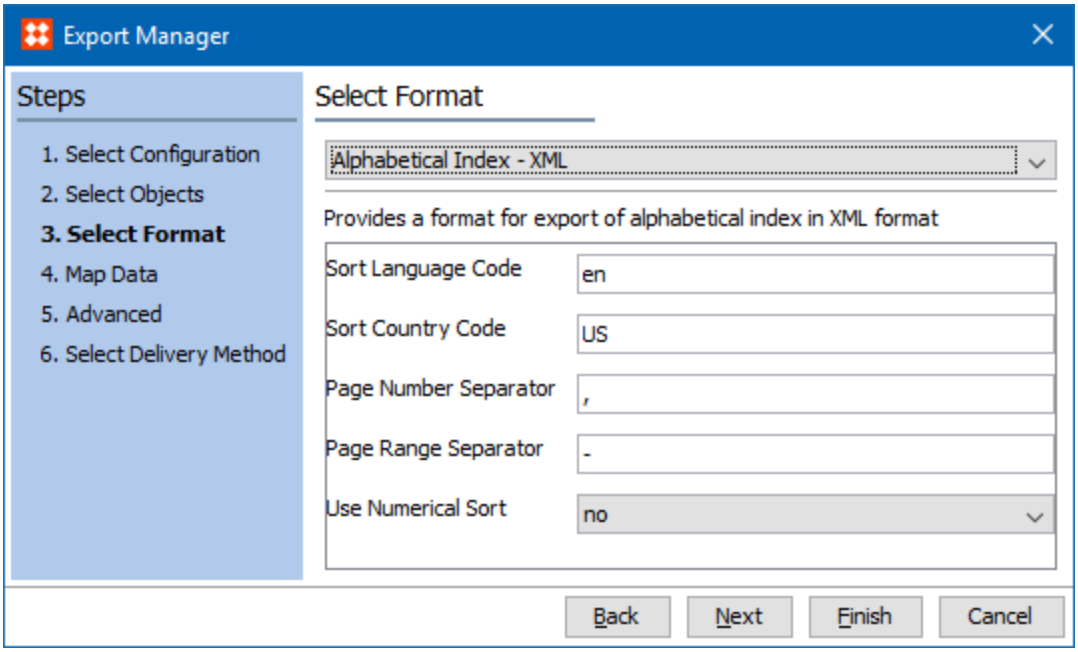
Mapping

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

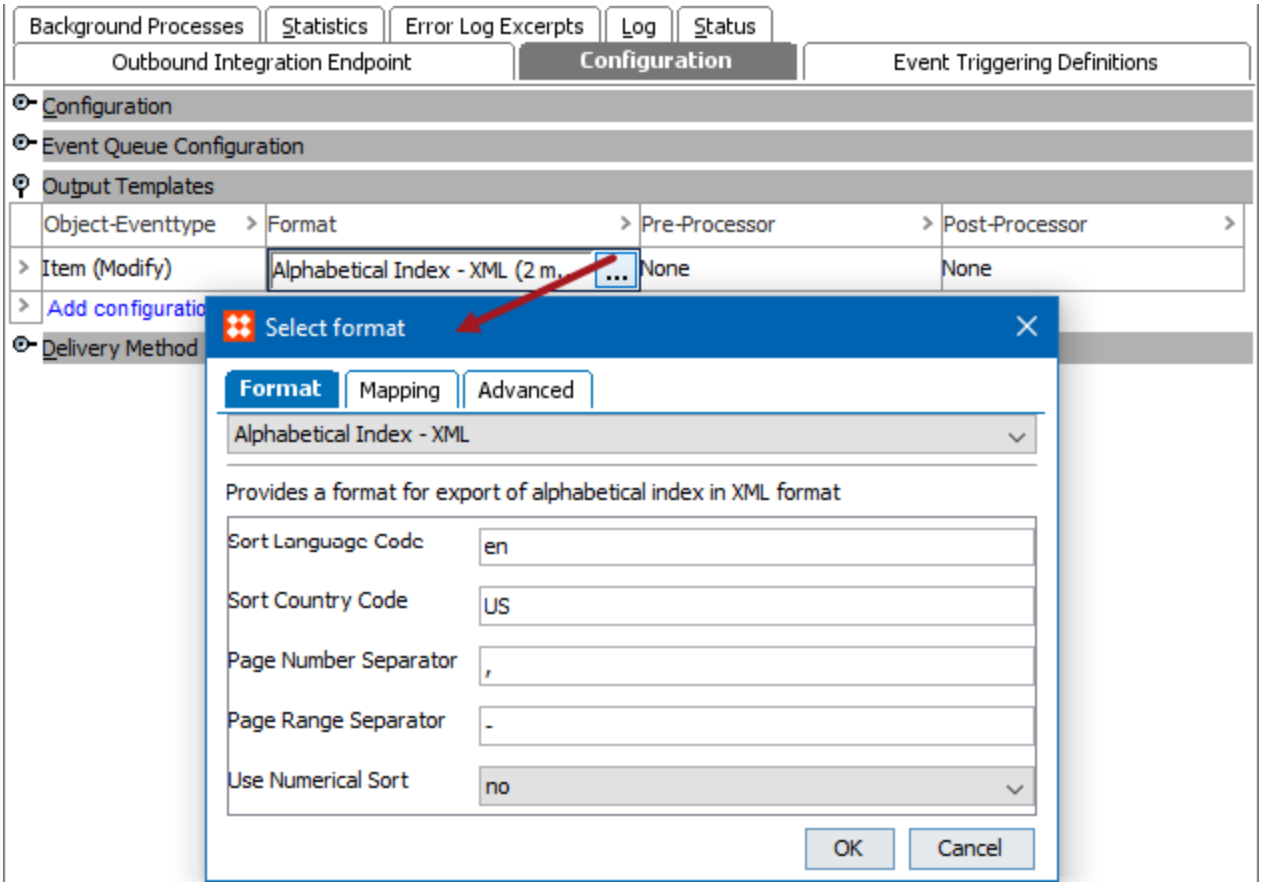
Outbound Data

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 - see **Creating a Data Export**
- OIEP - see **Creating an Outbound Integration Endpoint**

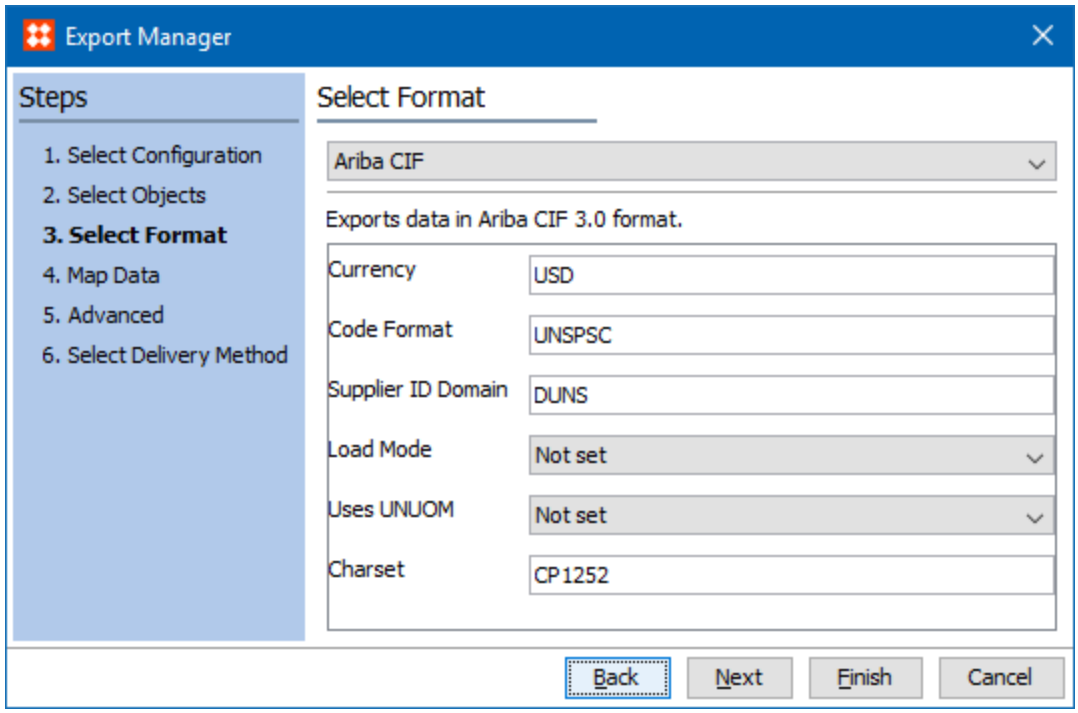
Mapping

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

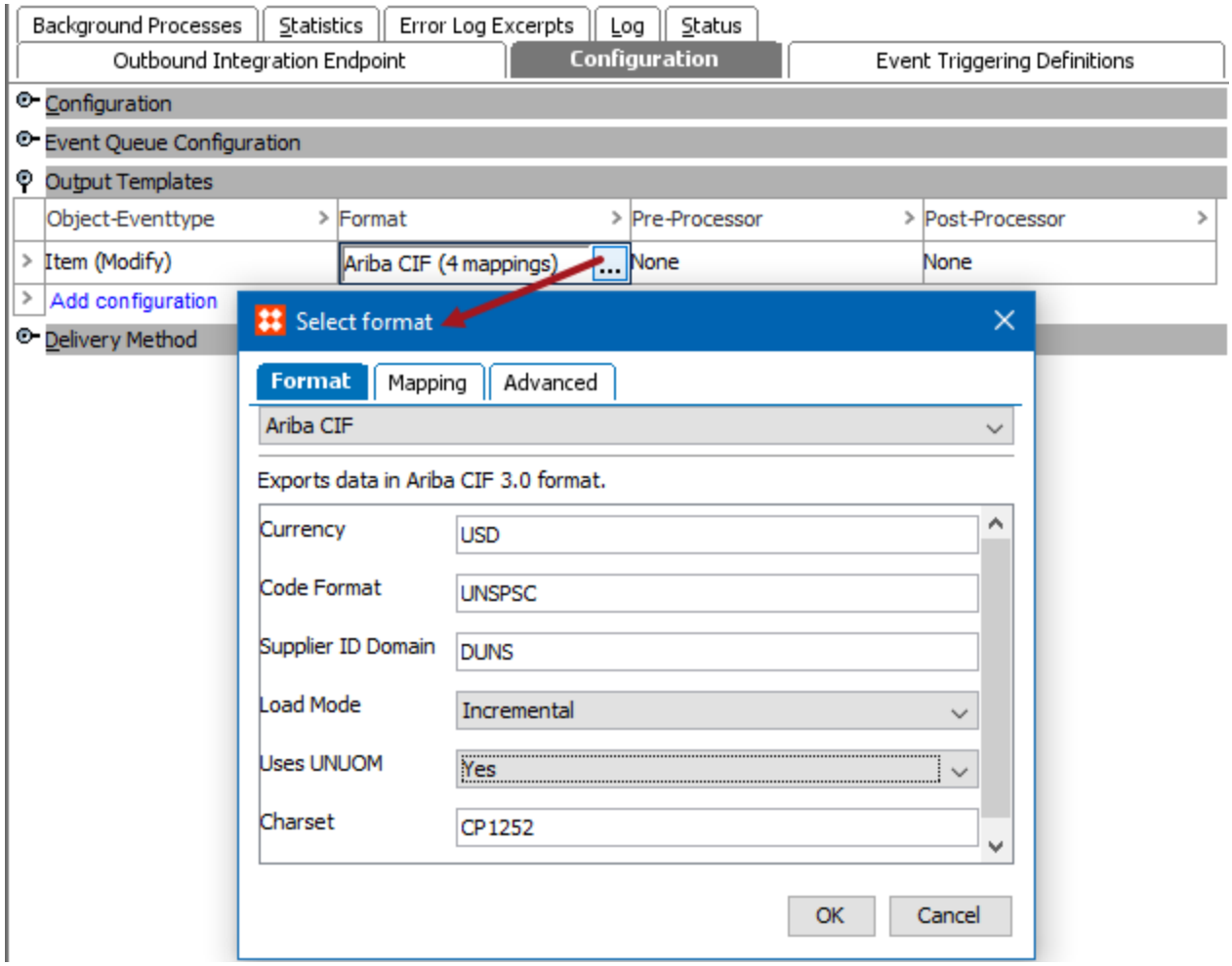
Outbound Data

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 - see **Creating a Data Export**
- OIEP - see **Creating an Outbound Integration Endpoint**

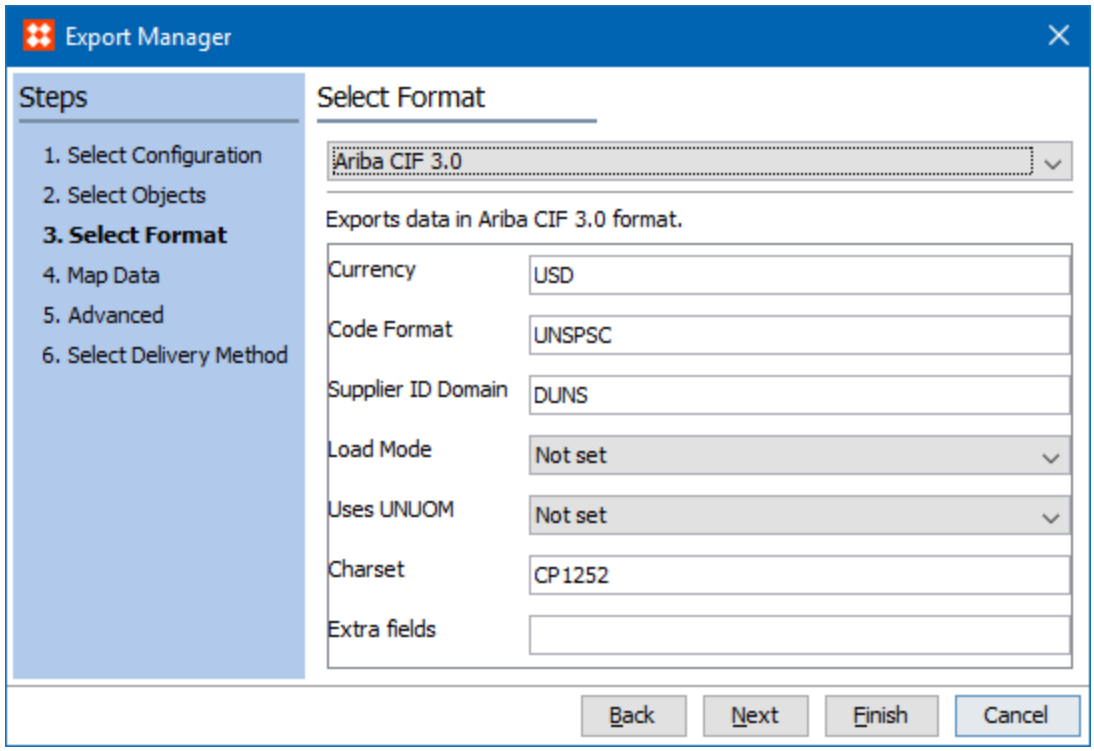
Mapping

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

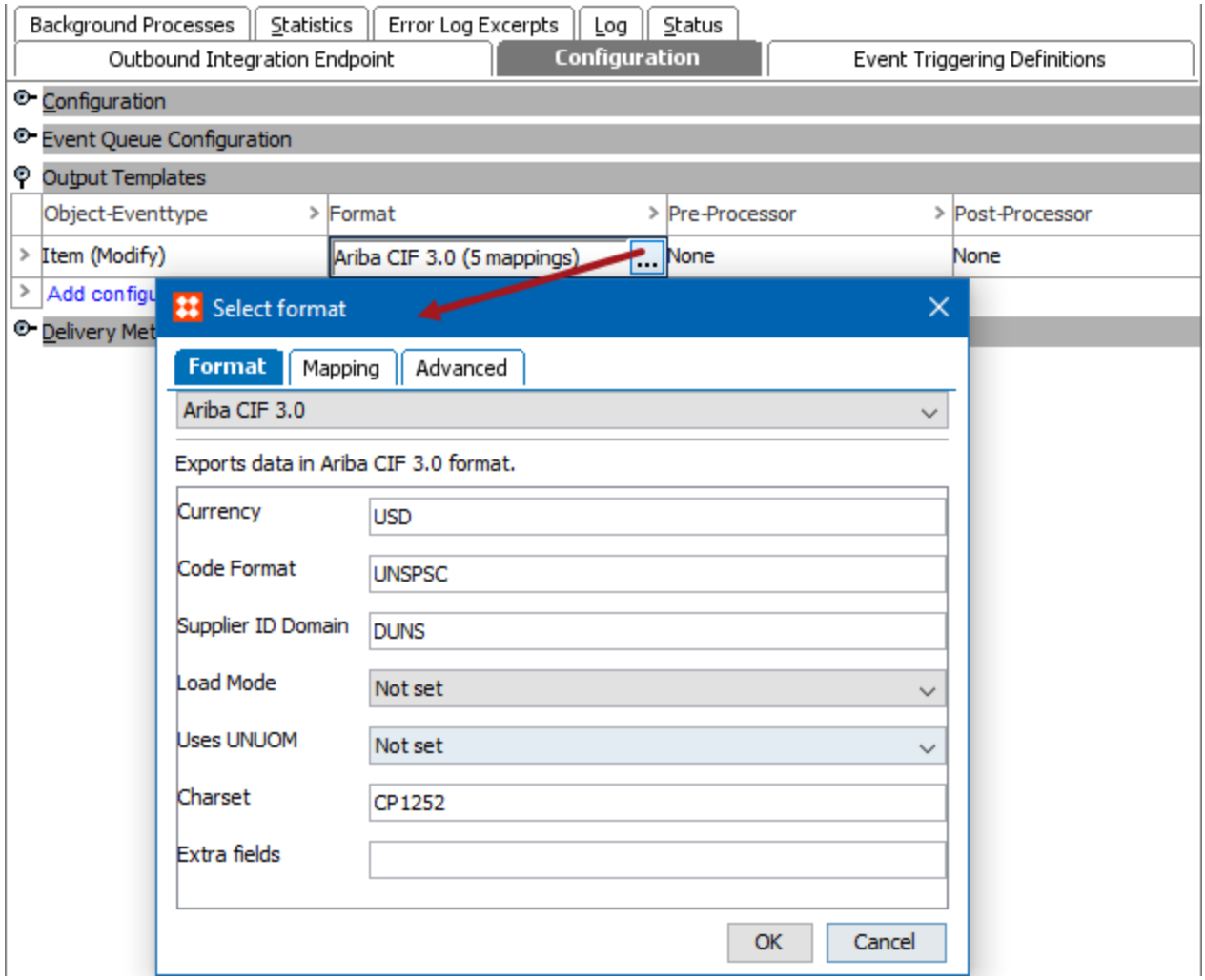
Outbound Data

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.

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

Format Availability

BMEcat is available for selection in:

- IIEP - see **Creating an Inbound Integration Endpoint**
- Import Manager - see **Creating a Data Import**
- Export Manager - see **Creating a Data Export**
- OIEP - see **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, see **Data Mapping**.

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, see **Aspect - Transform Outbound**.

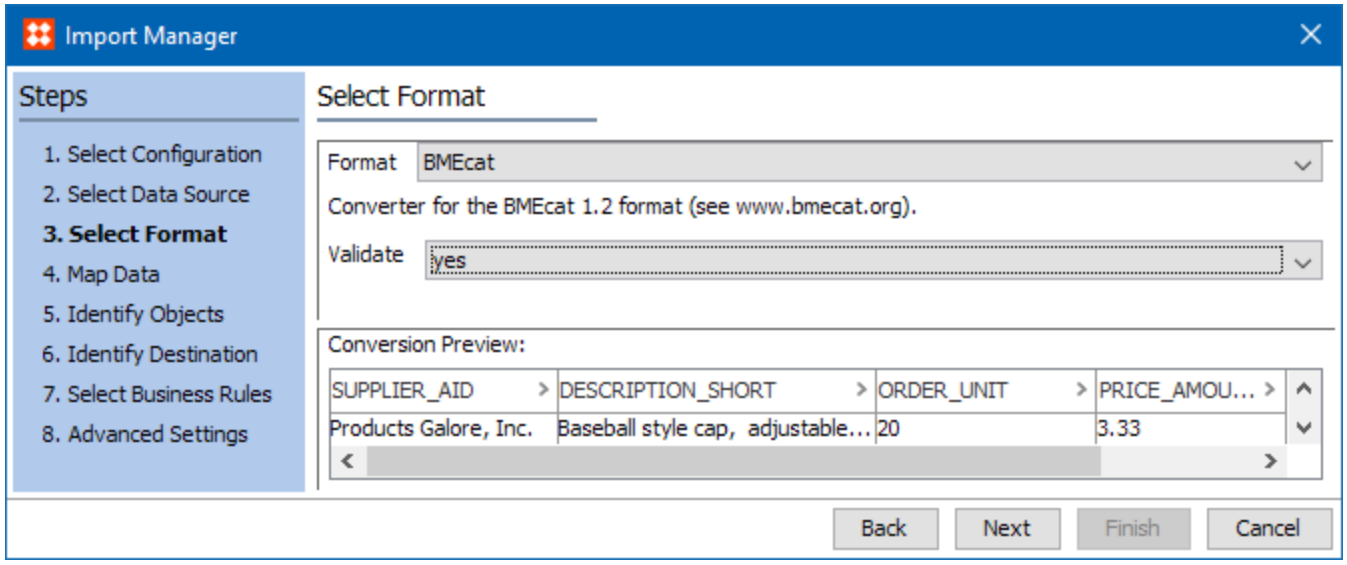
If mapping an ETIM file to the BMEcat2005 format, see the **LOV Value-ID or Value Aspect** topic.

Inbound Data

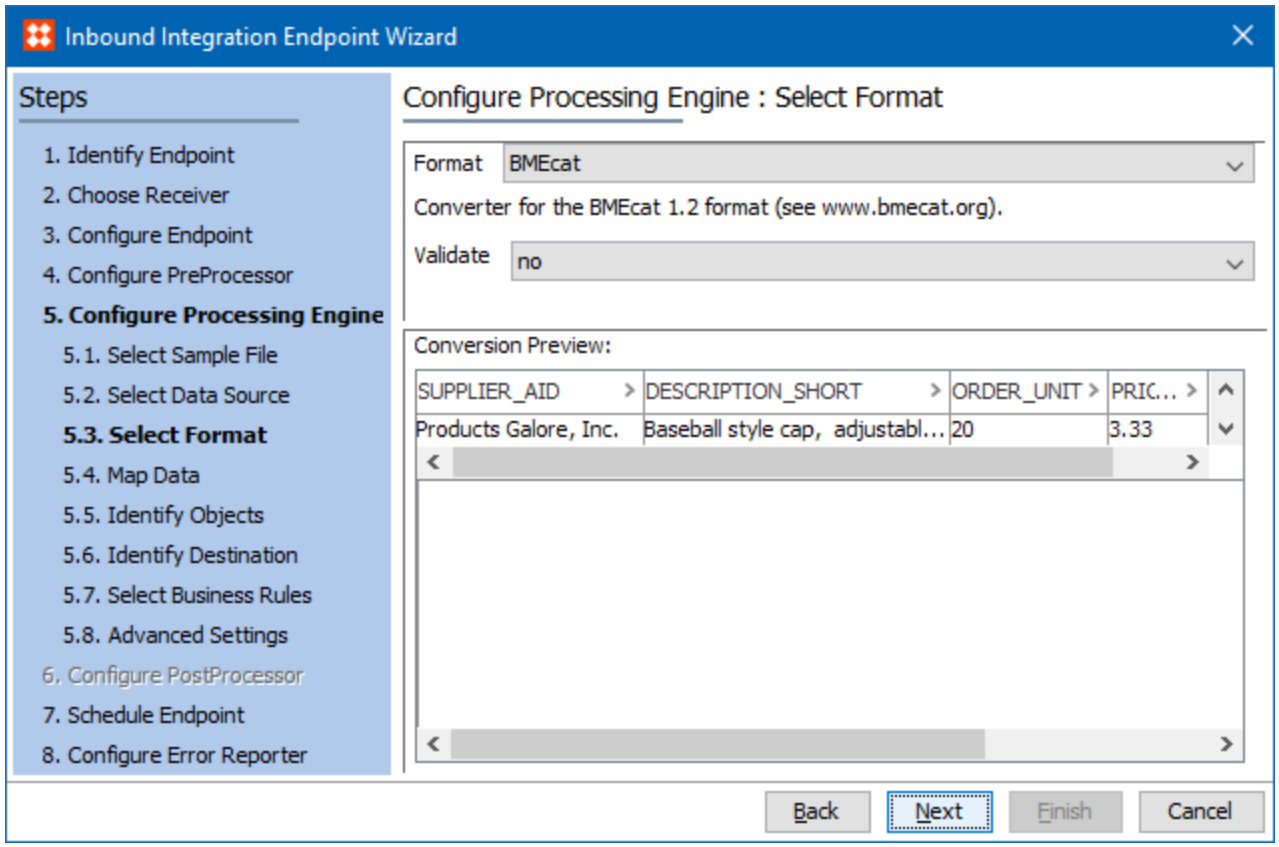
Inbound Parameters

- **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



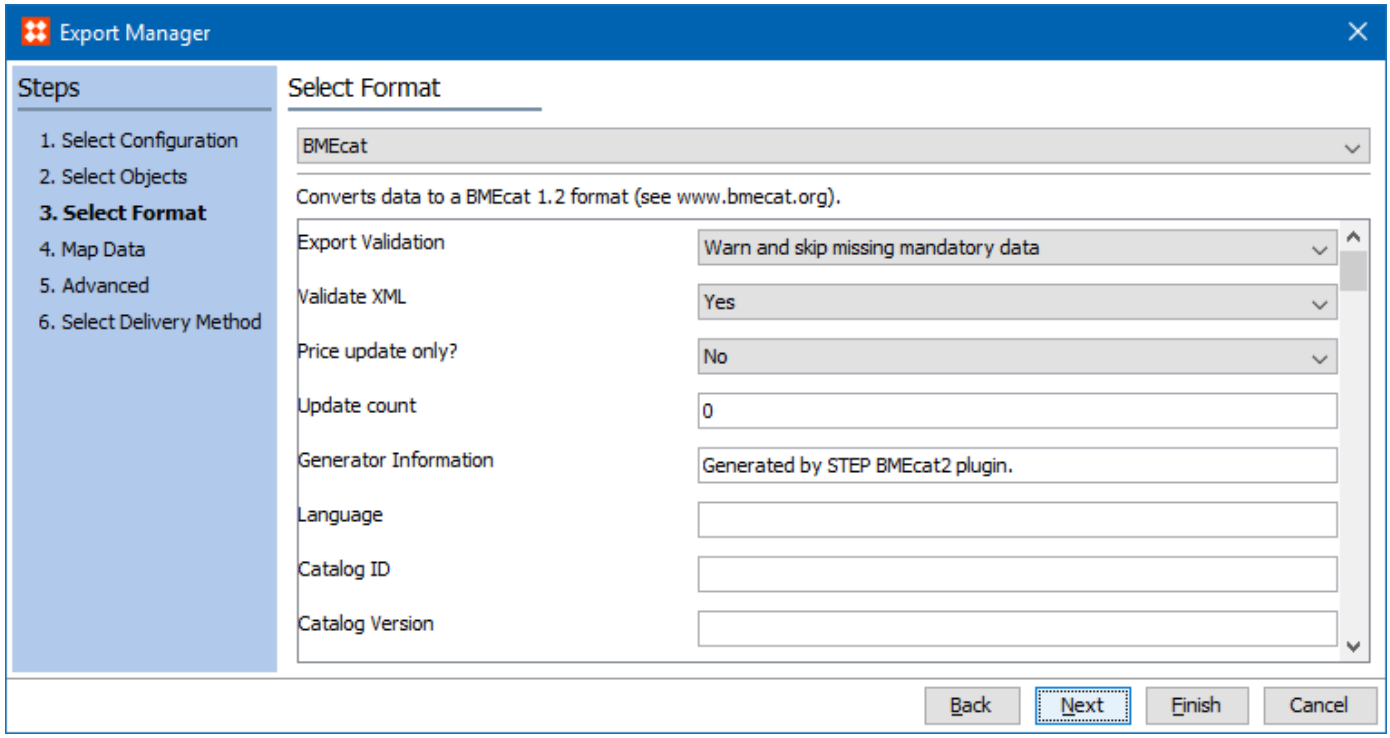
Outbound Data

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.

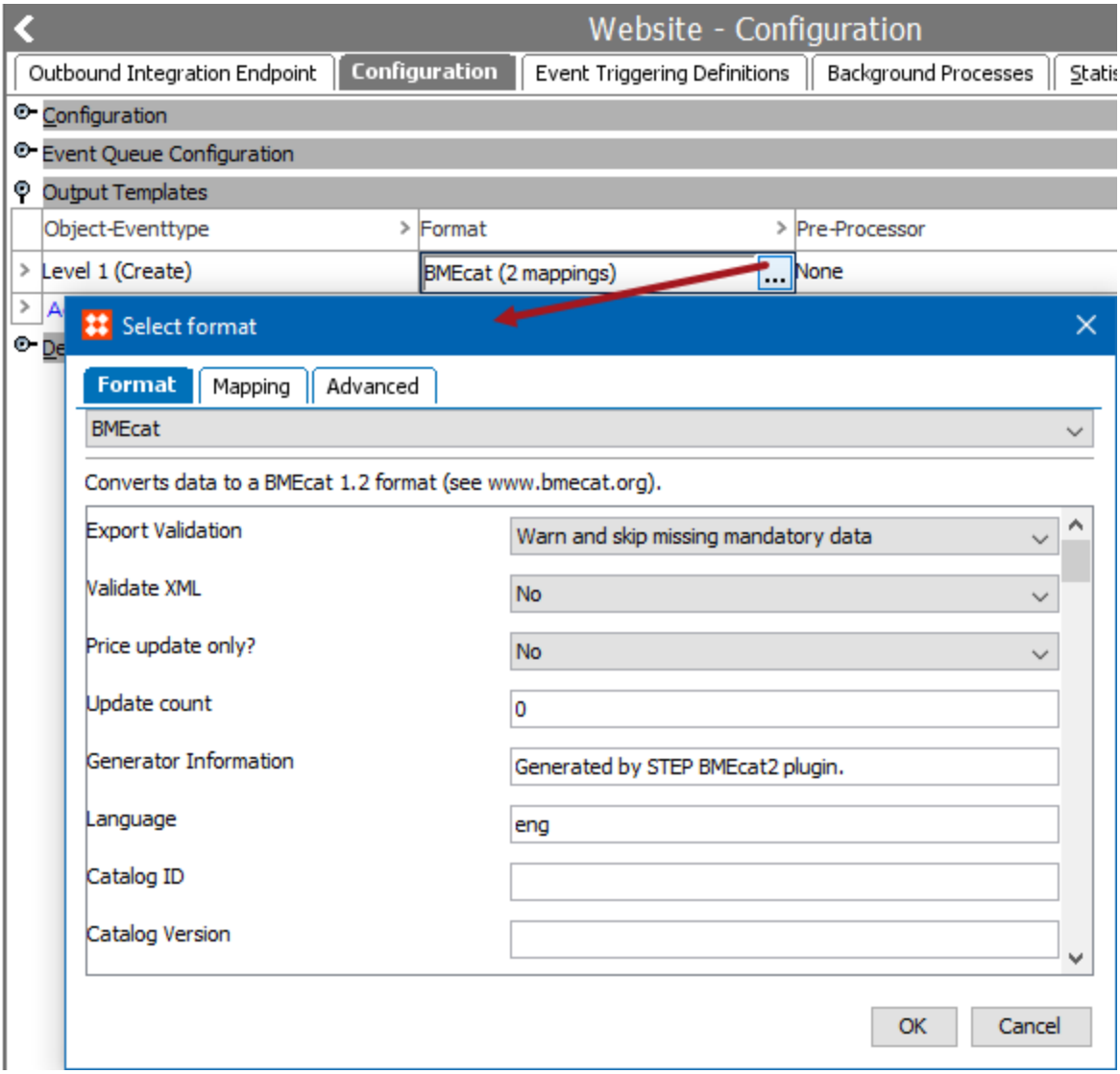
Outbound Parameters

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.
4. 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.
5. 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.
6. 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.
7. 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.

BMEcat 2005 is an add-on component. Contact your Stibo Systems representative to activate 'bmecat' on your STEP system.

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

Format Availability

BMEcat 2005 is available for selection in:

- IIEP - see **Creating an Inbound Integration Endpoint**
- Import Manager - see **Creating a Data Import**
- Export Manager - see **Creating a Data Export**
- OIEP - see **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, see **Data Mapping**.

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, see **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, see the **Inbound Map Data - Auto Map** topic in the **Data Mapping** documentation.

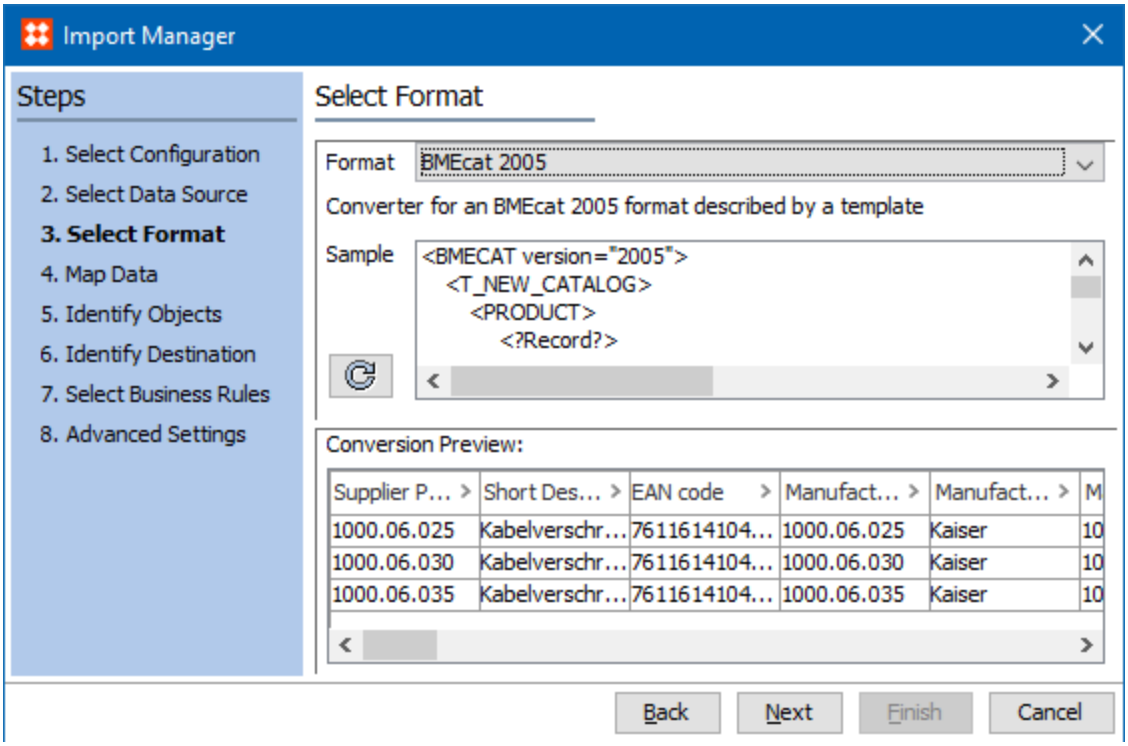
Inbound Data

Inbound Parameters

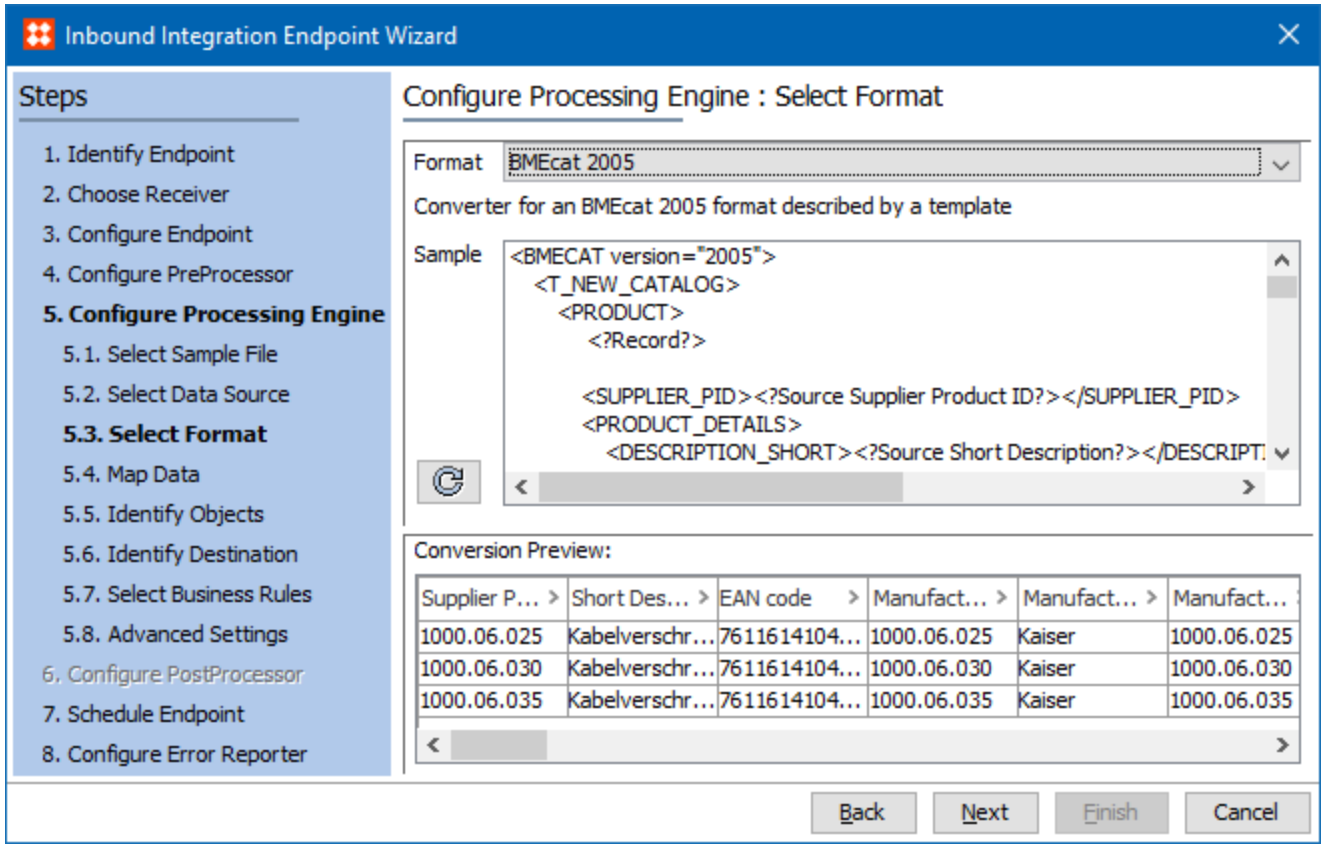
The following parameters are displayed on the inbound wizards:

- **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. 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.

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

Outbound Parameters

1. In the **Sample** parameter, an XML template file is loaded automatically but can be modified as required. This defines the format of the XML file to be exported.

Note: For more information on Generic XML instructions, see the **Generic XML Outbound Processing Instructions** topic in the **Generic XML Format** documentation

---

2. 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.

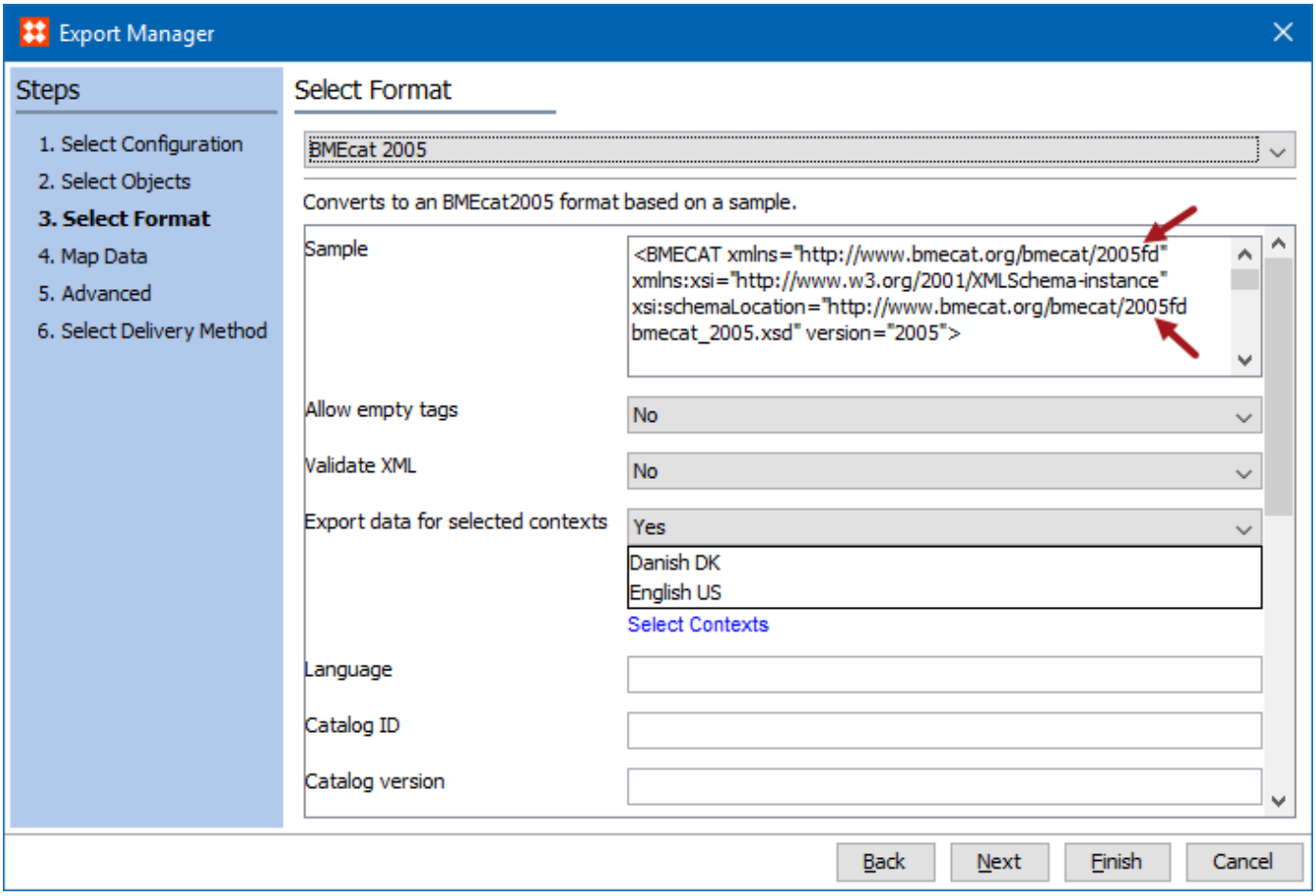
For example, setting this parameter to Yes enables all ETIM attributes, even those without a value in STEP, to be exported. Additionally, when ETIM values exported via BMEcat are empty, they should be represented by a dash (-). For information on transforming an empty attribute value to a text string,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

3. For the **Validate XML** parameter, **Yes** indicates that the BMEcat 2005 export file is validated against an XSD included in STEP. When validation fails, the background process also fails, and the problem is reported in the BGP execution report. If set to **No**, the validation is skipped and the BGP does not fail due to differences found when comparing to the XSD.

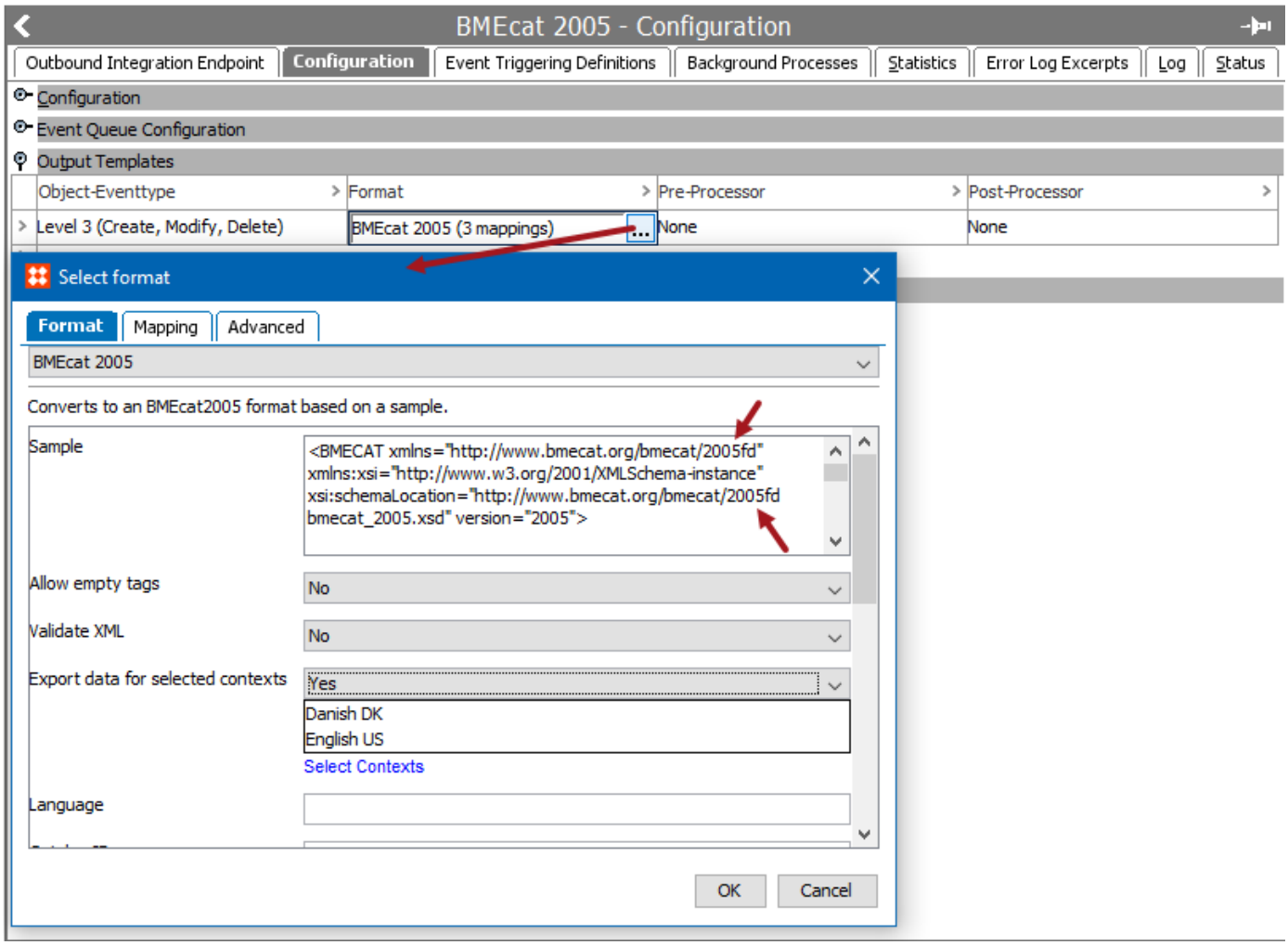
   Note: To successfully validate against the XSD, the sample file must include 'fd' option (as shown in two places in the images below).

4. The **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 selected contexts are listed in a text box. If set to **No**, the context set on Export Manager > Advanced step > Context parameter or set on OIEP > Configuration tab > Configuration flipper > Contexts parameter is used.
5. For an explanation of the remaining parameters, search the web to find and refer to the 'Specification BMEcat® 2005' 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


BMEcat 2005 - Configuration
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
Level 3 (Create, Modify, Delete)
BMEcat 2005 (3 mappings)
None
None
Select format
Format
Mapping
Advanced
BMEcat 2005
Converts to an BMEcat2005 format based on a sample.
Sample
<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">
Allow empty tags
No
Validate XML
No
Export data for selected contexts
Yes
Danish DK
English US
Select Contexts
Language
OK
Cancel

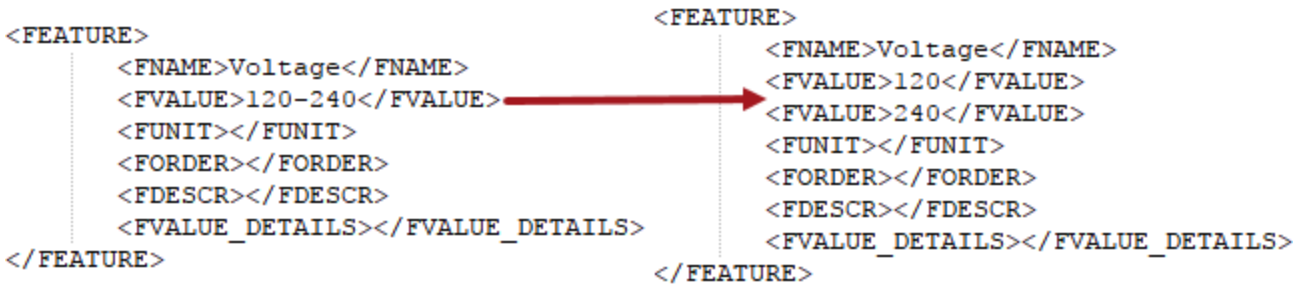
BMEcat 2005 for ETIM Data Configuration

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

Configuring the Export

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.

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, see the **Attributes Inherited from Classification - Data Source Outbound** topic in the **Data Mapping** 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 (objects and definitions).
- 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.
- 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 - see **Creating an Inbound Integration Endpoint**
- Import Manager - see **Creating a Data Import**
- Export Manager - see **Creating a Data Export**
- OIEP - see **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, see **Data Mapping**.

Inbound Data

CSV import allows creation of and updates to products, classifications, entities, attribute values and references. However, system setup objects (for example, attributes, 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, see **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

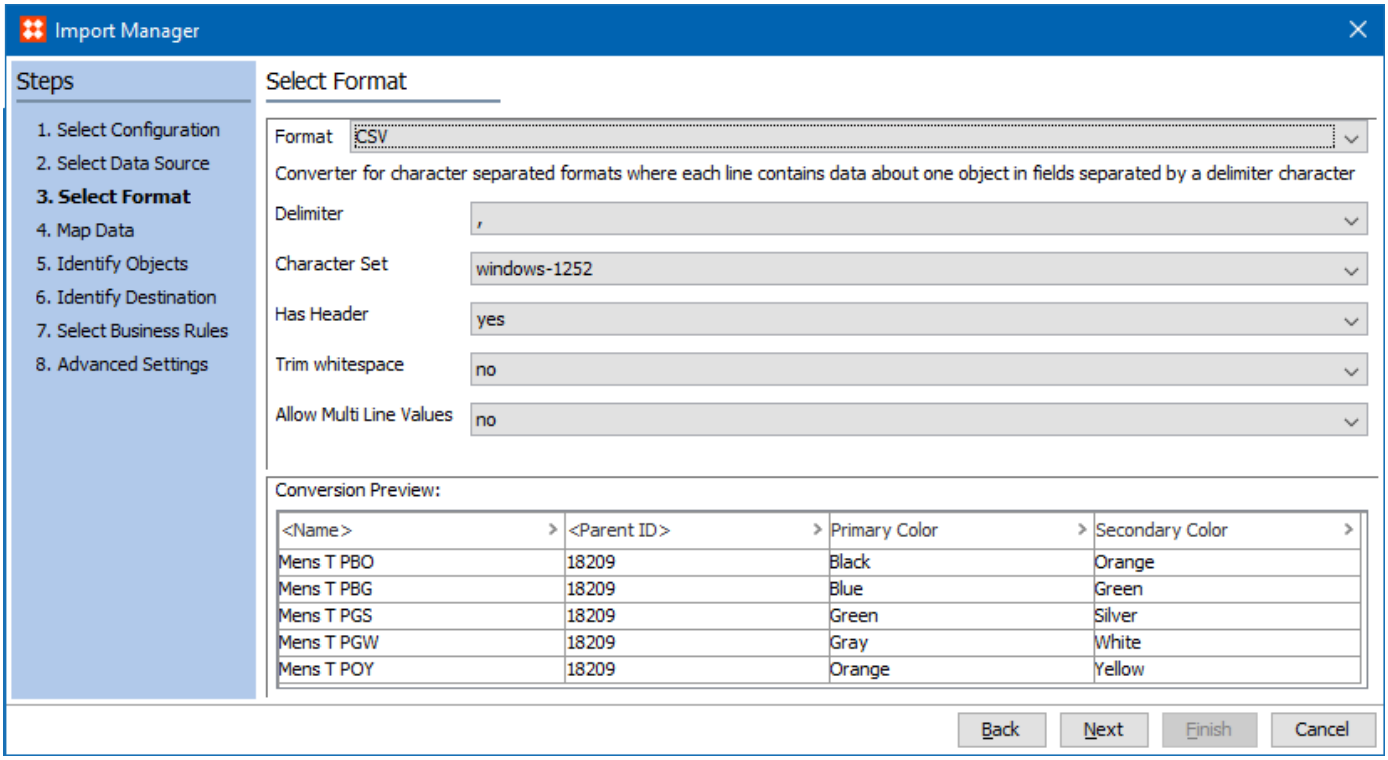
The following parameters are available in both Import Manager and IIEP.

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.

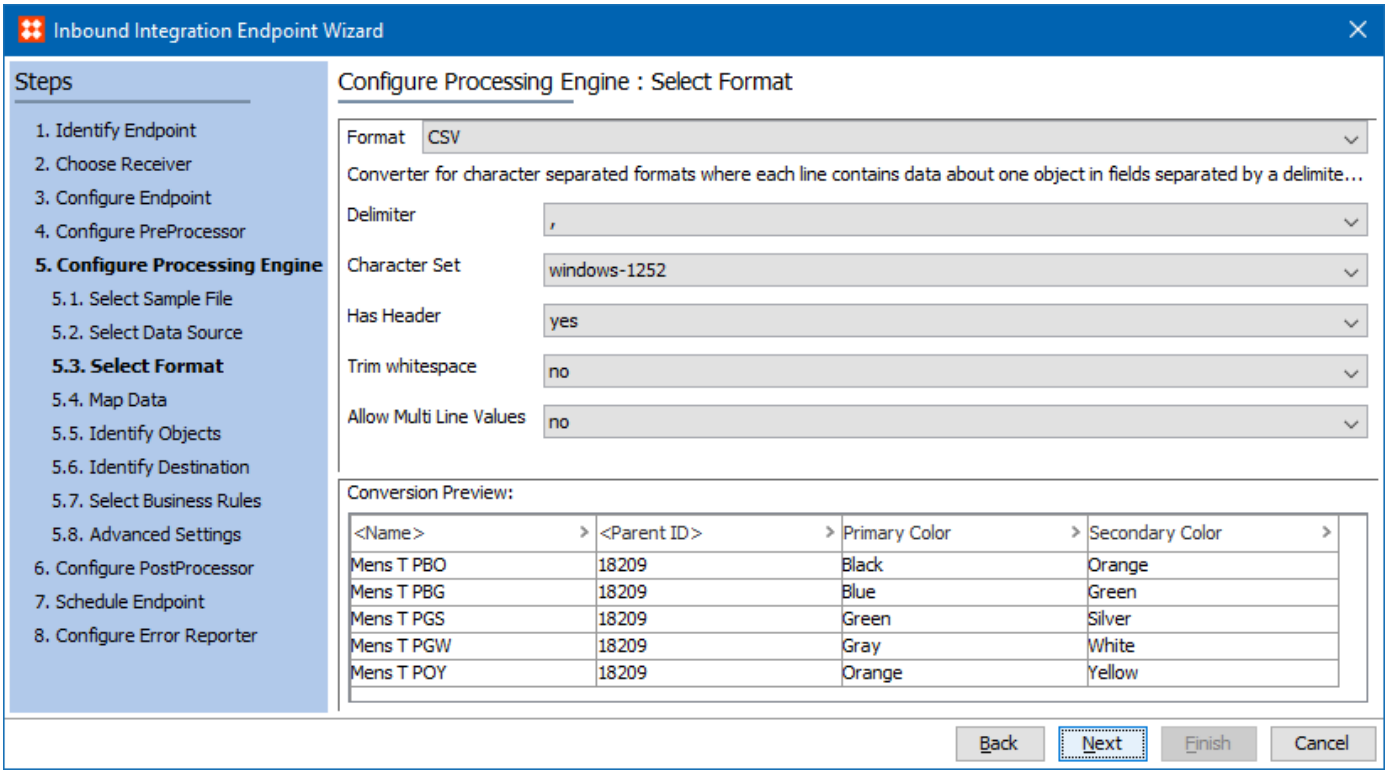
- **Yes** removes leading and trailing spaces in imported values.
- **No** leaves leading and trailing spaces in imported values.

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, see 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.

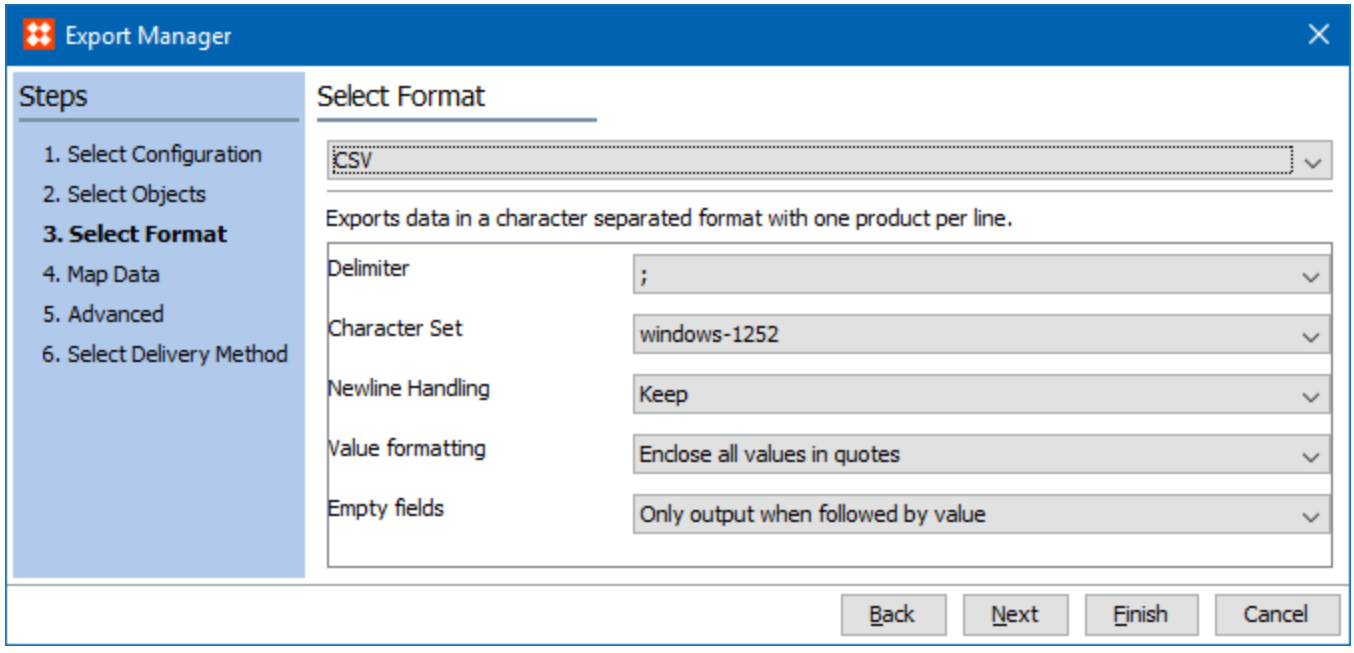
1. **Delimiter** - determines the character used to delimit the exported data. Options are semi-colon (;), comma (,), the tab character, the pipe symbol (|), or colon (:).
   - **Tab** is recommended to avoid splitting the data on printable characters that may occur within the values.
   - Only the delimiters 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.

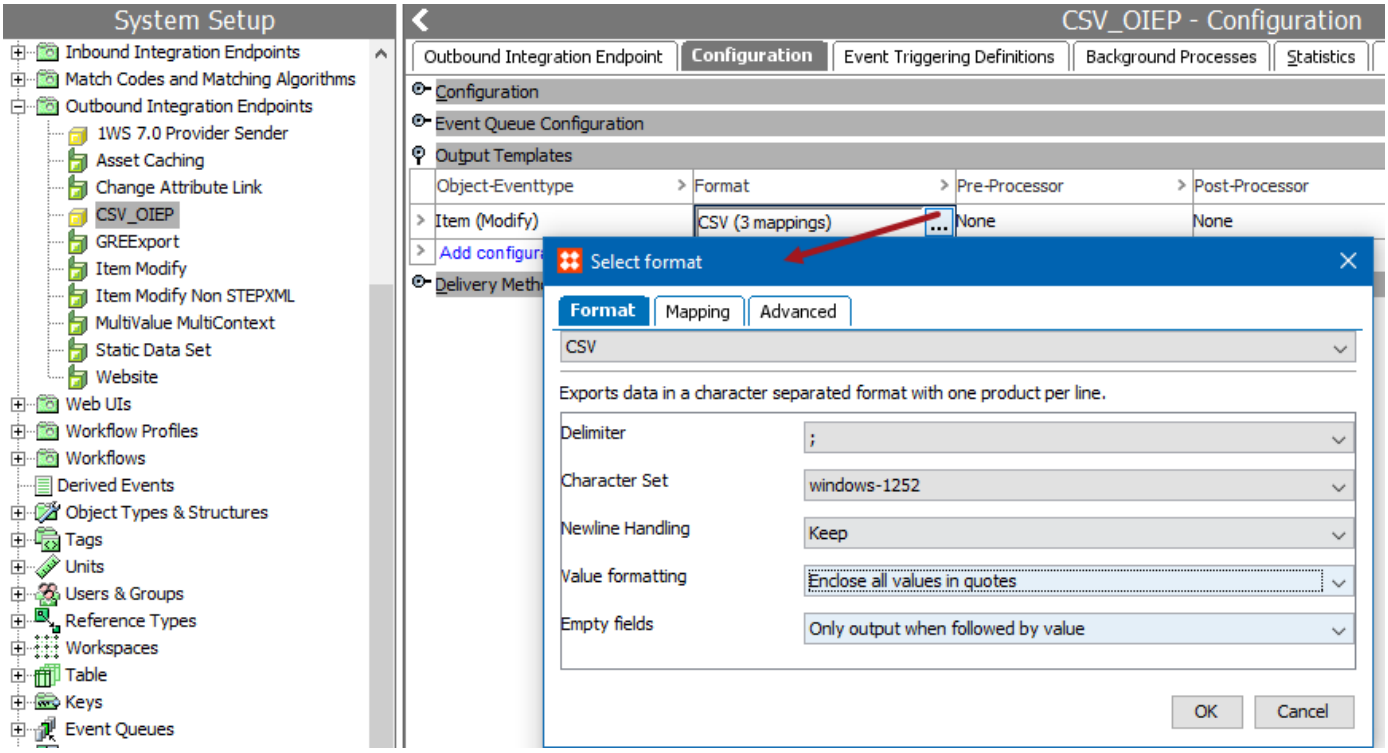
3. **Newline Handling** - determines how new lines (return codes) are handled when they occur in the data being exported.
   - **Convert to Space** (recommended) when new lines exist in the data being exported, to prevent data integrity issues caused by character-separated files having new lines within 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. *Attribute groups cannot be used in conjunction with this option.*

Important: If exporting attribute groups, empty attributes will not be exported. The specific attributes to be exported need to be mapped manually.

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 configur
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
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.

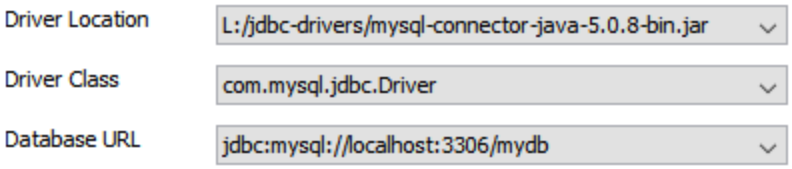
For details on a use case in which JDBC is used to update analytics data, see the **Analytics using JDBC Example** topic.

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. The first configuration entry must use number 1, and each additional entry must be numbered sequentially (JDBCDeliveryPlugin.DriverPath.1, JDBCDeliveryPlugin.DriverPath.2, JDBCDeliveryPlugin.DriverPath.3, etc.).



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.'

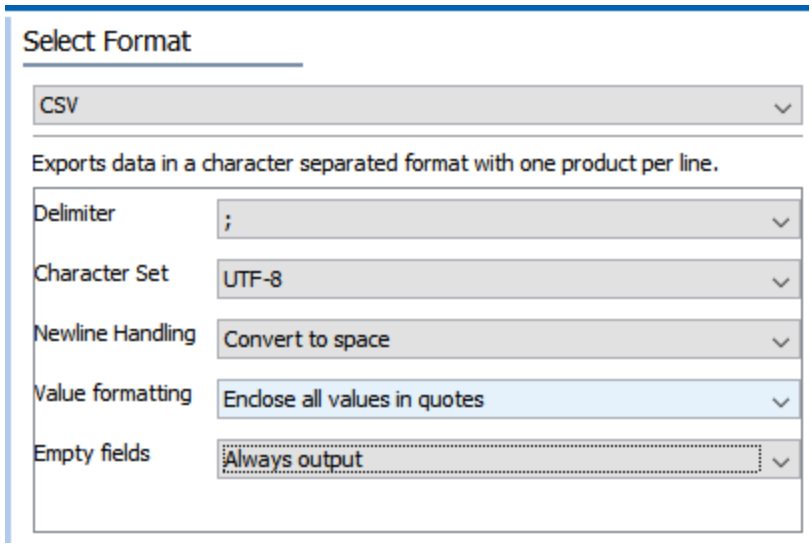
3. Prior to configuration, clicking the **Database URL** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **JDBCDeliveryPlugin.URL.[n]** property.

   ```
   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:

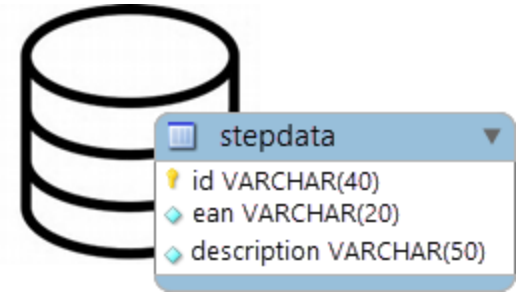


Map Data

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

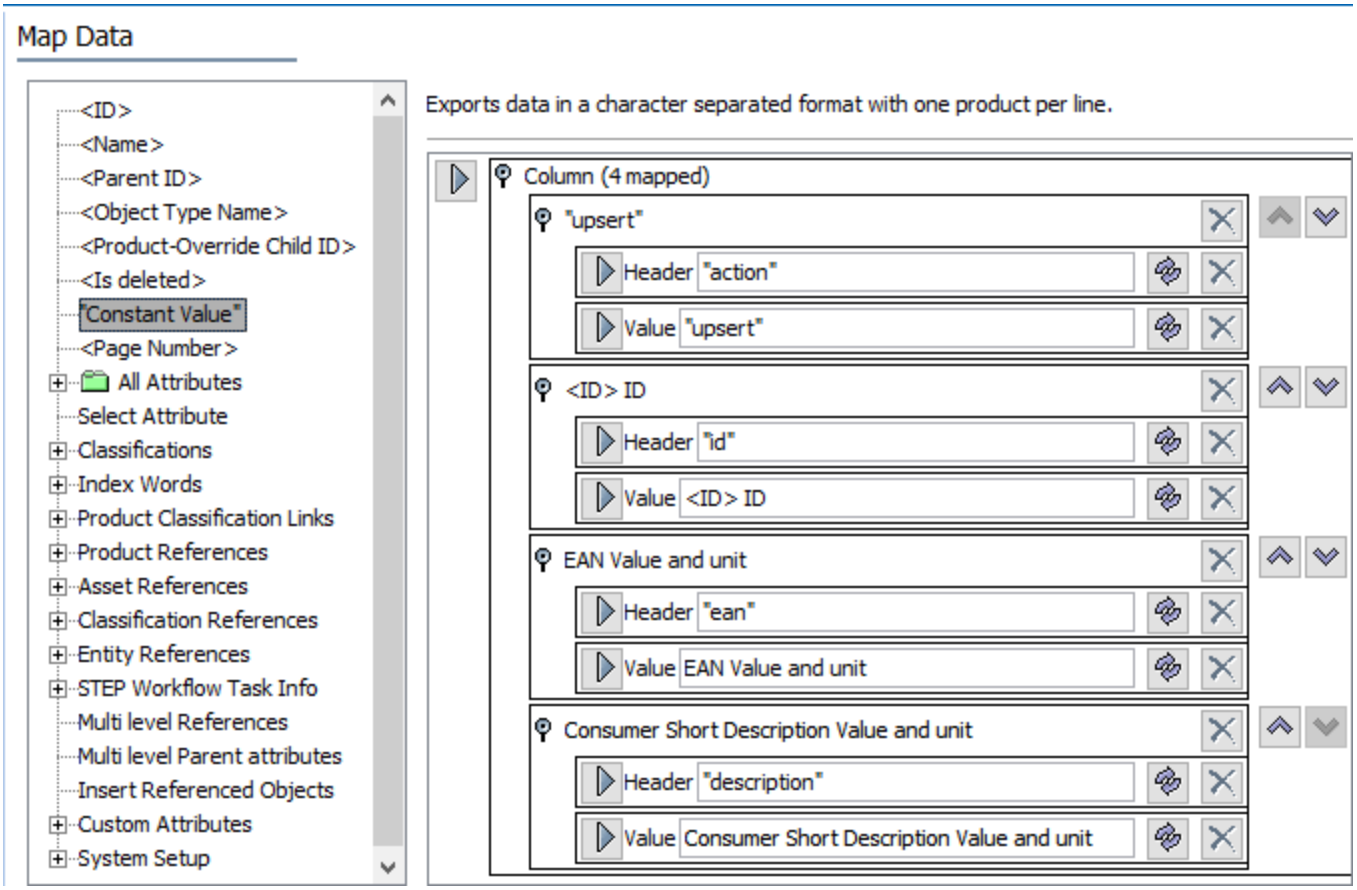
For example, consider the following external database sample table:



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, see 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):



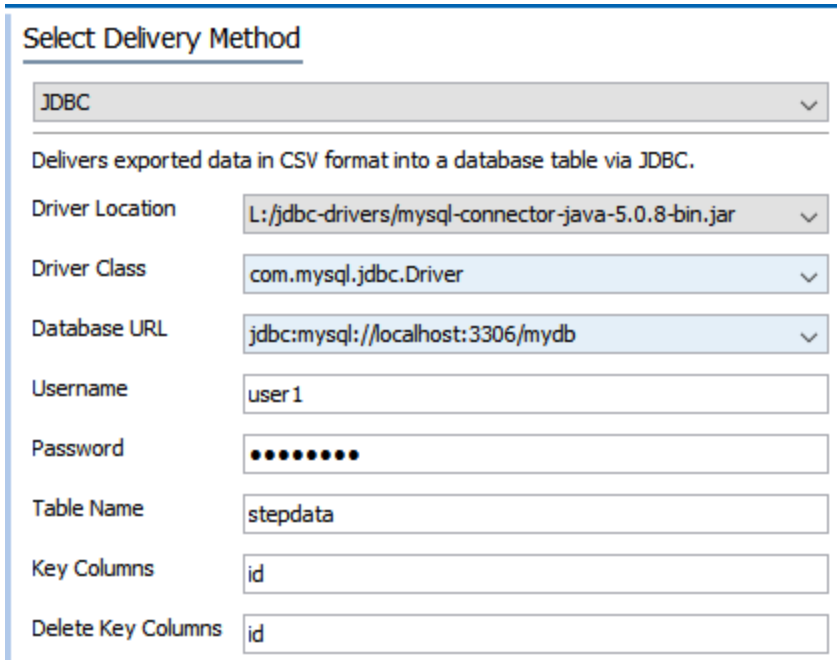
Important: For the external destination database, the JDBC delivery method assumes that the destination data table has a column used as a key, and a number of columns for values (for example, `key value1 value2 [value (n)]`.) This means a destination data table having only a single column (e.g., ID) 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.

Analytics using JDBC Example

One of the prime uses cases for deploying JDBC as a method of delivering data from STEP to a database is to populate dashboards for data analytics tools like Tableau and Qlik. To aid proper setup of a STEP integration with data analytics that makes use of the JDBC method, a use case specific to data analytics integrations is described below. For more information on setting up an analytics integration in the STEP Web UI, see the **Analytics Configuration** topic located in the Web User Interfaces guide.

One common approach to configuring data analytics dashboards is to enable display of historical data. As an example, a user has configured a Tableau dashboard to display regional sales data for a specific product. For this user, seeing sales figures for that same product a year ago, or two years ago, would greatly enhance the utility of the dashboard. The way to enable display of historical data using the JDBC method of delivering data is done in the mapping step of the Export Manager and Outbound Integration Endpoint (OIEP).

To implement this users will need to:

- **Install required drivers** for JDBC
- **Configure properties** in the sharedconfig.properties file
- **Select format** of CSV, using the required settings

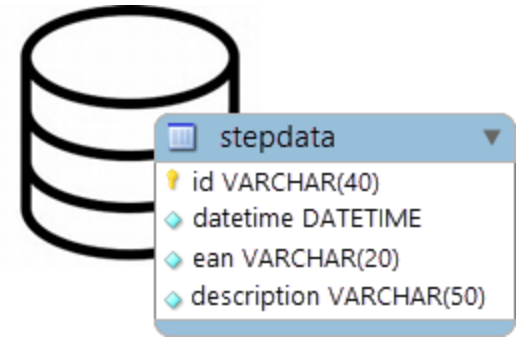
These steps are described in detail in the **Exporting Data via JDBC with CSV Format** topic.

The following steps, which are described in detail in the text below, are specific to this use case:

- **Map Data** to include the required action field and calculated attribute for date / time
- **Select Delivery Method** of JDBC, using the necessary settings

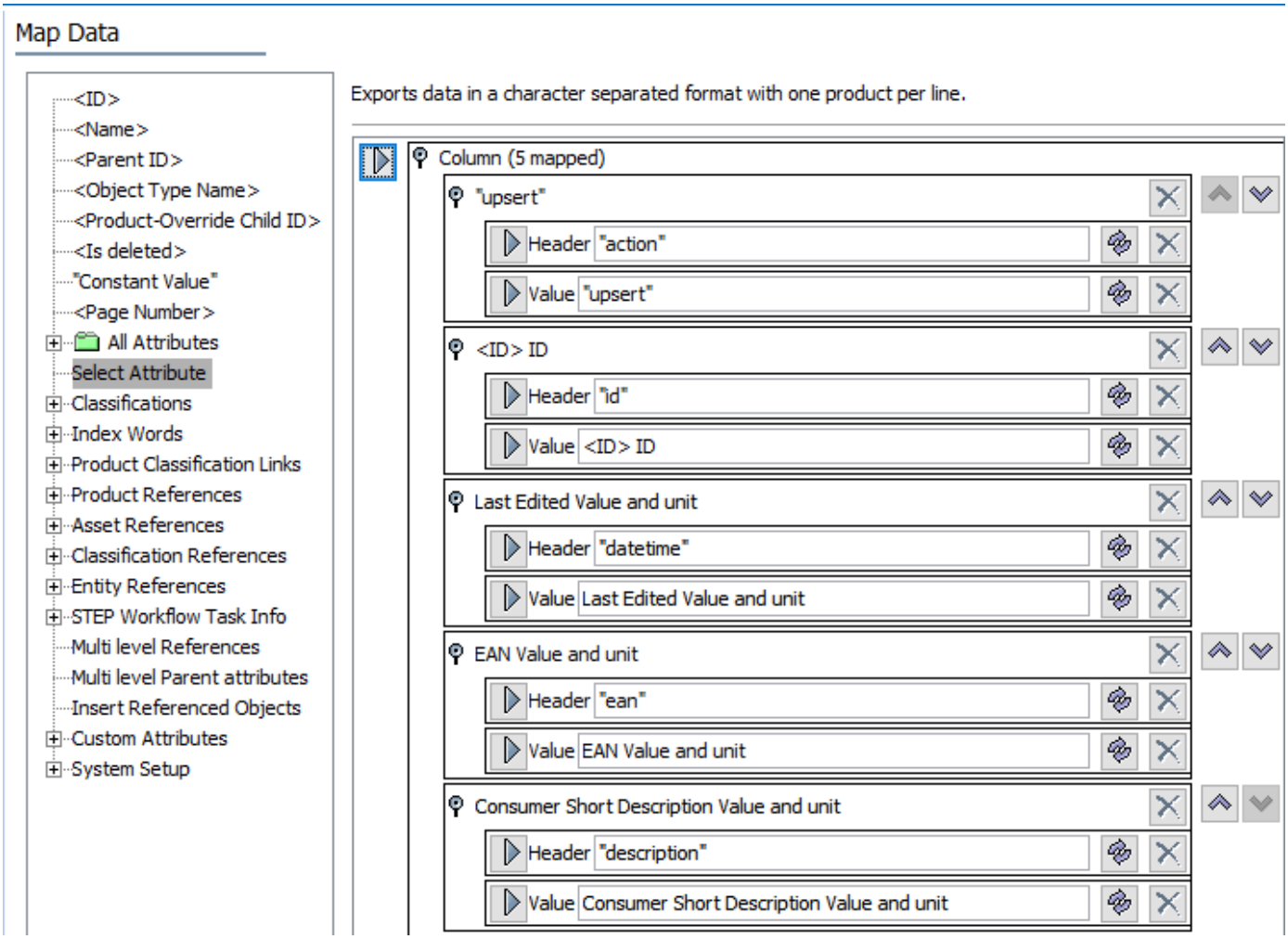
Map Data

With this example, data will be published to a table ('stepdata') that has the following layout:



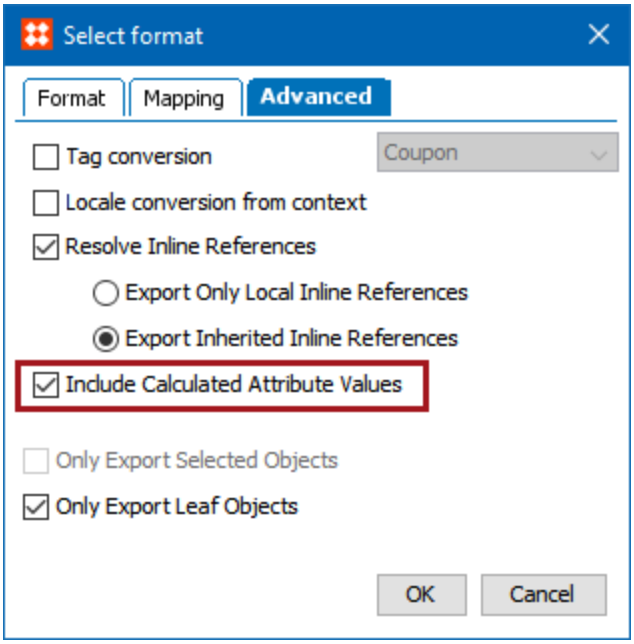
The table has the potential to contain multiple rows representing the same object, and the 'datetime' column will hold information about the latest STEP revision date. This can be achieved with a calculated attribute, mapped in the export configuration as described below. For more information on calculated attributes, see the **Calculated Attributes** topic in the **Getting Started** guide.

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



- 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, see 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.
- Create a calculated attribute using the function 'revisioneditdate()' and make it valid for your exported products. Then map the calculated attribute for export and update the header to match the column in the external table. Details on this function are included in the **Other Functions** topic of the **Calculated Attributes** section of the **System Setup / Super User Guide**.

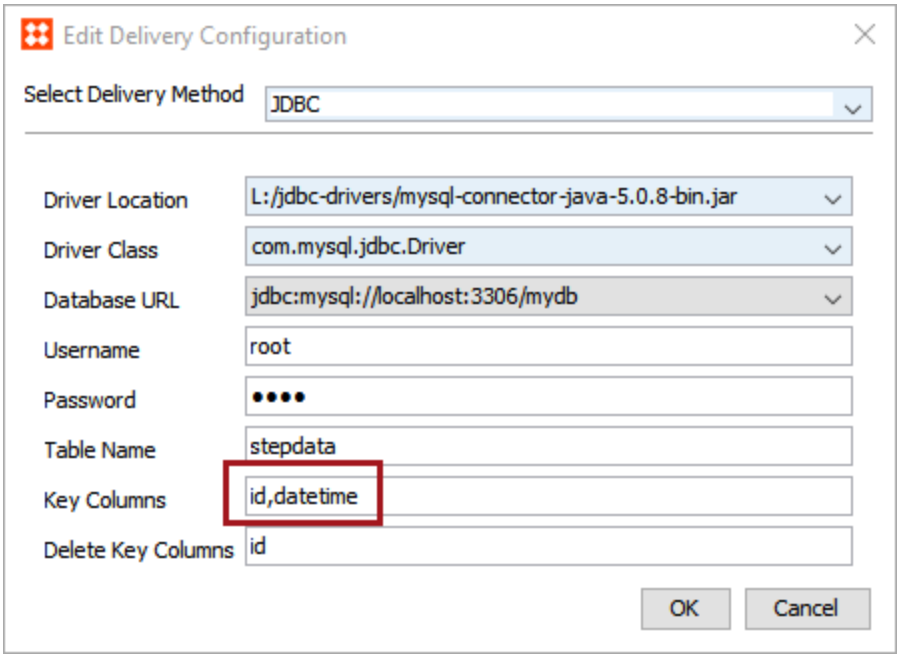
The example above uses the 'datetime' header, which is mapped to the 'Last Edited' calculated attribute. In addition to the other steps required to enact the JDBC method, the 'Include Calculated Attribute Values' parameter on the 'Advanced' step (shown below), must be checked.



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

The appropriate delivery configuration for this scenario can be seen below. Notice that both the 'id' and the 'datetime' column values are used as keys for the upsert action, meaning both values must match to determine whether to insert or update the object. When an exact match for both keys is found, the object is updated; when no match is found, the object is inserted.



The value of this configuration is that if multiple instances of an object with the same ID but different datetime values (representing, for instance, different revisions of the object) are exported, the upsert action will insert into the table all instances of the single object. The effect of this setup is that various revisions of the same STEP data will be published to the table, giving users the ability to view historical data for objects in a data analytics dashboard. Further, if the Web UI has been configured to display data analytics dashboards, then Web UI users can view historical data on STEP objects.

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 - see **Creating a Data Export**
- OIEP - see **Creating an Outbound Integration Endpoint**

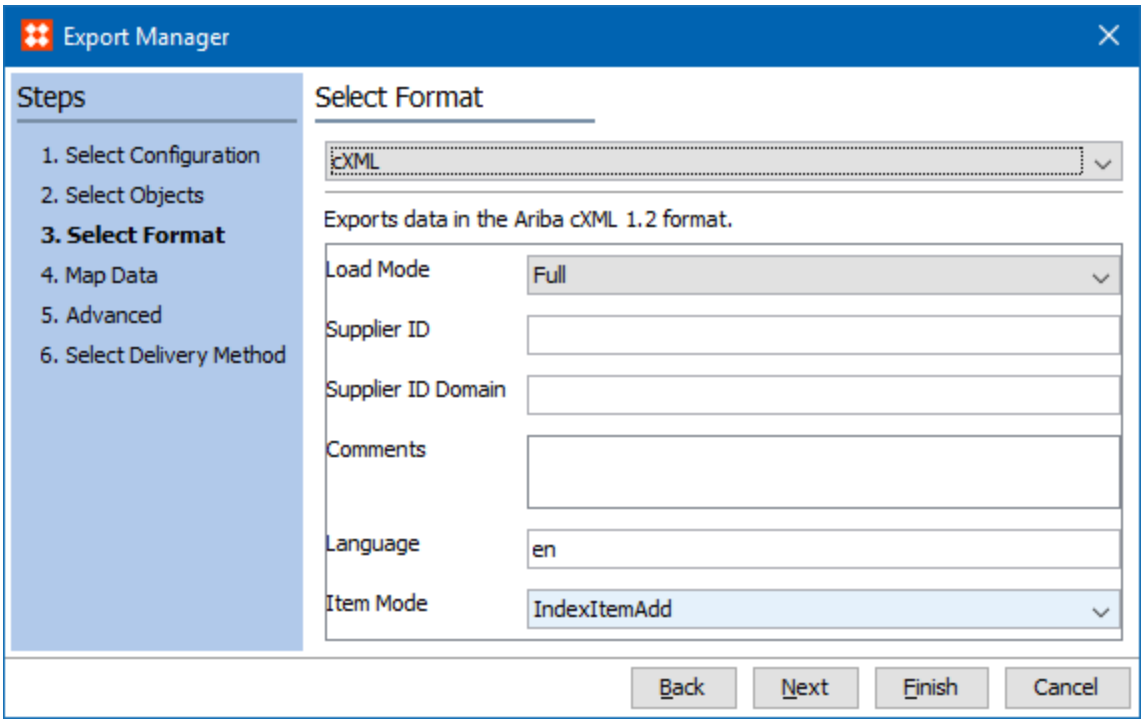
Mapping

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

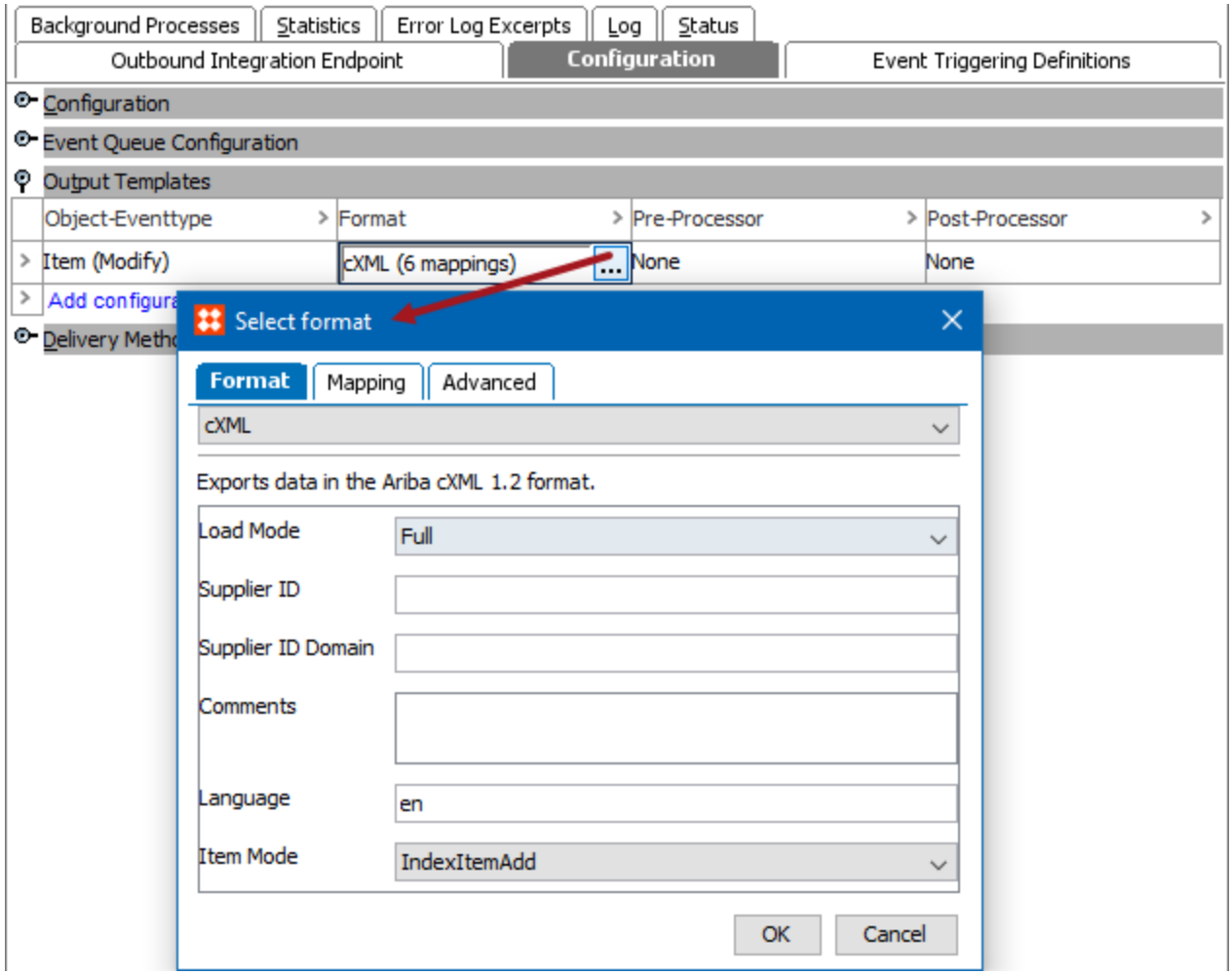
Outbound Data

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

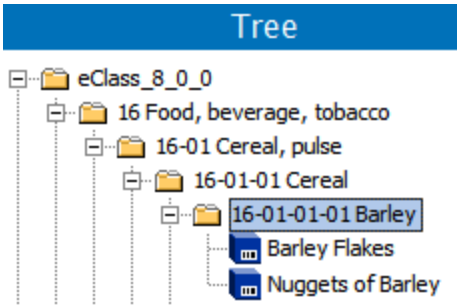
eCl@ss Format

eCl@ss® 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 eCl@ss, search the web.

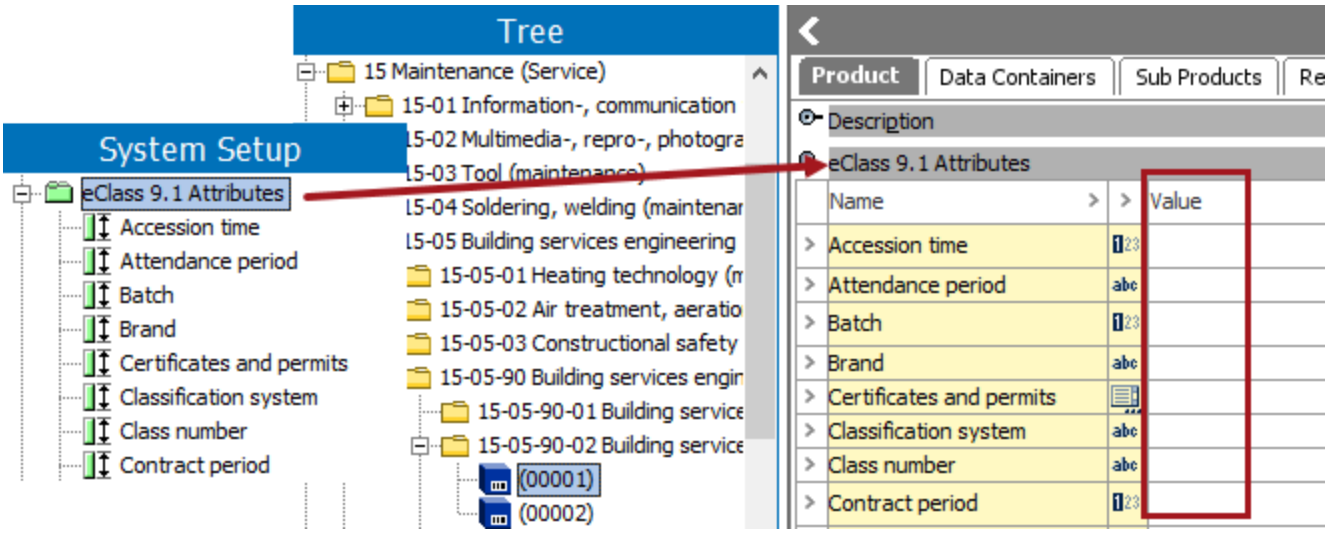
eCl@ss is an add-on component. Contact your Stibo Systems representative to activate 'eclassimporter' on your STEP system.

With eCl@ss you can import the Basic CSV and XML formats for any language using versions 7.0, 7.1, 8.0, 8.1, 9.0, and 9.1 and 10.0.1. For more information on the eCl@ss license, contact your Stibo Systems account manager.

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



When a product is linked to an eCl@ss classification, via a product-to-classification link type with 'Inheritance of Specification Attributes / Data Container Types' set to 'Yes,' the eCl@ss 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, see the **Inheritance Example for a Product to Classification Link Type** topic in the **System Setup / Super User Guide** documentation.



Format Availability

eCl@ss import format is available for selection only in the eCl@ss Classification Import wizard. For details, see **eCl@ss Classification Import**.

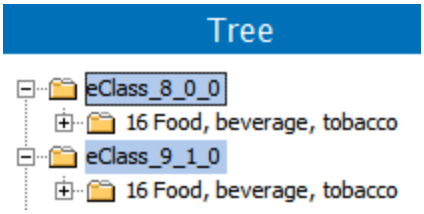
Mapping

When using eCl@ss attributes that have been created by an eCl@ss CSV import file, there is no need to recreate attributes already in STEP to gain the benefits of using standard eCl@ss attributes. After importing an eCl@ss classification hierarchy from CSV, you can map each eCl@ss attribute to a STEP attribute to merge the attributes and delete the eCl@ss attribute. Mapping functionality is not available with eCl@ss XML imports. For details, see **Map eCl@ss Attributes to STEP Attributes**.

Important: Typically, industry standard attributes should not be altered, so limiting access to edit eCl@ss attributes via user privileges is recommended.

Multiple Versions

The eCl@ss 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 eCl@ss within the same system.

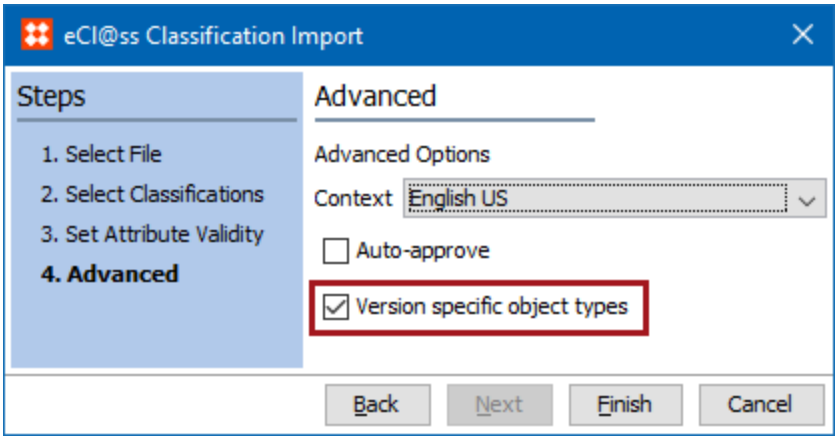


Update objects to a newer classification version

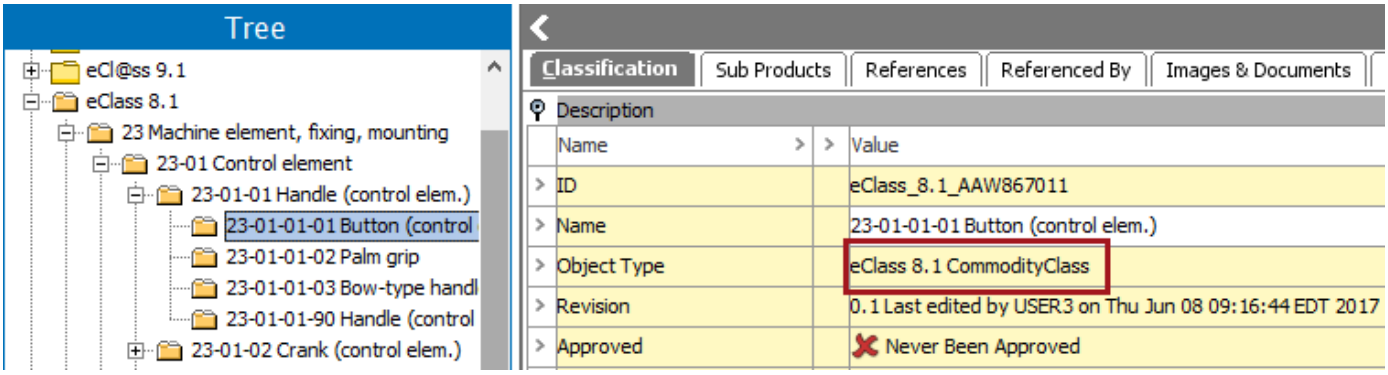
If you want to update existing objects to a newer eCl@ss classification version, you can re-classify them using the Bulk Update operation 'Industry Classification Synchronization.' For more information, see the **References and Links: Industry Classification Synchronization Operation** topic of the **Bulk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



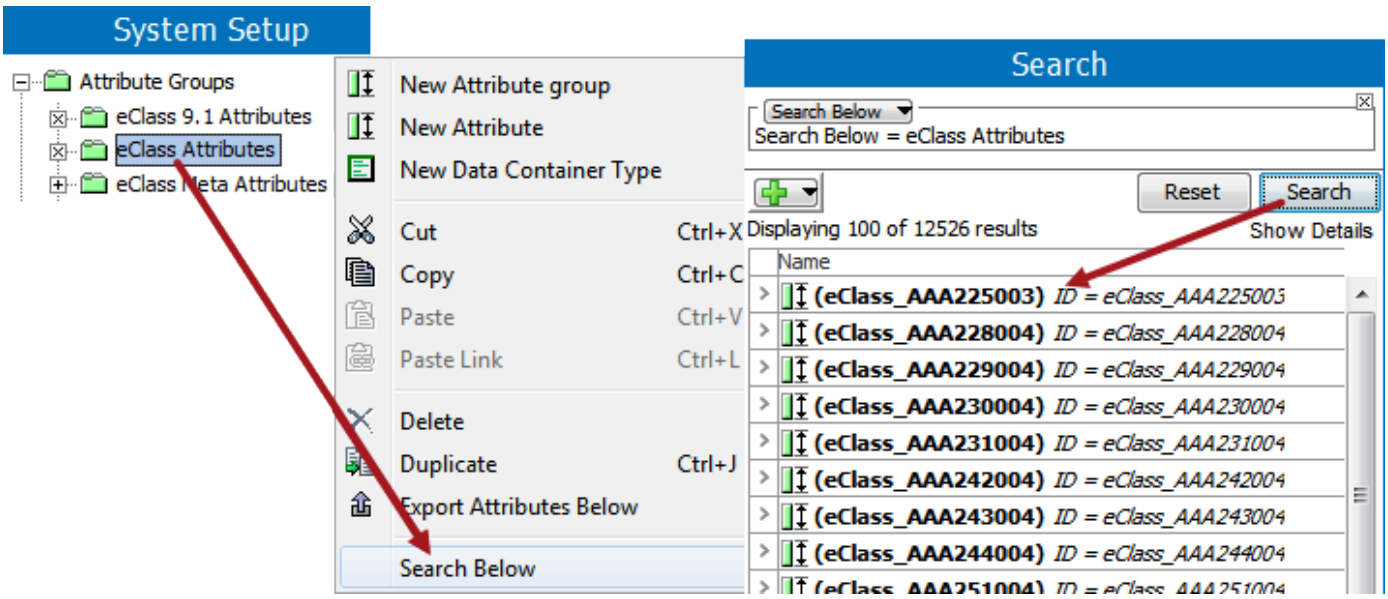
Selecting this option during import will automatically create a new version-specific object type structure for eCl@ss classifications if one does not already exist for that version.



For more information, see the **eCl@ss Classification Importer** section of the documentation.

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 eCl@ss 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.

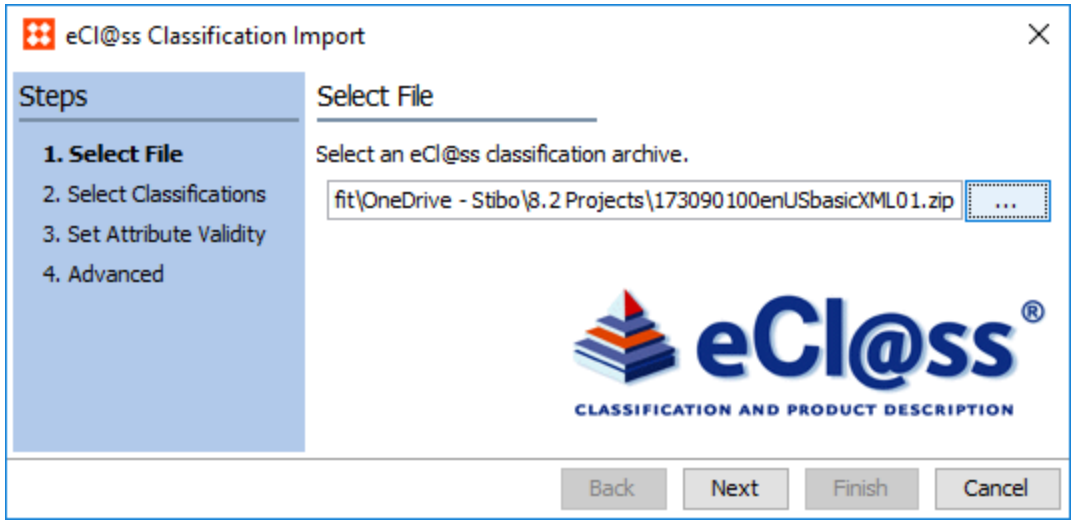
Important: Units and LOVs can only be imported via eCl@ss XML files, *not* CSV.

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 eCl@ss 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, see the **Attribute Groups** topic in the **System Setup / Super User Guide** documentation.
- **eClass Meta Attributes** - description attributes for metadata on attributes, units, LOVs, and classifications.

eCl@ss Classification Import

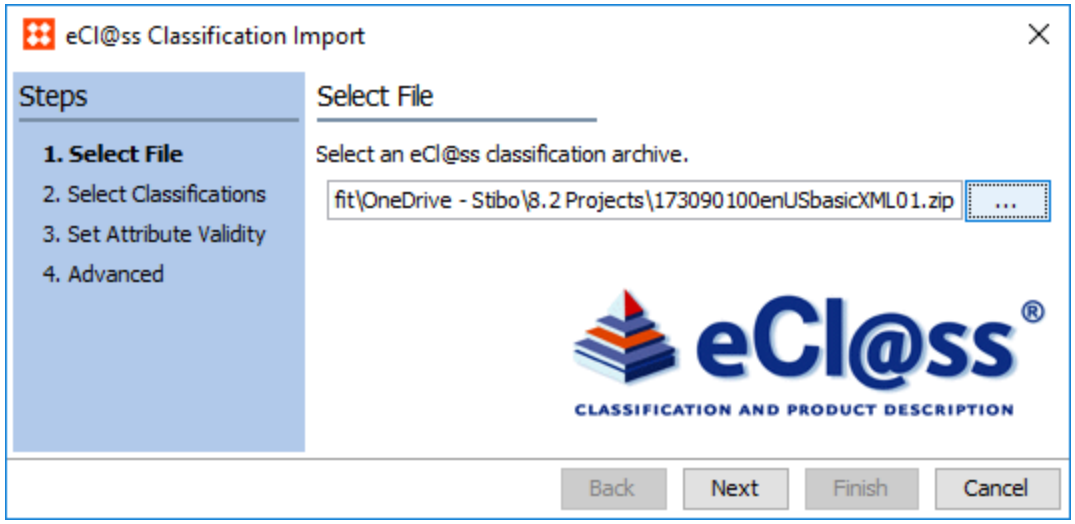


eCl@ss Classification Import

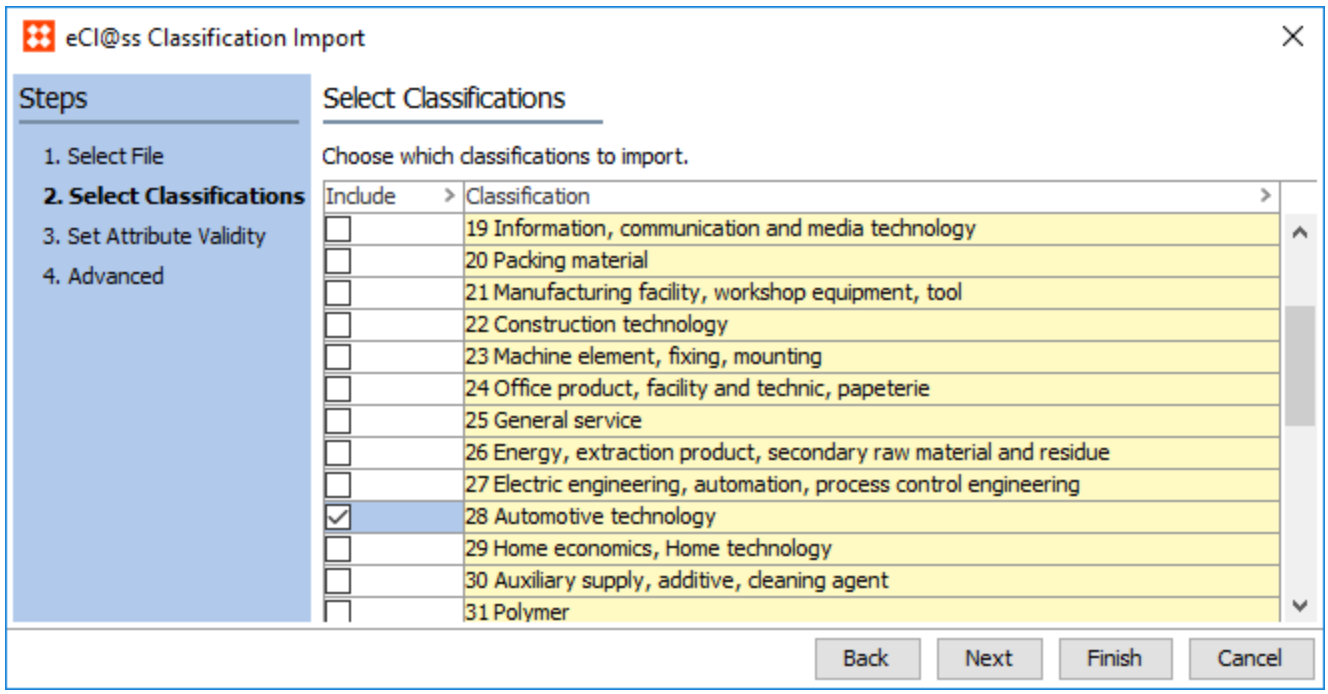
Use the following steps to start the eCl@ss import.

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

Note: The language of the file is indicated within the file name. For example, in the image below, '...enUS...' indicates that the file contains English / US.

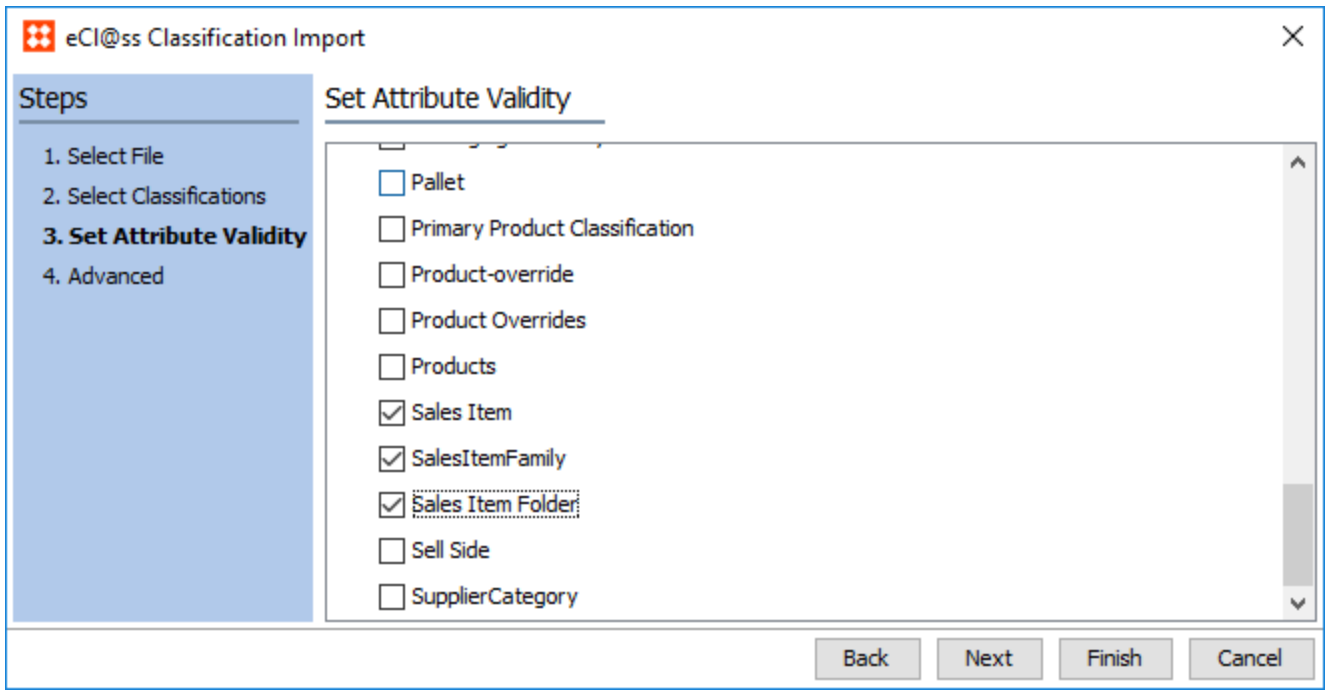


3. For **Select Classifications**, select all classifications to import, and click **Next**.



4. For **Set Attribute Validity**, select the valid product object types for the imported eCl@ss attributes. This step is optional, and only available for eCl@ss 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.

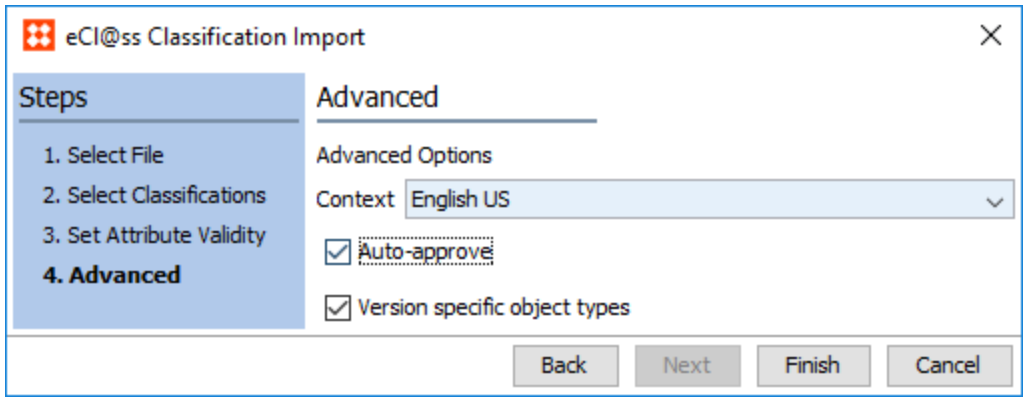


Important: If attribute validity is not set during the import it will have to be set manually later, otherwise you will not be able to maintain these attributes on linked eCl@ss products.

5. For **Advanced**, select the context for the language of the eCl@ss file you are importing. Check the 'Auto-approve' checkbox if imported classifications should be approved automatically.

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

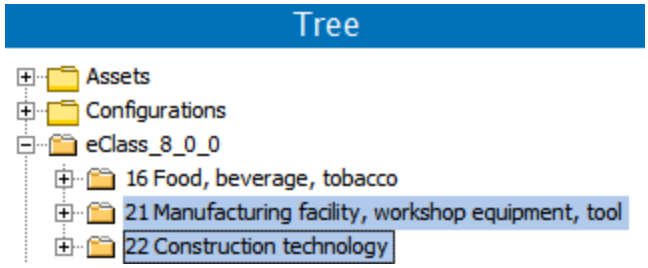
   Click **Finish** to begin the import.



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 eCl@ss languages for different contexts:

   - **Configure the language displayed in the Tree for the eCl@ss 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 eCl@ss data:** Run the import wizard again for each eCl@ss language required.

Map eCl@ss Attributes to STEP Attributes

eCl@ss classifications include eCl@ss specification attributes. Links are inherited down the product hierarchy, which allows the attributes to be displayed on the product within an eCl@ss [version] Attributes flipper.

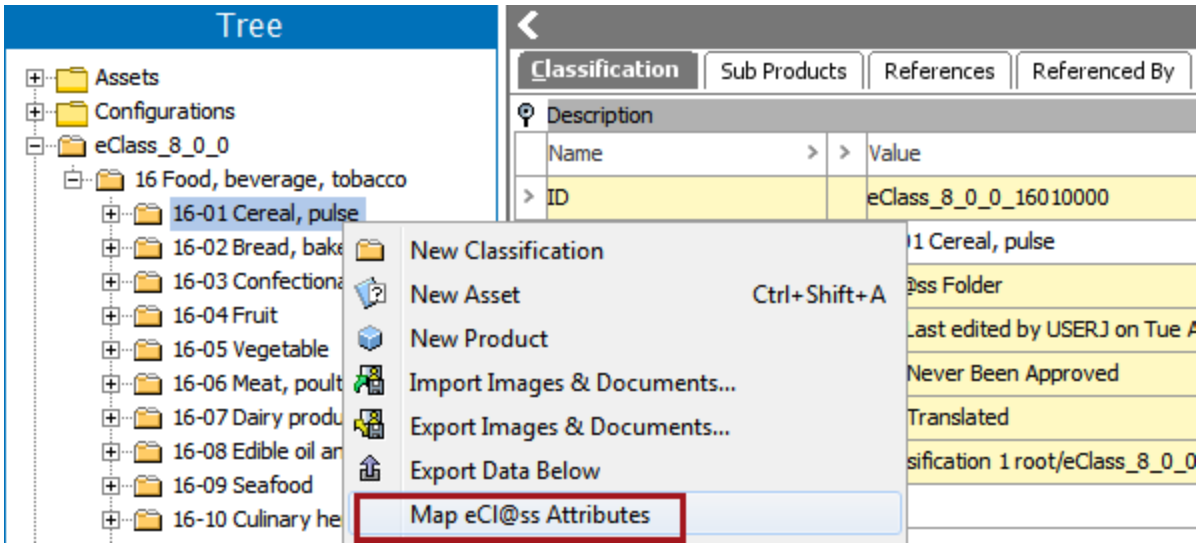
Important: Typically, industry standard attributes should not be altered, so limiting access to eCl@ss attributes via user privileges is recommended.

There is no need to recreate attributes you already have in STEP to gain the benefits of using standard eCl@ss attributes. After importing an eCl@ss classification hierarchy, you can map each eCl@ss attribute to a STEP attribute to merge the attributes and delete the eCl@ss attribute.

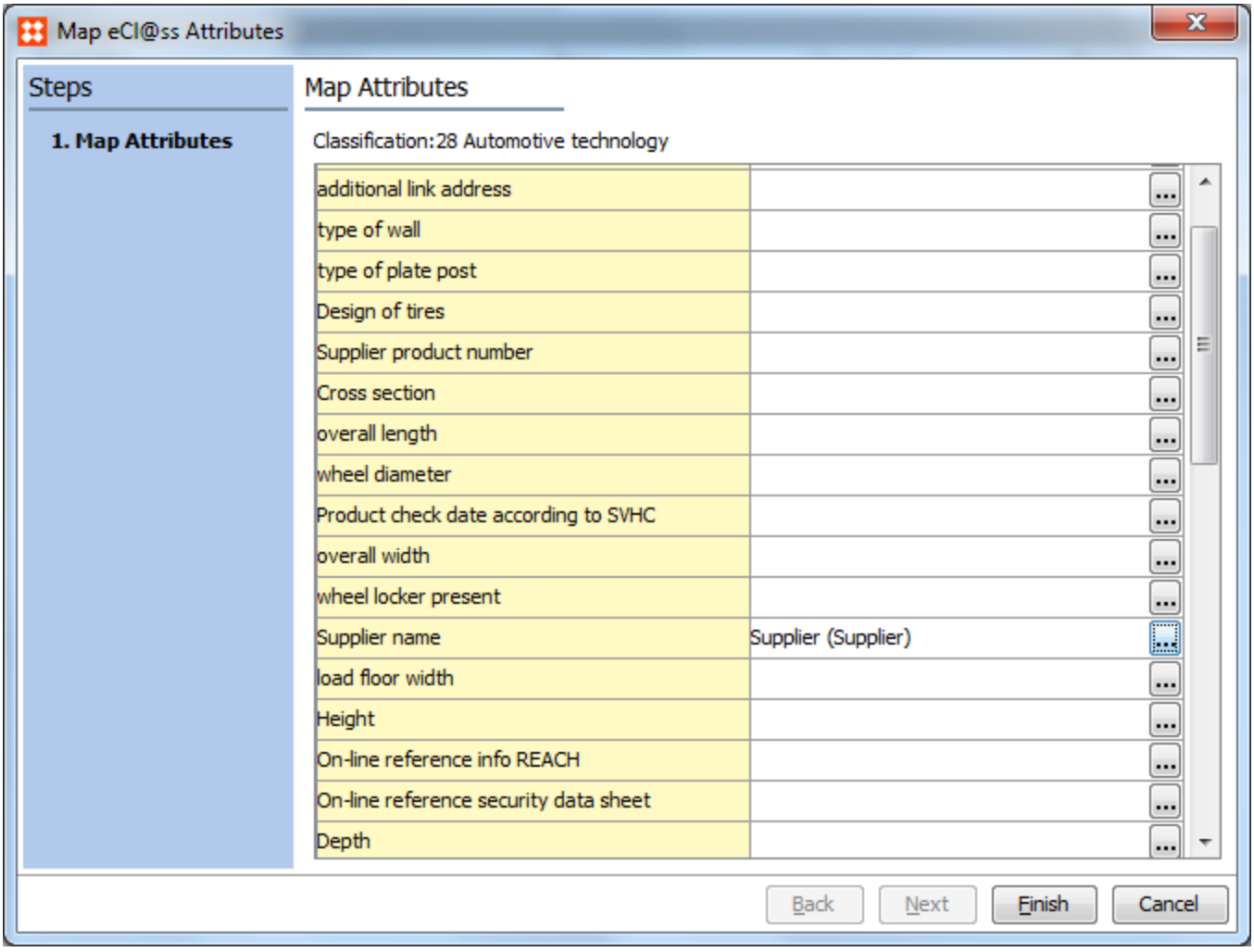
Note: The 'Map eCl@ss Attributes' option is not available for eCl@ss attributes imported via XML.

Mapping eCl@ss Attributes to STEP Attributes

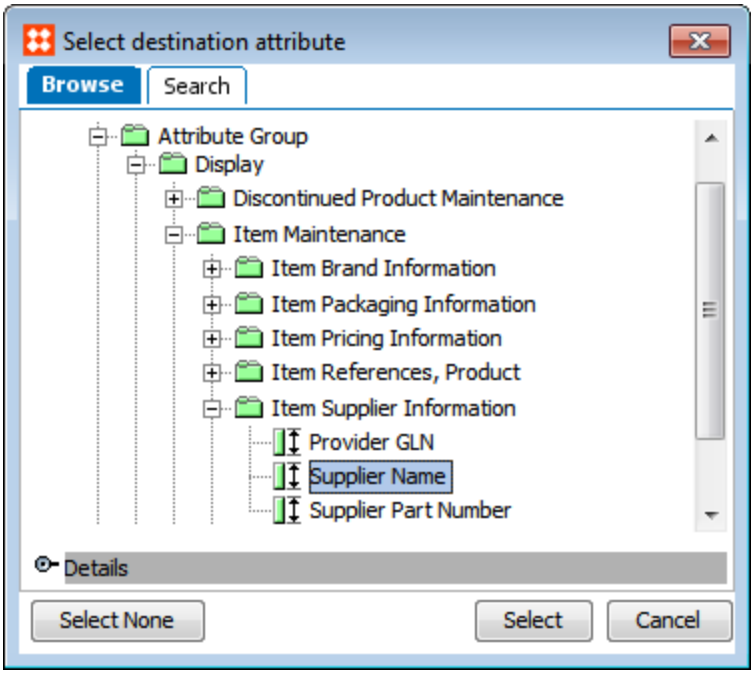
1. In the Tree, right-click the eCl@ss classification to map attributes *from* (the Source Attribute), and select the **Map eCl@ss Attributes** option.



2. In the wizard, click the ellipsis button (...) for the relevant eCl@ss attribute (the Source Attribute). The Source Attributes listed are those that exist for the selected classification.



3. Search or browse for the STEP attribute to map *to* (the Destination Attribute), select it and click the **Select** button. The eCl@ss attribute will be merged into this STEP attribute.



4. Click **Finish** on the Map eCl@ss Attributes wizard. Once merged, the eCl@ss attribute is deleted from both the available attributes in the wizard Source Attribute column and from the System Setup > eClass Attributes group folder.

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 - see **Creating an Inbound Integration Endpoint**
- Import Manager - see **Creating a Data Import**

Mapping

This format does not require a data map.

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, see **Creating a Data Import** or see **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.

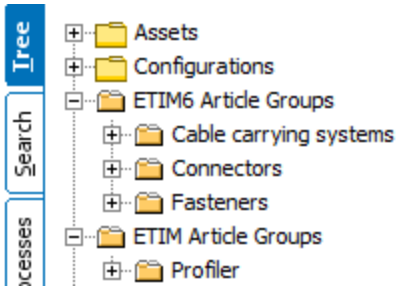
Multiple versions can exist in STEP simultaneously using multiple individual hierarchy structures.

- ETIM 5 uses CSV and each file contains a single language. For more information about the ETIM 5 formats, see **ETIM and ETIM v2Format**.
- ETIM 6 and ETIM 7 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, see **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.

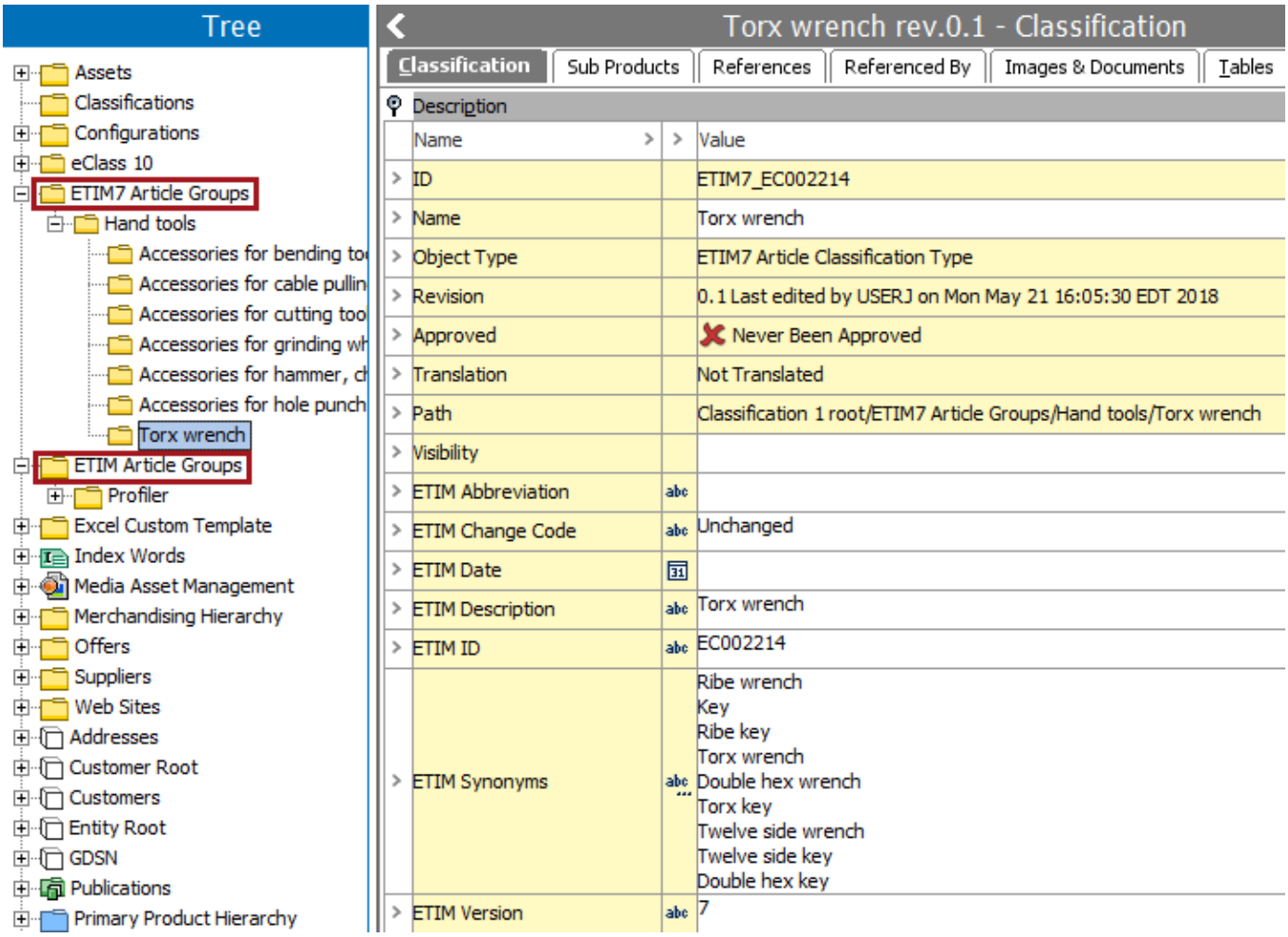


Each imported ETIM article group classification will include the following metadata attributes on the Classification tab within the Description flipper (as shown in the image below).

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.

Attribute Name	Description
ETIM Description	Description from ETIM, matches Name attribute.
ETIM ID	ID from ETIM, matches ID attribute.
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, see **ETIM and ETIM v2 Format**.

ETIM IXF

- ETIM IXF, versions 6 and 7- 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.

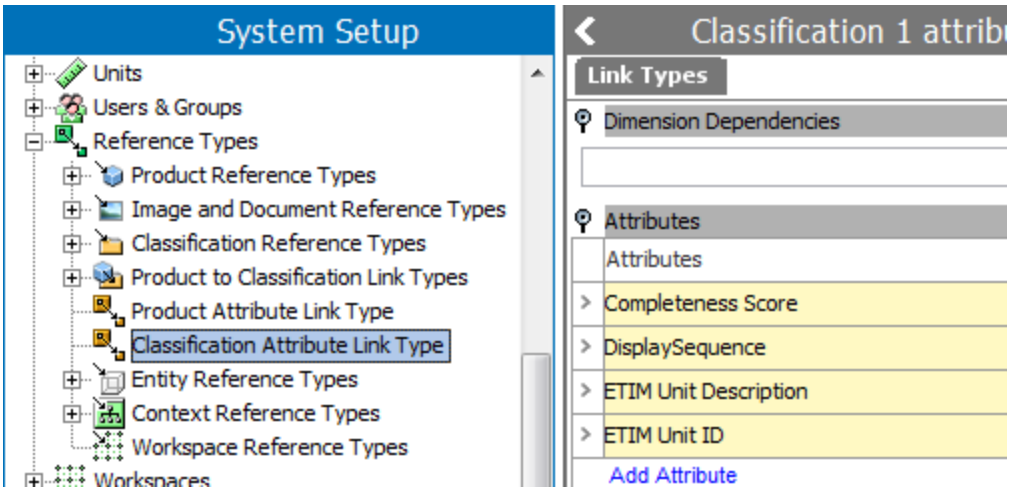
For more information about the ETIM IXF format, see **ETIM IXF Format**.

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.

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 STEP 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, see the **ETIM Format** documentation.

Format Availability

ETIM format is available for selection in:

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

Mapping

This format does not require a data map.

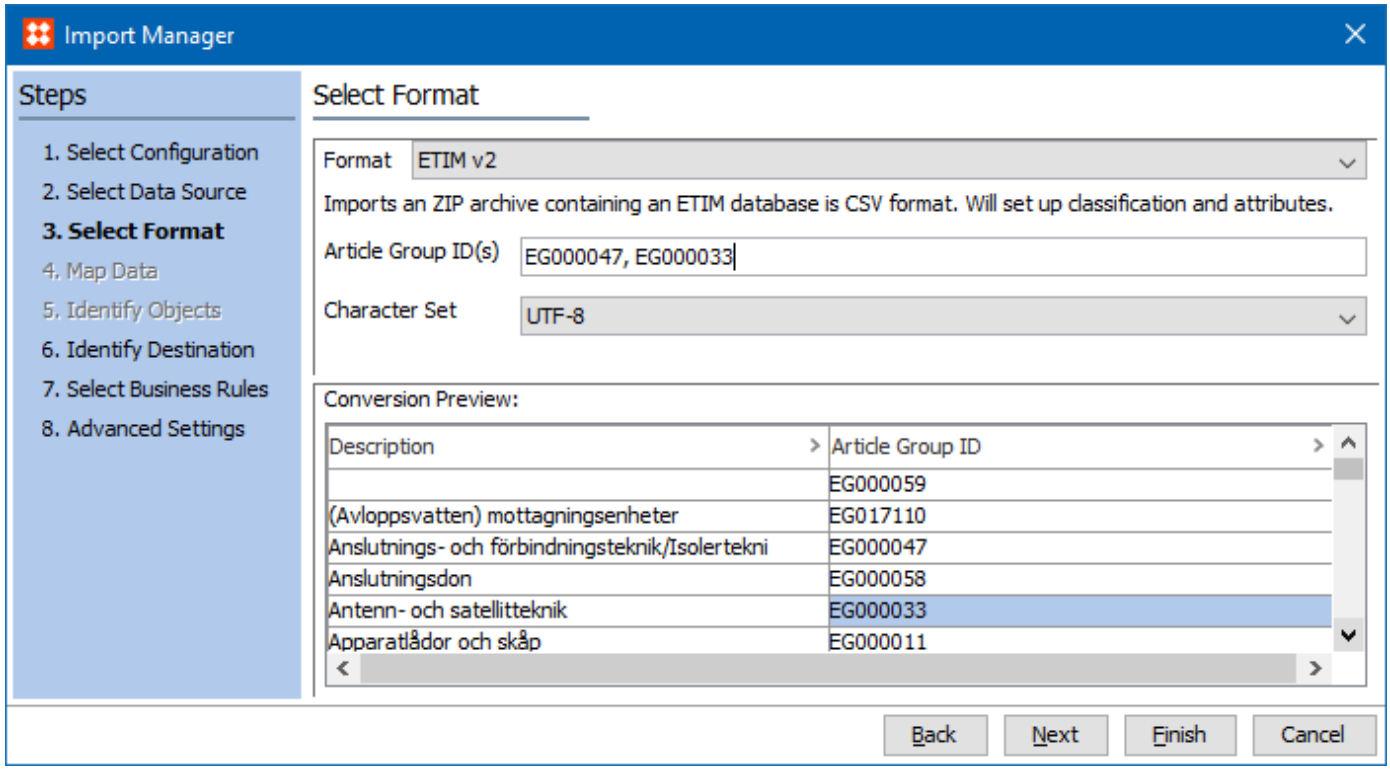
Inbound Data

Inbound Parameters

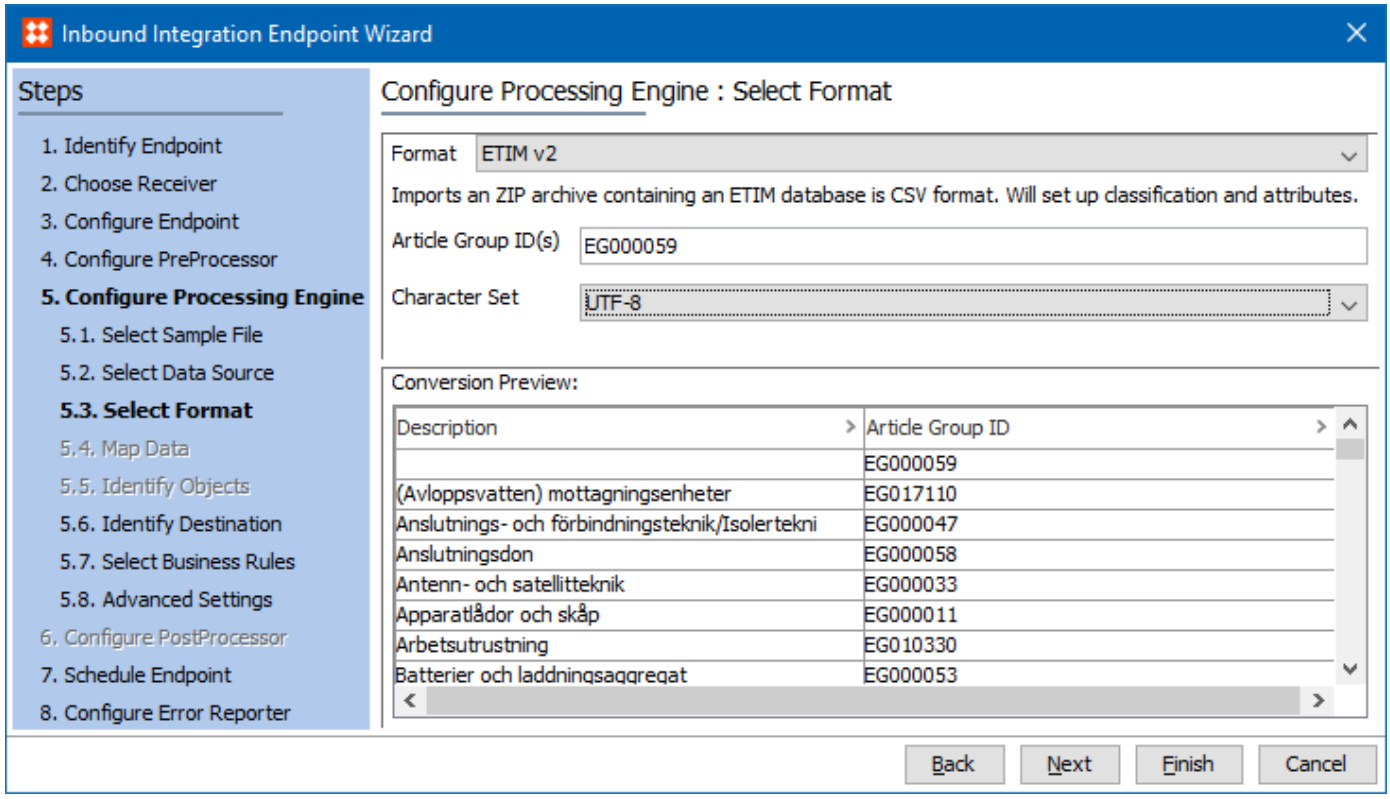
- **Article Group ID(s)** allows you to list one or more classifications as follows:
  - A single Article Group ID to import one classification.
  - A comma-separated list of Article Group IDs to import several classifications.
  - Blank to import the full ETIM classification system.
- **Character Set** 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 see 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. Configure Error Reporter
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 and 7 and STEP also supports the dynamic, or pre-release versions in between major releases. For more information about ETIM functionality in STEP, see **ETIM Format**.

Format Availability

ETIM IXF format is available for selection in:

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

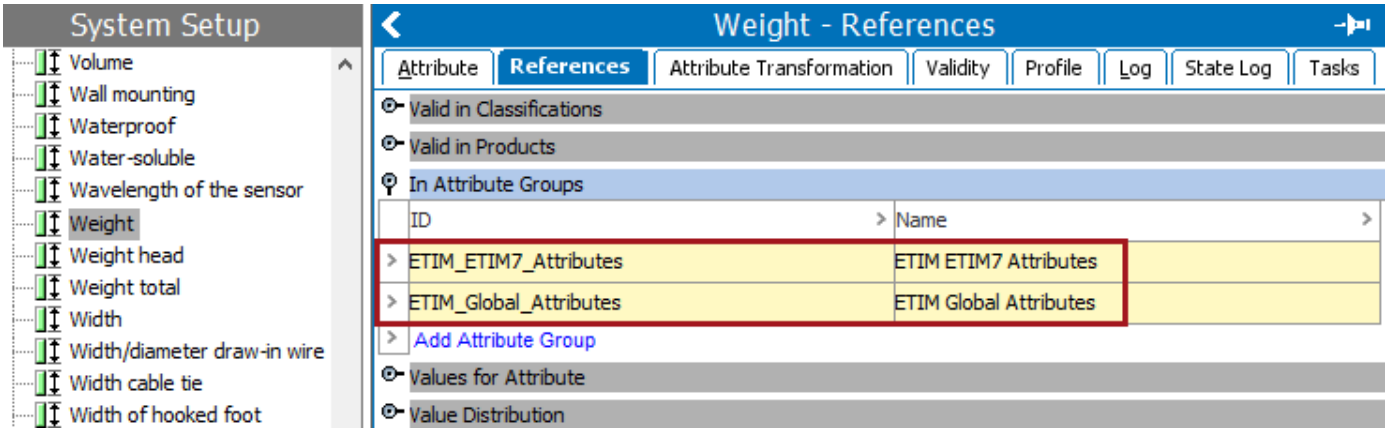
Mapping

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.

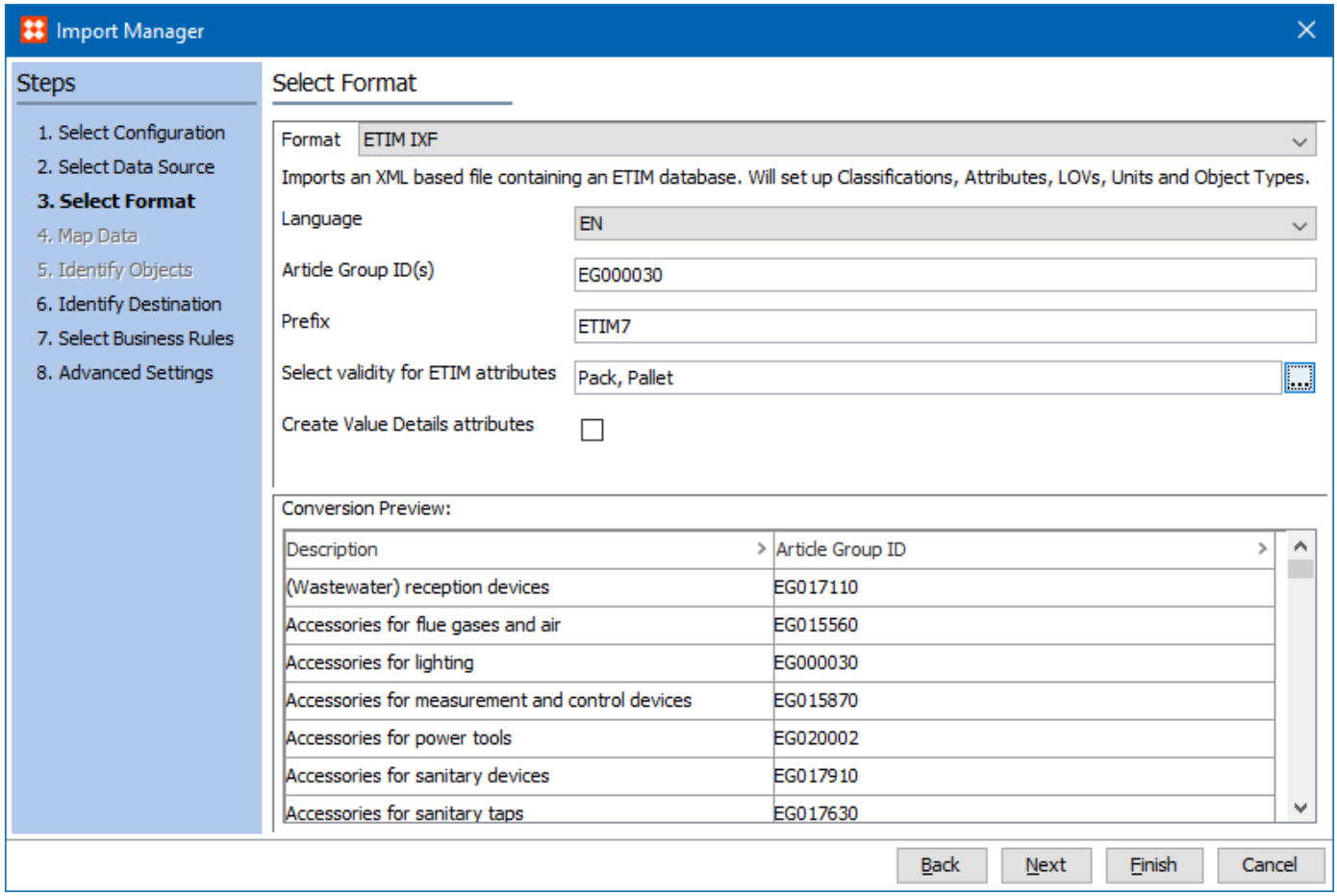
- **Article Group ID(s)** allows the user to list one or more classifications as follows:
  - A single Article Group ID to import one classification.
  - A comma-separated list of Article Group IDs to import several classifications.
  - Blank to import the full ETIM classification system.
- **Prefix** 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. 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 see 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.

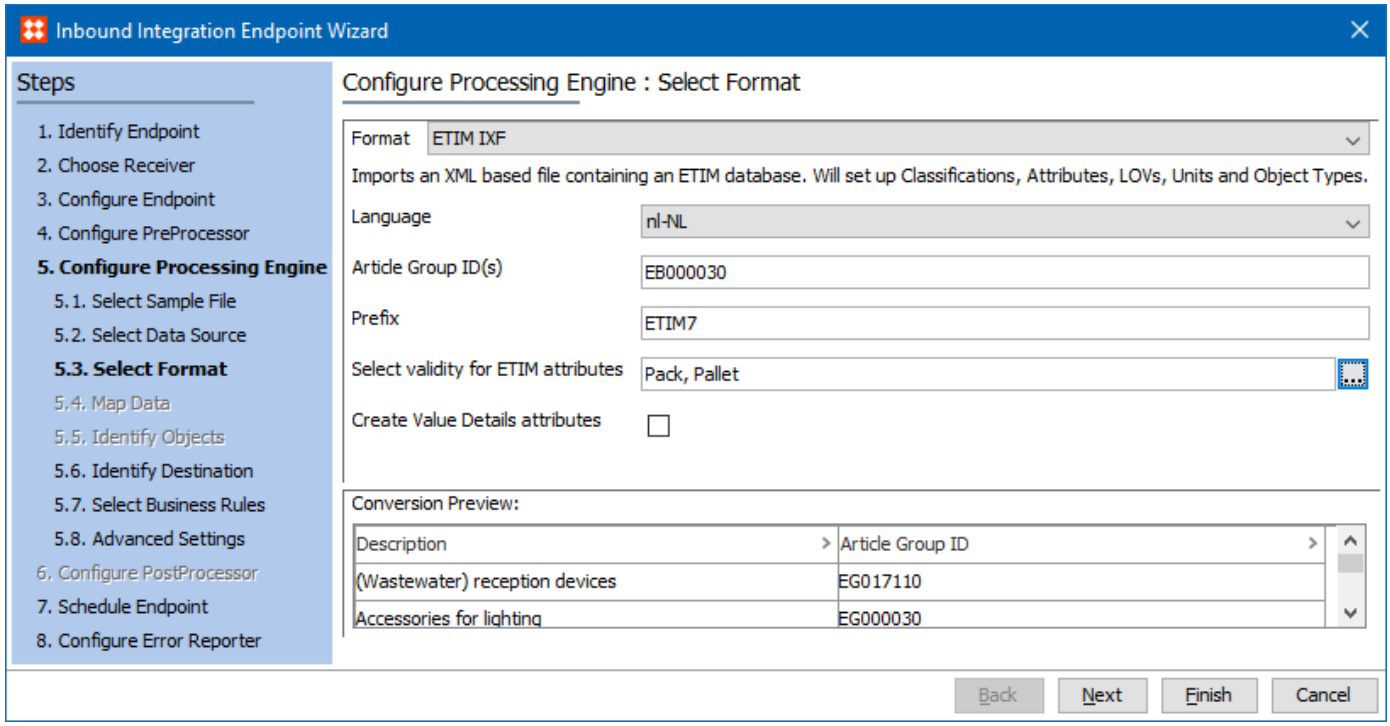
- **Set validity for ETIM attributes** allows the user to select the object types that are valid for the ETIM attributes. ellipses Click the ellipsis button (...) to display a list of object types that can be checked for validity in both the Import Manager and IIEP.
- **Create Value Details attributes**, 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.
- **Conversion Preview** displays article groups that were read from the ETIM import file in the selected language.

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

Import Manager



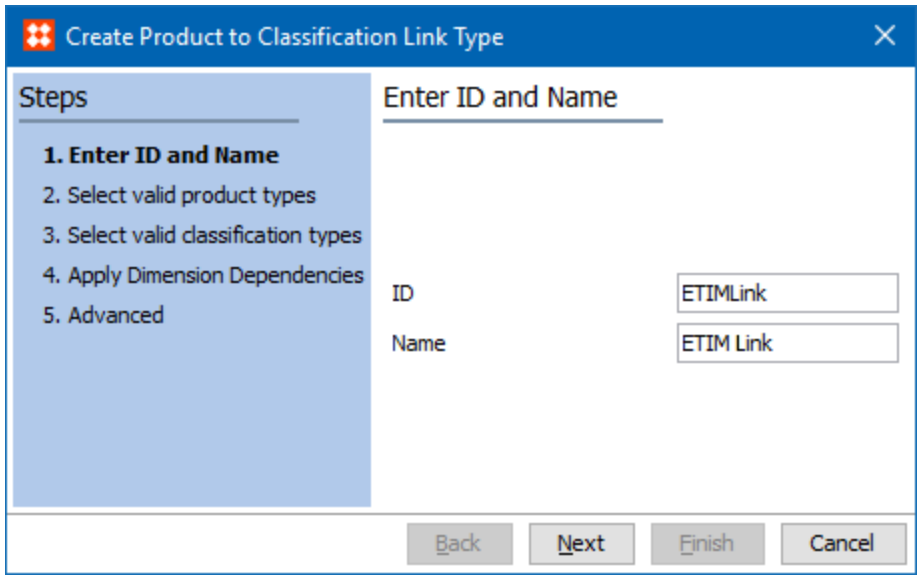
IIEP


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. Configure Error Reporter
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
Conversion Preview:
Description
Article Group ID
(Wastewater) reception devices
EG017110
Accessories for lighting
EG000030
Back
Next
Finish
Cancel

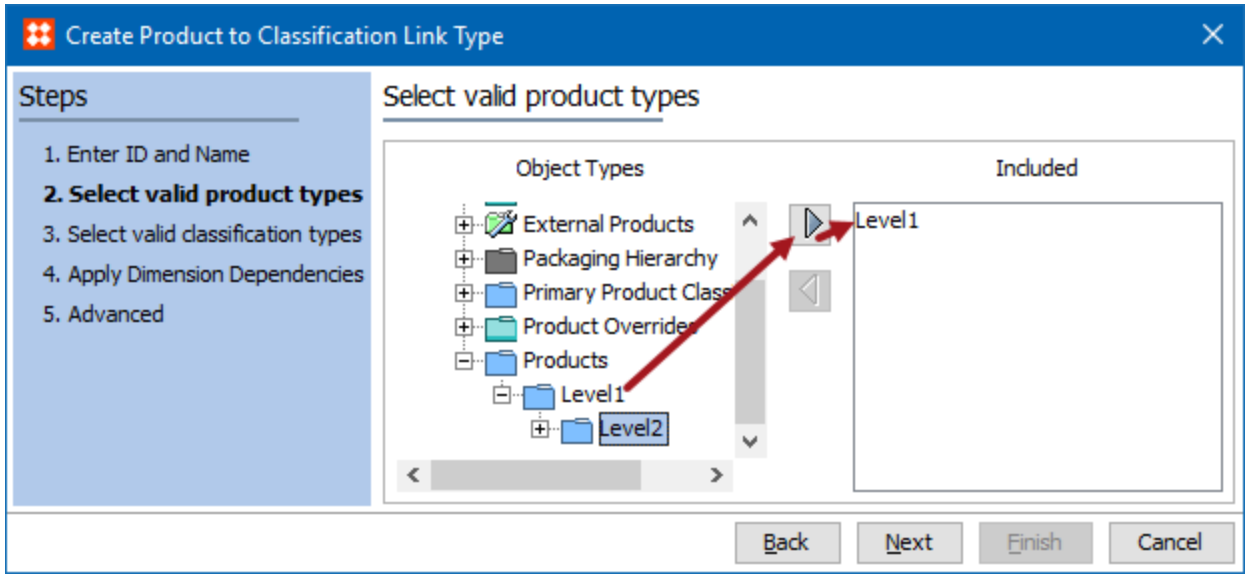
Create Product to ETIM Classification Link

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

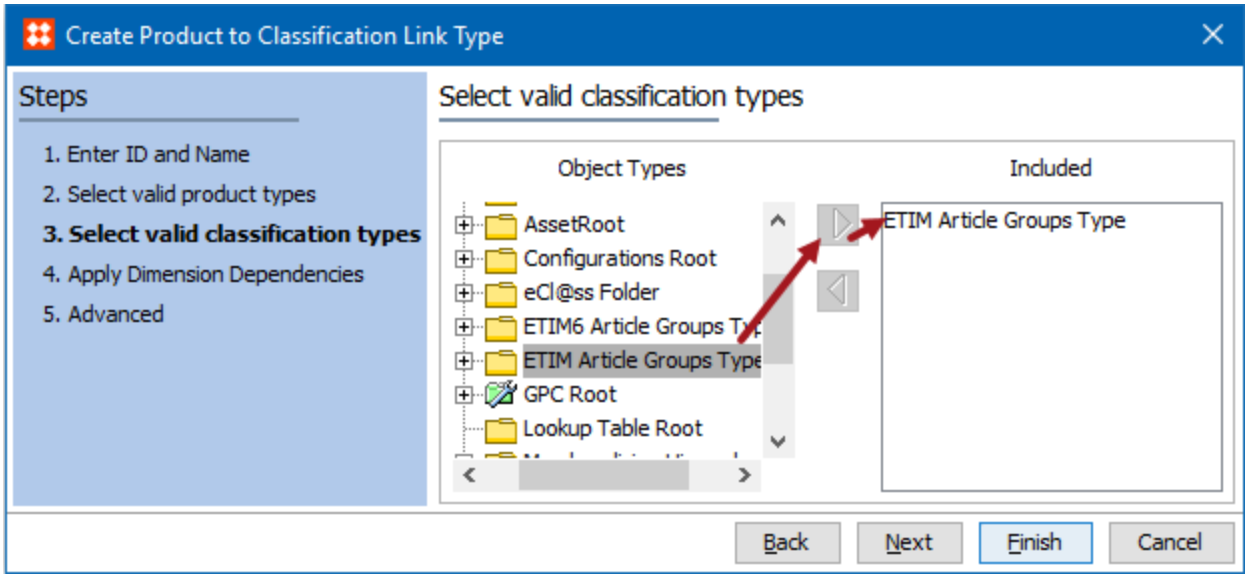
1. System Setup > Reference Types > **Product to Classification Link Types**, right-click and create a new Product to Classification Link Type.
2. Type an ID and Name for the new link type and click the **Next** button.



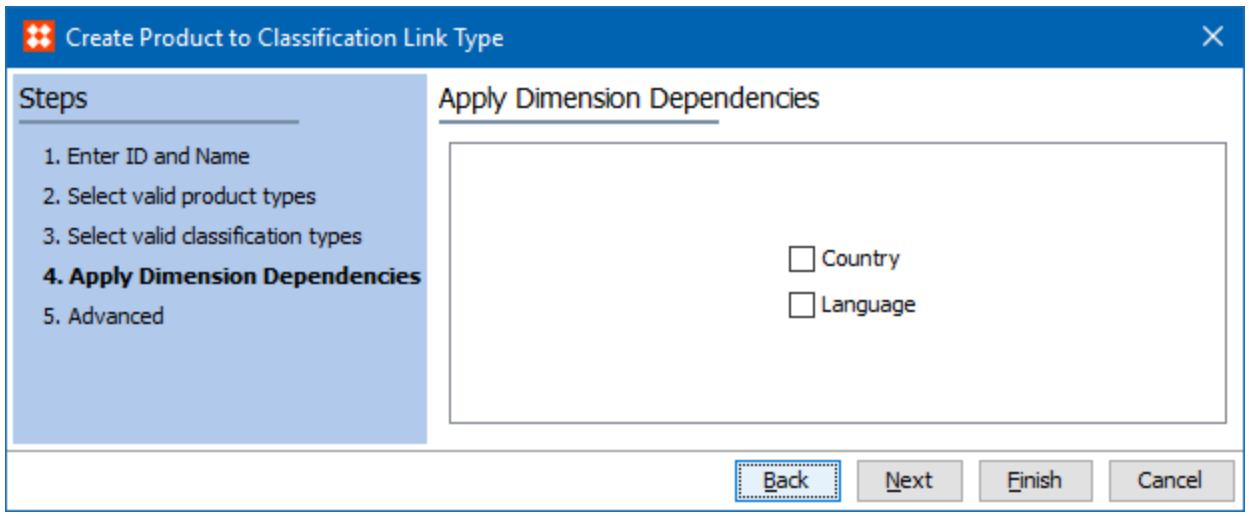
3. Select the valid product type(s), click the arrow button to move to the 'Included' list, and click the **Next** button.



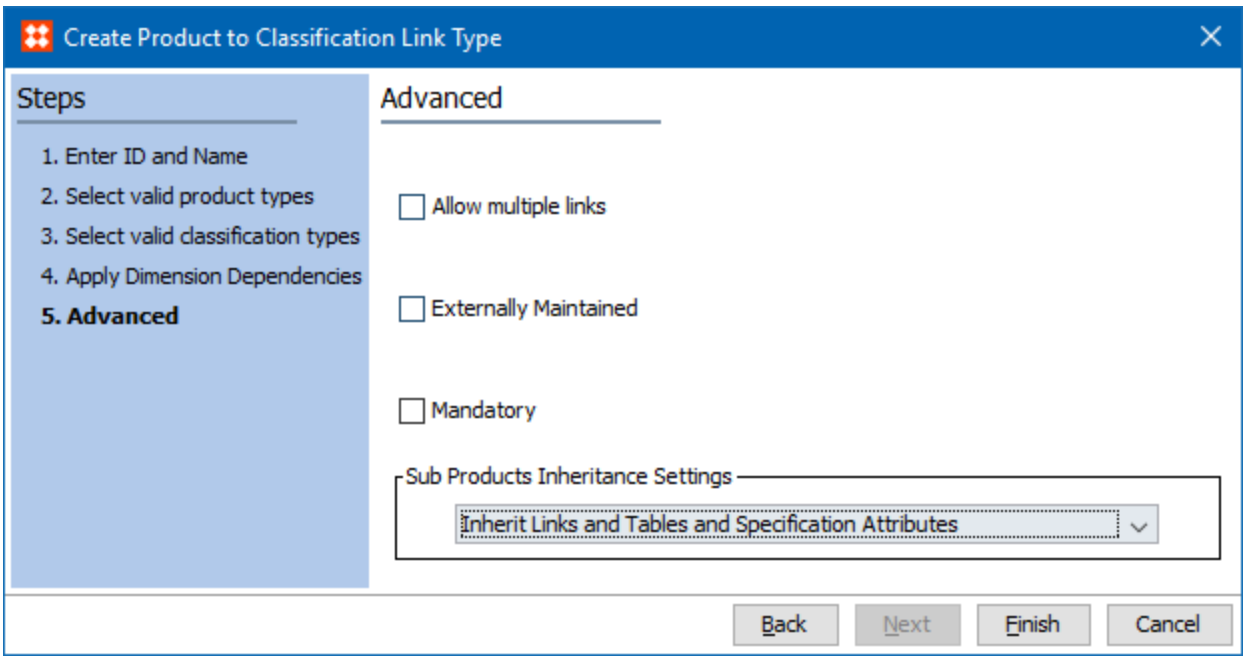
4. Select the valid classification type(s), click the arrow button to move to the 'Included' list, and click the **Next** button.



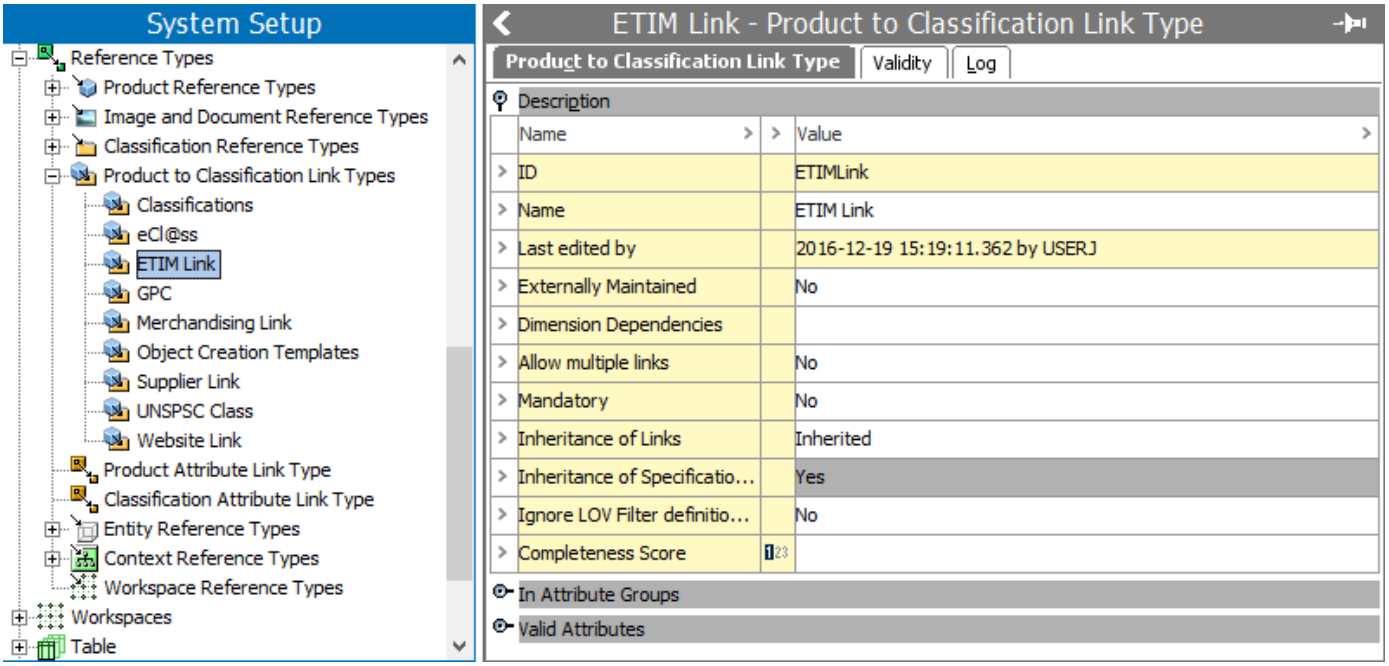
5. Select any necessary dimension dependencies and click the **Next** button.



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



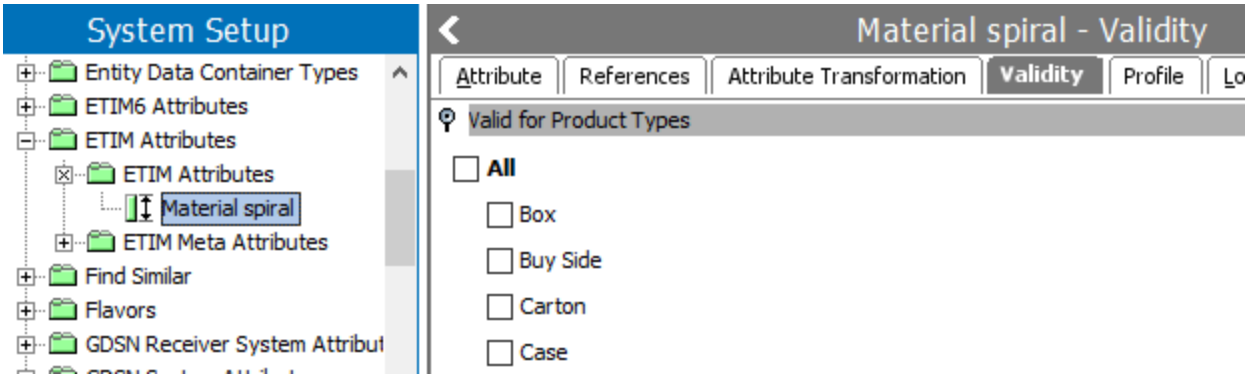
7. Click the **Finish** button and verify the correct settings are displayed in the editor.



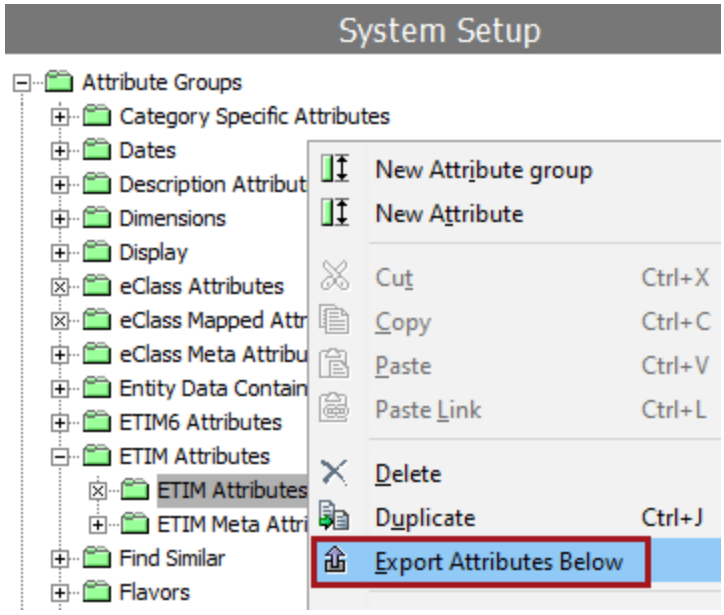
Set Validity on ETIM Attributes for Products

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

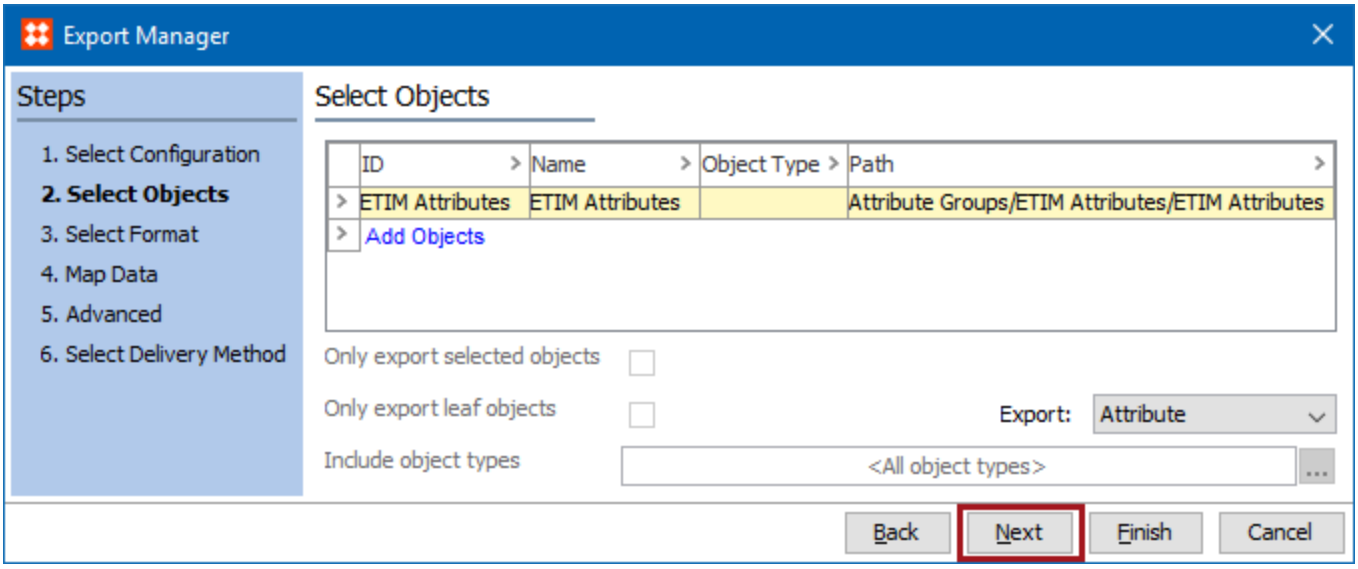
Use the following steps to set the valid product type on the Validity tab for the exported ETIM attributes.



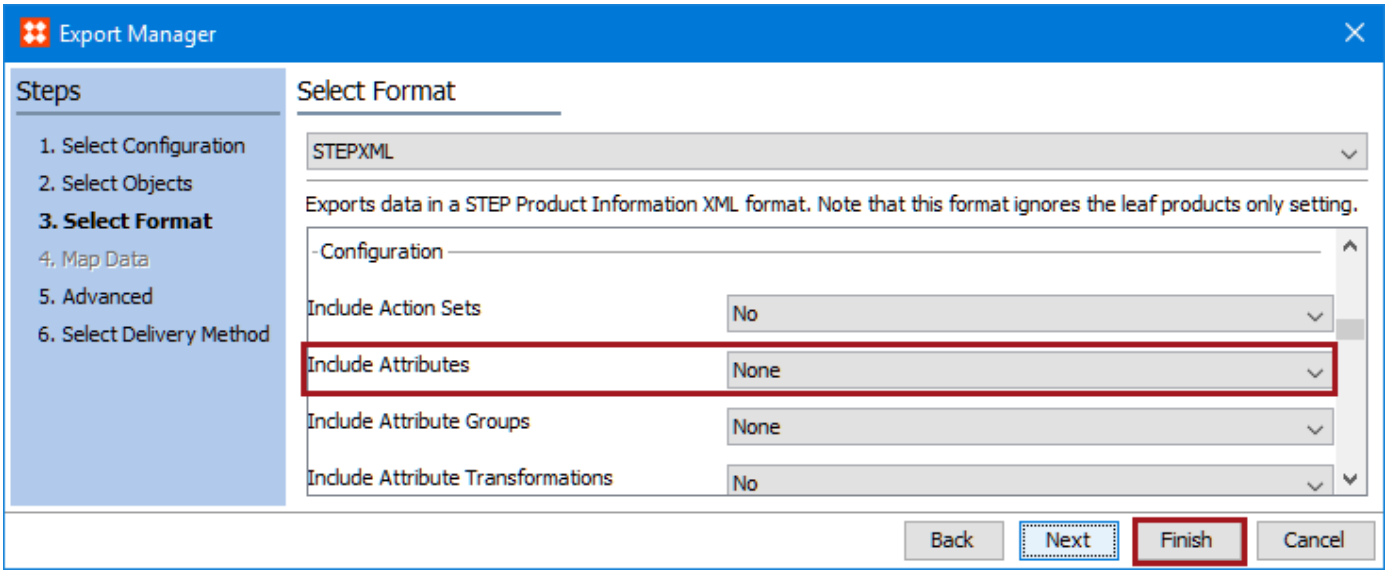
1. Determine the product object type that should be selected on the ETIM Attributes Validity tab.
2. System Setup > Attribute Groups > ETIM Attributes > ETIM Attributes, right-click and select **Export Attributes Below**. This assumes that you want to set validity on all ETIM attributes. If needed, select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



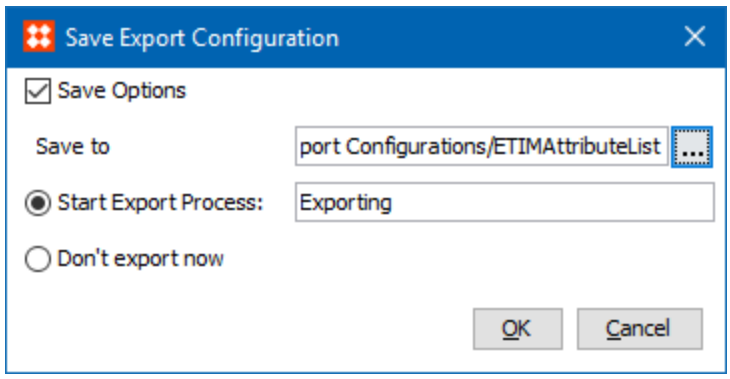
3. The Export Manager displays with the selected ETIM Attributes, click the **Next** button.



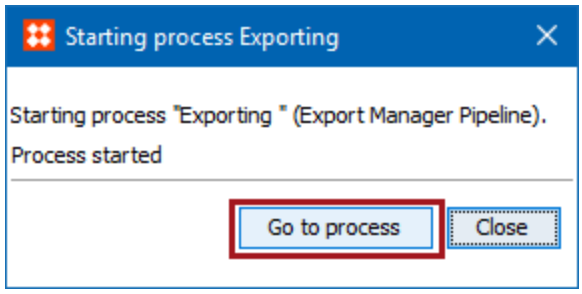
4. To ensure the smallest XML export file, on Select Format step:
   - Set **STEPXML** as the format.
   - Set the **Include Attributes** parameter to **Selected**.
   - Set all other parameters to No / None.
   - Click **Finish**.



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.

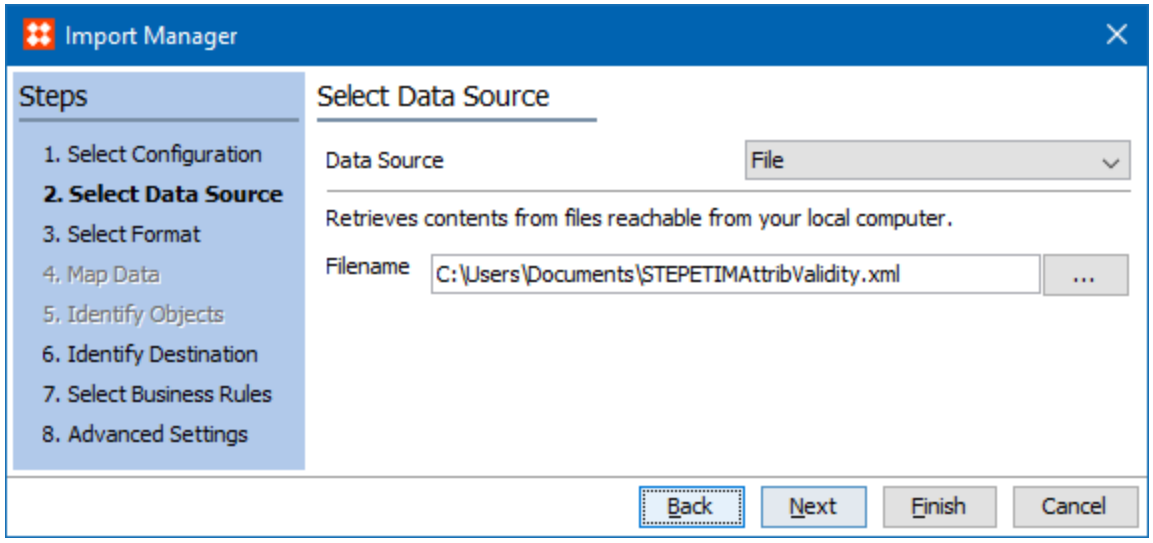


7. Open the export file for editing and make the following updates as shown in the image below. In the example, 'Item' is the product object type that will be set as valid.

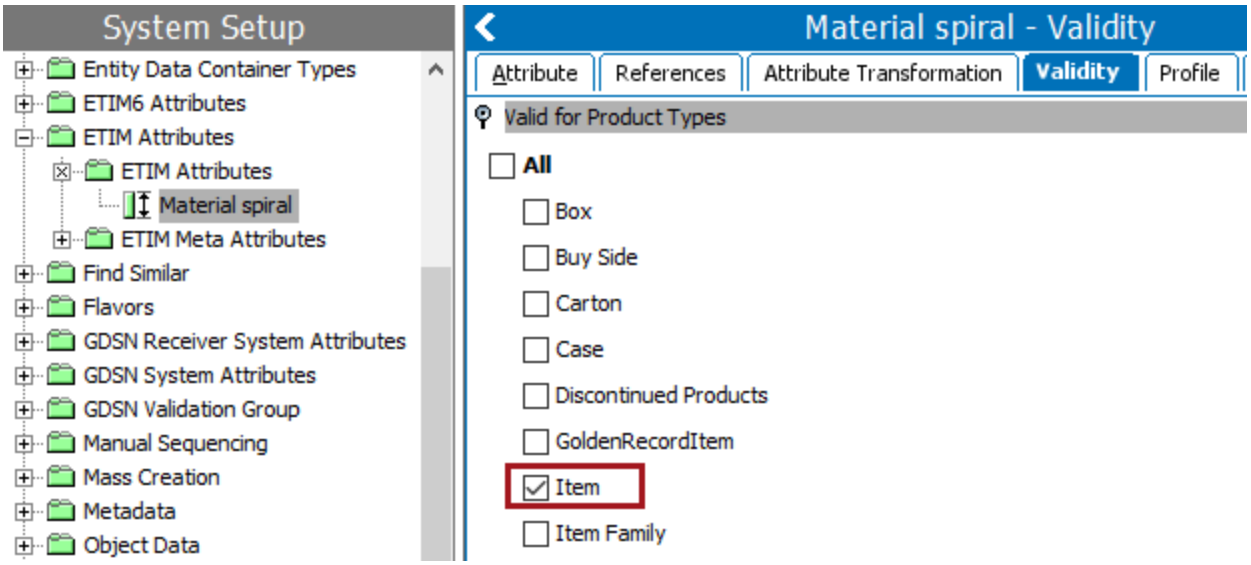
   Below each instance of the <AttributeGroupLink AttributeGroupID="ETIM Attributes"/> tag, add the text **<UserTypeLink UserTypeID="Item"/>**, substituting the text 'Item' with the name of your own desired product object type determined in the first step. The double quote marks are required.

```
<?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, see **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.

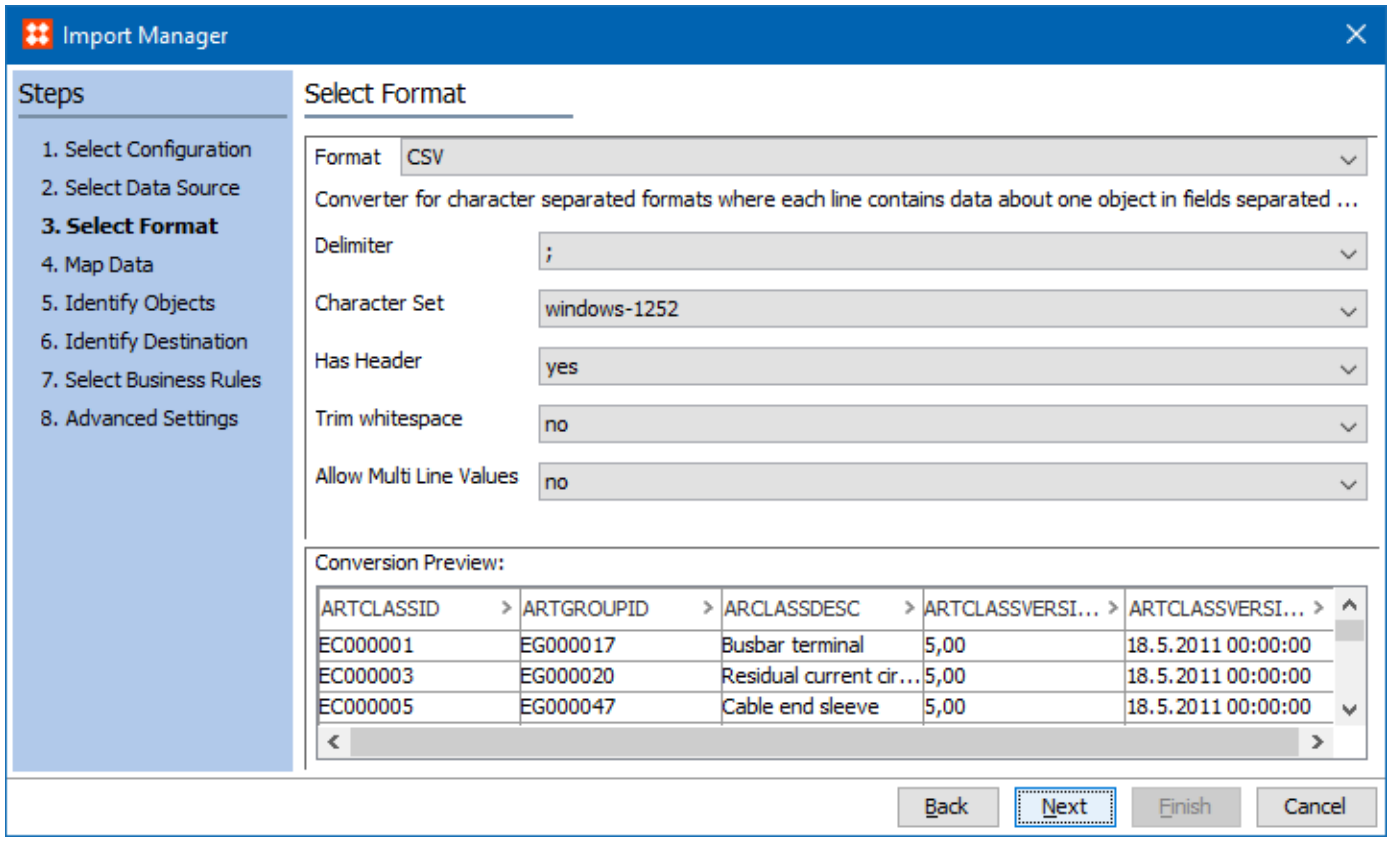
1. Start the Import Manager and complete the following steps:

- **Import Manager - Select Configuration**
- **Import Manager - Select Data Source**

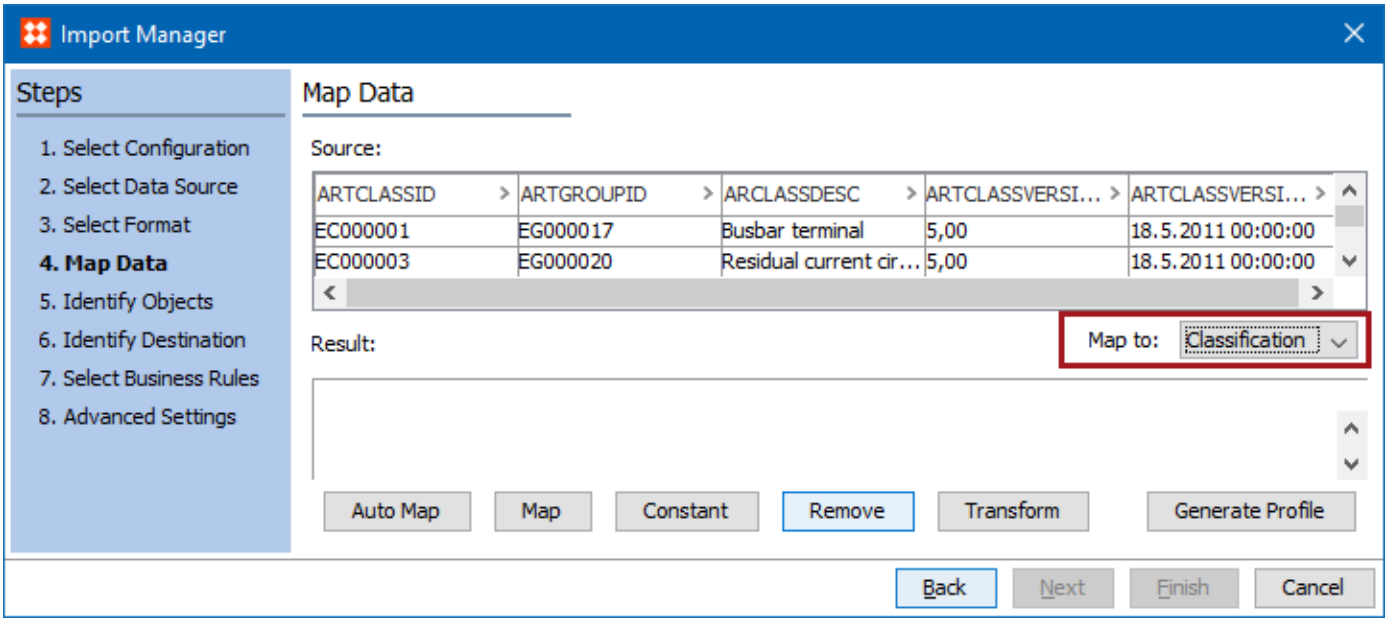
2. For the Import Manager Select Format step, select the appropriate ETIM version.

- ETIM version 5 using **ETIM and ETIM v2 Format**
- ETIM versions 6 and 7 using **ETIM IXF Format**

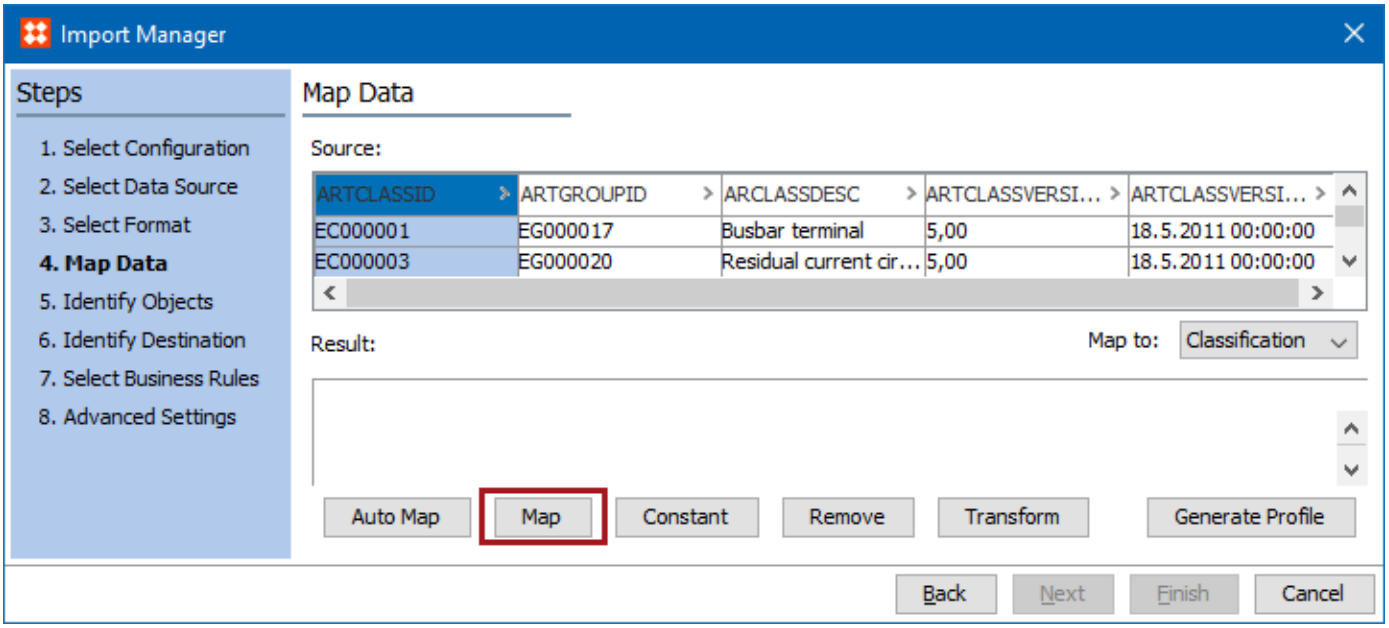
3. On Import Manager Select Format, choose **CSV** and leave the other settings as default. The Conversion Preview is displayed.



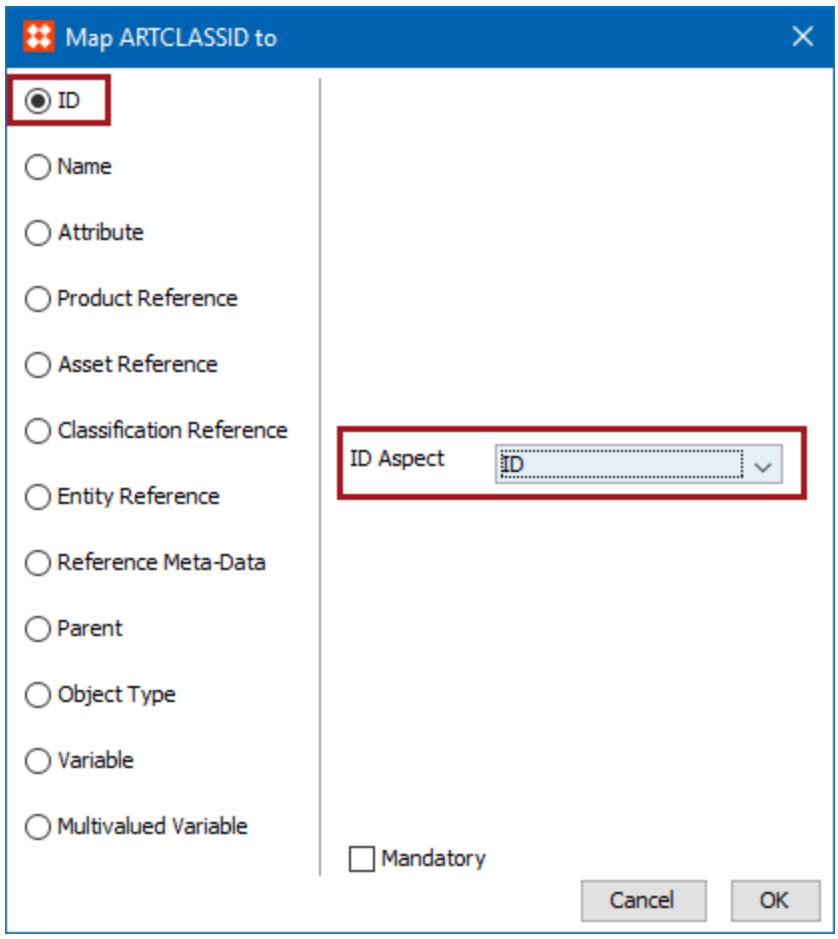
4. On Import Manager Map Data, set **Map to** as **Classification**.



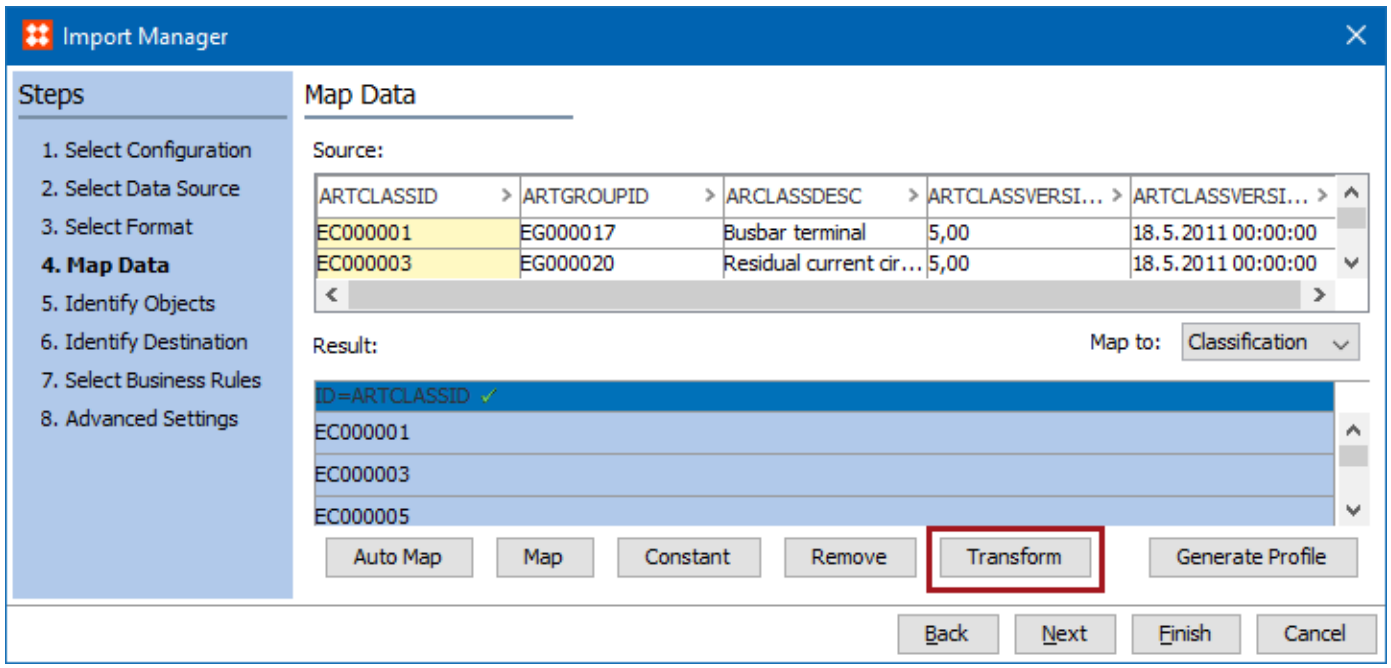
5. In the Source table, select the **ARTCLASSID** column and click the **Map** button below the Result table.



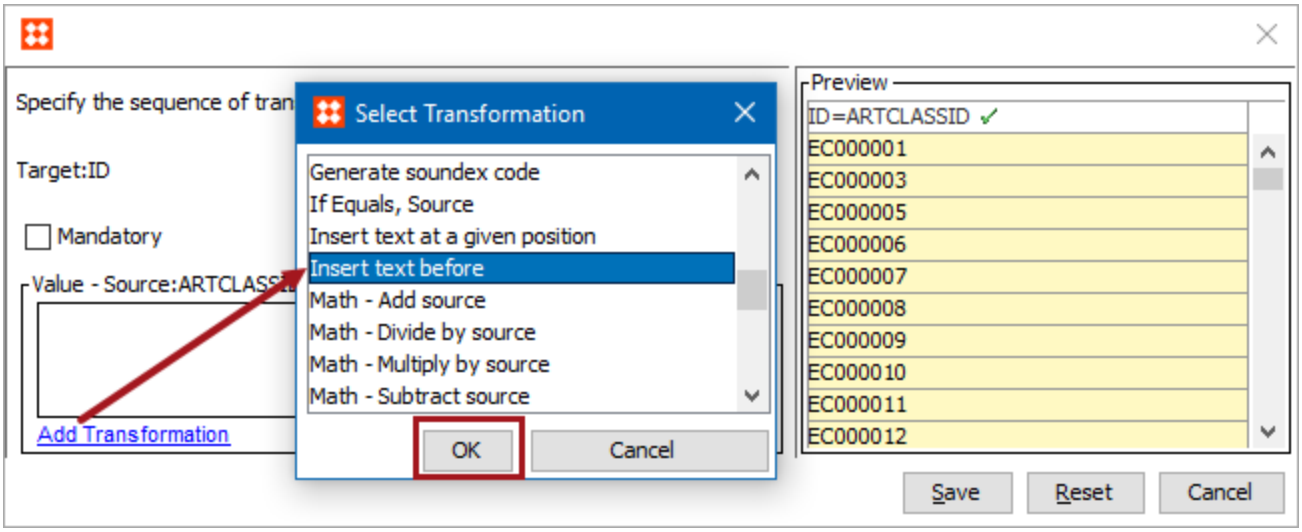
6. In the Map To window, select the **ID** radio button, set the **ID Aspect** to **ID**, and click **OK**.



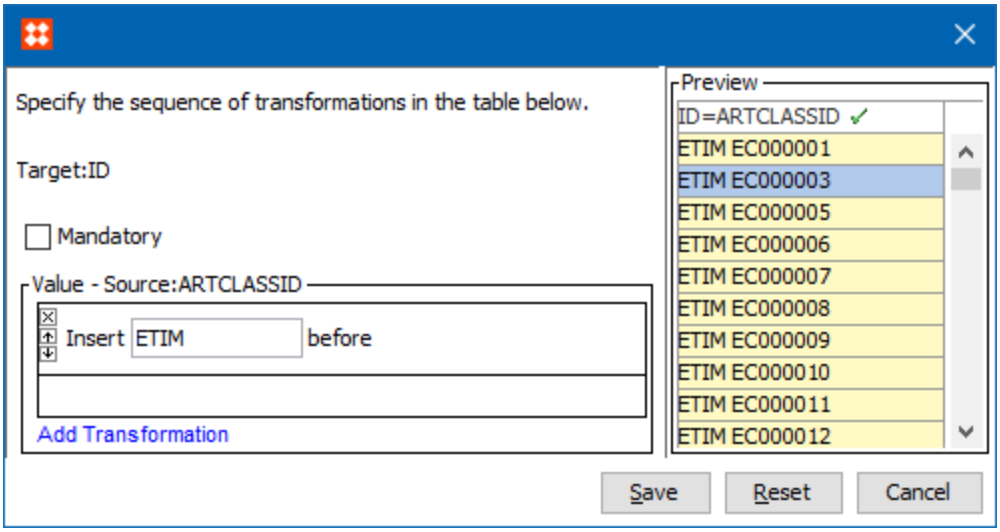
7. In the Result table, select the **ID=ARTCLASSID** column and click the **Transform** button.



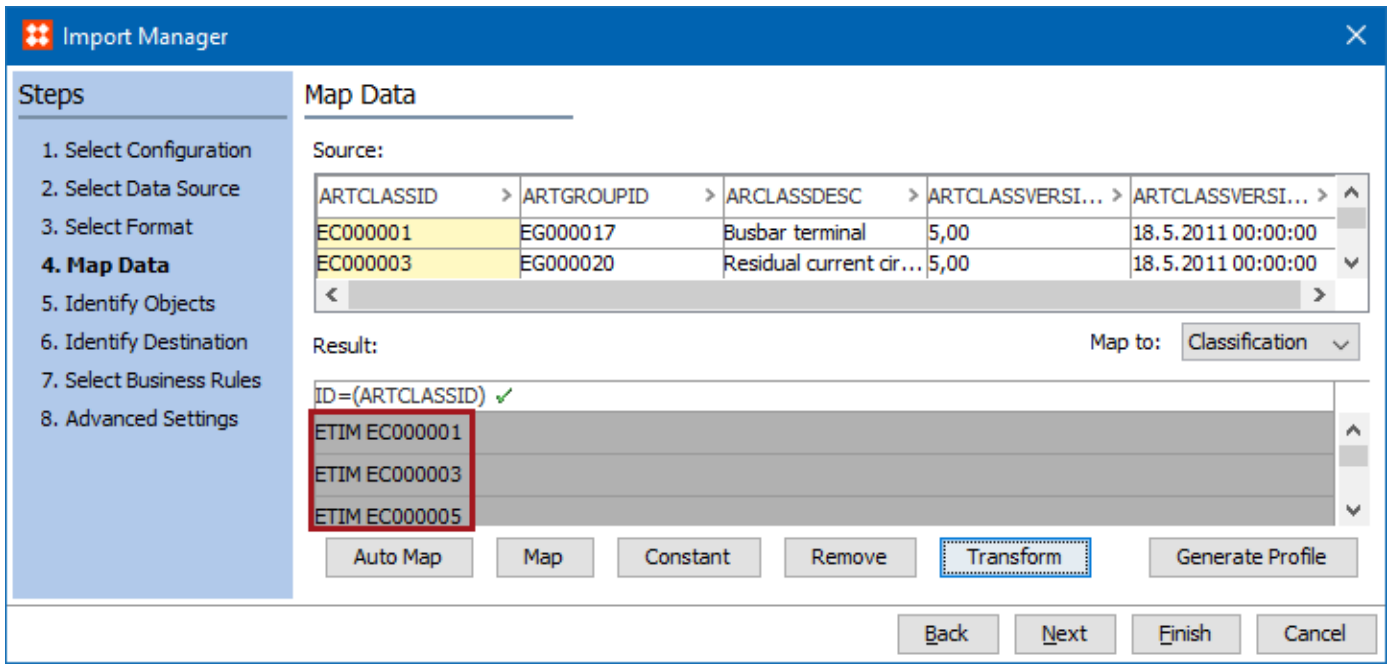
8. In the Transformation dialog, click the **Add Transformation** link, select the **Insert text before** option, and click **OK**.



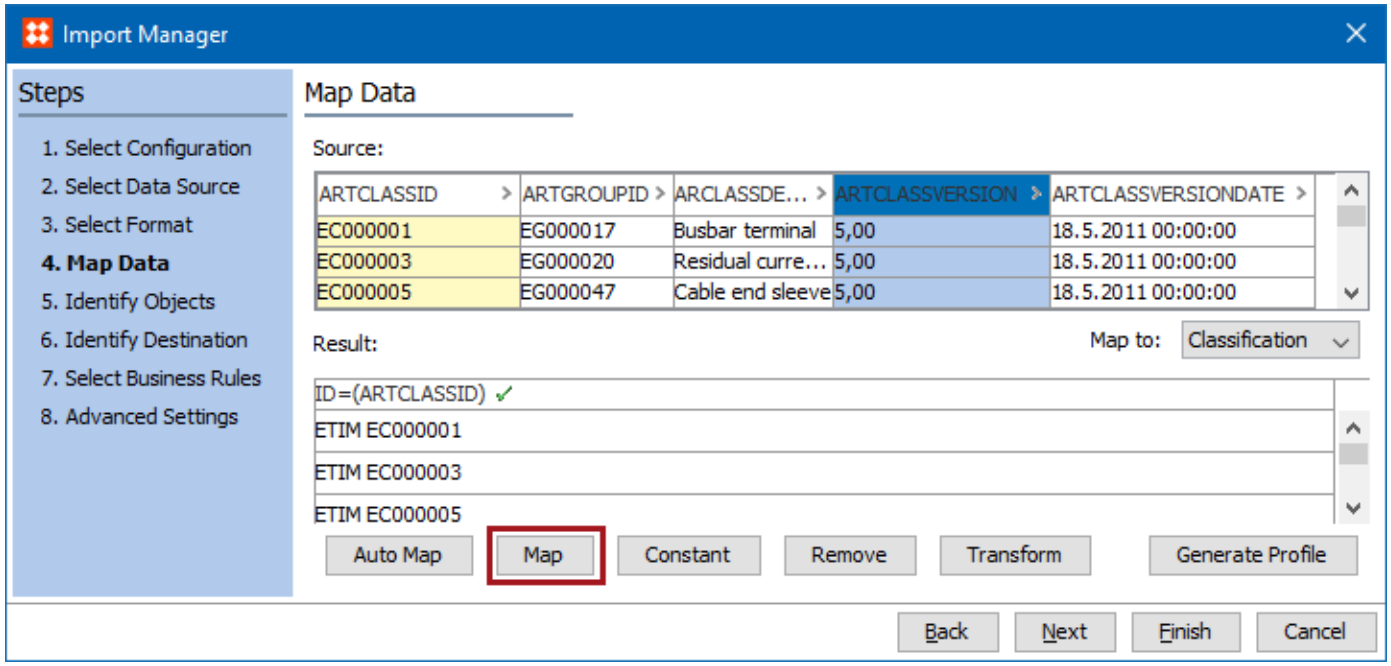
9. Enter the ETIM version your want to import (including a trailing space) as the value to insert in order to match the ID in STEP and click the **Save** button. This text entered is based on the suffix added when the ETIM Article Groups were imported. The results are displayed in the Preview pane.



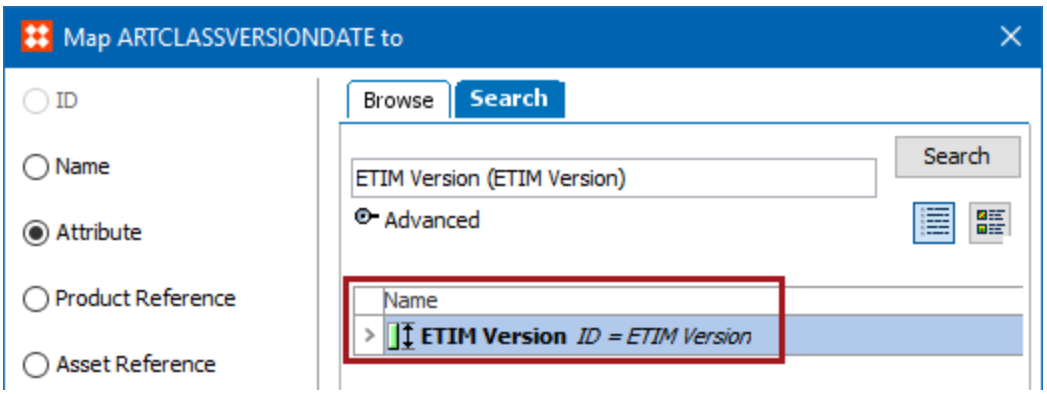
10. The Result table ID=ARTCLASSID column now displays the selected text preceding the ID originally displayed in the Source table.



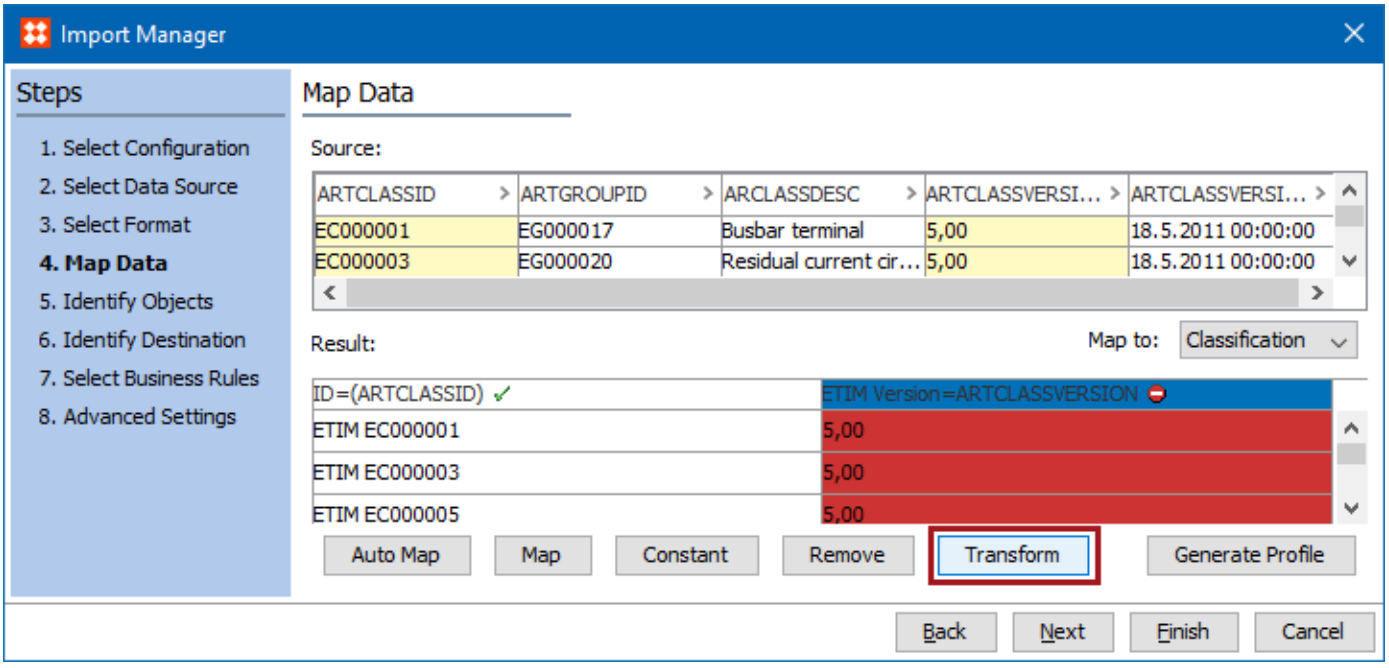
11. In the Source table, select the **ARTCLASSVERSION** column and click the **Map** button below the Result table.



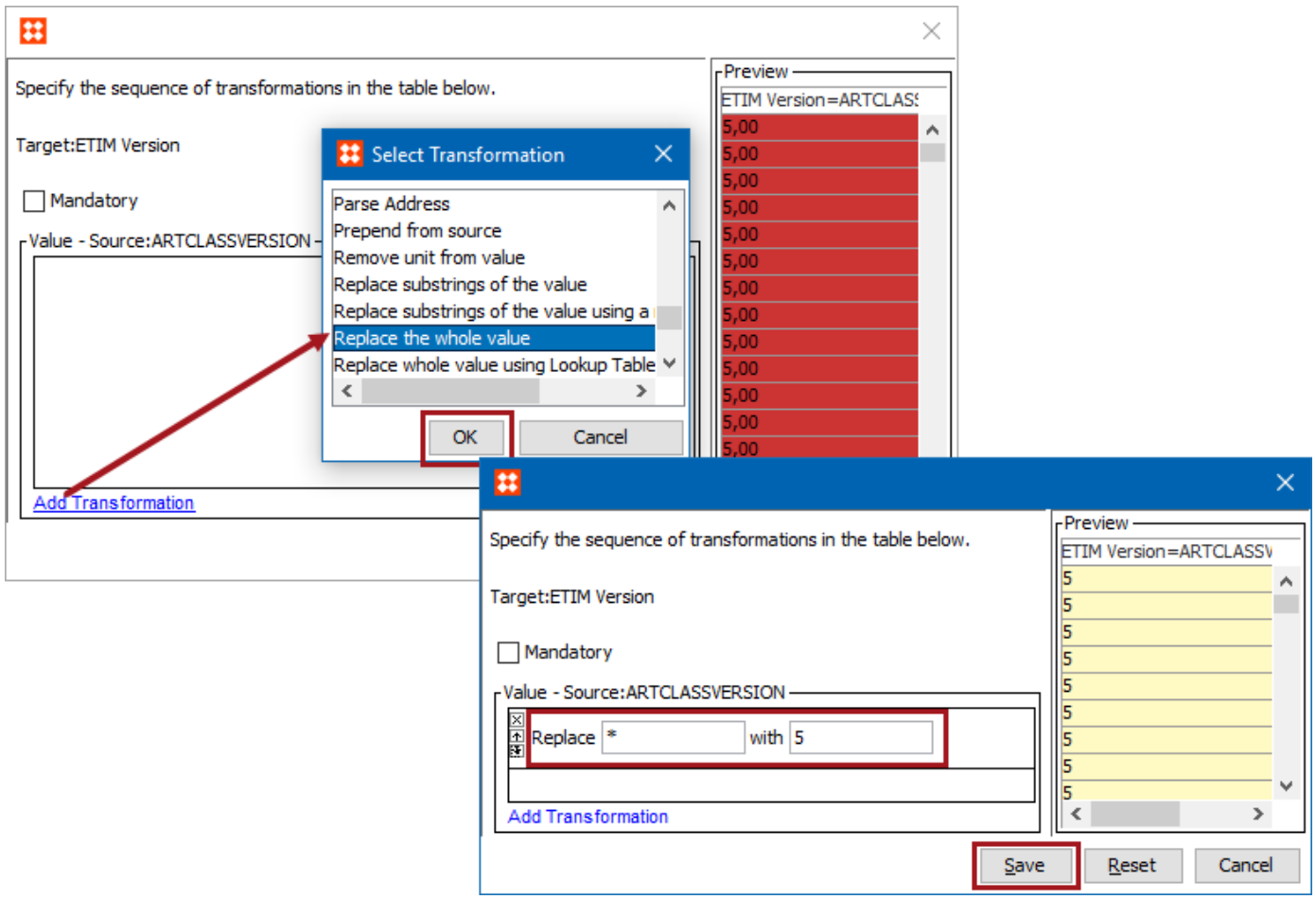
12. In the Map To window, select the **Attribute** radio button, search to find and select the **ETIM Version** attribute, and click **OK**.



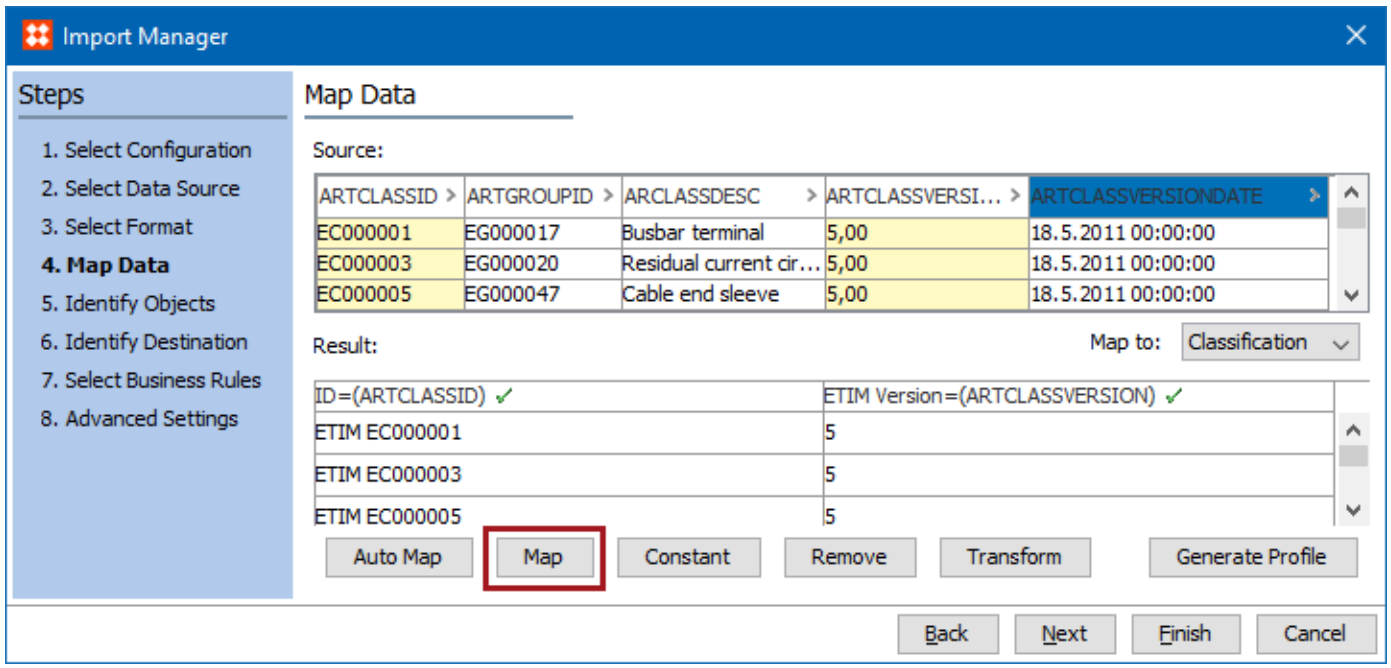
13. The ETIM Version attribute has a maximum length of three (3) characters, so it displays as an error in the Result table. Select the **ETIM Version=ARTCLASSVERSION** column and click the **Transform** button.



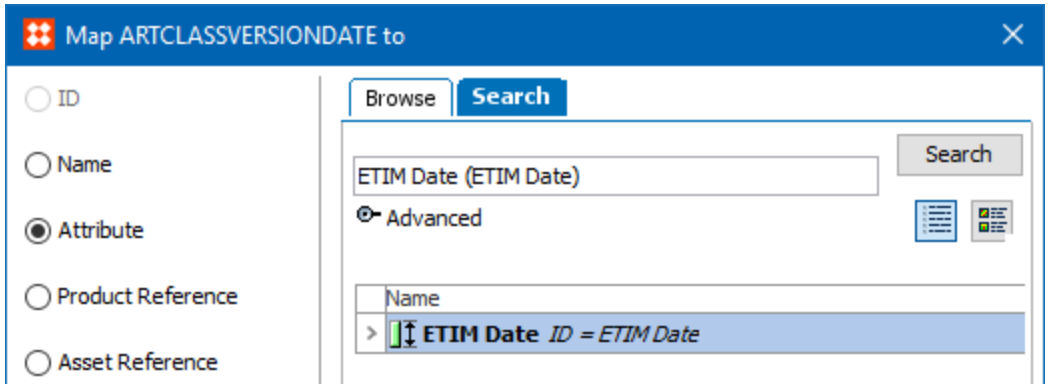
14. In the Transformation dialog, click the **Add Transformation** link, select the **Replace the whole value** option. For the values, add **replace * with** and enter a single digit based on the version being imported, and click the **Save** button.



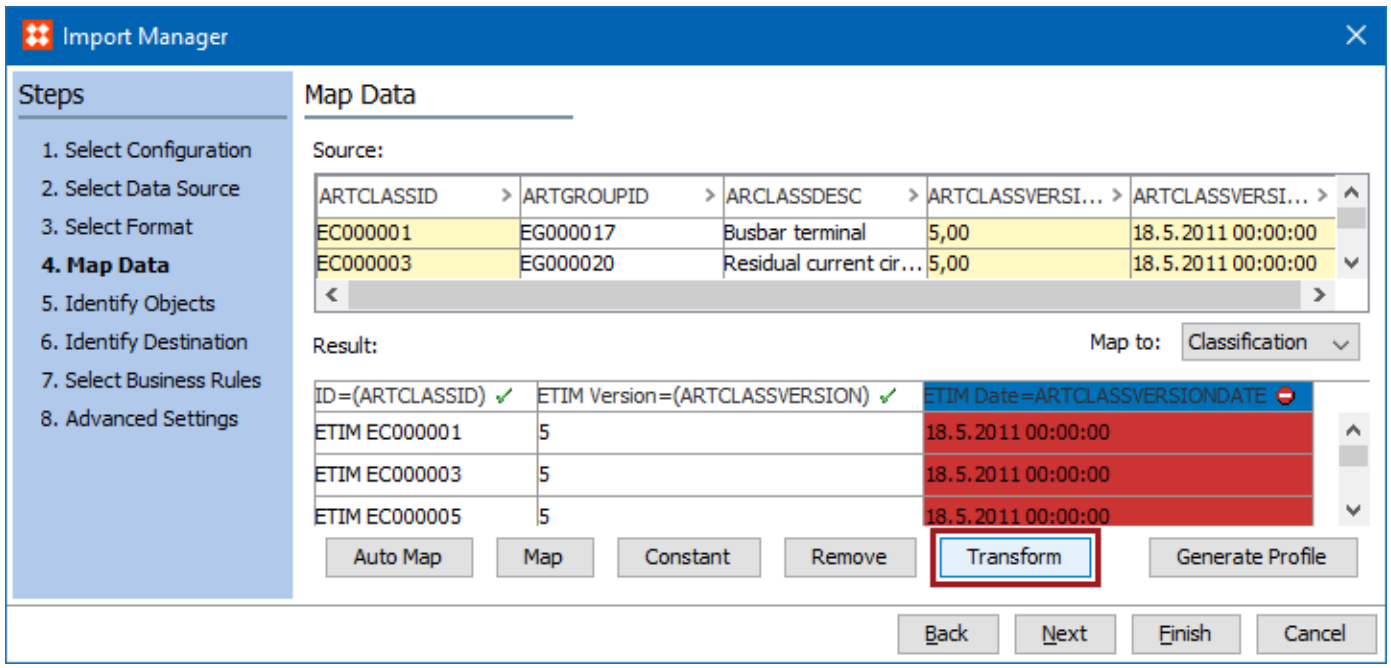
15. The Result table ID=ARTCLASSVERSION column now displays the selected single digit for all rows. In the Source table, select the **ARTCLASSVERSIONDATE** column and click the **Map** button below the Result table.



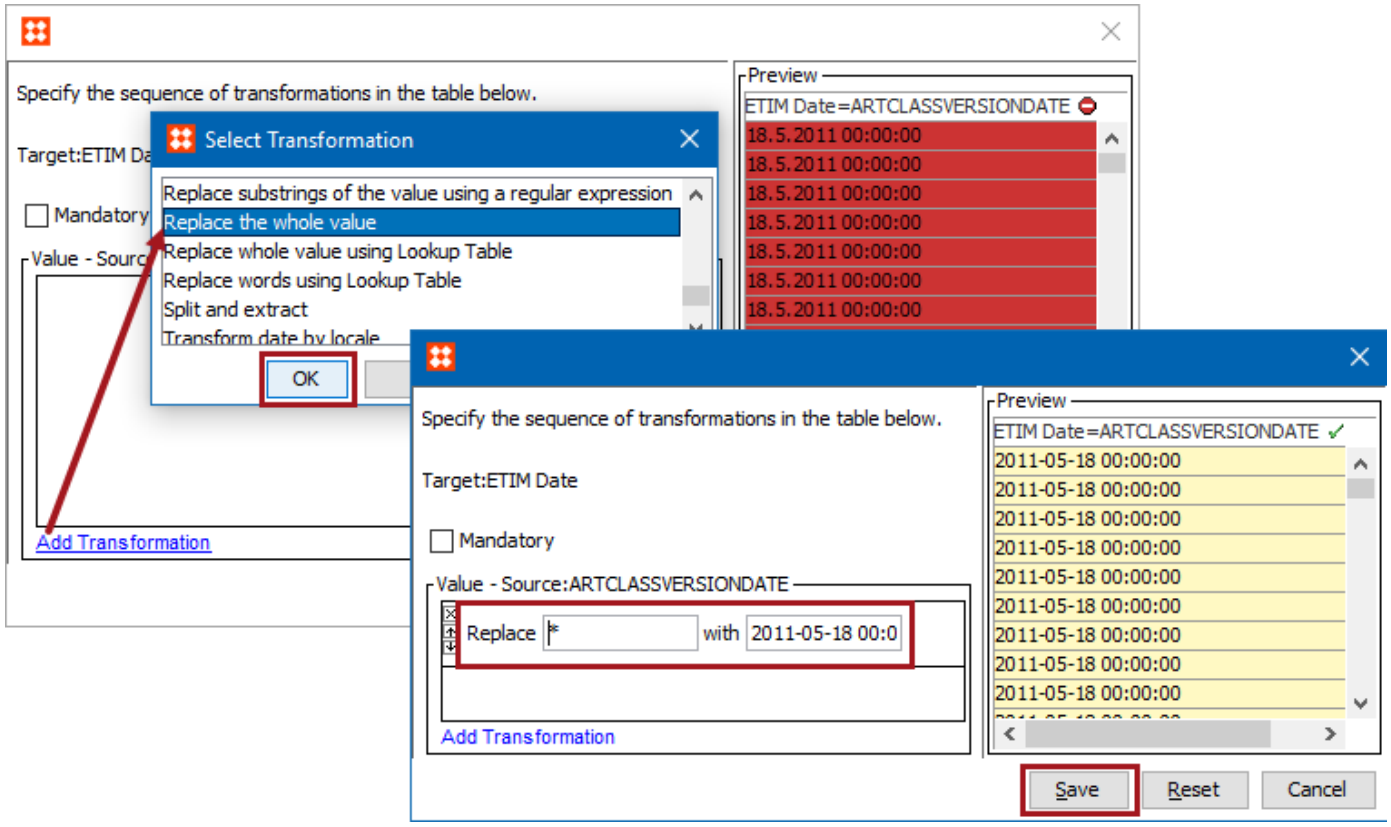
16. In the Map To window, select the **Attribute** radio button, search for and select the **ETIM Date** attribute, and click **OK**.



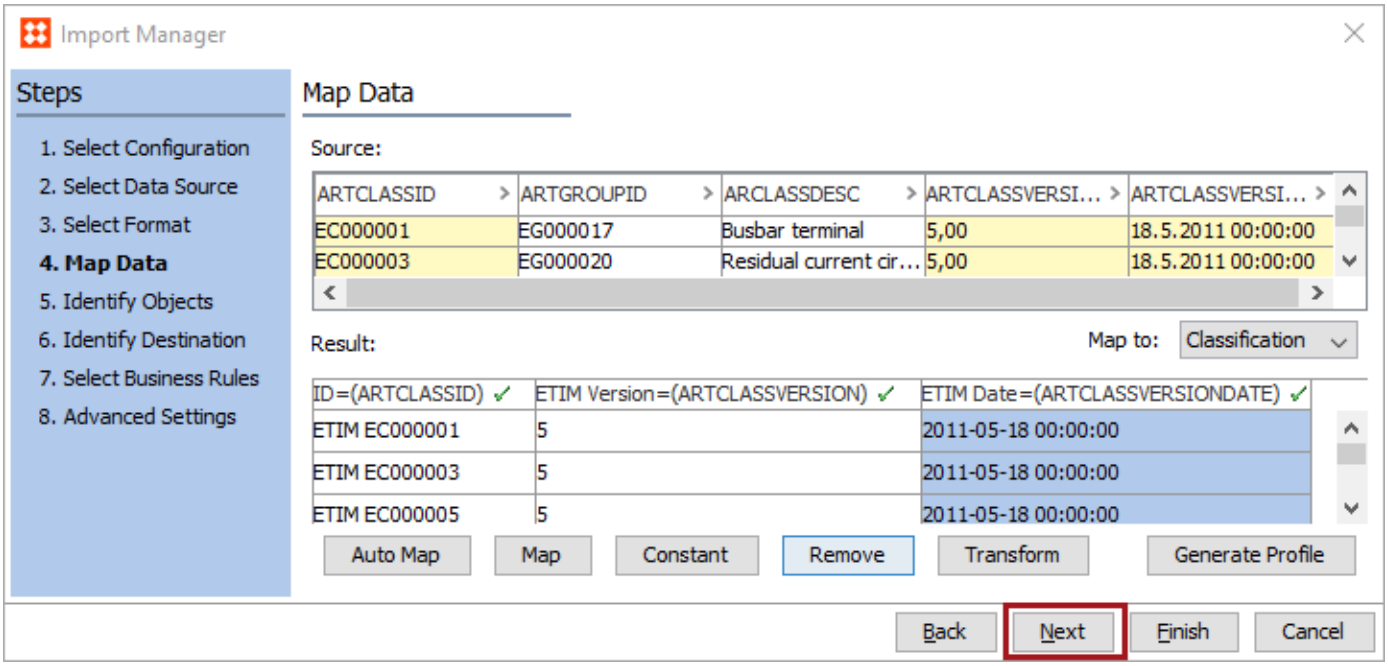
17. The ETIM Date attribute has a validation type of 'ISO Date and Time' format, it does not match the format in the ETIM file, so it displays as an error in the Result table. Select the **ETIM Date=ARTCLASSVERSIONDATE** column and click the **Transform** button.



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.

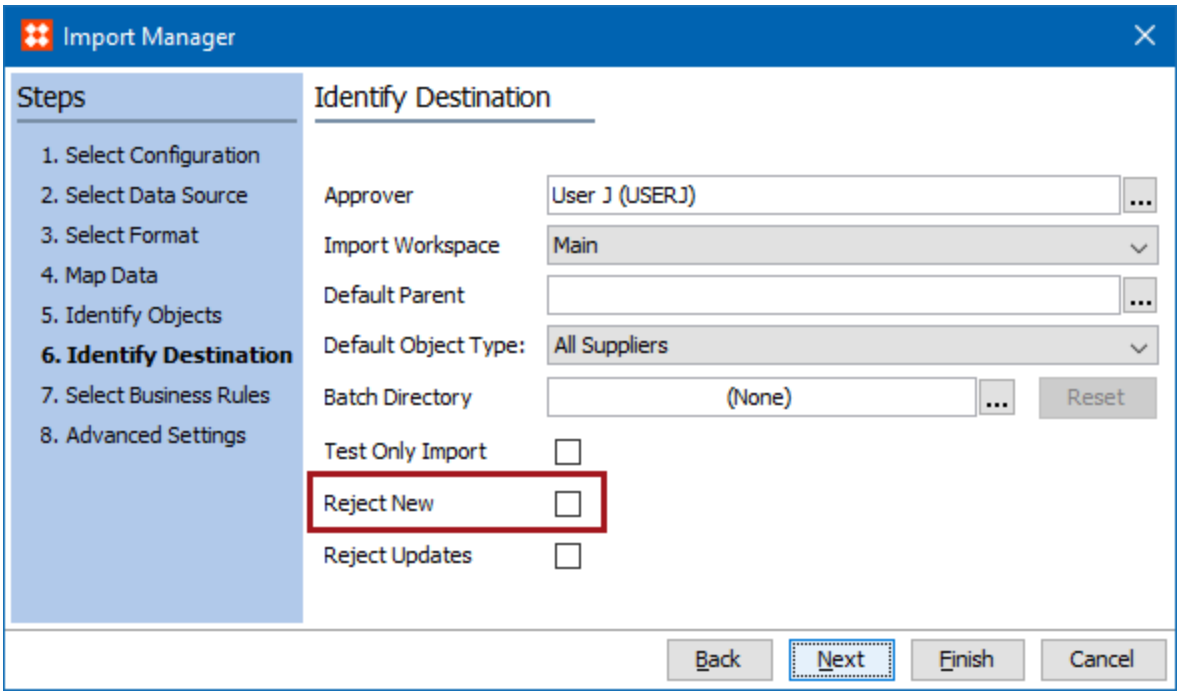


19. All three (3) mapped columns now show valid values in the Result table. Click **Next** (selecting defaults) until the Import Manager Identify Destination step displays.

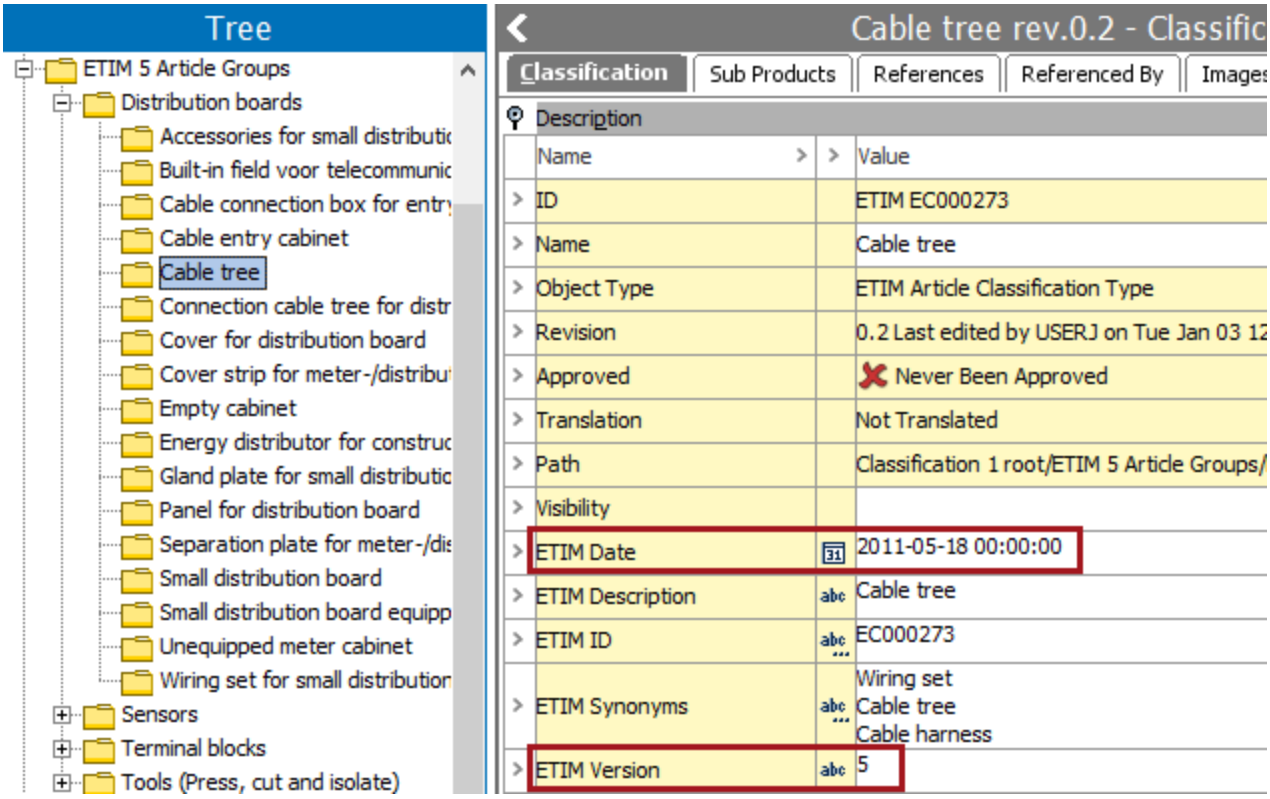


20. Determine how to set the **Reject New** checkbox and click the **Finish** button:

- **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.

The available *export* versions for Excel within STEP are as follows. However, versions of Excel newer than 2007 can be used for imports, and selecting 2007 for an export still means that the exported data will be readable by a later version of Excel.

- Excel 2003 - limit 65,536 rows
- Excel 2007 - limit 1,048,576 rows

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, see 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.
- 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.

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

Format Availability

Excel is available for selection in:

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

- Export Manager - see **Creating a Data Export**
- OIEP - see **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, see **Data Mapping**.

Inbound Data

Excel import allows creation of and updates to products, classifications, entities, attribute values and references. However, system setup objects (for example, attributes, 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.
- 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. See the 'Inbound Parameters' subsection below for more information.
- Enter attribute values and units 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.

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 within this guide.

Excel can be used to modify or create multiple attributes at the same time within System Setup. For details, see 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.

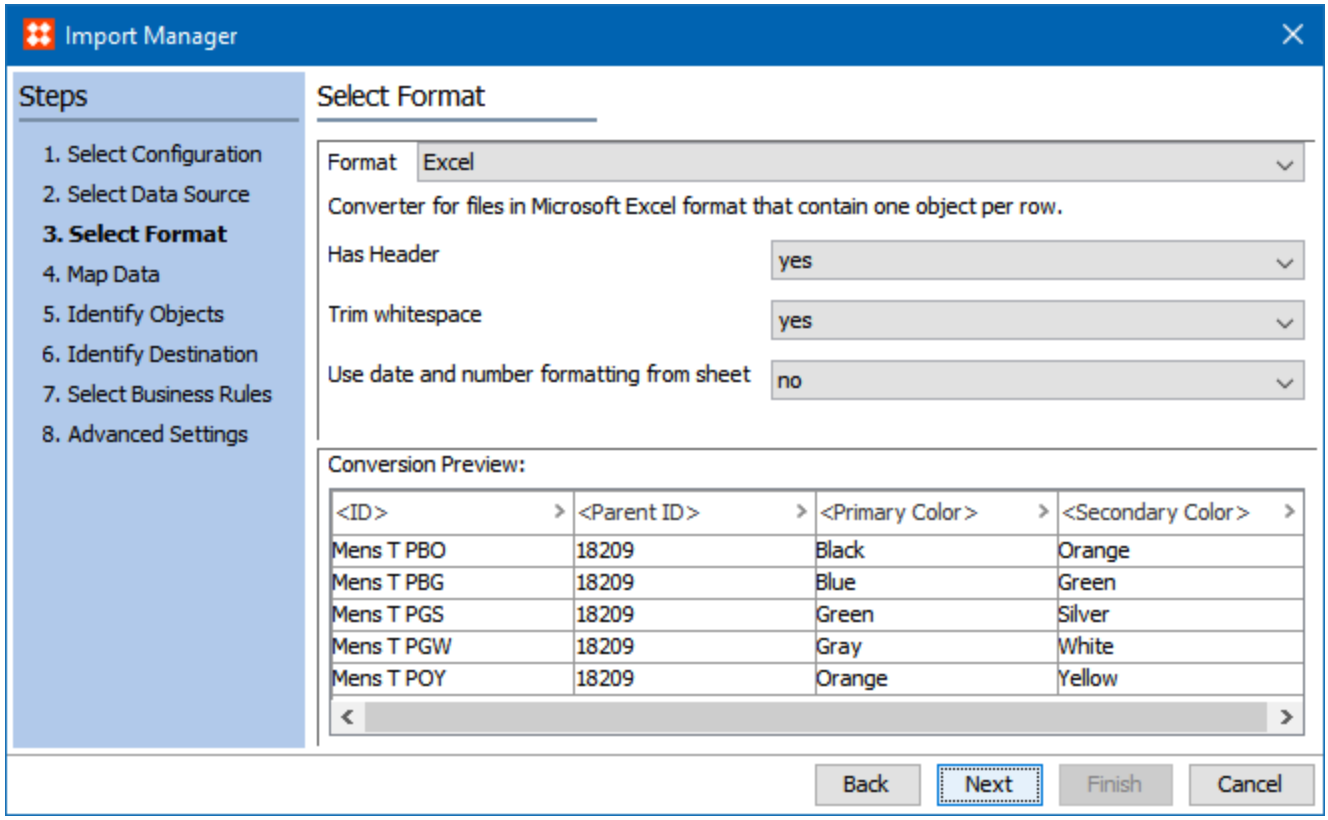
This functionality differs from imports using STEPXML or CSV files.

Inbound Parameters

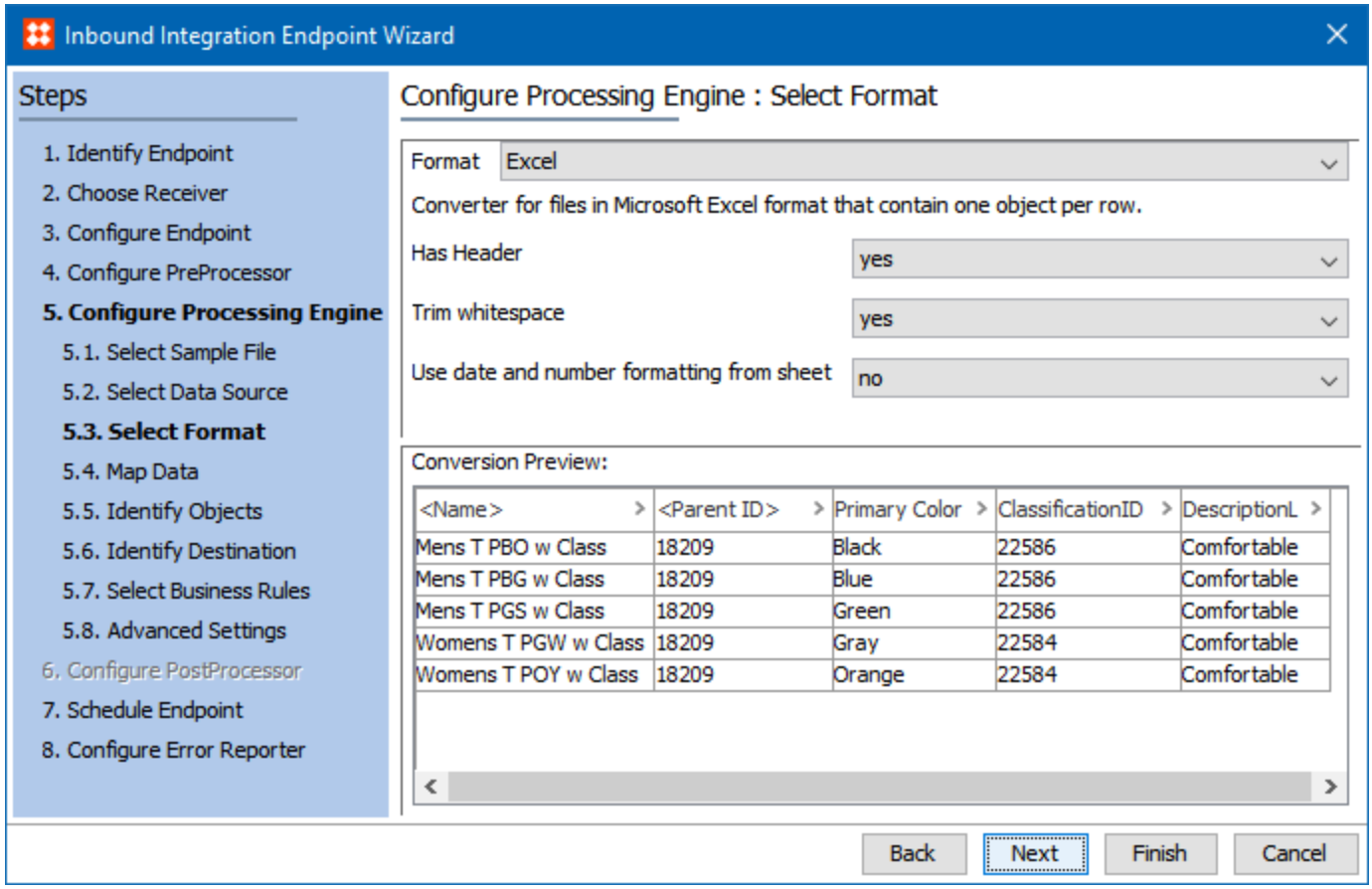
The following parameters are available in both Import Manager and IIEP.

- **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, see the **Date and Number Formatting with Excel** topic.

Import Manager



IIEP

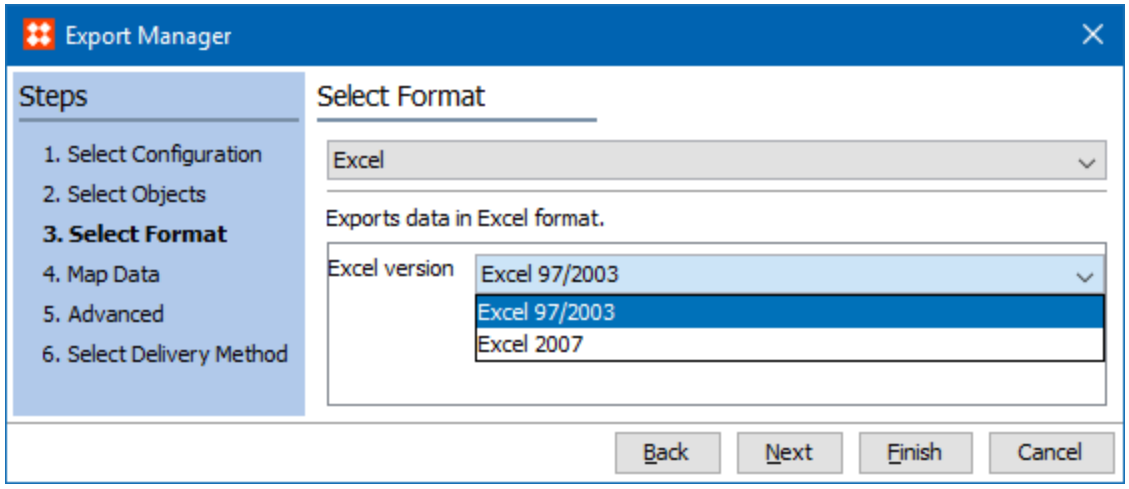


Outbound Data

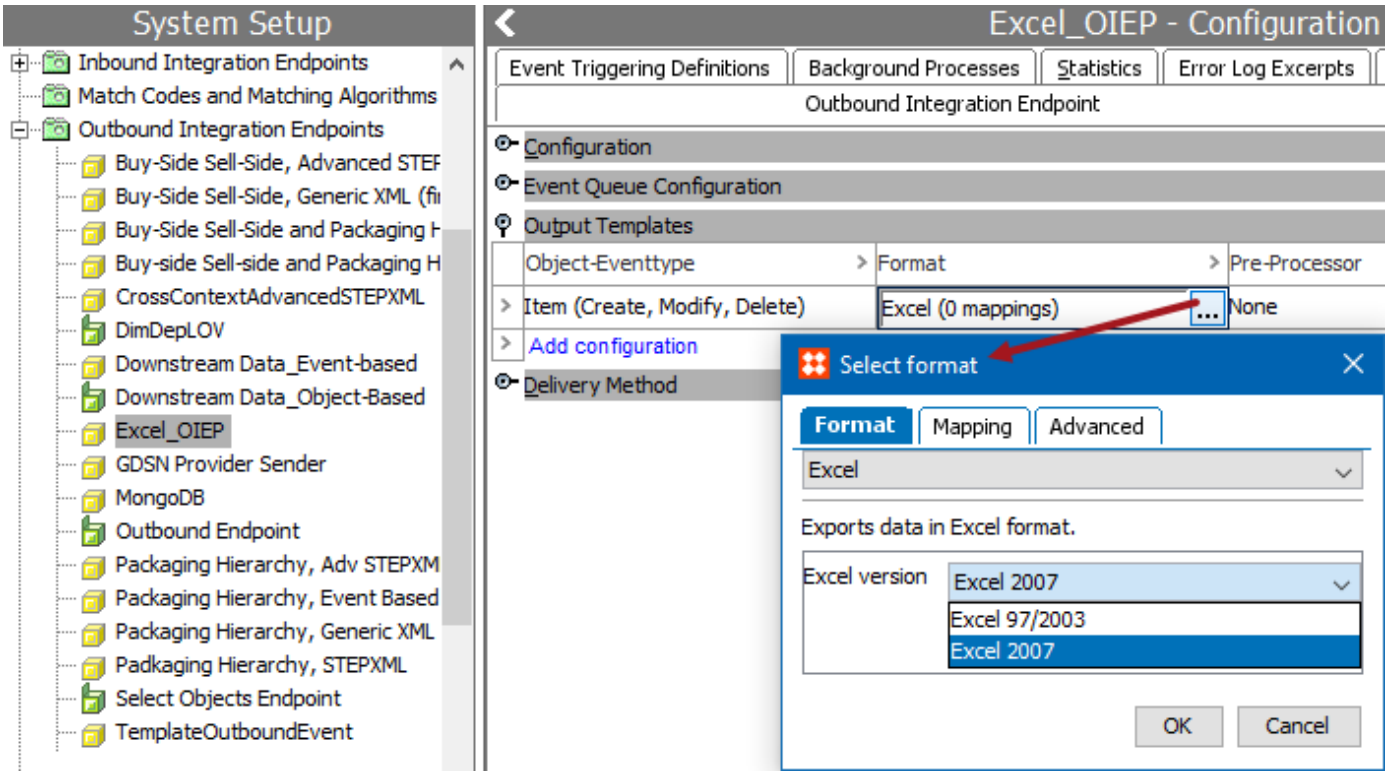
When data leaves STEP via Excel format, the UTF-8 character set is used to export the data.

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, see the **Aspect - Transform Outbound** section of the **Data Mapping** documentation.

Export Manager



OIEP

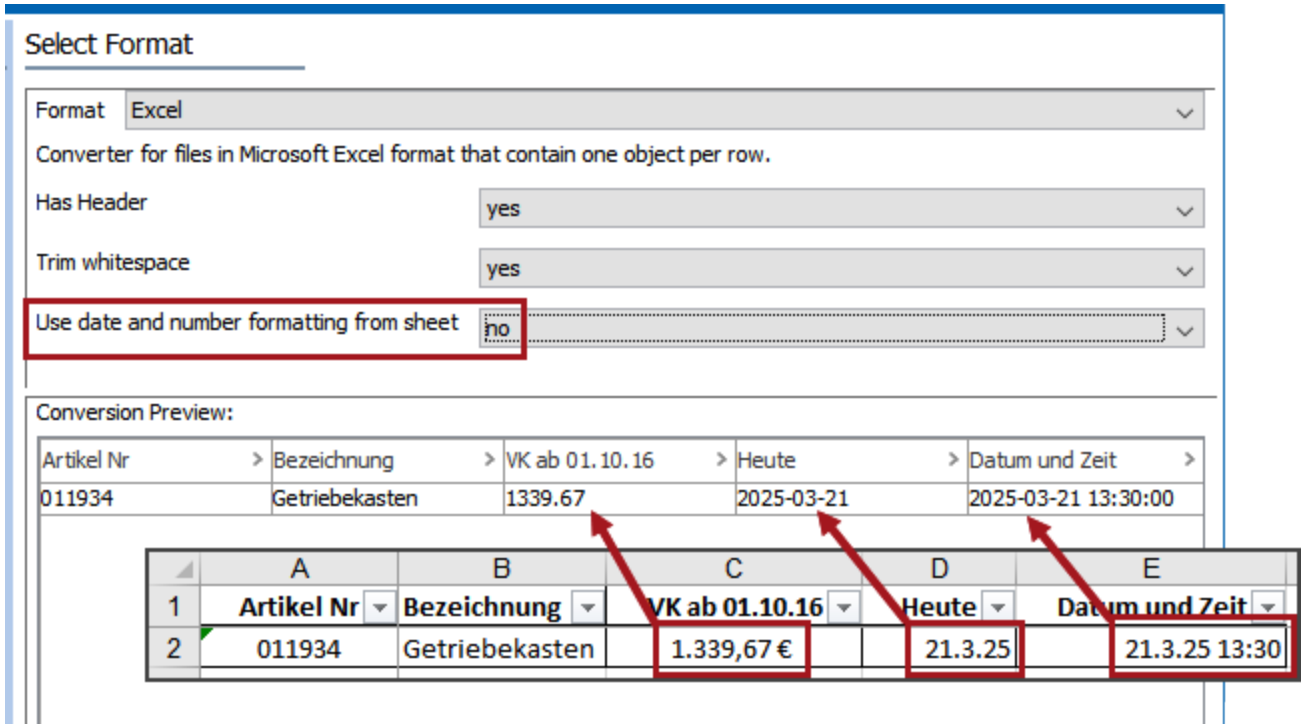


Date and Number Formatting with Excel

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.

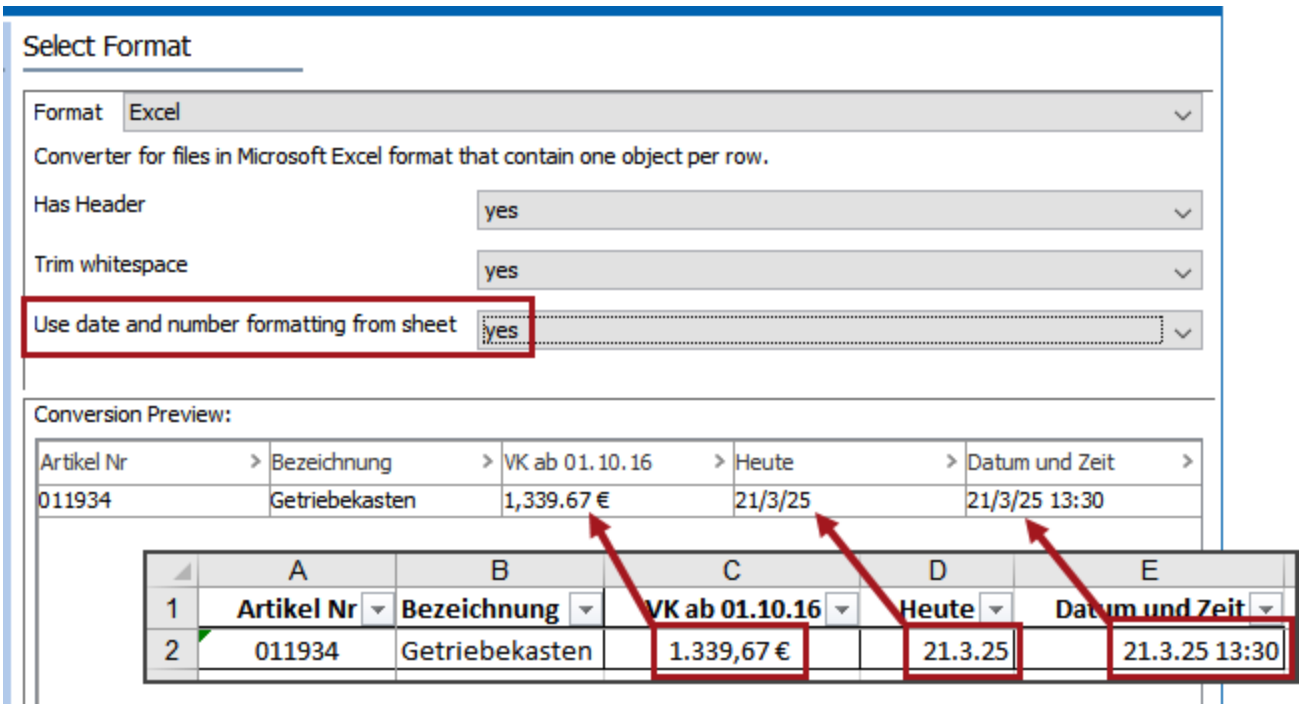
As an example of the 'no' setting, the below screenshot shows the Conversion Preview for a spreadsheet created on a computer with its region set to Germany, displaying currency in a Euro format and dates in a German format. When **no** is selected for 'Use date and number formatting from sheet,' the 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, see 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, see the **Excel Format** topic.

Importing Multi-Valued References or Links with Excel

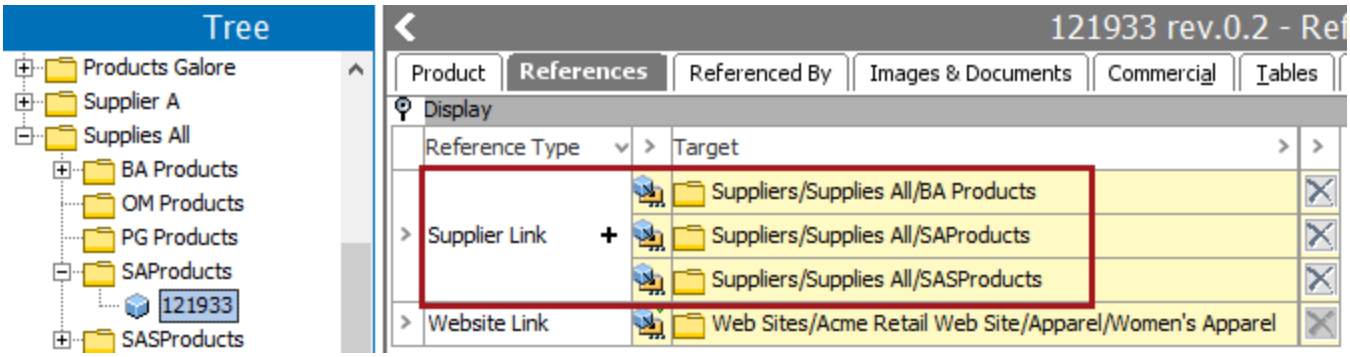
When using Excel to import a reference type that allows multiple references, it is important to keep in mind the following points to ensure the intended references are applied in STEP.

- 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

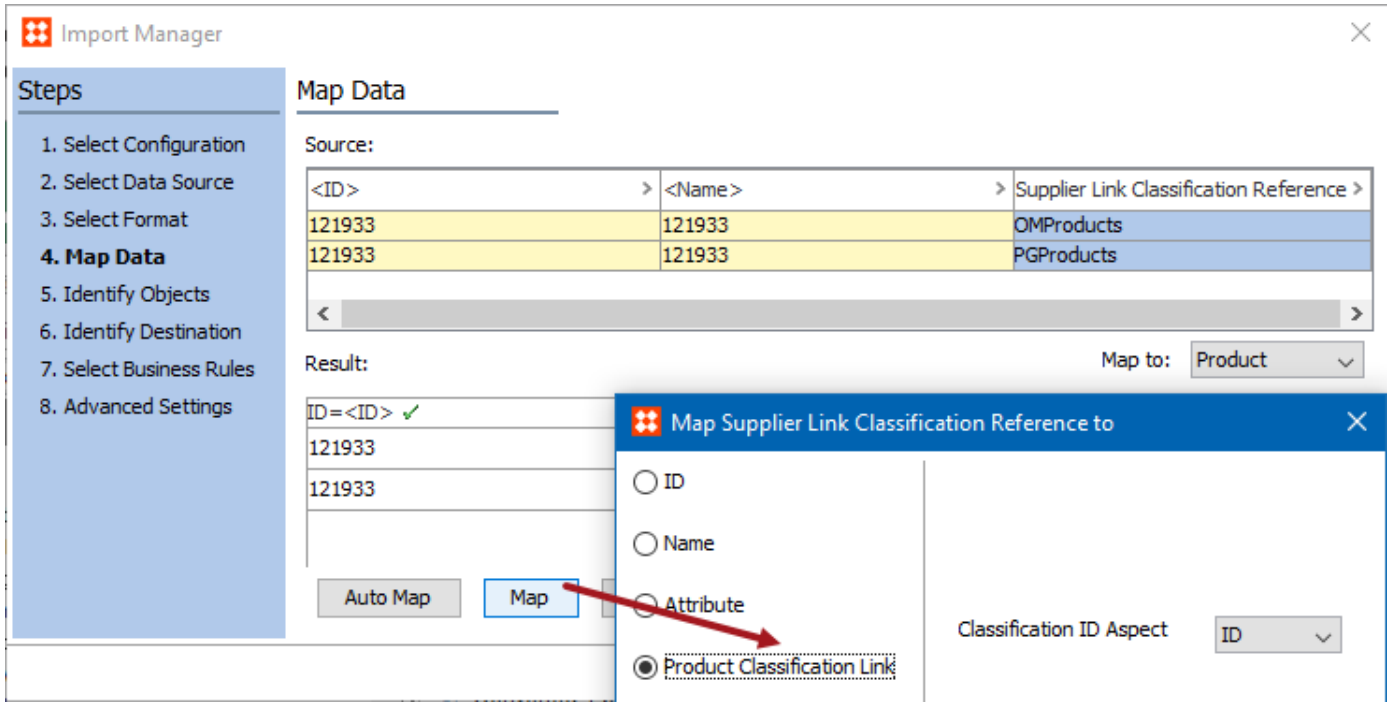
Consider the scenario shown below where in STEP the product with ID 121933 is already linked to three classifications (ID BAProducts, ID SAProducts, and ID SASProducts) using the product to classification link type Supplier Link. This link type has the 'Allow multiple links' option enabled.



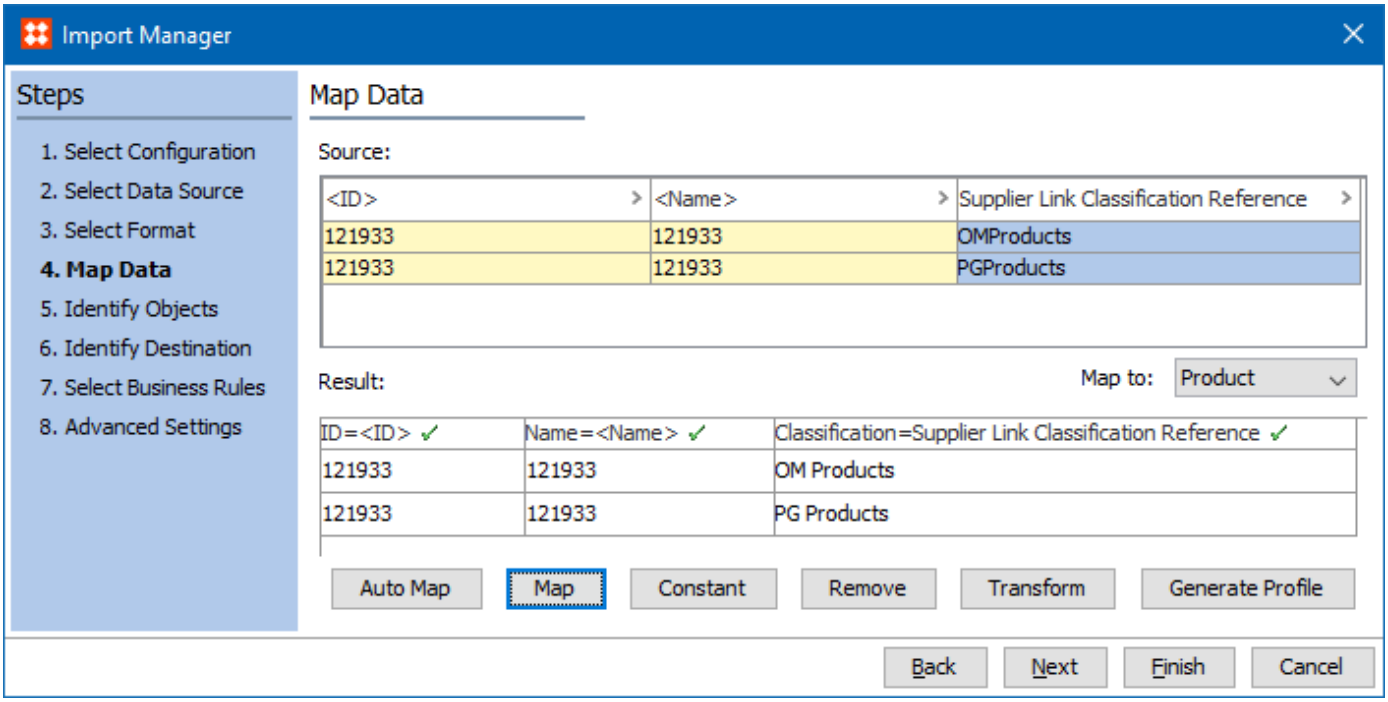
An Excel import file has been prepared and includes two different valid classifications (ID OMProducts and ID PGProducts) for the product to classification link type with the ID Supplier Link.

	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.

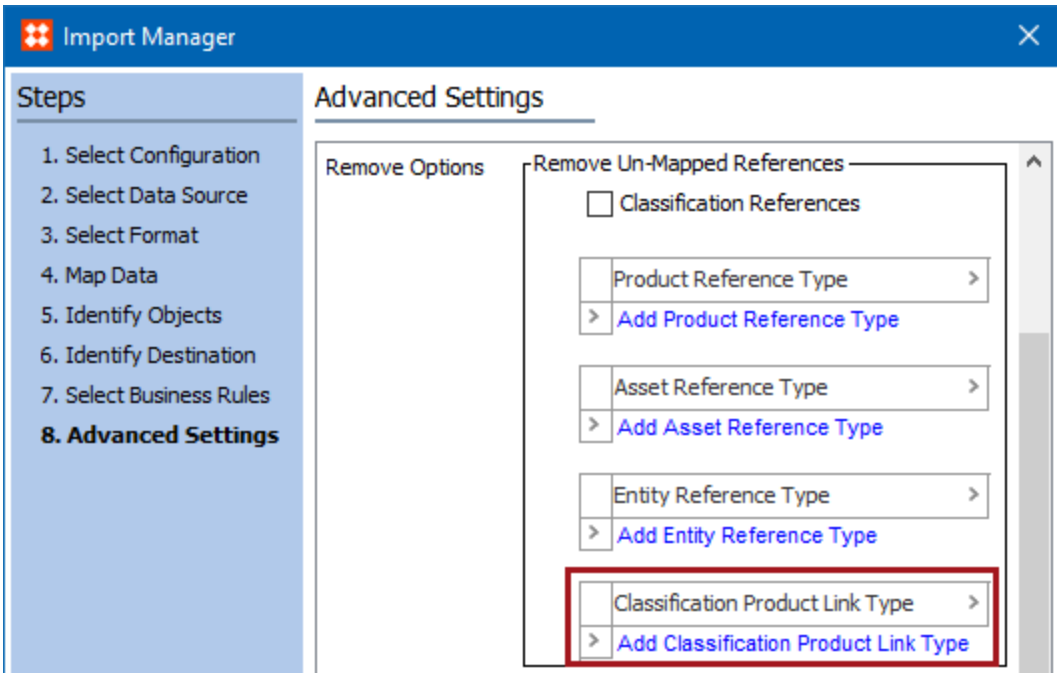


The multi-value classification references are displayed and ready to import.

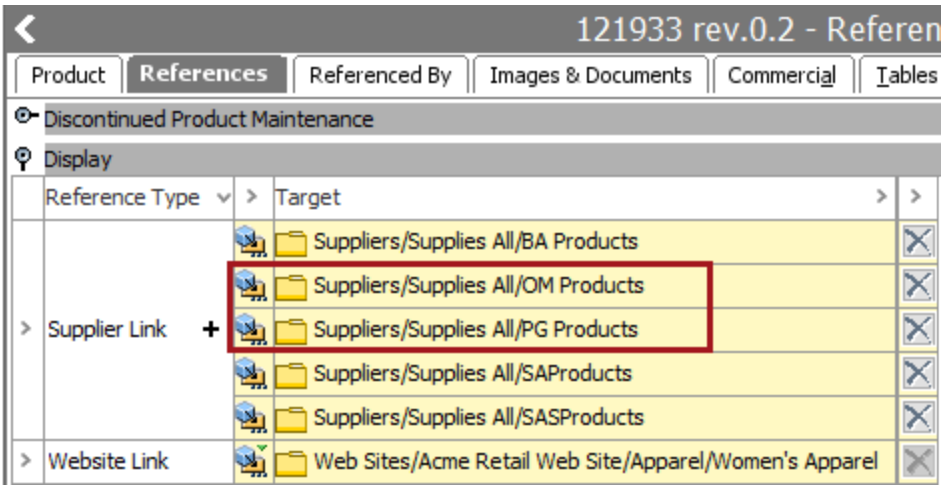


Complete the import wizard steps, noting that the selection on the Advanced tab determines how existing data in STEP is affected.

Remove Un-Mapped Classification References Not Set (Default)

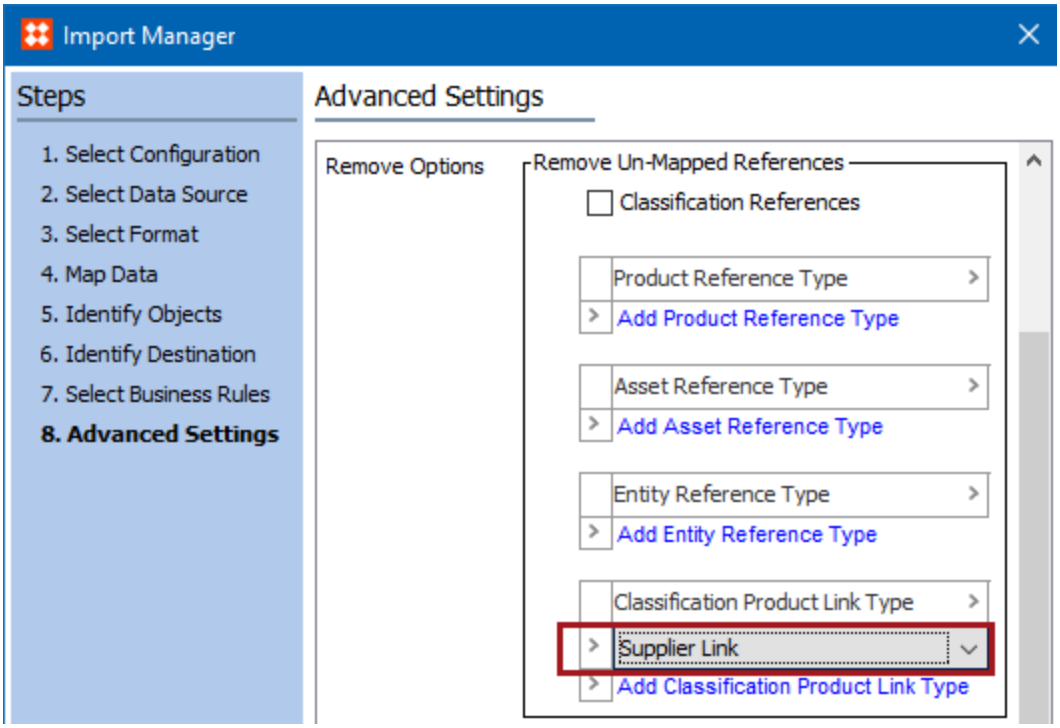


The imported classifications are added to the existing ones for the mapped product to classification link type.

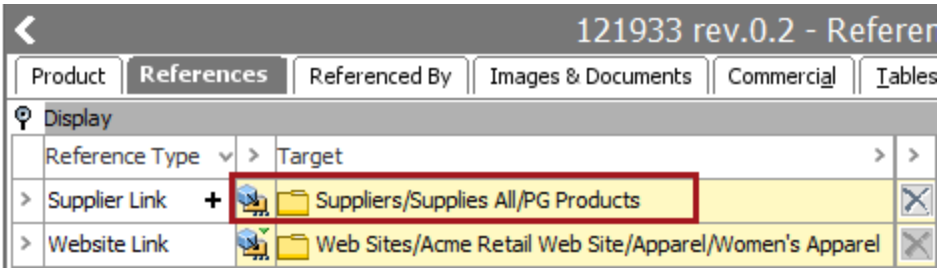


Setting the Remove Un-Mapped Classification References option indicates that after the import, STEP will include only the product to classification link types (based on ID) contained in the import file.

Remove Un-Mapped Classification References Set



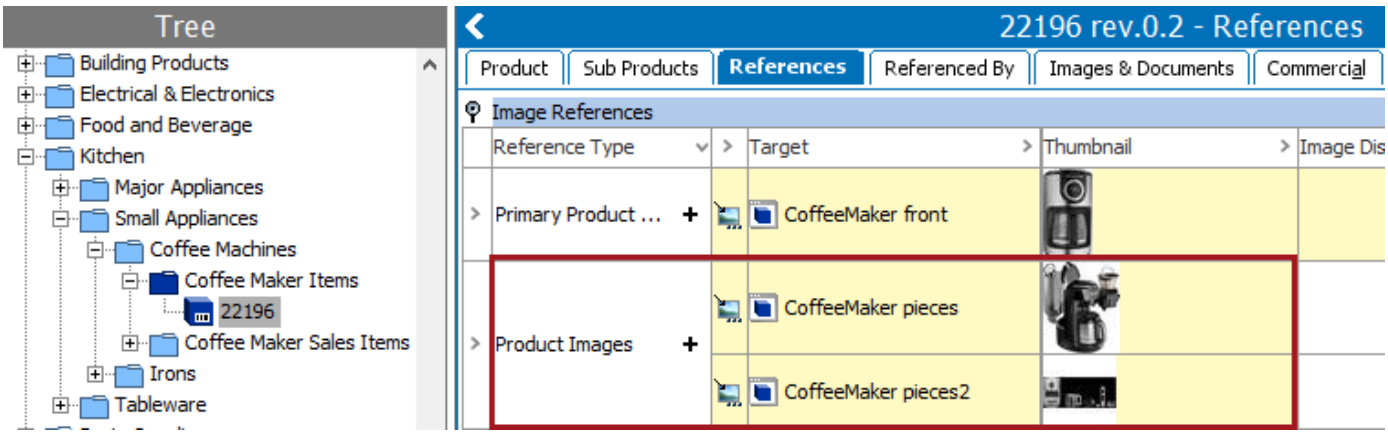
The imported classifications replace the existing ones for the mapped product to classification link type. Note that since each row in the Excel file is processed individually, only the final link for the object remains after the import is completed.



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

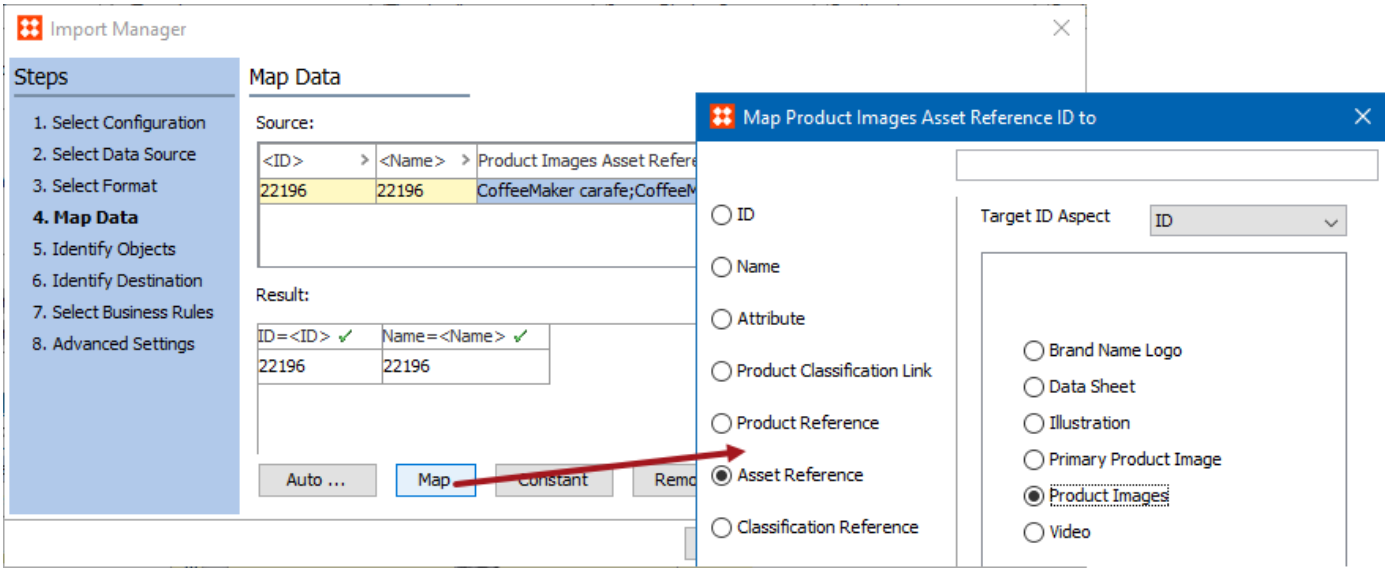
Consider the scenario shown below where in STEP a product already references two images (IDs CoffeeMaker pieces and CoffeeMaker pieces2) using the Product Images reference type. This reference type has the 'Allow multiple references' option enabled.



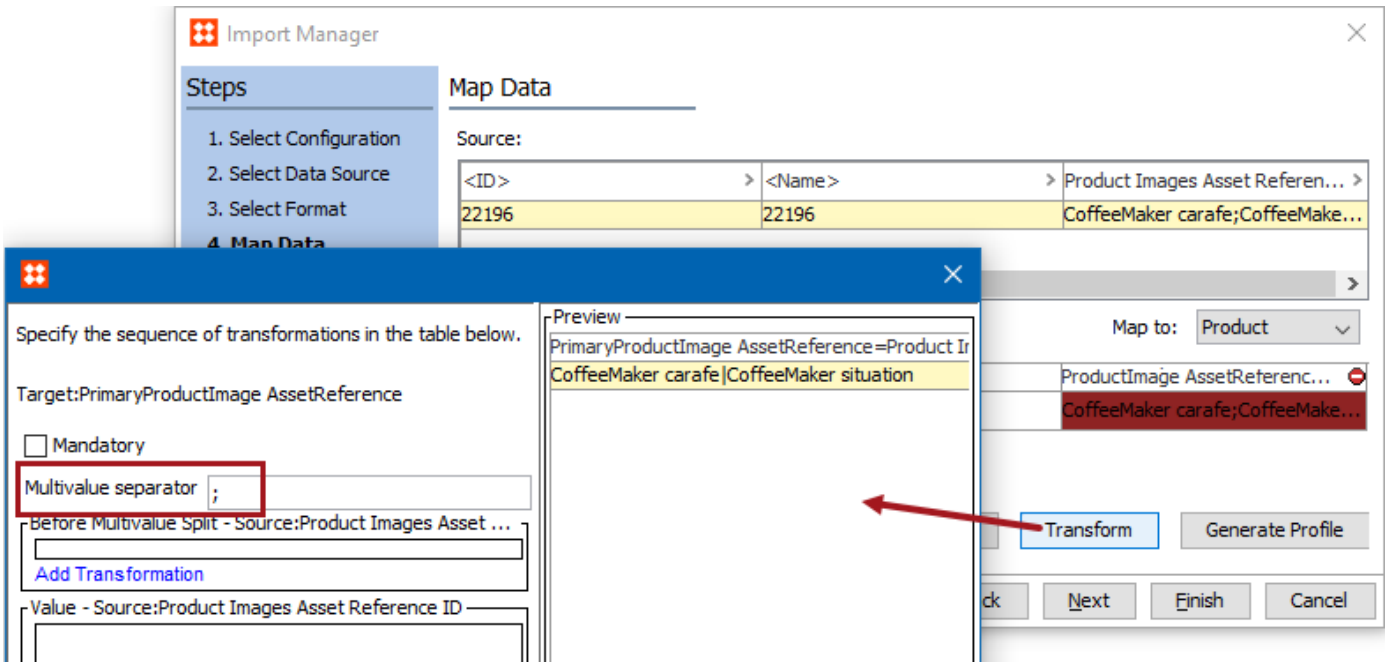
An Excel import file has been prepared and includes two different images (IDs CoffeeMaker carafe and CoffeeMaker situation) for the Product Images reference type.

	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.

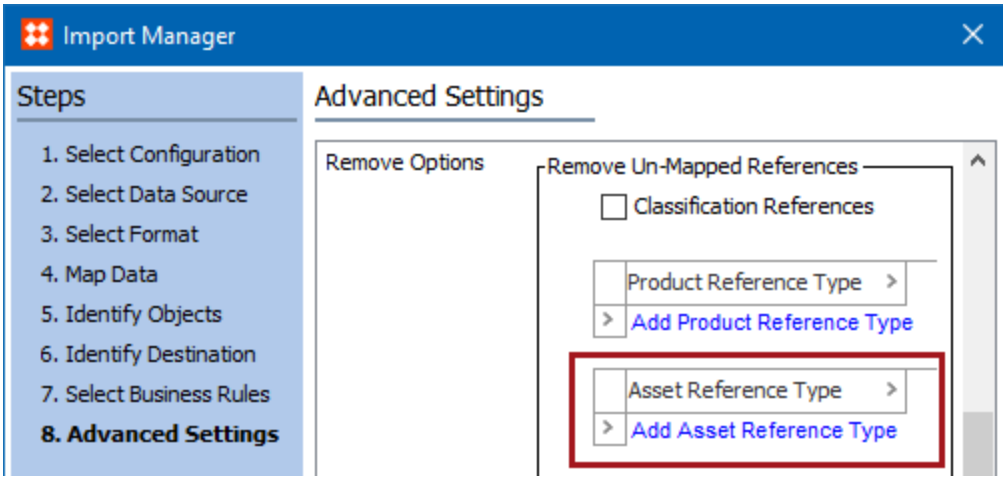


Finally, transform the multi-value references to an input format that STEP can read [replace each semicolon (;) with a pipe (|)], and verify that the Preview pane no longer shows a red background (which indicates an error).



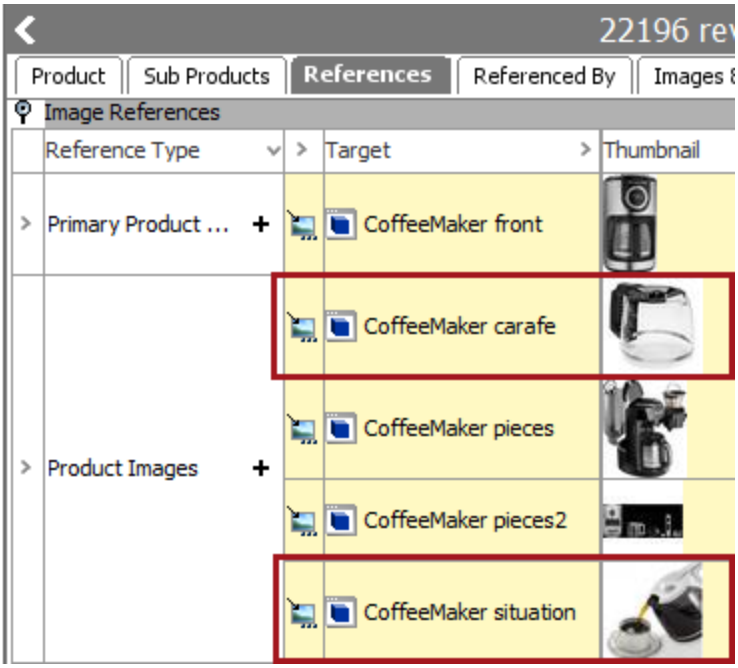
Complete the import wizard steps, noting that the selection on the Advanced tab determines how existing data in STEP is affected.

Remove Un-Mapped Asset References Not Set (Default)



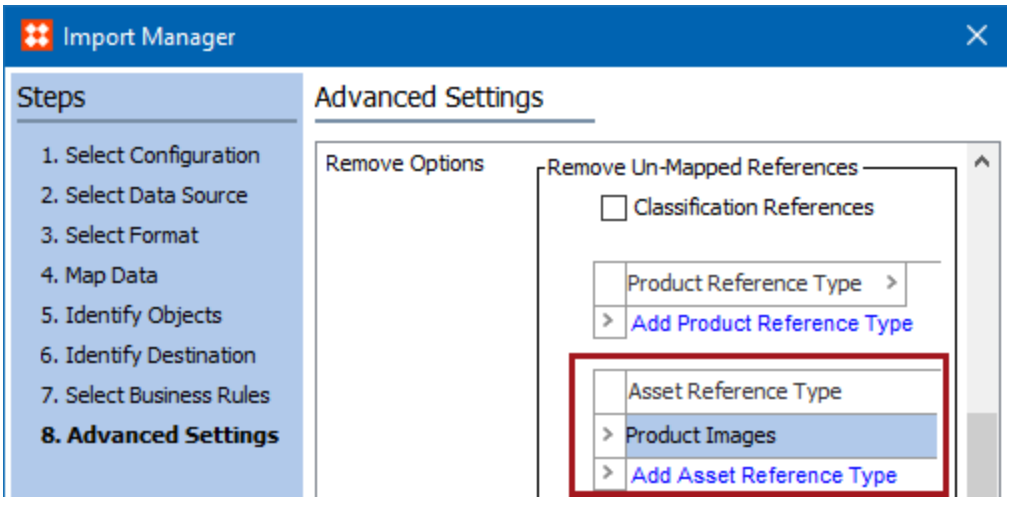
The imported images are added to the existing images for the selected reference type.

Remove Un-Mapped Asset References Not Set (Default)



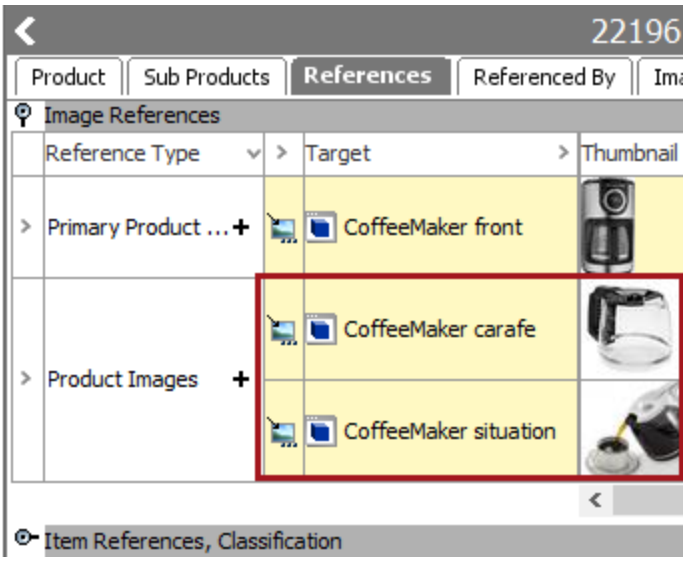
Setting the Remove Un-Mapped References options indicates that after the import, STEP will include only the references contained in the import file.

Remove Un-Mapped Asset References Set to Asset Reference Type



Remove Un-Mapped Asset References Set to Asset Reference Type

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

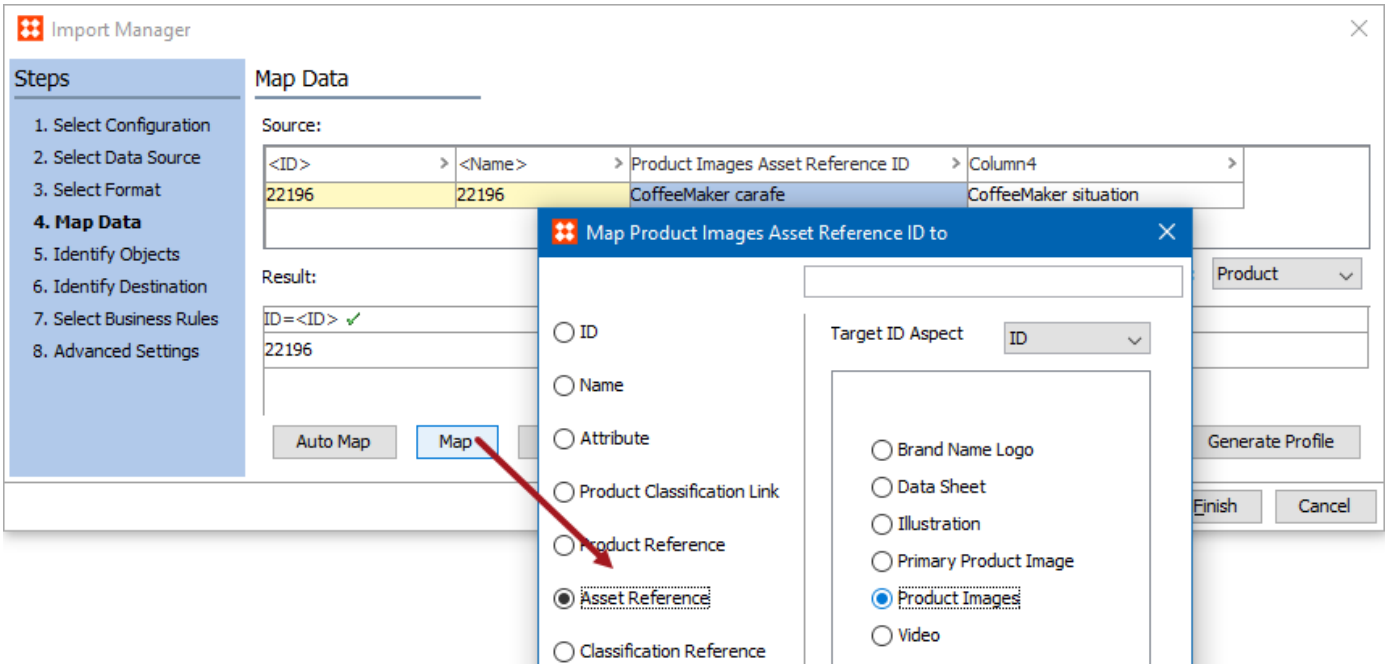


Object listed once, multiple references in individual cells

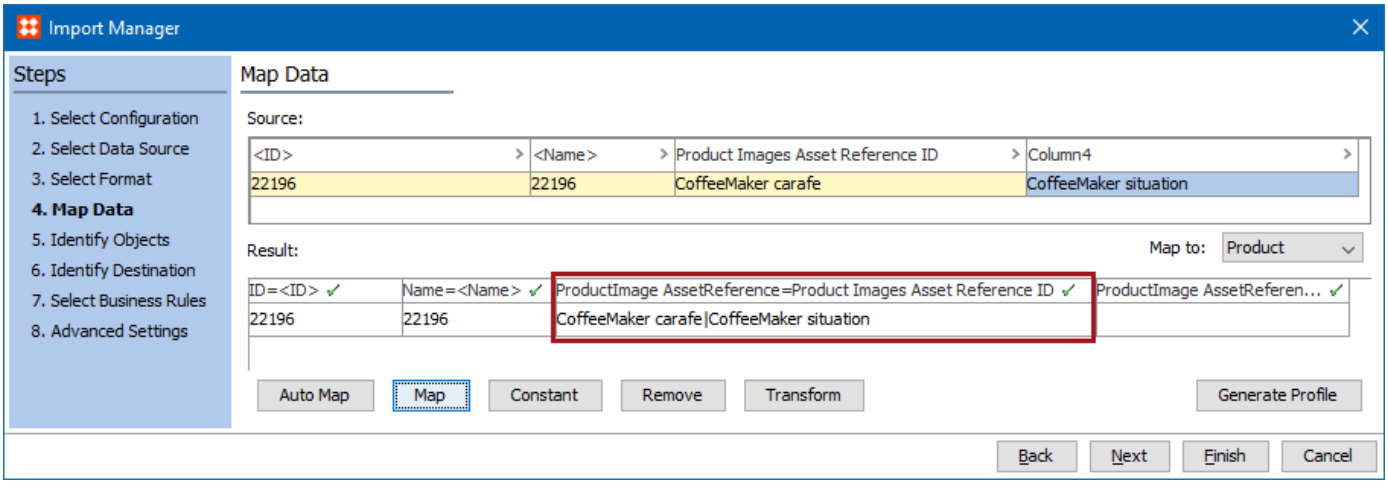
Consider the same scenario shown above where in STEP a product already references two images (IDs CoffeeMaker pieces and CoffeeMaker pieces2) using the Product Images reference type. This reference type has the 'Allow multiple references' option enabled. An Excel import file has been prepared and includes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

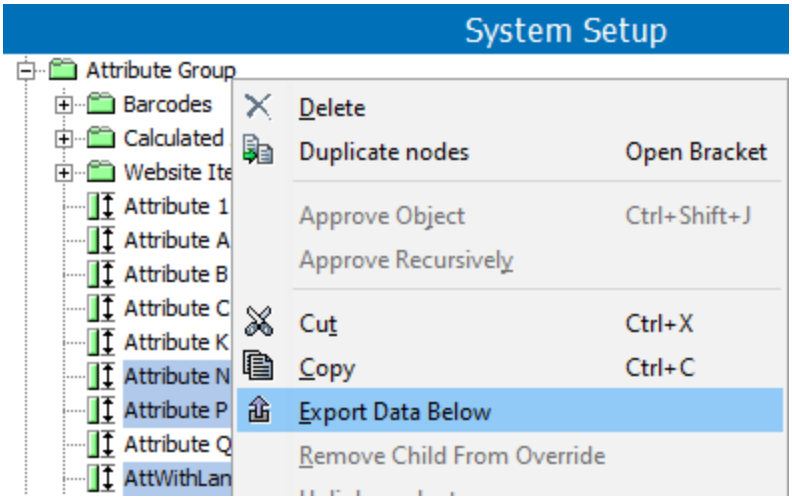
The easiest way to manage STEP attributes using Excel is to first export one or more attributes, then modify the Excel output, and finally import the modified file back into STEP. Use the configuration defined below to perform all necessary steps.

Configuration

1. In System Setup, select one or more attributes to export as follows:

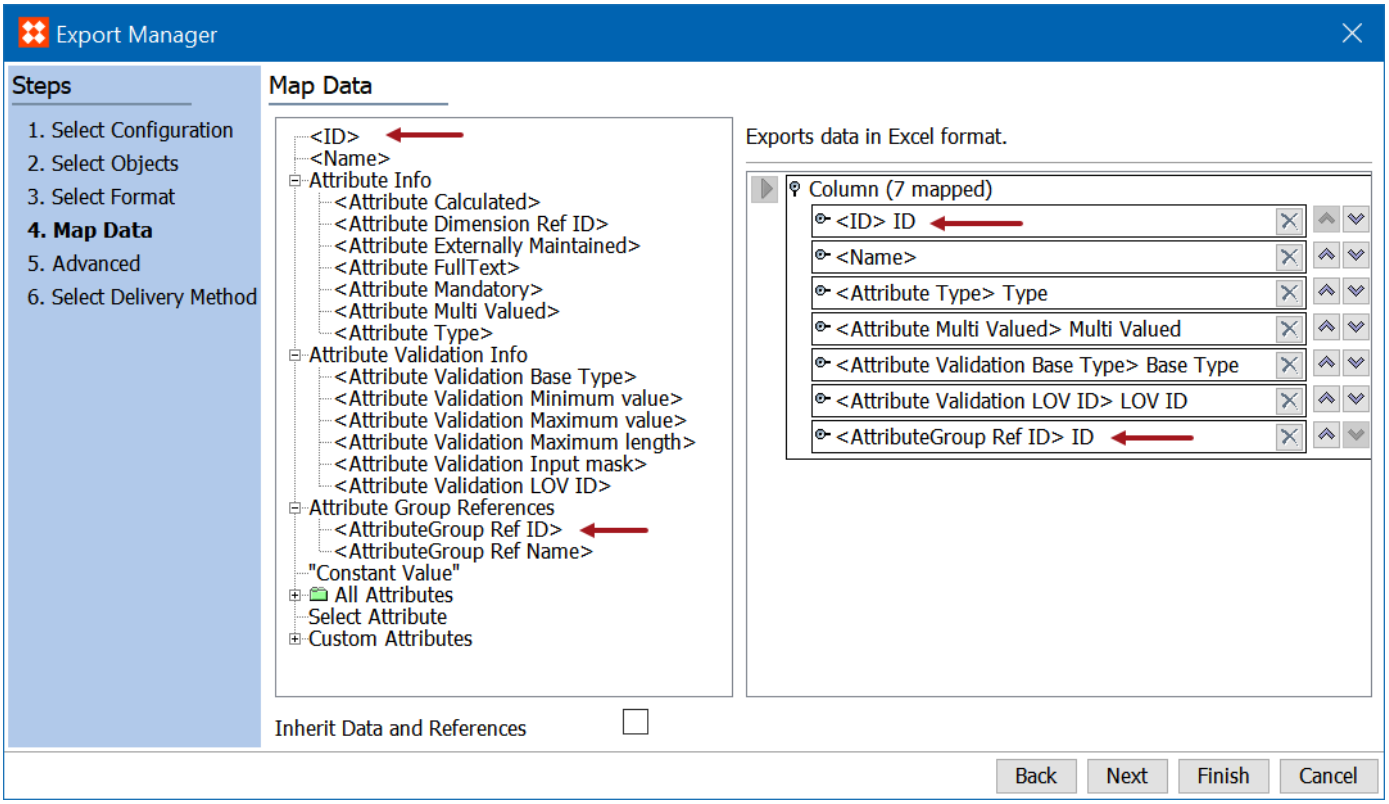
- To update existing attributes, select all attributes to be modified.
- To create new attributes, select one attribute of each Validation Base Type to represent the attributes that will be created. These attributes will create a template to be modified for the new attributes. Validation Base Types are defined in the Validation Base Type table within the **Validation Rules** topic in the **System Setup / Super User Guide**.

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.

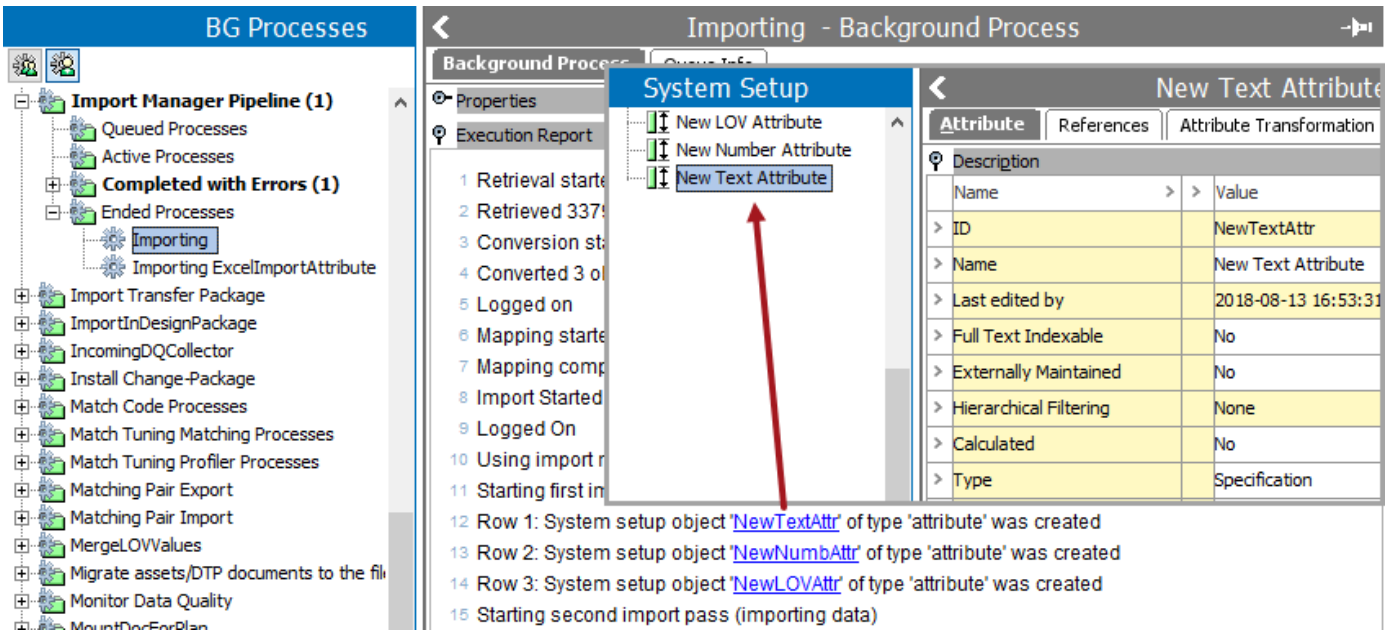
- On the Select Data Source step, choose your modified Excel file, as defined in the **Import Manager - Select Data Source** section.
- On the Map Data step, in the **Map to** parameter, select **Attribute** as the super type, as defined in the **Inbound Map Data - Map To** section.
- On the Map Data 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.


BG Processes
Import Manager Pipeline (1)
Queued Processes
Active Processes
Completed with Errors (1)
Ended Processes
Importing
Importing ExcelImportAttribute
Import Transfer Package
ImportInDesignPackage
IncomingDQCollector
Install Change-Package
Match Code Processes
Match Tuning Matching Processes
Match Tuning Profiler Processes
Matching Pair Export
Matching Pair Import
MergeLOVValues
Migrate assets/DTP documents to the fil
Monitor Data Quality
Importing - Background Process
Properties
Execution Report
System Setup
New LOV Attribute
New Number Attribute
New Text Attribute
New Text Attribute
Attribute
References
Attribute Transformation
Description
Name
Value
ID
NewTextAttr
Name
New Text Attribute
Last edited by
2018-08-13 16:53:31
Full Text Indexable
No
Externally Maintained
No
Hierarchical Filtering
None
Calculated
No
Type
Specification
Row 1: System setup object 'NewTextAttr' of type 'attribute' was created
Row 2: System setup object 'NewNumbAttr' of type 'attribute' was created
Row 3: System setup object 'NewLOVAttr' of type 'attribute' was created
Starting second import pass (importing data)

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 2003
- 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, see 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

Instructions for installing components can be found in the **SPOT Program** topic in the System Administration documentation.

Format Availability

Excel Custom Template is available for selection in:

- Export Manager - see **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, see **Data Mapping**.

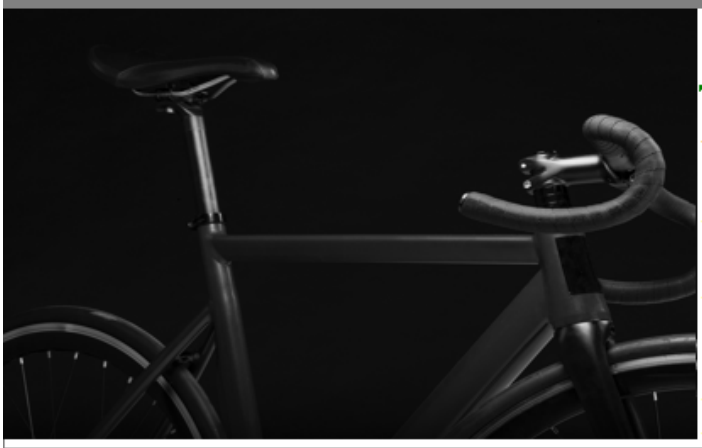
Output

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

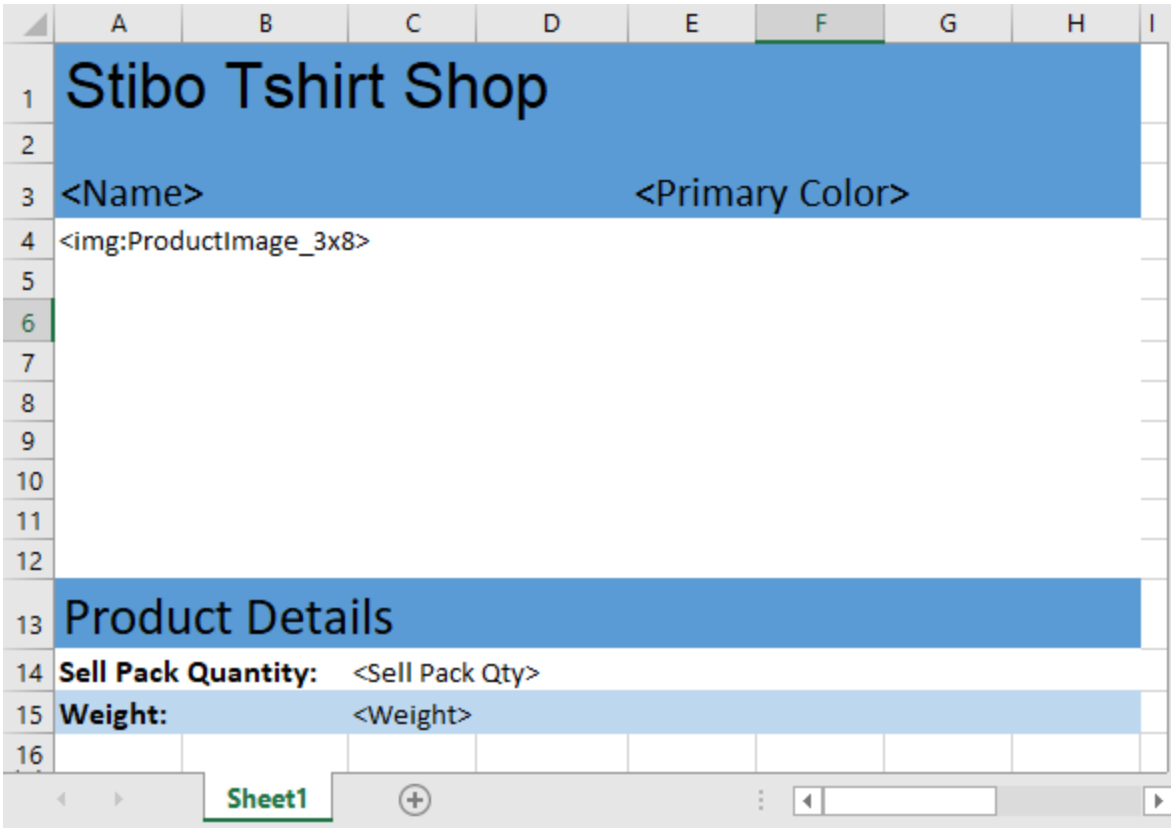
To configure an export of objects using the Excel Custom Template, follow these steps:

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.

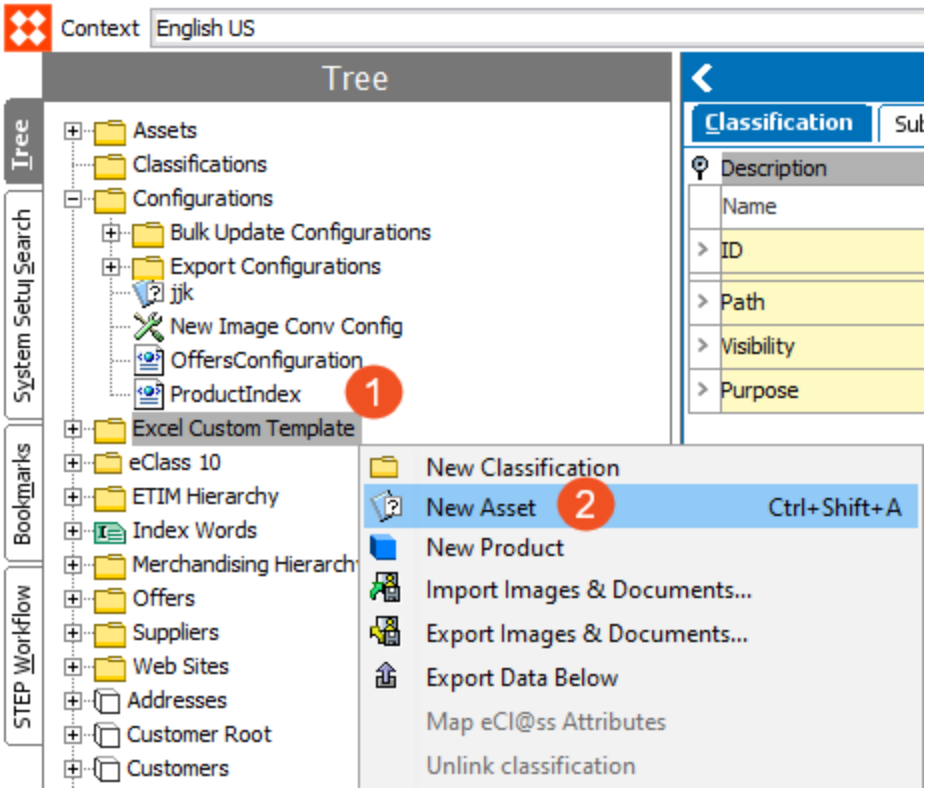
   STEP images may also be brought in, but the placeholder tag for images must be formatted differently.

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

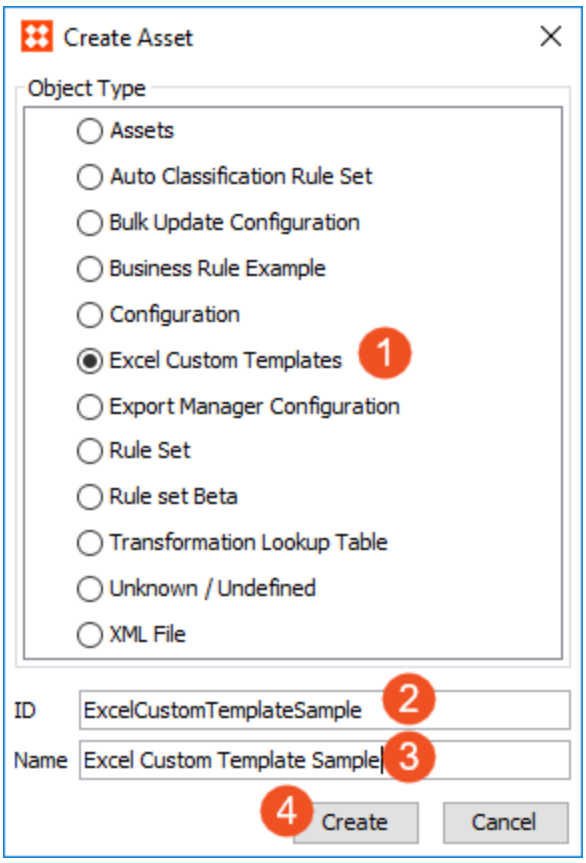
   Just as with attribute placeholder tags, image placeholder tags must begin and end with the greater than (<) and less than (>) symbols. Inside of these tags, the image placeholder tags must begin with 'img:.' Any text can be used following the colon, but recommended practice is to insert text relevant 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:



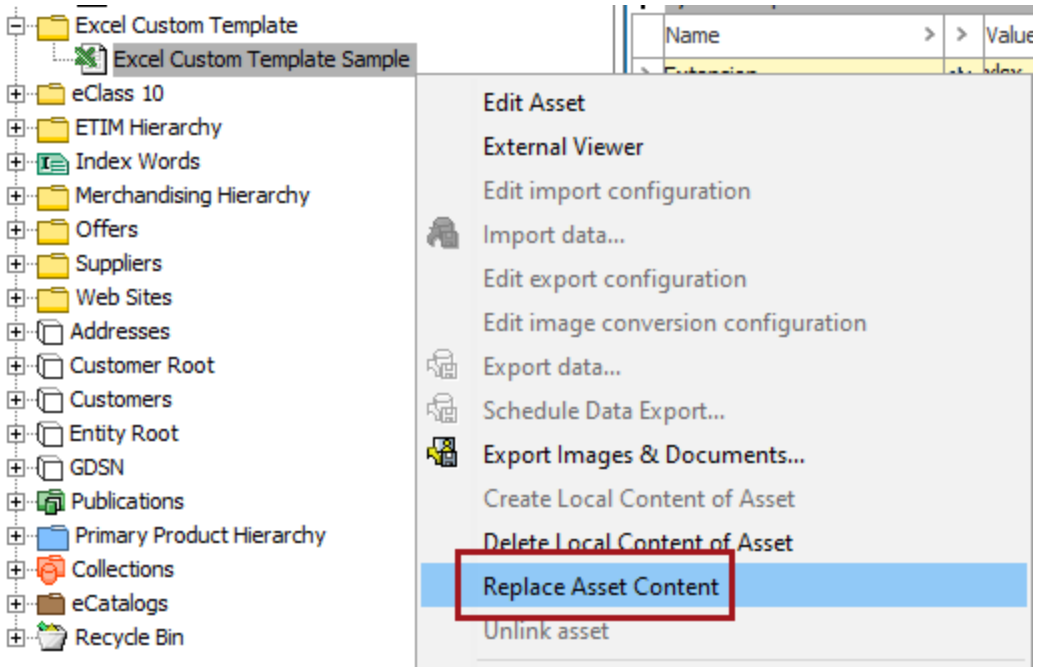
2. **Create a Custom Excel Template asset type** – In the System Setup tab in workbench, expand the Object Types & Structures node and select the 'Assets' node. Right-click 'Assets' and select 'New Object Type.' Assign an ID and name to a new object type ('Custom Excel Templates' would be appropriate), 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.'



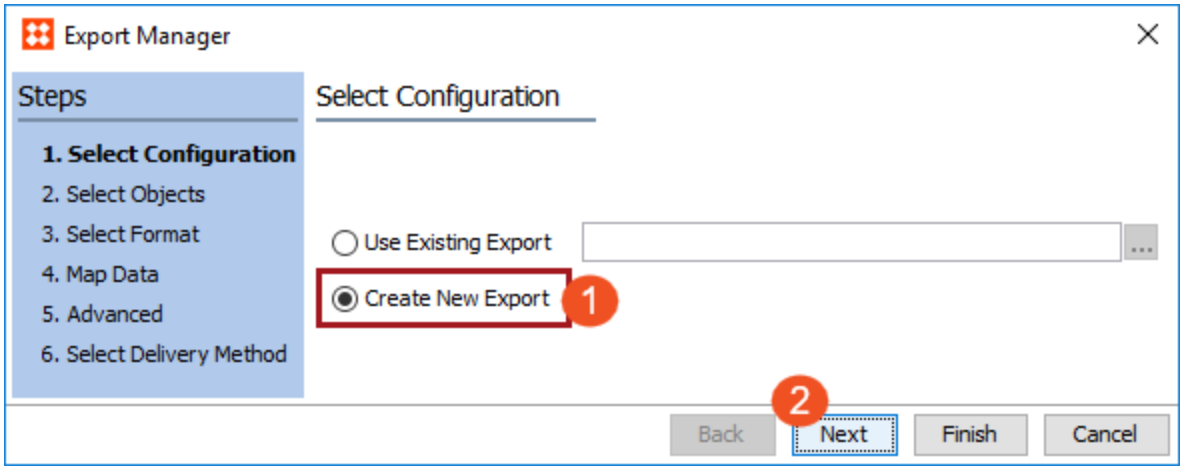
In the 'Create Asset' dialog, select the asset object type created in step two, in this case it is 'Excel Custom Templates,' and assign an ID and name to the template being saved.



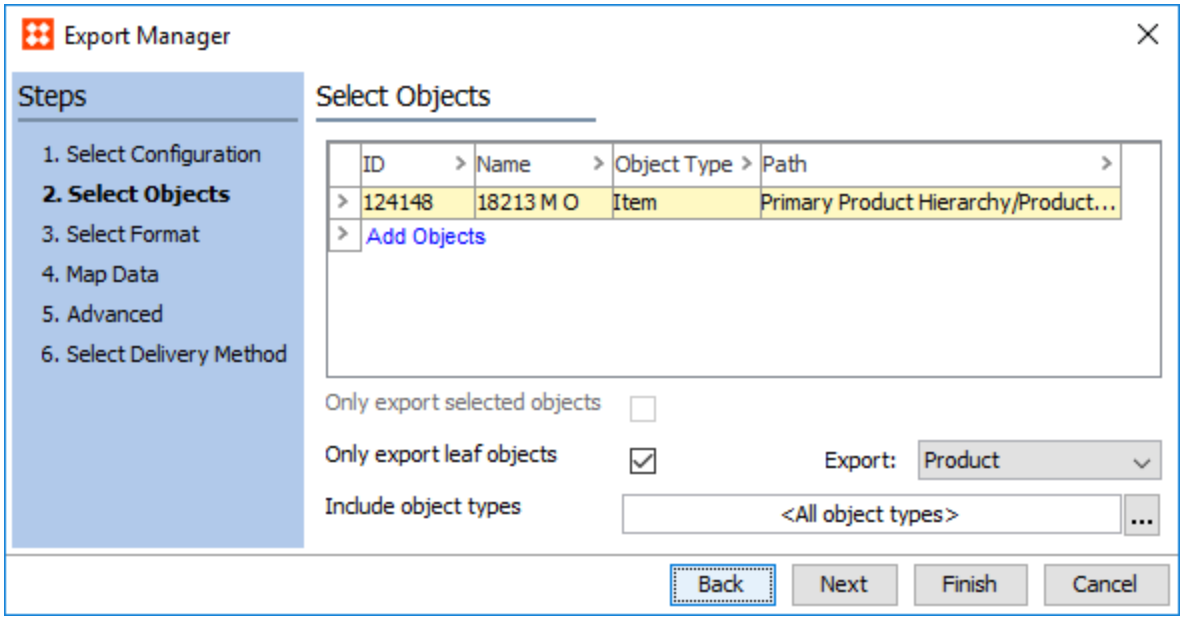
Be sure to also upload the template to the asset by right-clicking the asset and selecting 'Replace Asset Content.' STEP will prompt the user to select a file from the user's system to upload to the asset object.



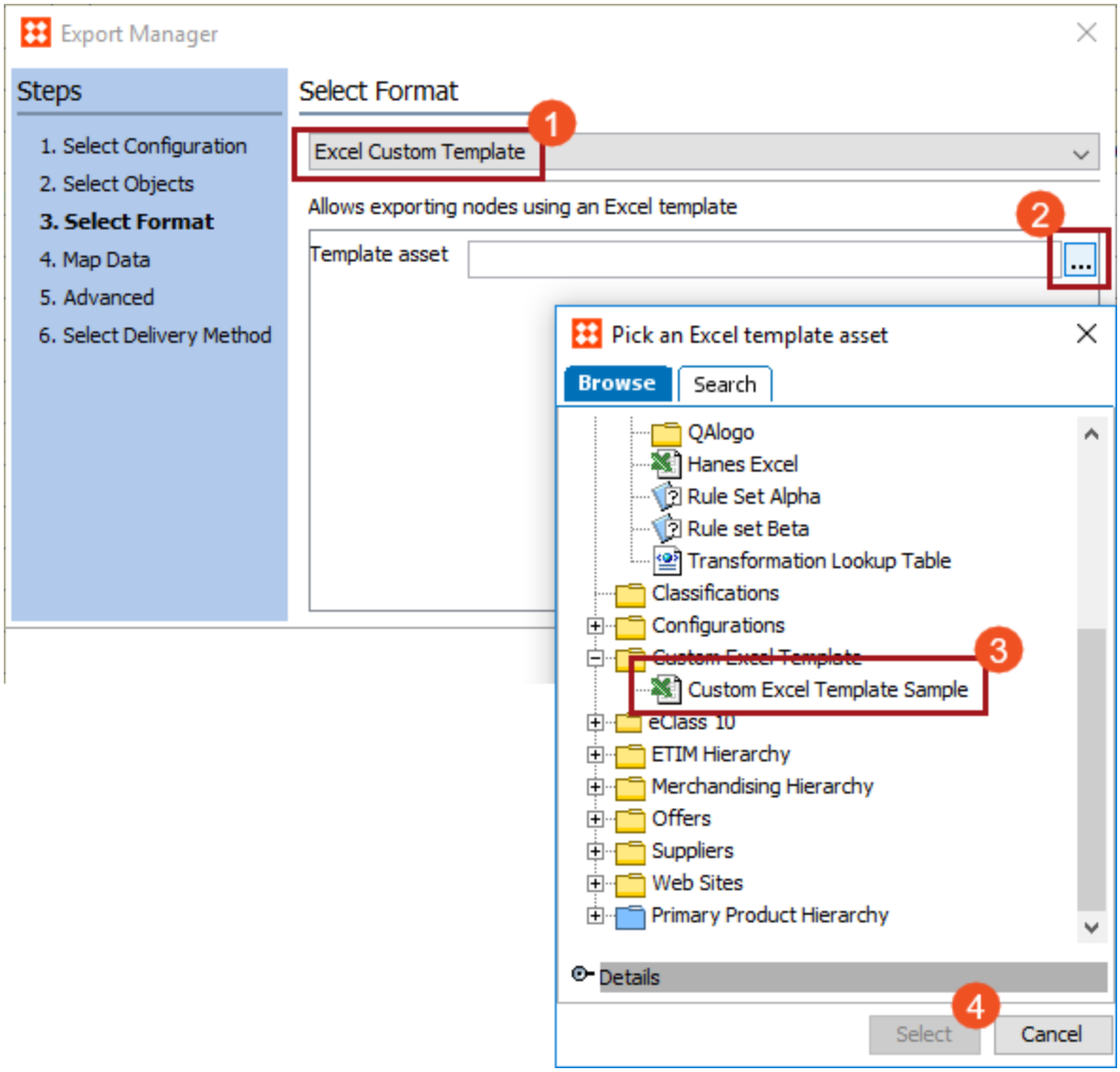
4. **Create an export configuration asset** – In order to retain the mappings that users establish in the Export Manager during configuration of their custom Excel export, it is useful to create and save an export configuration asset into which the export configuration can be saved. To create an export 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, see 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.

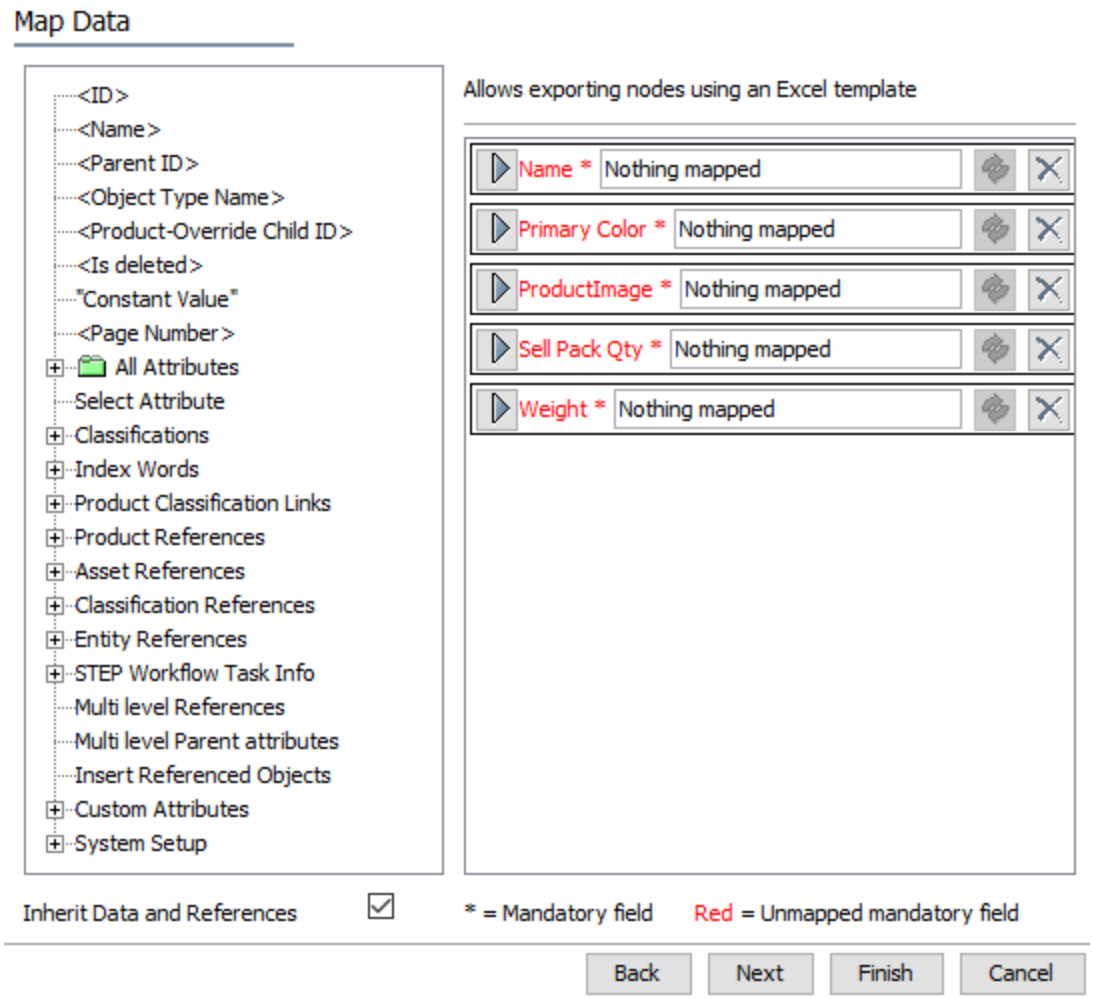


- **Select Format** – Select the 'Excel Custom Template' export format from the dropdown, and then, in the 'Pick an Excel template asset' dialog, select the Excel Custom template asset that was saved in step three.

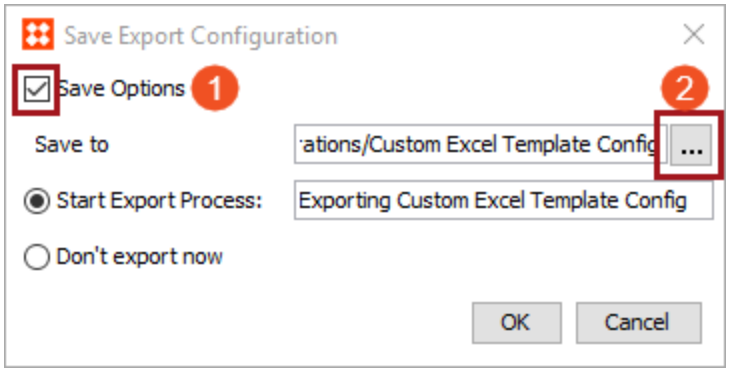


- **Map Data** – In the 'Map Data' screen, users will map the placeholder tags inserted into the custom Excel template to STEP attribute values. Listed in the right panel (mapping targets) are the placeholder tags that STEP read from the Excel template. In the left panel are the data sources. The placeholder 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.'

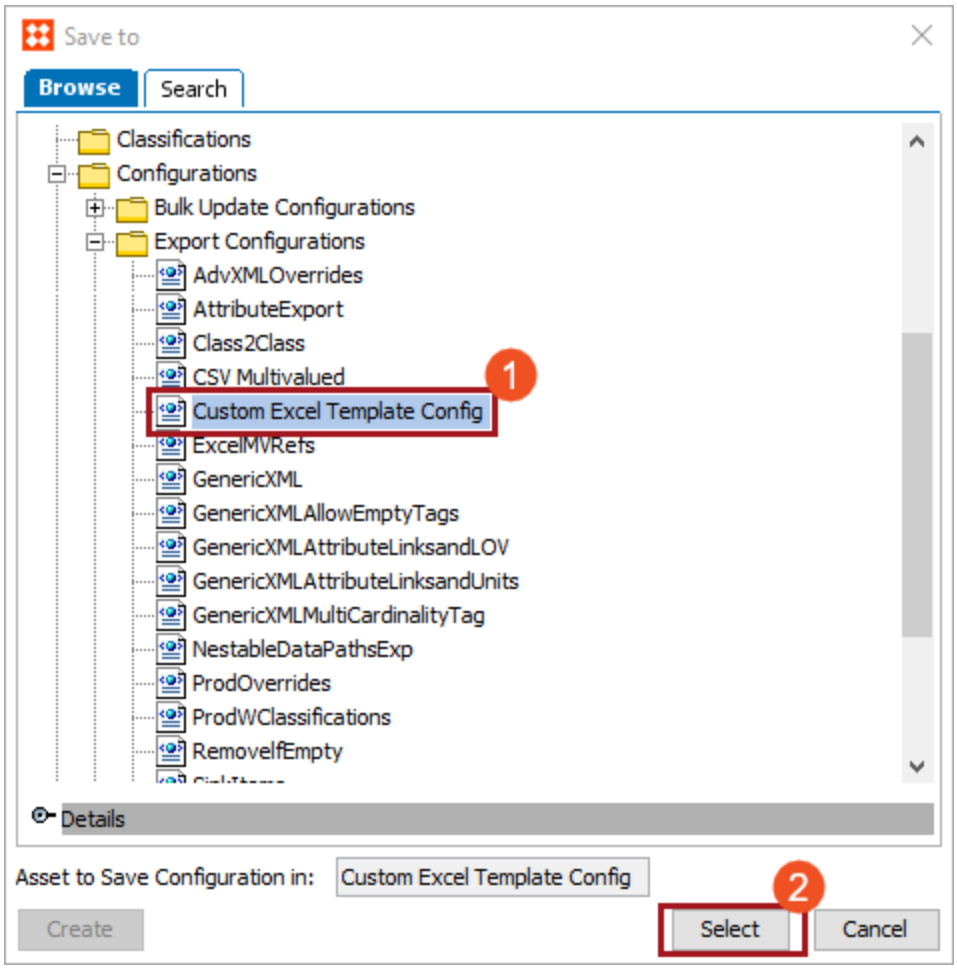
  For more information on mapping data, see the **Outbound Map Data - Data Source** topic.



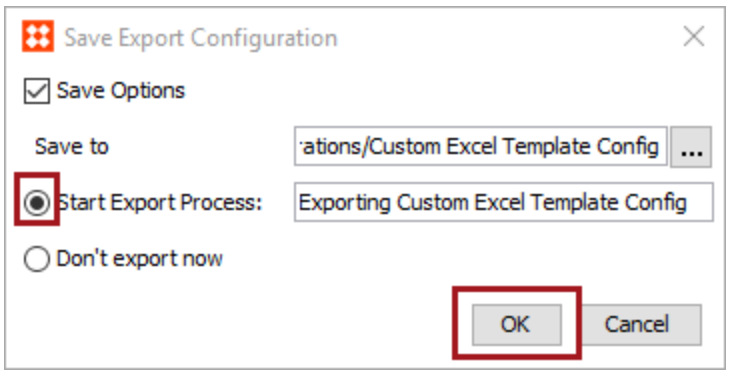
- **Advanced** – Make default selections, if applicable.
- **Select delivery method** – Select delivery method for the Excel document. For this example, 'File' has been chosen. Then click 'Finish.' Single-object exports will export to a single Excel file; multiple-object exports will export to a zip 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 ellipses 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.

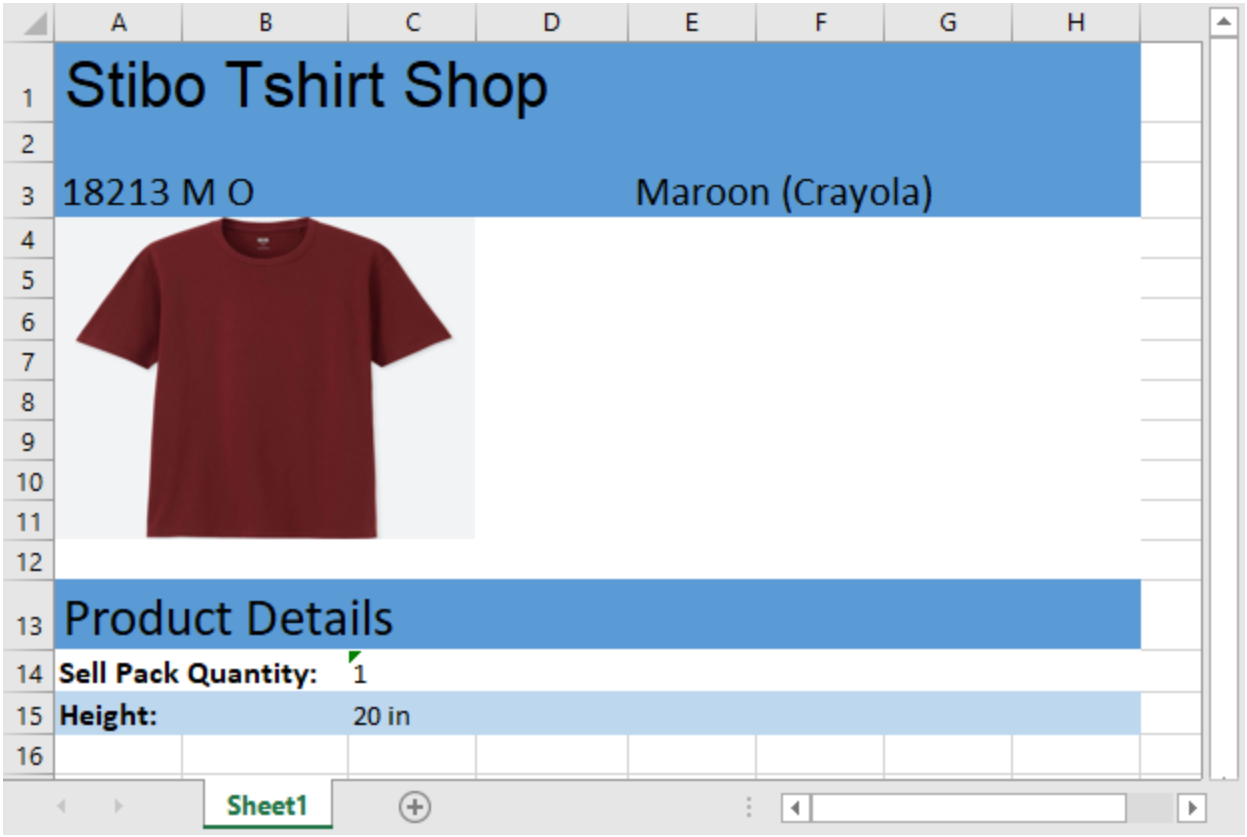
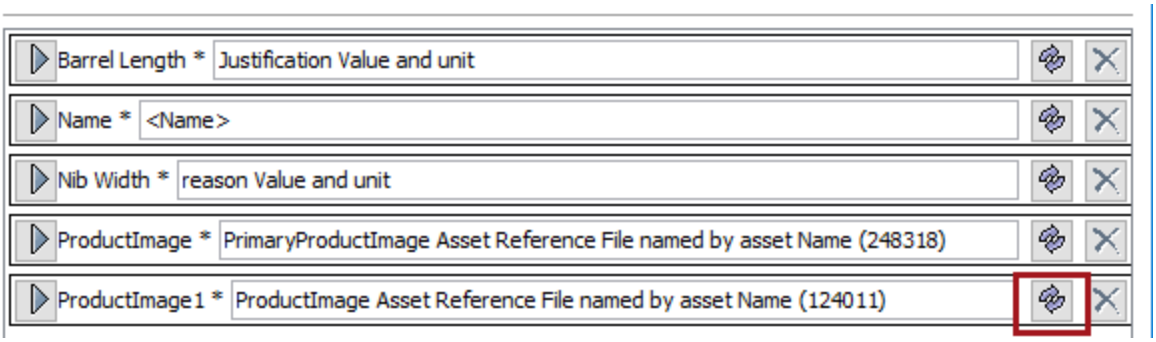
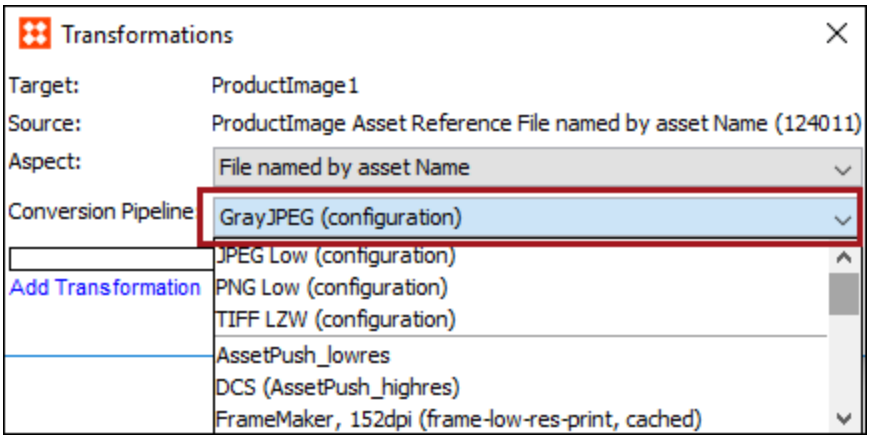


Image conversion configurations with Excel Custom Template

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, see 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 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 the Supplier 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 the Supplier Web UI. An import configuration should also be created and defined on the 'SmartSheetImportAction' in the Supplier Web UI (in case an import configuration is not specified in the export configuration).

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

Format Availability

Smartsheets are designed to be used from the Supplier Web UI **only** and using them outside of this context will diminish their capabilities. Furthermore, future Smartsheet enhancements will most likely also depend on Web UI.

Important: Objects should **not** be exported to Excel Smartsheets from STEP Workbench. Smartsheet templates exported from STEP 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 the Supplier 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, see 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.

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.
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 the Supplier 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.

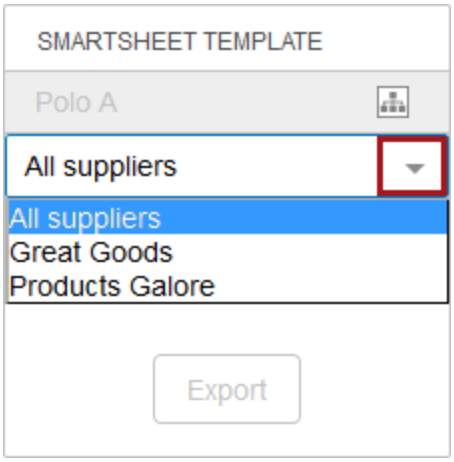
Product Initiation

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.



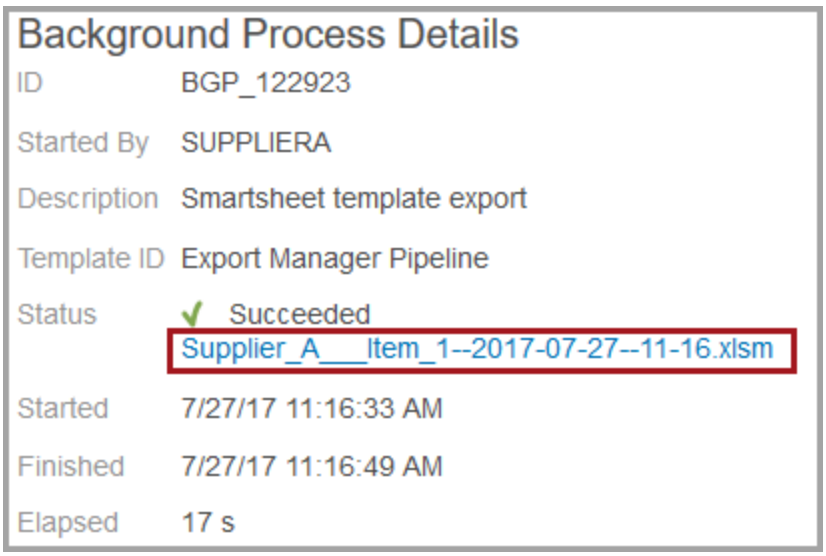
2. Select a valid node based on the object type(s) specified in the Smartsheet template and enable the OK button. An 'Invalid Selection' error is displayed and the OK button is not enabled when an invalid node is selected.

   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.

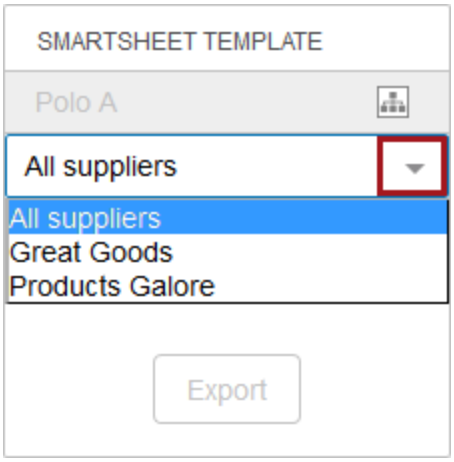
3. Click the **Export** button to start the background process and export the template.
4. On the status dialog, click the link to the background process.

   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.



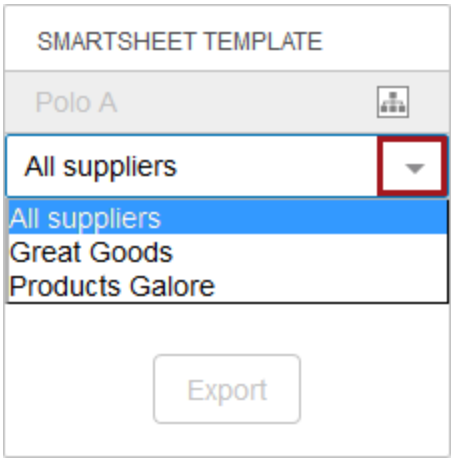
For more information on navigating and editing a Smartsheet, see the **Using a Smartsheet** section of the documentation.

For information about the color coding of the cells, see 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.

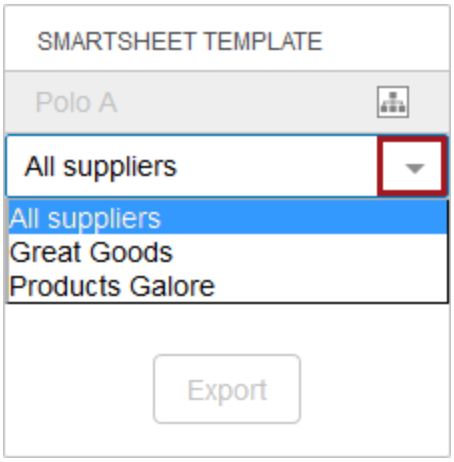
   For more information on validating a Smartsheet, see the **Using a Smartsheet** section of the documentation.

8. Once the sheet has been validated, save the document.
9. Navigate back to the Supplier 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.

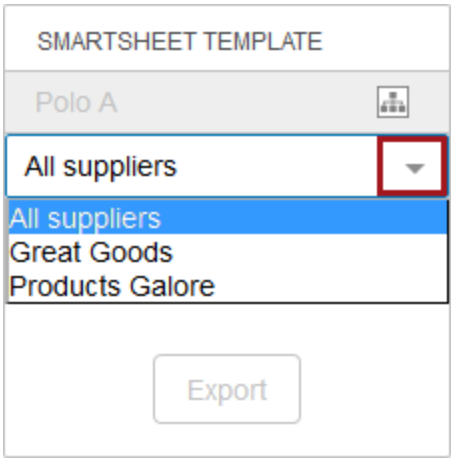
Exporting a Template with a Supplier Selector

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.

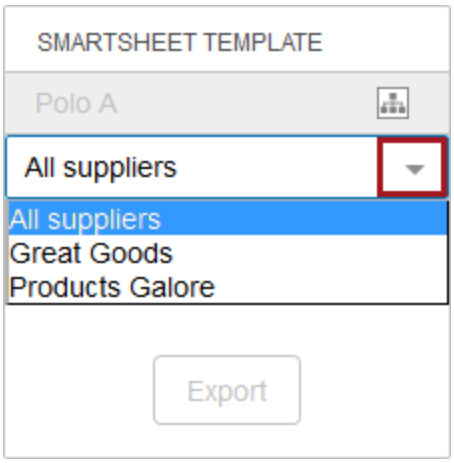
For information on how to configure a Smartsheet template with a supplier selector column, see the **Smartsheet Data and Template Configurations** section of the documentation.

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 

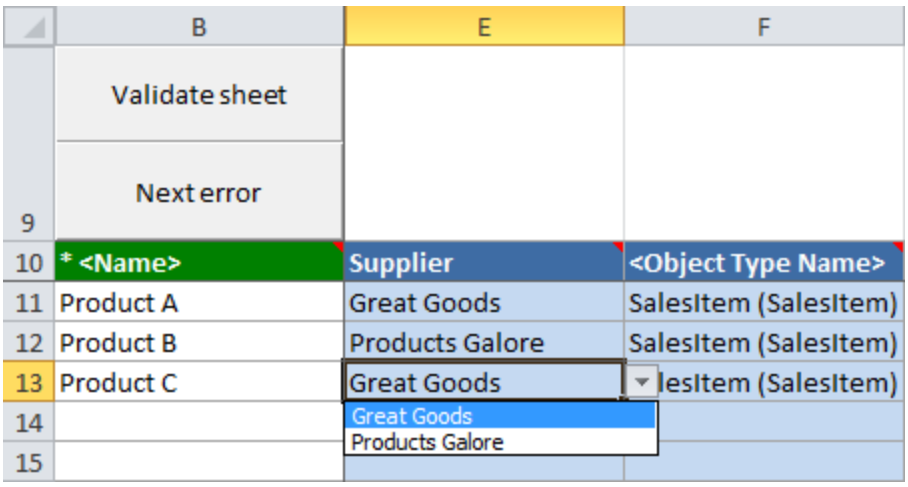
and select a valid node.
2. If the user is a member of multiple supplier groups the supplier dropdown will appear underneath the node picker. If the user is only a member of a single supplier user group, the supplier dropdown will not be present.



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

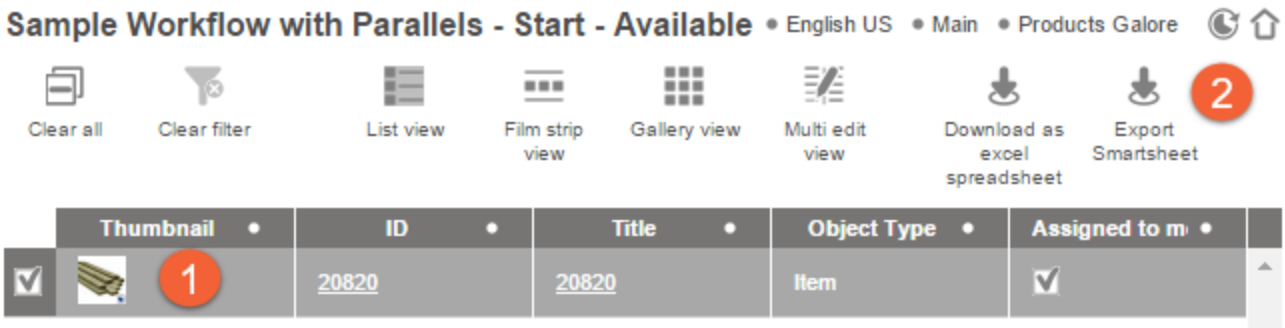
Several setup steps are required before Smartsheets can be used to maintain product data in STEP:

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.
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 the Supplier 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.

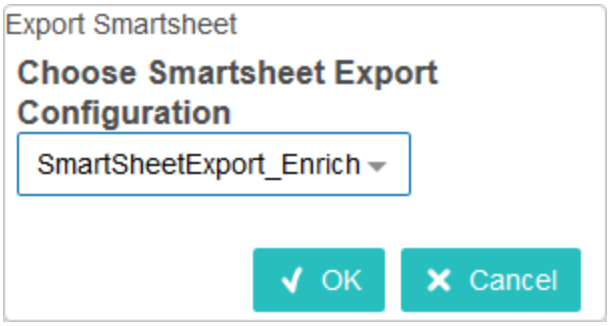
Product Maintenance

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:

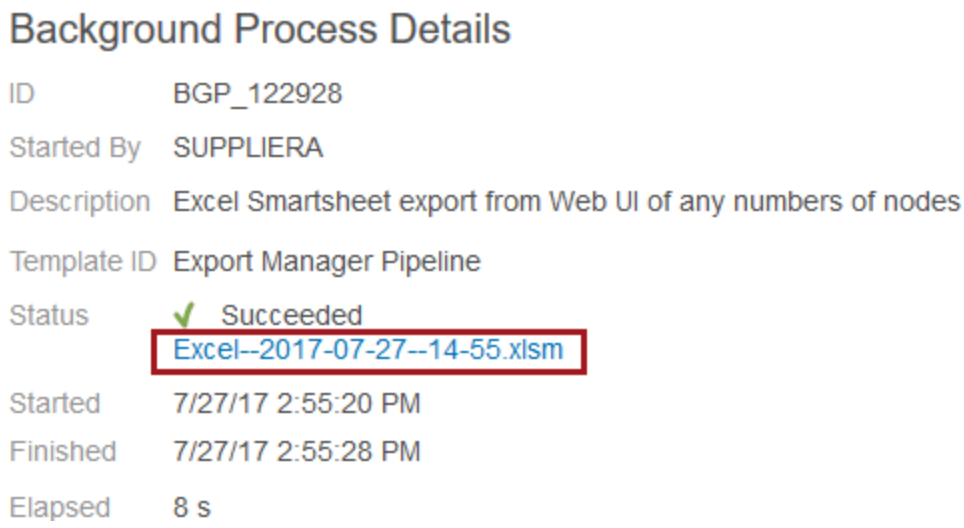
1. In the Supplier Web UI, navigate to the workflow widget and click the relevant workflow state. Then, on the Tasklist screen, individually select the products you wish to export, or use multi-select, or the Select All button.
2. Click the Smartsheet Export Action button. In the sample image, this button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



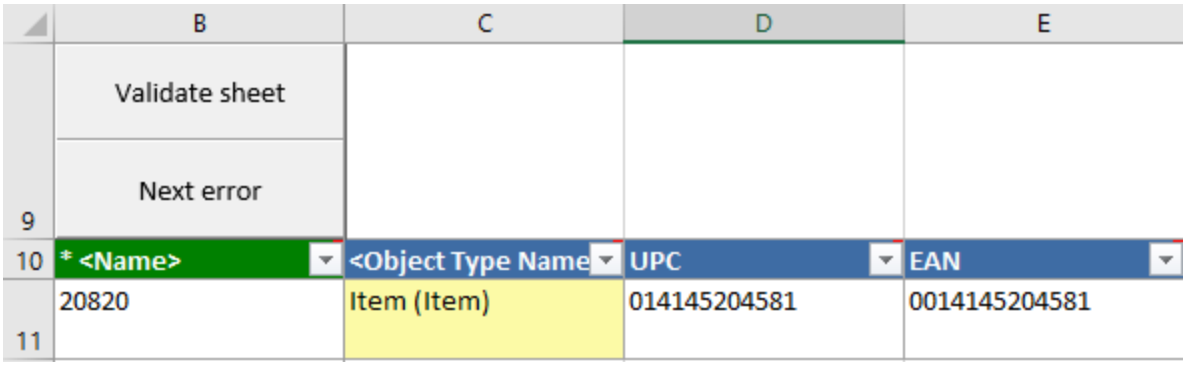
3. In the export dialog, select an export configuration from the dropdown and click **OK**.



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.

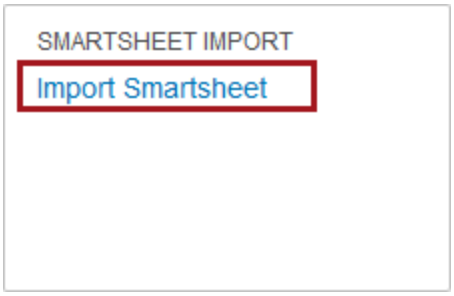


For more information on navigating and editing a Smartsheet, see the **Using a Smartsheet** section of the documentation.

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

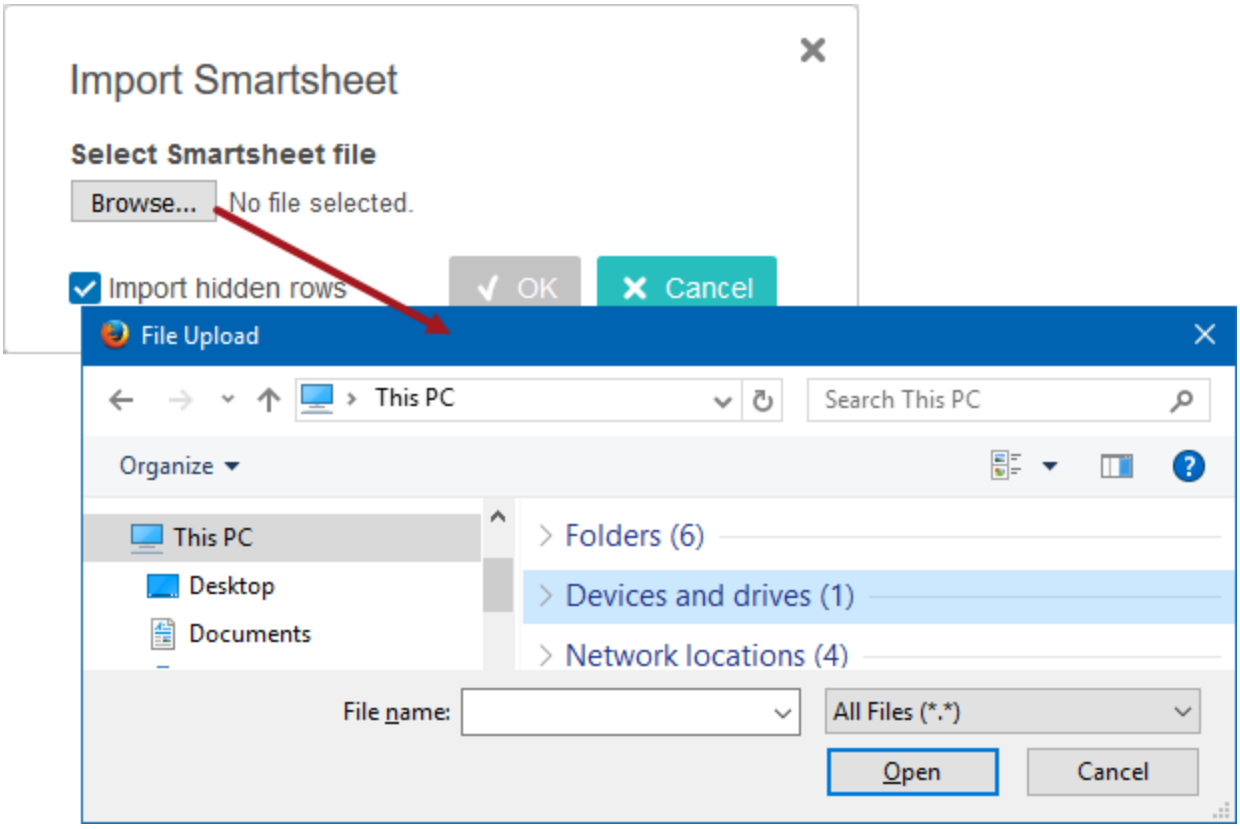
For more information on validating a Smartsheet, see the **Using a Smartsheet** section of the documentation.

7. Once the sheet has been validated, save the document.
8. Navigate back to the Supplier 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.

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, see 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. For Excel 97/2003, the maximum number of exported LOVs is 250. 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, see 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.

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

For detailed information about the Export Manager, see the **Export Manager** section of the documentation.

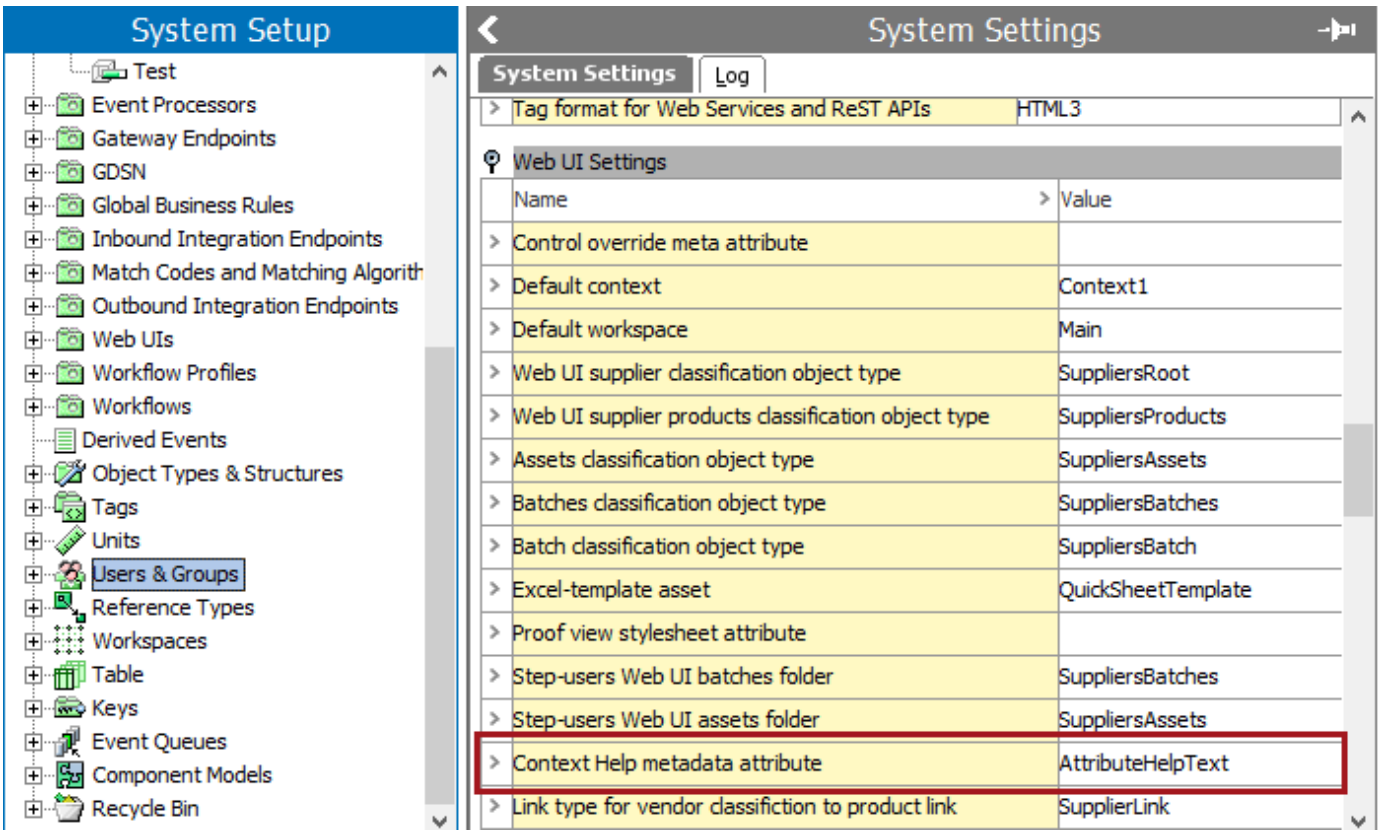
Important: Smartsheets are designed to be used from the Supplier Web UI **only** and using them out of this context will diminish their capabilities. Furthermore, future Smartsheet enhancements will most likely also depend on Web UI. Specifically, objects should **not** be exported to Excel Smartsheets from STEP Workbench. Smartsheet templates exported from STEP 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.

Export Manager Wizard

Select Objects Step

On this step, select the objects for export.

Note: 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.

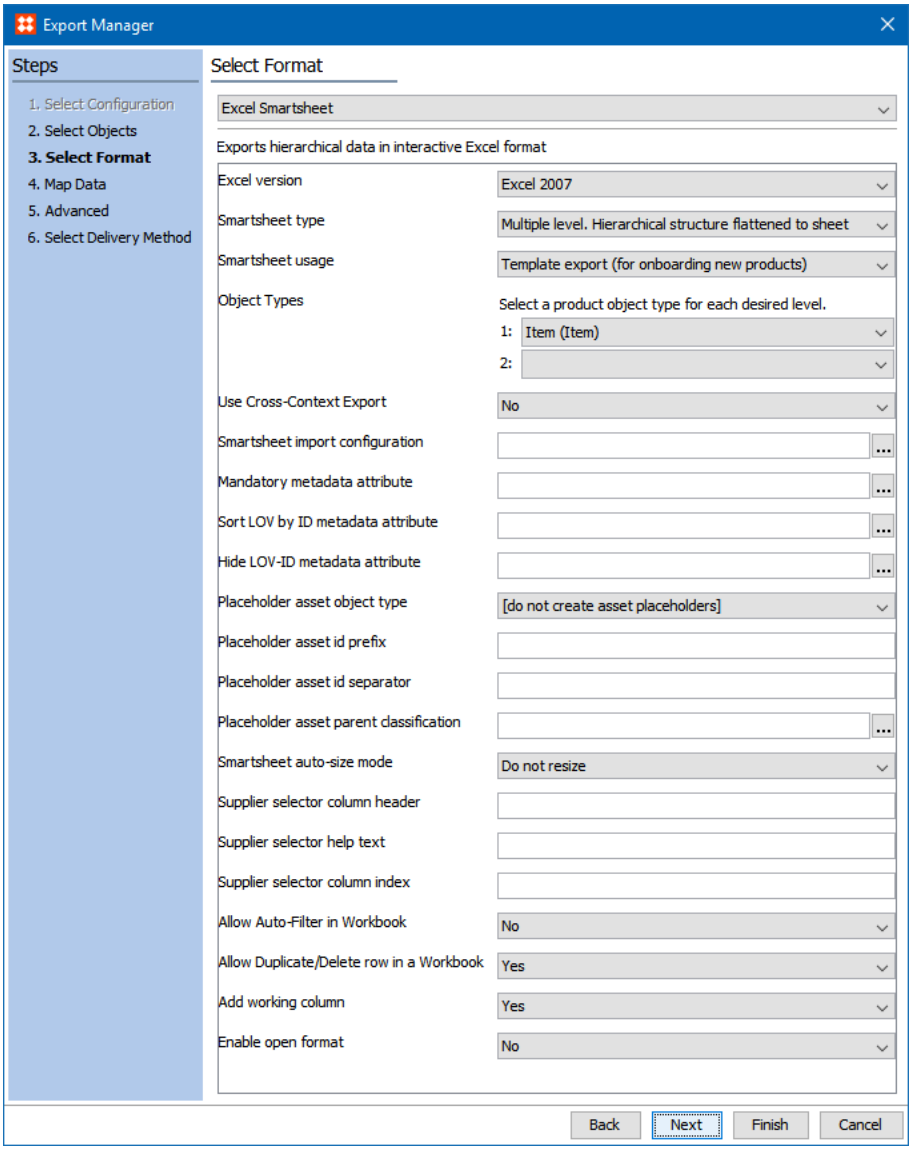


For both data export and template configurations, apply the following settings:

- 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.

Select Format Step

On this step, select **Excel Smartsheet** as the file export format.



1. In the **Excel version** parameter, use the dropdown to select Excel 2007. This parameter is mandatory. 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 **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.

   This parameter is mandatory.

For more information on Smartsheet types, see the **Using a Smartsheet** section of the documentation.

3. In the **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.

   This parameter is mandatory.

   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 **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.

   This parameter is mandatory.

5. In the **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 **Select Context** under the dropdown and choose the context(s) from the window that appears.

   This parameter is mandatory.

6. In the **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, see the **Smartsheet Import Configurations** section of the documentation.

7. In the **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 is optional.

For more information, see the **Mandatory Attributes** section of the **System Setup / Super User Guide** documentation.

8. In the **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.

   This parameter is optional.

   For more information, see the **Adding IDs to Values in LOV** section of the **System Setup / Super User Guide** documentation.

9. In the **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.

   This parameter is optional.

10. In the **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.

    This parameter is mandatory.

11. In the **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. This field is optional.

12. In the **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.

    This field is optional.

13. In the **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.

    This field is optional.

14. In the **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. This field is mandatory.

15. In the **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.

    This field is optional.

16. In the **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. This field is optional.

17. In the **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. This field is optional.

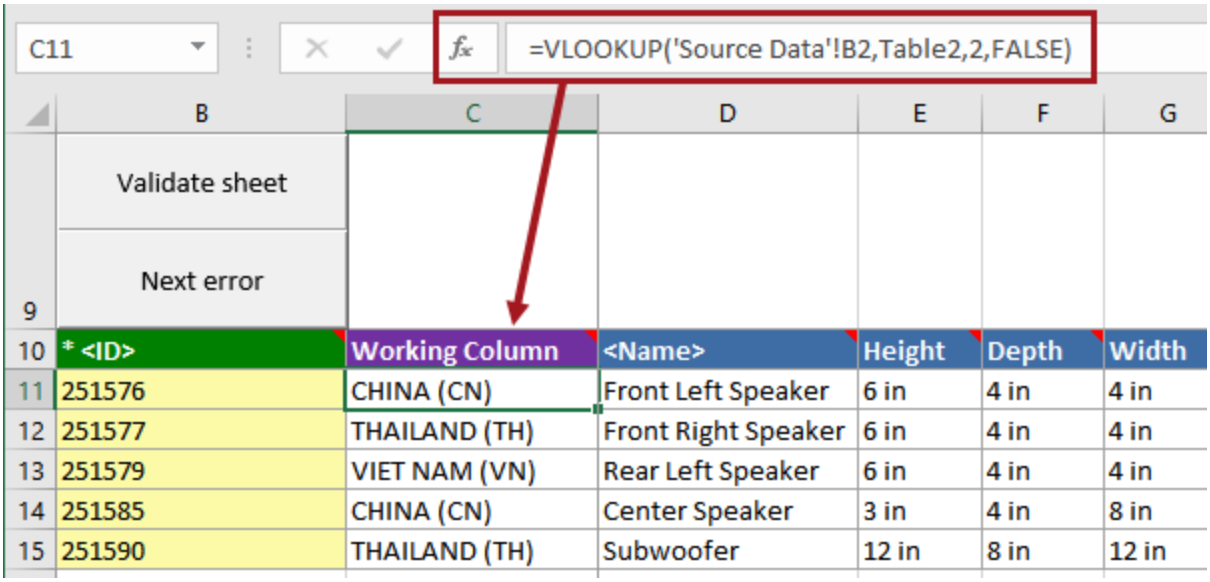
18. For the **Allow Auto-Filter in Workbook** parameter, determine whether or not rows can be filtered in the Smartsheet via the dropdown. This field is mandatory.

19. For the **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. This field is mandatory.

20. 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).

---

21. 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, see the **Open Format Spreadsheet** topic in this guide.

22. Click **Next** to move on to the next screen.

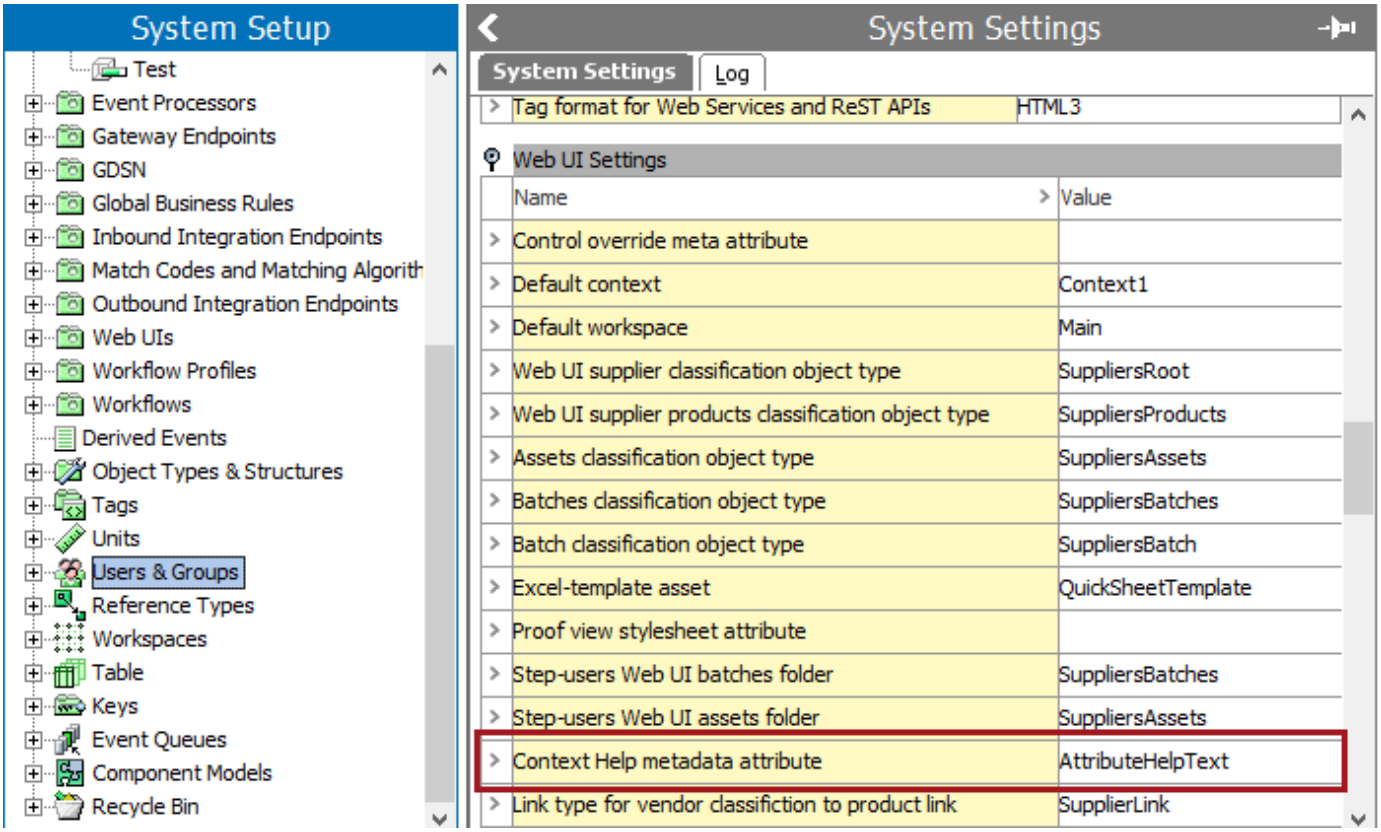
Map Data Step

Different 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.

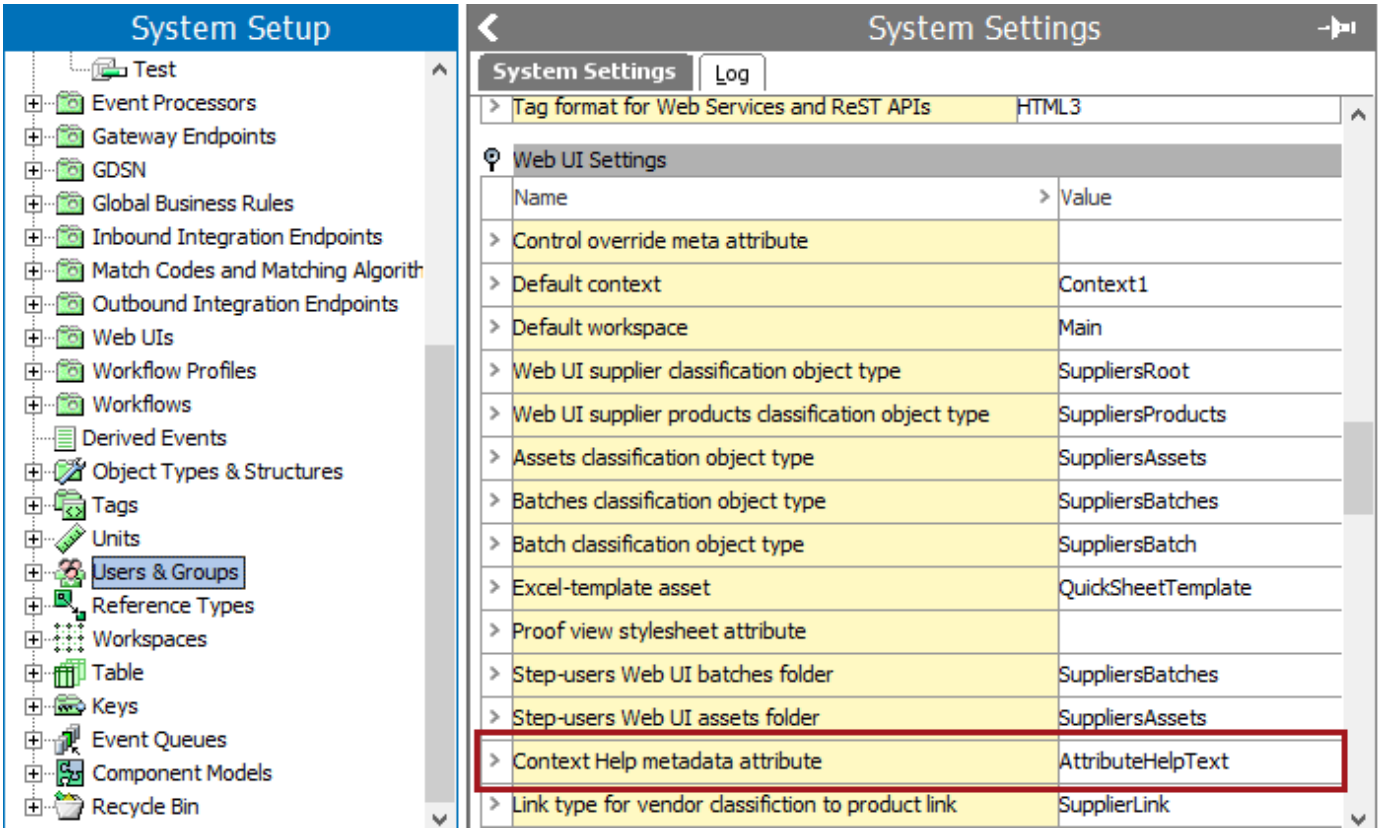
- **Mandatory** makes the column appear mandatory in the downloaded Smartsheet. If the Smartsheet is validated in Excel, cells that are empty but marked as mandatory will appear as invalid. However, no errors are generated when the Smartsheet is imported. To mark a column as mandatory, click the transformation button ( ) and enter any value in the window that appears.

For more information, see the **Mandatory Attributes** section of the **System Setup / Super User Guide** 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.

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 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 Step

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

Saving and Modifying the Configuration

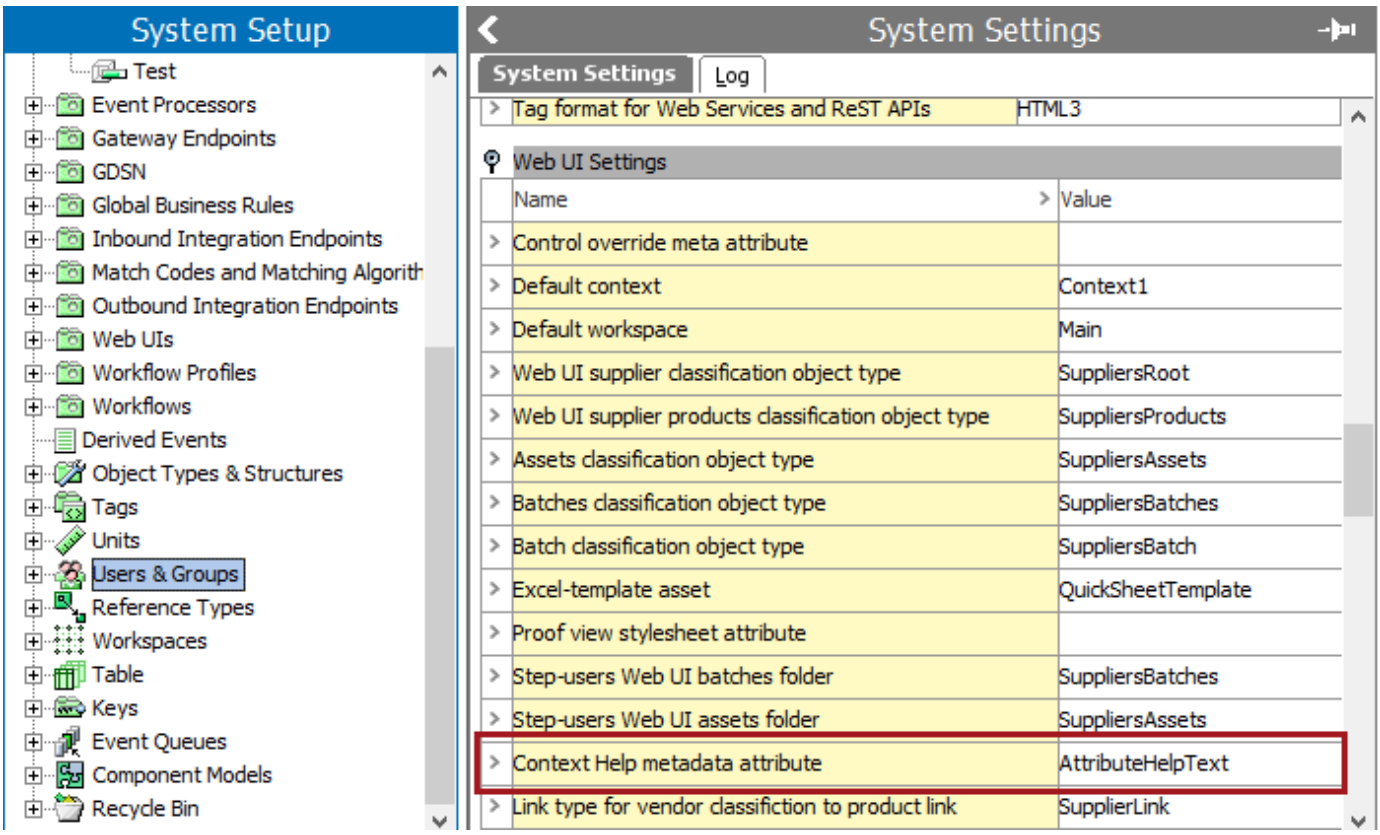
Saving an export configuration is described in the **Running a Data Export** topic of the **Export Manager** documentation.

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

Additional Configuration Options

Smartsheet column headers are capable of displaying tooltips when mousing over them. The text provided in the tooltips 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.


System Setup
Test
Event Processors
Gateway Endpoints
GDSN
Global Business Rules
Inbound Integration Endpoints
Match Codes and Matching Algorith
Outbound Integration Endpoints
Web UIs
Workflow Profiles
Workflows
Derived Events
Object Types & Structures
Tags
Units
Users & Groups
Reference Types
Workspaces
Table
Keys
Event Queues
Component Models
Recycle Bin
System Settings
System Settings
Log
Tag format for Web Services and ReST APIs
HTML3
Web UI Settings
Name
Value
Control override meta attribute
Default context
Context1
Default workspace
Main
Web UI supplier classification object type
SuppliersRoot
Web UI supplier products classification object type
SuppliersProducts
Assets classification object type
SuppliersAssets
Batches classification object type
SuppliersBatches
Batch classification object type
SuppliersBatch
Excel-template asset
QuickSheetTemplate
Proof view stylesheet attribute
Step-users Web UI batches folder
SuppliersBatches
Step-users Web UI assets folder
SuppliersAssets
Context Help metadata attribute
AttributeHelpText
Link type for vendor classifiction to product link
SupplierLink

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, see 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, see the **Smartsheet Data and Template Configurations** section of the documentation.

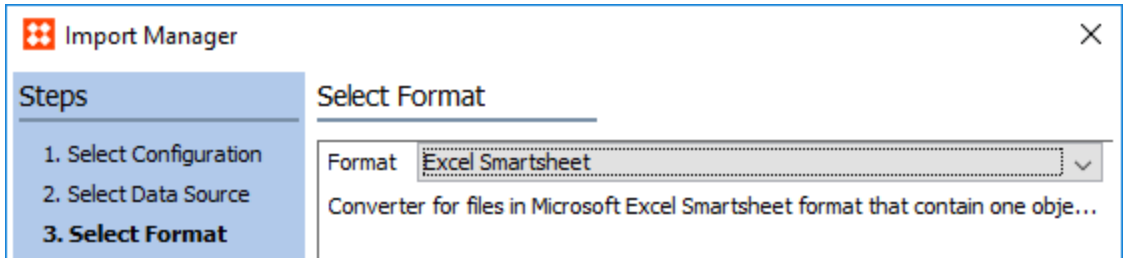
Creating an Import Configuration

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, see the **Import Manager** section of the documentation.

Select Format Step

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



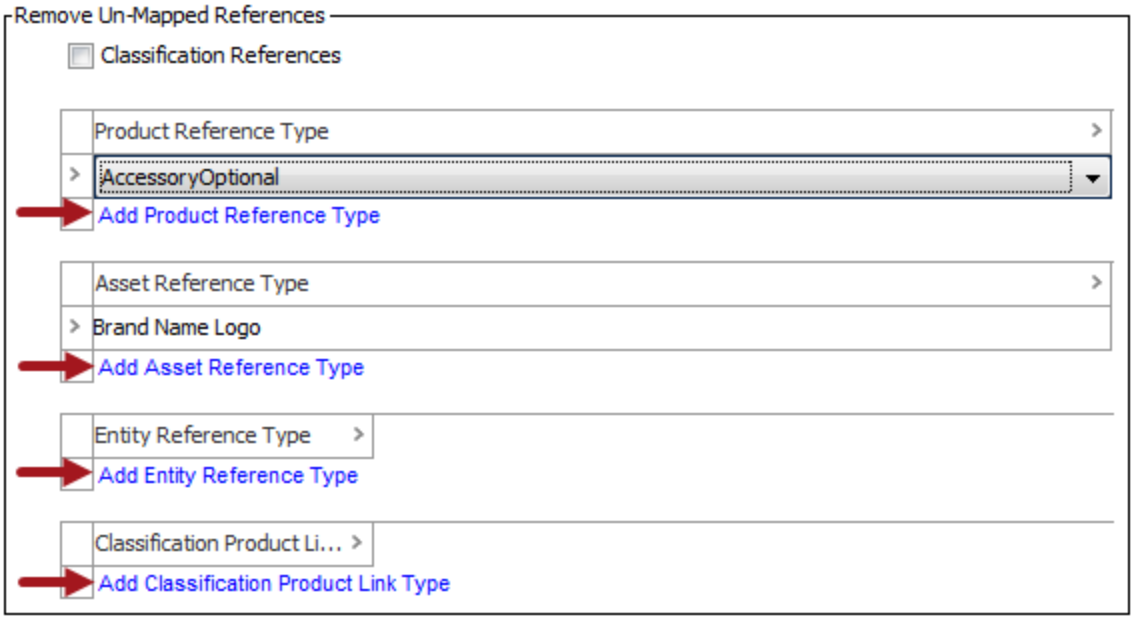
Important: If users encounter an error during the import process, it could be due to the template being used is set to an older version of Excel. It is recommended that users use the newest, supportable version of Excel. For more information on supported versions of Excel, see the current **Platform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

Advanced Settings Step

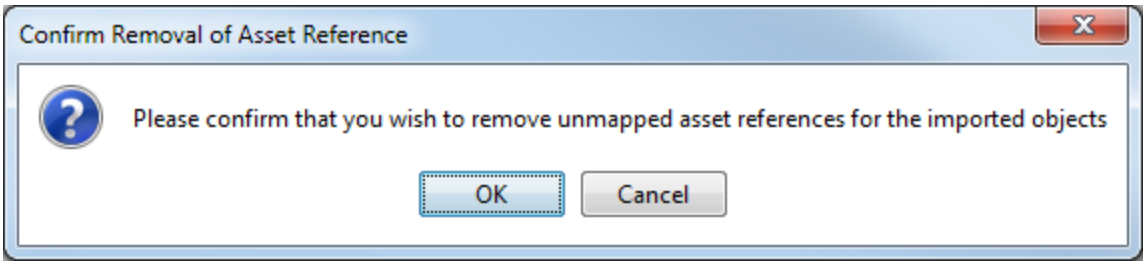
If desired, it is possible to delete / replace references when importing a Smartsheet.

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

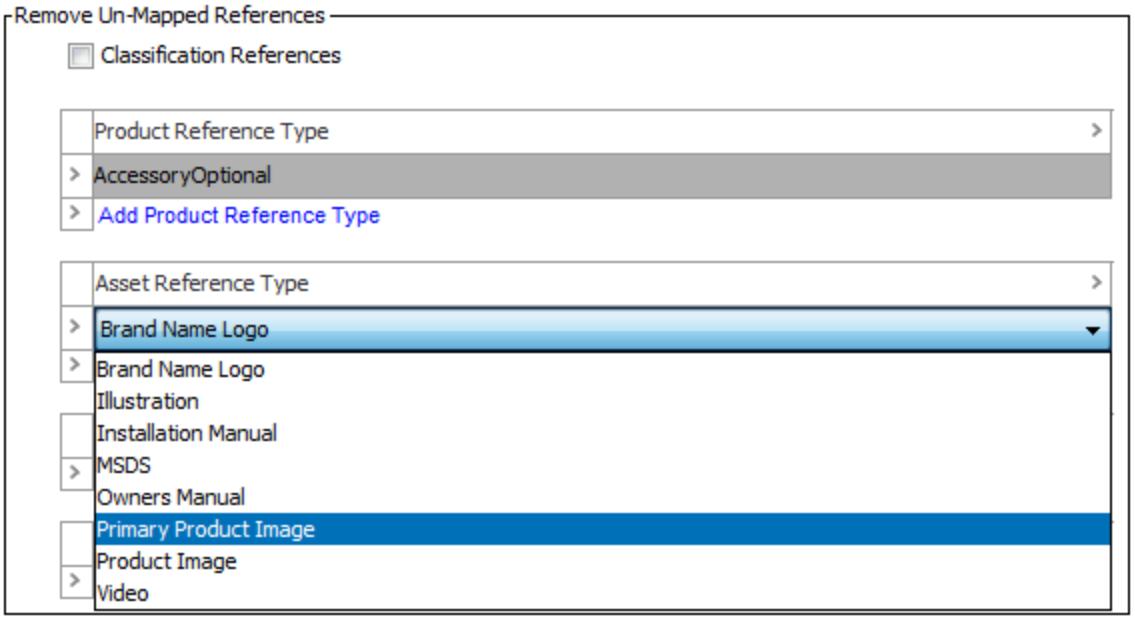
1. In the **Remove Un-Mapped References** parameter, click the relevant 'Add Reference Type' option.



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, see the **Smartsheet Background Processes and Queues** section of the documentation.

Saving and Modifying the Configuration

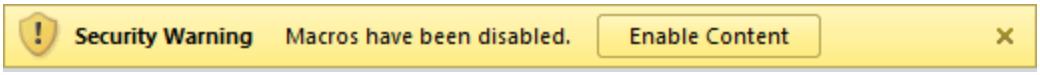
Saving an import configuration is described in the **Running a Data Import** topic of the **Import Manager** documentation.

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

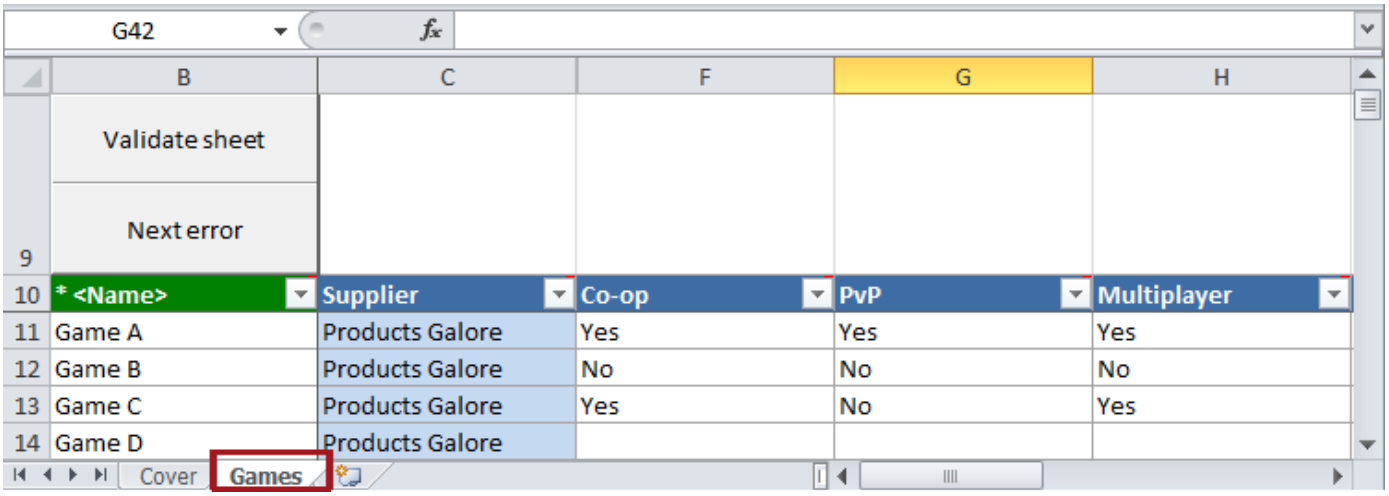
Unlike typical Excel sheets, Smartsheets are very restrictive and must be handled in a very specific way.

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.



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.

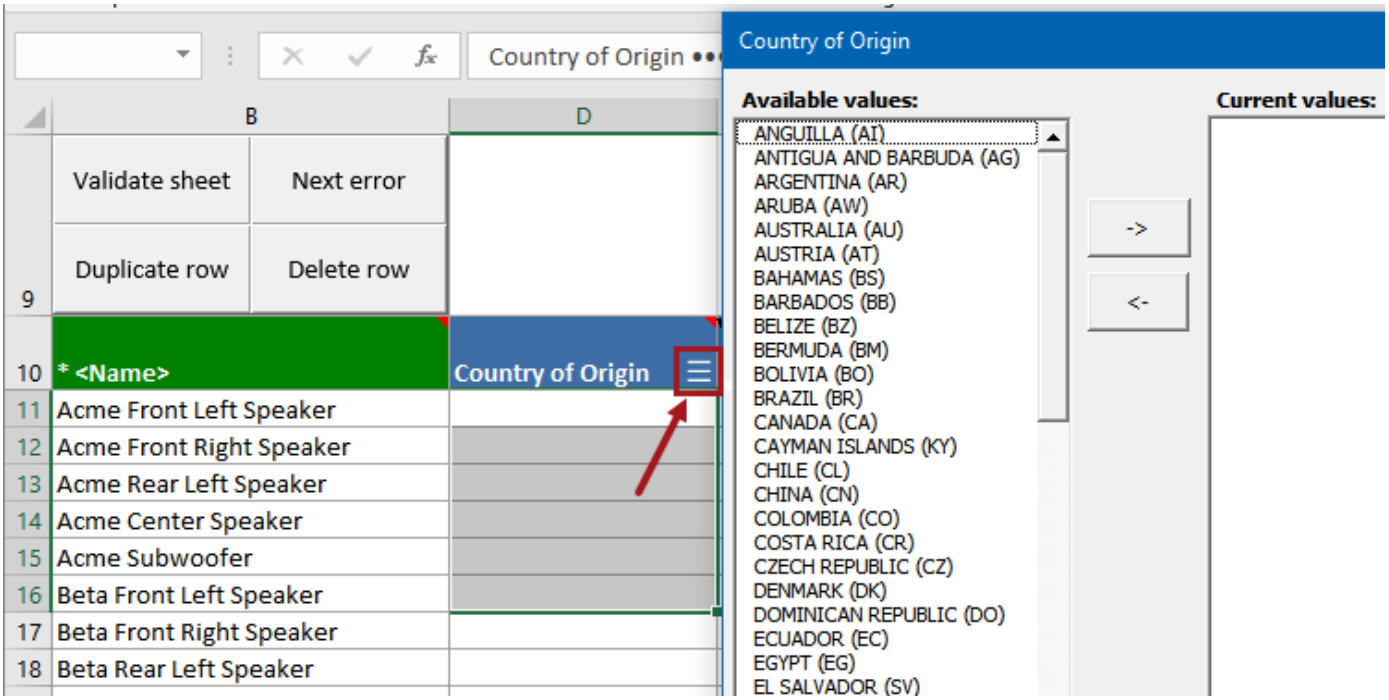
Entering Values for Multi-Valued LOVs and References

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

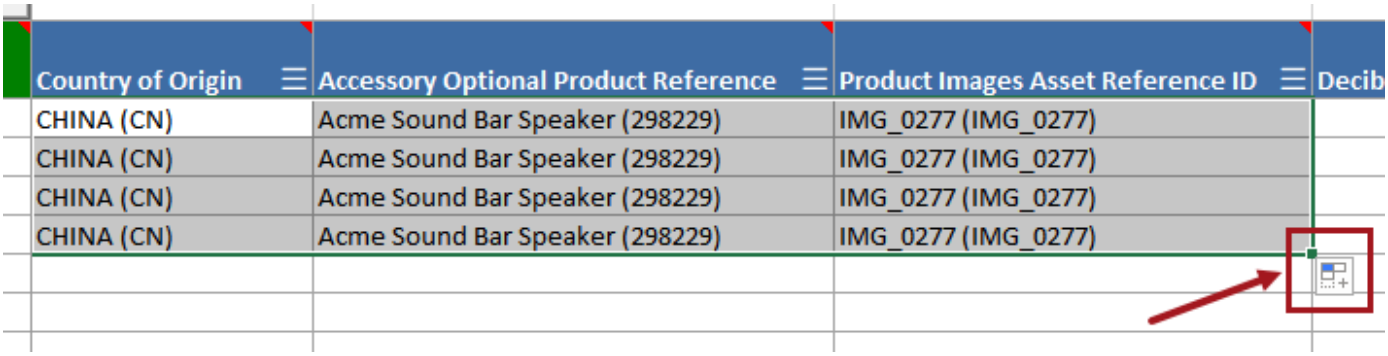
1. Select one or more cells in the column in which you want to add data. Both adjacent and non-adjacent cells within the same column can be selected. Hold the Ctrl key while clicking the relevant cells to select non-adjacent cells.

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.



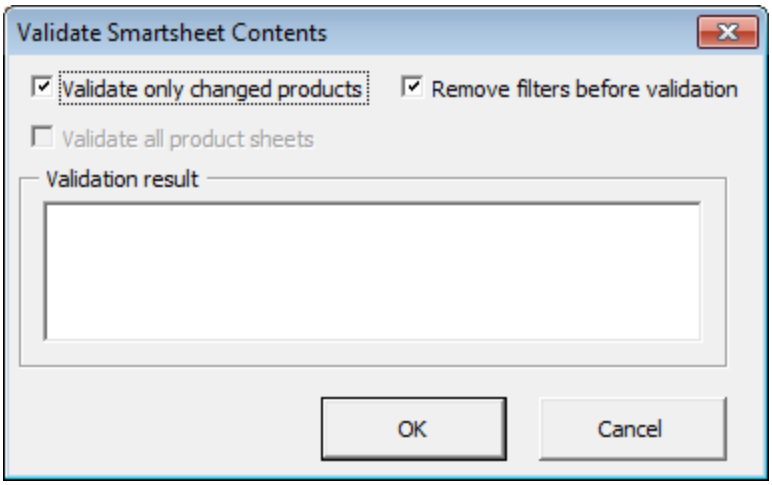
Like other values in Smartsheets, values populated from multi-valued LOVs, multi-product reference types, multi-asset reference types, and parent IDs can also be dragged down in adjacent columns using the Excel fill handle, as shown in the below screenshot. For more information on LOVs, see the **Smartsheet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



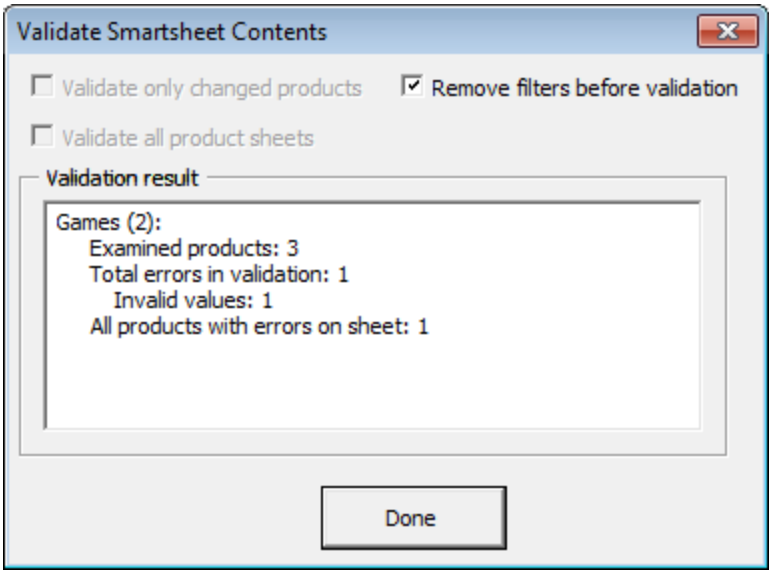
Validating the Sheet

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. 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 that need to be corrected.



Errors 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 from the business condition are shown in the first column.

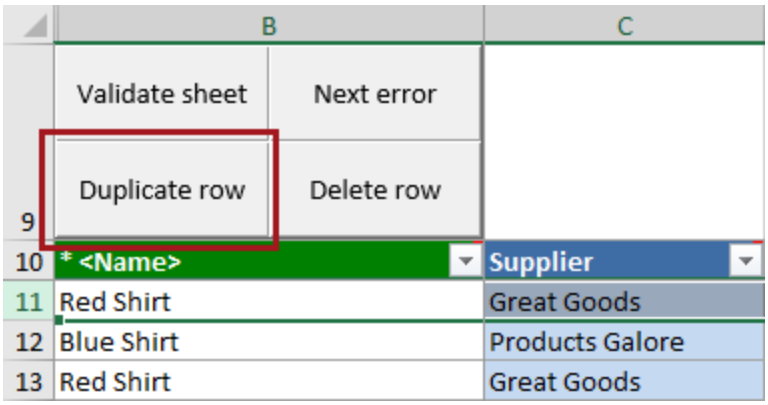
5. After resolving the errors, save the file.

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

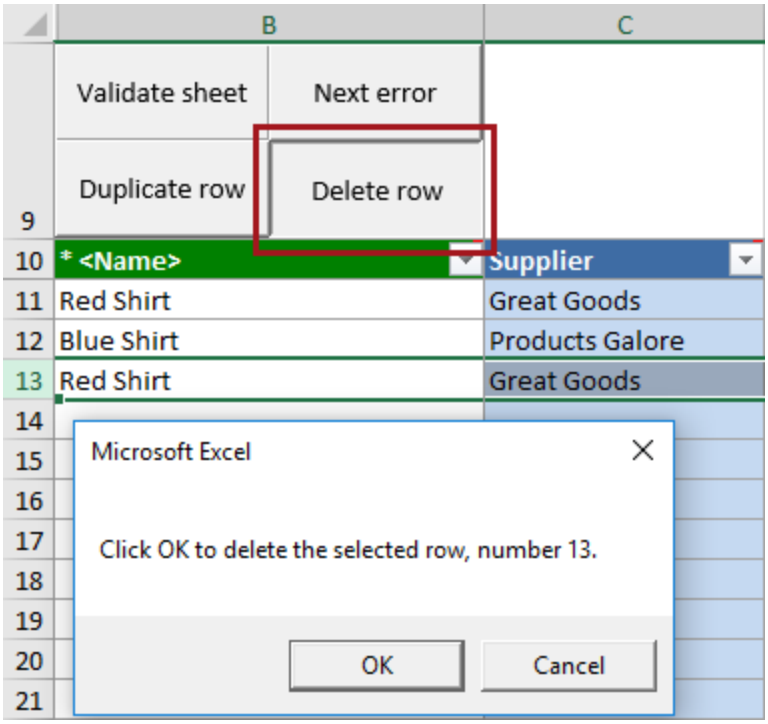
When initiating products using a Smartsheet template, users can duplicate data on a row-by-row basis, potentially saving them a lot of time. Note, however, that rows can only be copied in Smartsheet templates, and may need to be configured to do so.

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.



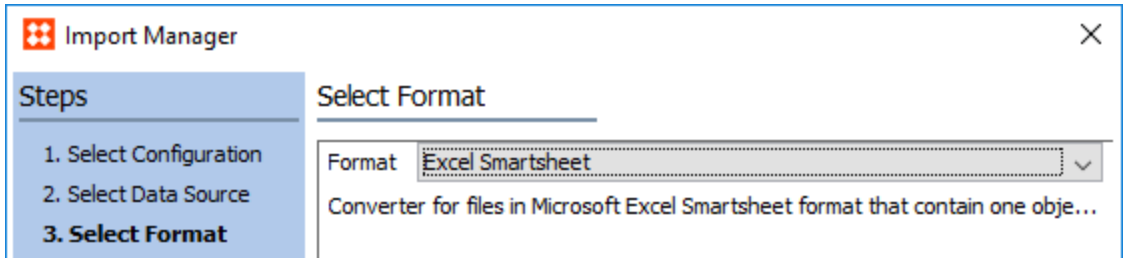
For more information on configuring these functions, see the **Smartsheet Data and Template Configurations** section of the documentation.

Smartsheet 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.

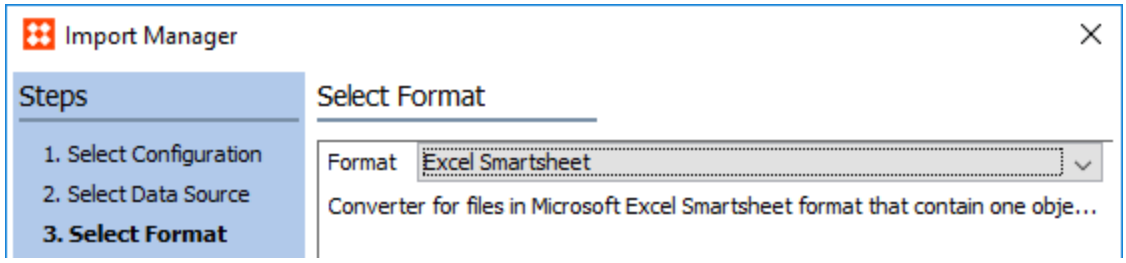
For each product category selected for download, a sheet will be added to the Excel workbook.



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. This functionality works best when there is a large overlap between the set of attributes that are valid for each of the object types.

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.



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.

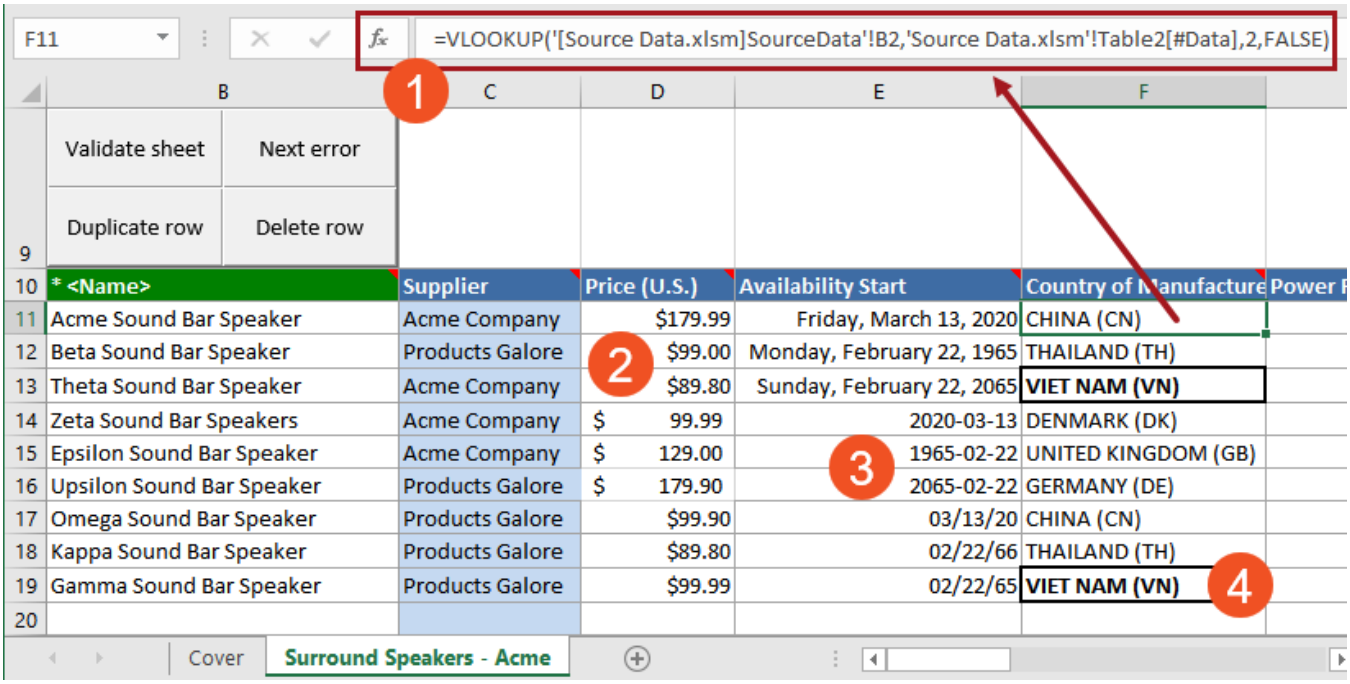
Open Format Smartsheet

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:

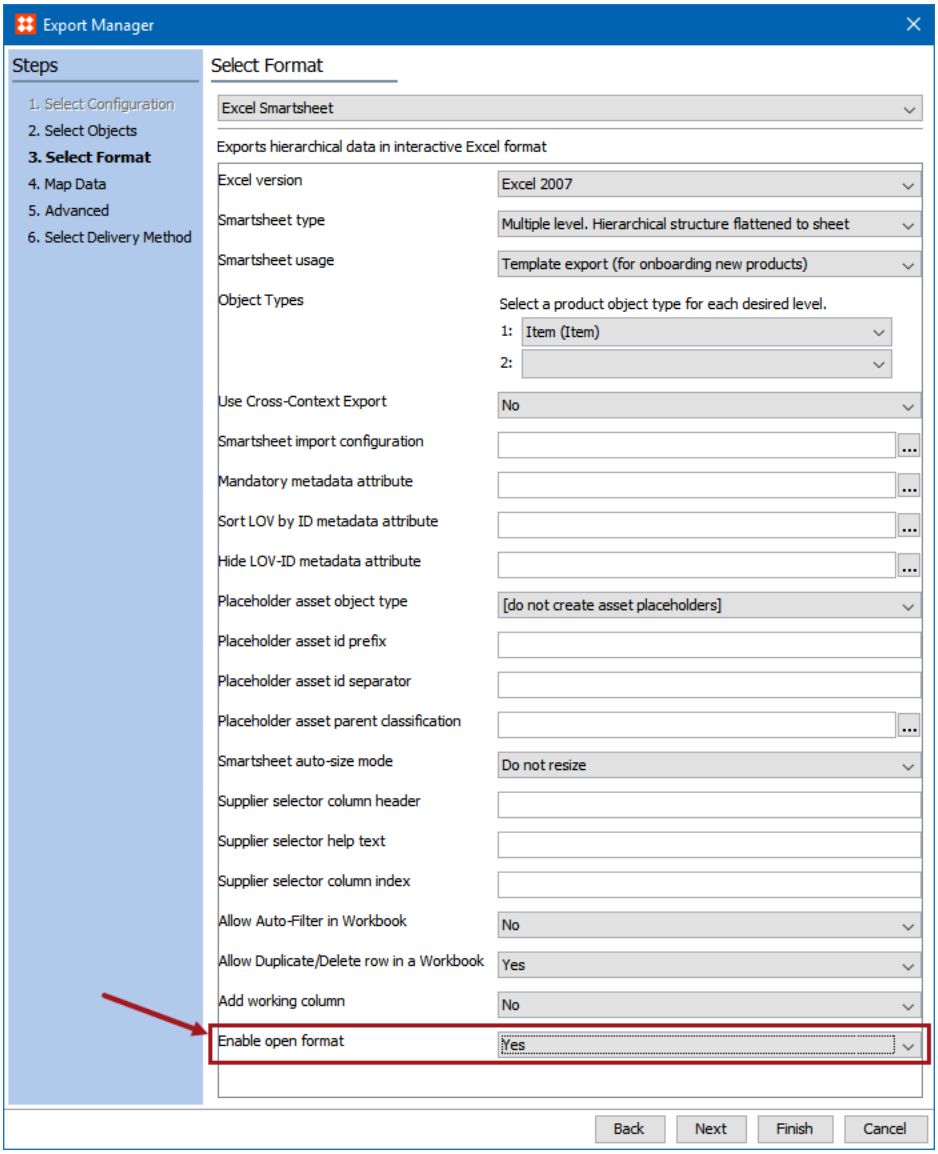
1. Excel formulas. In this example, the formula uses a VLOOKUP function to pull data from an external spreadsheet.
2. Currency and Accounting number formats.
3. Various date formats.
4. Text and cell formatting. This example uses bold text and a thick cell border.



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, see the **Smartsheet Data and Template Configurations** topic in this guide.

Note: This option is disabled and set to 'No' if the Excel version is set to Excel 97/2003.



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 you see 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 strikethrough, 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, see 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.

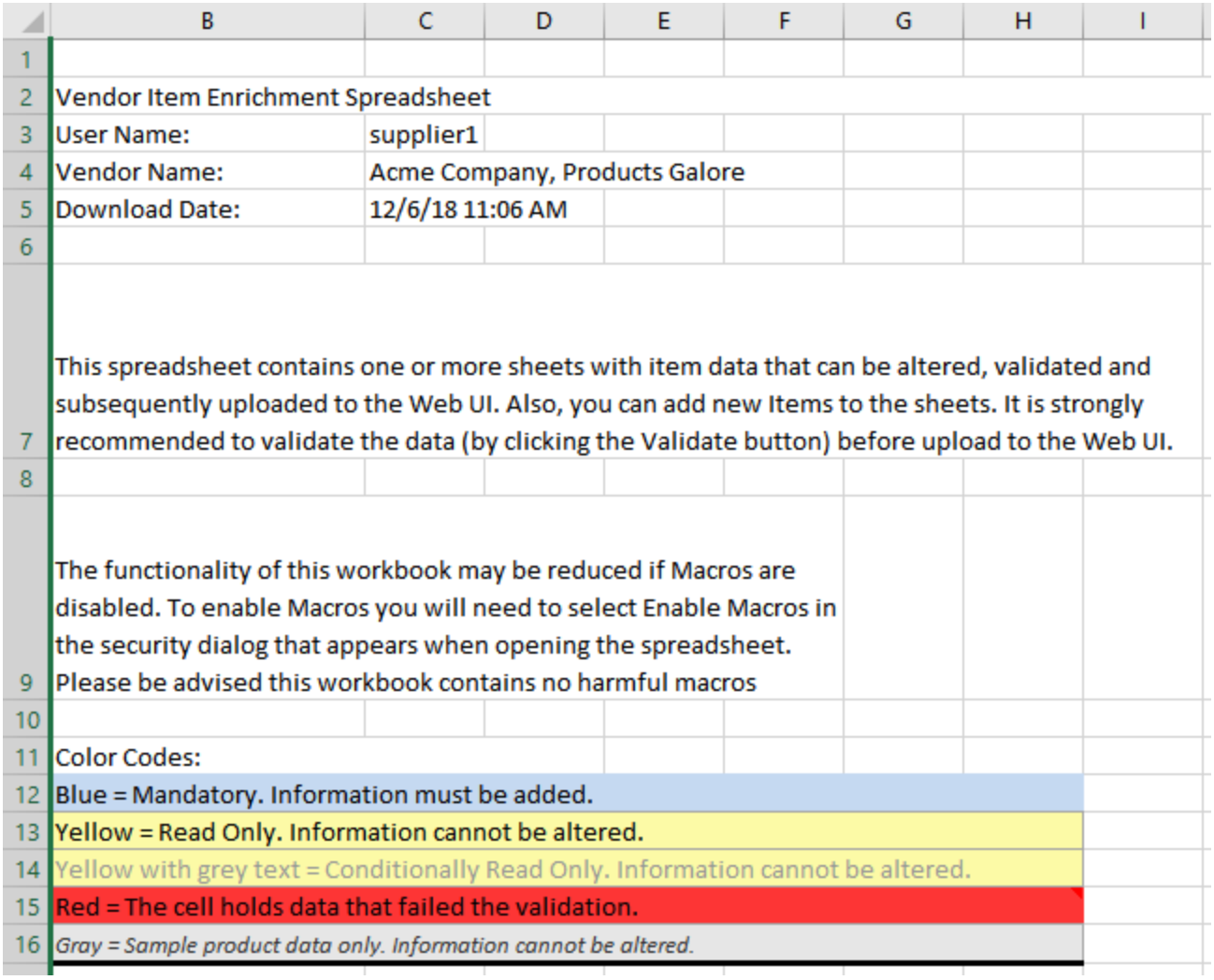
System Configuration

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, see the **Smartsheet Background Processes and Queues** topic in this guide.
- Allow a supplier user to see all tasks across all suppliers for which the user has privileges, in both the workbench and Web UI. For more information, see the **Smartsheet Multi-Supplier View** topic in this guide.
- Limit supplier browsing based on user privileges. For more information, see the **Smartsheet System Properties** topic in this guide.
- Override the URL of the Web UI server when exporting a Smartsheet. For more information, see the **Smartsheet System Properties** topic in this guide.
- Add Smartsheet validation to Excel's Trusted Header authentication. For more information, see 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, see 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.



	B	C	D	E	F	G	H	I
1								
2	Vendor Item Enrichment Spreadsheet							
3	User Name:	supplier1						
4	Vendor Name:	Acme Company, Products Galore						
5	Download Date:	12/6/18 11:06 AM						
6								
7	This spreadsheet contains one or more sheets with item data that can be altered, validated and subsequently uploaded to the Web UI. Also, you can add new Items to the sheets. It is strongly recommended to validate the data (by clicking the Validate button) before upload to the Web UI.							
8								
9	The functionality of this workbook may be reduced if Macros are disabled. To enable Macros you will need to select Enable Macros in the security dialog that appears when opening the spreadsheet. Please be advised this workbook contains no harmful macros							
10								
11	Color Codes:							
12	Blue = Mandatory. Information must be added.							
13	Yellow = Read Only. Information cannot be altered.							
14	Yellow with grey text = Conditionally Read Only. Information cannot be altered.							
15	Red = The cell holds data that failed the validation.							
16	*Gray = Sample product data only. Information cannot be altered.*							

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 grey text = Conditionally Read Only. Information will be ignored upon import and validation.

For more information on open format Smartsheets, see the **Open Format Smartsheet** topic in this guide.

Smartsheets Special Attribute Handling

Smartsheets support and enforce conditional attributes and workflow state mandatory attributes.

Conditional Attributes

When exporting products with conditional attributes any cells containing invalid conditionally displayed values will be automatically locked for editing. In order 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, see the **Conditional Attribute Display** section of the **System Setup** documentation.

Workflow State Mandatory Attributes

When maintaining objects in a Smartsheet that have been exported from a workflow state (i.e., from the Web UI Tasklist), any cells containing attributes that are mandatory for that particular state are highlighted blue. Additionally, unless the condition driving the mandatory status of the attributes returns 'False', the user must enter values for all applicable attributes before the sheet can be validated.

	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

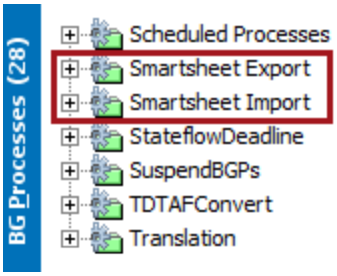
Important: Attributes mandatory for transitions are not enforced in Smartsheets as the selected transition is not known.

For more information, see the **Conditionally Mandatory Workflow Attributes** section of the **STEP Workflows** documentation.

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.

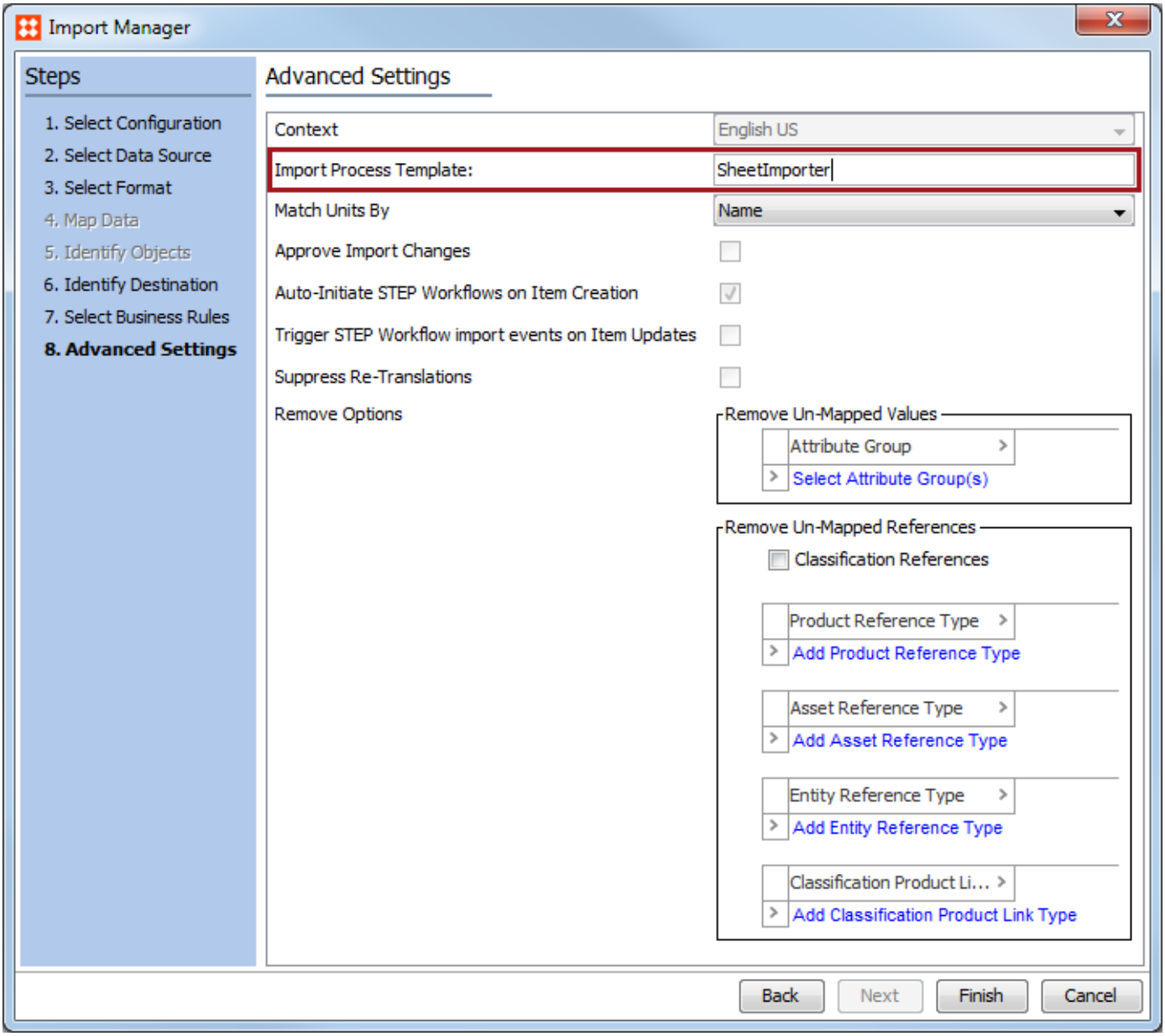
Important: 'BackgroundProcess.ProcessType.SheetExporter.Queue' and 'BackgroundProcess.ProcessType.SheetImporter.Queue' default to the same queue as Export Manager and Import Manager, and must be changed in sharedconfig.properties before they can run in their own queues. This step must be completed before 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.

See the **Import Manager** section of the documentation for more information.

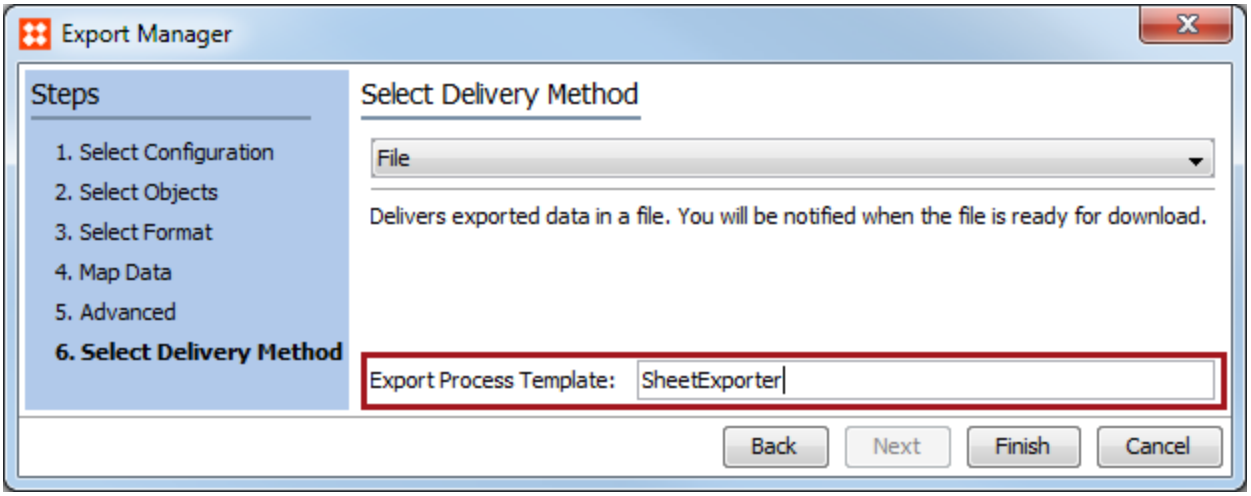


Using Smartsheet Export Background Process

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.

See the **Export Manager** section of the documentation for more information.


Export Manager
Steps
1. Select Configuration
2. Select Objects
3. Select Format
4. Map Data
5. Advanced
6. Select Delivery Method
Select Delivery Method
File
Delivers exported data in a file. You will be notified when the file is ready for download.
Export Process Template:
SheetExporter
Back
Next
Finish
Cancel

Smartsheet Multi-Supplier View

A supplier user can see 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 see all objects (e.g., products 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 GLNs (Global Location Numbers) 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 must select the specific supplier for which the products 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 see any objects belonging to the suppliers for which they are a member, regardless if 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 / Super User Guide** 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 STEP 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.

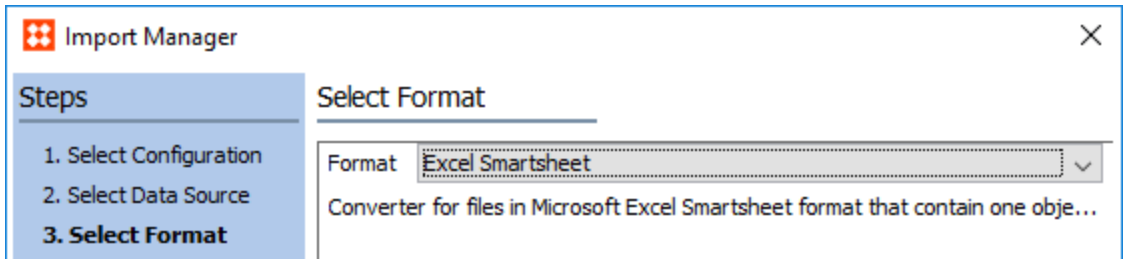
Trusted Headers Authentication

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

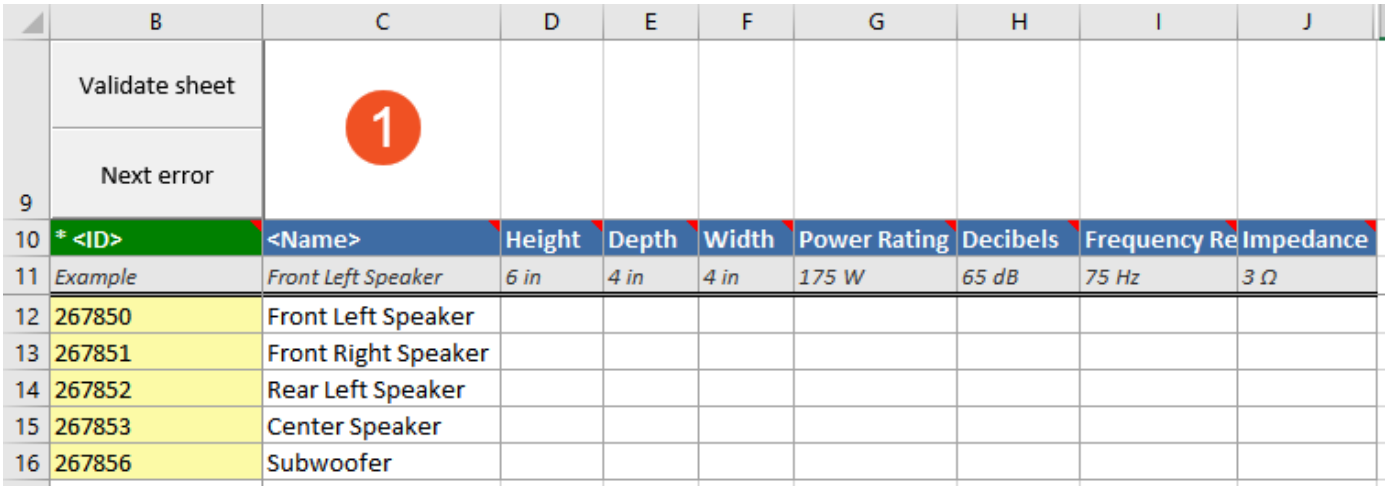
When using the Smartsheet template, the default max is 1,000 rows. Using the case-sensitive 'SmartSheet.PreformattedTemplateRows' property, you can increase the generated default up to 5,000 rows.



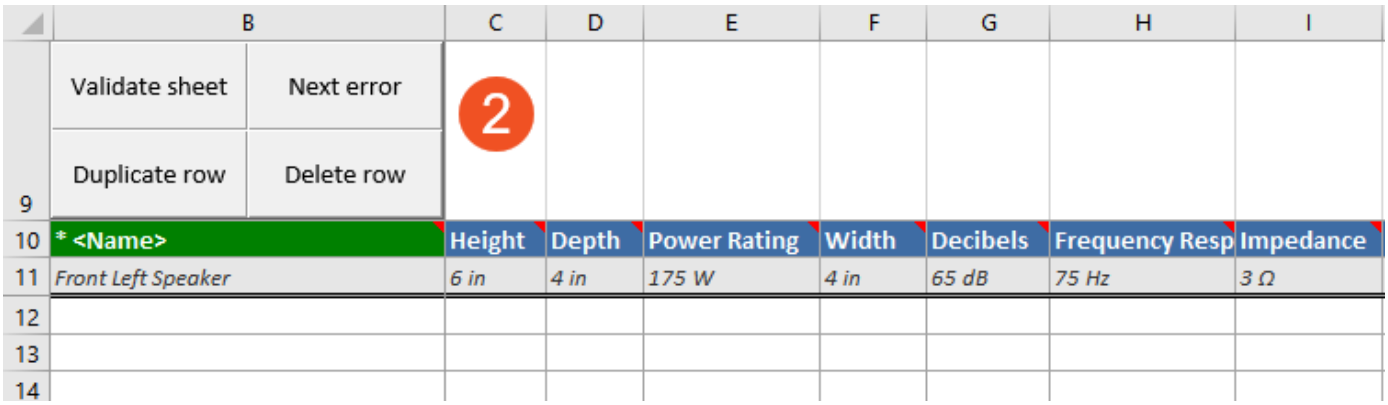
Smartsheet Template Product Row

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 / Duplicate row / Delete row	2						
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, see 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.

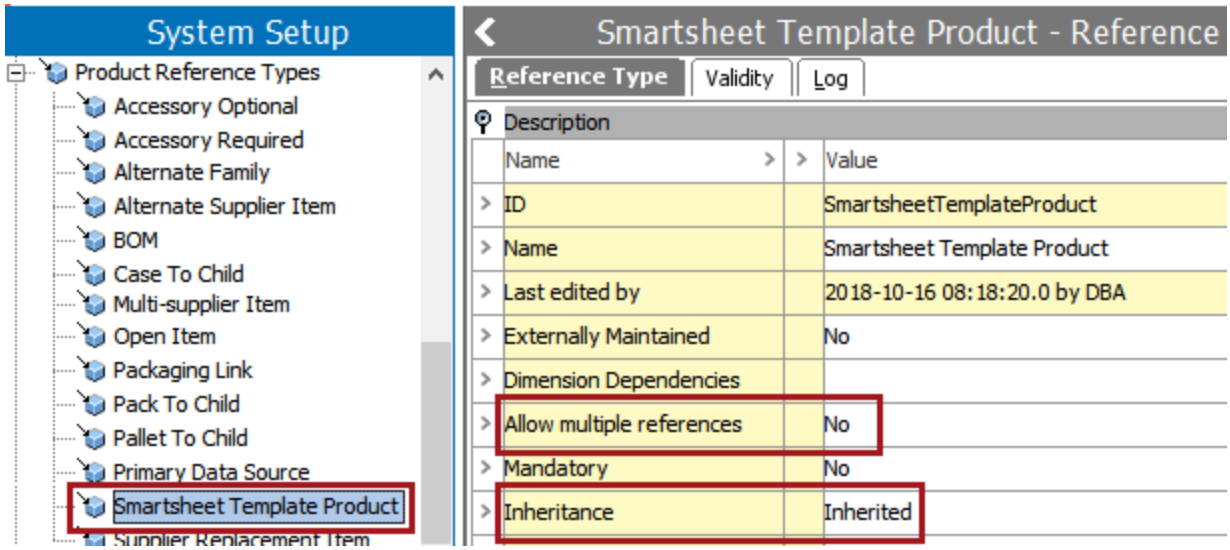
Smartsheet Template Product Reference

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:

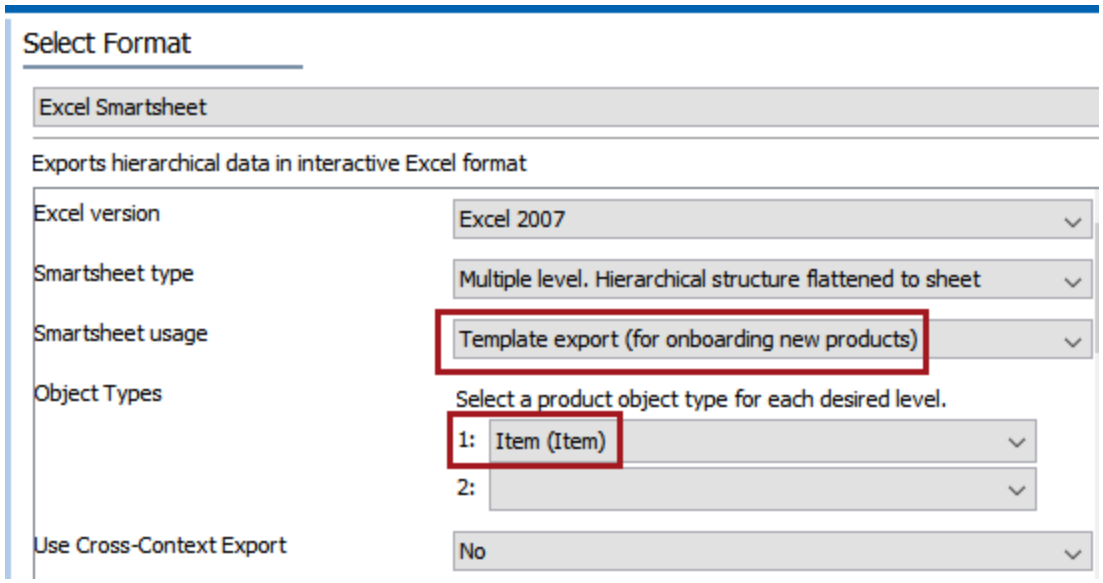
- **Allow multiple references** is set to **No** because only one source product should contain the example attribute values.
- **Inheritance** is set to **Inherited** because the example product values should be applicable to child products within the category.



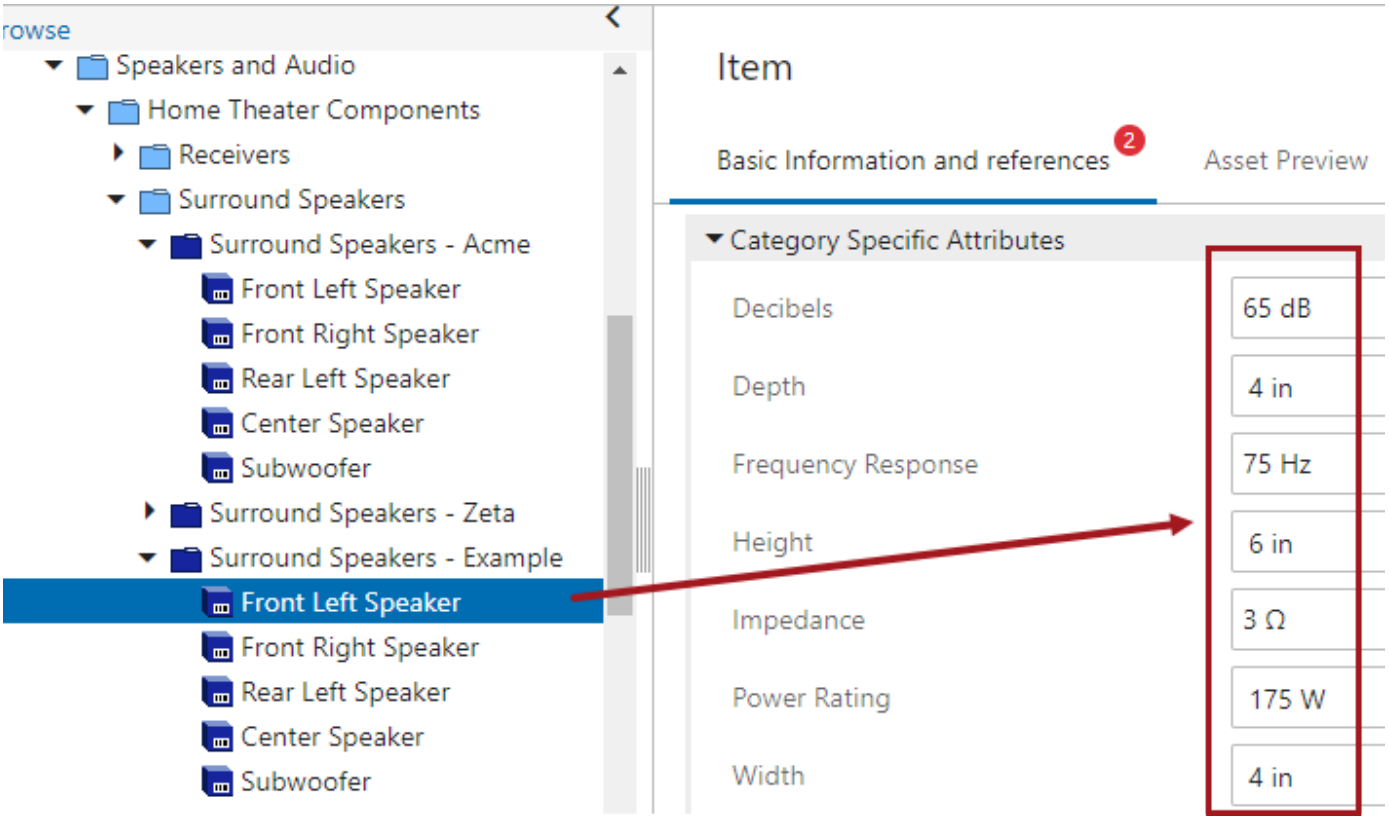
Example Setup

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.

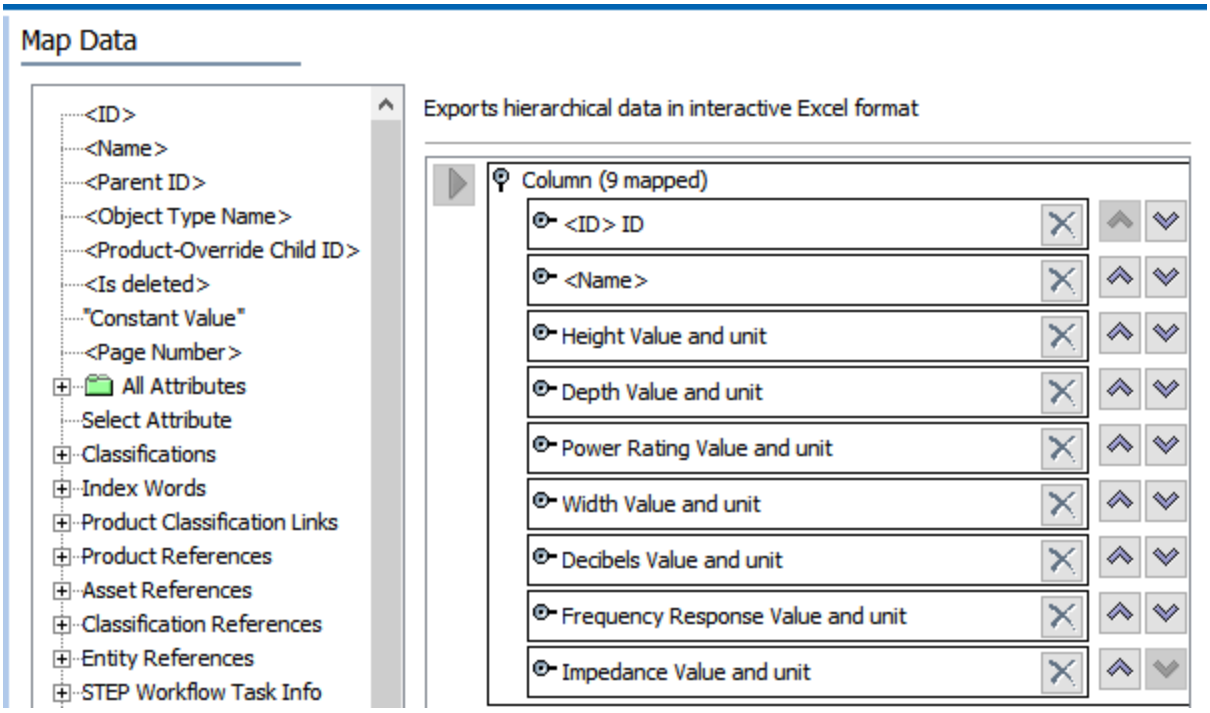
1. In this example, the Smartsheet Export Configuration is set to 'Template export' using the object type of 'Item.'



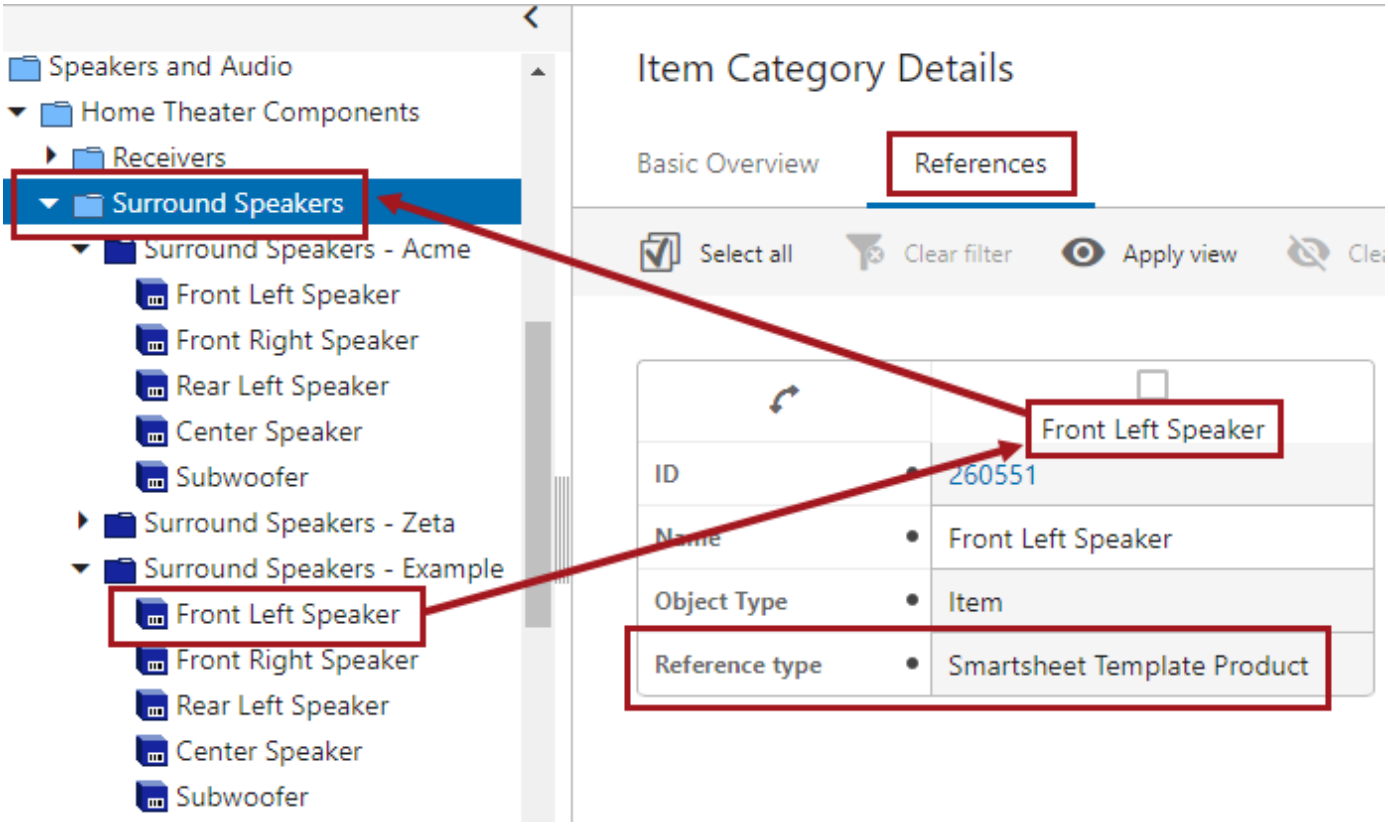
2. The example product is an Item object named 'Front Left Speaker.' It contains values for category-specific attributes such as Decibels, Frequency Response, Height, Width, and so forth.



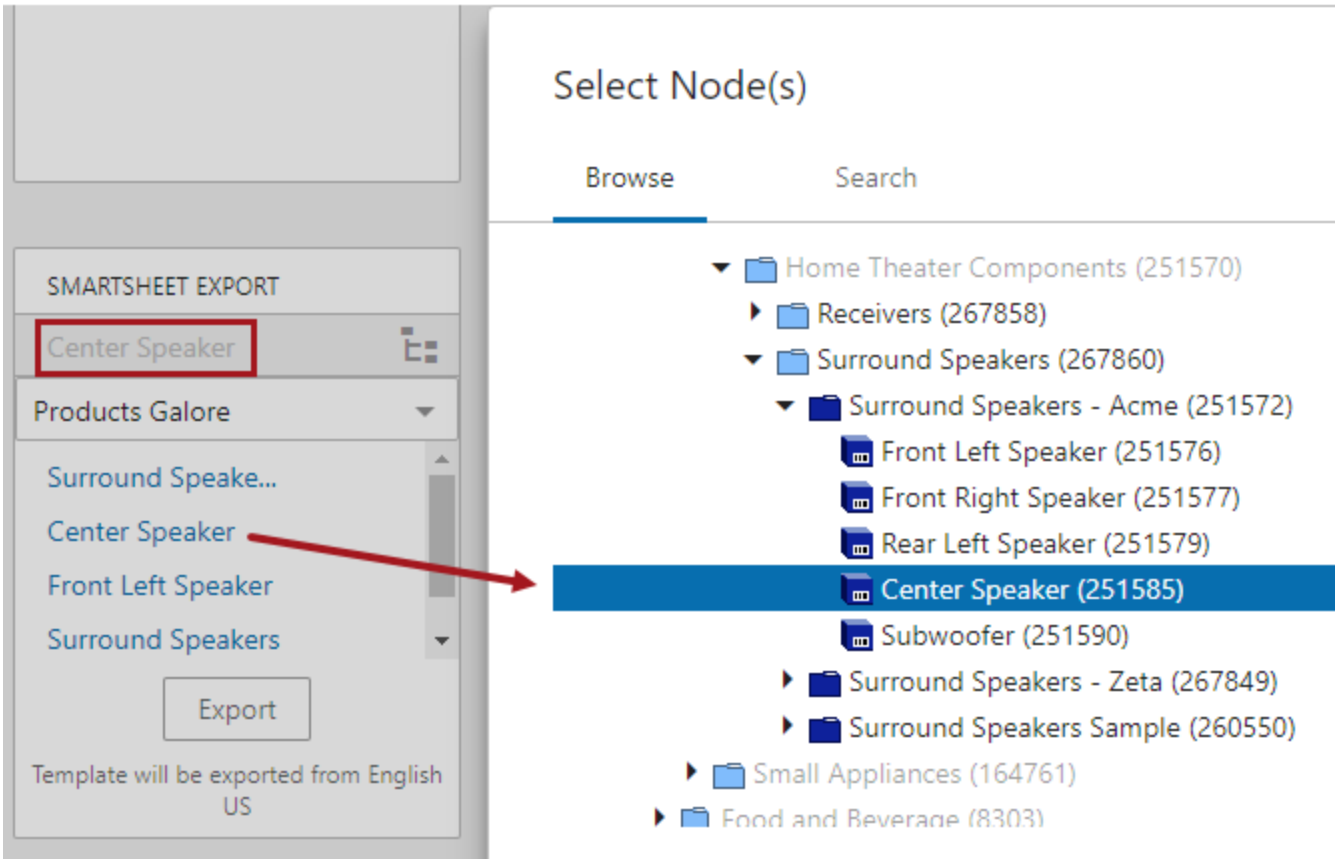
3. The same attributes have been mapped in the Smartsheet export configuration.



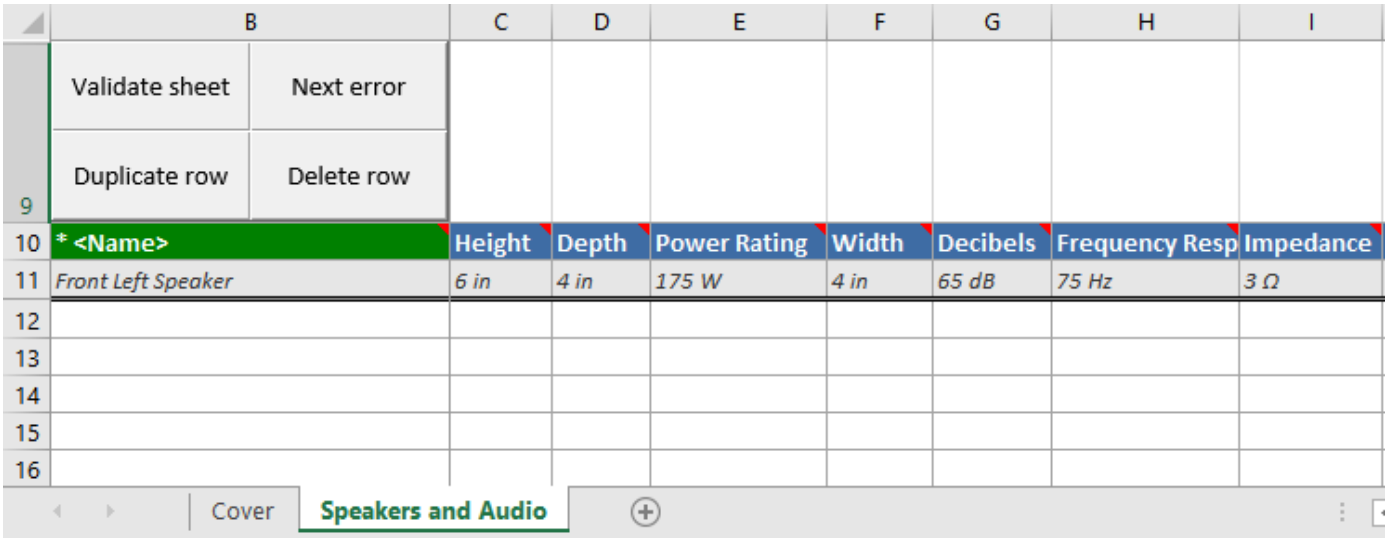
4. The 'Front Left Speaker' example product is referenced by the top-level of the product category ('Surround Speakers') using the Smartsheet Template Product reference.



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 - see **Creating an Inbound Integration Endpoint**
- Export Manager - see **Creating a Data Export**
- OIEP - see **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, see **Data Mapping**.

For inbound mapping examples, see the **FAB-DIS Import Format** topic.

For outbound mapping examples, see the **FAB-DIS Export Format** topic.

Inbound Data

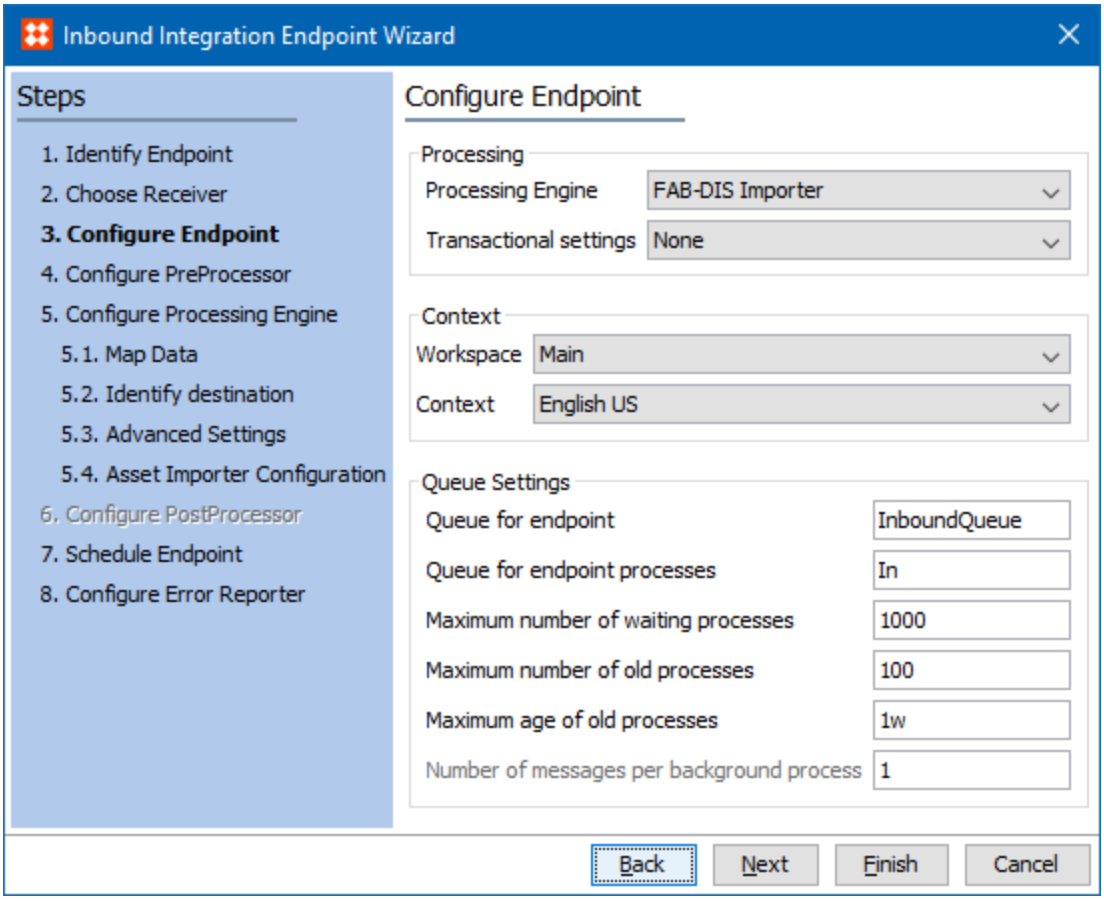
The FAB-DIS format is not available from Import Manager, but you can achieve the same result by manually running an IIEP using the FAB-DIS Importer processing engine.

Ultimately, Web UI provides the easiest FAB-DIS data import via the **File Loading Widget**. Web UI monitoring is available via the **Background Process Notification Component**. Information about both of these is included in the **Web User Interfaces / Web UI Setup and User Guide** documentation.

IIEP

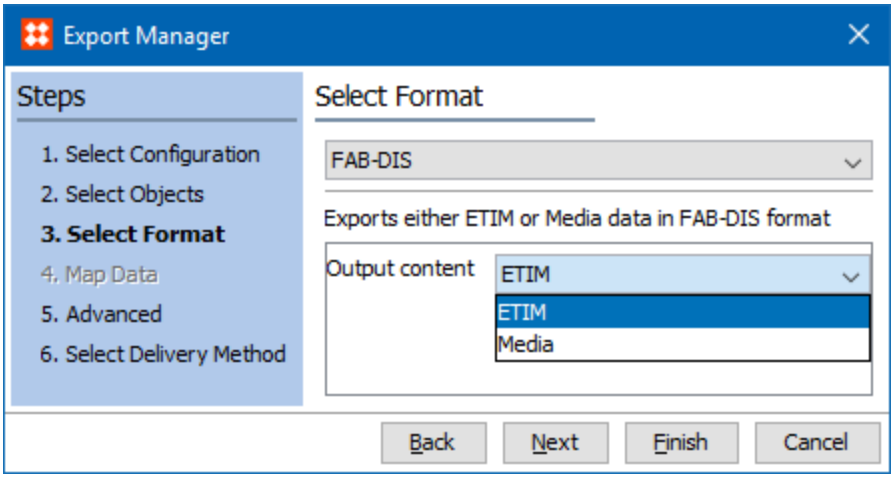
Minimum manual mapping is required when configuring the IIEP since the FAB-DIS Importer processing engine automatically maps the ETIM attributes included in the FAB-DIS import file.

Important: To import successfully, a FAB-DIS file must have only the expected columns, and no additional columns. For details, see the **Prerequisites** section of the **FAB-DIS Import Format** topic.

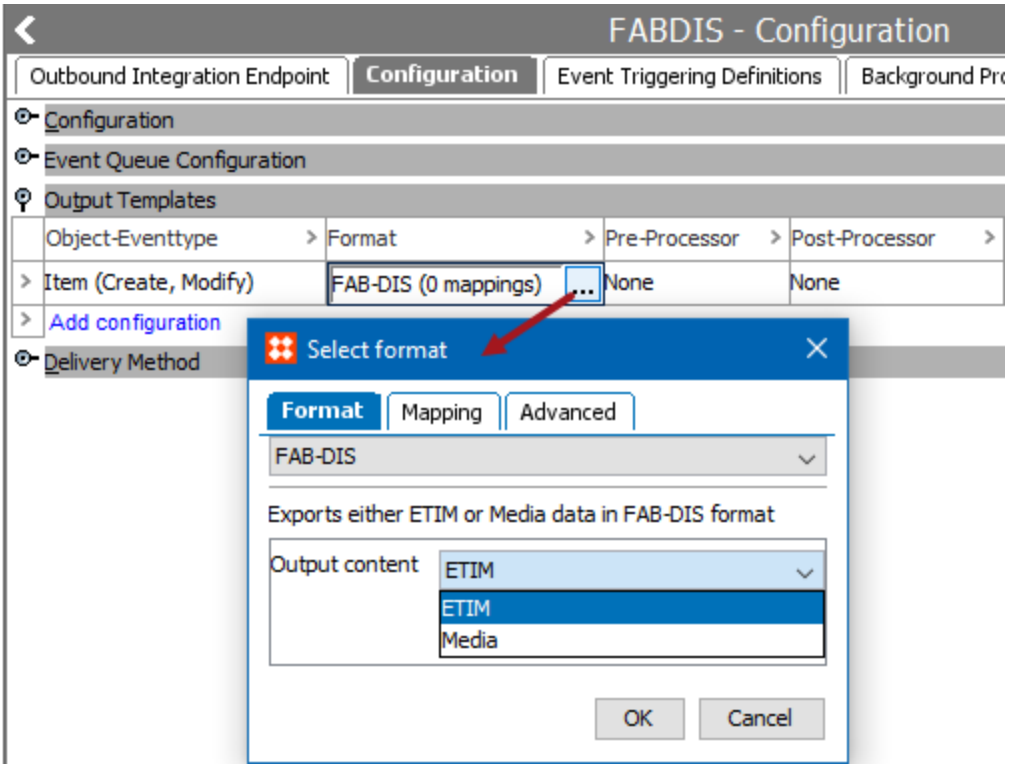


Outbound Data

Export Manager



OIEP


FABDIS - Configuration
Outbound Integration Endpoint
Configuration
Event Triggering Definitions
Background Pr
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.

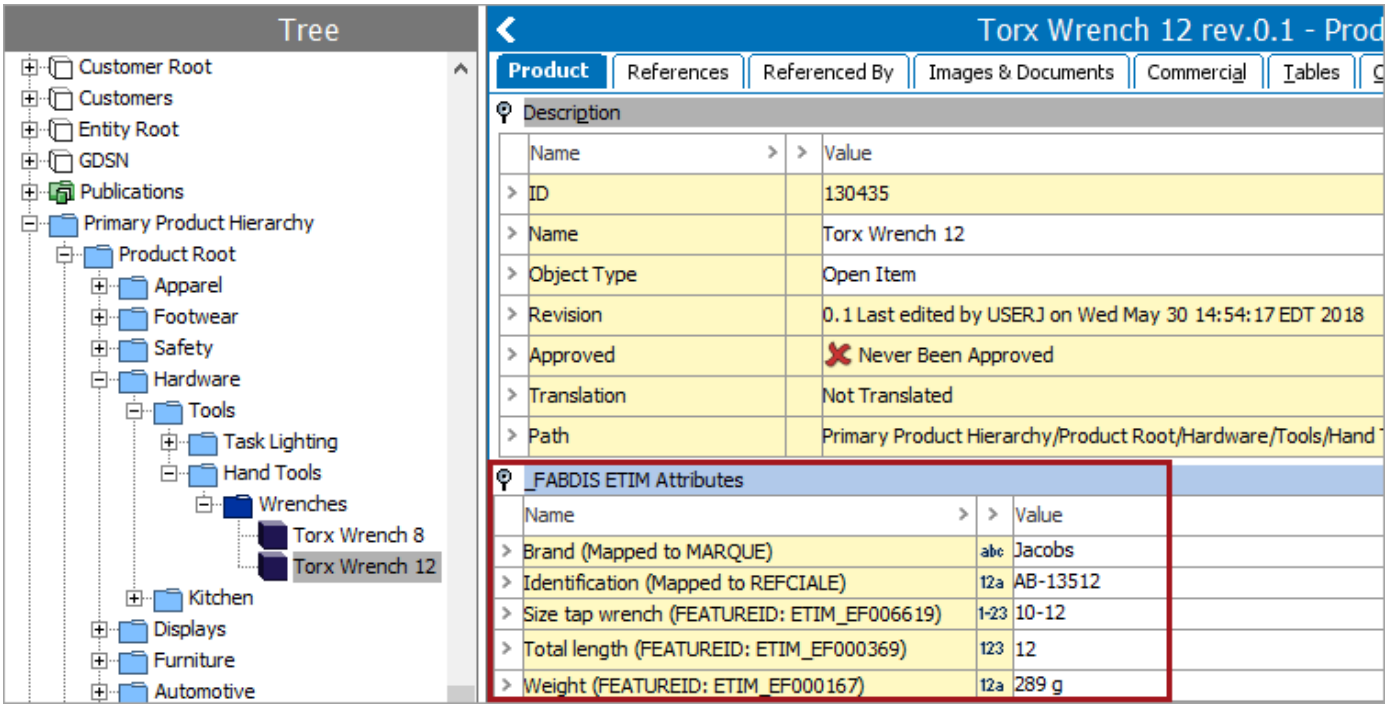
STEP Data for ETIM Mapping Targets

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, see 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

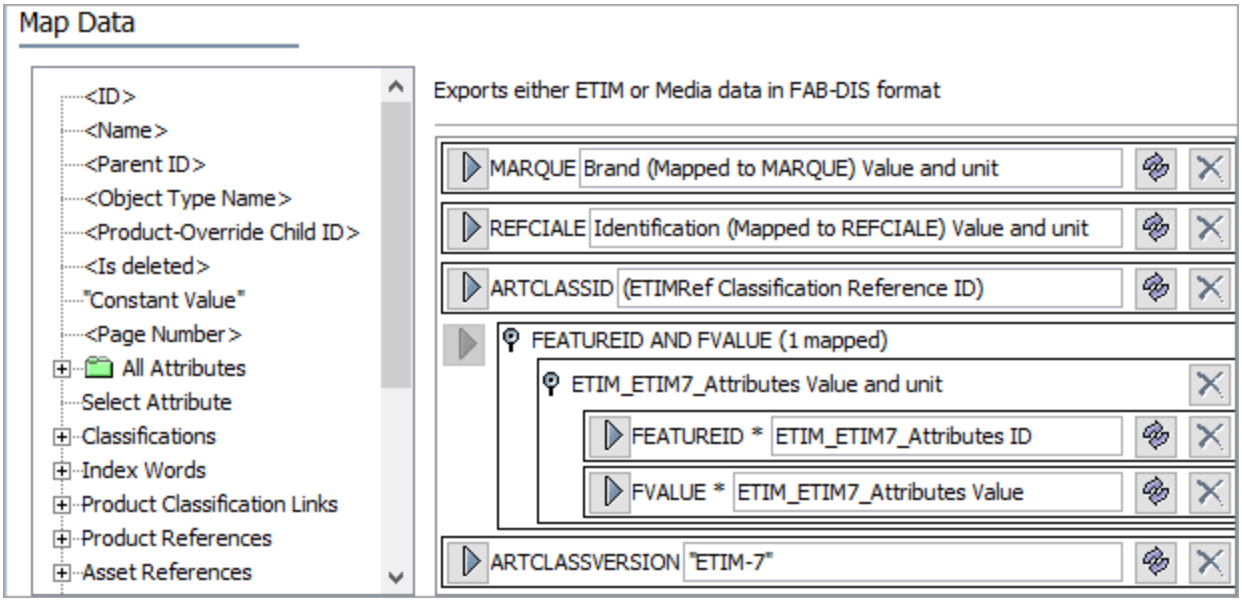
The following attributes are assigned to the '_FABDIS ETIM Attributes' group, and include values for the selected product. To clarify the mapping and output displayed within the images below, mapping information has been added to each attribute name in parentheses.



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 / Super User Guide** 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 / Super User Guide** 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.

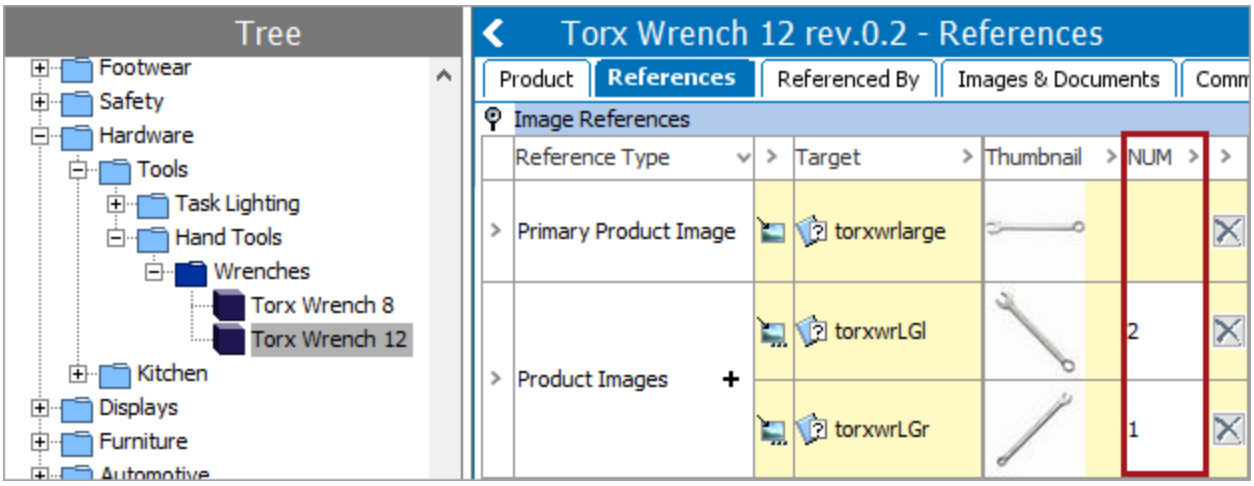
   ---

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.

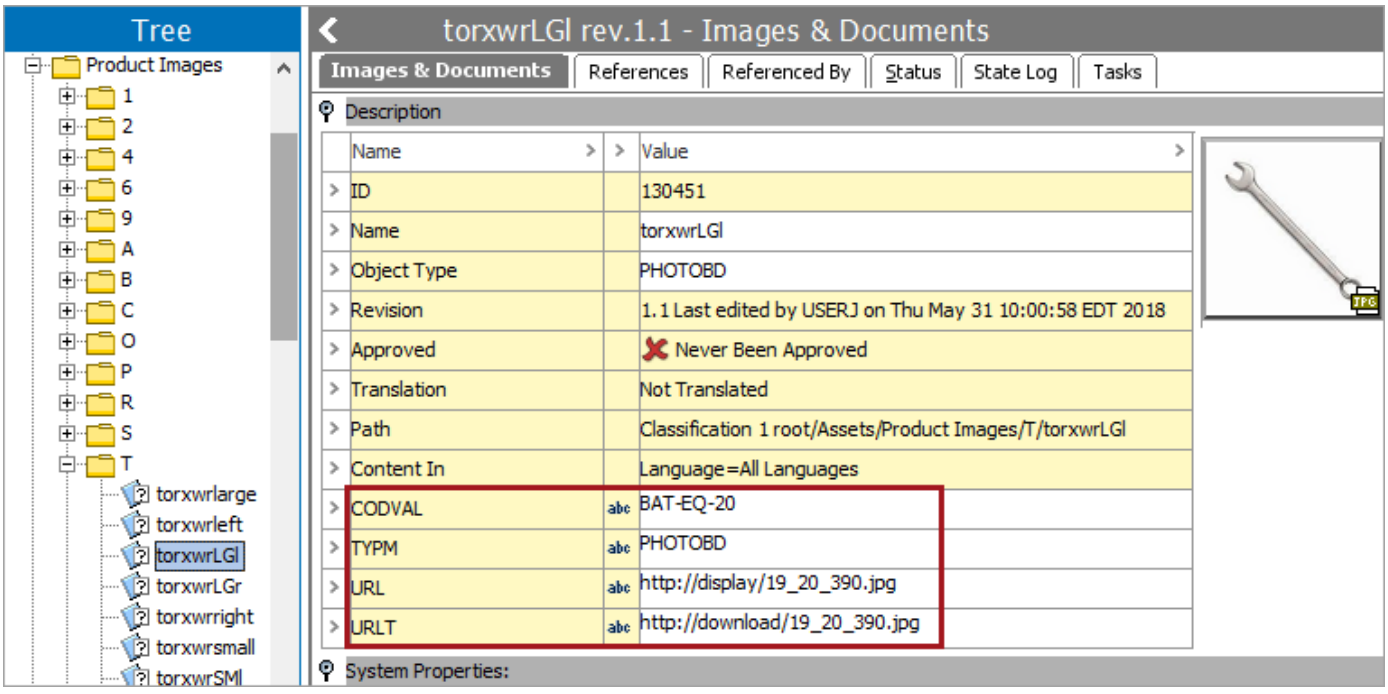
7. URL (text) is the URL to display the document or photo on a merchant site. This is an attribute, possibly a calculated attribute, on the asset itself.
8. URLT (text) is the URL to download the document or photo. This is an attribute on the asset itself.

Sample Data

The NUM value is stored as metadata on the product References tab.



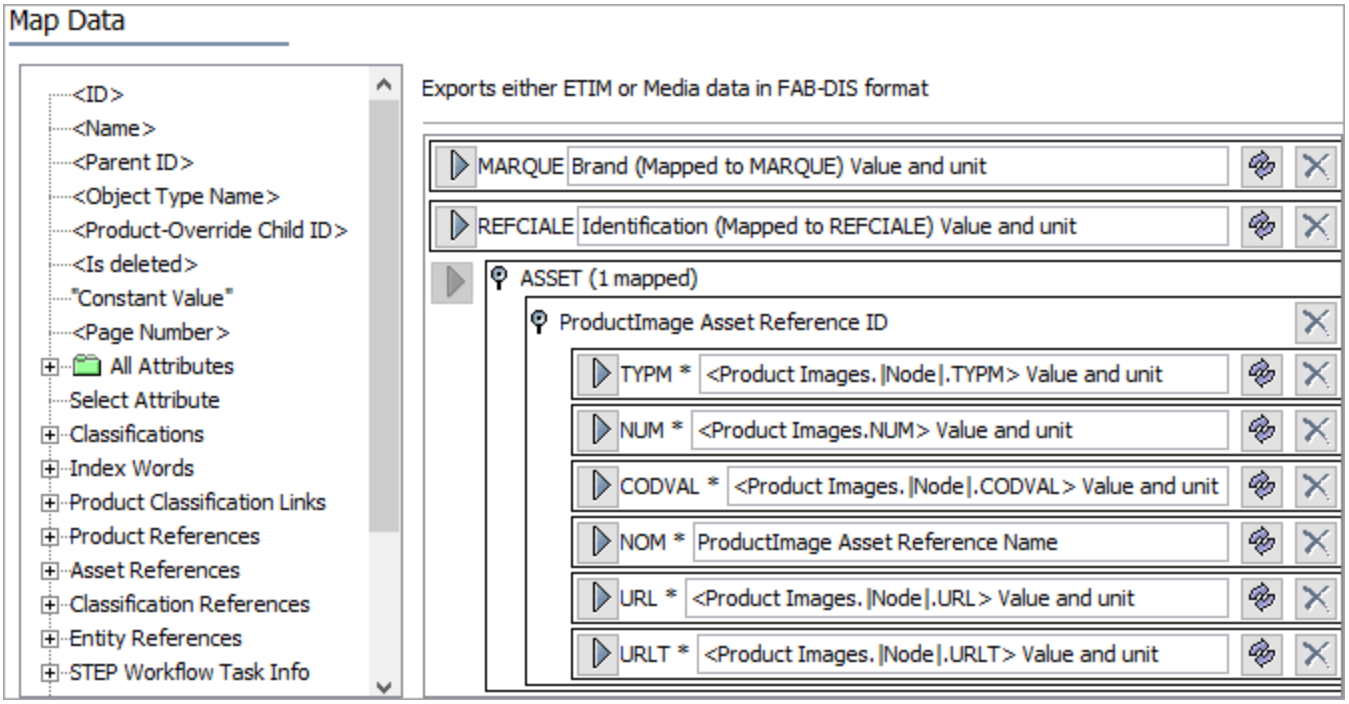
The remaining values are stored as metadata on the asset, as shown below for the Product Images reference with NUM = 2.



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

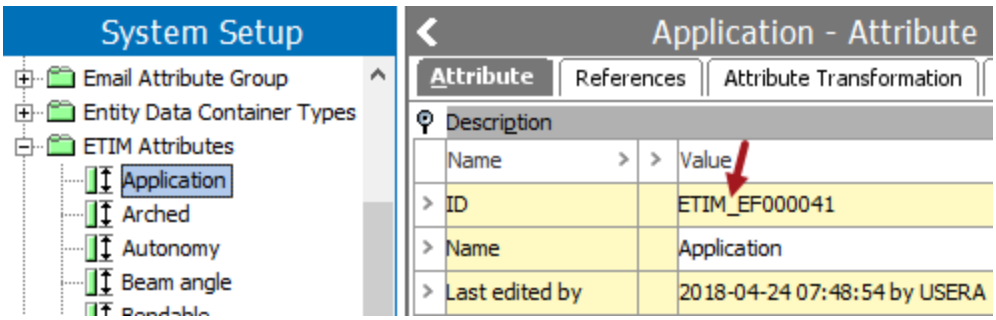
Importing FAB-DIS data from the ETIM and Media tabs of an Excel file is available in an IIEP via the FAB-DIS Importer processing engine. Other tabs in the FAB-DIS file are ignored by the importer.

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, see 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, see 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, see 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

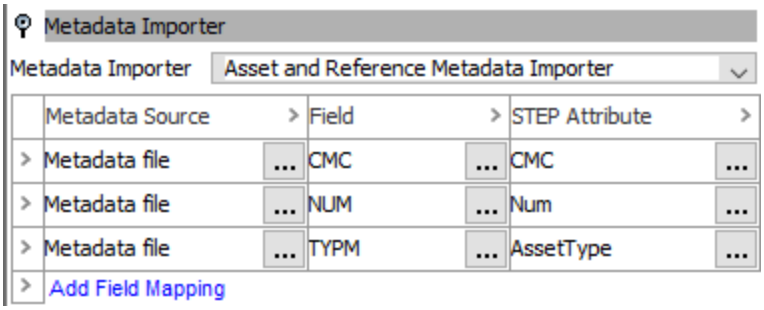
Asset Importer functionality is used to handle assets from the FAB-DIS Media tab. Media tab data triggers an asset importer background process and therefore, an asset importer configuration is required.

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, see 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, see 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.

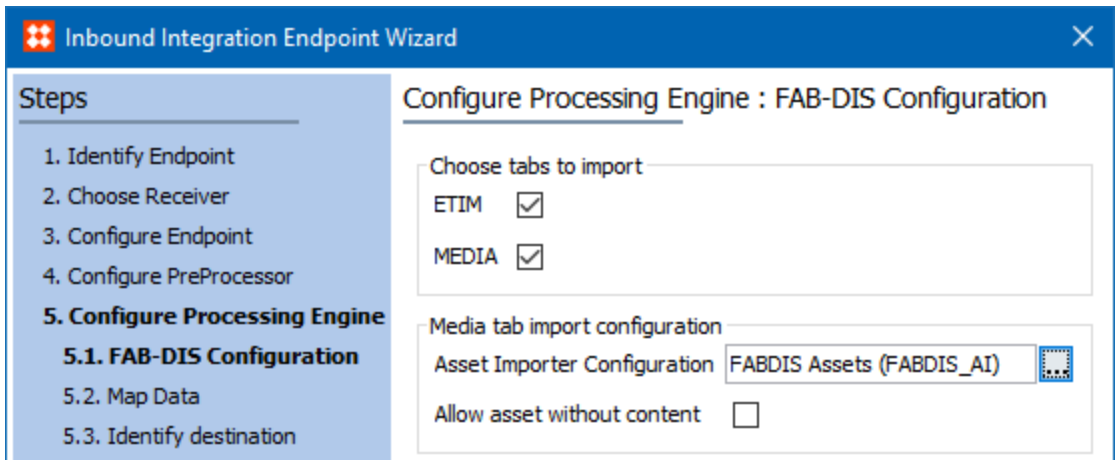
- To set the same reference type for all assets in the import file, leave the 'Reference Type Field' parameter blank, and set the 'Default Reference Type' parameter as the required reference type.

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, see 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, see 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:



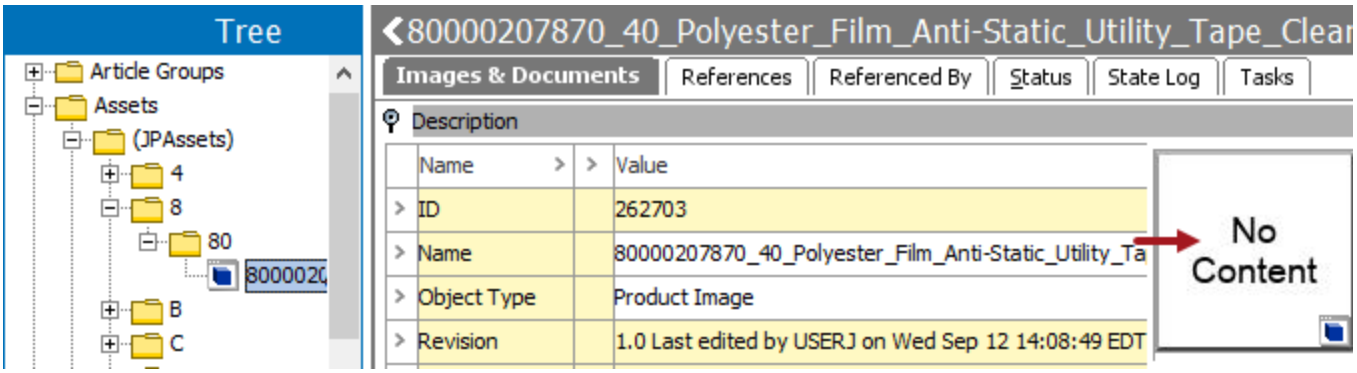
Note: To enable the **Finish** button, at least one checkbox must be checked on the FAB-DIS Configuration step, and one Source field must be mapped on the Map Data step.

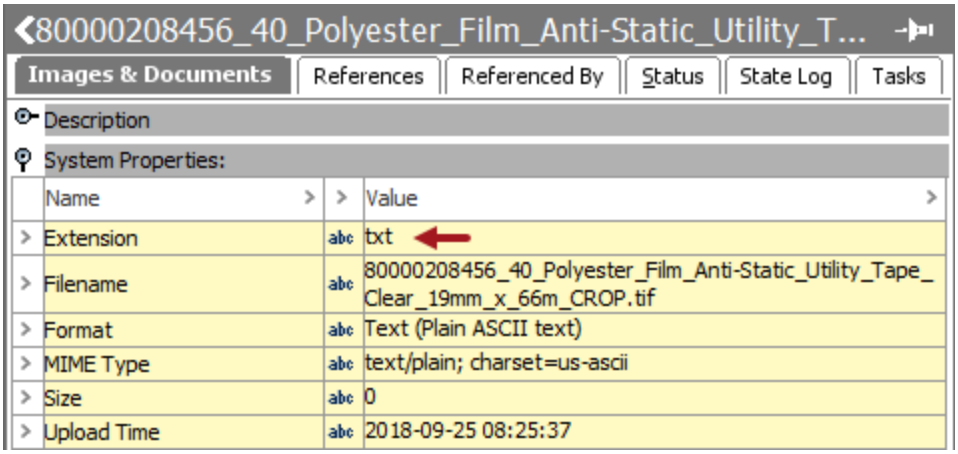
- Check the **ETIM** checkbox if data from the FAB-DIS ETIM tab should be processed.
- Check the **MEDIA** checkbox if data from the FAB-DIS Media tab should be processed. New products identified on the Media tab only, based on the REFCIALE column, are created in order to connect them to the imported 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.



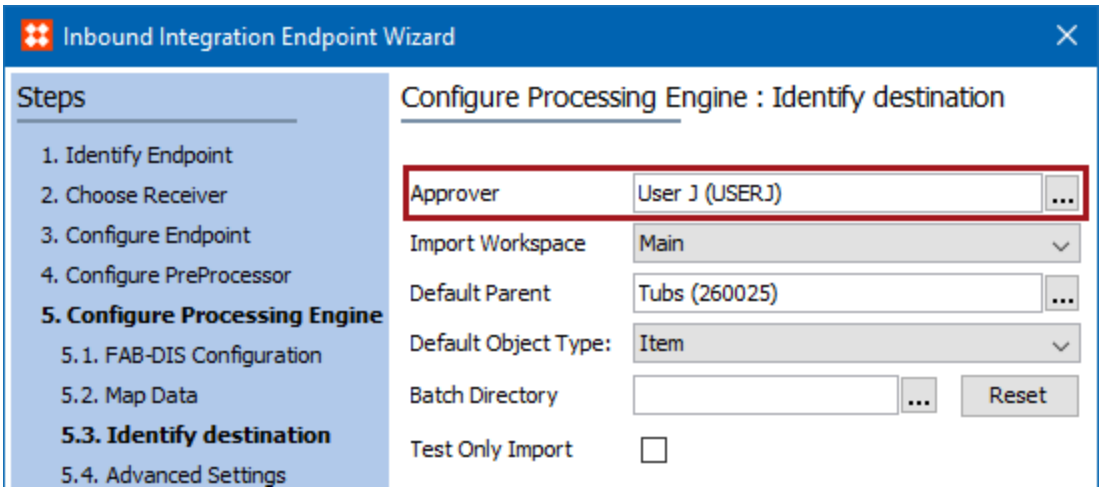


Considerations

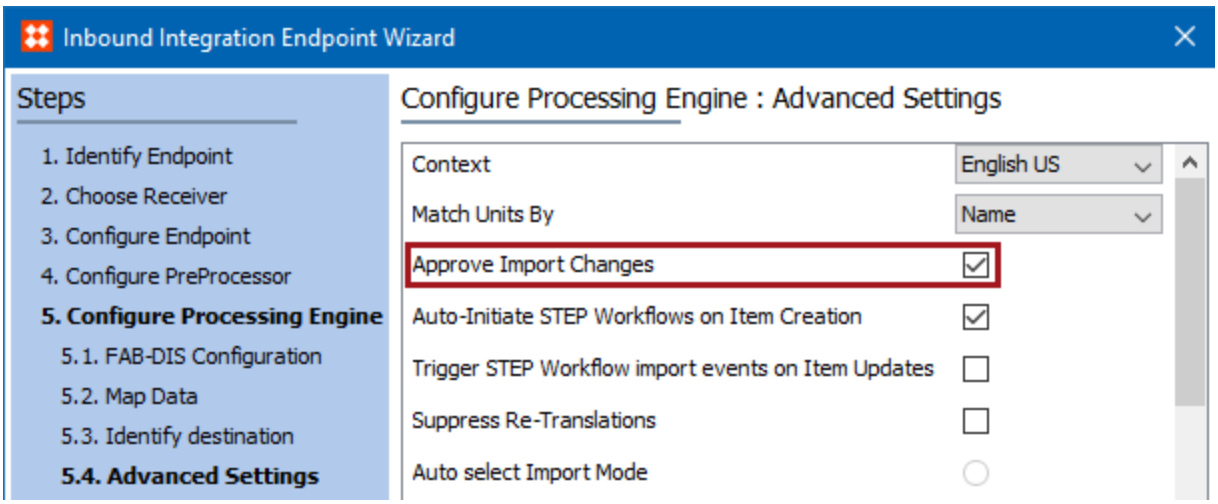
For the following scenarios, it is advised to use only one of these options, not both:

- 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, see 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, see 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, see 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

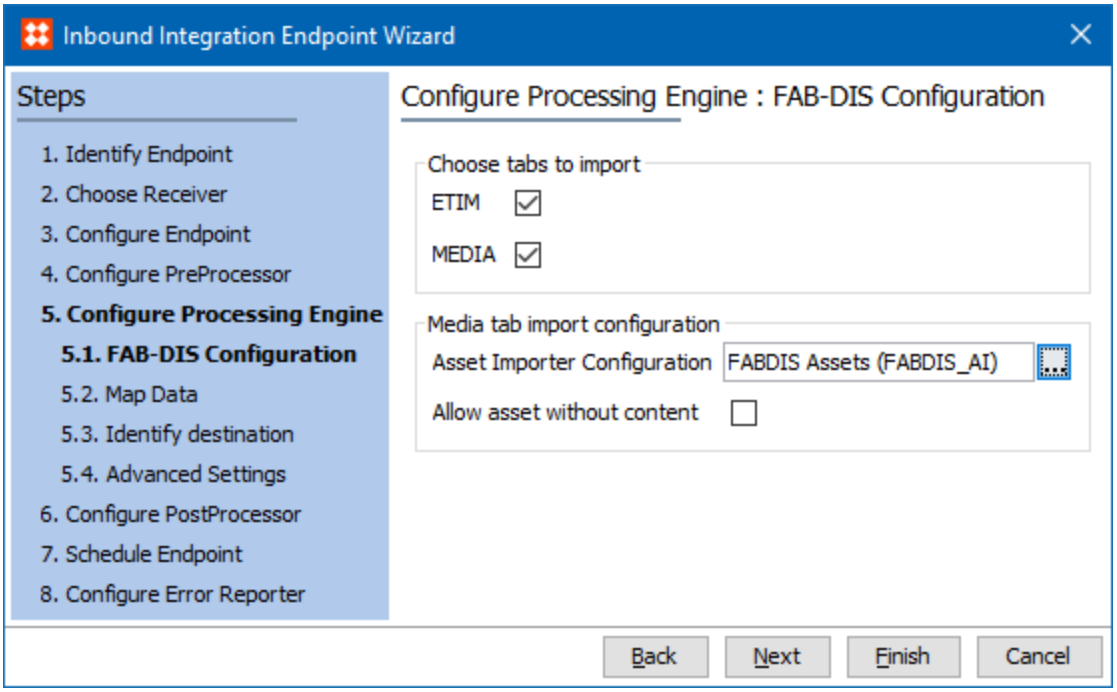
The full mapping process is addressed in the following sections of this topic:

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

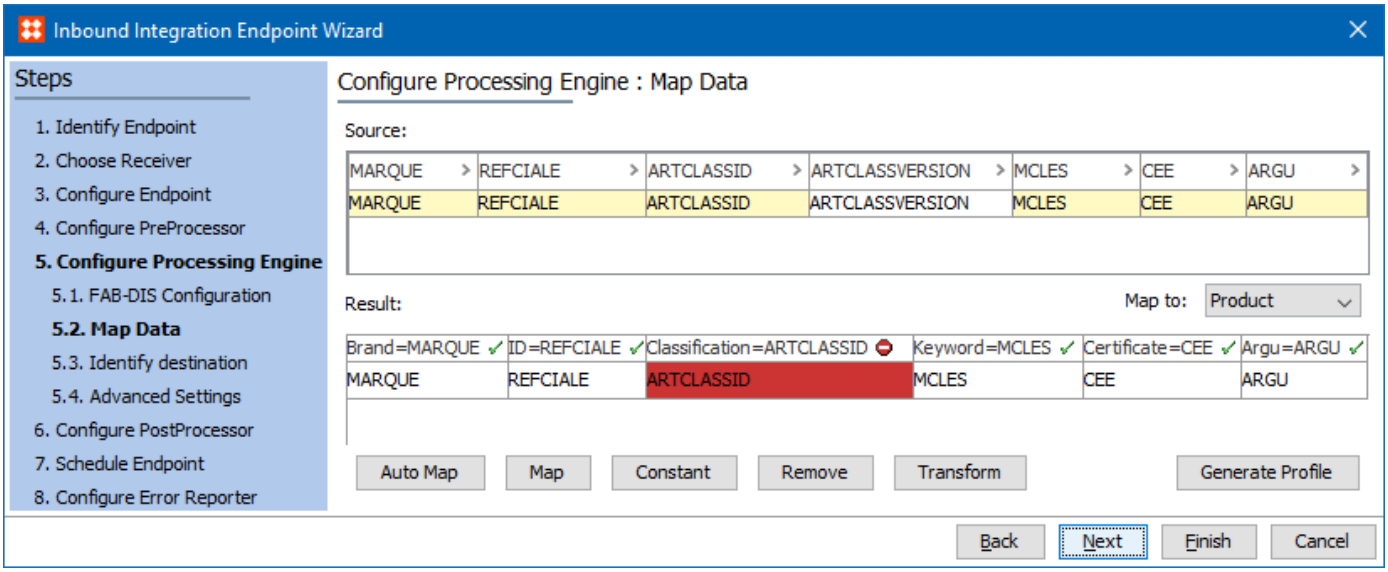
For an IIEP, the FAB-DIS Configuration step of the FAB-DIS Importer processing engine includes checkboxes for selecting the data to be imported. These checkboxes determine the mapping options available in the following Map Data step.



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, see the **Product to Classification Link Types** topic in **System Setup / Super User Guide** 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, see 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, see 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, see the **Media Mapping via Asset Importer Configuration** section below.

For details on all mapping options, see 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, see the **List of Values (LOVs)** topic in **System Setup / Super User Guide** 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, see 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, see the **Asset Importer Configuration** topic in the **Digital Assets** documentation.

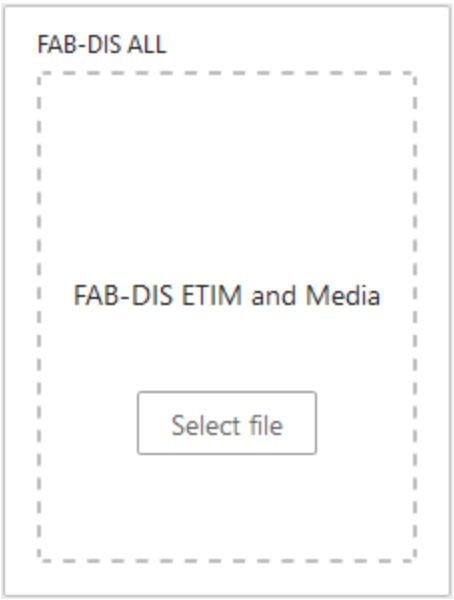
FAB-DIS Import Web UI Configuration

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, see 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, see 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, see 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.	Manually select a classification for the new product, or reimport with the new product's REFCIALE on the ETIM tab.

Error	Reason	Resolution
	The new product is created but cannot be added to a 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

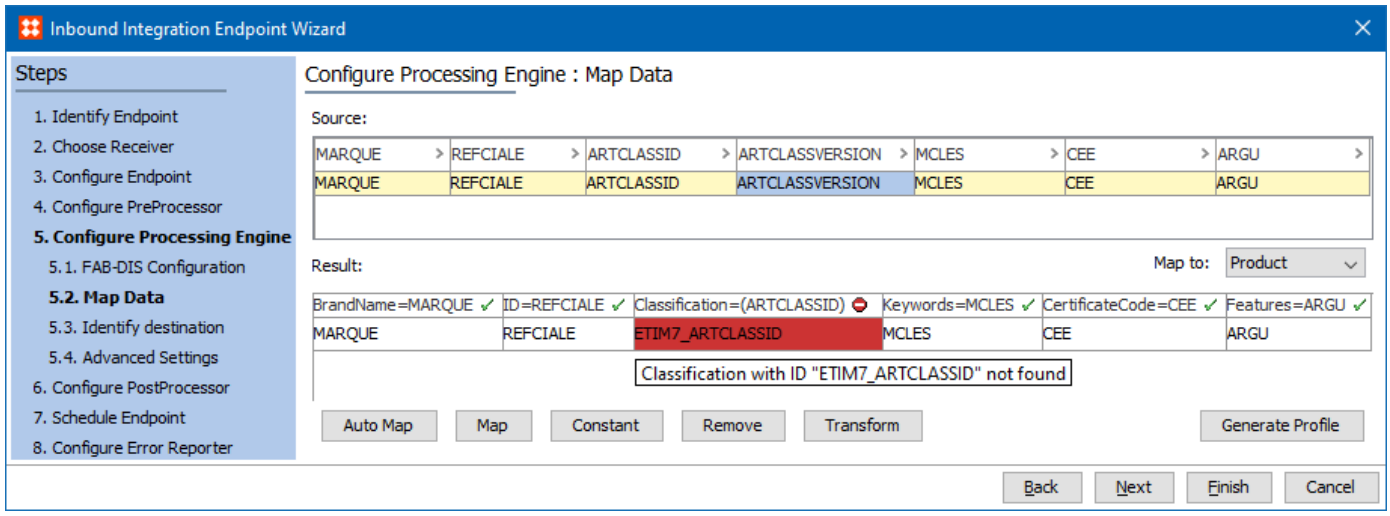
  'EFI08530' is the 'Hoisting load' attribute, and is of validation base type 'Number Range,' so two rows are used to supply the range values

'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'

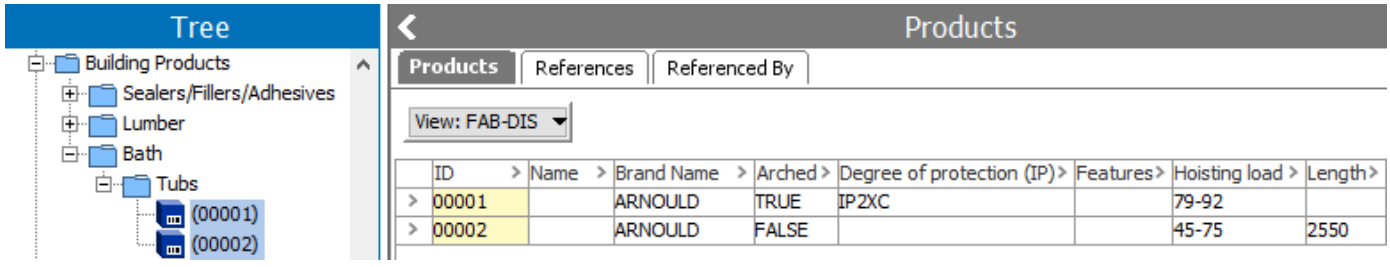
The TYPM values of MCLES, CEE, and ARGU indicate product data sources on the Media tab. For details, see the **Media Mapping** section below.

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.

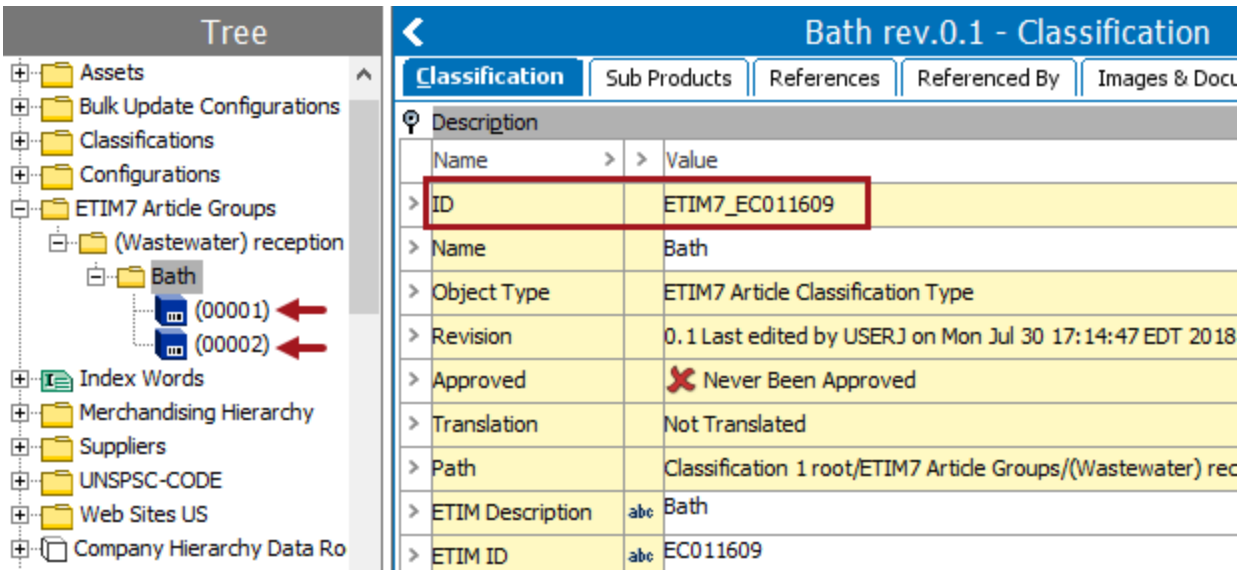


ETIM Result

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.

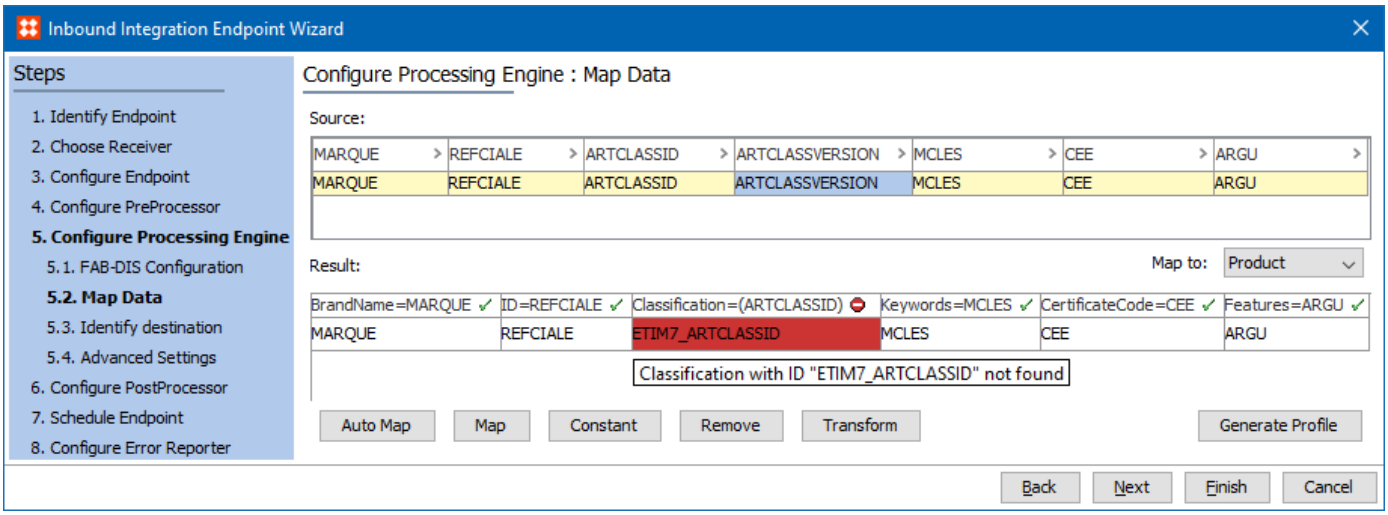
Important: The values in the TYPM column represent a variety of data that belongs to the product or the asset, such as ARGU or CMC. The CODVAL source is not mapped, but data from this column is used when the TYPM column values are MCLES, CEE, or ARGU.

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).

Tree

- Bath
  - Tubs
    - (00001)
    - (00002)
    - (00005)
    - (00006)
    - (00007)
    - (00008)

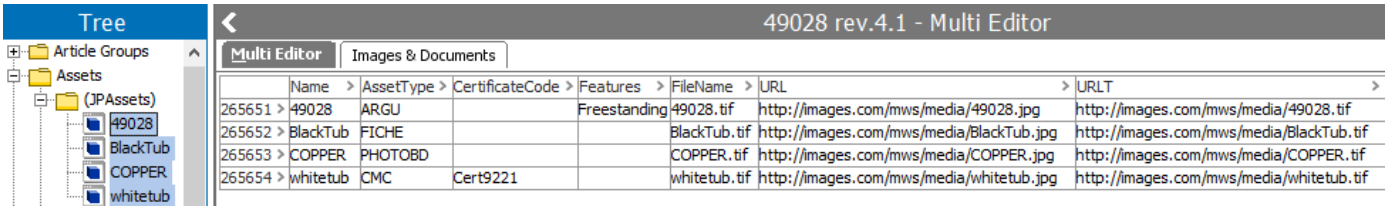
Products

Products | References | Referenced By

View: FAB-DIS

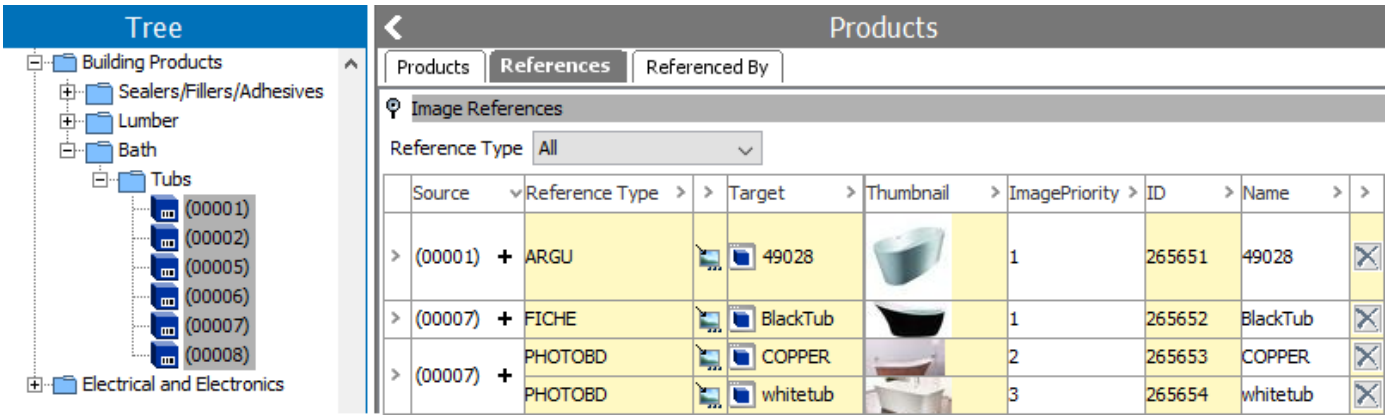
ID	Name	Brand Name	Features	Keywords	Arched	CertificateCode	Degree of protection (IP)	Hoisting load	Length
00001		ARNOULD			TRUE		IP2XC	79-92	
00002		ARNOULD			FALSE			45-75	2550
00005		ARNOULD		KW1 | KW2					
00006		ARNOULD				Cert7846			
00007		ARNOULD							
00008		ARNOULD	Jet						

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 - see **Creating an Inbound Integration Endpoint**
- Import Manager - see **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, see **Data Mapping**.

Inbound Data

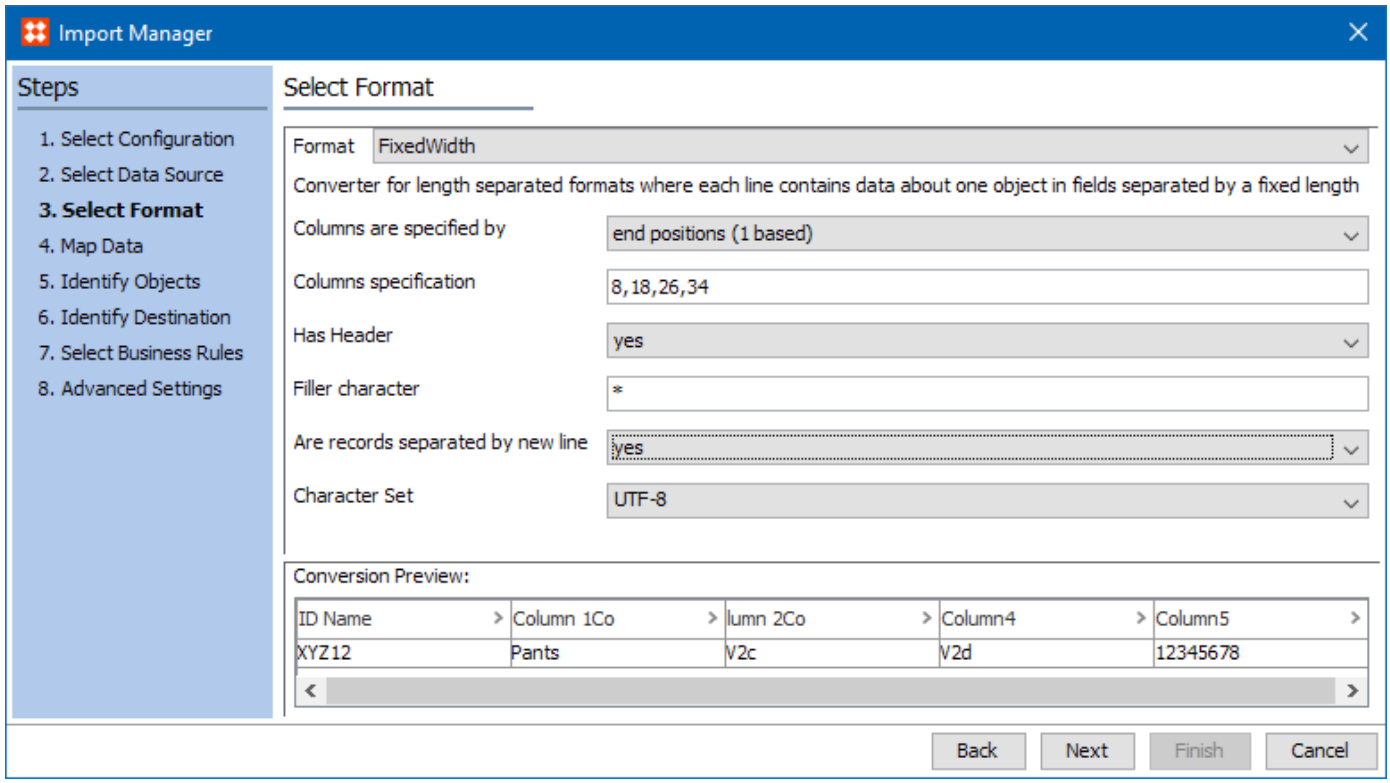
For examples of ways to modify fixed width data for import and further explanations on the inbound parameters, see **FixedWidth Format Configuration Examples**.

Inbound Parameters

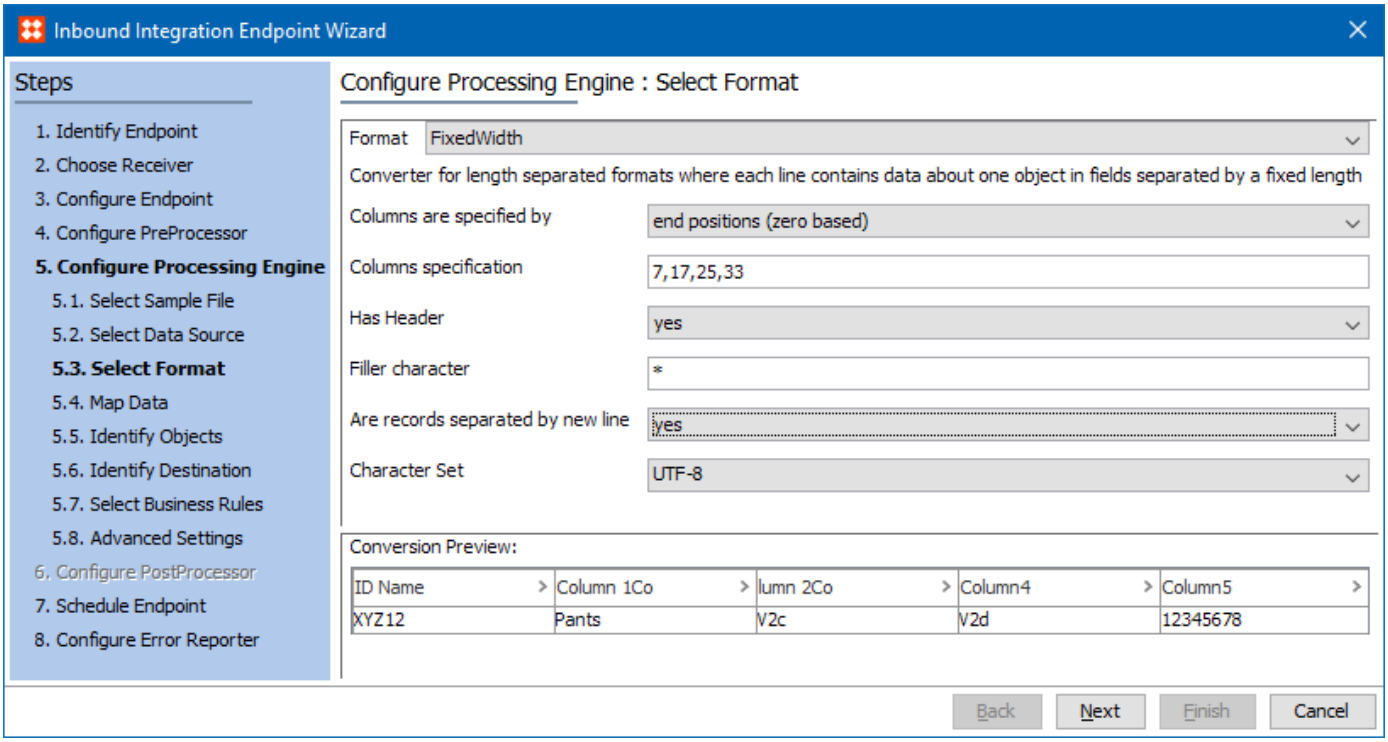
- **Columns are specified by** allows you to determine how the initial element is identified via the following options:
  - **end positions (zero-based)** - the last character counted is counted with a 'zero' value.
  - **end positions (1 based)** - every character is counted and indicates the last character 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



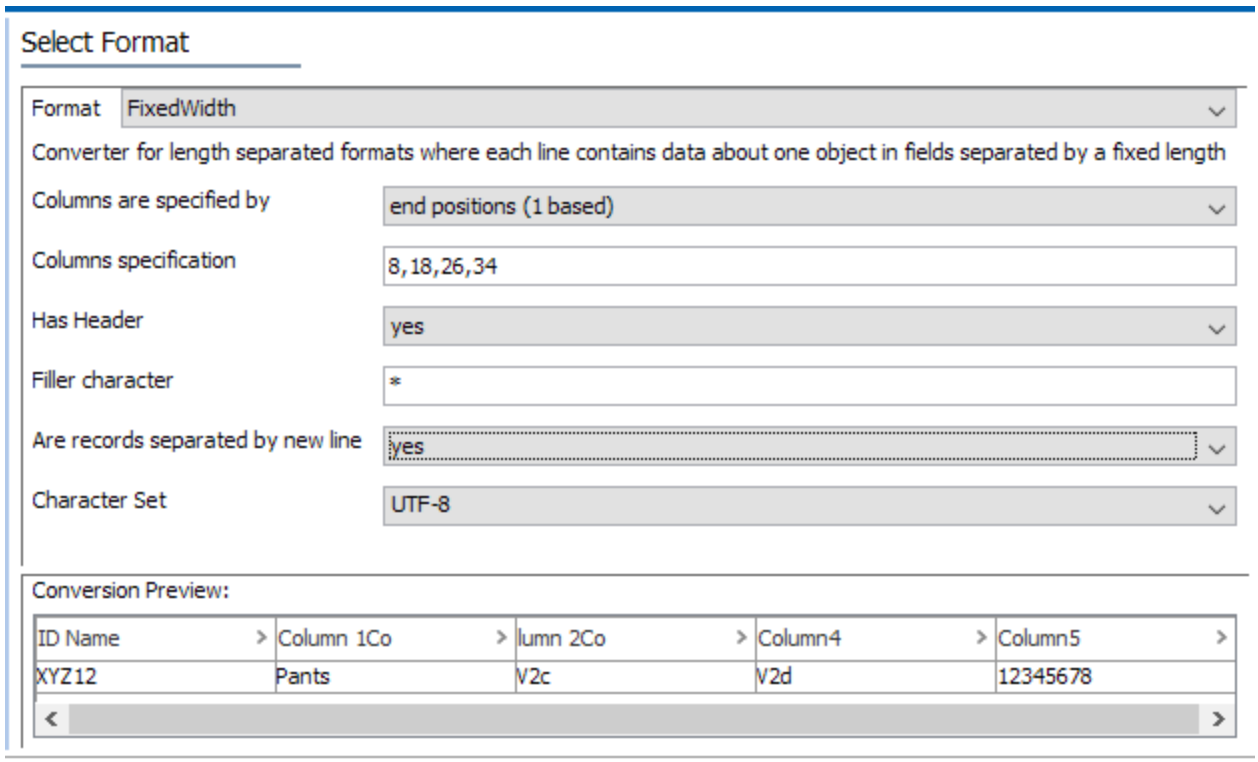
IIEP


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. Configure Error Reporter
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, see the **FixedWidth Format** documentation.



Example Data

Each of the examples below use the following input file.

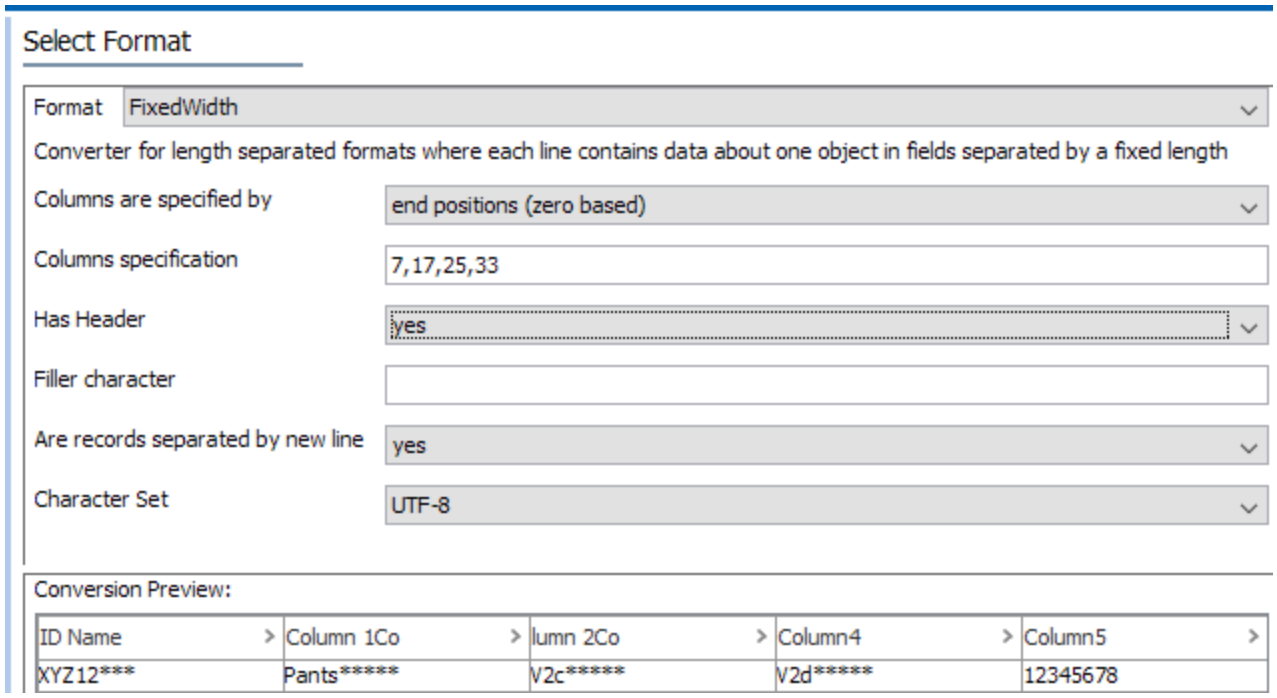
```
ID      Name    Column  1Column 2Column 3
XYZ12***Pants*****V2c*****V2d*****12345678
```

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 it's 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.

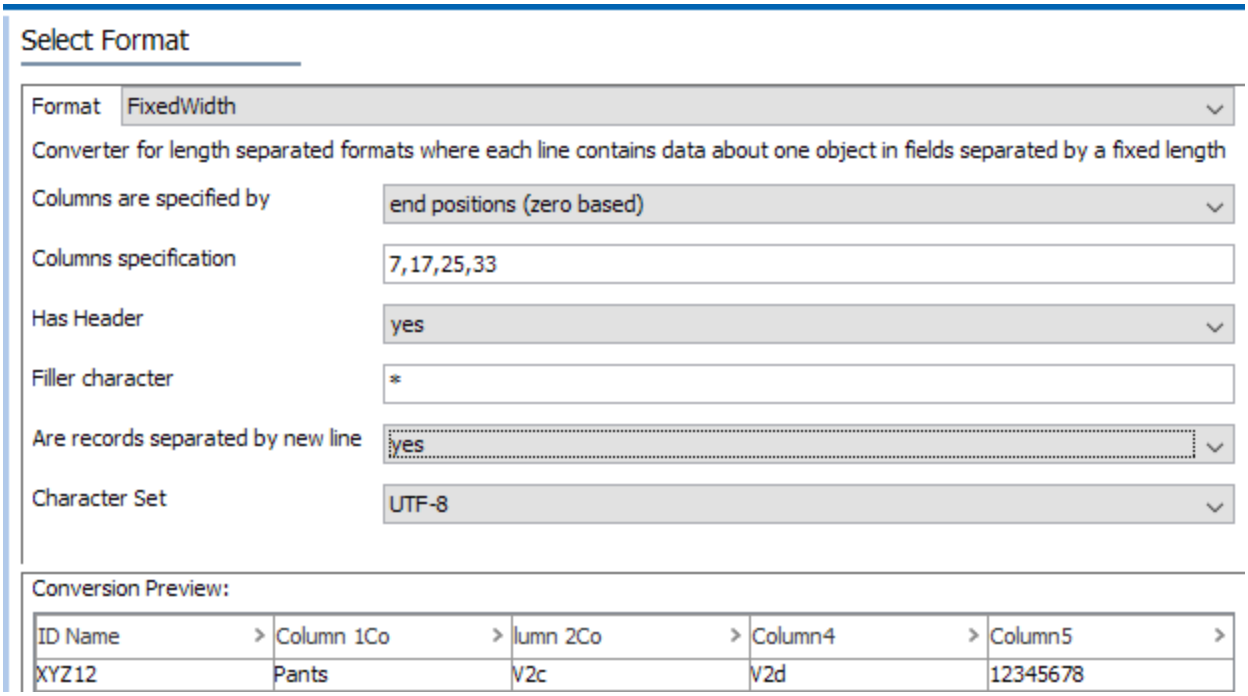
1. **Columns are Specified by** is set to **end positions (zero based)**.
2. **Columns Specification** is set to **7,17,25,33**.
3. **Has Header** is set to **Yes**.
4. **Are records separated by new line** is set to **Yes**.
5. **Character set** is set to **UTF-8**.

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.

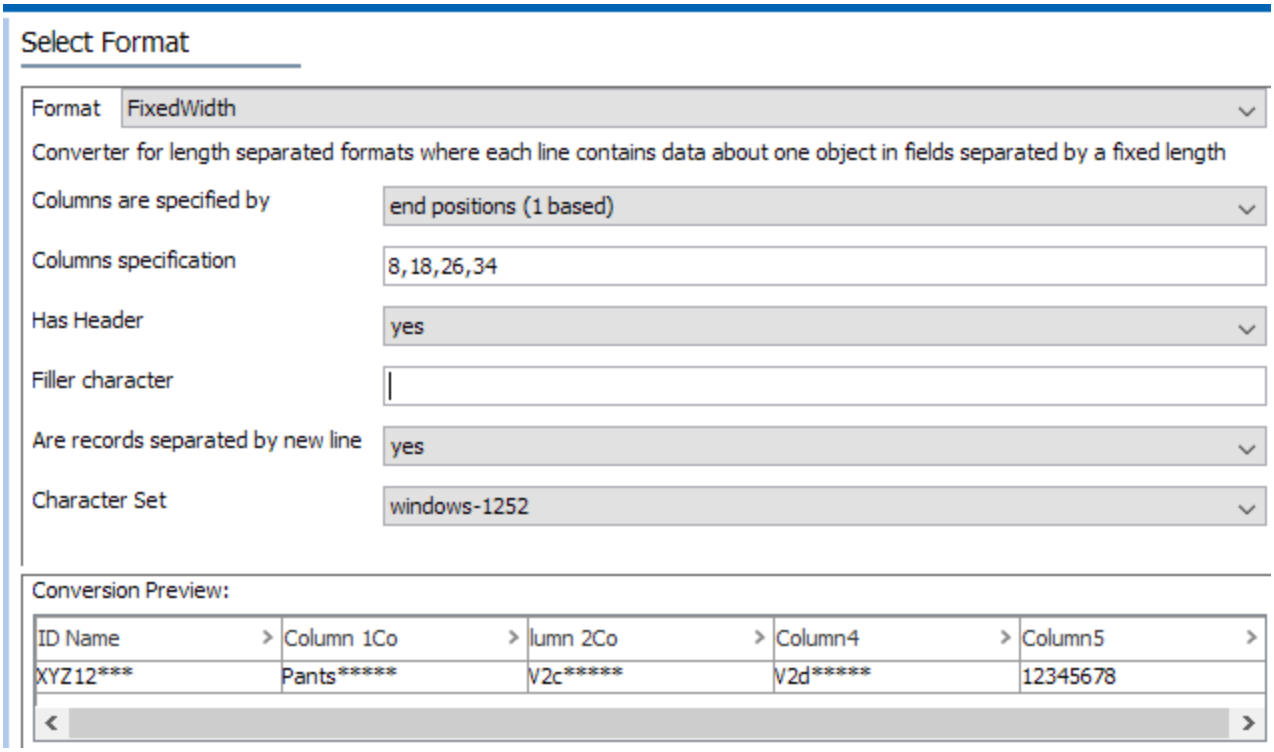


End Positions (One Based) Example

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.

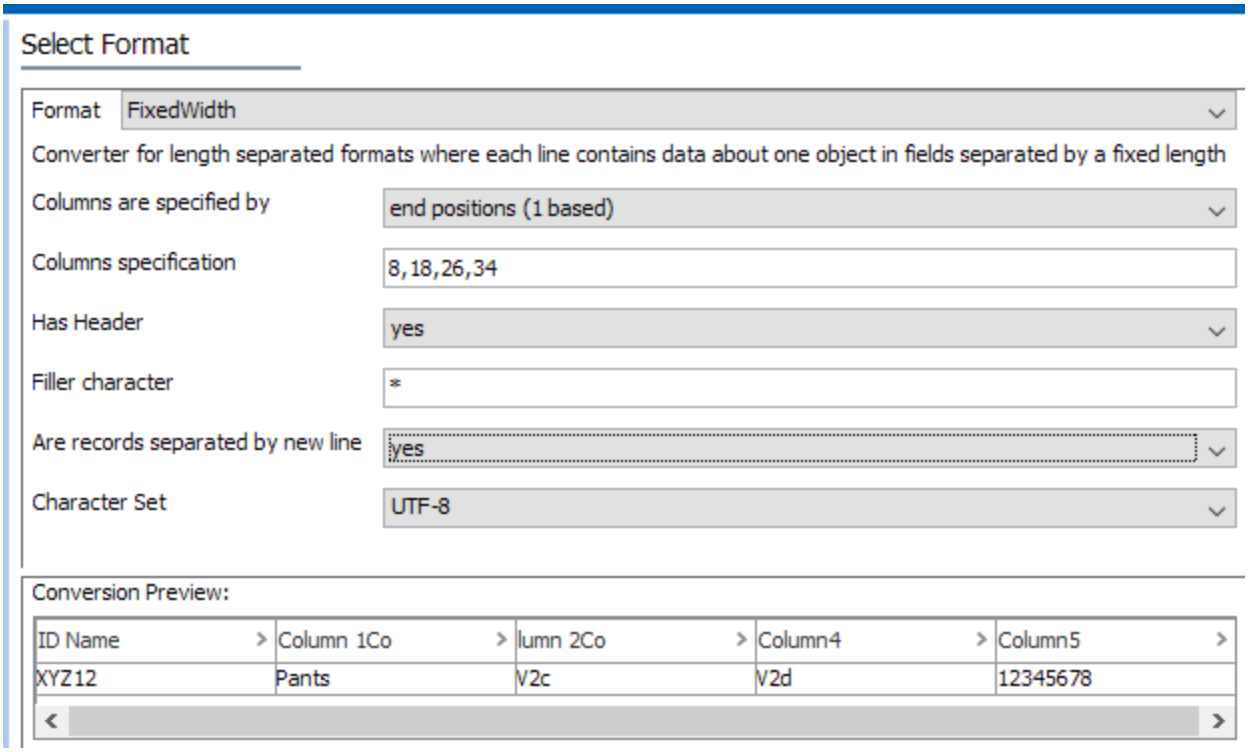
1. **Columns are Specified by** is set to **end positions (1 based)**.
2. **Column specification** is set to **8,18,26,34**.
3. **Has Header** is set to **Yes**.
4. **Are records separated by new lines** is set to **Yes**.
5. **Character set** is set to **UTF-8**.

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.



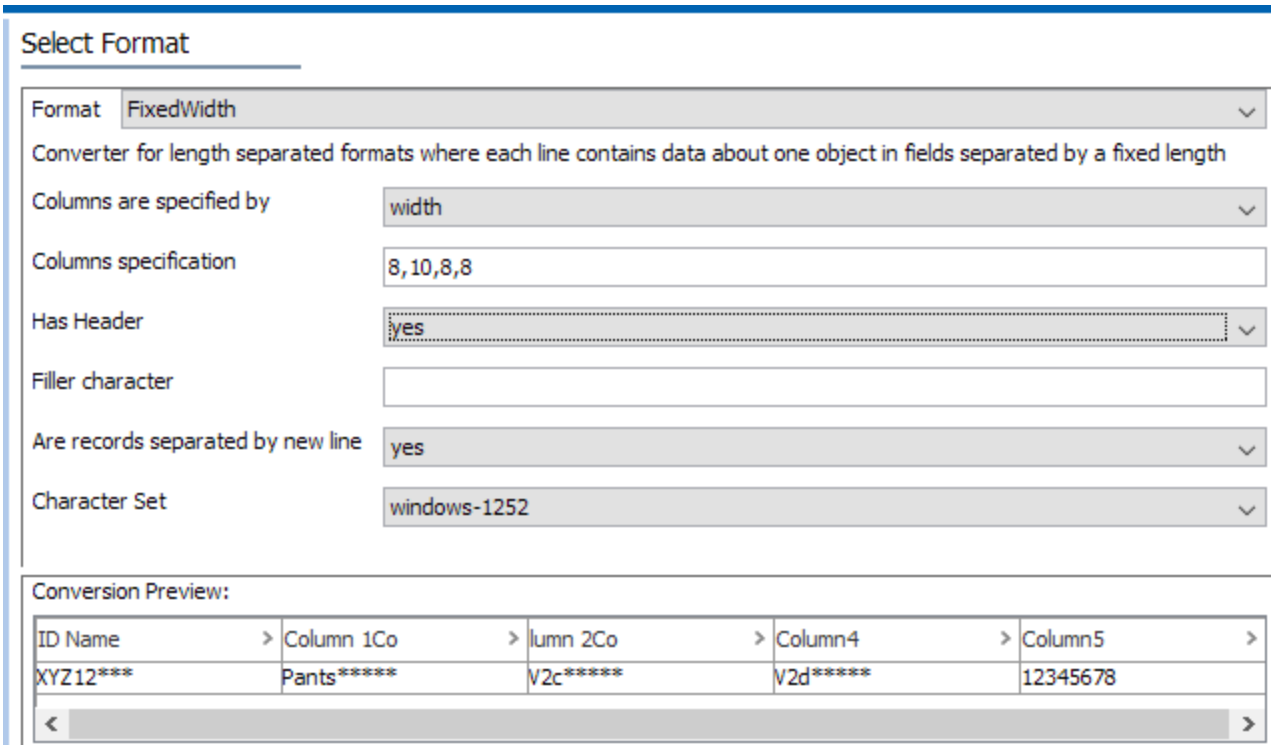
Width Example

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.

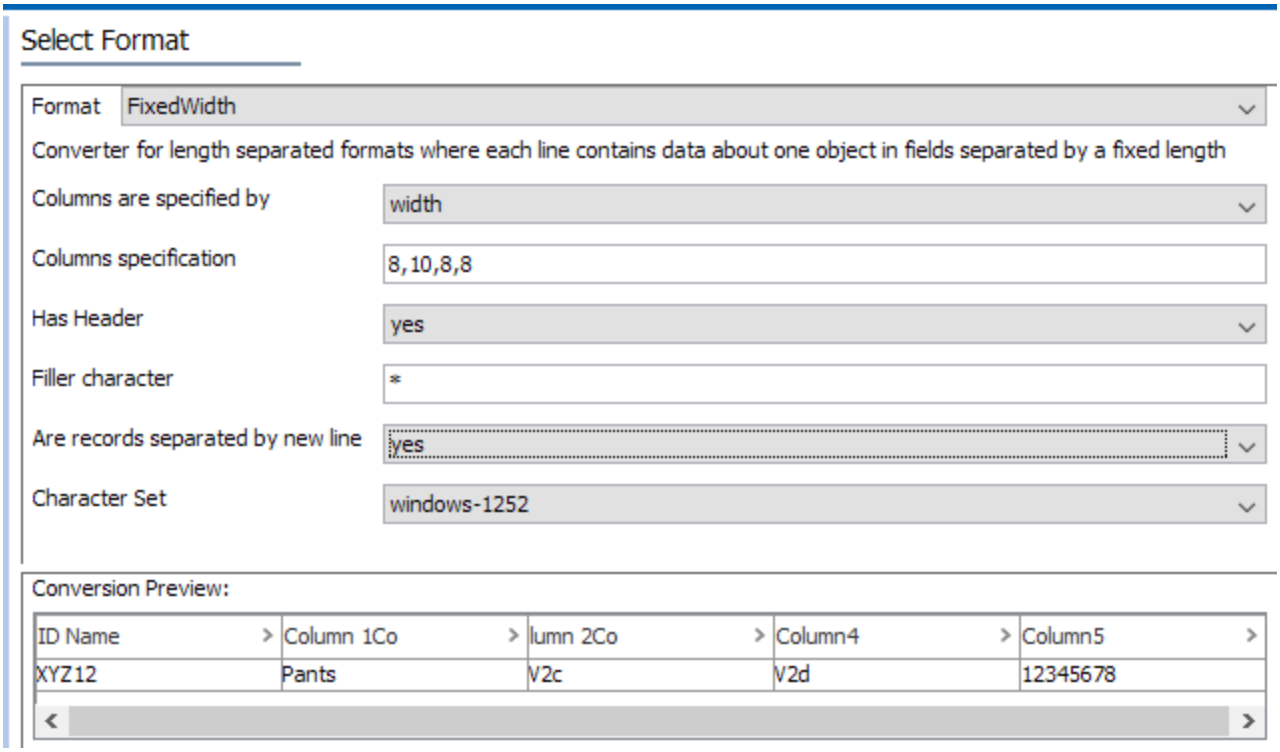
1. **Columns are Specified by** is set to **width**.
2. **Column specification** is set to **8,10,8,8**.
3. **Has Header** is set to **Yes**.
4. **Are records separated by new lines** is set to **Yes**.
5. **Character set** is set to **UTF-8**.

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 - see **Creating an Inbound Integration Endpoint**
- Import Manager - see **Creating a Data Import**
- Export Manager - see **Creating a Data Export**

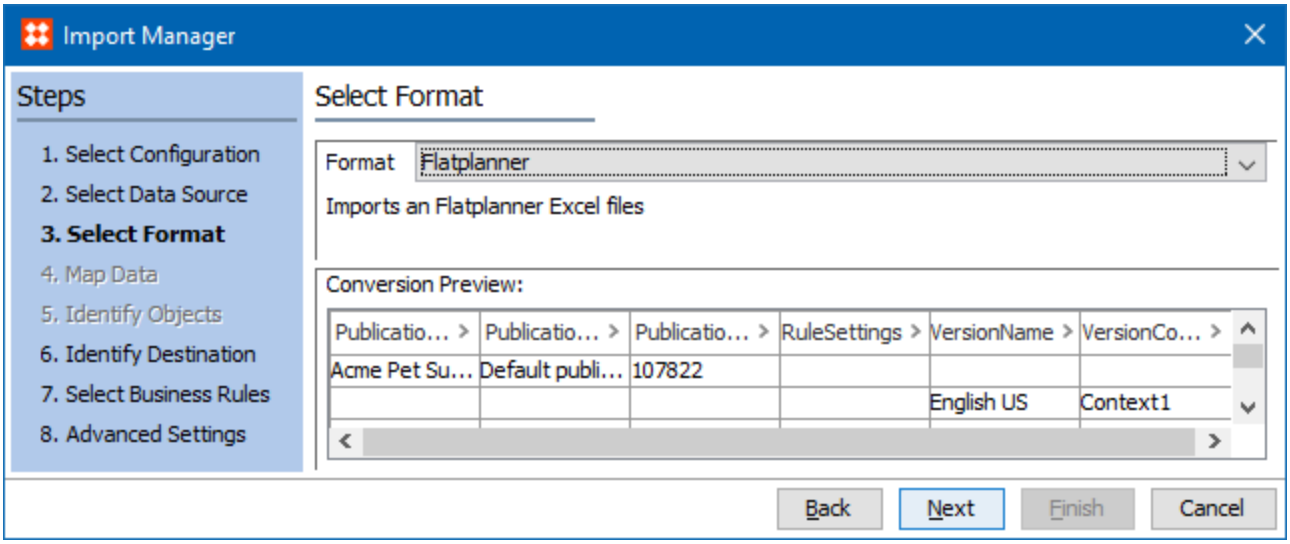
Mapping

Mapping data is not required and the Map Data step is disabled.

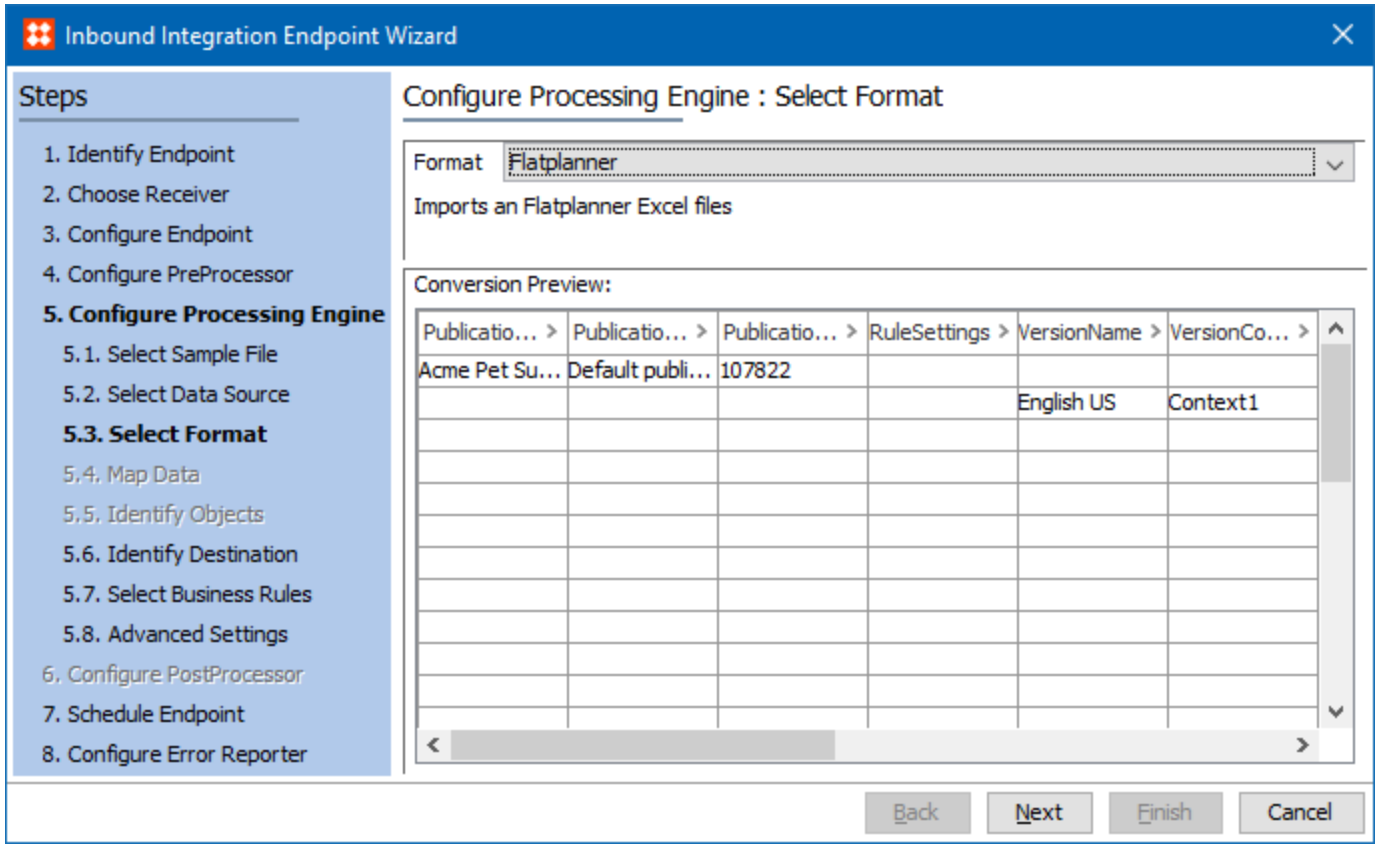
Inbound Data

The Flatplan Excel format is displayed as 'Flatplanner' format for inbound data.

Import Manager



IIEP

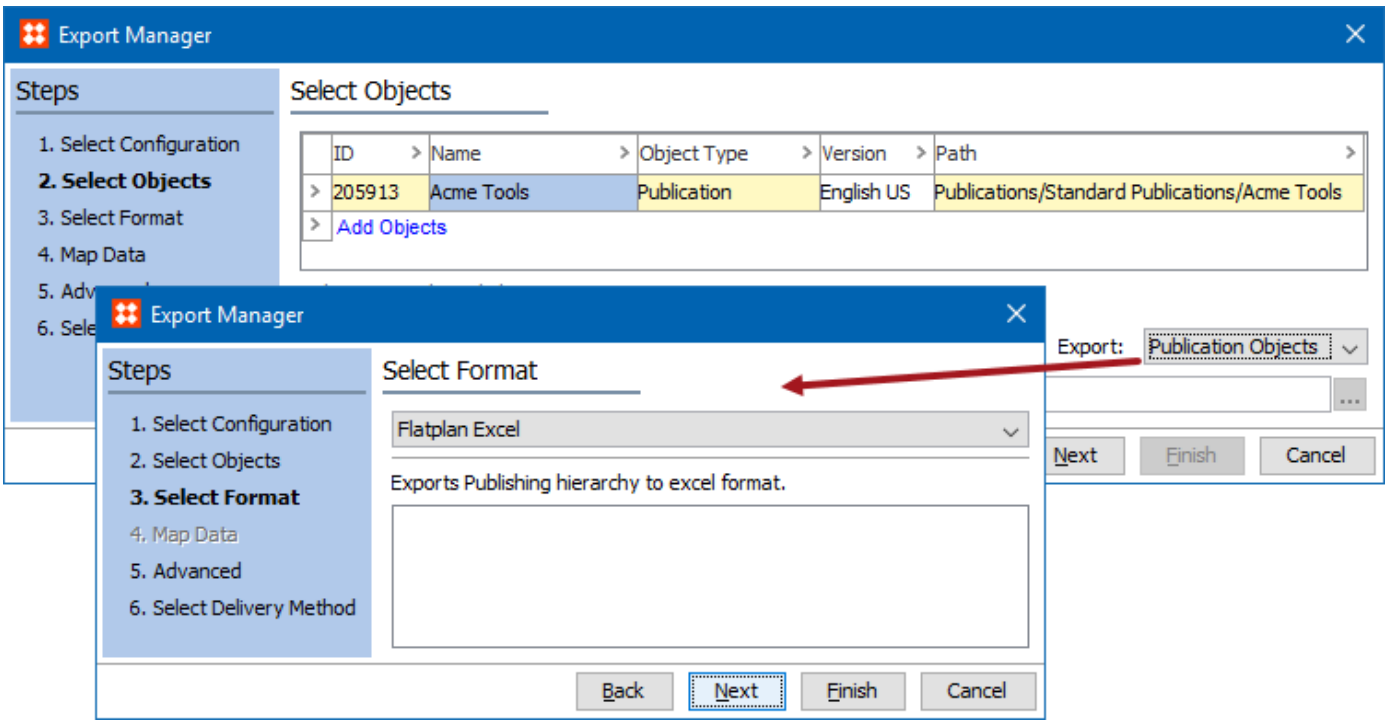


Outbound Data

For step-by-step instructions on how to export a Publication Excel or Flatplan Excel spreadsheet from a Flatplanner publication in STEP, see **Exporting a Publication Excel Sheet** in the **STEP Publisher** documentation.

Export Manager

After selecting a publication on Tree, right-click and choose **Export Data Below** to see 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 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 - see **Creating an Inbound Integration Endpoint**
- Import Manager - see **Creating a Data Import**
- Export Manager - see **Creating a Data Export**
- OIEP - see **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, see **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:

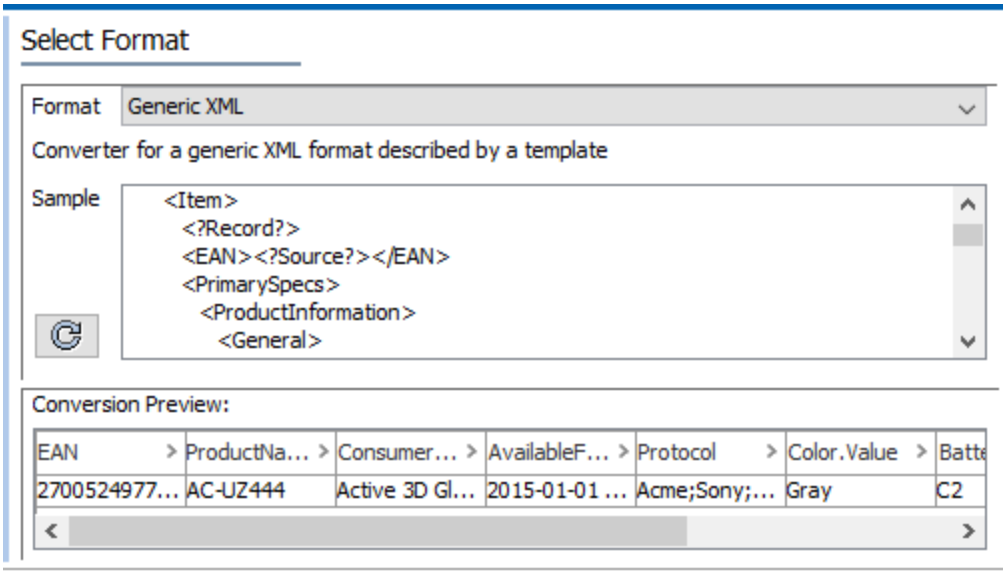
```
<Products>
   <Product>
      <Name/>
   </Product>
</Products>
```

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>.

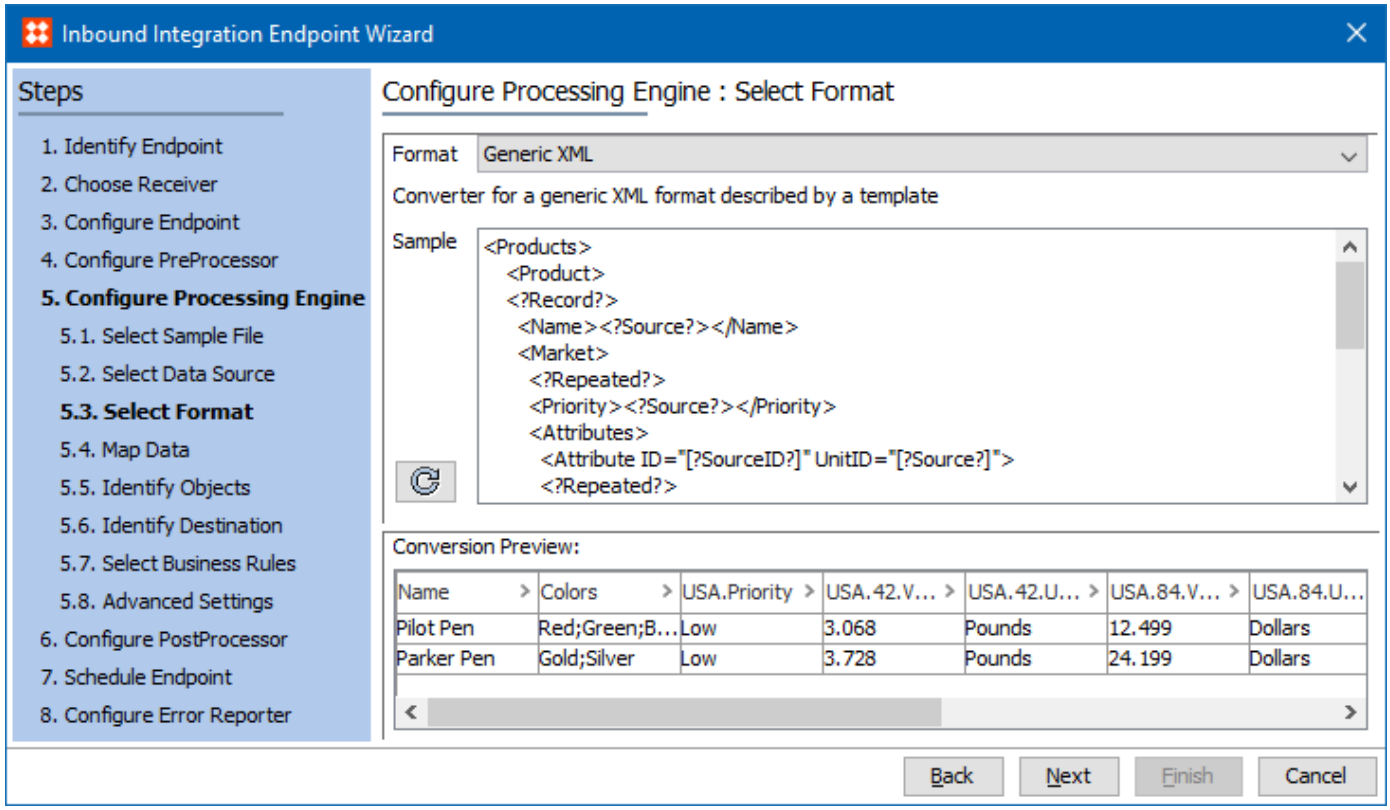
For details about the available processing instructions, see the **Generic XML Inbound Processing Instructions** documentation.

For details about mapping the processing instructions to STEP data, see **Inbound Map Data Options** documentation.

Import Manager



IIEP



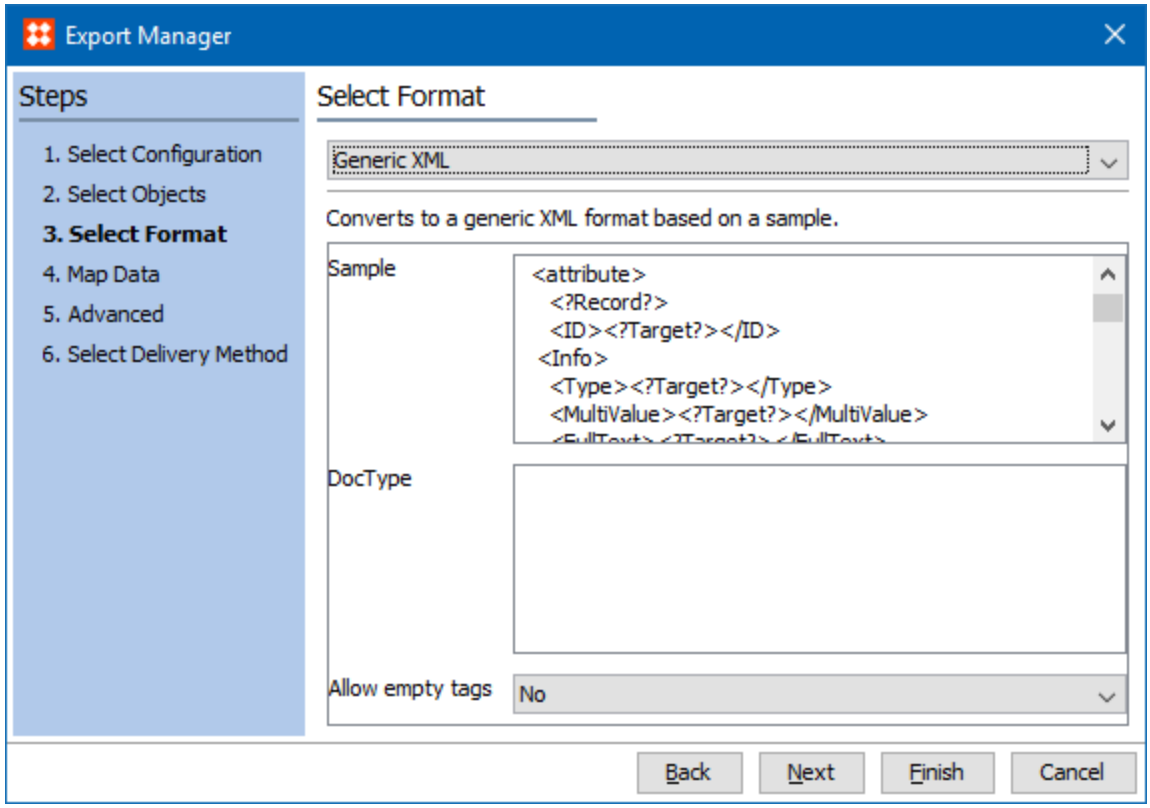
Outbound Data

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, see **Outbound Map Data Options** documentation.

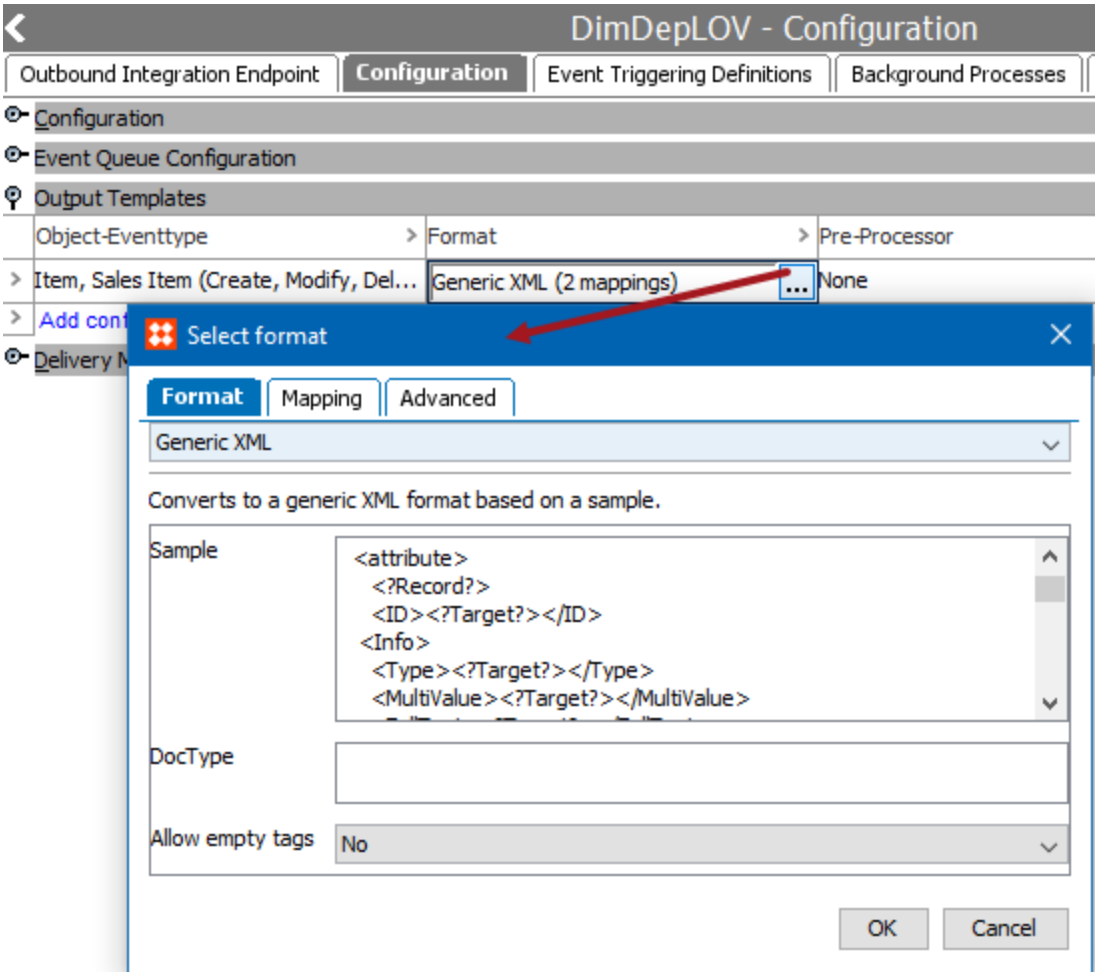
Since no undo functionality is available in the Template field, it is good practice to use an external XML editor program for creating and editing a template. To test a template's validity for output, paste the XML into STEP.

For details about the available processing instructions, some of which are only available in an OIEP, see 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 Instructions 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, see the following sections:

- Generic XML Import Simple 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. See 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, see the **Preventing Duplicate Field Names** section of the **Source Processing Instructions in Generic XML** documentation.

Record Processing Instructions 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>
```

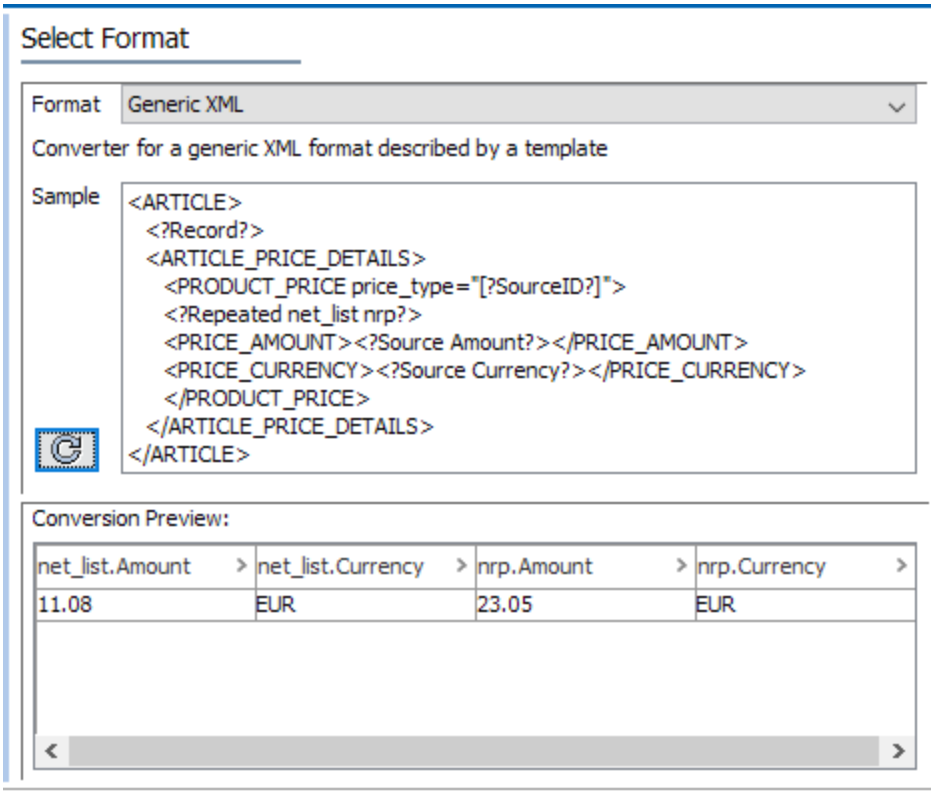
```
    <PRODUCT_PRICE price_type="net_list">
       <PRICE_AMOUNT>11.08</PRICE_AMOUNT>
       <PRICE_CURRENCY>EUR</PRICE_CURRENCY>
       <TAX>0.19</TAX>
       <LOWER_BOUND>1</LOWER_BOUND>
    </PRODUCT_PRICE>
    <PRODUCT_PRICE price_type="nrp">
       <PRICE_AMOUNT>23.05</PRICE_AMOUNT>
       <PRICE_CURRENCY>EUR</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>
```

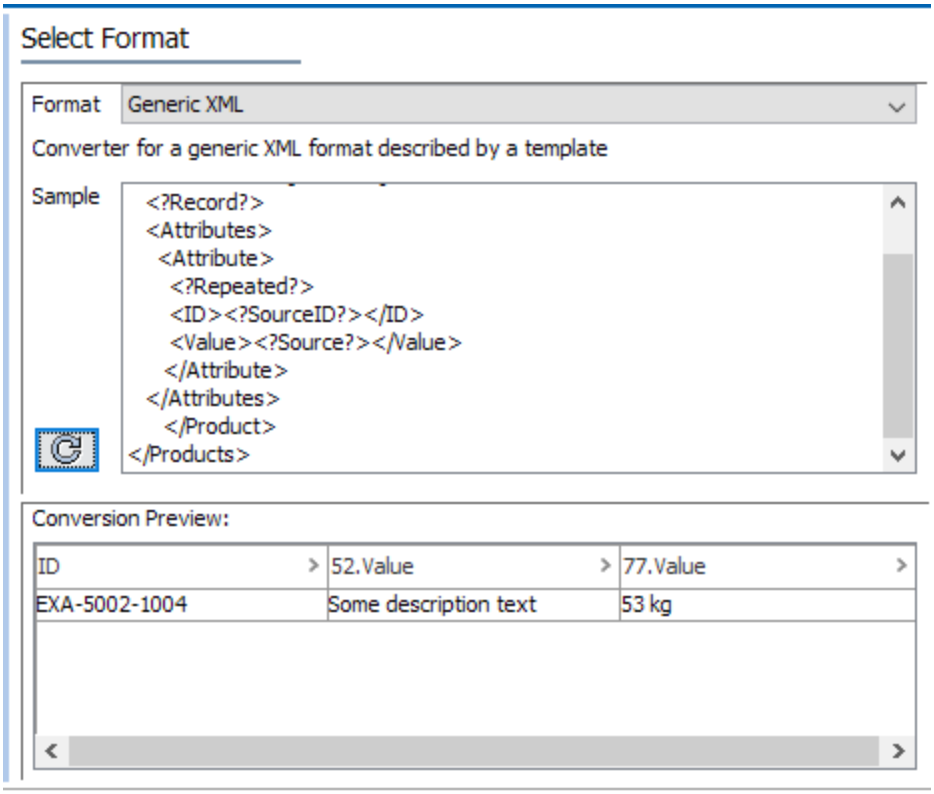
When used on the input document, this template outputs the fields 'ID', 'Value-52', 'Value-77.' In this case, the subtree from 'Value' and down is said to be in repeated scope.

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 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.'

```
<Products>
   <Product>
      <Name>Pilot Pen</Name>
      <Market>
         <Priority>Low</Priority>
         <Attributes>
            <Attribute ID="42" UnitID="Pounds">3.068</Attribute>
            <Attribute ID="84" UnitID="Dollars">12.499</Attribute>
         </Attributes>
         <MarketID>USA</MarketID>
      </Market>
      <Market>
         <Priority>High</Priority>
         <Attributes>
            <Attribute ID="42" UnitID="Kg">1534</Attribute>
            <Attribute ID="84" UnitID="Euro">22.995</Attribute>
        </Attributes>
      <MarketID>EU</MarketID>
      </Market>
      <Colors>
         <Color>Red</Color>
         <Color>Green</Color>
         <Color>Blue</Color>
      </Colors>
    </Product>
    <Product>
        <Name>Parker Pen</Name>
        <Market>
```

```
            <Priority>Low</Priority>
          <Attributes>
                  <Attribute ID="42" UnitID="Pounds">3.728</Attribute>
               <Attribute ID="84" UnitID="Dollars">24.199</Attribute>
          </Attributes>
          <MarketID>USA</MarketID>
         </Market>
         <Market>
              <Priority>Extreme</Priority>
              <Attributes>
                 <Attribute ID="42" UnitID="Kg">1864</Attribute>
                 <Attribute ID="84" UnitID="Euro">29.999</Attribute>
              </Attributes>
        <MarketID>EU</MarketID>
        </Market>
        <Colors>
           <Color>Gold</Color>
           <Color>Silver</Color>
        </Colors>
      </Product>
  </Products>
```

Template

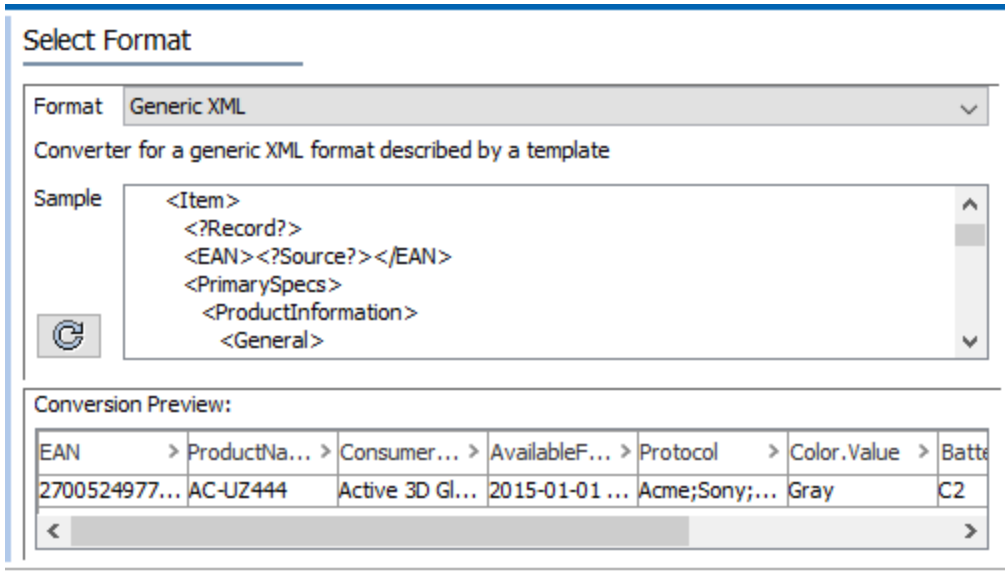
Copy the following text and paste it in the Import Manager Select Format step, Sample field.

```
  <Products>
      <Product>
      <?Record?>
        <Name><?Source?></Name>
        <Market>
          <?Repeated?>
          <Priority><?Source?></Priority>
          <Attributes>
            <Attribute ID="[?SourceID?]" UnitID="[?Source?]">
            <?Repeated?>
            <?Source Value?>
```

```
            </Attribute>
          </Attributes>
        <MarketID><?SourceID?></MarketID>
          </Market>
          <Colors>
            <Color><?MultiSource Colors?></Color>
          </Colors>
        </Product>
    </Products>
```

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, see the **Inbound Map Data - Transform** documentation.



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>
```

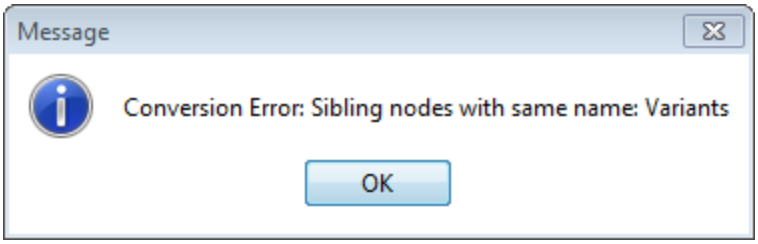
Creating a Template

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:



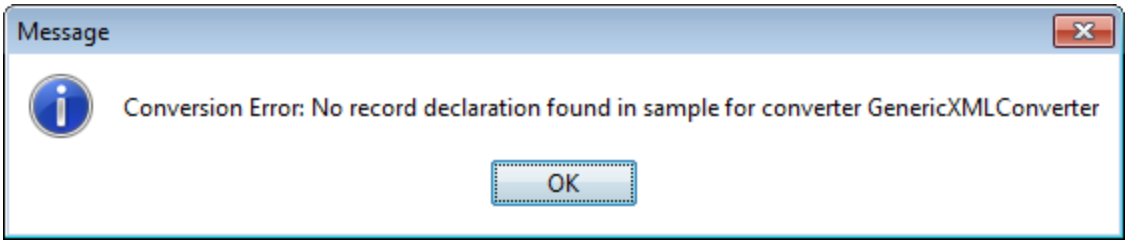
Remove of repetitions of the 'Variants', 'Manual', and 'Protocol' elements, until only one of each remains. The Sample field now displays the following:

```
   <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>
```

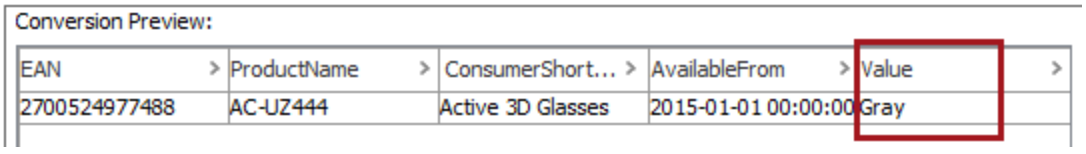
```
            <PrimarySpecs>
               <ProductInformation>
                  <General>
                     <ProductName><?Source?></ProductName>

<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 see 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>
  <Value><?Source?></Value>
</Variants>
```



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.

```
<Variants>
    <Name>Color</Name>
    <Value><?MultiSource?></Value>
</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

```
<Variants>
    <?Repeated?>
    <Name><?SourceID?></Name>
    <Value><?Source?></Value>
</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

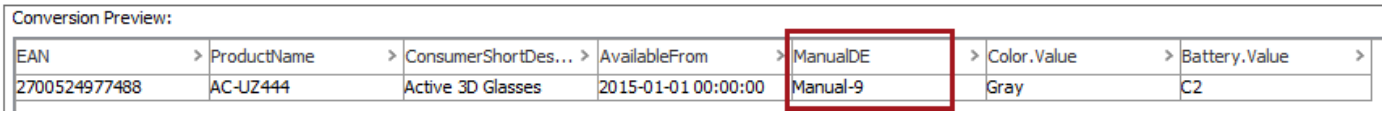
To see how each option affects the outcome, make the following updates in the Sample template and then click the Sample reload button to update the Conversion Preview.

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>
```

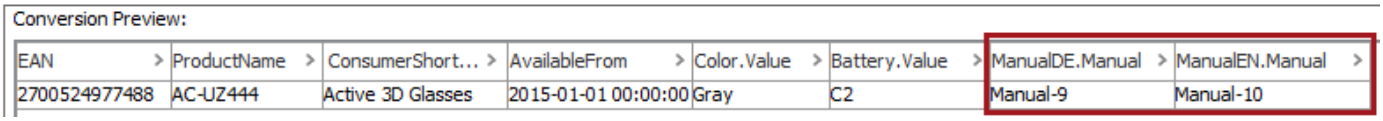


Conversion Preview:

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>
```

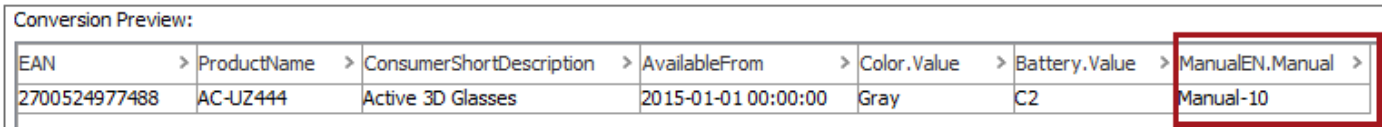


Conversion Preview:

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>
```



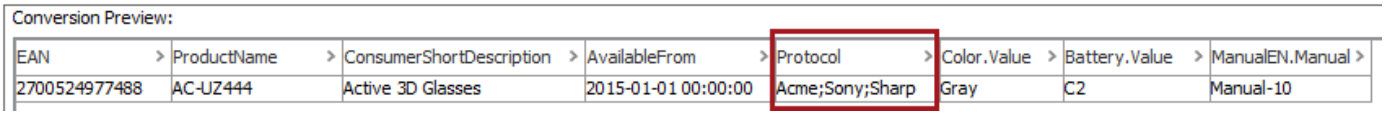
Conversion Preview:

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
- 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, see the **Generic XML Sample Export** section.

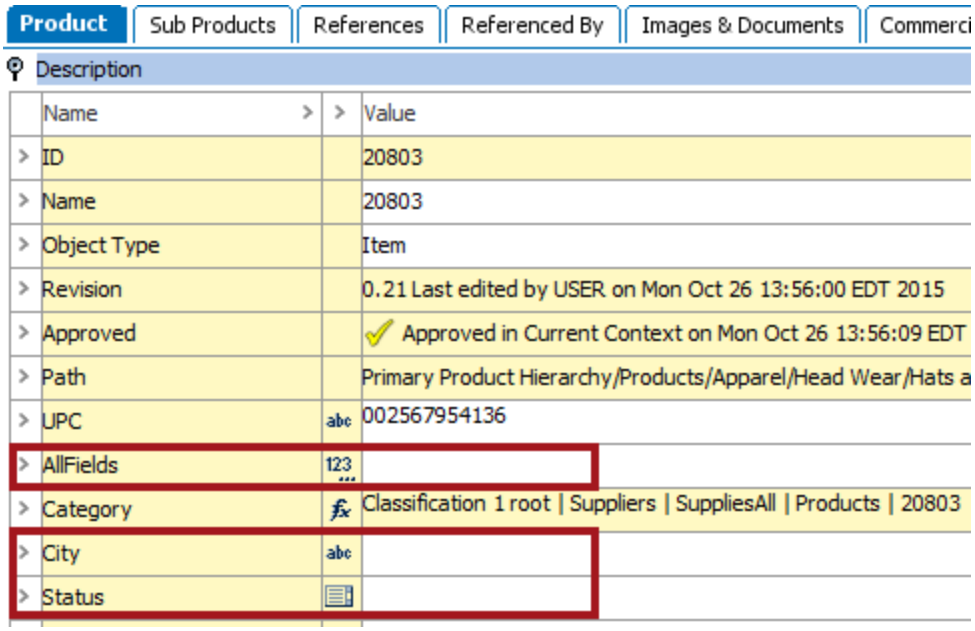
Allow Empty Tags Parameter in Generic XML

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, see the **RemoveIfEmpty Processing Instructions in Generic XML** section.

Objects

The object selected for export includes attributes without values.



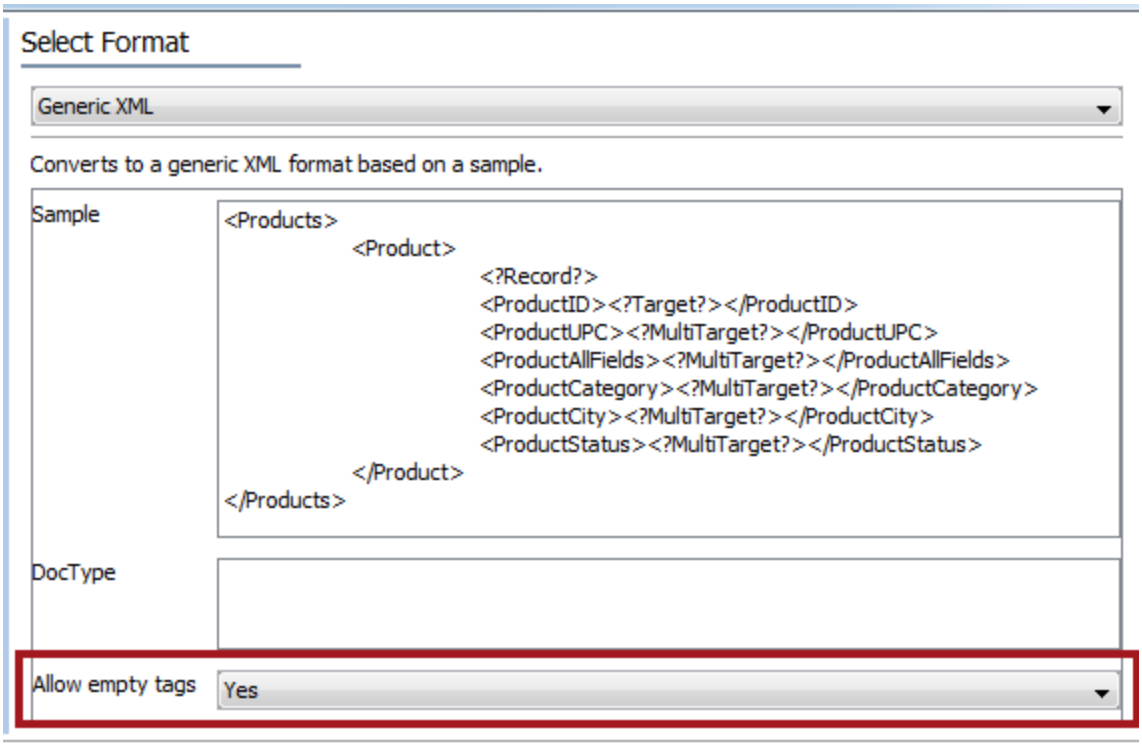
Template

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>
```

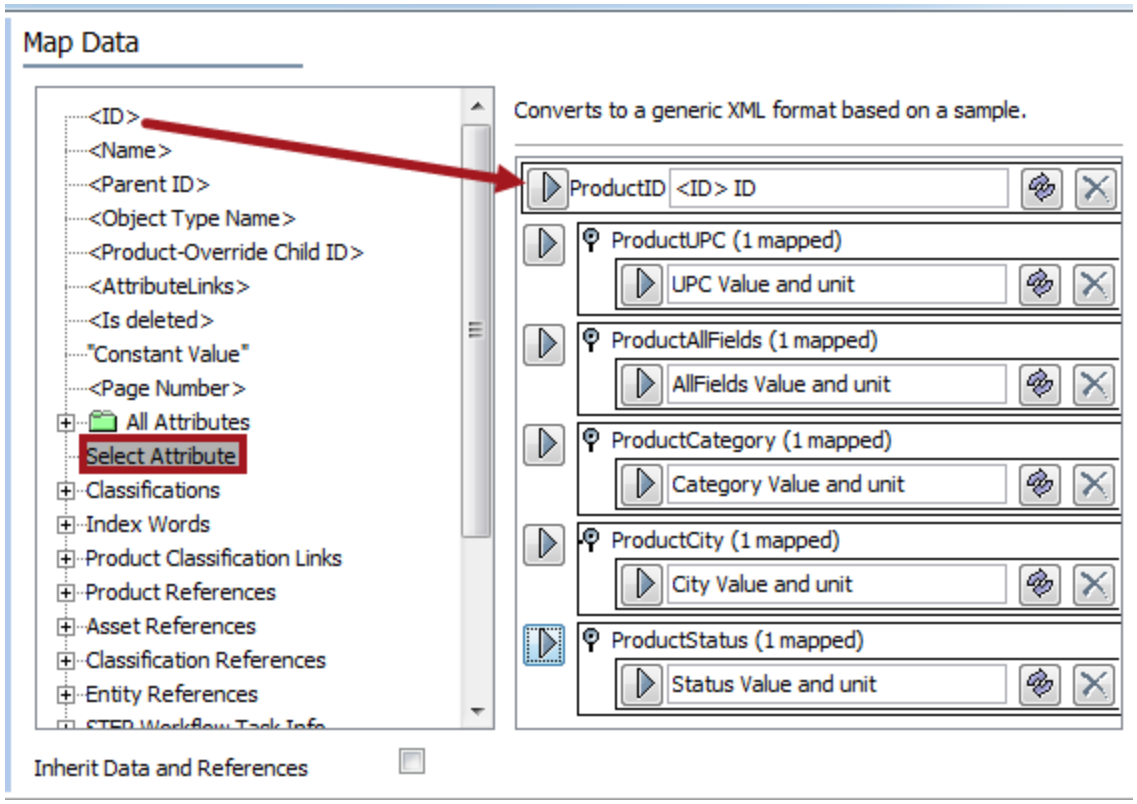
```
    </Product>
</Products>
```

Set **Allow empty tags** parameter to **Yes** if tags without values should be output or set to **No** if empty tags should be eliminated.



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>
```

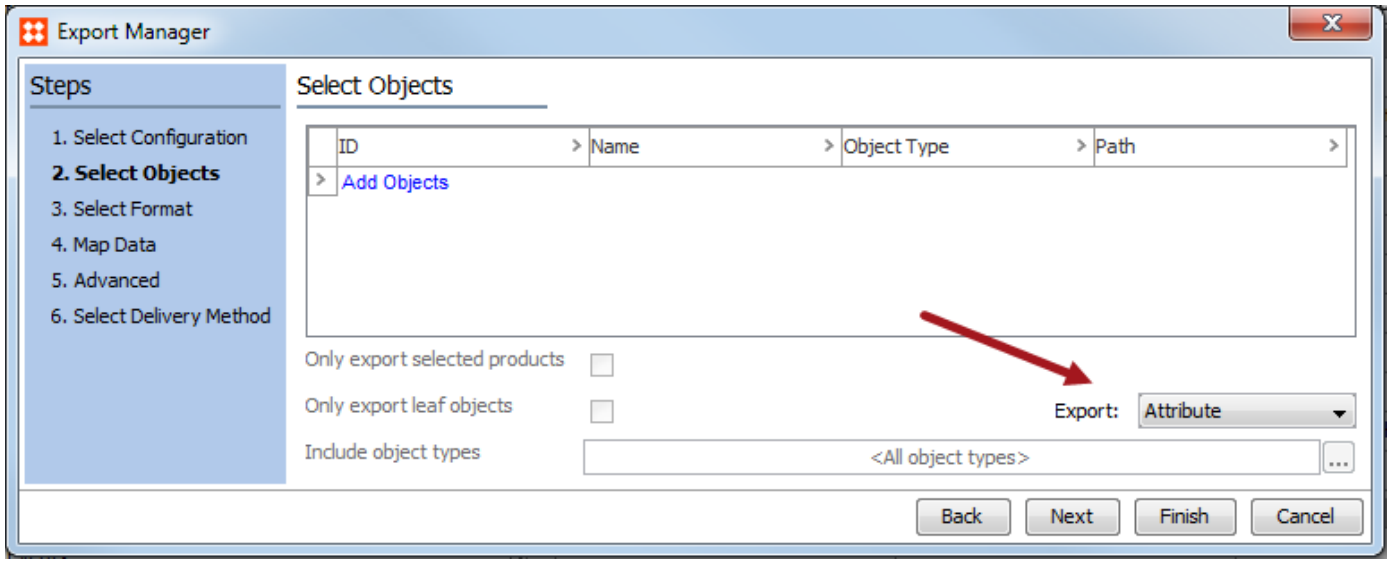
Attributes Processing Instructions in Generic XML

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.

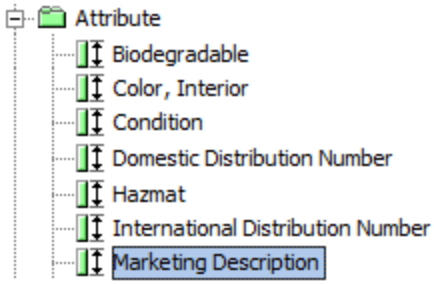
Follow the steps below to extract all available attribute data. If some data points on attributes are not required, they can be removed from the template and the mapping.

Objects

Choose 'Attribute' for the export.



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.

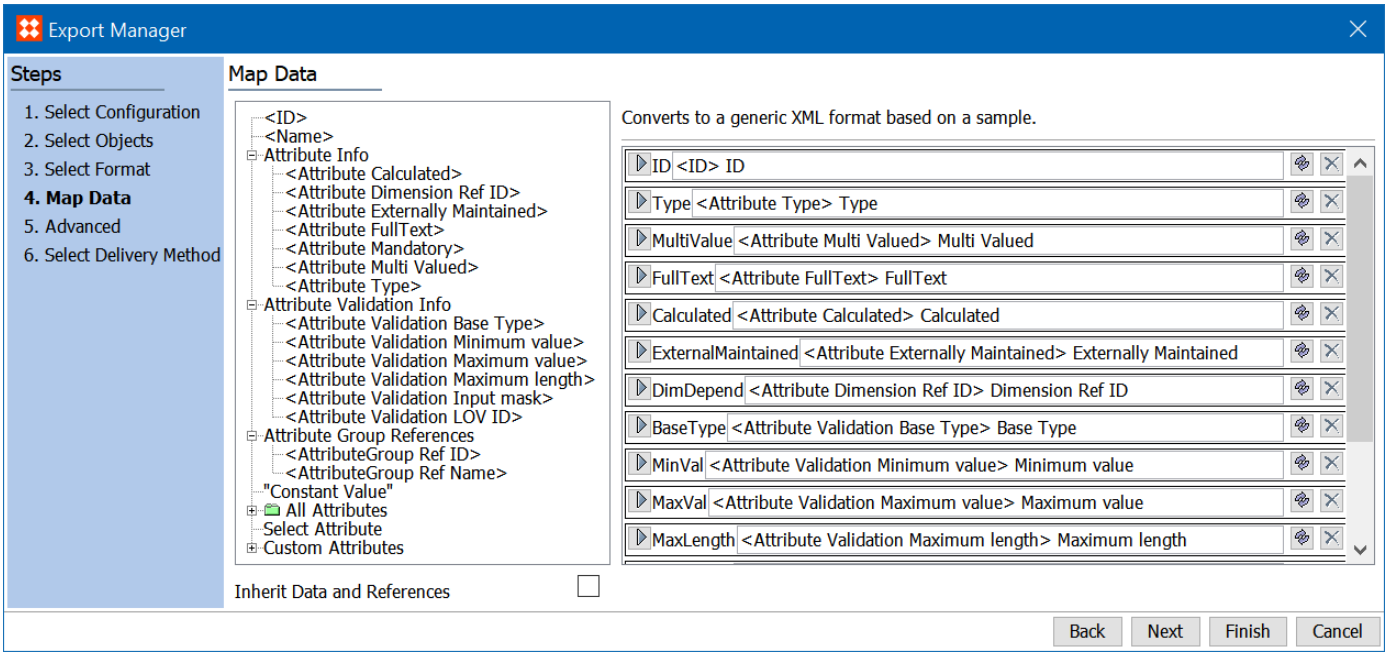
Template

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.

See the **Attribute Aspects in Generic XML** documentation for a description of each available data point.

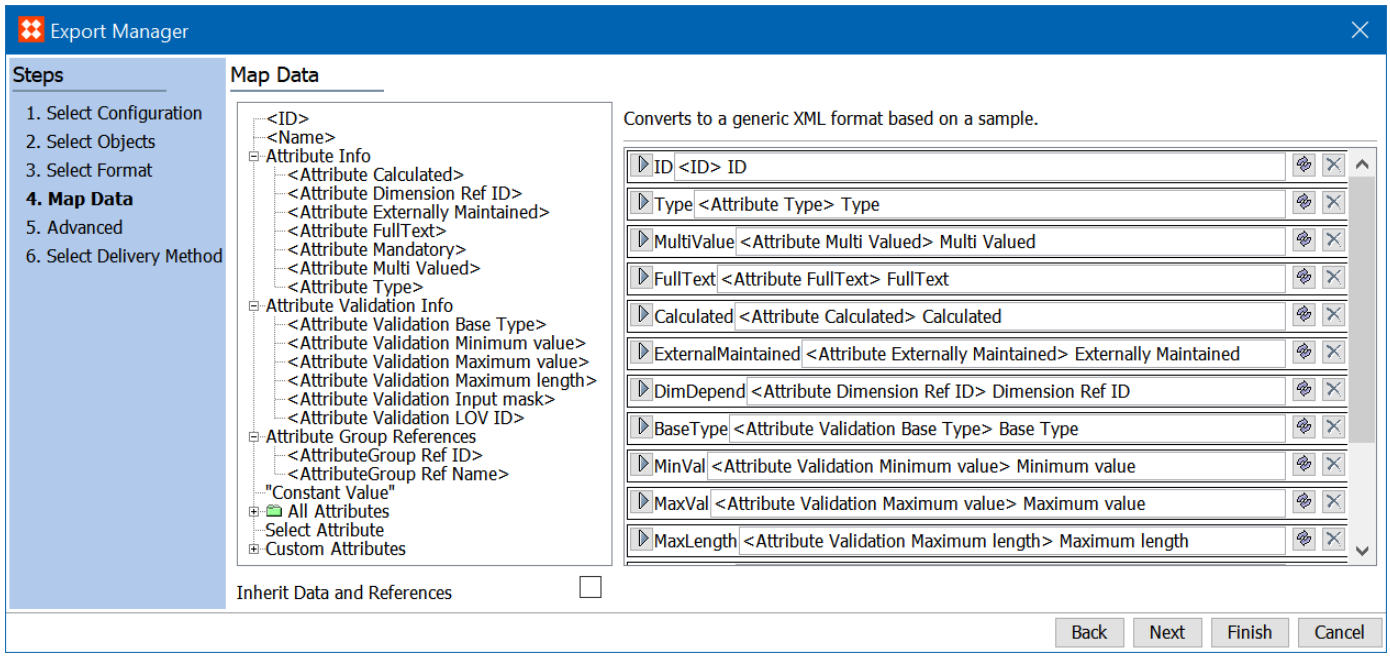
Results

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/>).
Attribute	Extracts the attribute validation input mask, if available.

Aspect	Description
Validation Input mask	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, see **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:

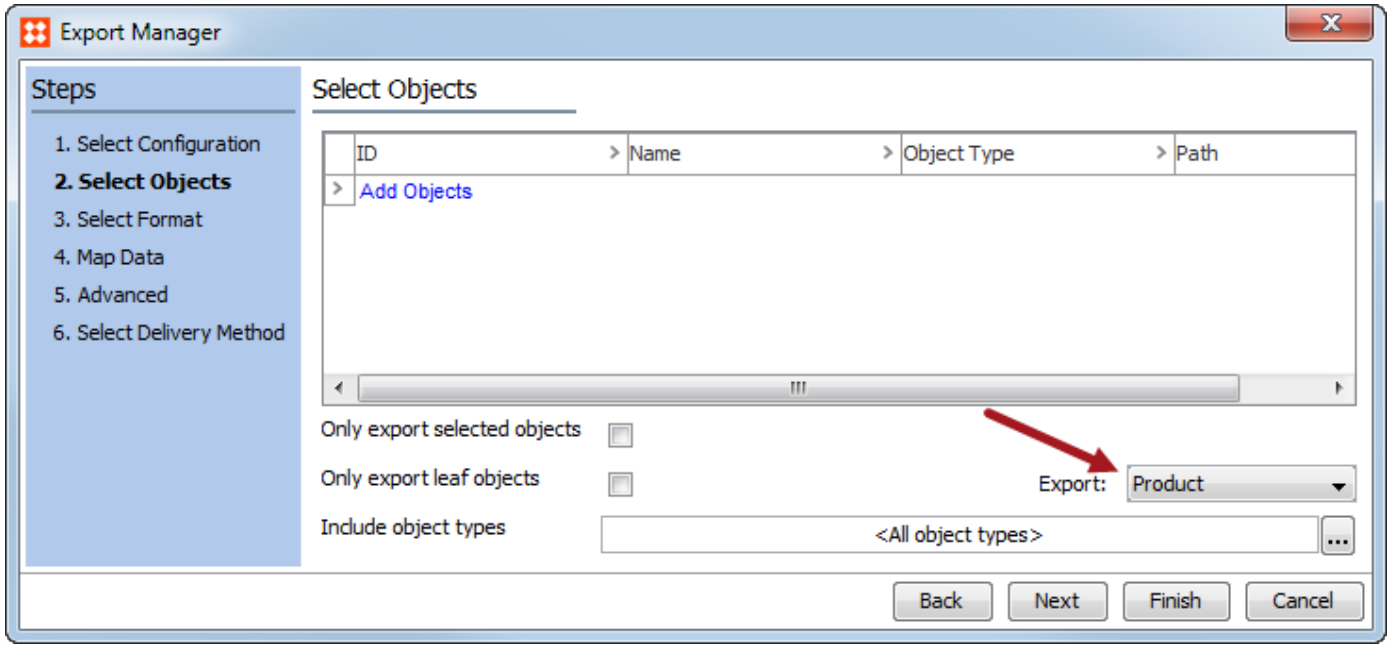
- the ID and Name of the Product
  - the ID and Name of the linked attributes on the Product
    - the ID, Name, and Values of the LOV on any of the linked attributes

Units for linked attributes can also be mapped and exported as discussed in **Attribute Links and Units Processing Instructions in Generic XML**.

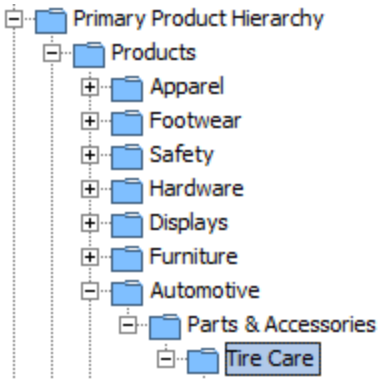
The data path data source is also available using the IDoc MATMAS 05 format.

Objects

Choose **Product** as object type to export.



Use the **Add Objects** link to select the data to export.



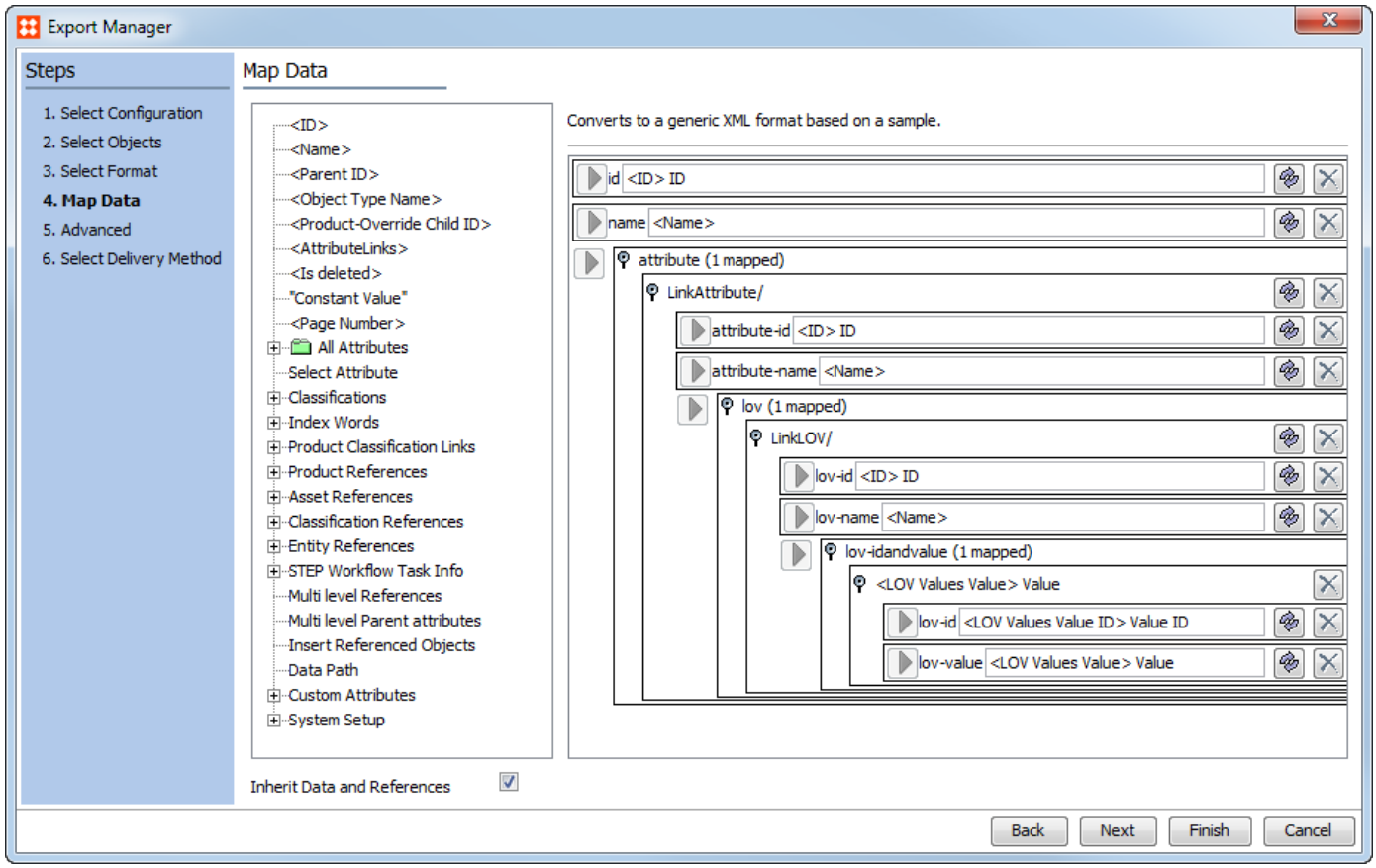
Template

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, see **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.

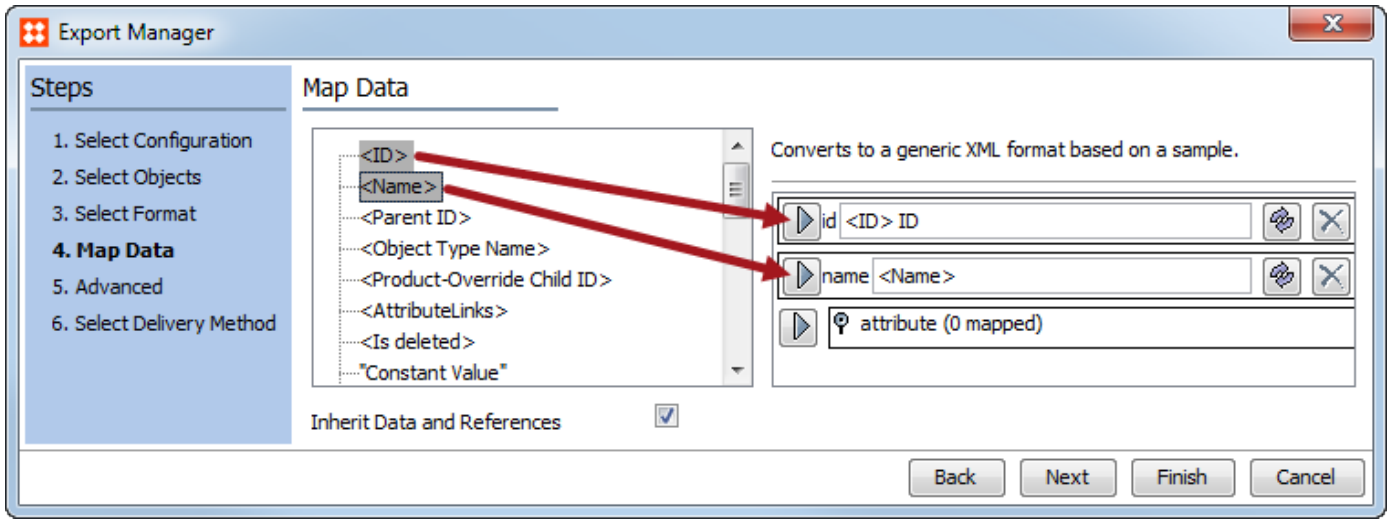
Note: Only local attribute links are exported.

```
<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>
```

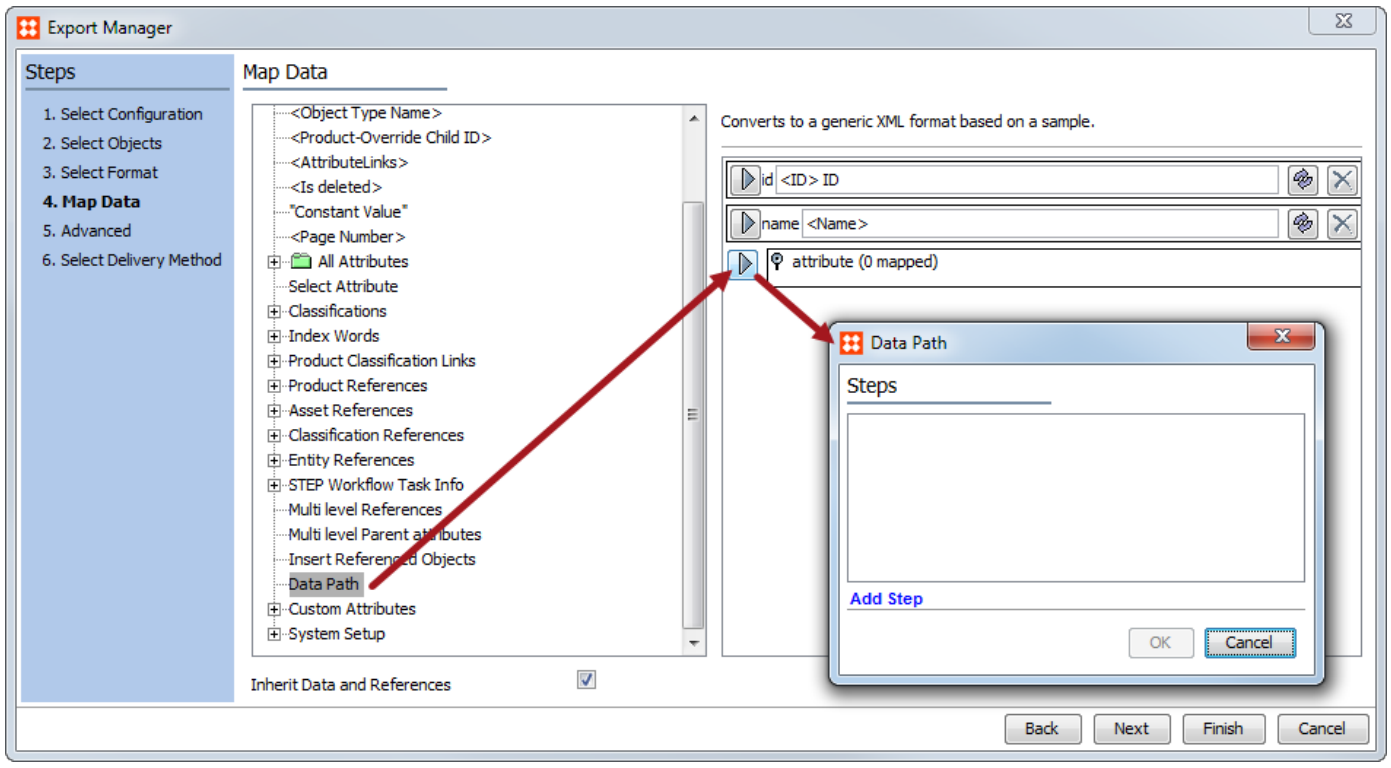
Mapping Attribute Links, LOVs, and LOV Values in Generic XML

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.

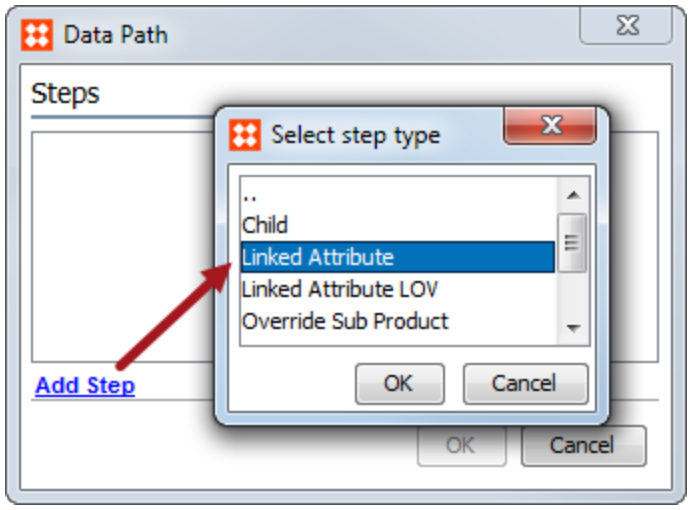
1. Map **ID** and **Name** for the Product.



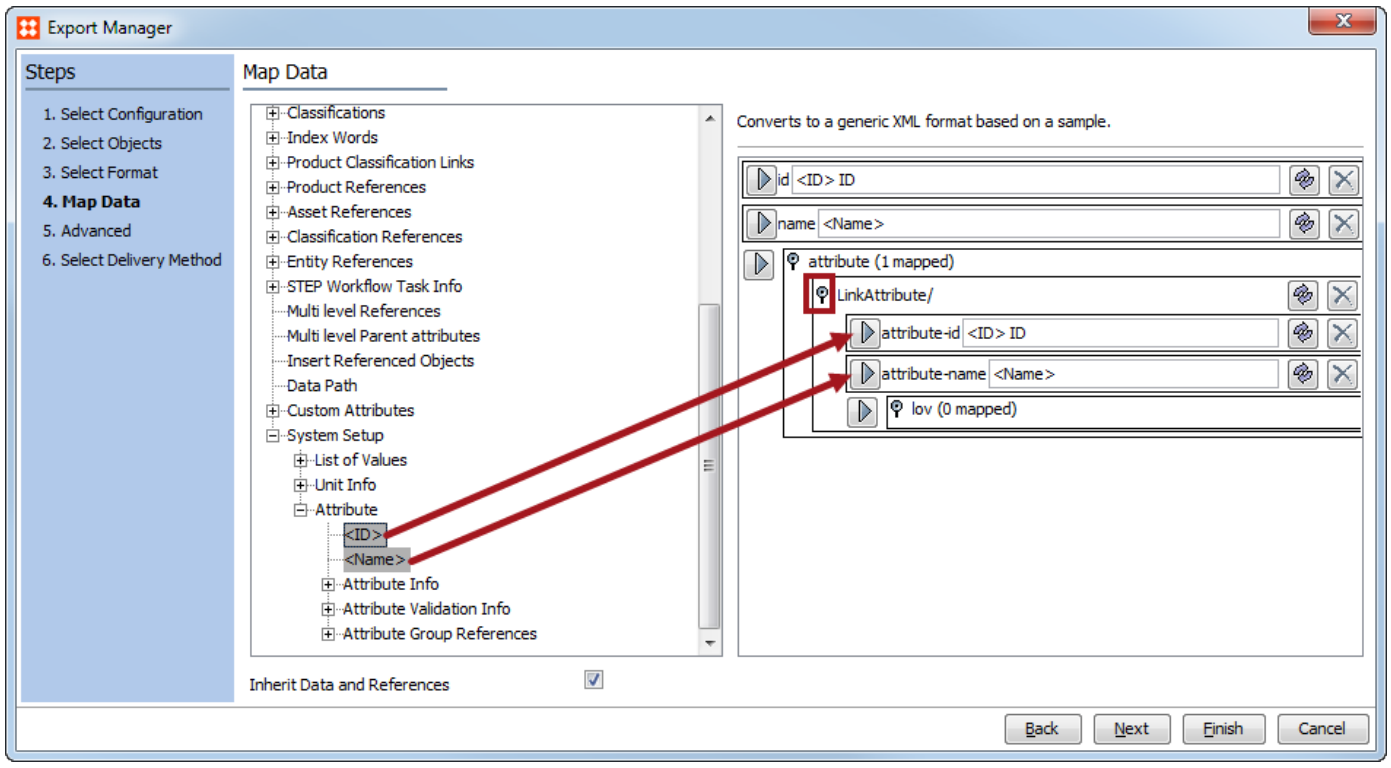
2. Map **Data Path** to attribute.



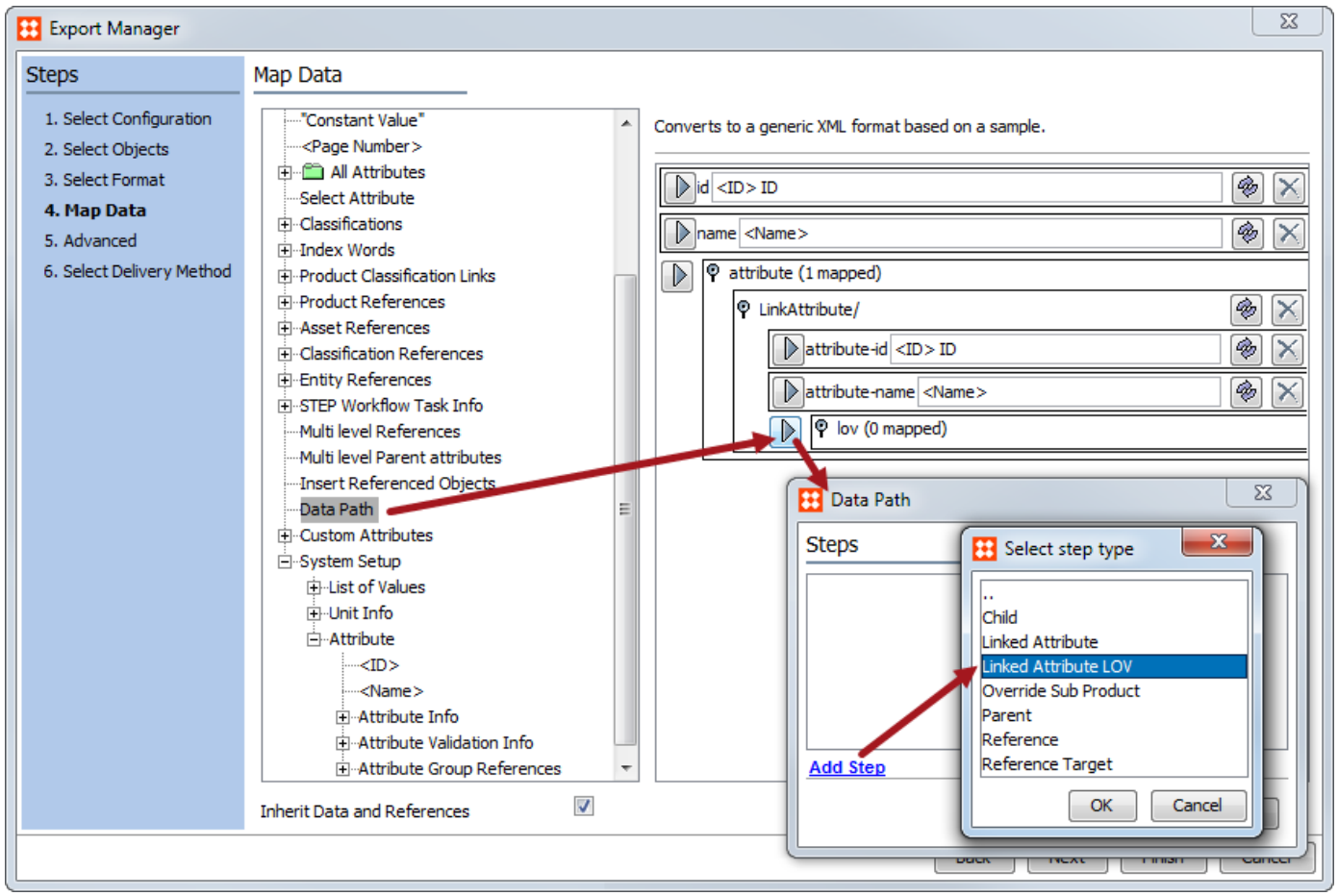
3. Click the **Add Step** link and select **Linked Attribute**.



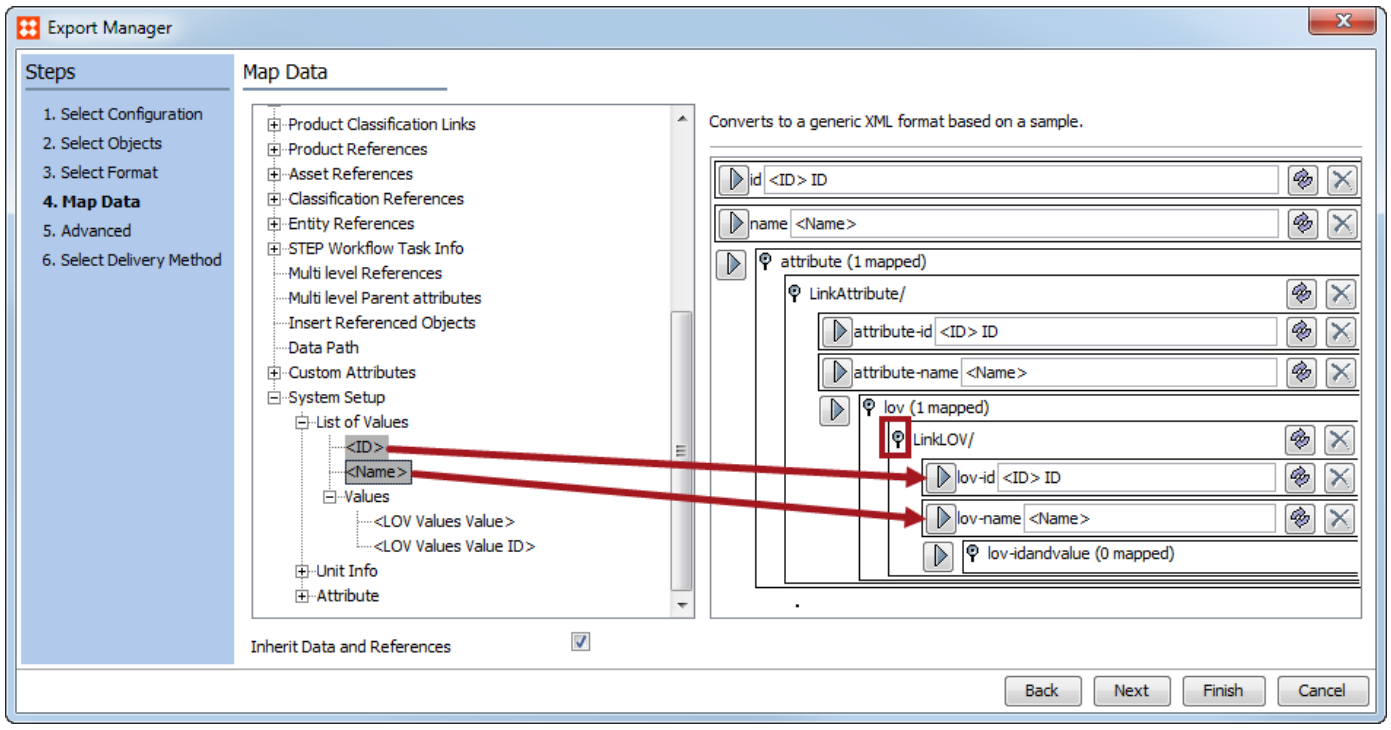
4. Click OK to close the **Select step type** and **Data Path** dialogs.
5. Open the LinkAttribute/ flipper and map Attribute **ID** and **Name**.



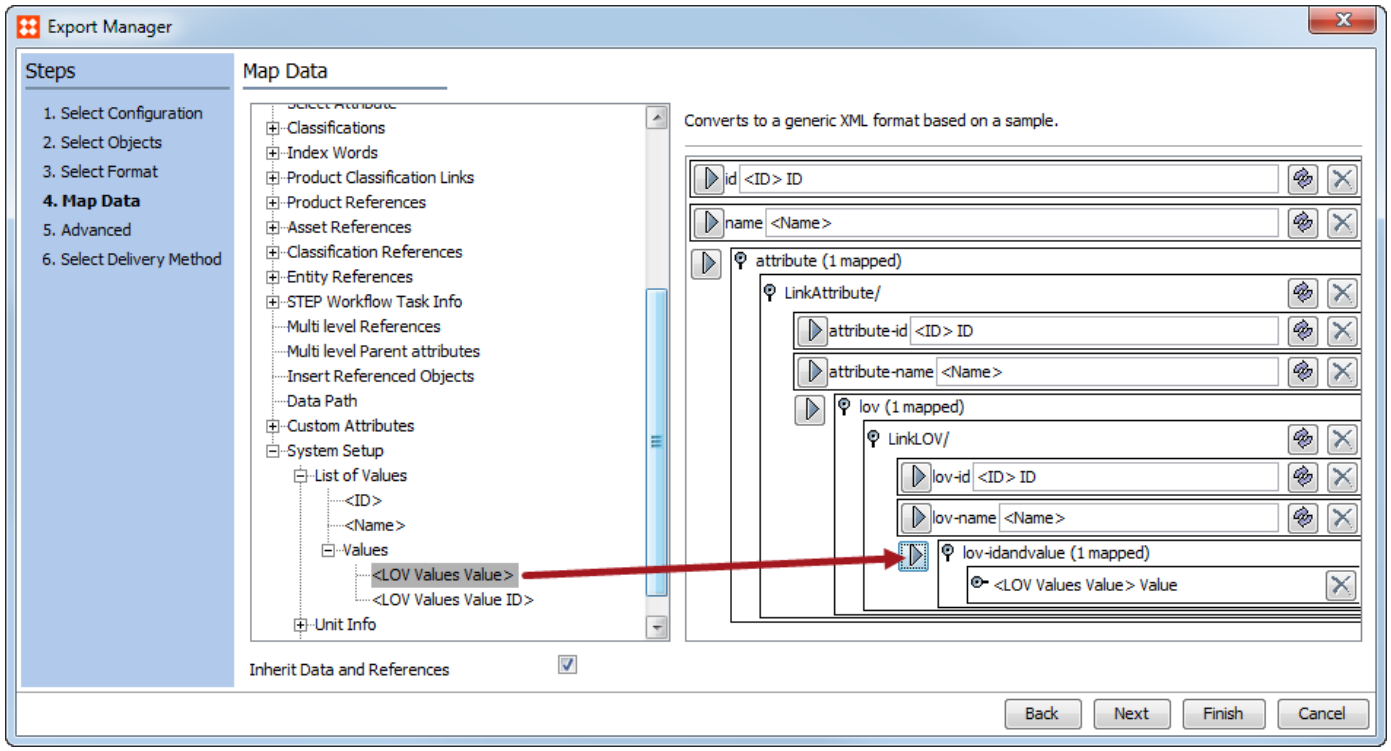
6. Map **Data Path** to lov, click **Add Step** and select **Linked Attribute LOV**.



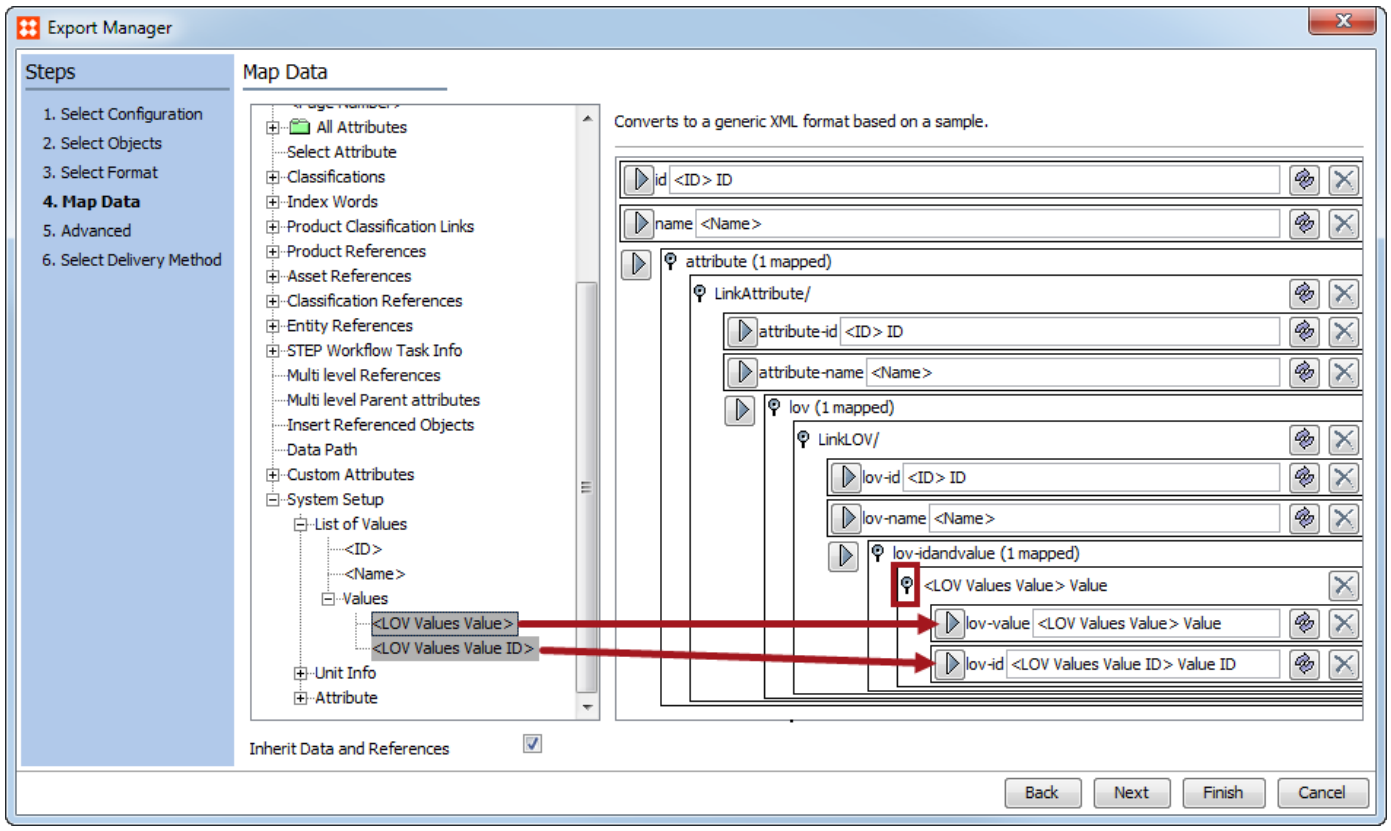
7. Click OK to close the **Select step type** and **Data Path** dialogs.
8. Open the LinkLOV/ flipper and map Attribute **ID** and **Name**.



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**.


Export Manager
Steps
1. Select Configuration
2. Select Objects
3. Select Format
4. Map Data
5. Advanced
6. Select Delivery Method
Map Data
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
Multi level Parent attributes
Insert Referenced Objects
Data Path
Custom Attributes
System Setup
List of Values
<ID>
<Name>
Values
<LOV Values Value>
<LOV Values Value ID>
Unit Info
Attribute
Converts to a generic XML format based on a sample.
id <ID> ID
name <Name>
attribute (1 mapped)
LinkAttribute/
attribute-id <ID> ID
attribute-name <Name>
lov (1 mapped)
LinkLOV/
lov-id <ID> ID
lov-name <Name>
lov-idandvalue (1 mapped)
<LOV Values Value> Value
lov-value <LOV Values Value> Value
lov-id <LOV Values Value ID> Value ID
Inherit Data and References
Back
Next
Finish
Cancel

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:

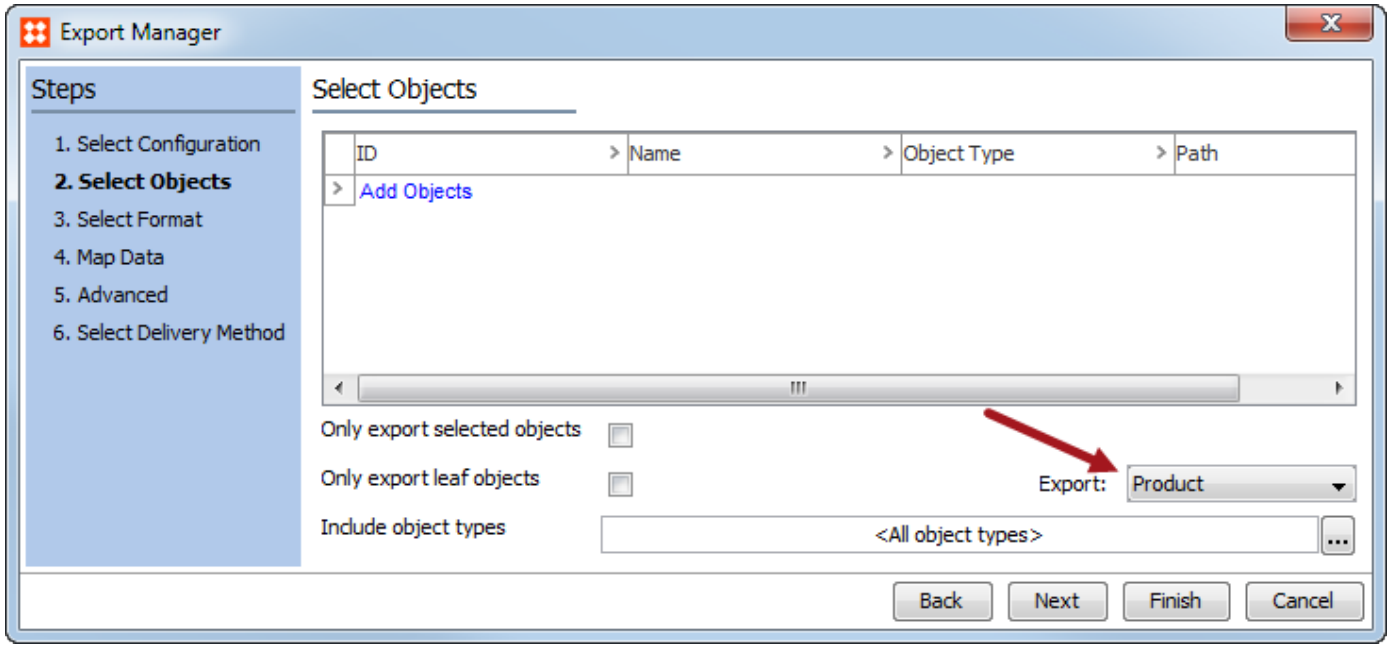
- the ID and Name of the Product
  - the ID and Name of the linked attributes on the Product
    - the ID, Name, and Base of the Unit on any of the linked attributes

LOVs for linked attributes can also be mapped and exported as described in **Attribute Links and LOVs Processing Instructions in Generic XML**.

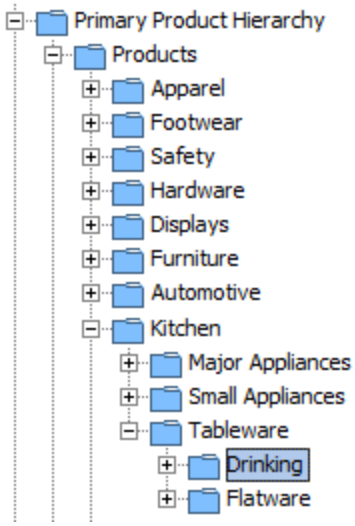
The data path data source is also available using the IDoc MATMAS 05 format.

Objects

Choose **Product** as object type to export.



Use the **Add Objects** link to select the data to export.



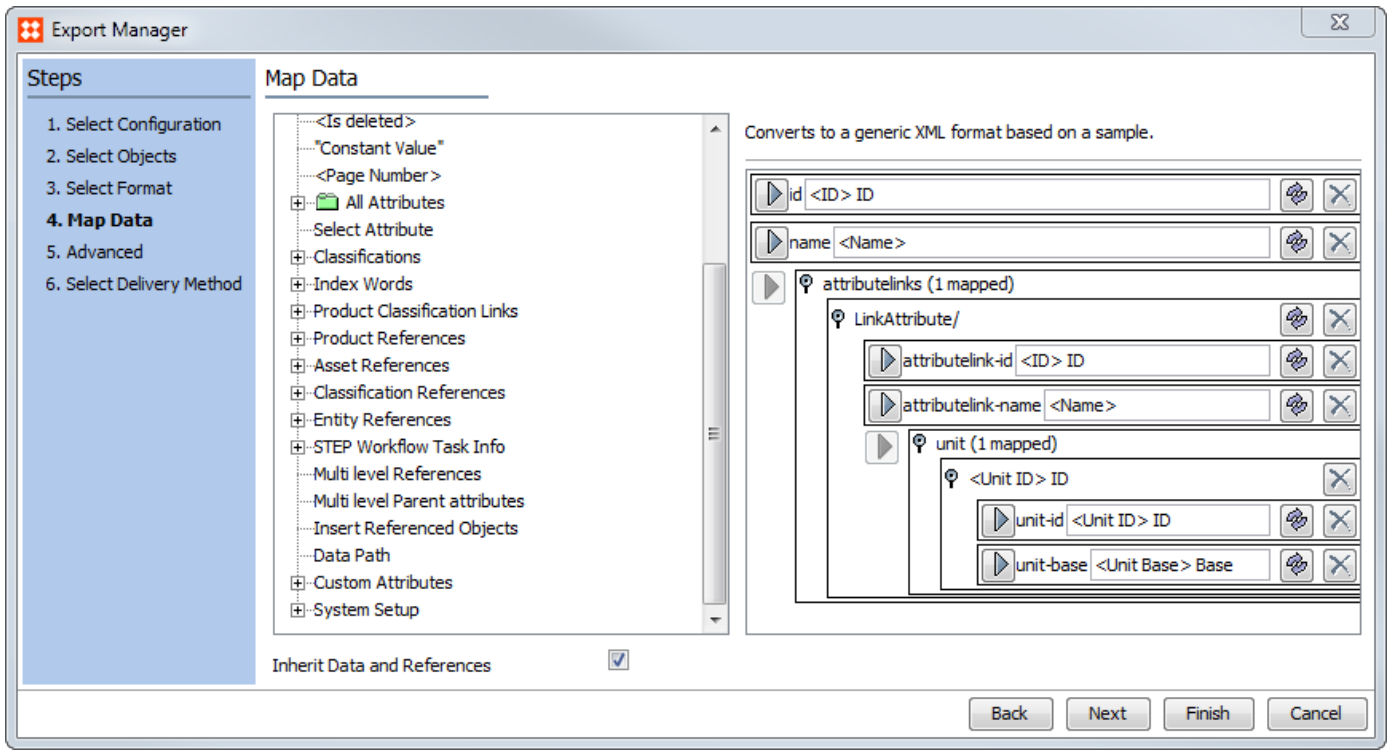
Template

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, see 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.

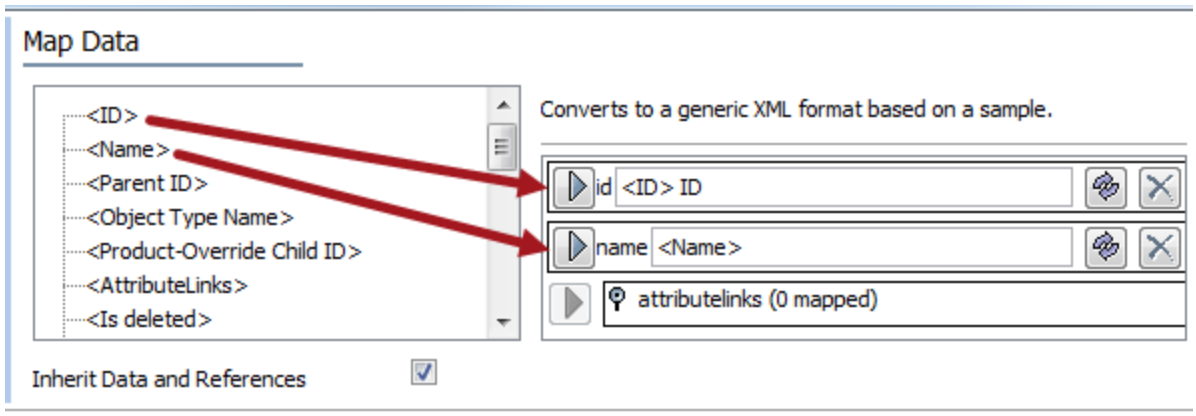
```
<products>
  <product>
    <id>8318</id>
    <name>Drinking</name>
    <attributelinks>
      <id>Capacity</id>
      <name>Capacity</name>
      <unit>
        <attributelink-unit-id>unece.unit.FTK</attributelink-unit-id>
        <attributelink-unit-base>unece.unit.MTK</attributelink-unit-base>
      </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

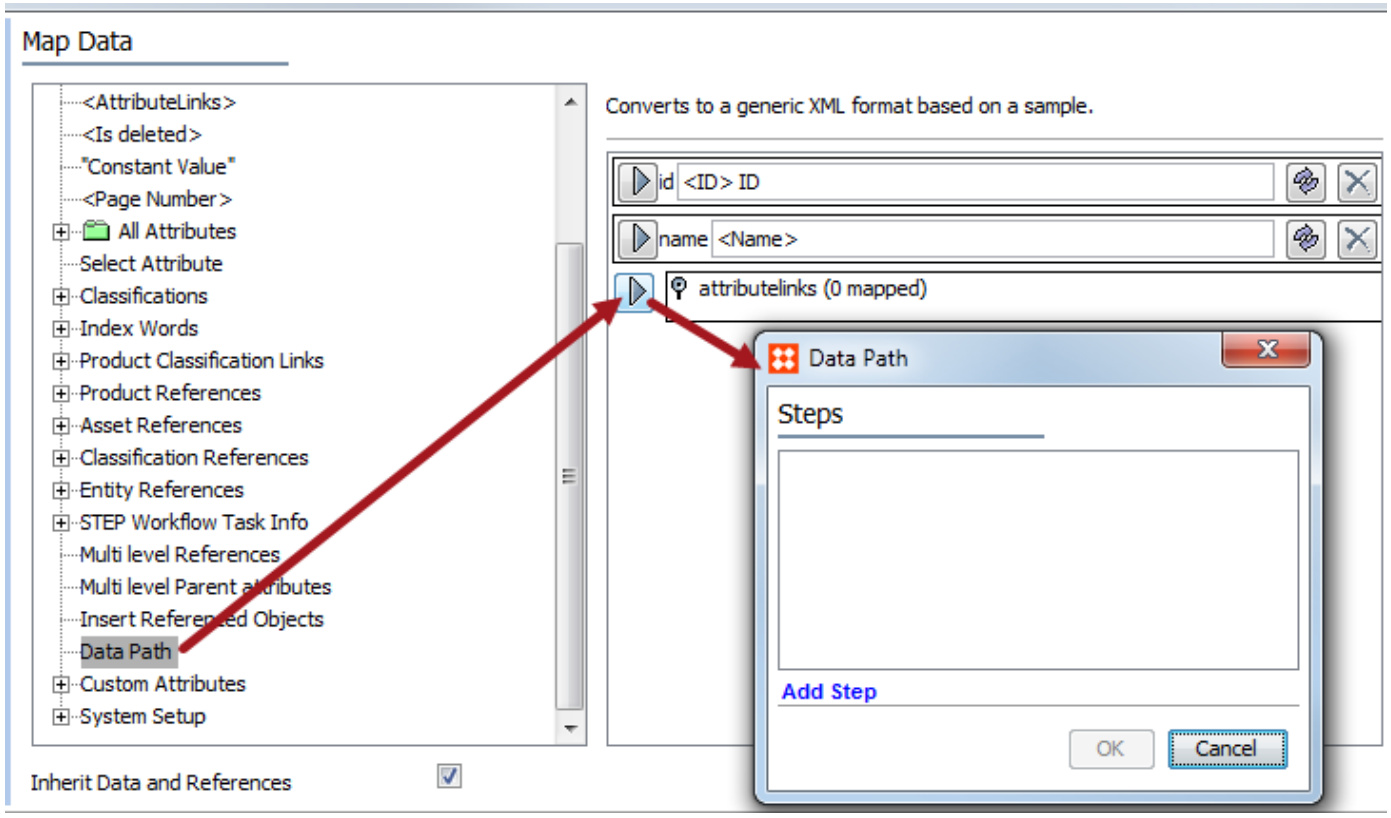
The **Attribute Links and Units Processing Instructions in Generic XML** section discusses additional setup for a Generic XML export. Use the following steps to map the Product, linked attributes, and their Unit information.

The data path data source is also available using the IDoc MATMAS 05 format.

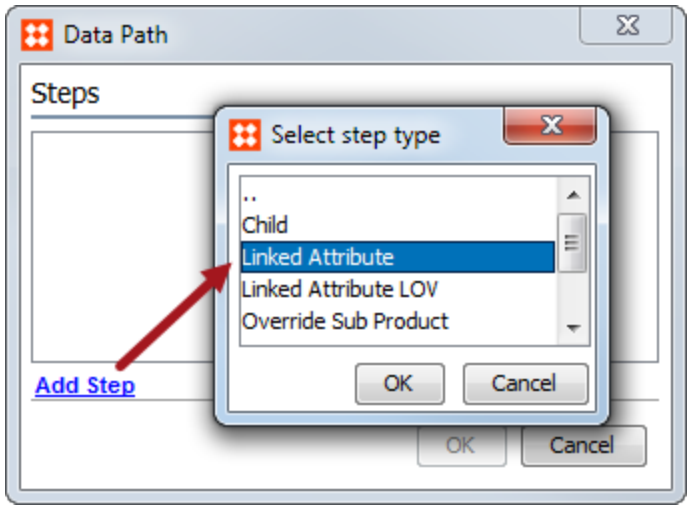
1. Map **ID** and **Name** for the Product.



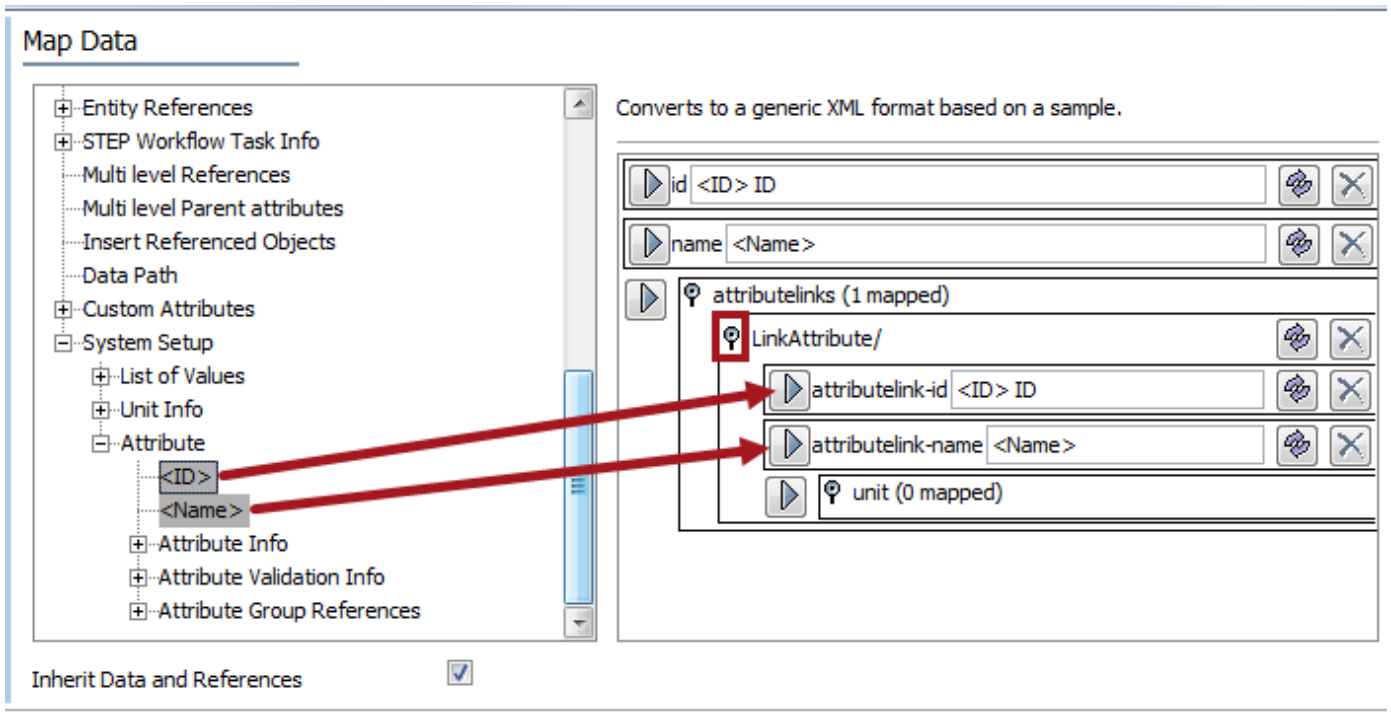
2. Map **Data Path** to attributelinks.



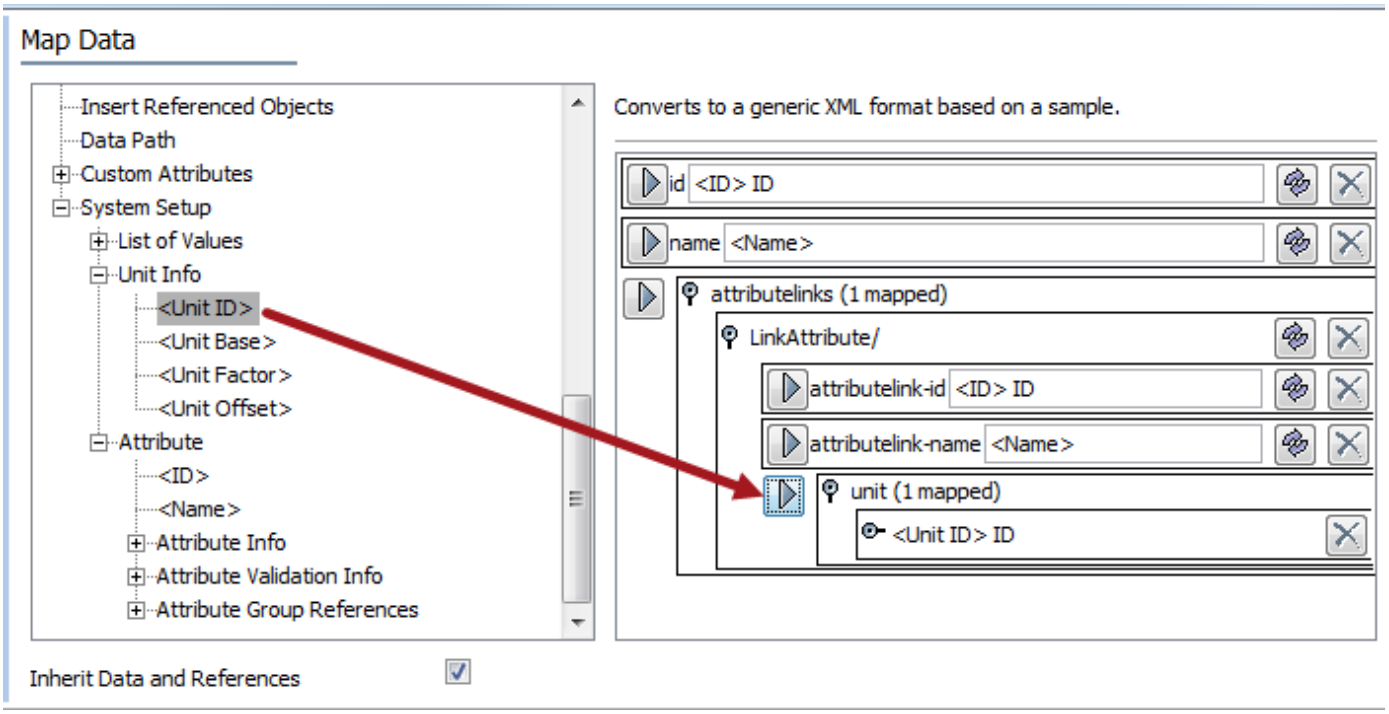
3. Click the **Add Step** link and select **Linked Attribute**.



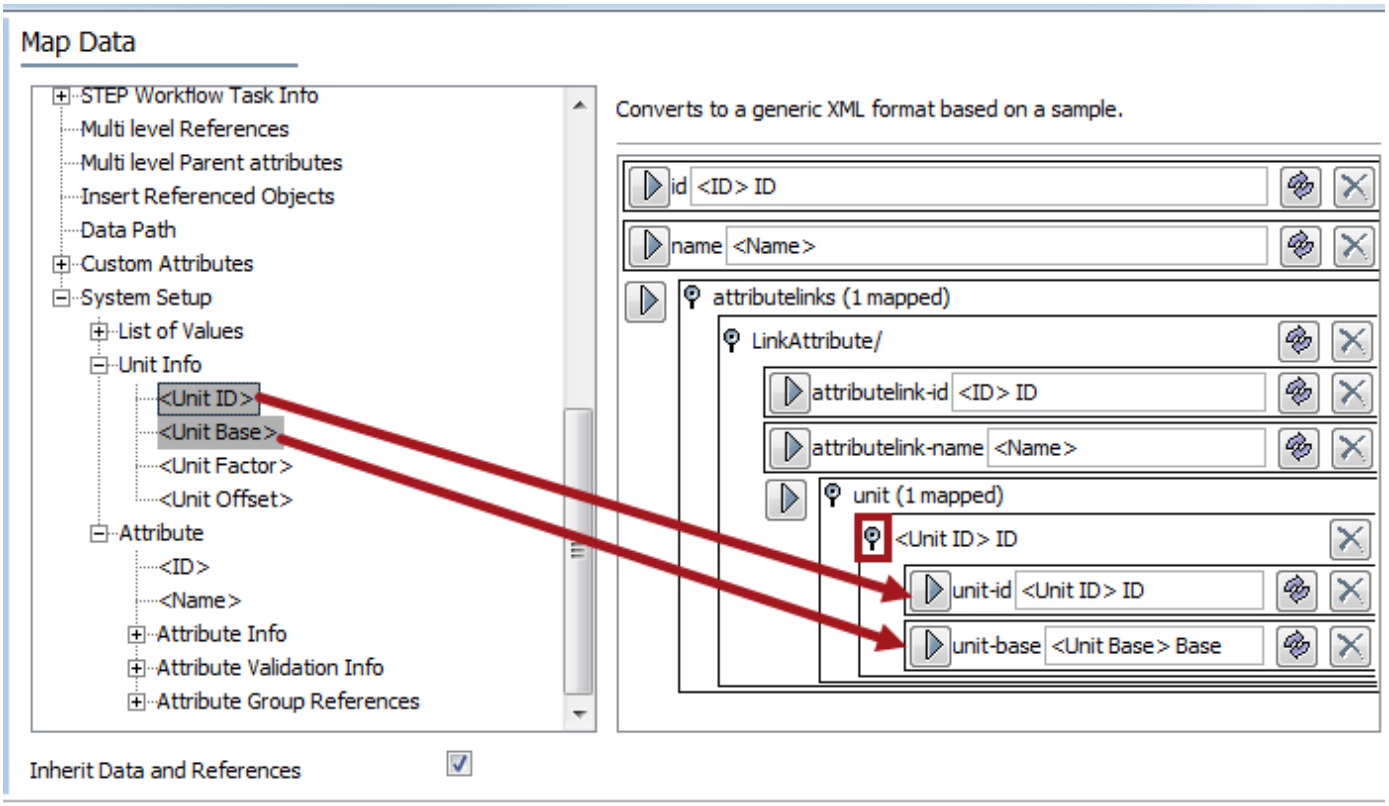
4. Click OK to close the **Select step type** and **Data Path** dialogs.
5. Open the LinkAttribute/ flipper and map Attribute **ID** and **Name**.



6. Map **Unit ID** to unit.



7. Open the <Unit ID> flipper and map Unit **ID** and **Base**.

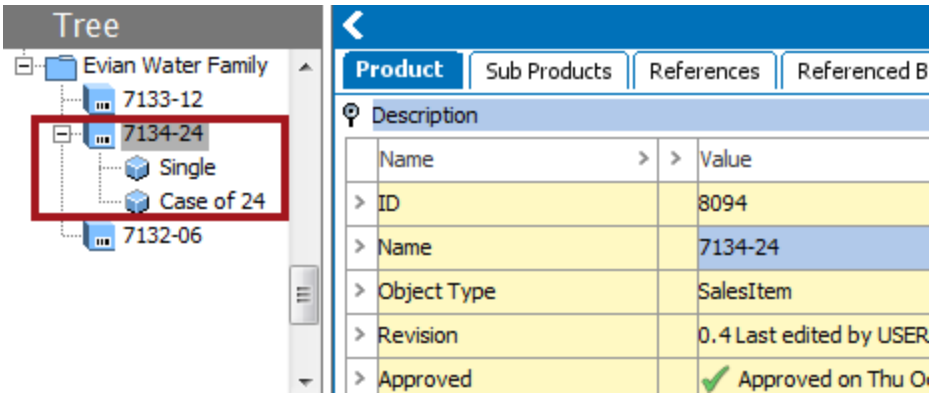


DoNotEmit Processing Instructions in Generic XML

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.

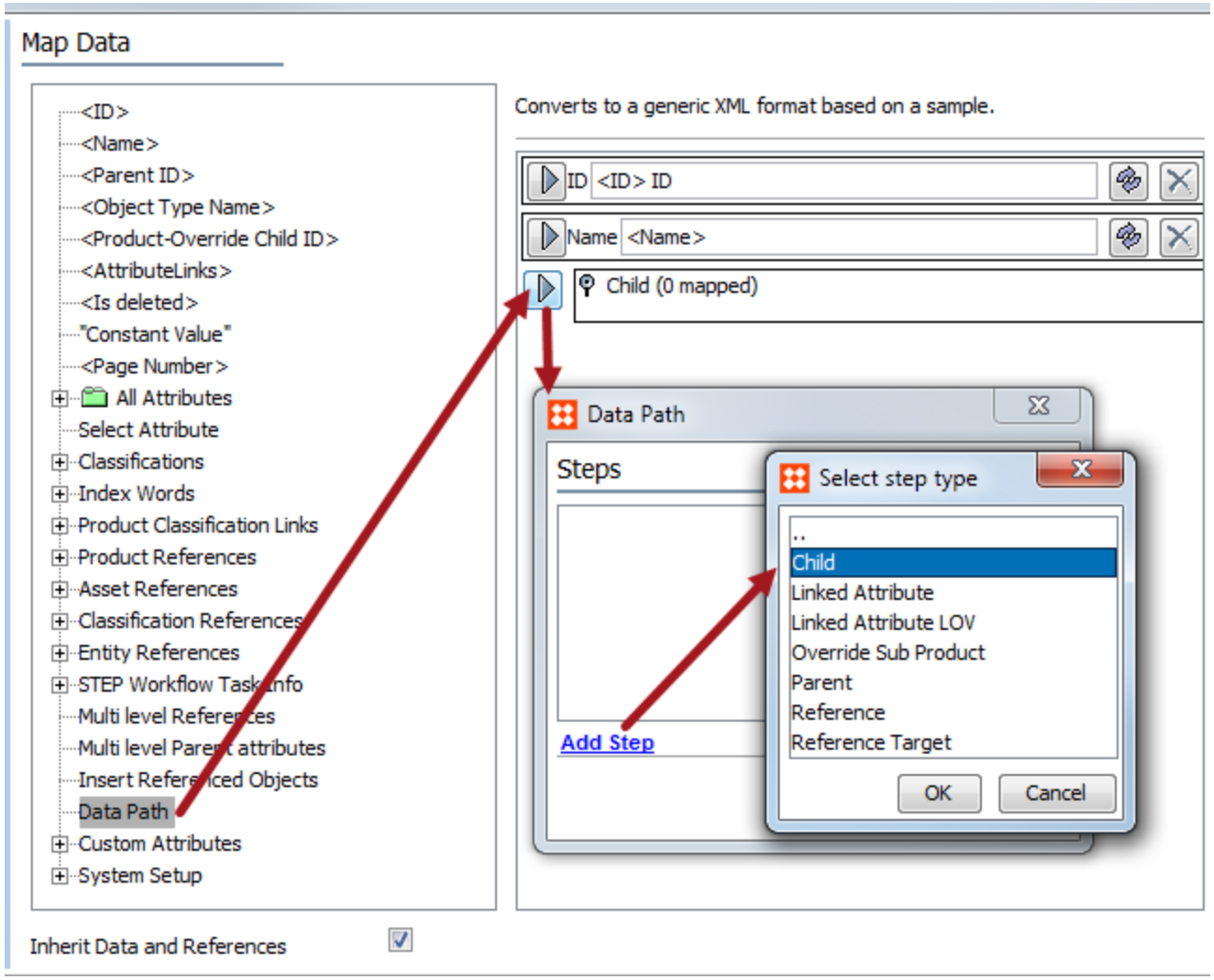
Without DoNotEmit	With DoNotEmit
<Families> <Family ID="[?Target ID?]"> <?Record?> <Name><?Target Name?></Name> <Children> <Child> <?MultiTarget?> <ID><?Target ID?></ID> <Name><?Target Name?></Name> </Child>	<Families> <Family ID="[?Target ID?]"> <?Record?> <Name><?Target Name?></Name> <Children> <Child> <?DoNotEmit?> <?MultiTarget?> <ID><?Target ID?></ID> <Name><?Target Name?></Name> </Child>

Without DoNotEmit	With DoNotEmit
</Children> </Family> </Families>	</Children> </Family> </Families>

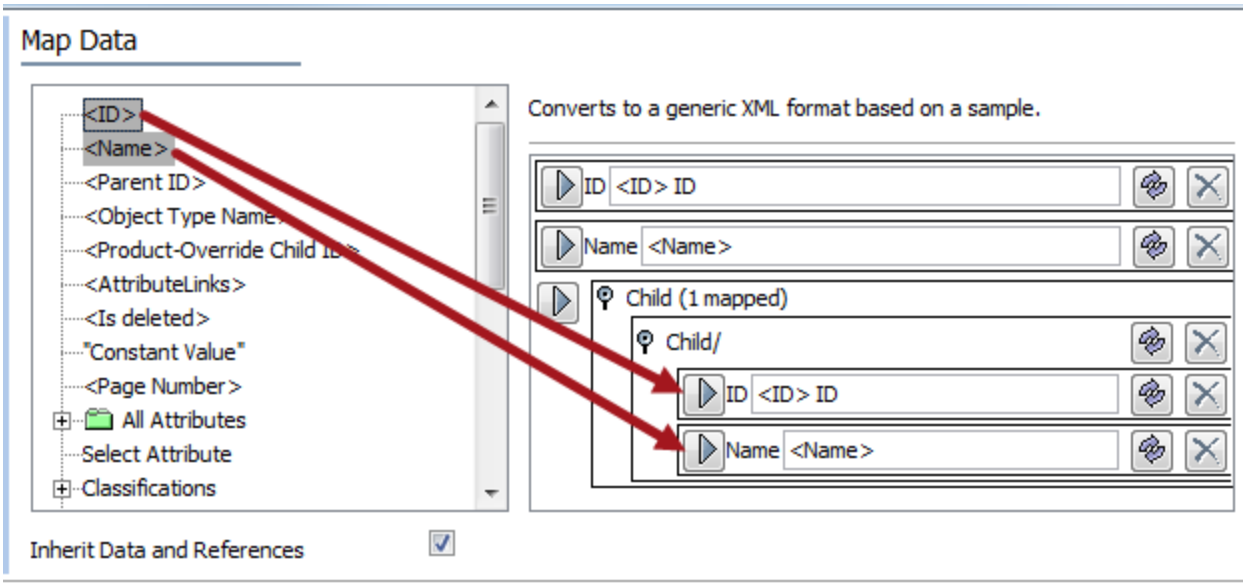
Mapping

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.

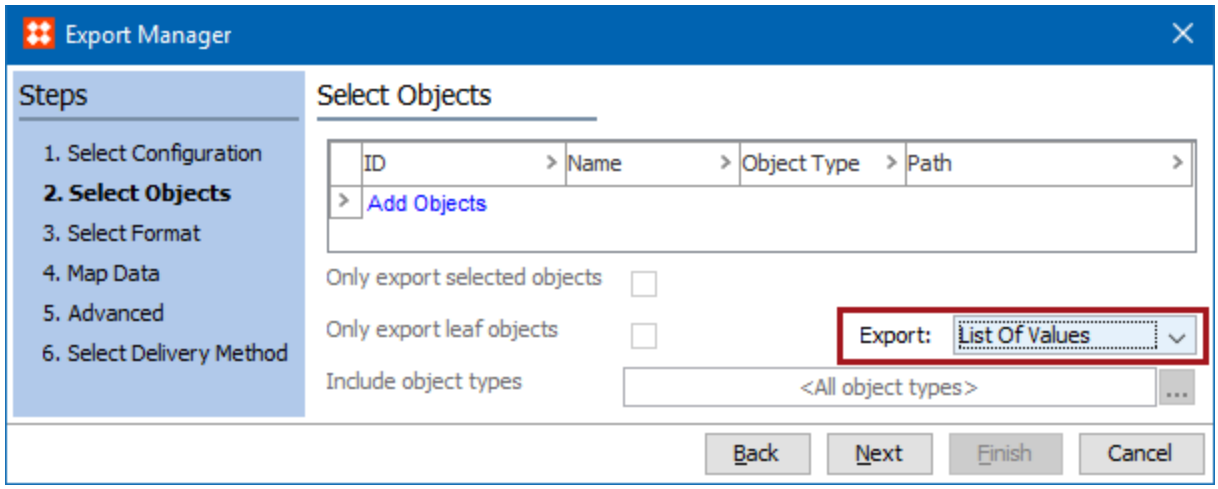
Without DoNotEmit	With DoNotEmit
<Families> <Family ID="8094"> <Name>7134-24</Name> <Children> <Child> <ID>8082</ID> <Name>Single</Name> </Child> <Child> <ID>107807</ID> <Name>Case of 24</Name> </Child> </Children> </Family> </Families>	<Families> <Family ID="8094"> <Name>7134-24</Name> <Children> <ID>8082</ID> <Name>Single</Name> <ID>107807</ID> <Name>Case of 24</Name> </Children> </Family> </Families>

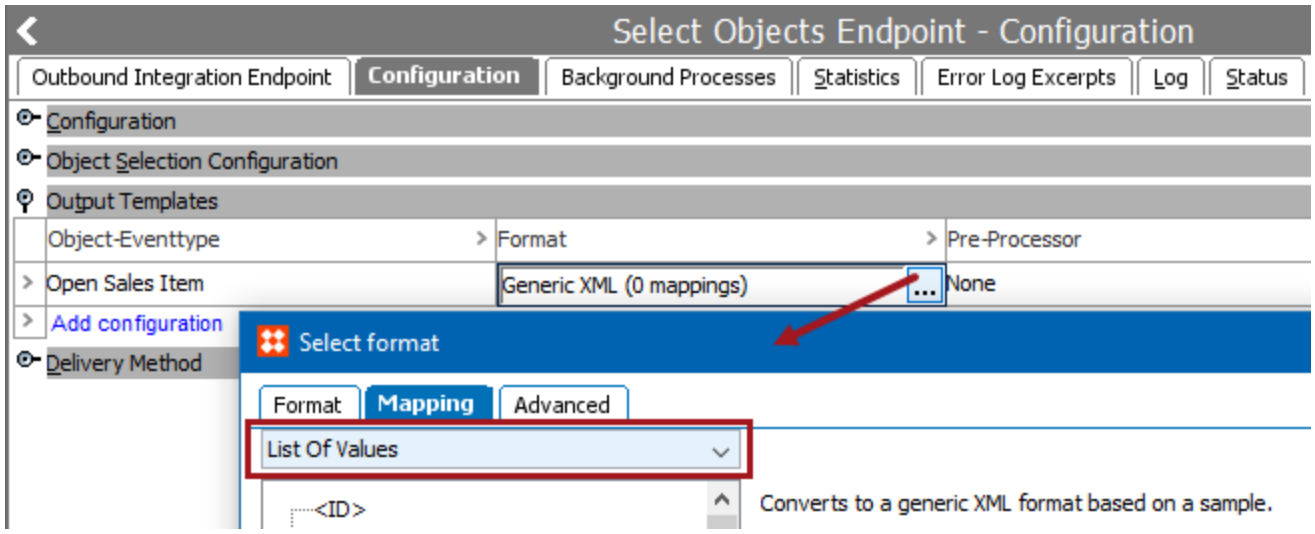
List of Values Object Type in Generic XML

Export tools require that you select objects and provide both a sample template and mapping when exporting Generic XML.

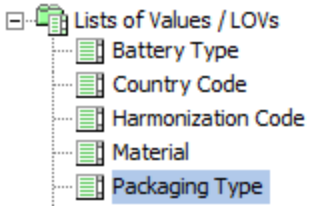
Objects

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, see the **Export Manager - Select Objects** topic.

For OIEP, see the **Object Types** section of the **OIEP - Event-Based - Event Triggering Definitions Tab** topic or the **OIEP - Select Objects - Object Selection Configuration Flipper** topic.

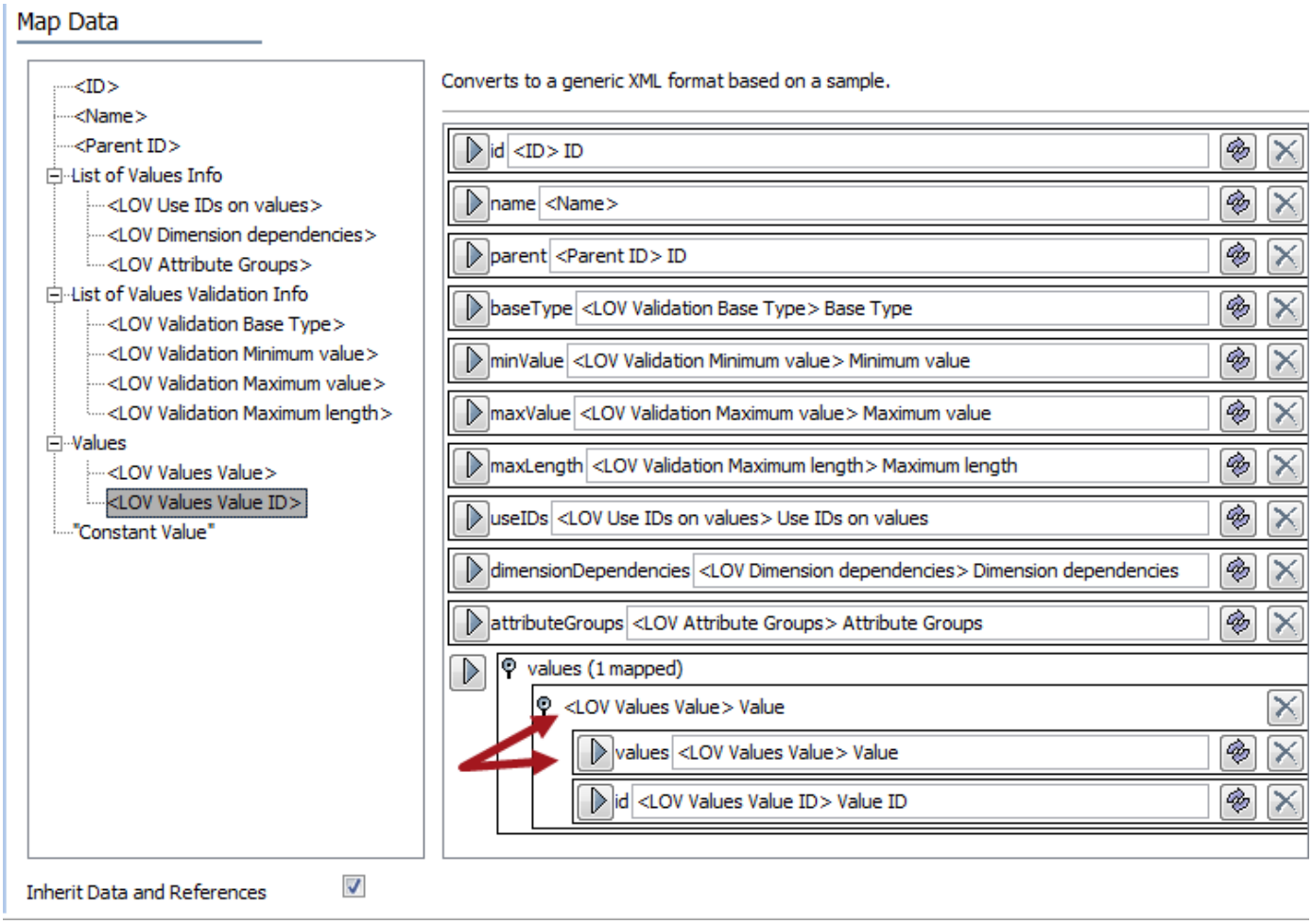
Template

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

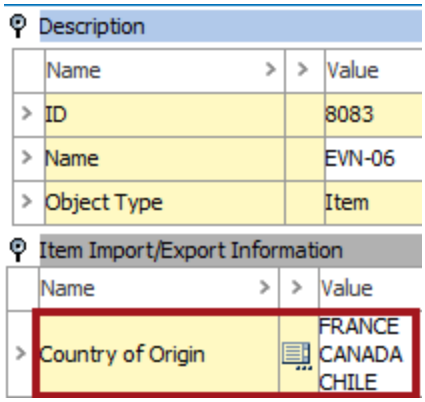
There are two Generic XML options for exporting fields that allow multiple values.

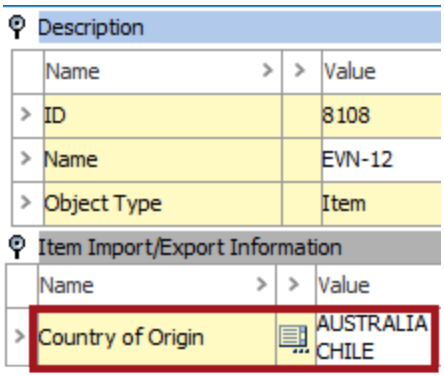
- Repeated Target exports a delimited list of the values in a single tag
- MultiTarget exports each value individually, in separate tags

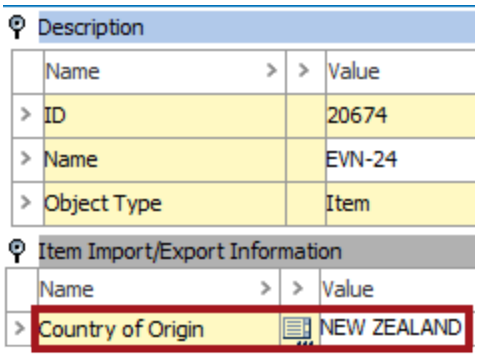
MultiTarget is usually the preferred option.

Objects

The Country of Origin multi valued attribute is exported for each of these selected products:





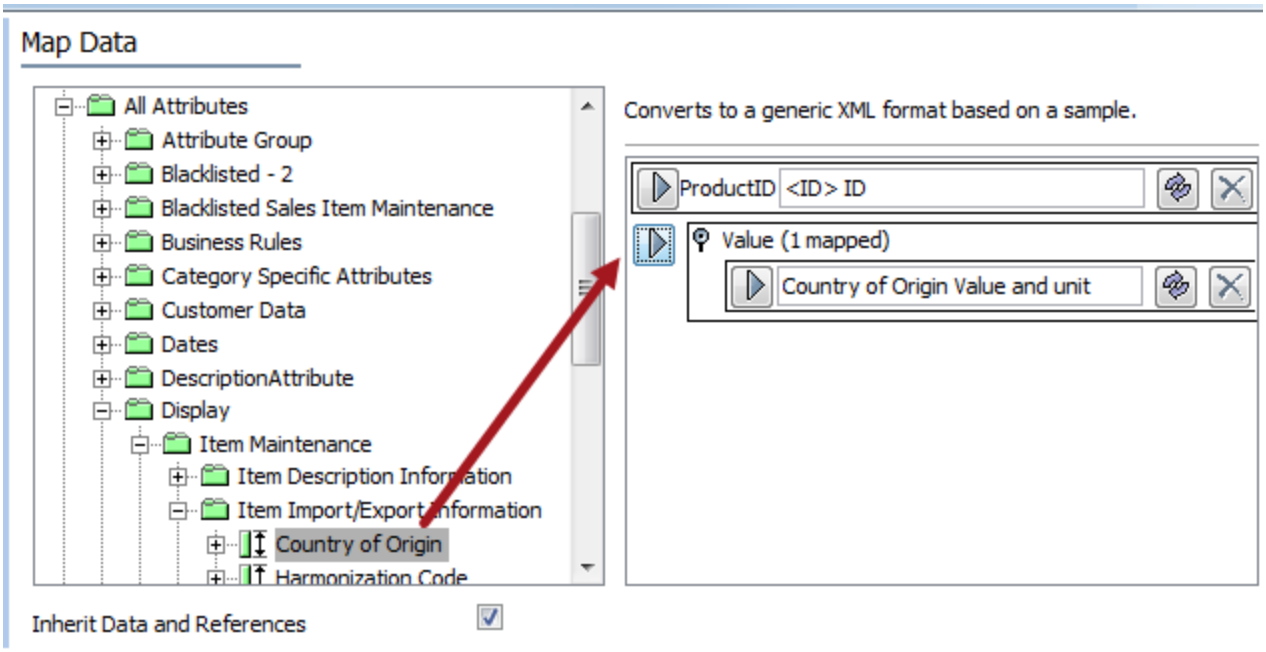


Template

Repeated Target	MultiTarget
<Products> <Product> <?Record?> <ProductID><?Target?></ProductID> <Value><?Target?></Value> <Value><?Target?></Value> </Product> </Products>	<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	MultiTarget
<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>	<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>

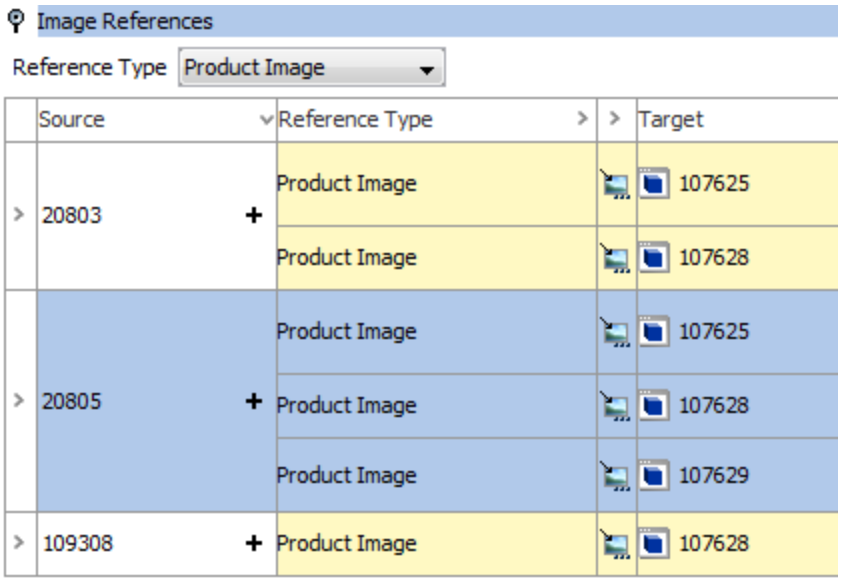
Multi-Cardinality Tags in Generic XML

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.

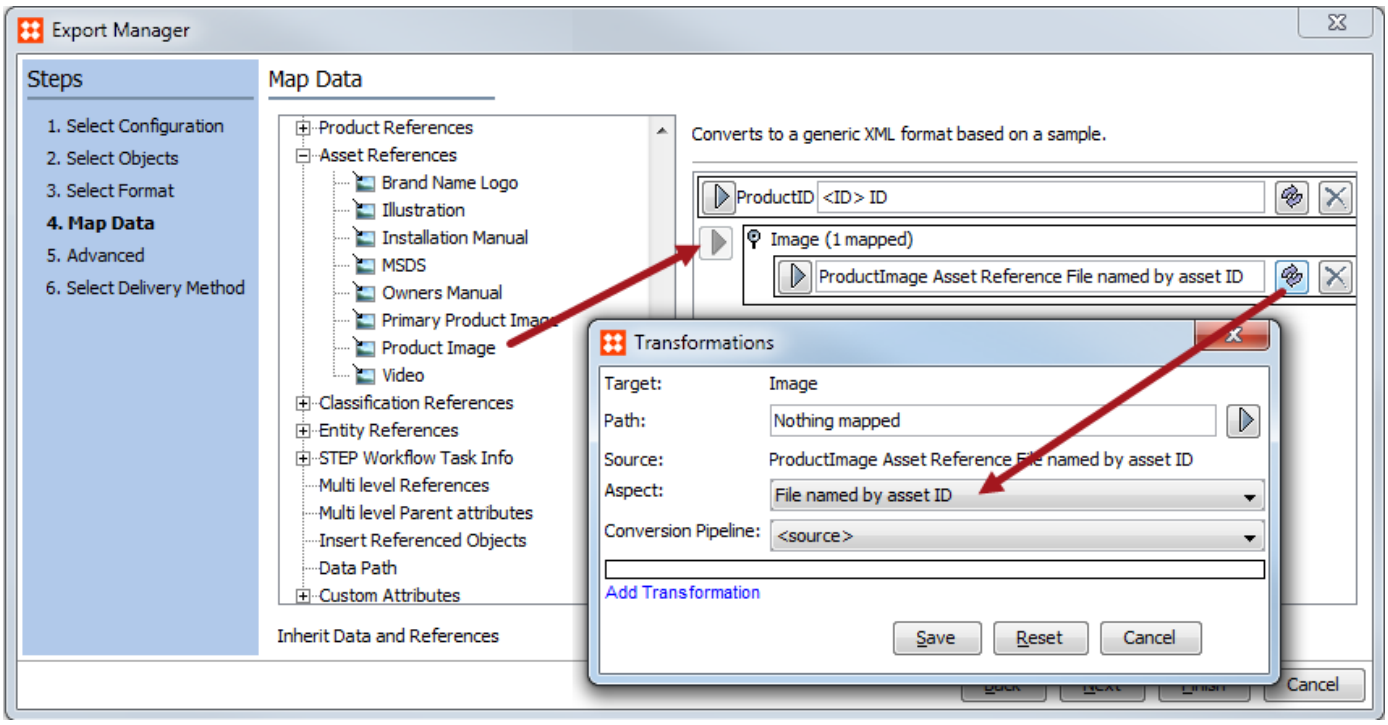
Objects

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

```
<Products>
   <Product>
      <?Record?>
      <ProductID><?Target?></ProductID>
      <Image><?MultiTarget?></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 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

```
<products>
   <product>
      <?Record?>
      <ProductID><?Target?></ProductID>
      <image>
      <?MultiTarget?>
         <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.

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:

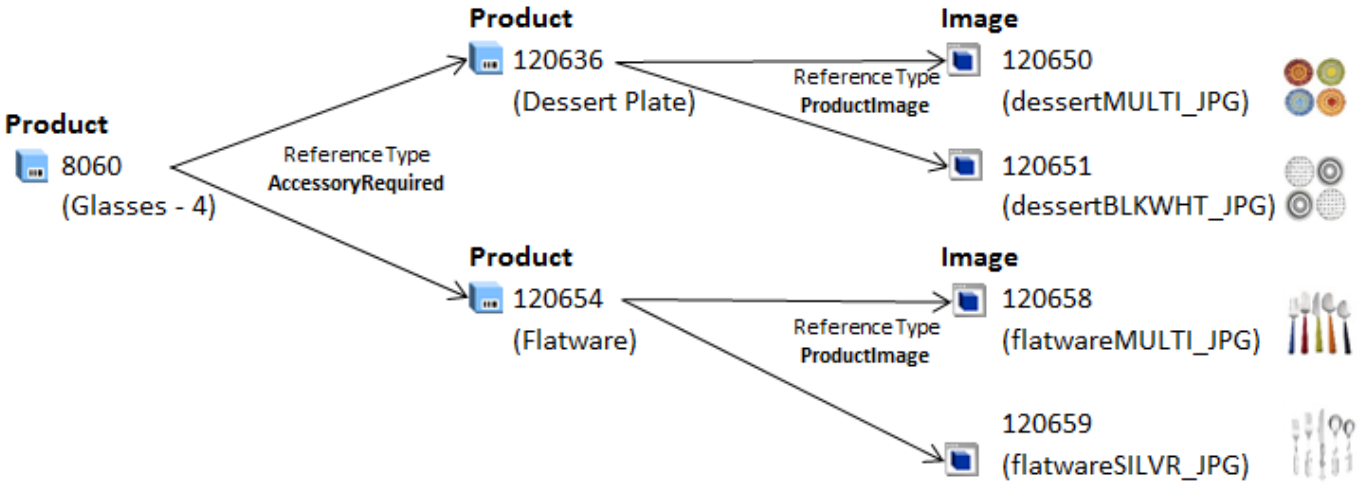
- ID
- Name
- Parent ID
- Object-Type

- 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>
```

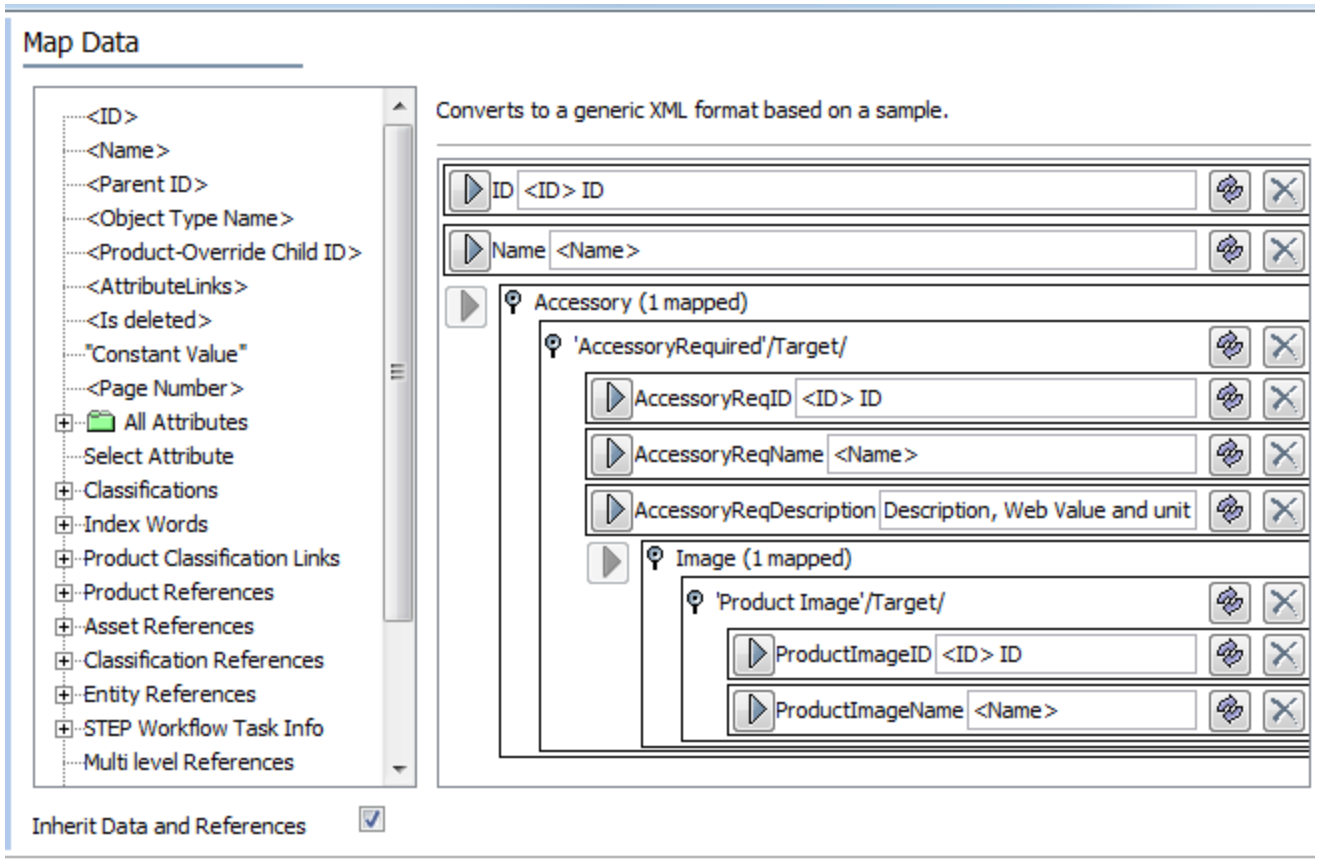
```
            <Image>
            <?MultiTarget?>
            <ID><?Target ProductImageID?></ID>
            <Name><?Target ProductImageName?></Name>
            </Image>
        </Accessory>
    </Product>
</Products>
```

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



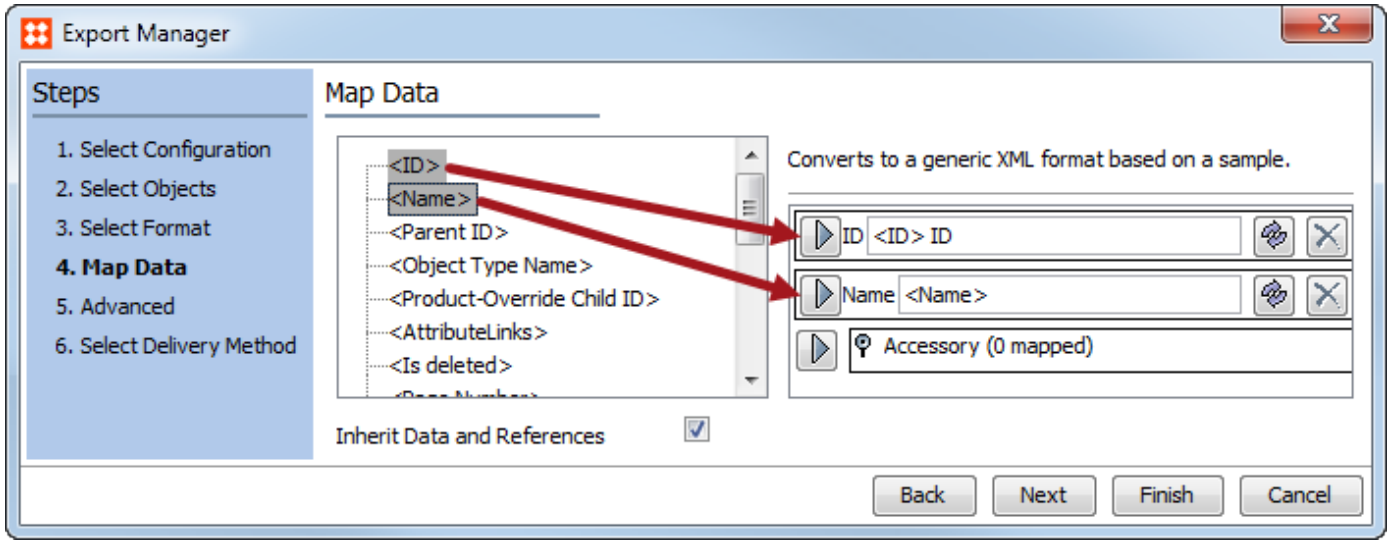
For step-by-step mapping instructions, see the **Mapping for Nestable Data Paths in Generic XML Example** documentation.

Results

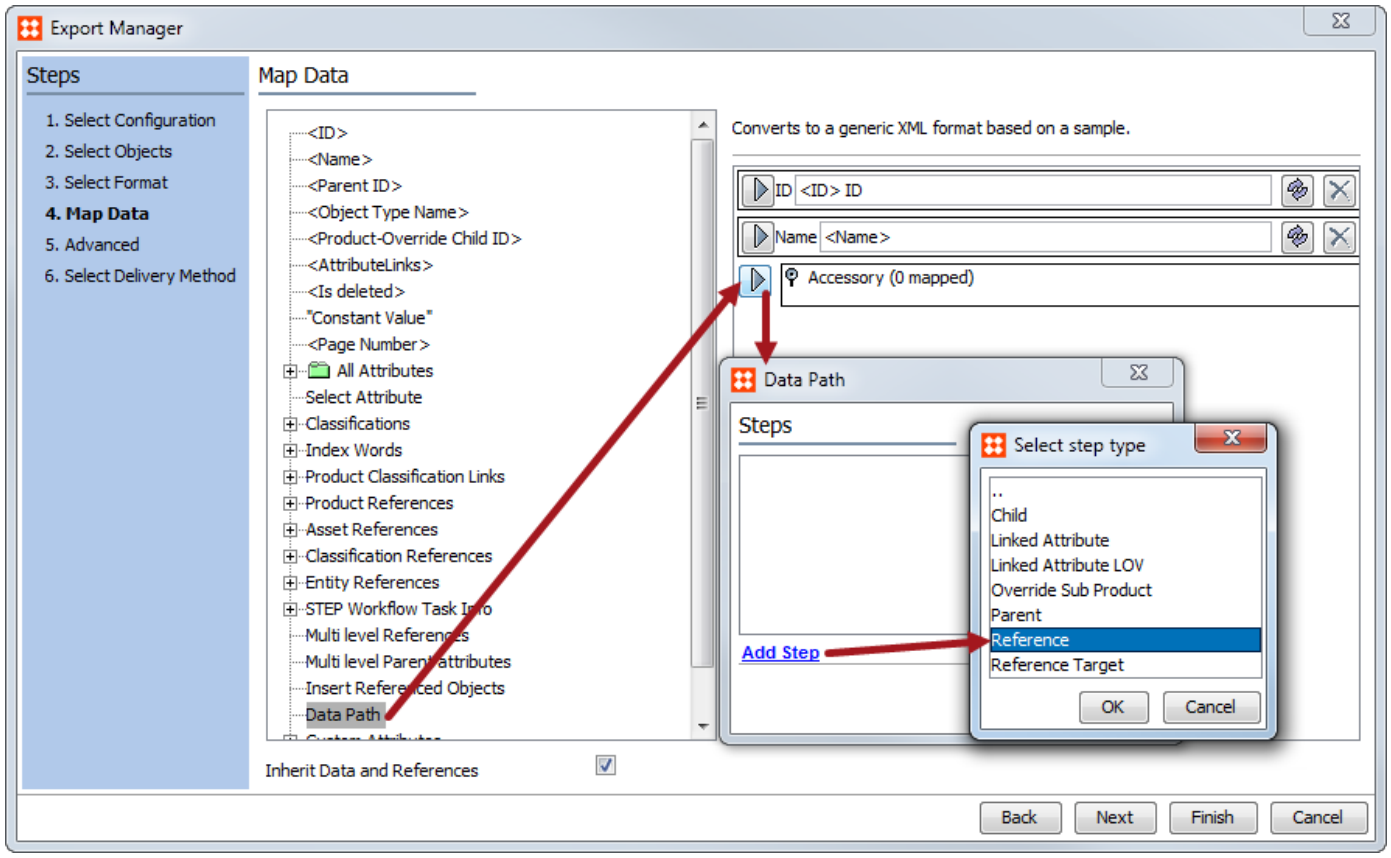
```
<Products>
  <Product>
    <ID>8060</ID>
    <Name>Glasses - 4</Name>
    <Accessory>
      <ID>120636</ID>
      <Name>Dessert Plate</Name>
      <Description>Set of 4 festive dessert plates.</Description>
      <Image>
        <ID>120650</ID>
        <Name>dessertMULTI_JPG</Name>
      </Image>
      <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

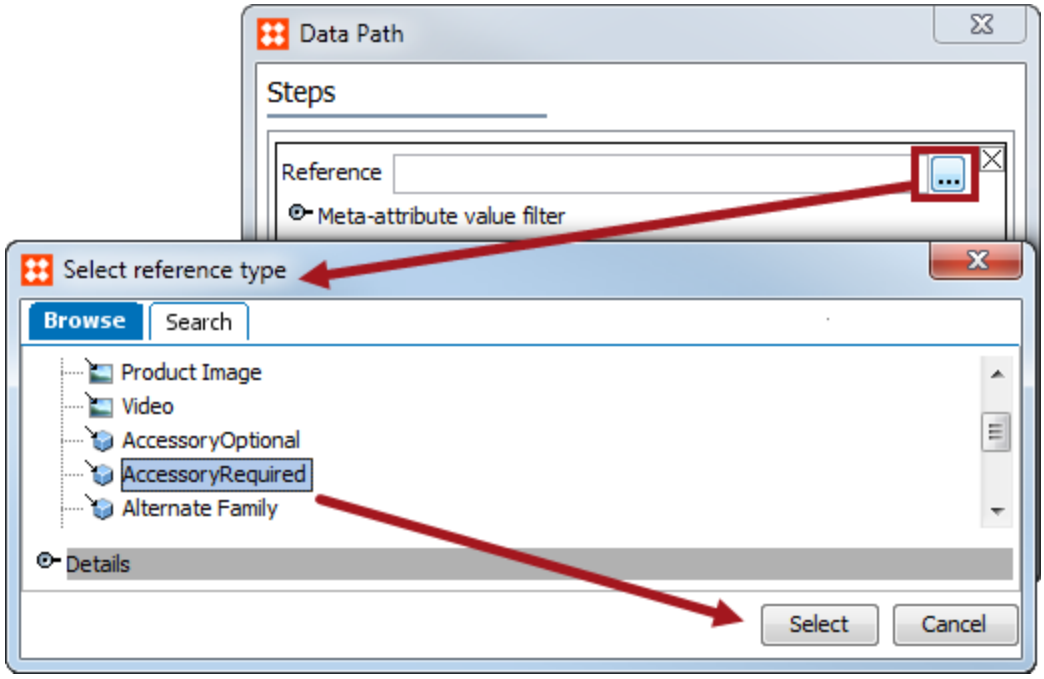
1. Map **ID** and **Name** for Level 1.



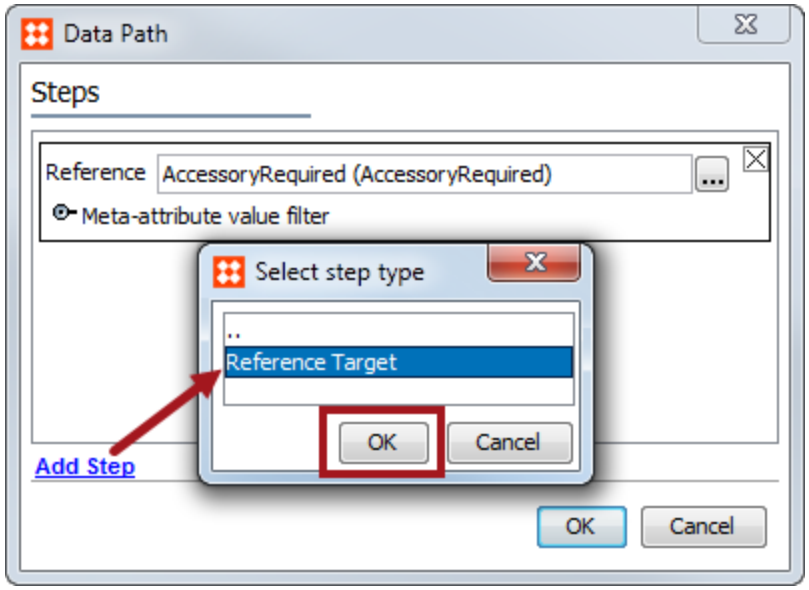
2. Map **Data Path** to Accessory, click the **Add Step** link, select the **Reference** step type, and click **OK**.



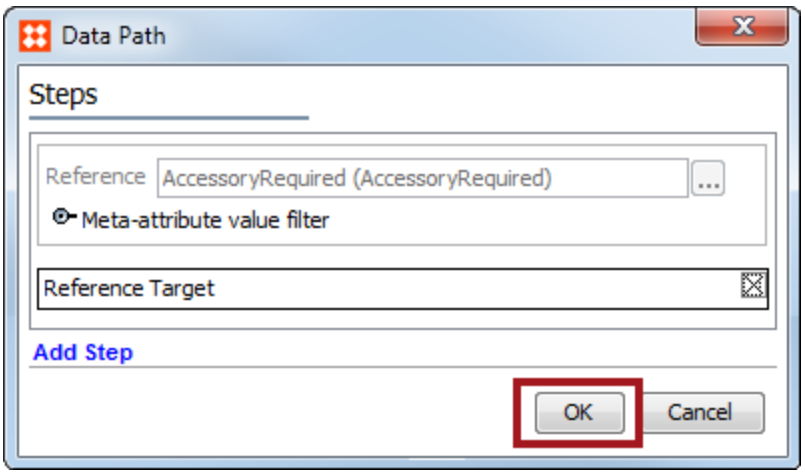
3. On the Data Path dialog, click the ellipsis button (...), select AccessoryRequired reference type, and click the **Select** button.



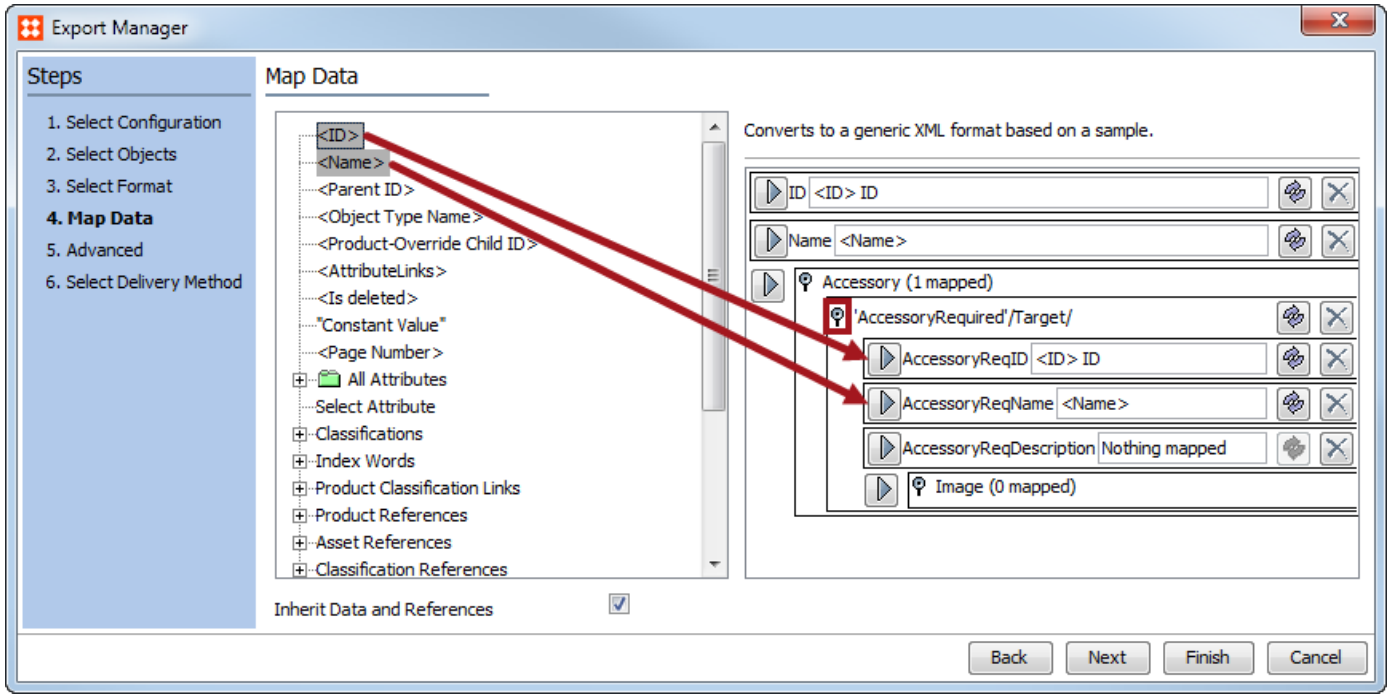
4. Click the **Add Step** link, select **Reference Target** for Accessory and click **OK**.



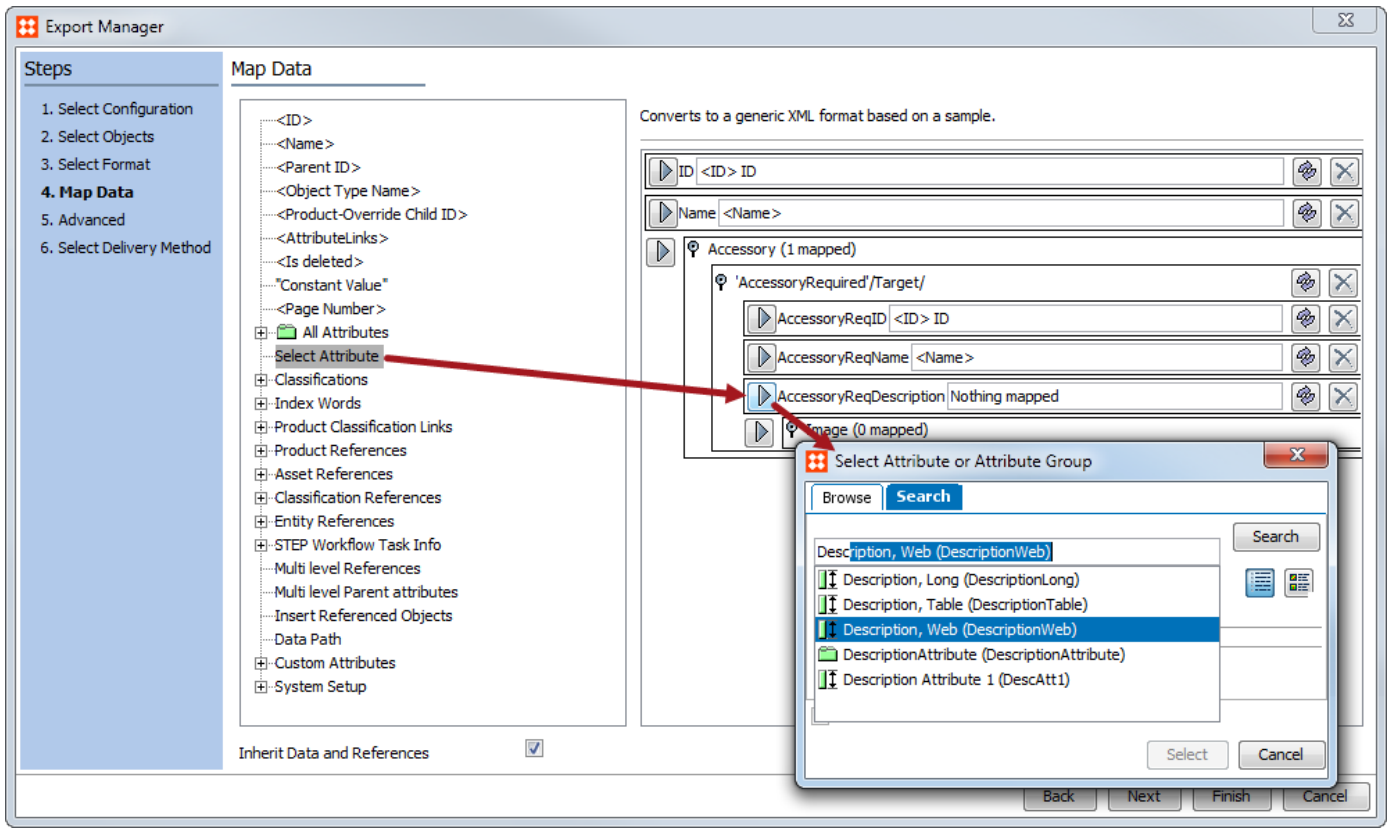
5. Click **OK** to confirm the Reference and Reference Target Data Path fields for the Accessory.



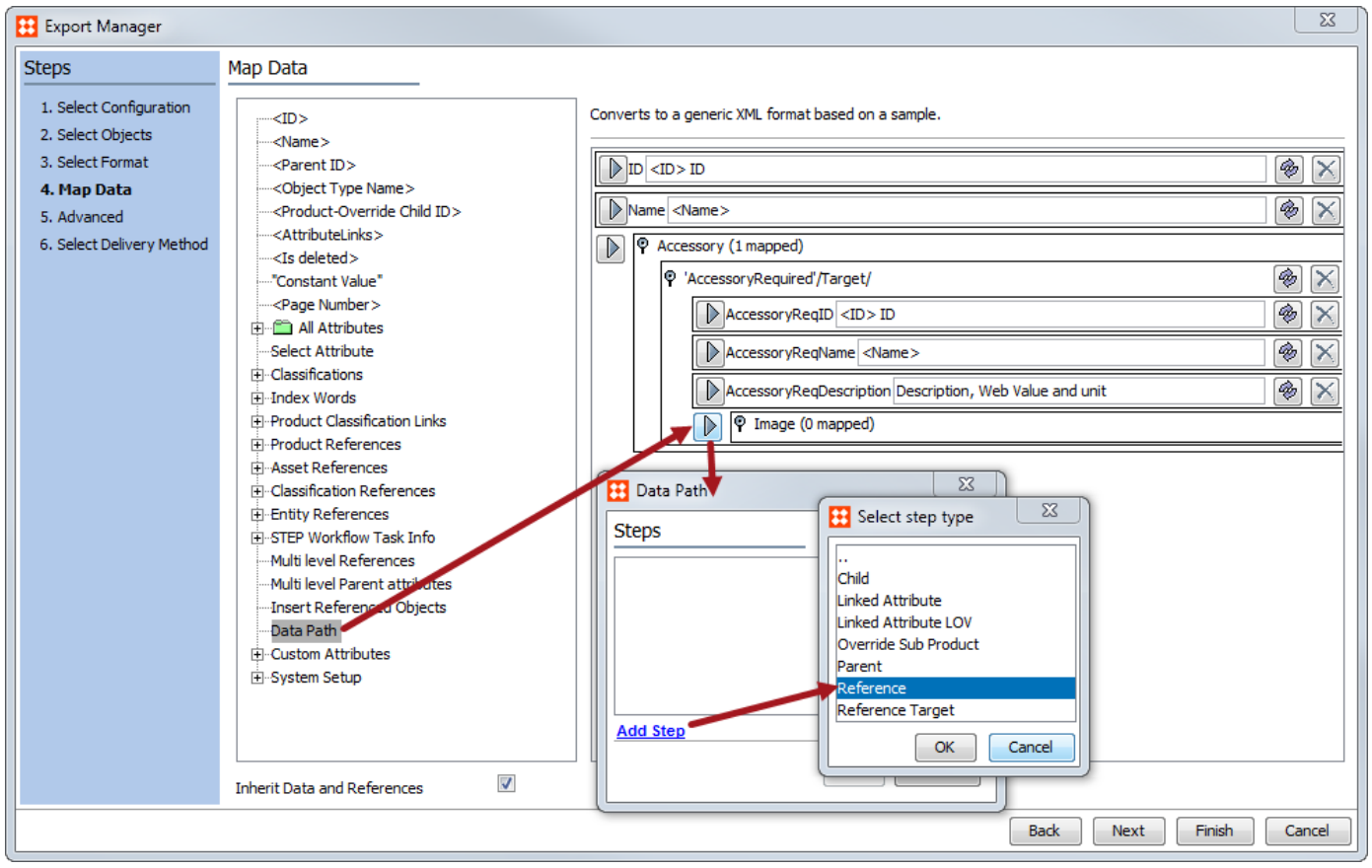
6. Open the 'AccessoryRequired'/Target/ flipper to display the nested mapping options. Map **ID** and **Name** for the Level 2 products being exported.



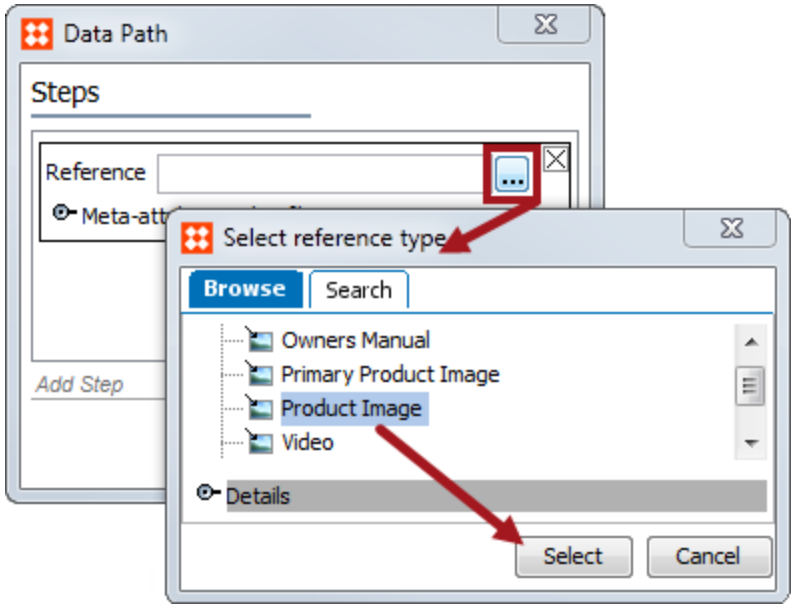
7. Use **Select Attribute** to map 'Description, Web' for Level 2 (products) and click the **Select** button.



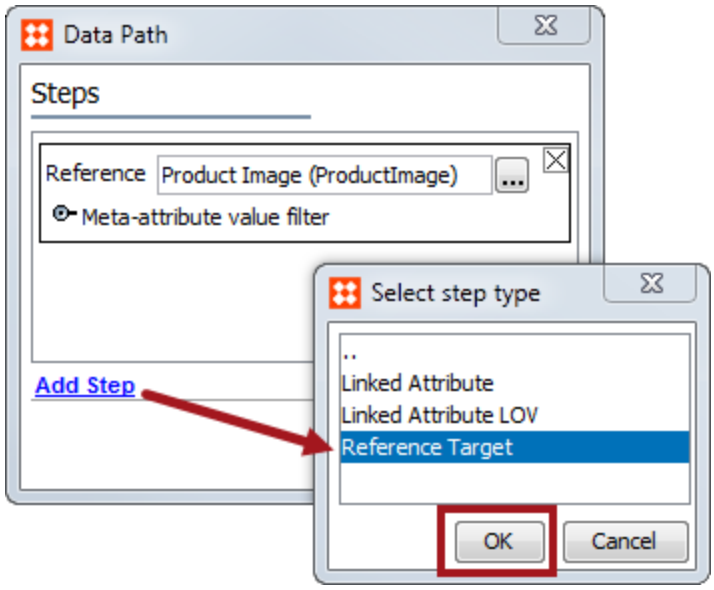
8. Map **Data Path** to Image, click the **Add Step** link, select the **Reference** step type, and click **OK**.



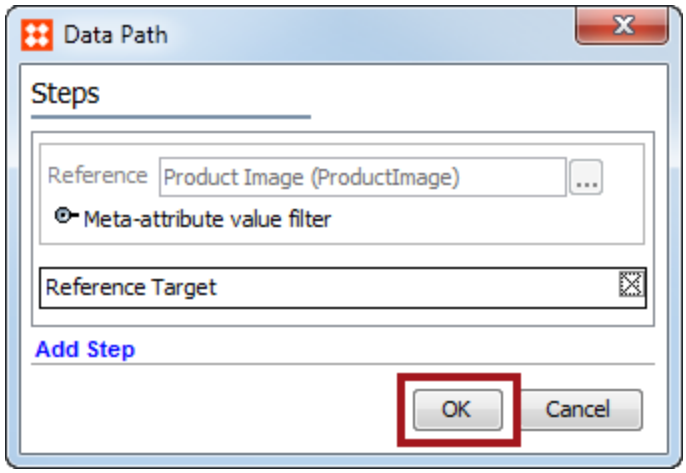
9. On the Data Path dialog, click the ellipsis button (...), select ProductImage reference type, and click the **Select** button.



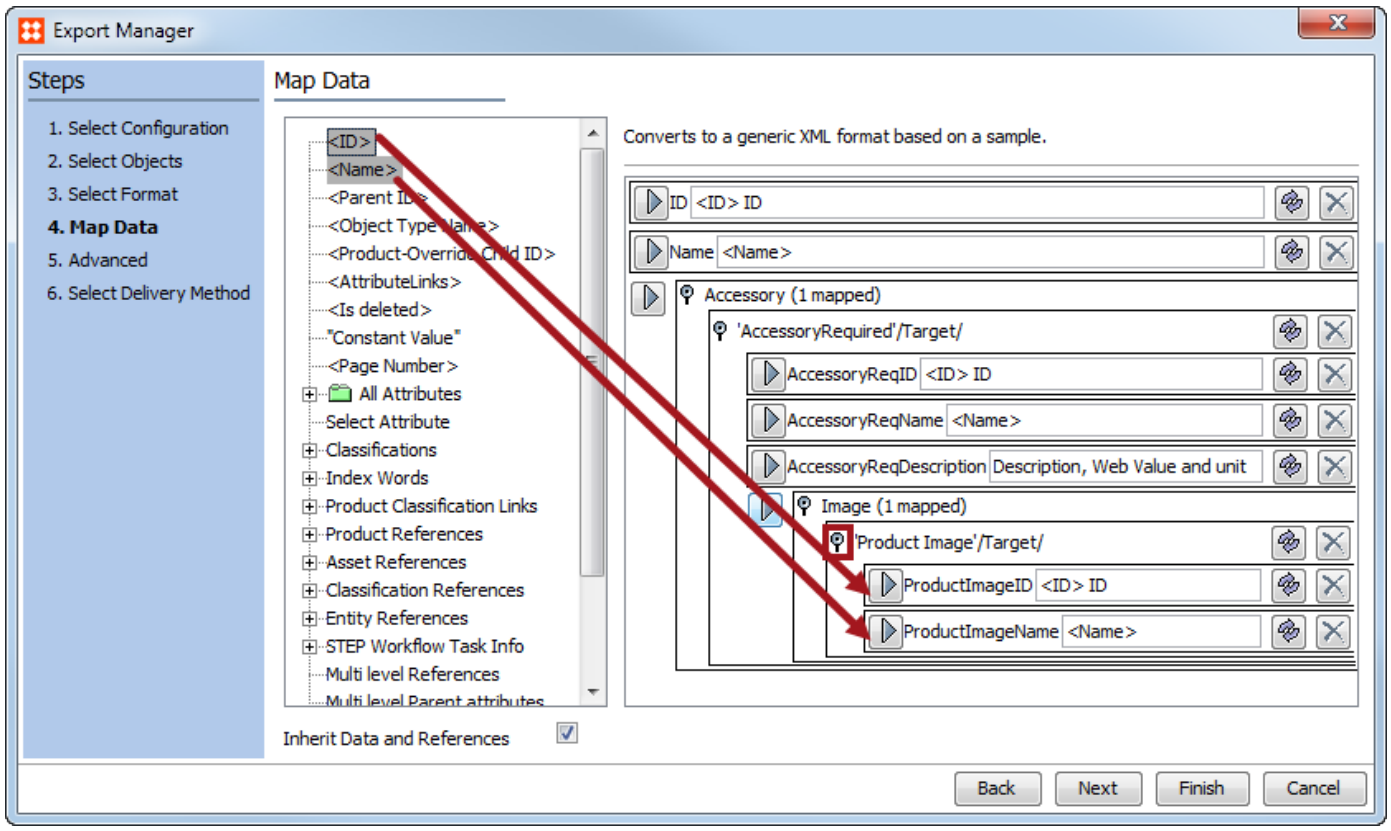
10. Click the **Add Step** link, select **Reference Target** for Image and click **OK**.



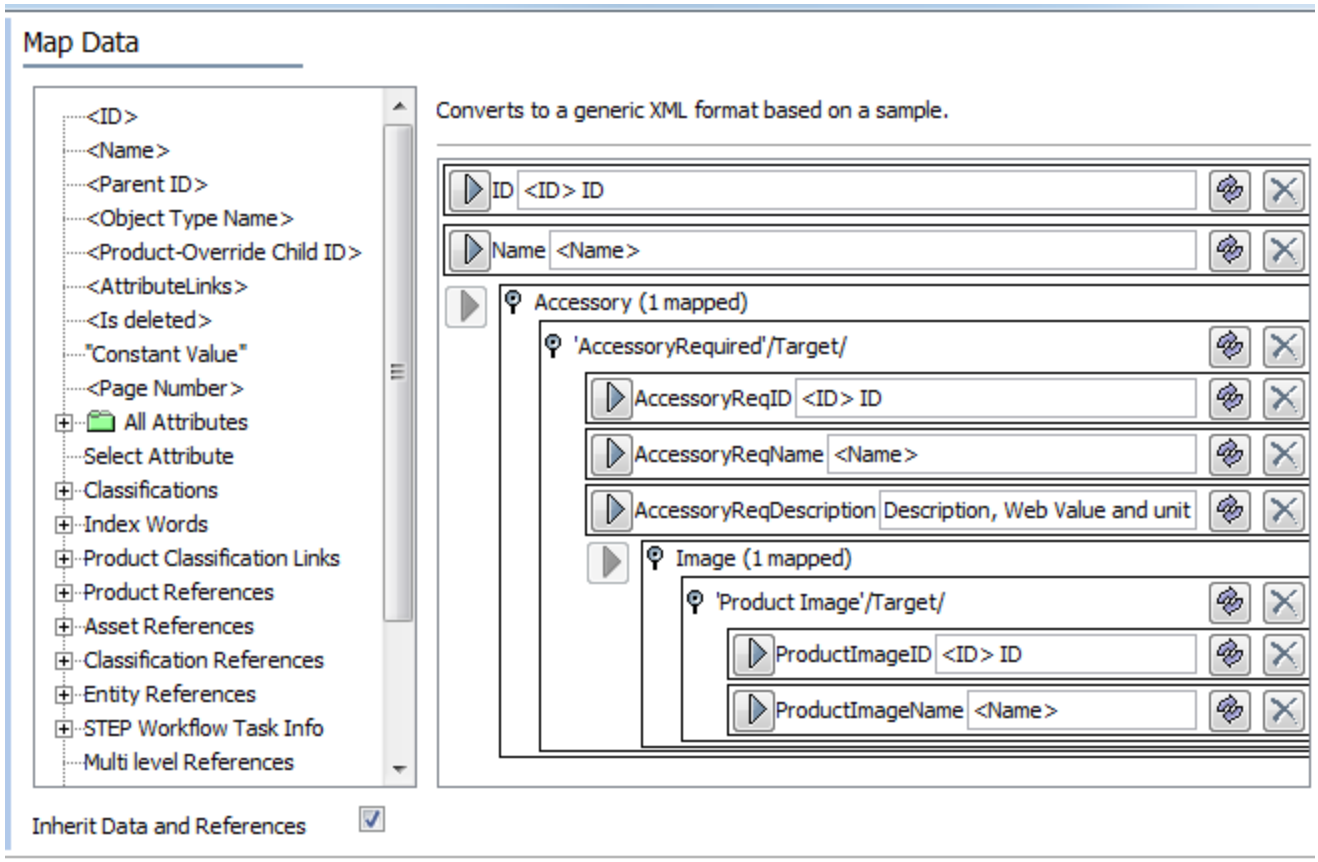
11. Click **OK** to confirm the Reference and Reference Target for the Image.



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

RemoveIfEmpty Processing Instructions in Generic XML

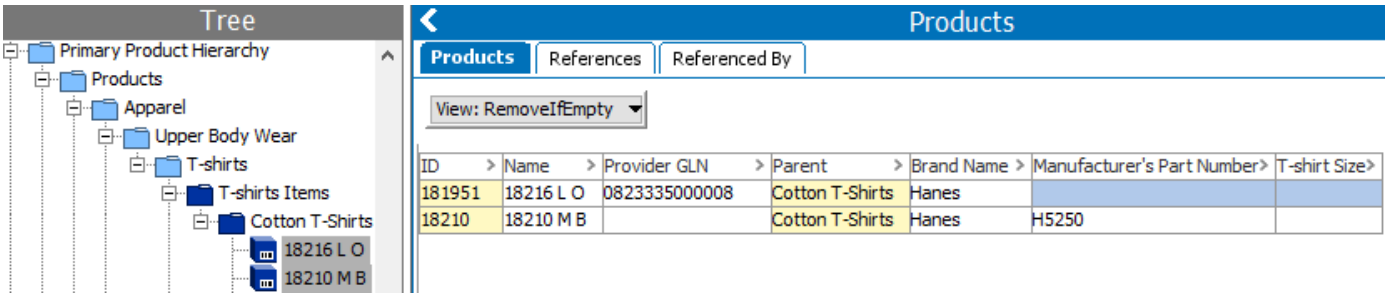
In an outbound Generic XML template, the Generic XML instruction **RemoveIfEmpty** makes it possible to remove the open and close tags surrounding a value if there is no value for any **one** of the attributes in the section.

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

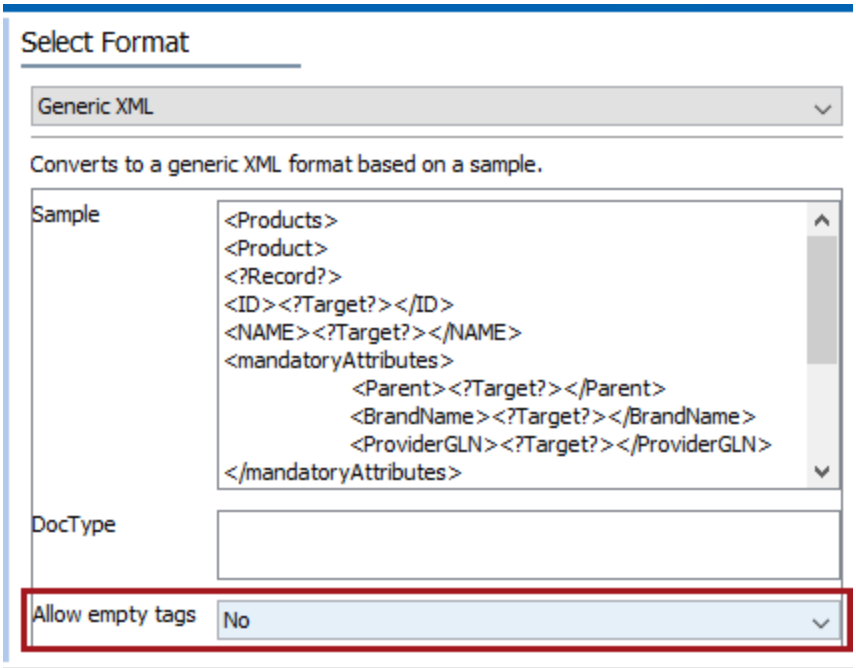
Choose **Generic XML** format and provide the following template text in the Sample field.

With RemoveIfEmpty	With 'RemoveIfEmpty recursive'
<Products> <Product> <?Record?> <ID><?Target?></ID> <NAME><?Target?></NAME> <mandatoryAttributes> <Parent><?Target?></Parent>	<Products> <Product> <?Record?> <ID><?Target?></ID> <NAME><?Target?></NAME> <mandatoryAttributes> <Parent><?Target?></Parent>

With RemoveIfEmpty	With 'RemoveIfEmpty recursive'
<BrandName><?Target?></BrandName> <ProviderGLN><?Target?></ProviderGLN> </mandatoryAttributes> <optionalAttributes> **<?RemoveIfEmpty?>** <PartNo><?Target?></PartNo> <Size><?Target?></Size> </optionalAttributes> </Product> </Products>	<BrandName><?Target?></BrandName> <ProviderGLN><?Target?></ProviderGLN> </mandatoryAttributes> <optionalAttributes> **<?RemoveIfEmpty recursive?>** <PartNo><?Target?></PartNo> <Size><?Target?></Size> </optionalAttributes> </Product> </Products>

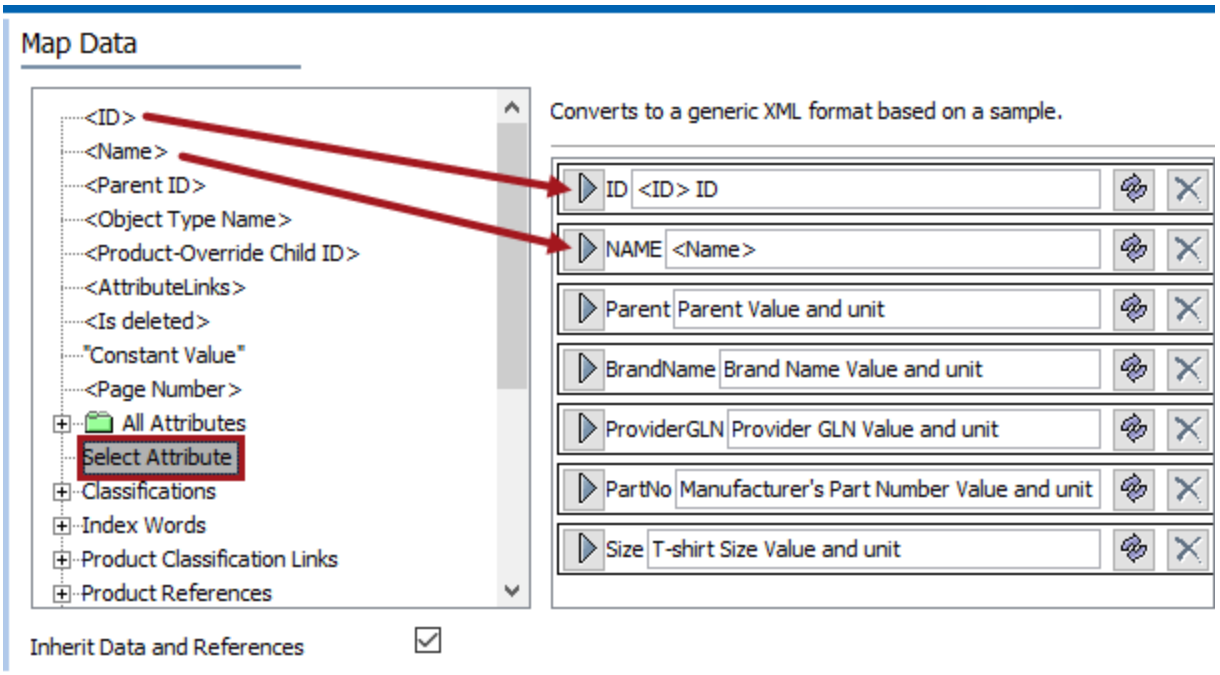
- The RemoveIfEmpty and/or 'RemoveIfEmpty recursive' tags are not used below the 'mandatoryAttributes' section since all attributes should be output, even if they are empty.
- The RemoveIfEmpty and/or 'RemoveIfEmpty recursive' tags are included below the 'optionalAttributes' section since that information 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

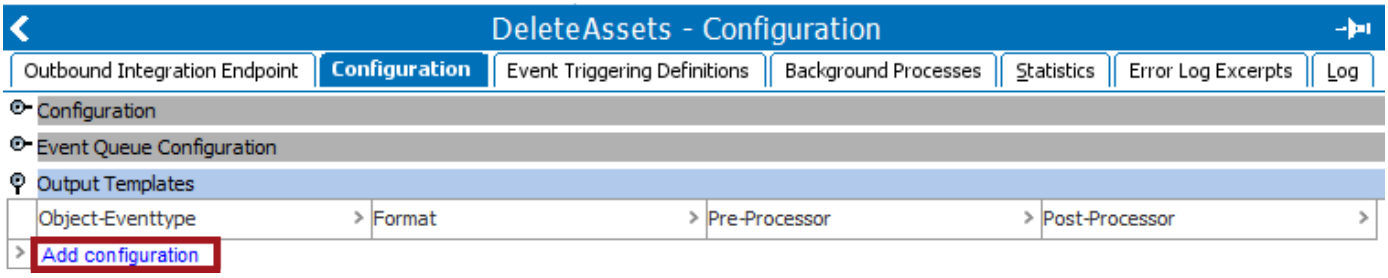
With RemoveIfEmpty	With 'RemoveIfEmpty recursive'
<Products> <Product> <ID>181951</ID> <NAME>18216 L O</NAME> <mandatoryAttributes> <Parent>Cotton T-Shirts</Parent> <BrandName>Hanes</BrandName> <ProviderGLN>0823335000008</ProviderGLN> </mandatoryAttributes> </Product> <Product> <ID>18210</ID> <NAME>18210 M B</NAME> <mandatoryAttributes> <Parent>Cotton T-Shirts</Parent> <BrandName>Hanes</BrandName> <ProviderGLN/> </mandatoryAttributes> </Product> </Products>	<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.

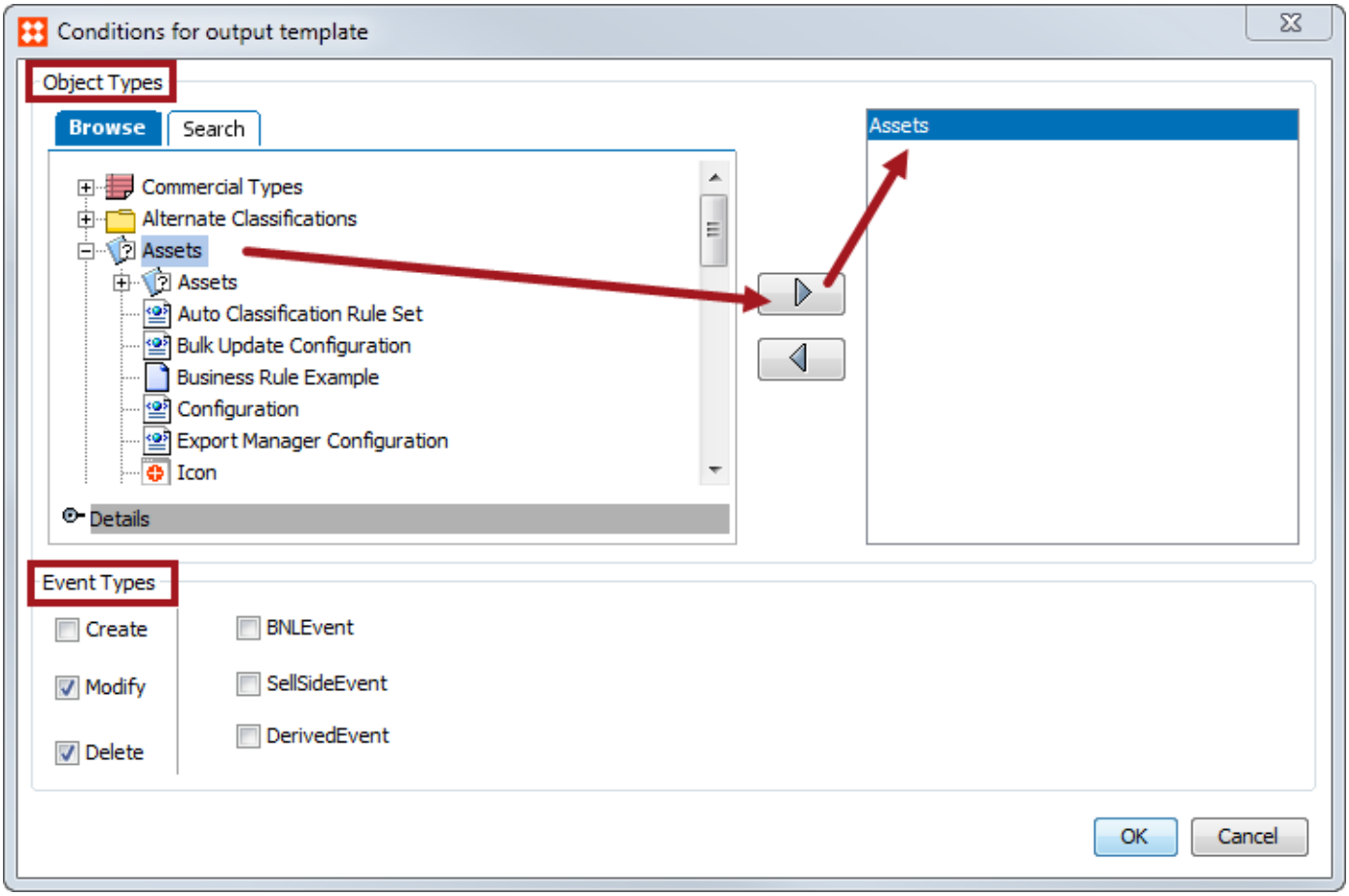
Object Type and Event Type Selection

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.



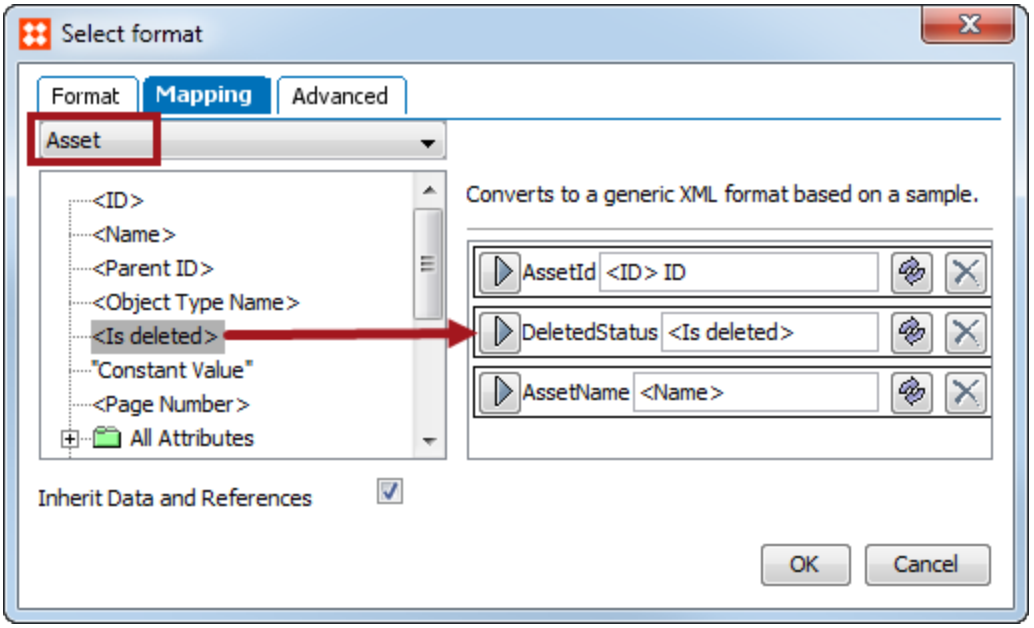
Template

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.

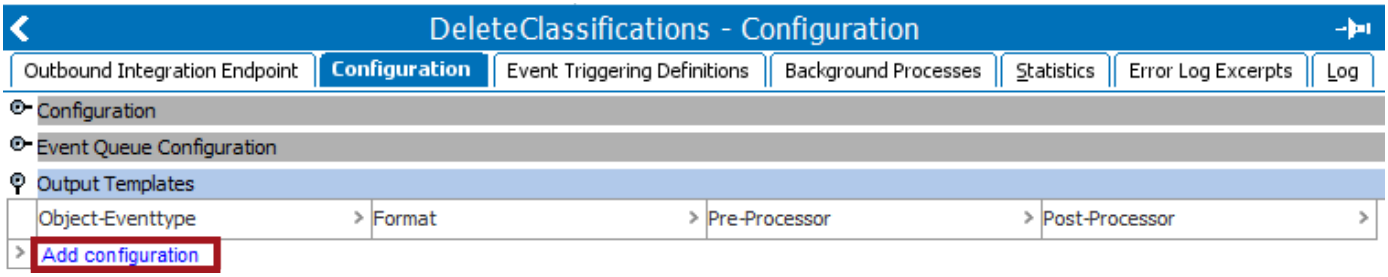
For modified Assets, the Deleted Status displays **" "**, which indicates the asset is not deleted. All mapped fields are exported for Create and Modify events.

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.

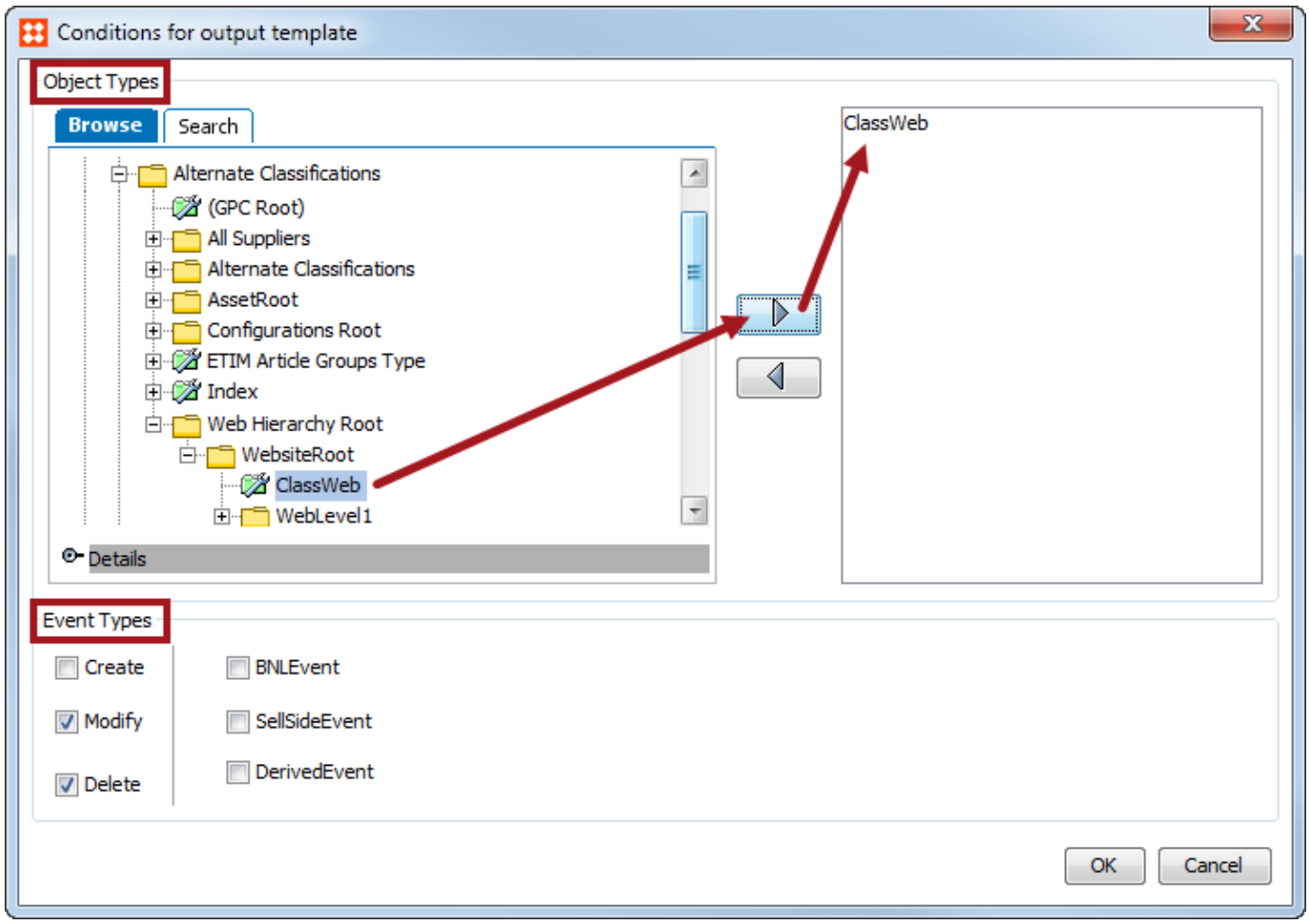
Object Type and Event Type Selection

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.



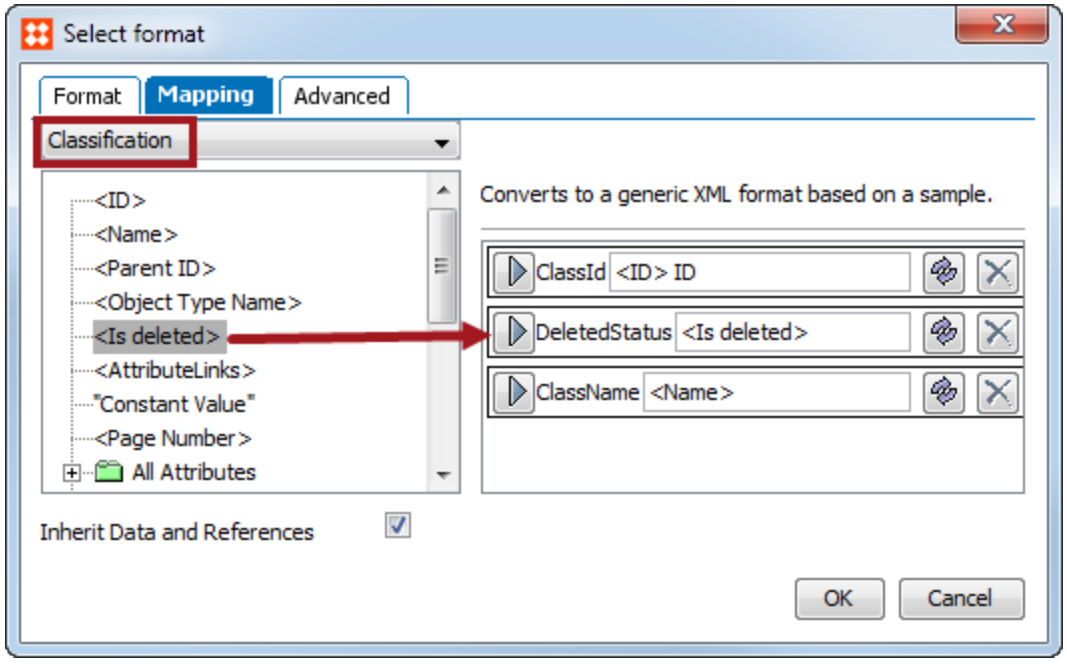
Template

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.

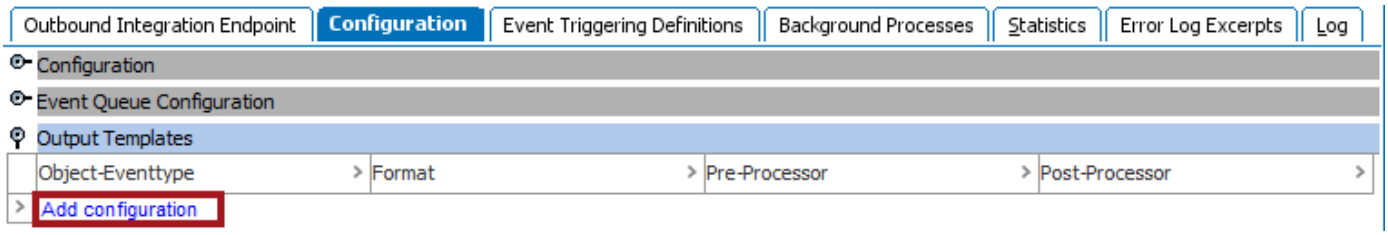
For modified Classifications, the Deleted Status displays **" "**, which indicates the classification is not deleted. All mapped fields are exported for Create and Modify events.

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.

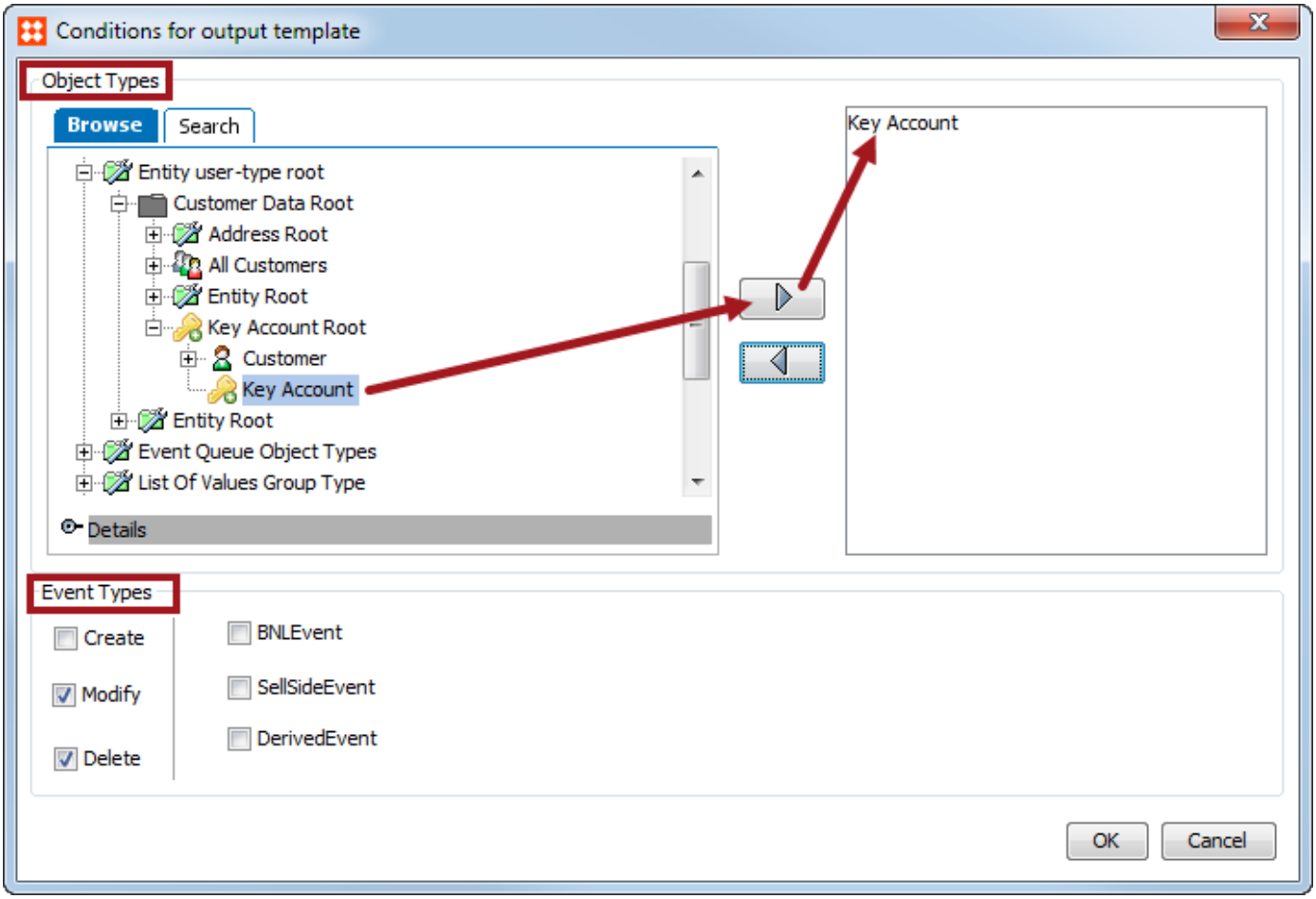
Object Type and Event Type Selection

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.



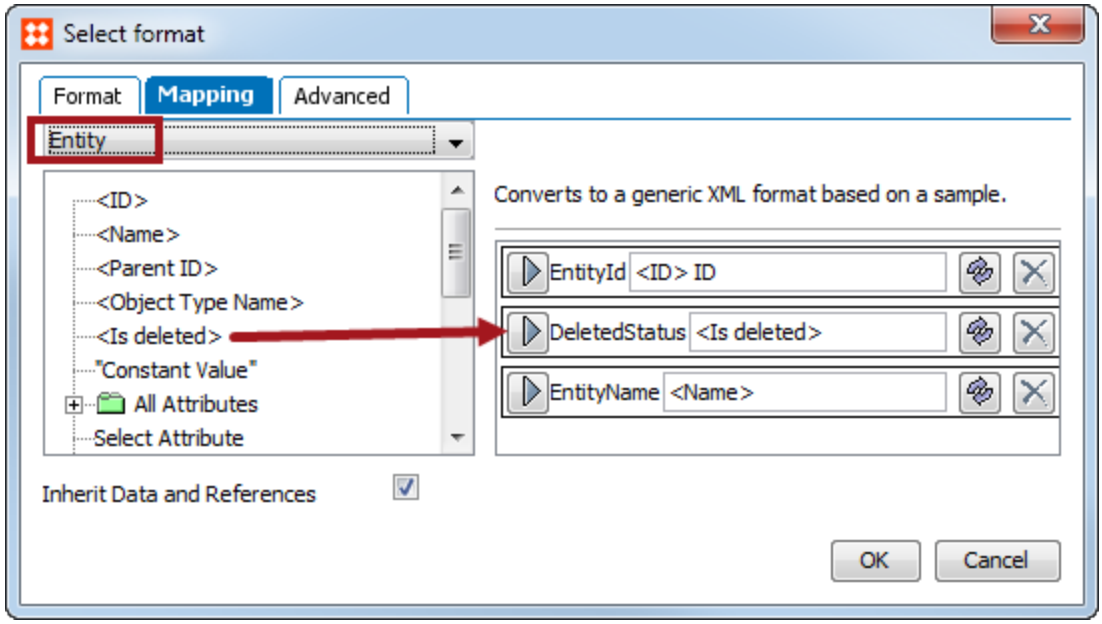
Template

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.

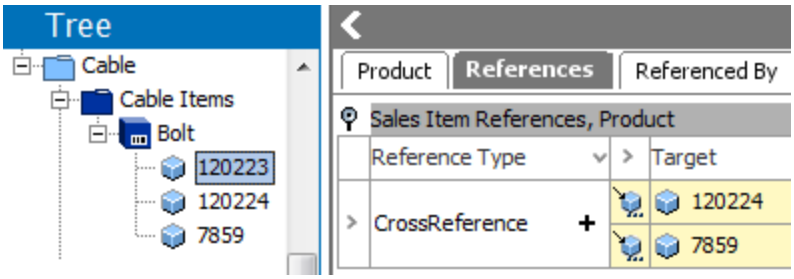
For modified Entities, the Deleted Status displays **" "**, which indicates the entity is not deleted. All mapped fields are exported for Create and Modify events.

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.

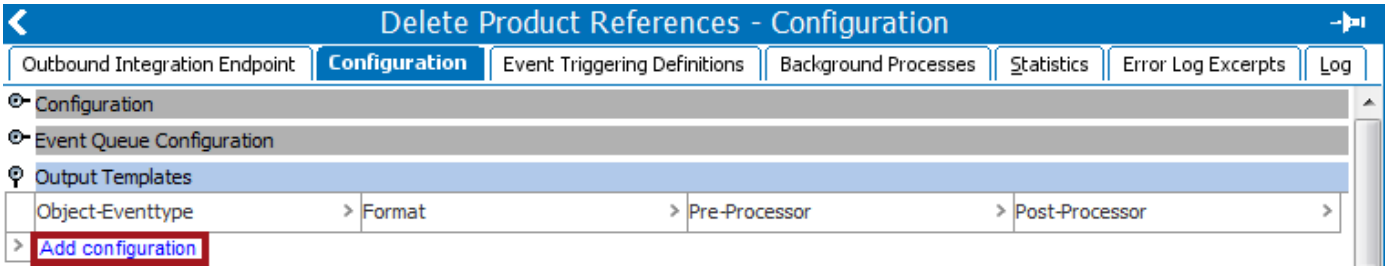
Product

On product 120223, the Reference Type of CrossReference to Target 7859 must be deleted in order to trigger the OIEP.

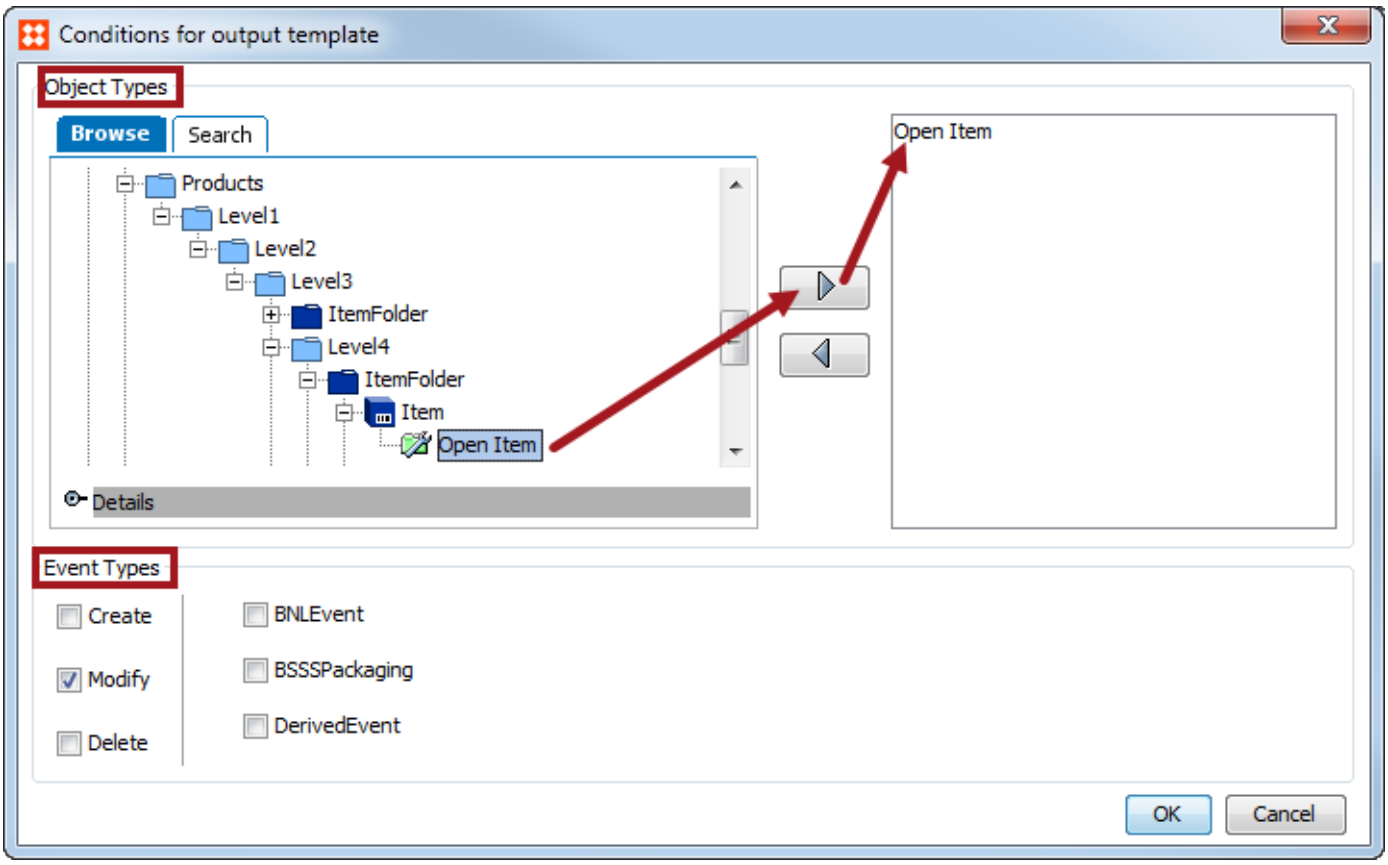


Object Type and Event Type Selection

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

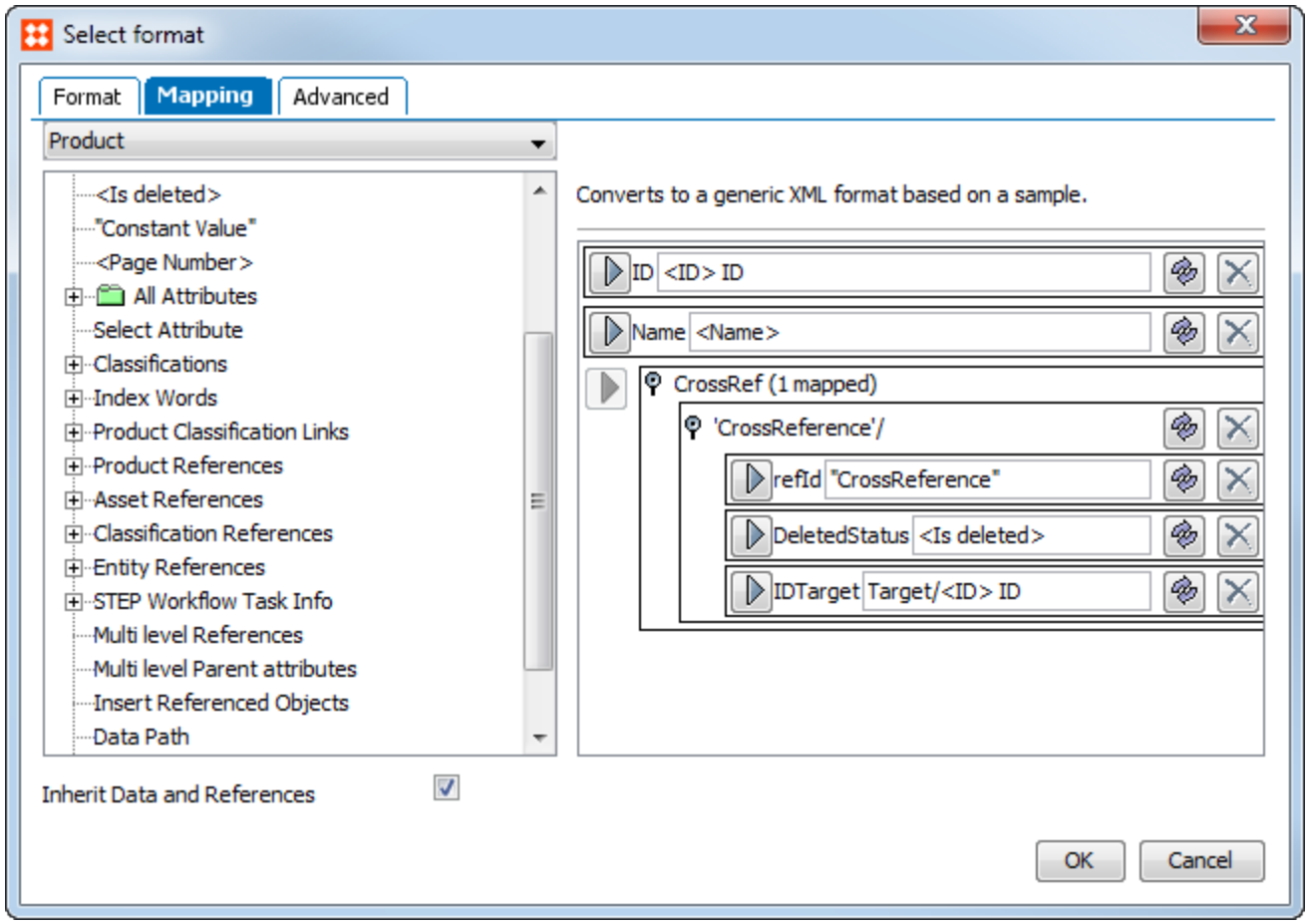
In the Output Template, click the ellipsis button (...), and set the format to **Generic XML**.

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, see 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:

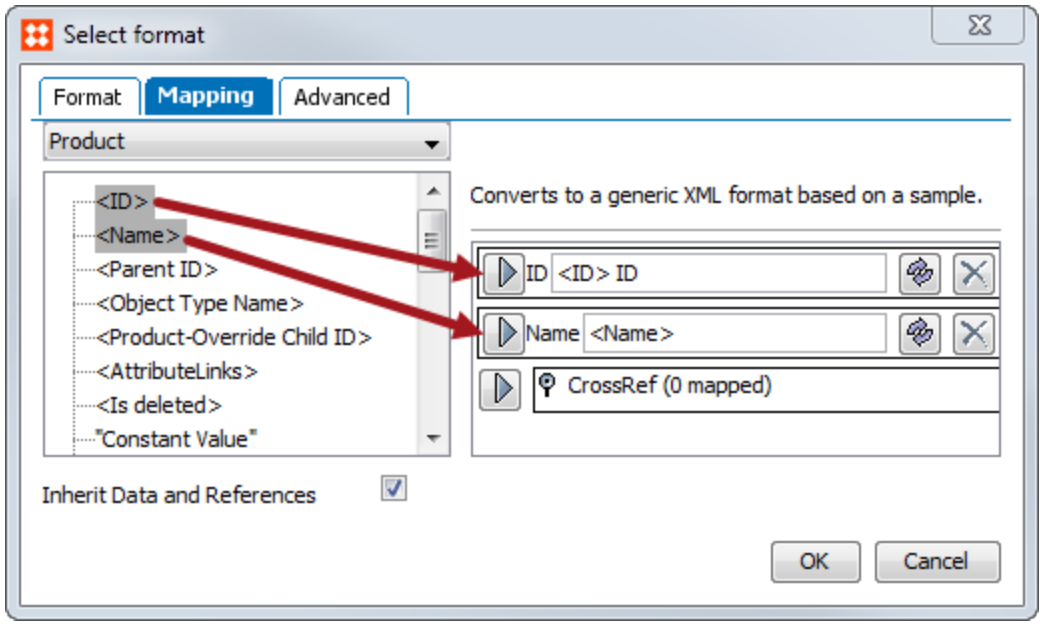
```
<Products>
  <Product ID="120223">
    <Name>120223</Name>
    <CrossRefs>
      <CrossRef refTypeID="CrossReference" DeletedStatus="">
        <IDTarget>120224</IDTarget>
      </CrossRef>
      <CrossRef refTypeID="CrossReference" DeletedStatus="IsDeleted">
        <IDTarget>7859</IDTarget>
      </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.

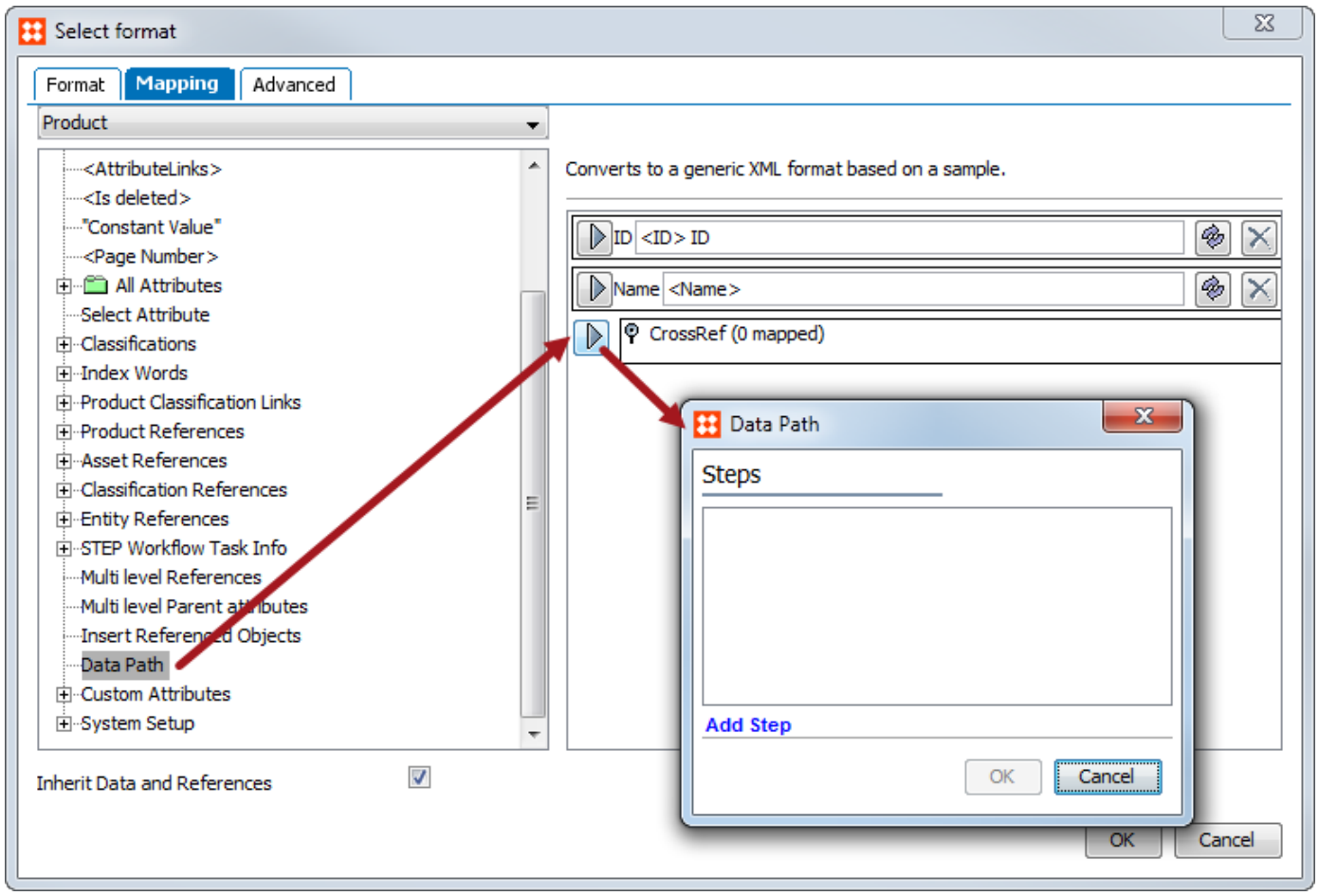
Mapping to Report Deleted Product References with a Generic XML OIEP

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.

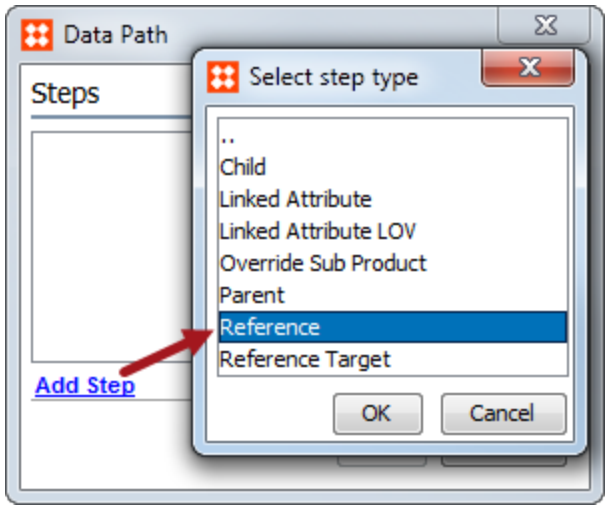
1. Map **ID** and **Name** for the Product.



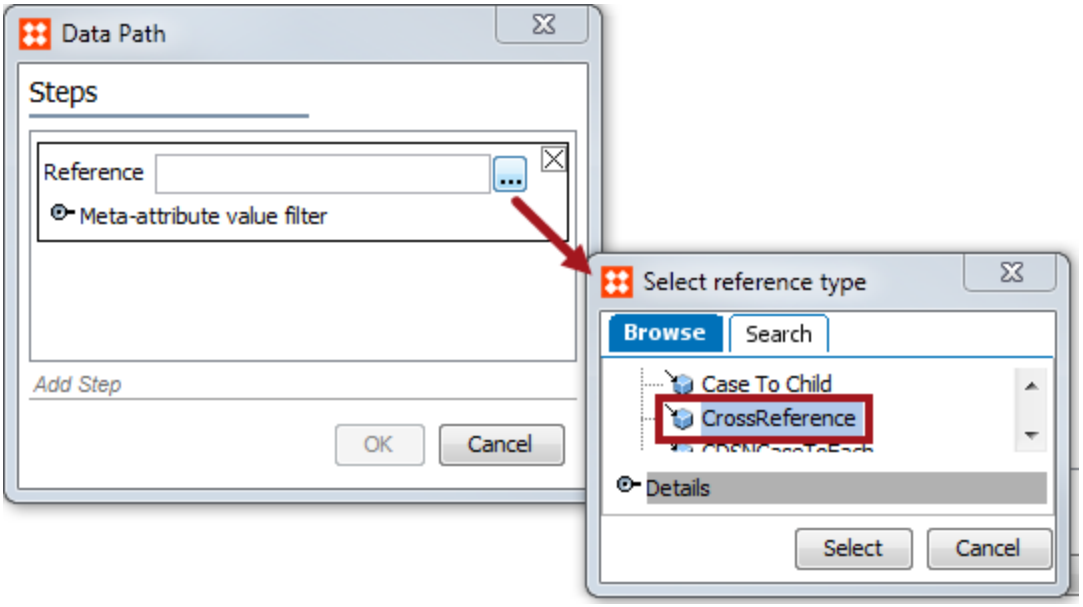
2. Map **Data Path** to CrossRef.



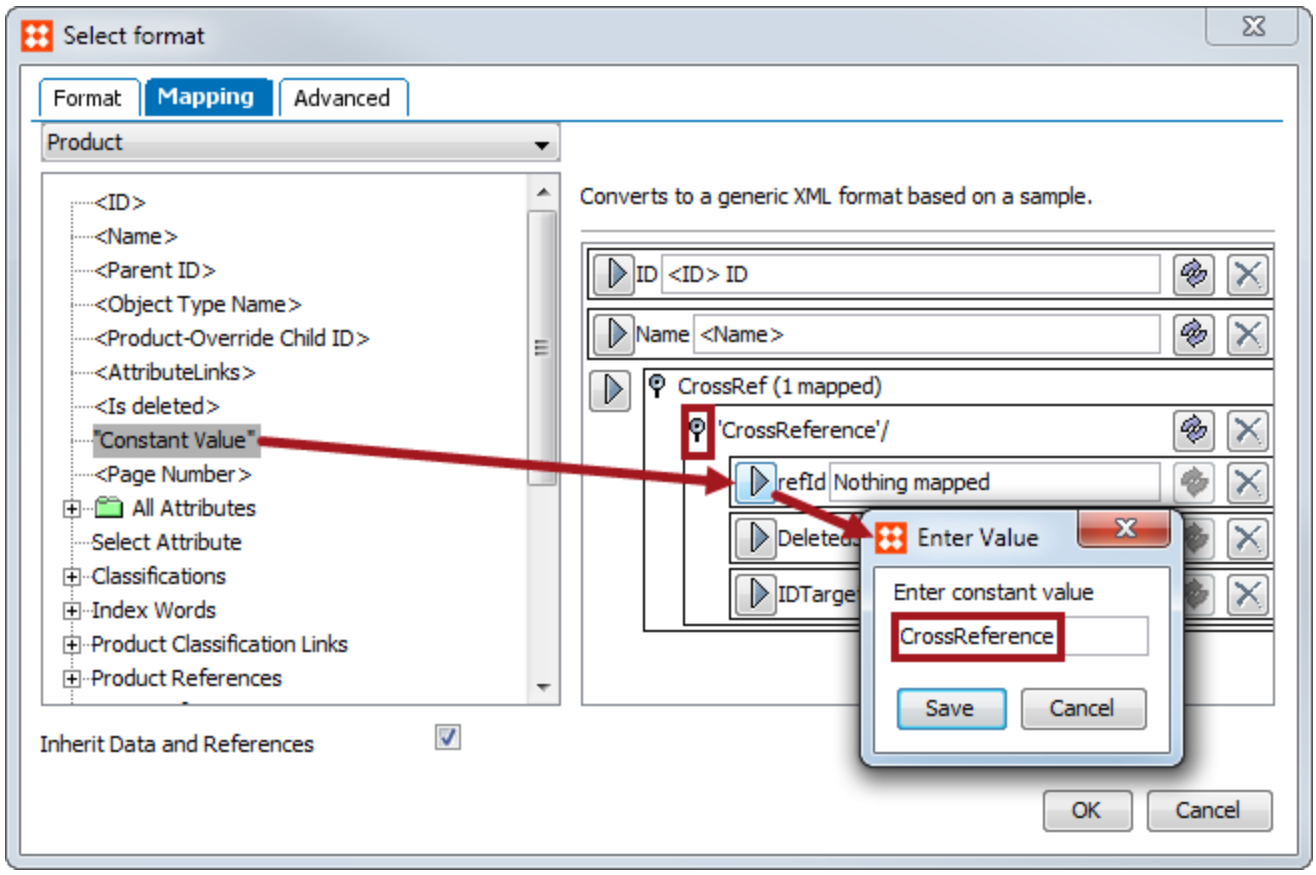
3. Click the **Add Step** link and select **Reference**.



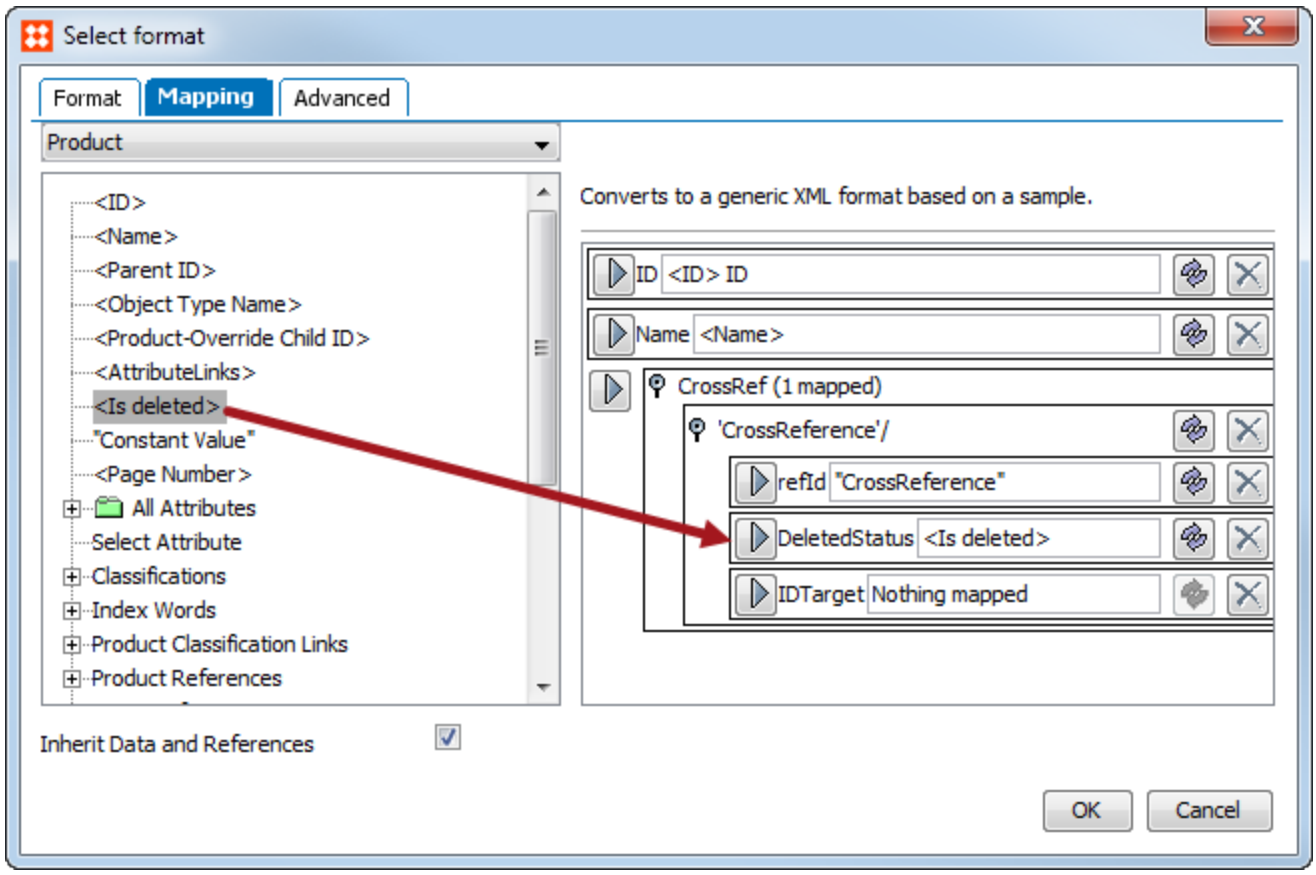
4. Click **OK** to close the **Select step type** dialog.
5. On the Data Path dialog, click the ellipsis button (...) and select the **Reference Type** to monitor, then click the **Select** button.



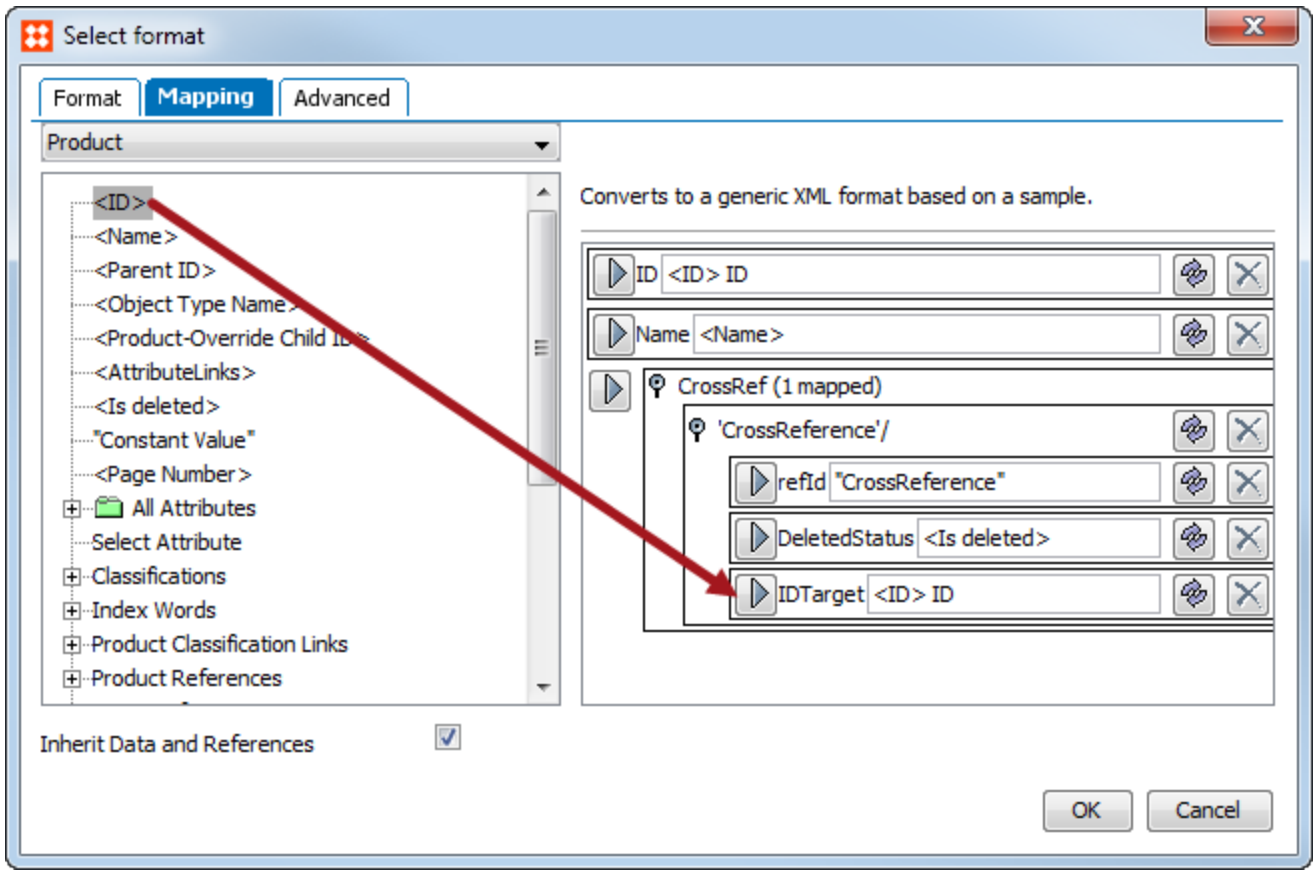
6. Click **OK** to close the **Data Path** dialog.
7. Open the 'CrossReference'/ flipper, map **Constant Value** to refID, add the text, and click the **Save** button.



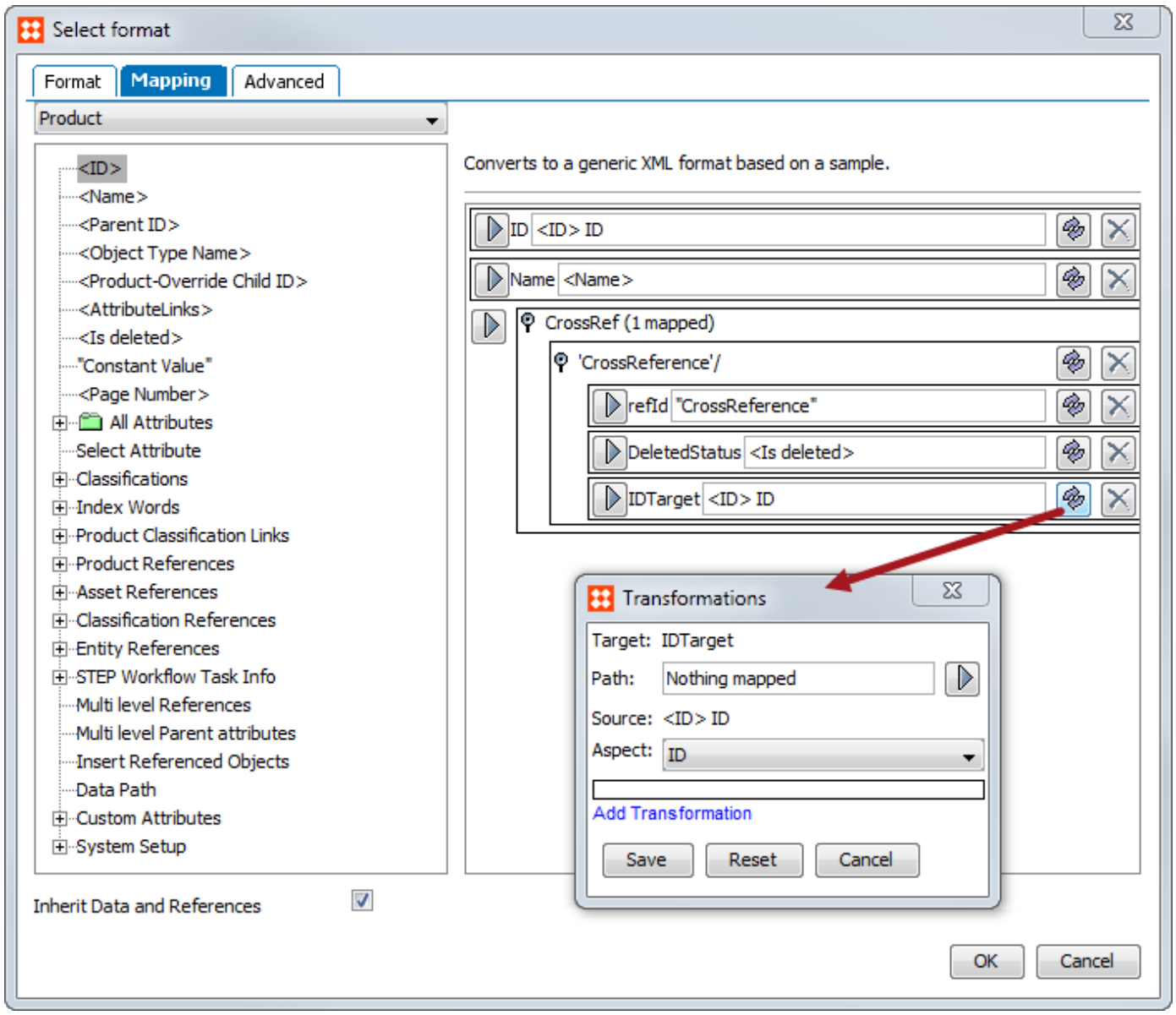
8. Map **<Is deleted>** to DeletedStatus.



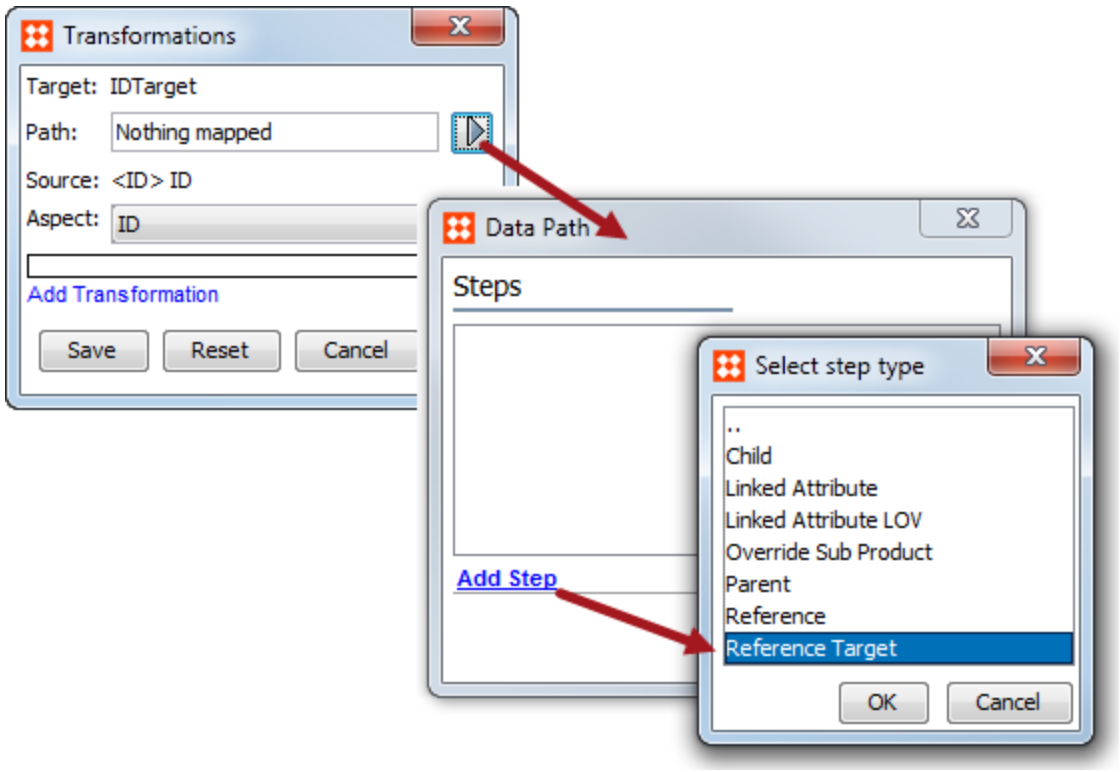
9. Map **ID** to IDTarget.



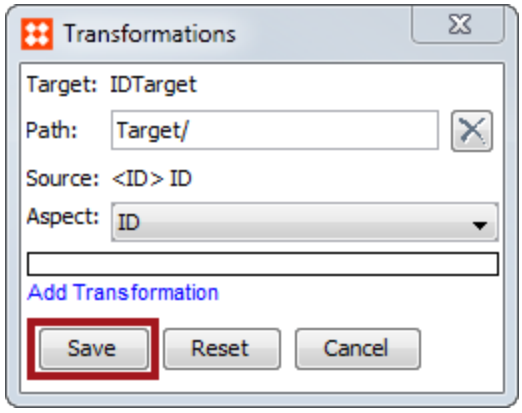
10. Click the **Transformations** button.



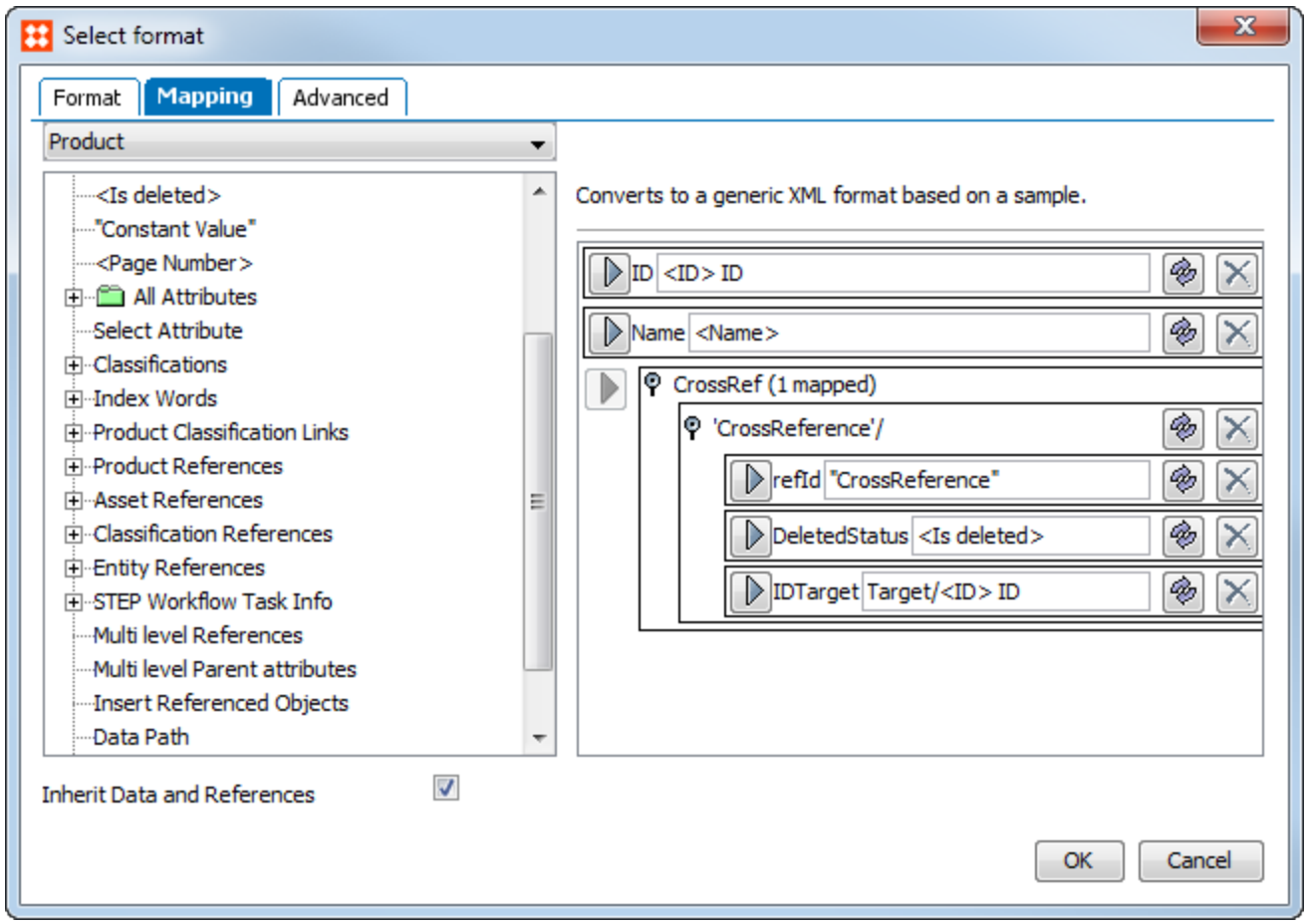
11. Click the **Path** button, click the **Add Step** link, and select **Reference Target**.



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.

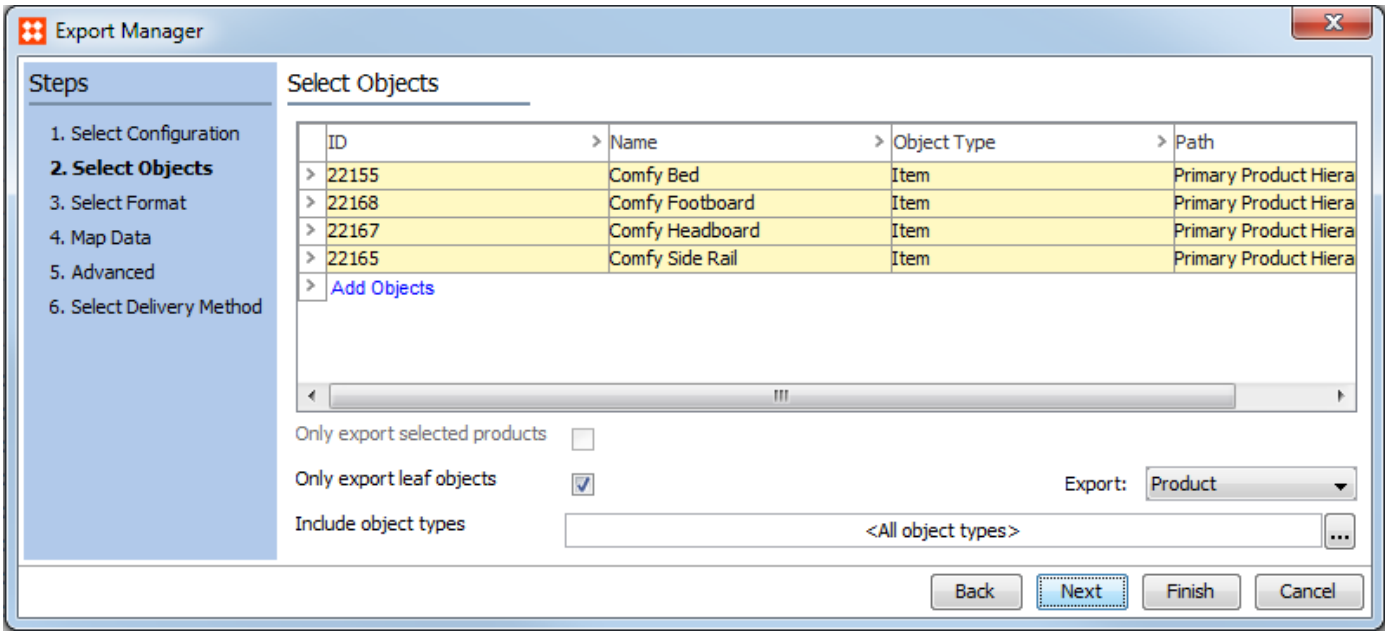

Select format
Format
Mapping
Advanced
Product
<Is deleted>
"Constant Value"
<Page Number>
All Attributes
Select Attribute
Classifications
Index Words
Product Classification Links
Product References
Asset References
Classification References
Entity References
STEP Workflow Task Info
Multi level References
Multi level Parent attributes
Insert Referenced Objects
Data Path
Converts to a generic XML format based on a sample.
ID <ID> ID
Name <Name>
CrossRef (1 mapped)
'CrossReference'/
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.

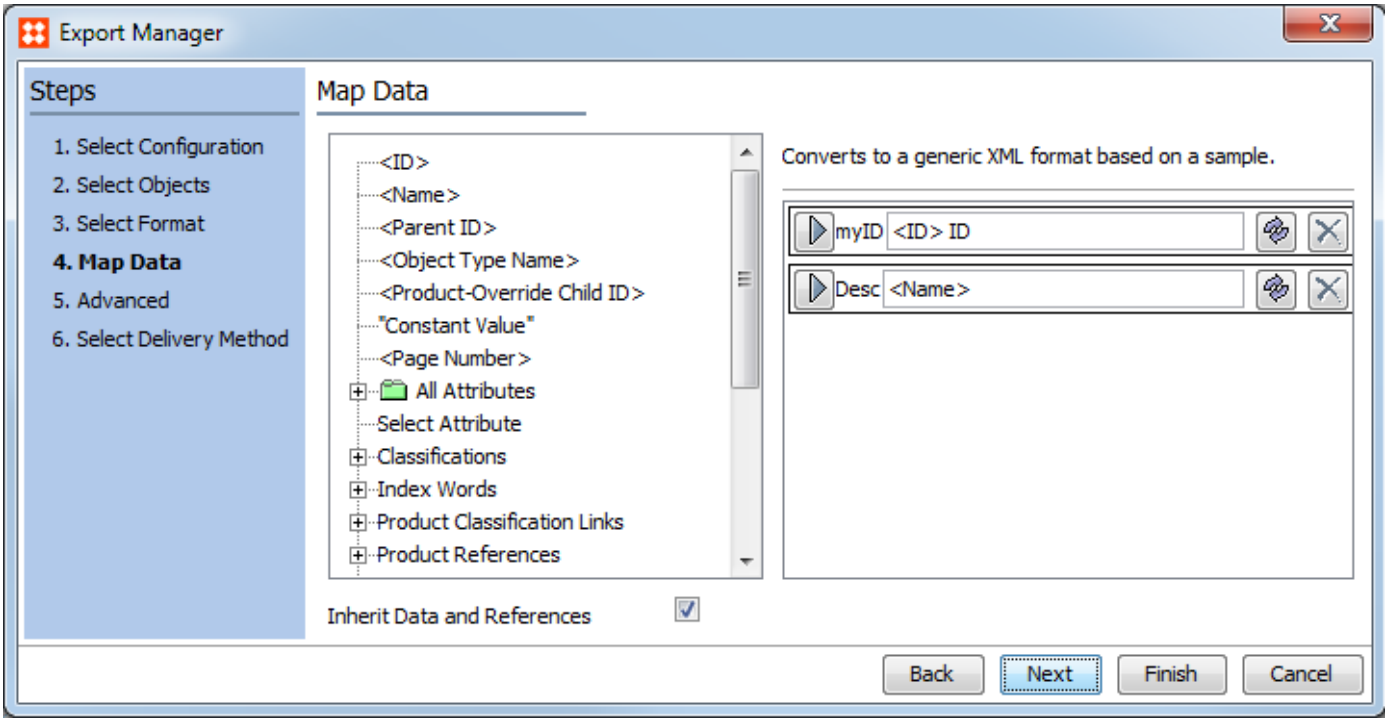
Objects



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

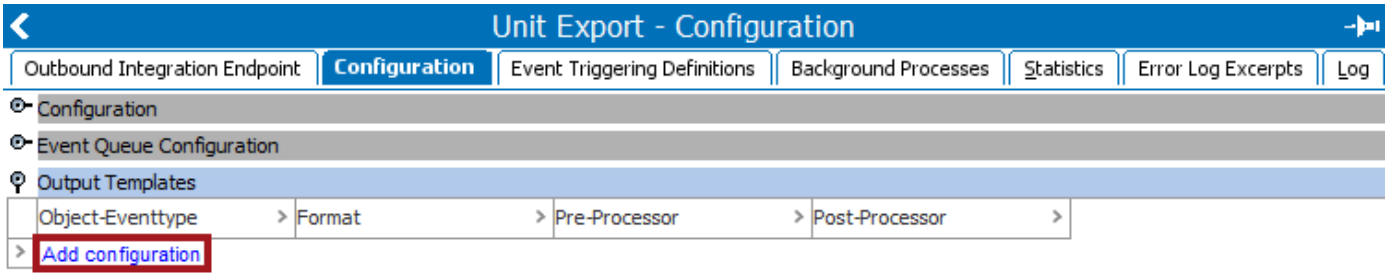
- There must be exactly one <?Record?> in the template.
- Mapping targets may only be used inside <?Record?> scope.
- Each mapping target must have a unique label.

Units with a Generic XML OIEP

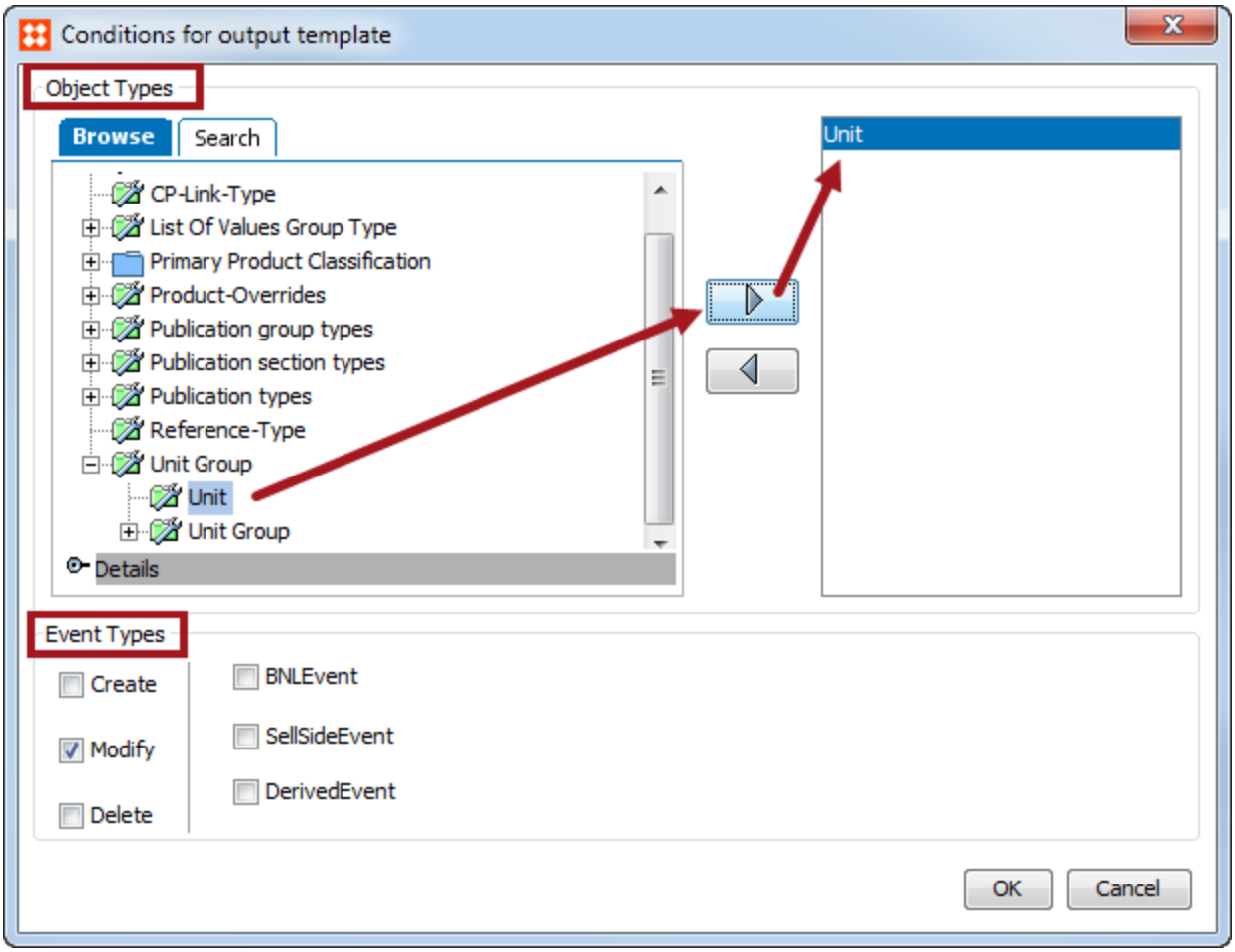
Outbound Integration Endpoints (OIEPs) allow the export of unit information.

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

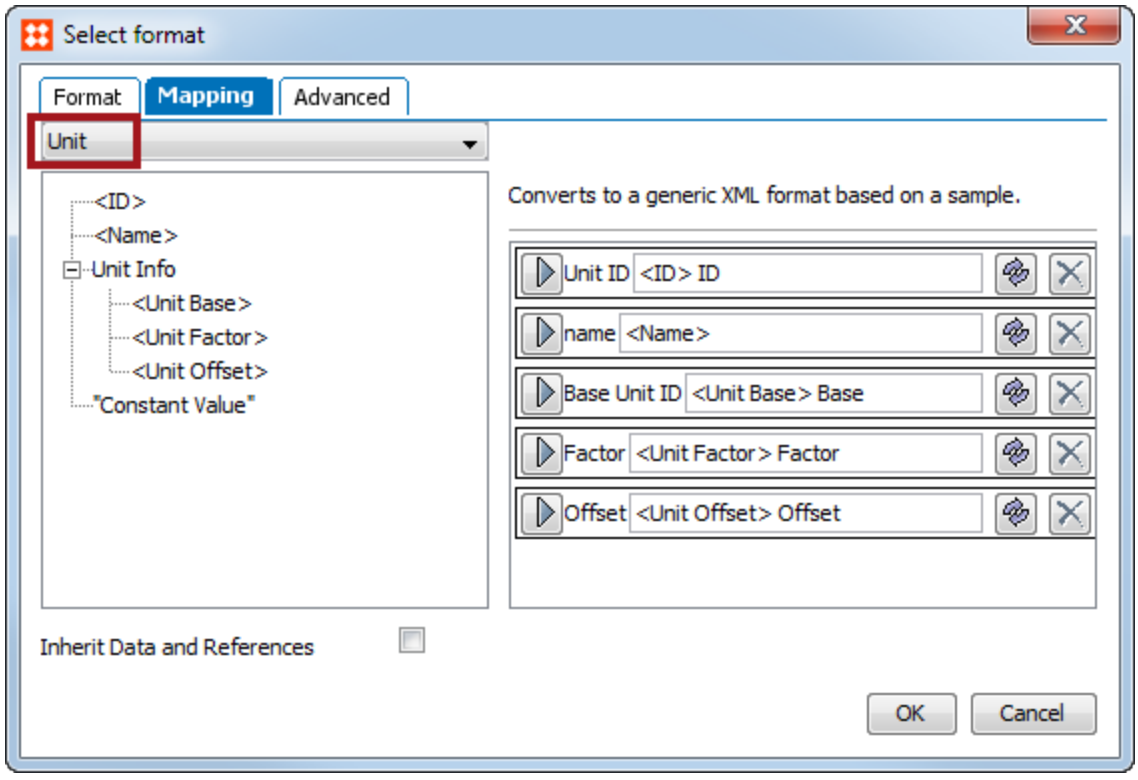
In the Output Template, click the ellipsis button (...), and set the format to **Generic XML**.

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.



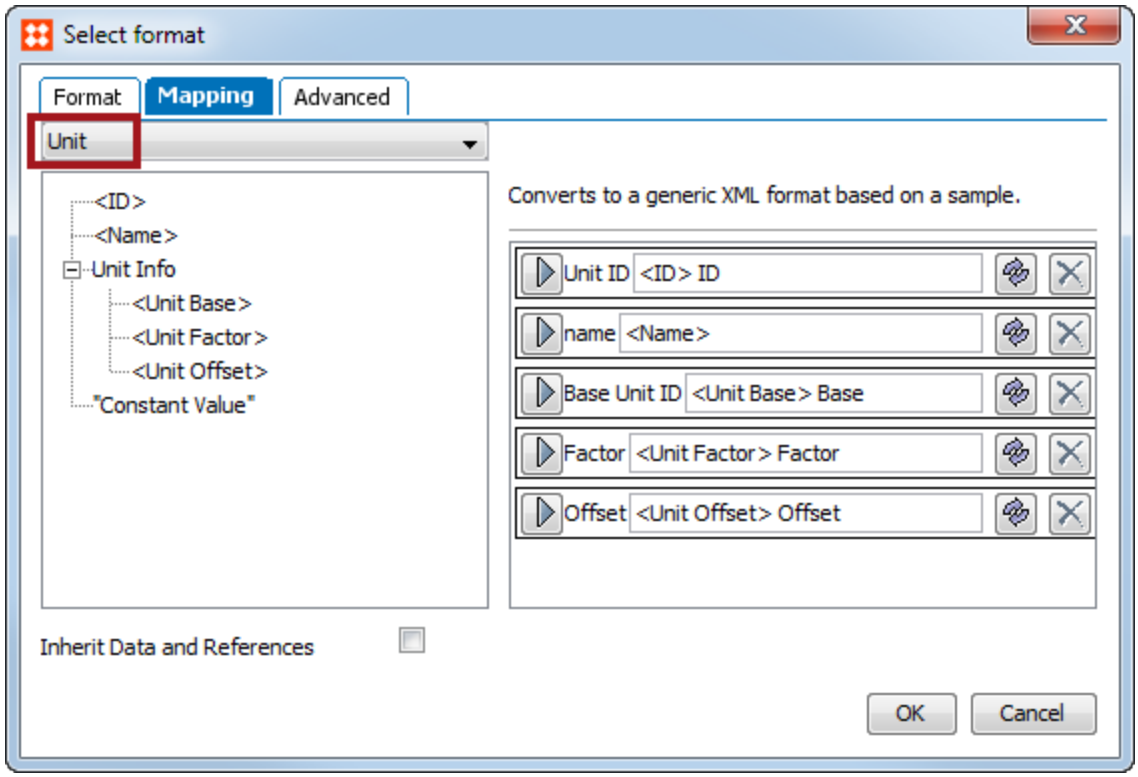
See the **Mapping Units Aspects with Generic XML OIEP** section for more information.

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.

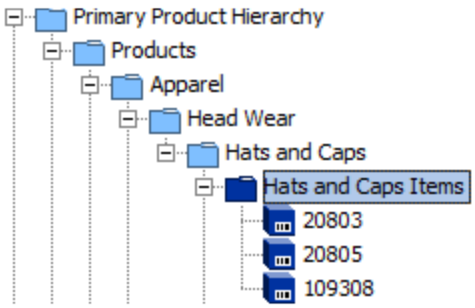
Aspect	Description
	For example, if a unit has a base unit conversion of 'value(g) = .001 * value(kg),' then '.001' is extracted.
Unit Offset	Extracts the conversion factor that should be added to a value with the unit to get the value in the base unit. For example, if a unit has a base unit conversion 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.

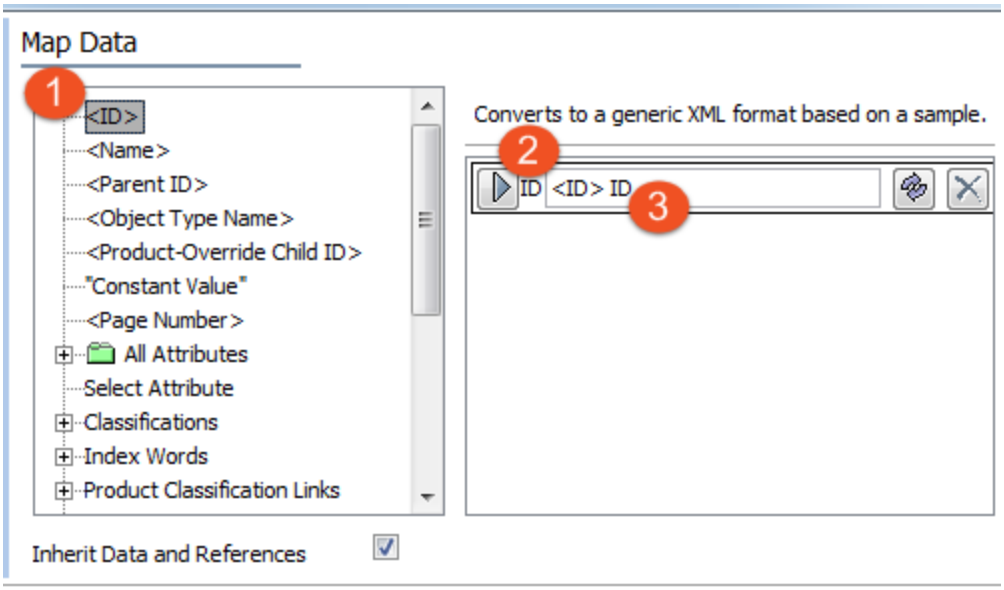


Template

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 - see **Creating an Inbound Integration Endpoint**
- Import Manager - see **Creating a Data Import**
- Export Manager - see **Creating a Data Export**
- OIEP - see **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, see **Data Mapping**.

Inbound Data

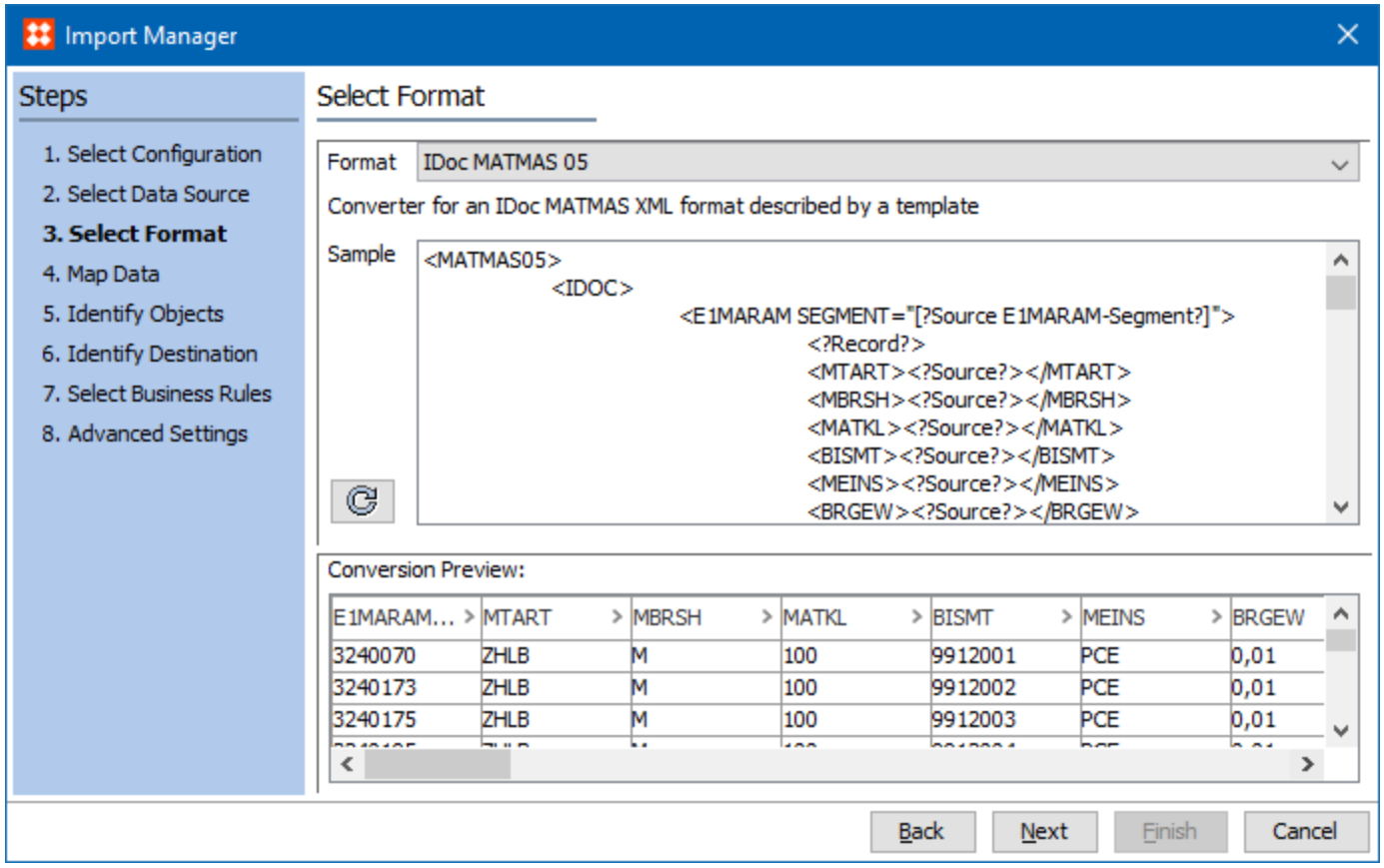
After importing the required SAP attributes into STEP, then use a source sample file from the SAP system to import values for the products. Mapping is required to identify where the values will live in STEP, and may also include data transformations.

Inbound Parameters

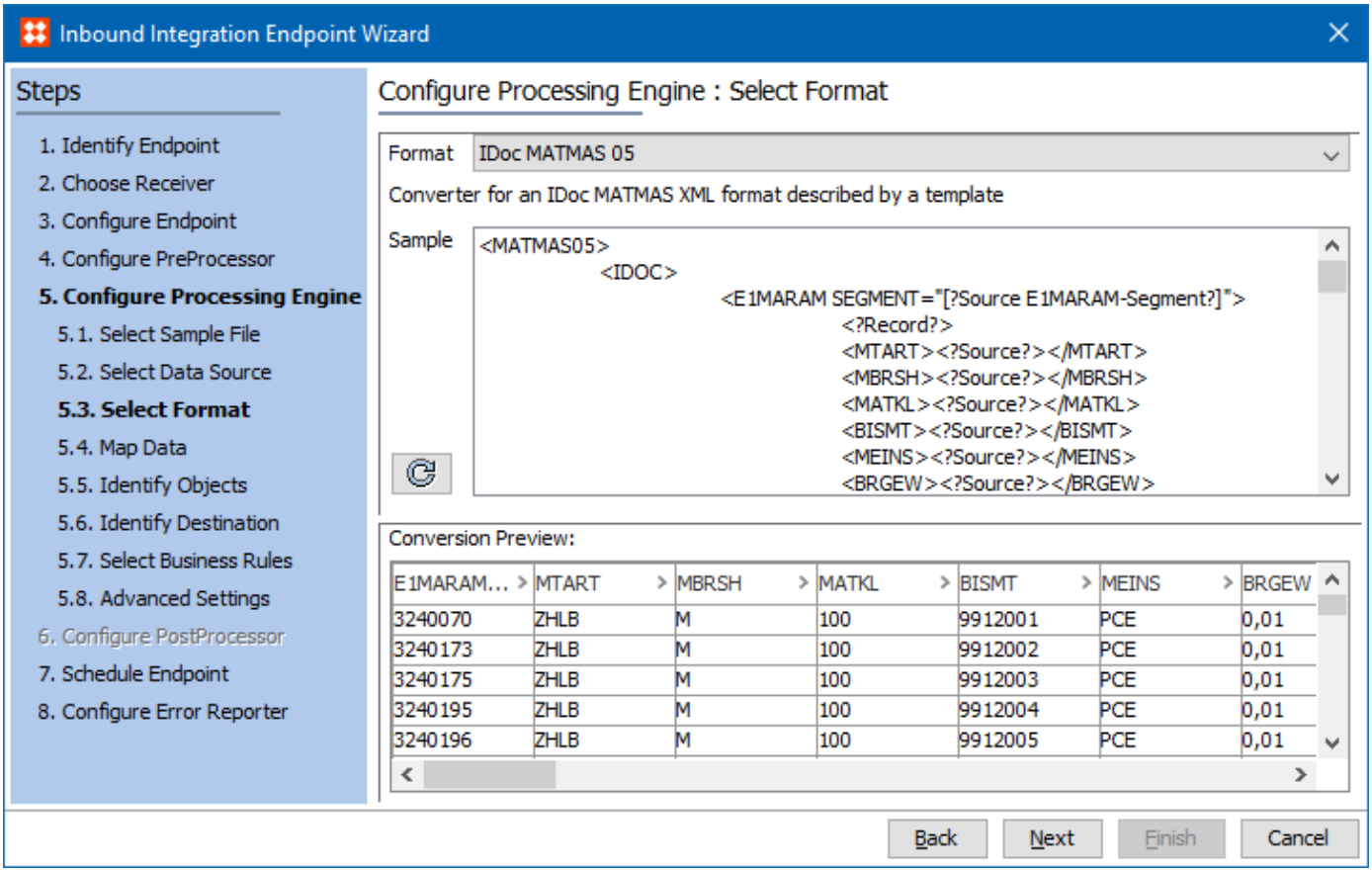
The following parameters are available in both Import Manager and IIEP.

- **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

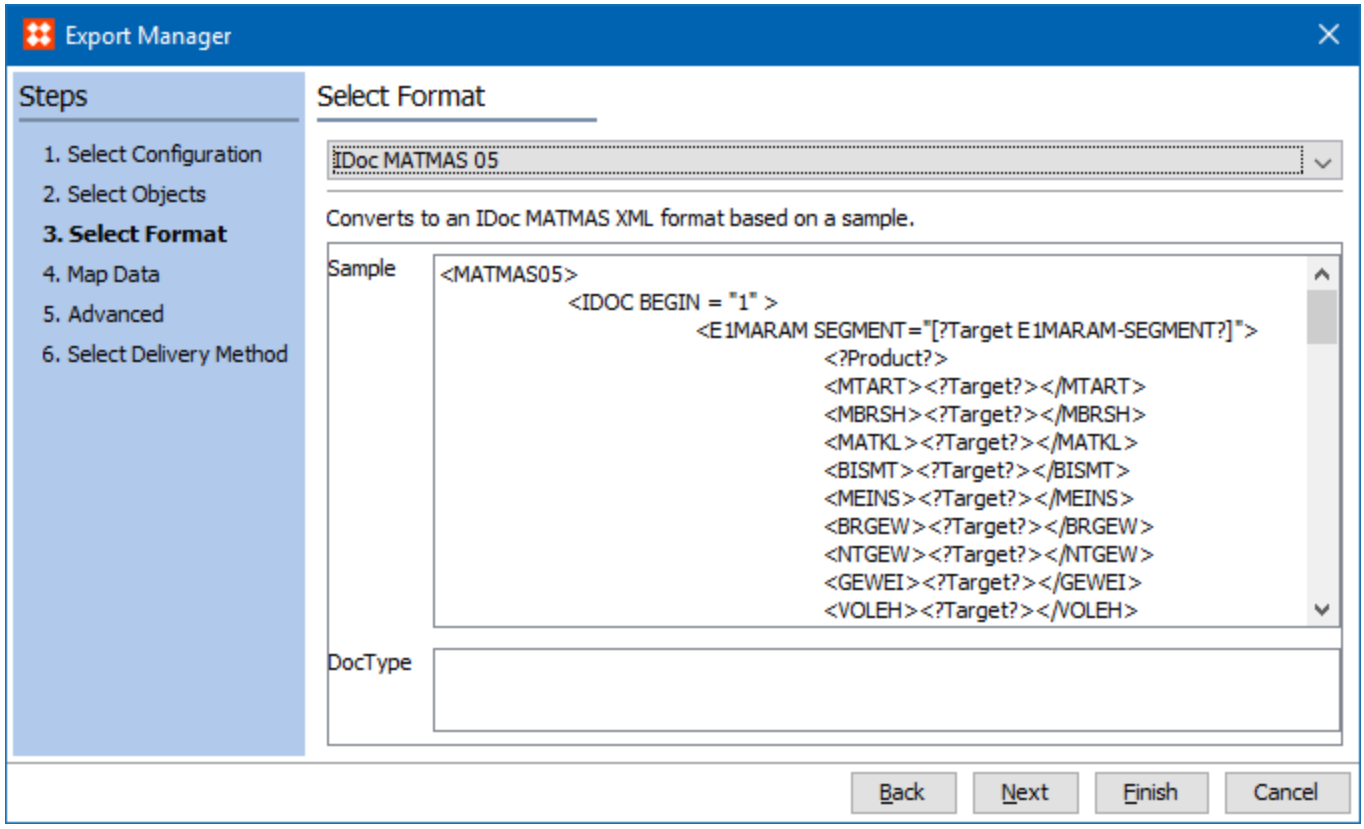
Mapping is required to identify where the values live in STEP, and may also include data transformations.

Outbound Parameters

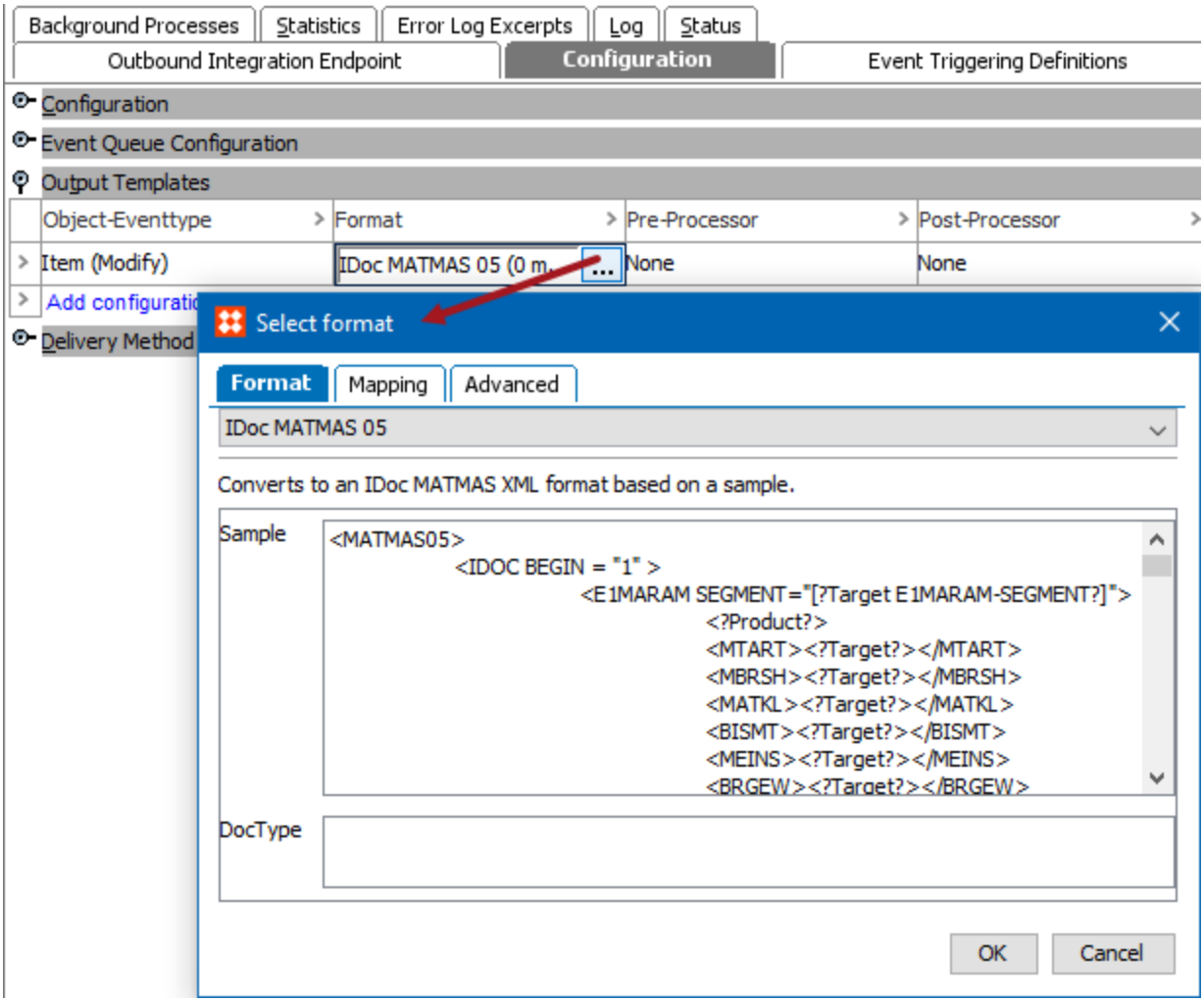
The following parameters are available in both Export Manager and OIEP.

- **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 STEP Publisher ('drag and drop'), Flatplanner, and AutoPage.

Format Availability

Publication Excel is available for selection in:

- IIEP - see **Creating an Inbound Integration Endpoint**
- Import Manager - see **Creating a Data Import**
- Export Manager - see **Creating a Data Export**

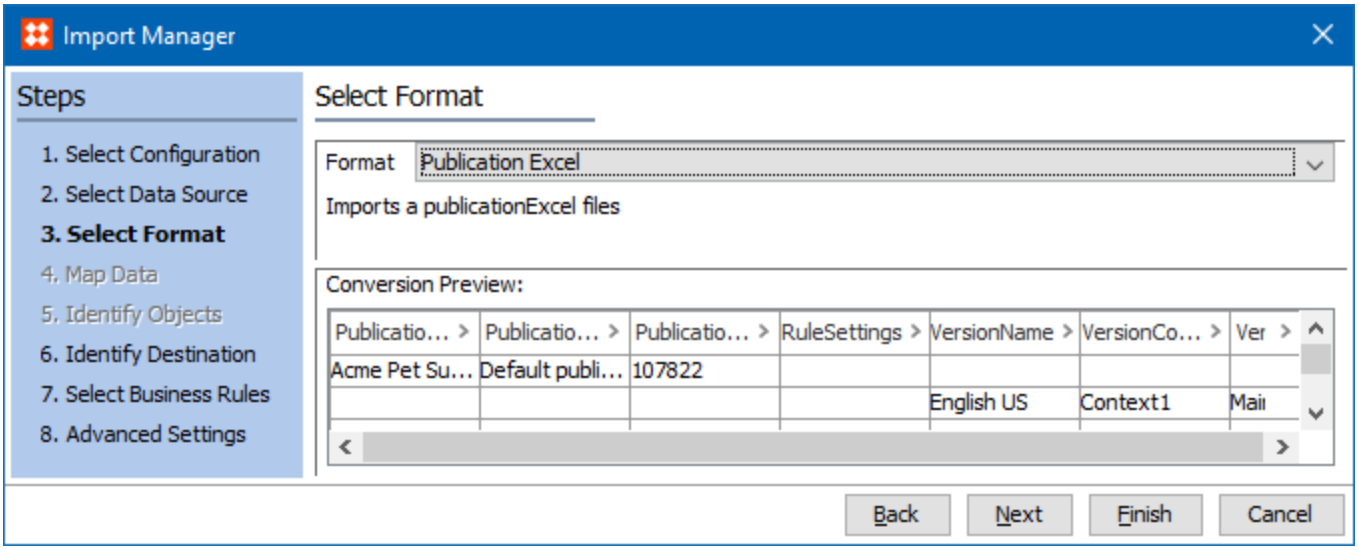
Mapping

Mapping data is not required and the Map Data step is disabled.

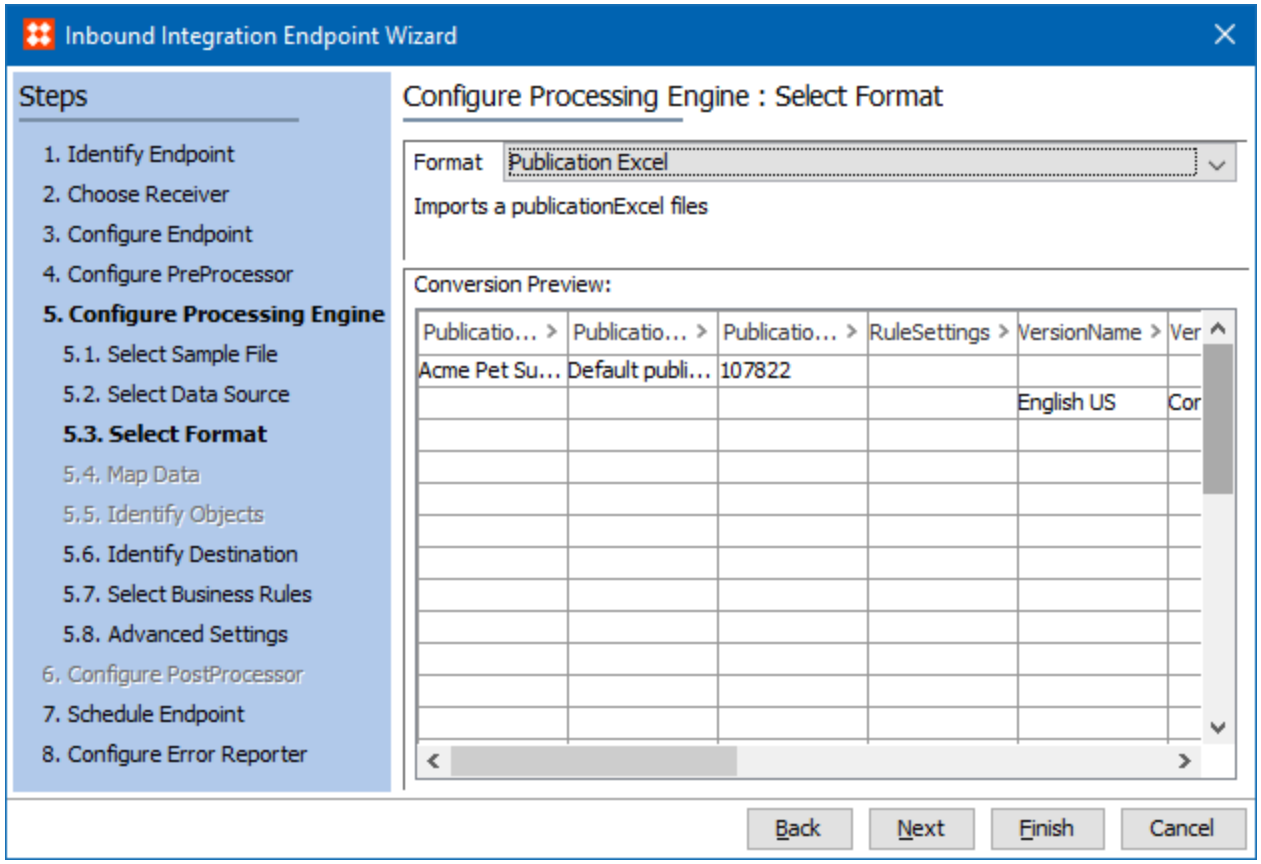
Inbound Data

For more information on the Publication Excel import process, see **Importing Publications in Excel**.

Import Manager



IIEP



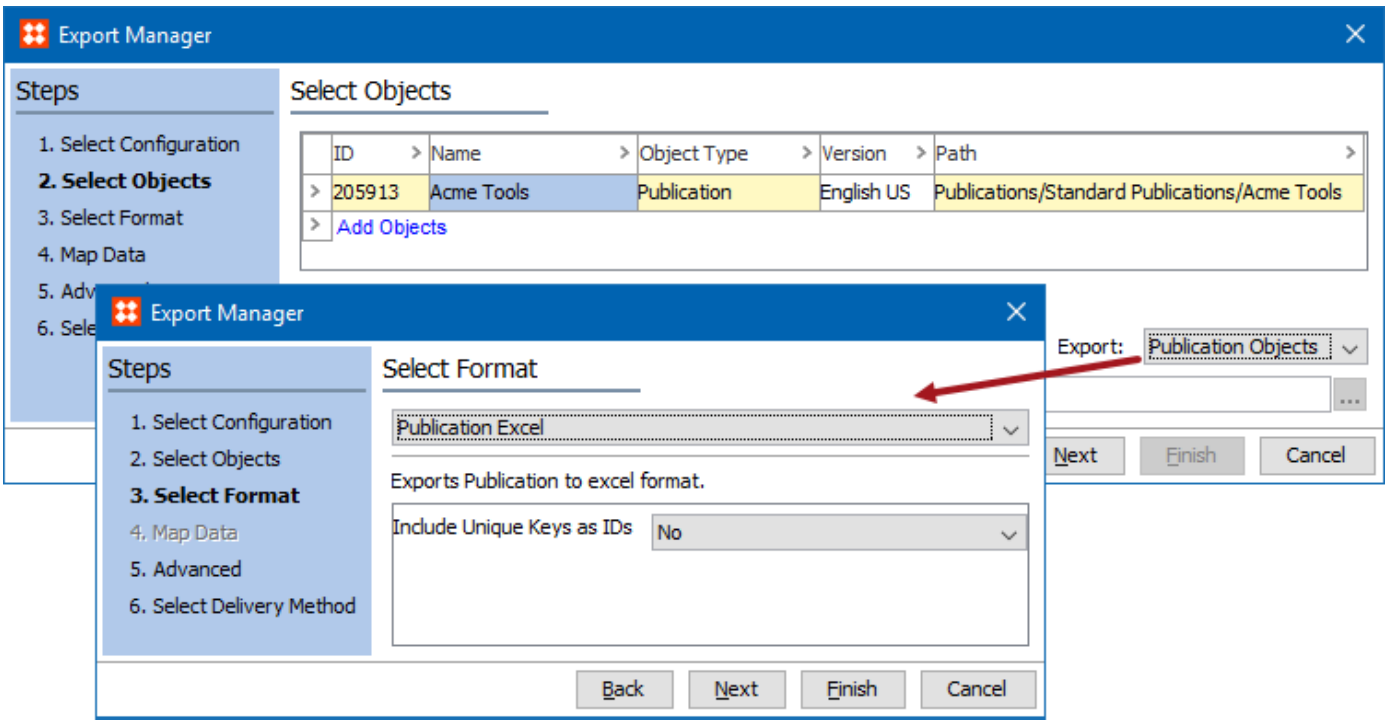
Outbound Data

For more information on the Publication Excel export process, see **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, see **Exporting a Publication Excel Sheet** in the **STEP Publisher** documentation.

Export Manager

After selecting a publication on Tree, right-click and choose **Export Data Below** to see the Publication Excel format option in Export Manager.


Export Manager
Steps
1. Select Configuration
2. Select Objects
3. Select Format
4. Map Data
5. Adv
6. Sele
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 - see **Creating a Data Export**
- OIEP - see **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, see **Data Mapping**.

Outbound Data

Mapping is required to identify where the values live in STEP, and may also include data transformations.

Outbound Parameters

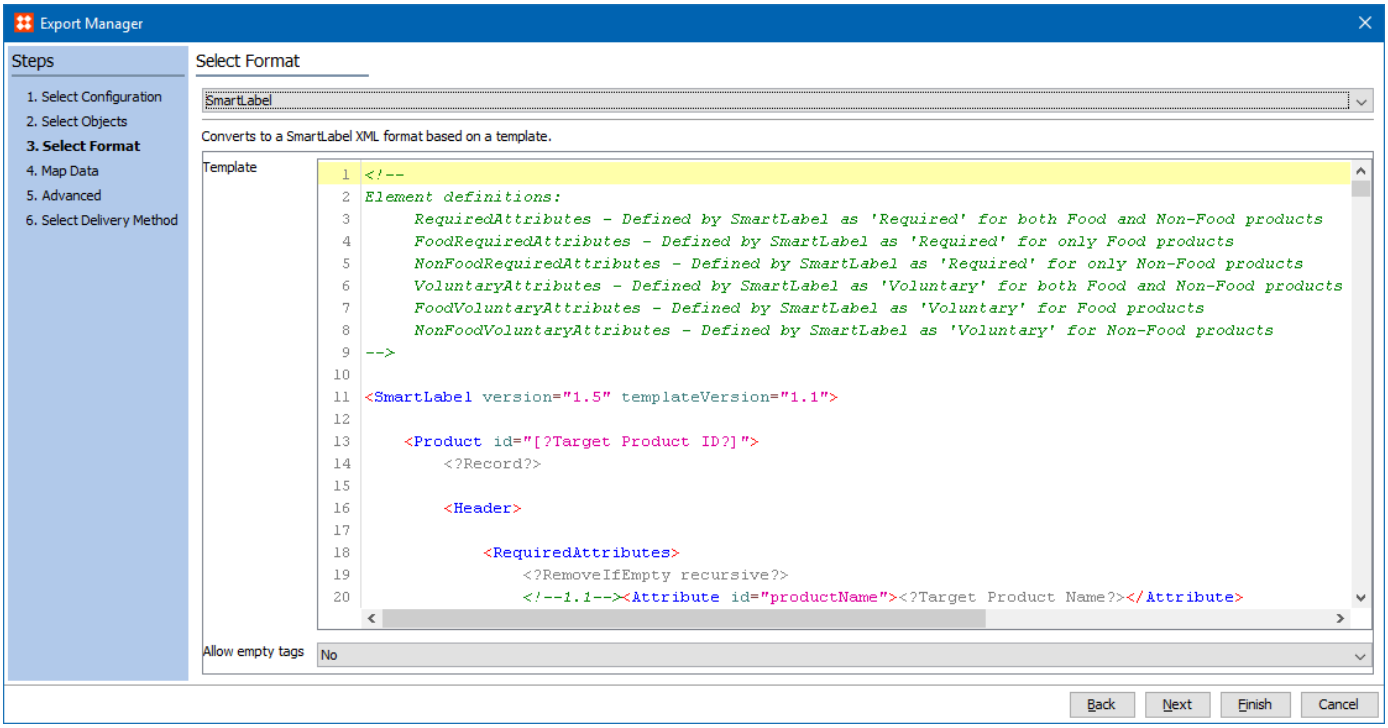
The following parameters are available in both Export Manager and OIEP.

- **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, see the **Configuring the SmartLabel Template** documentation.

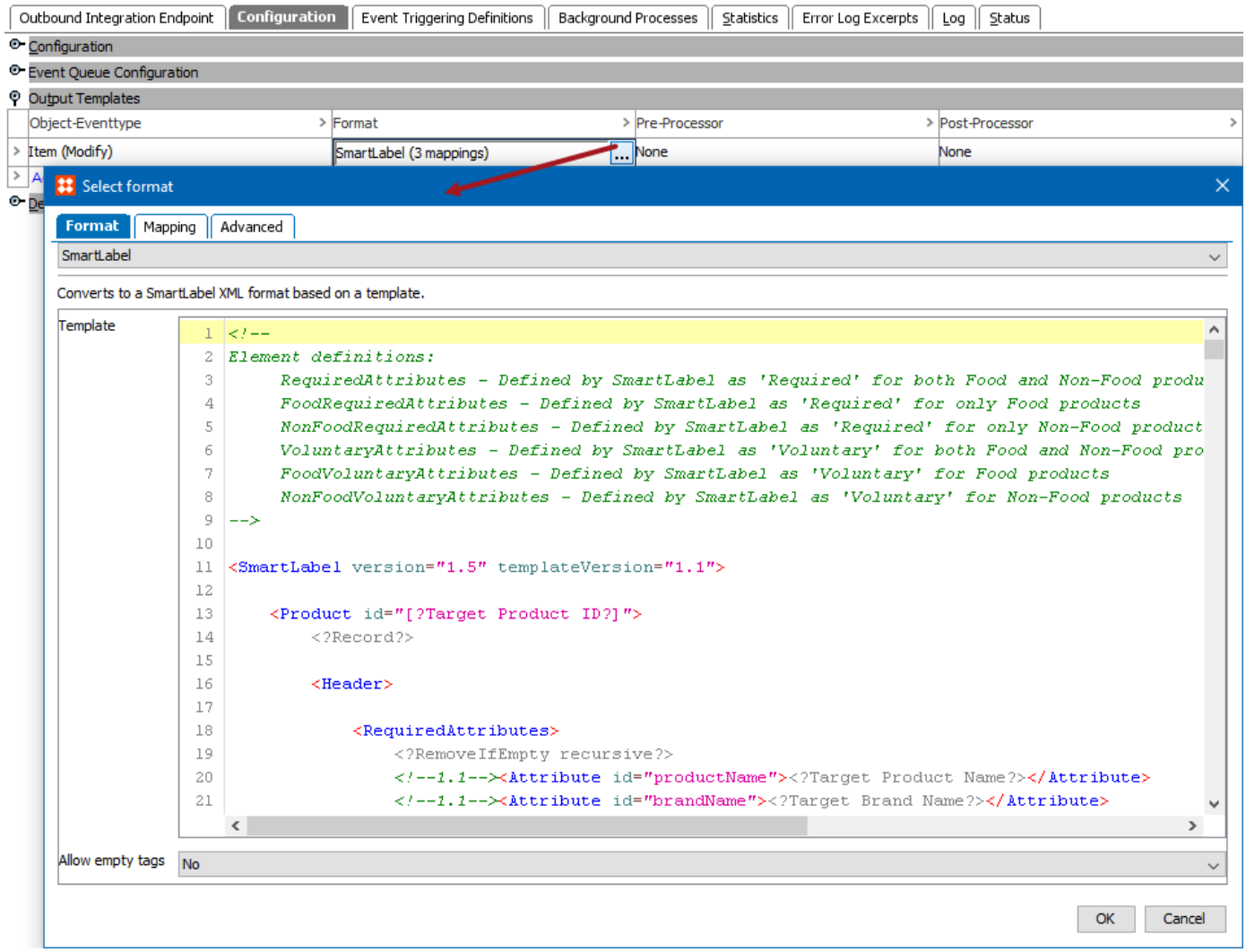
  'RemoveIfEmpty recursive' is applied to all sections to prevent unnecessary tags from being exported. For more information, see **RemoveIfEmpty Processing Instructions in Generic XML** in the **Generic XML Format** documentation.

- **Allow empty tags** - for details about this parameter, see 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, see **Running a Data Export**. For information about an OIEP output template, see **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.

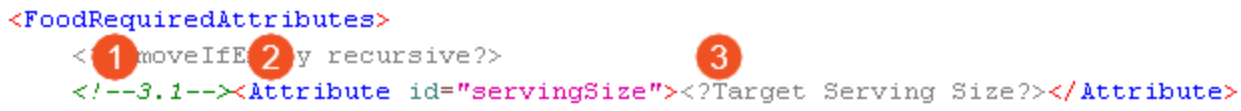
Required Attributes

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 seen 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.



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, see 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.

Format Availability

STEPXML is available for selection in:

- IIEP - see **Creating an Inbound Integration Endpoint**
- Import Manager - see **Creating a Data Import**
- Export Manager - see **Creating a Data Export**
- OIEP - see **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, see the **Unique Keys** section of the **System Setup / Super User Guide** 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

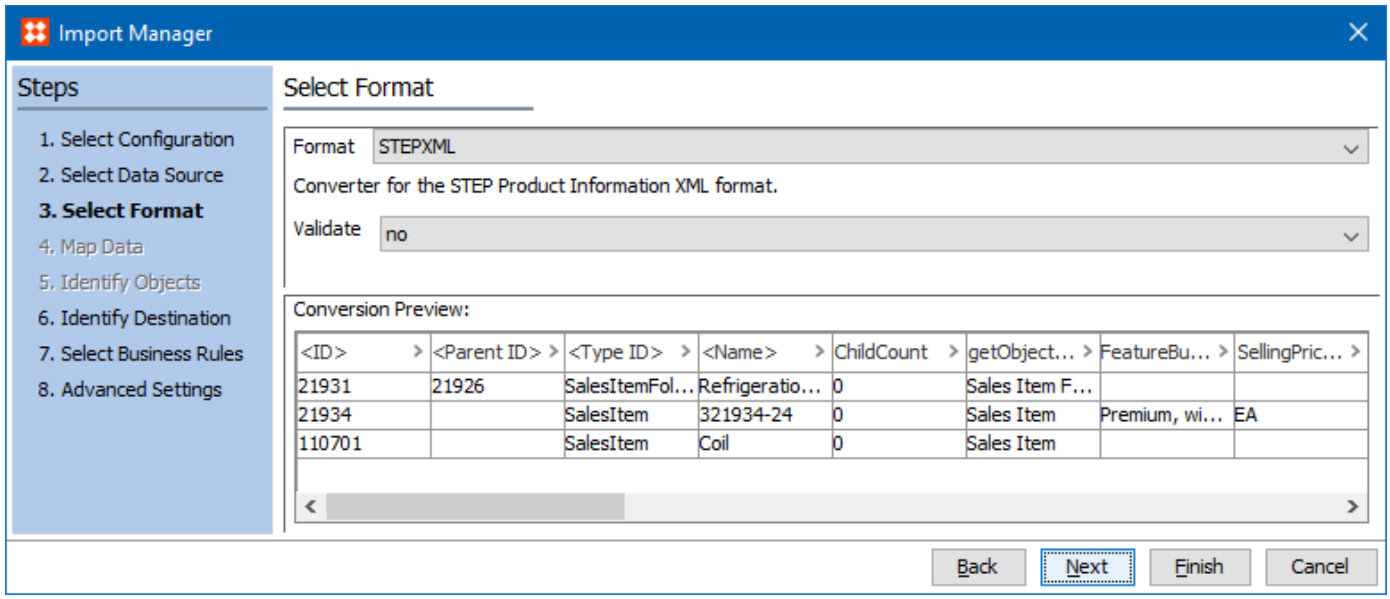
When importing STEPXML files, the values are imported exactly as provided in the file for the specified attributes. This means that when a blank value is imported, an attribute that previously had a value is overwritten as blank. If the value being deleted was inherited, the result is not a blank field, but the inherited value is restored.

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

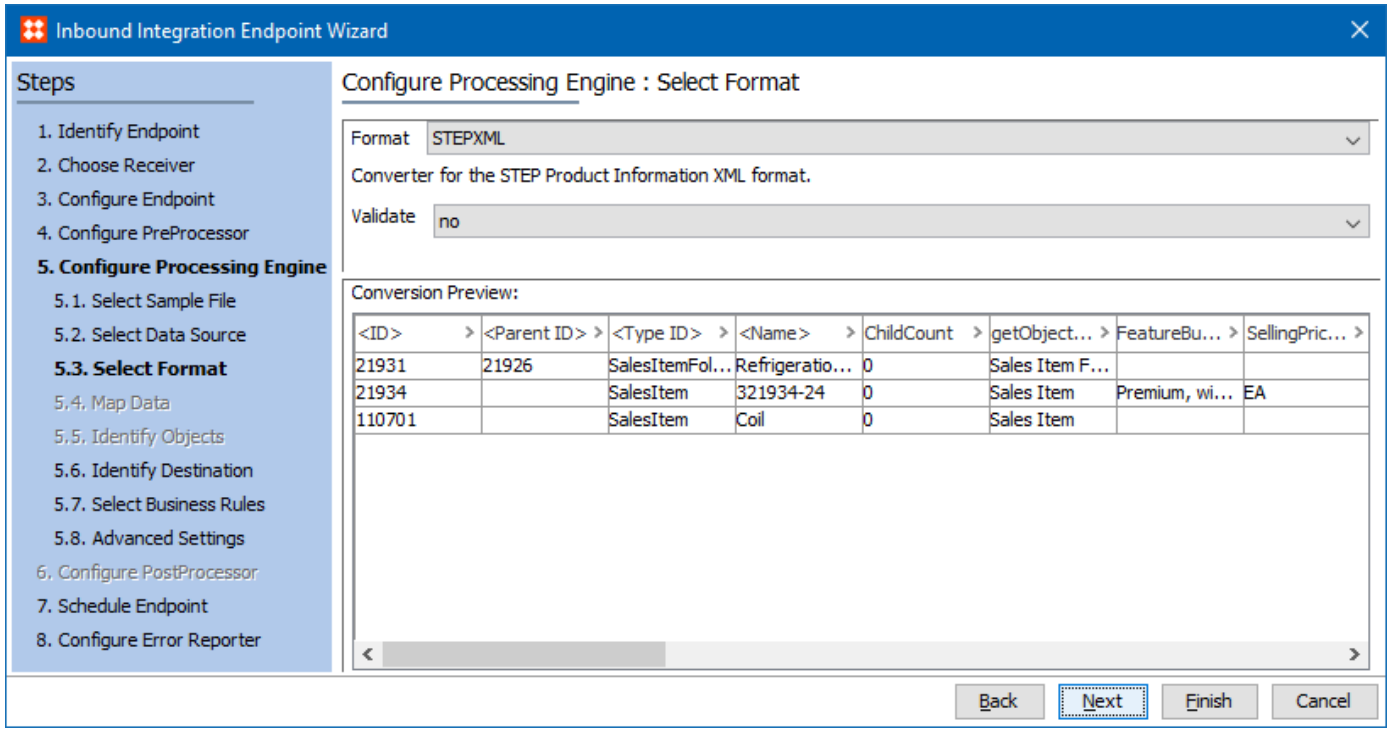
Inbound Parameters

An inbound file can create new objects, update existing objects, and delete existing values and links, among other options. For more information, see the **STEPXML Tags and Examples** documentation.

Import Manager



IIEP



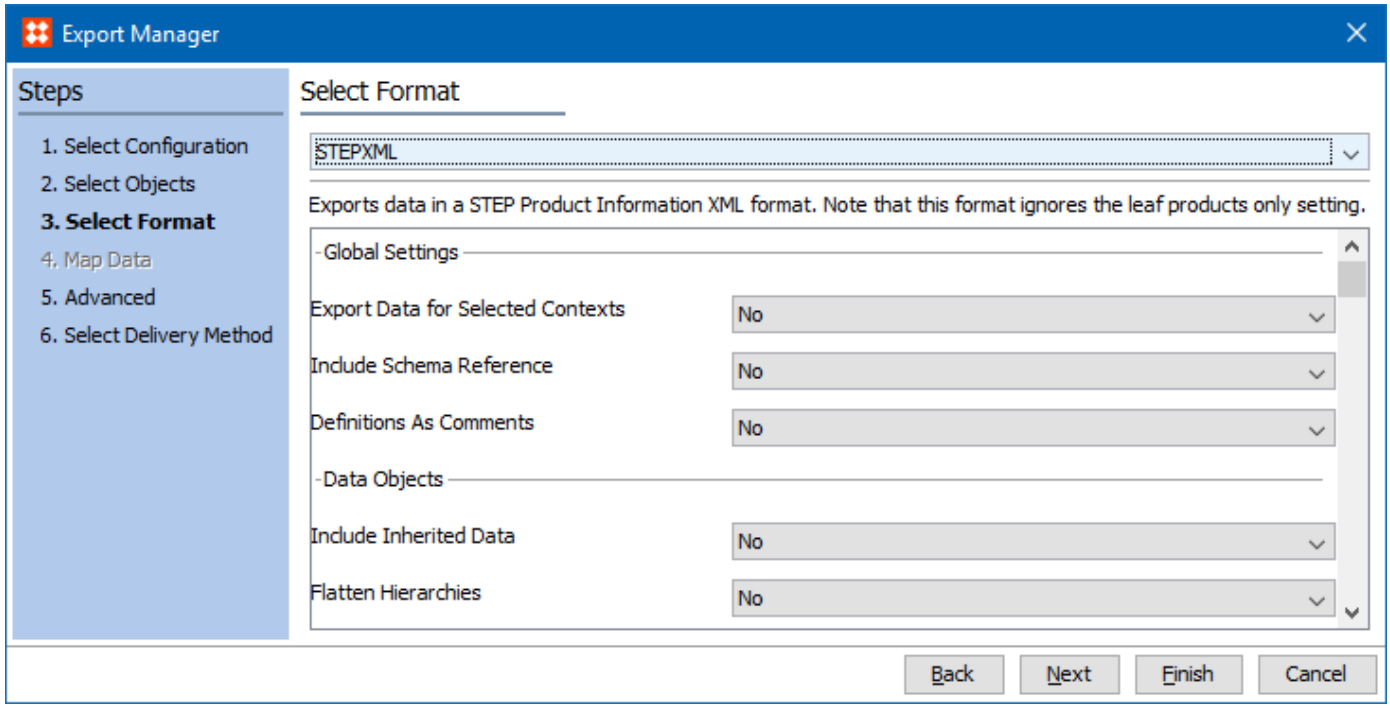
Outbound Data

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.

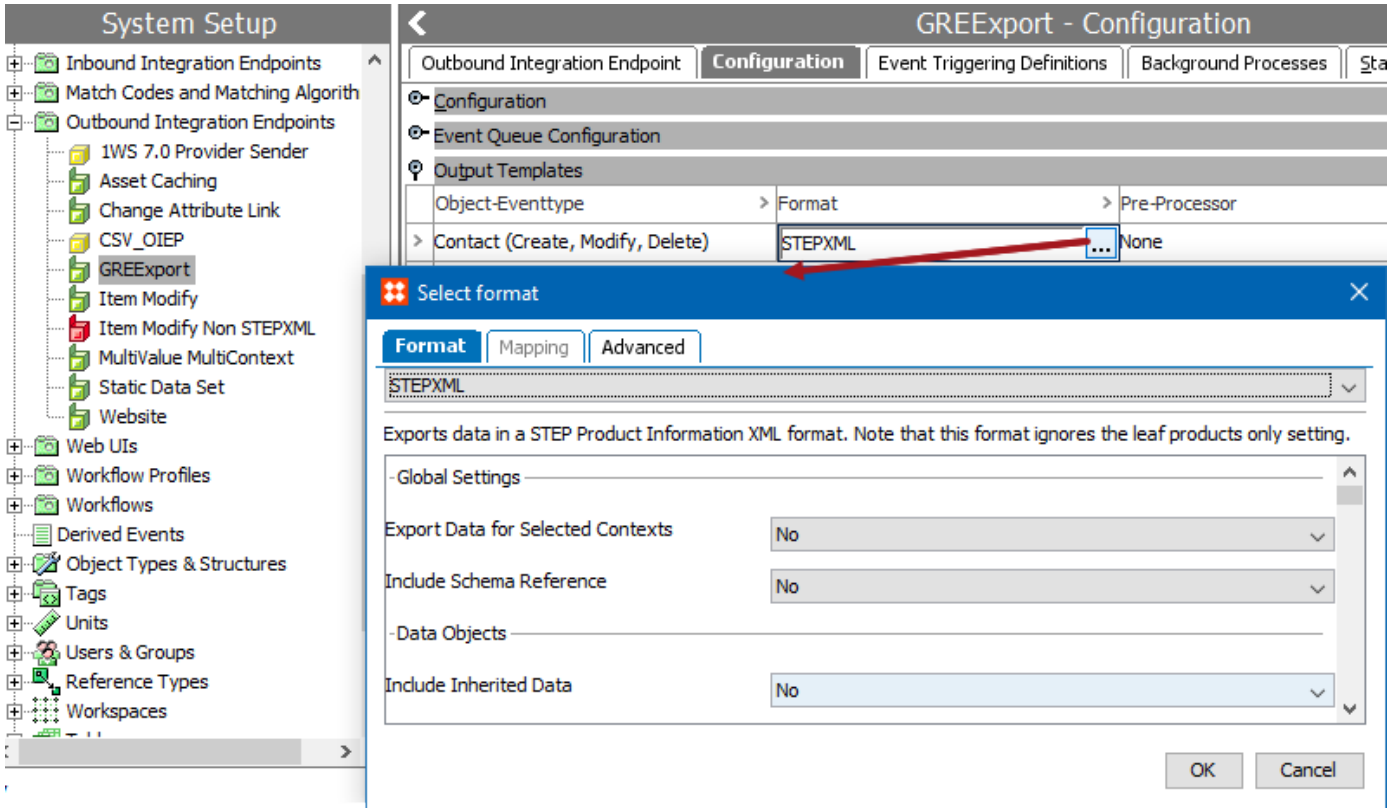
Outbound Parameters

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, see the **STEPXML Outbound Parameters** documentation.

Export Manager



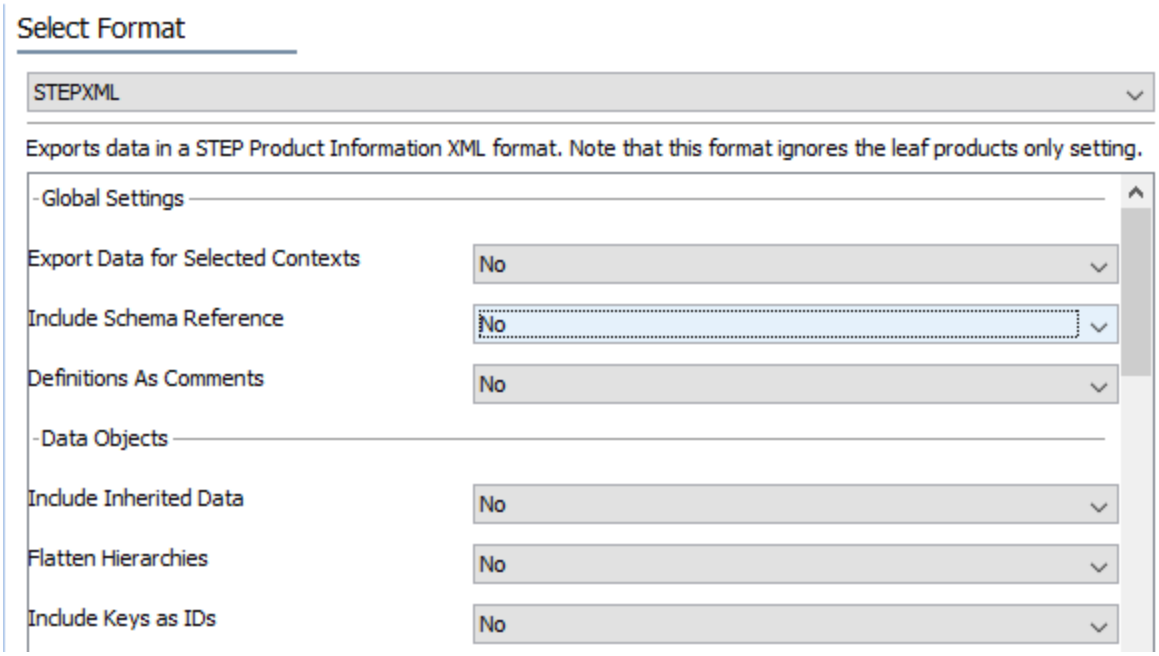
OIEP



STEPXML Outbound Parameters

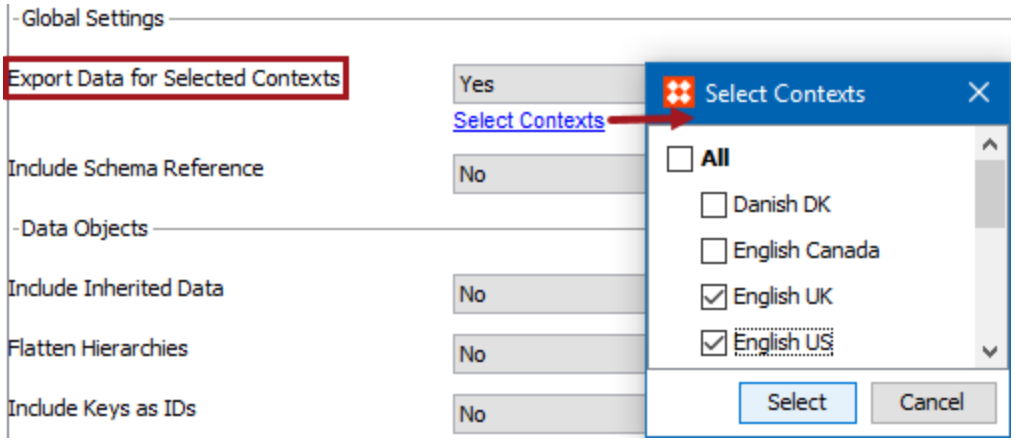
The following parameters are available when exporting using the STEPXML format and are separated into the following categories:

- **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.



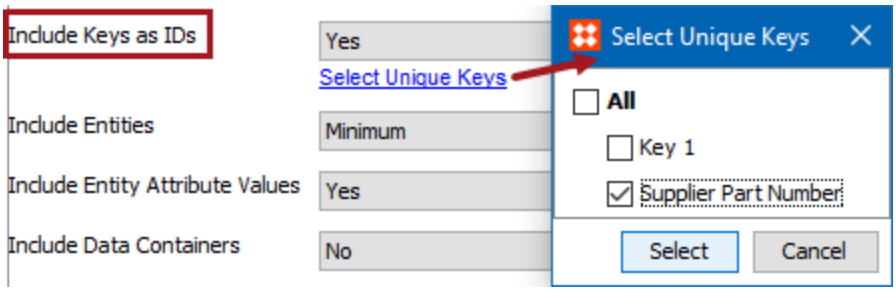
Global Settings

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, see the **Contexts** section of the **System Setup / Super User Guide** documentation or the **Multi-Context Data in STEPXML** topic.
Include Schema Reference	Select **No** or **Yes** to output a reference to the schema file. For more information about schema, click the **STEP API Documentation** button on the STEP Start Page.
Definitions as Comments	Select **No** or **Yes** to output a object definitions as comments. For more information, see the **Configuration Management** documentation.

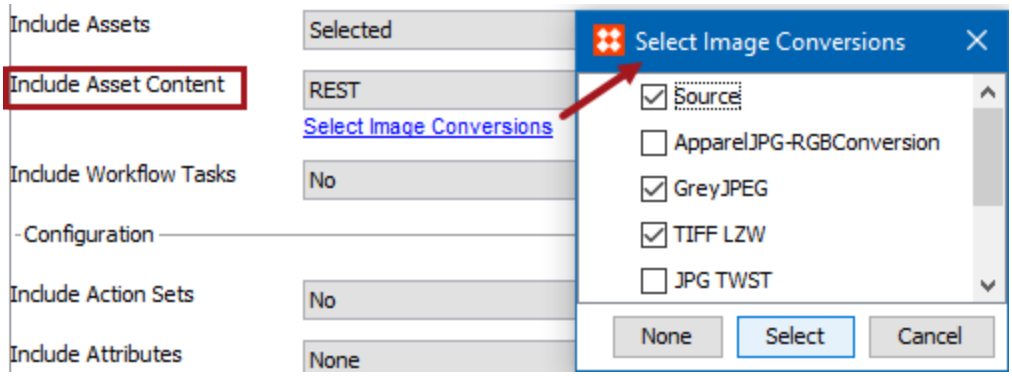
Data Objects

Data Objects Parameter	Description
Include Inherited Data	Select **No** or **Yes** to control the output of inherited values and references. For more information, see **Export Inheritance in STEPXML Example** 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

Data Objects Parameter	Description
	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, see **Object Type Hierarchy** and related topics in the **Getting Started / User Guide**.
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, see the **Unique Keys** section of the **System Setup / Super User Guide** documentation. For an Advanced STEPXML template, this option is included in the IncludeKey Tag. For more information, see 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. • Select **Minimum** or **Selected** to output entities based on the explanation in the **Minimum, Referenced, and Selected in STEPXML** 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, see **Entities** in the **Getting Started / User Guide**.

Data Objects Parameter	Description
Include Entity Attribute Values	Select **No** or **Yes** to control output of entity values. For more information, see **Entities** in the **Getting Started / User Guide**.
Include Data Containers	Select **No** or **Yes** to control output of data containers when exporting entities. For more information, see **Data Containers** in the **System Setup / Super User Guide**.
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 products based on the explanation in the **Minimum, Referenced, and Selected in STEPXML** 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, see **Products** in the **Getting Started / User Guide**.
Include Product Attribute Values	Select **No** or **Yes** to control output of product values. For more information, see **Products** in the **Getting Started / User Guide**.
Include Overridden Products	Select **No** or **Yes** to control output of product overrides. For more information, see **Product Override** in the **Getting Started / User Guide**.
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 classifications based on the explanation in the **Minimum, Referenced, and Selected in STEPXML** documentation. • Select **Referenced** to output all classifications in the hierarchy. • Select **None** to output no classifications or classification references. For more information, see **Classifications** in the **Getting Started / User Guide**.

Data Objects Parameter	Description
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. Note that Tables cannot be imported. For more information, see **Tables**.
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, see **Tables**.
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 **Selected** or **Minimum** to output assets based on the explanation in the **Minimum, Referenced, and Selected in STEPXML** documentation. • Select **None** to output no asset text, references, nor digital contents. For more information, see **Assets** in the **Getting Started / User Guide**. 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, see **Assets Tag in STEPXML** in the **Data Exchange**.
Include Asset Content	Requires that the 'Include Assets' parameter is set to All, Selected, or Minimum. Select **None** to exclude asset contents. 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.

Data Objects Parameter	Description
	 For details about setting up export of asset content, see **Assets and Content with STEPXML** in the **Digital Assets** documentation For more information, see **Assets** in the **Getting Started / User Guide**.
Include Workflow Tasks	Select **No** or **Yes** to control exporting STEP workflow task information for objects with active tasks. For more information, see **Workflows**.

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, see **Action Sets** in the **System Setup / Super User Guide**.
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** documentation. • Select **None** to output no attributes.

Configuration Parameter	Description
	For more information, see **Attributes** in the **System Setup / Super User Guide**.
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** documentation. • Select **None** to output no attribute groups. For more information, see **Attribute Groups** in the **System Setup / Super User Guide**.
Include Attribute Transformations	Select **No** or **Yes** to control output of attribute transformations and their configurations. For more information, see **Attribute Transformations** in the **System Setup / Super User Guide**.
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** documentation. • Select **No** to output no bulk update configurations. For more information, see **Bulk Updates**.
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** 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, see **Business Rules**.

Configuration Parameter	Description
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, see **Collections** in the **Getting Started / User Guide**.
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, see the **Component Models** topic in the **System Setup / Super User Guide**.
Include Contexts	Select **No** or **Yes** to control output of dimensions, dimension points, contexts, applied dimension points, and locale settings. For more information, see **Contexts** in the **System Setup / Super User Guide**.
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, see **Contexts** in the **System Setup / Super User Guide**.
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** documentation. • Select **None** to output no data container types. For more information, see **Data Containers** in the **System Setup / Super User Guide**.
Include Derived Event Types	Select **No** or **Yes** to control output of objects created in System Setup as Derived Event Type. For more information, see **Derived Events** in the **System Setup / Super User Guide**.
Include eCatalogs	Select **No** or **Yes** to control output of eCatalog configurations and price lists. Does not include product selection lists.

Configuration Parameter	Description
	For more information, see **eCatalogs**.
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** documentation. • Select **None** to output no event processors. For more information, see **Event Processors** in the **System Setup / Super User Guide**. The configurations are exported in Base64 and can only be imported without modifications into another STEP system.
Include Event Queues	Select **No** or **Yes** to control output of event queues and applied message templates, output formatting, and triggering definitions.
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** documentation. • Select **None** to output no export configurations. For more information, see **Maintaining a Saved Export Configuration** in **Data Exchange**.
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** documentation. • Select **None** to output no import configurations. For more information, see **Maintaining a Saved Import Configuration** in **Data Exchange**.
Include Integration	Control output of integraton endpoint ID, name, and configurations. The configurations are exported in Base64 and can only be imported without modifications into another

Configuration Parameter	Description
Endpoints	STEP system. • Select **All** to output all integraton endpoints. • Select **Selected** to output only integraton endpoints in the selected hierarchy based on the explanation in the **Minimum, Referenced, and Selected in STEPXML** documentation. • Select **None** to output no integraton endpoints. For more information, see the integration endpoint topics in **Data Exchange**.
Include Key Definitions	Select **No** or **Yes** to control output of definitions of objects created as keys. For more information, see the **Unique Keys** in the **System Setup / Super User Guide**.
Include Link, Reference and Object Types	**Important:** Handle reference types and object types with an ID that start with **stibo.** carefully since they are fundamental objects. System specific object types, edge types, and system specific reference types are exported as follows: • Select **All** to output all link, 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** documentation. • Select **None** to output no link, reference, and object types. For more information, see **Reference and Link Types** in the **System Setup / Super User Guide**.
Include List of Values	LOV definitions and values in LOVs, including applied meta attributes, validation base 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** documentation. • Select **None** to output no LOV definitions or values. For more information, see **List of Values (LOVs)** in the **System Setup / Super User Guide**.

Configuration Parameter	Description
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, see the **Matching, Linking, and Merging** documentation.
Include Matching Algorithms	Control output of matching algorithms, including ID, name, links, and configuration. For more information, see the **Matching, Linking, and Merging** documentation. • 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** 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.
Include Setup Entities	Control output of setup entities including ID, name, links, and configuration. 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** documentation. • Select **None** to output no setup entities.
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** documentation. • Select **None** to output no setup groups. For more information, see **Setup Groups** in the **System Setup / Super User Guide**.
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

Configuration Parameter	Description
	Minimum, Referenced, and Selected in STEPXML documentation. • Select **None** to output no workflow status flags. For information, see the **Status Flags** topic in **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 For more information, see **System Settings** in the **System Setup / Super User Guide**.
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** documentation. • Select **None** to output no table types. For more information, see **Tables**.
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, see **Tags** in the **System Setup / Super User Guide**.

Configuration Parameter	Description
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** documentation. • Select **None** to output no transformation lookup table configurations. For information, see the **Transformation Lookup Tables** topic in **Resource Materials** online help.
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** documentation. • Select **None** to output no unit groups or units. For more information, see **Units** in the **System Setup / Super User Guide**.
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. • Select **All** to output all users or user groups. • Select **Selected** to output users or user groups based on the explanation in the **Minimum, Referenced, and Selected in STEPXML** documentation. • Select **None** to output no users or user groups. For more information, see **Users and Groups** in the **System Setup / Super User Guide**.
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** documentation. • Select **None** to output no Web UI configurations.

Configuration Parameter	Description
	For more information, see **Managing Web UI Configurations** in the **Web User Interfaces / Web UI Setup and User Guide**.
Include Websites	Select **No** or **Yes** to control output of websites, with and without workspace. The legacy 'Web Publisher' used this option to publish data incrementally from STEP. OIEPs provide improved functionality for this same purpose. For more information, see the **Data Exchange**.
Include Workflows	Workflow information, including ID, name, links, valid object types, and configuration are exported as follows. 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** documentation. • Select **None** to output no workflows. For more information, see **Workflows**.

Publishing

Publishing Parameter	Description
Include Publication Groups	Select **No** or **Yes** to control output of publication groups. For more information, see **Publication Hierarchy Elements** in **STEP Publisher (InDesign) / STEP Publisher User Guide**.
Include Publications	Select **No** or **Yes** to control output of publications. For more information, see **Publication Hierarchy Elements** in **STEP Publisher (InDesign) / STEP Publisher User Guide**.
Include Publication Sections	Select **No** or **Yes** to control output of publication sections. For more information, see **Publication Hierarchy Elements** in **STEP Publisher (InDesign) / STEP Publisher User Guide**.

Publishing Parameter	Description
Include Planned Pages	Select **No** or **Yes** to control output of planned pages. For more information, see **Publication Hierarchy Elements** in **STEP Publisher (InDesign) / STEP Publisher User Guide**.
Include Autopage References	Select **No** or **Yes** to control output of AutoPage references. For more information, see **Exporting and Importing AutoPage Publications in STEPXML** in the **AutoPage** documentation.
Include Baskets	Select **No** or **Yes** to control output of details of Flatplanner baskets. For more information, see **Flatplanner Baskets** in **STEP Publisher (InDesign) / STEP Publisher User Guide**.
Include Pagination Rules	Select **No** or **Yes** to control output of details of pagination rules. For more information, see **Exporting and Importing AutoPage Publications in STEPXML** in the **AutoPage** documentation.
Include Actual Page Data	Select **No** or **Yes** to control output of details of actual page data. For more information, see **Saving Mounted Planned Pages to STEP** in **STEP Publisher (InDesign) / STEP Publisher User Guide**.

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. Although this is not an all-inclusive list, the most commonly required data can be handled using the following tags, XML attributes, and setup:

- AssetContent Tag in STEPXML
- Assets Tag in STEPXML
- AttributeLink Tag in STEPXML
- Auto Approve in STEPXML
- Backward Compatibility in STEPXML
- Classifications Tag 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 Products in STEPXML
- Filter References in STEPXML
- Filter Values in STEPXML
- IncludeKey Tag in STEPXML
- Inherit References in STEPXML
- Inherit Values in STEPXML
- Minimum, Referenced, and Selected in STEPXML
- Multi-Context Data in STEPXML
- Products Tag in STEPXML
- Referenced and Embedded XML Attributes in STEPXML
- ReplacementRules Tag in STEPXML
- SequenceProduct Tag in STEPXML
- SingleUpdateMode in STEPXML
- STEP-ProductInformation Tag in STEPXML

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. See 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 Input' 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, see **Assets and Content with STEPXML** in the **Digital Assets** documentation.

Asset Content Templates

When exporting asset content via Advanced STEPXML, an input 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, see **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 STEP 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>iVBORw0KGgoAAAANSUhEUgAAAScAAAE6CAIAAADbcANTAAAAAXNSR0IArs4c6QAAAARnQU1BAACxjwv8YQUAAAAJcEhZcwAADsMAAA7DAcdvqGQAAAMkmSU
  etqnqetVadc8O3/q2d+P9/TSiHX2yU3S02yu4KuP1/81BWLXXEqjrKx/NH2W7eFMraTixJdKpuEVF2V8BtKAtl1VJHrKqjfDx/lO06VbcU0am6RUTZ3TLD2tKIGsrH80fZrlN
  NH2W7TtUtRXSqbhERjoJy2fVSQTlqDbGqjvK1+aNs16m6pYh2qy57U4dt+1//63+Vjf7t3/7v///2///qv/5prahS0Sfn//J//46q9F1NIzf/8n/+zIEADjXYFlKtOliiaBzV
  FmgWwFLA/MaKeSQ3Z84cBYalUnsGuyovZe4arhrRSjYvJsQwlS1n1NyyE0sZHZZhUlQa/Pa3v02Brux3ZDZ79uzbbrtt1qxZUwpMnz596NChZ5555jvvvKP9P//zP4sYGocfe
  PHHP/7x3/3d36277rr9+/cfOXLk5ptv3rt377/+67+eNm3am2++iRMffPPB3g305l1o7nYpINMpb+bFNPjwww+PPPLIvn37/uQnP+natevZZ59dORp8TWEpoFrqZiOV2whGA+
  ...
  ...
  ...
  jyFJstskmpKdsvIbMKjw7z0LCHhhsLsy5hvAffMadBkbm4uIhXvb29lQc0yGDiIXaR39jt6+vDMdZ41JmdnZ35fH5vb+/h4YHhn2vEYJUoBOc2IsKsSx5PT0/yaLtcLheLRb
  CcM/WWODJq4zIc3l5ubu7u7S0lMlkenp6/gq+51ooFNbX1yuVCnLKk+PD4ylQGWsWRcjZl3yqM9atHFJVHx8fDw7Ozs8PNzZ2Tk6Orq7u5NjgMPq3zZGPLMuRsy6BCO2YBQ
  iYeGQ6PDPDRs/CRmXcJQOTw0HBod5qFh4ycx6xKGyuGh4dDoMA8NG+JWZcmukzgAAAAABJRU5ErkJggg==</BinaryContent>
</AssetBinaryContent>
<AssetBinaryContent ImageConversionConfigurationID="185016" MIMEType="image/jpeg" Checksum="6510942e4e5f4a82c4d72a3ce896fab45cde8caf">
  <BinaryContent>/9j/4AAQSkZJRgABAQEAXwBfAAD/4gGoSUNDX1BST0ZJTEUAAQEAAAGYQURCRQIQAABtbnRyR1JBWVhZWiAHzwAGAAMAAAAAAABhY3NwQVBQTAAAAABub
  JkZXNjAAAA9AAAAGl3dHB0AAABYAAAABRia3B0AAABdAAAABRrVFJDAAABiAAAAA50ZXh0AAAAAENvcHlyaWdodCAxOTk5IEFkb2JlIFN5c3RlbXMgSW5jb3Jwb3JhdGVkAA
  AAAAAAAAAAAAAAAAAAAAAWFlaIAAAAAAAAPNUAAEAAAABFs9YWVogAAAAAAAAAAAAAAAAAAAAAGN1cnYAAAAAAAAAAQIzAAD/2wBDABsSFBcUERsXFhceHBsgKEIrKCUlKFE
  BgMHAwQCAwAAAAABAgMEEQUSISIxQQYTFFFhcTKBwSNCUpGhsfBicuEzgtHxJKJDU7L/2gAIAQEAAD8A0wIAAJAIABIAIBJBIAAAAAAABB5XNxTtaMqtWWIr9TP19euZzbp7I
  NUD4nGNSDhJZjJYa8ygvezjWZWdT/ZP6M8bPs5XqPN1NUo/hi8tl7Z6fbWa+wpJPrJ72/mdYAAAAAAAABBmO0dr3Vz38fhqb/mVEuj/ABELy8yw0K8drfKMn9nV4ZfRmvJIM
  obns3UU821WMovpPc0e1l2cpwe1dz7xr7kd0S7pU4UoKFOKjFcklg+wAAAAAAAAAACAZLXbXw185QXDPiRWzWG2uTIjvfJtvkl5m10q28JYUqTXHjMvdnYCASQSAAAAAAAAA
  Z</BinaryContent>
</AssetBinaryContent>
```

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>
```

Name

The <Name/> tag returns the object name as output.

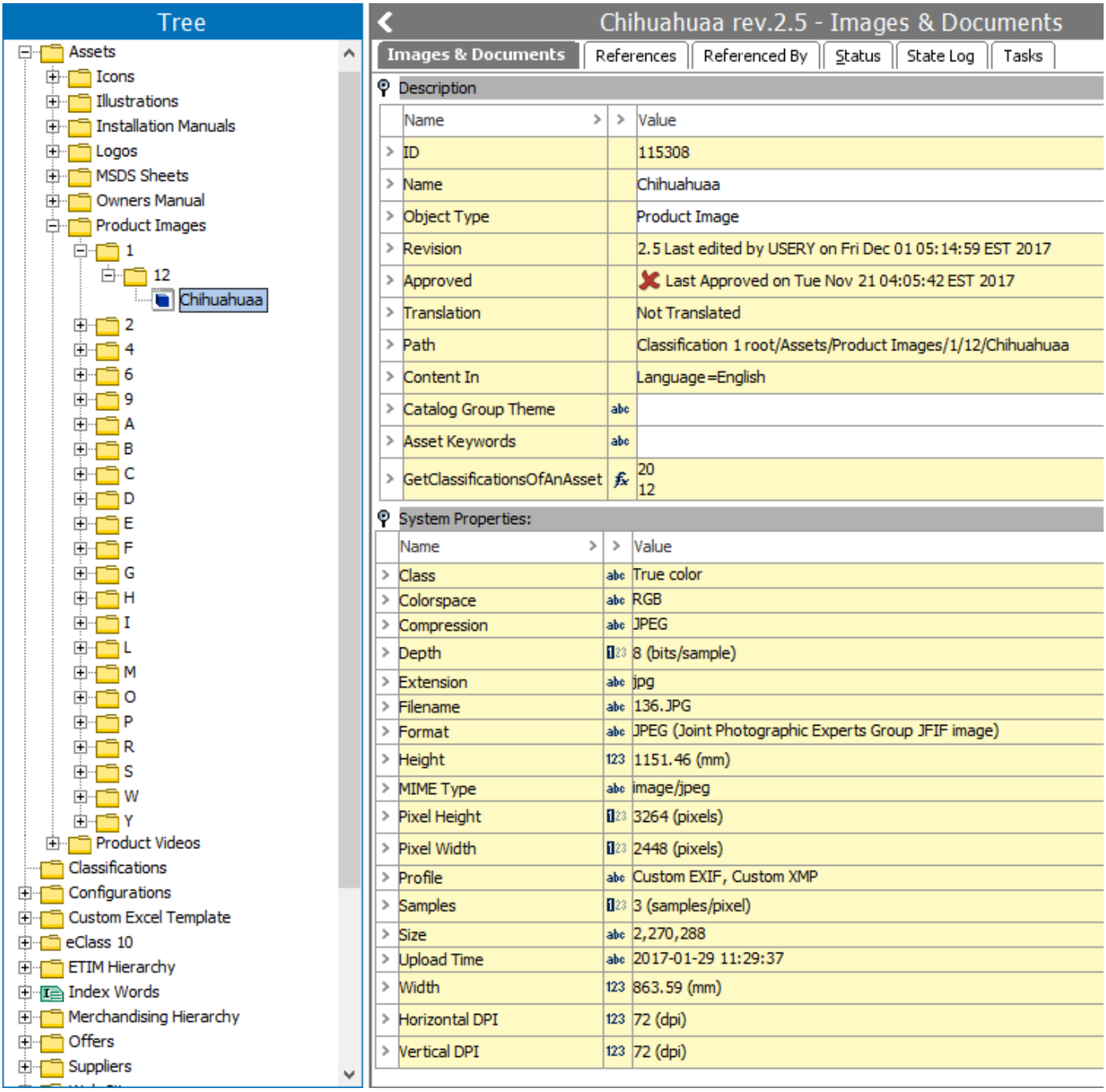
Values

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.

AttributeLink Tag in STEPXML

The **AttributeLink** tag outputs local attribute links for the products, product-overrides, or classifications in the export.

```
<AttributeLink/>
```

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>

Results

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, see **STEP-ProductInformation Tag in STEPXML**.

Backward Compatibility in STEPXML

The STEPXML schema is extended continuously but should always be backwards 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).

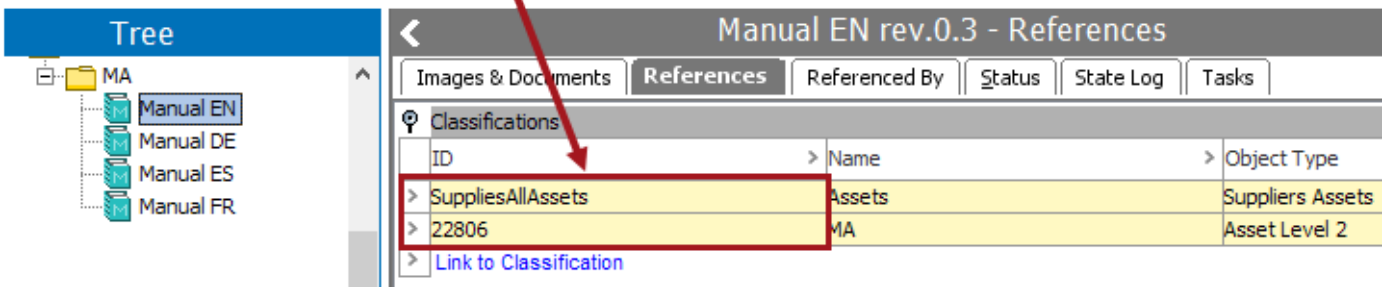
```
<Attribute ID= "Depth" Multivalued="false' DefaultUnitID="unece.unit.CMT" ProductMode="Normal" FullTEXTIndext="false"
  ExternallyMaintained="false" Derived="false" Selected="true" Referenced="true"></Attribute>
  </Attribute>
```

- **Derived** – attribute element that specifies if the attribute is calculated.

```
<Value AttributeID="CountryOfOrigin" Derived="true">GB</Value>
<Value AttributeID="ManufacturerEAN" Derived="true">9513864492123</Value>
```

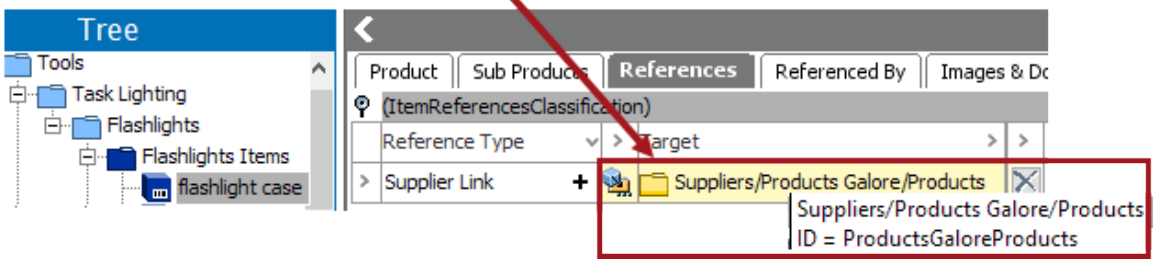
- **ClassificationReference** – used inside both the Product and Asset elements. For assets, this is the parent link (assets can have multiple parents).

```
<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"
```



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>
```

When an XML attribute is absent, the default setting is used. For example:

```
<Classifications>
    <Classification IncludeParent="true"/>
</Classifications>
```

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, see 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, see 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, see 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>
```

Create Objects in STEPXML

New objects can be created in the Tree and in System Setup using the STEPXML import.

Note: To create an STEPXML import template, first export relevant data, and use that as base for the import file.

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, see 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, see **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, see the **Creating Objects with Active Keys** topic of the **Keys** documentation within the **System Setup / Super User Guide**.

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>
```

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.

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 three 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.
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.

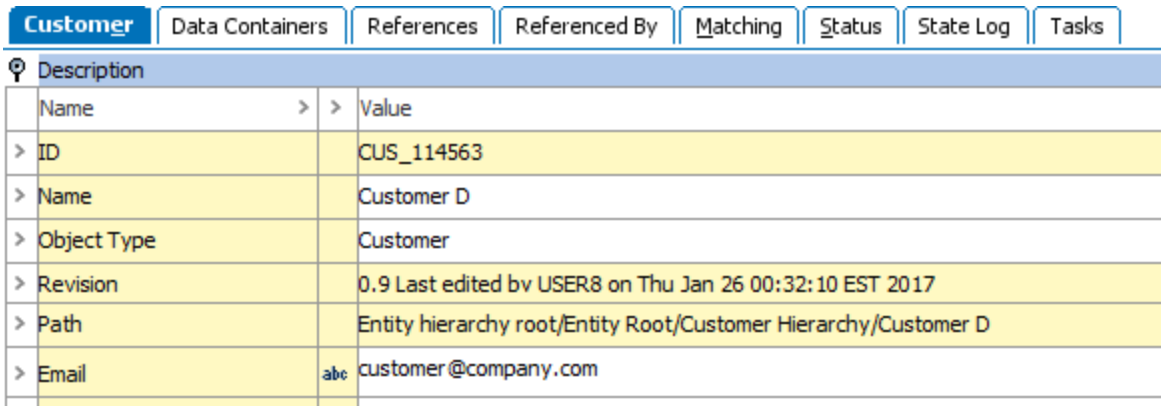
If a data container type allows for multiple containers, then it is indicated by the **MultiDataContainer** tag. Individual data containers are identifiable by the **DataContainer** tag. The Value tag, with the AttributeID XML attribute, is used to show / set up the attributes and attribute values.

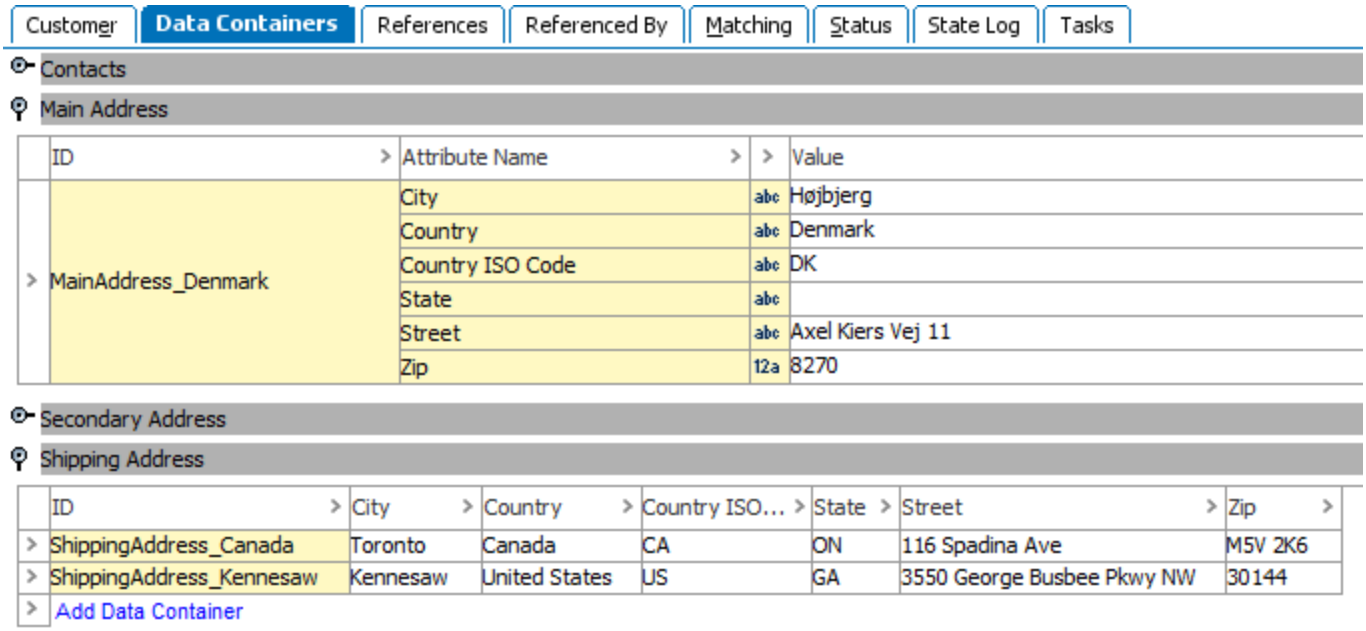
Example STEPXML code is shown below:

```
<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 STEP Workbench:





Delete Objects in STEPXML

Products, entities, classifications, assets, attributes, LOVs, and units 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 possibly entities), delete moves the object to the Recycle Bin.
- For attributes, LOVs, units, and global revisable entities, delete purges the object.

For more information on revisability, see **Revisions** or **Revisability on Entity Object Type**.

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>
```

Note: The <DeleteAttributeOrGroup> tag can only be used to report the deletion of attributes and attribute groups to downstream systems during export. The tag does not allow mass delete of attributes nor attribute groups from STEP via STEPXML during import. Deleting attributes and attribute groups is a manual task that must be performed in System Setup.

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, see 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.

```
<STEP-ProductInformation ExportTime="2015-12-31 15:24:43"
  <DeleteProducts>
    <DeleteProduct ID="124311"/>
  </DeleteProducts>
</STEP-ProductInformation>
```

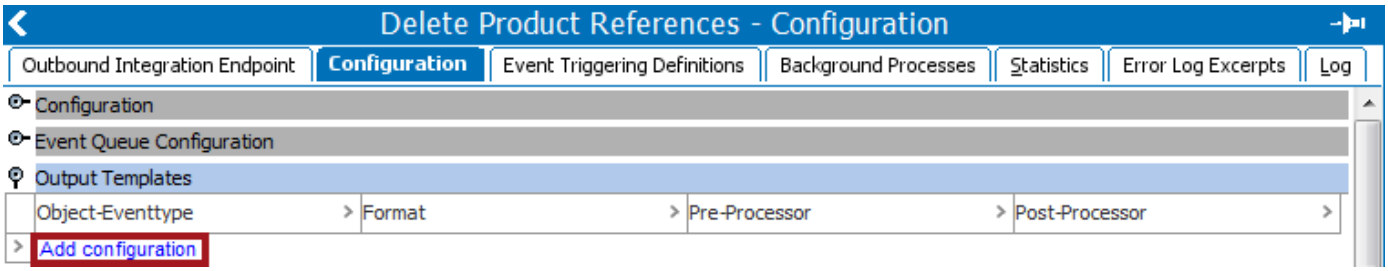
For more information on event-based OIEPs, see the **Outbound Integration Endpoints** documentation.

Delete Product Reference in STEPXML

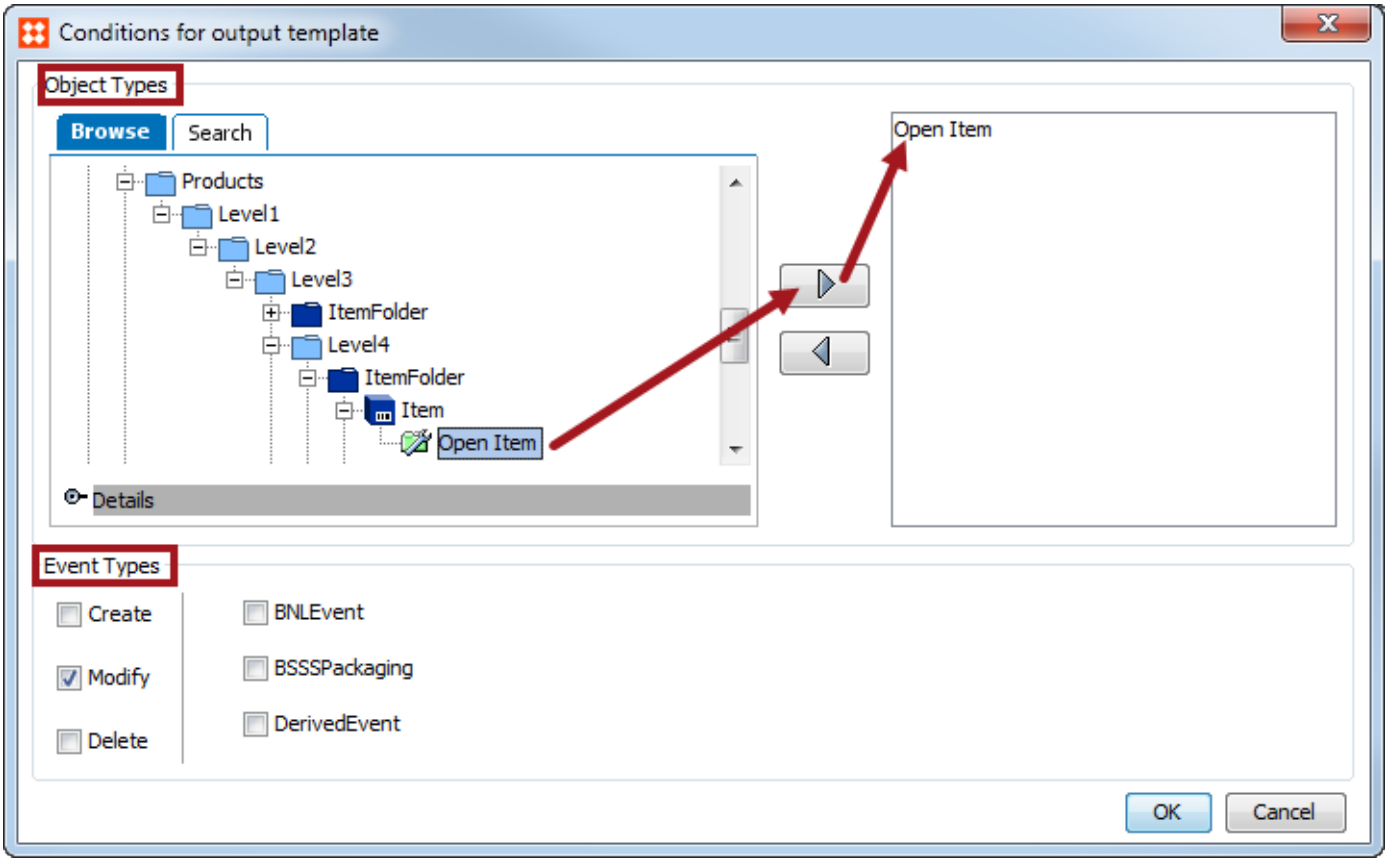
When an Outbound Integration Endpoint (OIEP) is configured to listen for Modify events, use the **STEP-ProductInformation** tag with the **ExportDeletedData** STEPXML attribute to output information about deleted product references. After deleting the reference, approving the product triggers the OIEP.

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.

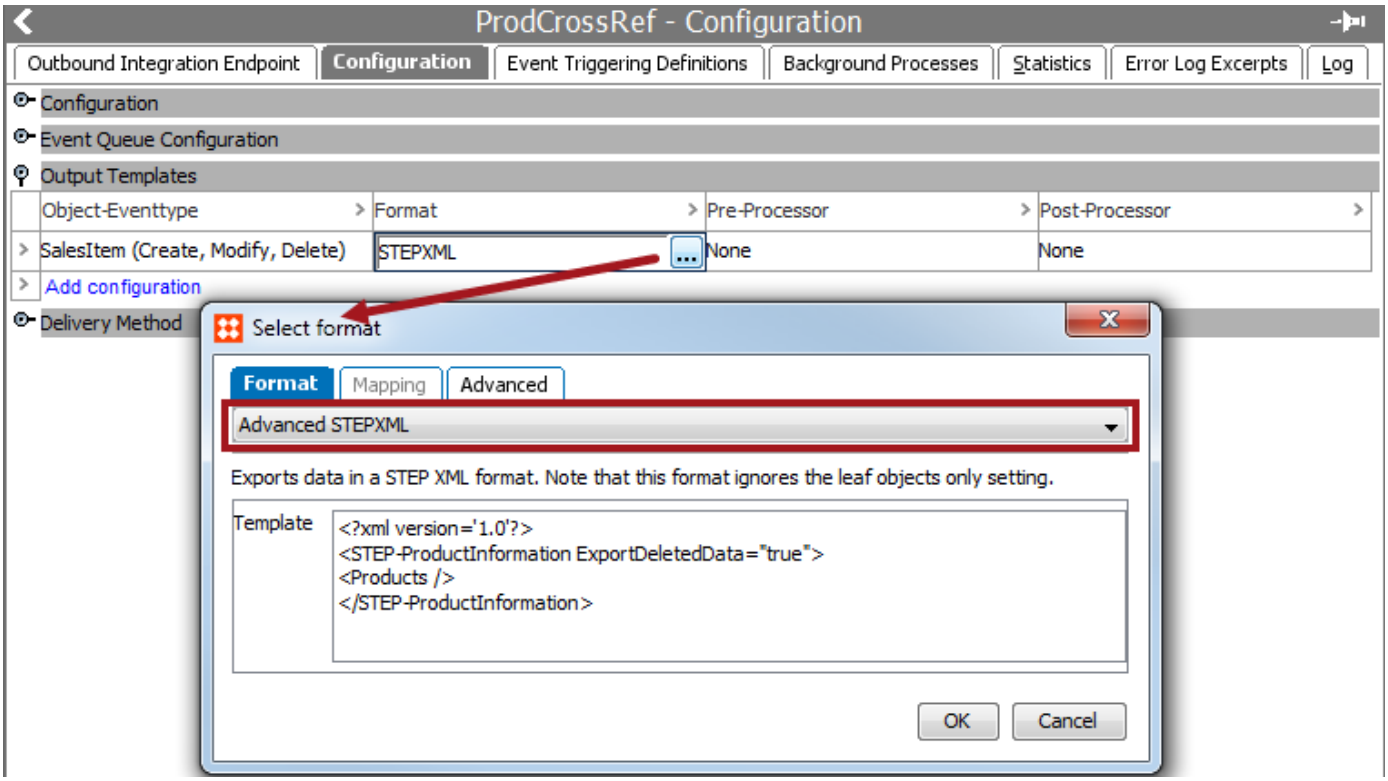


Template

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.

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.'

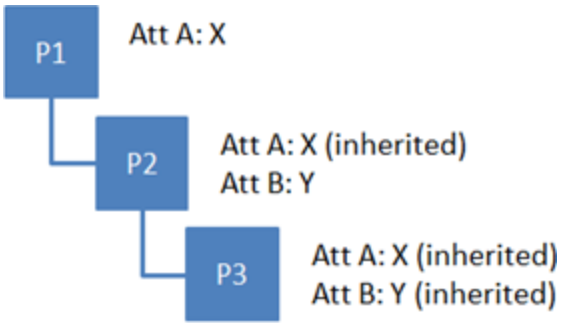
```
<?xml version="1.0" encoding="utf-8"?>
<STEP-ProductInformation ContextID="Context1" WorkspaceID="Main">
  <Products>
    <Product ID="L6576">
      <Name>Panasonic Viera TC-P65VT50</Name>
      <Values>
        <Value AttributeID="Manufacturer's Description"/>
        <Value AttributeID="Horizontal 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.

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, see 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.

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 is available on entities and products.

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, see 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"/>
```

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.

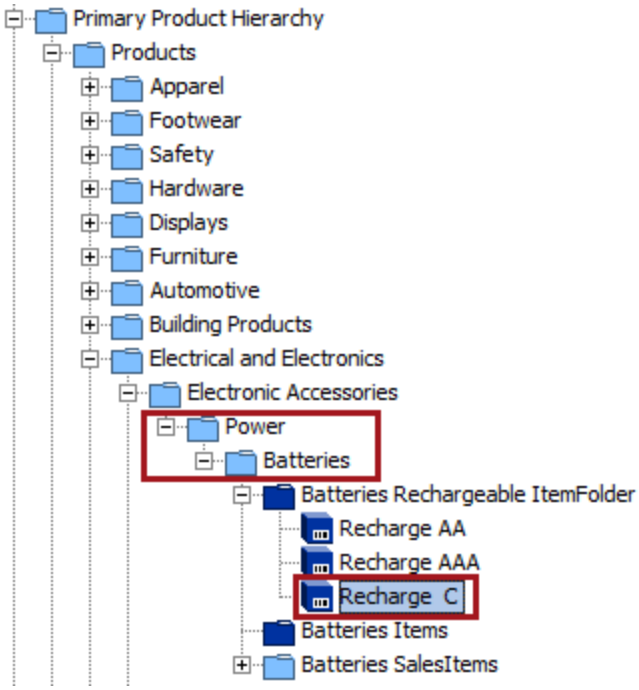
Use multiple instances of the **FilterUserType** tag to include all of the required objects and to filter out the objects that are not needed.

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.

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:

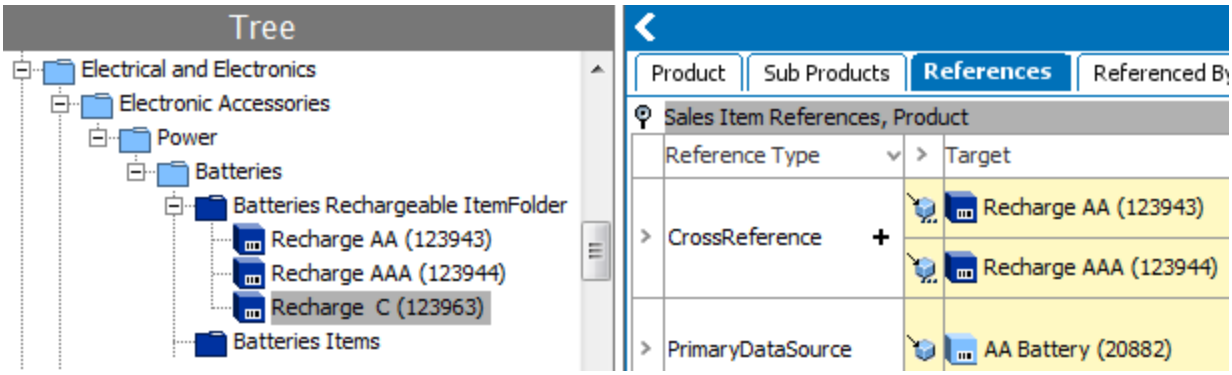
```
<ProductCrossReference Type="CrossReference"/>
```

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>
```

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, see the online version of this 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, see the **Unique Keys** section of the **System Setup / Super User Guide** documentation.

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>
   ...
...
```

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.

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

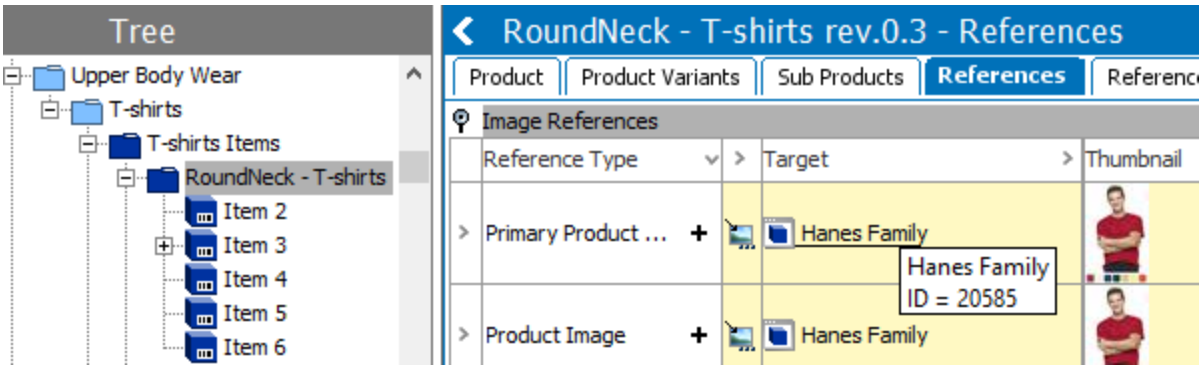
To inherit references to a particular object type, the following tag must include the product object type that references should be inherited to. If the type is not added, then all reference types will be exported.

For example:

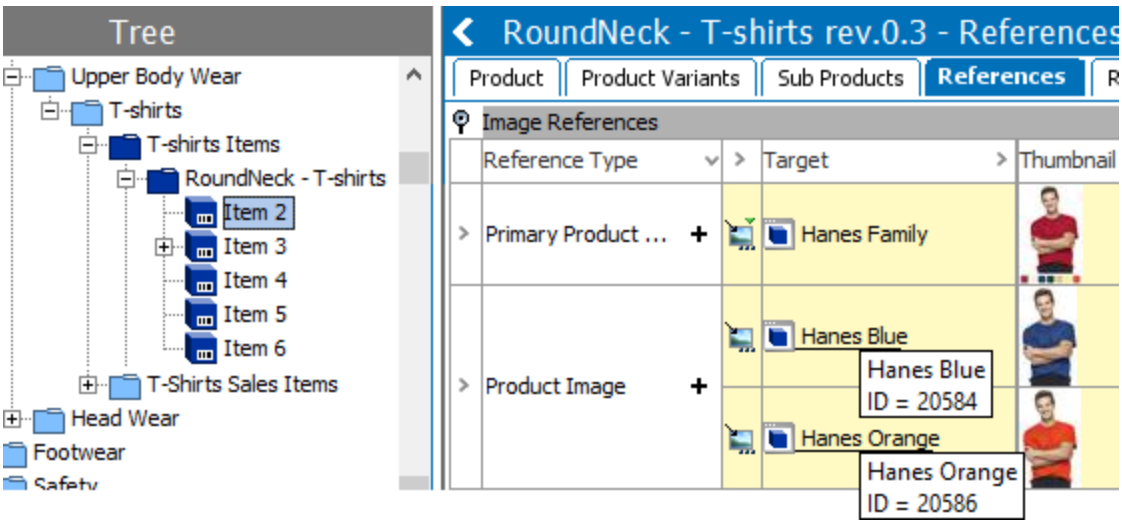
```
<AssetCrossReference Type="ProductImage" IncludeInheritedReferences="Level 3"/>
```

Input 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/>
       <Values IncludeInheritedAttributes="Level 3"/>
<AssetCrossReference Type="ProductImage" IncludeInheritedReferences="Level 3"/>
or
<AssetCrossReference IncludeInherited="true"/>
       <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, even when an inherited value exists, no value is output.

The IncludeInherited and IncludeInheritedAttributes XML attributes allow inherited values to output to top-level objects or to specified object types.

IncludeInherited

Local values are included for all objects in the output. To avoid duplicate values throughout the file, inherited values are only included once, and display for the top-level 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.

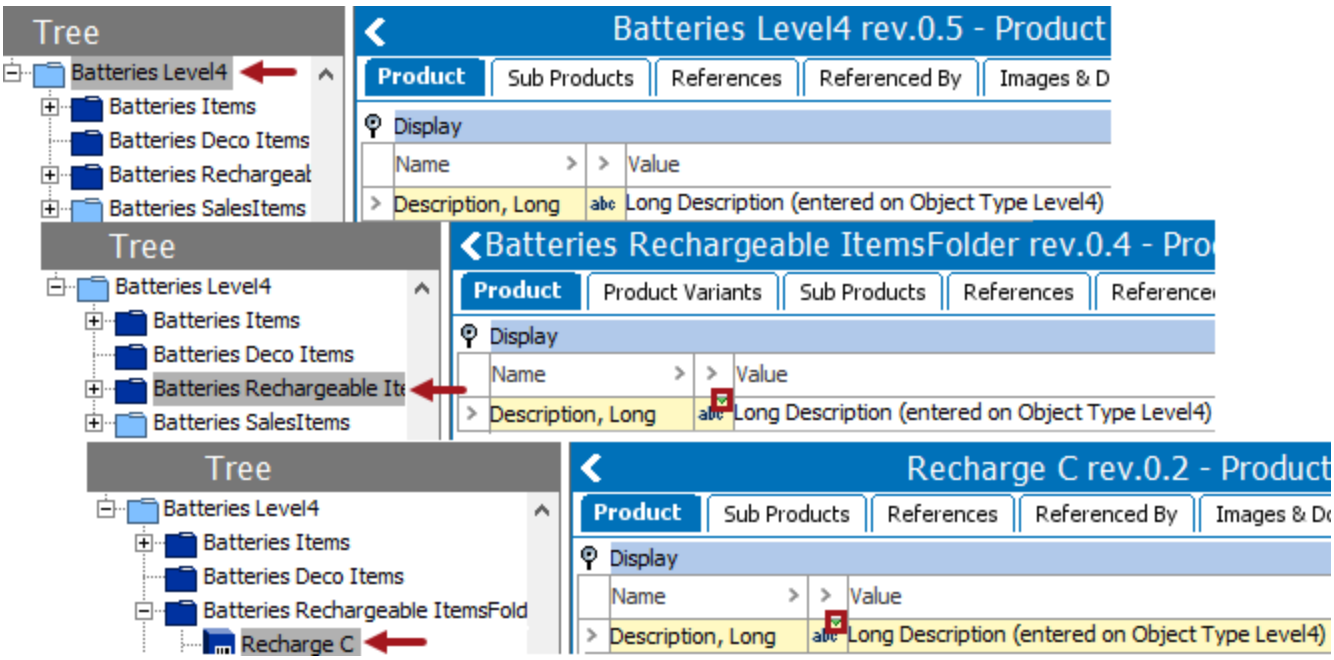
```
<Values IncludeInheritedAttributes="ItemFolder"/>
<Values IncludeInheritedAttributes="Item"/>
```

Note: IncludeInheritedAttributes will show attribute values on object types where they are not valid. For this reason, they cannot be imported back into STEP. The import process fails with errors saying each attribute is not valid for the object.

Input Data Example

With the IncludeInherited (set to true) or the IncludeInheritedAttributes XML attributes in the Advanced STEPXML Template parameter, the following data is exported:

- The 'Batteries Level4' object has a type of 'Level 4' and has a local value on the 'Description, Long' attribute.
- The 'Batteries Rechargeable ItemsFolder' object has a type of 'ItemFolder' and an inherited value (as indicated by the green down arrow) on the 'Description, Long' attribute.
- The 'Recharge C' object has a type of 'Item' and an inherited value (as indicated by the green down arrow) on the 'Description, Long' attribute.



Template

With the IncludeInherited XML attribute (set to true) or the IncludeInheritedAttributes XML attribute in the Advanced STEPXML Template parameter, the above data is exported:

IncludeInherited	IncludeInheritedAttributes
<?xml version="1.0" encoding="UTF-8"?> <STEP-ProductInformation> <Products ExportSize="Minimum"> <Product> <Name/> <Values **IncludeInherited="true"**/> </Product> </Products> </STEP-ProductInformation>	<?xml version="1.0" encoding="UTF-8"?> <STEP-ProductInformation> <Products ExportSize="Minimum"> <Product> <Name/> <Values **IncludeInheritedAttributes="ItemFolder"**/> <Values **IncludeInheritedAttributes="Item"**/> </Product> </Products> </STEP-ProductInformation>

Note: The XML attribute ExportSize="Minimum" is required on the Products tag for IncludeInheritedAttributes to get attributes to be inherited to the particular object type.

Results

Two (2) levels are exported from the product hierarchy: ItemFolder and Item.

- Local and inherited values are output for the top level (ItemFolder)
- Only local values are displayed for the lower level (Item)

IncludeInherited

```
<STEP-ProductInformation ExportTime="2015-12-23 13:52:56" ExportContext="Context1" ContextID="Context1" W

  <Products>
    <Product ID="123942" UserTypeID="ItemFolder" ParentID="20878">
      <Name Changed="true">Batteries Rechargeable ItemFolder</Name>
      <Values>
        <Value AttributeID="PrimaryColor">Green (entered on Object Type ItemFolder)</Value>
        <Value AttributeID="DescriptionLong">Long Description (entered on Object Type Level4)</Value>
      </Values>
      <Product ID="123963" UserTypeID="Item">
        <Name Changed="true">Recharge  C</Name>
        <Values>
          <Value AttributeID="ProductGroupDescription">Description (entered on Object Type Item)</Value>
        </Values>
      </Product>
    </Product>
  </Products>
</STEP-ProductInformation>
```

Three (3) levels are exported from the product hierarchy: Level4, ItemFolder, and Item.

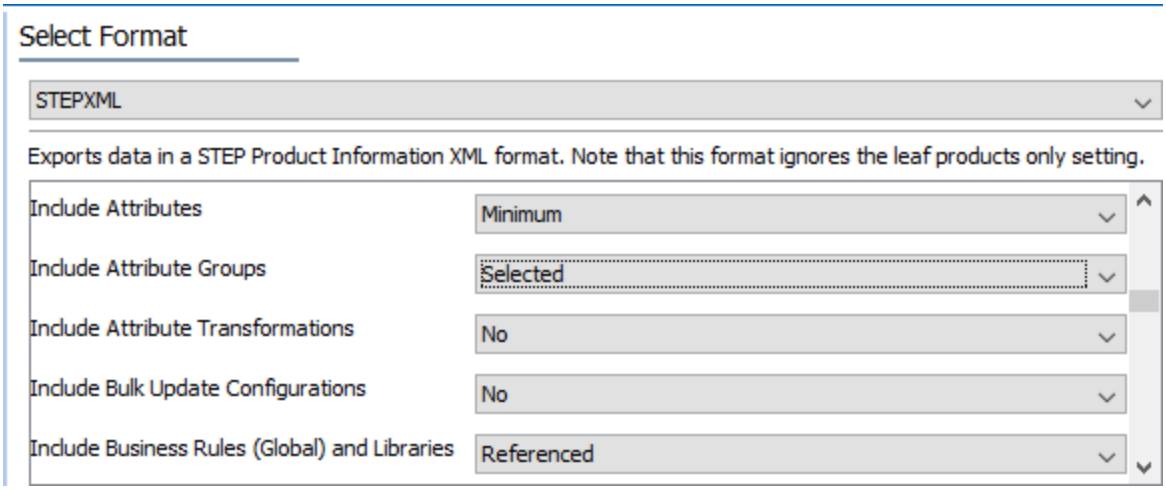
- Only local values are output for the top level (Level4)
- Inherited values are displayed for both of the lower levels (ItemFolder and Item)

IncludeInheritedAttributes

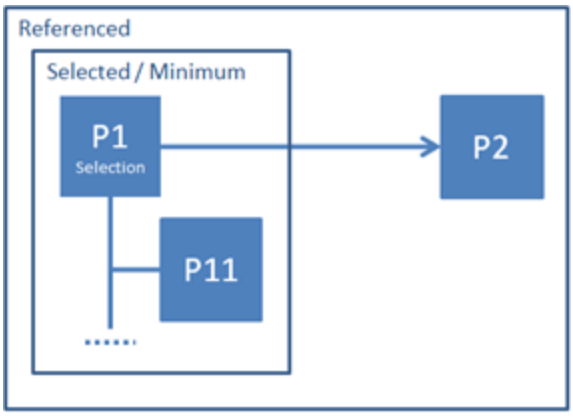
```
<STEP-ProductInformation ExportTime="2015-12-23 13:24:52" ExportContext="Context1" ContextID="Context1" Wo

  <Products>
    <Product ID="20878" UserTypeID="Level4" ParentID="20876">
      <Name Changed="true">Batteries Level4</Name>
      <Values>
        <Value AttributeID="DescriptionLong">Long Description (entered on Object Type Level4)</Value>
      </Values>
      <Product ID="123942" UserTypeID="ItemFolder">
        <Name Changed="true">Batteries Rechargeable ItemFolder</Name>
        <Values>
          <Value AttributeID="PrimaryColor">Green (entered on Object Type ItemFolder)</Value>
          <Value AttributeID="DescriptionLong">Long Description (entered on Object Type Level4)</Value>
        </Values>
        <Product ID="123963" UserTypeID="Item">
          <Name Changed="true">Recharge  C</Name>
          <Values>
            <Value AttributeID="PrimaryColor">Green (entered on Object Type ItemFolder)</Value>
            <Value AttributeID="DescriptionLong">Long Description (entered on Object Type Level4)</Value>
            <Value AttributeID="ProductGroupDescription">Description (entered on Object Type Item)</Value>
          </Values>
        </Product>
      </Product>
    </Product>
  </Products>
</STEP-ProductInformation>
```

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.

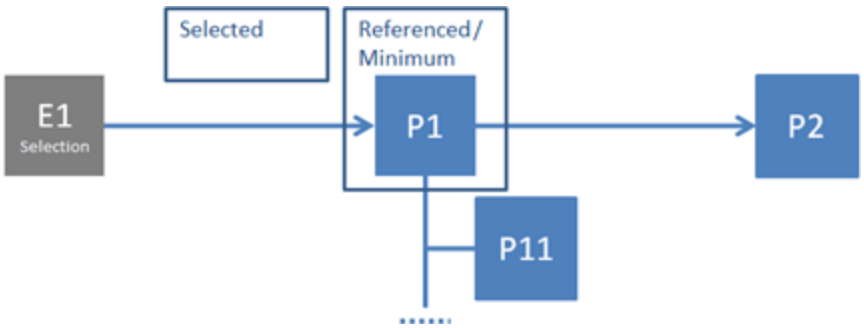


In the example below, product P1 has been selected as root node for the export. The Selected and Minimum options cause P1 and its descendants to be exported. The Referenced setting will also include P2 since it is referenced from P1.

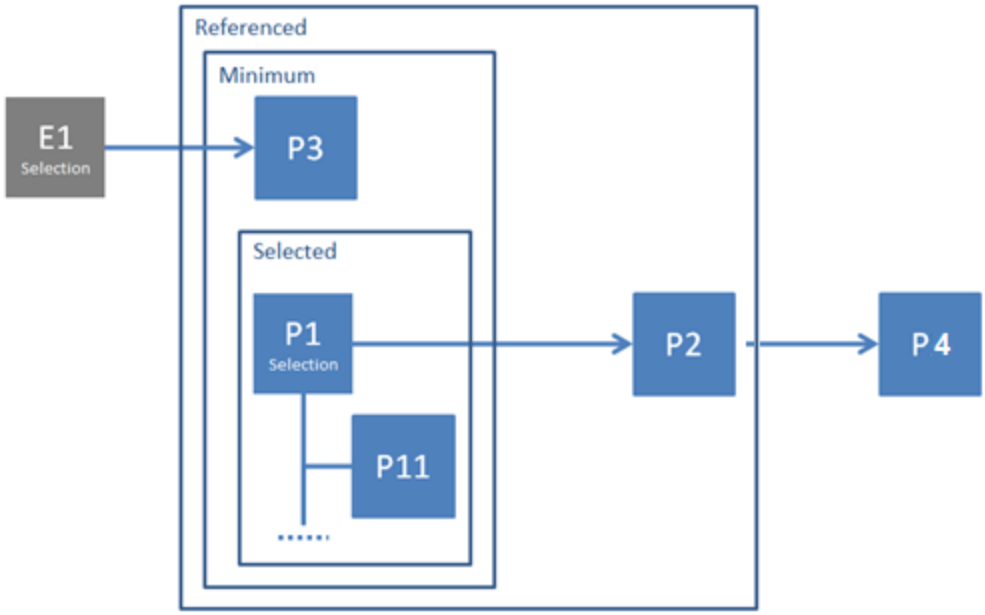


When exporting a product hierarchy, selecting Minimum outputs only the objects used in the selected product hierarchy.

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.



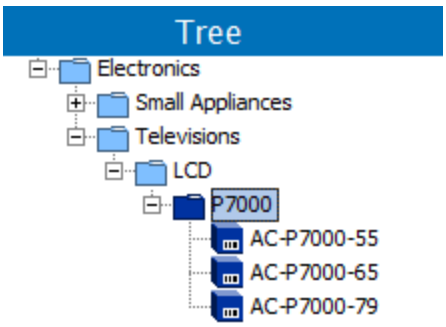
In the following example, the reason for having three options is demonstrated. Both a non-product (E1) and a product (P1) are selected as root nodes for the export. With this selection, Referenced, Minimum, and Selected will have three different meanings.



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">
            <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>
    <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>
    <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"/>
      <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>
```

1
2
3
4

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>
```

Multi-Context Data 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.

Input 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, see 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, see the **OIEP - Post-Processor - Context Splitter** topic in the **Outbound Integration Endpoints** documentation.

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>
```

The following XML attributes are available:

ExportSize

- Minimum - default setting. Other options are Selected and All.

For details about export size, see 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, see 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, see the **Filter References in STEPXML** section.

ProductCrossReference

The <ProductCrossReference/> tag returns all product cross references as output. For more information about cross reference output, see the **Filter References in STEPXML** section.

ClassificationReference

The <ClassificationReference/> tag returns all classification cross references as output. For more information about cross reference output, see the **Filter References in STEPXML** section.

Additionally, see 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>
```

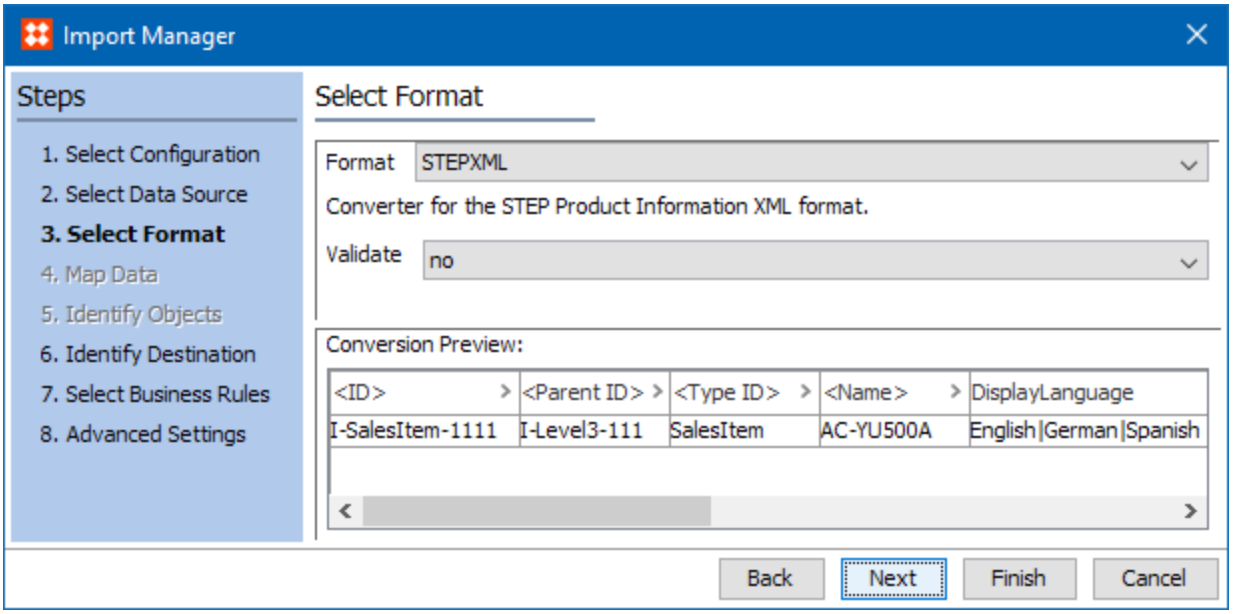
Product with Units, References, and Multi-Values Attributes Example

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, see 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, see the **Owns Product Links on Alternate Classifications Object Type** topic in the **System Setup / Super User Guide** documentation.

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 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

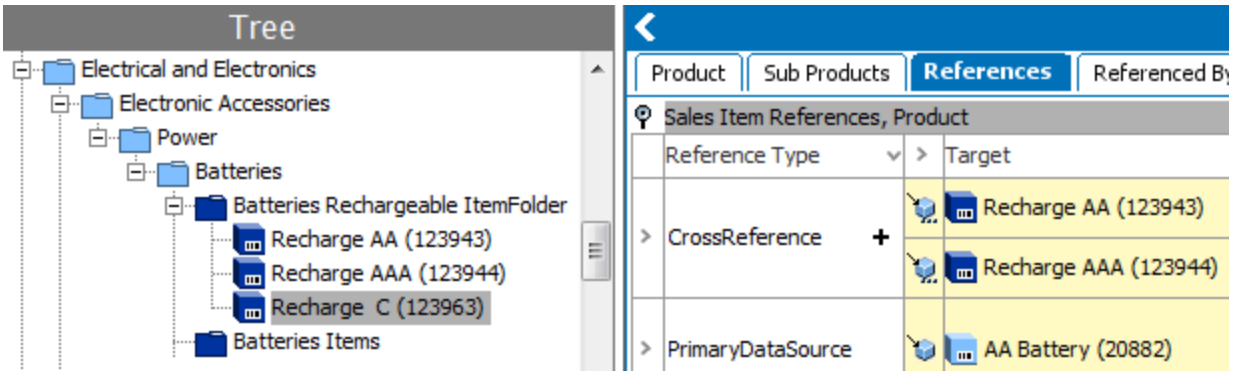
The MetaData tag is used to output attribute values maintained on the product reference itself.

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/> <Values/>

Referenced Without Embedded	Referenced With Embedded
</Product> </ProductCrossReference> </Product> </Products> </STEP-ProductInformation>	</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

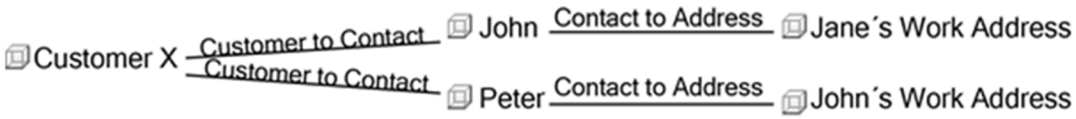
Referenced Without Embedded

```
<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>
```

ReplacementRules Tag in STEPXML

The STEPXML import functionality generally runs in a mode where properties not represented in the XML file are left untouched by the import. For 'list properties' (multiple instances of the same XML element at the same level) such as 'Value' elements inside the 'Values' element for a product or 'TargetUserTypeLink' elements for a reference type definition, this means that special processing instructions must be used to express that properties not present in the import file must be removed from the system as part of the import.

All replace options must be configured in supertype-specific subsections of the 'ReplacementRules' element.

Using the ReplacementRules Tag

While you can delete elements using the ReplacementRules tag, use caution as this removes **all** elements of the indicated type not present in the import file for **all** loaded objects.

See below for an example, plus a full list of tags and their descriptions:

```
<?xml version="1.0" encoding="utf-8"?>
<STEP-ProductInformation ContextID="Context1" WorkspaceID="Main">
   <ReplacementRules>
     <Attributes>
        <ReplaceAttributeGroupLinks/>
        <ReplaceUserTypeLinks/>
        <ReplaceUnitLinks/>
     </Attributes>
     <LinkTypes>
       <ReplaceUserTypeLinks/>
       <ReplaceTargetUserTypeLinks/>
     </LinkTypes>
     <UserTypes>
       <ReplaceUserTypeLinks/>
     </UserTypes>
     <UserGroups>
       <ReplacePrivilegeRules/>
     </UserGroups>
     <Products>
       <ReplaceCrossReferences CrossContext="Y"/>
       <ReplaceCrossReferences ReferenceTypeID="MyReferenceType" CrossContext="Y"/>
       <ReplaceOverrideSubProducts/>
       <ReplaceValues CrossContext="Y"/>
       <ReplaceValues AttributeGroupID="MyAttrGrp" CrossContext="Y"/>
       <ReplaceValues AttributeID="MyAttr" CrossContext="Y"/>
       <ReplaceAttributeLinks CrossContext="Y"/>
       <ReplaceClassificationReferences CrossContext="Y"/>
       <ReplaceClassificationReferences TypeID="MyClassificationProductLinkType"/>
       <ReplaceDataContainers/>
       <ReplaceDataContainers DataContainerTypeID="MyDcType"/>
     </Products>
     <Classifications>
```

```
        <ReplaceCrossReferences CrossContext="Y"/>
        <ReplaceCrossReferences ReferenceTypeID="MyReferenceType" CrossContext="Y"/>
        <ReplaceAttributeLinks CrossContext="Y"/>
        <ReplaceProductReferences CrossContext="Y"/>
        <ReplaceProductReferences TypeID="MyClassificationProductLinkType"
CrossContext="Y"/>
        <ReplaceValues CrossContext="Y"/>
        <ReplaceValues AttributeGroupID="MyAttrGrp" CrossContext="Y"/>
        <ReplaceValues AttributeID="MyAttr" CrossContext="Y"/>
      </Classifications>
      <Entities>
        <ReplaceCrossReferences CrossContext="Y"/>
        <ReplaceCrossReferences ReferenceTypeID="MyReferenceType" CrossContext="Y"/>
        <ReplaceDataContainers/>
        <ReplaceDataContainers DataContainerTypeID="MyDcType"/>
        <ReplaceValues CrossContext="Y"/>
        <ReplaceValues AttributeGroupID="MyAttrGrp" CrossContext="Y"/>
        <ReplaceValues AttributeID="MyAttr" CrossContext="Y"/>
      </Entities>
      <Assets>
        <ReplaceCrossReferences CrossContext="Y"/>
        <ReplaceCrossReferences ReferenceTypeID="MyReferenceType" CrossContext="Y"/>
        <ReplaceValues CrossContext="Y"/>
        <ReplaceValues AttributeGroupID="MyAttrGrp" CrossContext="Y"/>
        <ReplaceValues AttributeID="MyAttr" CrossContext="Y"/>
      </Assets>
    </ReplacementRules>
    <!-- Data omitted -->
</STEP-ProductInformation>
```

Attributes

Tag	Description
ReplaceAttributeGroupLinks	For attribute definitions included in the XML file, replace existing attribute group links with those provided in the file. Notice that an attribute must always be linked to / included in at least one attribute group.
ReplaceUserTypeLinks	For attribute definitions included in the XML file, replace existing user type links with those provided in the file. User type links are the links between an attribute and an object type that make the attribute valid for instances of the object type. The update expressed with this instruction can only be performed if there a no present (non-historical) values for the attribute on instances of the object type(s) for which the validity is sought removed.
ReplaceUnitLinks	For attribute definitions included in the XML file, replace existing unit links with those provided in the file. Unit links are the links between an attribute and a unit that make the unit valid for the attribute. The update expressed with this instruction can only be performed if there a no present (non-historical) values for the attribute using the unit(s) to be made invalid. Notice that if the attribute 'Validation' element in which 'UnitLink' elements reside is left out for attributes in the XML file and this instruction is used, all existing valid units will be sought removed.

LinkTypes

Tag	Description
ReplaceUserTypeLinks	For link type definitions (reference types and classification product link types) included in the XML file, replace existing user type links with those provided in the file. User type links 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' always represent the product side and 'target user type links' represent the classification side, regardless of the link ownership. The update expressed with this instruction can only be performed if there a no present (non-historical) links from instances of the object type(s) sought made invalid.
ReplaceTargetUserTypeLinks	For link type definitions (reference types and classification product link types) included in the XML 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' always represent the product side and 'target user type links' represent the classification side, regardless of the link ownership. The update expressed with this instruction can only be performed if there a no present (non-historical) links to instances of the object type(s) sought made invalid.

UserTypes

Tag	Description
ReplaceUserTypeLinks	For user type definitions (i.e., object type definitions) included in the XML file, replace existing user type links with those provided in the file. User type links are in this context links between object types that make an object type legal below another (relation is owned by the child type). The update expressed with this instruction can only be performed if the links sought removed are not used in any present (non-historical) revisions. I.e., if the update expresses that instances of object type A may no longer be children of instances of object type B, the update cannot be performed if there is present data with this relationship. Note that user defined object types always must have at least one user type link.

UserGroups

Tag	Description
ReplacePrivilegeRules	For user group definitions included in the XML file, replace existing privilege rules with those provided in the file. Notice that especially with this instruction, due diligence must be observed. If, for example, all privilege rules are removed for all users, no one will be able to access the STEP system on which the file is imported.

Products

Tag	Description
ReplaceCrossReferences	For products included in the XML file, replace existing cross references owned by the products (not including classification product links) with those provided in the file. This instruction can either be configured to apply for all reference types or by setting the 'ReferenceTypeID' attribute to be configured to only apply for references of a specific reference type. The instruction can further, via the 'CrossContext' attribute, be configured to work only for the import context, replacing only references local to the import context (default) or for all contexts. If a reference type is not dimension dependent, the CrossContext attribute value will have no effect. For a given reference type matched by multiple rules, with regards to the 'CrossContext' setting, the setting from the most specific instruction will be used.
ReplaceOverrideSubProducts	For product overrides included in the XML file, replace override sub products with those provided in the file. Override sub products are normal product objects 'adopted' by a product override. Note that product override children also are represented using the 'OverrideSubProduct' elements in STEPXML. Such 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.
ReplaceValues	For products included in the XML file, replace existing values owned by the products with those provided in the file. This instruction can either be configured to apply for all attributes, attributes below a specific group ('AttributeGroupID' attribute) or a specific attribute ('AttributeID' attribute). The instruction can further, via the 'CrossContext' attribute, be configured to work only for the import context, replacing only values local to the import context (default) or for all contexts. If an attribute is not dimension dependent, the CrossContext attribute value will have no effect. For a given attribute matched by multiple rules, with regards to the 'CrossContext' setting, the setting from the most specific instruction will be used. If the attribute is in multiple groups with diverting values for 'CrossContext', “N” (No) will overrule “Y” (Yes).
ReplaceAttributeLinks	For products included in the XML file, replace existing attribute links with those provided in the file. The instruction can, via the 'CrossContext' attribute, be configured to work only for the import context, replacing only links visible in the import context (default) or for all contexts. If attribute links are not dimension dependent, the CrossContext attribute value will have no effect.
ReplaceClassificationReferences	For products included in the XML file, replace existing classification references (also known as classification product links) owned by the products with those provided in the file. This instruction can either be configured to apply for all classification product link types or by setting the 'TypeID' attribute to be configured to only apply for links of a specific classification product link type.

Tag	Description
	The instruction can further, via the 'CrossContext' attribute, be configured to work only for the import context, replacing only links local to the import context (default) or for all contexts. If a classification product link type is not dimension dependent, the CrossContext attribute value will have no effect. For a given classification product link type matched by multiple rules, with regards to the 'CrossContext' setting, the setting from the most specific instruction will be used.
ReplaceDataContainers	For products included in the XML file, replace existing data container objects owned by the products with those provided in the file. This instruction can either be configured to apply for all data container types or by setting the 'DataContainerTypeID' attribute to be configured to only apply for links of a specific data container type.

Classifications

Tag	Description
ReplaceCrossReferences	The functionality for classifications is identical to what is described for products in Products > ReplaceCrossReferences.
ReplaceAttributeLinks	The functionality for classifications is identical to what is described for products in Products > ReplaceAttributeLinks.
ReplaceProductReferences	For classifications included in the XML file, replace existing product references (also known as classification product links) owned by the classifications with those provided in the file. This instruction can either be configured to apply for all classification product link types or by setting the 'TypeID' attribute, be configured to only apply for links of a specific classification product link type. The instruction can further, via the 'CrossContext' attribute, be configured to work only for the import context, replacing only links local to the import context (default) or for all contexts. If a classification product link type is not dimension dependent, the CrossContext attribute value will have no effect. For a given classification product link type matched by multiple rules, with regards to the “CrossContext” setting, the setting from the most specific instruction will be used.
ReplaceValues	The functionality for classifications is identical to what is described for products in Products > ReplaceValues.

Entities

Tag	Description
ReplaceCrossReferences	The functionality for entities is identical to what is described for products in Products > ReplaceCrossReferences.
ReplaceDataContainers	The functionality for entities is identical to what is described for products in Products > ReplaceDataContainers
ReplaceValues	The functionality for entities is identical to what is described for products in Products > ReplaceValues.

Assets

Tag	Description
ReplaceCrossReferences	The functionality for assets is identical to what is described for products in Products > ReplaceCrossReferences.
ReplaceValues	The functionality for assets is identical to what is described for products in Products > ReplaceValues.

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/>
```

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.

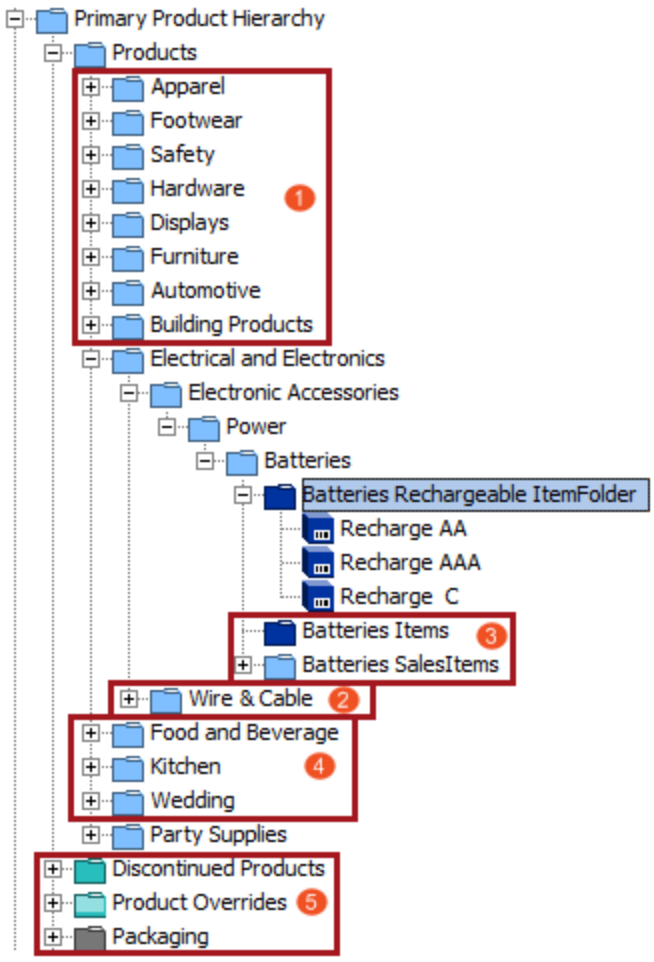
Note: When the Manually Sorted option is not activated, objects are output in what may appear to be a random order since they are not ordered by ID or name.

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, see **Single-Update Mode**.

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, see **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">
```

Note: The true and false settings for all XML attributes are case-sensitive and must be lower case as shown in the above example.

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, see 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

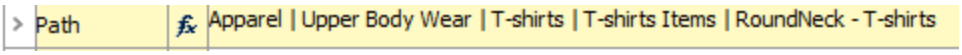
- False - default setting
- True - change markers indicate if data has been added to or removed from a product. This includes attribute values, product references and classification references.

For more information, see 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:



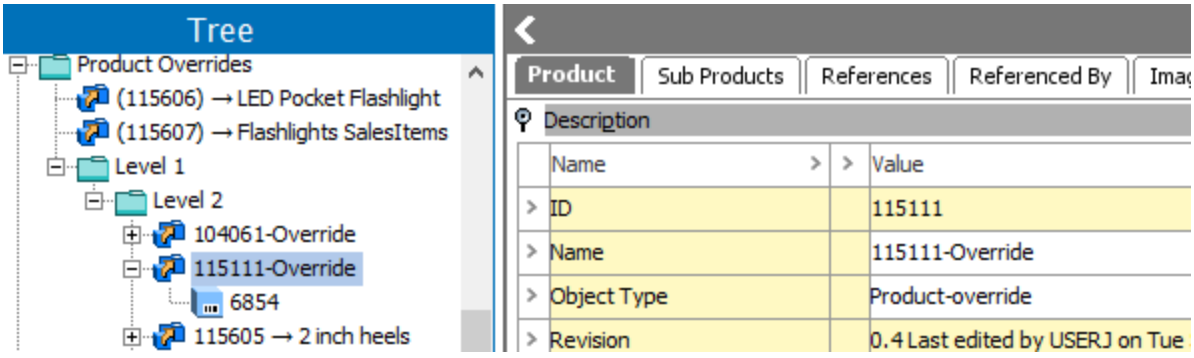
An excerpt from the output generated includes this data:

```
106 <Value AttributeID="Path" Derived="true">Apparel | Upper Body Wear | T-shirts | T-shirts Items | RoundNeck - T-shirts</Value>
```

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:

```
52  ...
53  <Product ID="115111" UserTypeID="std.AdaptorType" OverridesProductID="" ParentID="101608">
54    <Name>115111-Override</Name>
55    <Values>
58    </Values>
59    <OverrideSubProduct ProductID="6854"/>
60    ...
```

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:

	Name		Value
>	Long Item Description	abc	One of our best-selling T-shirts, available in Red or Black.

An excerpt from the output generated includes this data:

```
25  <Values>
26    ...
27    <Value AttributeID="LongItemDescription">One of our best-selling T-shirts, available in Red or Black.</Value>
28   ...
29  </Values>
```

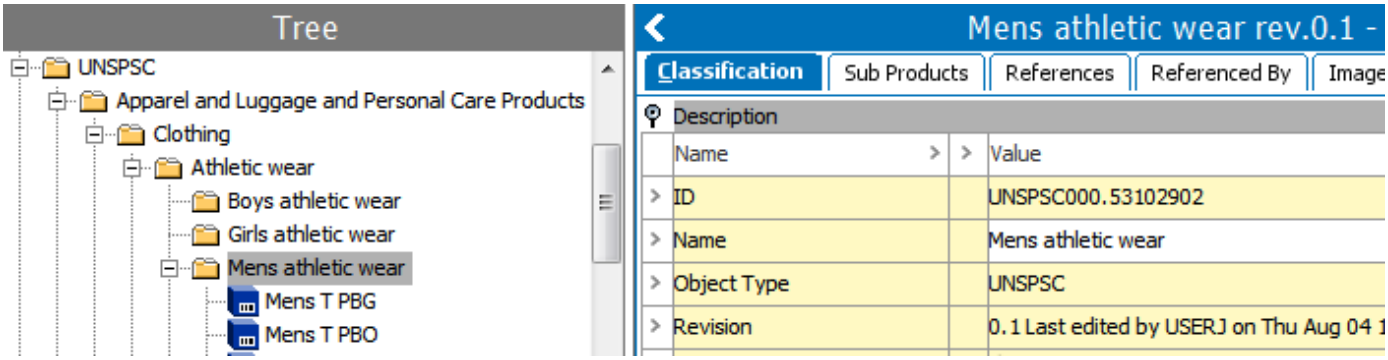
Validation

- None - default. Other option is are Schema and XSD

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, see the **Bulk Updates** documentation.



Format Availability

UNSPSC is available for selection in:

- IIEP - see **Creating an Inbound Integration Endpoint**
- Import Manager - see **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:

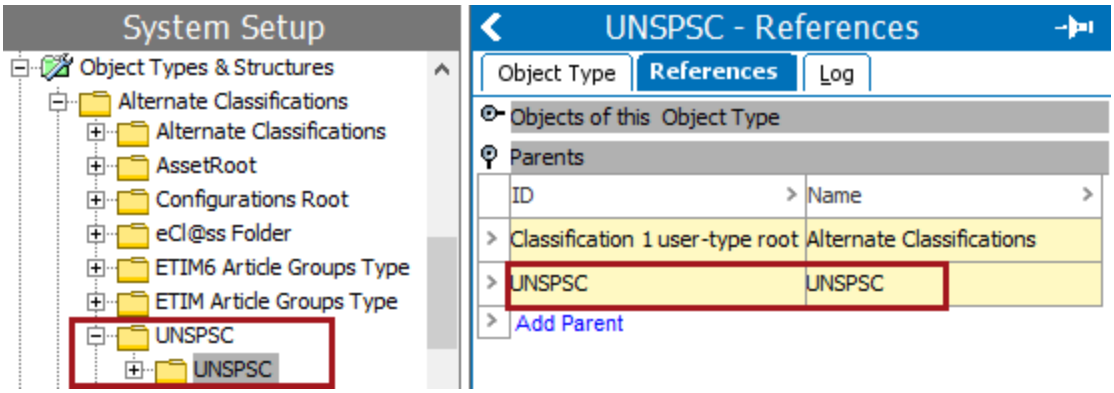
Version	File Types Included in ZIP	File Prep Required	File to Import
13	One (1) Excel file with six (6) tabs	Extract before import	Excel file
17, 18, 19, 20, 21	Six (6) individual .txt files	None	.ZIP file

Multiple versions of UNSPSC classifications can exist simultaneously in STEP, provided different UNSPSC classification ID prefixes are used.

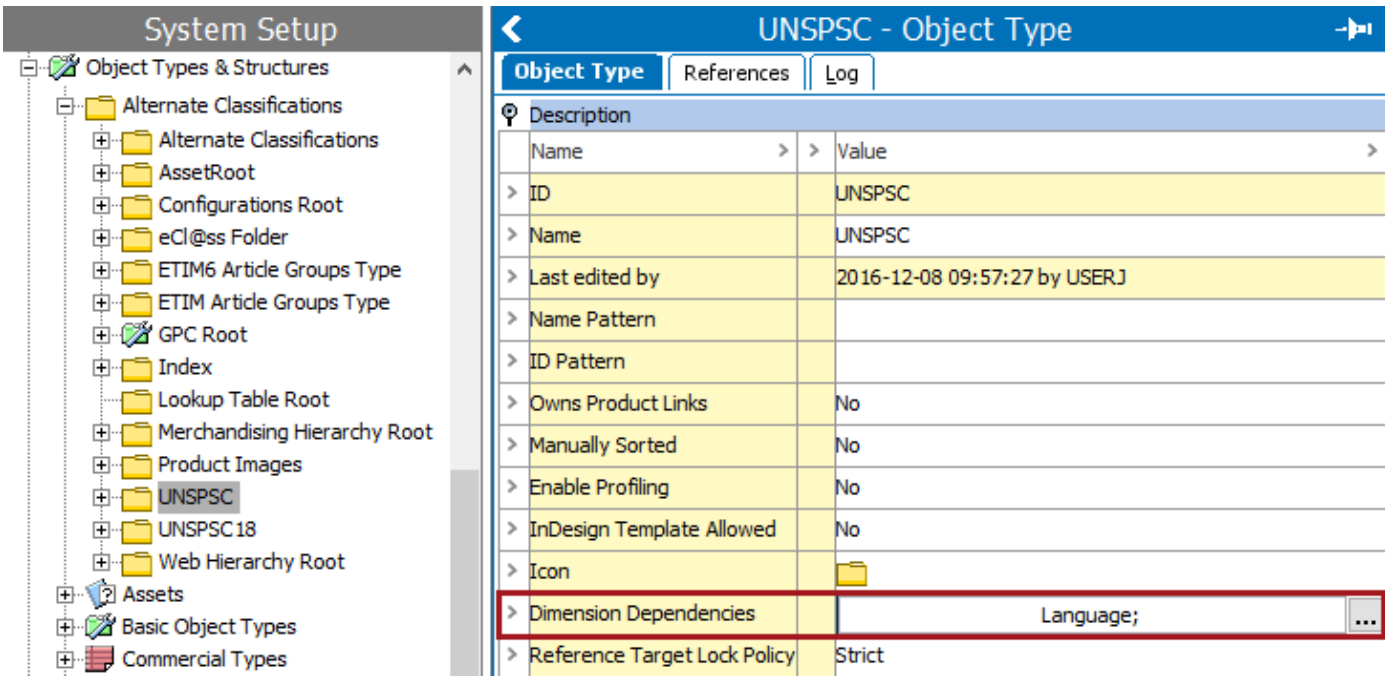
Prerequisites

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.



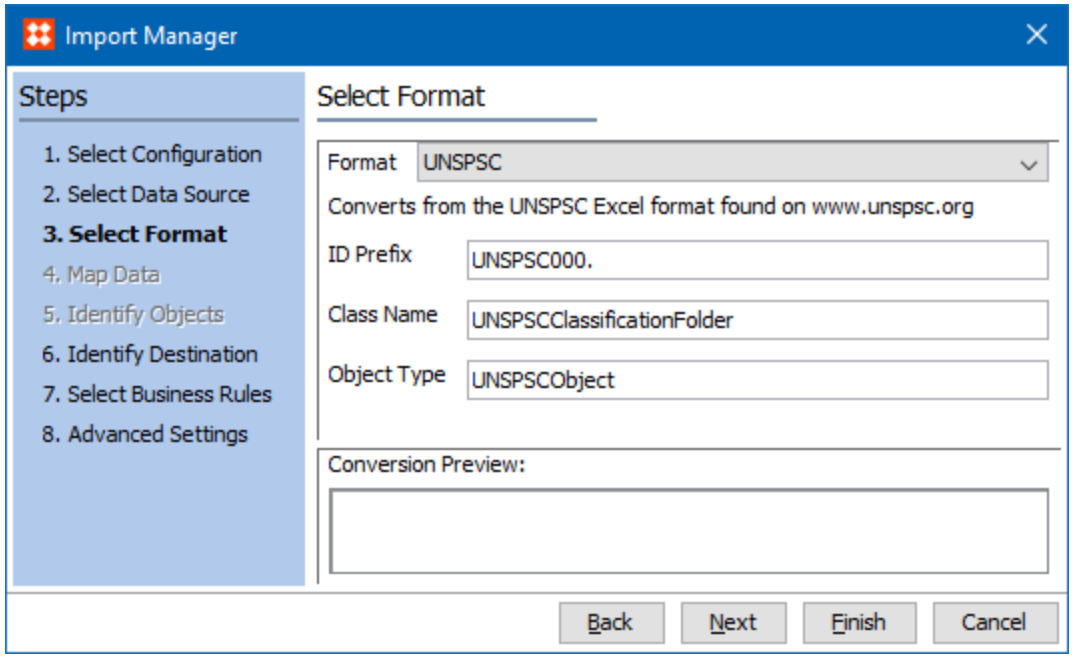
To import multiple UNSPSC languages for different contexts, on the top-level UNSPSC folder, set the Dimension Dependencies parameter to Language.



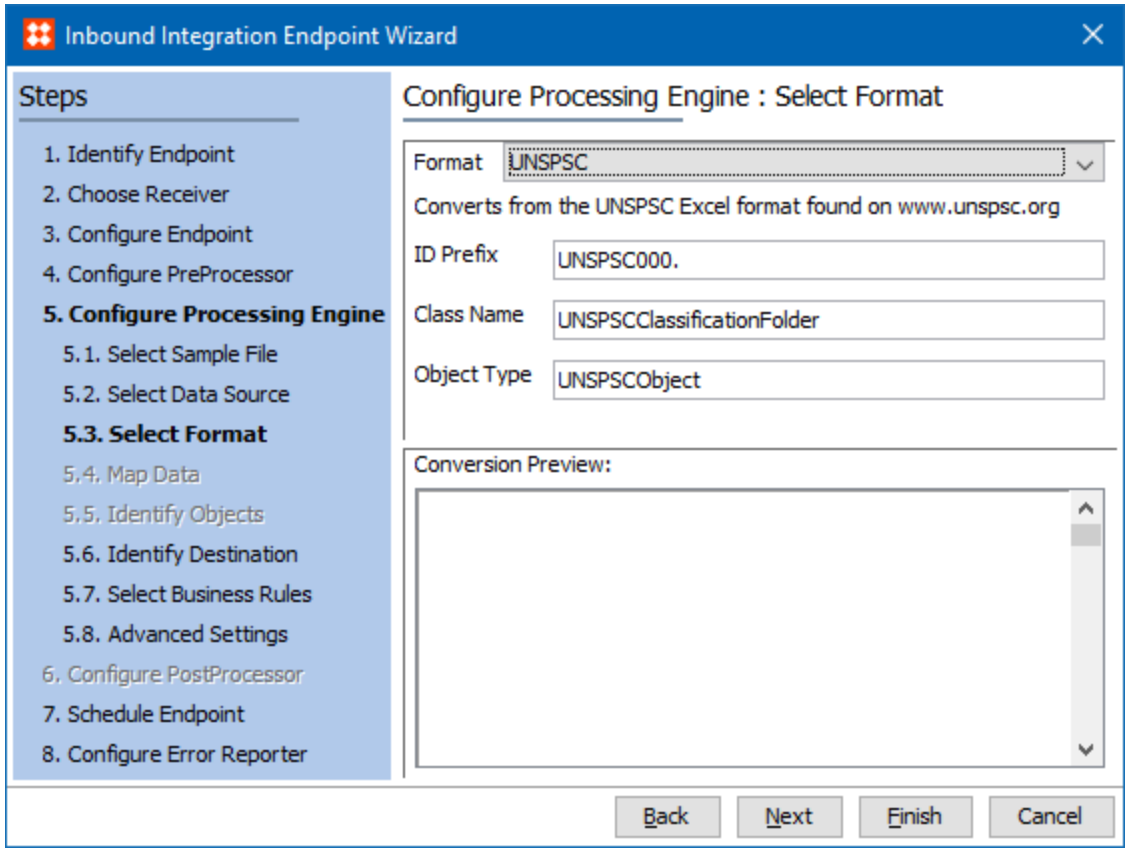
Inbound Parameters

- **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 (see 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 - see **Creating a Data Export**
- OIEP - see **Creating an Outbound Integration Endpoint**

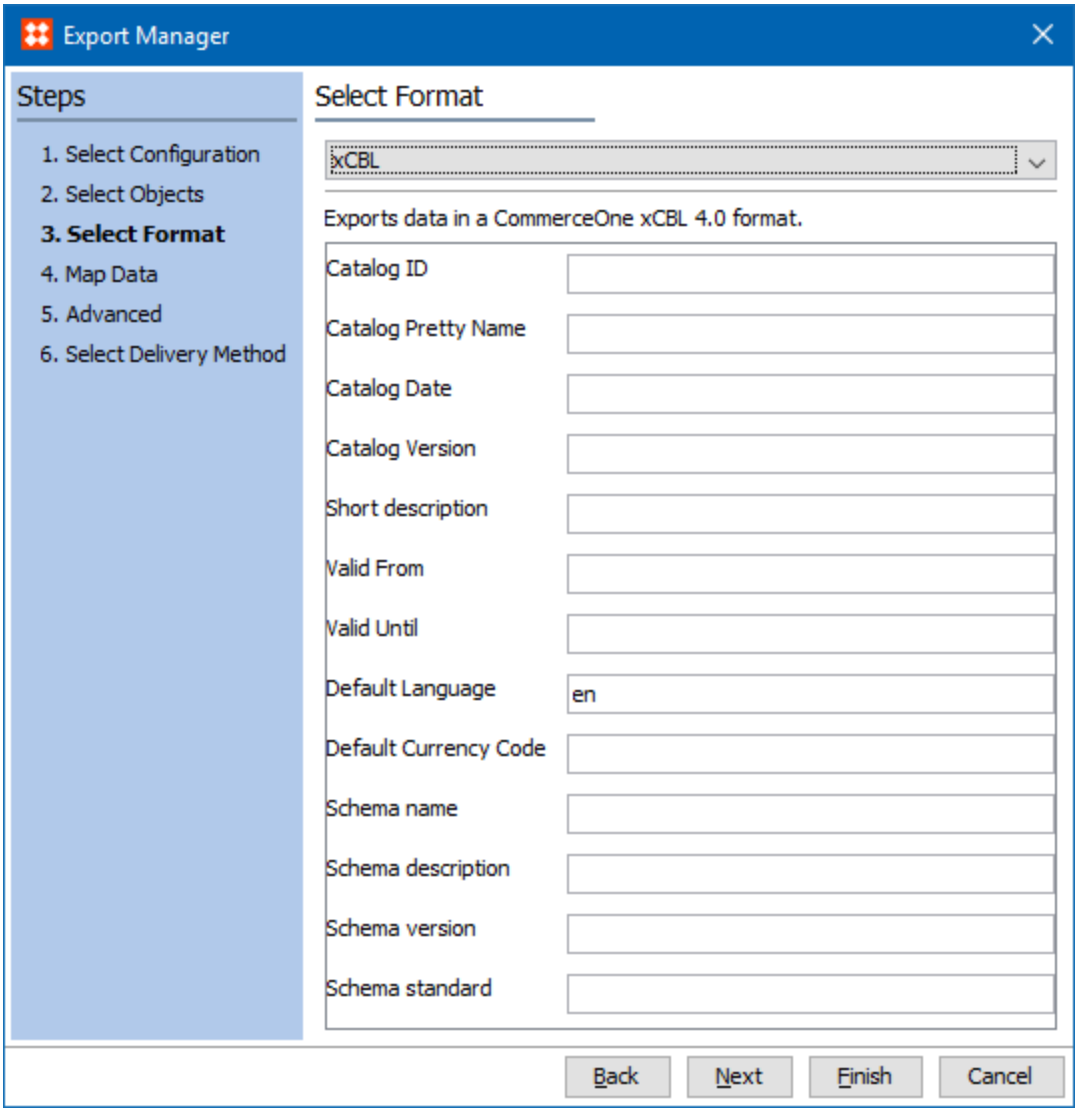
Mapping

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

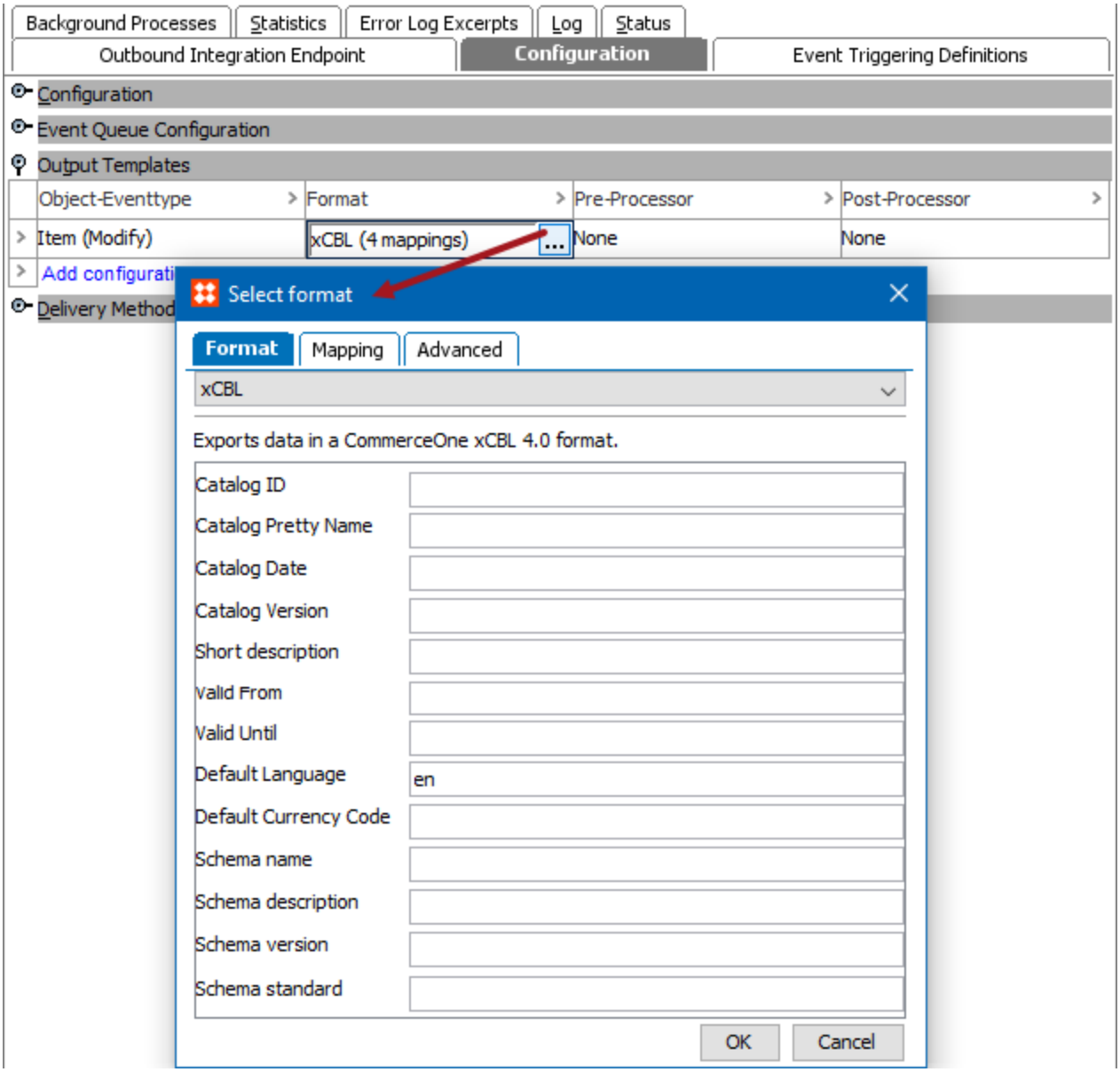
Outbound Data

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, see **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, see **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, see **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

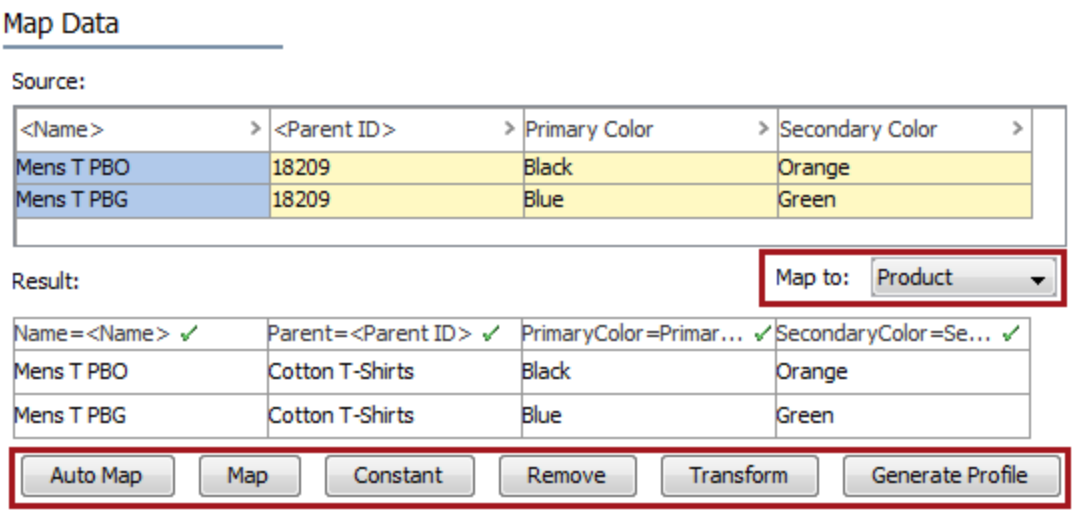
XML Formats

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, see 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, see 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, see 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, see 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, see the **Inbound Map Data - Remove** section.

For additional guidance, see 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, see 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, see 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, see the **Inbound Map Data - Generate Profile** section.

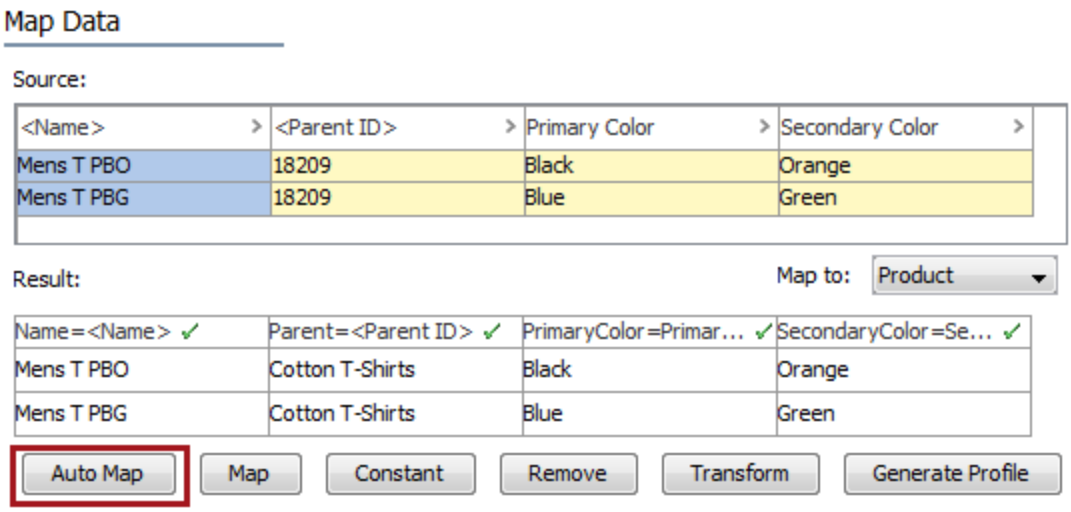
Tips for Mapping Data

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 / 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, see **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
<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

Header Text	Description
	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, see 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.'

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.

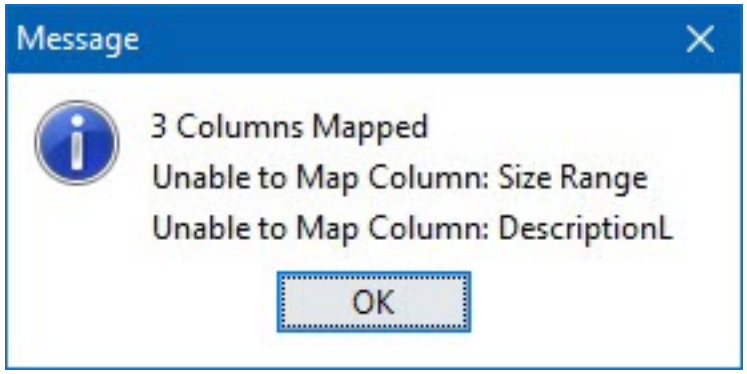
Auto Map Example

The file being imported includes 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

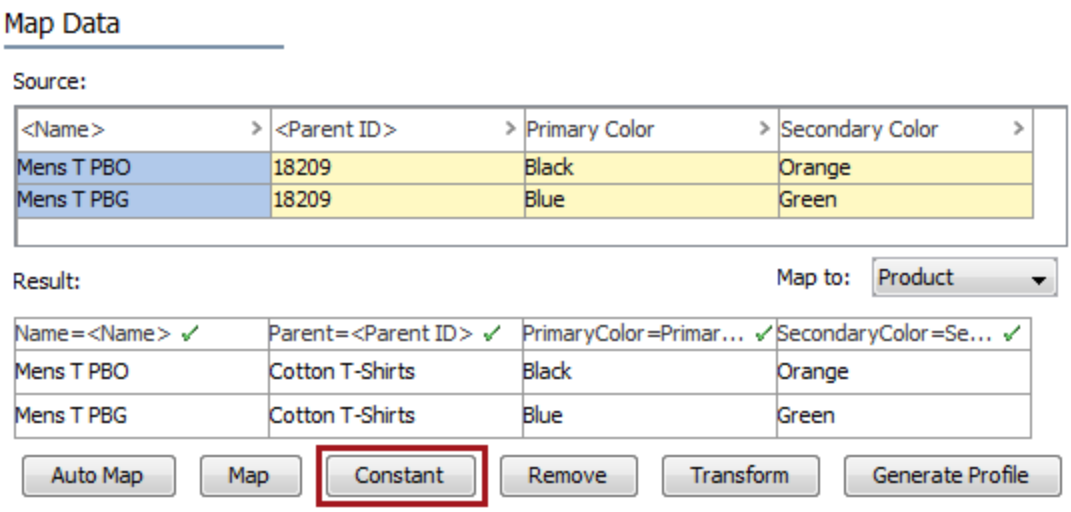
3. Review the **Result** panel. Make any adjustments or transformations if needed.

Note: 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.

4. When the mapping is correct, click **Next**.

Inbound Map Data - Constant

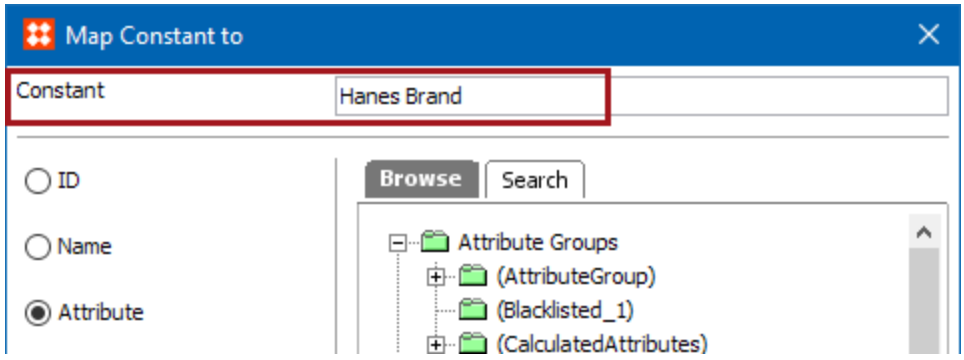
You can add an attribute value to all records by mapping a constant value. Common setup uses this feature for filling some object attributes.



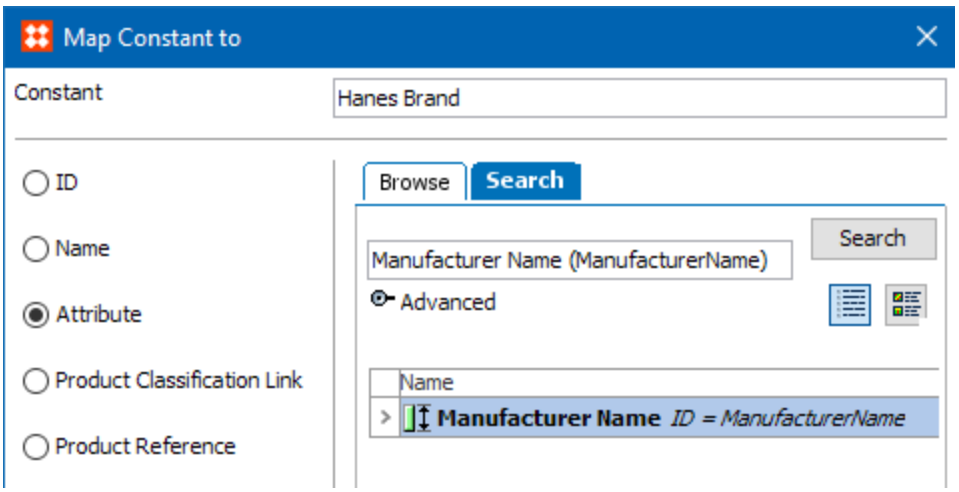
For more information about the additional import steps, see **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

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**.



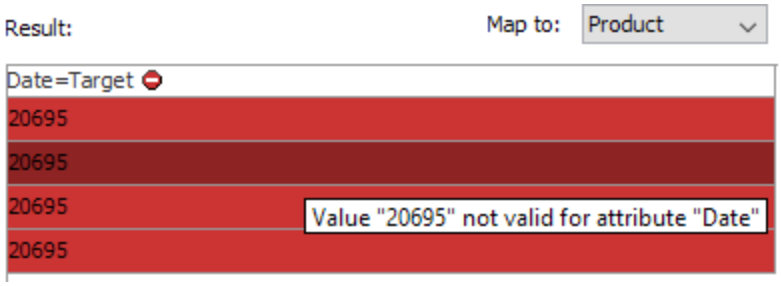
4. The **Result** panel displays the constant value as if it exists in the input file.

Result:

Map to: Product

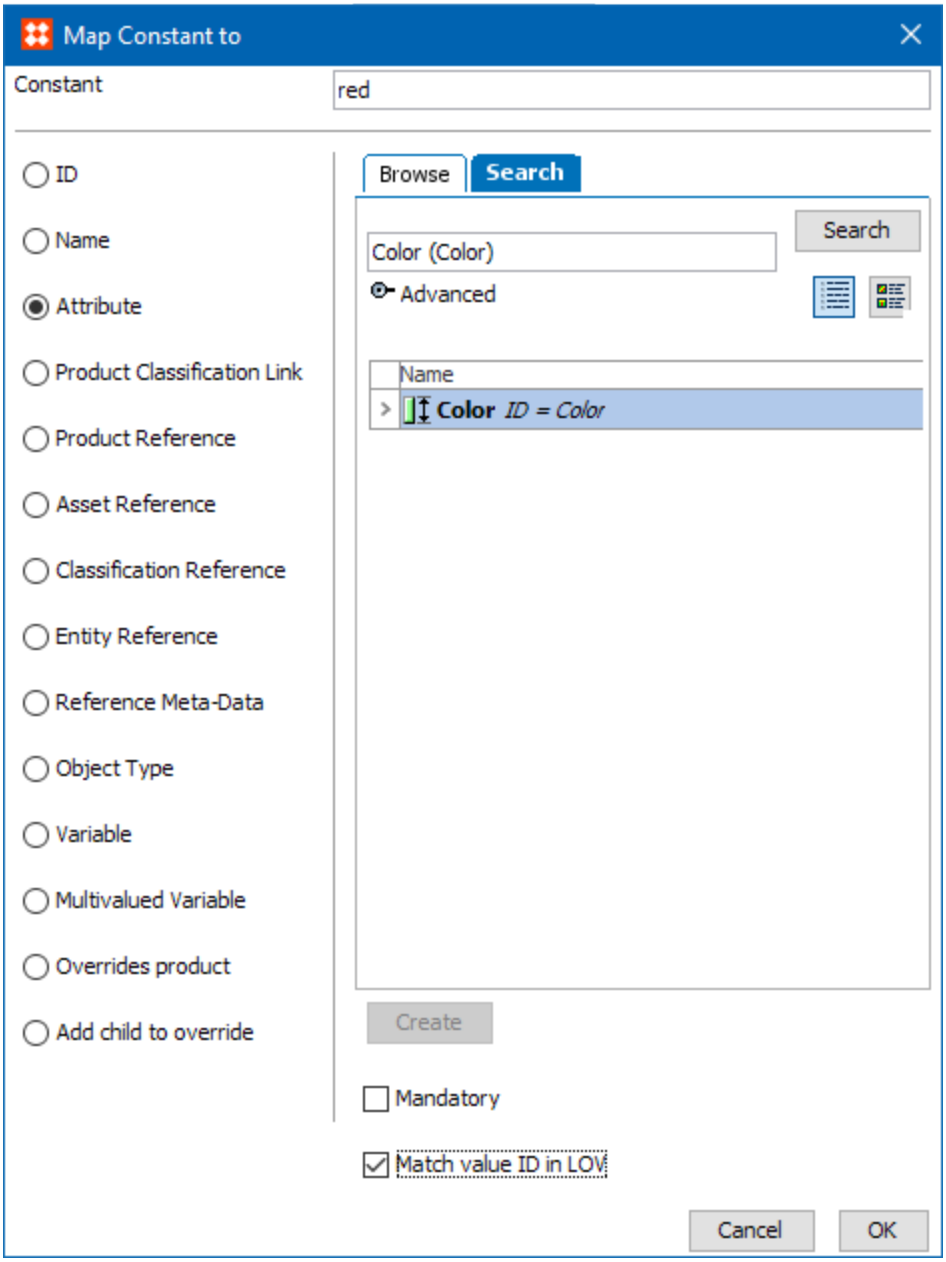
ID=<ID> ✓	Name=<Name> ✓	Parent=<Parent ID> ✓	PrimaryColor=(PrimaryColor) ✓	ManufacturerName=Hanes Brand ✓
18210	18210 M B	Cotton T-shirts	Blue	Hanes Brand
18212	18212 L B	Cotton T-shirts	Red	Hanes Brand
18213	18213 M O	Cotton T-shirts	Black	Hanes Brand
18216	18216 L O	Cotton T-shirts	Green	Hanes Brand
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.

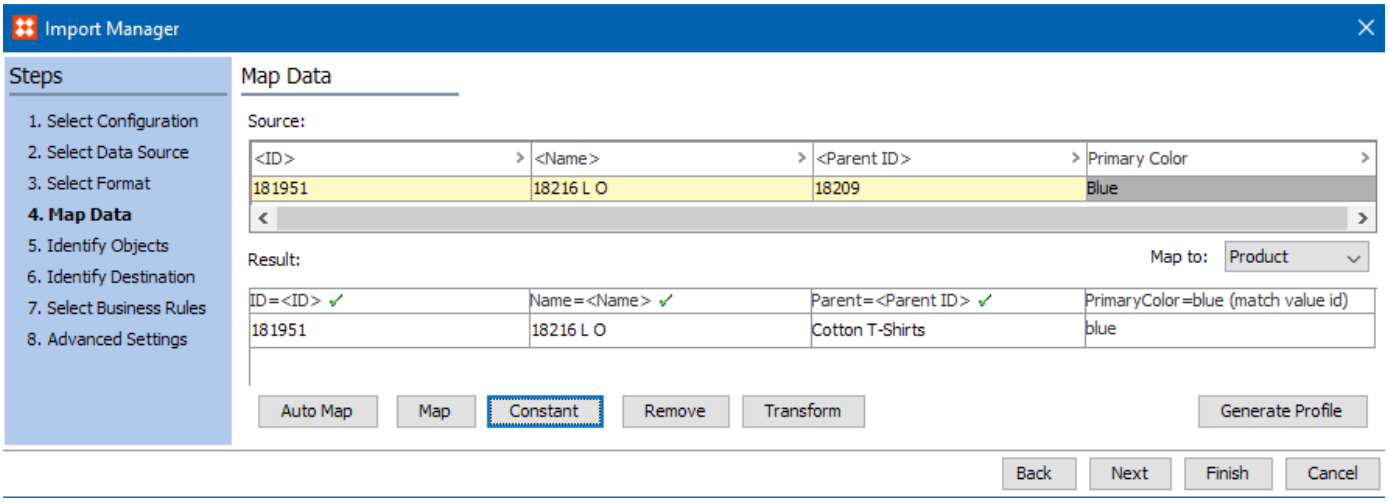


A user can also add the constant value to the LOV attribute, they must enter the value of the LOV, which is case sensitive.

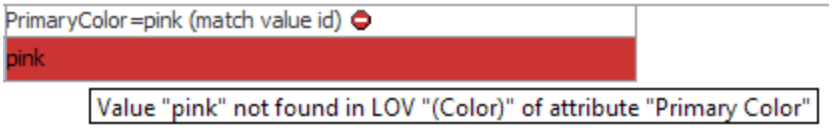
If the LOV attribute holds an LOV with value IDs, then the user can enter the value ID. If the option 'Match value ID in LOV' is checked, the IDs are matched against the IDs in the LOV, and values are applied based on the match, as shown below.



The Result panel displays the mapped values below, and matches the LOV ID.

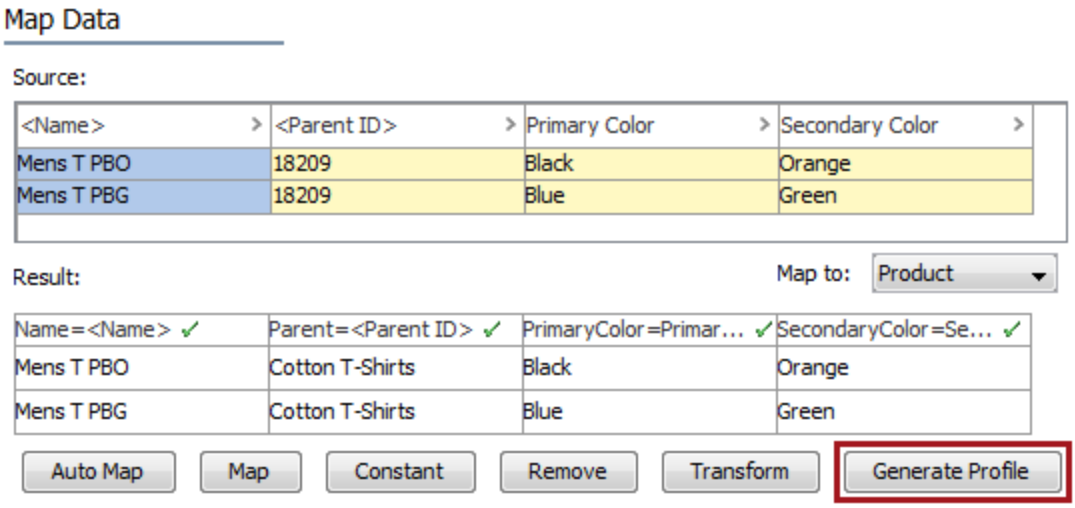


Note: The LOV ID is case sensitive, if there is a mismatch with the LOV ID that exists in STEP, validation will return an error as shown below.



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.



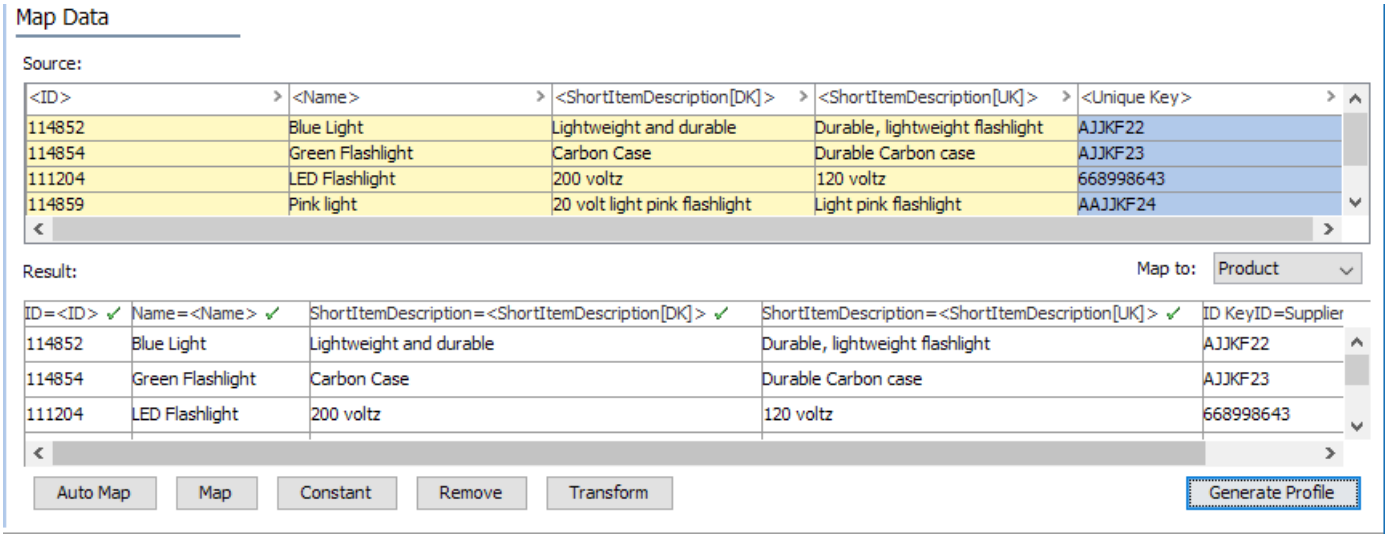
For more information about the additional import steps, see **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Profiling Setup

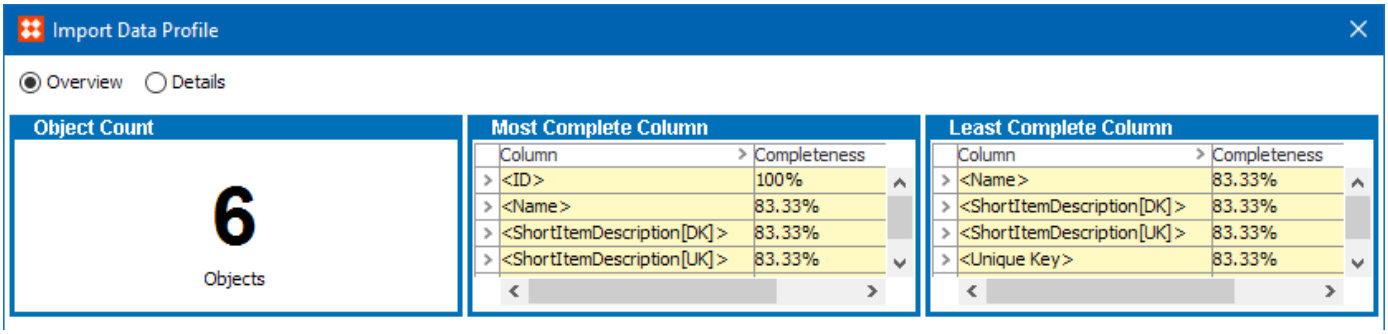
To limit the number of profiled rows and prevent out of memory exceptions, use the property **Import.Profile.Max.Products** in the sharedconfig.properties file. The default value for the property is 10,000. If the number of rows in the import file exceeds the limit, the remaining rows are not profiled.

Profiling Data

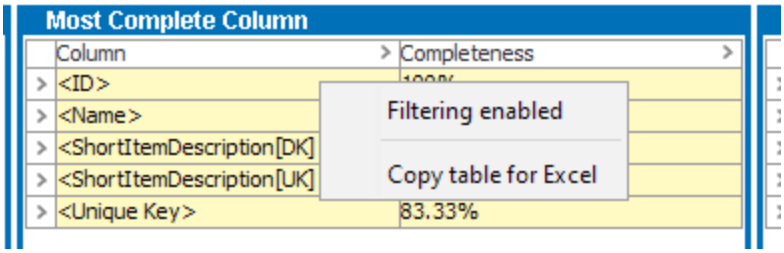
1. Map all required data on the Map Data page and click the Generate Profile button to display the Import Data profile.



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:

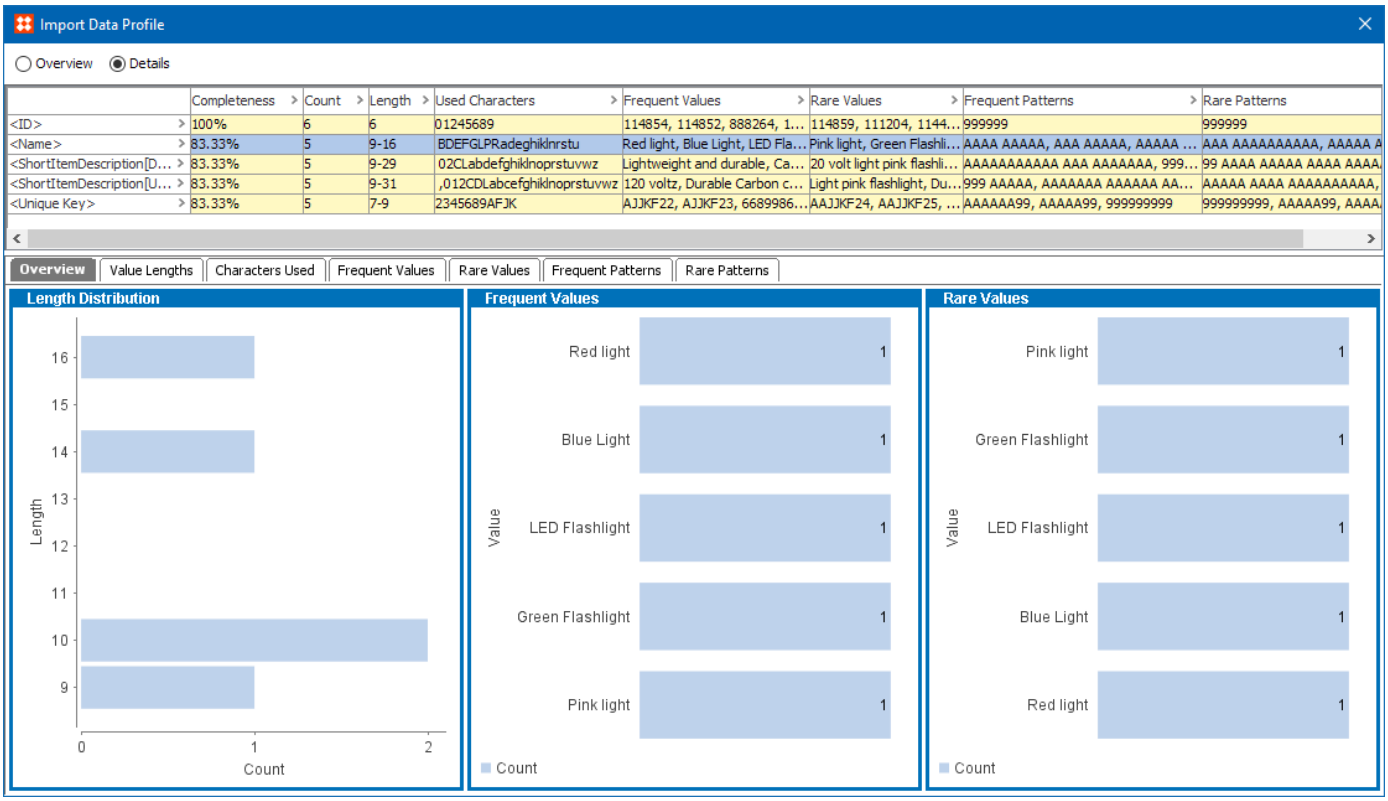


When **Filtering Enabled** is selected, you will get a dropdown list which allows you to filter the based on: Empty Values, Non Empty Values, Equals, Begins with, Ends with, Contains, '<' (lesser than), and '>' (Greater than) options which is nothing but a criteria. Once used, based on the criteria selected, 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.
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.

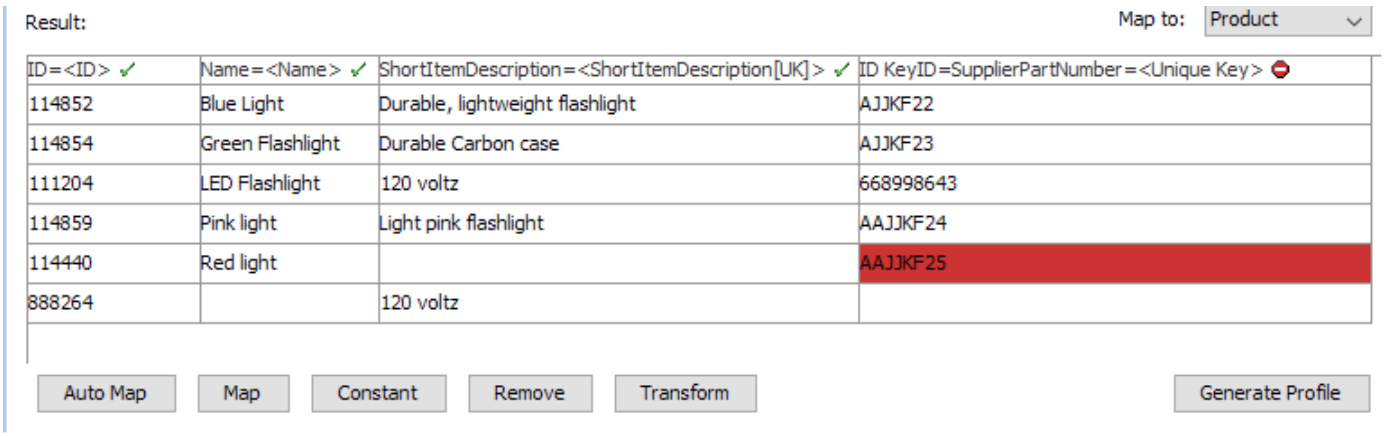
Column	Description
Used Characters	List of the characters used for values in the column
Length	Displays value lengths, for example, 3-99 or 10.
Frequent Values	Displays a comma-separated list of the most frequently appearing values. The list is sorted ascending on frequency.
Rare Values	Comma-separated 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, see the **Data Profiling** documentation.

On the Fly 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 the fly. 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.



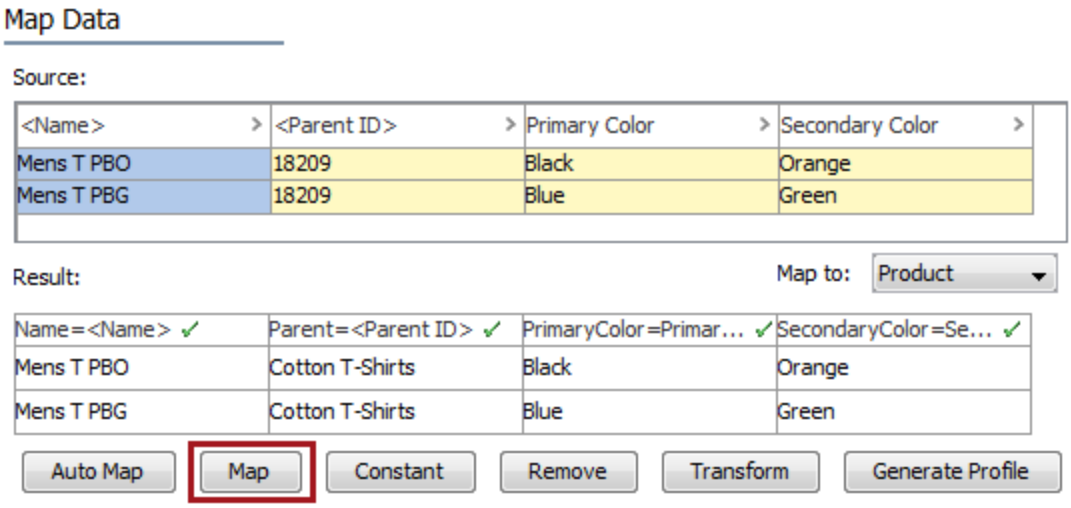
Mouse over a cell with validation errors and the hover text provides information about the problem. If you apply transformations to a result column, the validation feature is also available in the Transformation Preview. Data is re-validated with every change. See the **Inbound Map Data - Transform** 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.

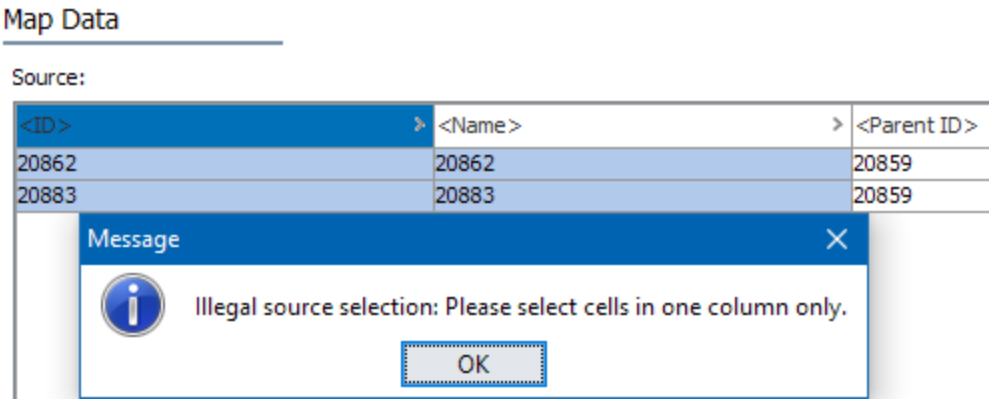
It is possible to start an import process even if validation errors have been discovered and the same errors are displayed in the background process in the execution report.

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, see the **Inbound Map Data - Auto Map** documentation.



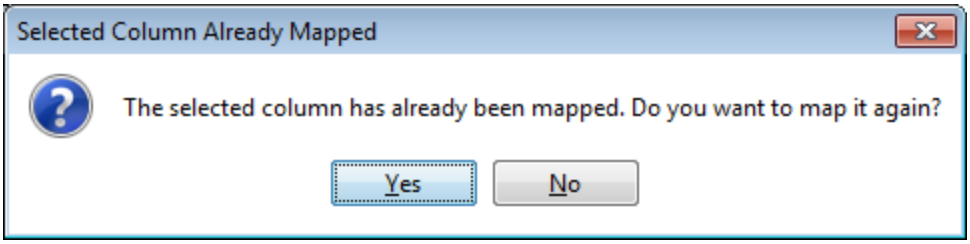
Source columns must be mapped individually. If multiple source columns are selected prior to clicking the Map button, a message is displayed saying the source selection is not valid.



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, see **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, see **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Mapping Options

These mapping options are available both in IIEP and Import Manager.

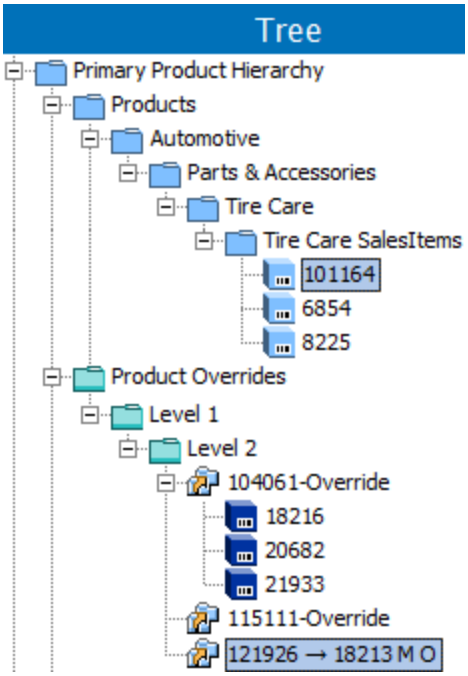
See the following topics for the specific options available for manually mapping, based on the object type selected in the **Map to** dropdown parameter.

- 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 Attribute - Map Inbound
- 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
- 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, see **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. See the **Product-Override Object Types and Commercial Object Types** topic in the **System Setup / Super User Guide** 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, see the **Product Overrides** topic in the **Getting Started / User Guide** 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.

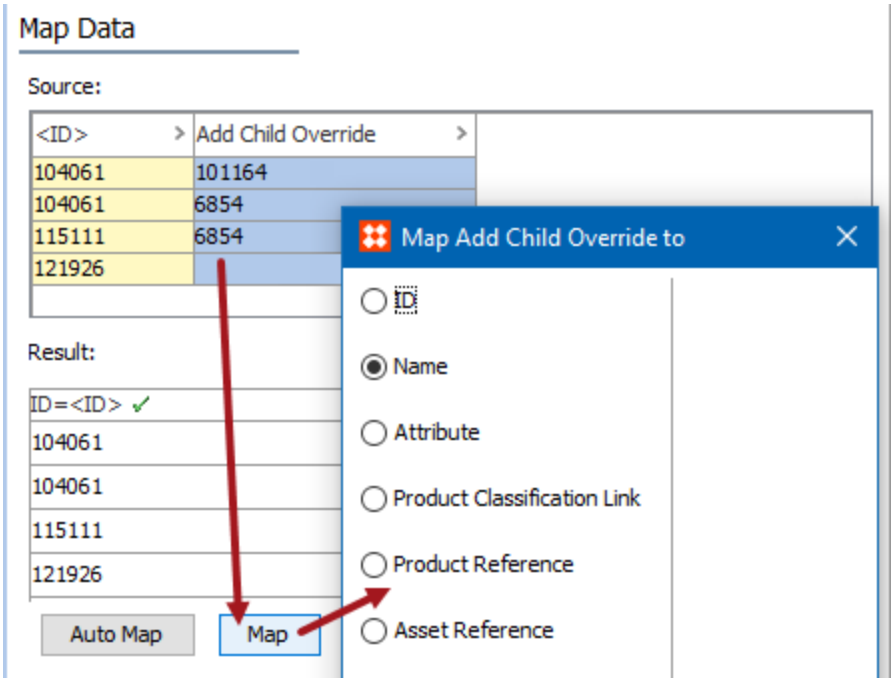
This example shows an Excel import file (although any inbound format can be used). The file adds two objects as children for one product override (ID 104061), and one of those objects is also added as a child on another product override (ID 115111). Any additional data on the final row in this file will 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	

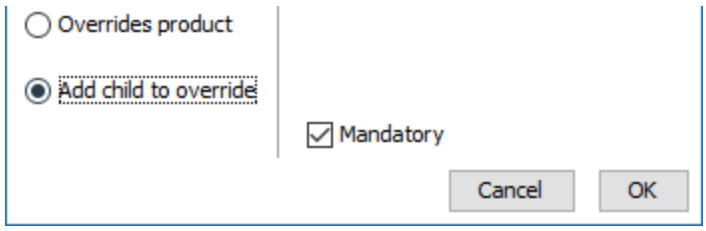
Map Add Child to Override

Once System Setup is configured, an object can be imported as a child to a product override using Import Manager or Inbound Integration Endpoint.

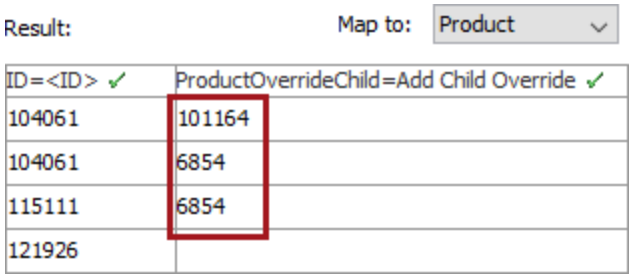
1. Use Auto Map to map the ID, or map it manually. For more information, see **ID or Key - Map Inbound**.
2. In the Source section, first select the Add Child Override column and then click the **Map** button to display the 'Map ...to' dialog. The label of the field is used as the title of this dialog.



3. On the Map ... to dialog, select the **Add child to override** radio button.



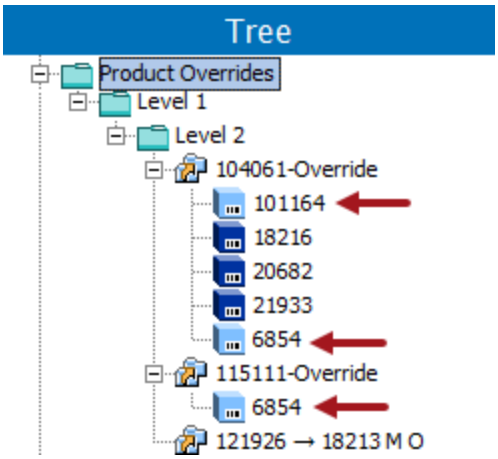
4. Check the **Mandatory** option appropriately:
   - If **checked**, imported objects must have an Add Child Override value in the file. If an Add Child Override value does not exist for an object, the object is skipped and not imported.
   - If **unchecked**, all objects are imported regardless if 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.



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, see **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 / key (not the name) of the source object for the reference (product, classification, or entity).
- The inbound file must contain the ID / 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.

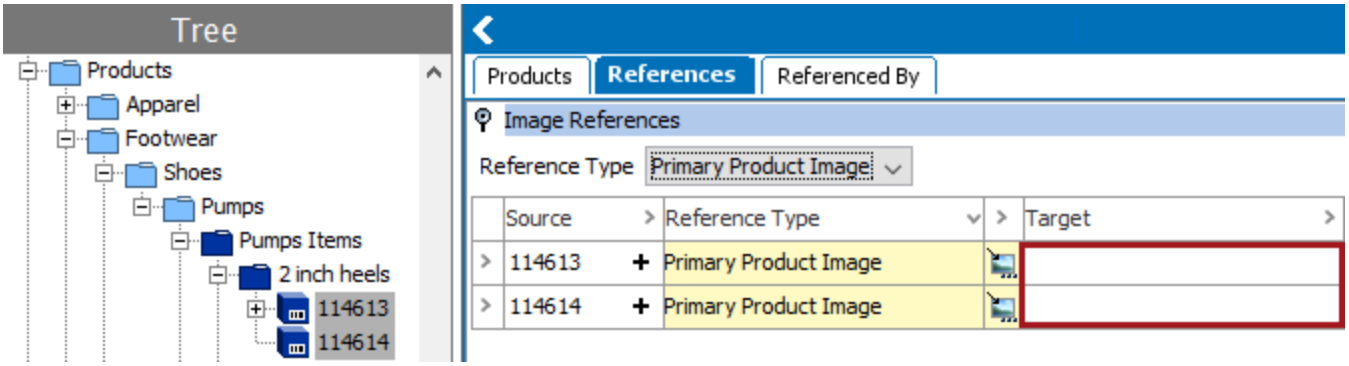
If the objects to be imported or the Target to be mapped is identified via a unique key, you must verify that the Key required is available and activated. For more information, see the **Activating and Deactivating Keys** topic of the **System Setup / Super User Guide** documentation.

Map Product to Asset Reference

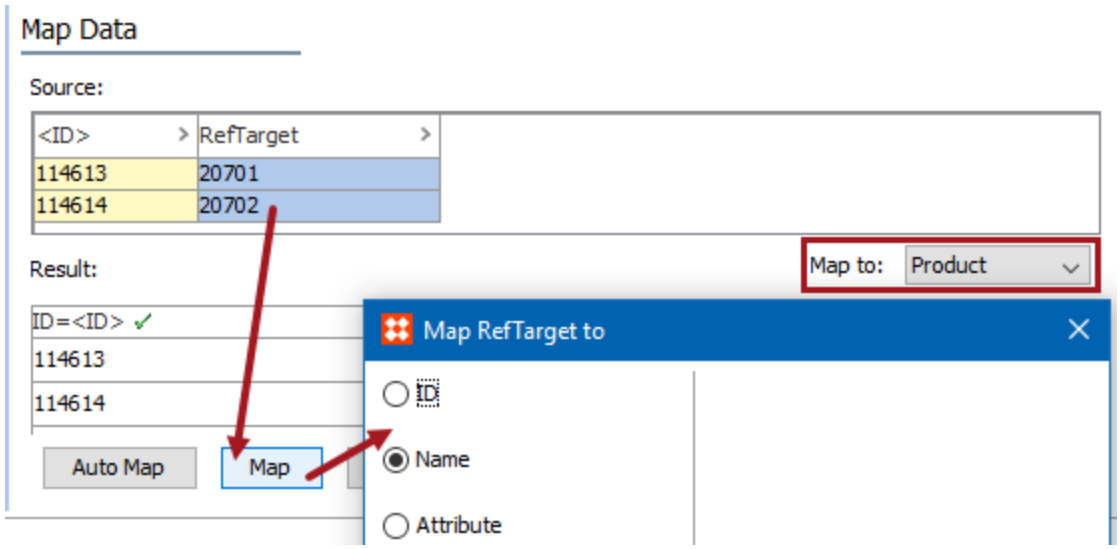
This example shows an Excel import file (although any inbound format can be used). The file includes target asset objects to be used as the primary product image for two products (indicated by the ID column).

A	B
<ID>	RefTarget
114613	20701
114614	20702

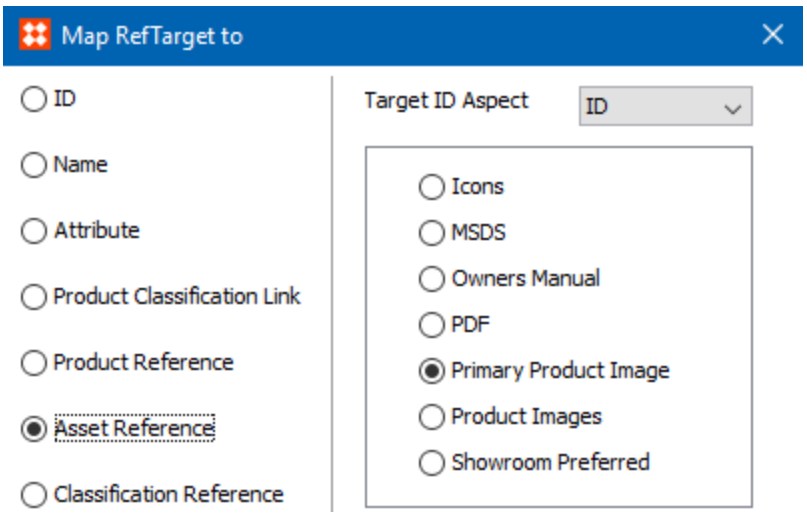
The import will specify an asset that should reference a product using the reference type 'Primary Product Image.' Before the import, there are no image references for either source product, as shown simultaneously in the image below.



1. In the Source window, select the ID or Key column and map it. For more information see 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.



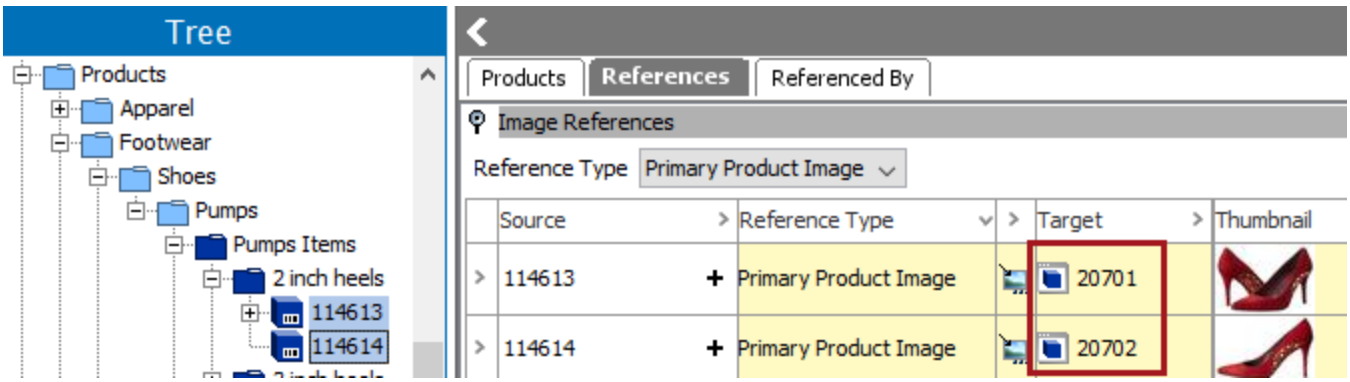
4. On the 'Map ...to' dialog, select the **Asset Reference** radio button to display the list of available reference types on your system. Select the radio button of the desired reference type. Because the Product super type was selected previously, only product-to-asset references are displayed.



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.



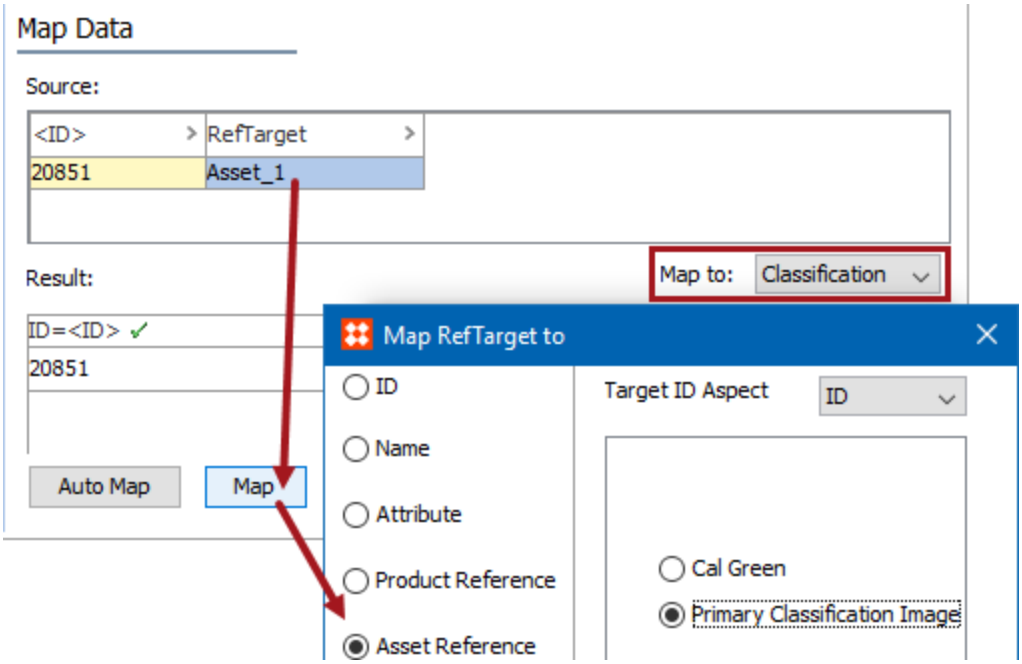
Map Classification to Asset Reference

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 / classification, verify that the object and Entity / Asset / 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.

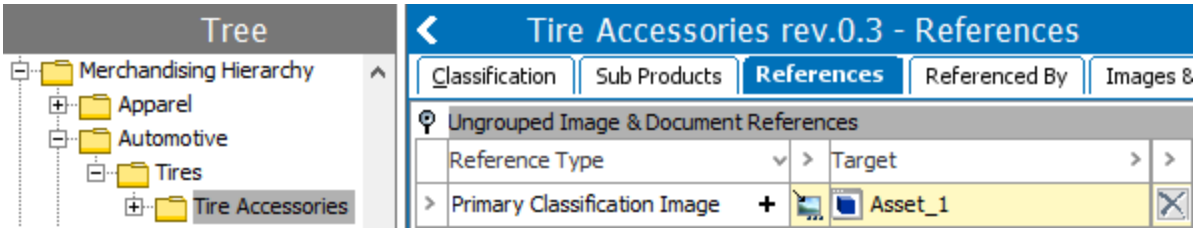
A	B
<ID>	RefTarget
20851	Asset_1

- 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.



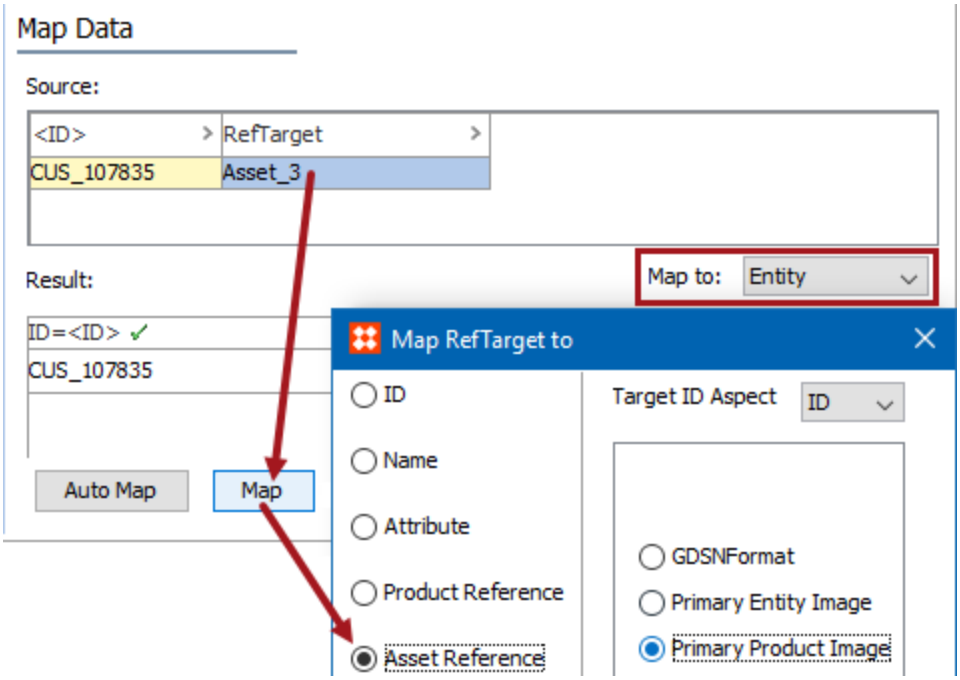
Map Entity to Asset Reference

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.

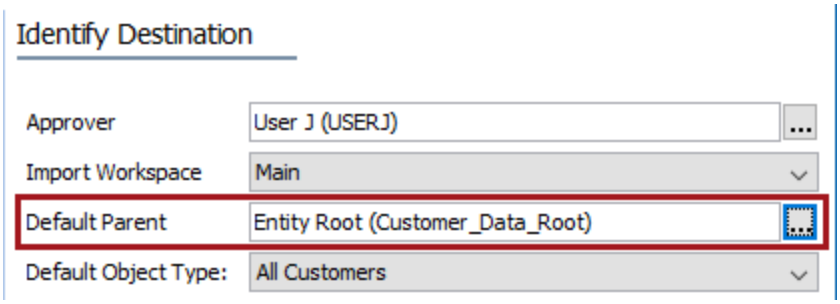


- Only entity-to-asset references are displayed after selecting the Asset Reference radio button on the **Map ... to** dialog. The Result section shows the selected entity-to-asset reference type.

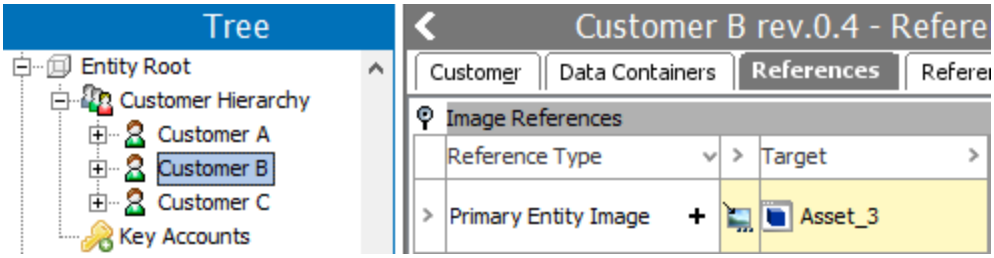
Result: Map to: Entity

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

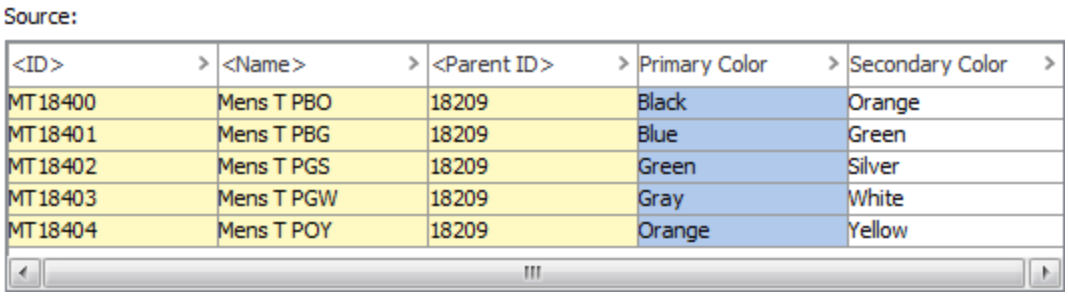
Mapping for attributes is available for both attributes that already exist in STEP and those that need to be created, as defined below.

The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, see **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

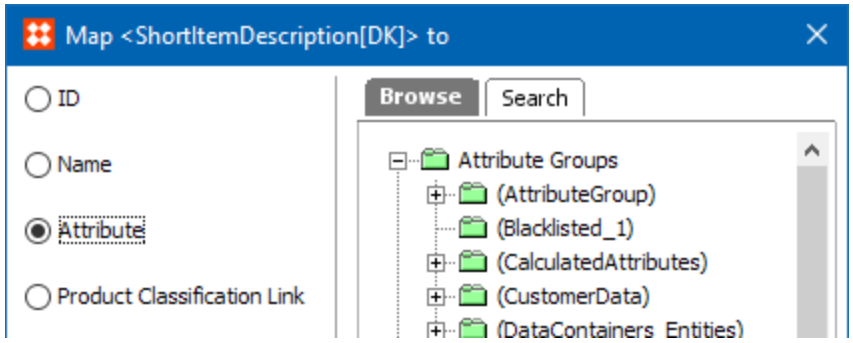
Map to an Existing Attribute

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.

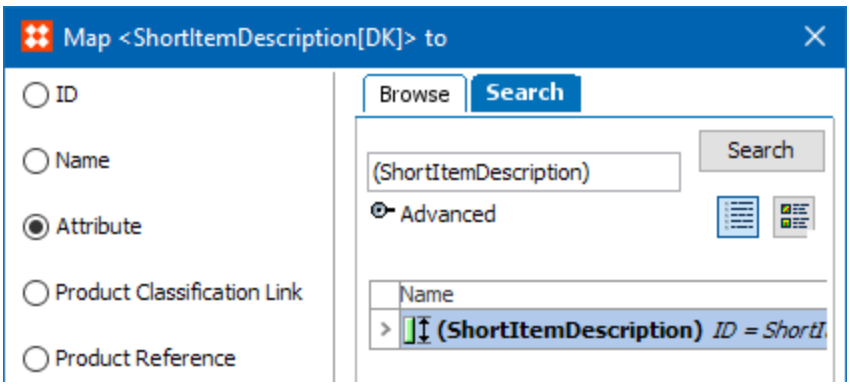
1. In the Source section, select the column of data that you want to map to an attribute and click the **Map** button.



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.

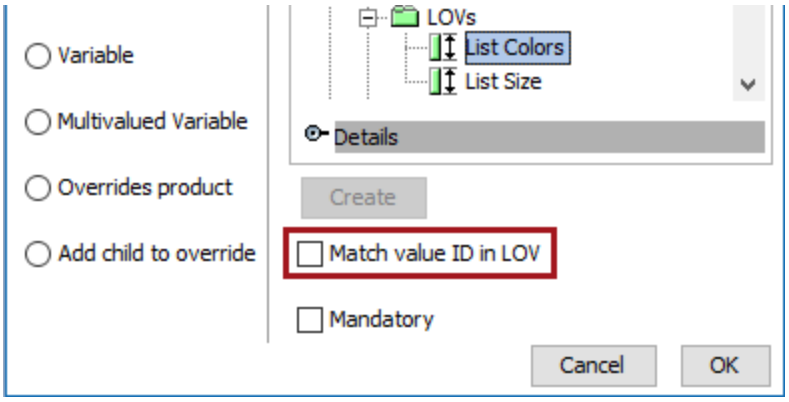


Search is available on all or part of the attribute's name or ID. Click the advanced search flipper to use other options, such as searching for attributes with a specific value.

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

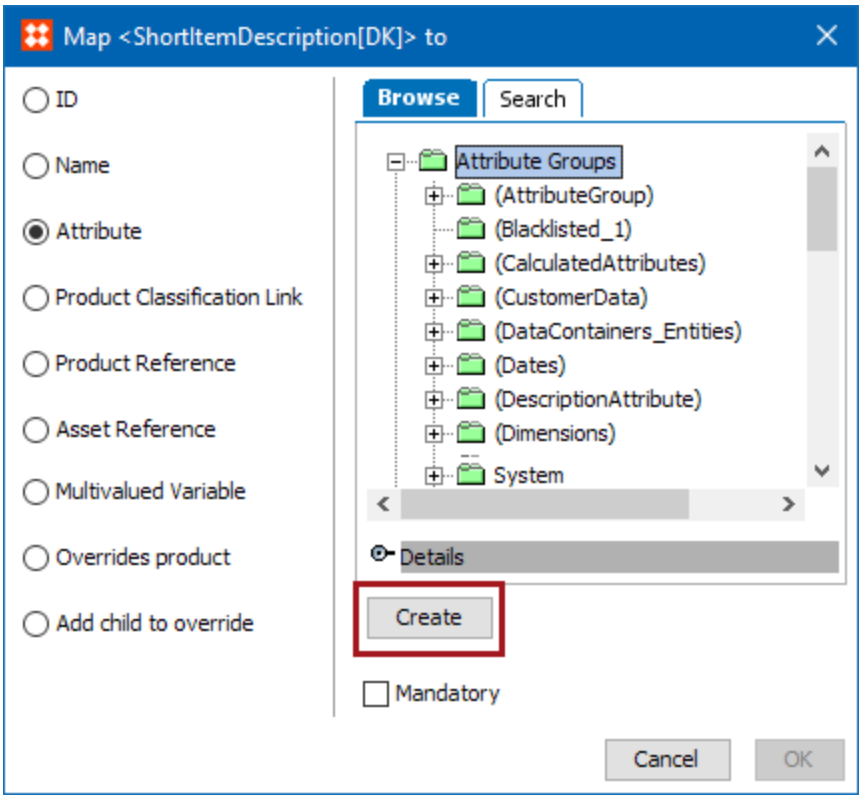
ID=<ID> ✓	Name=<Name> ✓	Parent=<Parent ID> ✓	PrimaryColor=Primary Color ✓
MT18400	Mens T PBO	Cotton T-Shirts	Black
MT18401	Mens T PBG	Cotton T-Shirts	Blue
MT18402	Mens T PGS	Cotton T-Shirts	Green
MT18403	Mens T PGW	Cotton T-Shirts	Gray
MT18404	Mens T POY	Cotton T-Shirts	Orange

7. Complete the mapping and initiate the import.

Create a New Attribute and Map

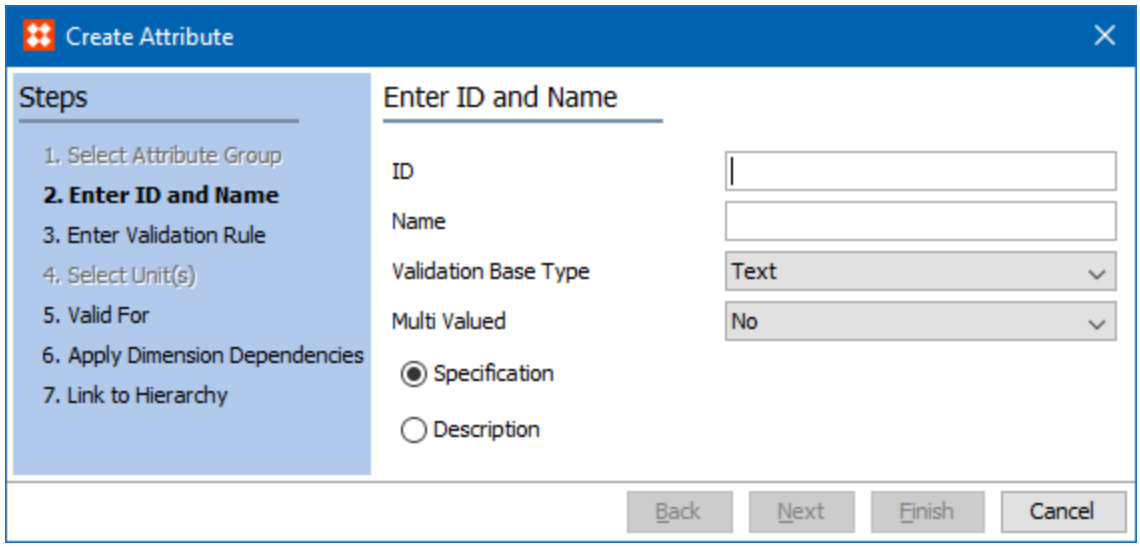
When a data column in your input file is an attribute that does not yet exist in STEP, you can create the new STEP attribute during the import process, and then map the data column to it.

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.

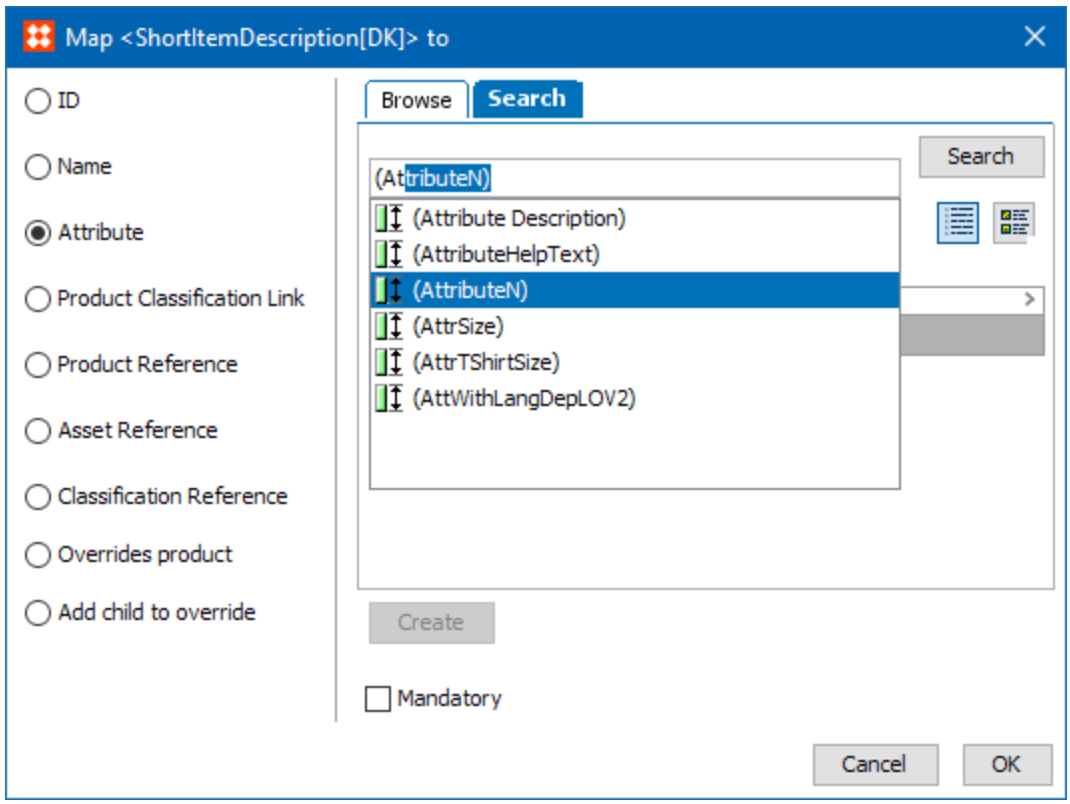


2. In the Create Attribute window, provide required information for the new attribute. For details on how to create an attribute, see the **Creating Attributes** topic in the **System Setup / Super User Guide** documentation.

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

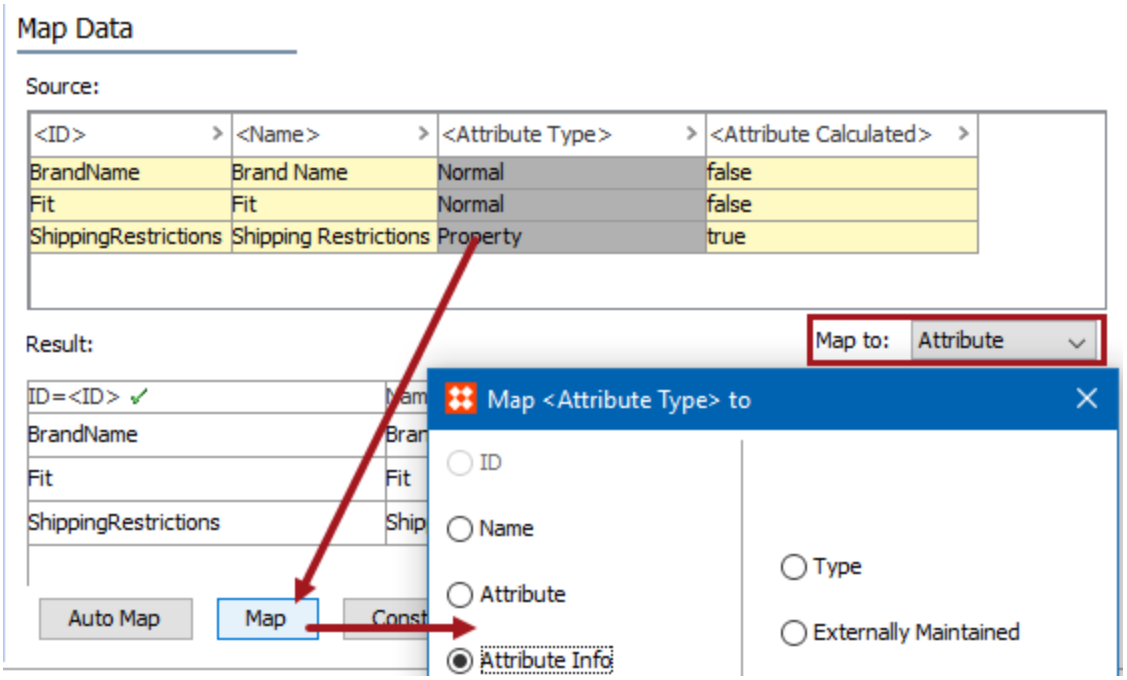
The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, see **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Verify Import File

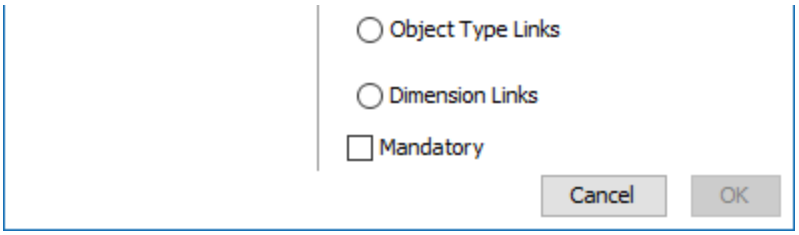
The import file must include the attribute ID or key, and must following the format defined in the **Attribute Info File Structure Requirements** section below.

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, see **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.

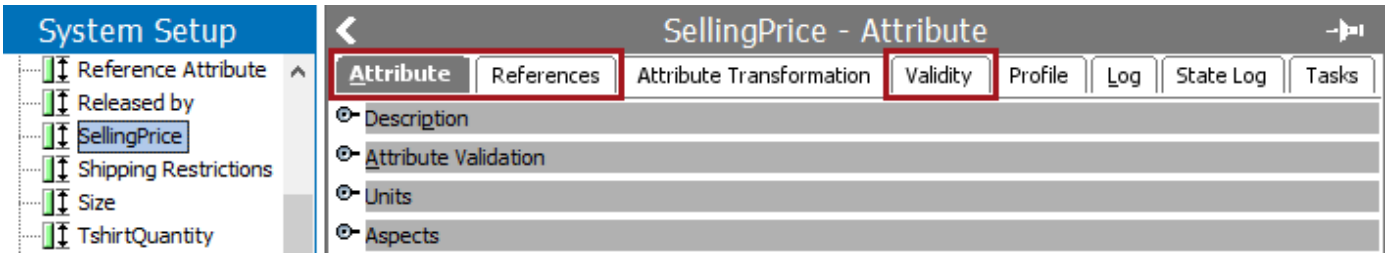
Result: Map to: Attribute

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, single update mode is required.
- **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, single update mode is required.
- **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, see 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, see the **Calculated Attributes** section of the **System Setup / Super User Guide** documentation.
- **Calculated Value Template** can only be used when the Calculated field is set to 'yes' in the workbench (see 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 more information on calculated attributes, see the **Calculated Attributes** section of the **System Setup / Super User Guide** 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, see the **Calculated Attributes** section of the **System Setup / Super User Guide** 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, single update mode is required.

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, see the **Validation Rules** section of the **System Setup / Super User Guide** documentation.

  For more information on units, see the **Units** section of the **System Setup / Super User Guide** documentation.

  For more information on regular expression, see 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	

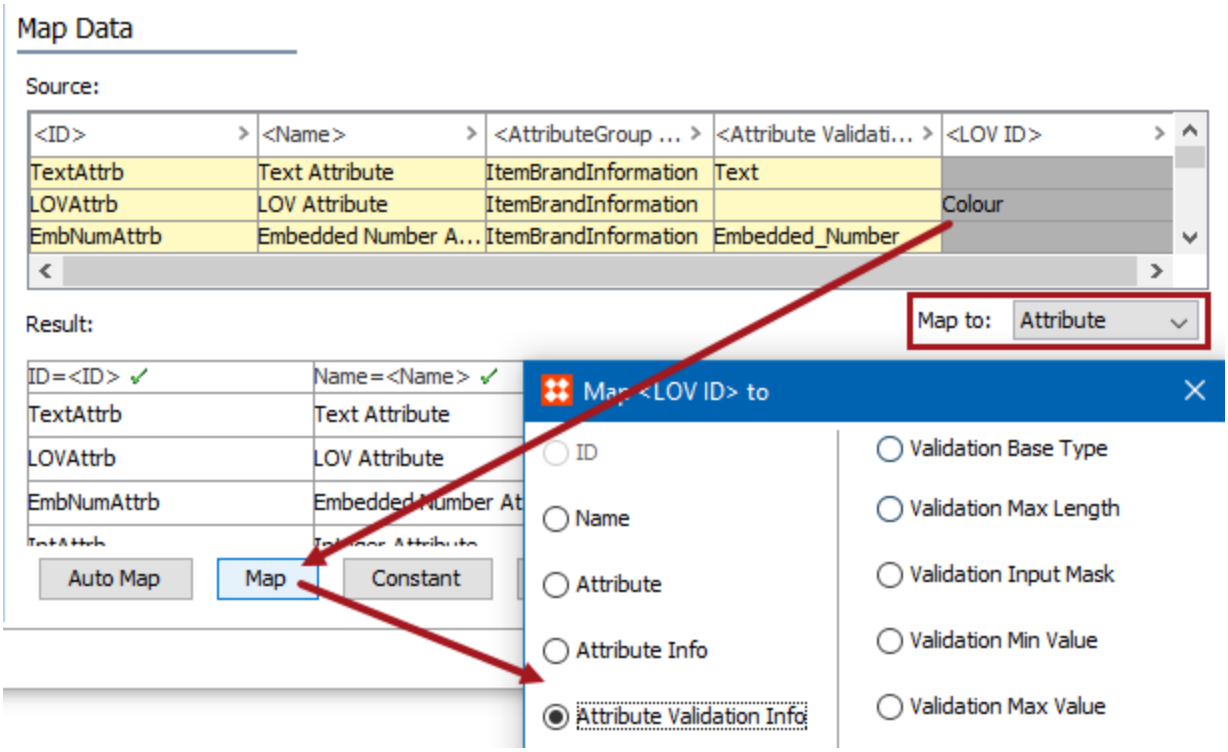
The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, see **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Verify Import File

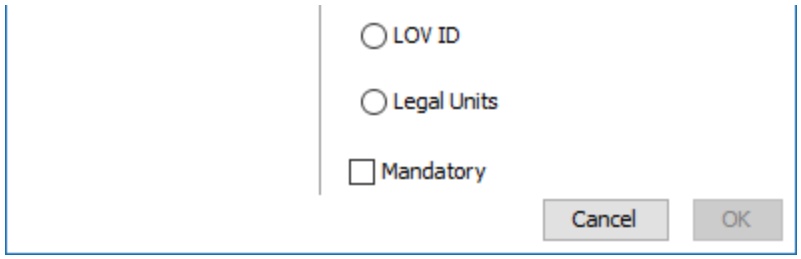
The import file must include the attribute ID, and must follow the format defined in the **Attribute Validation Info File Structure Requirements** section below.

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, see **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:

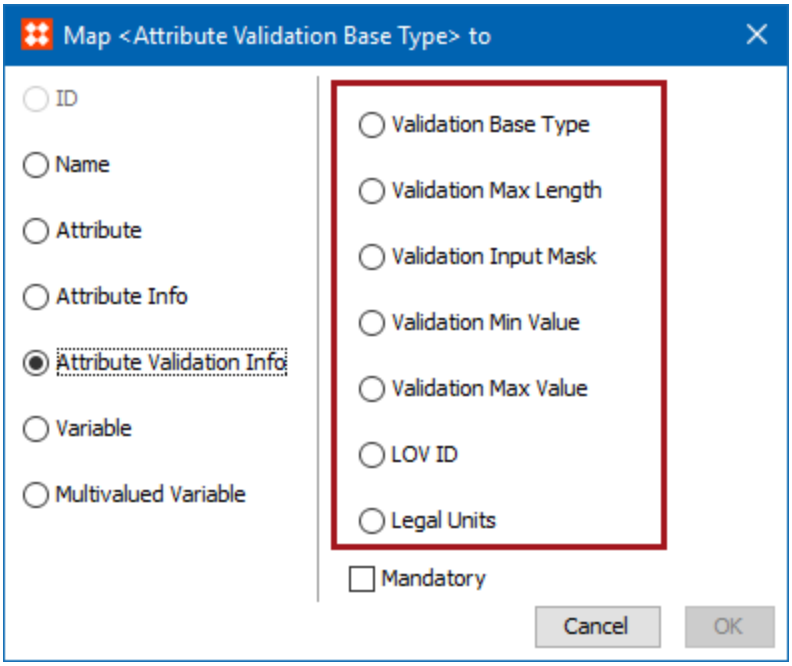
Map to: Attribute

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	

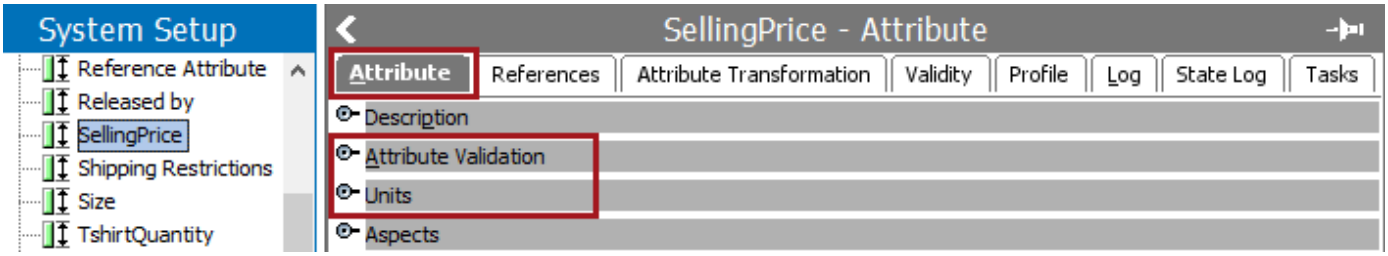
7. Complete the mapping and initiate the import.

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, see 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, see the **Validation Rules** section of the **System Setup / Super User Guide** 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, see the **Input Masks** topic in the **System Setup / Super User Guide** 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, see the **List of Value (LOVs)** topic in the **System Setup / Super User Guide** 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, see the **Units** topic in **System Setup / Super User Guide** documentation.

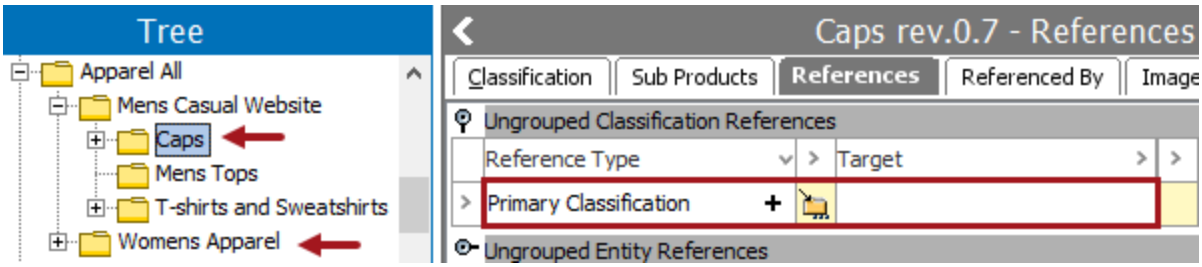
   This field is never output in any export file, and must be added manually to an input file. If more than one unit type is being imported for an attribute, then IDs should be separated with a delimiter, for example a semicolon.

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, see **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).



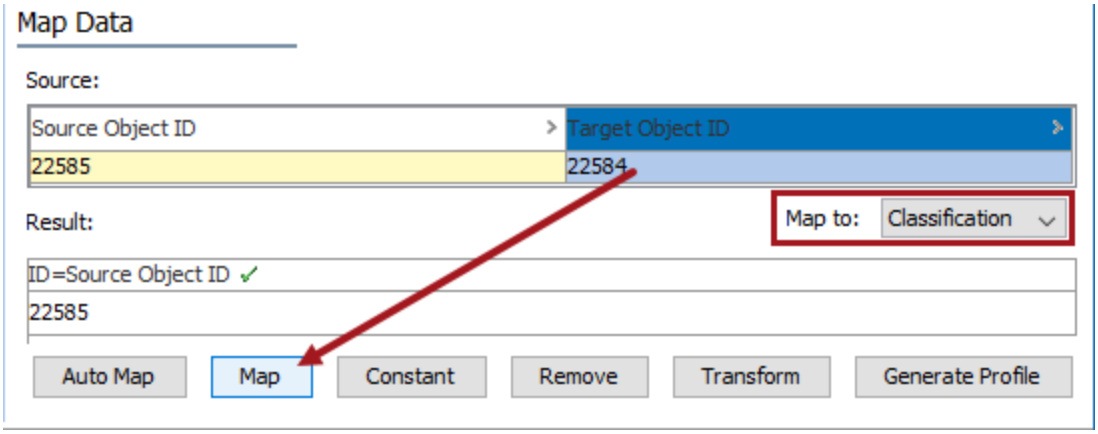
Prerequisites

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 / unique key of the source object, and the other with the ID / 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

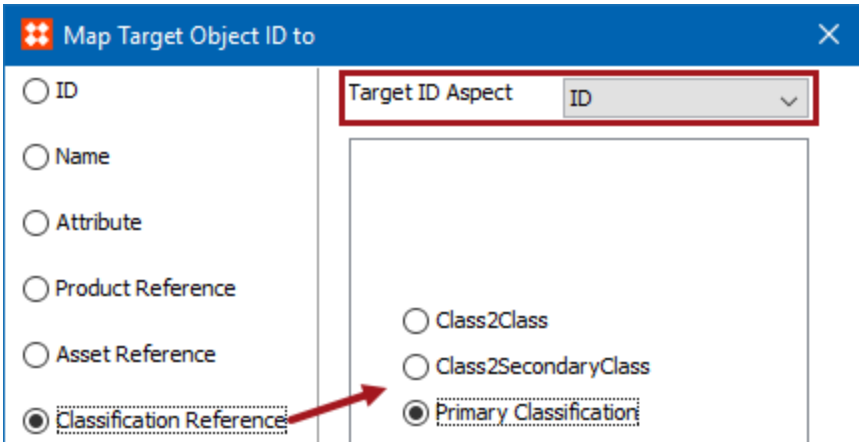
1. Below the Source section, use the **Map to** dropdown to select the **Classification** object super type.
2. Map the source ID or key column as the ID using the Auto Map button, or manually. For more information, see the **ID or Key - Map Inbound** topic.
3. In the Source section, select the target 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.

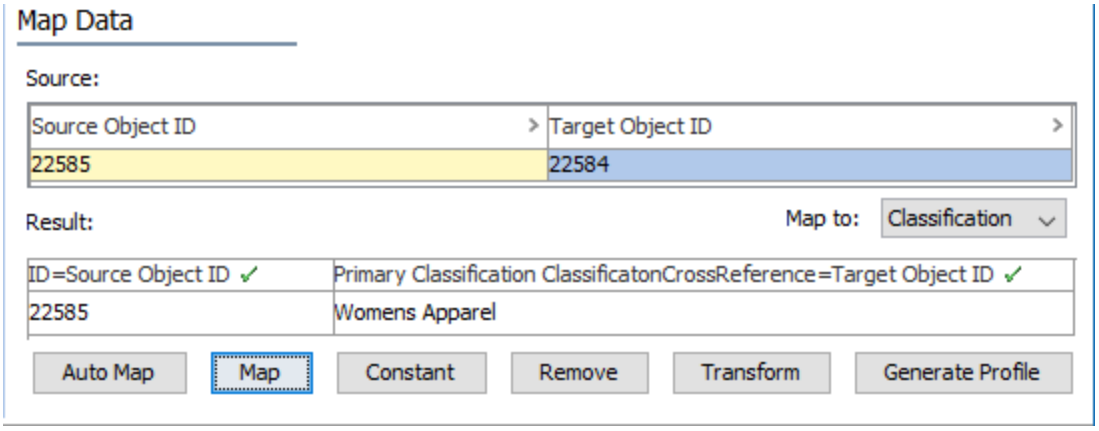
5. For the **Target ID Aspect** parameter:
   - Select **ID** when the inbound file includes the ID for the source.
   - Select the **Key** from the dropdown when the inbound file includes a key for the target.



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.

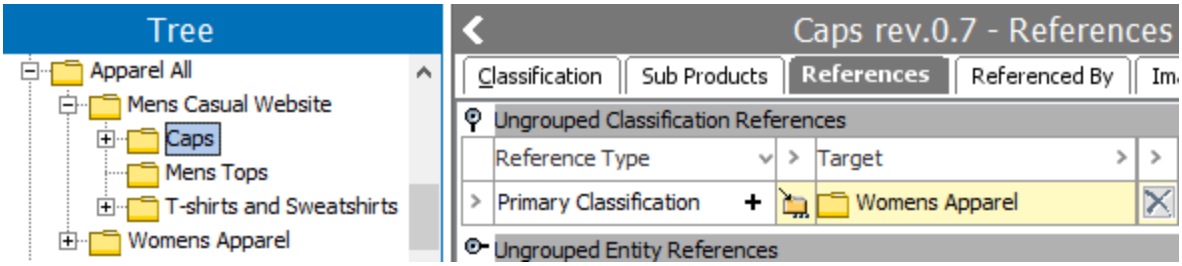
7. Click **OK** and the **Result** panel displays the name of the selected reference target object. A validated data column is marked with a green check mark as shown below.



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 Attribute - Map Inbound

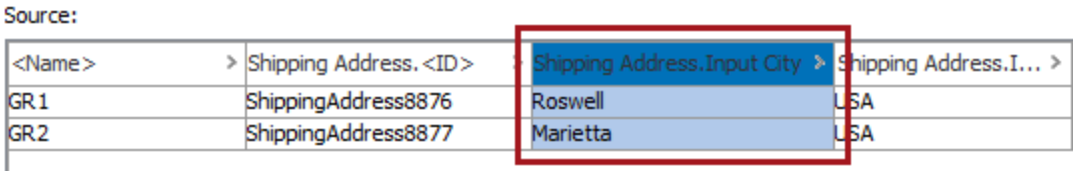
The Data Container Attribute option allows you to map data container attributes to specific (new or existing) data containers when mapping to entity or product objects.

The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, see **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

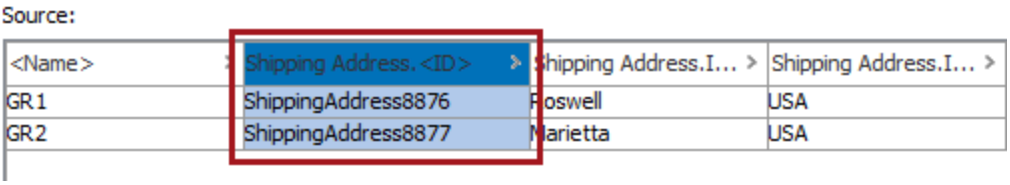
Map Data Container Attribute

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 'Shipping Address.Input City' column will be mapped to the appropriate data container type and attribute.



   Unless configured with auto-ID, a data container ID column (or variable) must also be specified in the attribute mappings. In this example, the 'Shipping Address.<ID>' column represents the data container IDs being imported.

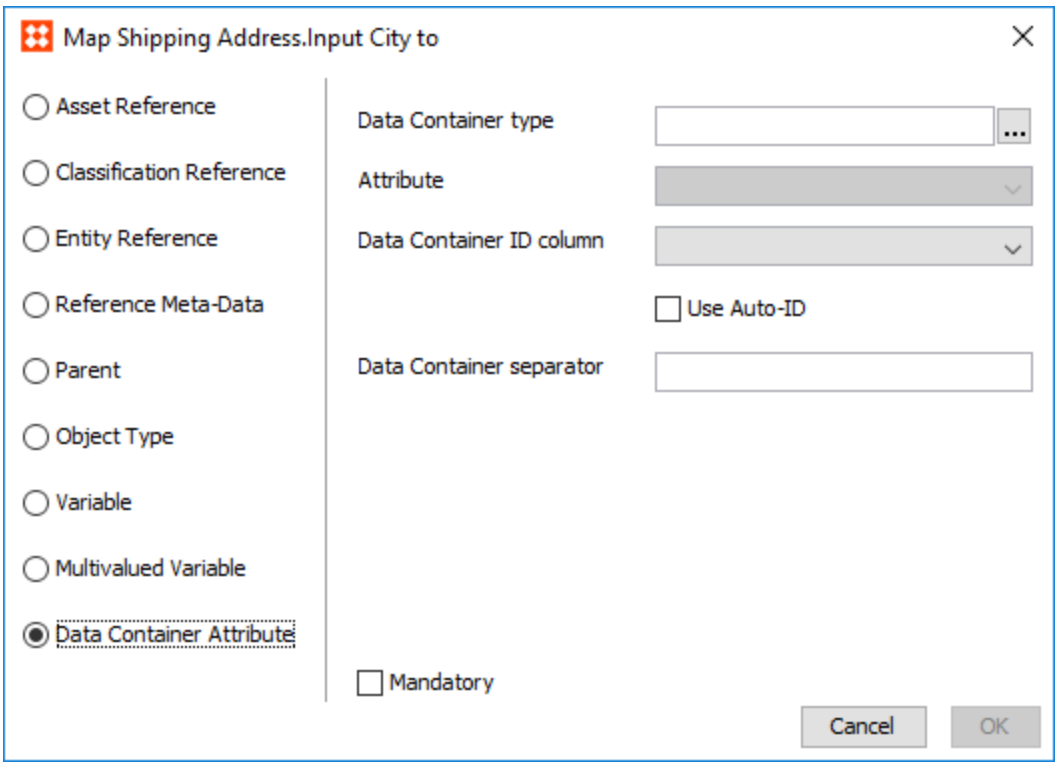


   Note that the data container type ID column should not be mapped directly, and the ID should instead be mapped to each individual attribute. Explicitly setting an ID for each data container in this way allows for subsequent imports to update the data container instances. If data container IDs are not 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.

   Instructions on mapping the ID to each attribute is described below.

   For more information on variables, see the **Variable - Map Inbound** section of the **Data Exchange** documentation.

3. In the 'Map ... to' dialog, select the **Data Container Attribute** radio button. This option is only available when the Entity or Product super type is specified in the 'Map To' dropdown.



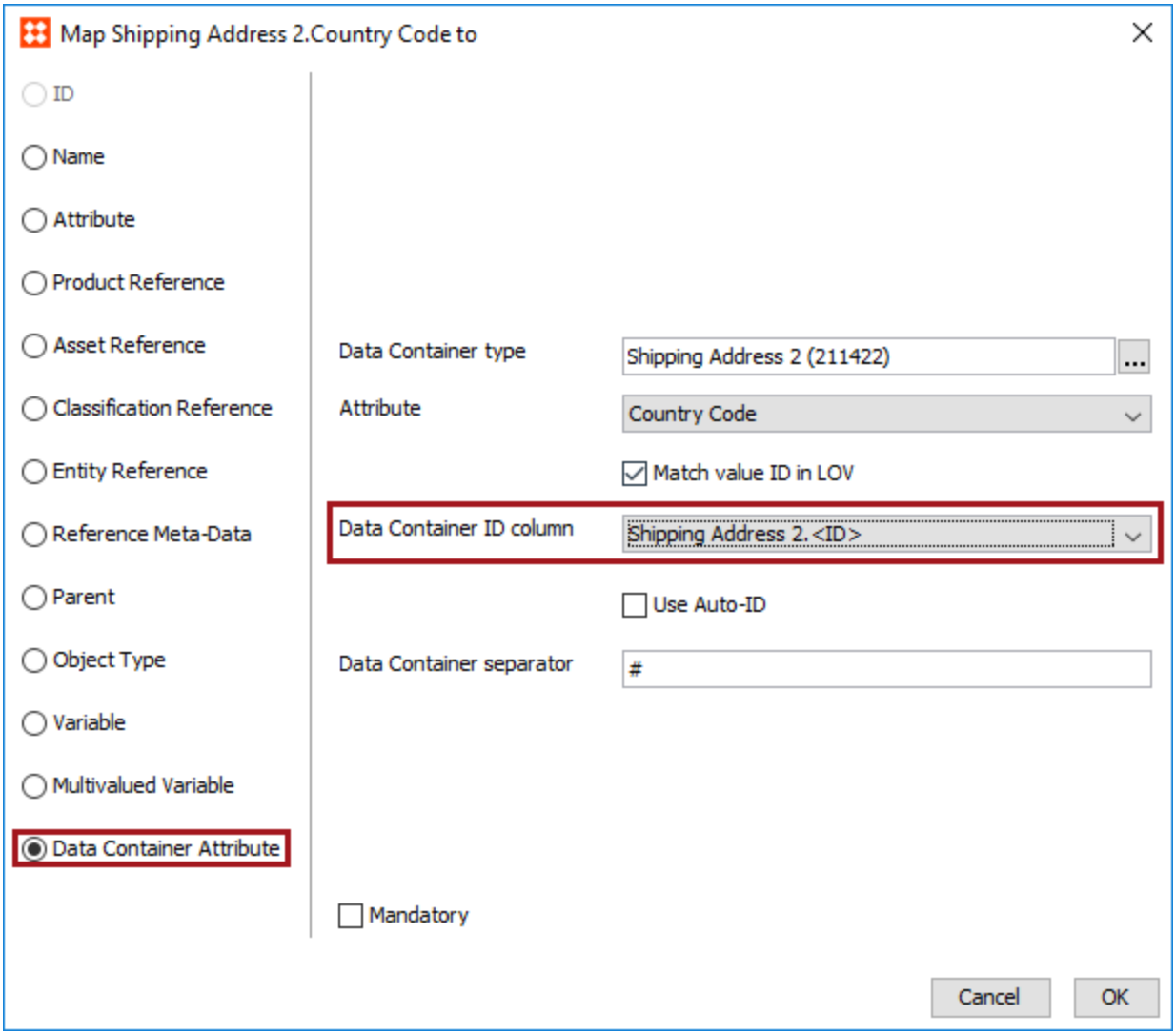
4. For **Data Container type**, Click the ellipsis button (...) and select the data container type associated with the attribute column being mapped.
5. Select the **Attribute** using the dropdown. Only attributes valid for the data container type mapped in the previous step will display.
6. The **Match 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, see the **List of Values (LOVs)** section of the **System Setup / Super User Guide** documentation.
7. 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.
8. 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.

9. 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 / updated accurately.

Note: All attributes being mapped to a specific data container type must use the same ID column or an error will display. Using the example below, Shipping Address2.Country Code (shown) and Shipping Address2.State (not shown) will share the same Data Container ID.



10. 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.
11. Click **OK** and the **Result** panel displays your selection.

Result: Map to: Entity

ID=<ID> ✓	Name=<Name> ✓	ShippingAddress.City=Shipping Address.City ✓
CUS_114563	Customer D	Kennesaw|Toronto|Paris

Delete Values - Map Inbound

When updating existing objects, use the following delete command to remove values for attributes or the name of the object that exists in STEP. 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, see the **Multivalued Variable - Map Inbound** topic.

The delete function works for both tabular and XML formats as shown below.

Importing the following tabular format import file, the primary color value would be removed if the column was mapped to the appropriate STEP object.

	A	B
1	<ID>	Primary Color
2	179915	[delete]

Importing the following XML format file, the existing value of the attribute 'ManufacturersPartNumber' would be removed, for the mapped column.

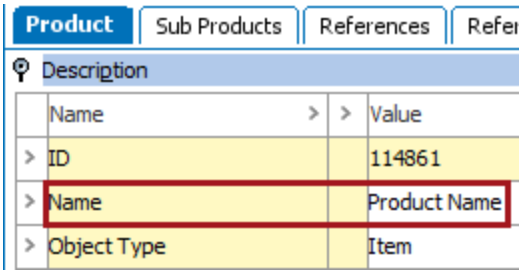
```
<Products>
    <Product ID="20883" UserTypeID="Item" ParentID="20859">
      <Name>20883</Name>
      <Value AttributeID="ManufacturersPartNumber">[delete]</Value>
    </Product>
</Products>
```

Map to Delete a Value

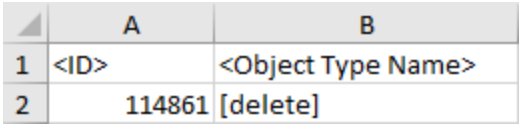
1. In the inbound file, add the **[delete]** command for the cells where existing data should be deleted.
2. Map the inbound data following the steps required for the type of data. Details for mapping each object type are included in the Inbound **Map Data - Map** topic.
3. Initiate the import.

Example

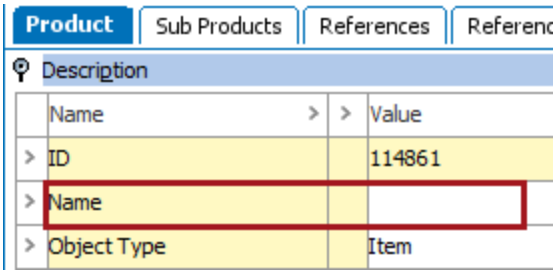
Consider the 'Name' attribute value on a product.



The following import file that is configured to delete the 'Name' attribute value on the product.



After the successful import, the 'Name ' attribute value is blank.



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, see **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Prerequisites

The inbound file must include:

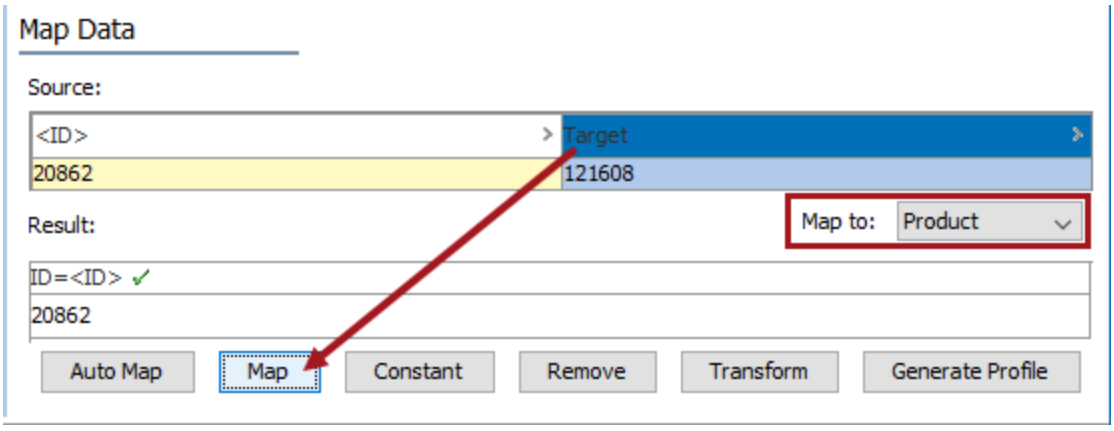
1. The ID / key (not the name) of the reference source: product, classification, or entity.
2. The ID / key (not the name) of the reference target: entity.
3. Objects that already exist in STEP.

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, see 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.

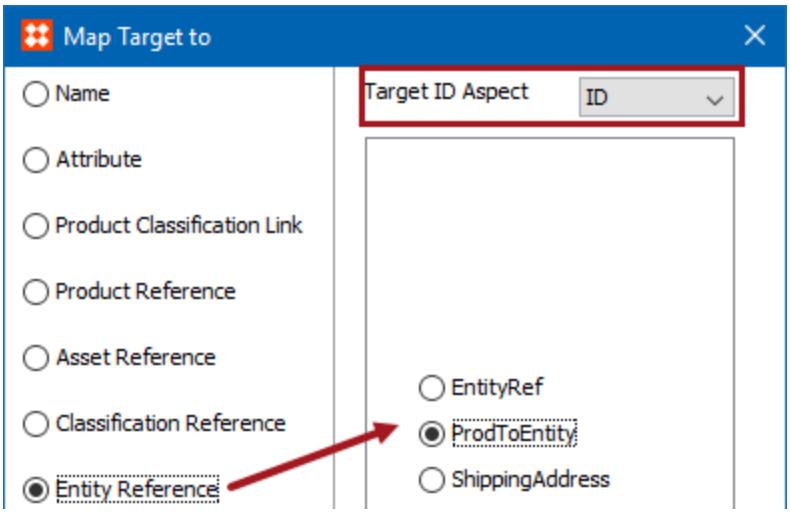
3. In the Source section, select the target column of data and click the **Map** button.



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.

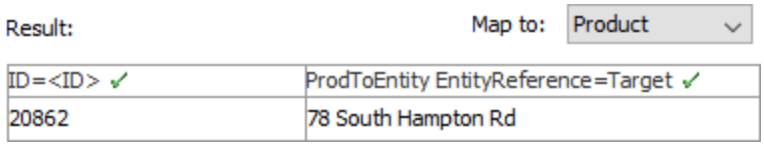
5. For the **Target ID Aspect** parameter:
   - Select **ID** when the inbound file includes the ID for the target.
   - Select the **Key** from the dropdown when the inbound file includes a key for the target.



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.



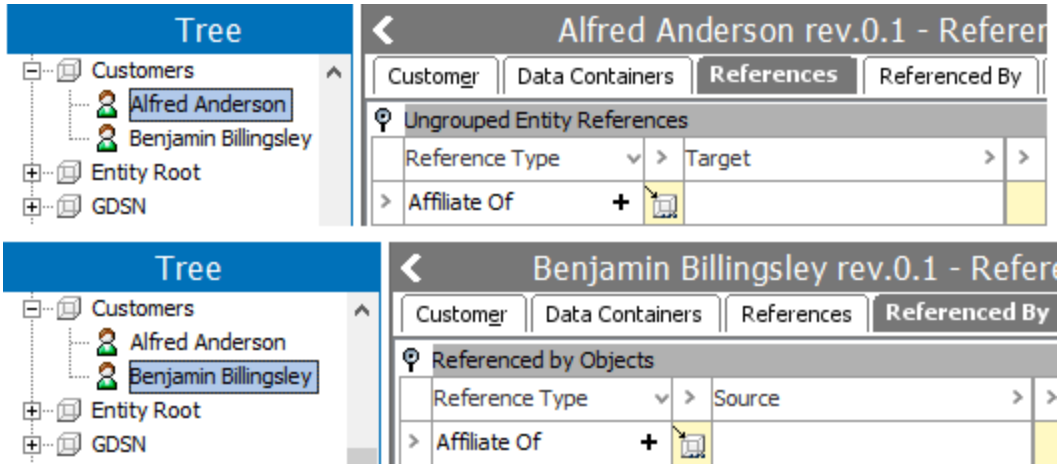
8. Complete any additional mapping.
9. If the Parent ID is not included in the inbound file, set the **Default Parent** parameter on the **Identify Destination** step. The parameter is the same for IIEP and Import Manager as defined in the **Import Manager - Identify Destination** topic.
10. Initiate the import.

Entity-to-Entity Reference Example

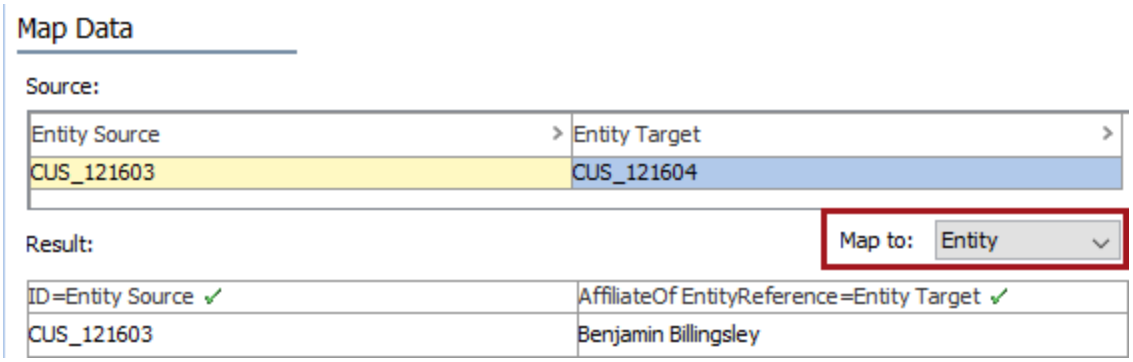
This sample file will be used to create an Affiliate Of reference between two entity objects holding customer data.

	A	B
1	Entity Source	Entity Target
2	CUS_121603	CUS_121604

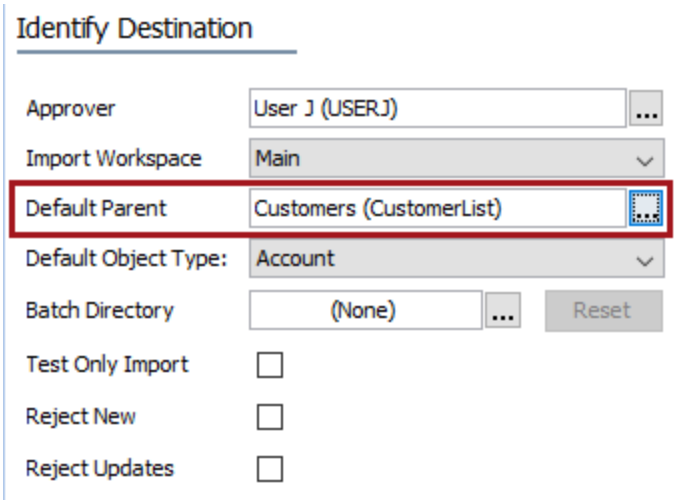
Before the import, the source entity has no Affiliate Of references.



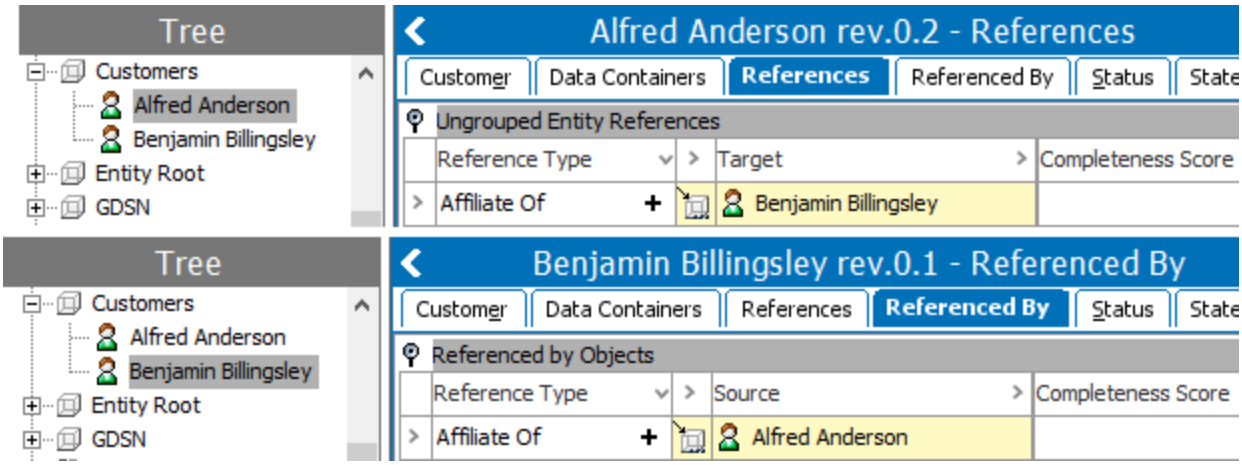
Select the Entity object super type before mapping the source and target entities.



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.



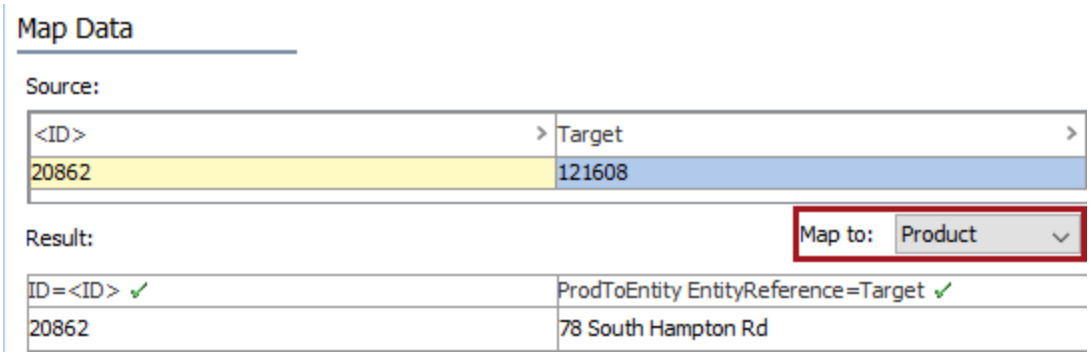
Entity-to-entity references can also be defined by the target entity's Source Record ID. For more information on mapping entity-to-entity references in this way, see the **Entity Reference via Source Record ID - Map Inbound** section of the **Data Exchange** documentation.

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.



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, see 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, see **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:

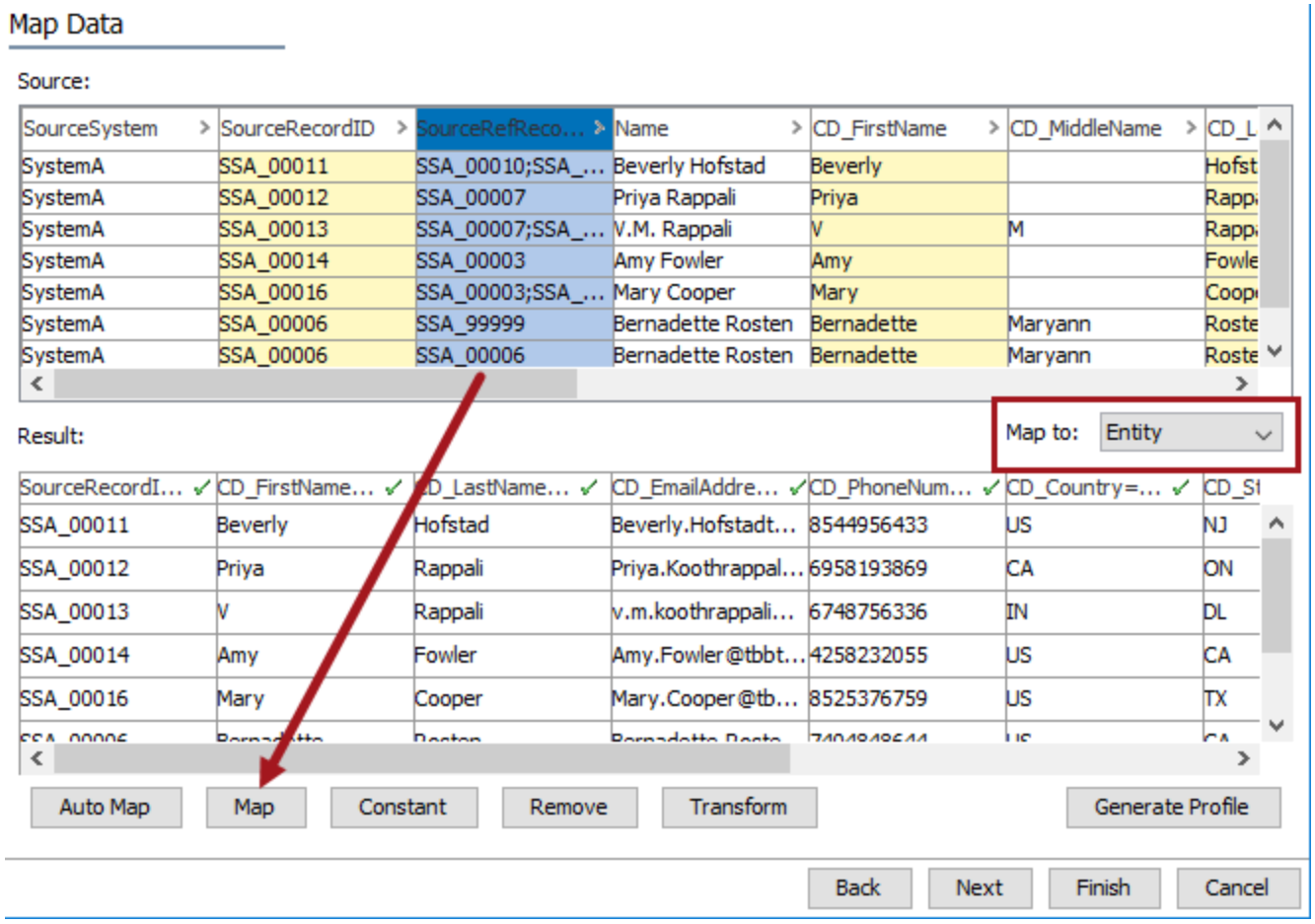
1. The ID / key (not the name) of the reference source: entity.
2. The Source Record ID key (not the name) of the reference target: entity.

Map Entity References

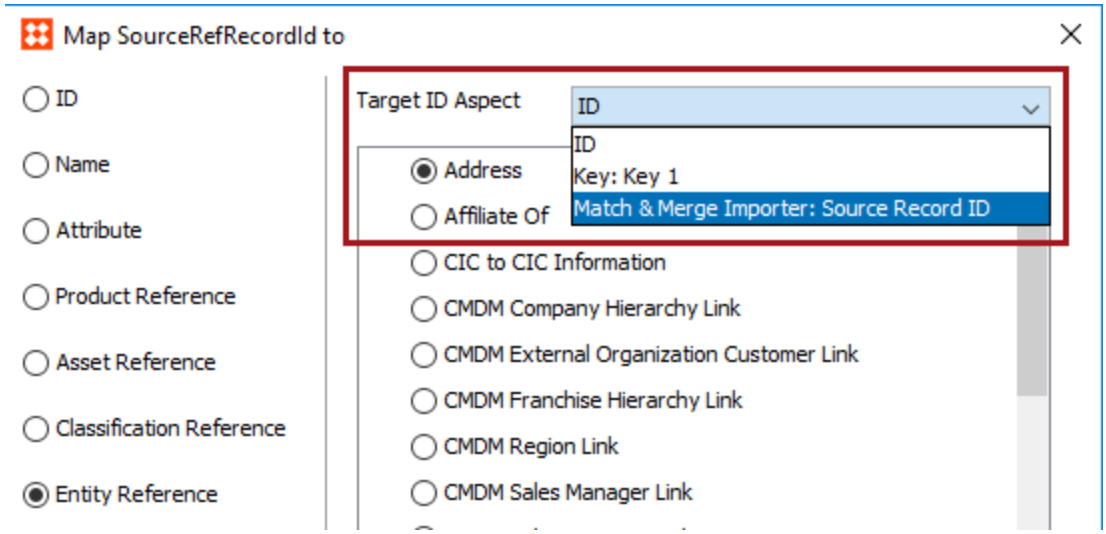
1. Below the Source section, use the **Map to** dropdown to select the object super type defined by the source of the reference type. For example, on an entity-to-entity, the source is entity.
2. Map the source ID or key column using the **Auto Map** button, or do so manually. For more information, see 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.

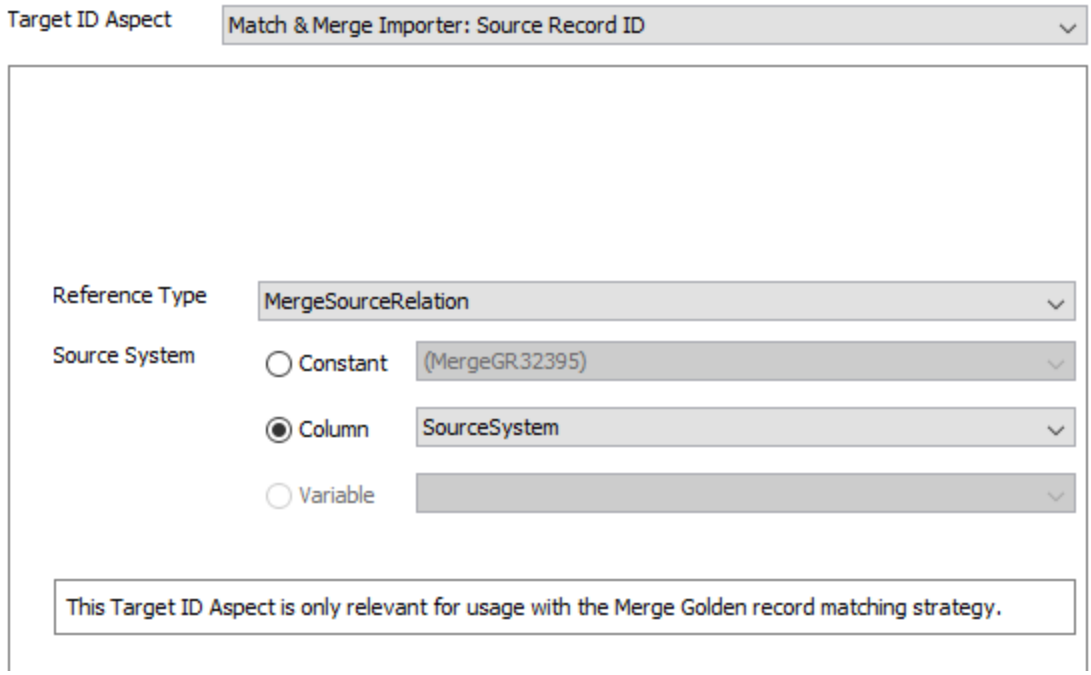
3. In the Source section, select the target column of data and click the **Map** button.



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.

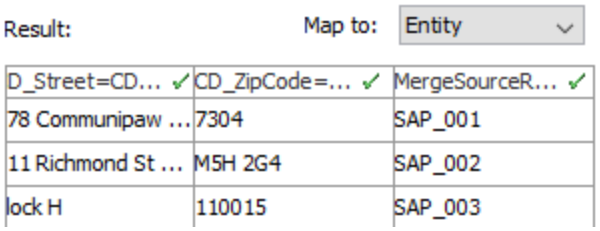


5. For the **Reference Type** parameter, choose the relevant Reference Type.
6. For the **Source System** parameter, choose the relevant Source System:
   - 'Constant' provides a constant value that can be chosen. The values include all the source system found in the system.
   - 'Column' lists the set of columns in the Excel / 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, see **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

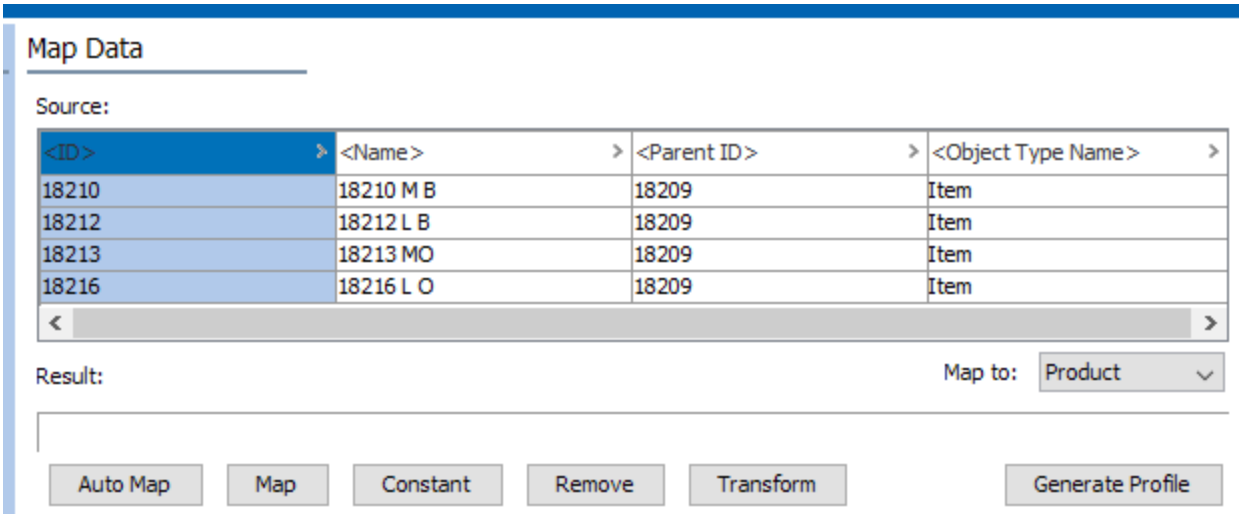
Mapping ID

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.

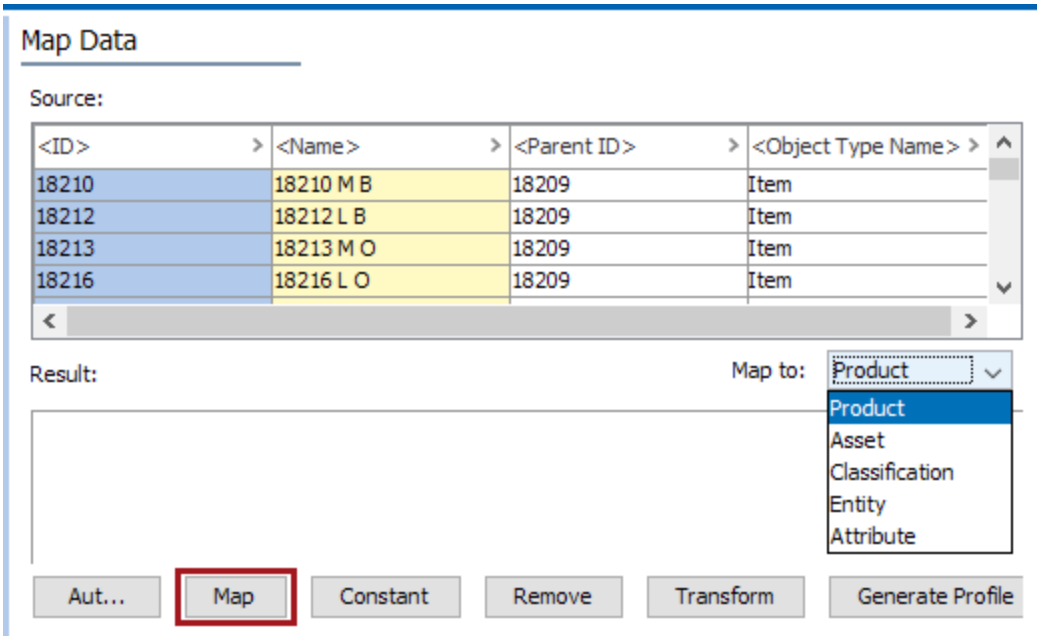
When updating existing products, you must specify the object ID as it exists in STEP.

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.

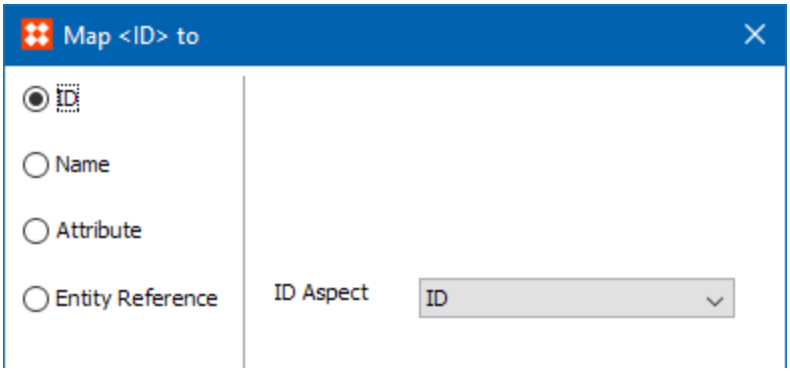
1. In the Source section, select the column that contains the object ID.



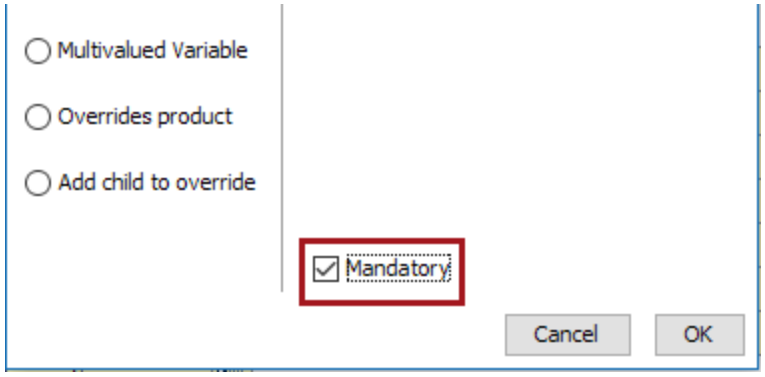
2. From the **Map to** dropdown, choose the option for the type of data being imported, and then click the **Map** button. In this example, products are being imported.

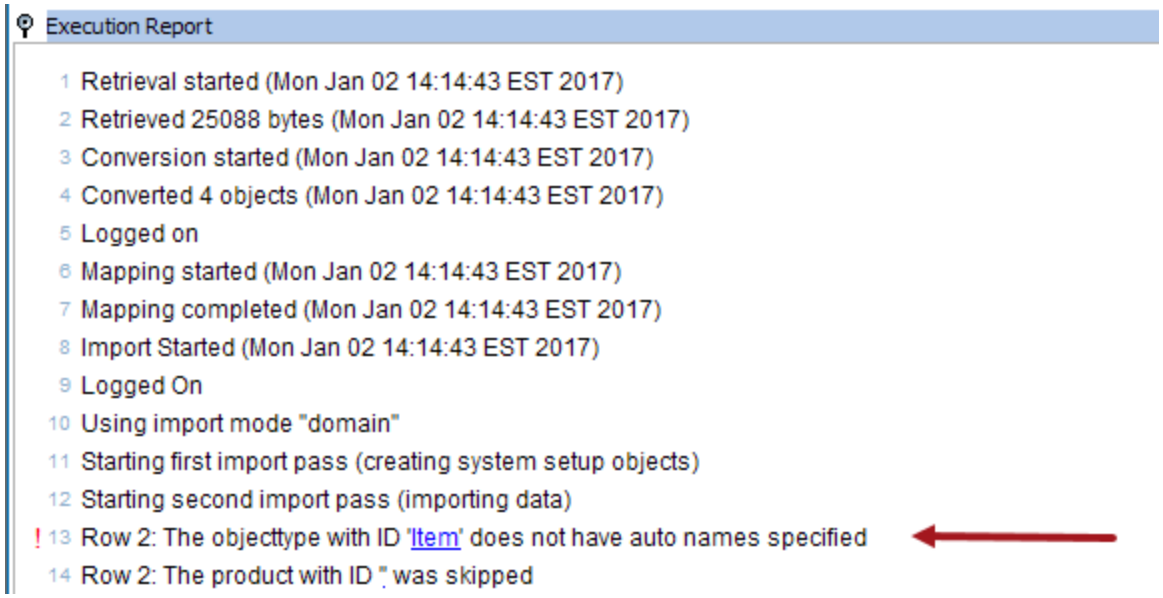


3. In the 'Map ... to' dialog, select **ID** radio button, and select the **ID** in the 'ID Aspect' dropdown list.

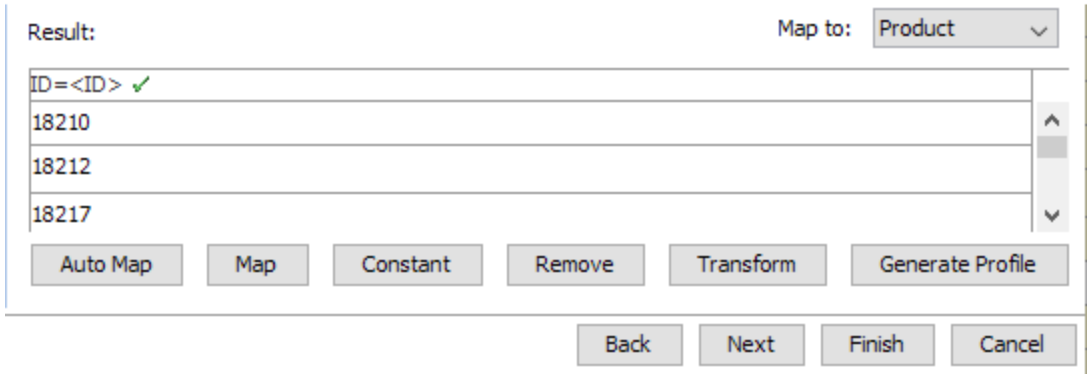


4. Determine if the 'Mandatory' option needs to be checked and do so accordingly:
   - **Check** the 'Mandatory' option only if each object being imported has an ID. If they do not all have IDs, then the object will be skipped, will not display in the Results field, and will not be imported.
   - **Unchecked** 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.



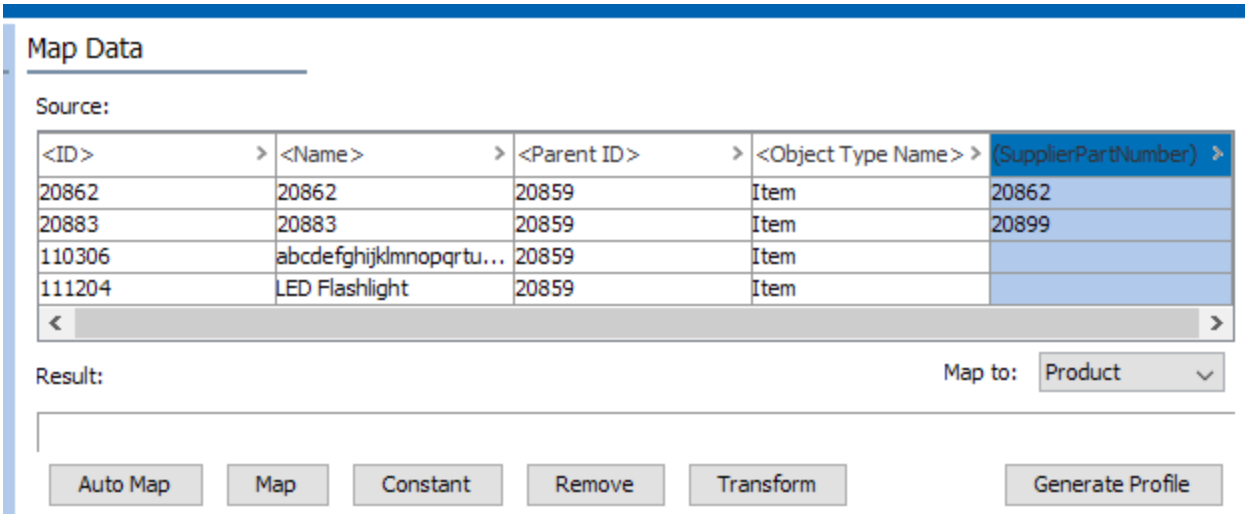
6. Complete the mapping and initiate the import.

Mapping Key

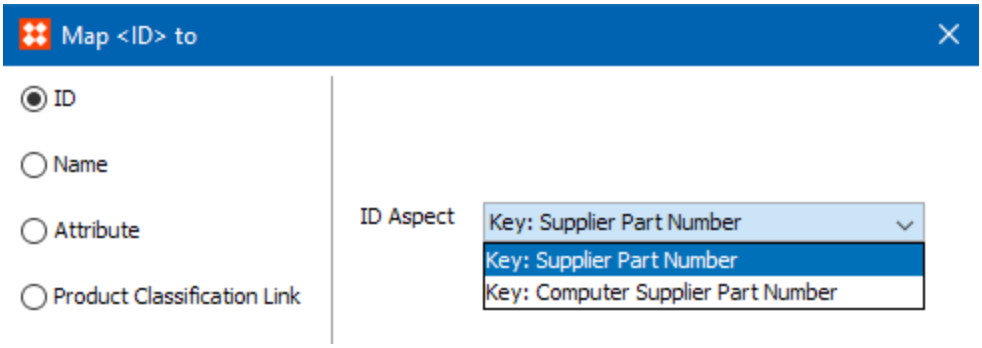
Objects, parent objects, and referenced objects can also be identified using keys. Keys are automatically generated when the underlying attributes are mapped during import.

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, see the **Unique Keys** section of the **System Setup / Super User Guide** 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.



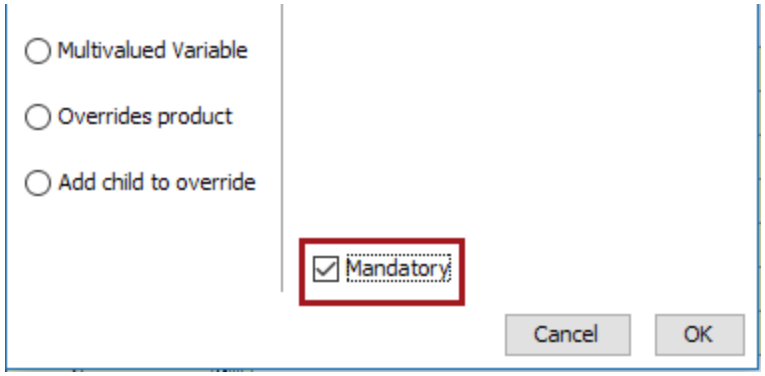
3. In the 'Map ... to' dialog, select the ID radio button option, and then select the respective key in the ID Aspect dropdown list. This indicates the ID of the current object is represented by the specified key.



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 / usage outside of being a way to transform a multivalued attribute, asset, classification, entity, or product during import.

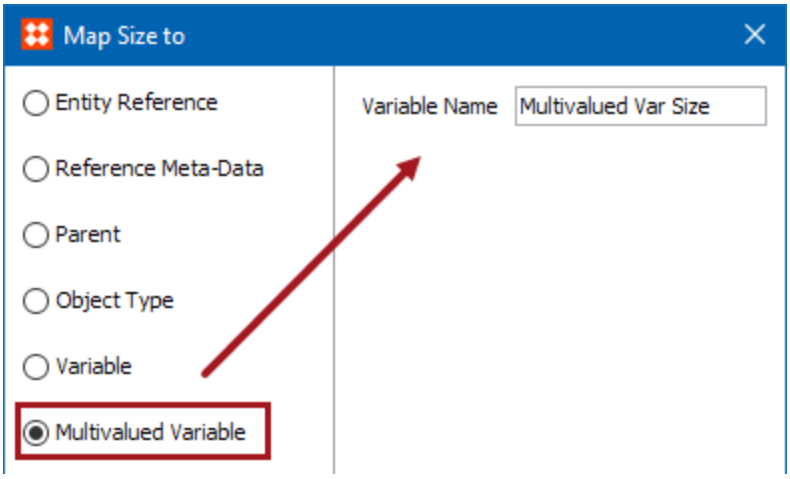
To map using a multivalued variable, follow the steps below. Note that the below example shows an Excel file being imported, though all standard inbound formats are accepted:

	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, see **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Map Multivalued Variable

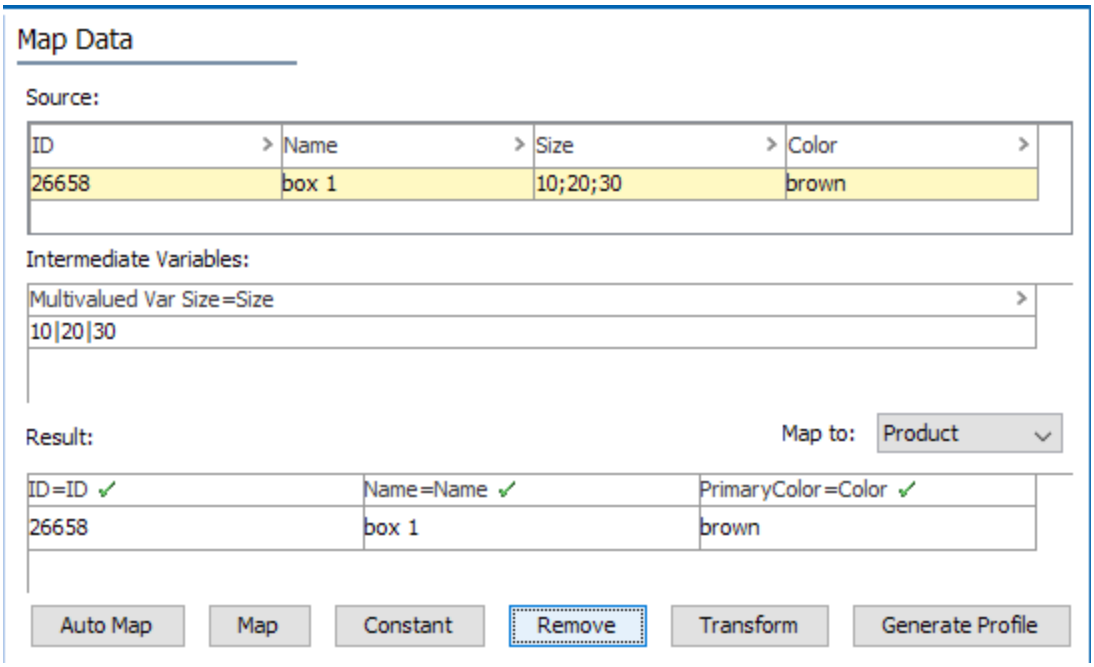
1. Use Auto Map to map the ID, or map it manually. For more information, see **ID or Key - Map Inbound**.
2. Map all other data required for import. For more information, see **Inbound Map Data - Map**.
3. Map the column that holds data that needs transformations on each value to the Multivalued Variable option, and type the name in the Variable Name field.



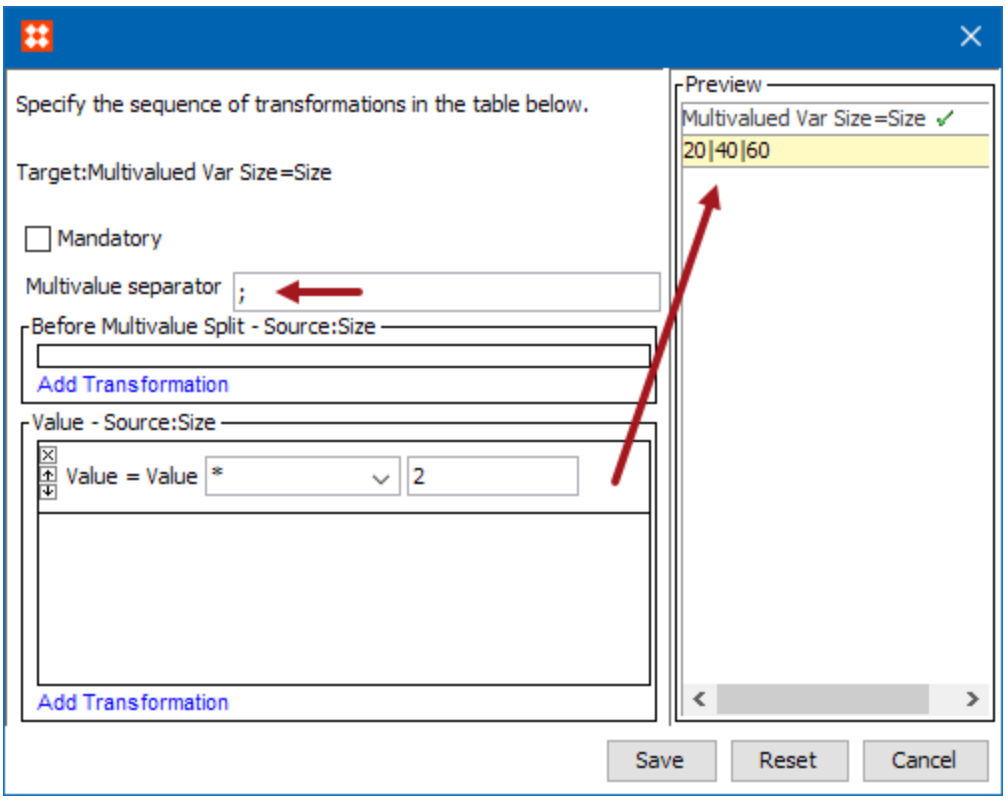
4. Check the **Mandatory** option appropriately:

- 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.

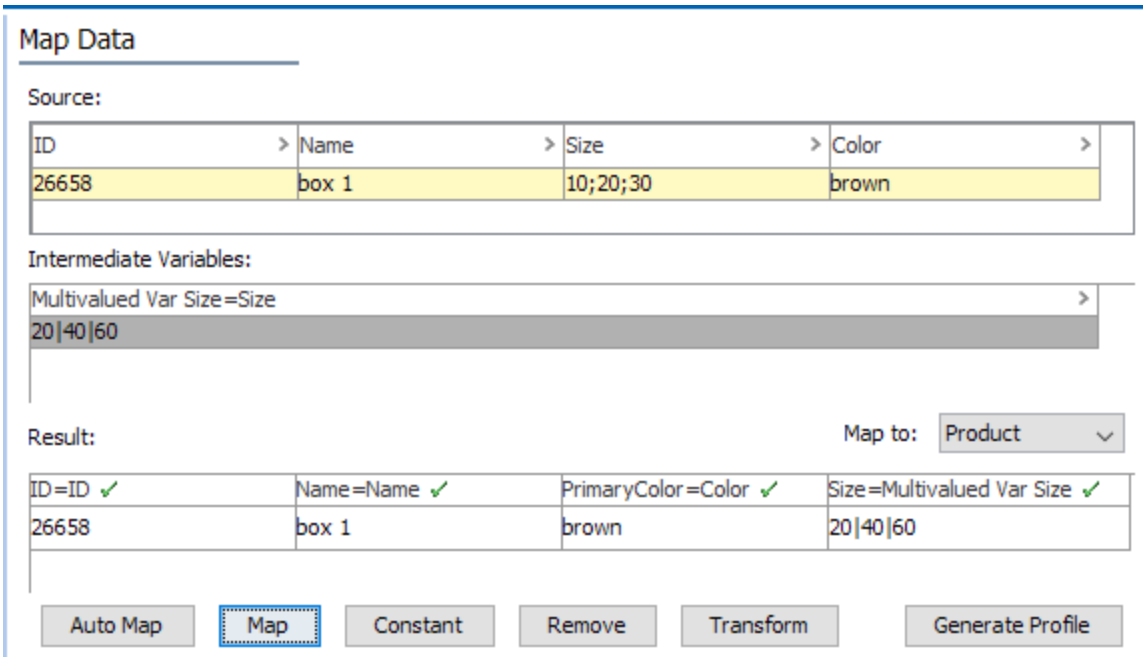
5. Click **OK** and the **Intermediate Variables** section displays the multivalued variable to temporarily hold modified data for additional transformations.



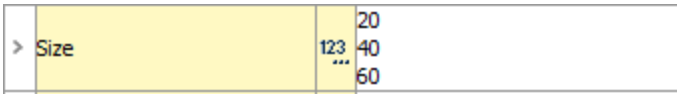
6. Click on the multivalued variable in the Intermediate Variables section, and click the **Transform** button.
7. Select any needed transformations. The Preview pane shows how the transformations modify each part of the value in the multivalued variable, as 'multivalued parts' are designated by the **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 see the **Transformations** topic in the **Resource Materials** online help.

For an example of the 'Append from source' transformation being used, see 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, see 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, see **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

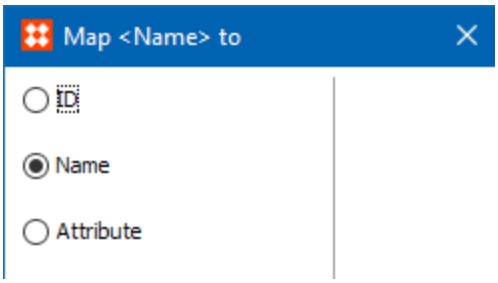
Map a Name

1. In the Source section, select the data column to map to the object name, and then click **Map**.

Source:

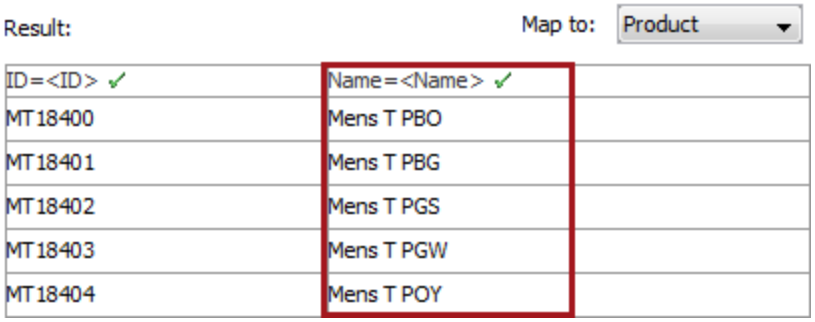
<ID>	<Name>	<Parent ID>	Primary Color	Secondary Color
MT18400	Mens T PBO	18209	Black	Orange
MT18401	Mens T PBG	18209	Blue	Green
MT18402	Mens T PGS	18209	Green	Silver
MT18403	Mens T PGW	18209	Gray	White
MT18404	Mens T POY	18209	Orange	Yellow

2. In the 'Map ... to' dialog (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, see **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:

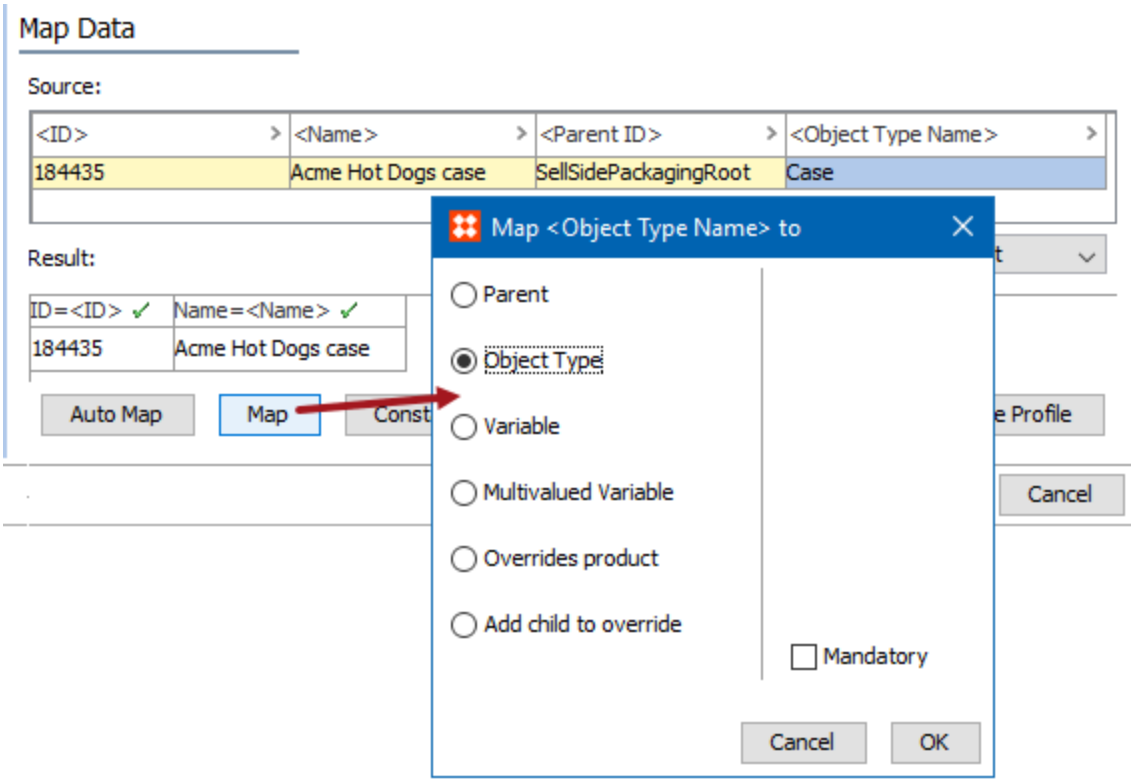
- Changing a globally revisable entity object type requires single update mode. See the **Single-Update Mode** topic in **System Setup / Super User Guide** documentation.
- Changing the object type of a child entity if the parent is globally revisable is not allowed.
- Changing the object type of the entity when the Term List is different is not allowed.

Mapping

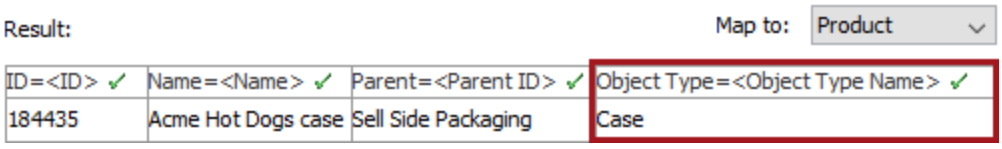
The steps for mapping to create new objects or to update existing objects are the same as follow. For prerequisites and examples of creating or updating, see the following sections below:

- **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, see 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:

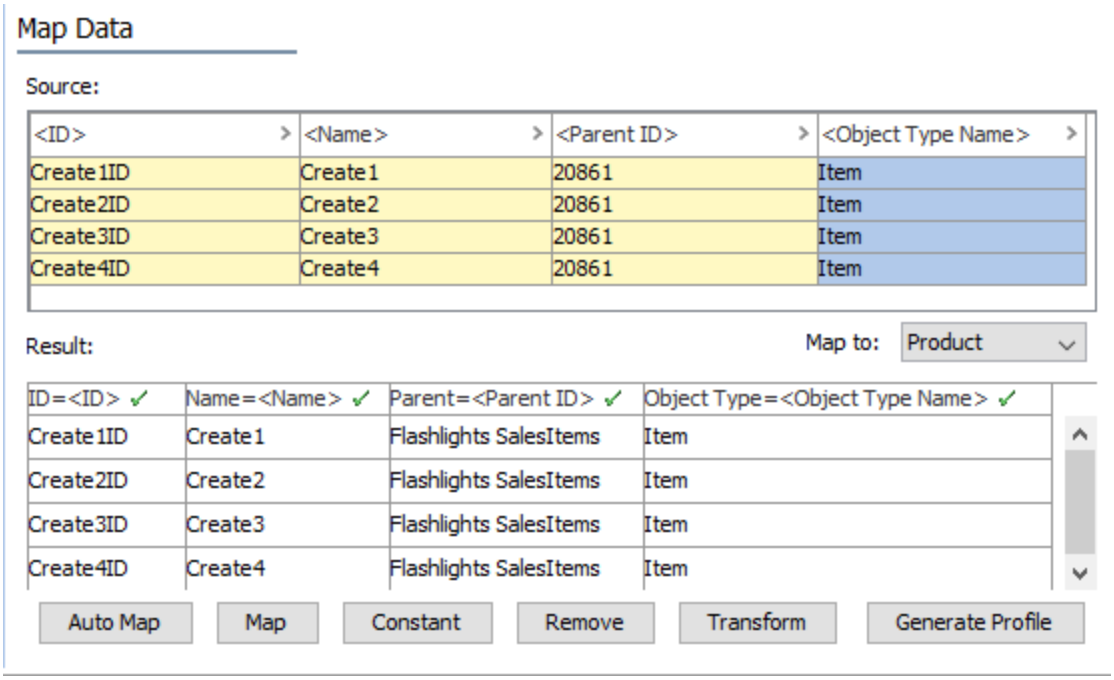
1. The inbound file must include: object ID (or key), parent ID, and object type.
2. The object type in the inbound file must already exist in STEP.
3. When the parent ID is not in the inbound file, and the object type is not valid, an error is displayed in the background process.

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.

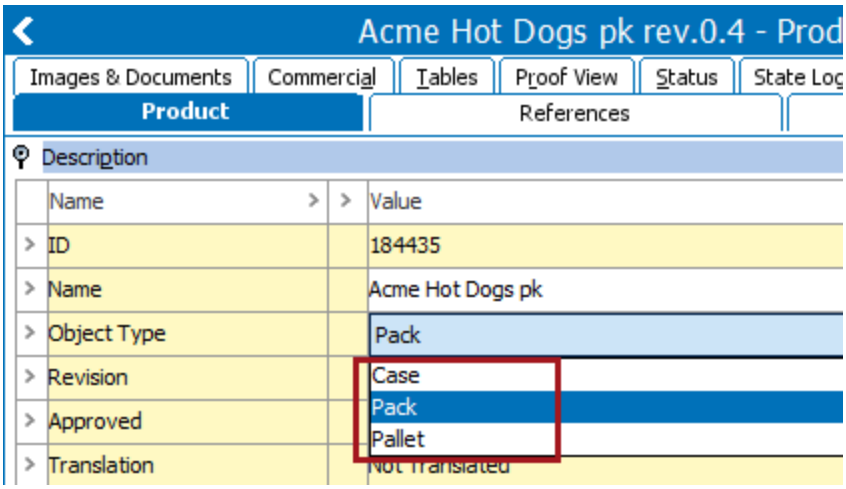
Prerequisites

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

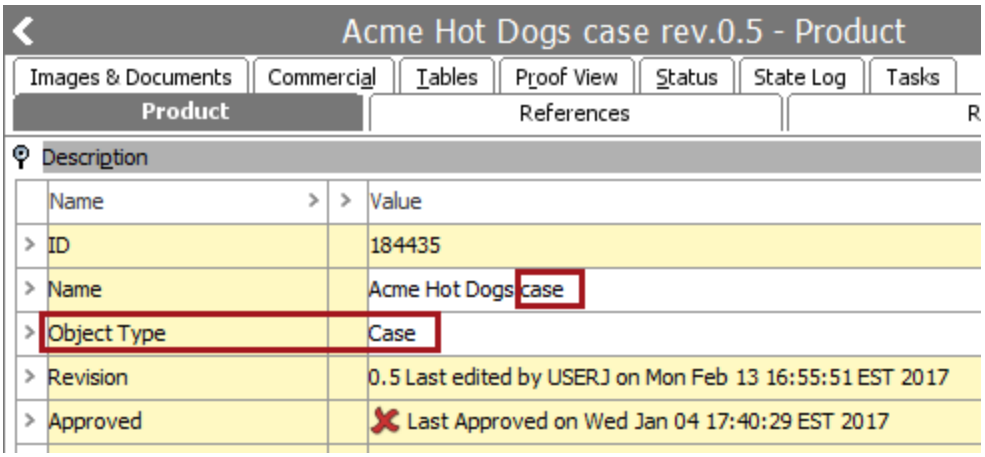
Inbound Data

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, see the **Product Override** topic in the Getting Started / User Guide documentation or the **Override Scenario** use case topic in the **Calculated Attributes** section of the **System Setup / Super User Guide** documentation.

The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, see **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. See the **Product-Override Object Types and Commercial Object Types** topic in the **System Setup / Super User Guide** 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.

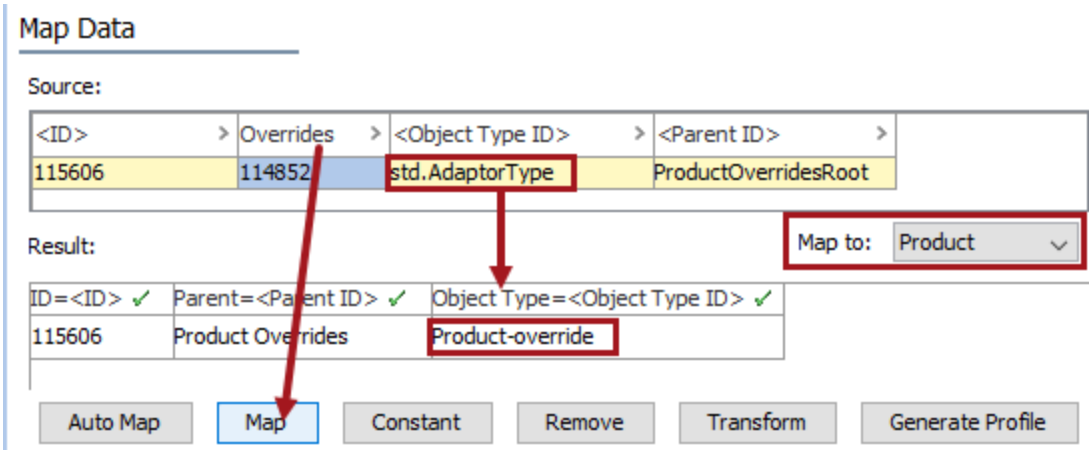
	A	B	C	D
1	<ID>	Overrides	<Object Type ID>	<Parent ID>
2	115606	114852	std.AdaptorType	ProductOverridesRoot

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, see the **ID or Key - Map Inbound** topic.

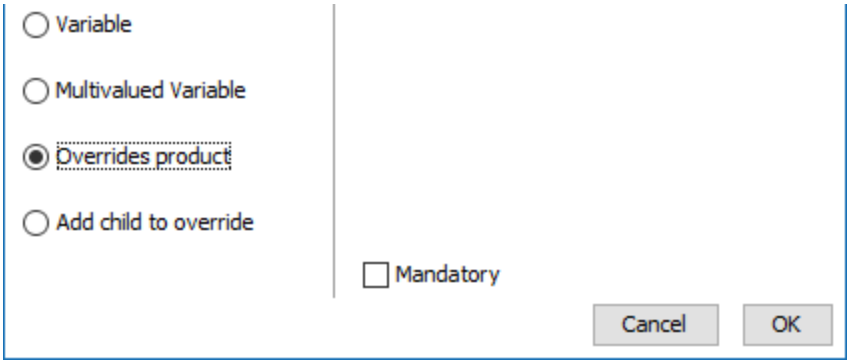
   Columns that include Predefined Header Text can be automatically mapped as defined in the **Inbound Map Data - Auto Map** topic. For this sample file, this includes the <ID>, <Object Type ID>, and <Parent ID> columns.

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.

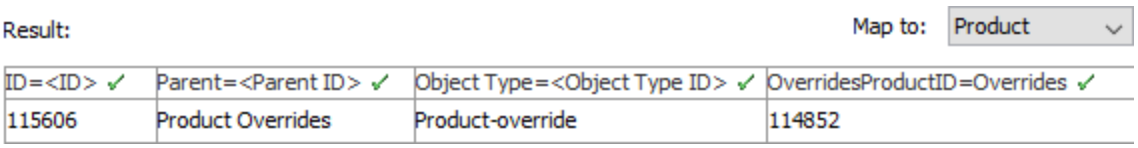
4. In the 'Map ... to' dialog, select the **Overrides product** radio button.



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.

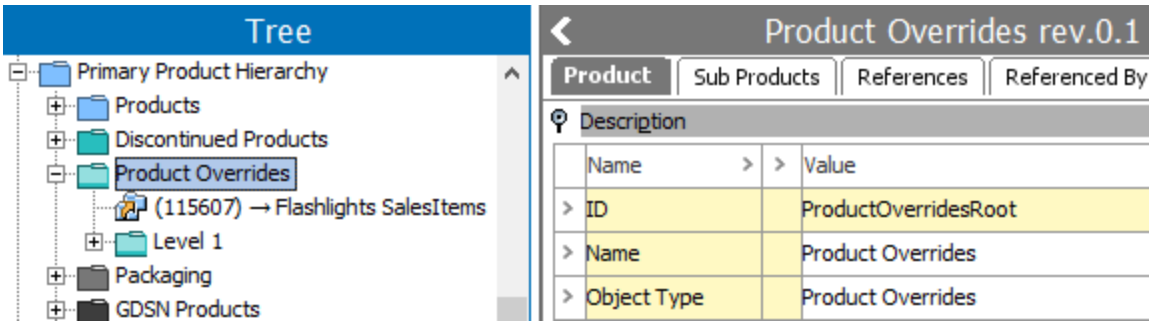
6. Click **OK** and the **Result** panel displays your selection of column and attribute. A validated data column is marked with a green check mark as shown below.



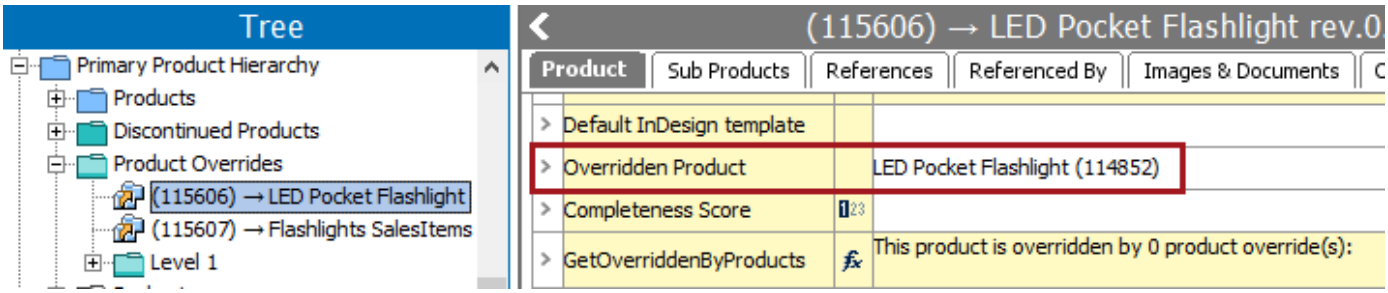
7. Complete the mapping and initiate the import.

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

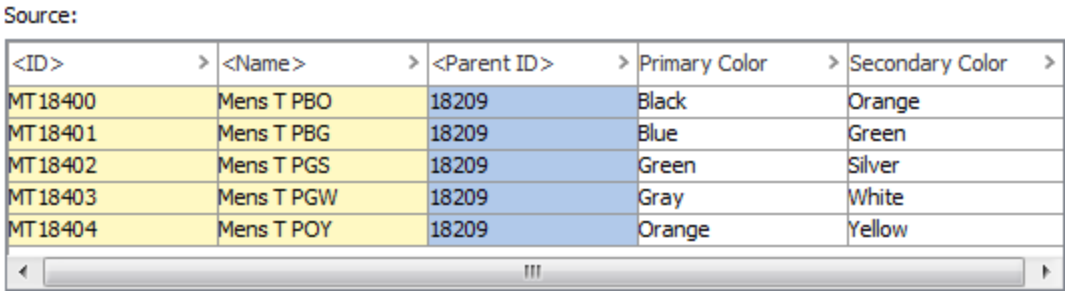
The parent column indicates where to locate, or relocate, the imported object(s). When updating attribute values for existing objects, mapping the parent column is not required, even if it exists in the input file. Common setup involves mapping the parent column when one of the following conditions is true:

- new objects are being imported into STEP
- existing objects within the hierarchy need to be relocated

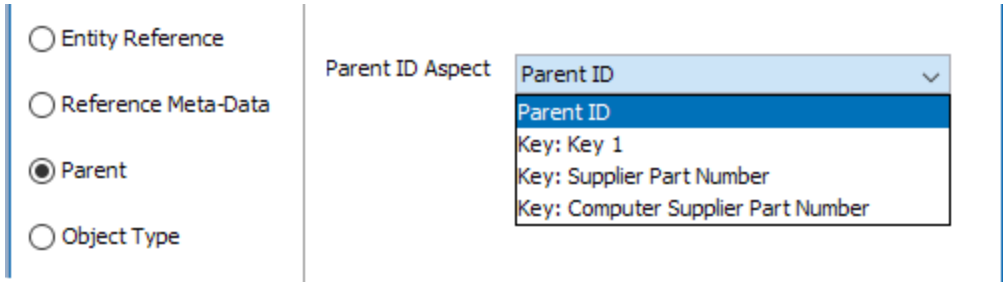
The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, see **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Map Parent

1. In the Source section, select the column of data that includes valid STEP IDs for parents, and click the **Map** button.

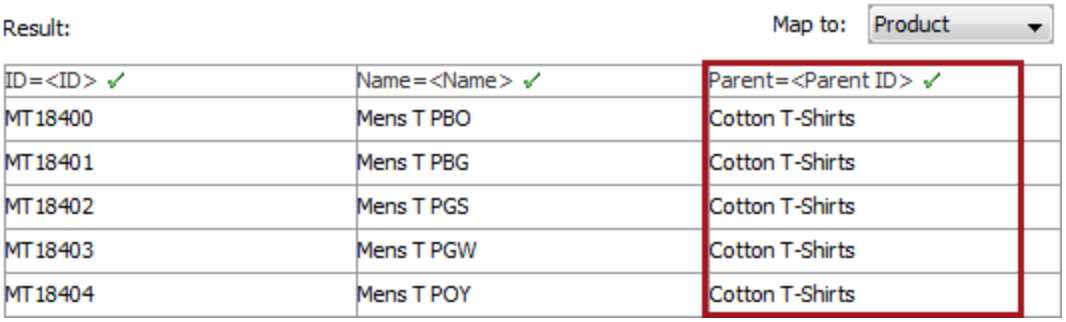


2. In the 'Map ... to' dialog (the label of the field is used as the title of this dialog.), select the **Parent** radio button. Select the 'Parent ID' option from the **Parent ID Aspect** dropdown list. To use a Key, instead of the Parent ID, select the appropriate option from the Parent ID Aspect dropdown.

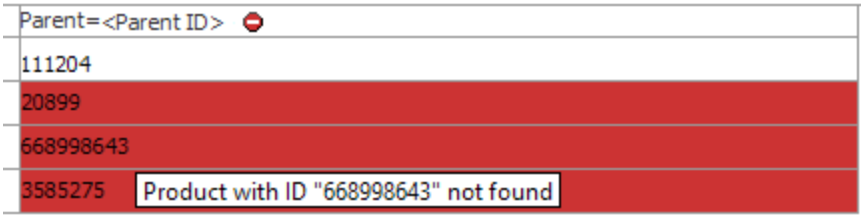


3. Check the **Mandatory** option appropriately:
   - If **checked**, imported objects must have a parent ID in the file. If a parent ID does not exist for an object, the object is skipped and not imported.
   - If **unchecked**, all objects are imported regardless if a parent ID is specified or not.

4. Click **OK** and the **Result** panel displays your selection.



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.



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 / Key of the product.
2. The ID / 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

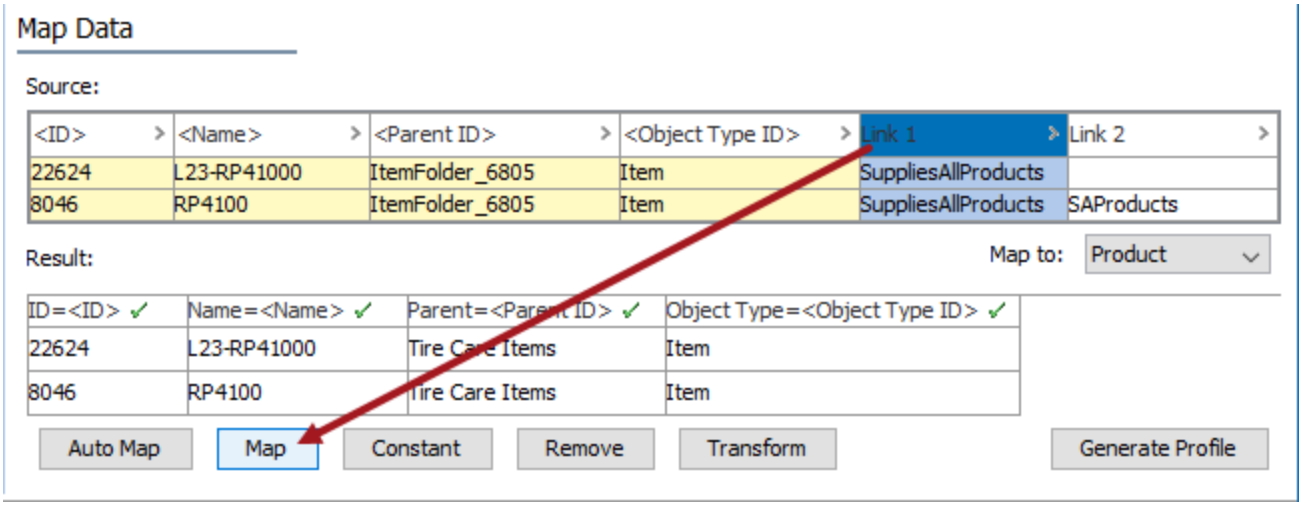
The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, see **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

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, see 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.

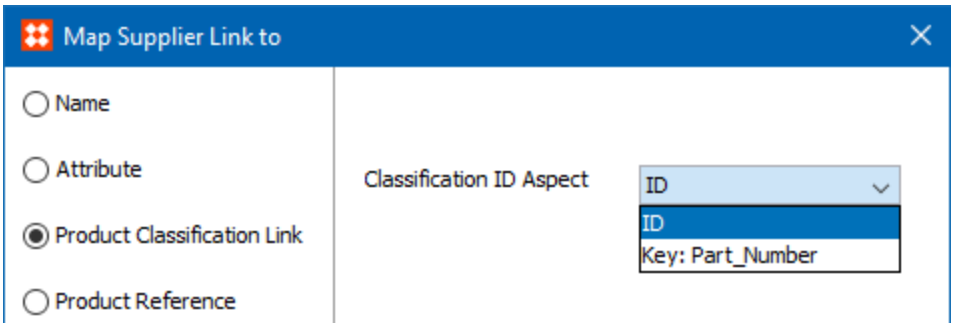
3. In the Source section, select the classification ID column of data and click the **Map** button.



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 / 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

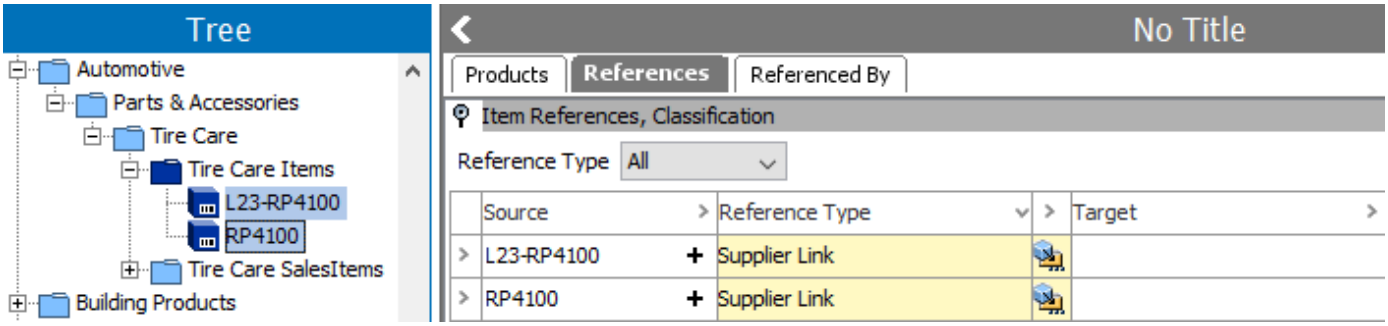
ID=<ID> ✓	Name=<Name> ✓	Parent=<Parent ID> ✓	Object Type=<Object Type ID> ✓	Classification=Link 1 ✓	Classification=Link 2 ✓
22624	L23-RP41000	Tire Care Items	Item	SASProducts	
8046	RP4100	Tire Care Items	Item	SASProducts|SAProducts	

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.

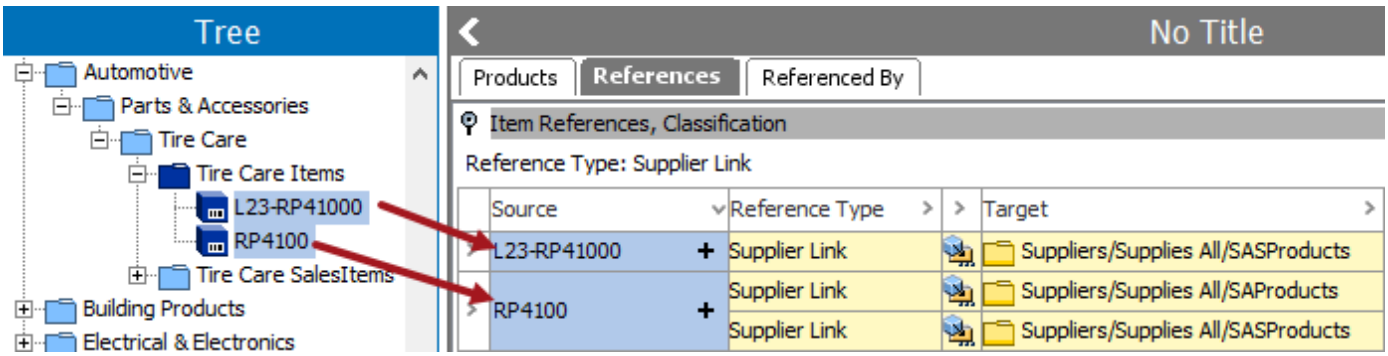
8. Complete the mapping and initiate the import.

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.


Tree
Automotive
Parts & Accessories
Tire Care
Tire Care Items
L23-RP41000
RP4100
Tire Care SalesItems
Building Products
Electrical & Electronics
No Title
Products
References
Referenced By
Item References, Classification
Reference Type: Supplier Link
Source
Reference Type
Target
L23-RP41000
Supplier Link
Suppliers/Supplies All/SASProducts
RP4100
Supplier Link
Suppliers/Supplies All/SAProducts
Supplier Link
Suppliers/Supplies All/SASProducts

Product Reference - Map Inbound

You can reference products, classifications, or entities IDs / Unique keys listed in one column of the input file to another column of products IDs / 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 / unique key of the source object, and the other wit the ID / 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, see **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

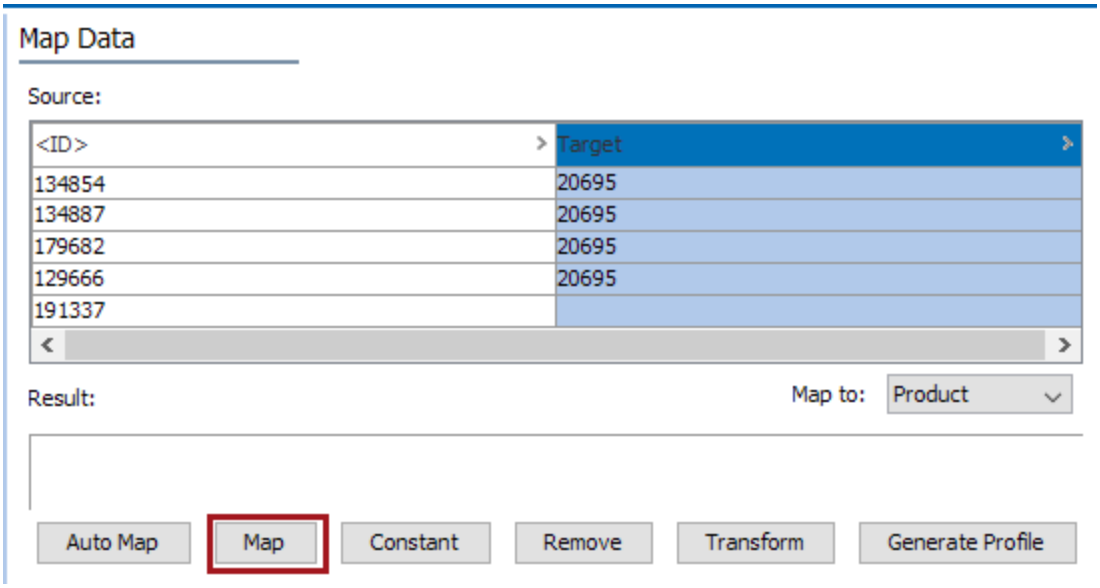
Prerequisites

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, see the **Activating and Deactivating Keys** topic of the **System Setup / Super User Guide** documentation.

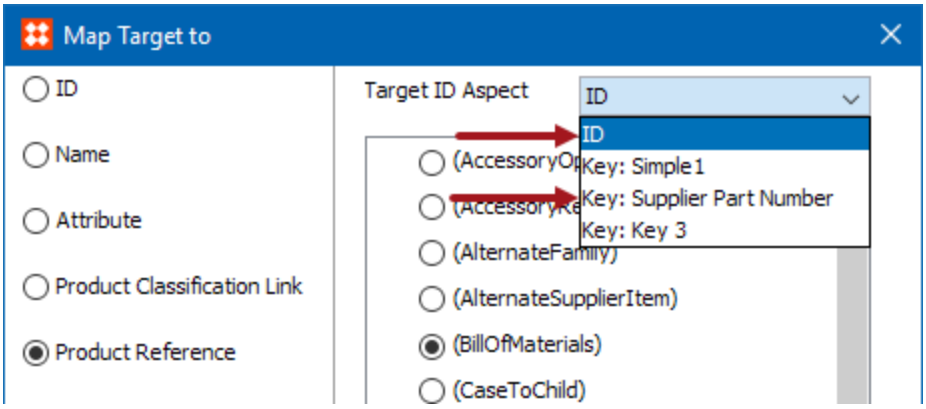
The following example assumes that the 'Allow multiple references parameter' on the Reference Type is set to Yes, which means that multiple products can be linked to the same product reference. For more information, see the **Multiple References for the Same Objects** topic within the **System Setup / Super User Guide** documentation.

Map Product Reference

1. In the Source window, select the ID or Key column and map it. For more information see the **ID or Key - Map Inbound** documentation.
2. Select the reference target column in the Source section and then click the 'Map' button.

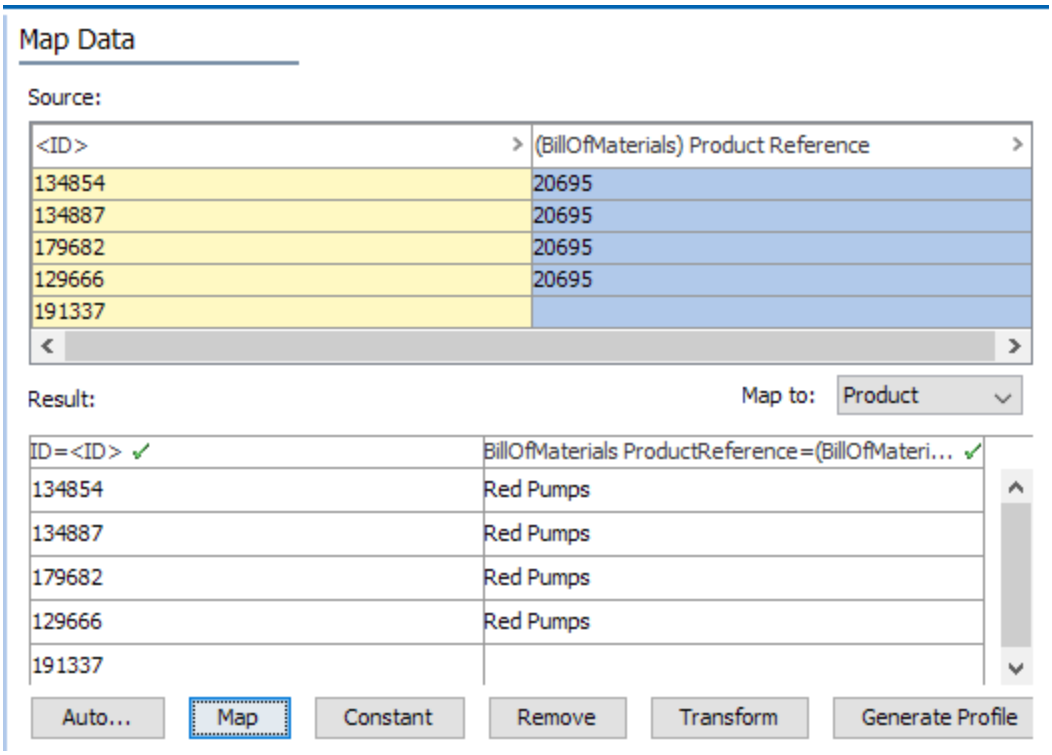


3. In the 'Map ... to' dialog, the label of the import field is used as the title. Make the following selections:
   - Select the **Product Reference** radio button option.
   - In the 'Target ID Aspect' dropdown list, select either ID or the appropriate key. See the **Prerequisites** section above if mapping by key.
   - Select the desired Reference type.

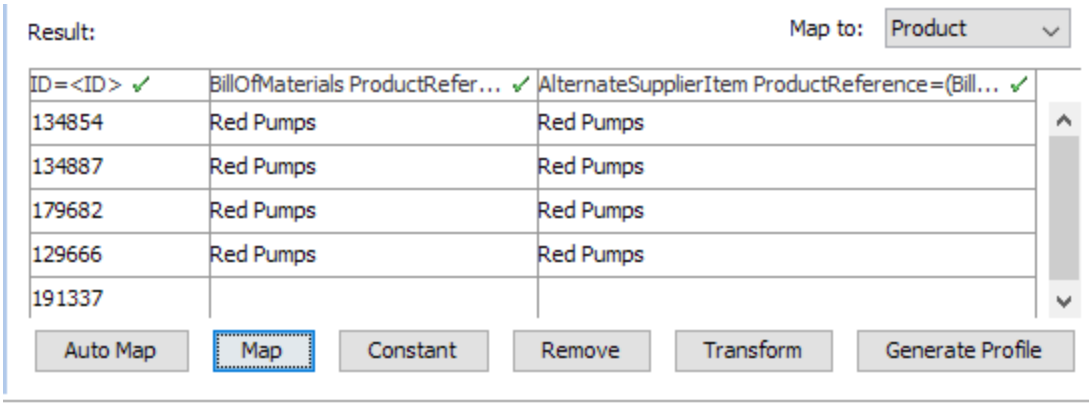


4. Check the **Mandatory** option appropriately:
   - If **checked**, imported objects must have a Target ID in the file. If a Target ID does not exist for an object, the object is skipped and not imported.
   - If **unchecked**, all objects are imported regardless if a Target ID is specified or not.

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.

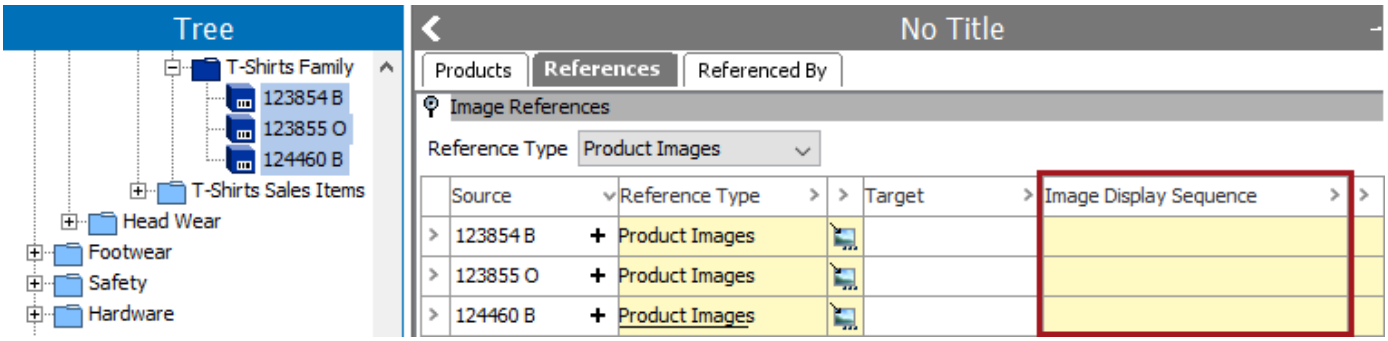


Reference Meta-Data - Map Inbound

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, see **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.



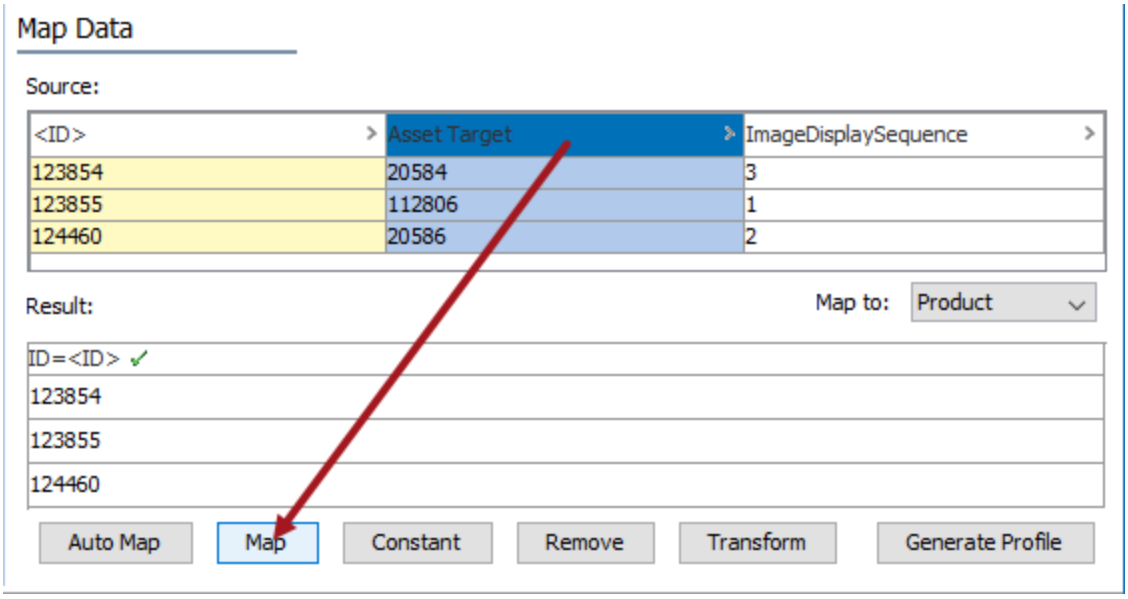
To import metadata attribute values into the 'Image Display Sequence' metadata attribute, an input file must contain columns holding the metadata attribute values, the Product IDs, and the referenced target asset IDs.

Map Reference Metadata

1. Map the source ID or key column using the Auto Map button, or manually. For more information, see 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.

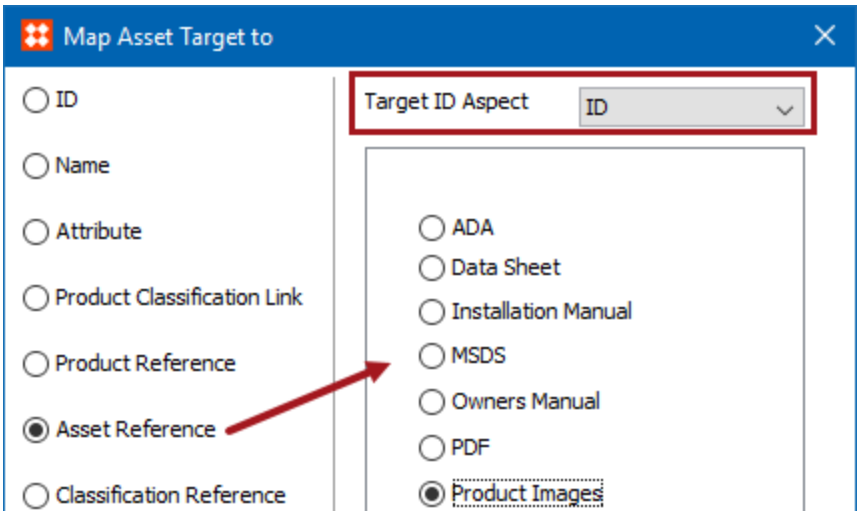
2. In the Source section, select the target column of data and click the **Map** button.



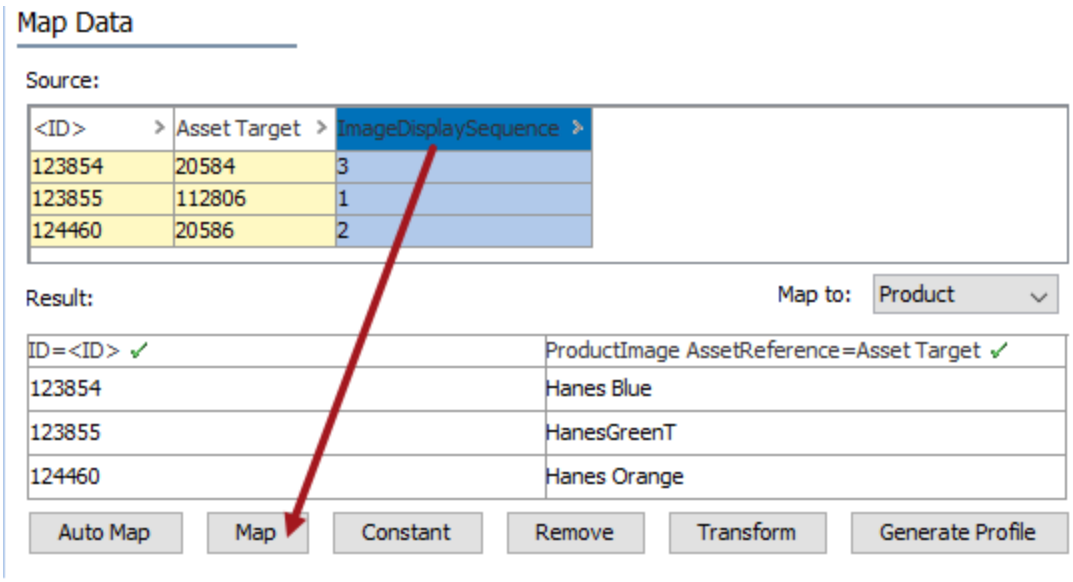
3. In the 'Map ... to' dialog (below), select the **Asset Reference** radio button, and then select the reference type.

   Only existing reference types are listed. You cannot specify or create new reference types during the import.

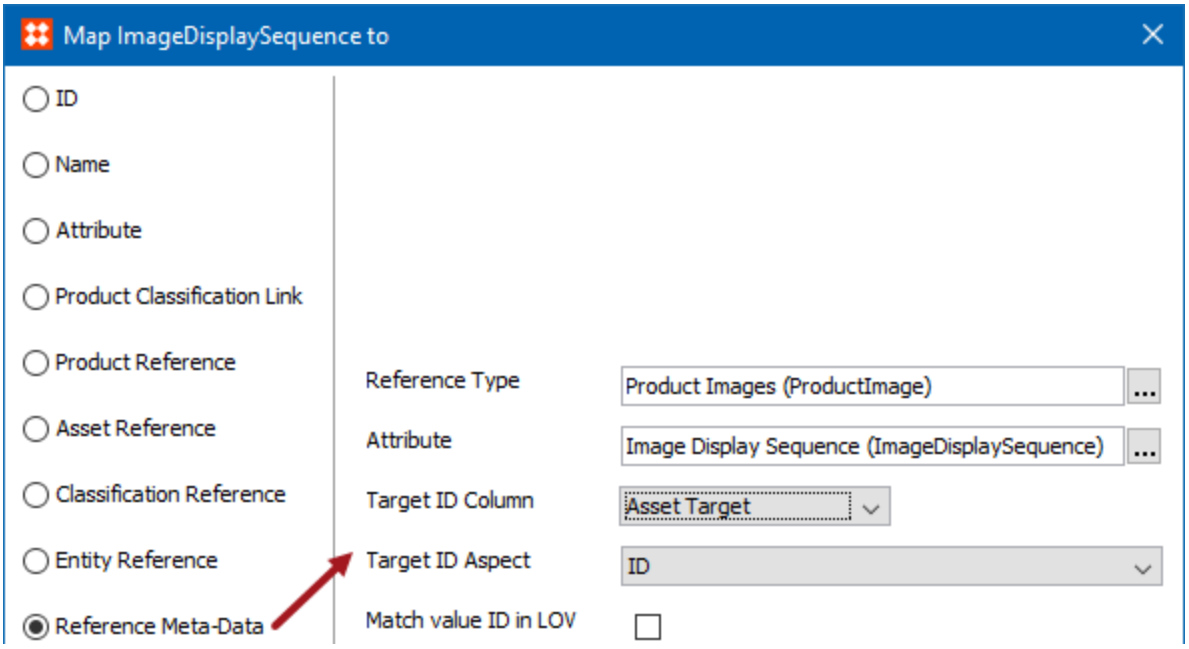
4. For the **Target ID Aspect** parameter:
   - Select **ID** when the inbound file includes the ID for the target.
   - Select the **Key** from the dropdown when the inbound file includes a key for the target.



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.

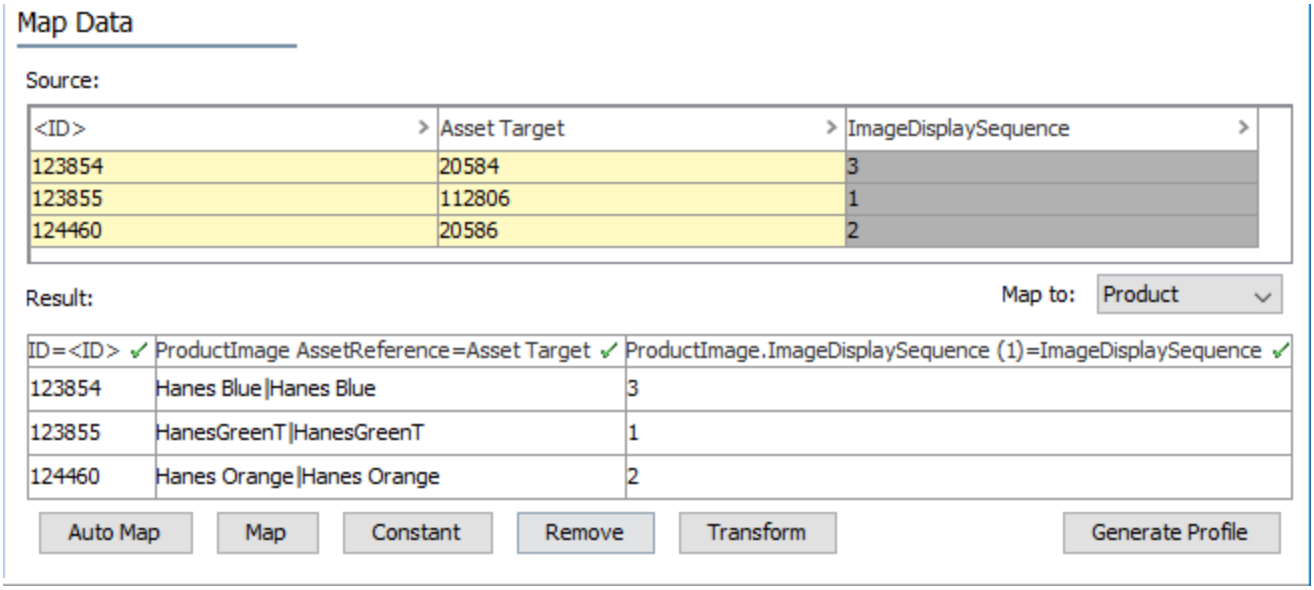


8. Complete the additional parameters as follows:
   - For **Reference Type**, select the type of reference that should receive the metadata values. In this example, the asset reference type 'Product Images' is selected.
   - For **Attribute**, select the attribute that should receive the metadata values. 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.

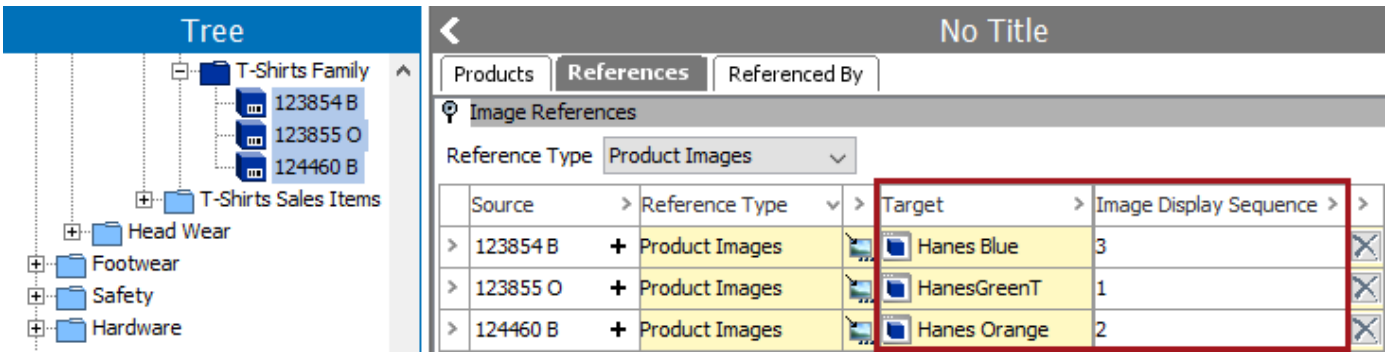
10. Click **OK** and the **Result** panel displays your selection of reference target. A validated data column is marked with a green check mark as shown below.



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

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.

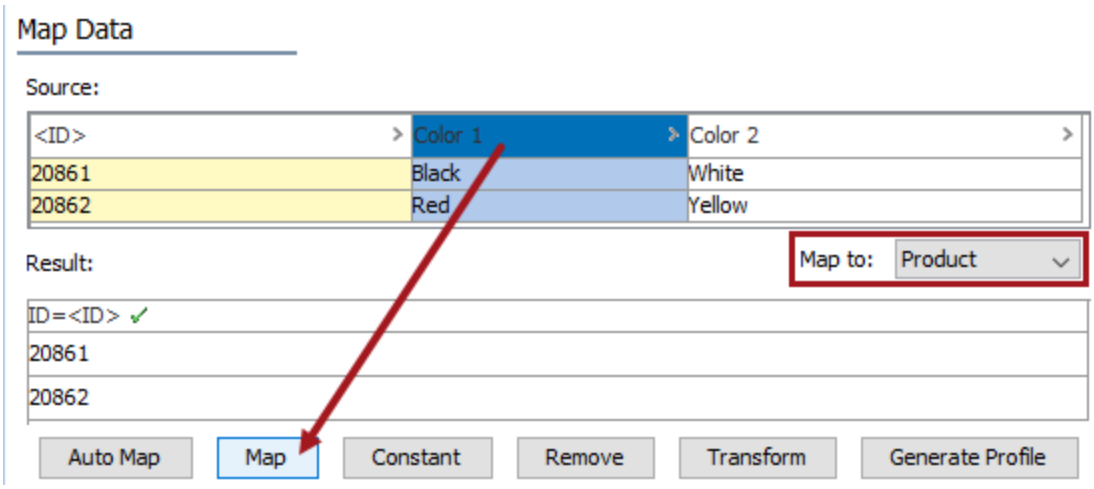
The Variable option is available for use with assets, attributes, classifications, entities, and products.

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, see the **Running a Data Import** topic in the **Import Manager** documentation. or the **IIEP - Configure STEP Importer Processing Engine** topic in the **IIEP** documentation.

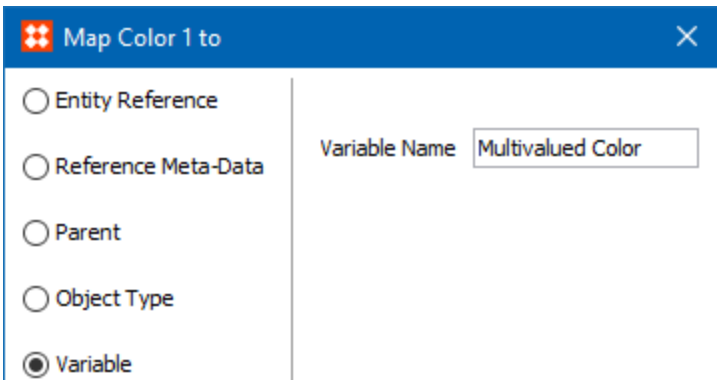
The mapping instructions are the same for both the Import Manager tool and the IIEP tool. For details about starting an import, see **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Map a Variable

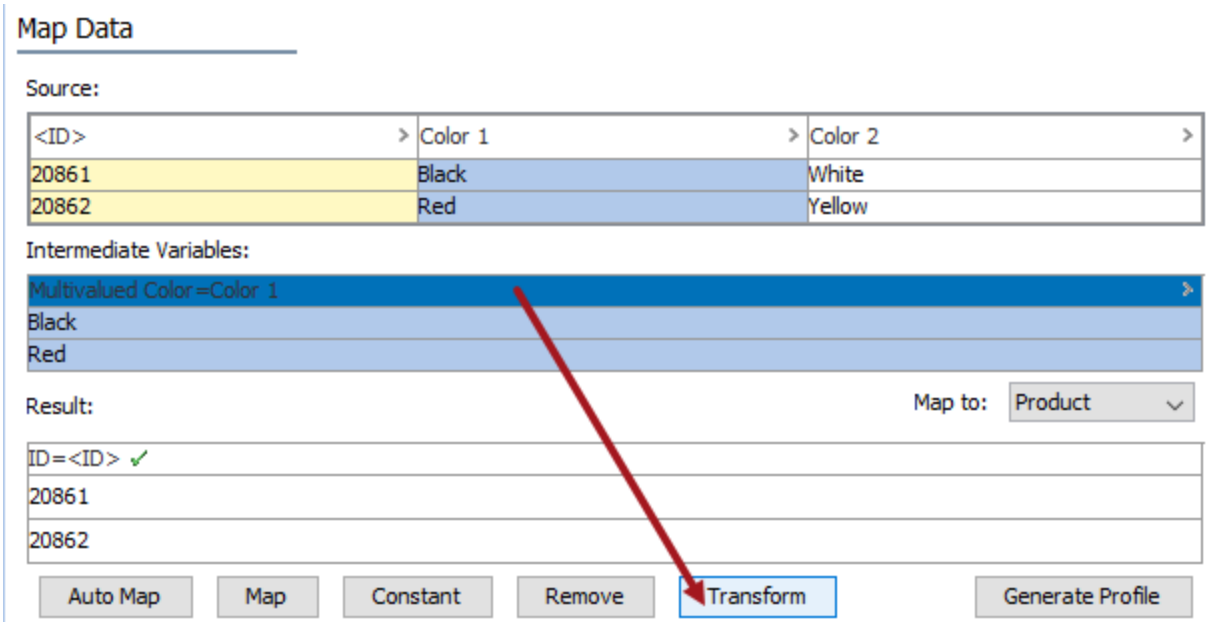
1. Below the Source section, use the **Map to** dropdown to select the object super type required by the import file.
2. In the Source section, select the column of data that you want to map to the variable and click the **Map** button.



3. In the 'Map ... to' dialog, select the **Variable** radio button and provide text to describe the purpose of the variable.

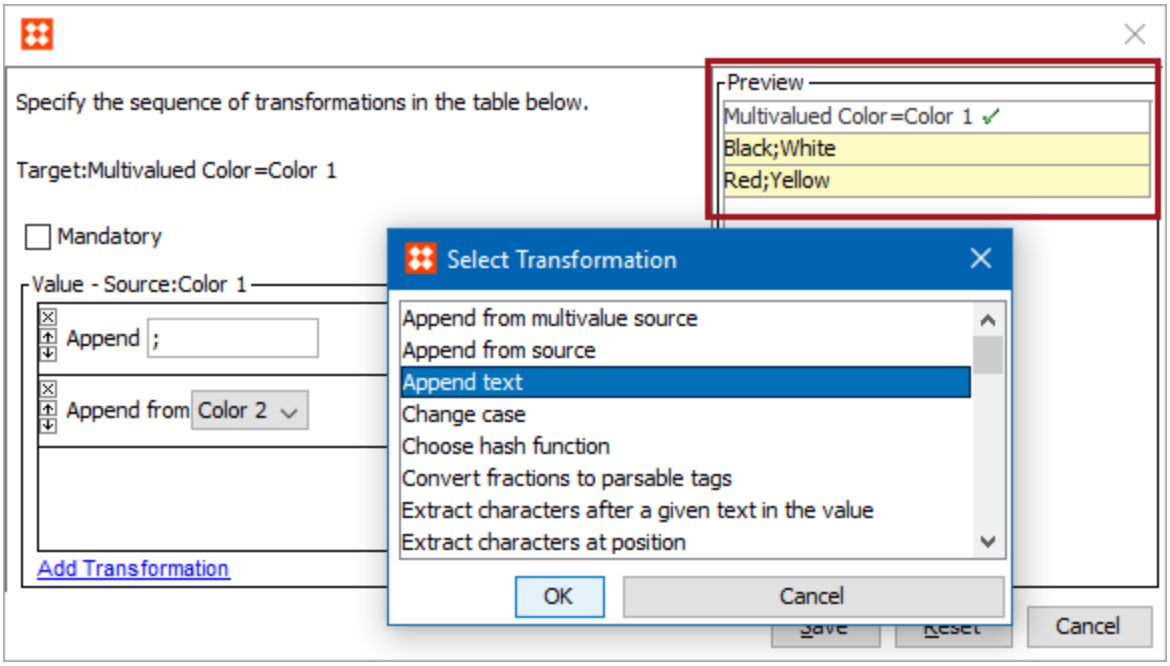


4. Click **OK** and the **Intermediate Variables** section is displayed on the Map Data dialog. The originally selected Source column is shown as mapped to the variable text provided. The data is now available for multiple transformations.

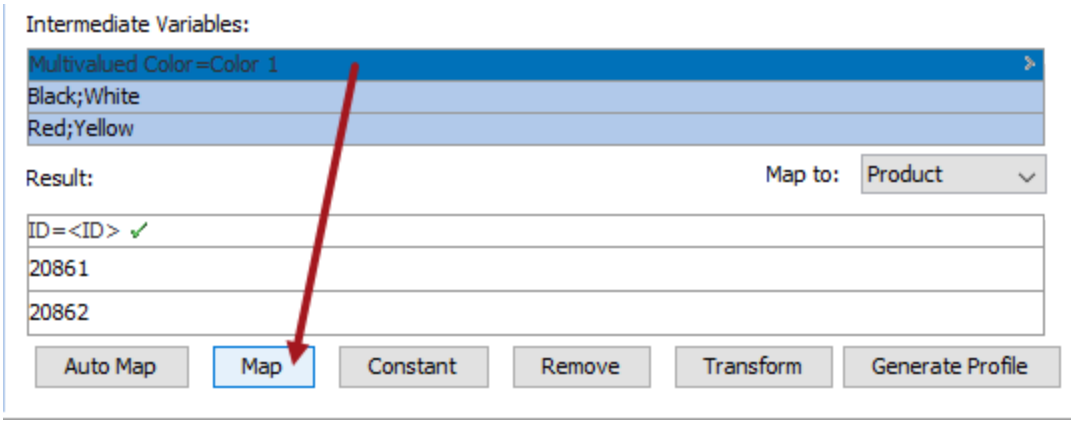


5. Select the intermediate variable column and click the **Transform** button. The transformations dialog is displayed. For more on the transformations, see the **Inbound Map Data - Transform** topic in the **Data Exchange** documentation.
6. Click the **Add Transformation** link to display the Select 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, see 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

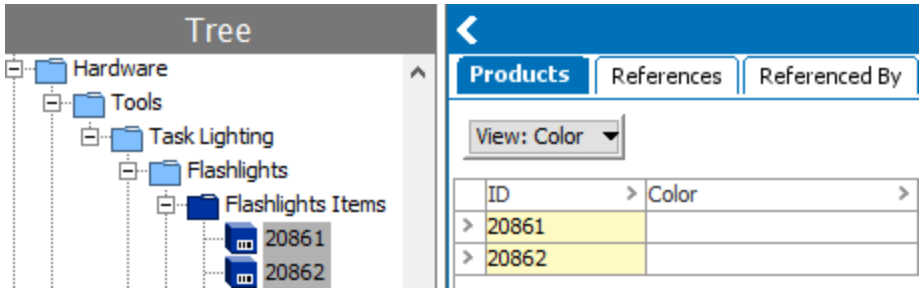
Result: Map to: Product

ID=<ID> ✓	Color=Multivalued Color ✓
20861	Black|White
20862	Red|Yellow

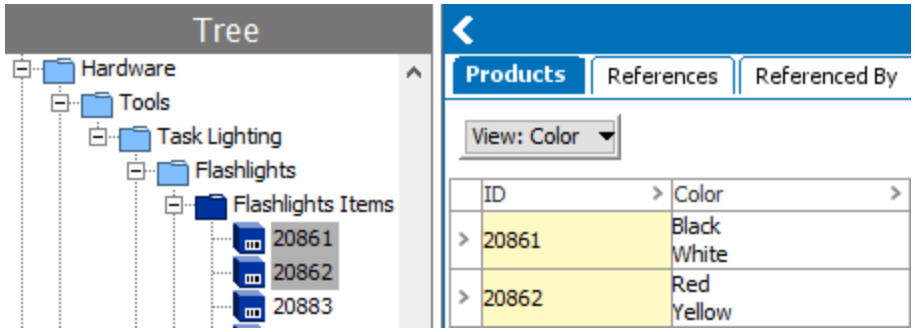
12. Complete the mapping and initiate the import.

Example

Consider two objects with a multivalued attribute named Color.



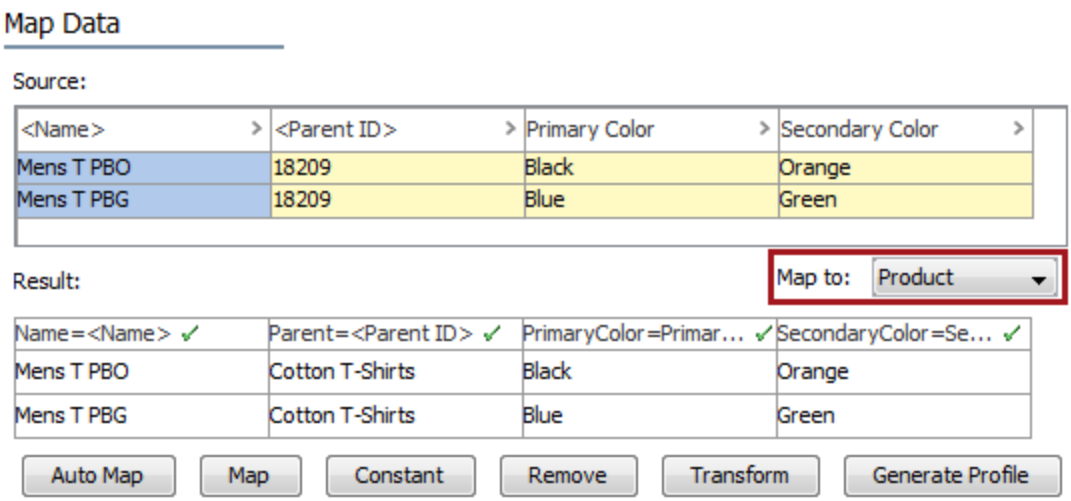
The import file includes two colors for each object, and the variable is used to hold the transformation that adds both colors to the same attribute.



For an example of the 'Append from source' transformation being used, see 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, see **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Mapping Based on Object Type

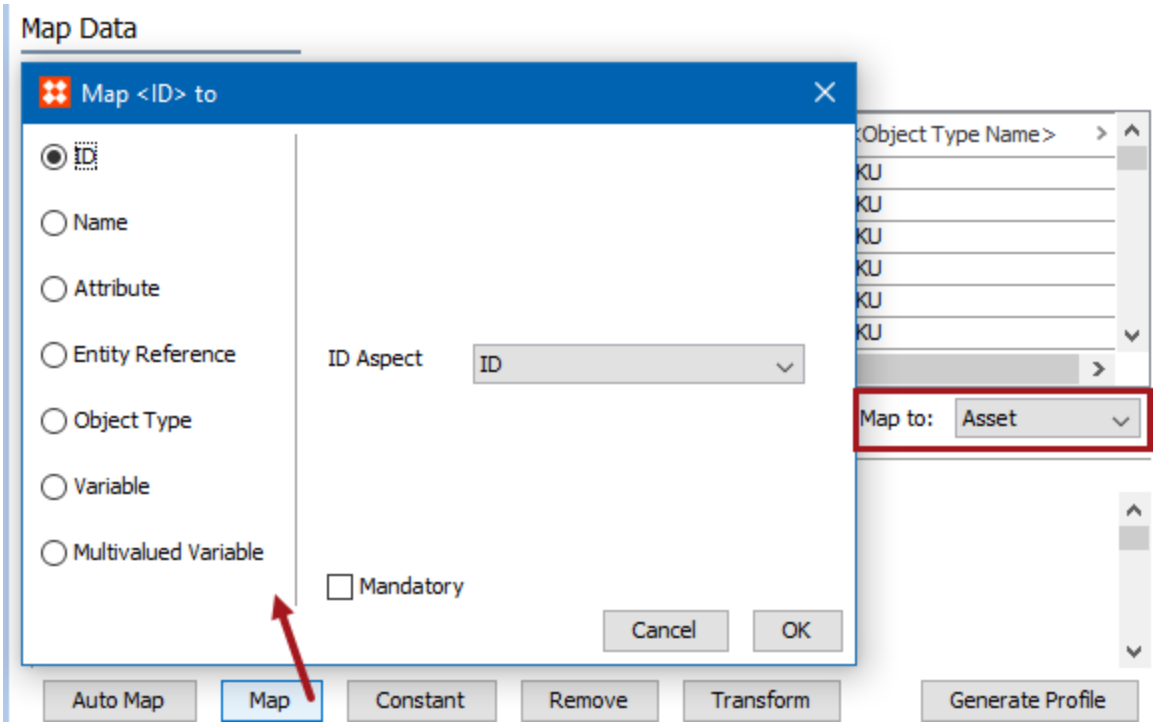
The following object types can be mapped:

- 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, see **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.



For more information, see the **Map Inbound** topics on the available options:

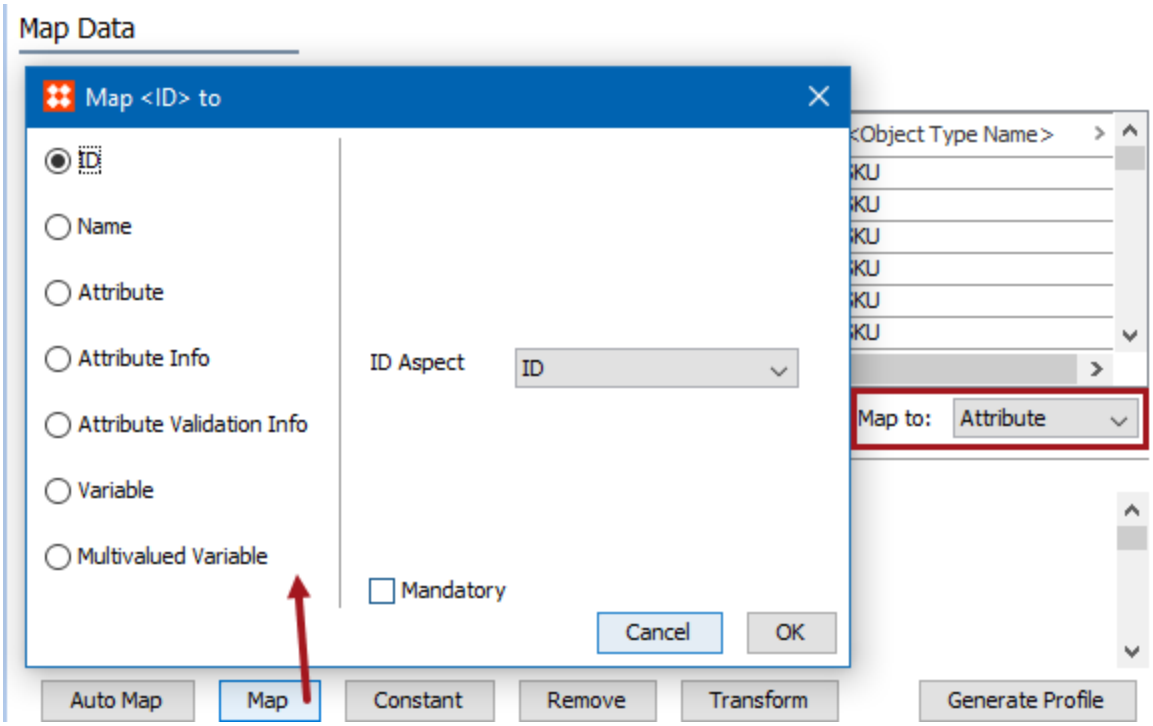
- ID or Key
- Name
- Attribute
- Entity Reference
- Object Type
- Variable
- Multivalued Variable

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, see **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.



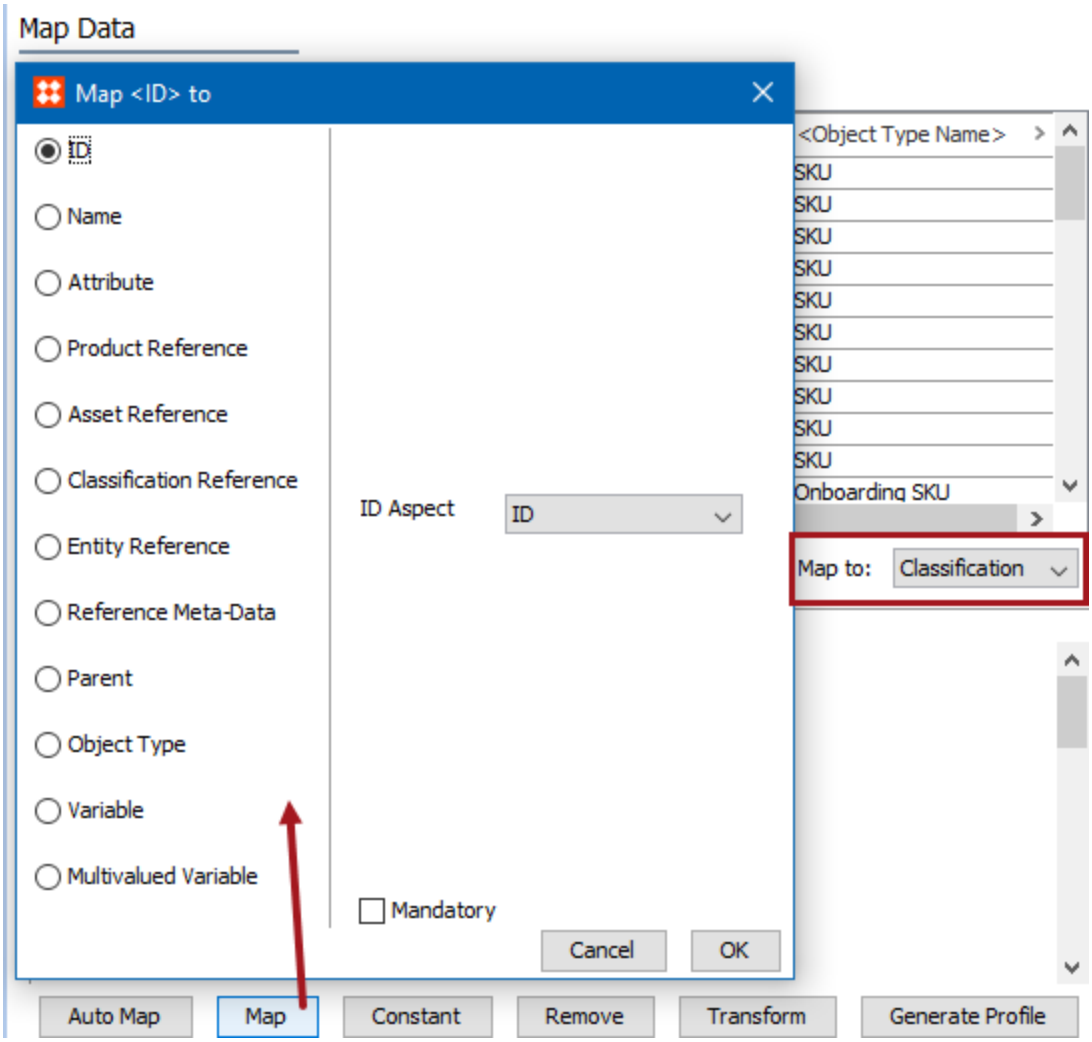
For more information, see the **Map Inbound** topics on the available options:

- 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, see **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.



For more information, see the **Map Inbound** topics on the available options:

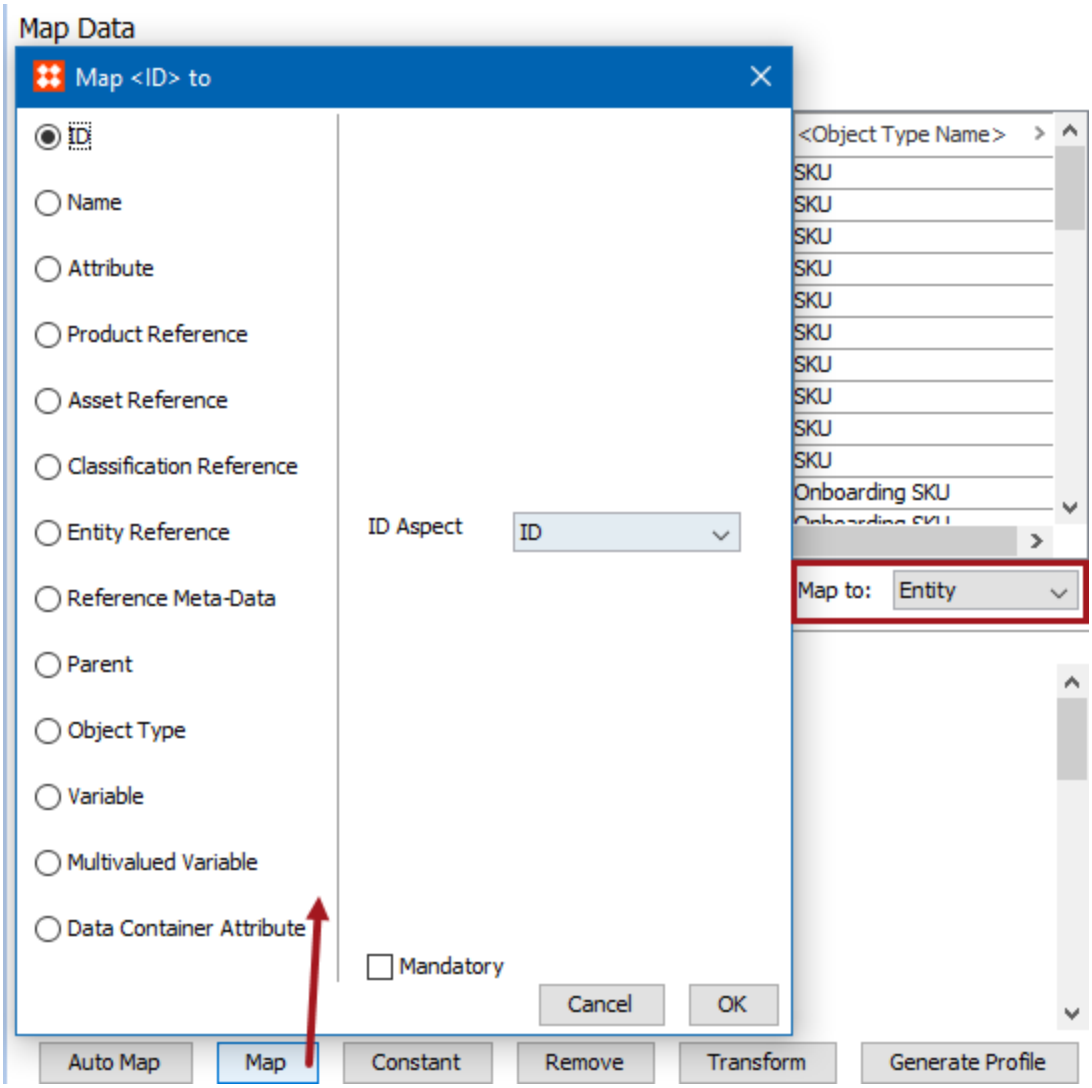
- ID or Key
- Name
- Attribute
- Product Reference
- Asset Reference

- 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, see **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.



For more information, see the **Map Inbound** topics on the available options:

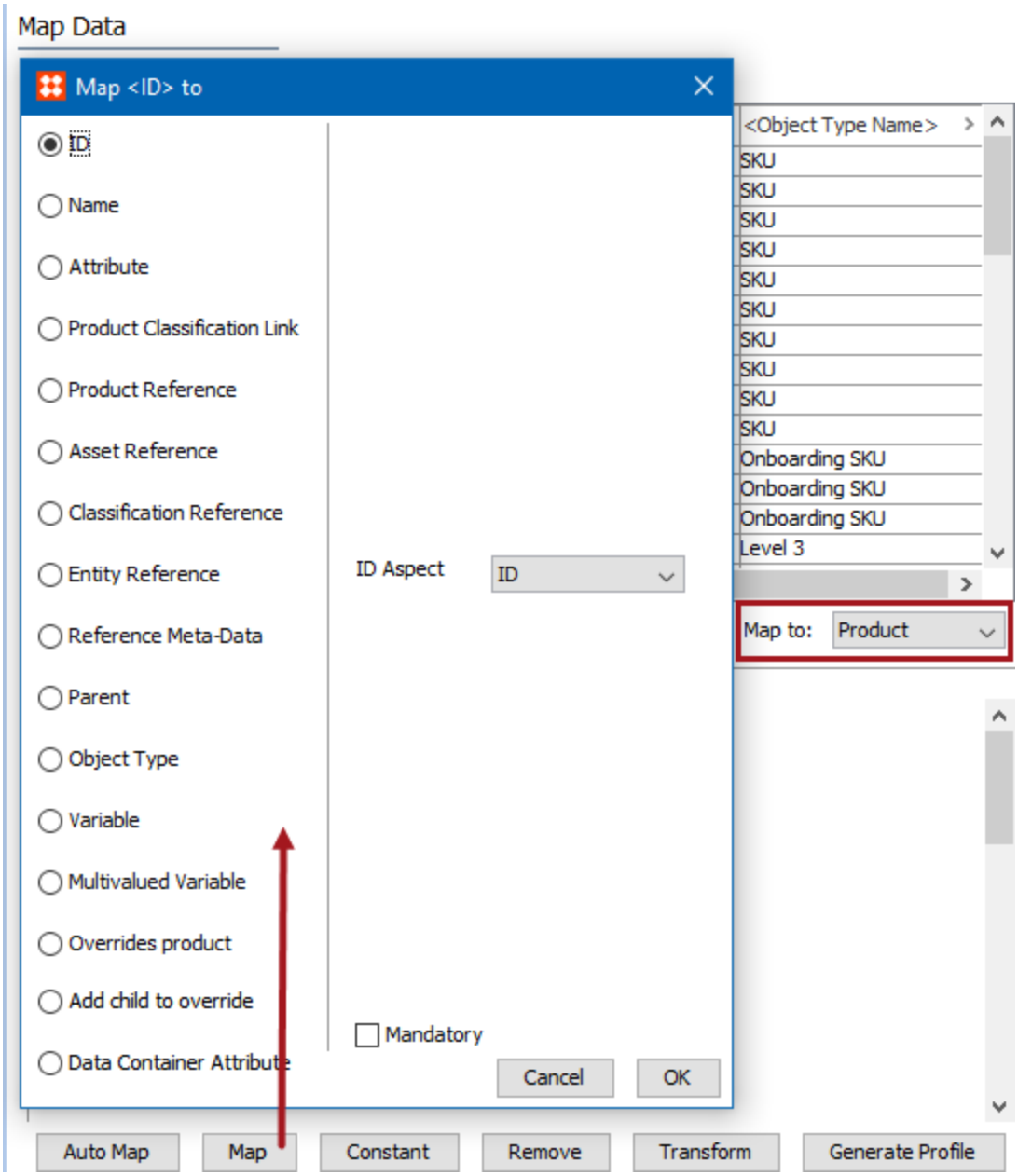
- ID or Key
- Name
- Attribute
- Product Reference

- Asset Reference
- Classification Reference
- Entity Reference
- Reference Meta-Data
- Parent
- Object Type
- Variable
- Multivalued Variable
- Data Container Attribute

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, see **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

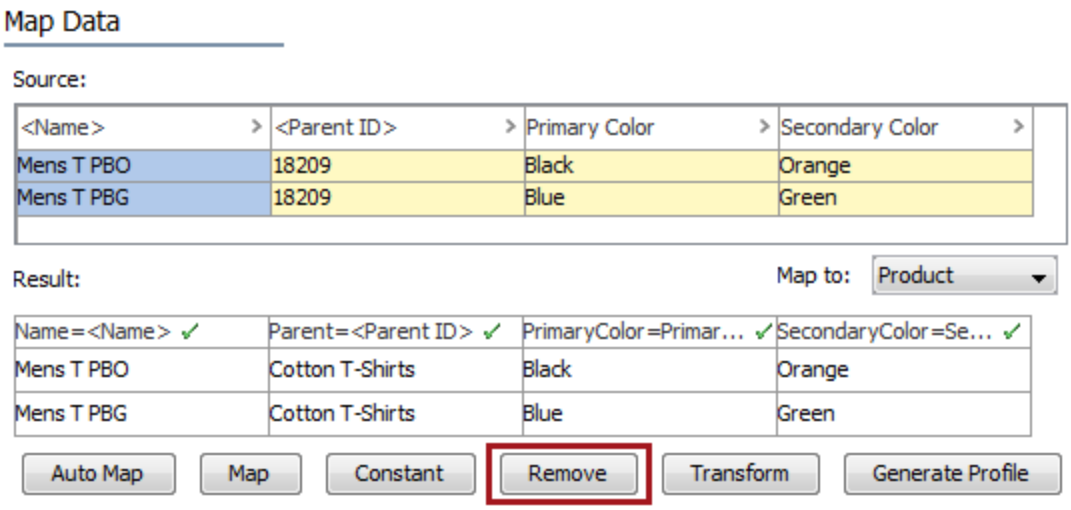


For more information, see the **Map Inbound** topics on the available options:

- 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

Inbound Map Data - Remove

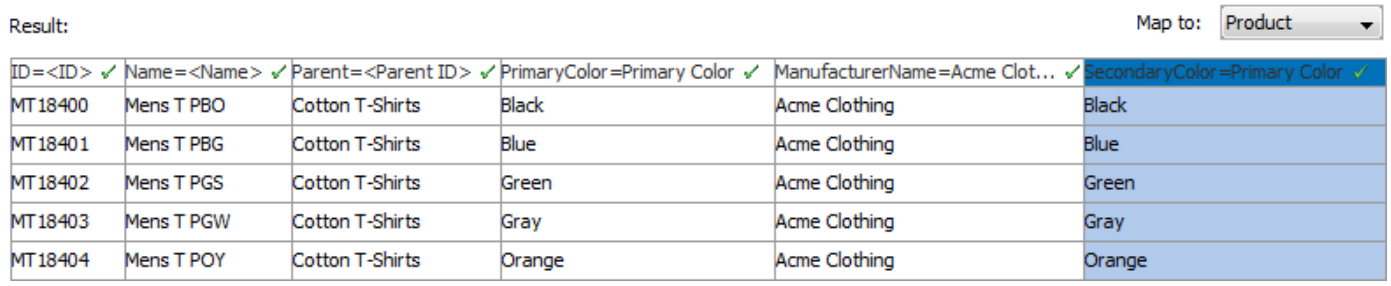
If the Result panel displays a column that you do not want to import, you can remove it.



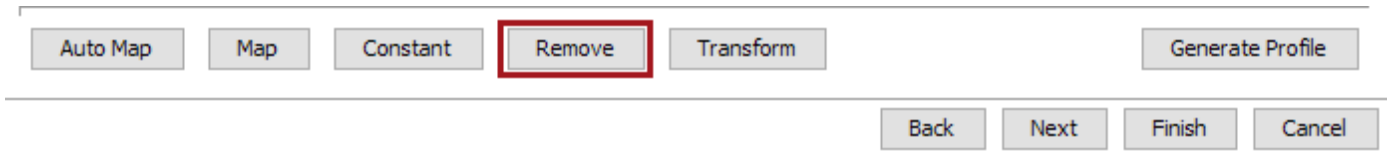
For more information about the additional import steps, see **Creating a Data Import** or **Creating an Inbound Integration Endpoint**.

Remove a Mapped Column

1. In the **Result** panel, select the mapped column to remove.



2. Click the **Remove** button.



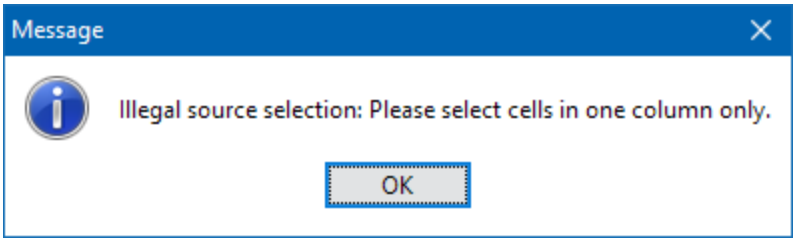
The **Result** panel no longer displays the column, and the removed data will not be imported.

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:

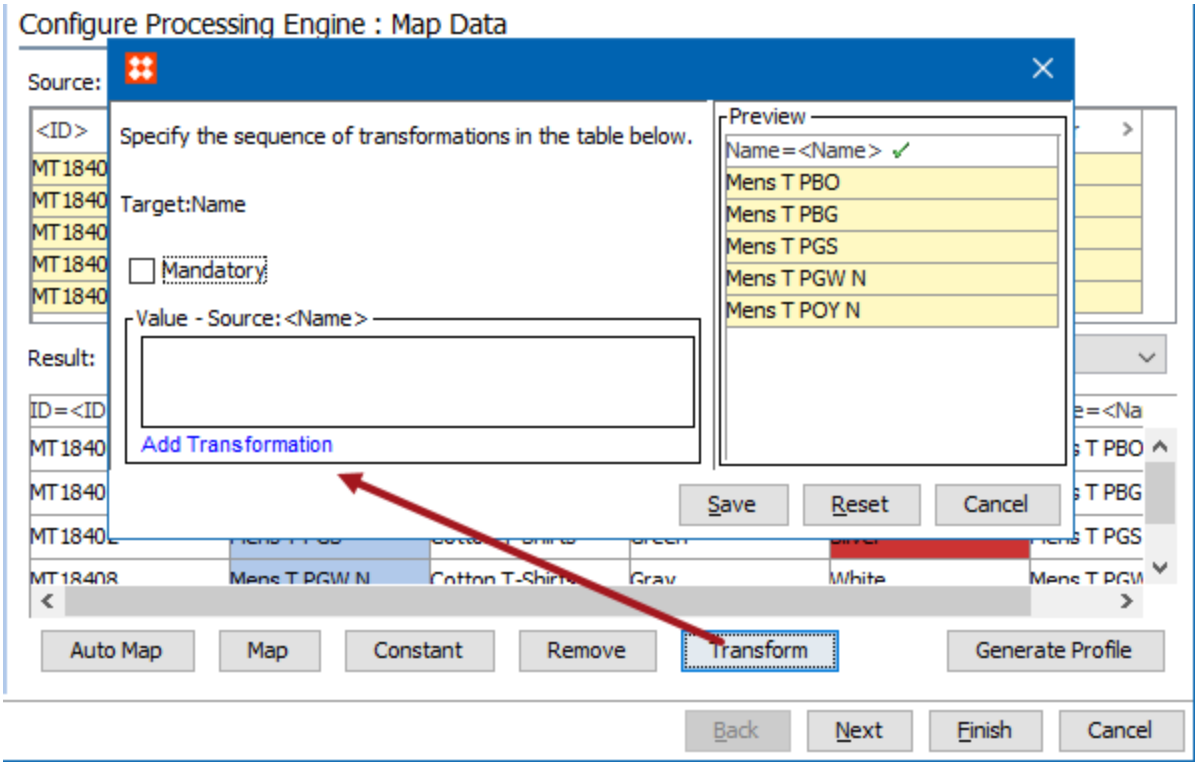
- You can apply multiple transformations to the columns of data, but be aware that they are processed sequentially, from top to bottom.
- When setting up transformations that are implemented with a saved configuration, it is recommended that you work with smaller load files first. You can then build 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, see **Transformation Examples**.

To use transformations on attributes in STEP, or on data in tables and publications, see **Attribute Transformations** in the **System Setup / Super User Guide** 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, see the **Transformations** topic in the **Resource Materials** online help.

For more information about the additional import steps, see **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, see **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 / 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:

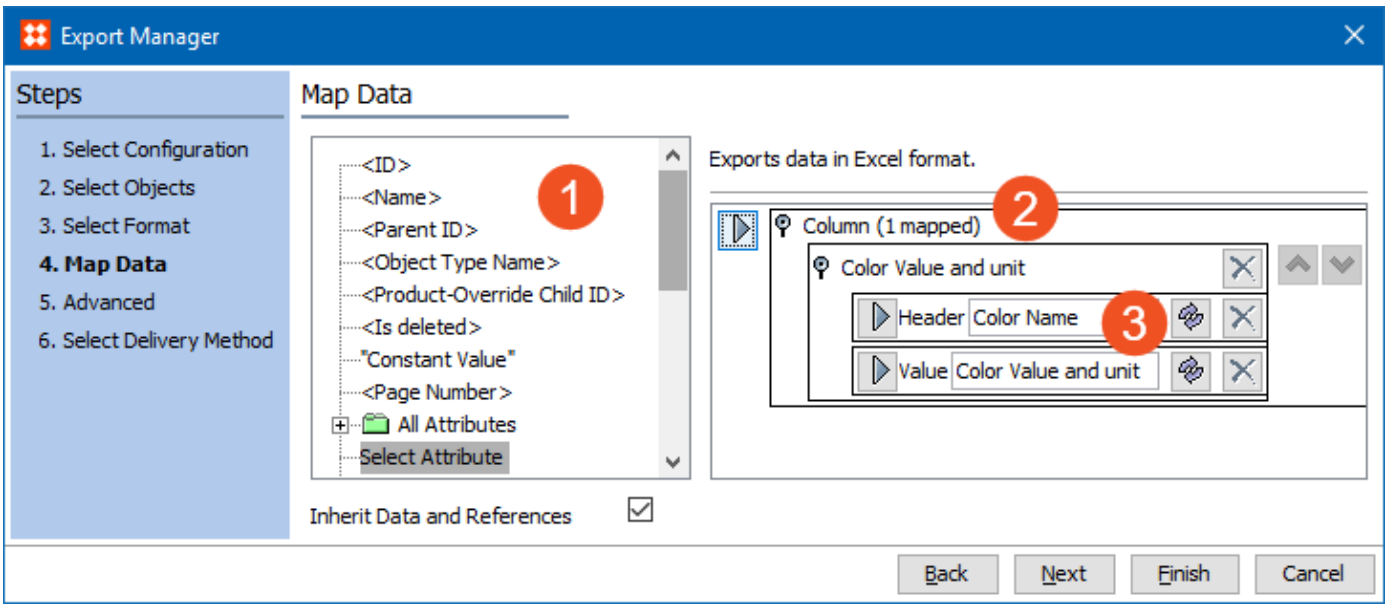
- BMEcat Format
- BMEcat 2005 Format
- Generic XML Format
- IDoc MATMAS 05 Format

Mapping in Outbound Tools

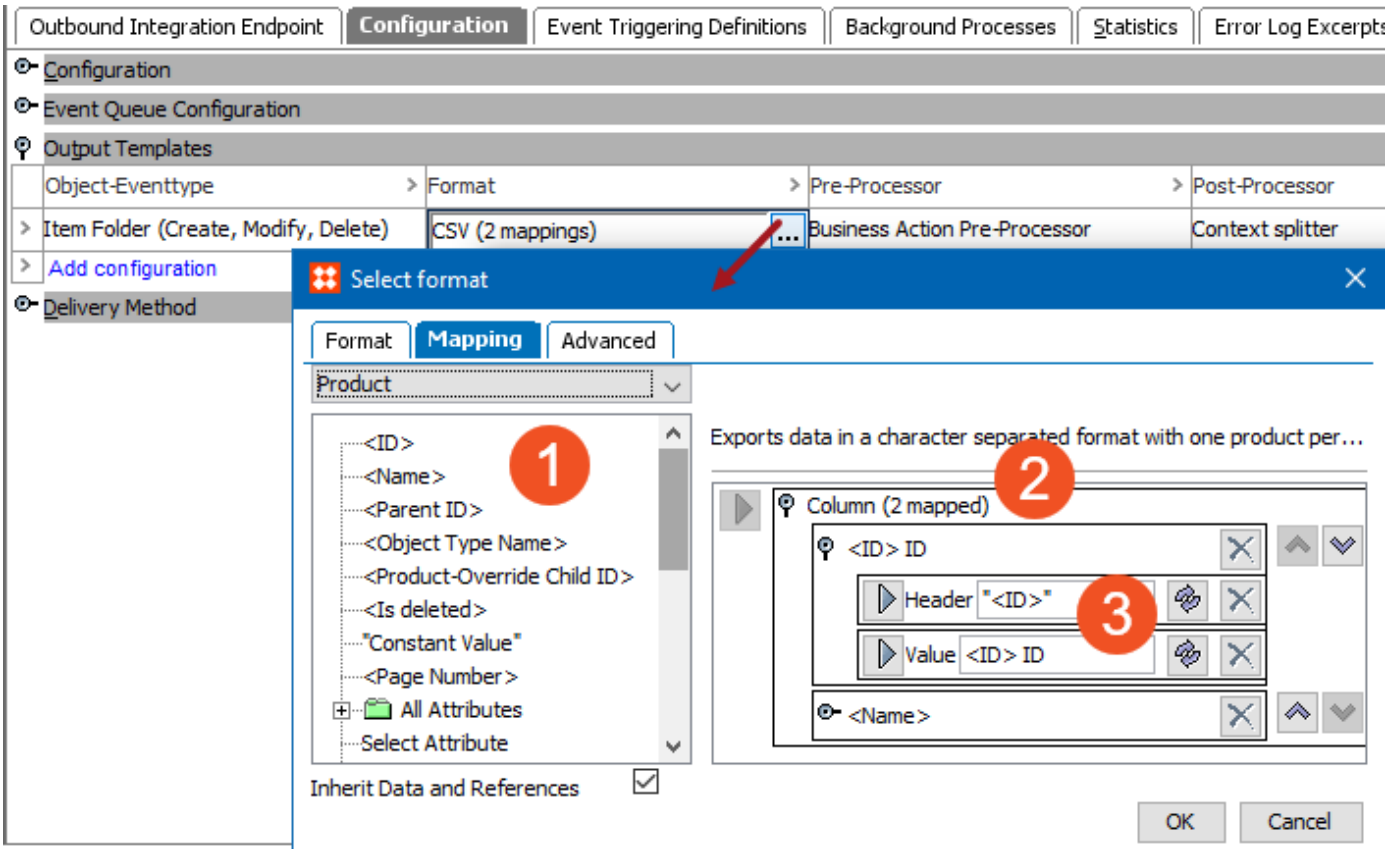
The user interface for the mapping step of the Export Manager tool and the Outbound Integration Endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter 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, see **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:

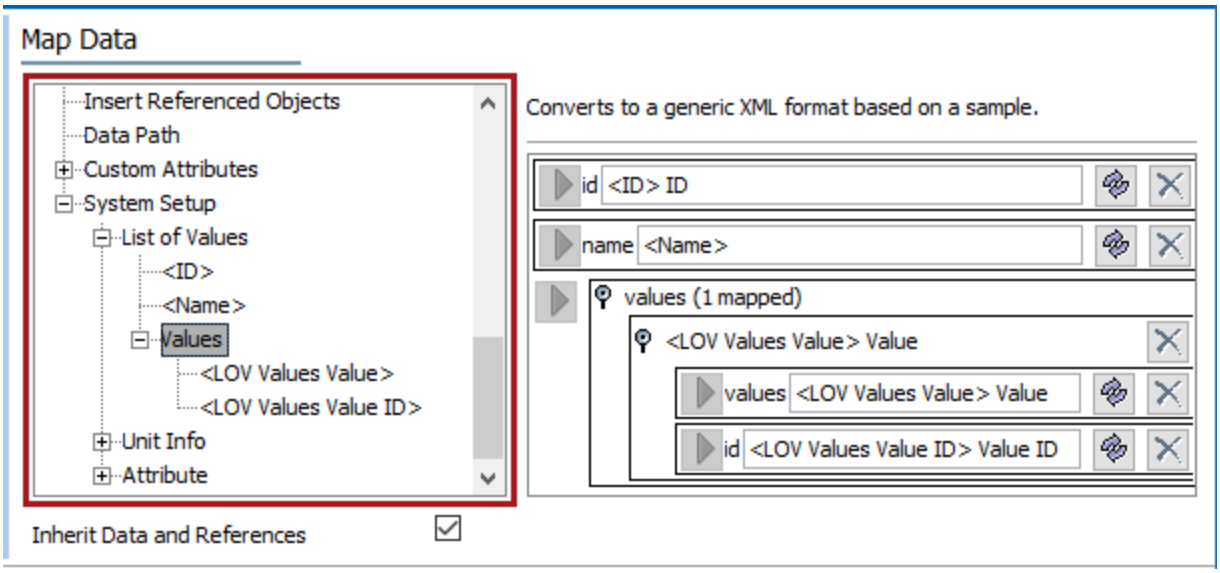
1. **Data Sources** indicate the original location of the information to be exported. The mapping page provides a way to associate the mapping targets with the STEP information available for the selected object types being exported. For more information on each source option, see the **Outbound Map Data - Data Source** documentation.
2. **Mapping Targets** are placeholders for the actual data being exported. The selected format determines which targets are available. See 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, see 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, see **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 Path - Data Source Outbound - for Generic XML and IDoc MATMAS 05 only
- Entity References - Data Source Outbound
- ID or Key - Data Source Outbound
- Index Words - Data Source Outbound - for STEP Publisher only
- Inherit Data and References - Data Source Outbound
- 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 STEP 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
- 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, see **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

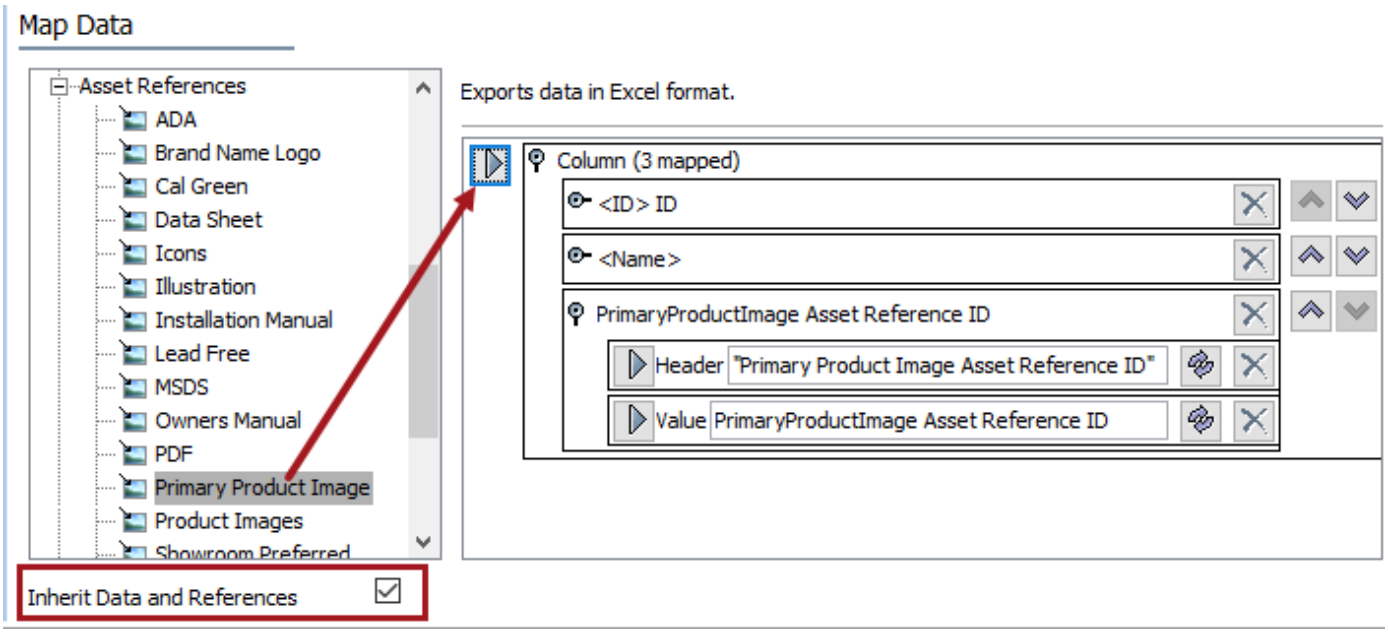
After mapping, most output options can be altered using transformations. For more information, see the **Outbound Map Data - Transform** documentation.

Mapping Asset References

1. Select the data to be exported for the output tool. For more information, see 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.

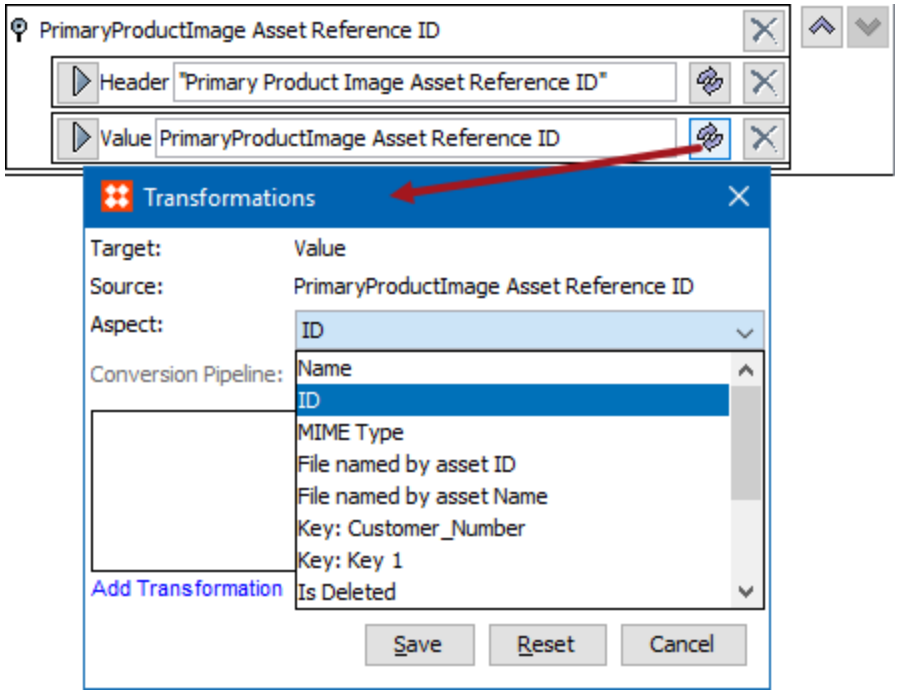
3. Click the right arrow button ( ) in the right panel to add the selected asset reference type.



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, see the Inheritance in 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, see the **Results** section below. For details on the options, see 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.

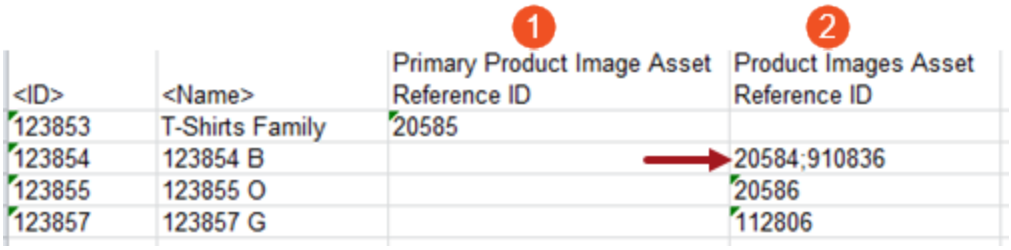
5. Apply any transformations necessary to change the output without changing the original data. See the **Outbound Map Data - Transform** topic for details.
6. Complete any additional mappings and initiate the export.

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.



<ID>	<Name>	Primary Product Image Asset Reference ID	Product Images Asset Reference ID
123853	T-Shirts Family	20585	
123854	123854 B		20584;910836
123855	123855 O		20586
123857	123857 G		112806

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, see the **MIME Types** section of the **System Setup / Super User Guide** 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

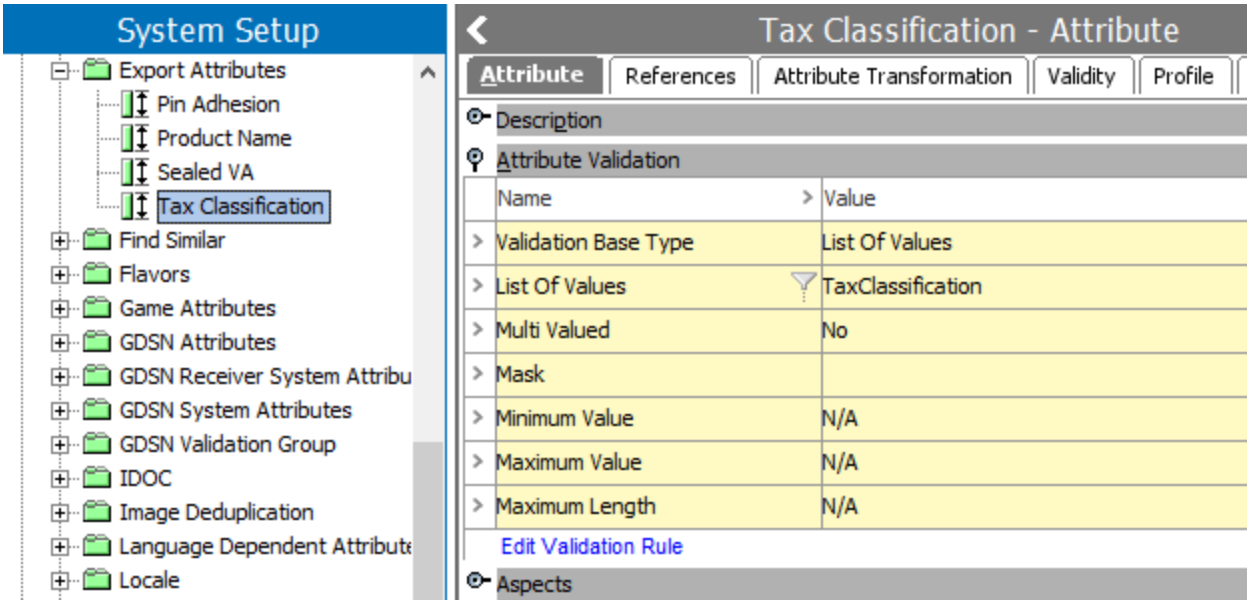
When the 'File named by asset Name' aspect is selected, multiple files are output in a .ZIP file:

- 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, see 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.

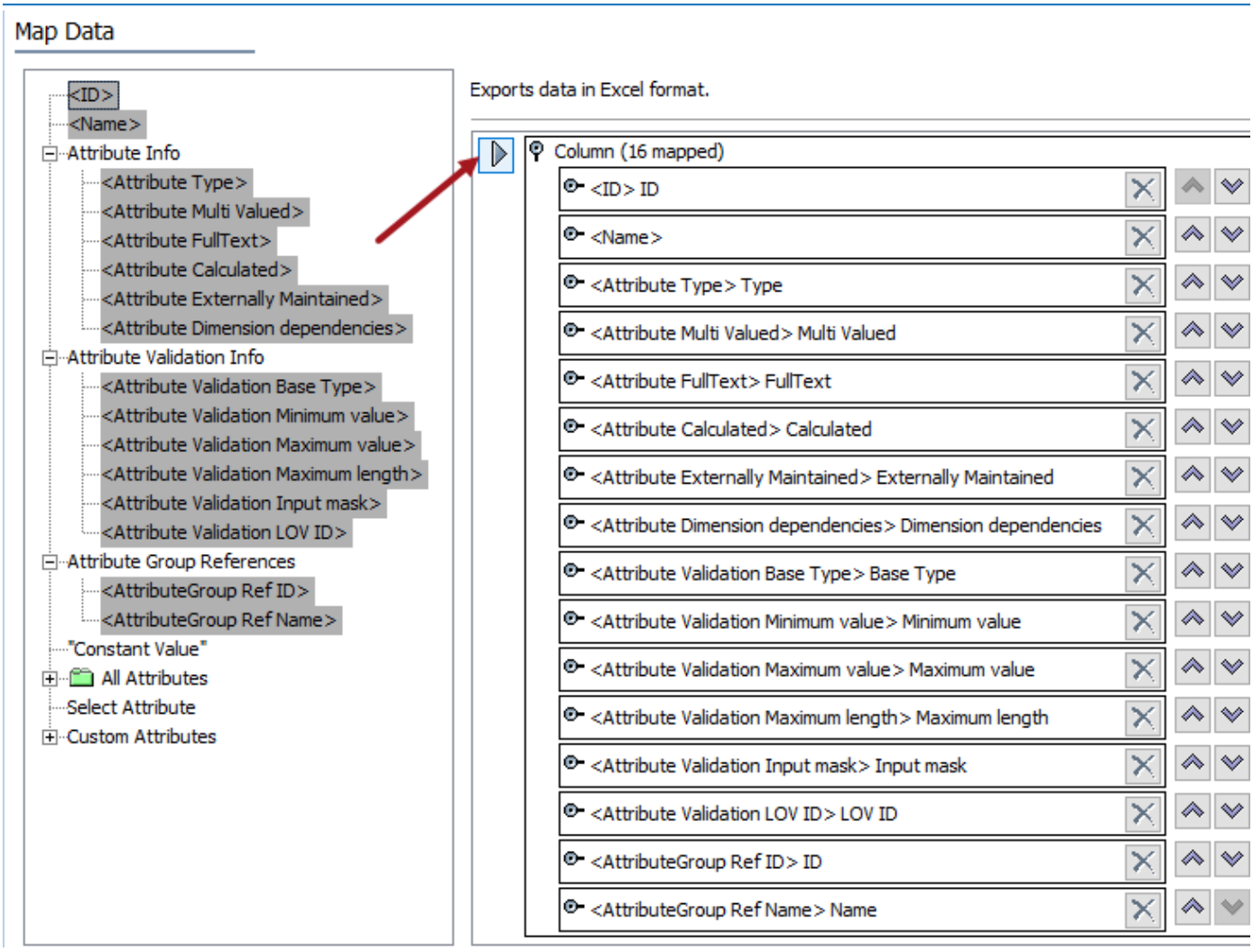
For more information about the additional wizard steps, see **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

After mapping, most output options can be altered using transformations. For more information, see the **Outbound Map Data - Transform** documentation.

Mapping Attribute Details

1. Select the data to be exported for the output tool. For more information, see the **Data Exchange** topic.
2. Select the attribute super object type based on your output tool. For more information, see 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. See 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

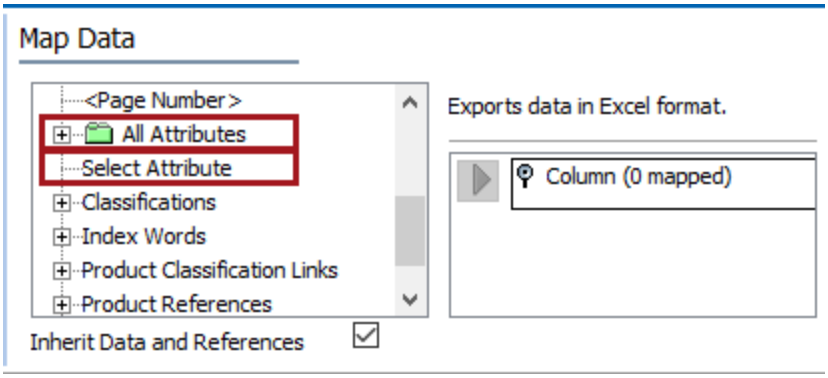
Attributes and data containers can be selected and exported for the objects being exported.

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, see 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, see **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

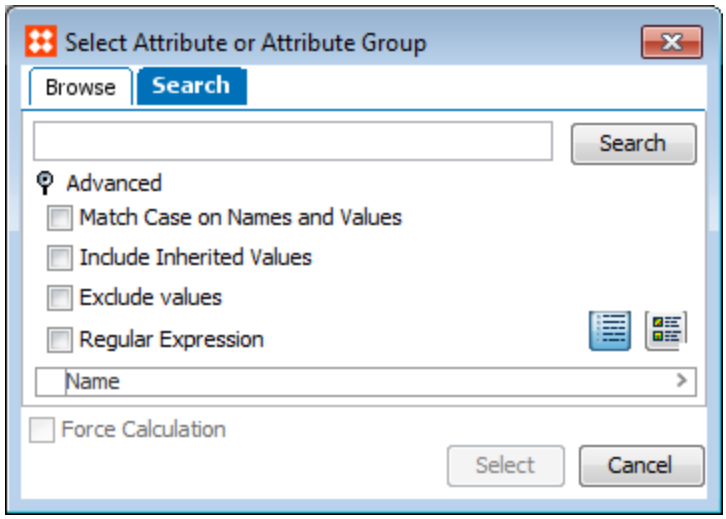
After mapping, most output options can be altered using transformations. For more information, see the **Outbound Map Data - Transform** documentation.

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, see 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, see the **Search: Advanced Options** topic in the **Getting Started / User Guide** 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. See the **Aspect - Transform Outbound** topic.
8. Complete any additional mappings and initiate the export.

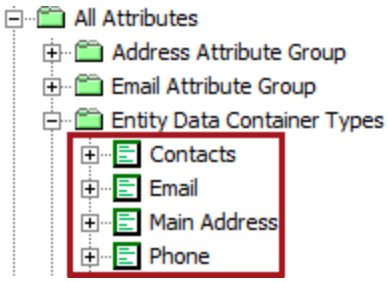
Mapping Attributes or Attribute Groups via All Attributes

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, see 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. See the **Aspect - Transform Outbound** topic.
5. Complete any additional mappings and initiate the export.

Mapping Entity Data Containers via All Attributes

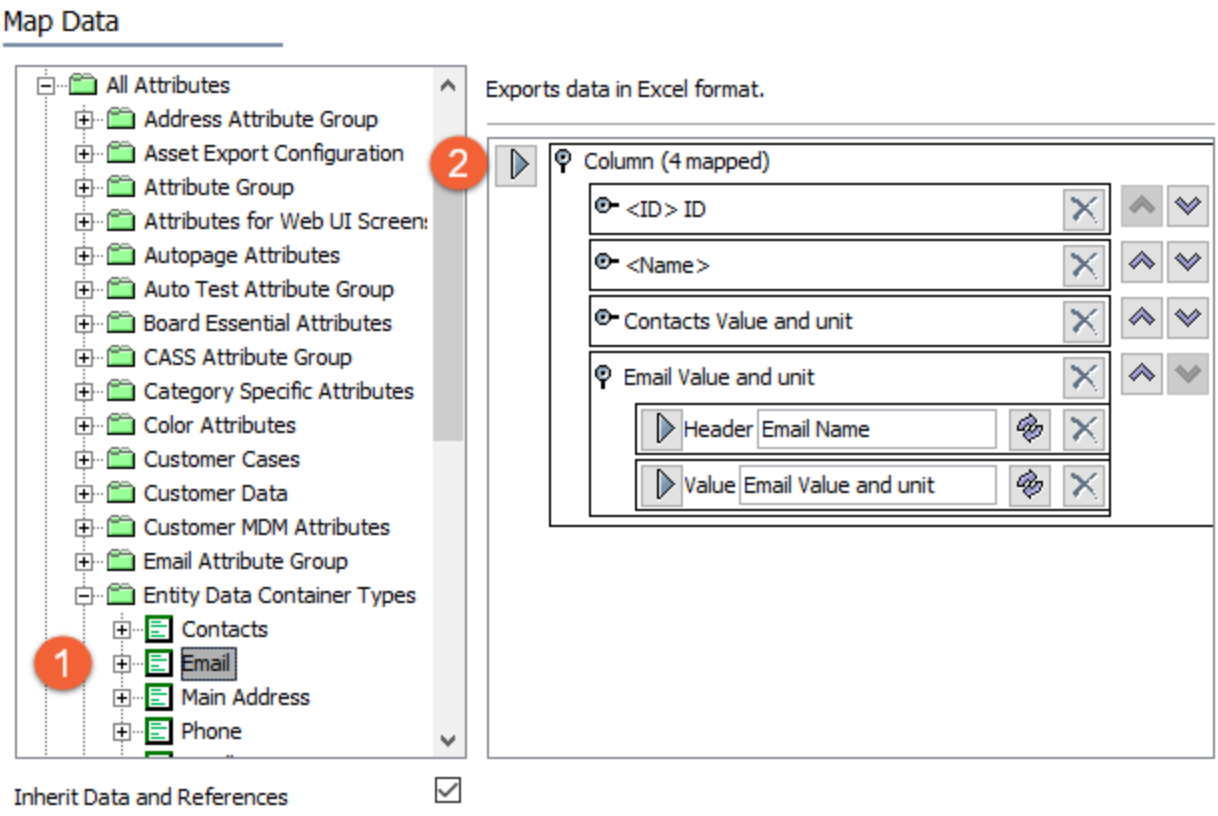
For exporting entity data containers into CSV or Excel formats (which require mapping), only **All Attributes** can be used. Using this option, data containers can be mapped directly to the export or included in a higher level selection.



For information on exporting entity and product data containers without mapping, see 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, see 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. See the **Outbound Map Data - Transform** topic.
6. Complete any additional mappings and initiate the export.

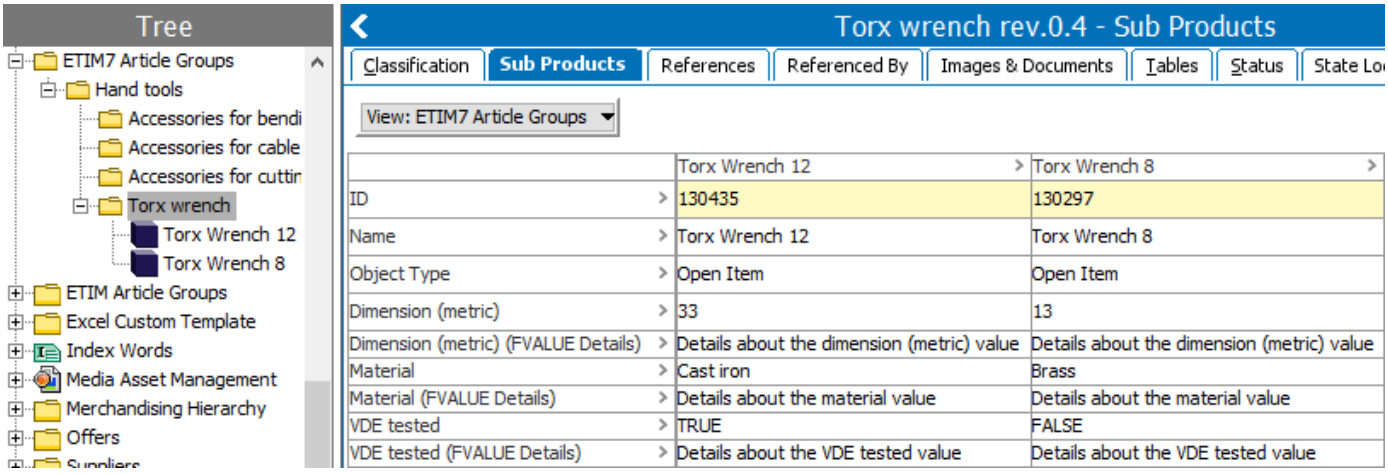
For information on handling entity data container transformations during export, see the **Data Container Separator - Transform Outbound** section of the **Data Exchange** documentation.

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.

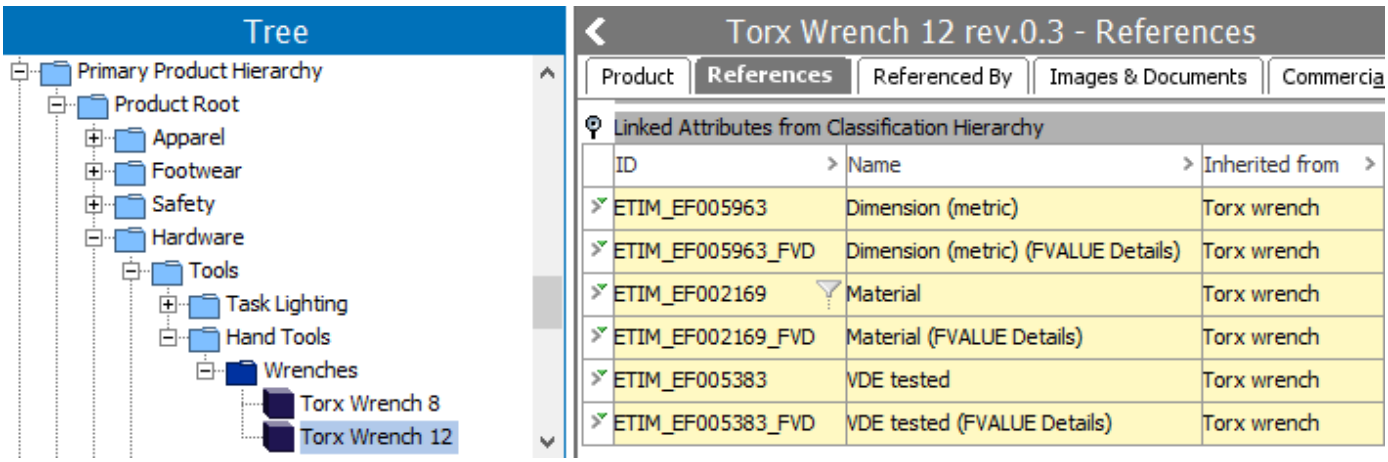
The option to map attributes inherited from a classification extracts attributes that are inherited from a selected classification (or child classification of the selected classification) to the product. So instead of mapping to an attribute group, you map to an ETIM classification, and the system analyzes 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 seen 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, see **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

After mapping, most output options can be altered using transformations. For more information, see the **Outbound Map Data - Transform** documentation.

Limitations

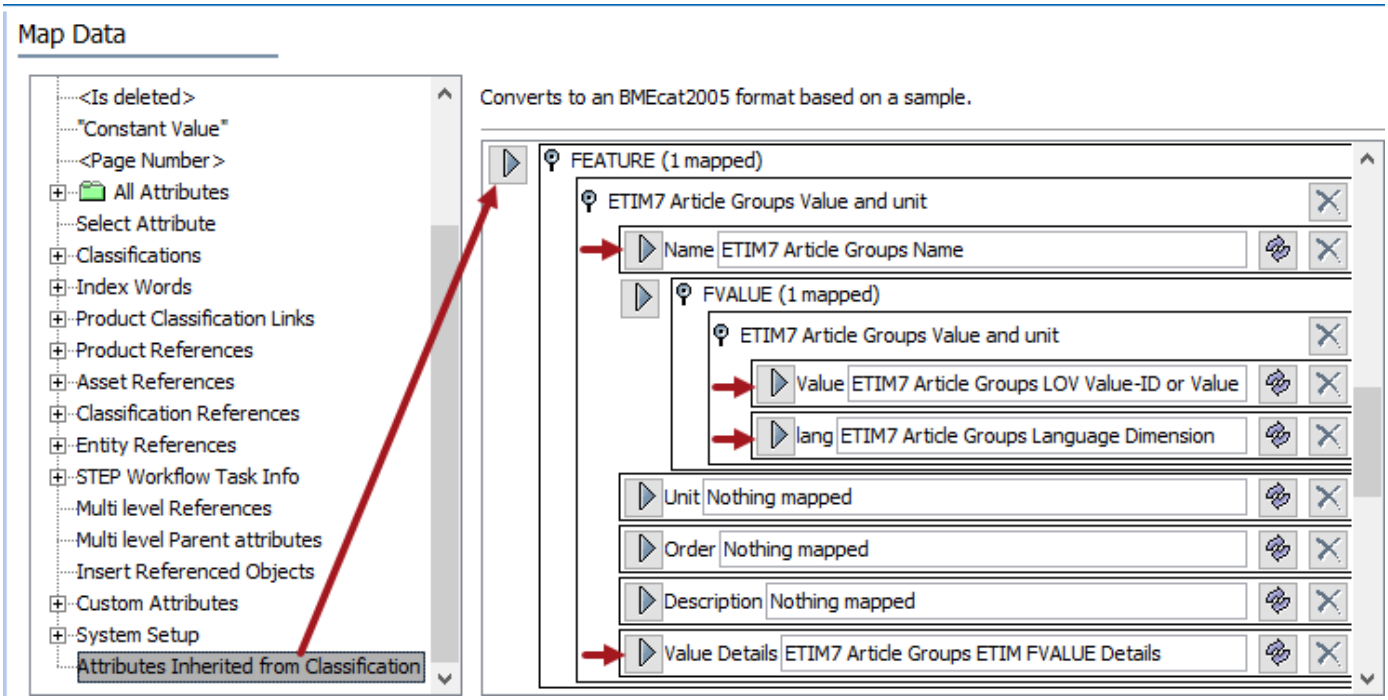
For ETIM product data, the data source can only be used to extract the following sub-fields in the FEATURE target via aspect parameters:

- 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, see 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 , see 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. See 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, see 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, see **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

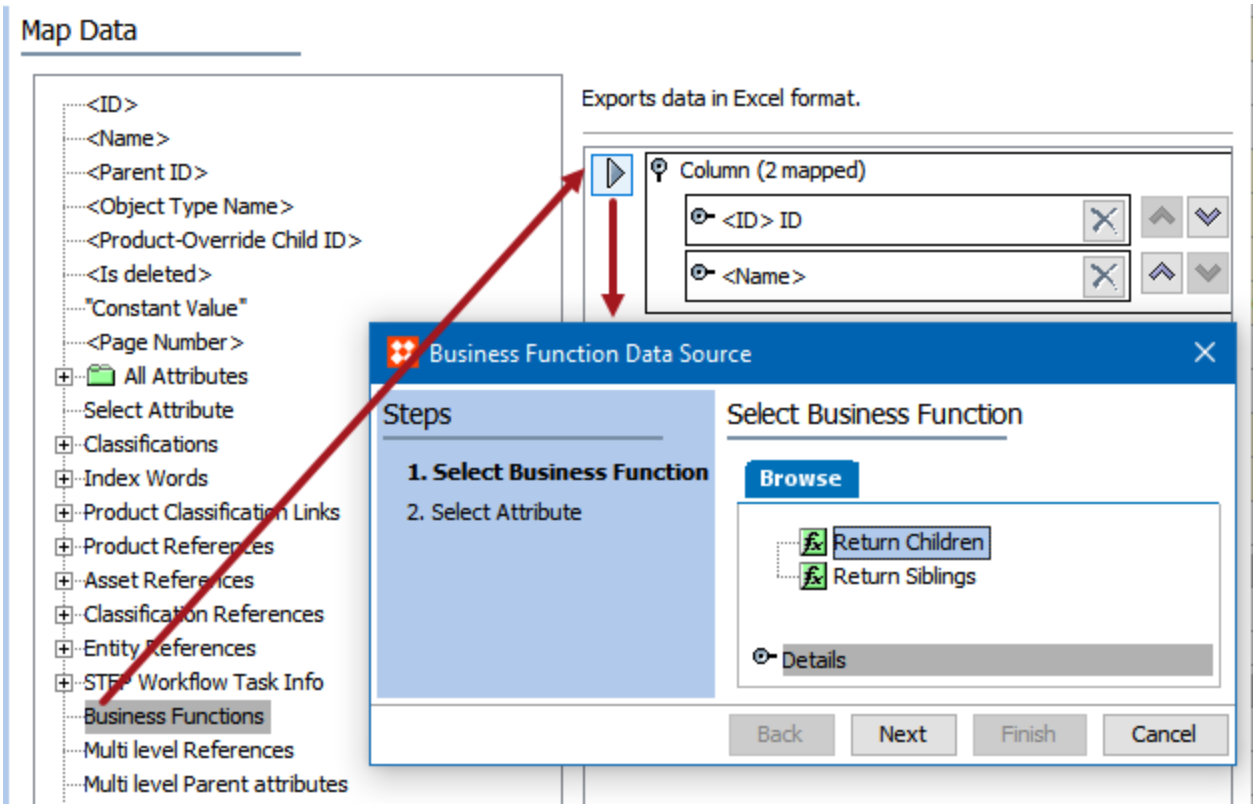
After mapping, most output options can be altered using transformations. For more information, see the **Outbound Map Data - Transform** documentation.

1. Select the data to be exported for the output tool. For more information, see 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.

   ---

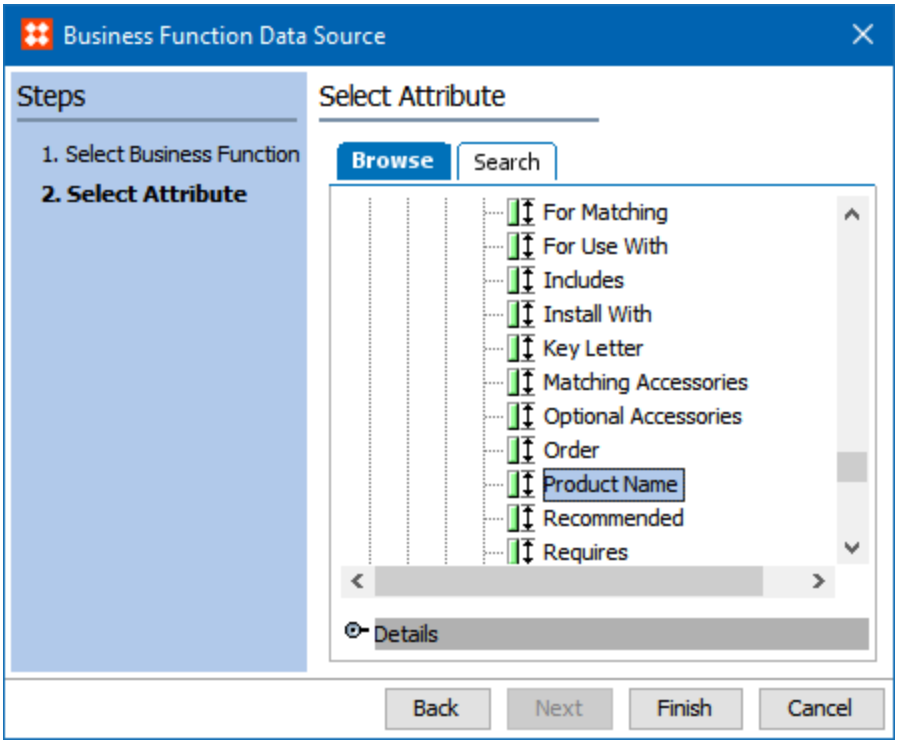
3. In the right panel, click the right arrow icon to display the **Business Function Data Source** wizard.



4. On the **Select Business Function** screen, select the relevant business function, then click **Next**.

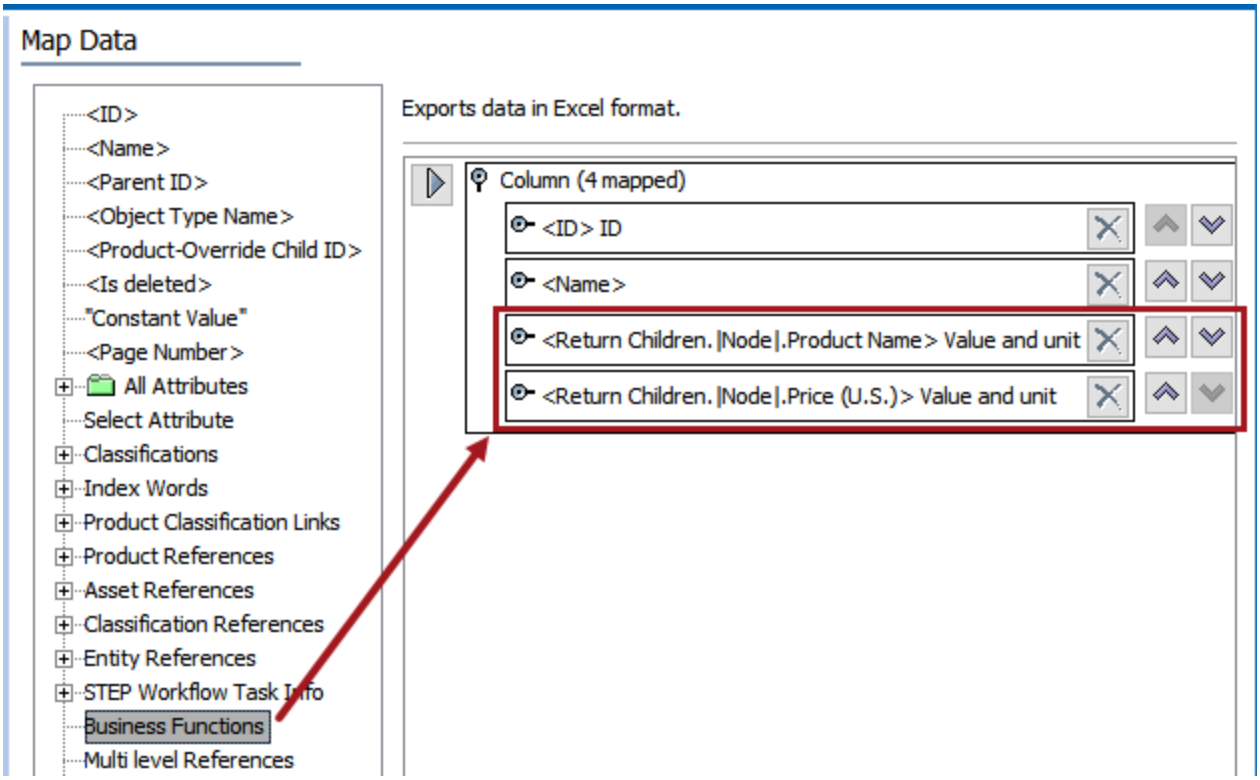
   Note: Only business functions that have **Node** as input and either **Node** or **List of Nodes** as output are valid for this functionality. As such, only these business functions display in the wizard.

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 / 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., **;**.

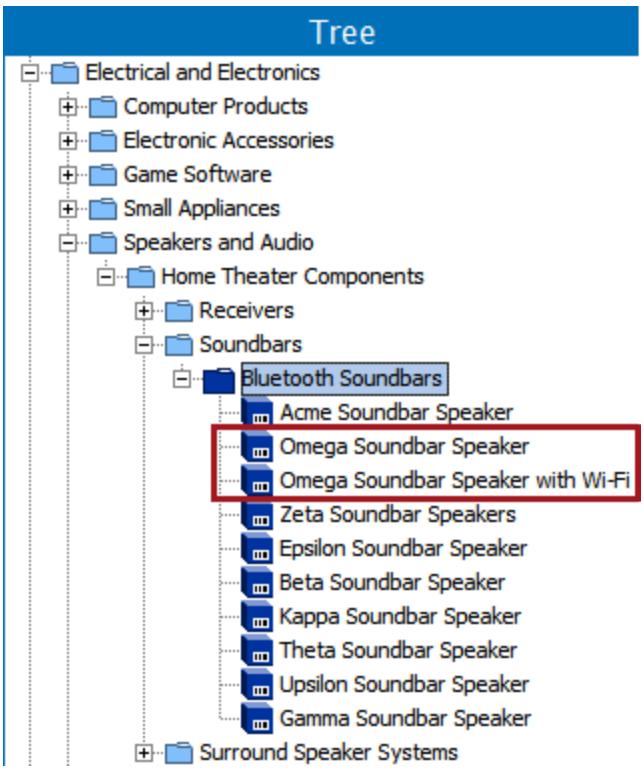


Example Use Cases and Results

Two sample use cases for this functionality are as follows:

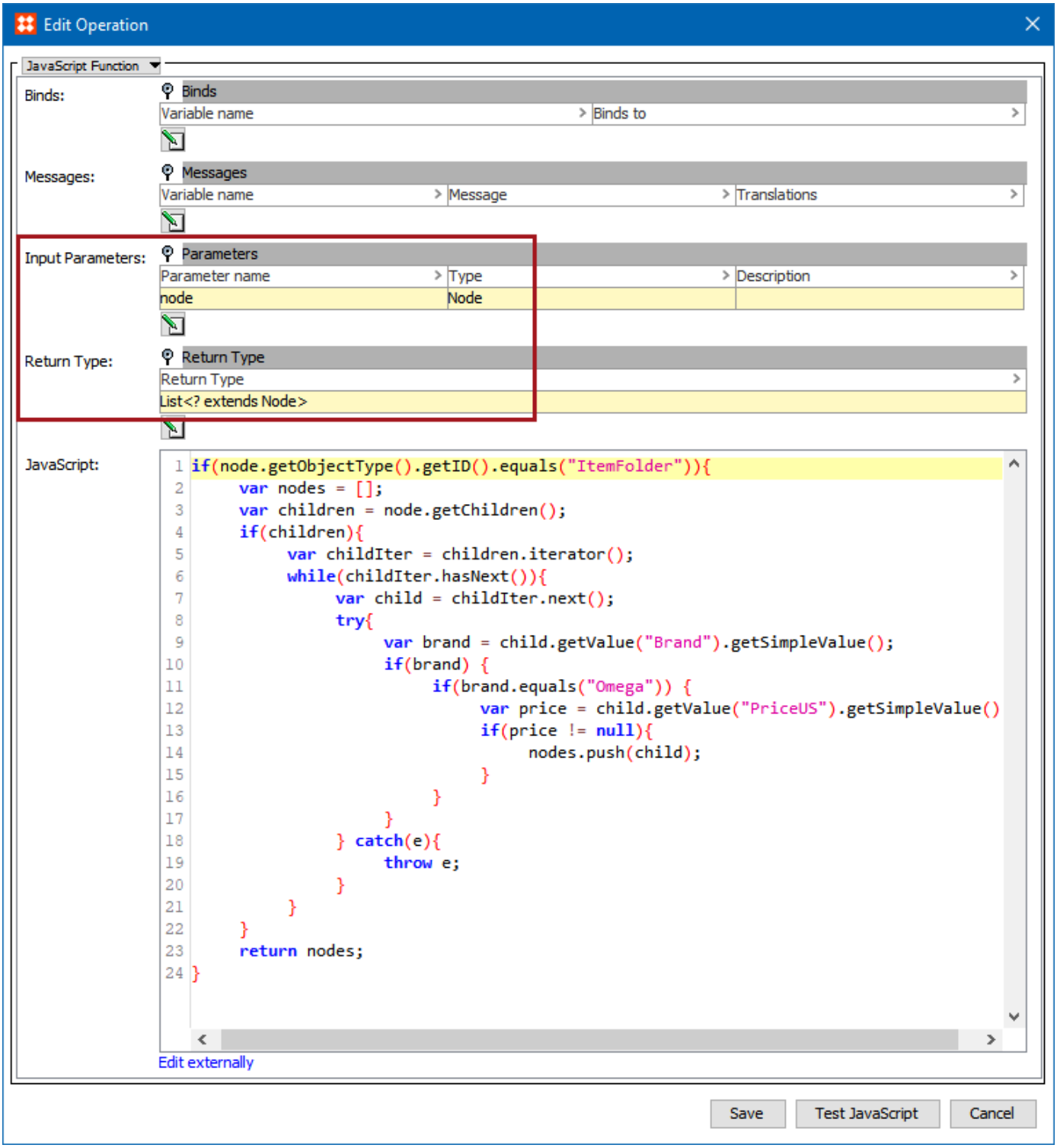
- Return a list of *child* product objects where the 'Brand' attribute equals a specified value (e.g., Brand = Omega). If a price value also exists for these objects, return the attribute value(s) from the node(s) to be used in an export.
- Return a list of *sibling* product objects where Brand = Omega. 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.



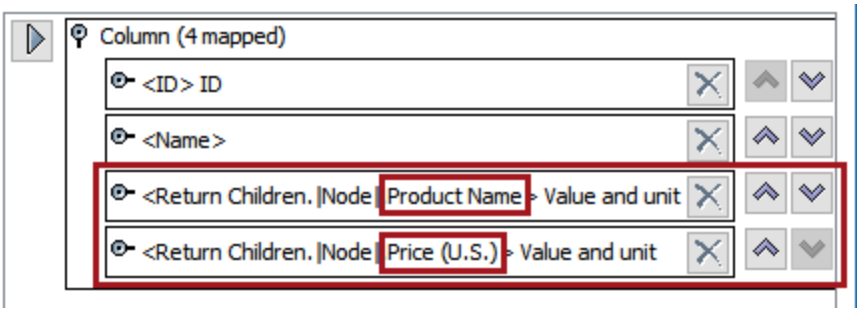
The code used in the above example is:

```
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.).

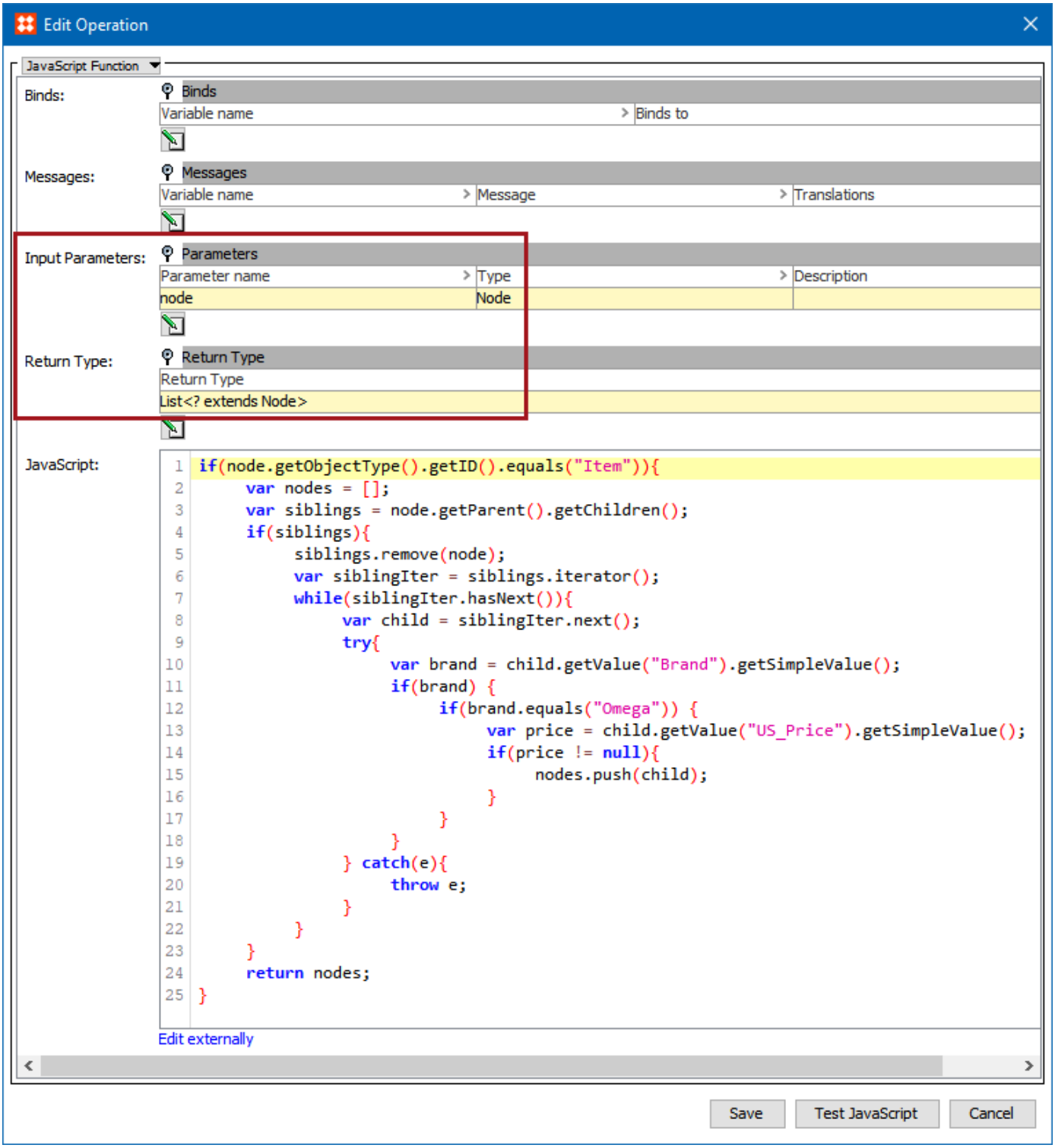


In the below example, the export has been performed from the Item Folder level (Bluetooth Soundbars) and has returned the Product Name and Price (U.S.) of the two Omega-branded children.

	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.



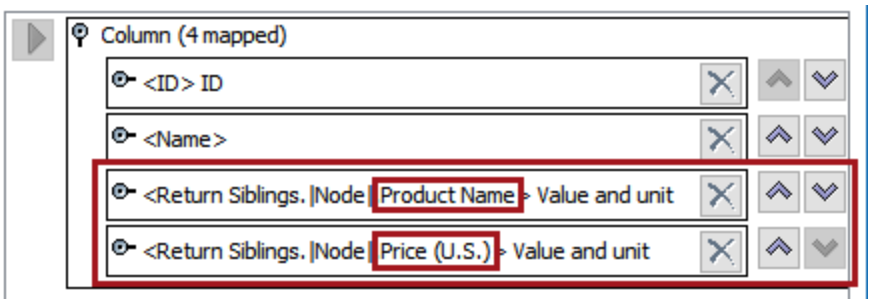
The code used in the above example is:

```
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.).



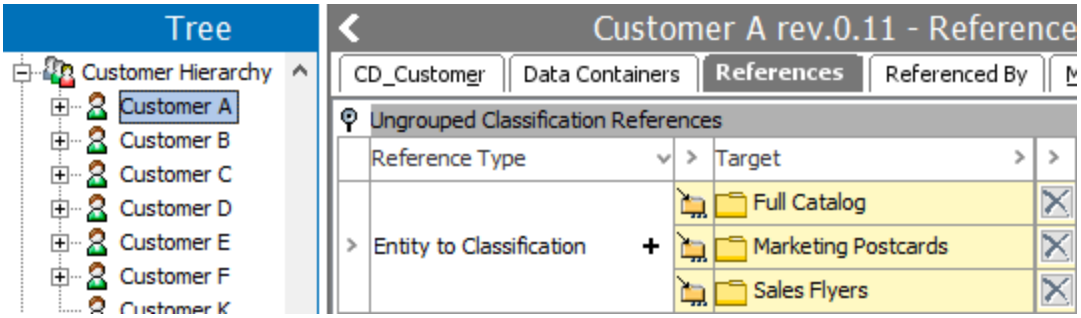
In the below example, the export has been performed from the Item level (Acme Soundbar Speaker) and has returned the Product Name and Price (U.S.) of the two Omega-branded siblings.

	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.



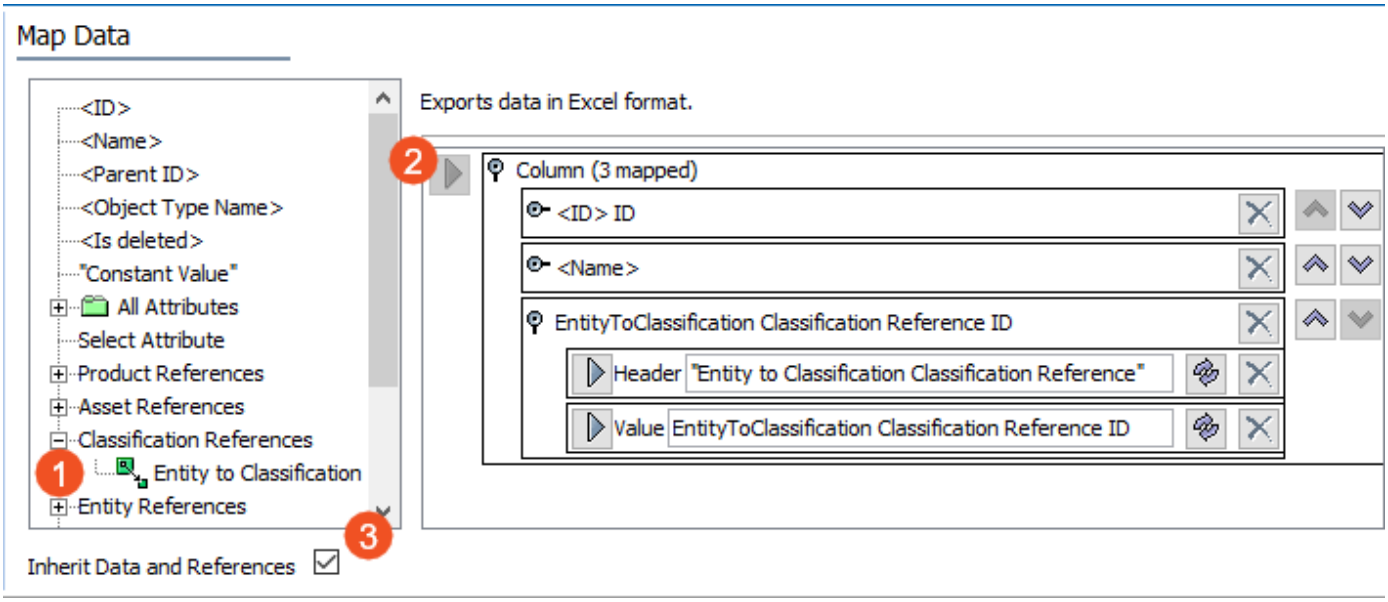
The user interface for the mapping step of the Export Manager tool and the Outbound Integration Endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter 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, see **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

After mapping, most output options can be altered using transformations. For more information, see the **Outbound Map Data - Transform** documentation.

Mapping Classification References

1. Select the data to be exported for the output tool. For more information, see 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, see the Inheritance in the **Inherit Data and References - Data Source Outbound** topic.
5. Apply any transformations, which can change the output without changing the original data. See the **Outbound Map Data - Transform** topic.
6. Complete any additional mappings and initiate the export.

Results

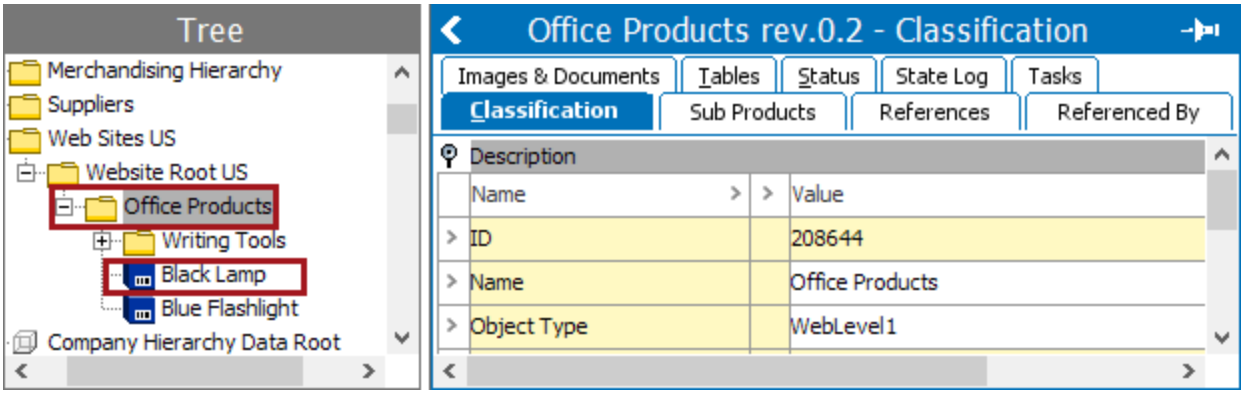
The output includes the classification reference type name as a header and the ID for each row.

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.

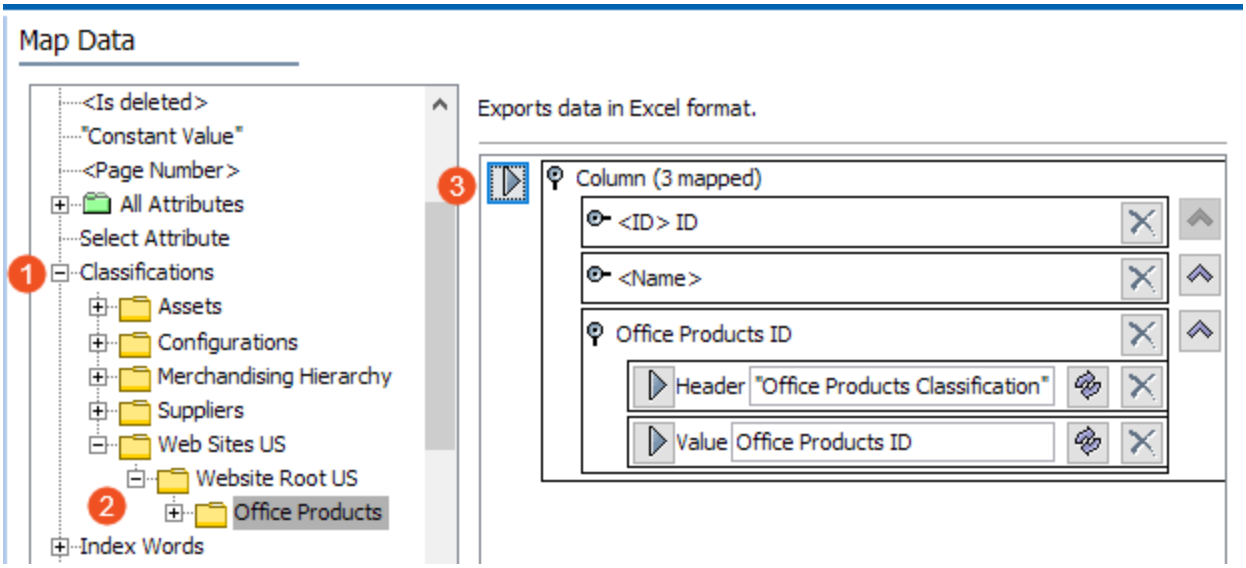
For more information about the additional wizard steps, see **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

After mapping, most output options can be altered using transformations. For more information, see the **Outbound Map Data - Transform** documentation.

Mapping Classifications

1. Select the data to be exported for the output tool. For more information, see 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. See 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.

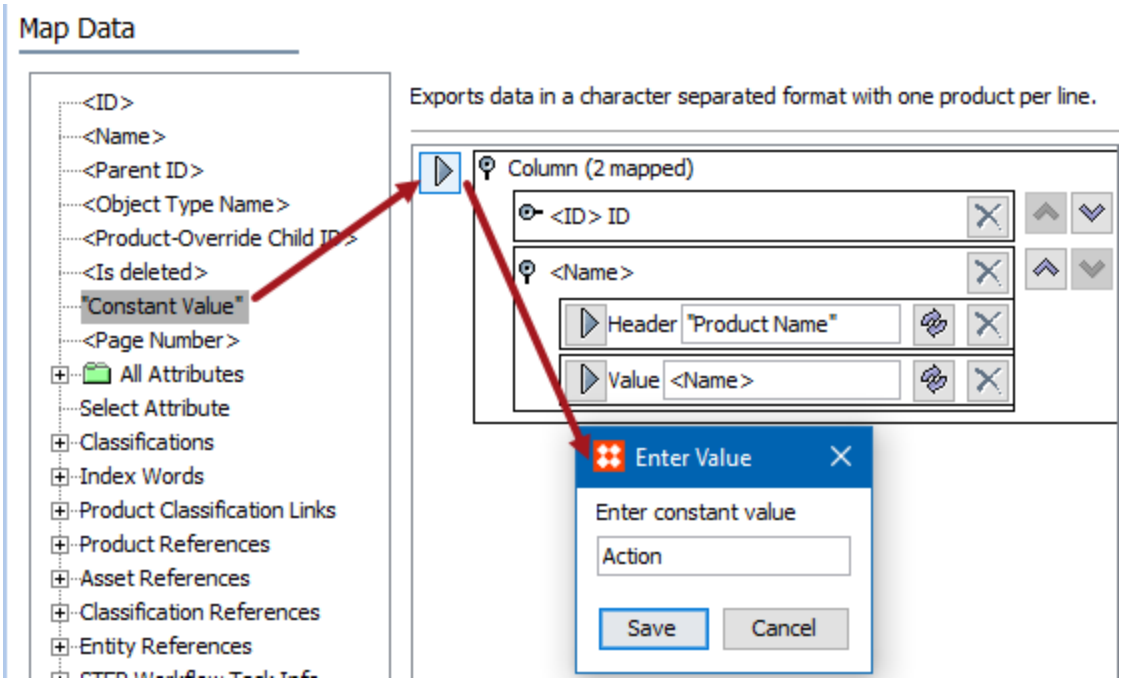
The user interface for the mapping step of the Export Manager tool and the Outbound Integration Endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter 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, see **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

After mapping, most output options can be altered using transformations. For more information, see the **Outbound Map Data - Transform** documentation.

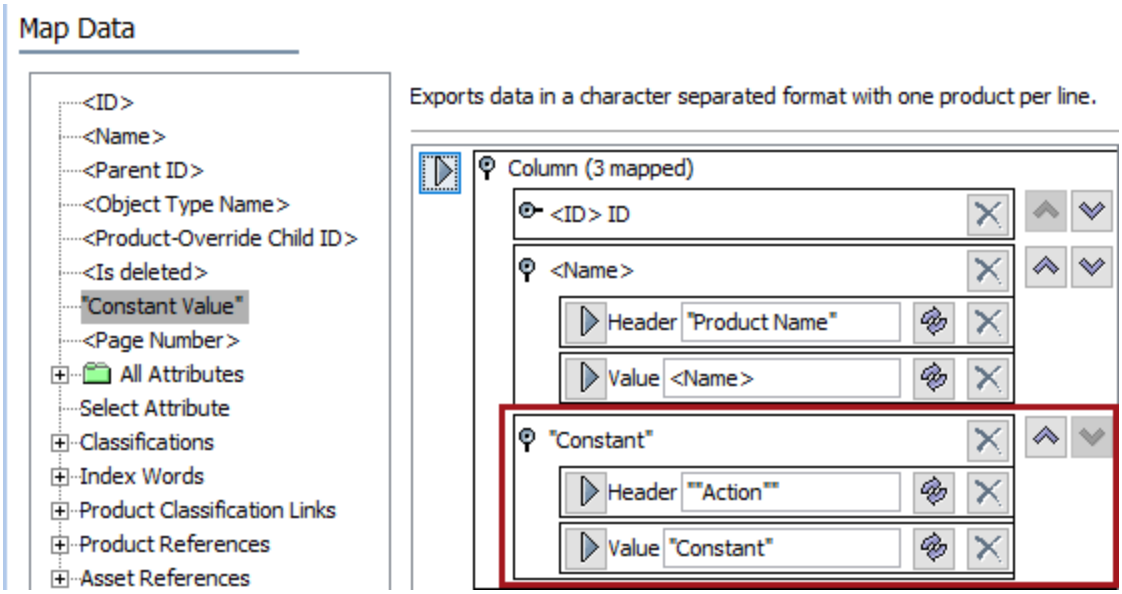
For an expanded example of mapping aspects and constants, see the **Header and Value Aspects Example** documentation.

Mapping Constant Value as a New Field

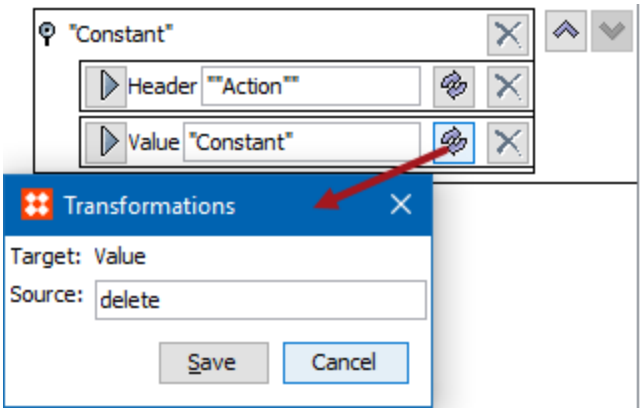


1. Select the data to be exported for the output tool. For more information, see the **Data Exchange** topic.
2. On the Map Data step, in the left panel, select **Constant Value**.
3. In the right panel, click the top-most right arrow ( ) to add a new field to the export file.

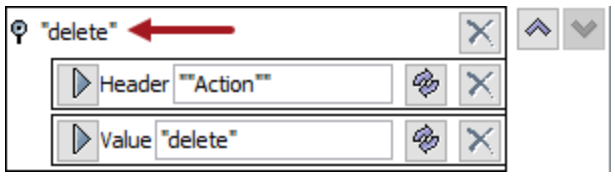
4. In the **Enter Value** dialog, enter the constant value to assign to all records in the export file, and click **Save**.
5. The new "Constant" filed is displayed using the text you entered as the Header.



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.



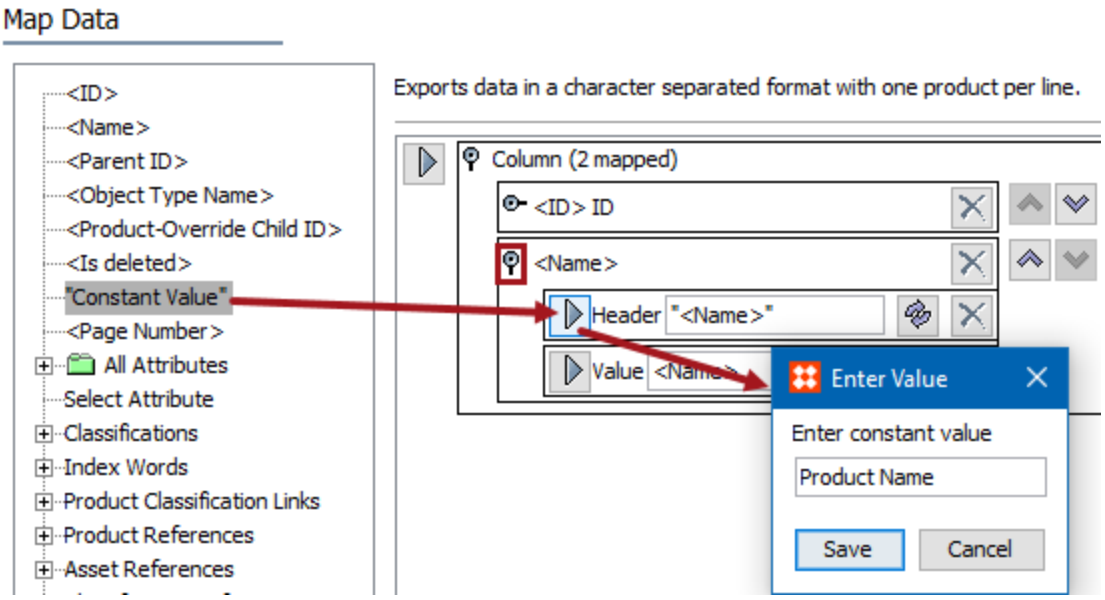
8. Complete the mapping and initiate the export.

Results

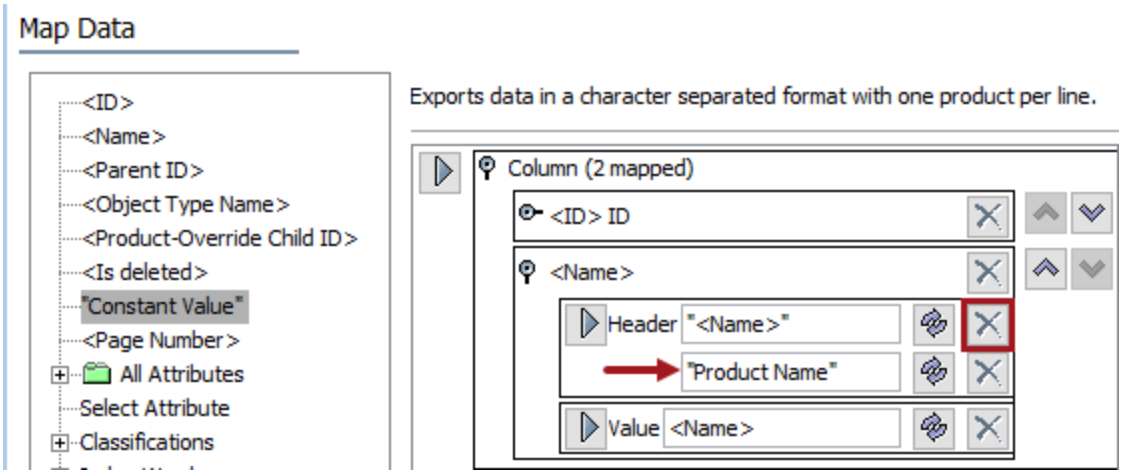
The exported file includes ID, product name and an action column with constant value “delete”.

<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.

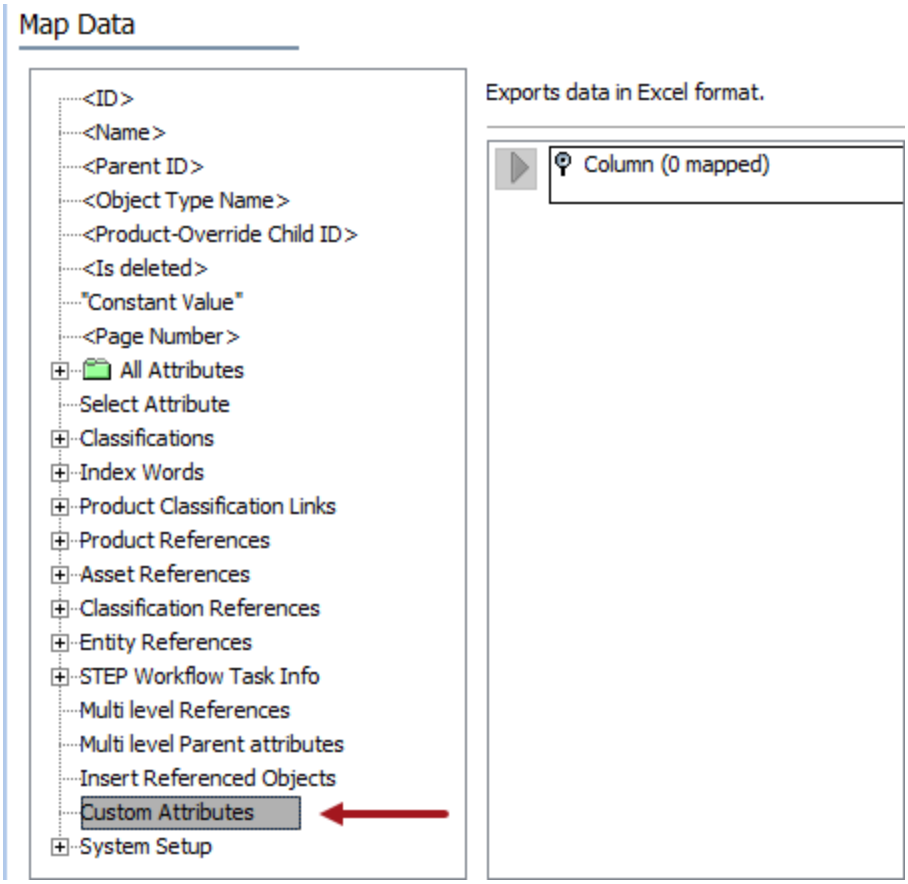
Results

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, see **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

After mapping, most output options can be altered using transformations. For more information, see the **Outbound Map Data - Transform** documentation.

Data Path - Data Source Outbound

The Data Path data source is only available with the Generic XML and IDoc MATMAS 05 format. The following mapping examples use the 'Data Path' data source:

- Nestable Data Paths in Generic XML
- Nestable Data Paths in Generic XML Example
- 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, see **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

After mapping, most output options can be altered using transformations. For more information, see the **Outbound Map Data - Transform** documentation.

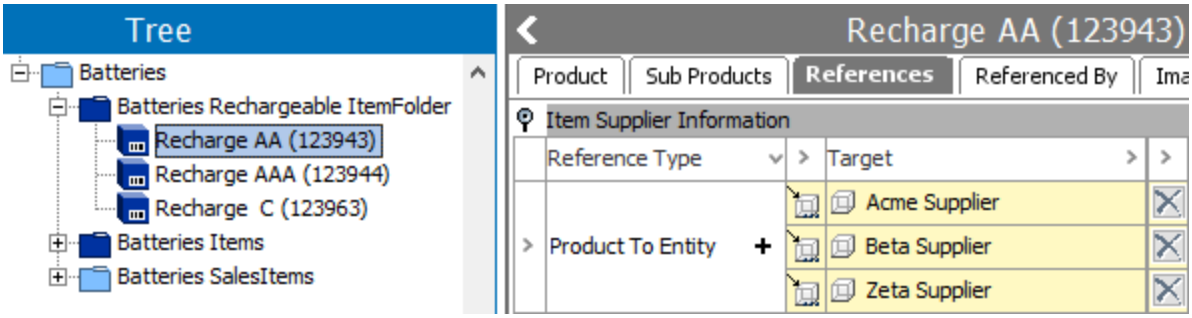
Entity References – Data Source Outbound

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, see 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.



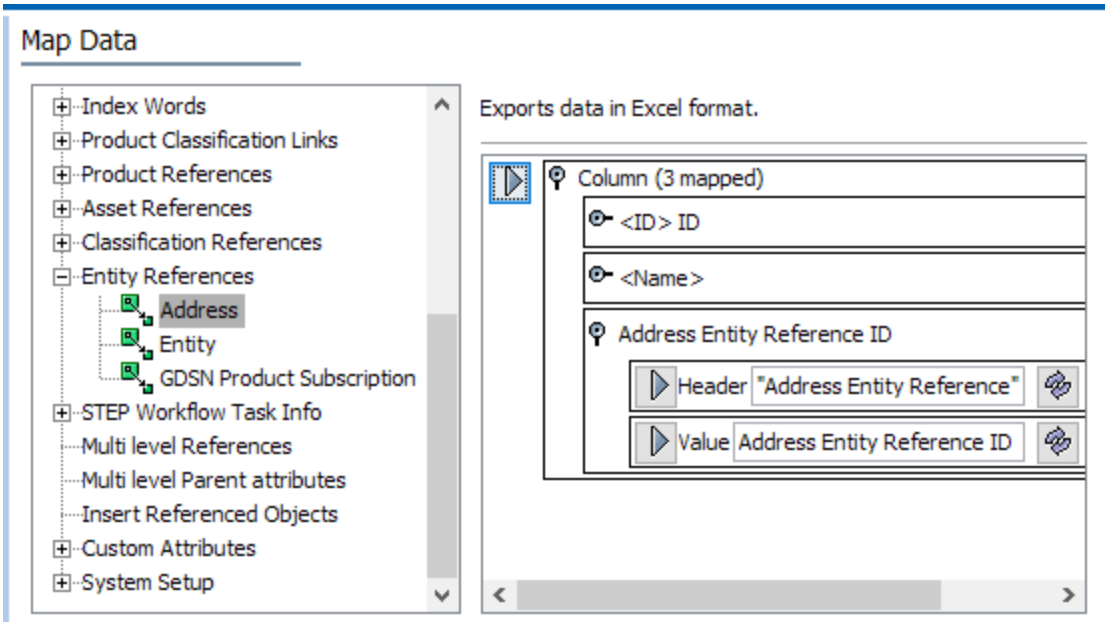
The user interface for the mapping step of the Export Manager tool and the Outbound Integration Endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter 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, see **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

After mapping, most output options can be altered using transformations. For more information, see the **Outbound Map Data - Transform** documentation.

Mapping Entity References

1. Select the data to be exported for the output tool. For more information, see 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. See 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, see the **Unique Keys** section of the **System Setup / Super User Guide** 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.

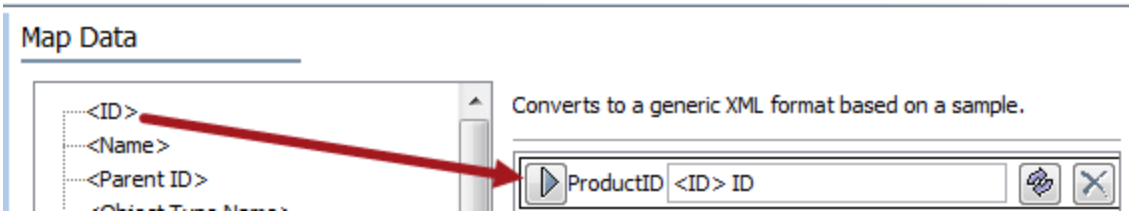
For more information about the additional wizard steps, see **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

After mapping, most output options can be altered using transformations. For more information, see the **Outbound Map Data - Transform** documentation.

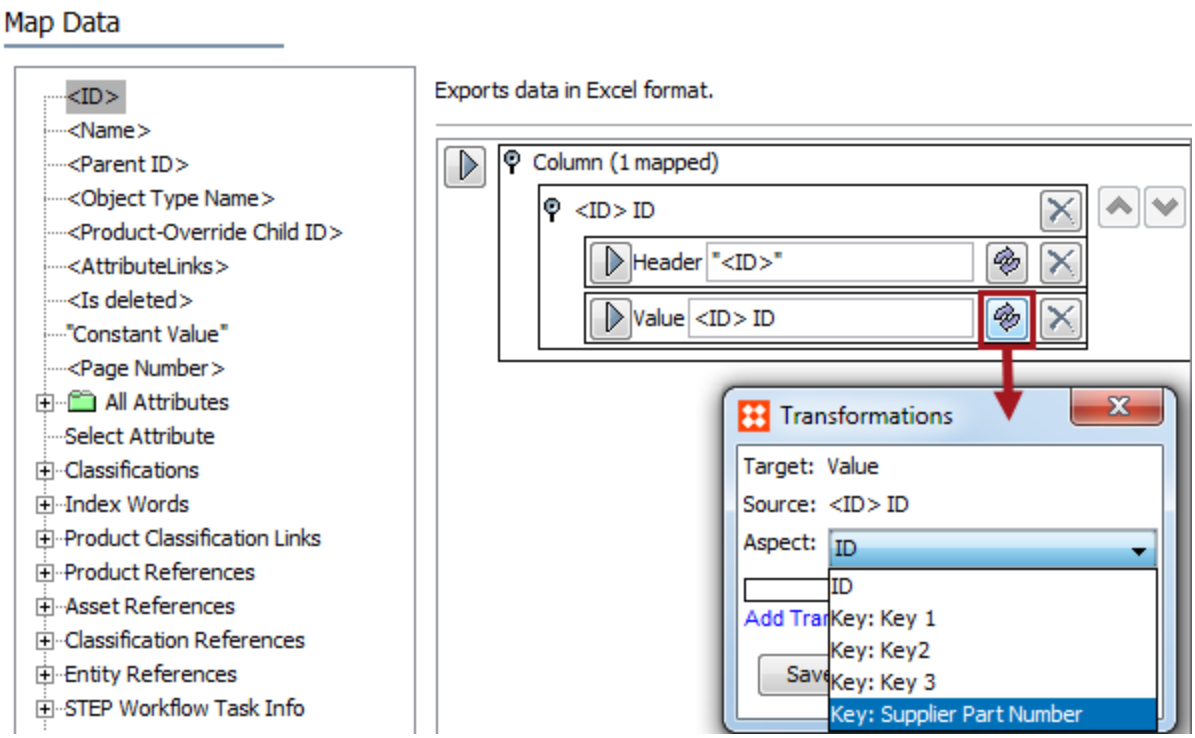
Mapping ID or Key

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, see the **Data Exchange** topic.
2. On the Map Data step, in the left panel, select **<ID>**.



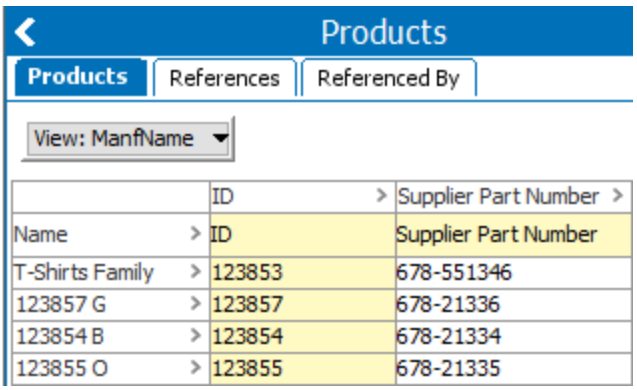
3. Click the right arrow button ( ) in the right panel to add the ID flipper as a column. If mapping the ID, skip to the last step.
4. If mapping a key, open the <ID> flipper, and under **Value**, click the **Transformation** icon ( ). In the **Transformations** dialog, select the relevant key from the **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

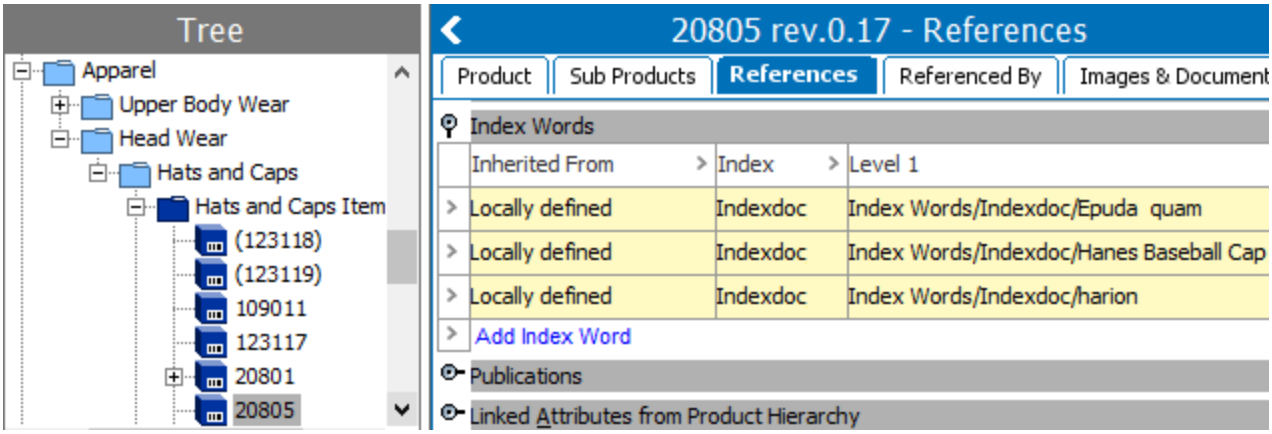
If only the 'Supplier Part Number' active key is mapped, the exported file includes the ID KeyID=SupplierPartNumber as header and its corresponding key values for each of the exported objects.

<ID KeyID=SupplierPartNumber>
678-551346
678-21334
678-21335
678-21336

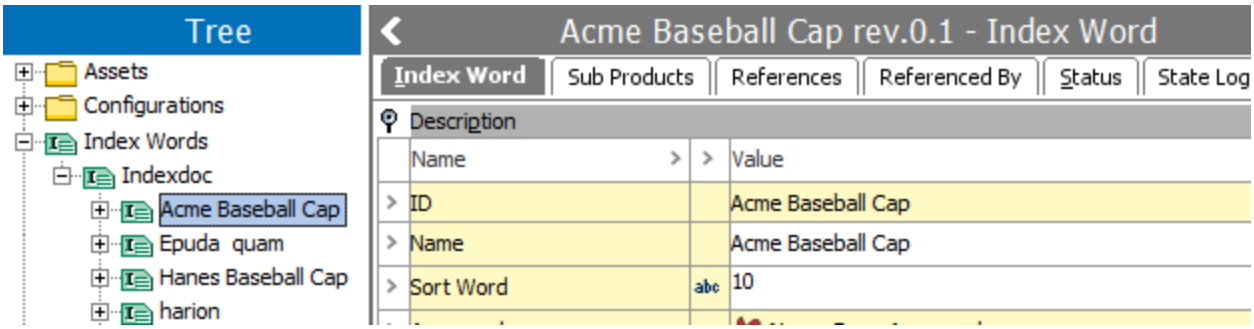
Index Words – Data Source Outbound

The option to map index words is available for an index hierarchy. For more information on indexes, see **Creating Document Indexes** in the **STEP Publisher** 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, see **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

After mapping, most output options can be altered using transformations. For more information, see the **Outbound Map Data - Transform** documentation.

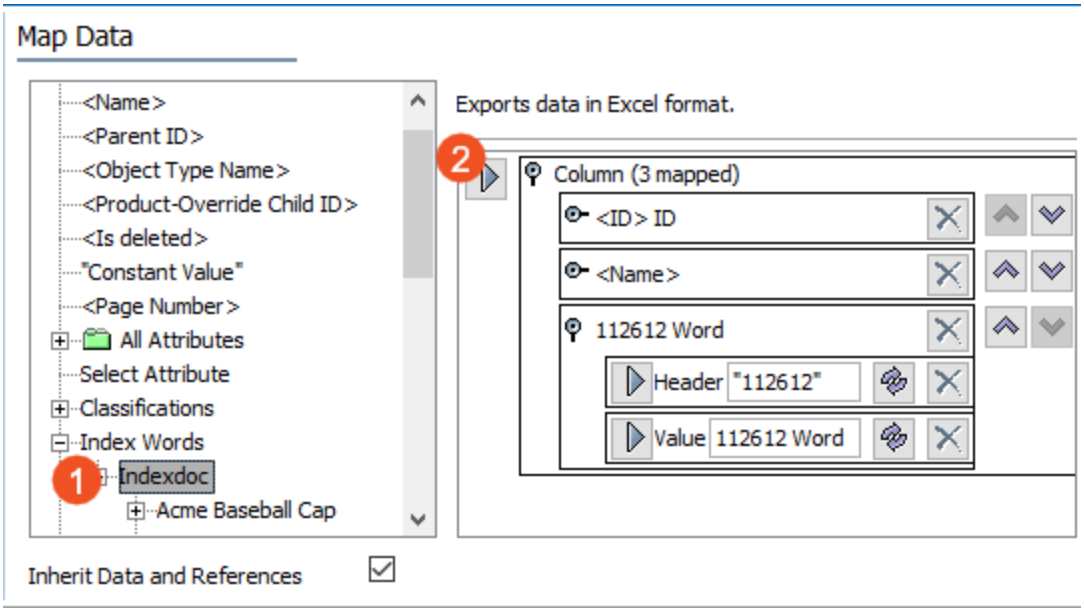
The index words of an index hierarchy can be exported as follows:

- 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, see the **Data Exchange** topic.
2. Select the Product super type. For more information, see **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. See the **Outbound Map Data - Transform** topic.
7. Complete any additional mappings and initiate the export.

Results of Index Words Linked to Products Using Word

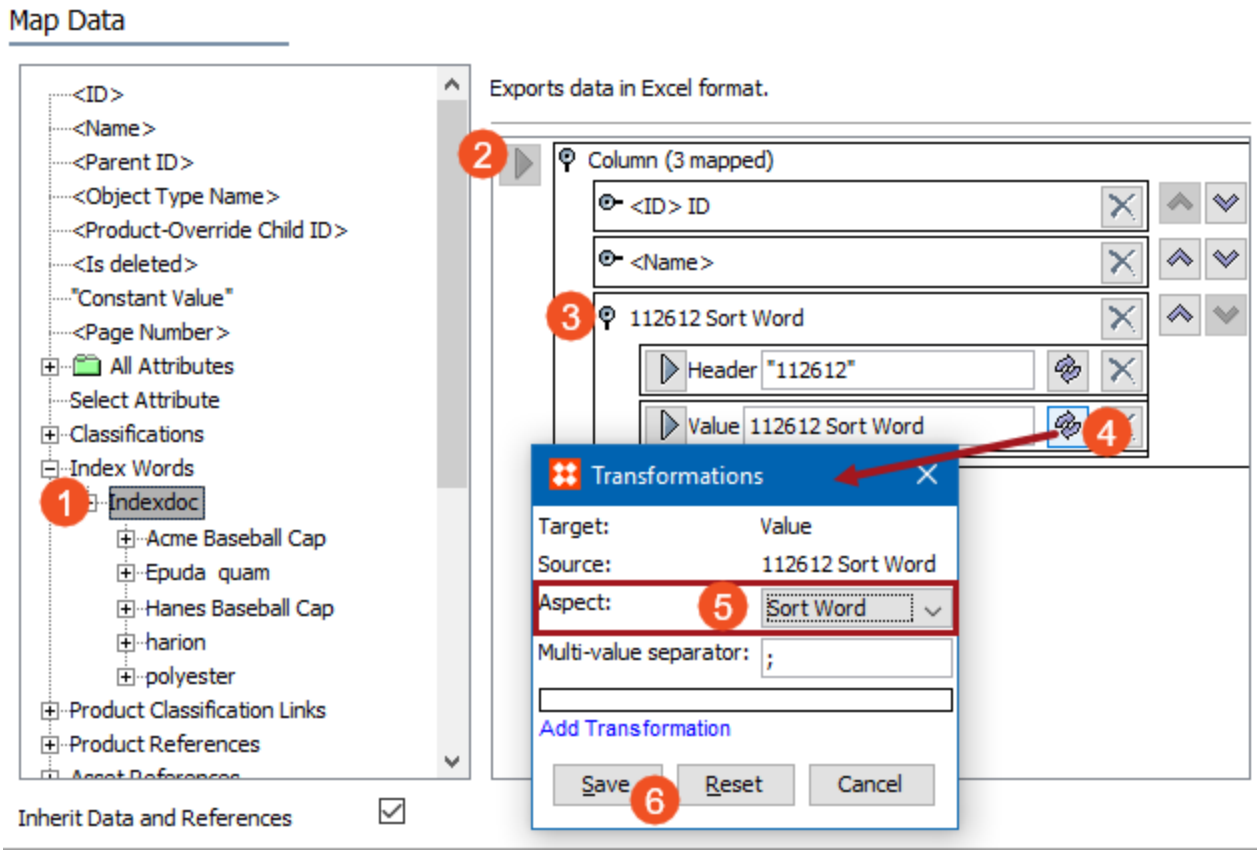
The output includes the ID and name (of product), and index word ID as headers.

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, see the **Data Exchange** topic.
2. Select the Product super type. For more information, see **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

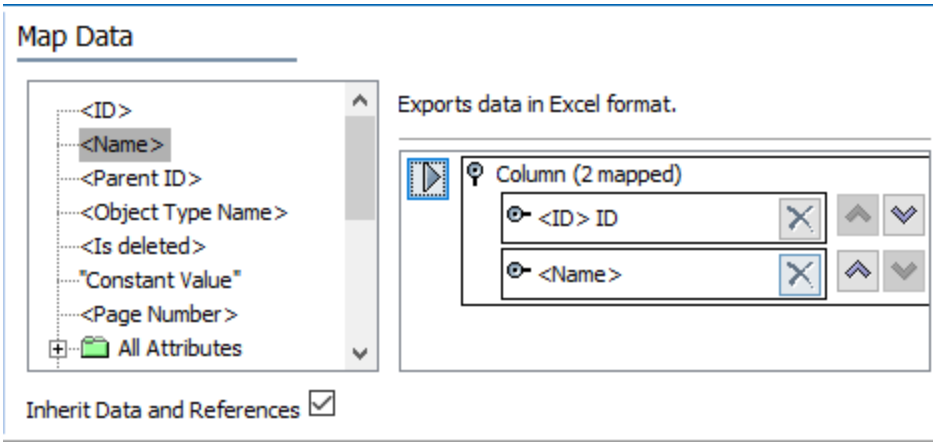
The output includes the ID and name of product, and index word ID as headers. Data includes the sort word attribute value defined on the index word that is linked to the product.

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, see the **Data Exchange** topic.
2. Select the Classification super type. For more information, see **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, see the Inheritance in the **Product Hierarchy** topic in the **Getting Started / User Guide** documentation.

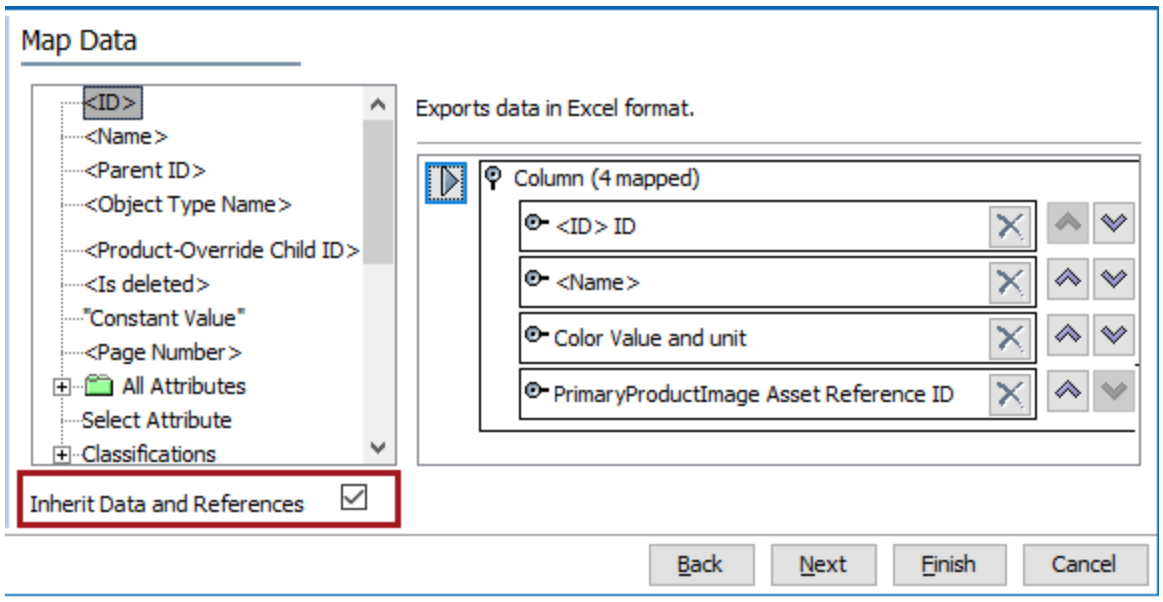
The user interface for the mapping step of the Export Manager tool and the Outbound Integration Endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter 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, see **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

After mapping, most output options can be altered using transformations. For more information, see the **Outbound Map Data - Transform** documentation.

Mapping Inherited Data and References

1. Select the data to be exported for the output tool. For more information, see 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.

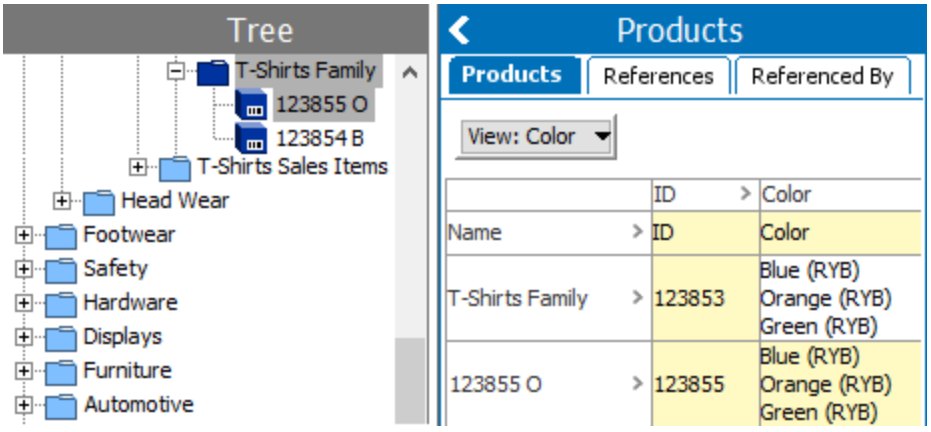


3. Use the necessary data sources to map the required data. For information, see **Outbound Map Data - Data Source**.
4. Initiate the export.

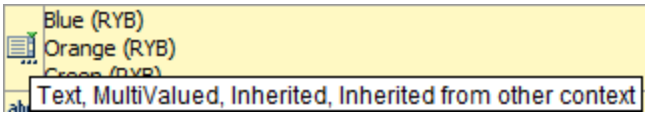
Results

The following data is exported with the results displayed below, based on the 'Inherit Data and References' parameter.

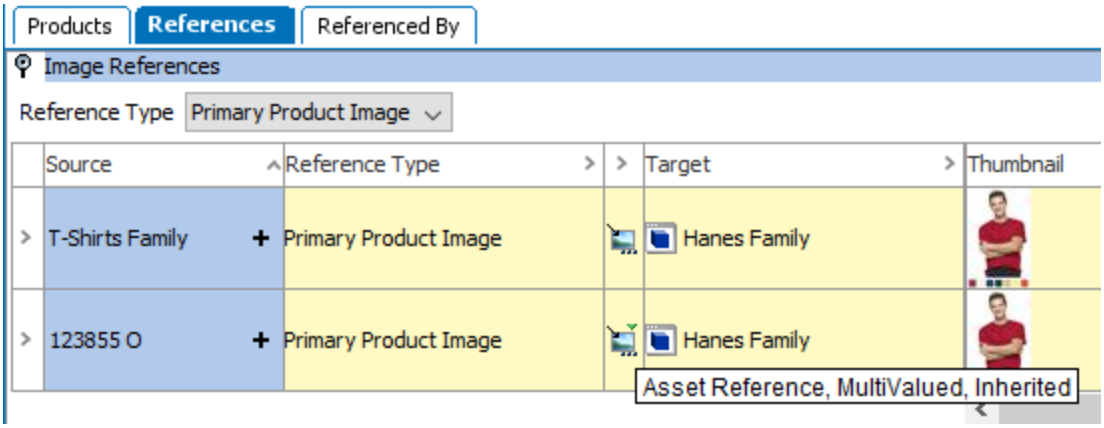
The Color attribute has a local value on the family object, and the values are inherited to the same attribute of the child object. The following shows a view created to show the values available for both the parent and the child.



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 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, in order to see that 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.

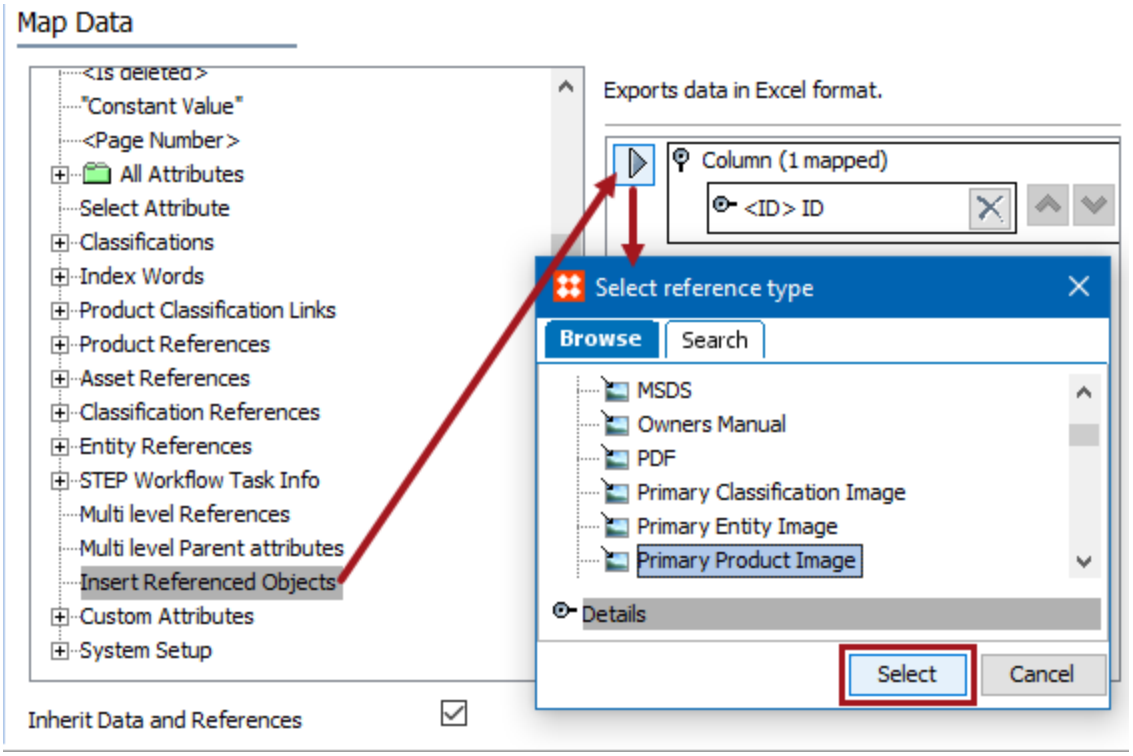
For more information about the additional wizard steps, see **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

After mapping, most output options can be altered using transformations. For more information, see the **Outbound Map Data - Transform** documentation.

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, see 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.

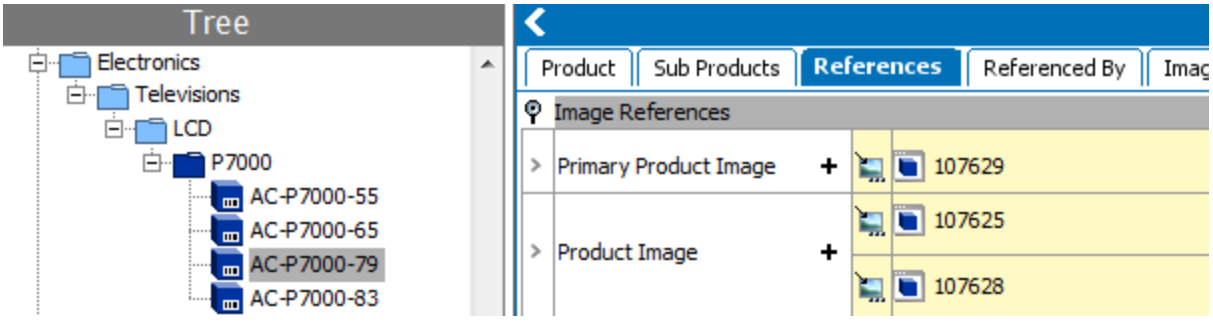


3. Check the **Inherit Data and References** option appropriately:
   - If **checked**, inherited data and references are included in the export.
   - If **unchecked**, inherited data and references is not included in the export.

   For information on inheritance, see the Inheritance in 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. See the **Aspect - Transform Outbound** topic. Example outputs are included in the **Results** section below.
5. Complete any additional mappings and initiate the export.

Results

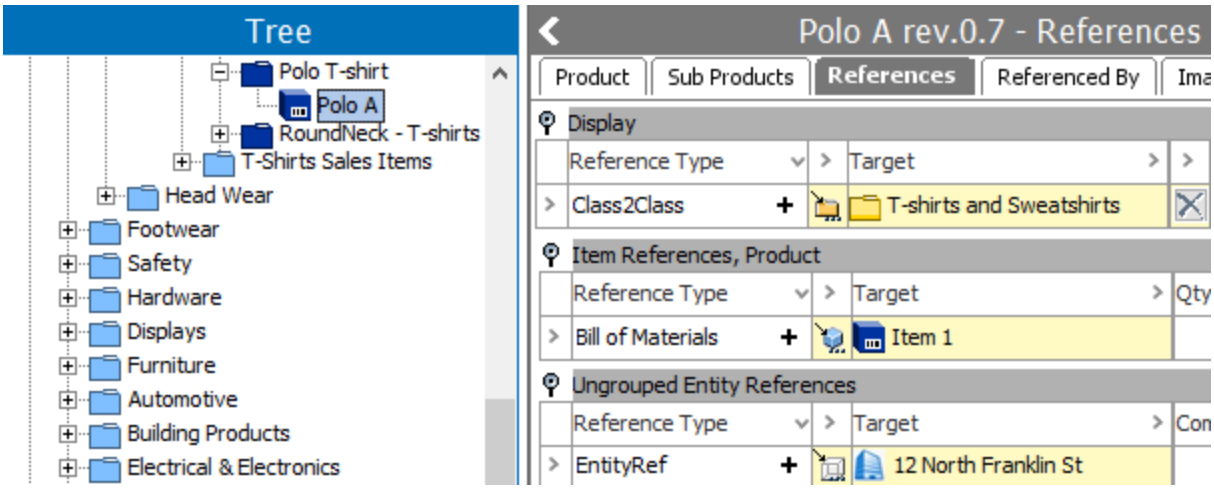
In this example of Asset References, the product being exported has three references to images:



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

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, see 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.

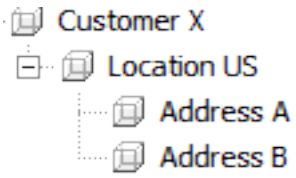
For more information about the additional wizard steps, see **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

After mapping, most output options can be altered using transformations. For more information, see the **Outbound Map Data - Transform** documentation.

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.

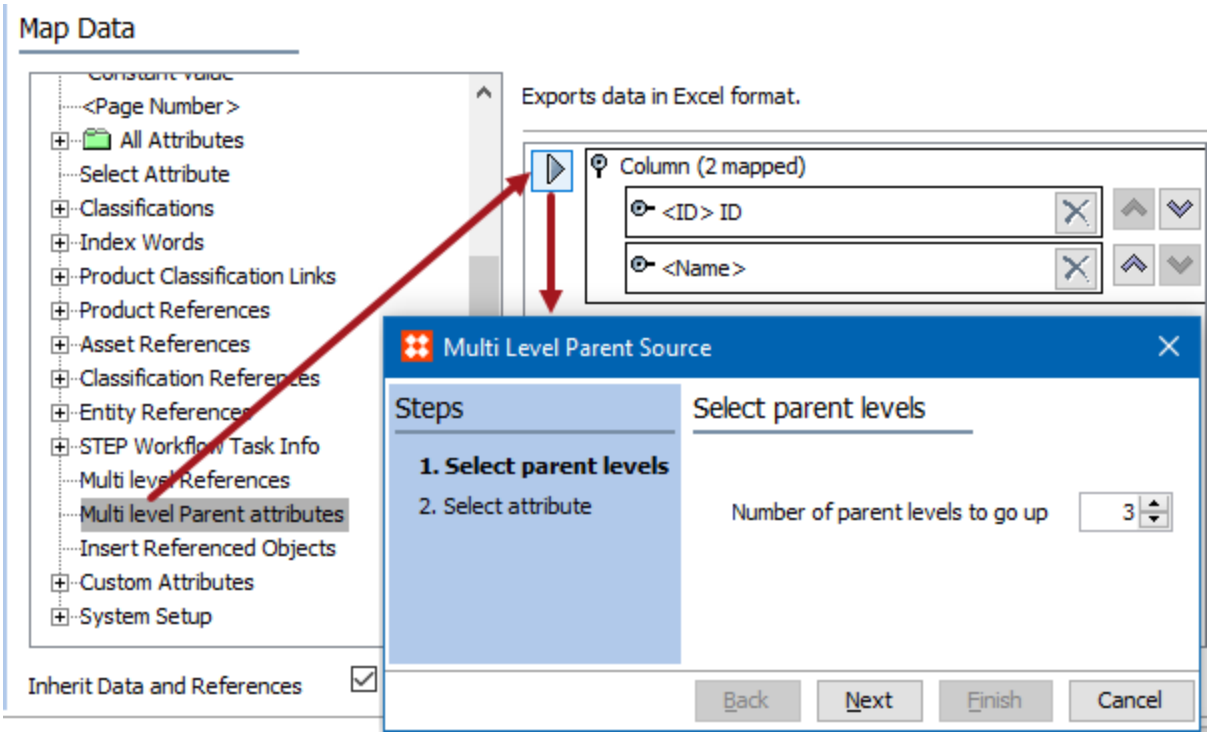
The user interface for the mapping step of the Export Manager tool and the Outbound Integration Endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter 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, see **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

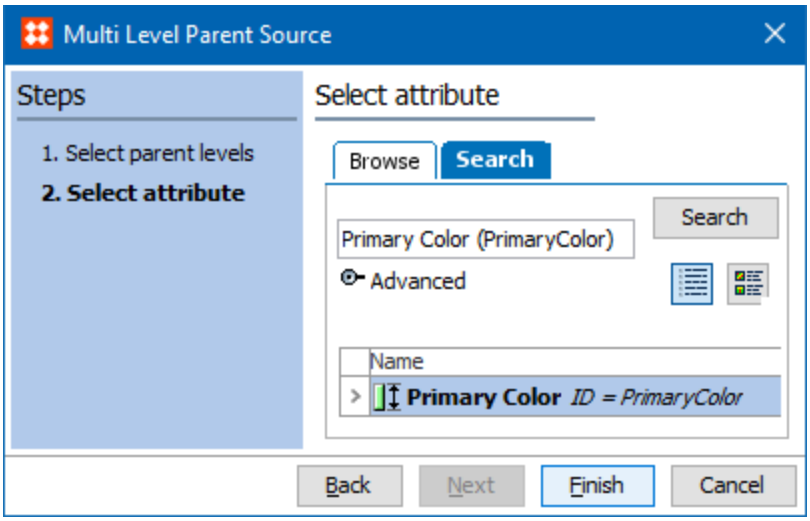
After mapping, most output options can be altered using transformations. For more information, see the **Outbound Map Data - Transform** documentation.

Mapping Multi Level Parent Attributes

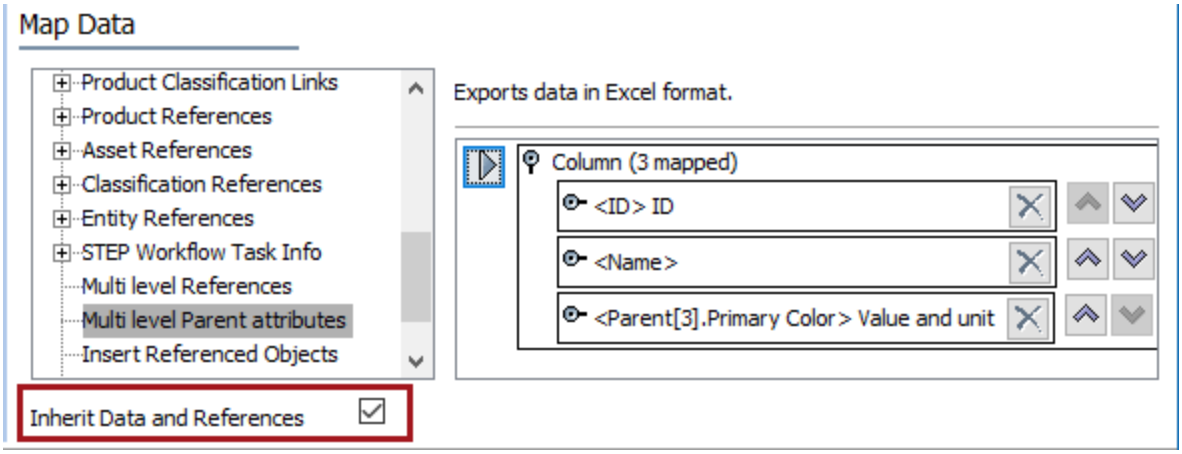
1. Select the data to be exported for the output tool. For more information, see the **Data Exchange** topic.
2. On the Map Data step, in the left panel, select the **Multi level Parent attributes** option, and in the right panel, click the arrow mapping button ( ) to display the Multi Level Parent Source 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**.

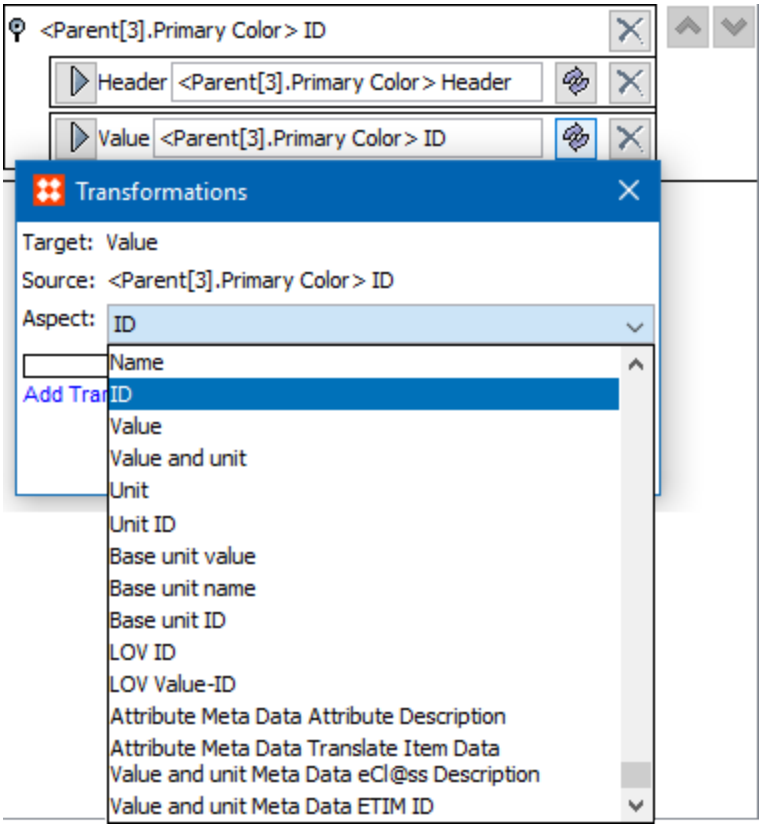


5. Back on the Map Data dialog, check the **Inherit Data and References** option appropriately:
   - If **checked**, inherited data and references are included in the export.
   - If **unchecked**, inherited data and references is not included in the export.



For information on inheritance, see the Inheritance in 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, see the **Aspect - Transform Outbound** topic.

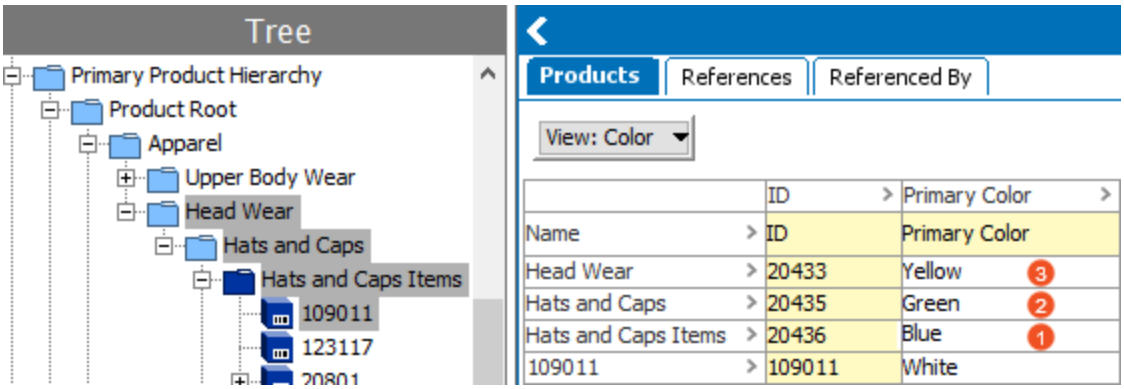


- Open the asset reference flipper to display the Header and Value elements.
- Click the transformation button ( ) for the Header or for the Value element to display the Transformations dialog and the Aspects parameter.
- Click the **Aspect** dropdown to display and select an option.
- Click the **Save** button to apply any aspect changes.

7. Apply any transformations necessary to change the output without changing the original data. See 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 is available for all formats that require mapping. This allows you to map data from referenced / 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.

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:

- Meta data attributes on references with values.
- Names of referenced objects.
- Attribute values on referenced 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, see **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

After mapping, most output options can be altered using transformations. For more information, see the **Outbound Map Data - Transform** documentation.

To demonstrate this data source option, this topic includes the following sections:

- Handling Multiple References
- Mapping Multi Level References
- Results

Handling Multiple References

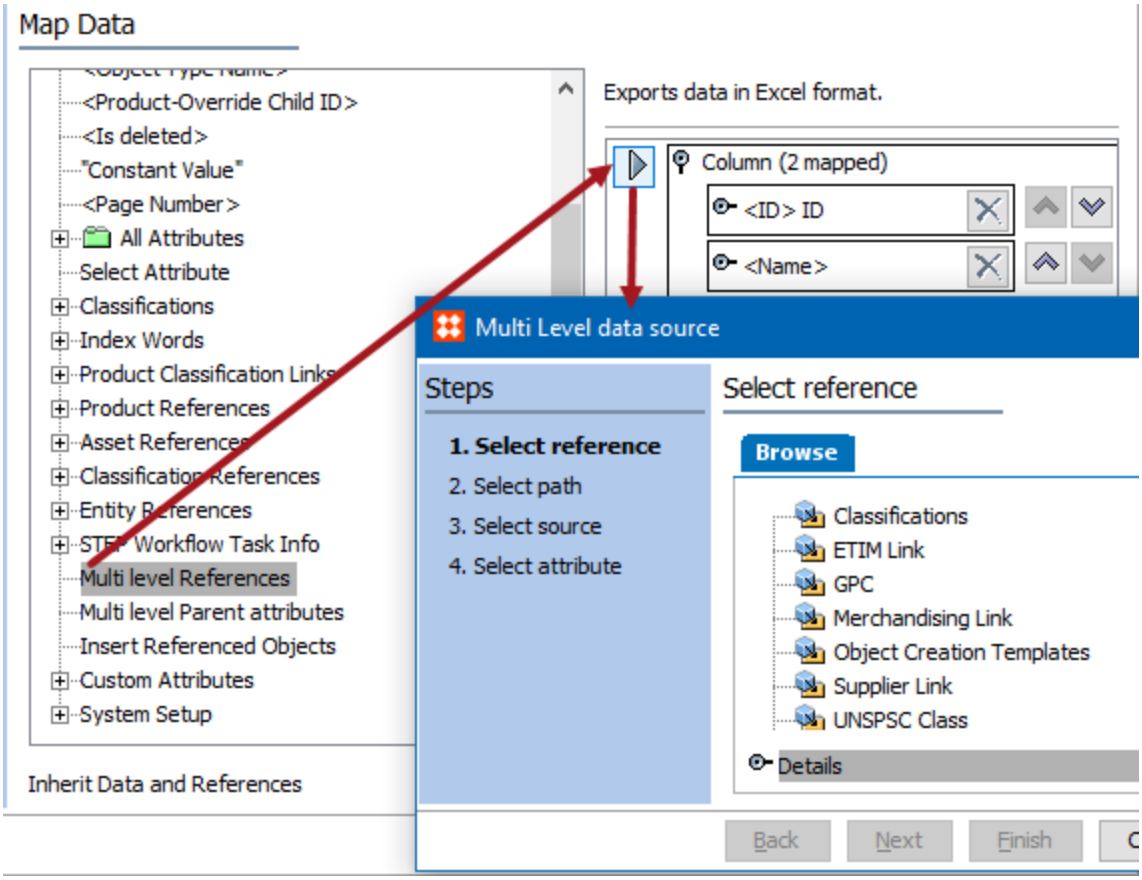
When multiple references exist, the options for the output are:

- All references in a single field, separated - This is the default setting and is achieved using the Multi Valued option.
- One reference per field - Using the Item option repeatedly, each reference can be output in individual fields.

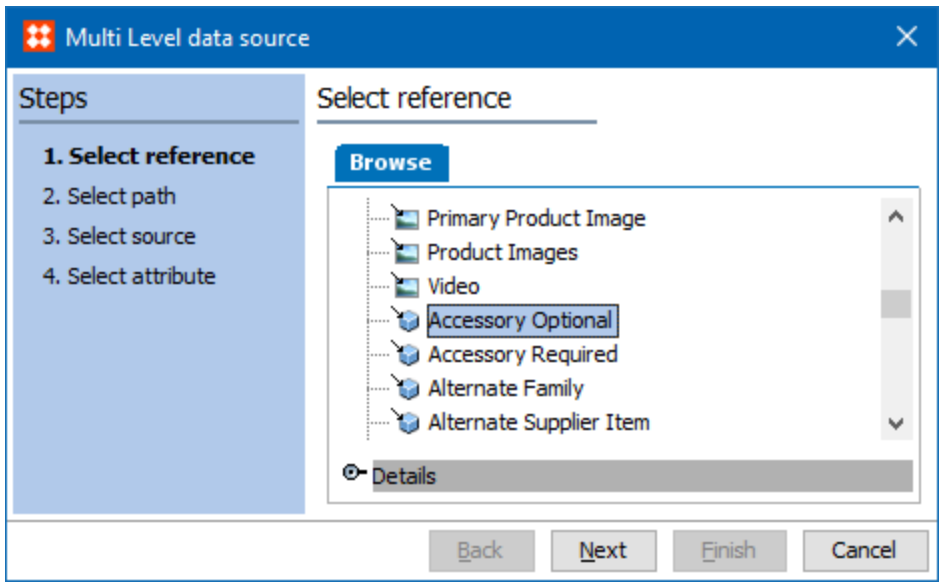
Details for using each option is described below.

Mapping Multi Level References

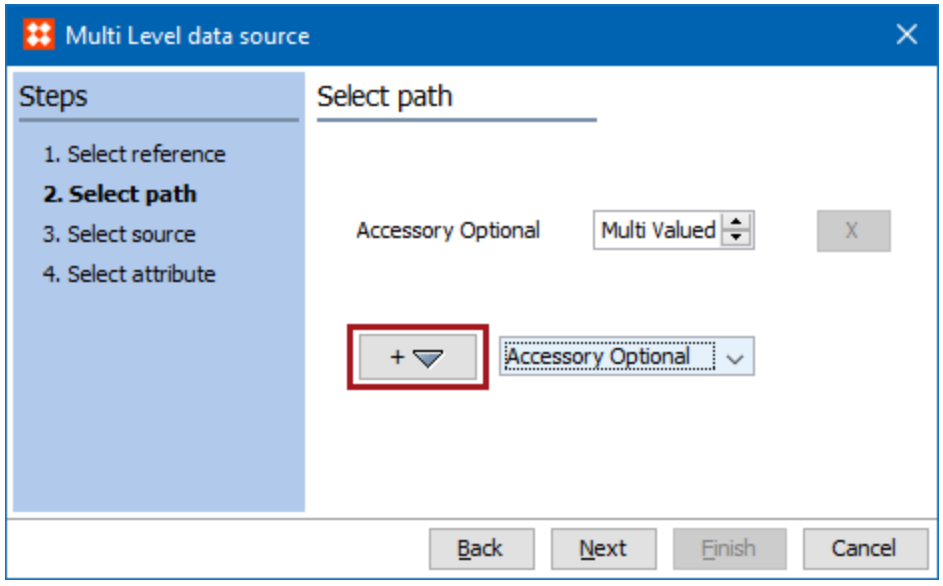
1. Select the data to be exported for the output tool. For more information, see the **Data Exchange** topic.
2. On the Map Data step, in the left panel, select **Multi level References** data source.
3. In the right panel, click the right arrow icon to display the **Multi Level data source** wizard.



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**.

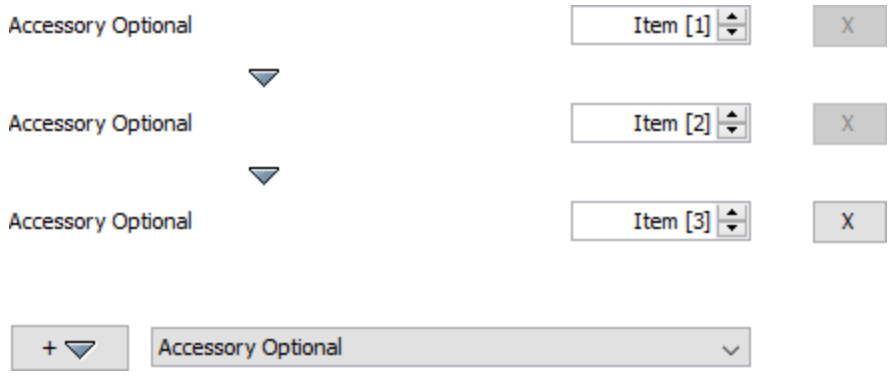
   Based on selected reference, the following options may be available:

   - **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.

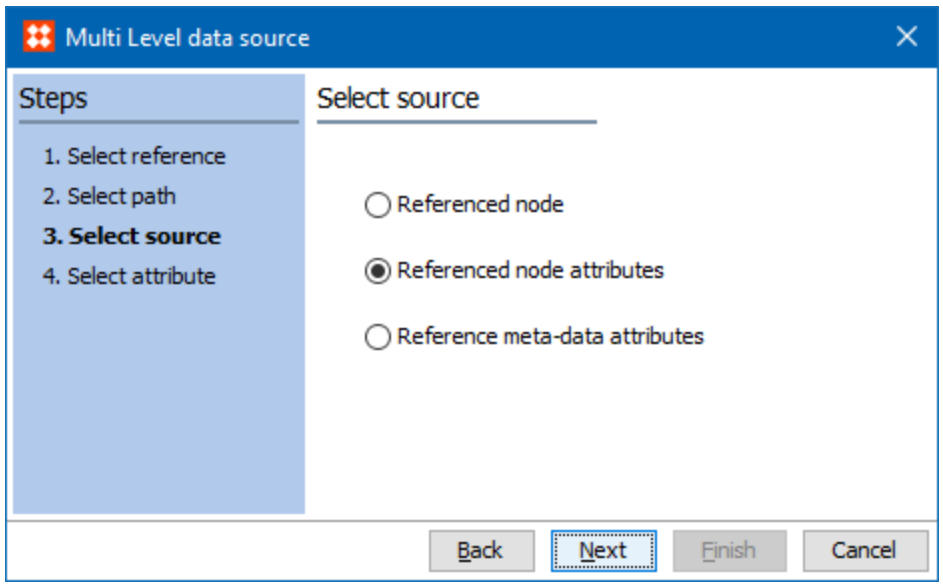
- **Item [ ]** - exports data from only a single referenced object, determined by the selected Item number. For example, Item [1] only exports data from the first referenced object.

- **Item [ ] with multiple references** - exports multiple references in individual fields (not a single, separated field as with the Multivalued option), first determine the maximum number of possible instances for the selected reference type per referencing object. Then change Multi Valued to Item [ ] option and use the plus / down arrow button, , until the maximum number is displayed.

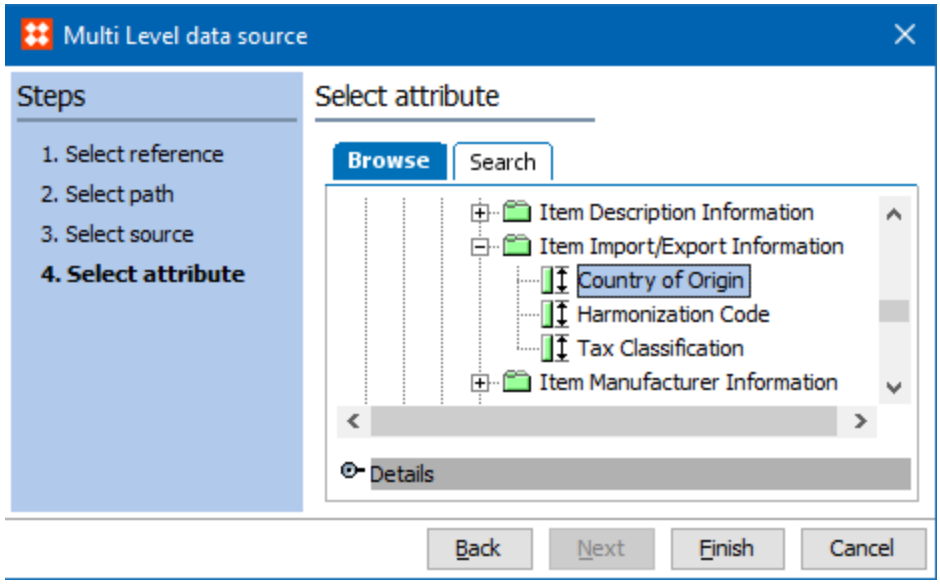


- **X** - Use the delete button to remove any unnecessary levels.

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**.

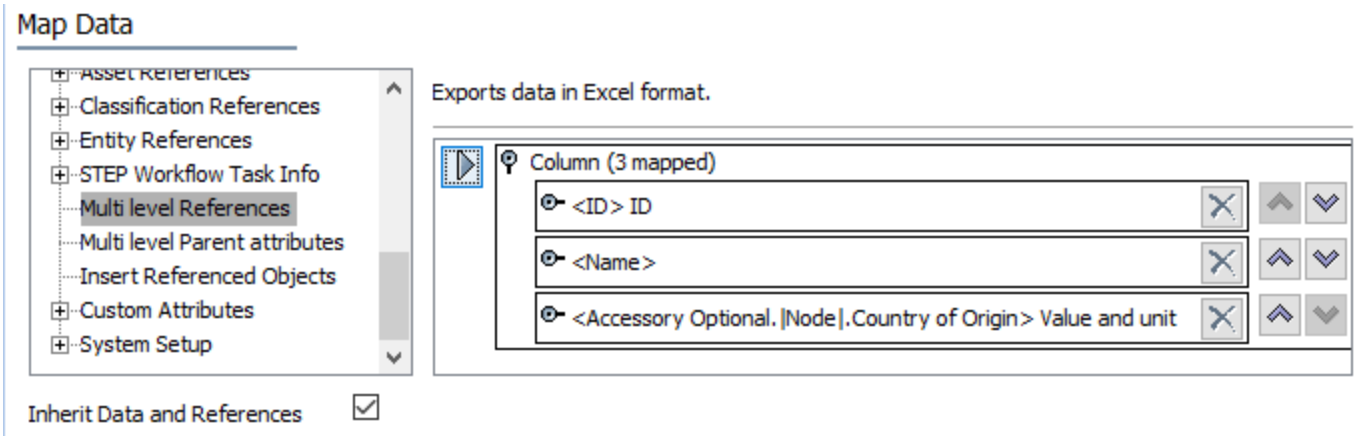


8. If needed, in **Select Attribute**, use **Browse** or **Search** to locate the attribute that holds the values to be exported and click **Finish**.



The multi-level reference mapping is complete and displayed on the Map Data dialog.

9. Complete any additional mappings and initiate the export.



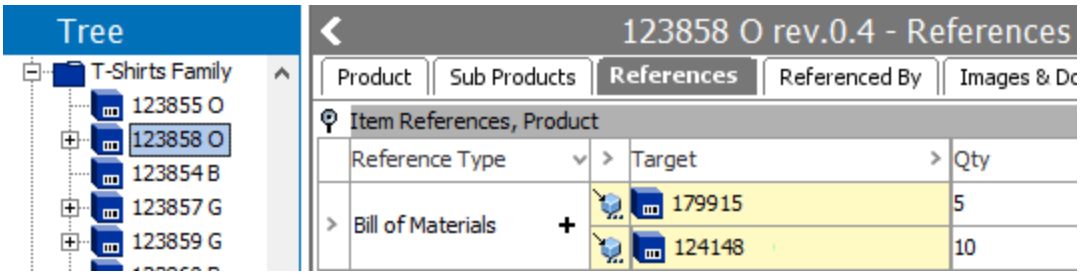
The single object output includes the values for both of the Accessory Optional references. Although this example used the Excel format, any other format would produce the same output. In the following image, '.|Node|.' is the referenced product of the exported product. This is included since on the **Select source** step, we chose 'Referenced node attributes' and then for the **Select attribute** step, we selected 'Country of Origin.'

	A	B	C
1	<ID>	<Name>	<Accessory Optional.|Node|.Country of Origin>
2	20805	Baseball Cap	SPAIN;CHINA

Results

The following section includes examples of exporting for multi- and single-valued reference types.

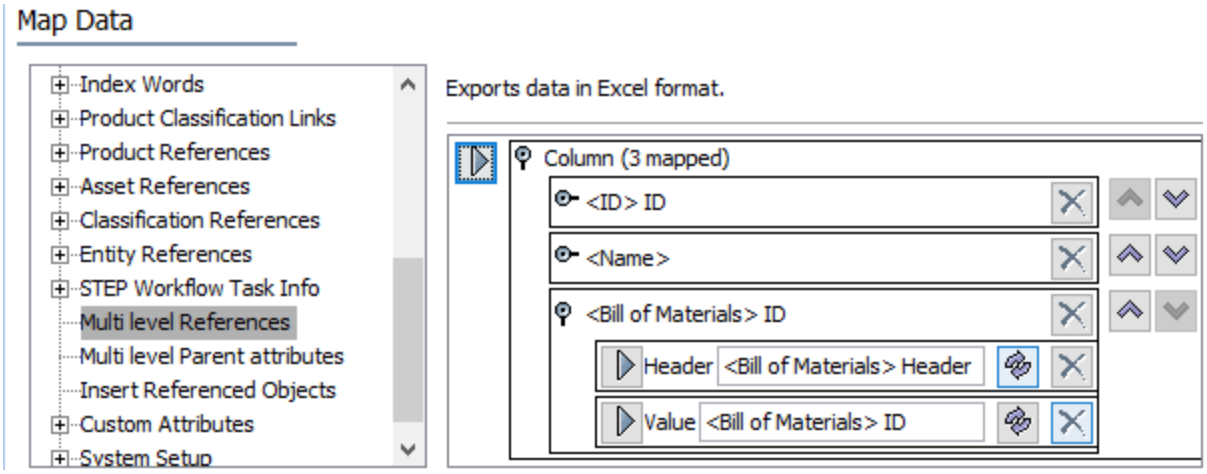
Export a product with references to a multi-valued reference type. For example, the product below has multiple references via the 'Bill of Material' reference type, and includes metadata attribute 'Qty.'



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:



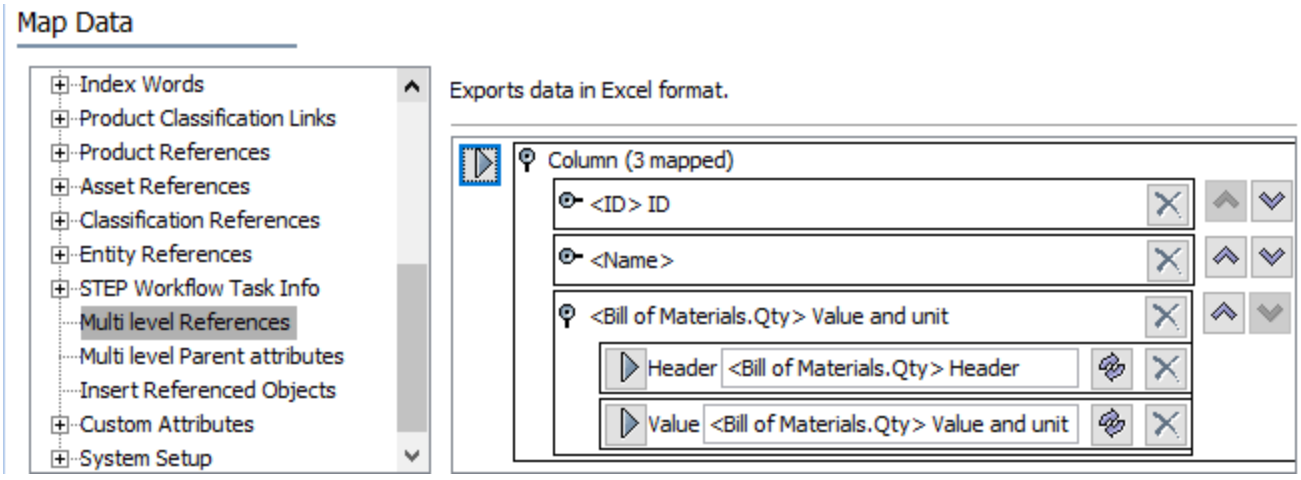
Once the export is initiated, the output includes the originally selected object ID and name, as well as the ID for each of the multiple references, separated by a semicolon (;).

	A	B	C
1	<ID>	<Name>	<Bill of Materials>
2	124458	123858 O	179915;124148

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 we did previously.
- On the **Select path** step, set 'Bill of Materials' as 'Multi Valued', as we did previously.
- On the **Select source** step, now choose 'Reference meta-data attributes.'
- On the **Select attribute** step, now choose the 'Qty' attribute.

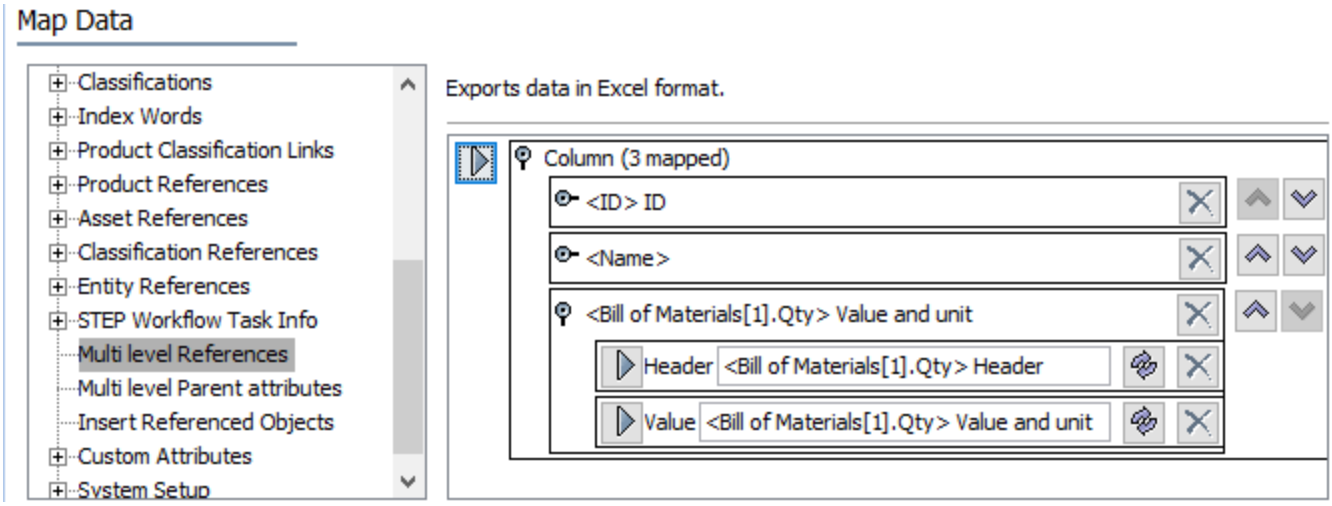
The final mapping is displayed as follows:



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' attribute.

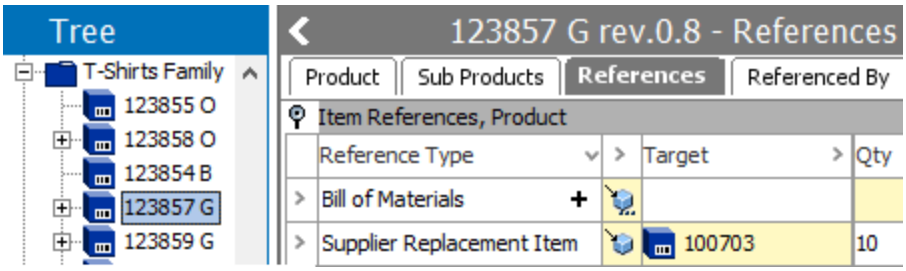
The final mapping is displayed as follows:



Once the export is initiated, the output includes the originally selected object ID and name, 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

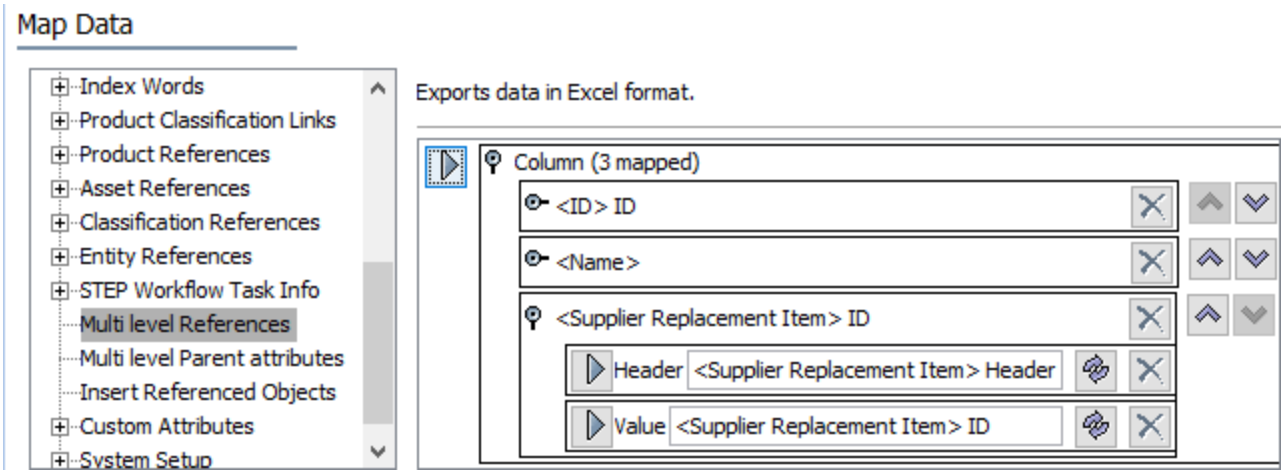
The same functionality is available for single-valued reference types, where only one item can be referenced. The following product includes a reference to another product with ID 100703 via the 'Supplier Replacement Item' reference type. No additional references of this type are allows, as indicated by the absence of the + sign following the reference type.



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.'

The final mapping is displayed as follows:



Once the export is initiated, the output includes the originally selected object ID and name, as well as the ID for the 'Supplier Replacement Item' referenced object.

	A	B	C
1	<ID>	<Name>	<Supplier Replacement Item>
2	123857	123857 G	100703

Exporting the same object again, but on the **Select source** step, we chose 'Referenced node attributes' and then for the **Select attribute** step, we selected 'Country of Origin' results in the following. As before, '.|Node|.' indicates the selected source was 'Referenced node attributes.'

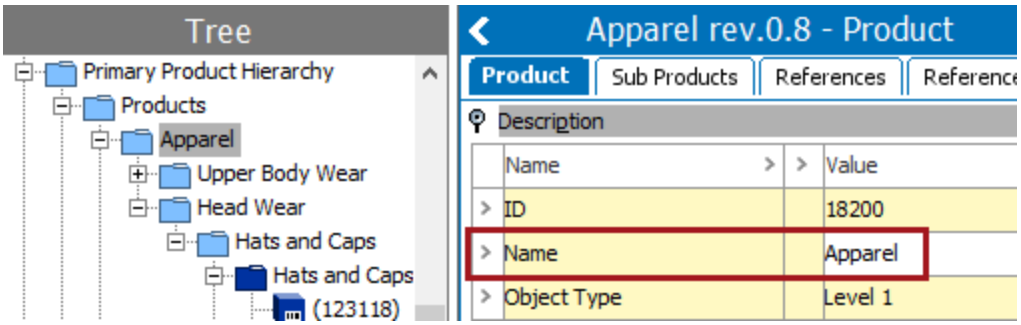
	A	B	C
1	<ID>	<Name>	<Supplier Replacement Item.|Node|.Country of Origin>
2	123857	123857 G	ANTIGUA AND BARBUDA

Exporting the same object again, but on the **Select source** step, we chose 'Reference meta-data attributes' and then for the **Select attribute** step, we selected 'Qty' results in the following.

	A	B	C
1	<ID>	<Name>	<Supplier Replacement Item.Qty>
2	123857	123857 G	10

Name - Data Source Outbound

The option to map a name allows output of the value of the Name attribute for the selected objects.



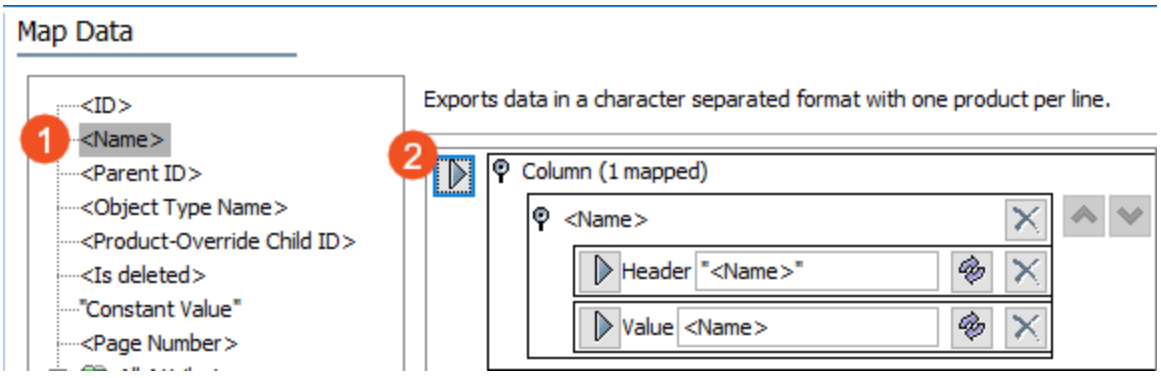
The user interface for the mapping step of the Export Manager tool and the Outbound Integration Endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter 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, see **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

After mapping, most output options can be altered using transformations. For more information, see the **Outbound Map Data - Transform** documentation.

Mapping Name

1. Select the data to be exported for the output tool. For more information, see 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. See the **Outbound Map Data - Transform** topic.
4. Complete any additional mappings and initiate the export.

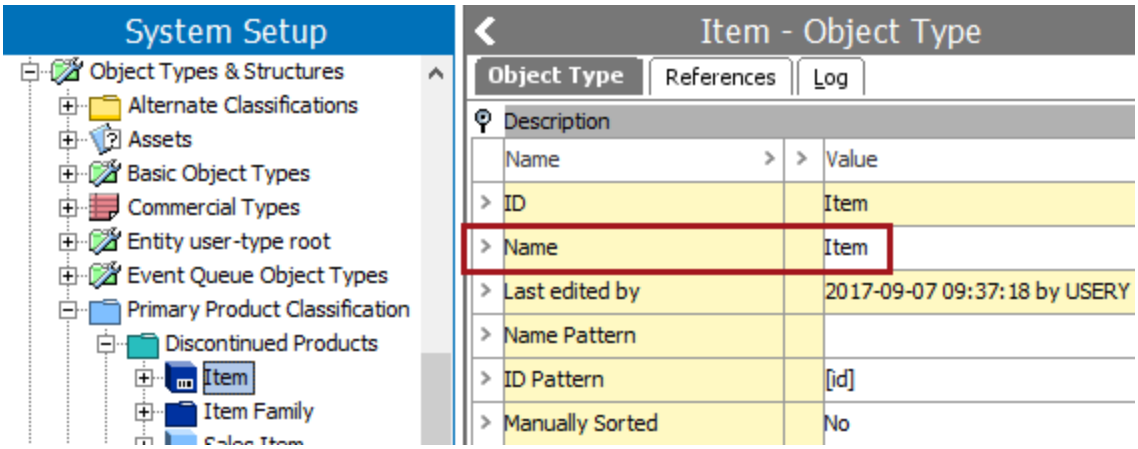
Results

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.



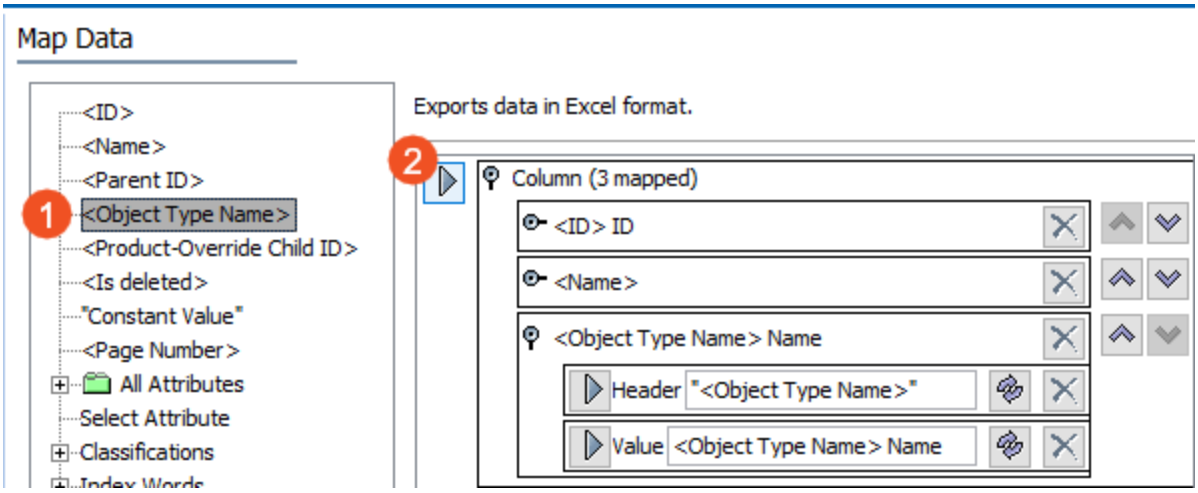
The user interface for the mapping step of the Export Manager tool and the Outbound Integration Endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter 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, see **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

After mapping, most output options can be altered using transformations. For more information, see the **Outbound Map Data - Transform** documentation.

Mapping Object Type Name

1. Select the data to be exported for the output tool. For more information, see the **Data Exchange** topic.
2. Select the super type for the desired object types. For more information, see **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, see 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, see **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

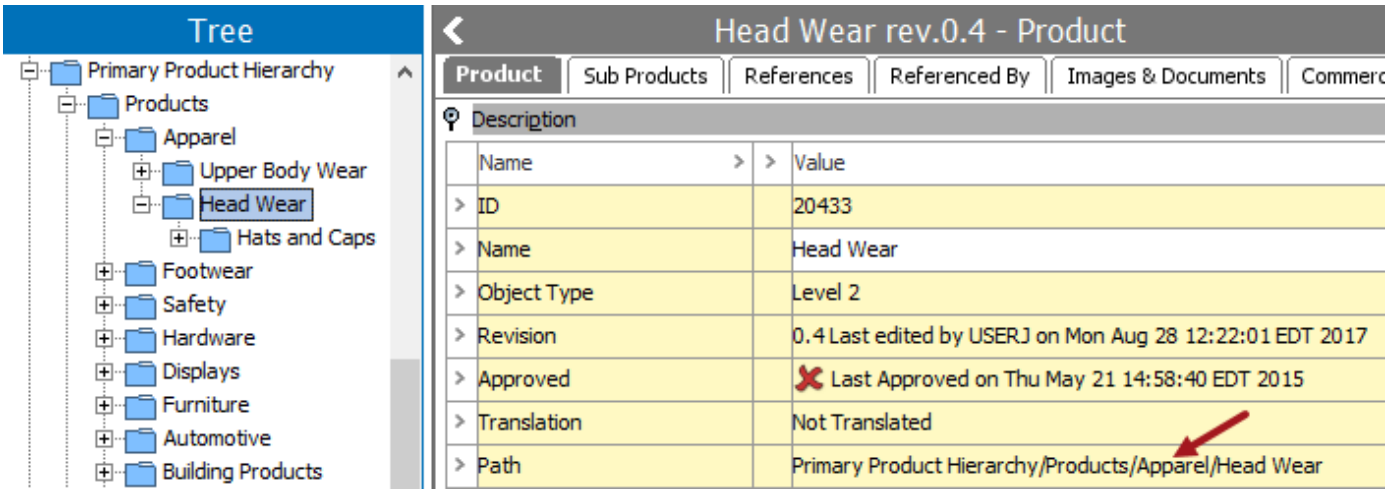
After mapping, most output options can be altered using transformations. For more information, see the **Outbound Map Data - Transform** documentation.

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, see the **Unique Keys** section of the **System Setup / Super User Guide** documentation.

The parent is displayed as part of the Path parameter on an object editor, and indicates the node directly above the selected object.



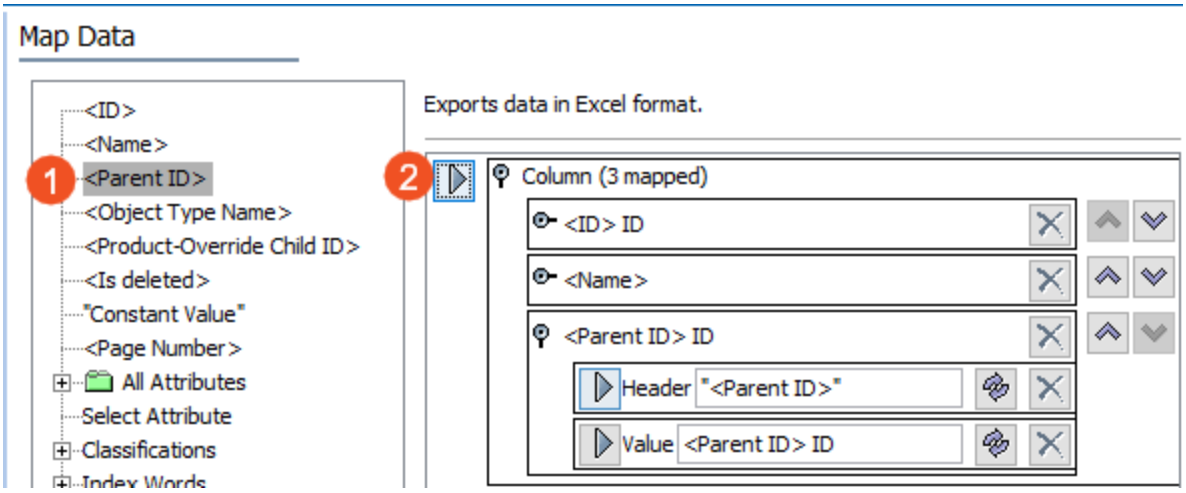
The user interface for the mapping step of the Export Manager tool and the Outbound Integration Endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter 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, see **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

After mapping, most output options can be altered using transformations. For more information, see the **Outbound Map Data - Transform** documentation.

Mapping Parent ID or Key

1. Select the data to be exported for the output tool. For more information, see 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. See the **Outbound Map Data - Transform** topic.
4. Complete any additional mappings and initiate the export.

Results

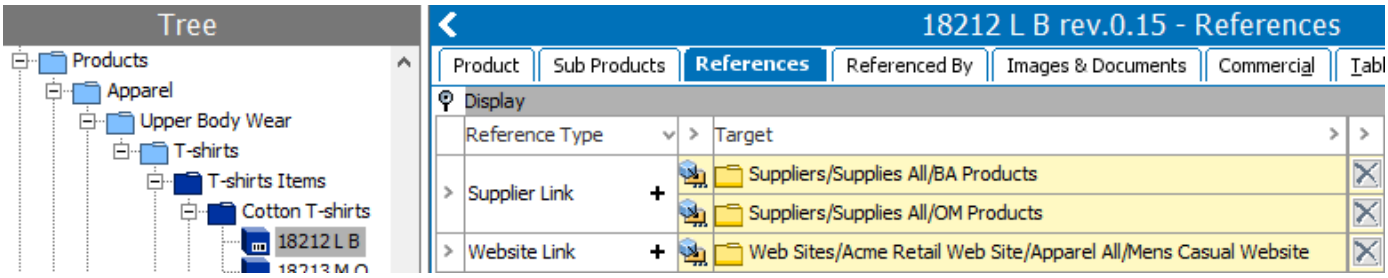
The output includes <Parent ID> as a header (or Object Type ID if transformed) and the value for each row.

	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.



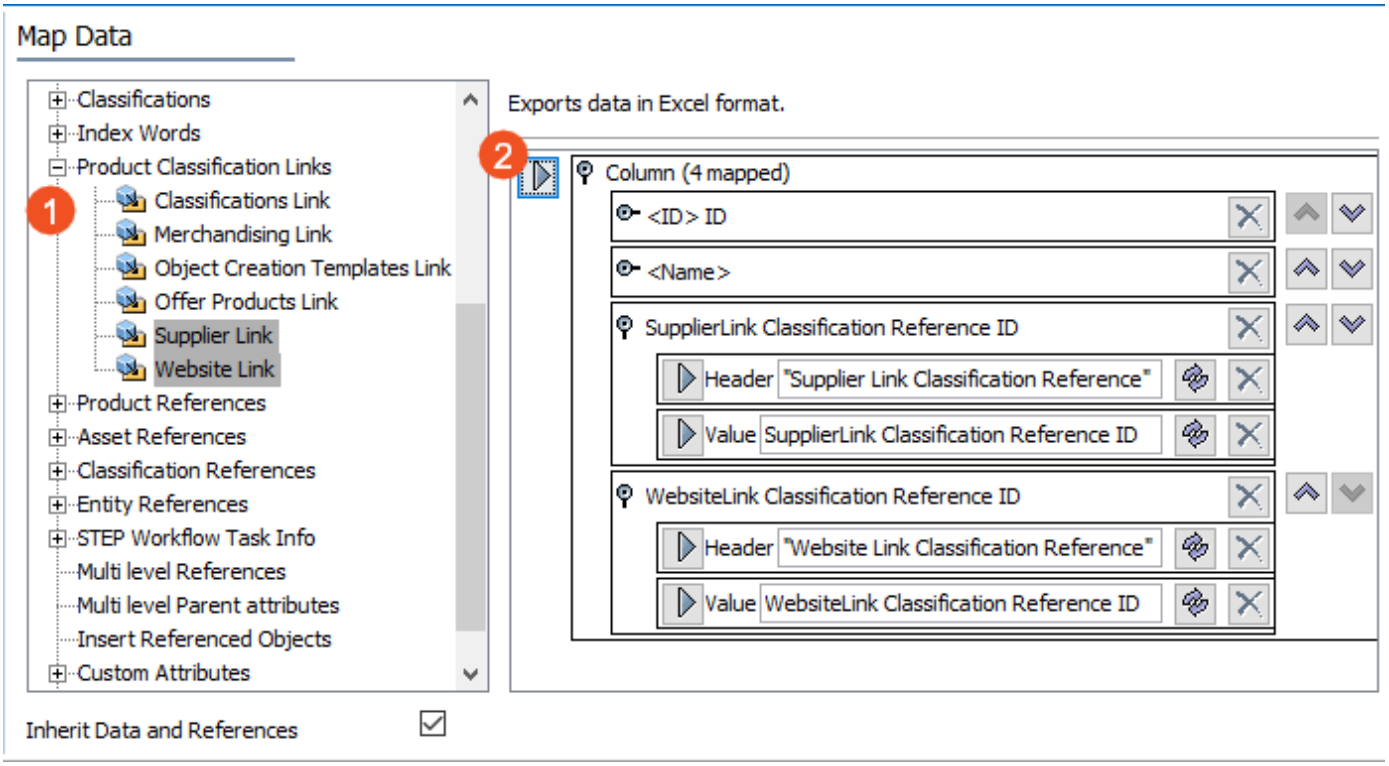
The user interface for the mapping step of the Export Manager tool and the Outbound Integration Endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter 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, see **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

After mapping, most output options can be altered using transformations. For more information, see the **Outbound Map Data - Transform** documentation.

Mapping Product Classification Links

1. Select the data to be exported for the output tool. For more information, see 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, see the Inheritance in 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. See the **Outbound Map Data - Transform** topic.
6. Complete any additional mappings and initiate the export.

Results

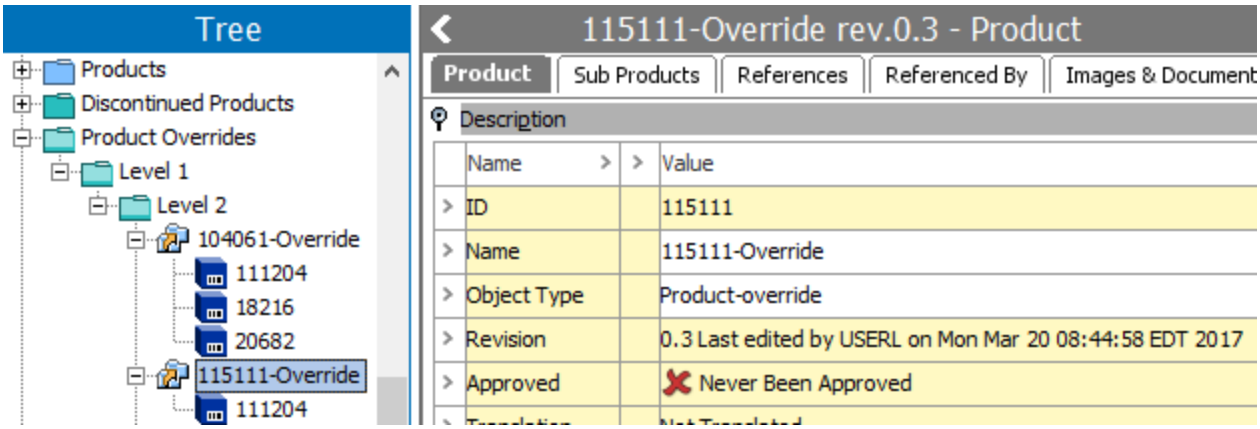
The output includes the product classification link type name as a header and the ID for each row.

When the product references multiple product classification links, the product classification link IDs are displayed in a semi-colon separated list.

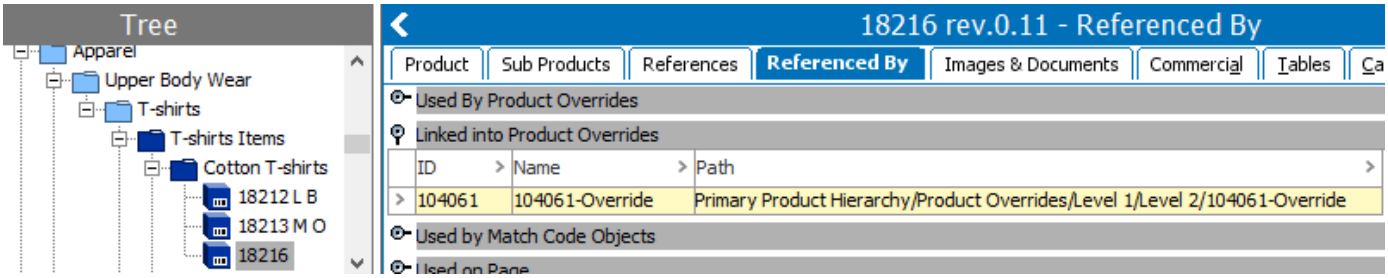
	A	B	C	D
1	<ID>	<Name>	Supplier Link Classification Reference	Website Link Classification Reference
2	18212	18212 L B	BAProducts;OMProducts	22583

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.

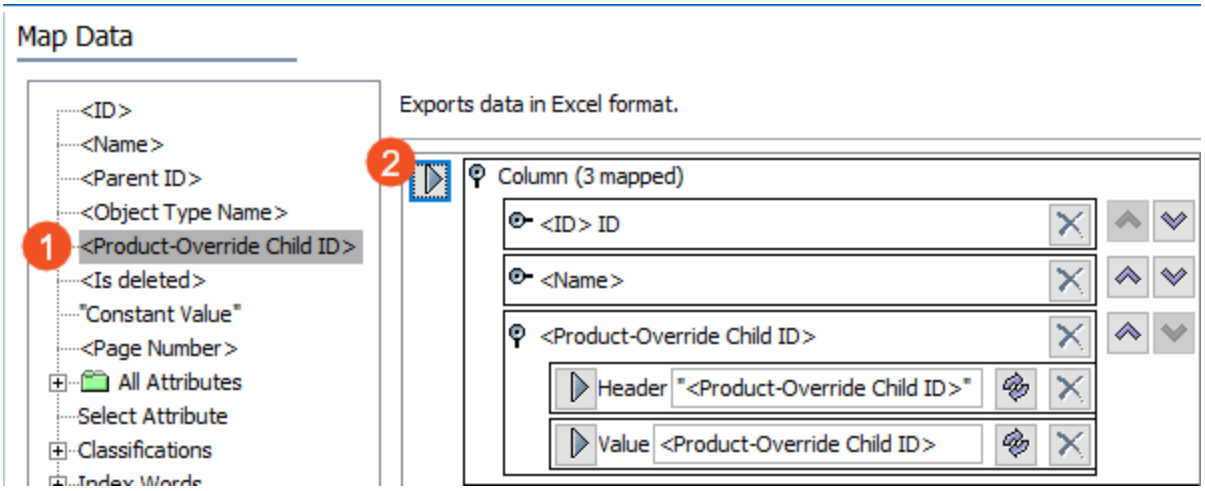
For more information about the additional wizard steps, see **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

After mapping, most output options can be altered using transformations. For more information, see the **Outbound Map Data - Transform** documentation.

Mapping Product-Override Child ID

1. Select the data to be exported for the output tool. For more information, see the **Data Exchange** topic.

2. Select the product super type and the product override objects for export. For more information, see **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, see the Inheritance in 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. See 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.

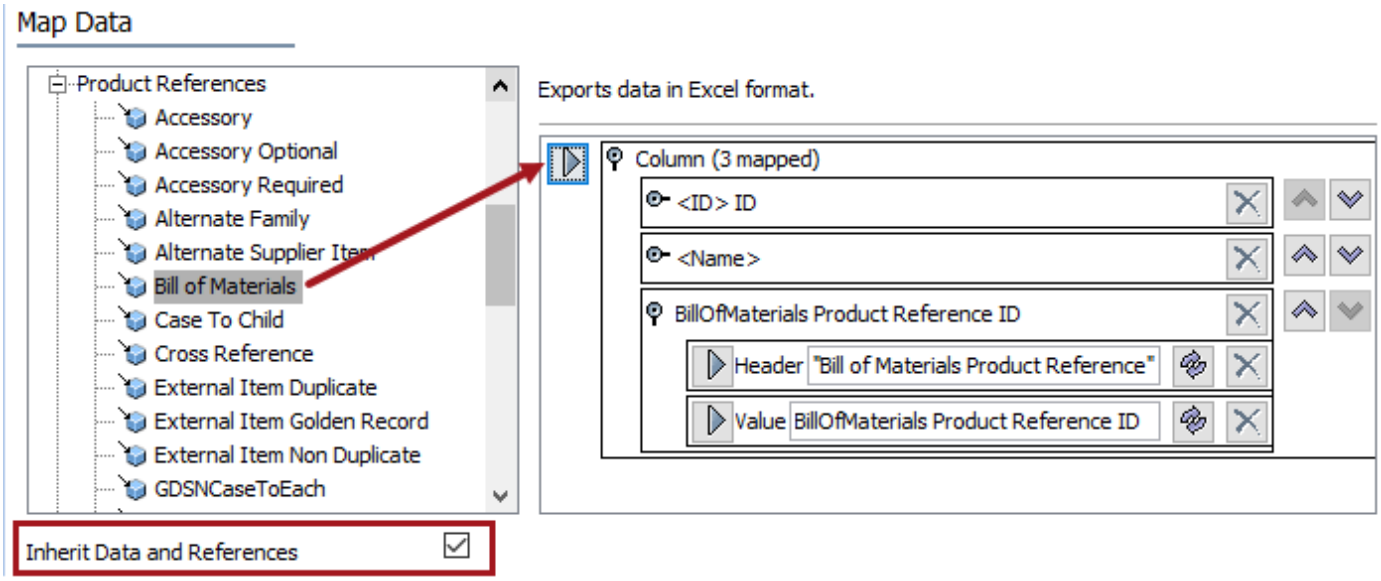
For more information about the additional wizard steps, see **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

After mapping, most output options can be altered using transformations. For more information, see the **Outbound Map Data - Transform** documentation.

1. Select the data to be exported for the output tool. For more information, see 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.

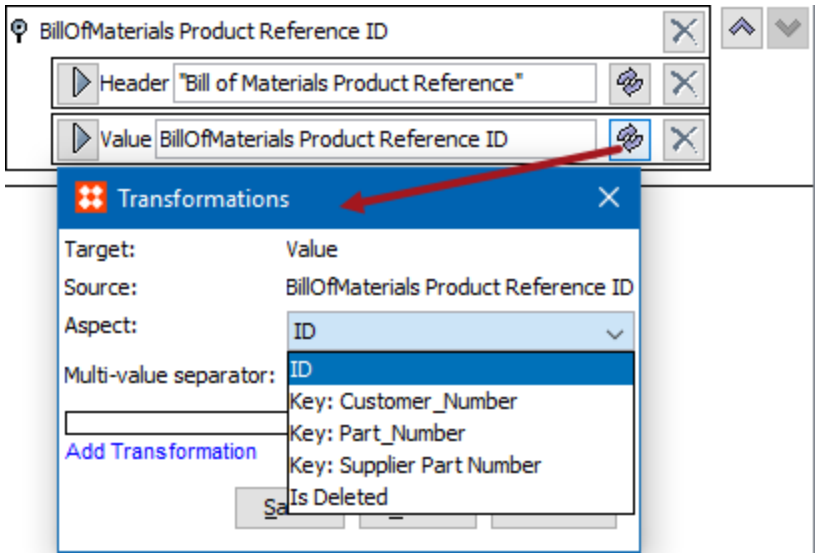


4. Check the **Inherit Data and References** option appropriately:

- 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, see the Inheritance in 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, see 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. See 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.

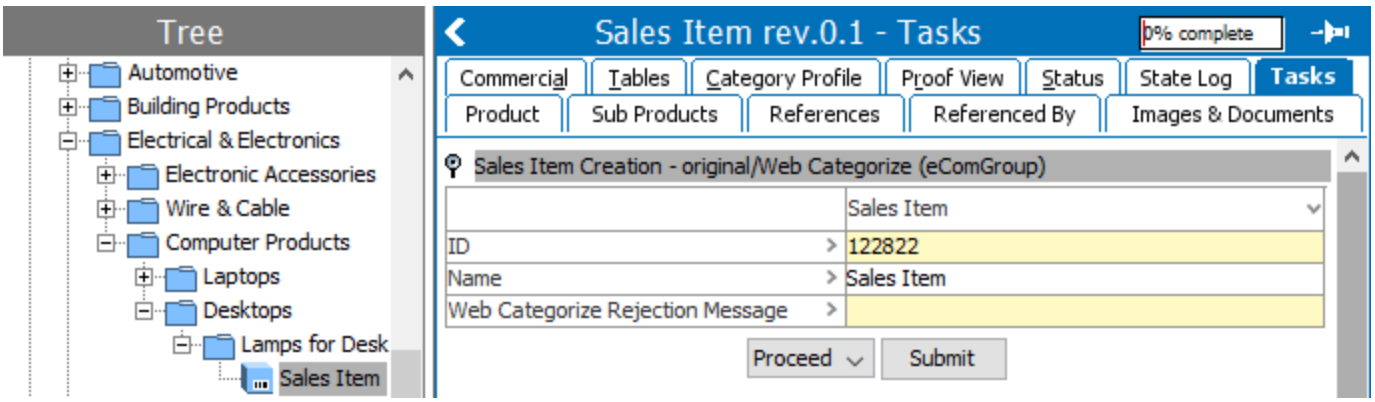
	A	B	C	D
1	<ID>	<Name>	Bill of Materials Product Reference	Supplier Replacement Item Product Reference
2	18212	Item 1		100703
3	18213	18213 M O		124148
4	18216	18216	123857;123854	123855

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

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, see the **Exporting Workflow Statuses** topic in the **Workflows** documentation.

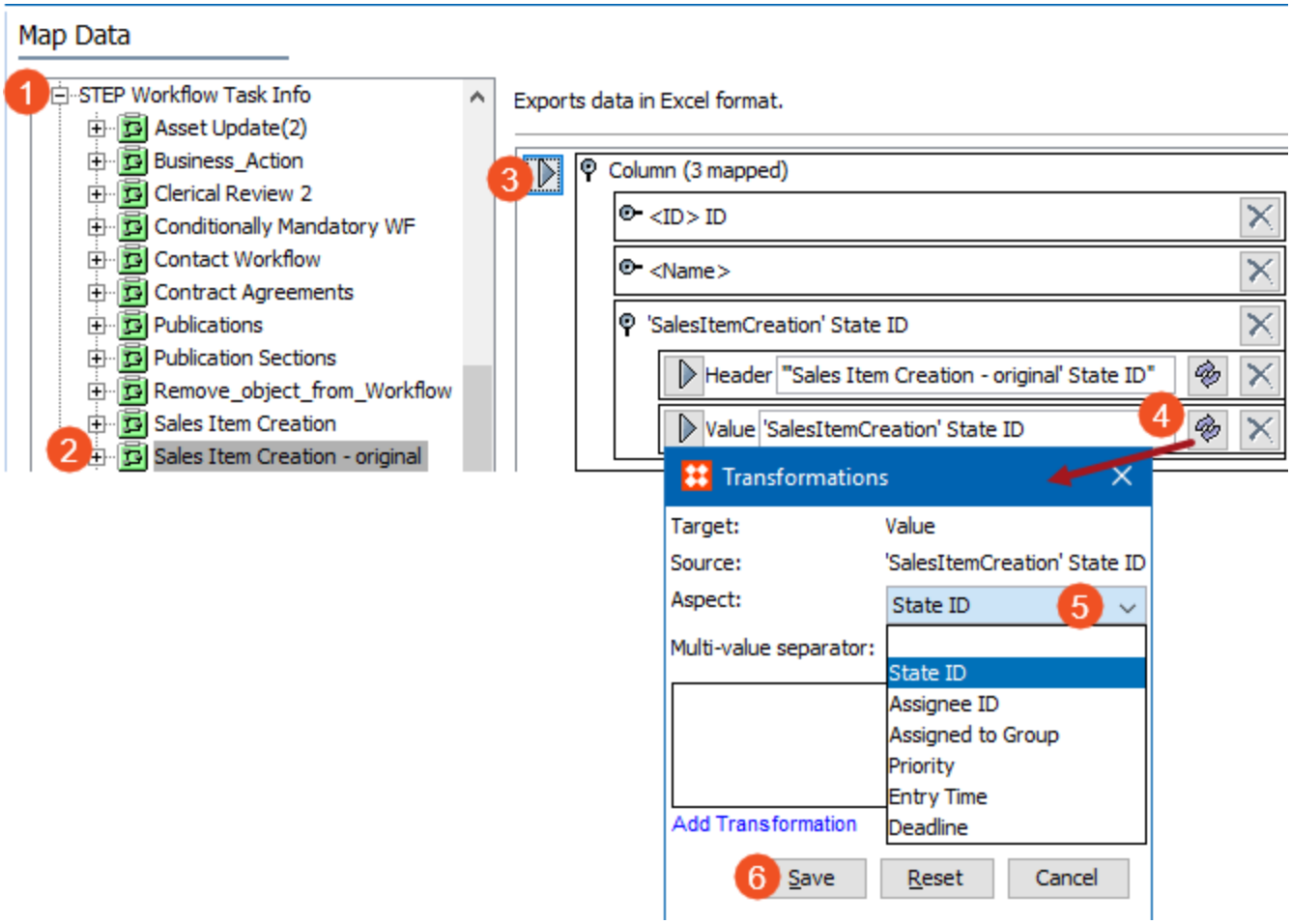
The user interface for the mapping step of the Export Manager tool and the Outbound Integration Endpoint tool are not exactly the same. For Export Manager, mapping is described in the **Export Manager - Map Data** topic. For OIEPs, mapping is handled in the Output Templates flipper under the Format parameter 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, see **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

After mapping, most output options can be altered using transformations. For more information, see the **Outbound Map Data - Transform** documentation.

Mapping STEP Workflow Task Info

1. Select the data to be exported for the output tool. For more information, see 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, see 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.

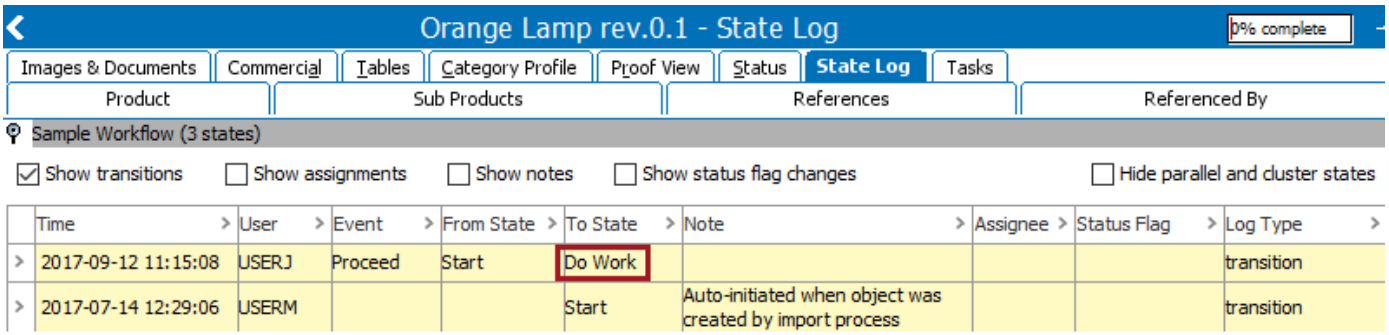
Results

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.

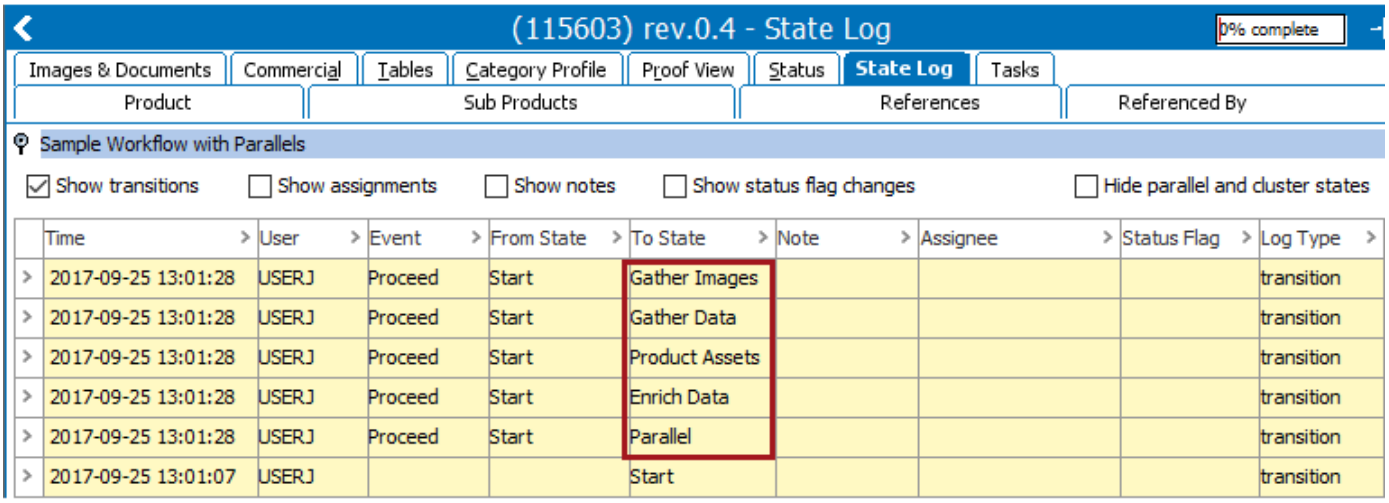
- **State ID** Data and Output

  When a product exists in a single workflow state for the workflow as shown below, the workflow state is displayed.



A	B	C
<ID>	<Name>	'Sample Workflow (3 states)' State ID
208655	Orange Lamp	DoWork

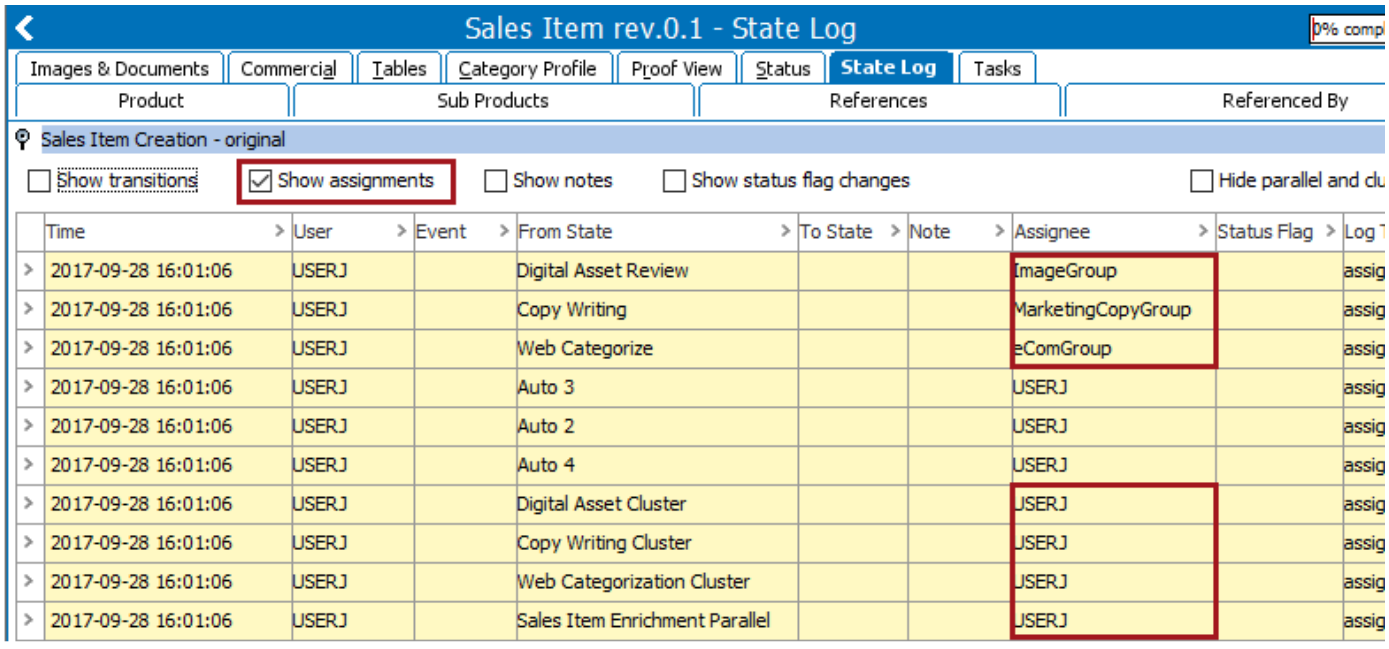
  When a product exists in multiple workflow states for the same workflow as shown below, all the relevant workflow states are displayed along with the Cluster and Parallel State IDs that the product exists within.



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

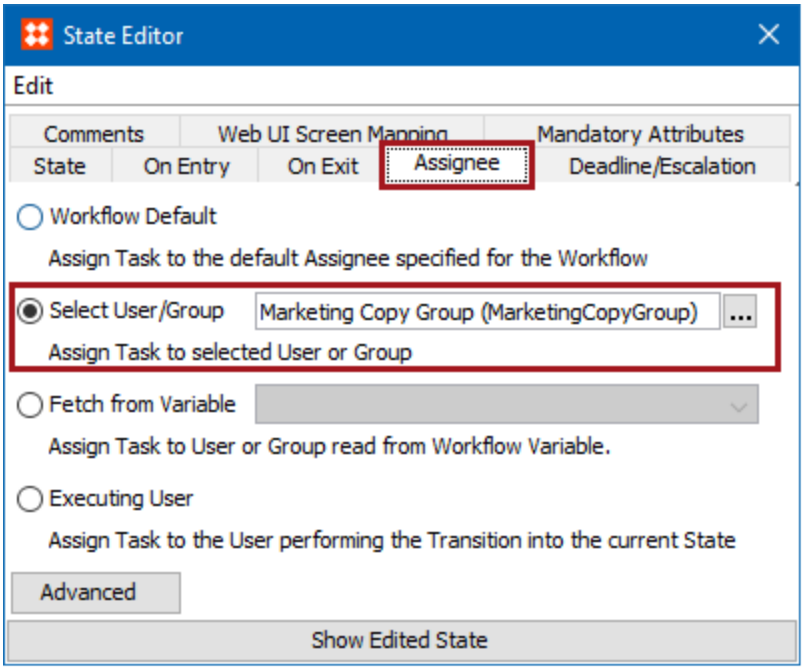
  When a workflow task has an assignee (set in the workflow State Editor on the Assignee tab), for the current states, the assignees are output. The following images show results for a workflow with a parallel that includes multiple clusters, so the product is assigned to multiple users at the same time.



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

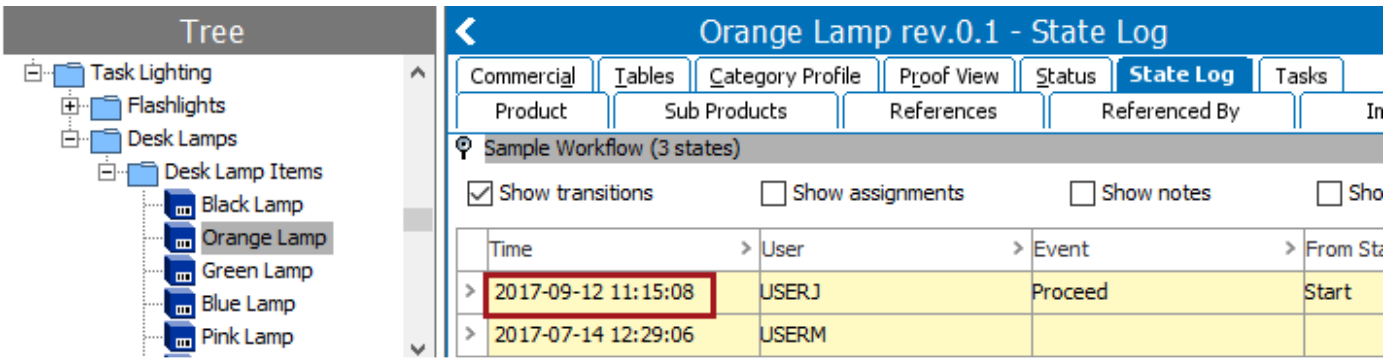
- **Priority** Data and Output

  The output includes only -1, which indicates the lowest priority. This legacy feature has been replaced with status flag functionality. For more information, see 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

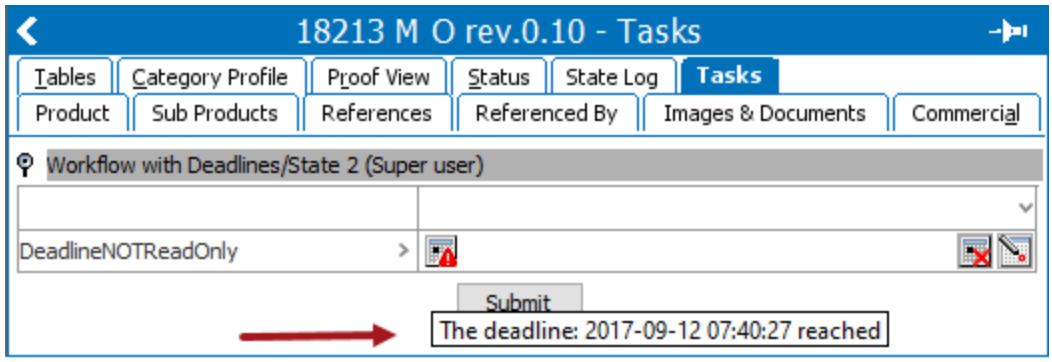
  When an object enters a workflow task state, the server time is recorded and can be output using this aspect. In the output, a single time is logged upon entry into a parallel or cluster, so the entry times do not correspond one-to-one with the times displayed on the State Log tab of the object for workflows 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, see the **Deadlines and Escalations in Workflows** topic and **Views and Mappings for Workflows** in the **Workflows** documentation.



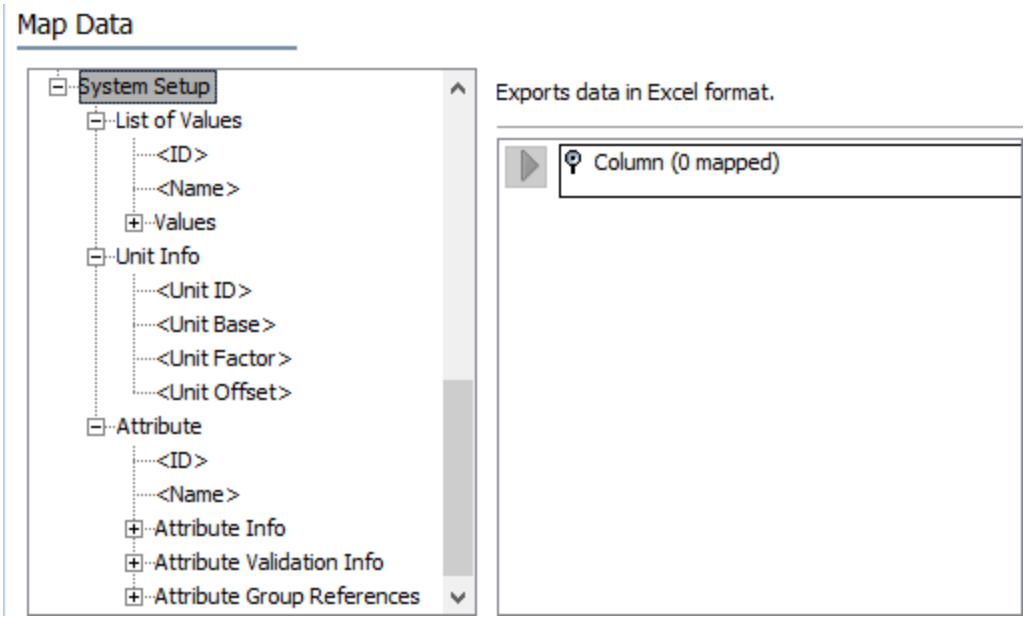
A	B	C
<ID>	<Name>	'Workflow with Deadlines' Deadline
18213	18213 M O	2017-09-12 07:40:27

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, see:

- 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, see **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

After mapping, most output options can be altered using transformations. For more information, see the **Outbound Map Data - Transform** documentation.

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.

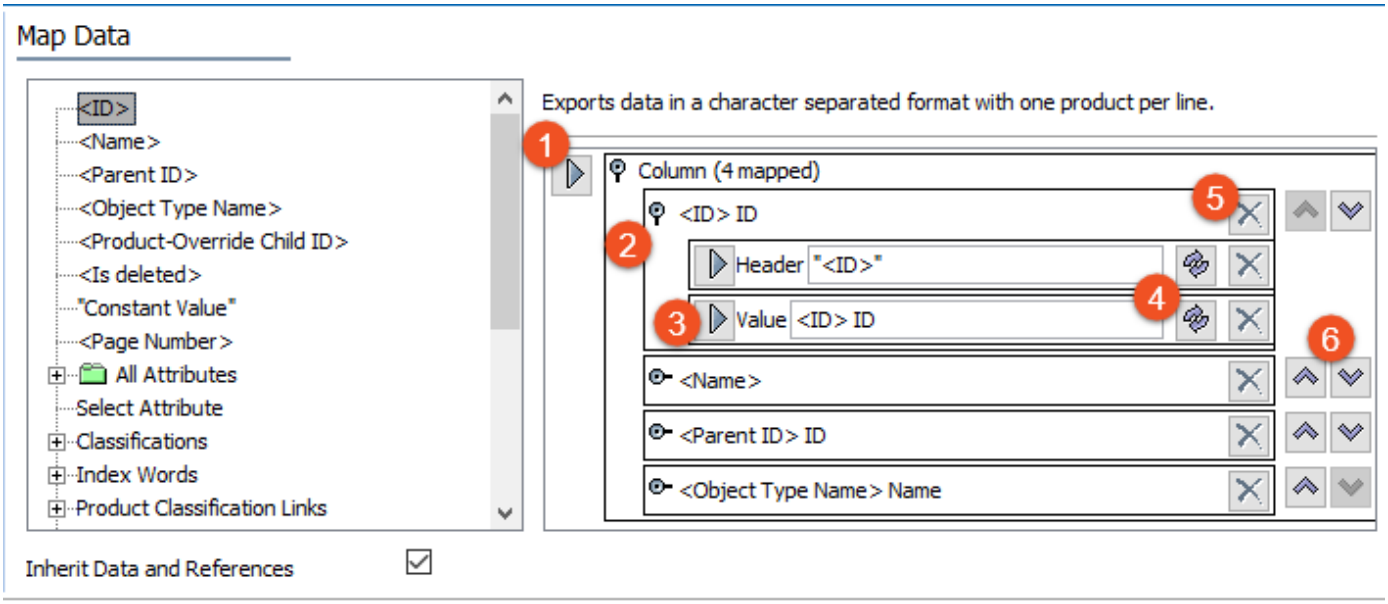
Generally, mapping data involves selecting a data source from the left panel and clicking the mapping button ( ) in the right panel. Based on the selected data source, the mapping rule appears immediately, or additional configuration is required. Ultimately, all data being exported is displayed as a 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, see **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.

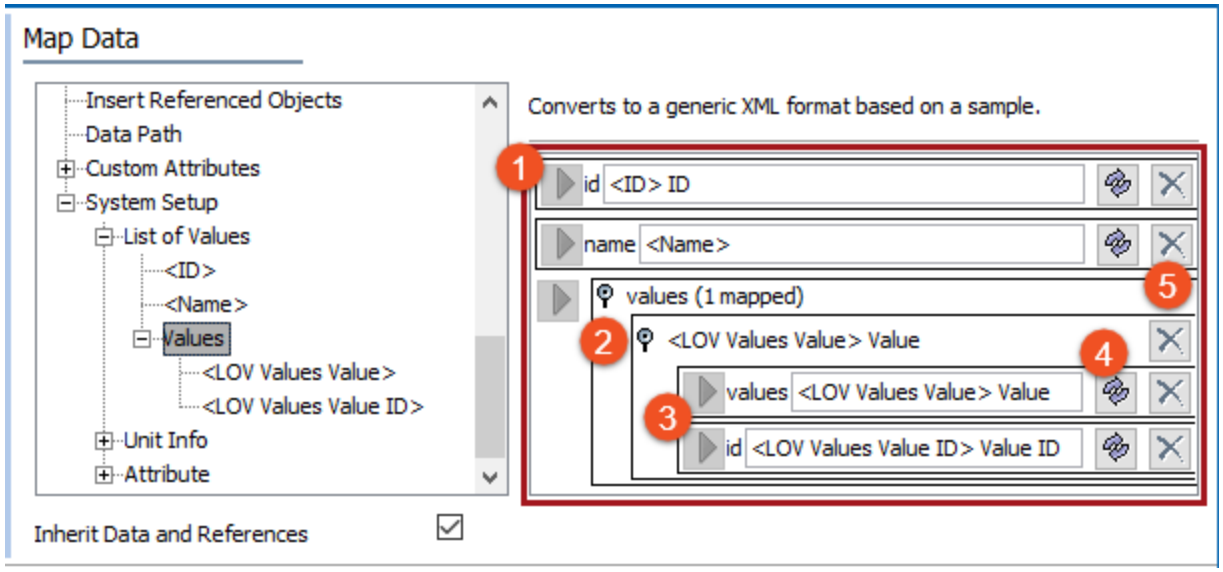


1. Mapping button assigns the selected data source to the selected mapping target.
2. A flipper is available when multiple mapping options are available for a single mapping target.
3. Mapping button assigns the selected data source to the selected mapping target within a flipper.
4. The transform button 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.

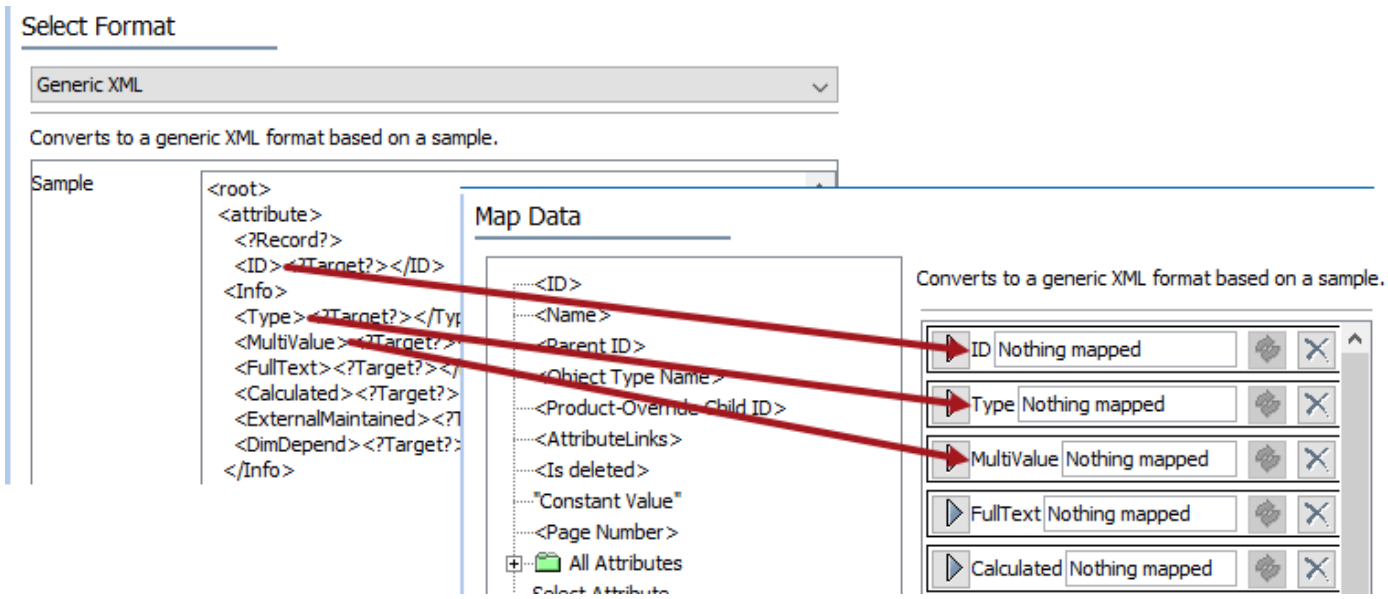
XML Format Targets

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.


Select Format
Generic XML
Converts to a generic XML format based on a sample.
Sample
<root>
<attribute>
<?Record?>
<ID><?Target?></ID>
<Info>
<Type><?Target?></Ty
<MultiValue><?Target?>
<FullText><?Target?></
<Calculated><?Target?>
<ExternalMaintained><?T
<DimDepend><?Target?>
</Info>
Map Data
<ID>
<Name>
<Parent ID>
<Object Type Name>
<Product-Override-Child ID>
<AttributeLinks>
<Is deleted>
"Constant Value"
<Page Number>
All Attributes
Converts to a generic XML format based on a sample.
ID Nothing mapped
Type Nothing mapped
MultiValue Nothing mapped
FullText Nothing mapped
Calculated Nothing mapped

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.

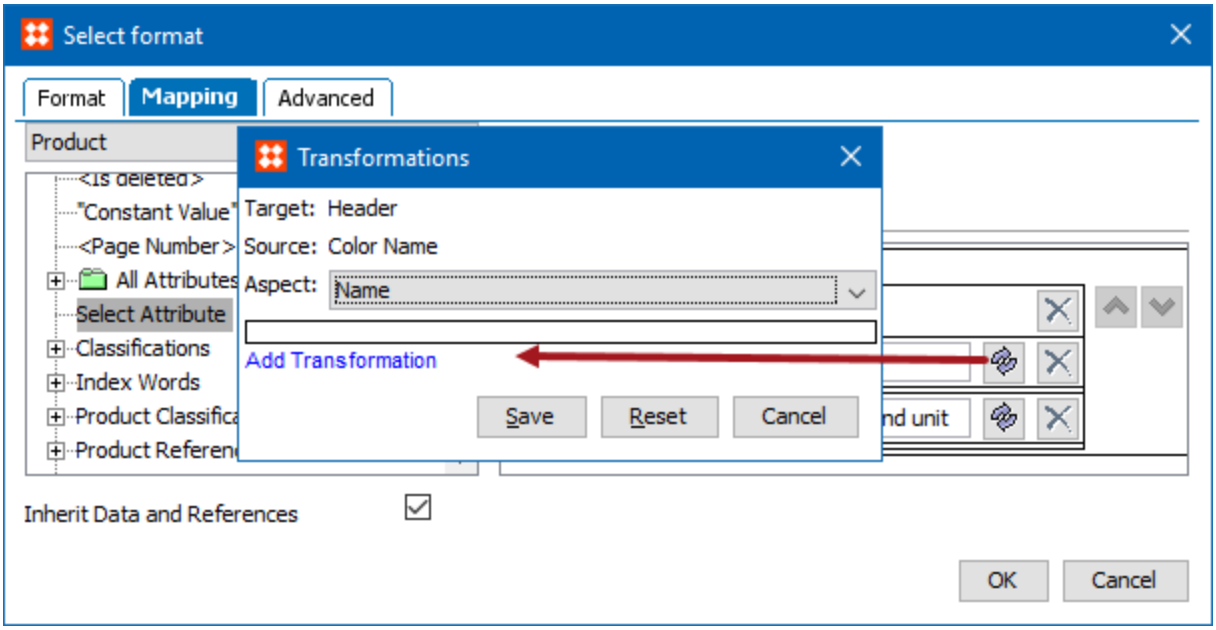
For examples of the most common transformations, see **Mapping Transformation Examples**.

To use transformations on attributes in STEP, or data in tables and publications, see **Attribute Transformations** in the **System Setup / Super User Guide** documentation.

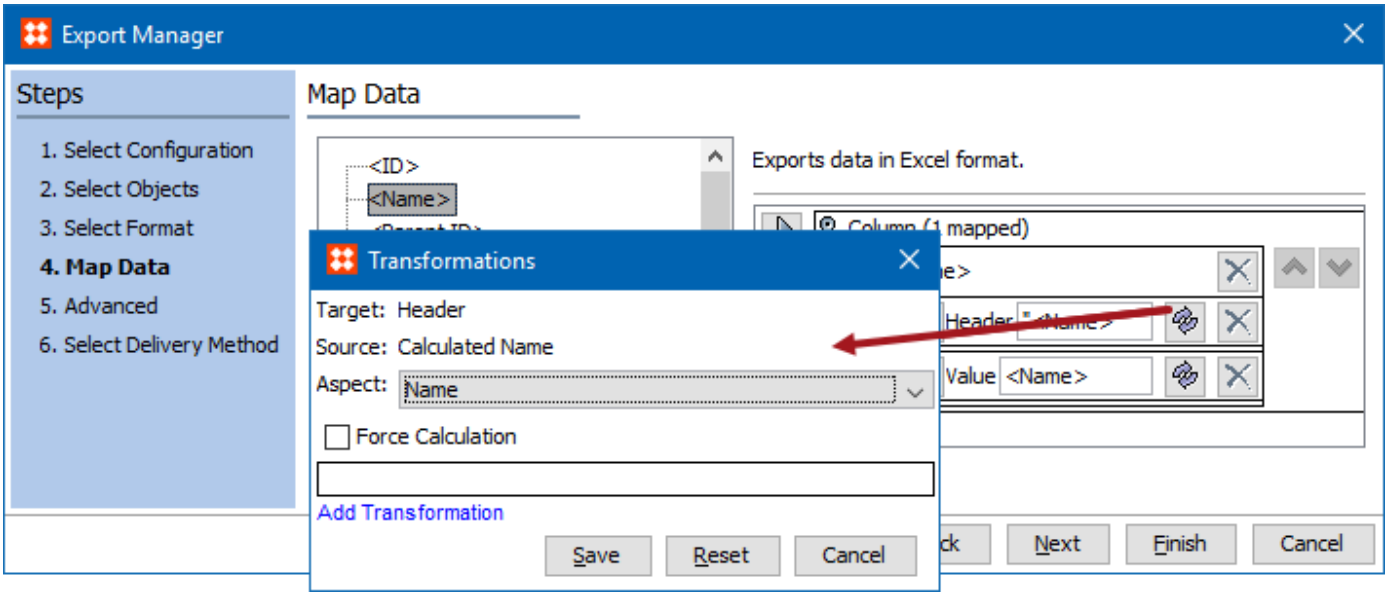
Transforming Data

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)

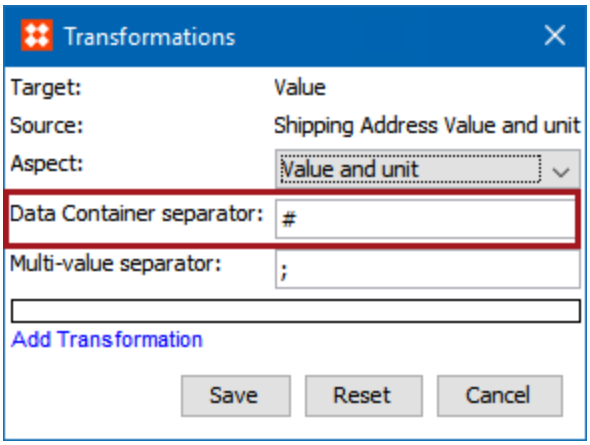
For details on each of the available transformation options, see the **Transformations** topic in the **Resource Materials** online help.

For more information about the additional wizard steps, see **Creating a Data Export** or **Creating an Outbound Integration Endpoint**.

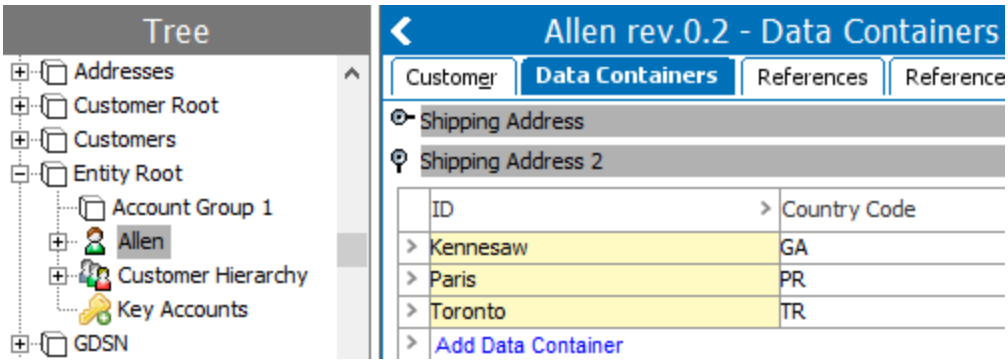
Data Container Separator - Transform Outbound

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.

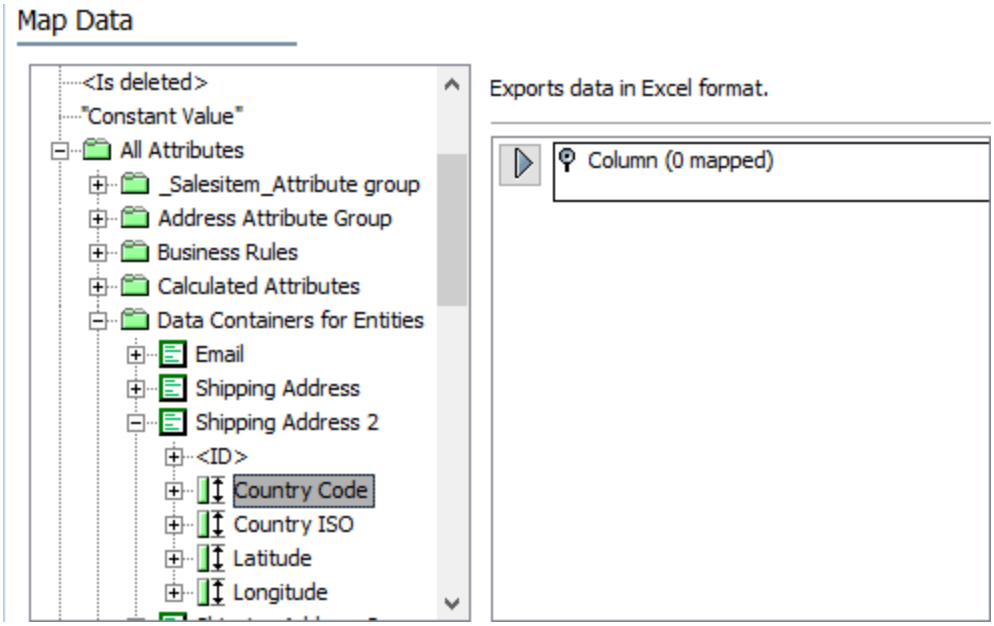


Configuration

1. In the output tool, choose Entity from the object type dropdown, and select the entity to be exported. For more information, see the **Data Exchange** topic.
2. On the Map Data step, in the left panel, expand the **All Attributes** data source, then expand the **Data**

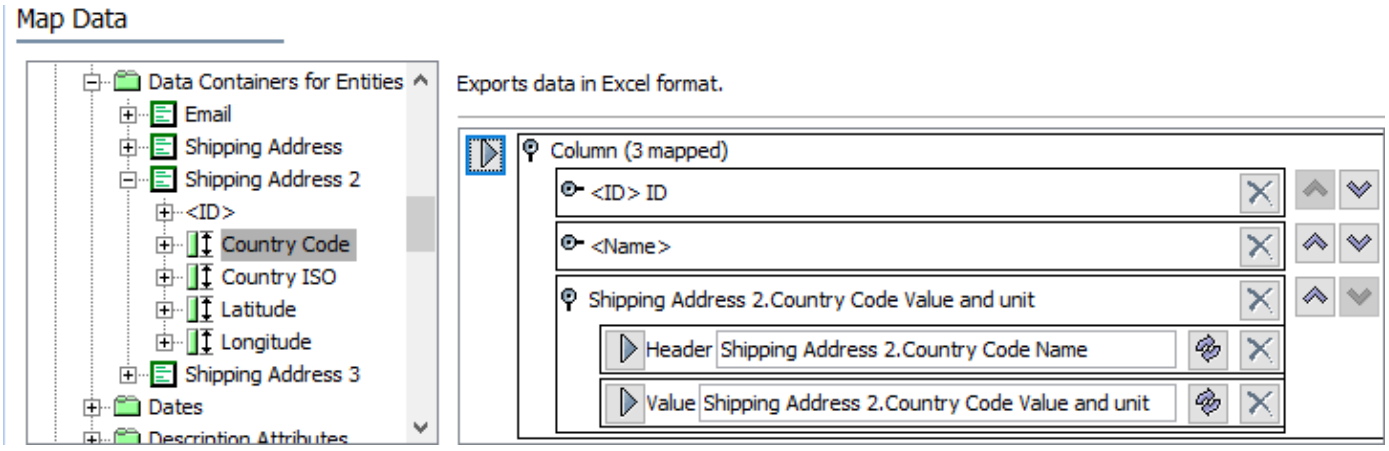
Containers for Entities group.

3. Expand the required data container, and then select the required attribute.



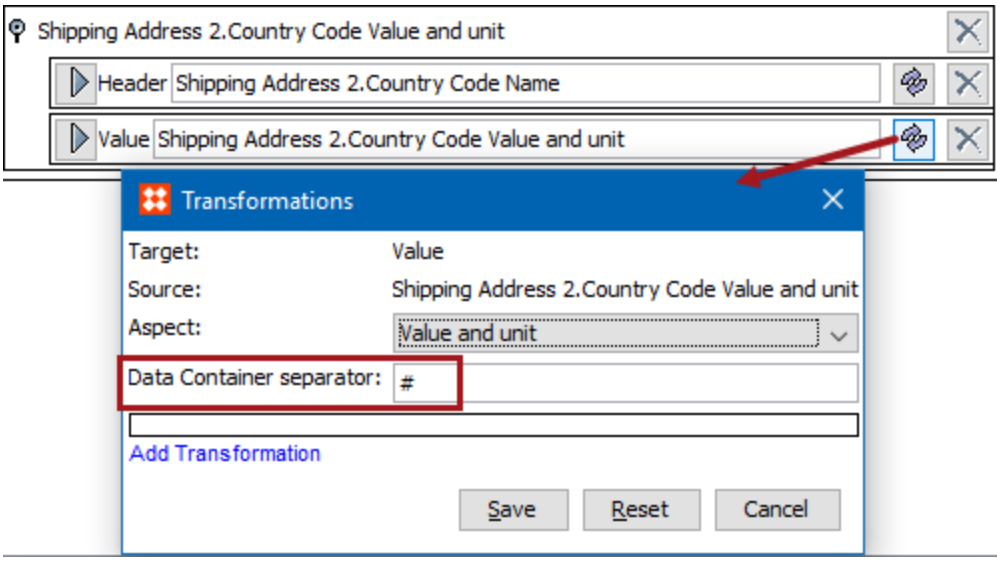
Important: The **All Attributes** data source can also be used to map the Data Container itself and get all of the attribute values associated with the data container. The **Select Attribute** data source does not give access to the data container type, nor the attributes.

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. See the **Outbound Map Data - Transform** topic.
9. Complete any additional mappings and initiate the export.

Results

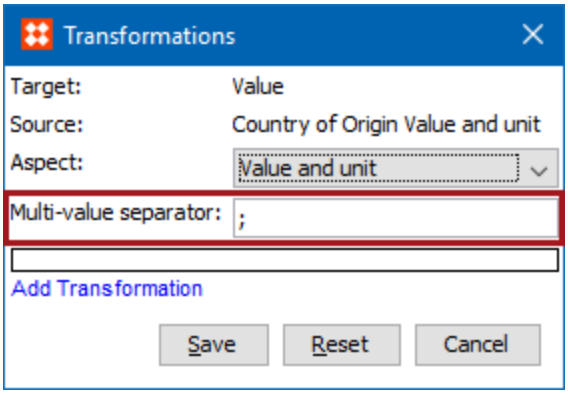
Using the data and configuration in the example above, the output of the entity object includes the entity ID and name, as well as the values in the 'Shipping Address 2' data container, separated by the default separator.

	A	B	C
1	<ID>	<Name>	Shipping Address 2.Country Code
2	CUS_114563	Allen	GA#TR#PR

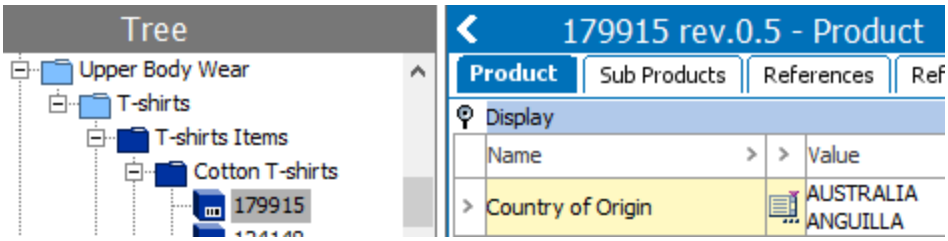
Multi-Value Separator - Transform Outbound

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.

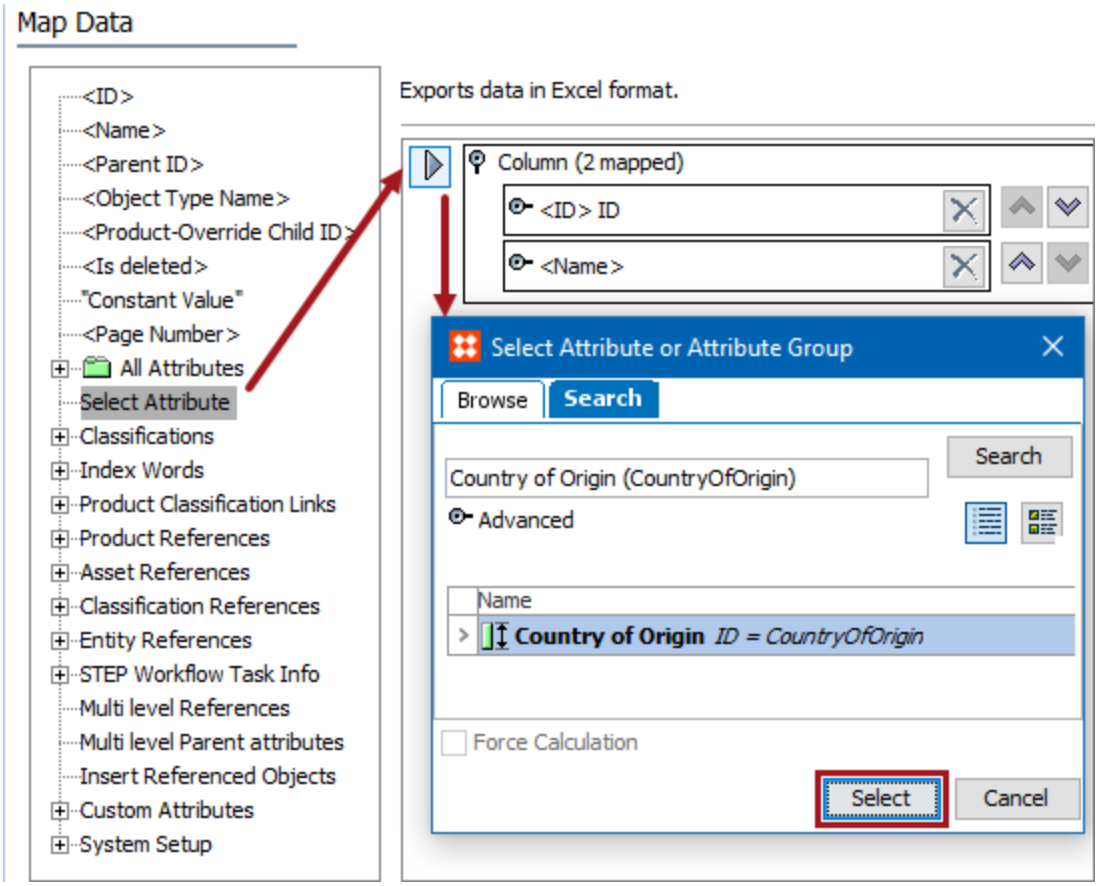


In this example, the product '179915' has a multivalued attribute 'Country Of Origin', which includes multiple values. The 'Multi-value separator' parameter is used to determine the text that is exported between the values.



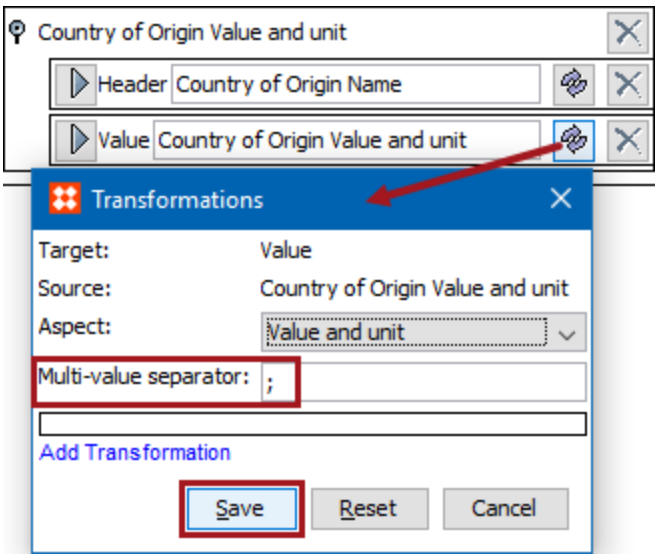
Configuration

1. In the output tool select the data to be exported. For more information, see 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. See the **Outbound Map Data - Transform** topic.
9. Complete any additional mappings and initiate the export.

Results

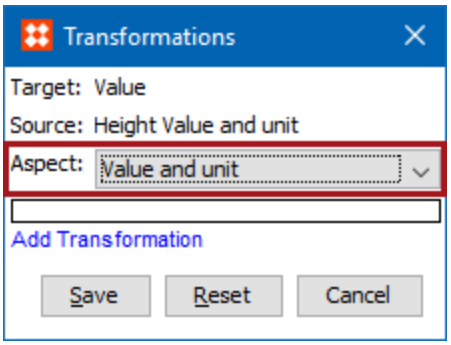
Using the data and configuration in the example above, the output of the product includes the ID and name, as well as the values in the 'Country of Origin' attribute, separated by the selected separator.

	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.

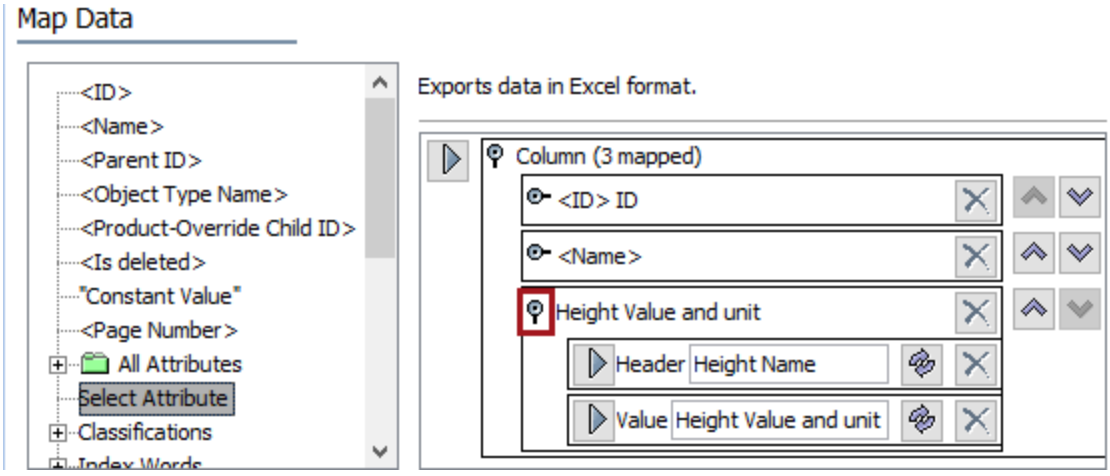
Important: 'Aspect' is only visible in the **Transformations** dialog for the data sources included in the **Data Sources with Aspect** below.



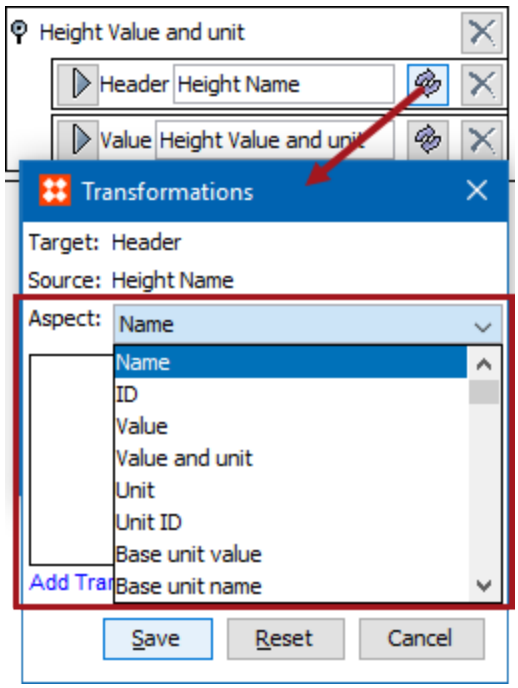
For a general example of mapping aspects, see the **Header and Value Aspects Example**. For specific examples of the available aspects, see the topics referenced in the **Aspect Data Elements Options** table below.

Configuration

1. In the output tool, select the data to be exported. For more information, see the **Data Exchange** topic.
2. On the Map Data step, in the left panel, select the data source. For more information, see 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, see 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, see 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. See 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**.

Aspect Data Element	Description
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**.
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	Extracts multiple files in a .ZIP file including the name of the digital

Aspect Data Element	Description
Name	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. 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, see the **MIME Types** topic of the **System Setup / Super User Guide** documentation.

Aspect Data Element	Description
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

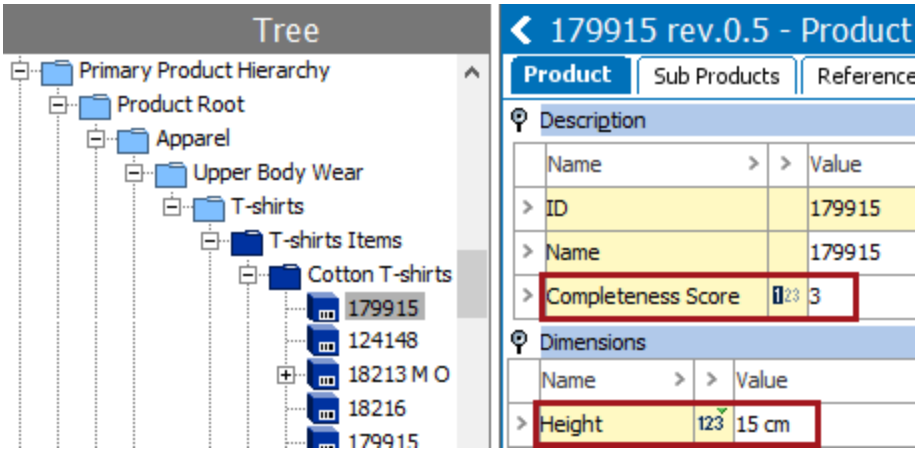
Attribute Meta Data Aspect

Metadata (description) attributes that are available on the mapped attribute are displayed with the prefix 'Attribute Meta Data.'

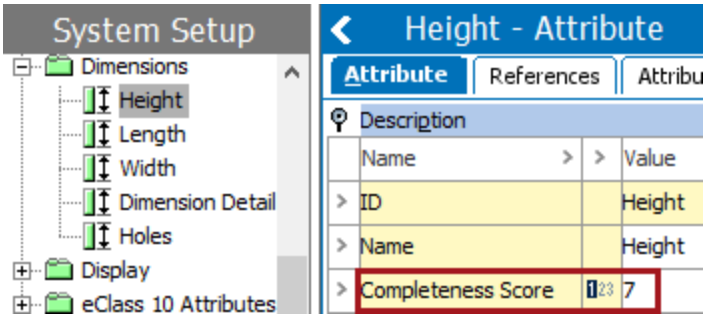
For general information on setting a Transformation aspect, see the **Aspect - Transform Outbound** topic.

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.

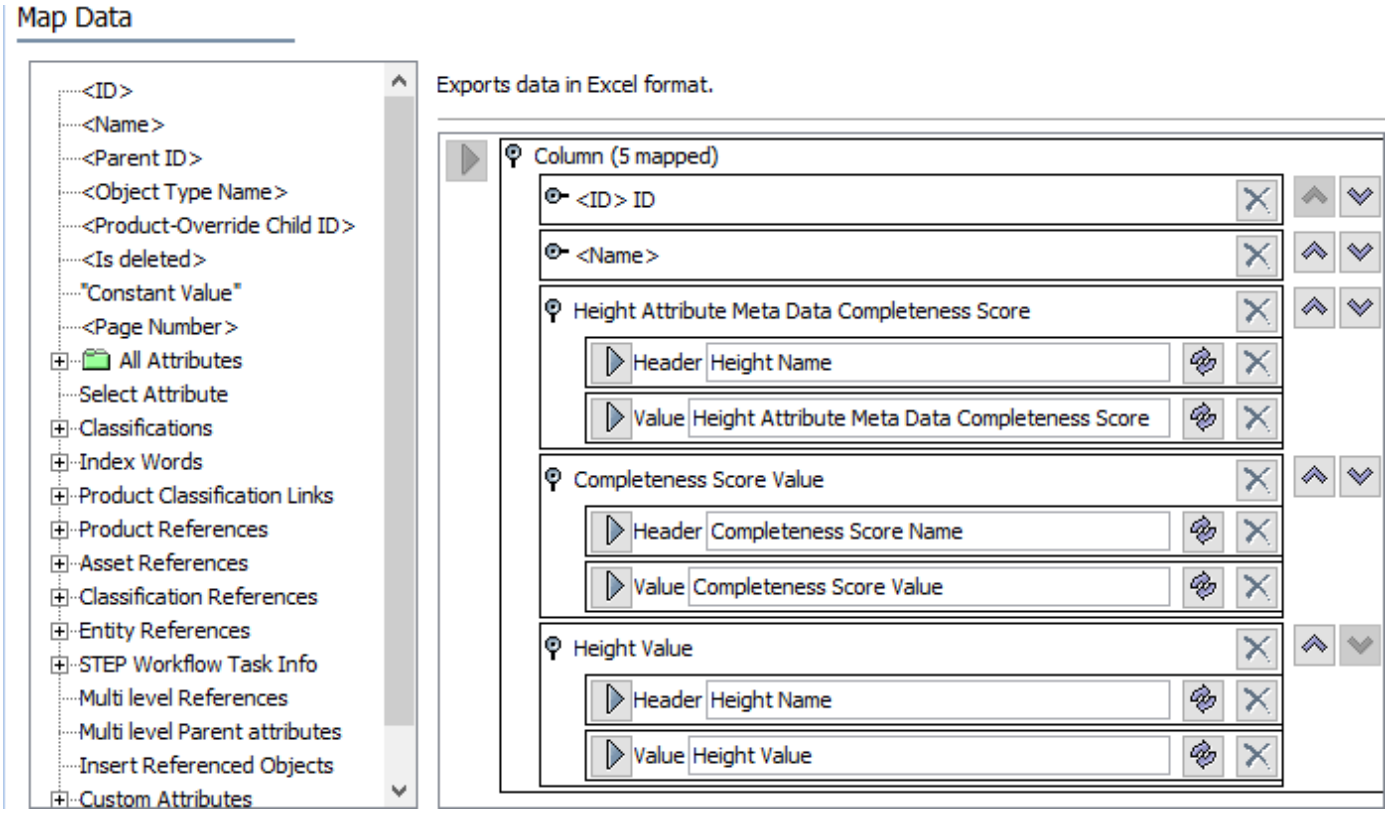


Configuration

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

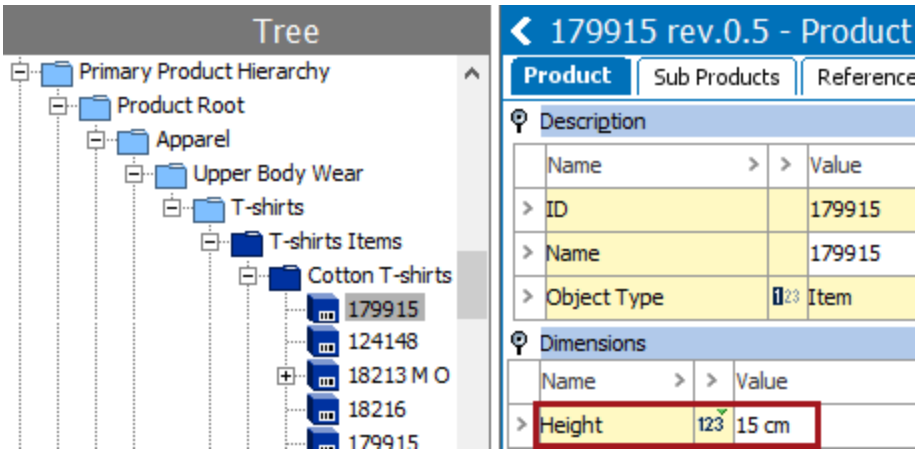
Attribute Type Aspect

The Validation Base Type of an attribute can be extracted using the 'Attribute Type' aspect. For more information on validation base types, see the **Validation Rules** topic in the **System Setup / Super User Guide**.

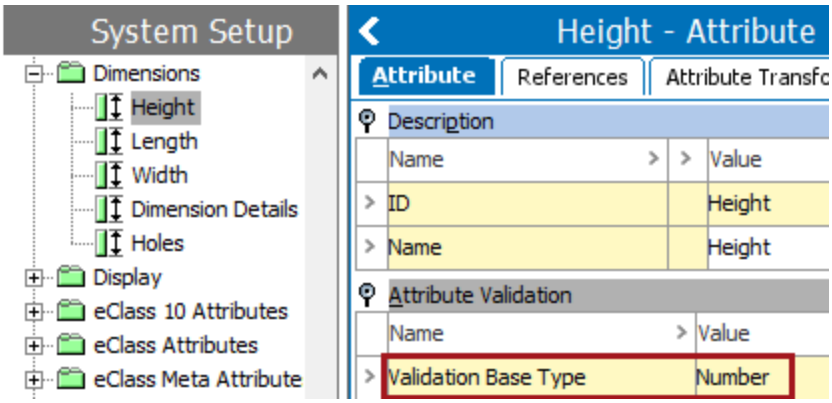
For general information on setting a Transformation aspect, see the **Aspect - Transform Outbound** topic.

For example, the description attribute 'Completeness Score' is valid on attributes and on products.

In the Tree, the product 179915 has the value of 15 for the Height attribute.



In System Setup, the 'Height' attribute has a Validation Base Type of Number.

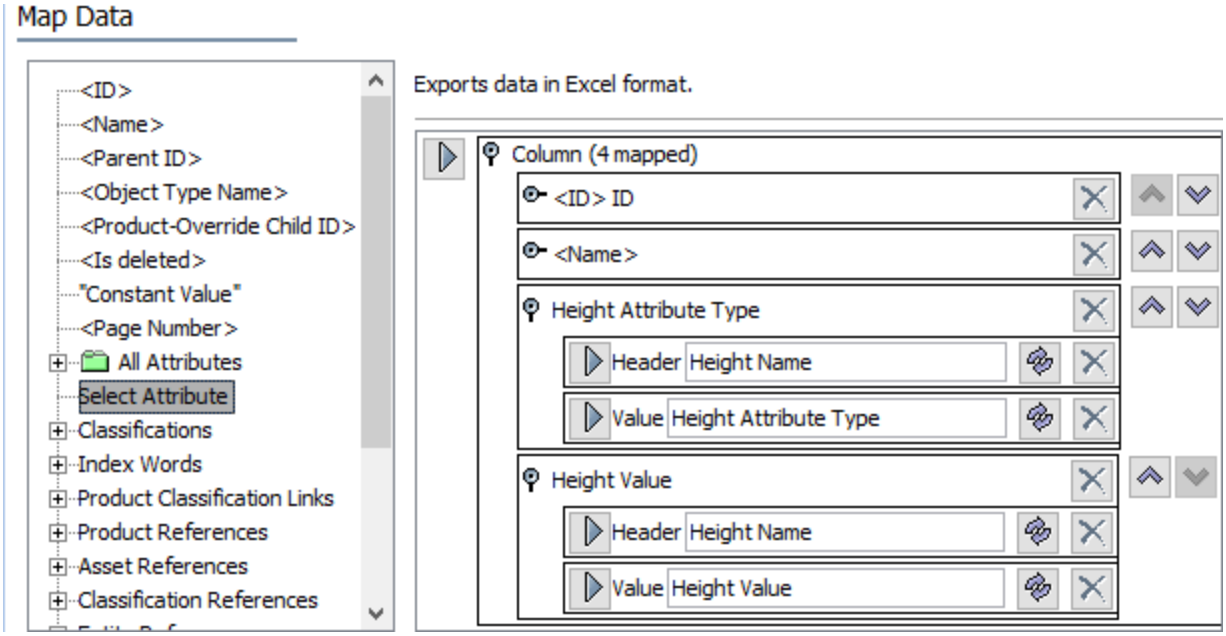


Configuration

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

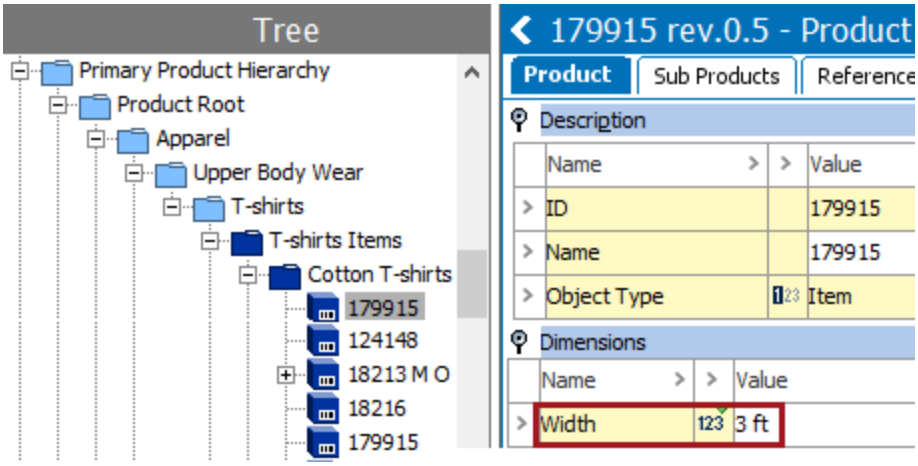
Base Unit ID, Name, and Value Aspects

Data for attributes that include a unit can be extracted using the following base unit aspects:

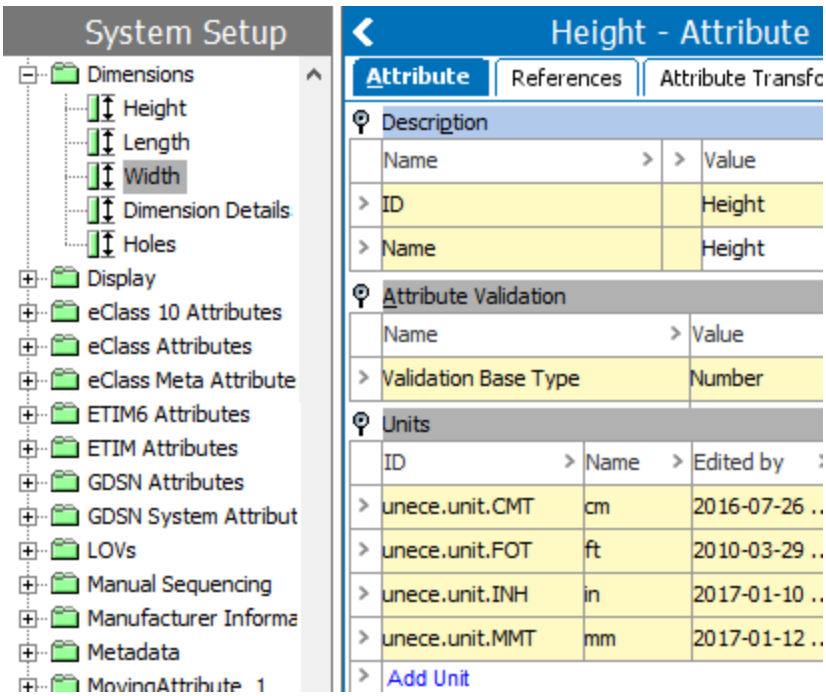
- **Base Unit ID** - Extracts the ID of the base unit that is associated with the unit of the attribute value.
- **Base Unit Name** - Extracts the name of the base unit that is associated with the unit of the attribute value.
- **Base Unit Value** - Extracts the attribute value according to the base 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, see the **Aspect - Transform Outbound** topic.

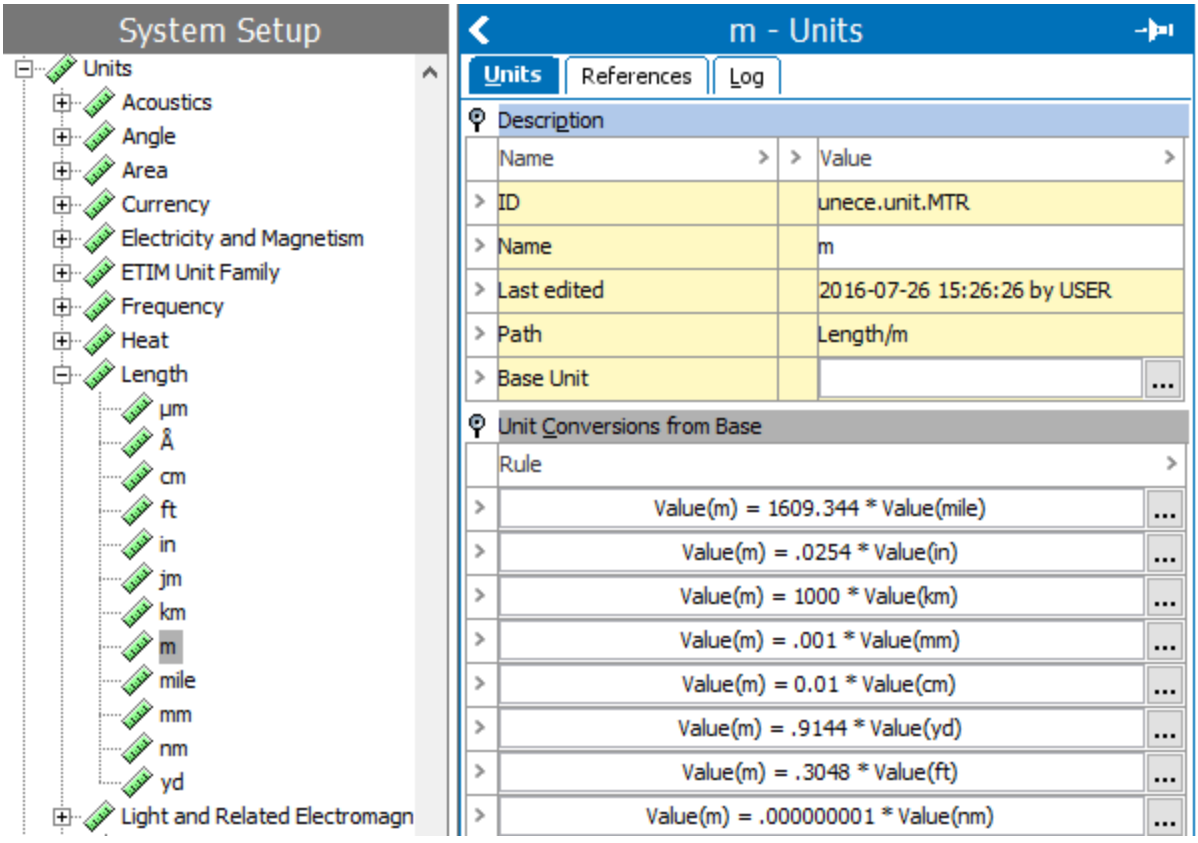
For example, for the Width attribute on a product, the value '3' is associated with the unit 'ft.'



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, see the **Units** topic in the **System Setup / Super User Guide**.

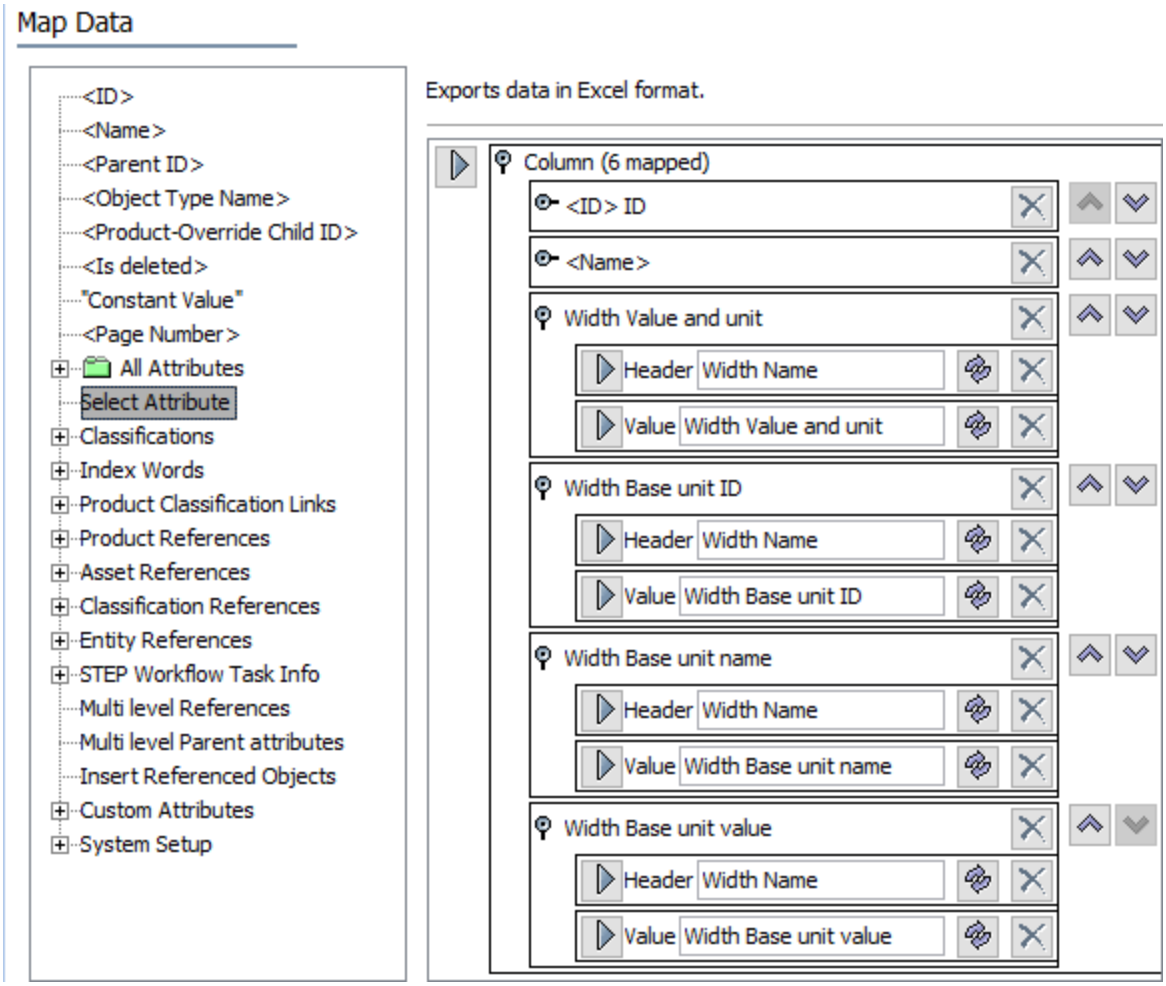


Configuration

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.

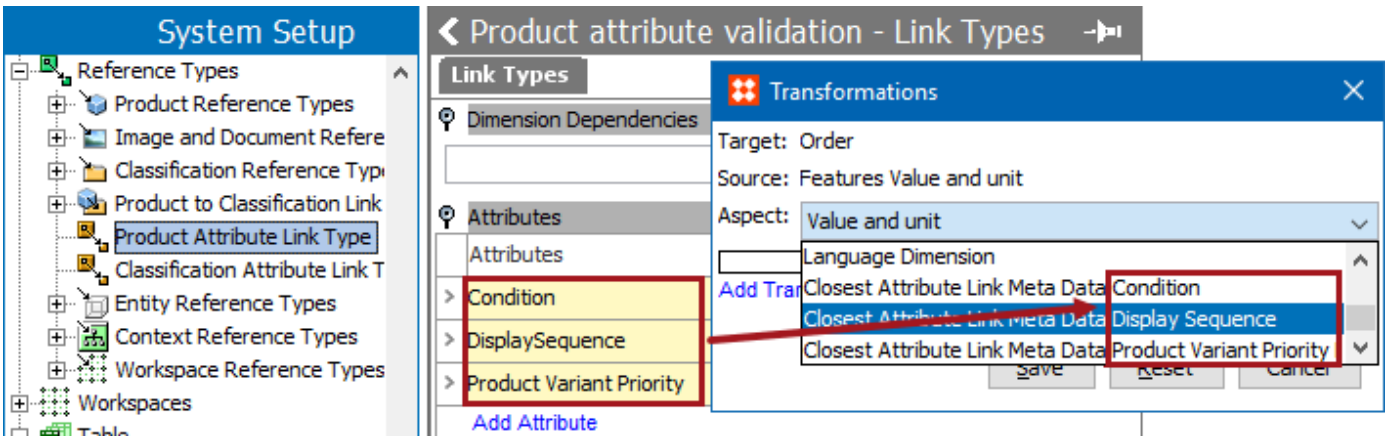
For more information on BMEcat 1.2, see the **BMEcat Format** topic. For more information on BMEcat 2005, see the **BMEcat 2005 Format** topic.

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.

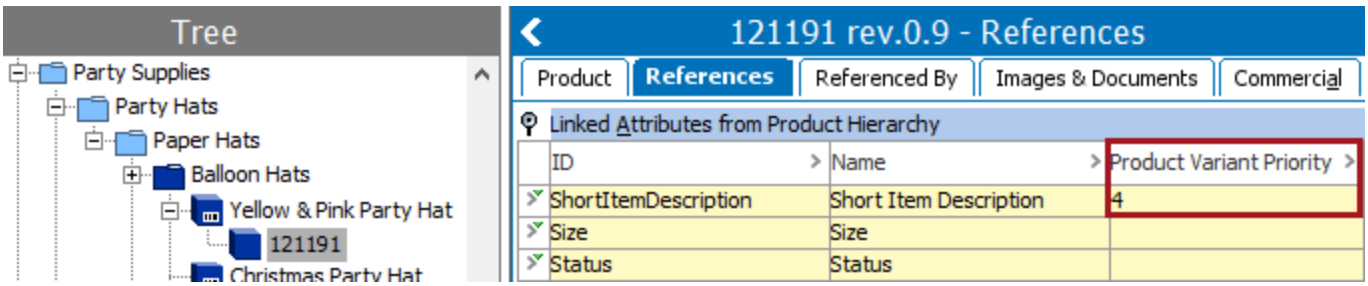
For general information on setting a Transformation aspect, see the **Aspect - Transform Outbound** topic.

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.

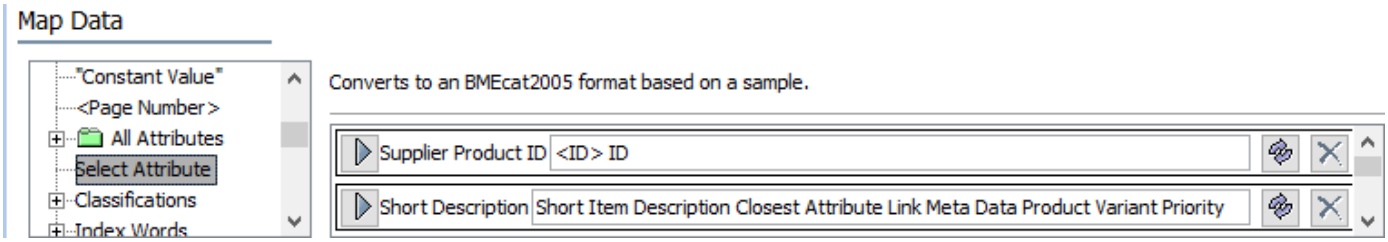
In the tree, product 121191 has an inherited value of '4' for the Product Variant Product metadata attribute displayed on the References tab under the Linked Attributes from Product Hierarchy flipper.



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

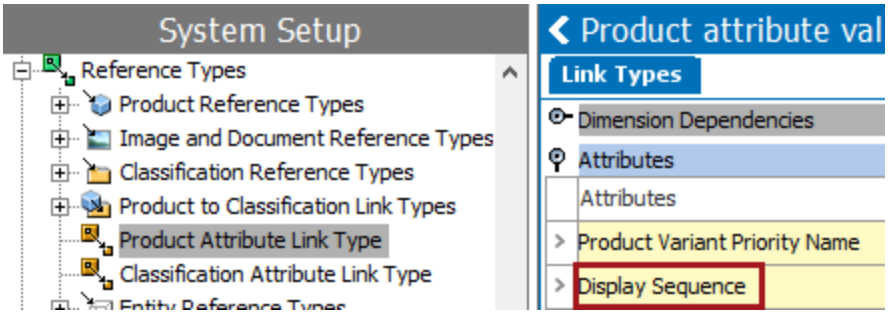
BMEcat 2005 is the only format that allows access to Product Attribute Link metadata attribute values, via the Closest Attribute Link Meta Data transformation aspect.

Inheritance of the Display Sequence attribute value used as metadata varies based on the configuration as follows:

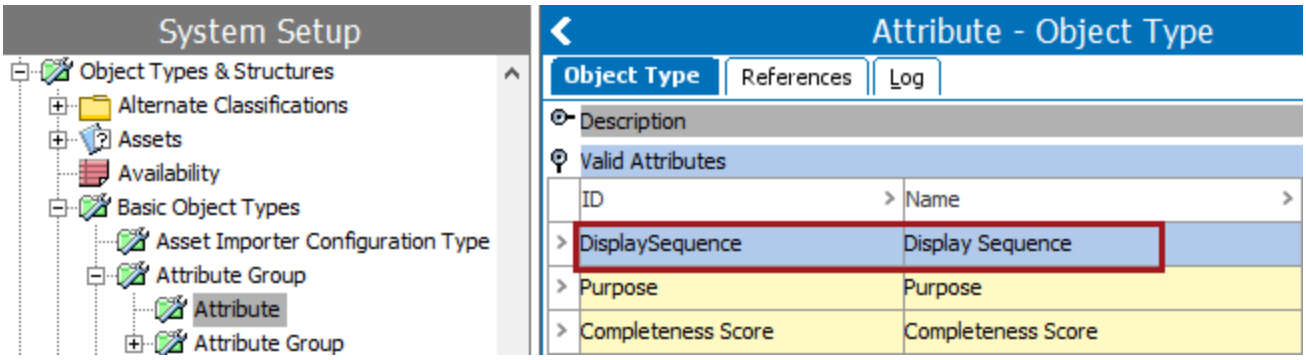
- Using the standard setup of a display sequence attribute, both explicit and inherited values via the Product Attribute Link are exported as expected. Standard setup is defined in the **Attribute Display Sequence** topic of the **Getting Started / User Guide** documentation.
- Configuring the Display 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:

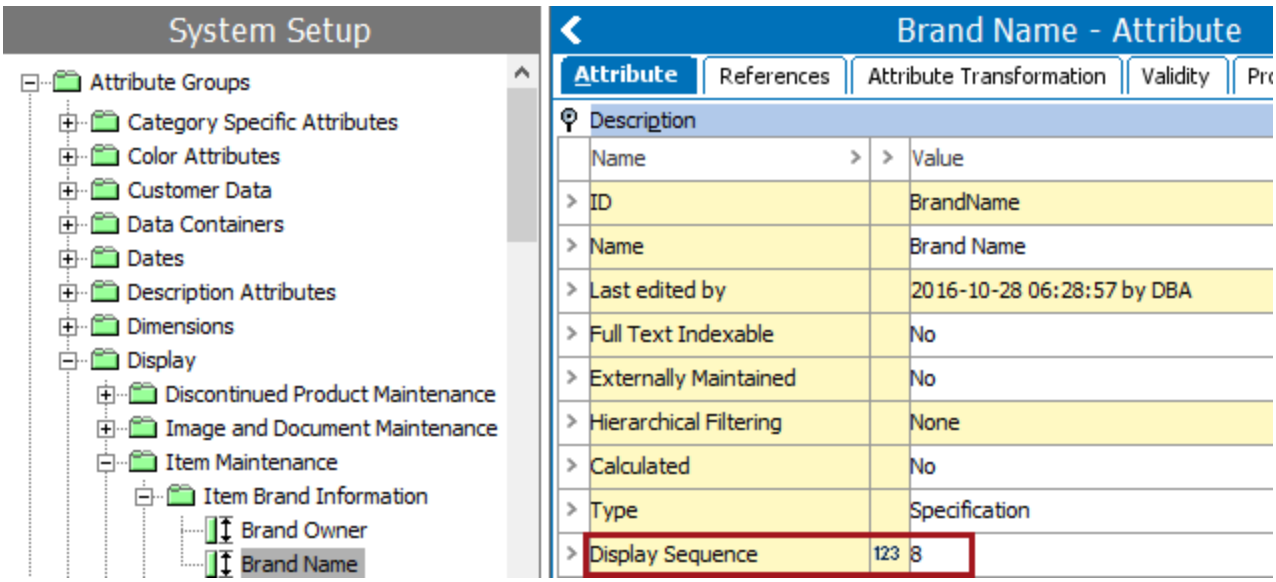
The Product Attribute Link Type includes a metadata attribute with the name Display Sequence. This is the attribute identified in System Setup as **Default Attribute to use as Attribute Display Sequence Attribute**.



The same attribute is valid as metadata on the attribute object type (below the Object Types & Structures node in System Setup).



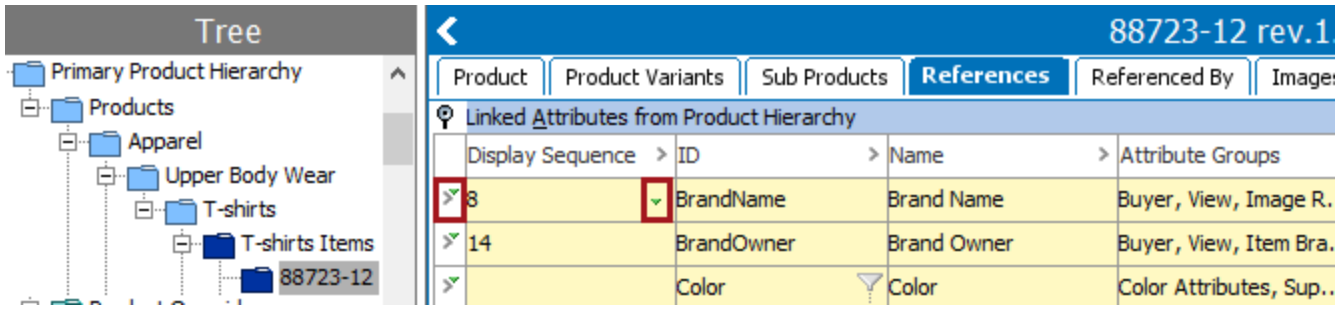
The value is then set on an attribute that is valid for a product.



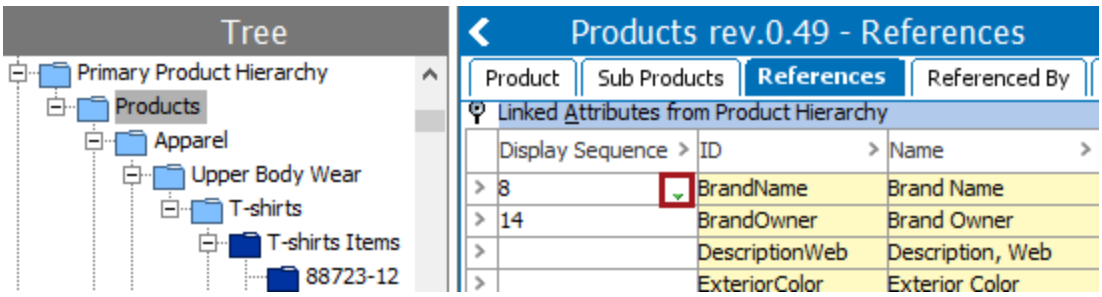
The value from the attribute is displayed on a product on the References tab, under the Linked Attributes from Product Hierarchy flipper.

Notice the two small green down arrows:

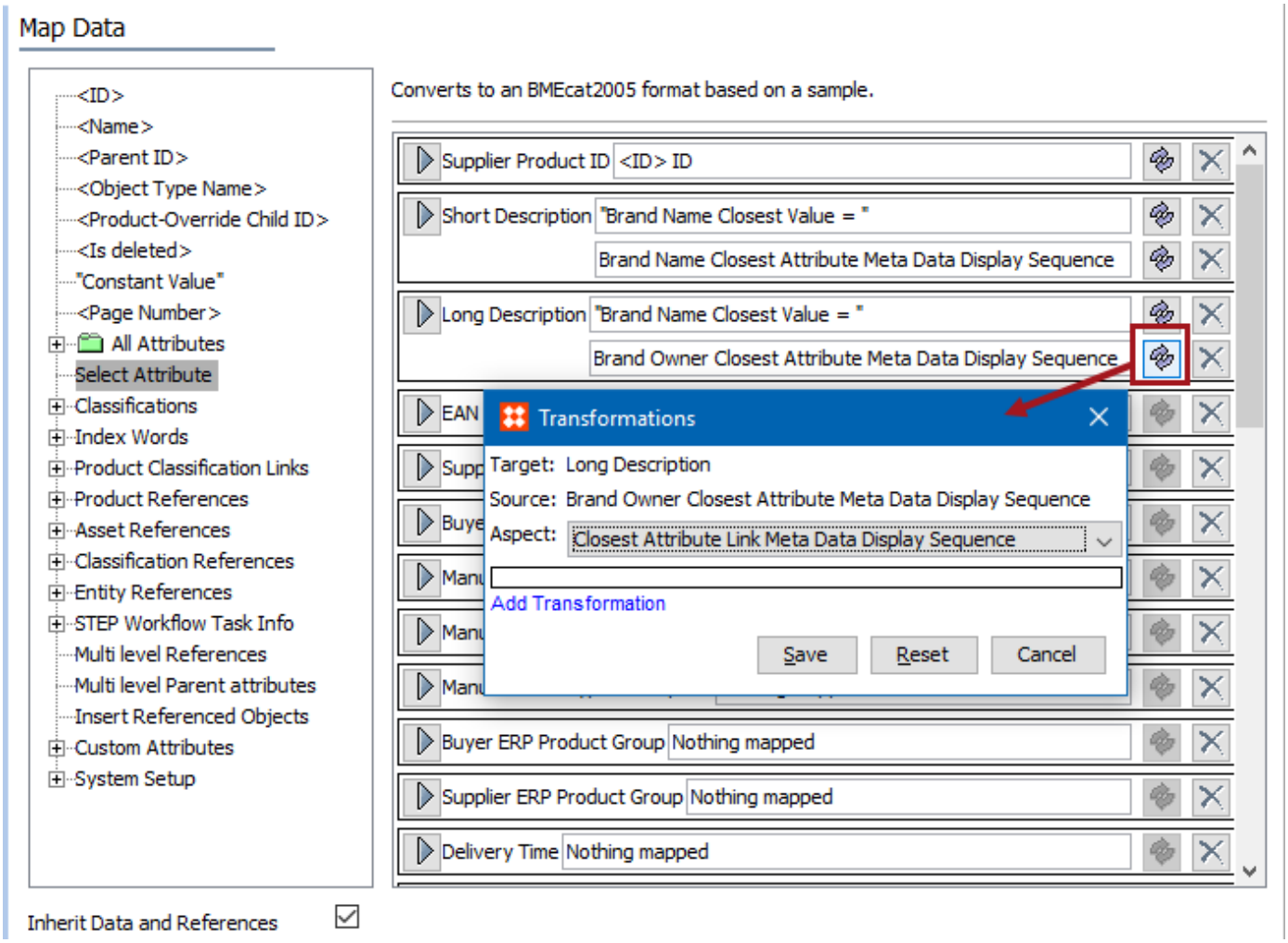
- The arrow in the first column (to the left of the Display Sequence column) indicates that the value is inherited from the location higher up the Tree that is identified by the Inherited From column.
- The arrow within the attribute value field indicates that the value is inherited from the attribute.



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/>
```

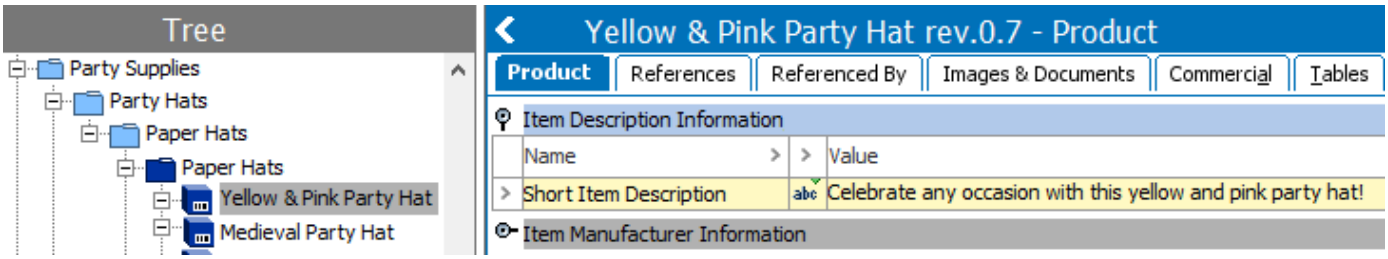
Dimension Dependencies Aspects

Data for the dimension dependencies parameter on attributes can be extracted using the following dimension dependency aspects:

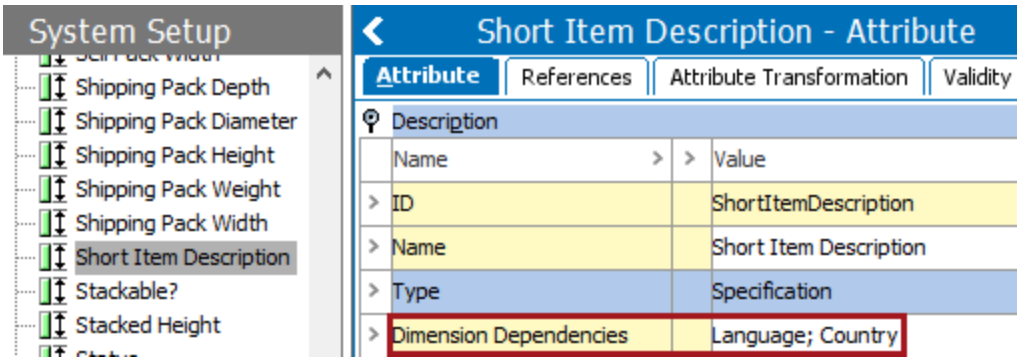
- **Dimension Dependencies** - Extracts the names of the dimension dependencies on the selected attribute.
- **[*name*] Dimension** - Extracts the ID of the dimension point selected by name, for the selected attribute, based on the exported context.

For general information on setting a Transformation aspect, see the **Aspect - Transform Outbound** topic.

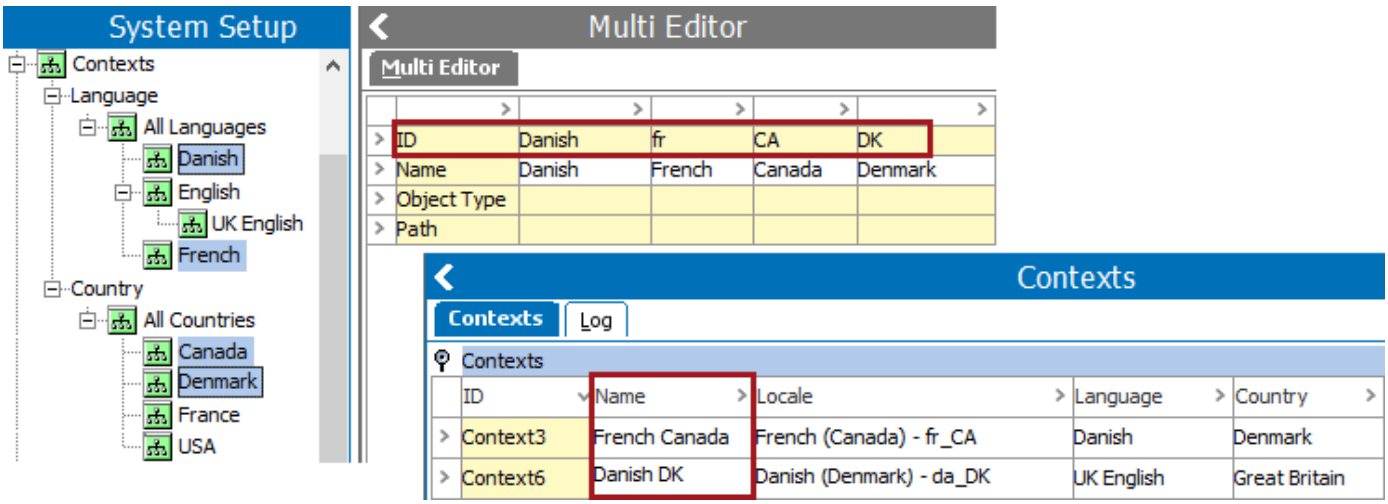
For example, the Tree displays that the Short Item Description attribute is valid on a product with ID 121179 and name 'Yellow & Pink Party Hat' and has a value in the contexts being exported.



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, see the **Dimensions, Dimension Points, and Contexts** topic in the **System Setup / Super User Guide**.

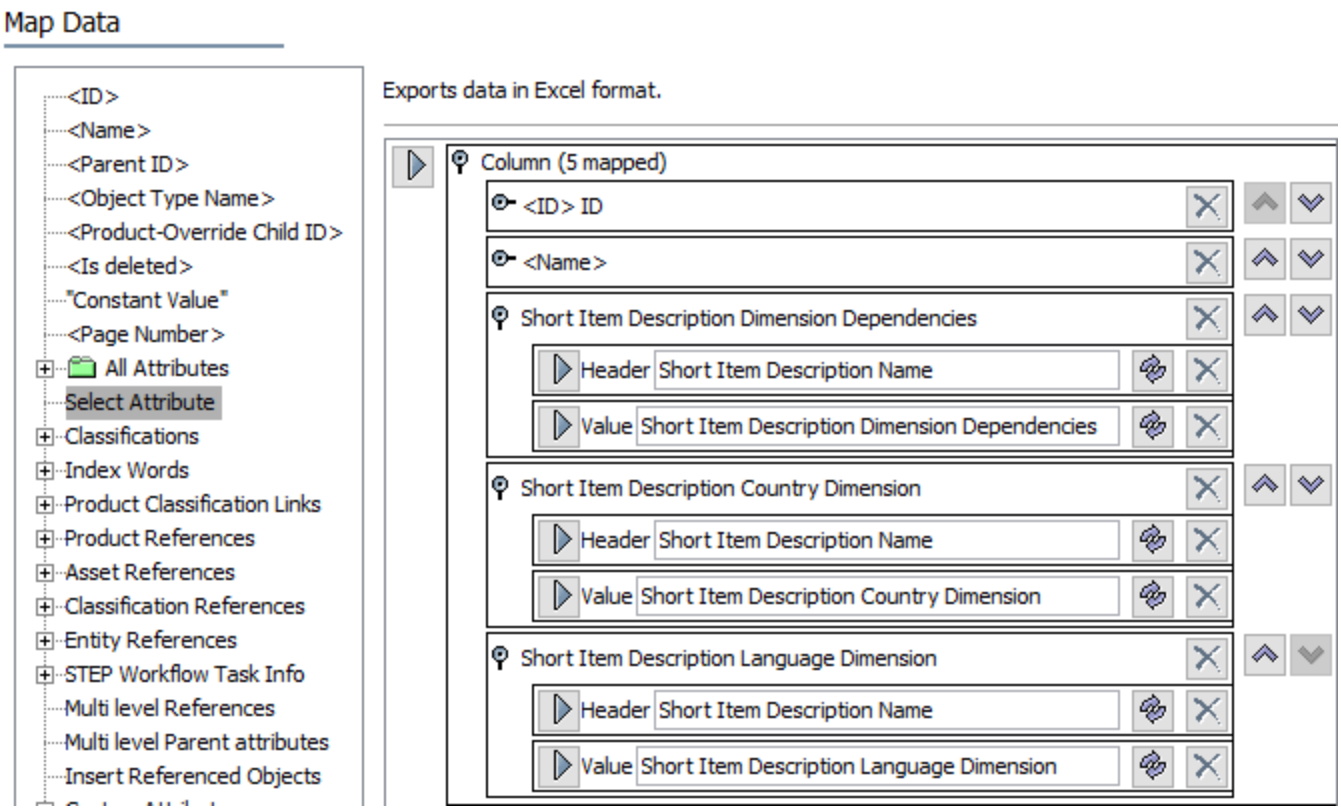


Configuration

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

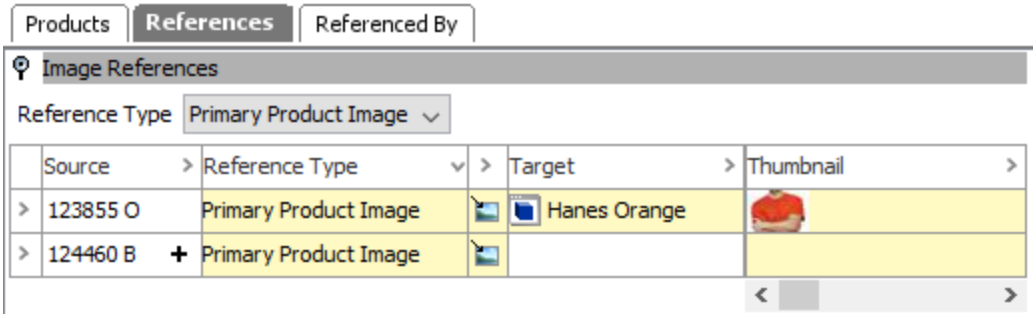
Is Deleted Aspect

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, see 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'.

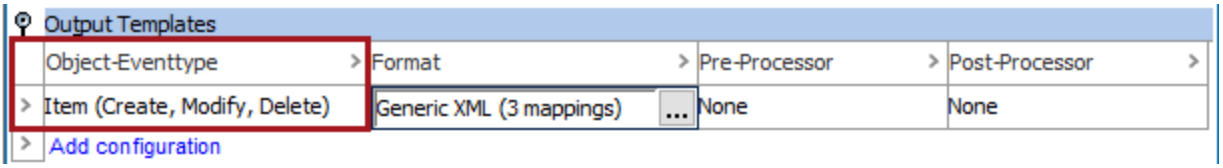


The Generic XML format output includes the 'IsDeleted' value only when the ID of the 'Primary Product Image' was deleted from the product.

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).



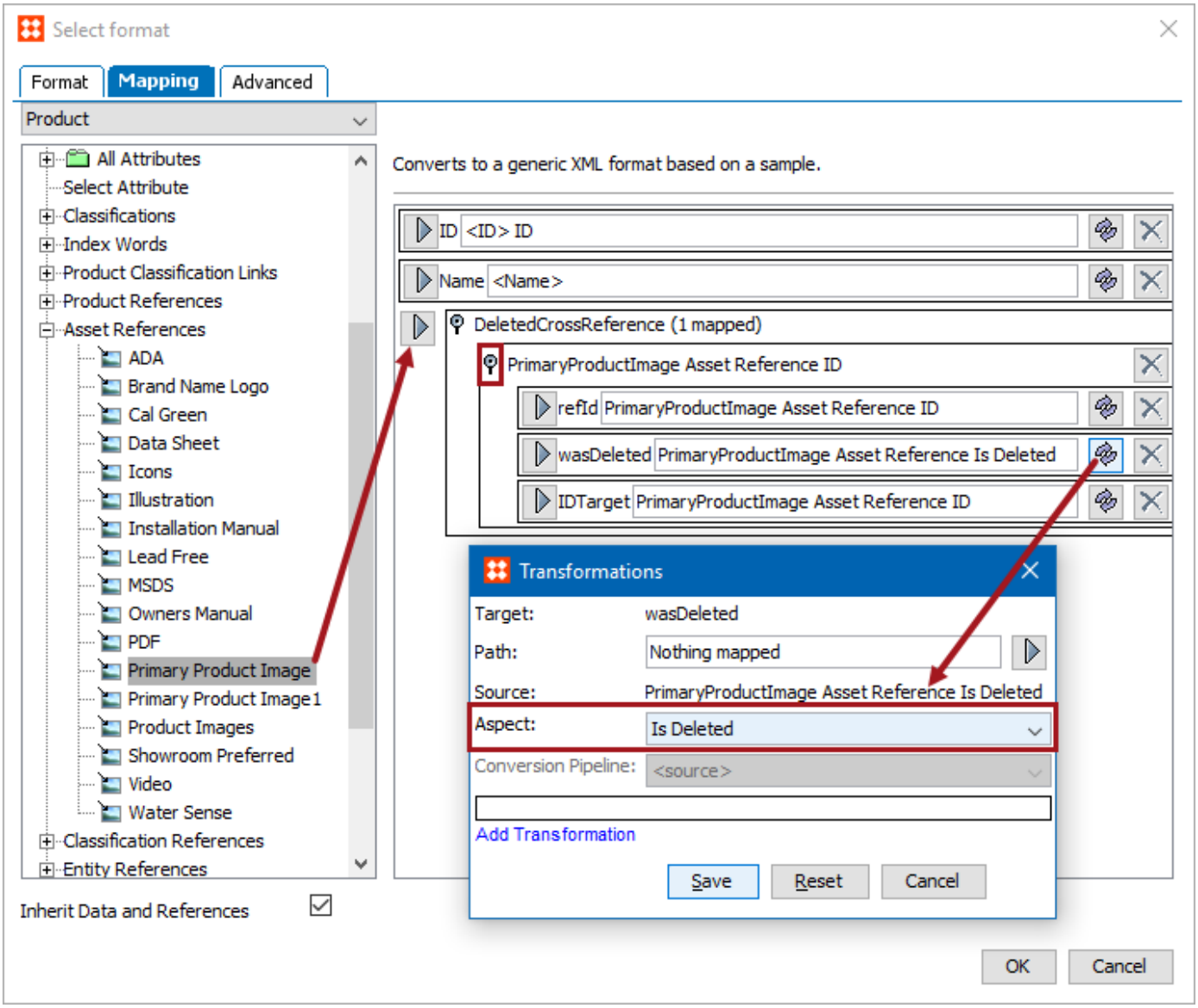
2. On the Configuration tab, on the Output Templates flipper, click into the **Format** parameter to display and click the ellipsis button (...).

   - On the Format tab, set the desired format and include a sample if required.

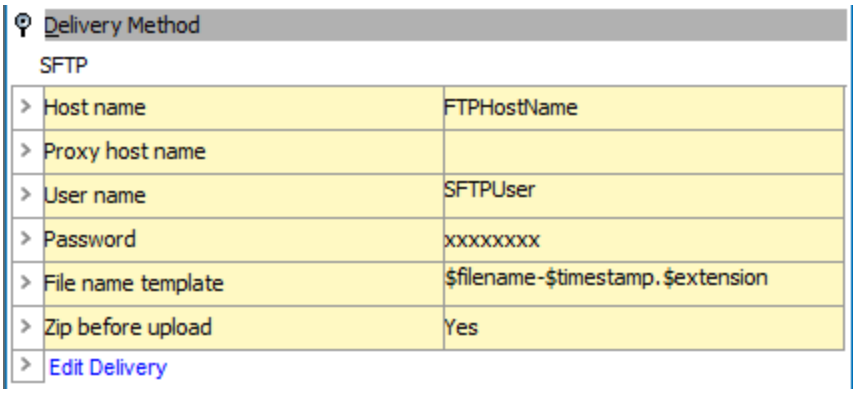
     For this example, the following Generic XML sample document is used:

```
<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>
```

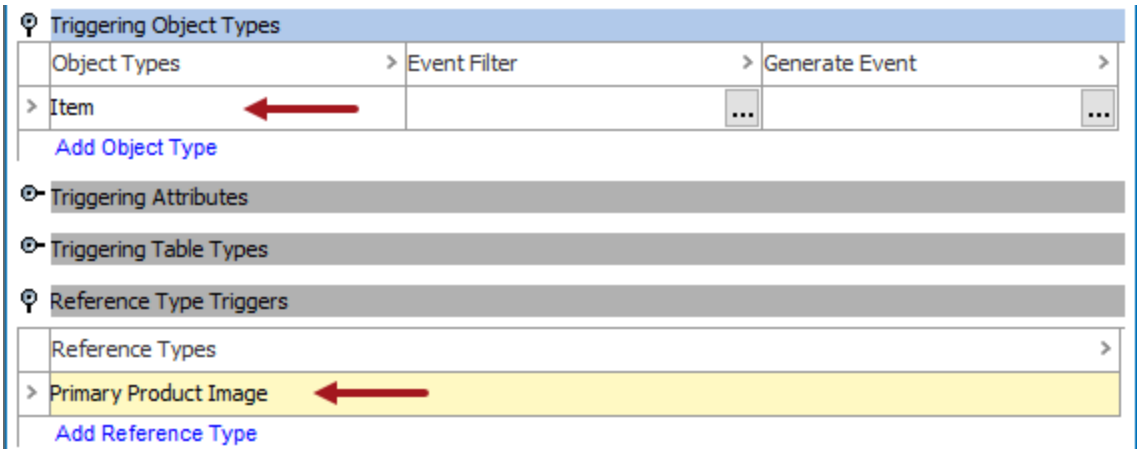
- On the Mapping tab, map the attributes, the reference type, and the transformation.



3. On the Configuration tab, on the **Delivery Method** flipper, set the delivery method parameters.



4. On the **Event Triggering Definitions** tab add the required object type(s) and reference type(s).



5. 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.

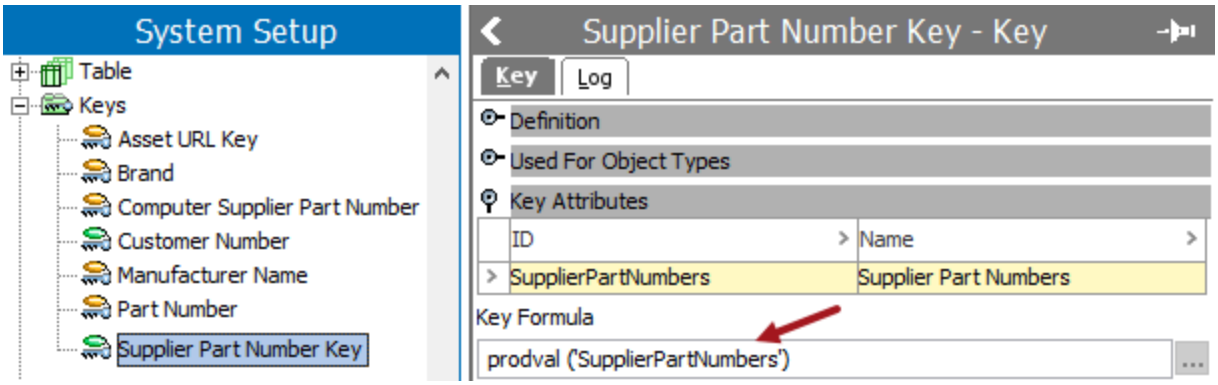
The ID data source is also used to define a unique key (in place of, or in addition to, the STEP ID) as required by external systems.

For general information on setting a Transformation aspect, see the **Aspect - Transform Outbound** topic.

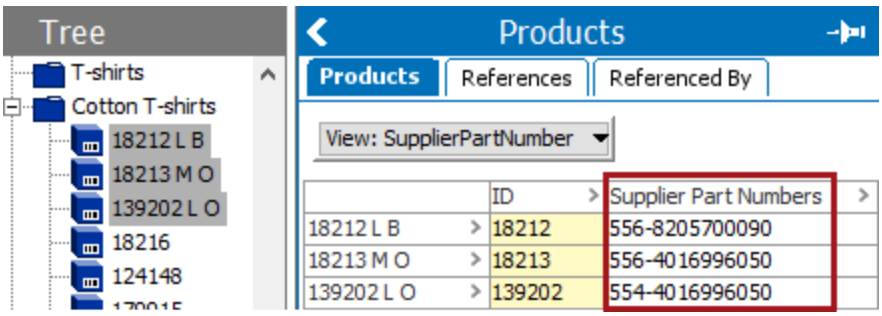
Prerequisite

Keys must be activated before they are displayed in the Map Data step of an export tool. For more information, see the **Unique Keys** topic of the **System Setup / Super User Guide** 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.



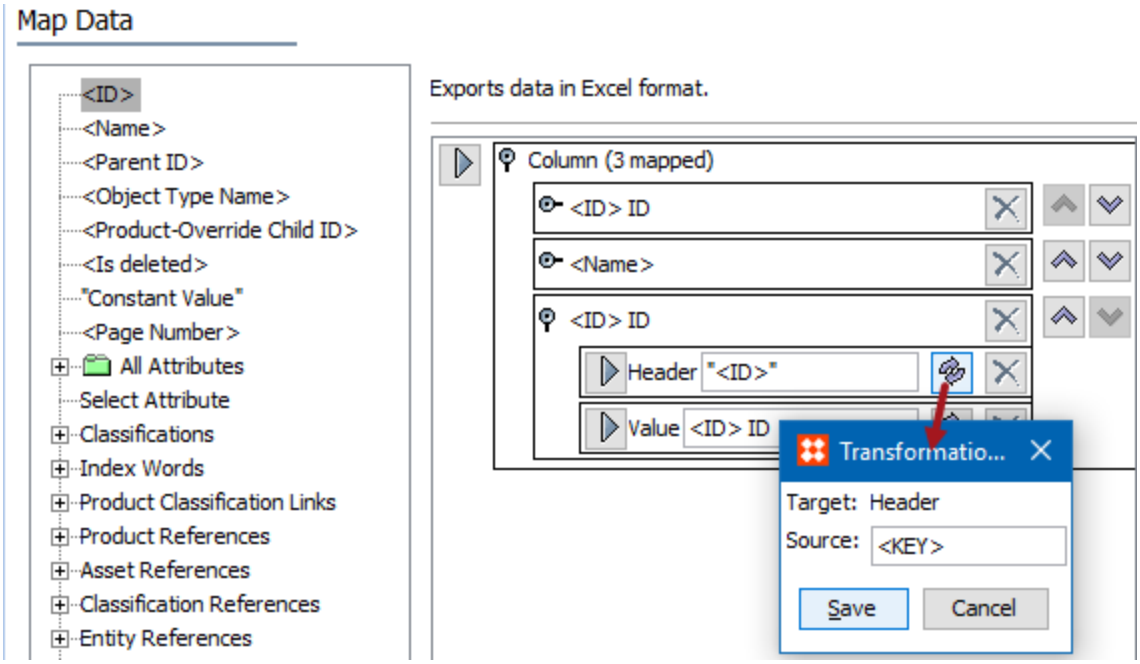
In the Tree, the products 18212, 18213, and 139202 have values for the attribute 'Supplier Part Numbers (SupplierPartNumbers)' which is used in the active unique key.



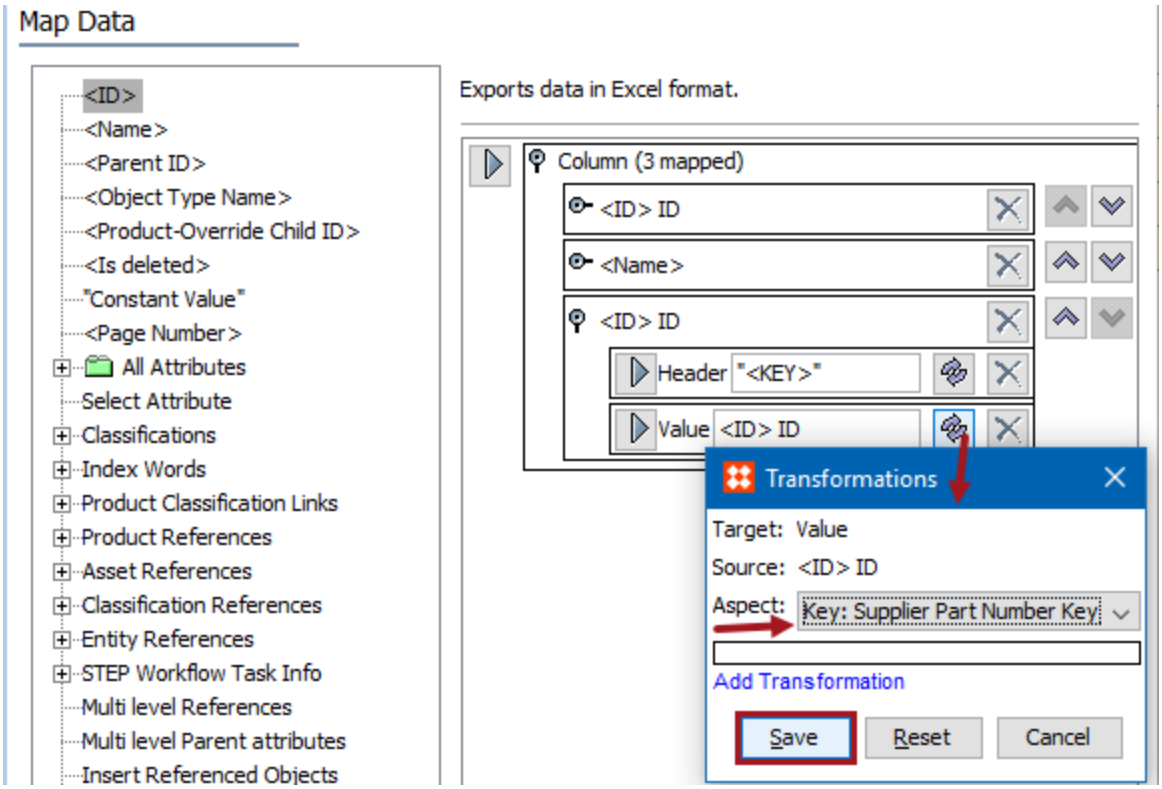
Configuration

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:

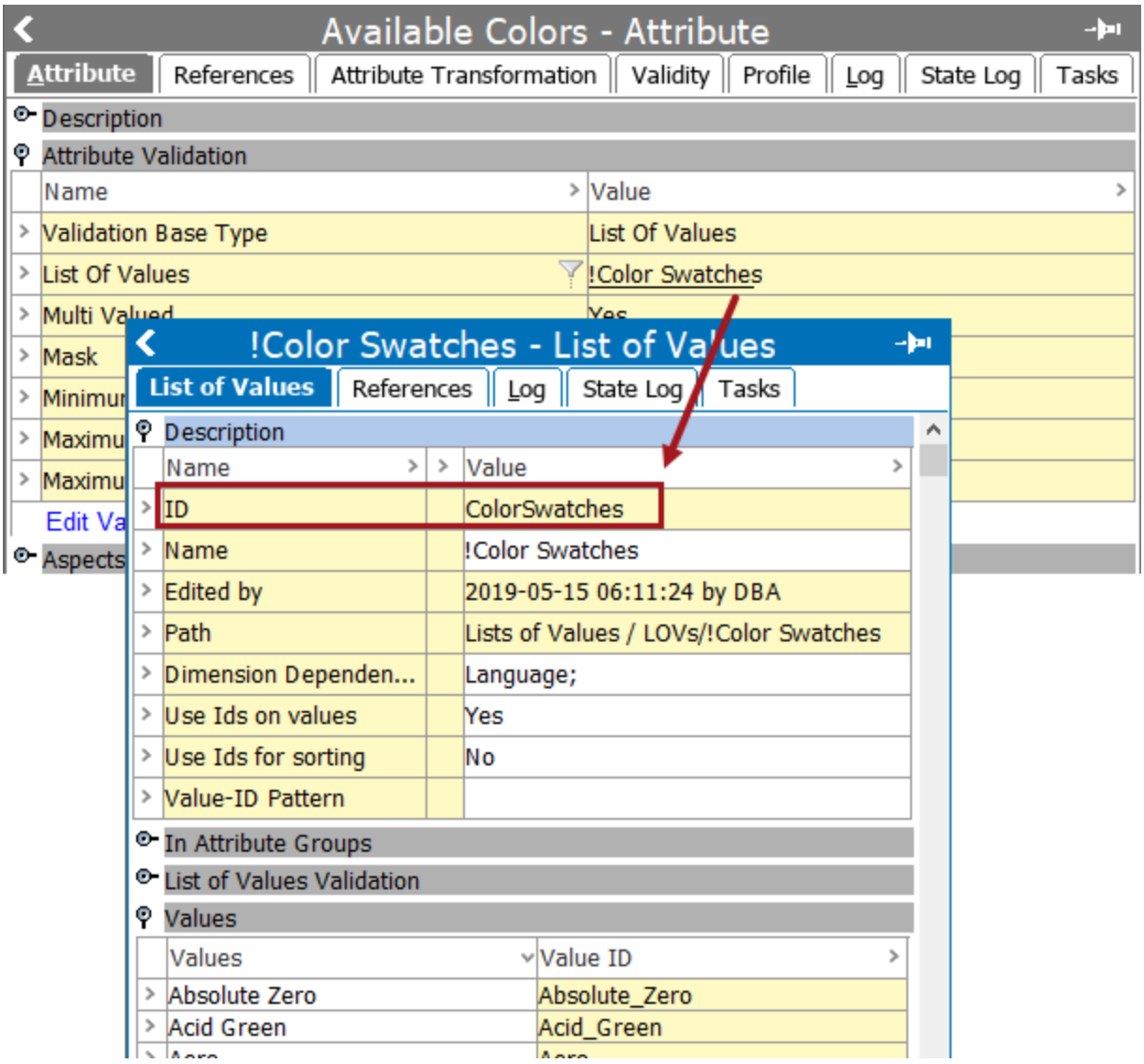
	A	B	C
1	<ID>	<Name>	<KEY>
2	18212	18212 L B	556-8205700090
3	18213	18213 M O	556-4016996050
4	139202	139202 L O	554-4016996050

LOV ID Aspect

The 'LOV ID' aspect allows for LOV IDs to be exported for the mapped LOV attributes. For details on LOVs , see the **List of Values (LOVs)** topic in the **System Setup / Super User Guide** documentation.

For general information on setting a Transformation aspect, see 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.

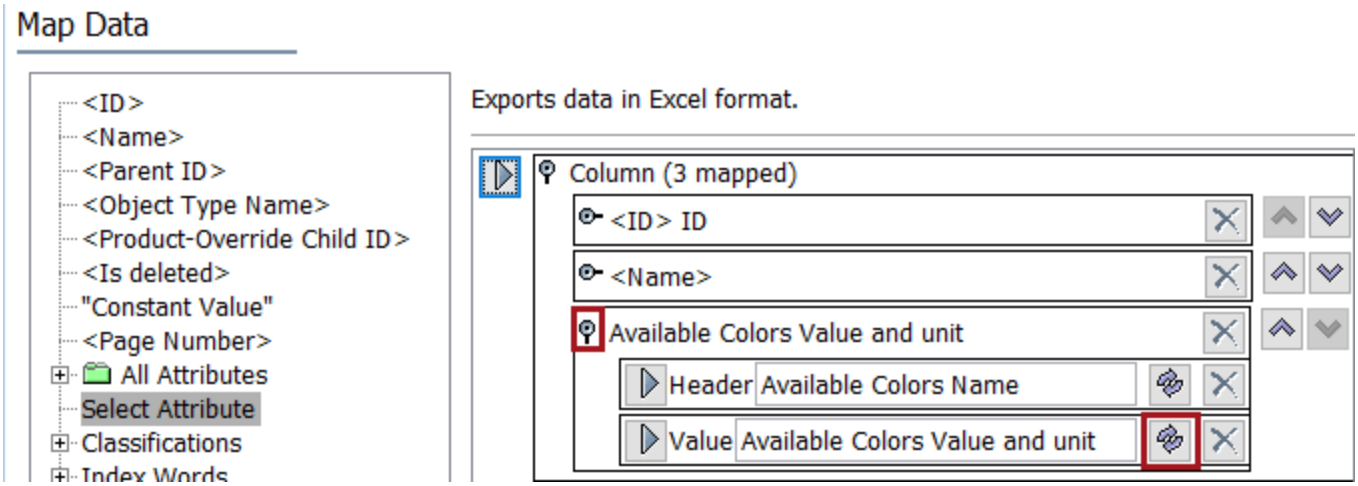


Additional options for exporting LOV aspect information are included in the following topics:

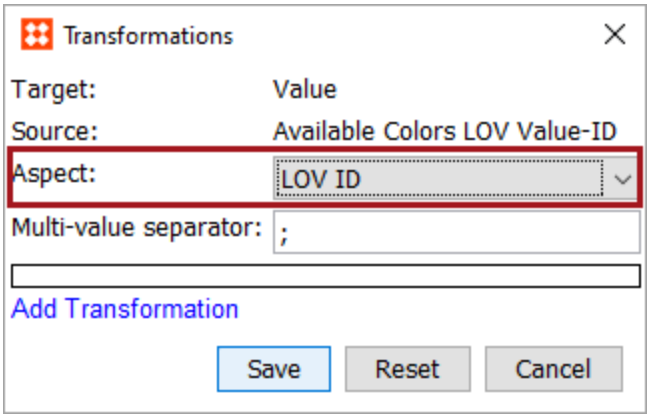
- LOV Value-ID Aspect
- LOV Value-ID or Value Aspect

Configuration

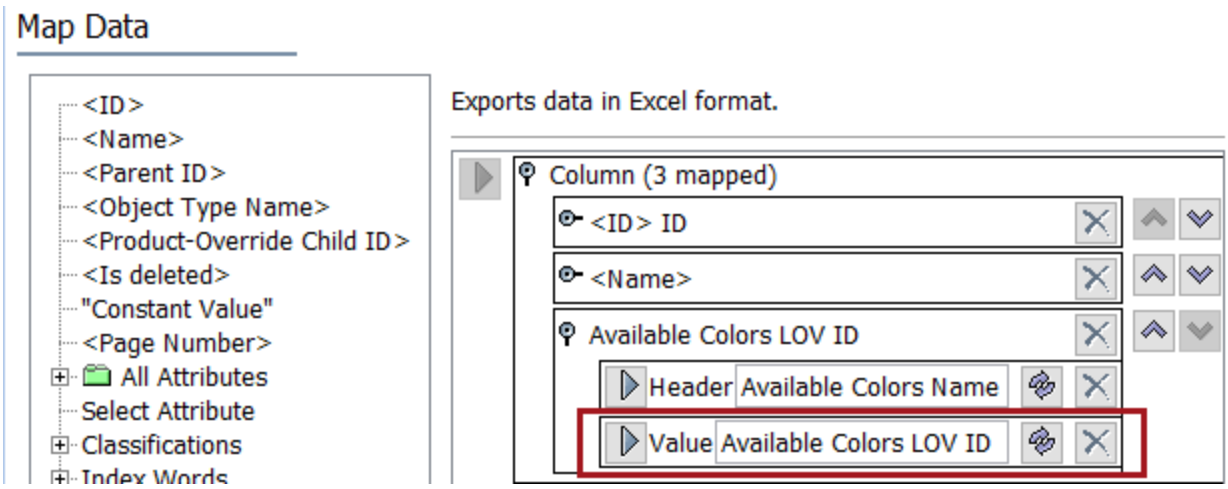
1. In the output tool, select the data to be exported. For more information, see the **Data Exchange** topic.
2. On the Map Data step, map the required attribute or attribute group using the **Select Attribute** data source. For more information, see **Mapping Attributes via Select Attribute** section 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

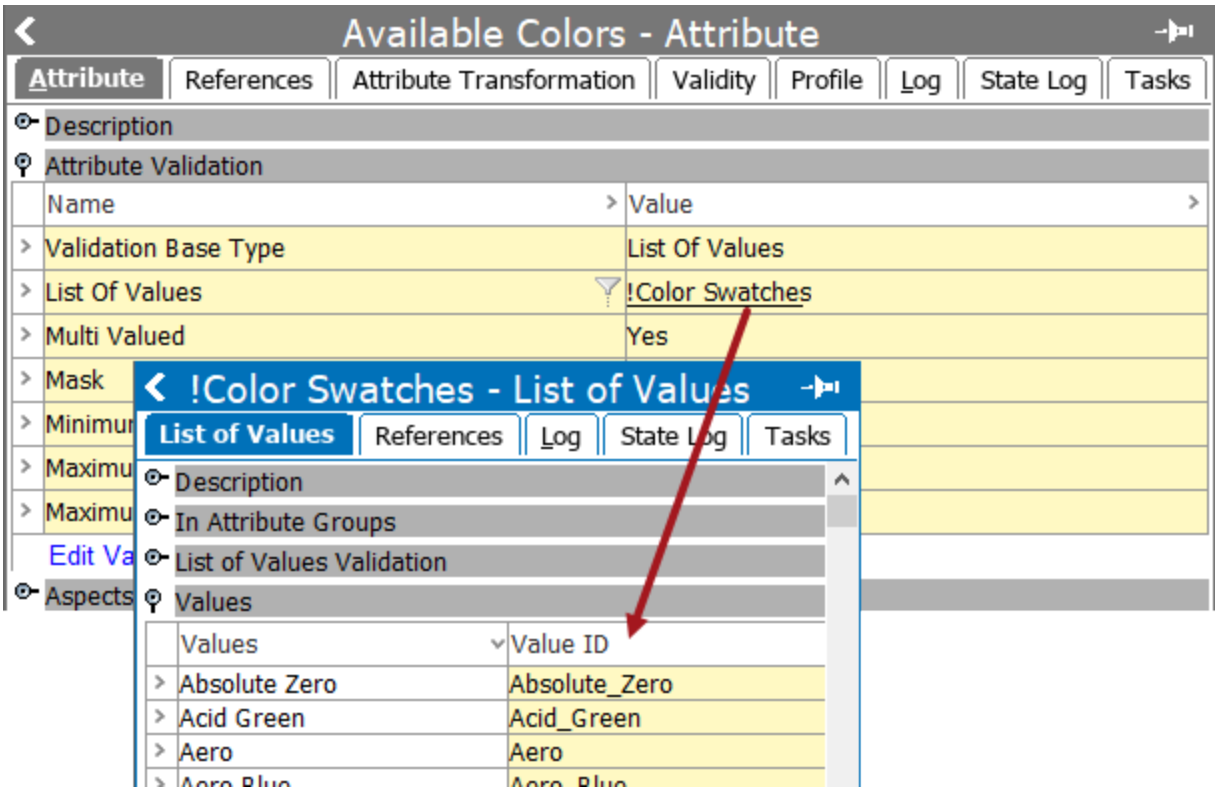
Note: If the LOV attribute allows multiple values, the LOV Value IDs are exported with the multi-value separator defined in the Transformations dialog. For more information, see the **Multi-Value Separator - Transform Outbound** topic.

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, see the **Use IDs on values** section of the **Creating an LOV** topic in the **System Setup / Super User Guide** documentation.

For general information on setting a Transformation aspect, see 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.



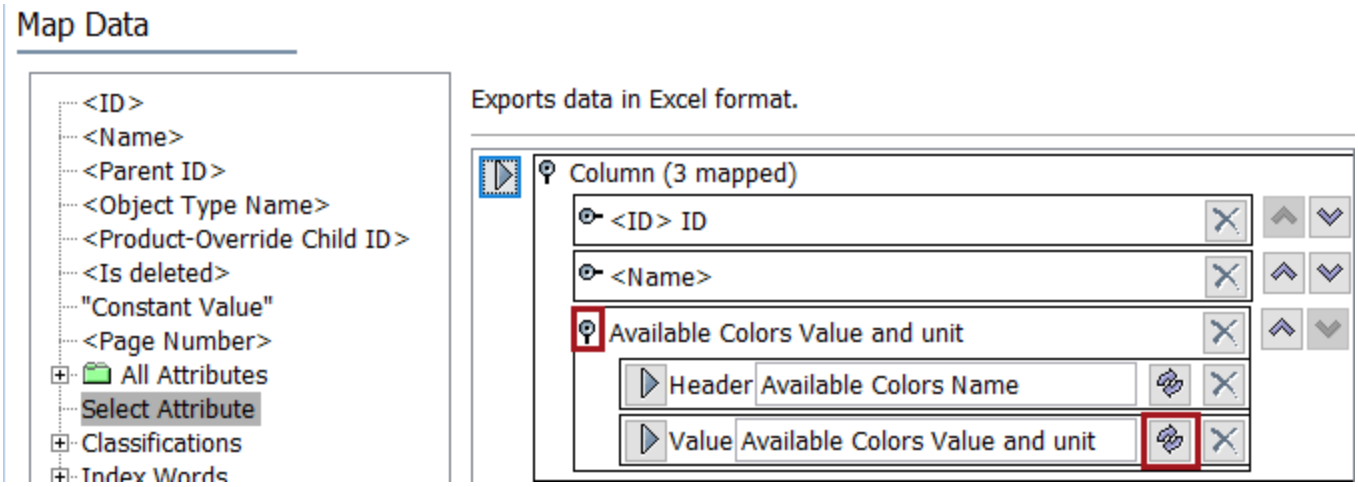
Additional options for exporting LOV aspect information are included in the following topics:

- LOV ID Aspect
- LOV Value-ID or Value Aspect

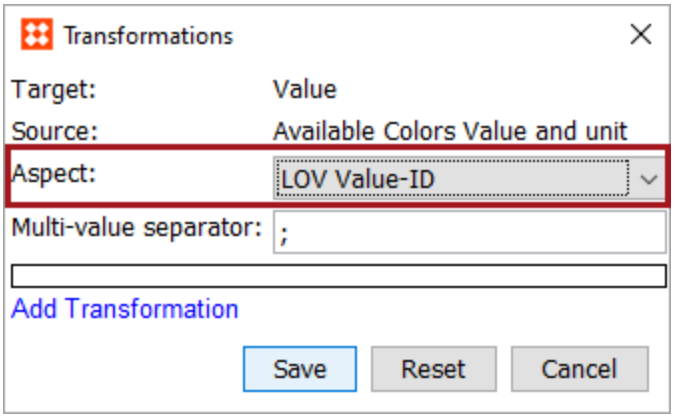
Configuration

1. In the output tool select the data to be exported. For more information, see 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, see **Mapping Attributes via Select Attribute** section of the **Attributes (and Data Containers) - Data Source Outbound** topic.

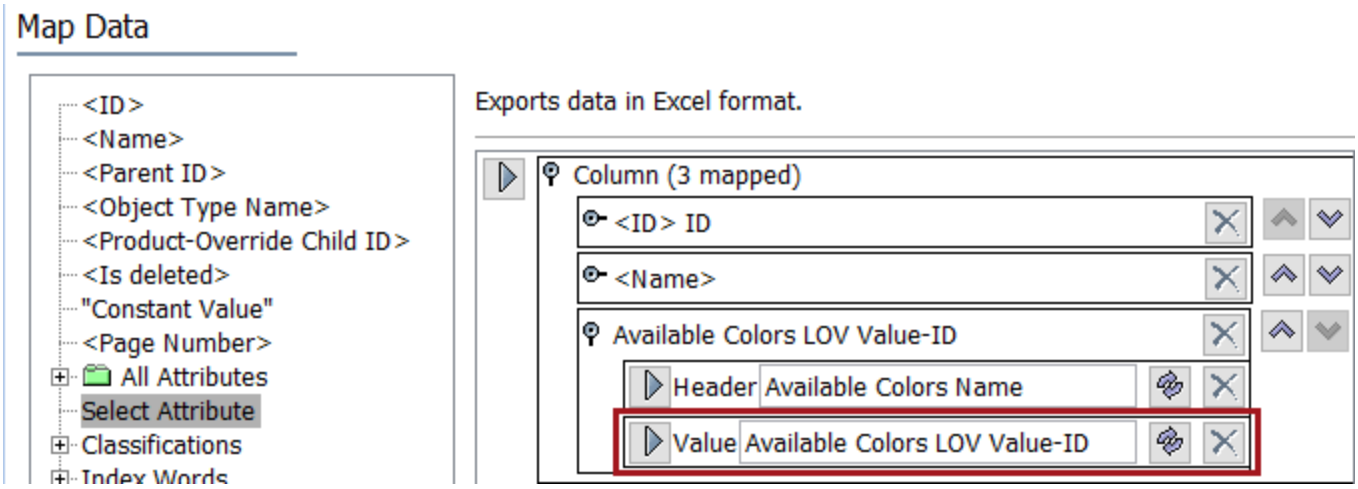
3. In the right panel, click the flipper for the attribute or group that should be extracted using LOV Value-ID. In the example below, the 'AvailableColors' attribute has been mapped.
4. On the Value row, click the **Transformation** button to display the Transformations dialog.



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, see the **Multi-Value Separator - Transform Outbound** topic.

LOV Value-ID or Value Aspect

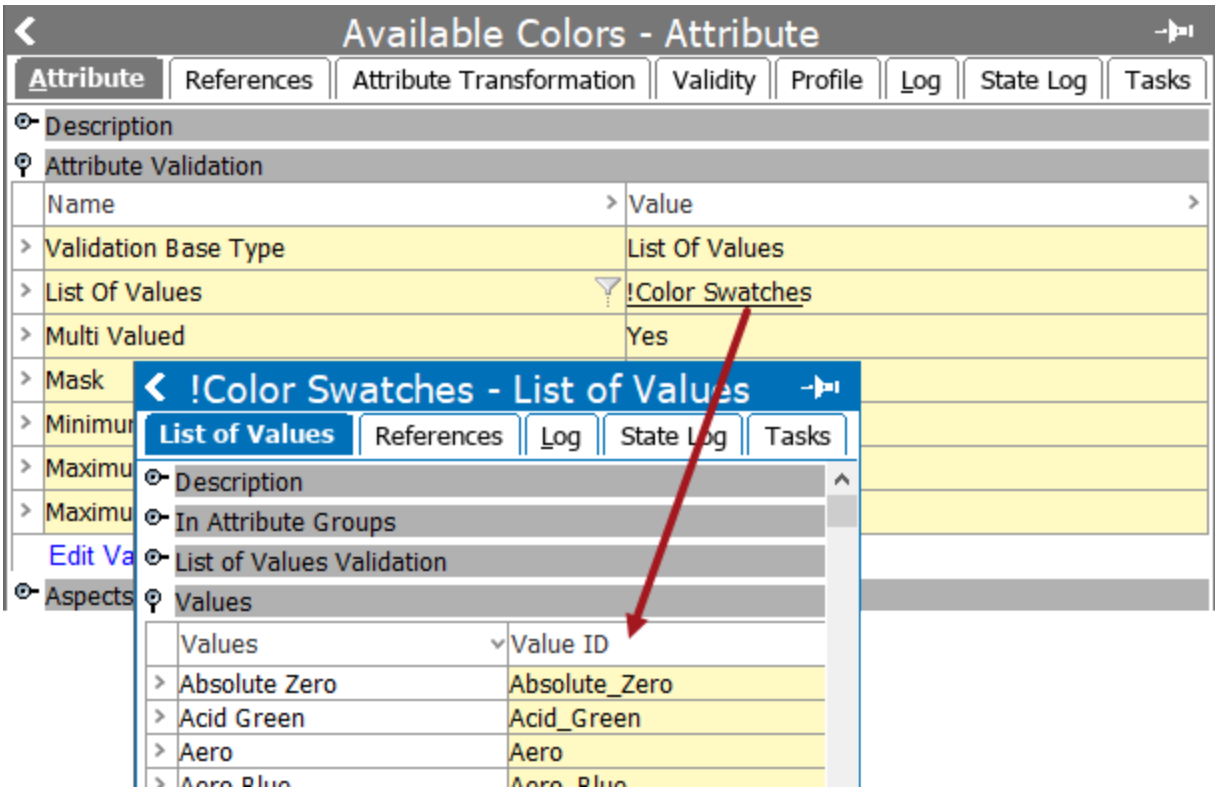
When mapping attributes or attribute groups, this aspect extracts the LOV Value-ID when present, otherwise it extracts the values in the attributes. For details on LOV Value IDs, see the **Use IDs on values** section of the **Creating an LOV** topic in the **System Setup / Super User Guide** documentation.

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.

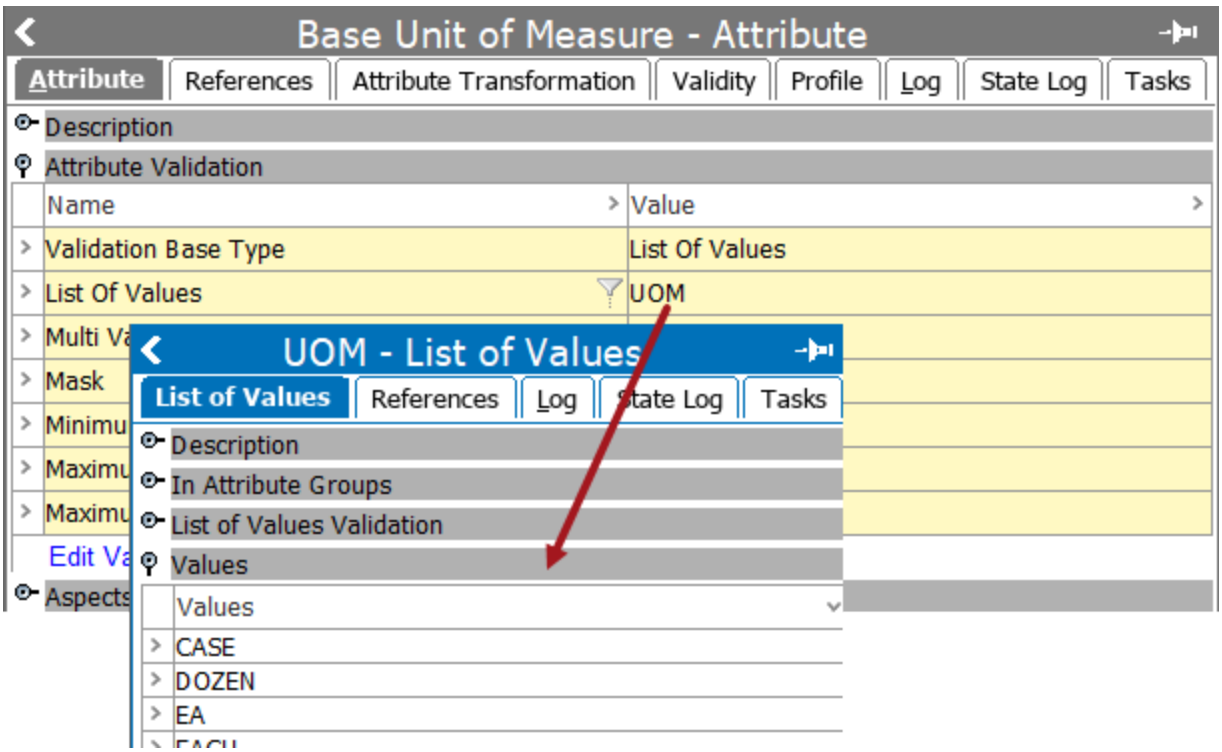
A common use for this aspect is when exporting ETIM via BMEcat2005 format, map an attribute group to the feature field and export the LOV Value IDs. See the **BMEcat 2005 Format Example** section below.

For general information on setting a Transformation aspect, see 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:



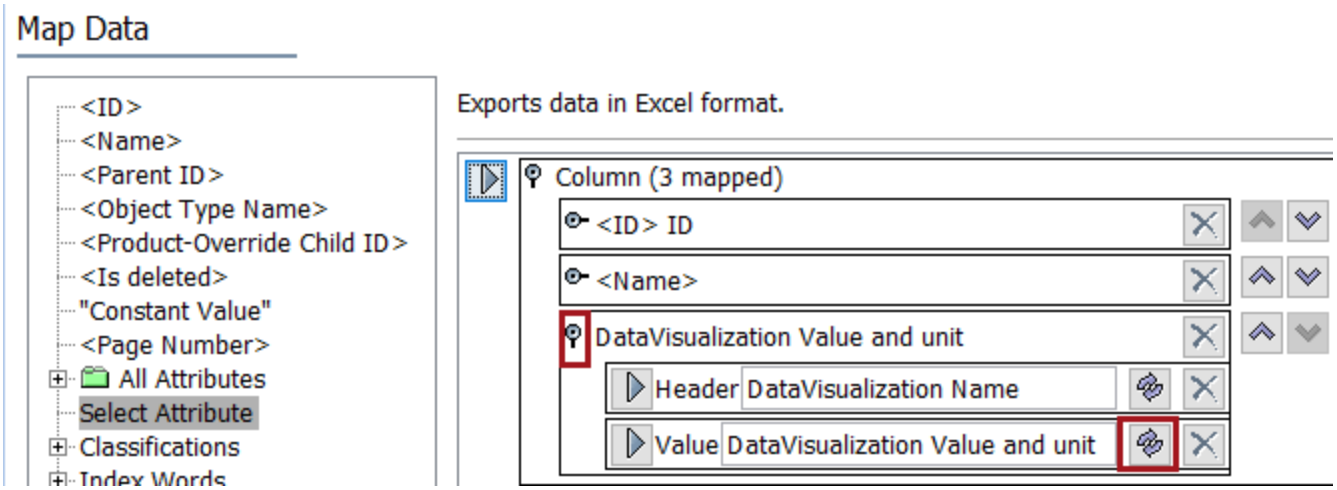
Using the single 'LOV Value-ID or Value' aspect for the attribute group results in exporting the Value IDs for the 'Available Colors' attribute and the Values for the 'Base Unit of Measure' attribute.

Additional options for exporting LOV aspect information are included in the following topics:

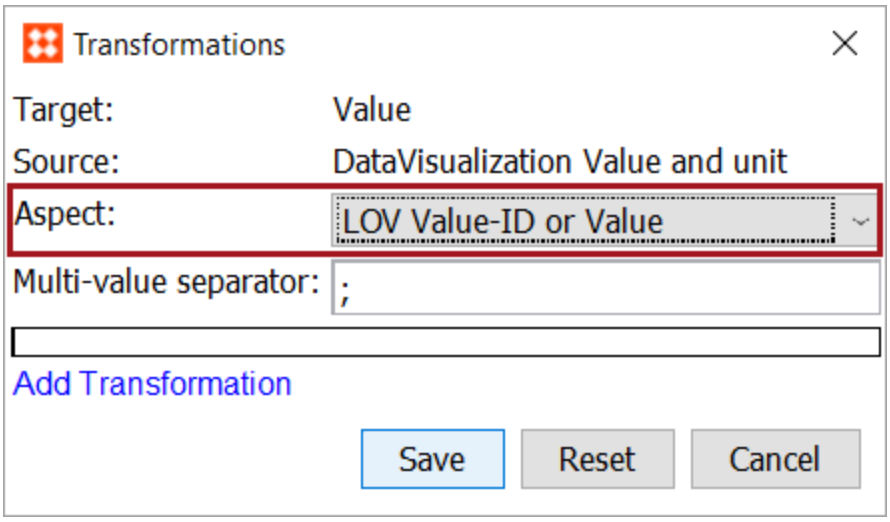
- LOV ID Aspect
- LOV Value-ID Aspect

Configuration

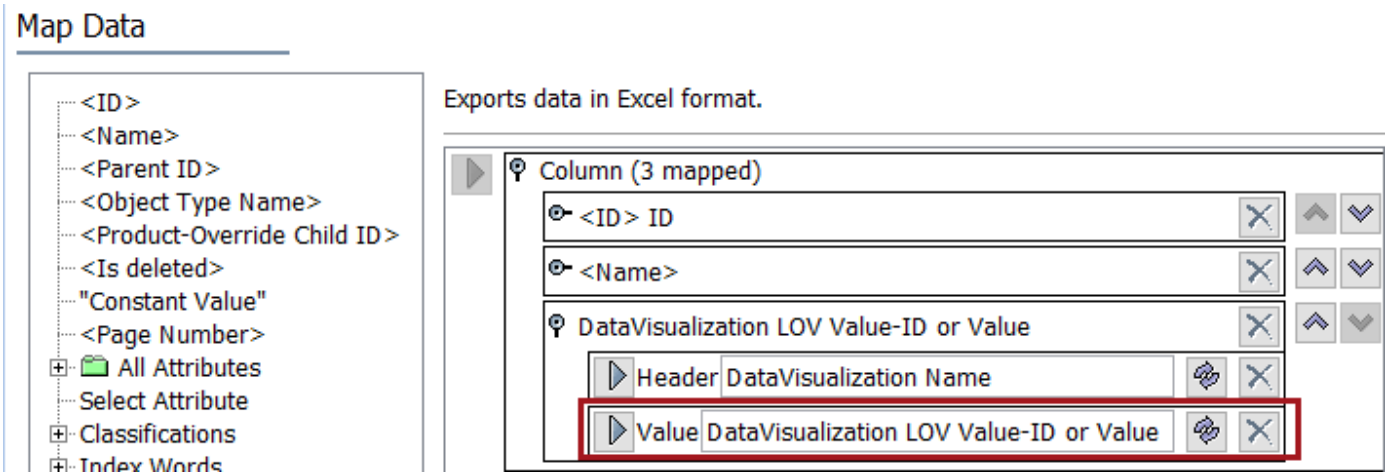
1. In the output tool select the data to be exported. For more information, see the **Data Exchange** topic.
2. On the Map Data step, map the required attribute or attribute group using the **Select Attribute** data source. For more information, see **Mapping Attributes via Select Attribute** section 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.

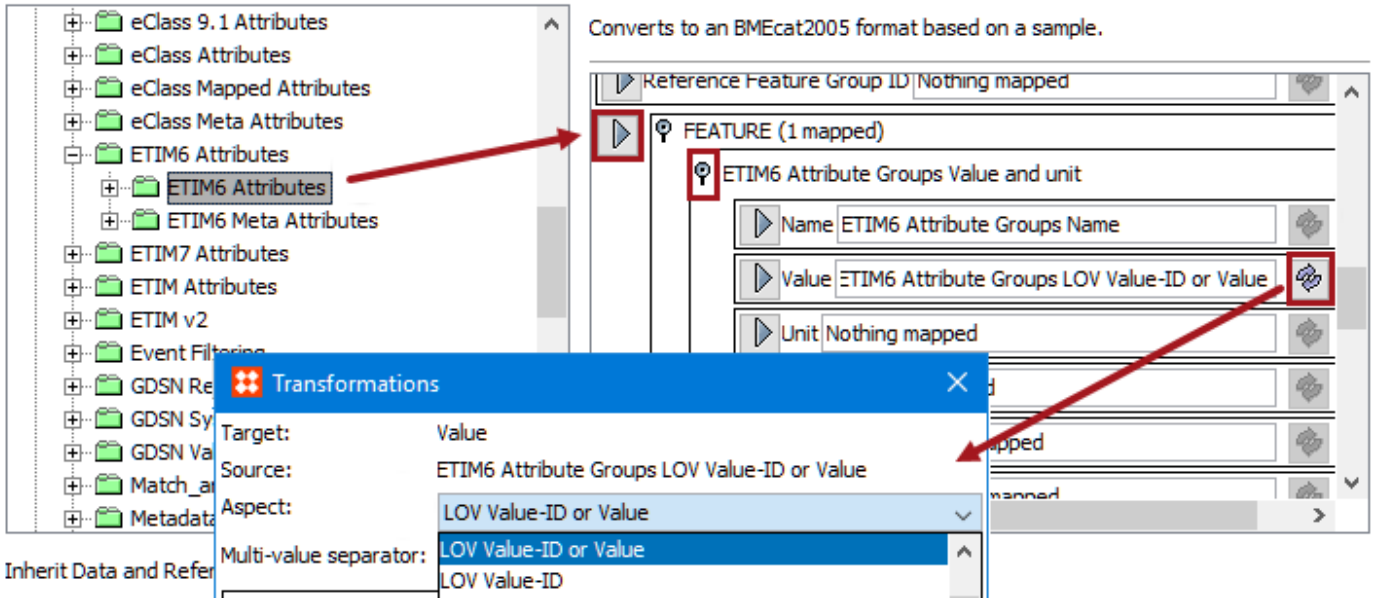
	A	B	C	D
1	<ID>	<Name>	Available Colors	Base Unit of Measure
2	18212	18212 L B	Acid_Green	CASE
3	18213	18213 M O	Aero	EA
4	18216	18216 L O	Absolute_Zero	DOZEN

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, see 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, see 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.'

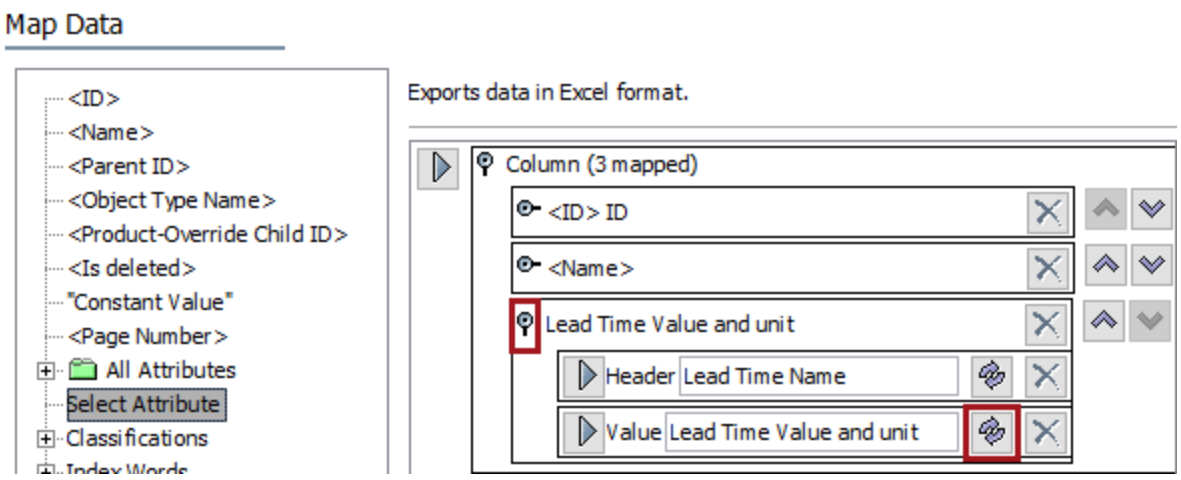
ID	Name	Lead Time
18212	18212 L B	12 mon
18213	18213 M O	3 dy
18216	18216 L O	4 wk

Additional options for exporting unit aspect information are included in the following topics:

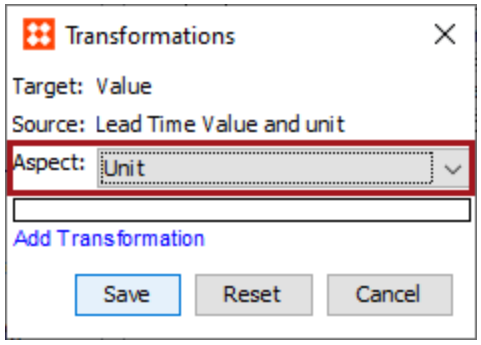
- Unit ID Aspect
- Value and Unit Aspect
- Value and Unit Meta Data Aspect

Configuration

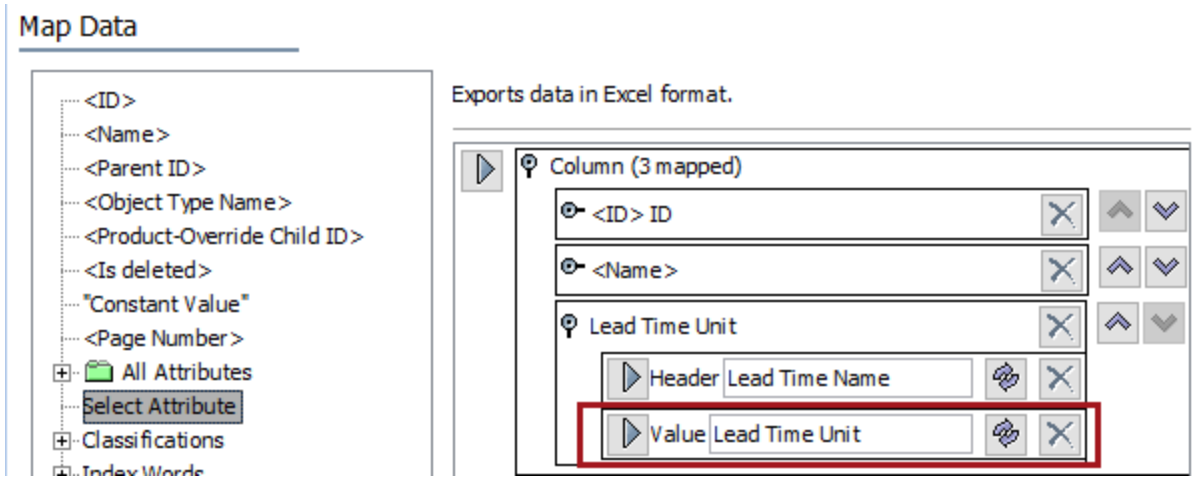
1. In the output tool, select the data to be exported. For more information, see the **Data Exchange** topic.
2. On the Map Data step, map the required attribute using the **Select Attribute** data source. For more information, see **Mapping Attributes via Select Attribute** section of the **Attributes 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

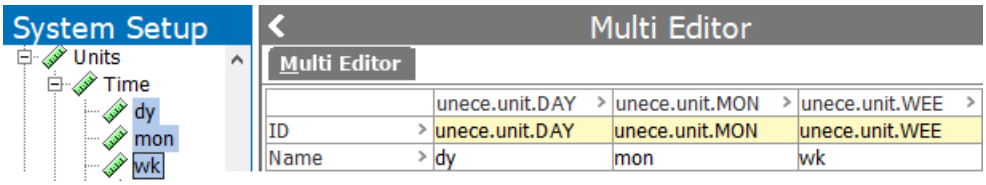
When working with values that may include units, use the Unit ID aspect option to extract the ID of the unit associated with the object attribute value (not the actual value).

For general information on setting a Transformation aspect, see 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.

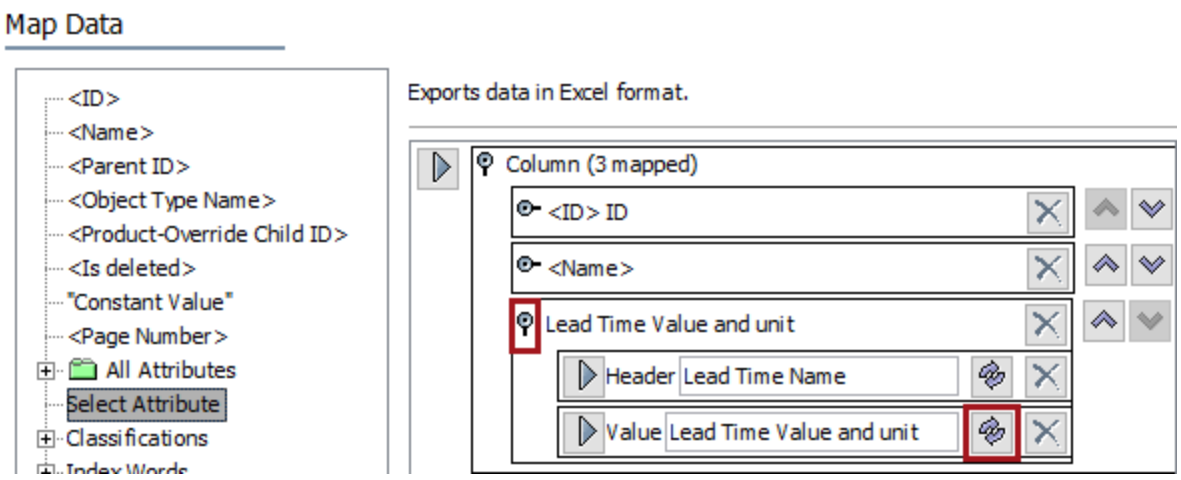


Additional options for exporting unit aspect information are included in the following topics:

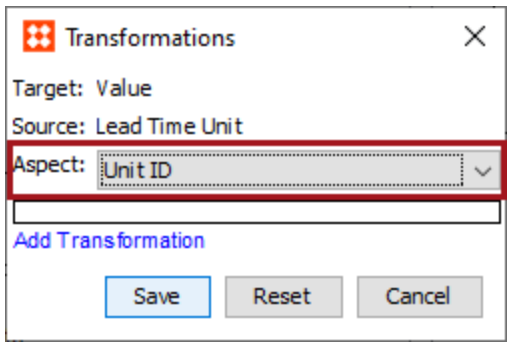
- Unit Aspect
- Value and Unit Aspect
- Value and Unit Meta Data Aspect

Configuration

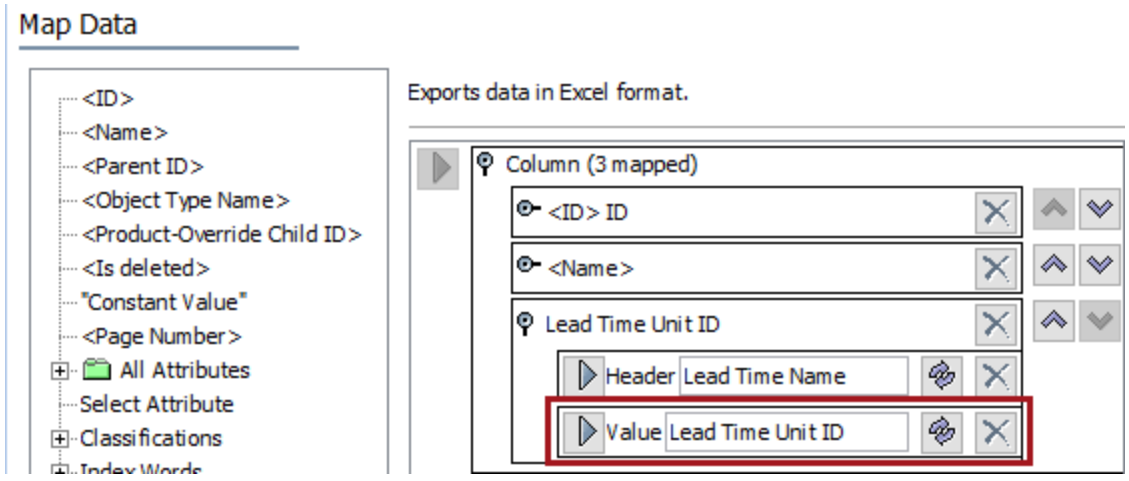
1. In the output tool select the data to be exported. For more information, see the **Data Exchange** topic.
2. On the Map Data step, map the required attribute using the **Select Attribute** data source. For more information, see **Mapping Attributes via Select Attribute** section of the **Attributes 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.



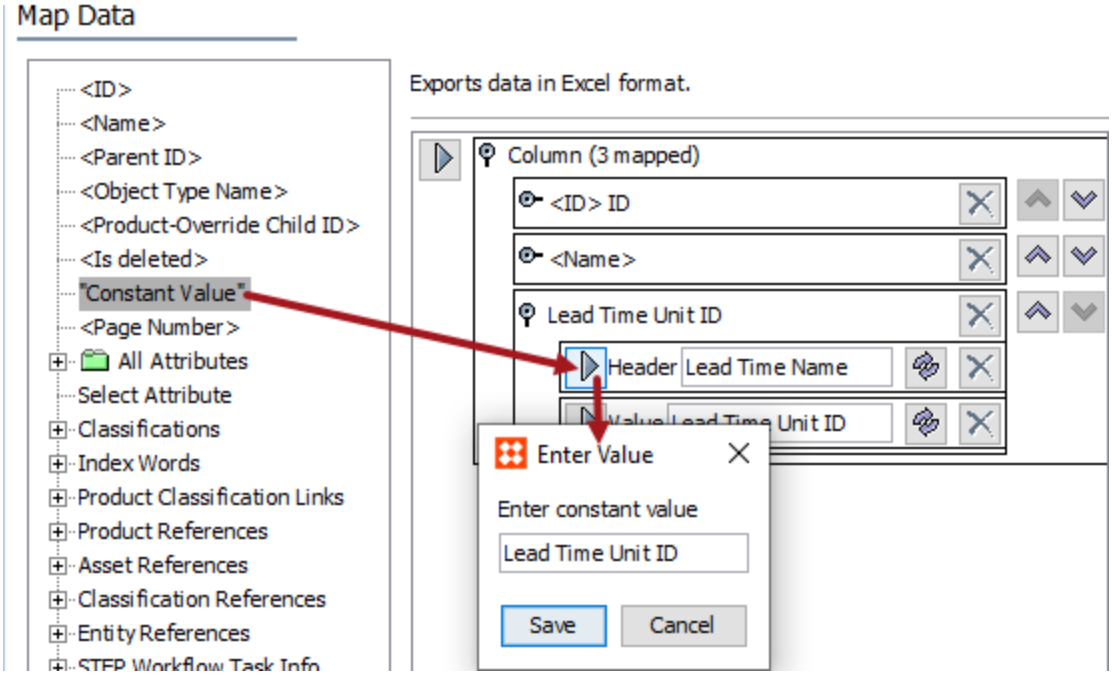
5. In the **Transformations** dialog, from the **Aspect** list, select the **Unit ID** aspect and click the **Save** button. In our example, the Unit ID aspect will export only the ID of the 'month' unit as 'unece.unit.MON', the **Value** aspect will export only the number part of '12,' while the **Value 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, see 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, see 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.

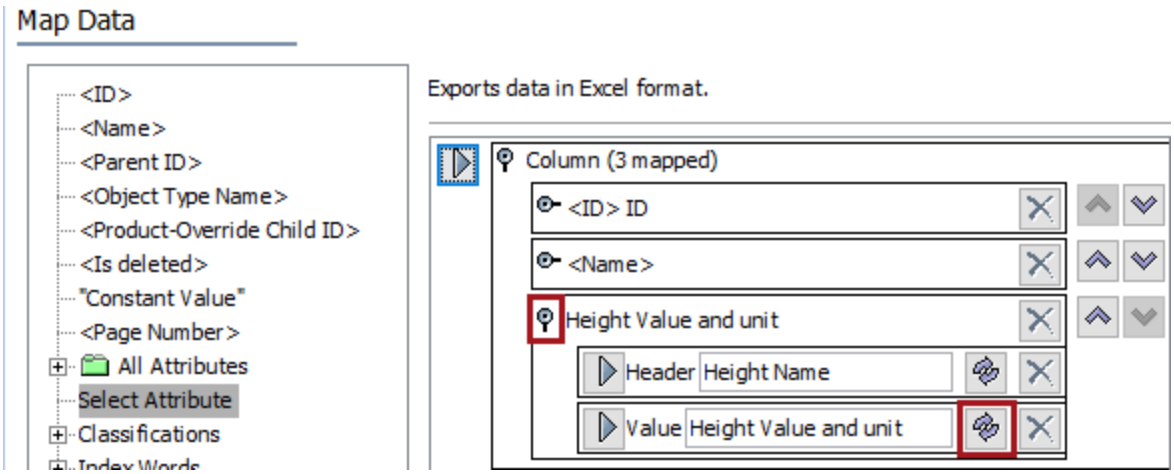
ID	Name	Height
18212	18212 L B	12 ft
18213	18213 M O	3 in
18216	18216 L O	4 cm

Additional options for exporting value aspect information are included in the following topics:

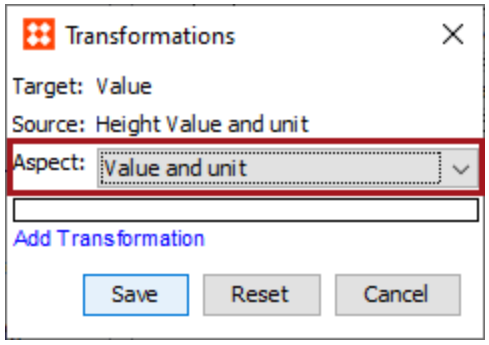
- Value Aspect
- Value and Unit Meta Data Aspect

Configuration

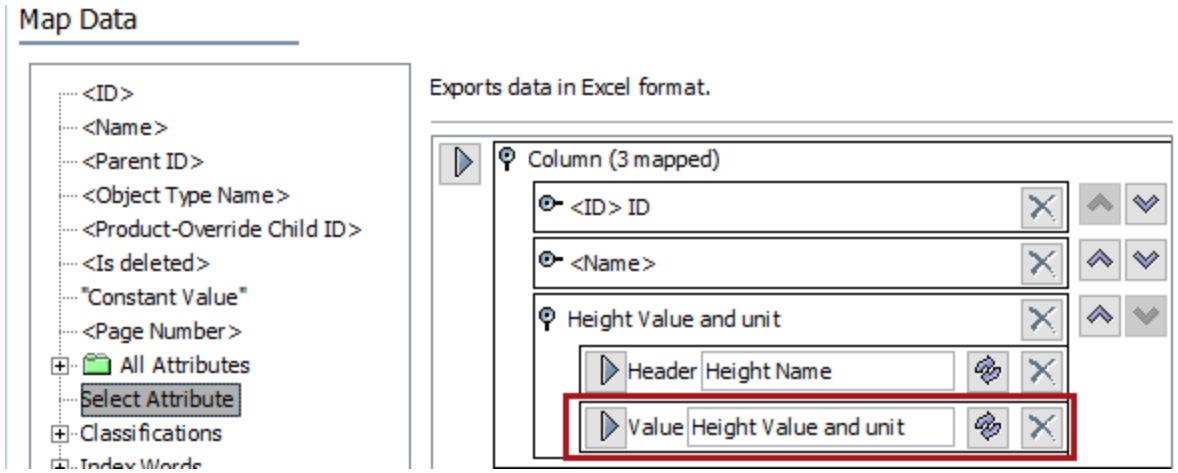
1. In the output tool select the data to be exported. For more information, see the **Data Exchange** topic.
2. On the Map Data step, map the required attribute using the **Select Attribute** data source. For more information, see **Mapping Attributes via Select Attribute** section of the **Attributes 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.

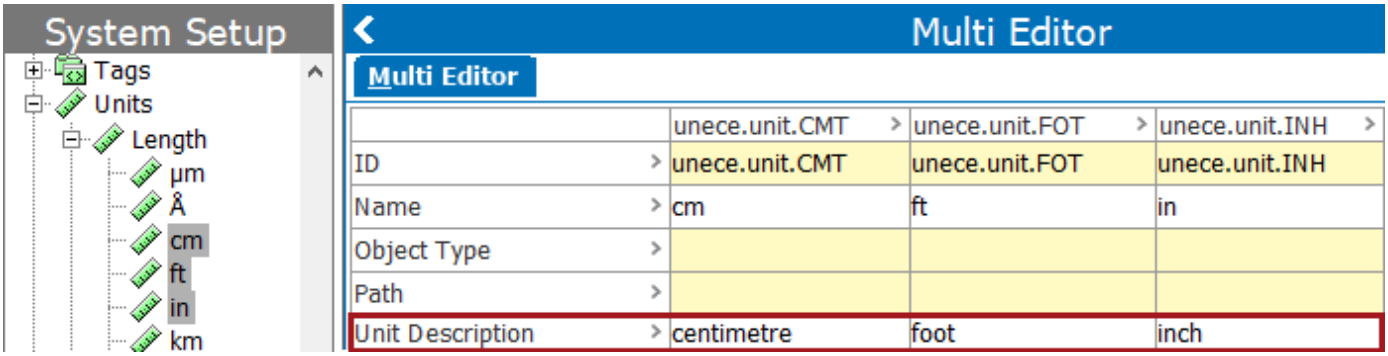
The metadata options available for selection are defined by the metadata attributes that are valid on the objects being exported.

For general information on setting a Transformation aspect, see 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.'

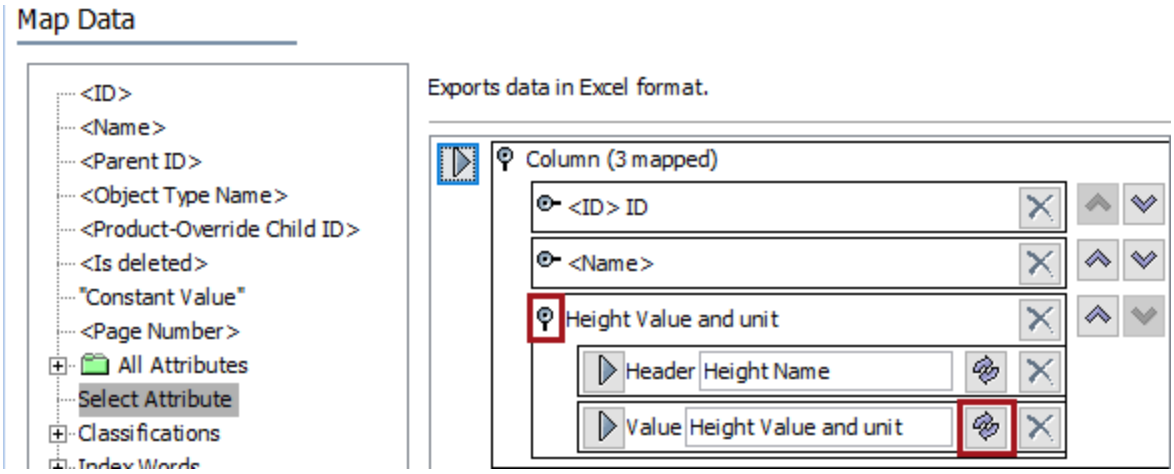
Additional options for exporting value aspect information are included in the following topics:

- Value Aspect
- Value and Unit Aspect

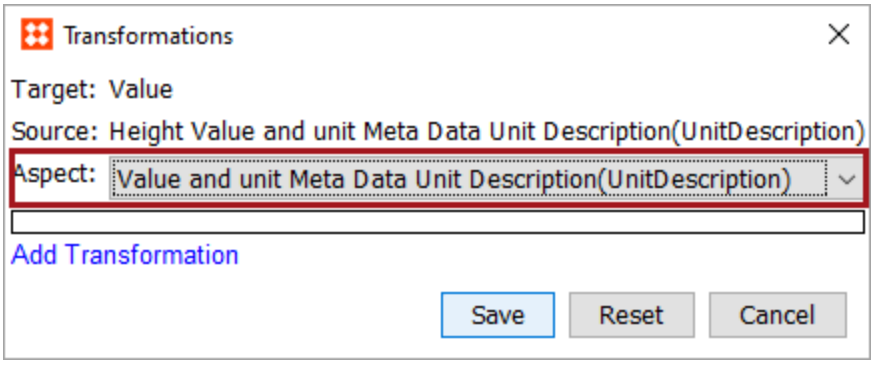
Configuration

1. In the output tool, select the data to be exported. For more information, see the **Data Exchange** topic.

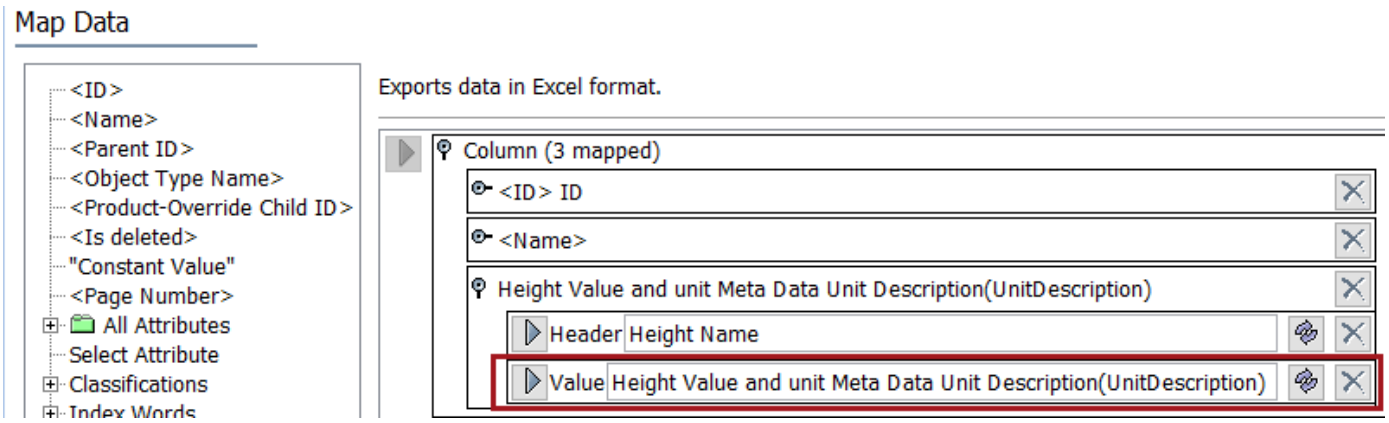
2. On the Map Data step, map the required attribute using the **Select Attribute** data source. For more information, see **Mapping Attributes via Select Attribute** section of the **Attributes and Data Containers - Data Source Outbound** topic.
3. In the right panel, click the flipper for the attribute 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, see 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.

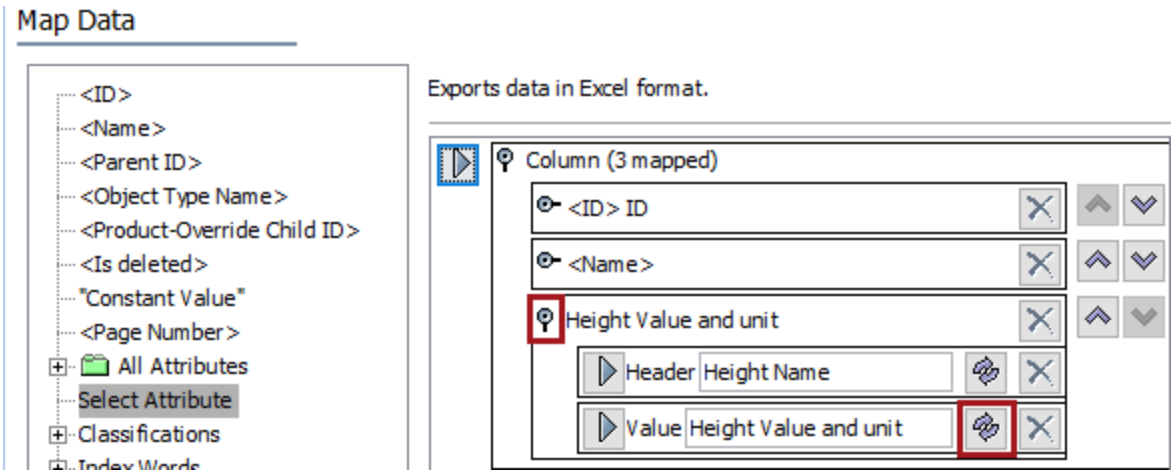
ID	Name	Height
18212	18212 L B	12 ft
18213	18213 M O	3 in
18216	18216 L O	4 cm

Additional options for exporting value aspect information are included in the following topics:

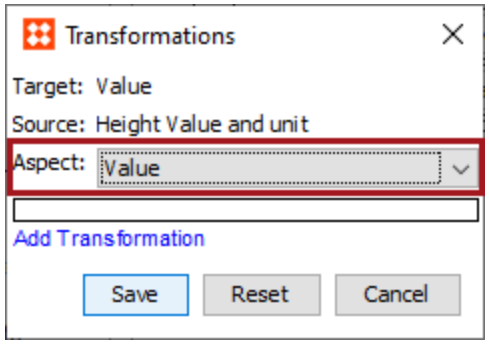
- Value and Unit Aspect
- Value and Unit Meta Data Aspect

Configuration

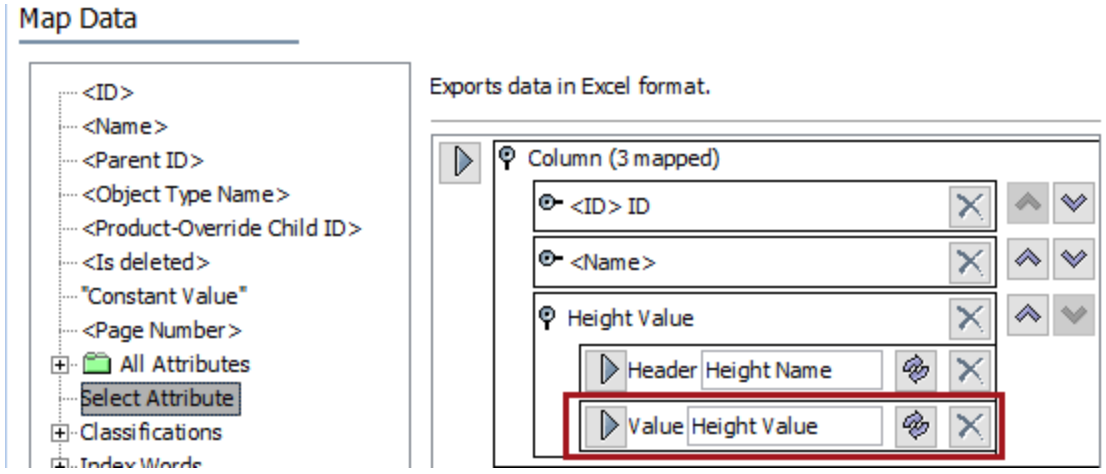
1. In the output tool, select the data to be exported. For more information, see the **Data Exchange** topic.
2. On the Map Data step, map the required attribute using the **Select Attribute** data source. For more information, see **Mapping Attributes via Select Attribute** section of the **Attributes (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 / Super User Guide** documentation.

For additional help importing data, see the **Creating a Data Import** section of the **Data Exchange** documentation. And, for additional help exporting data, see 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, see 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, see 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, see the **Inbound Map Data - Transform** section.

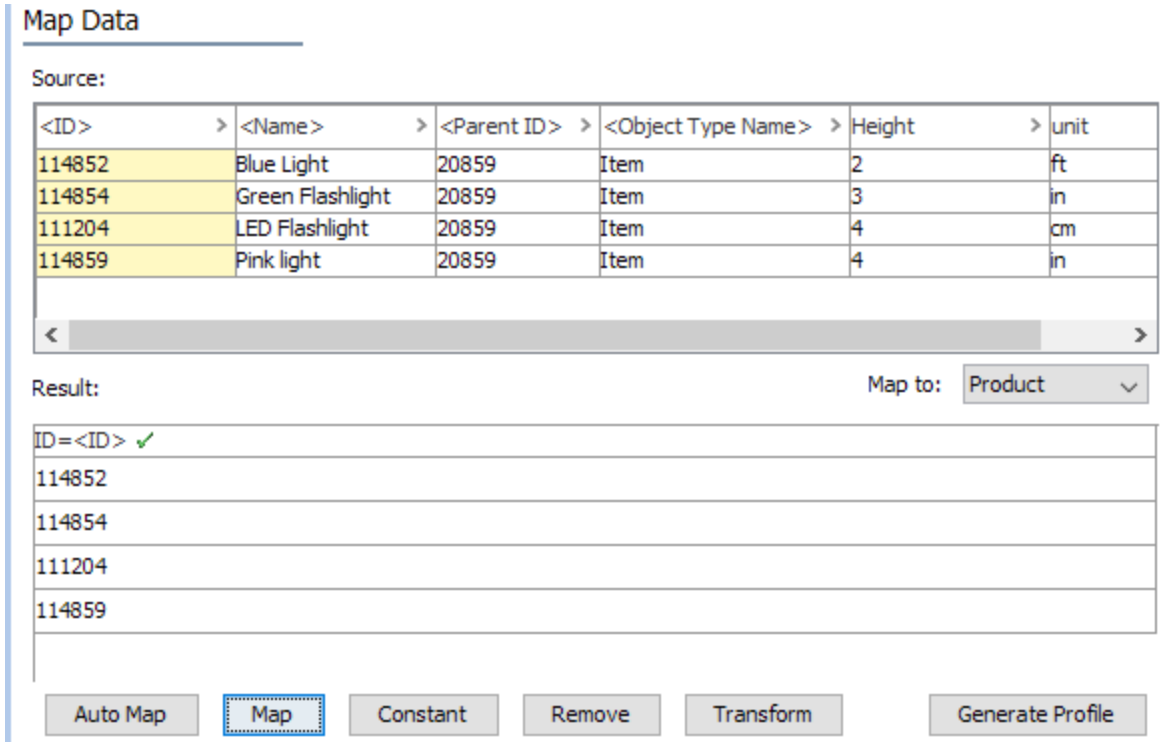
In this example, the source file is a CSV or XLS file with product data where one column has a value and the following column has the corresponding units for that value.

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.

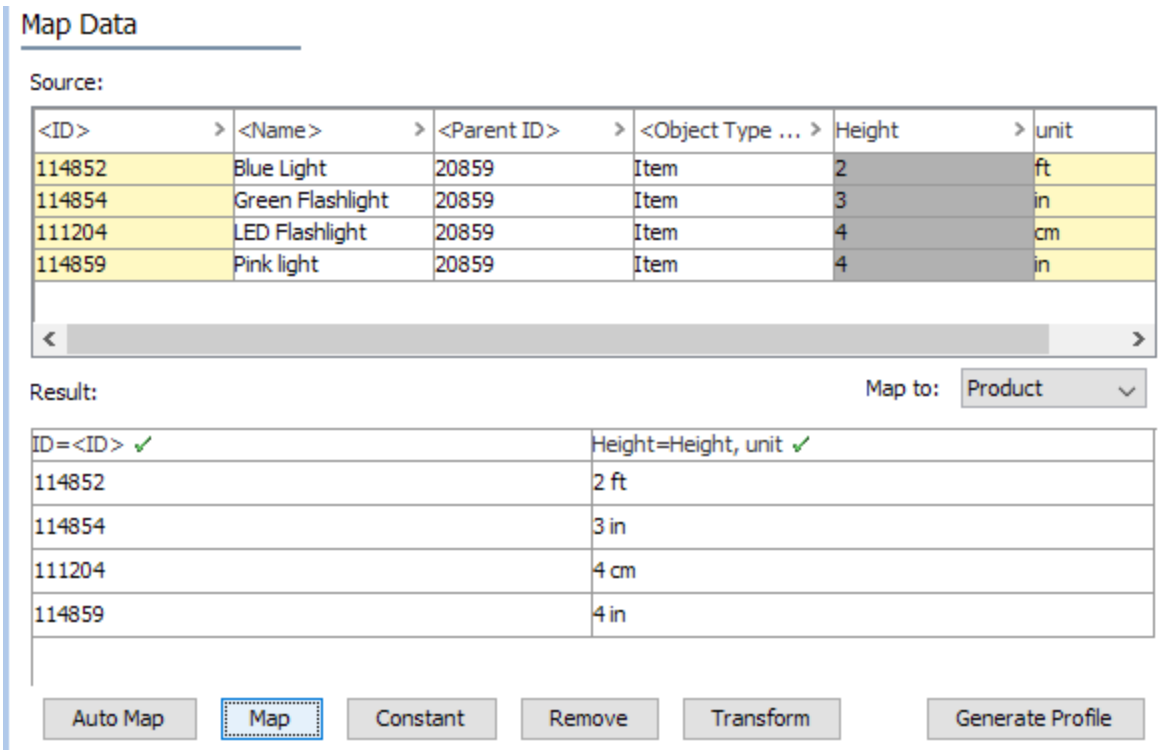
- External update: Concatenate the value and unit columns in the source file, and save the concatenated column values as text. In the import, map the concatenated column to the attribute.
- During import: Concatenate the value and unit columns using the mapping procedure as illustrated below.

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.

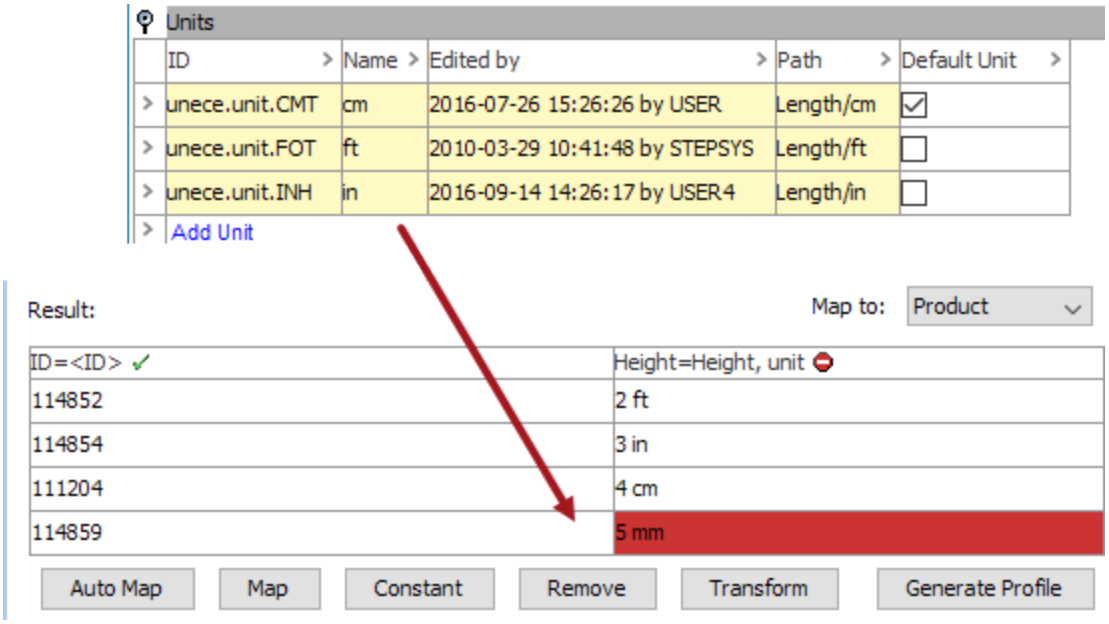


2. Select the column that contains the attribute values, in this case 'Height', and map it to the appropriate attribute.



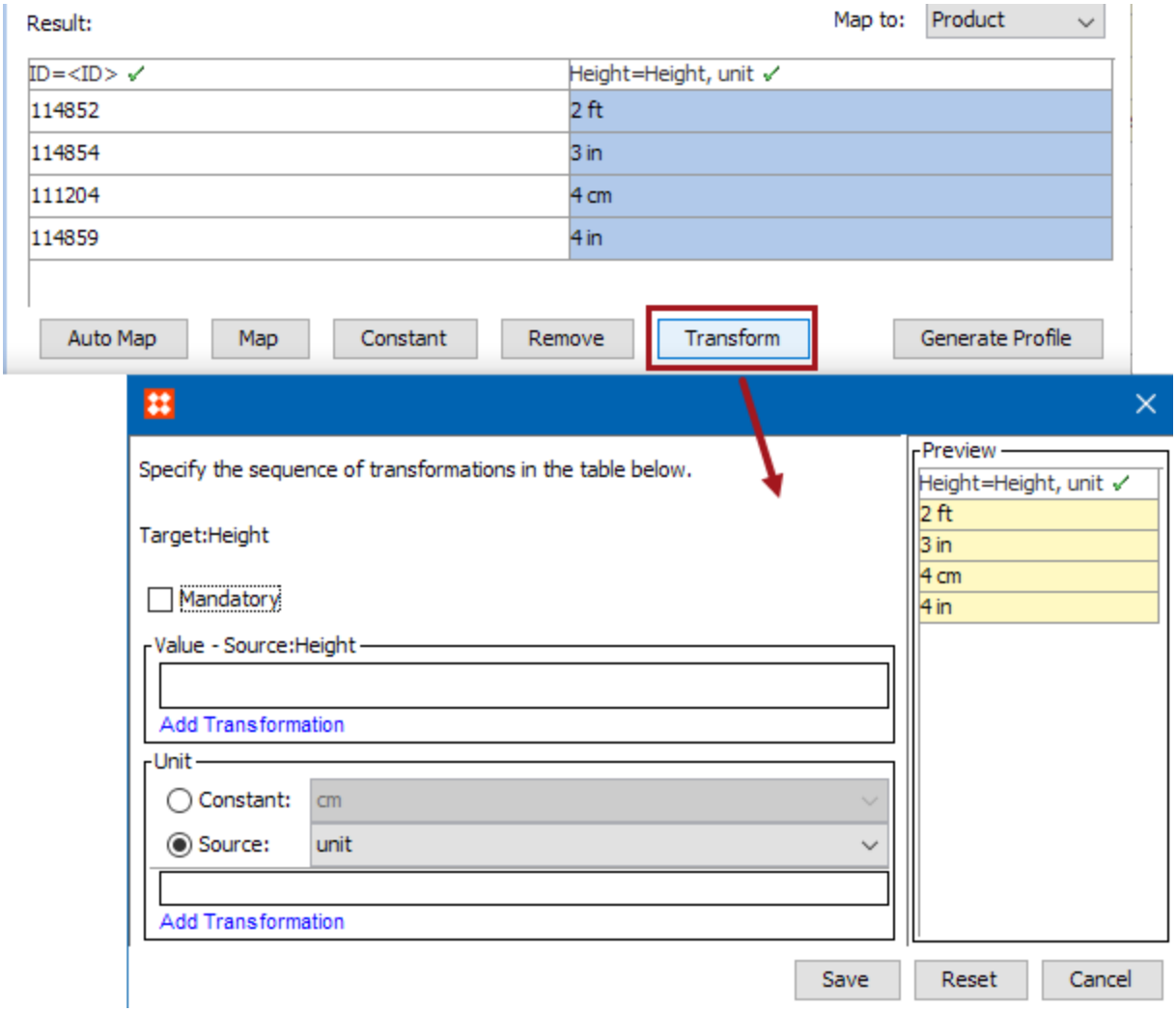
Note that the column has the corresponding units mapped automatically, and that the Result panel shows the value with the corresponding units. It can be observed in the header of the Result panel that **Height = Height, unit**. This mean that it is concatenated with the 'unit' column.

In order for it to concatenate correctly, the attribute must have the corresponding units in STEP. If it does not, an error is displayed.

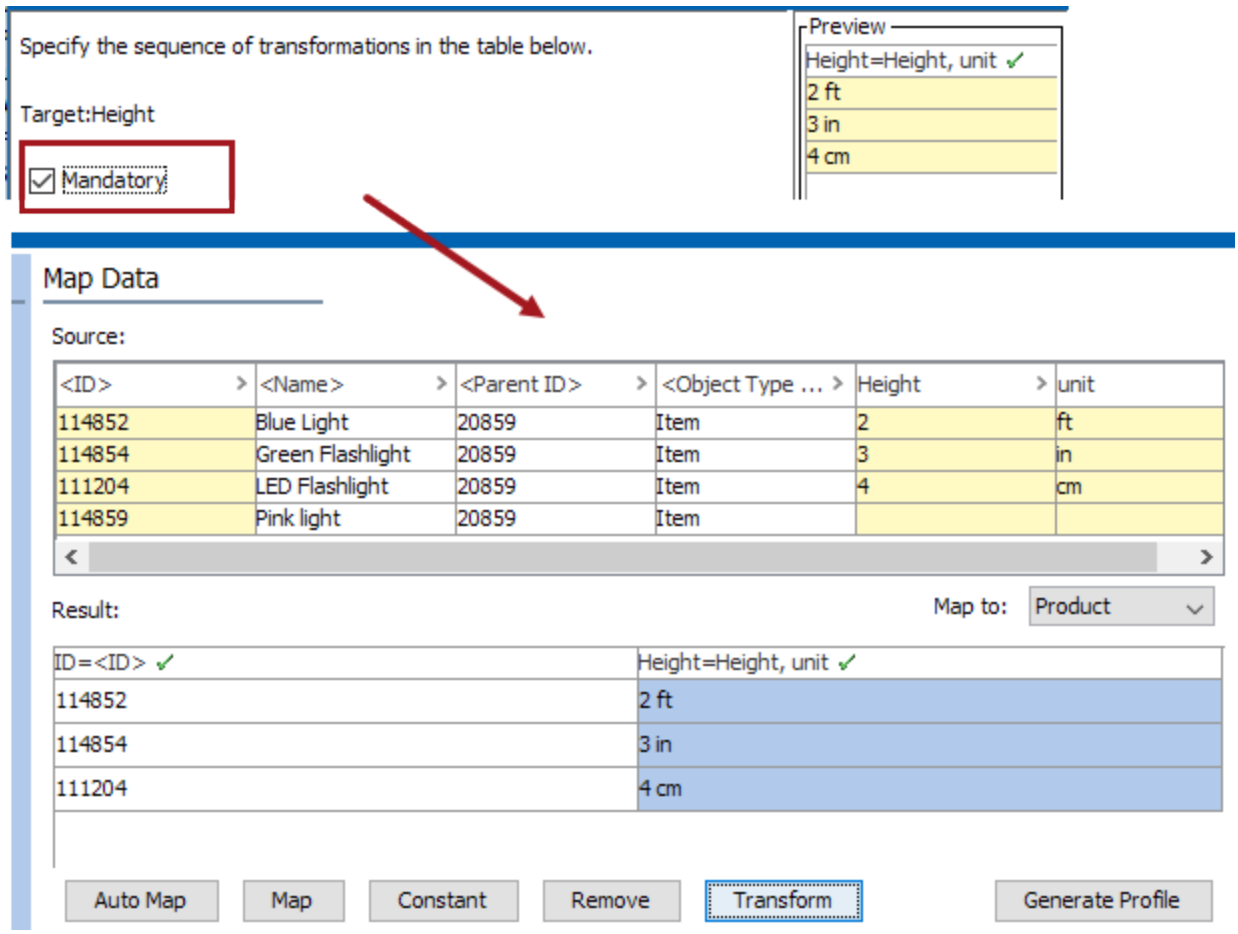


Note: For auto-mapping of the 'units' column to the attribute values, the imported file must include the units in the column following the column that holds the attribute values.

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, see 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, see the **Variable - Map Inbound** topic.

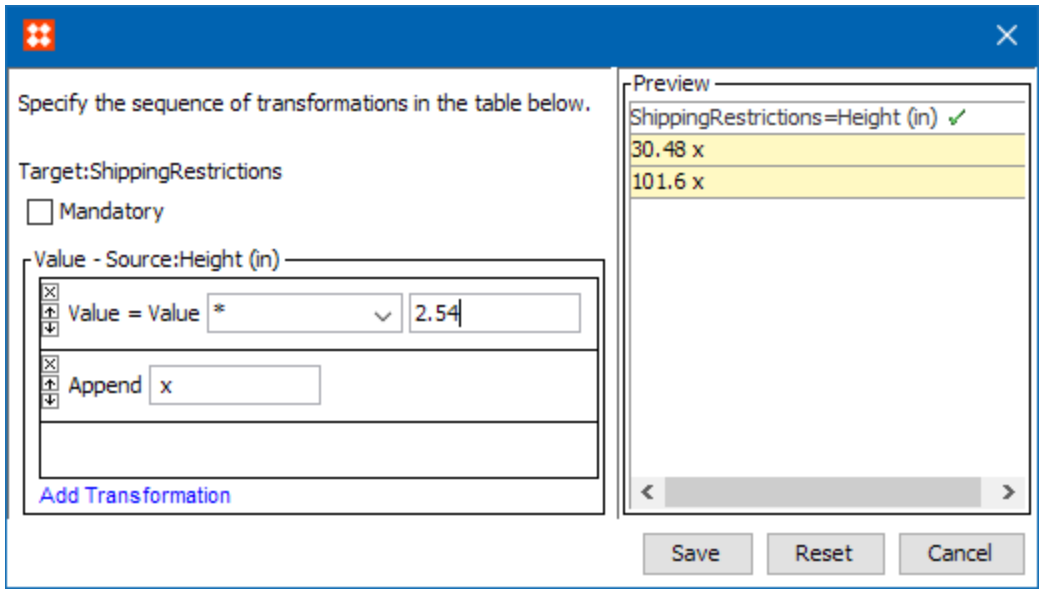
Important: Saving an import configuration saves variables and transformations, as well as the mappings, for use each time the same import is run.

This example includes the following transformations:

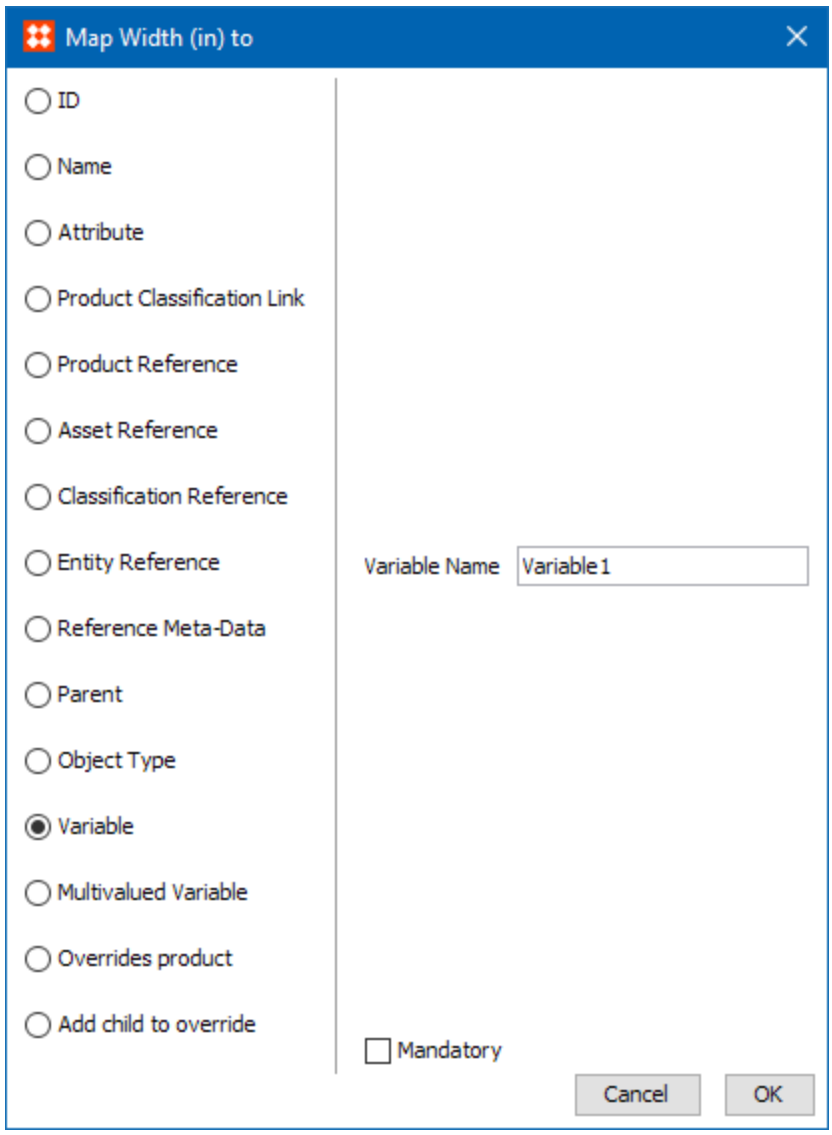
- Math operation (+,-,*,/)
- Append text
- Append from source

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].

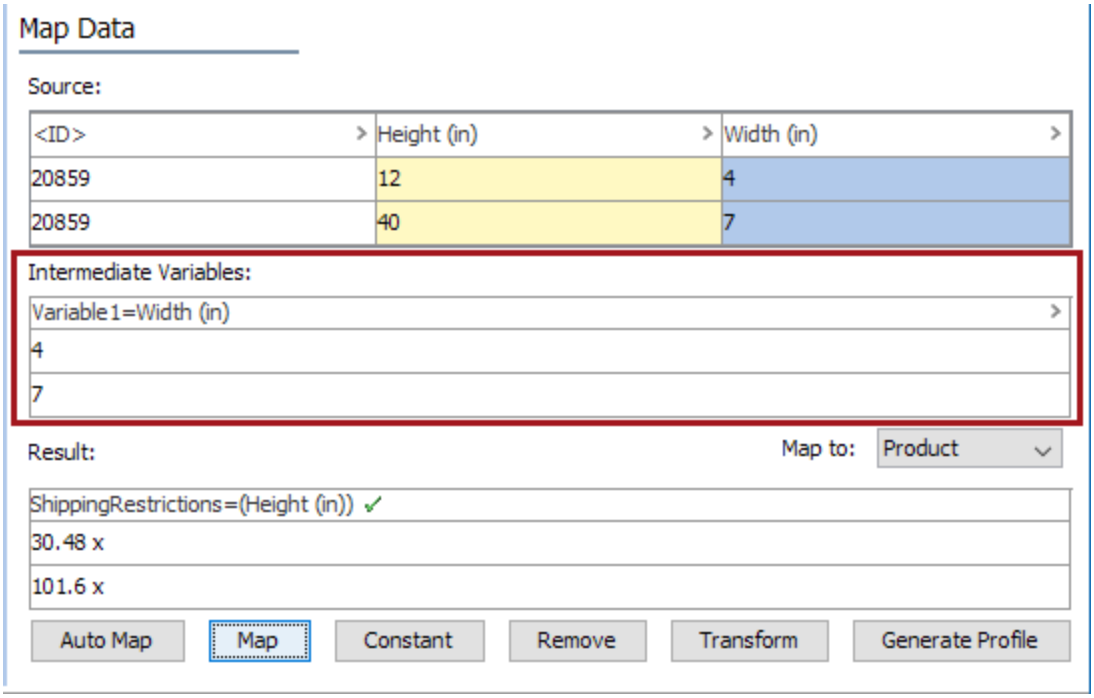
1. Map 'Height (in)' to the 'ShippingRestrictions' attribute and apply needed transformations. In this case the 'Math operation (+,-,*,/)' and the 'Append text' transformation were used.



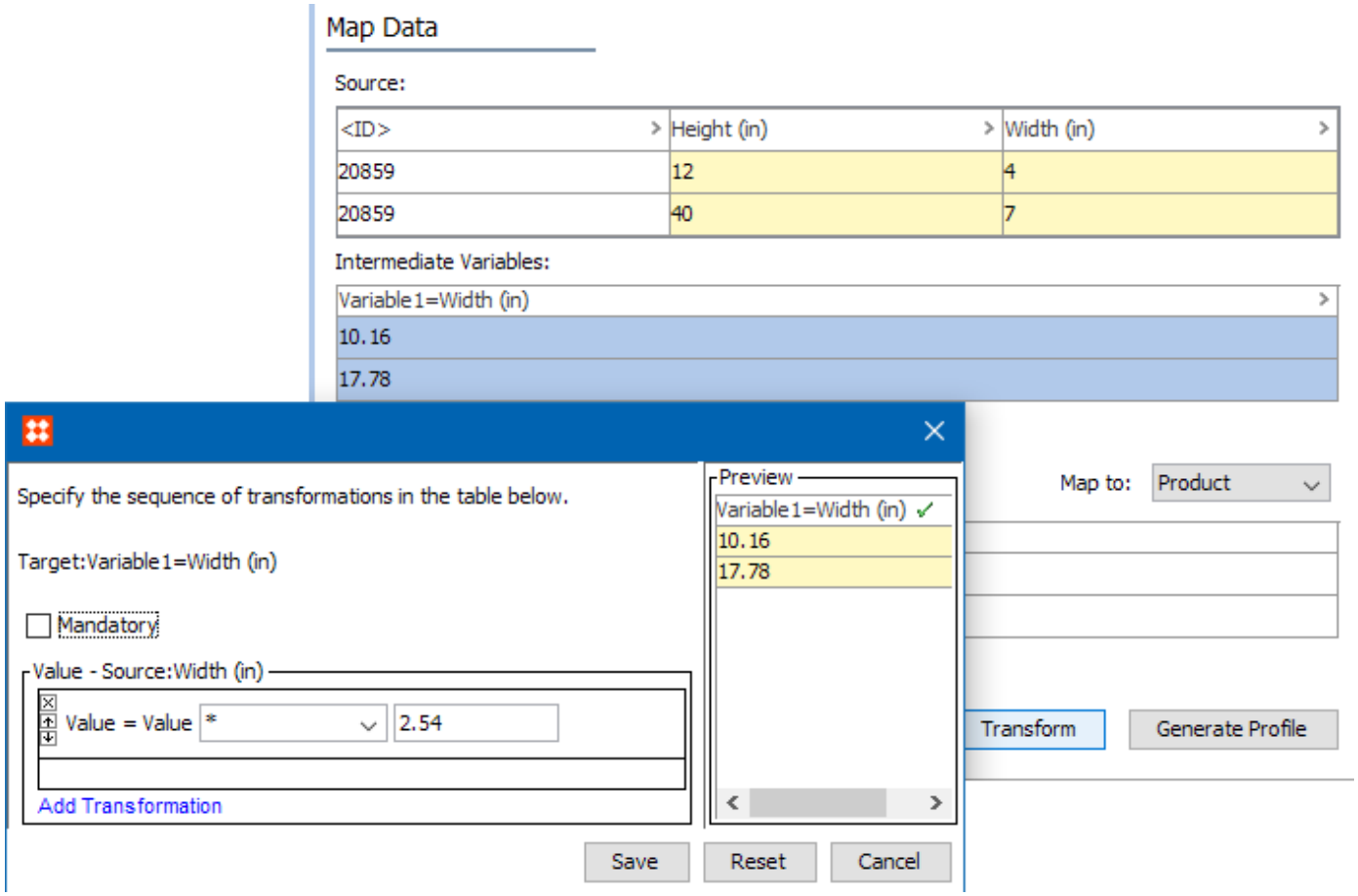
2. Manipulate the data in the 'Width (in)' column before it is appended by saving the transformations made in the first mapping, selecting the 'Width (in)' column in the Source area, and clicking the **Map** button. In the 'Map [column name] to' dialog, select the **Variable** radio button and enter a variable name.



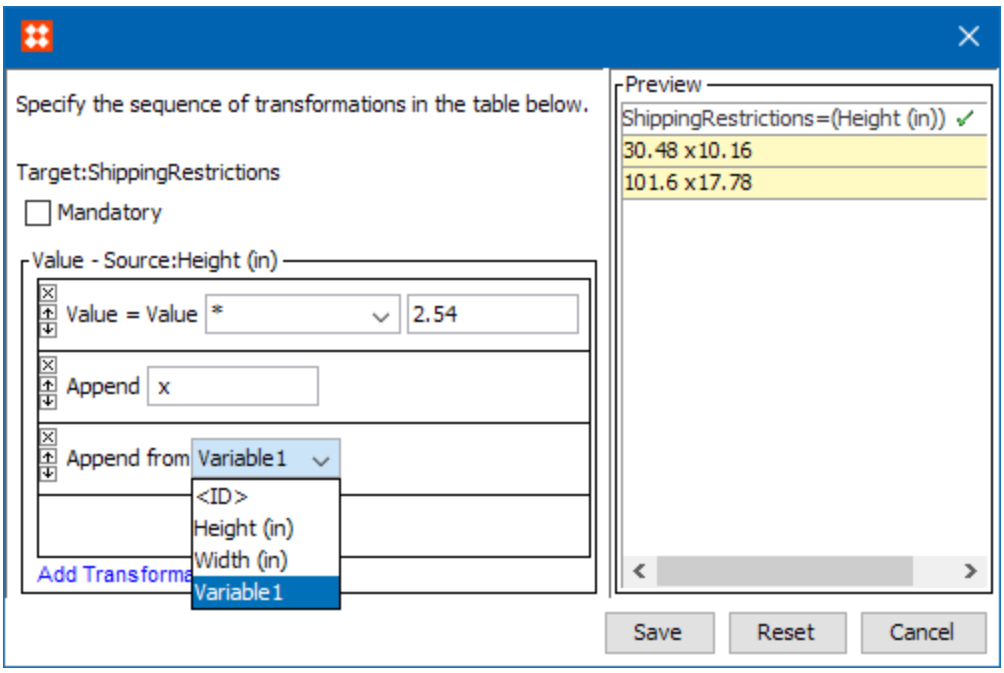
3. Click **OK** to create a new intermediate level, which is displayed between the Source and Result panels.



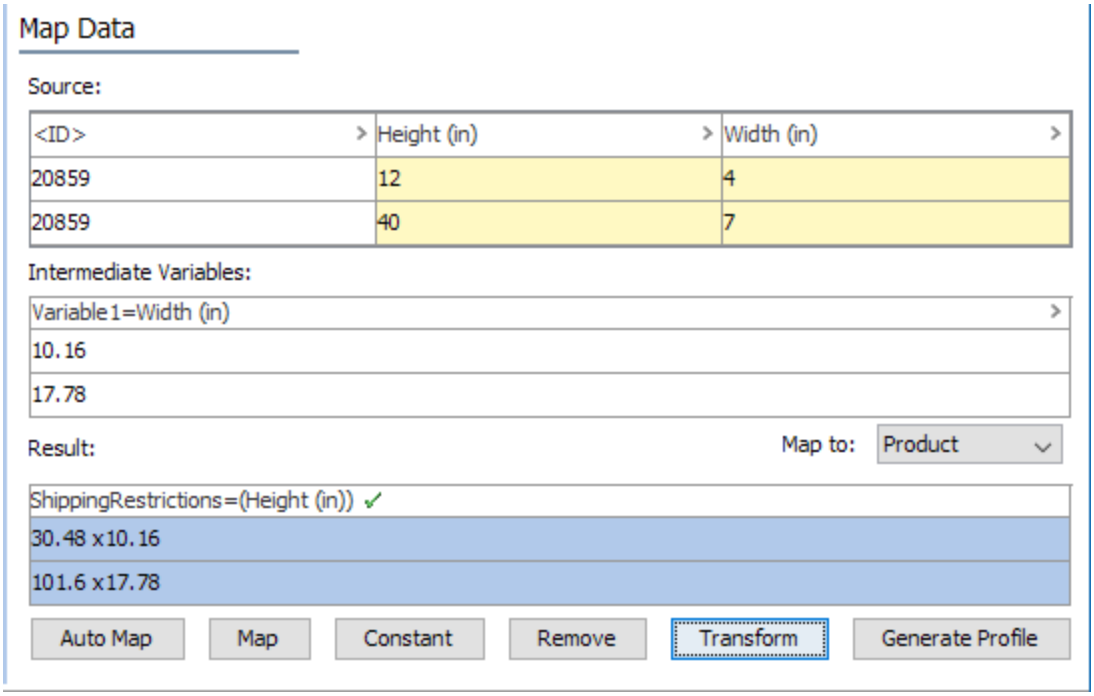
4. You can now transform this data before it is appended to the data already mapped to the 'ShippingRestrictions' attribute. Select the variable column, click **Transform**, and then add the transformation that change inches to centimeters. Click **Save**.



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

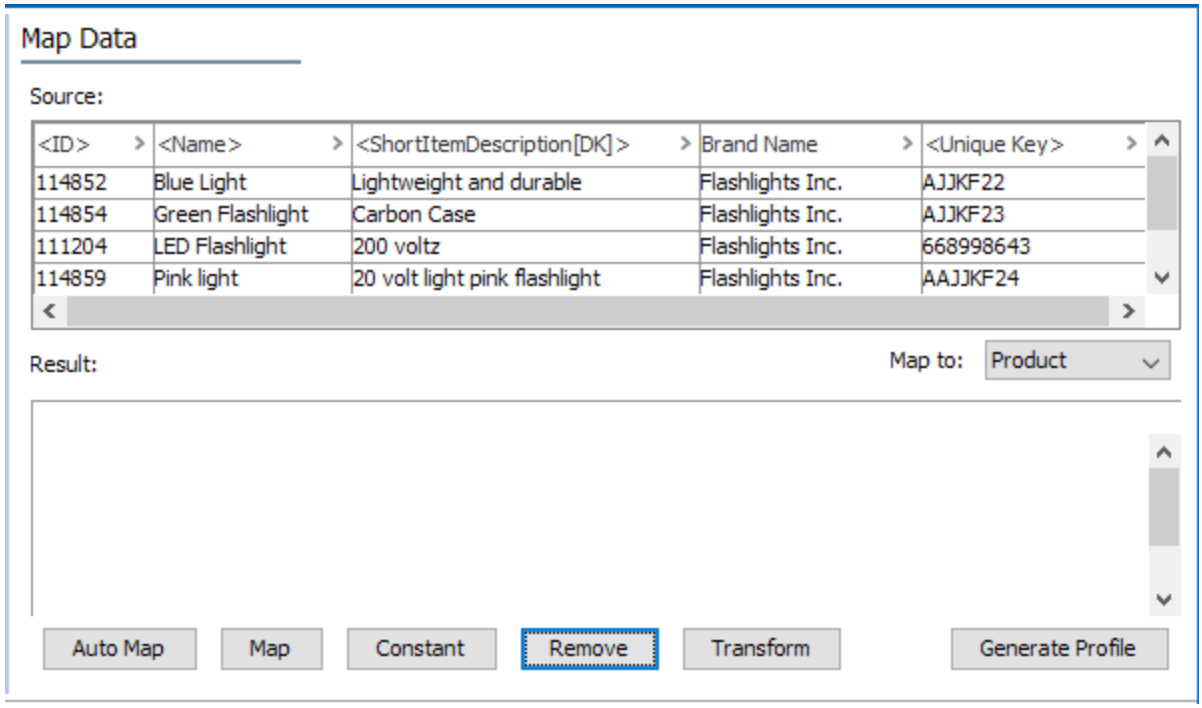
Transformations can improve the quality of the data you want to import. For general information about transformations, see the **Inbound Map Data - Transform** section.

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.

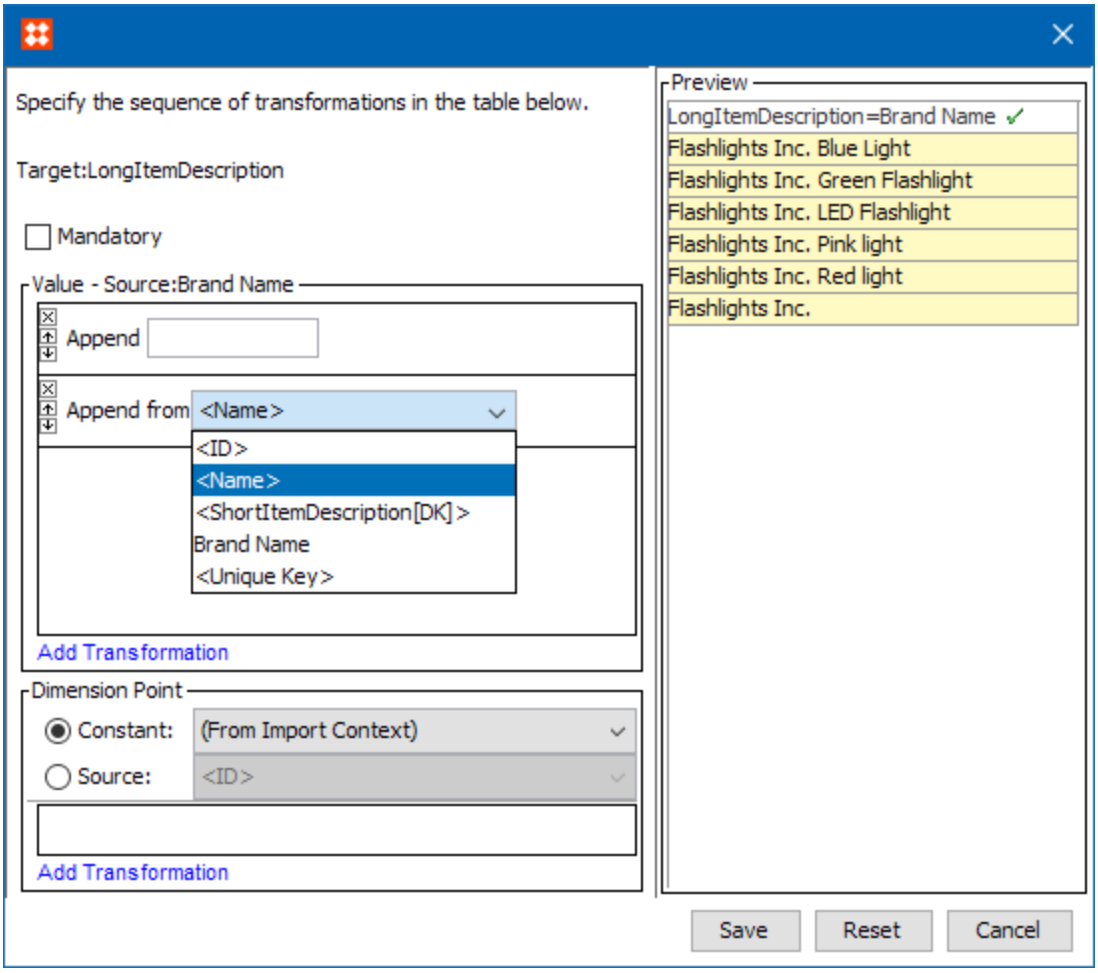
This example includes the following transformations:

- Append text
- Append from source

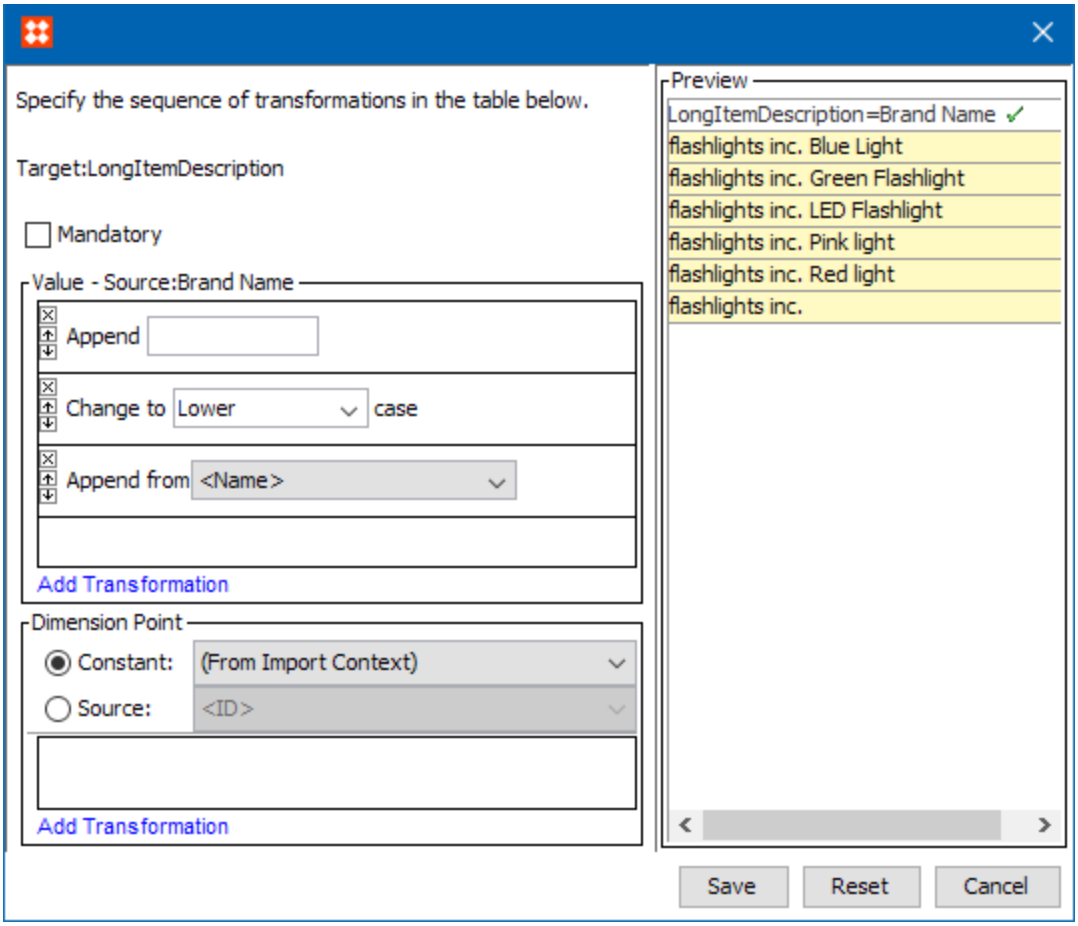
In this example, you have an import file where you want to concatenate the data in the 'Brand Name' column with the data in the '<Name>' column and map it into a attribute named 'LongItemDescription.'



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

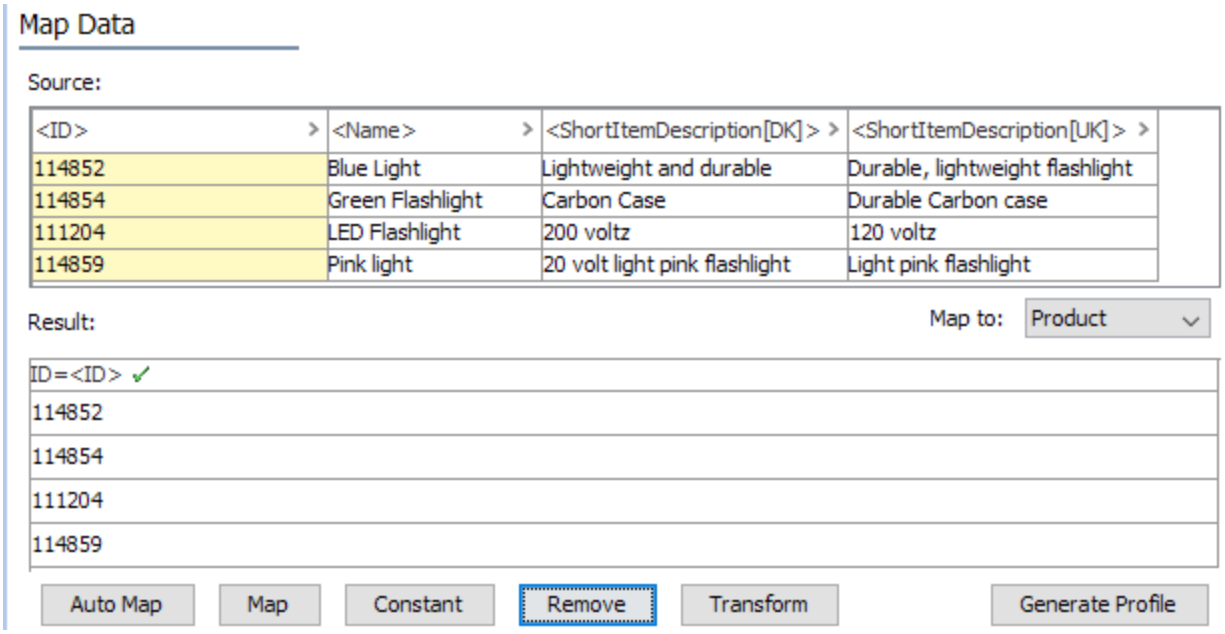
Transformations can improve the quality of the data you want to import. For general information about transformations, see the **Inbound Map Data - Transform** section.

This example includes the following transformations:

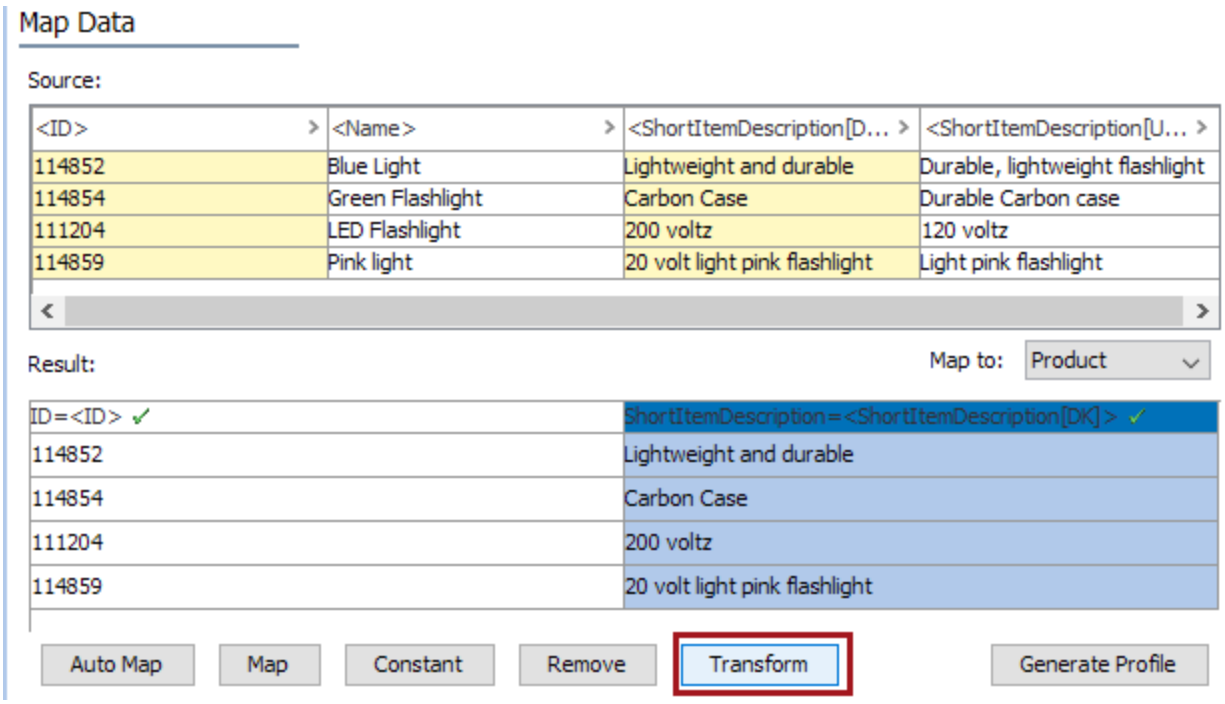
- Dimension Point

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.

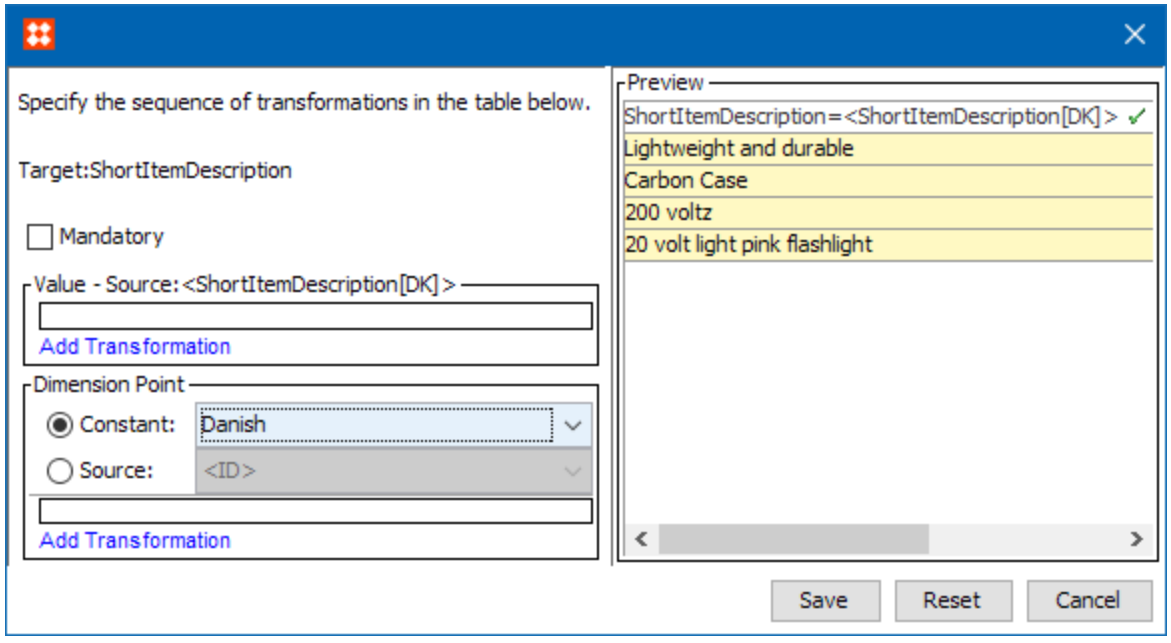
1. Map the object's **ID** in the import tool.



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**.

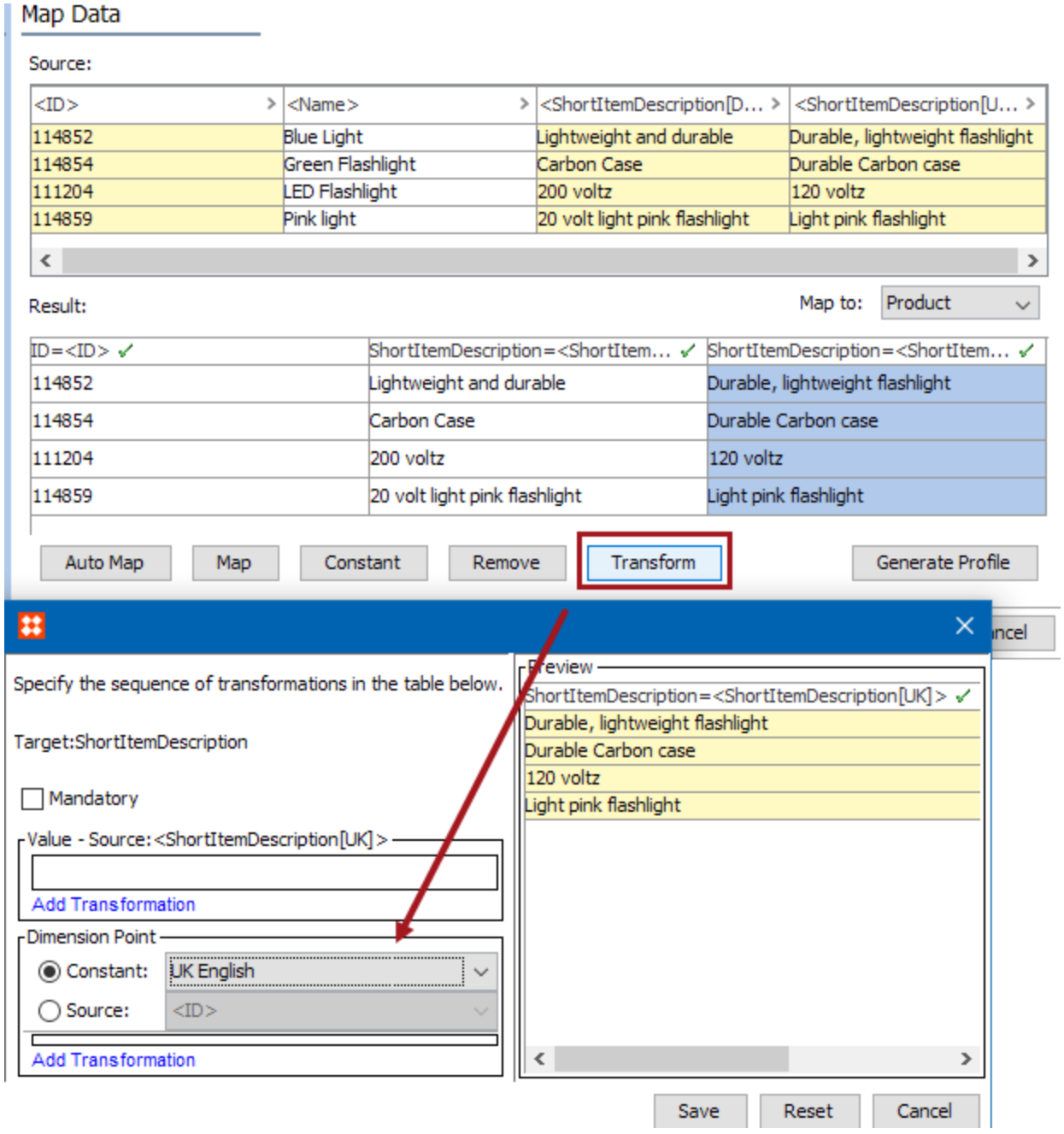


The Dimension Point section is shown because the attribute has a single dimension dependency. The default behavior is to use the language of the import context. The dimension point can either be mapped to:

- **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 / 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

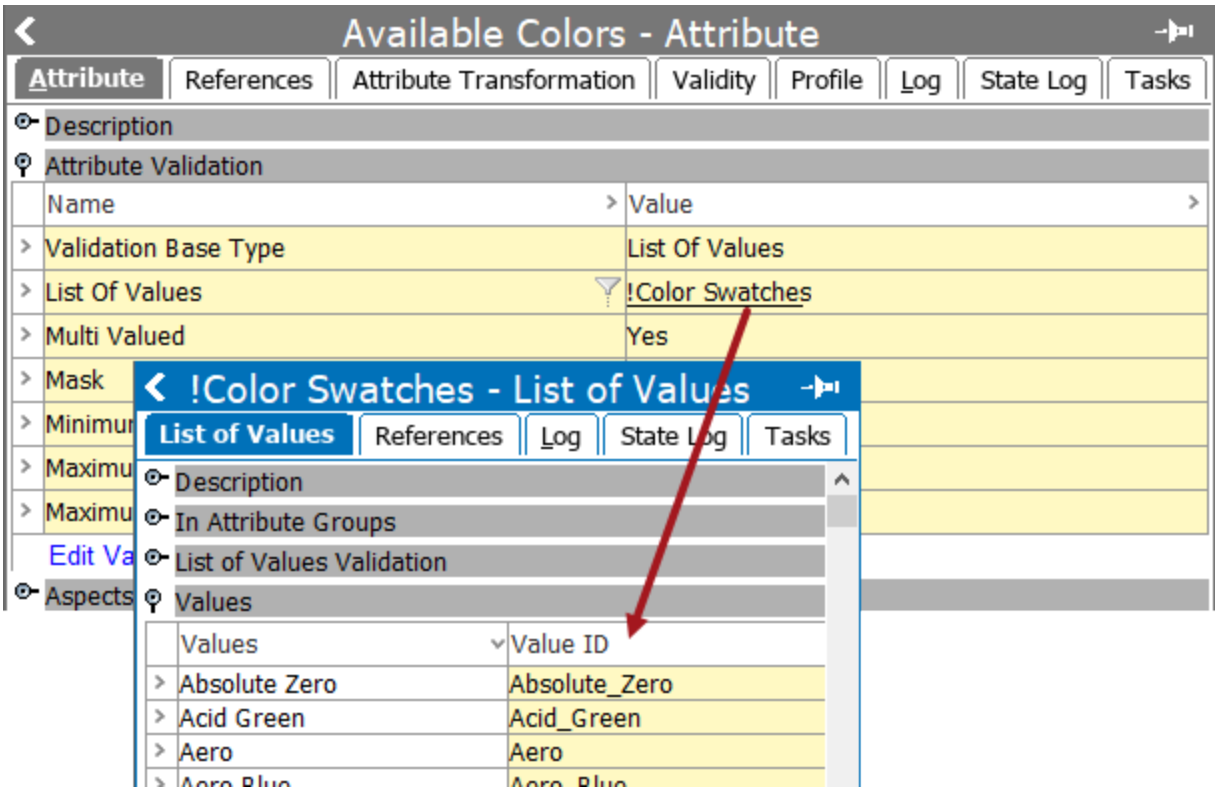
When mapping attributes or attribute groups, this aspect extracts the LOV Value-ID when present, otherwise it extracts the values in the attributes. For details on LOV Value IDs, see the **Use IDs on values** section of the **Creating an LOV** topic in the **System Setup / Super User Guide** documentation.

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.

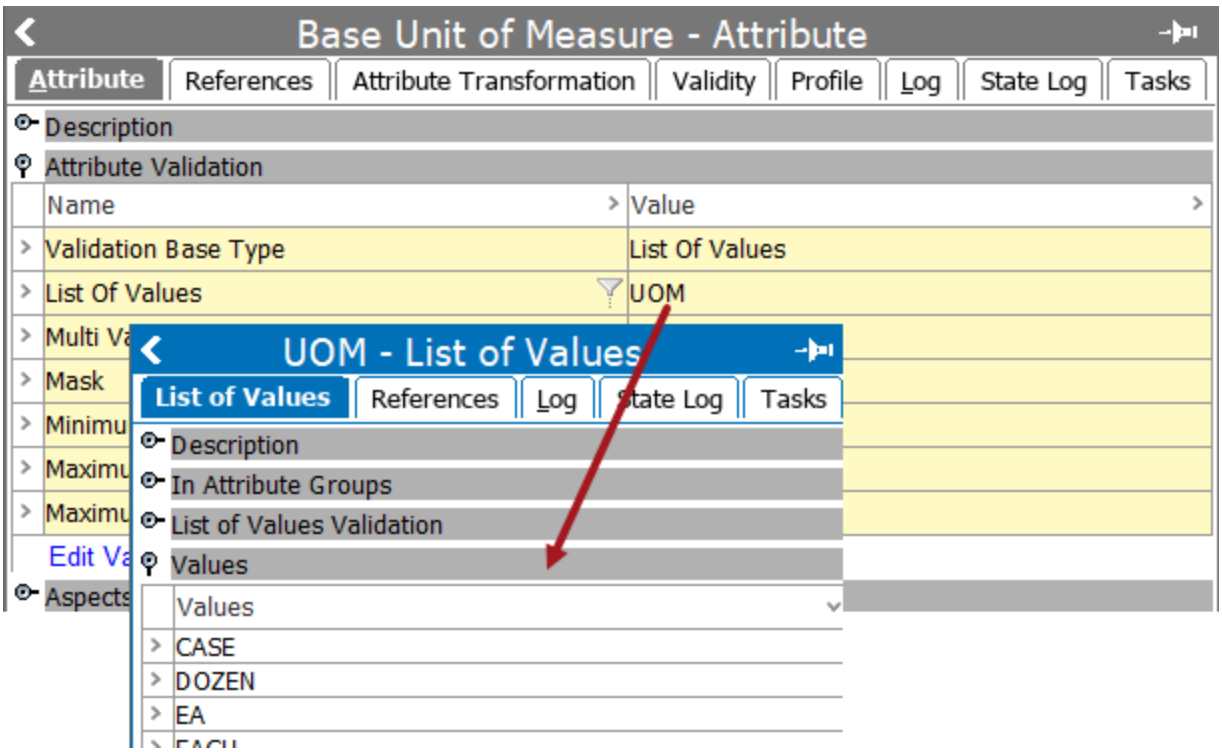
A common use for this aspect is when exporting ETIM via BMEcat2005 format, map an attribute group to the feature field and export the LOV Value IDs. See the **BMEcat 2005 Format Example** section below.

For general information on setting a Transformation aspect, see 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:



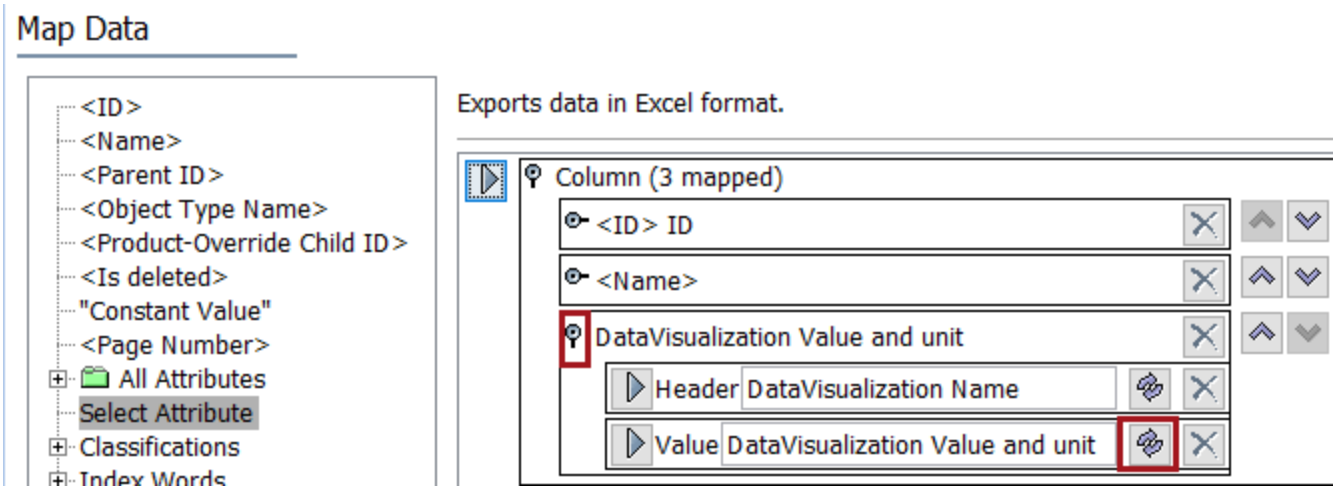
Using the single 'LOV Value-ID or Value' aspect for the attribute group results in exporting the Value IDs for the 'Available Colors' attribute and the Values for the 'Base Unit of Measure' attribute.

Additional options for exporting LOV aspect information are included in the following topics:

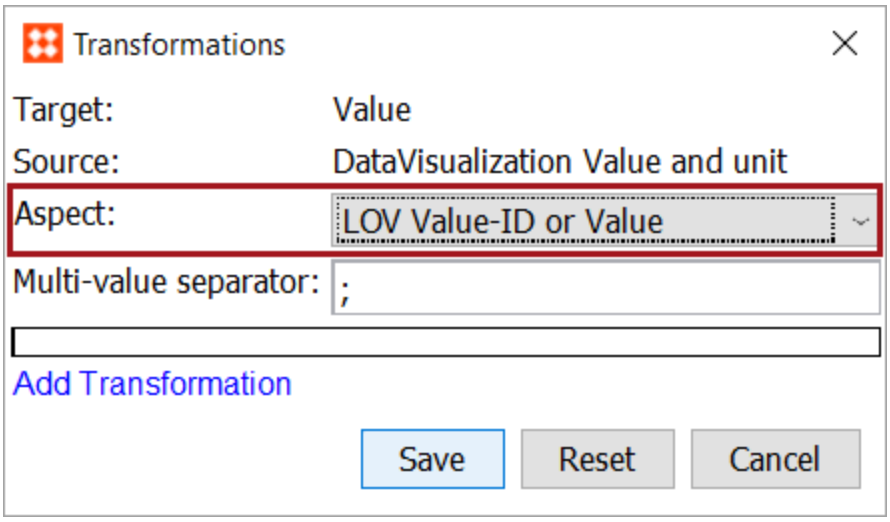
- LOV ID Aspect
- LOV Value-ID Aspect

Configuration

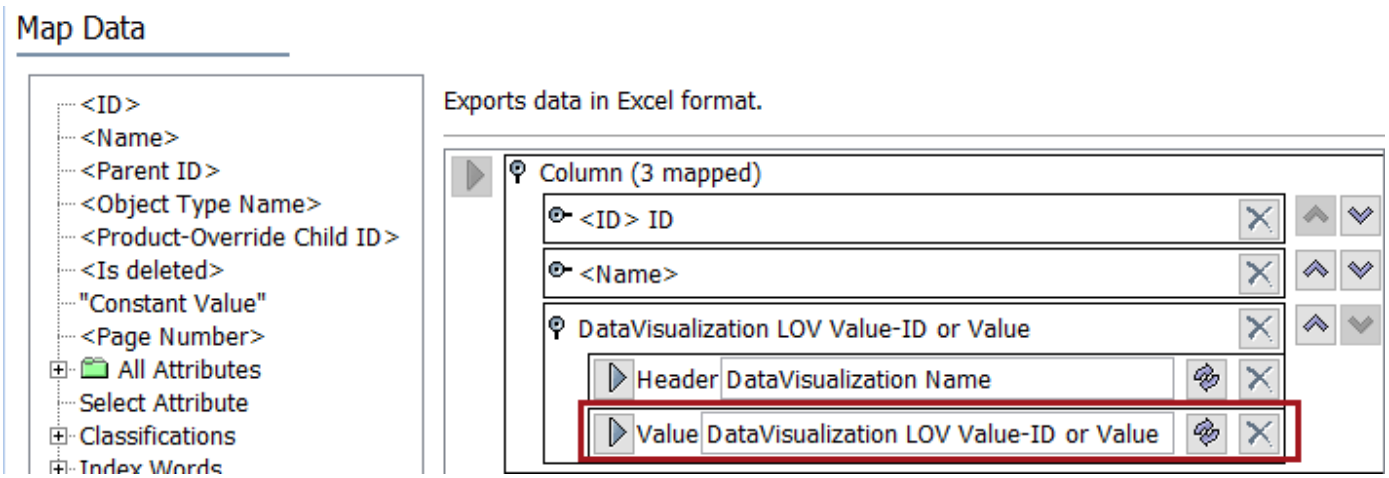
1. In the output tool select the data to be exported. For more information, see the **Data Exchange** topic.
2. On the Map Data step, map the required attribute or attribute group using the **Select Attribute** data source. For more information, see **Mapping Attributes via Select Attribute** section 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.

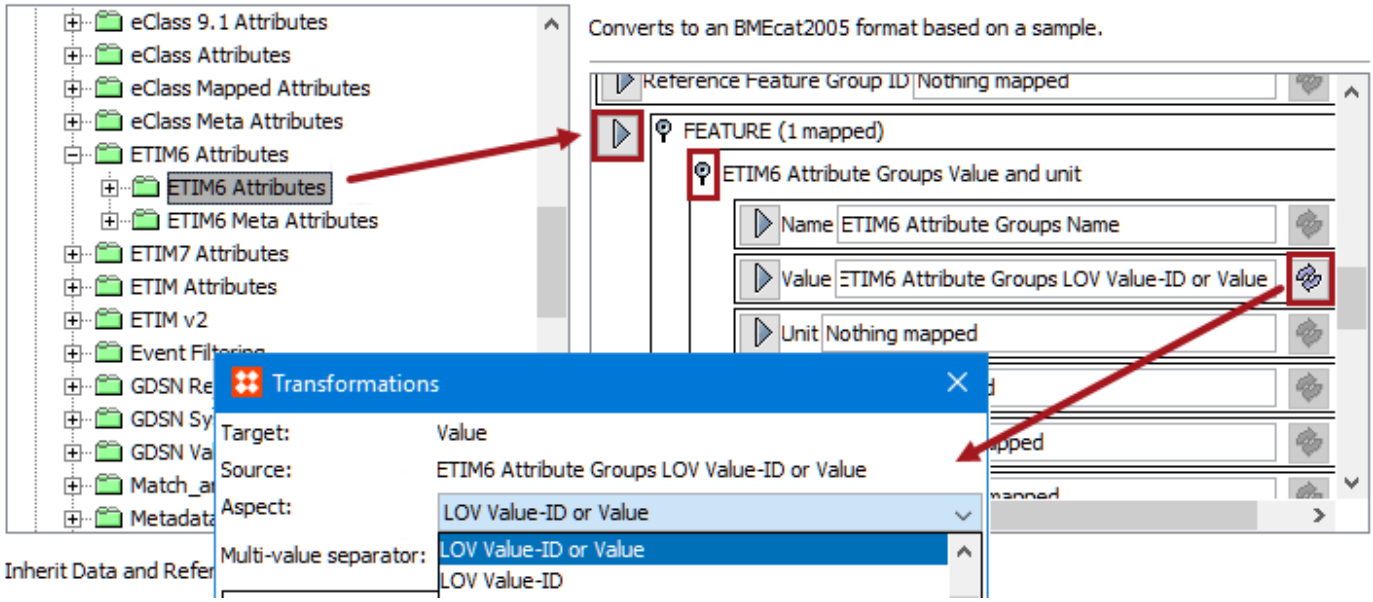
	A	B	C	D
1	<ID>	<Name>	Available Colors	Base Unit of Measure
2	18212	18212 L B	Acid_Green	CASE
3	18213	18213 M O	Aero	EA
4	18216	18216 L O	Absolute_Zero	DOZEN

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, see 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, see 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, see 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, see the **Transformations** topic in the **Resource Materials** online help.

Applying transformations

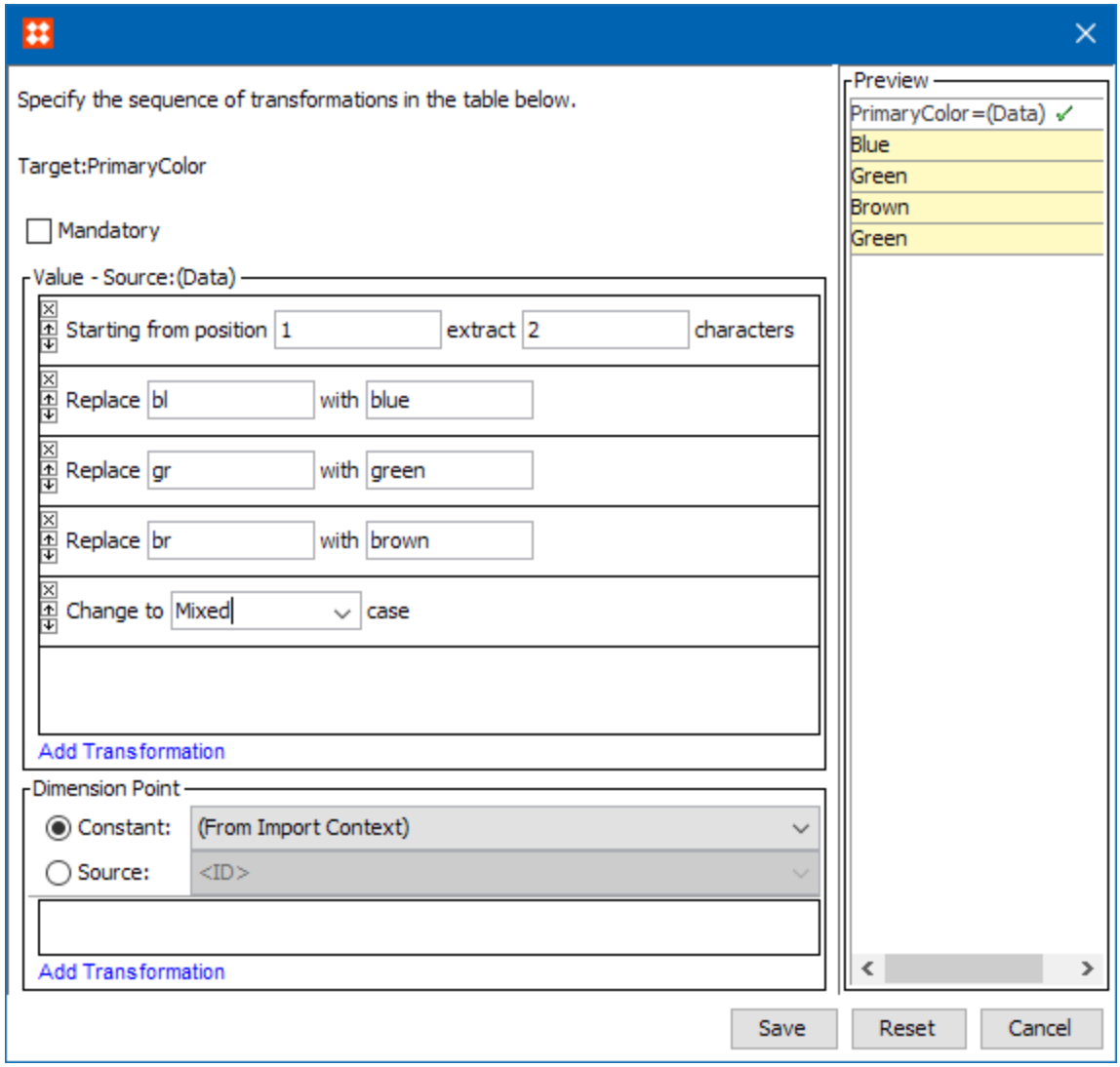
First map the inbound data columns and then apply the needed transformations.

1. Map the '<ID>' column to the ID in STEP. For details, see 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, see the **Mapping the Same Column Multiple Times** section of the **Inbound Map Data - Map** topic.

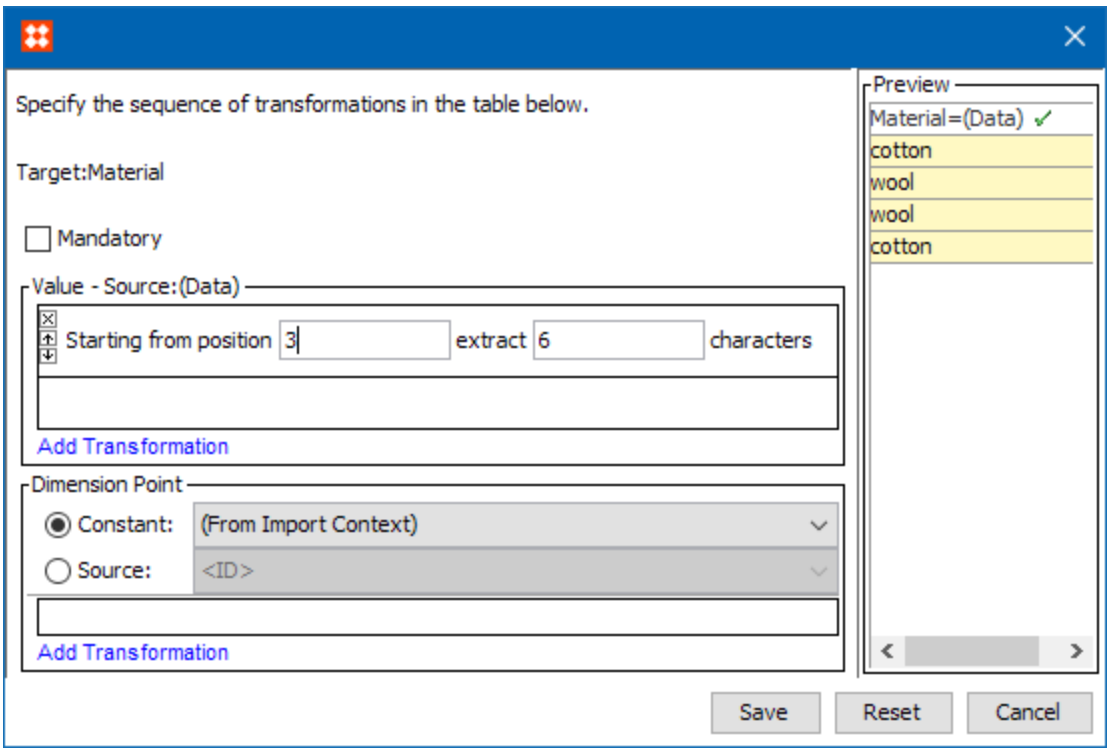
3. In the Results area, select the column mapped to the 'Primary Color' attribute, and click **Transform**.
4. Add required transformations:

- 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.

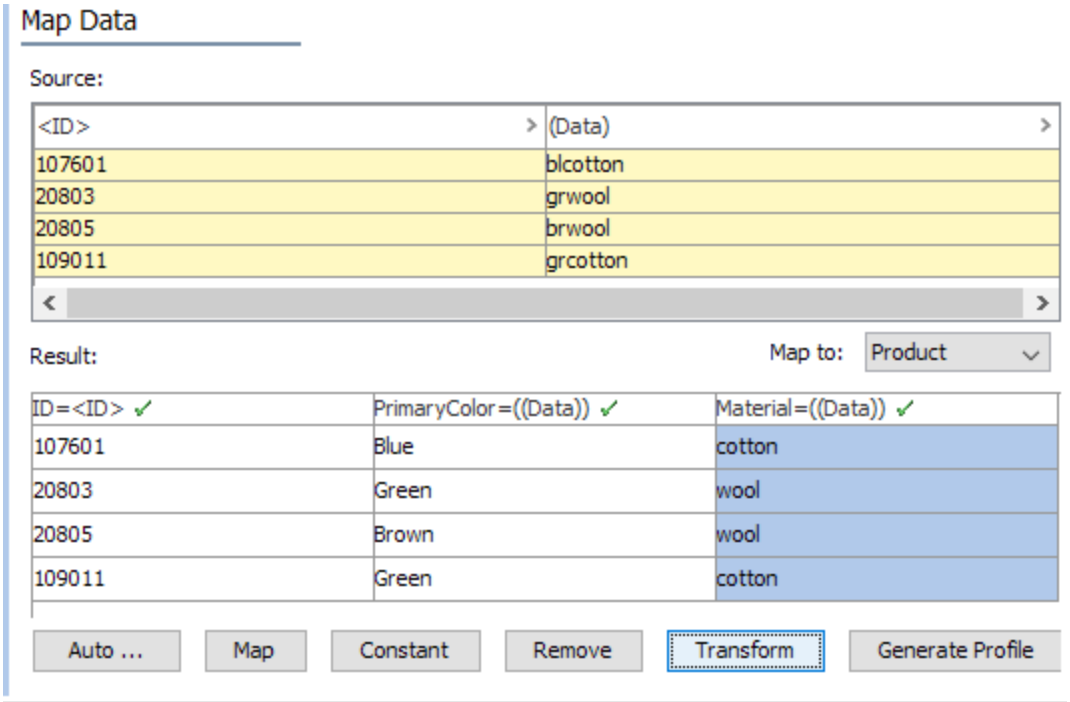


Note: Transformations work in sequence from top to bottom.

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

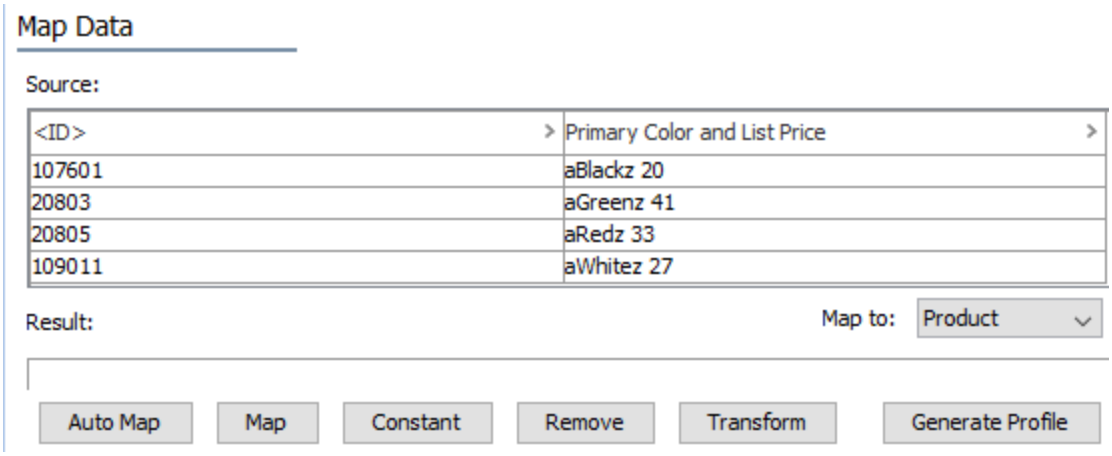
Transformations can improve the quality of the data you want to import. For general information about transformations, see the **Inbound Map Data - Transform** section.

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:

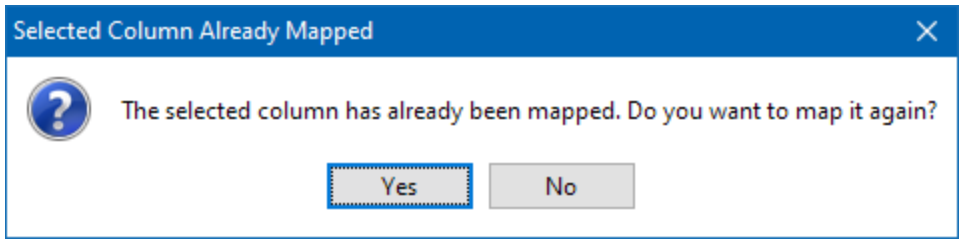
- Extract characters between two pieces of text in the value
- Extract characters after a given text in the value

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':



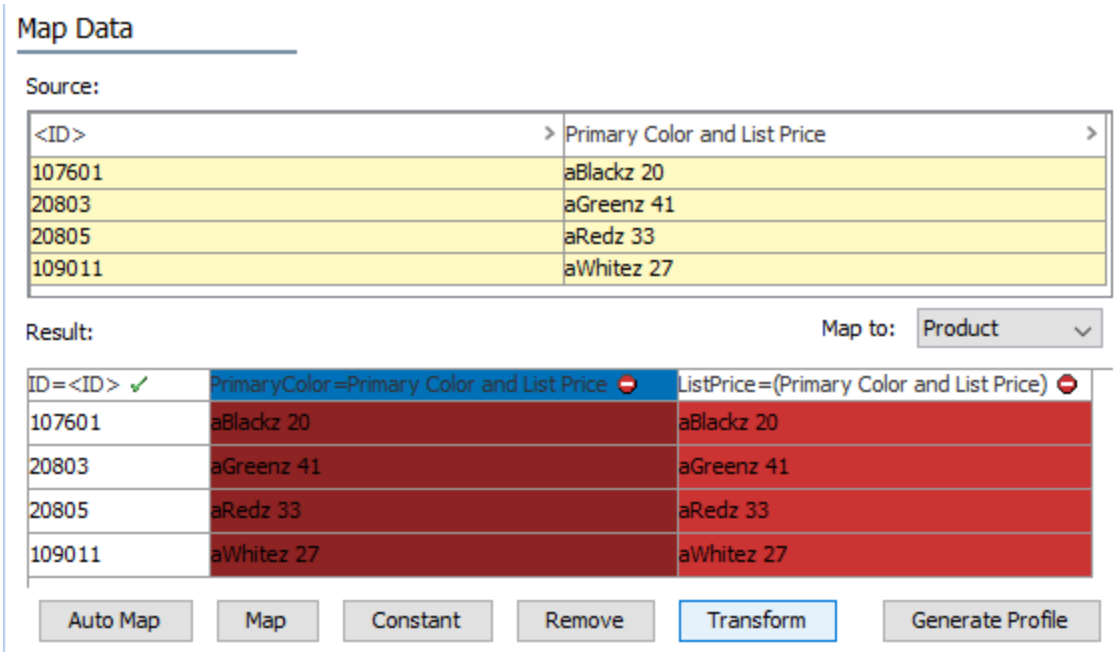
Configuration

1. Load the import file into the inbound tool. For more information, see the **Data Exchange** topic.
2. Map the object ID in the import tool. See **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.

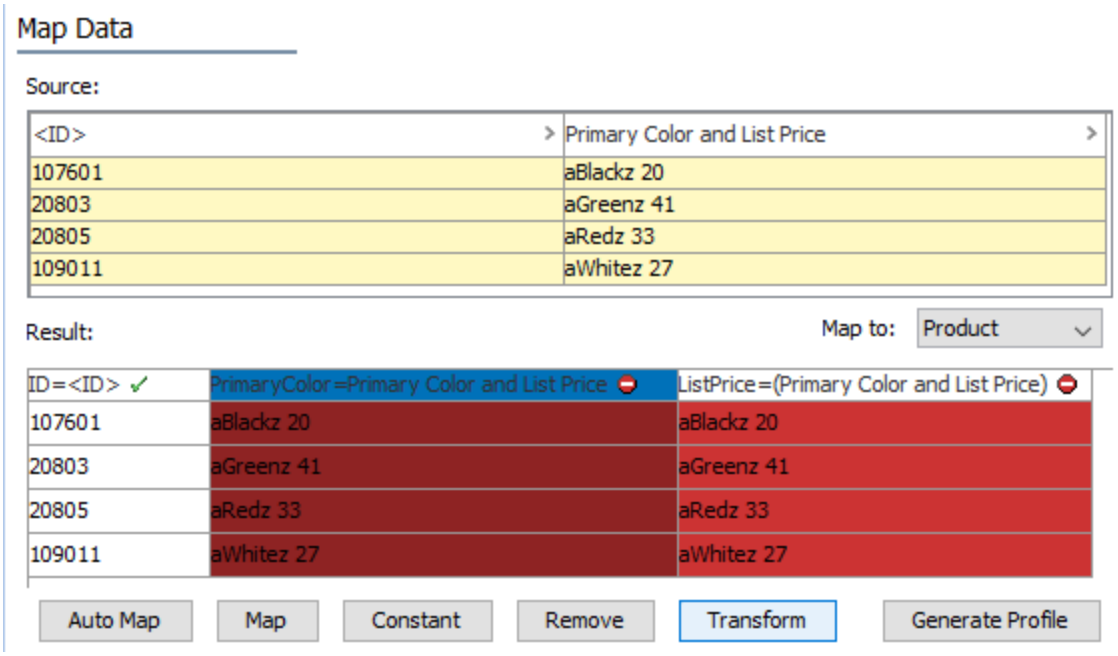


In this example, we map 'Primary Color and List Price' two times, once to the Primary Color attribute, and again to the List Price attribute. In our example, the Result section shows the newly mapped items in red because the current values are not valid.

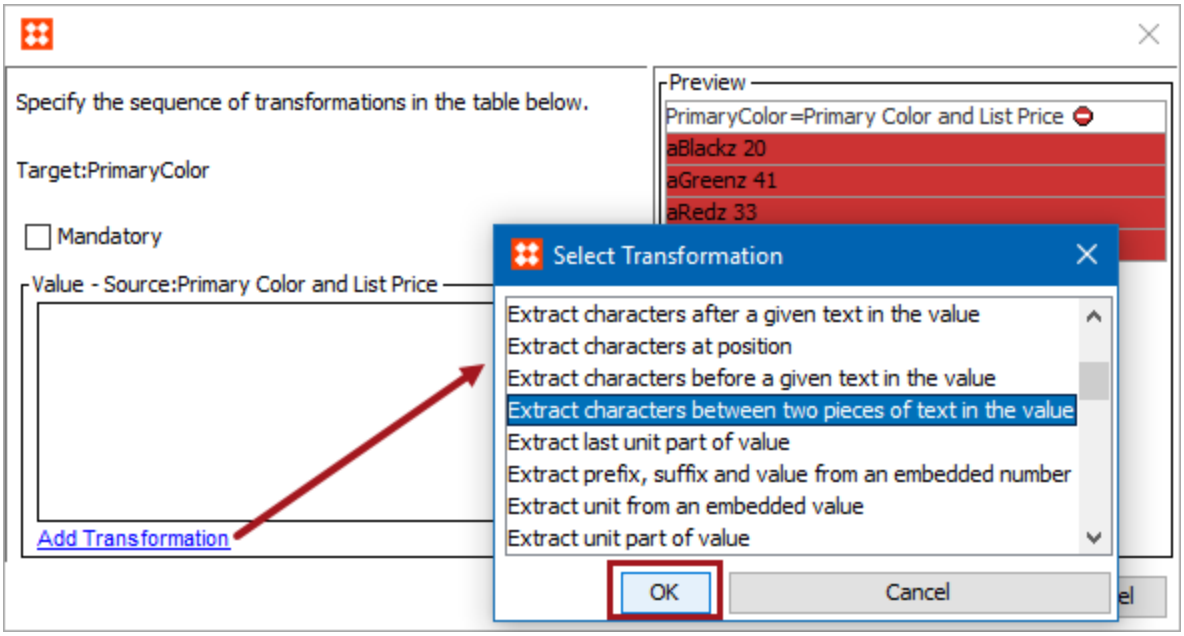
Note: Hover over a red cell to display the error message.



4. Select the one of the result columns mapped to an attribute and click the **Transform** button.



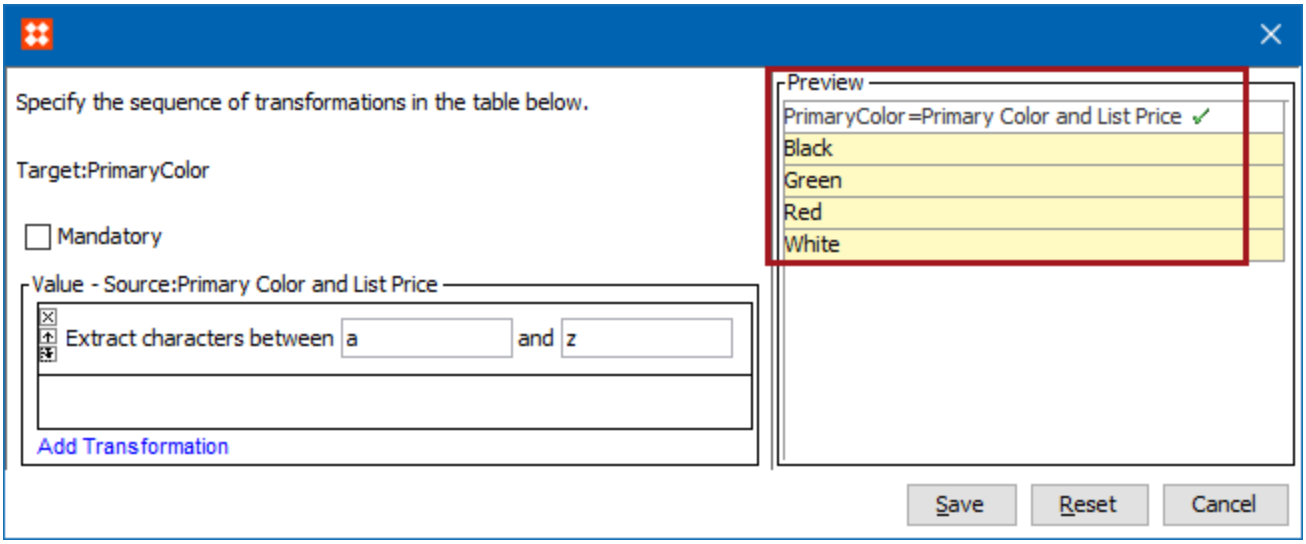
5. In the Transformation window, click on the **Add Transformation** link, select the **Extract characters between two pieces of text in the value** transformation, and click **OK**.



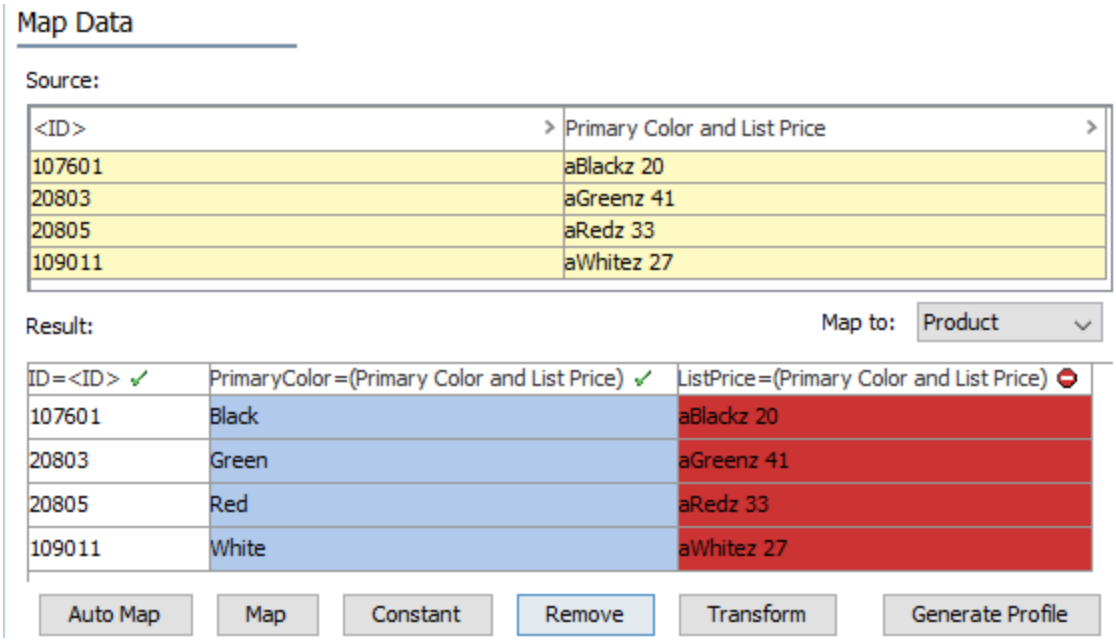
6. Enter the characters that should be used to determine that data that needs to be extracted and click out of the text fields. In this example, the color is always between the letters 'a' and 'z.' Click or tab out of the parameter to complete the configuration. The Preview pane shows the results of 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.

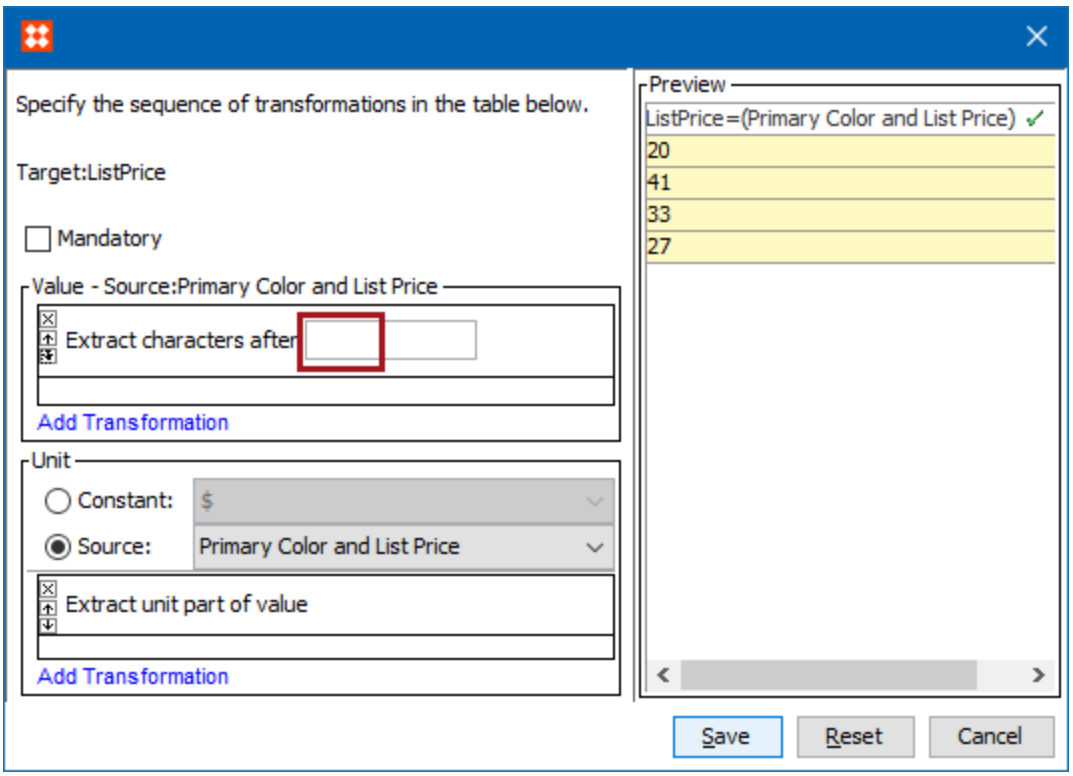
Important: Transformations are evaluated from the top down. Carefully consider the order of the transformations applied. To remove or rearrange transformations, use the Delete, Up arrow, and Down arrow buttons .



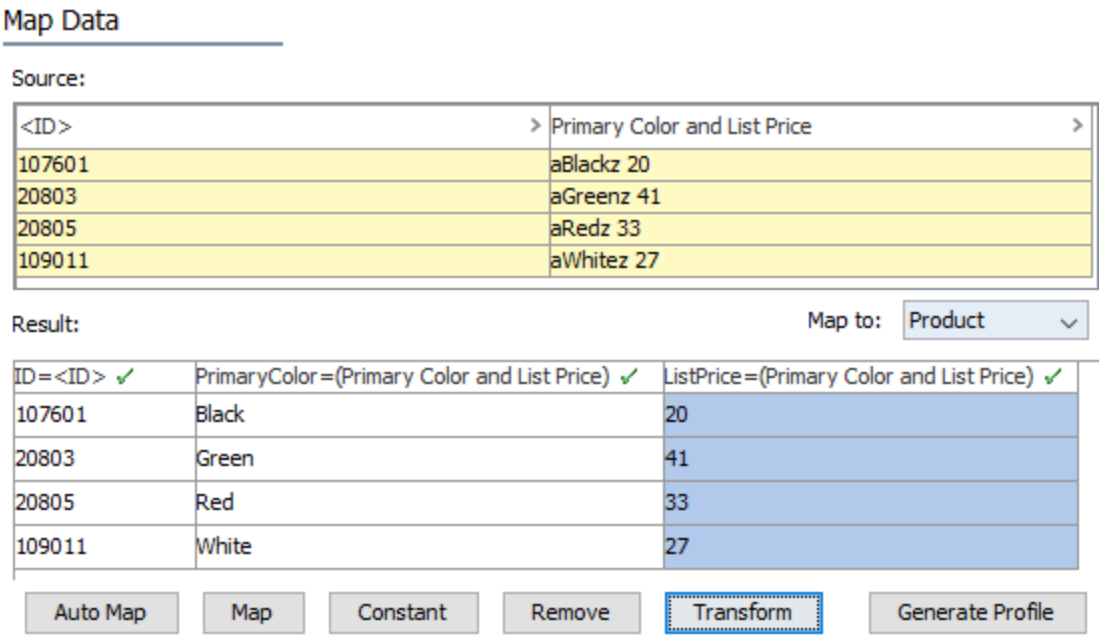
7. If necessary, check the 'Mandatory' option to skip objects with empty values during import.
8. Click **Save** to display the transformed data in the Map Data step.



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.



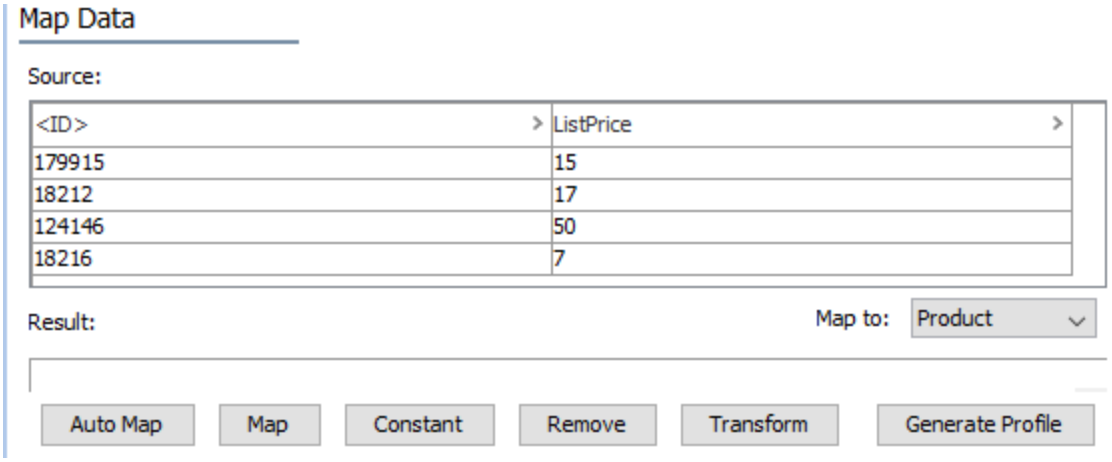
11. Click **Save** to display all transformed data in the Map Data step.



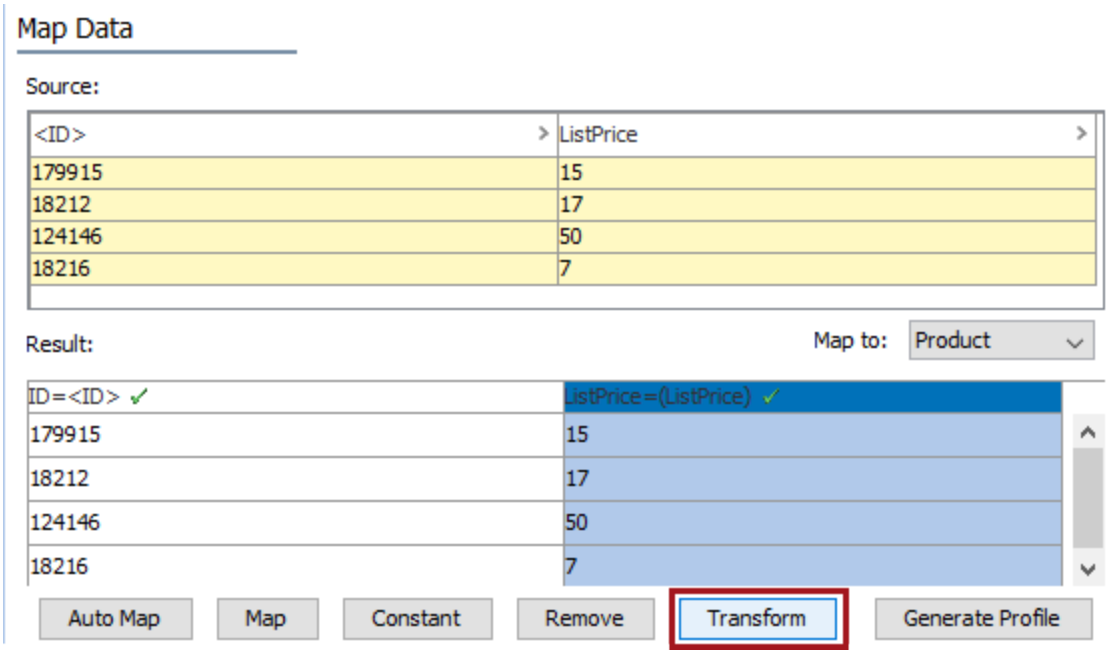
Format Number Example

Transformations can improve the quality of the data you want to import. For general information about transformations, see the **Inbound Map Data - Transform** section.

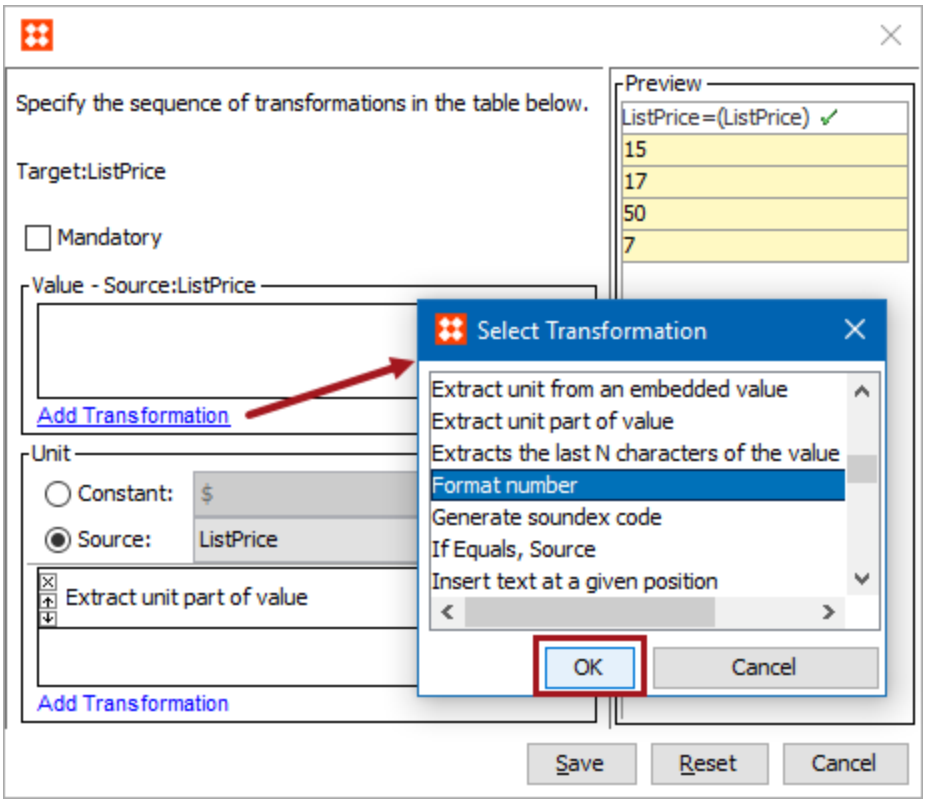
Transformations can be used to format numbers, for example, to set a specified number of decimals, as required by an attribute. For example, the following data is being imported, and contains no decimal places:



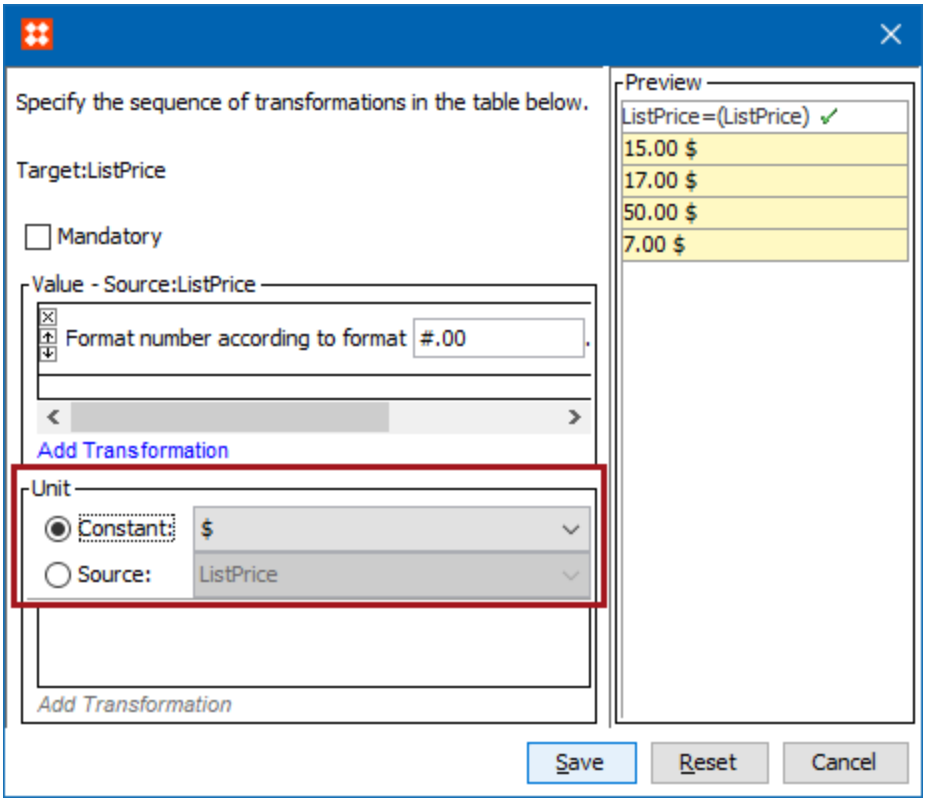
1. Map the object ID in the import tool.
2. Map the source column that holds the data to the required attribute. In this example, we map 'ListPrice' to the attribute of the same name.
3. Select the mapped column with the price data and click the **Transform** button.



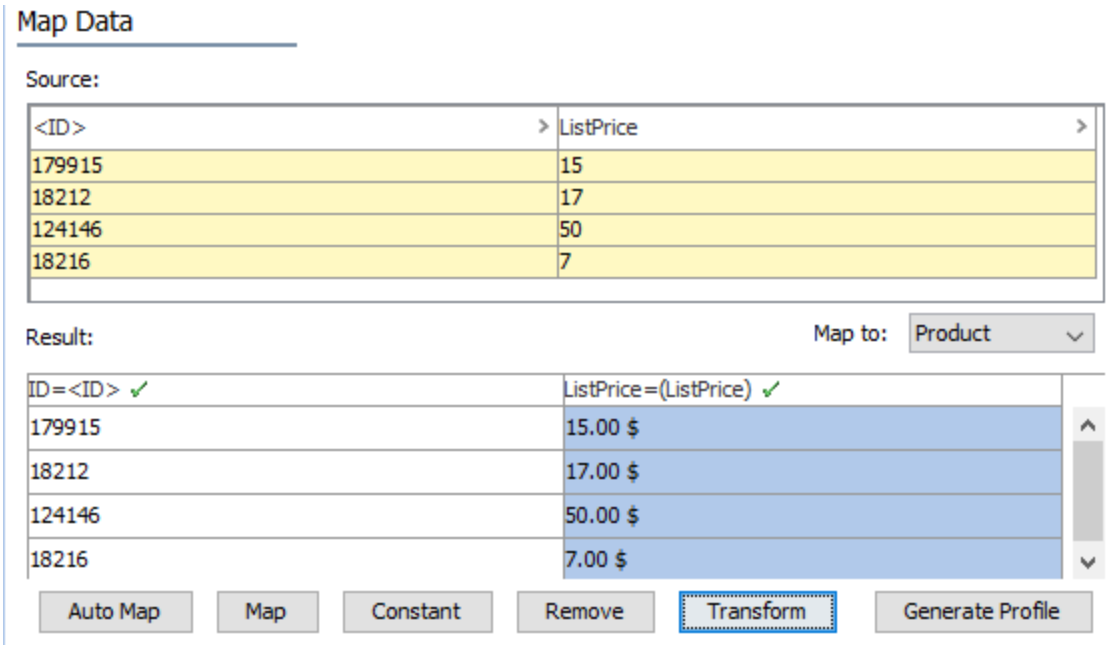
4. Click the **Add Transformation** link, and choose the 'Format number' transformation. Click **OK**.



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.



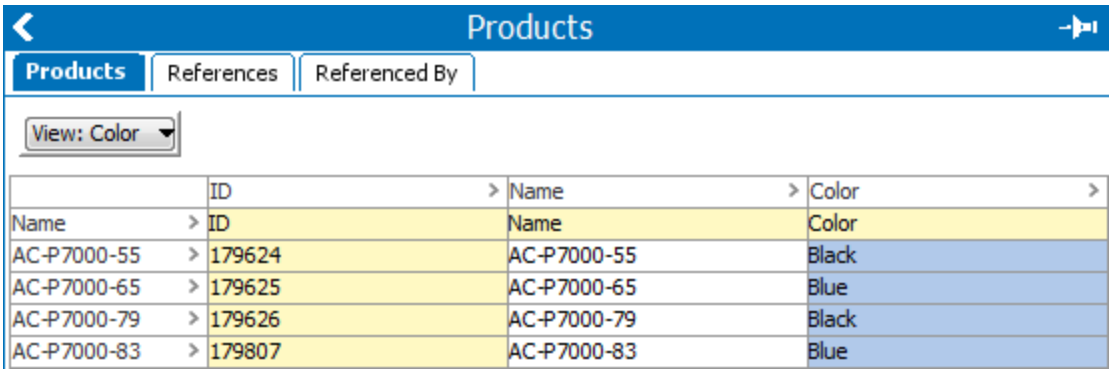
Header and Value Aspects Example

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, see the **Outbound Map Data - Data Source** documentation.

For more information on the available aspects, see the **Aspect - Transform Outbound** documentation.

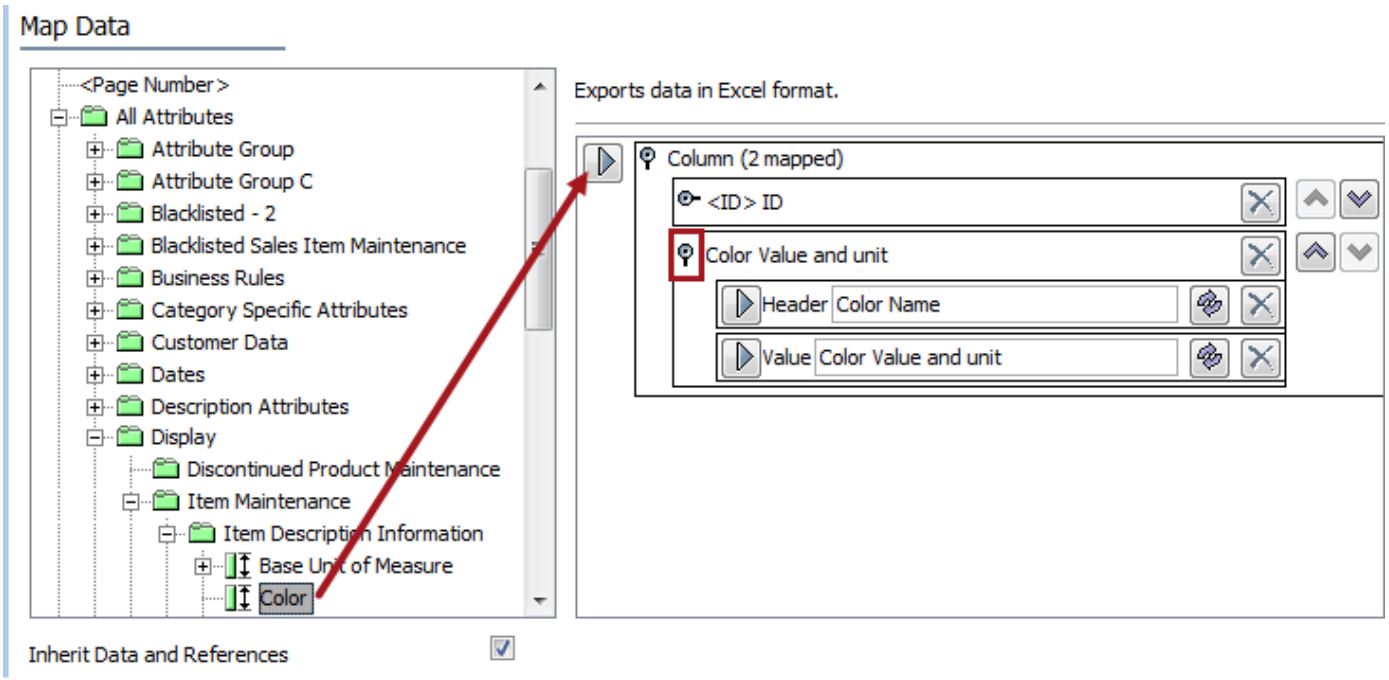
Default Header Aspect and Default Value Aspect

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.



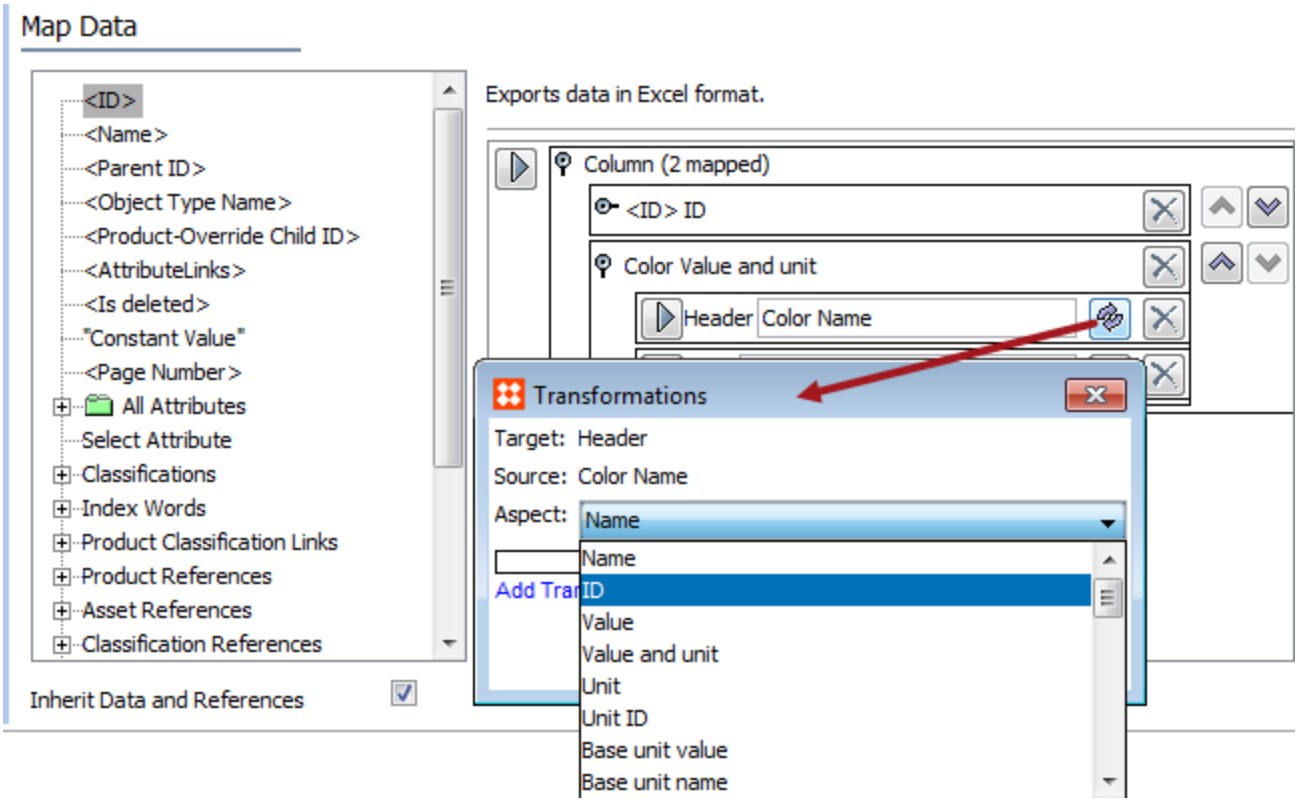
Results

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.



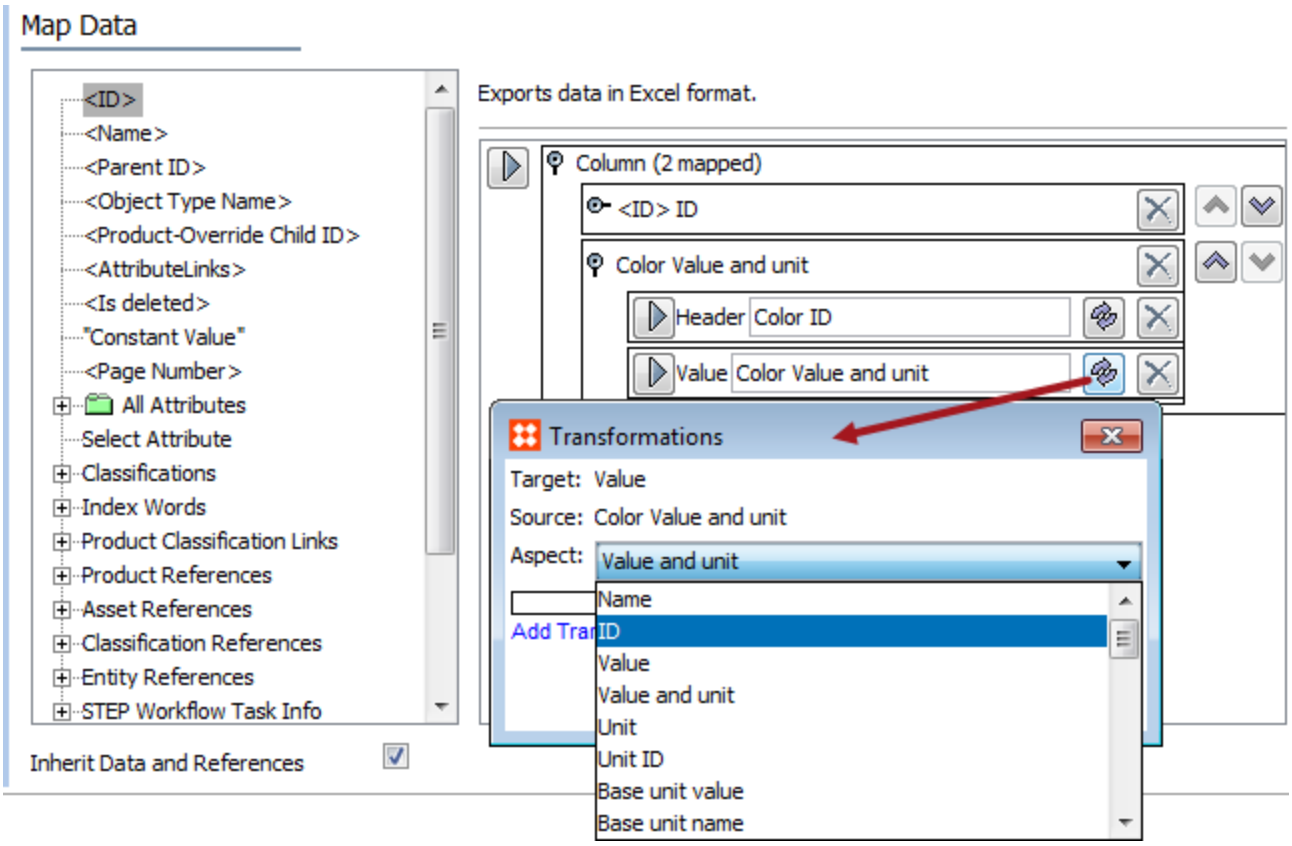
Results

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.



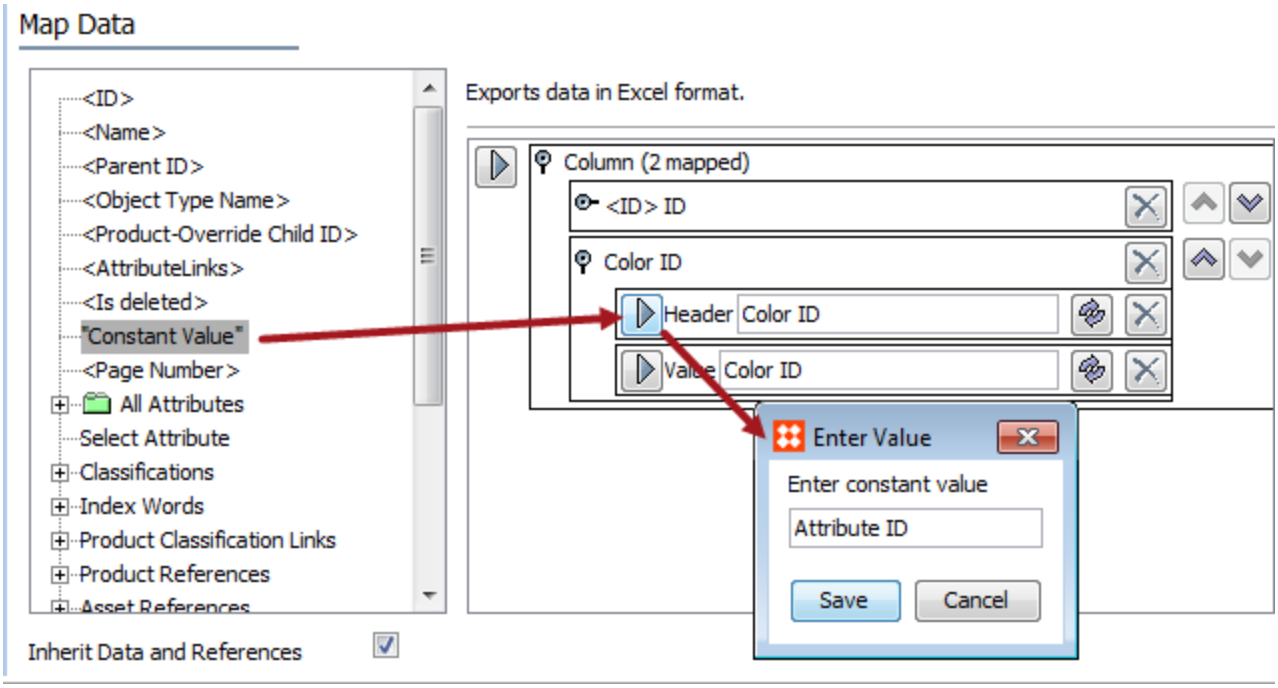
Results

Exporting with this mapping results in the following file, which includes the attribute ID in both the header and the value for each product:

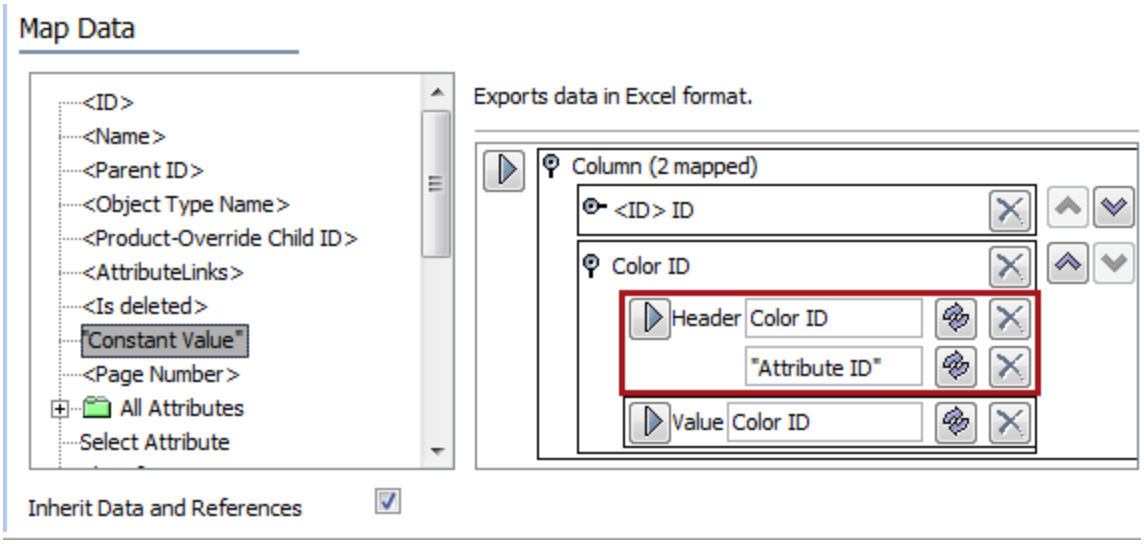
	A	B
1	<ID>	ColorID
2	179624	ColorID
3	179625	ColorID
4	179626	ColorID
5	179807	ColorID

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, see 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.'

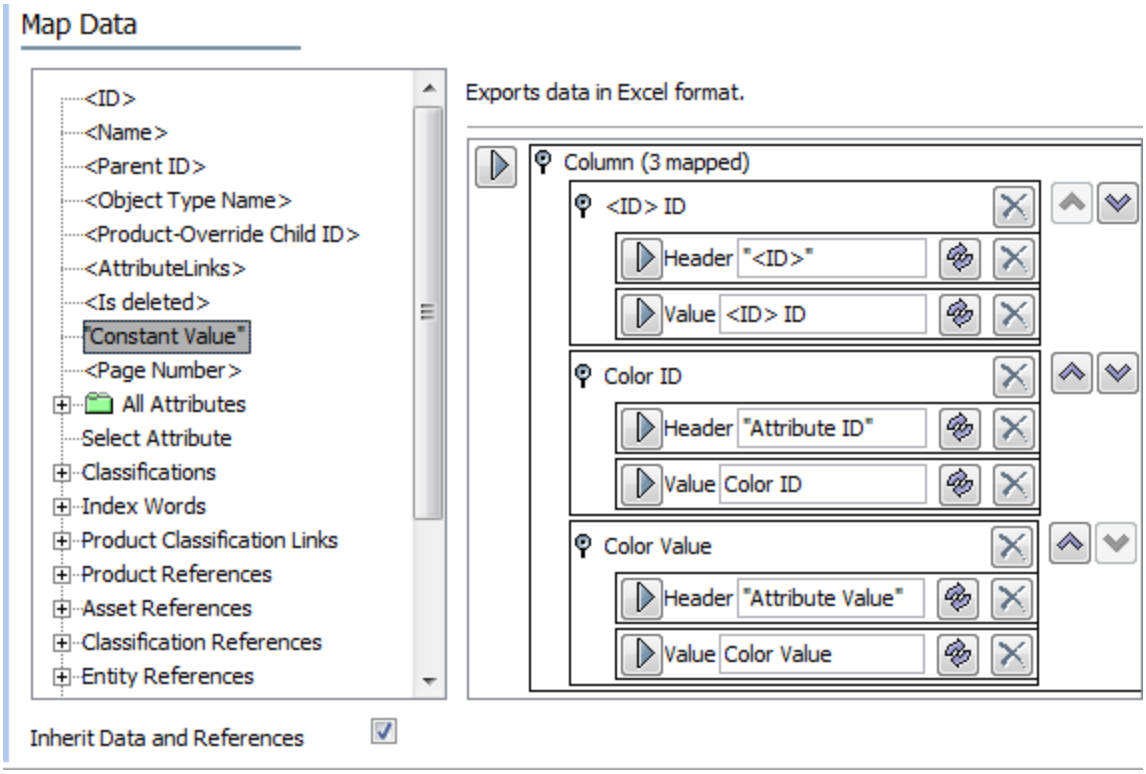
	A	B
1	<ID>	ColorIDAttribute ID
2	179624	ColorID
3	179625	ColorID
4	179626	ColorID
5	179807	ColorID

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

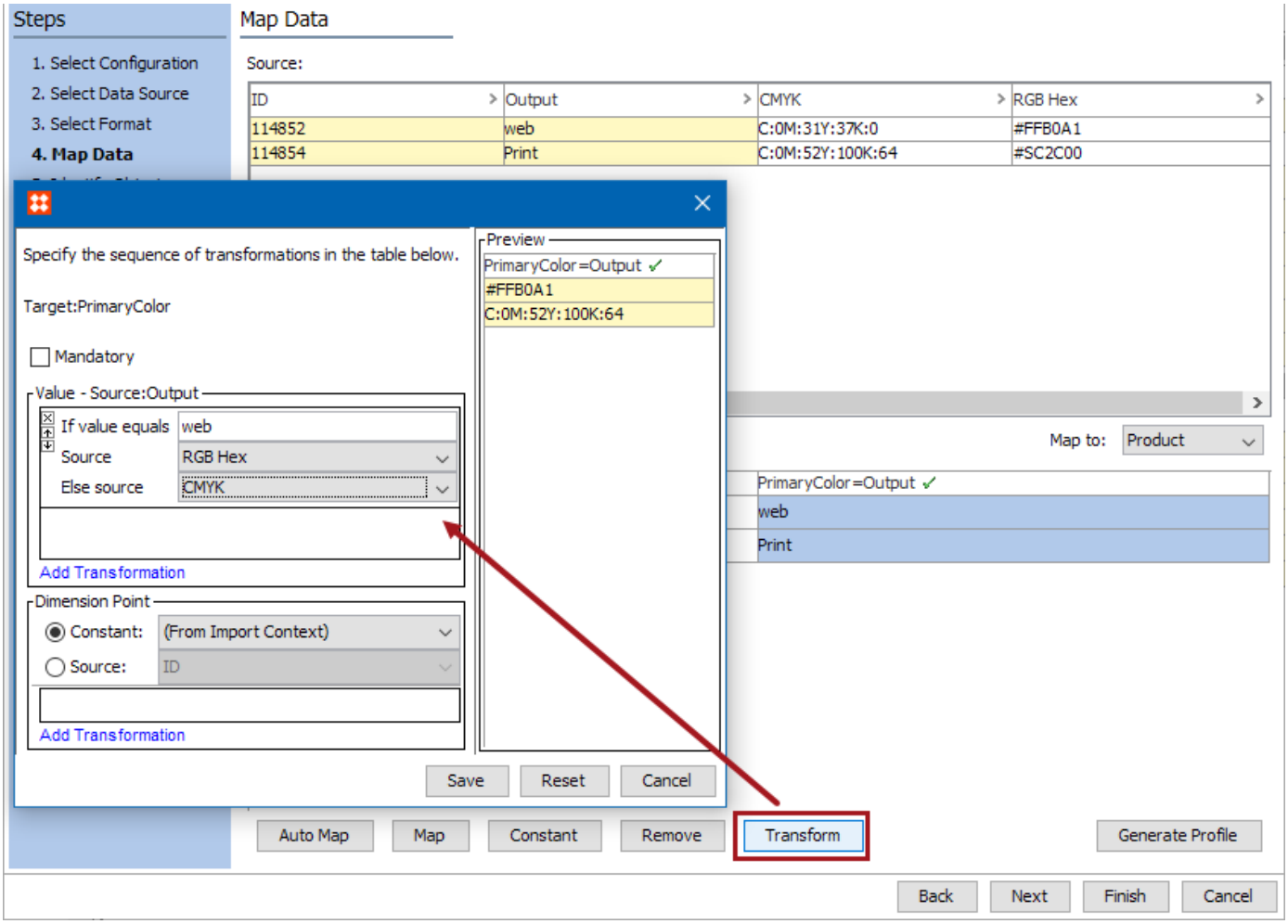
If Equals Source Example

Transformations can improve the quality of the data you want to import. For general information about transformations, see 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.

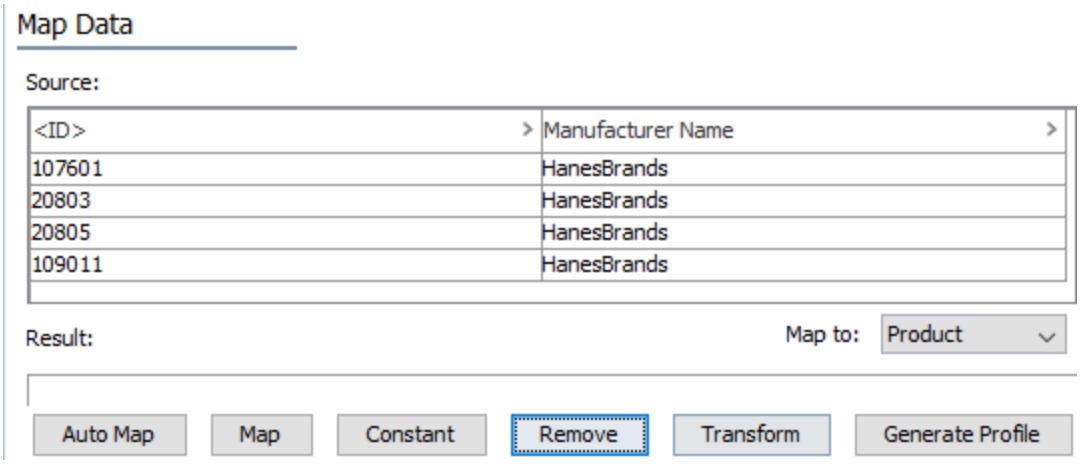


Insert Text Example

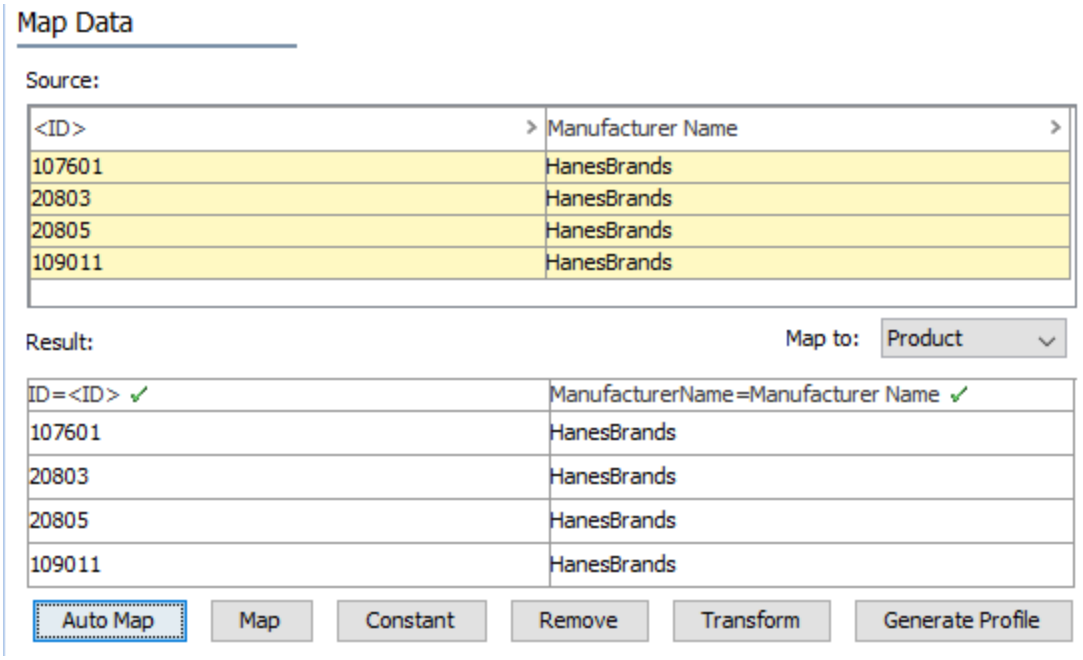
Transformations can improve the quality of the data you want to import. For general information about transformations, see the **Inbound Map Data - Transform** section.

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.

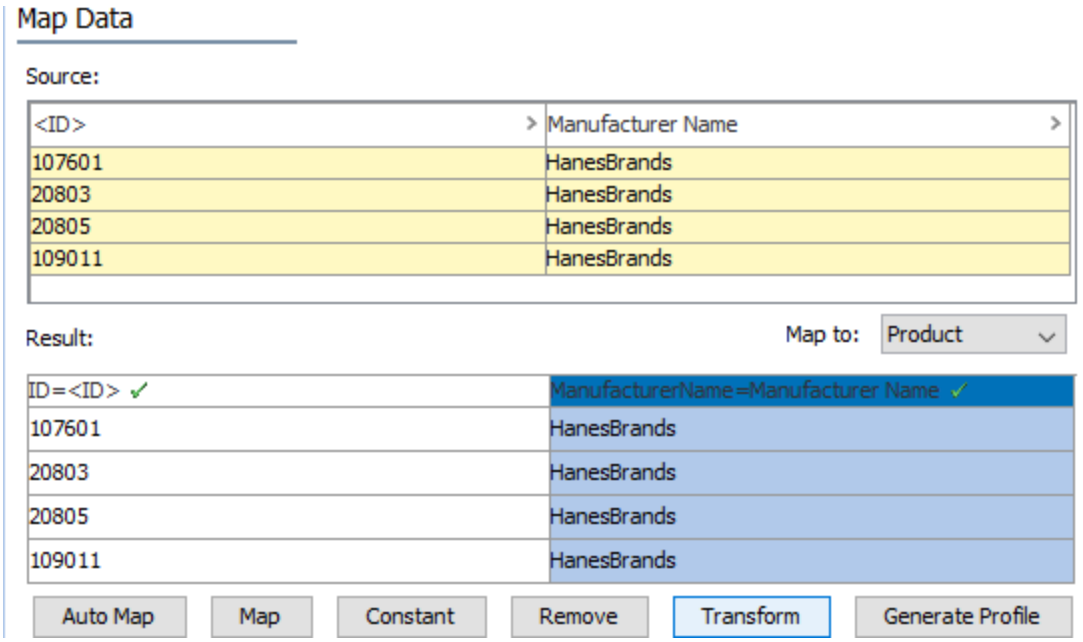
This example describes how to add new text to a field in an import file, and prepare it to be loaded into an attribute in STEP. The following data is being imported and contains incomplete data for the 'Manufacturer Name' attribute:



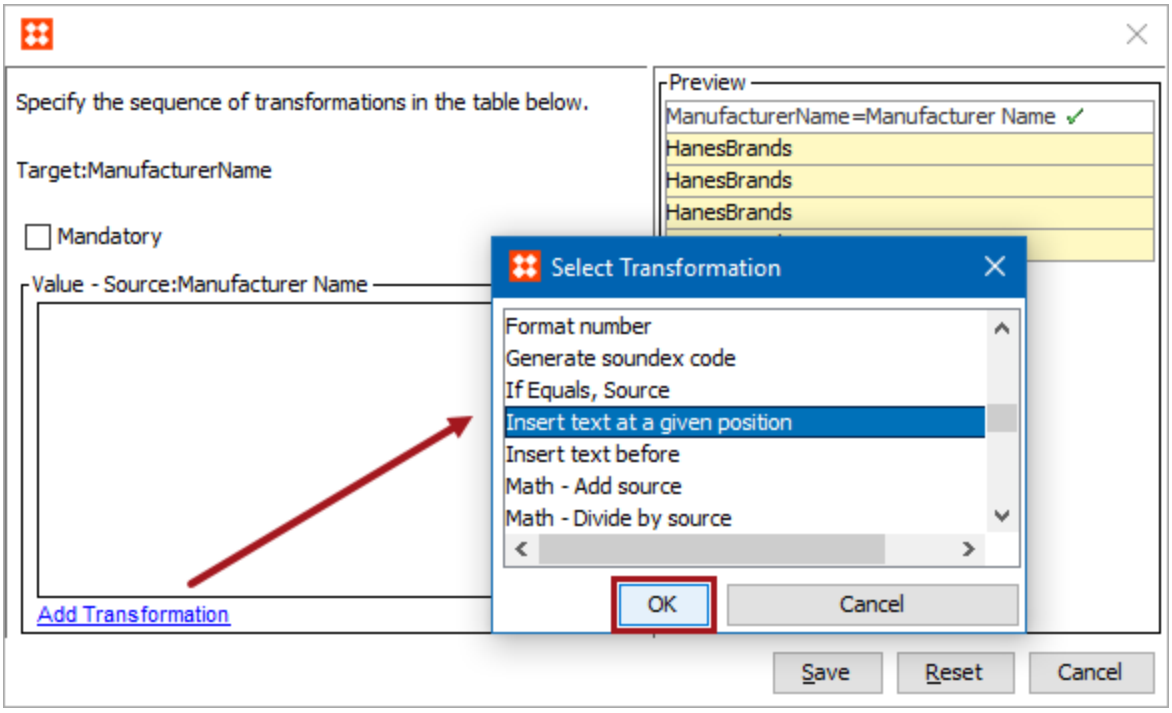
1. Load the import file into the inbound tool. For more information, see the **Data Exchange** topic.
2. Map the object ID in the import tool. See **ID or Key - Map Inbound** for details.
3. Map the next source column to the required attribute. In this example, we map 'Manufacturer Name' to the Manufacturer Name attribute.



4. Select the result column that contains data to be modified and click the **Transform** button.

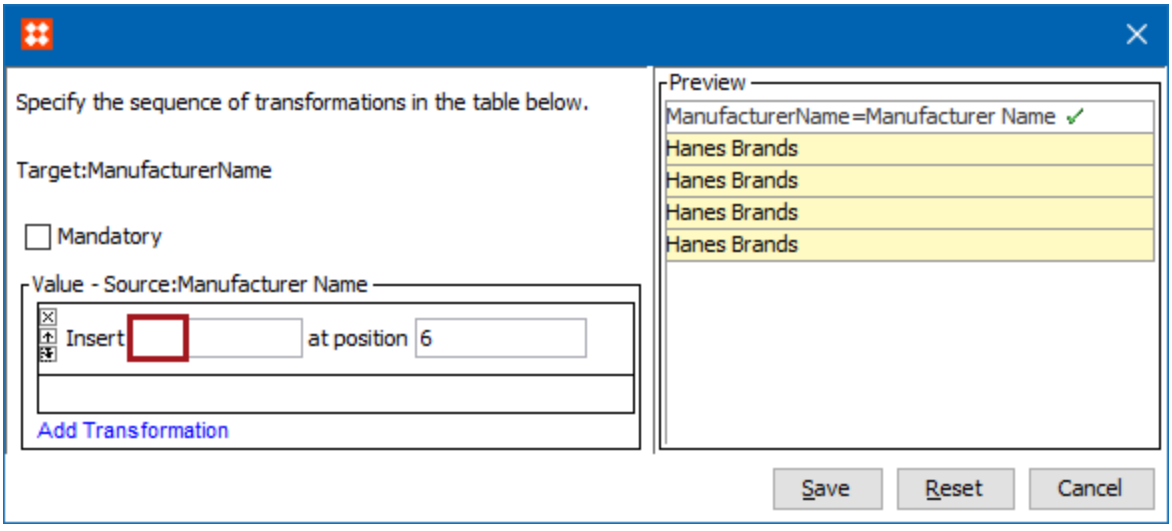


5. In the Transformation window, click on the **Add Transformation** link, select the **Insert text at a given position** transformation, and click **OK**.

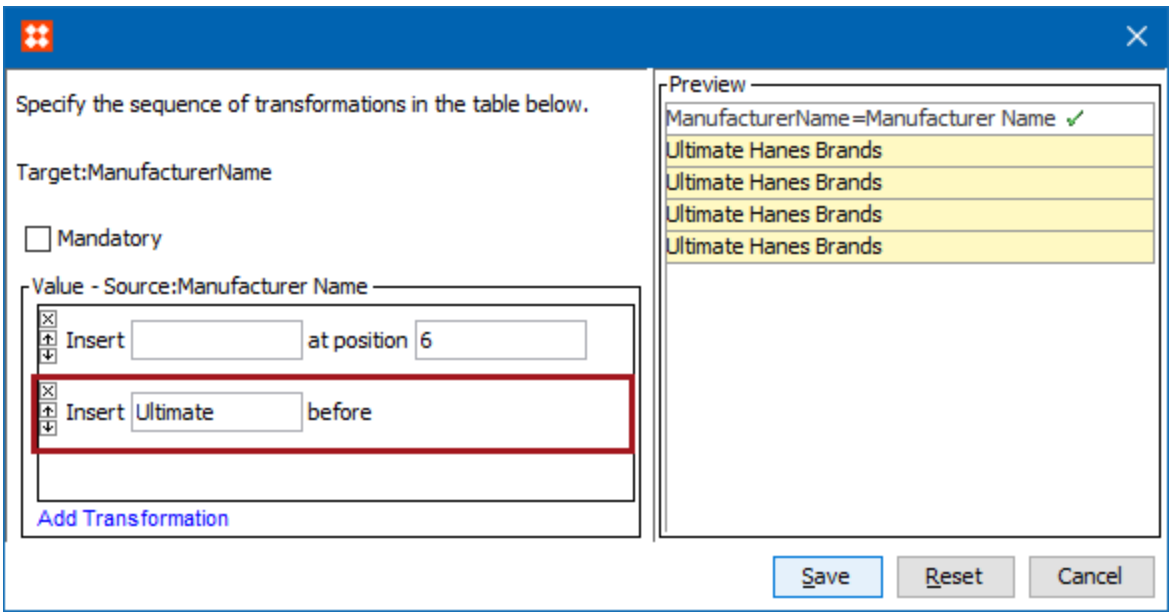


6. Although not obvious in the following image, add a space in the **Insert** parameter, and indicate the position of the space by adding a number in the **at position** parameter. In this example, since the space belongs between the words 'Hanes' and 'Brands', we enter the number 6 for the position 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.



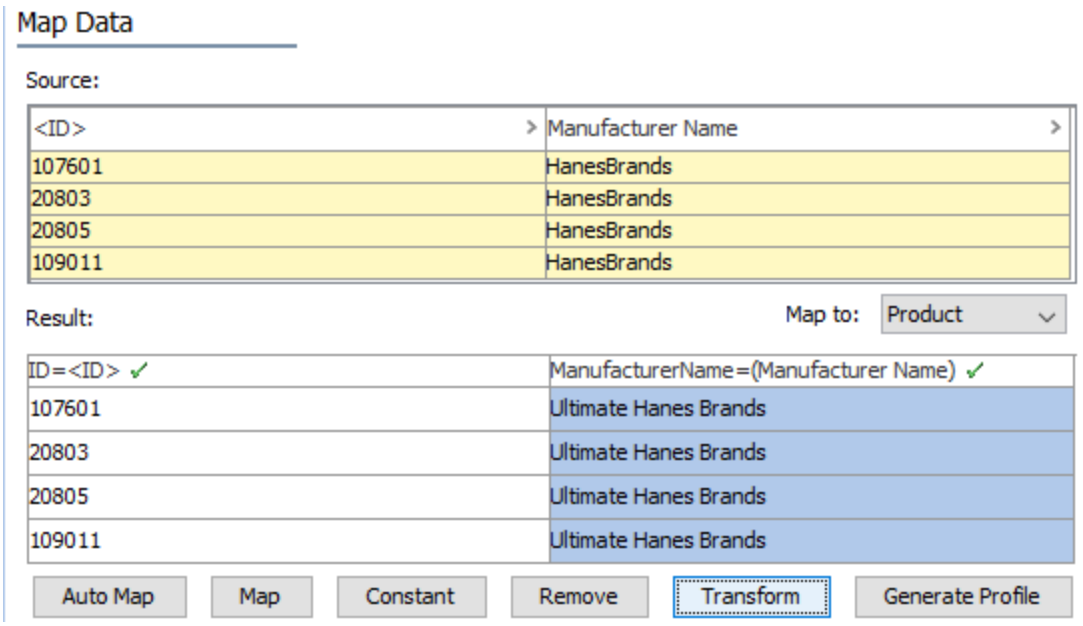
7. In the Transformation window, click on the **Add Transformation** link again, select the **Insert text before** transformation, and click **OK**.
8. In the **Insert before** parameter, add the text that should precede the existing value. Remember to add any necessary space characters. Click or tab 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.

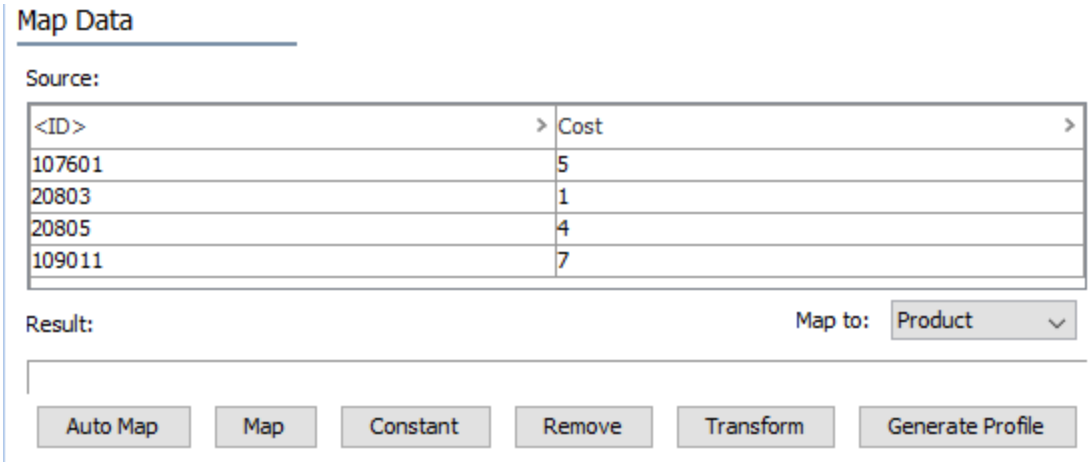

Map Data
Source:
<ID>
Manufacturer Name
107601
HanesBrands
20803
HanesBrands
20805
HanesBrands
109011
HanesBrands
Result:
Map to:
Product
ID=<ID>
ManufacturerName=(Manufacturer Name)
107601
Ultimate Hanes Brands
20803
Ultimate Hanes Brands
20805
Ultimate Hanes Brands
109011
Ultimate Hanes Brands
Auto Map
Map
Constant
Remove
Transform
Generate Profile

Math Operation Example

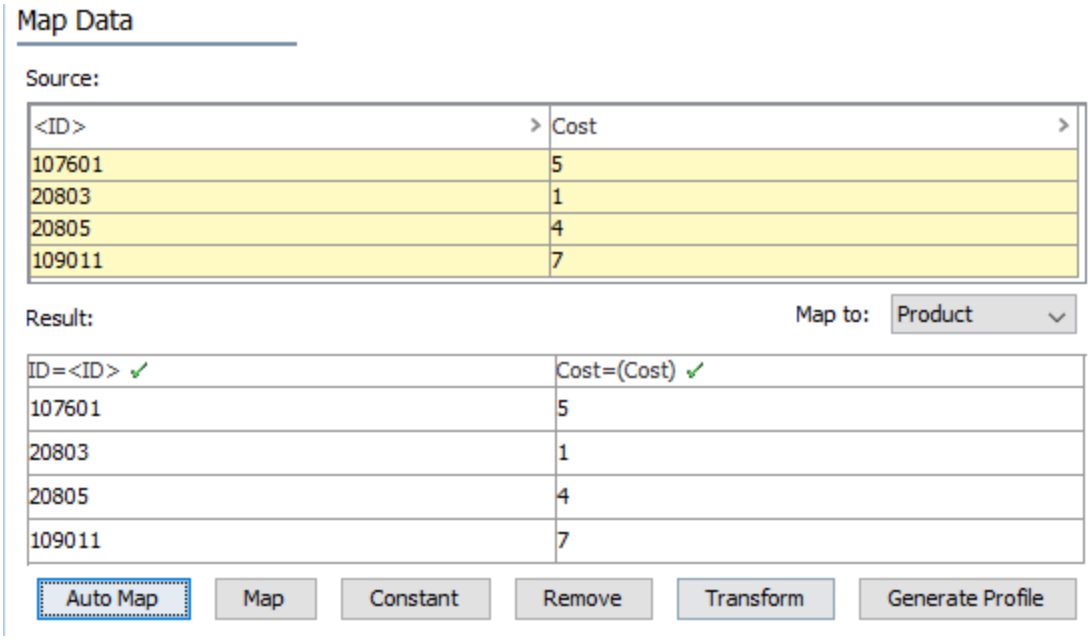
Transformations can improve the quality of the data you want to import. For general information about transformations, see the **Inbound Map Data - Transform** section.

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.

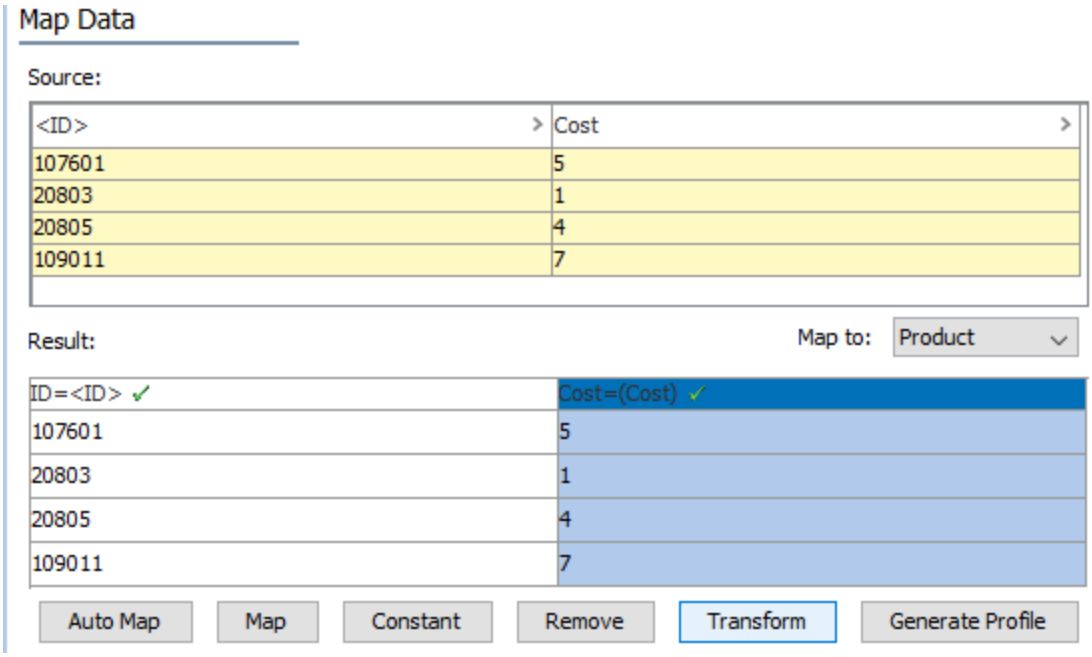
This example applies each of the math operations to a single field of the import file as shown below.



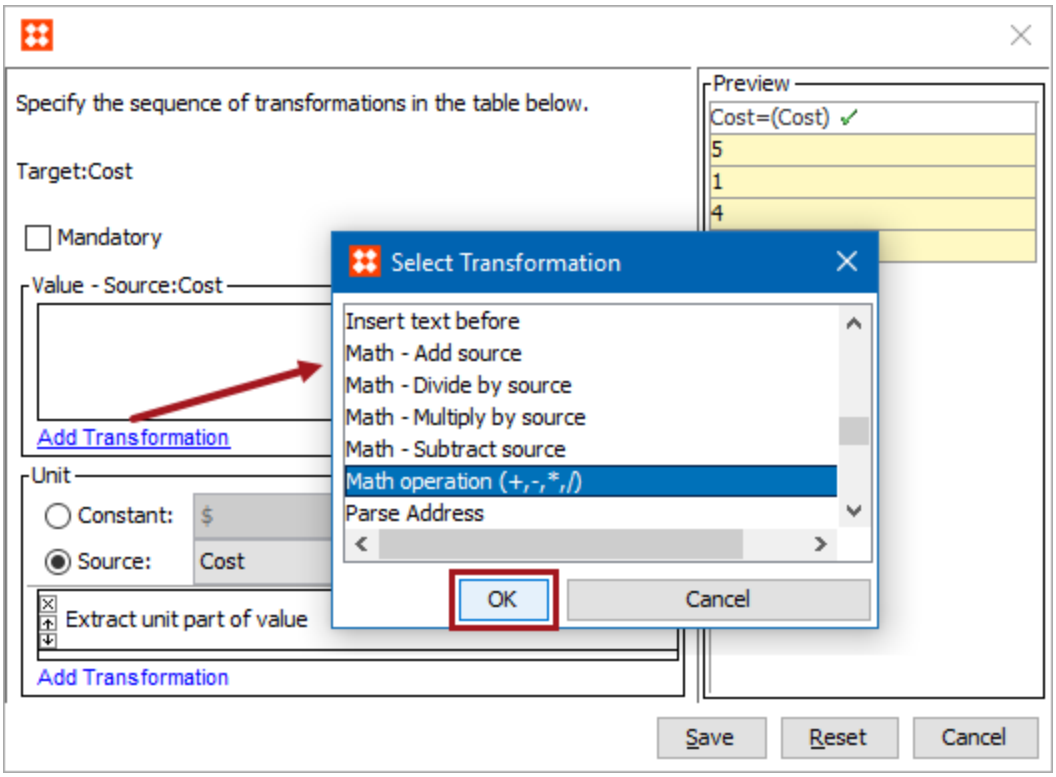
1. Load the import file into the inbound tool. For more information, see the **Data Exchange** topic.
2. Map the object ID in the import tool. See **ID or Key - Map Inbound** for details.
3. Map the next source column to the required attribute. In this example, we map 'Cost' to the Cost attribute.



4. Select the result column that contains data to be modified and click the **Transform** button.

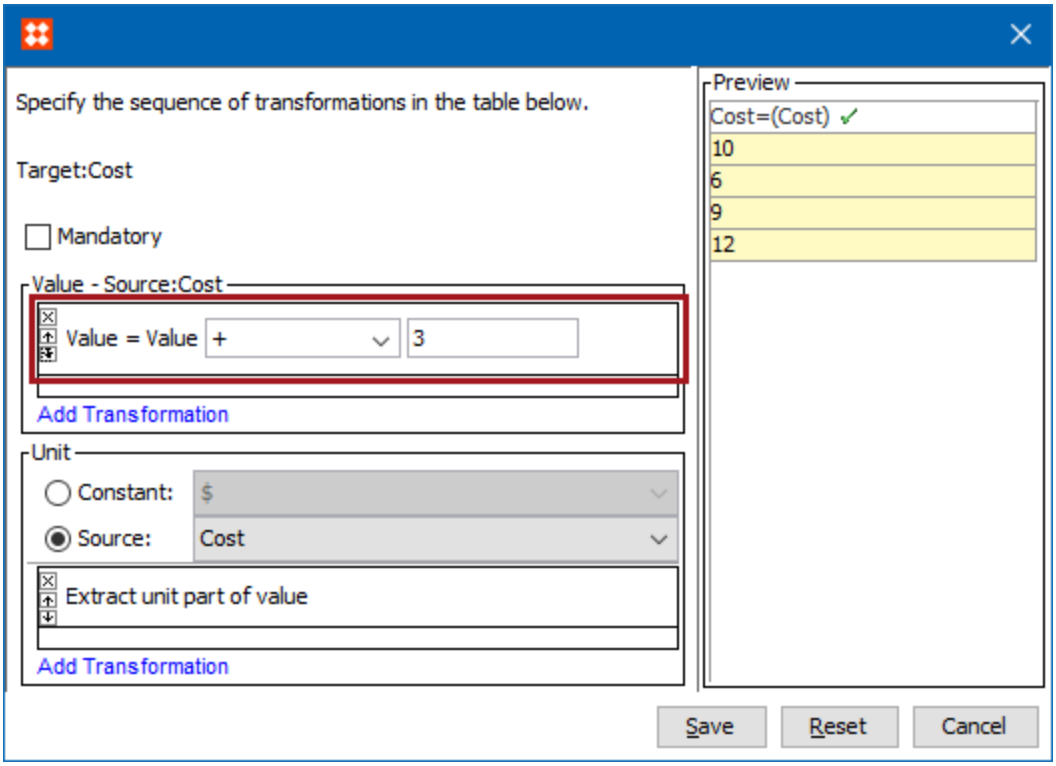


5. In the Transformation window, click on the **Add Transformation** link, select the **Math operation (+,-,*,/)** transformation, and click **OK**.



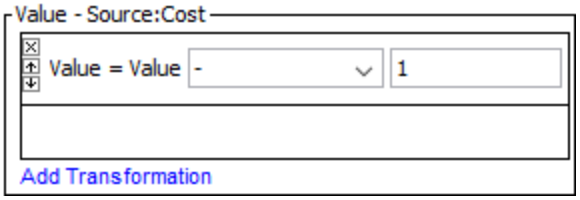
6. In the **Value** parameter, use the dropdown to choose the **+** math operation, and in the text field, enter the number that should be added to the inbound value. In this example, 3 is added to the value.

   Click or tab out of the parameter to complete the configuration. The Preview pane shows the results of the transformation.



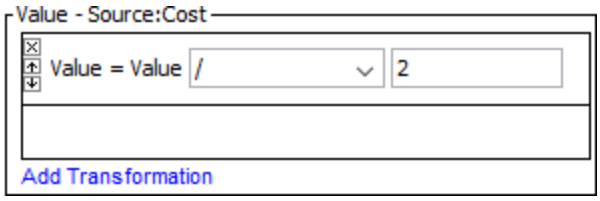
7. Click **Save** to display the transformed data in the Map Data step.
8. To illustrate each math operation, three additional attributes for Cost are used: Cost1, Cost2, and Cost3. Repeat steps 3 - 6, selecting one of the additional Cost attributes each time, confirming to map the same column again, 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.

Map Data

Source:

<ID>	Cost
107601	5
20803	1
20805	4
109011	7

Result: Map to: Product

ID=<ID> ✓	Cost=(Cost) ✓	Cost1=(Cost) ✓	Cost2=(Cost) ✓	Cost3=(Cost) ✓
107601	8	4	10	2.5
20803	4	0	2	0.5
20805	7	3	8	2
109011	10	6	14	3.5

Auto Map | Map | Constant | Remove | Transform | Generate Profile

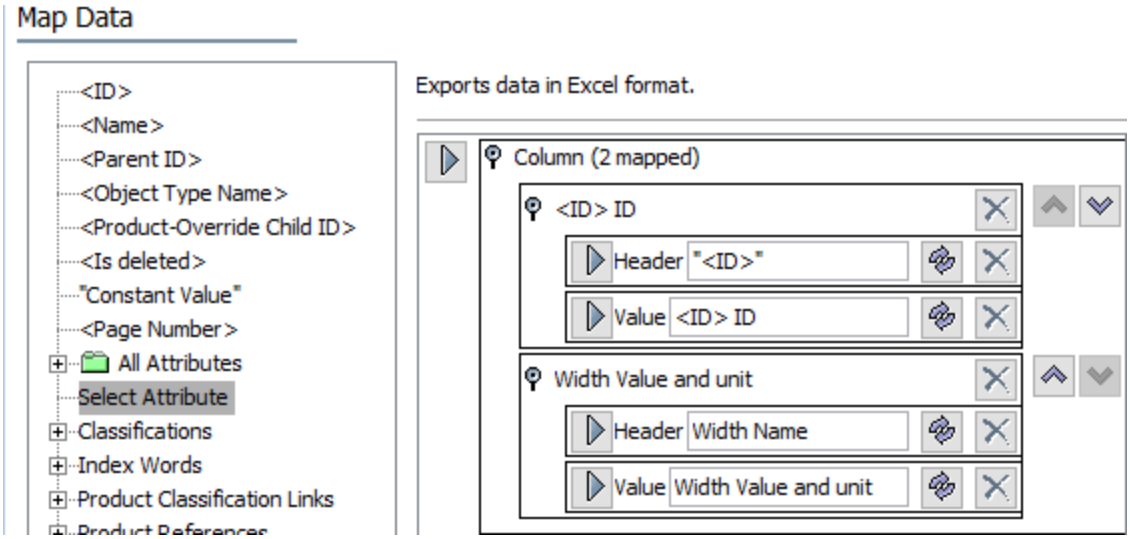
Merge Attributes and Constant Example

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).

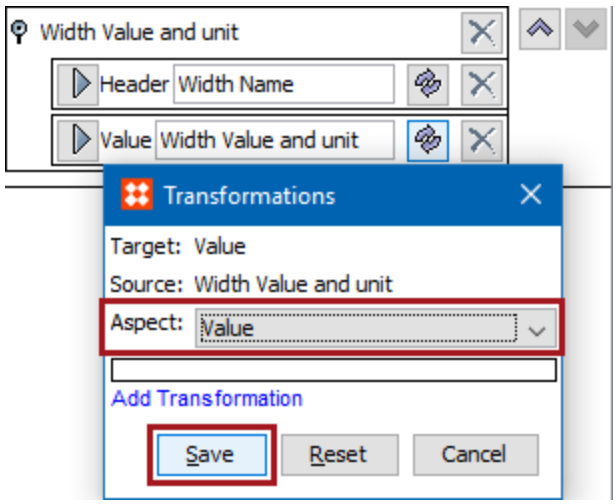
1. Map **<ID>**. See the **ID or Key - Data Source Outbound** topic for details.
2. Map **Select Attribute**, and search for the Width attribute. See the **Attributes and Data Containers - Data Source Outbound** topic for details.

   Two items have been mapped and display in the right panel.

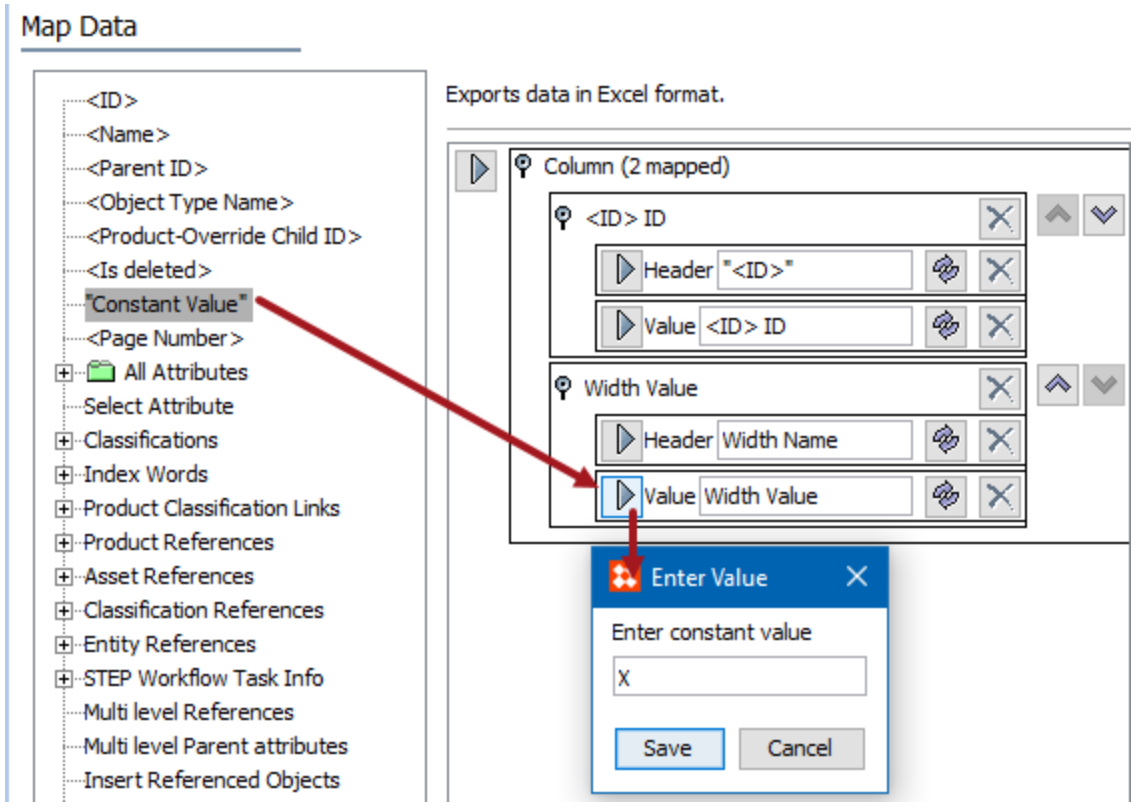


3. Open the Width attribute flipper to display the Header and Value elements.
4. Click the **Transformation** button ( ) for the width attribute Value element to display the Transformations dialog.
5. From the **Aspect** parameter dropdown select **Value**, and click **Save** to apply the change.

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.



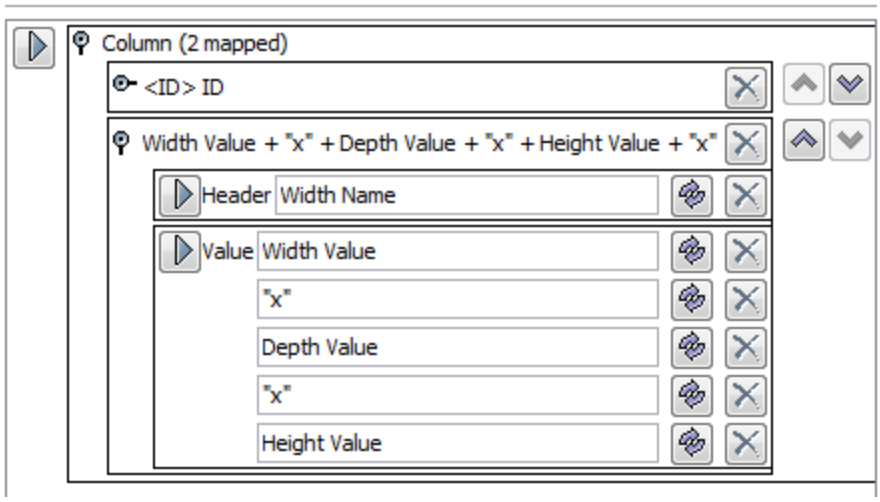
6. Map **Constant Value** to the Value element of your Width attribute in order to merge elements.
7. In the **Enter Value** dialog type **X** and click the **Save** button.



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, see 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, see 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 / 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).

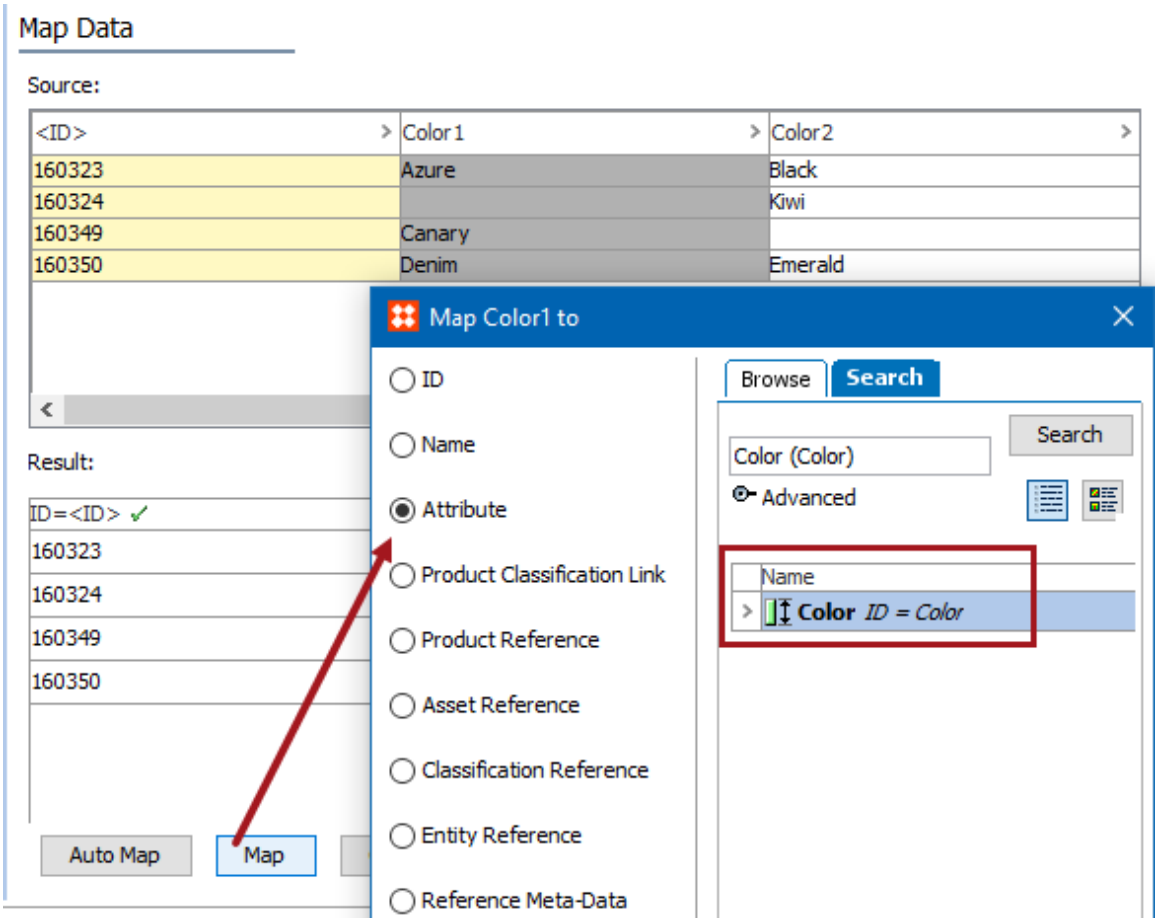
Below are the following examples of merging import data to result in values for a multi-valued attribute:

- **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.

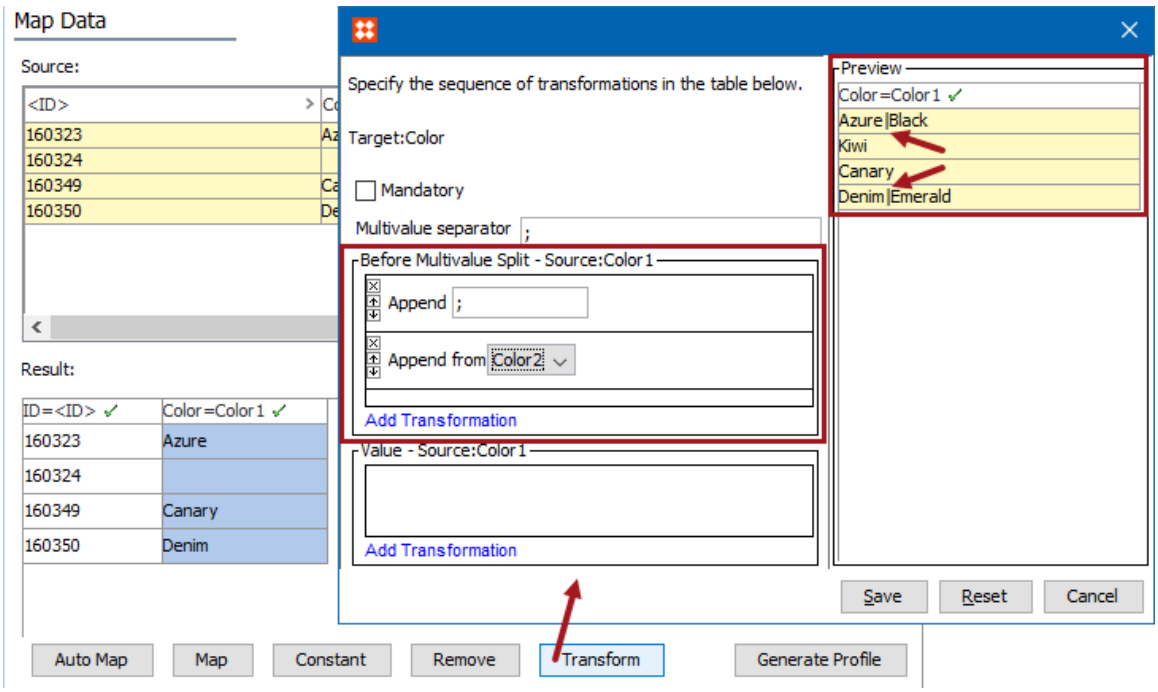
Merge single values from multiple columns

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.

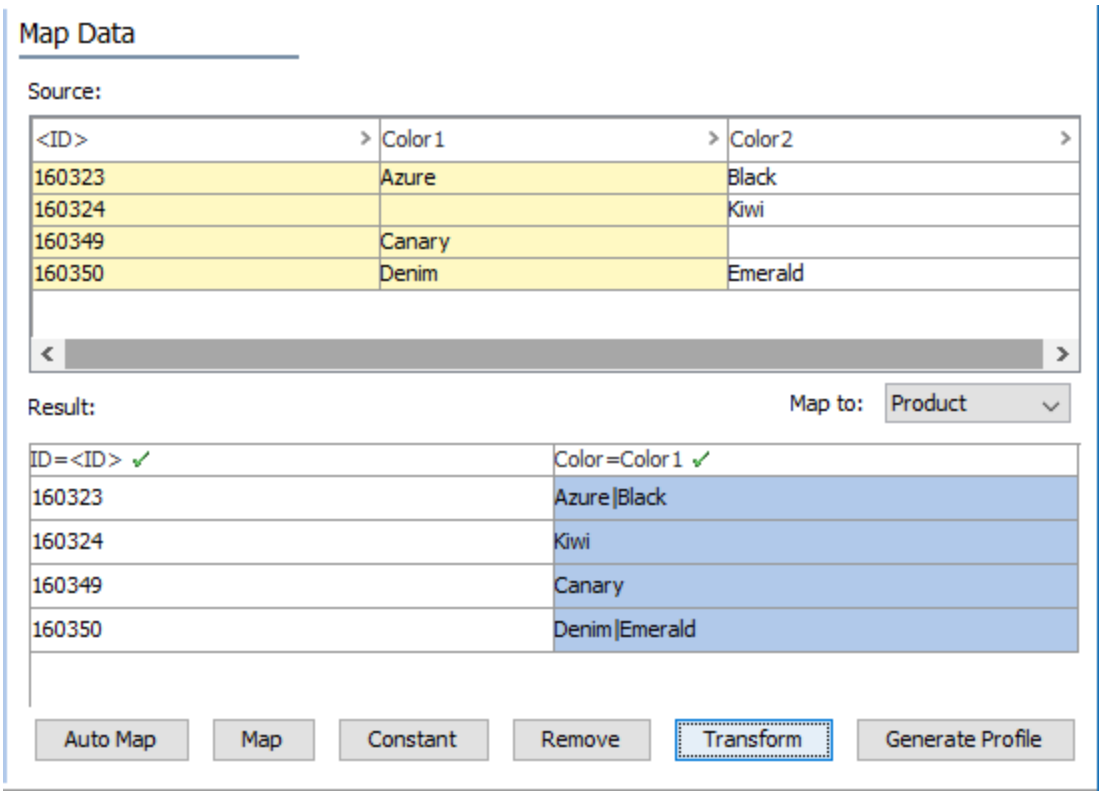


2. In the **Result** area, select the 'Color=Color 1' column and click the **Transform** button to display the transformation dialog.
3. In the **Before Multivalue Split** area, add the **Append text** with a semi-colon (;), and the **Append from source** transformation selecting the 'Color2' column from the dropdown.

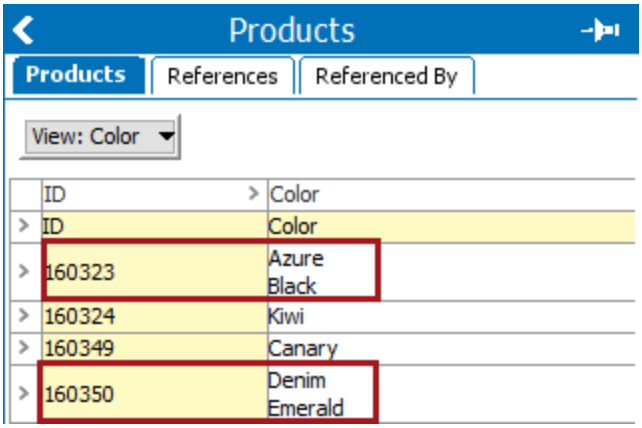


In the Preview area, verify that the rows with multiple values are displayed correctly with a pipe (|) separator. This is required for a successful import of multi-values.

4. Click **Save** to display the result of the transformation in the Result area. Continue navigating through the mappings and rest of the wizard.



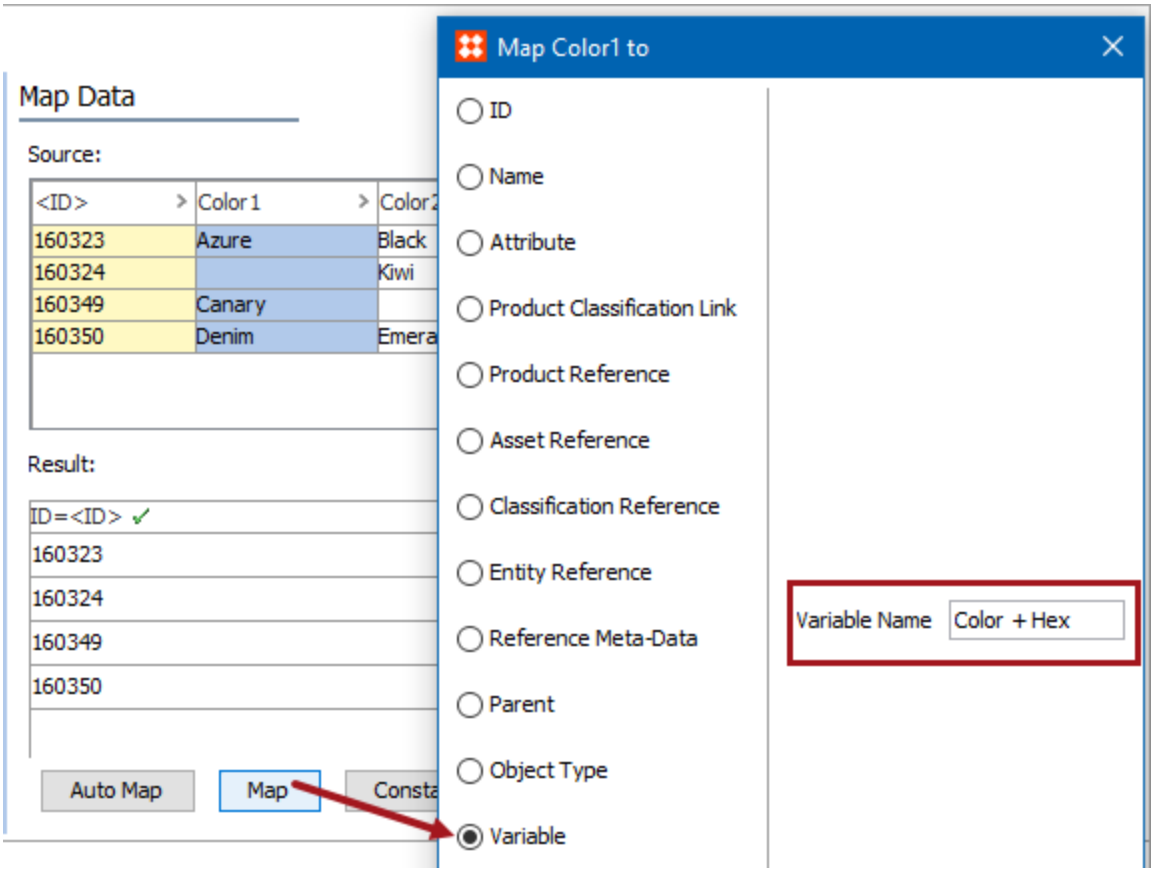
Viewing the products that were modified by the import, shows that the multi-valued Color attribute correctly includes multiple colors when available.



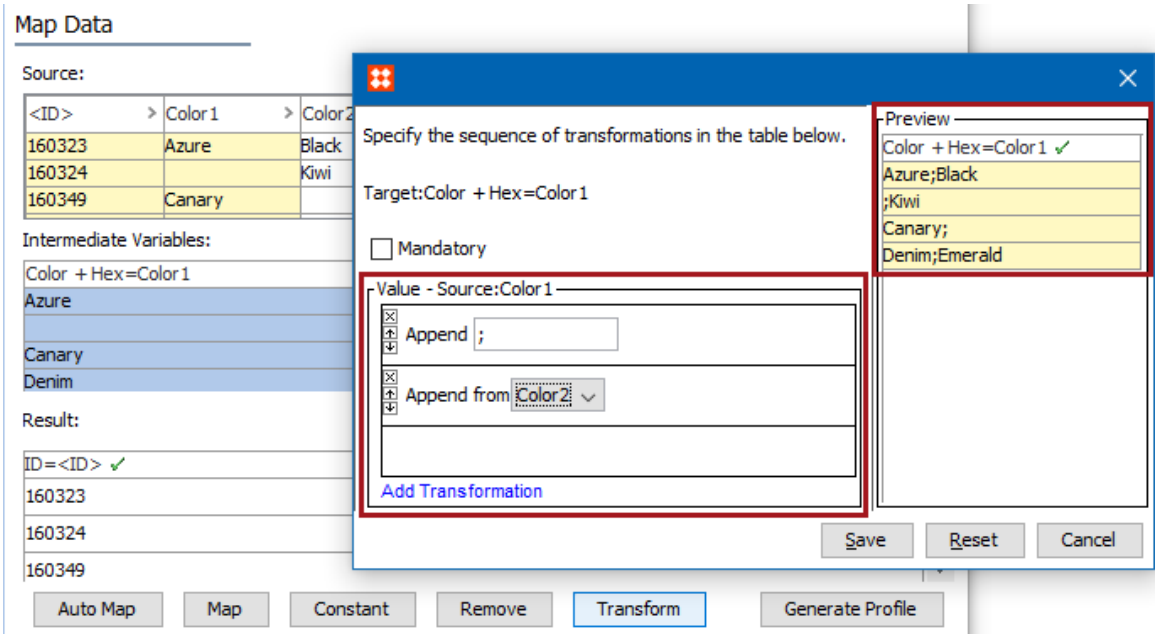
Merge single values from multiple columns using an Intermediate Variable

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.

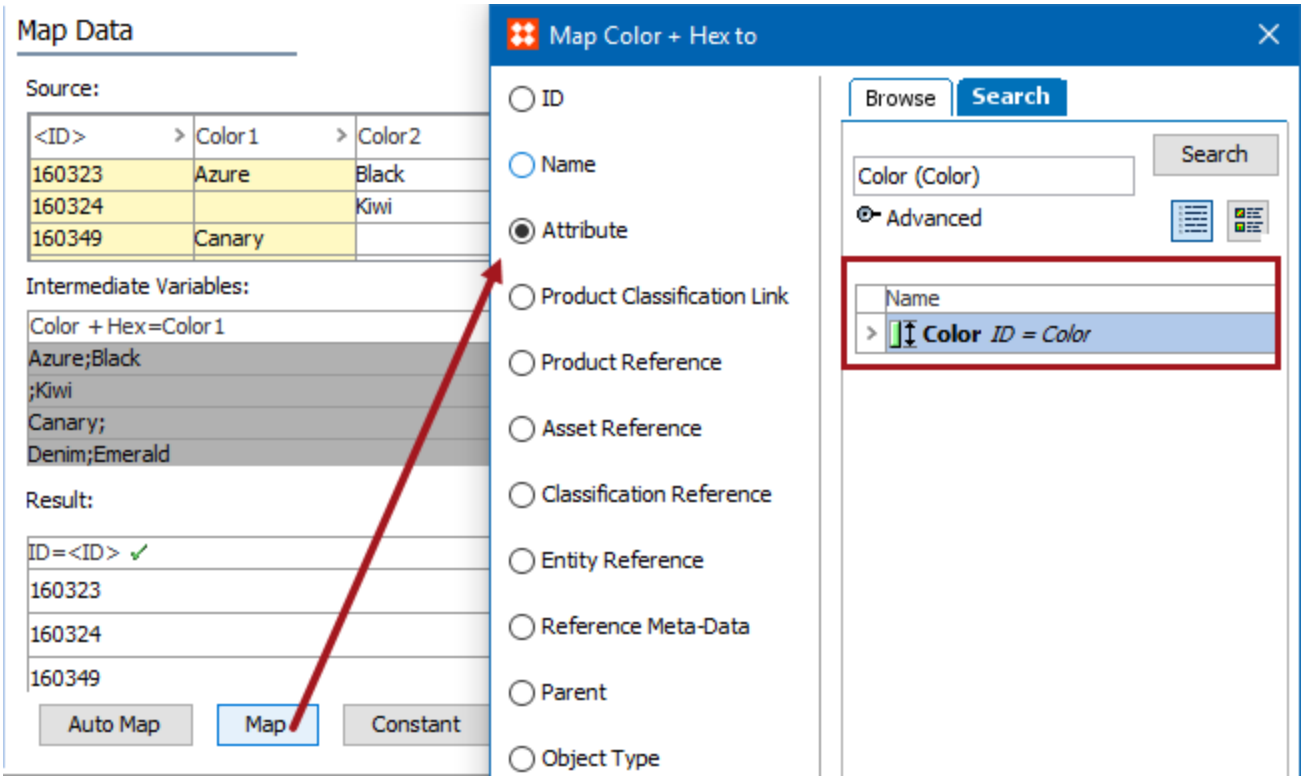


2. In the **Intermediate Variables** area, select the 'Color + Hex =Color 1' column and click the **Transform** button to display the transformation dialog.
3. In the **Value** area, add the **Append text** with a semi-colon (;), and the **Append from source** transformation selecting the 'Color2' column from the dropdown.

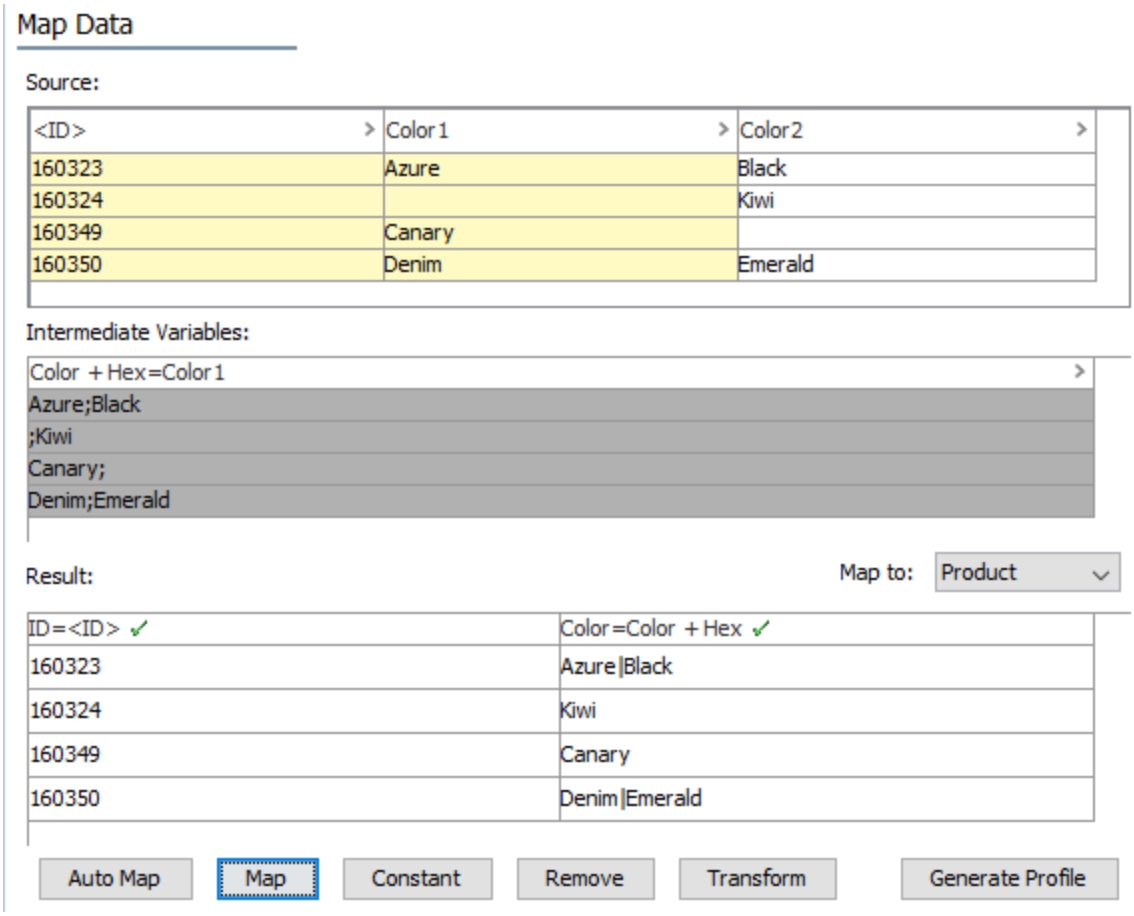


In the Preview area, verify that the rows with multiple values are displayed correctly with a semi-colon (;) separator.

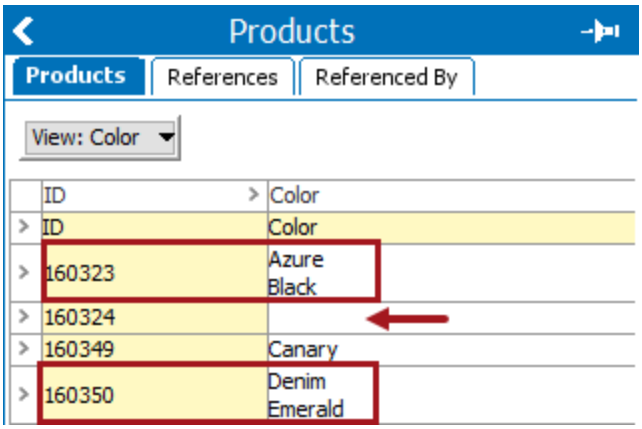
4. Click **Save** to display the result of the transformation in the Intermediate Variables area, select the 'Color + Hex =Color 1' variable column, and map it to the 'Color' attribute.



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.



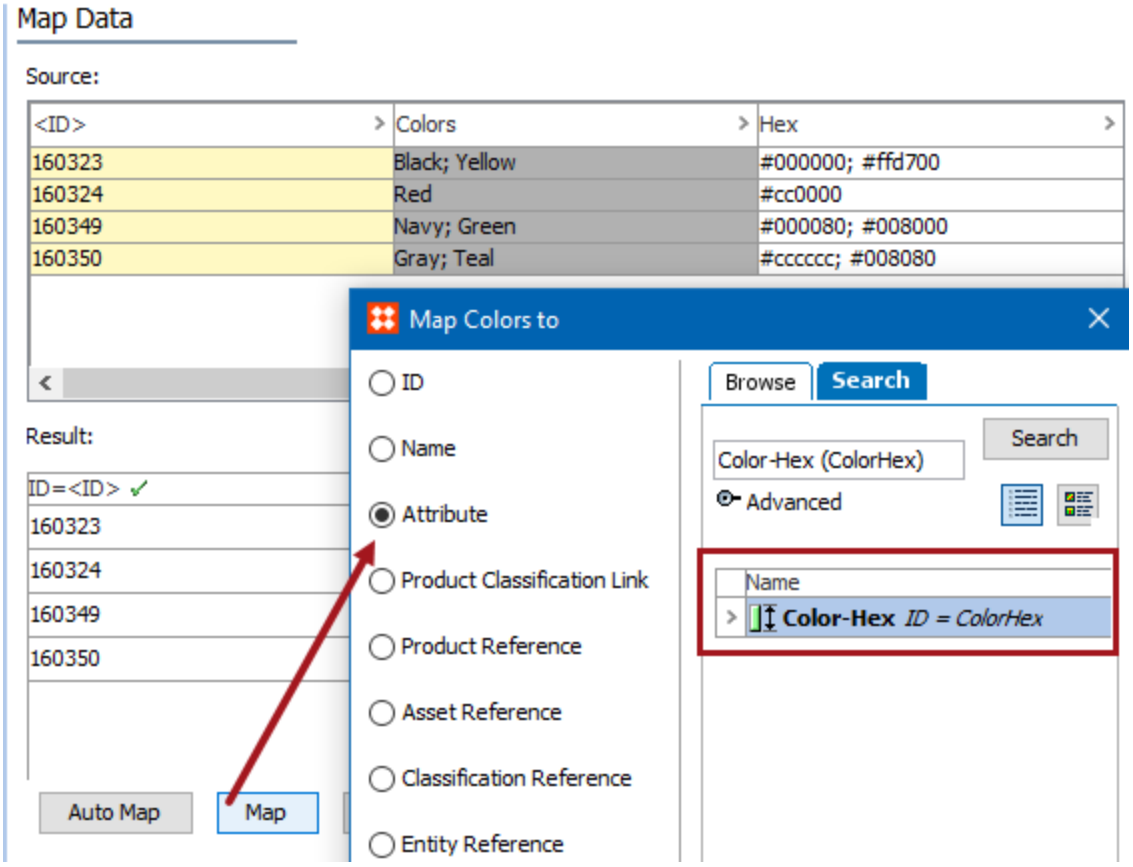
Viewing the products that were modified by the import, shows that the multi-valued Color attribute correctly includes multiple colors when available. However, note that when first column mapped has no value, indicated by the arrow below, that the import does not include the second value.



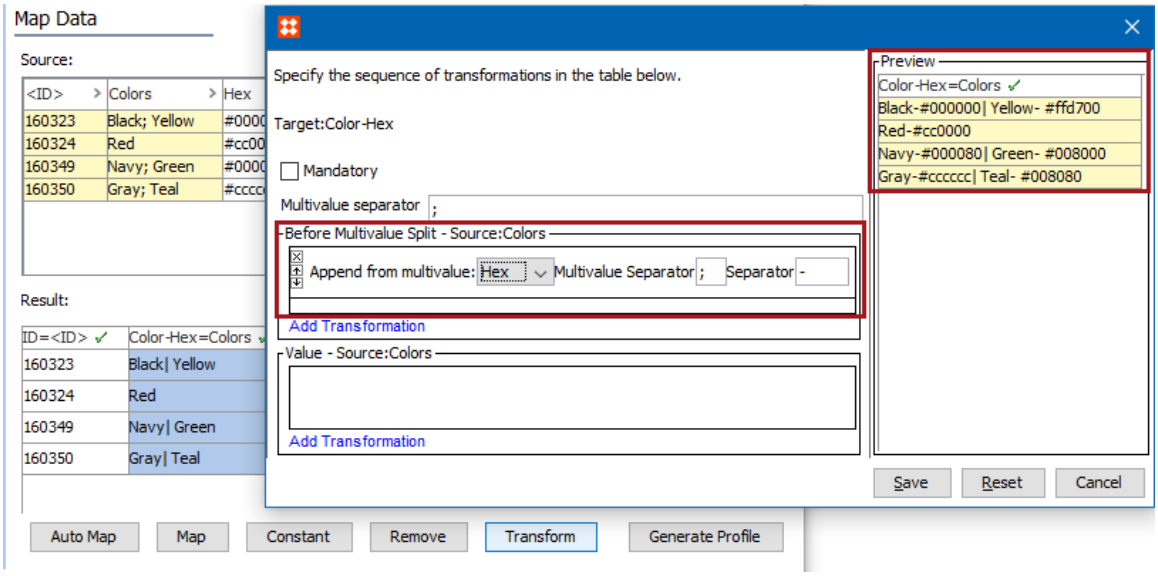
Merge multiple values from multiple columns

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.'

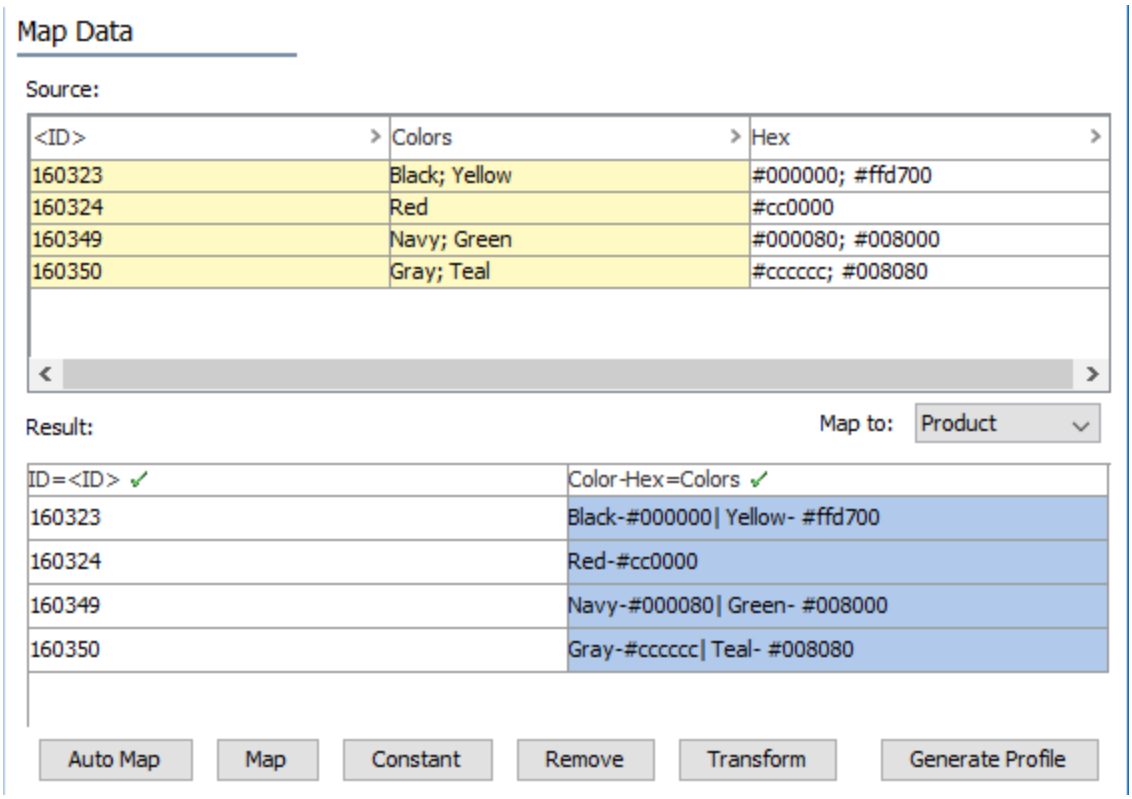


2. In the **Result** area, select the mapped 'Color-Hex=Colors' column, and click the **Transform** button to display the transformation dialog.
3. In the **Before Multivalue Split** area, add the **Append from multivalue source** transformation selecting the 'Hex' column from the dropdown. The separator, '-' in the example, is optional.

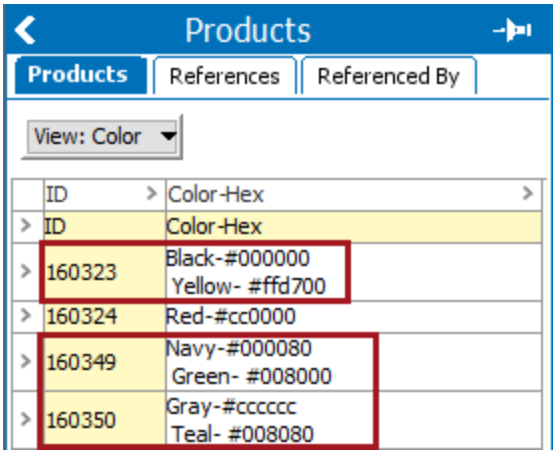


In the Preview area, verify that the rows with multiple values are displayed correctly with a pipe (|) separator. This is required for a successful import of multi-values.

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.

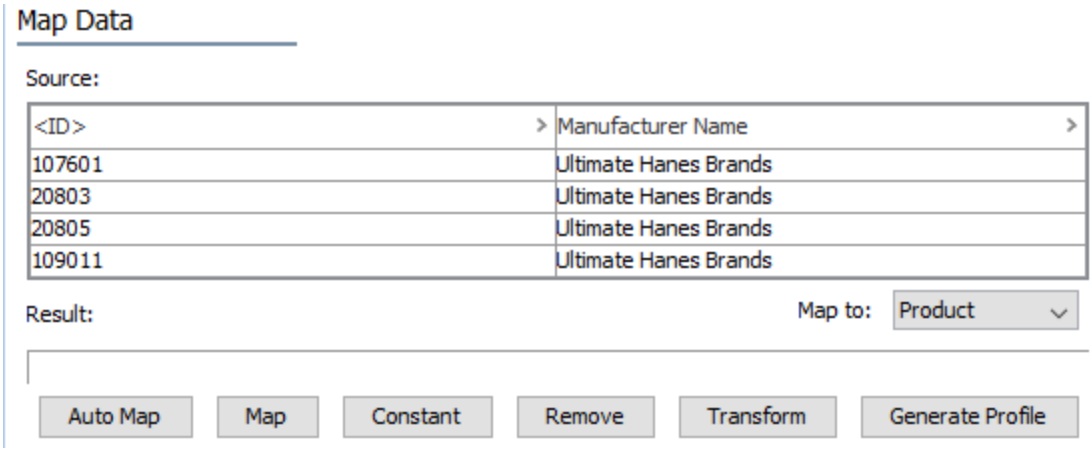


Replace Substrings of the Value Example

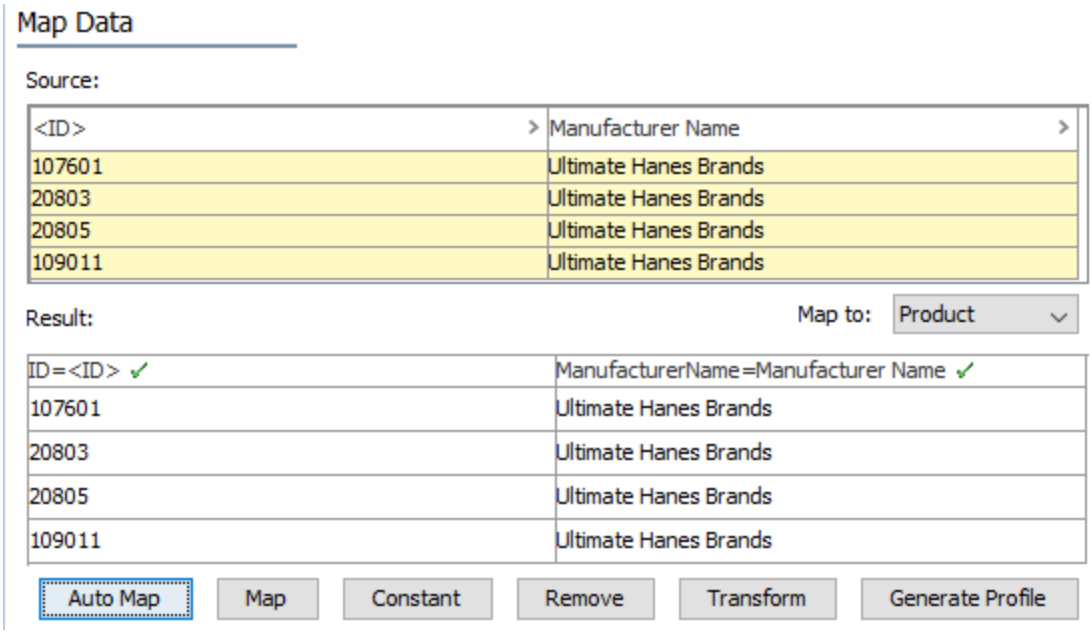
Transformations can improve the quality of the data you want to import. For general information about transformations, see the **Inbound Map Data - Transform** section.

The **Replace substrings of the value** transformation can be used to find and replace all matches of a specified text within a value.

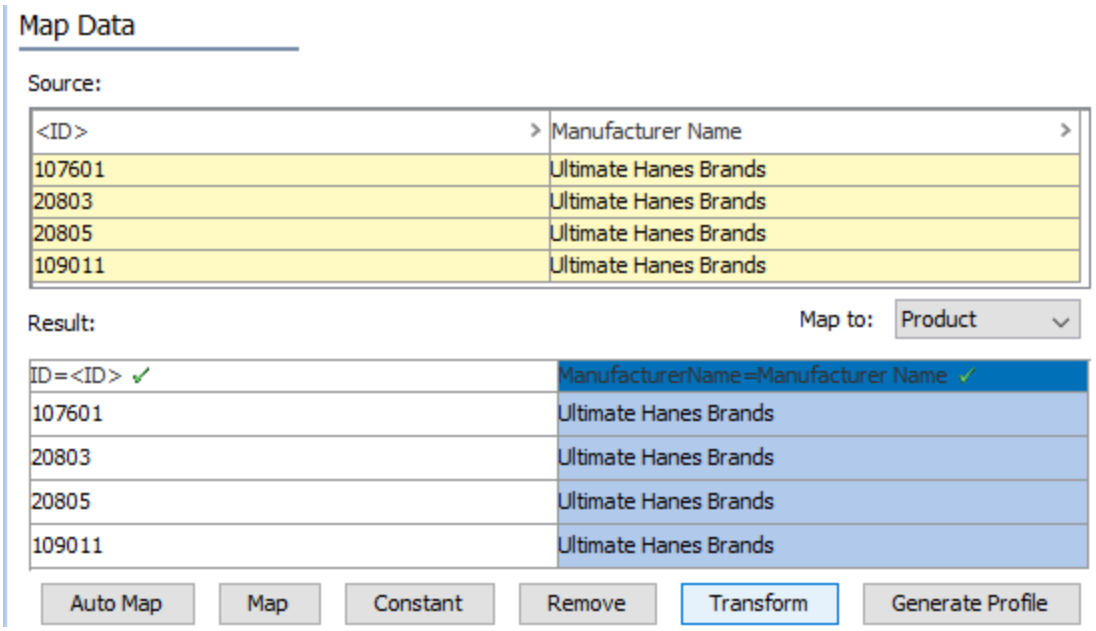
This example describes how text within a field in an import file is modified to be loaded into an attribute in STEP. The following data is being imported, and contains outdated data for the 'Manufacturer Name' attribute:



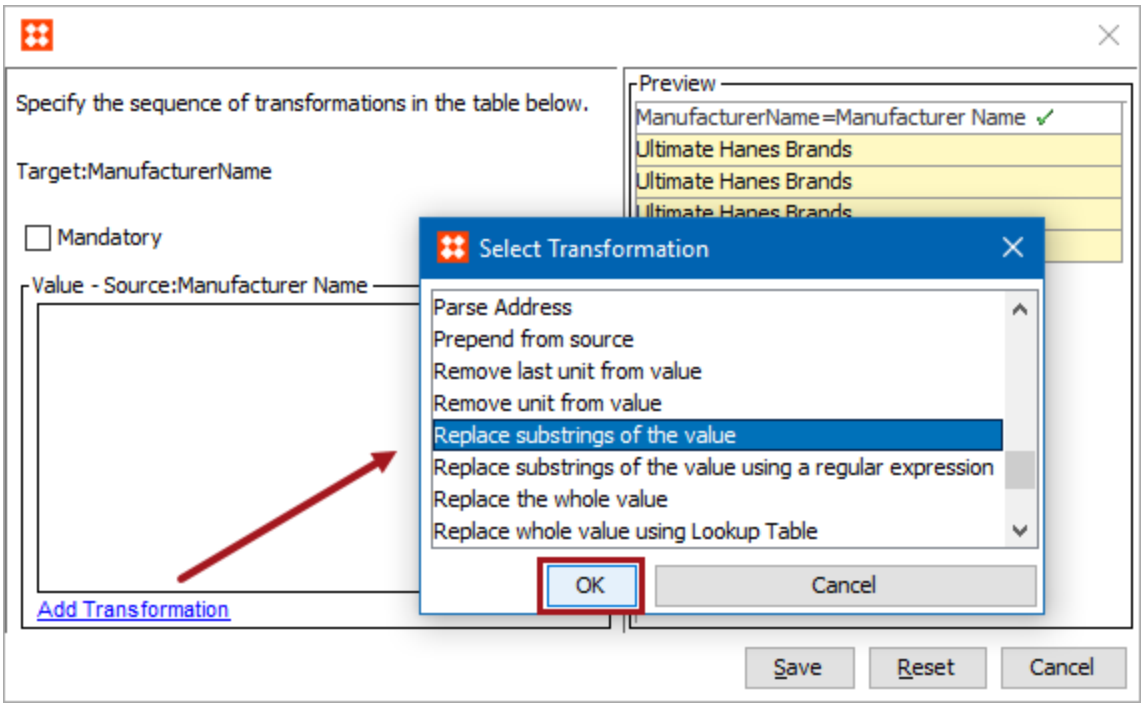
1. Load the import file into the inbound tool. For more information, see the **Data Exchange** topic.
2. Map the object ID in the import tool. See **ID or Key - Map Inbound** for details.
3. Map the next source column to the required attribute. In this example, we map 'Manufacturer Name' to the Manufacturer Name attribute.



4. Select the result column that contains data to be modified and click the **Transform** button.

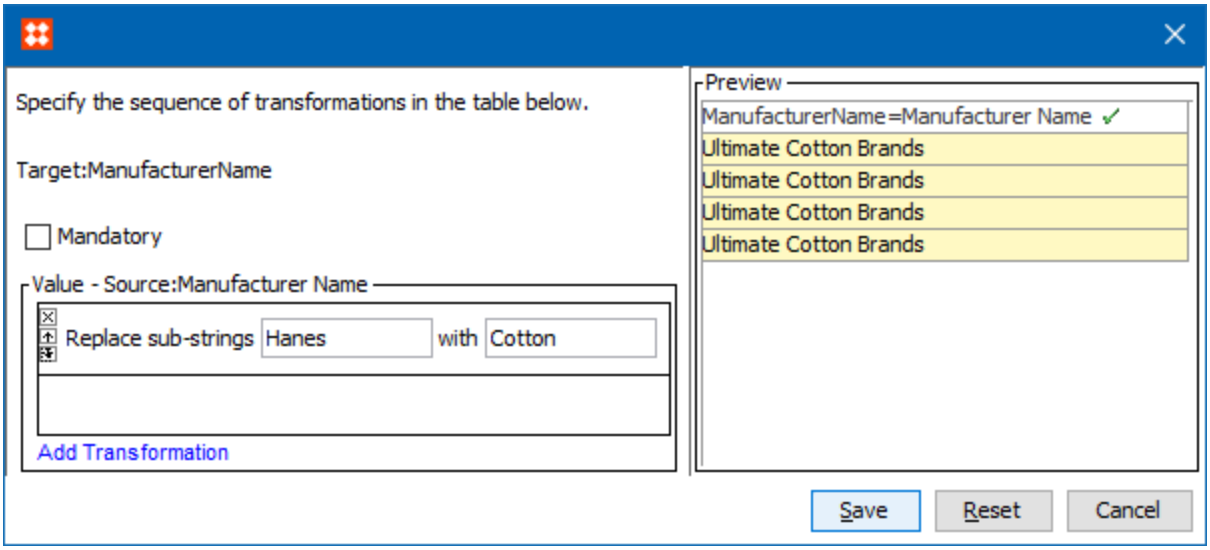


5. In the Transformation window, click on the **Add Transformation** link, select the **Replace substrings of the value** transformation, and click **OK**.



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.

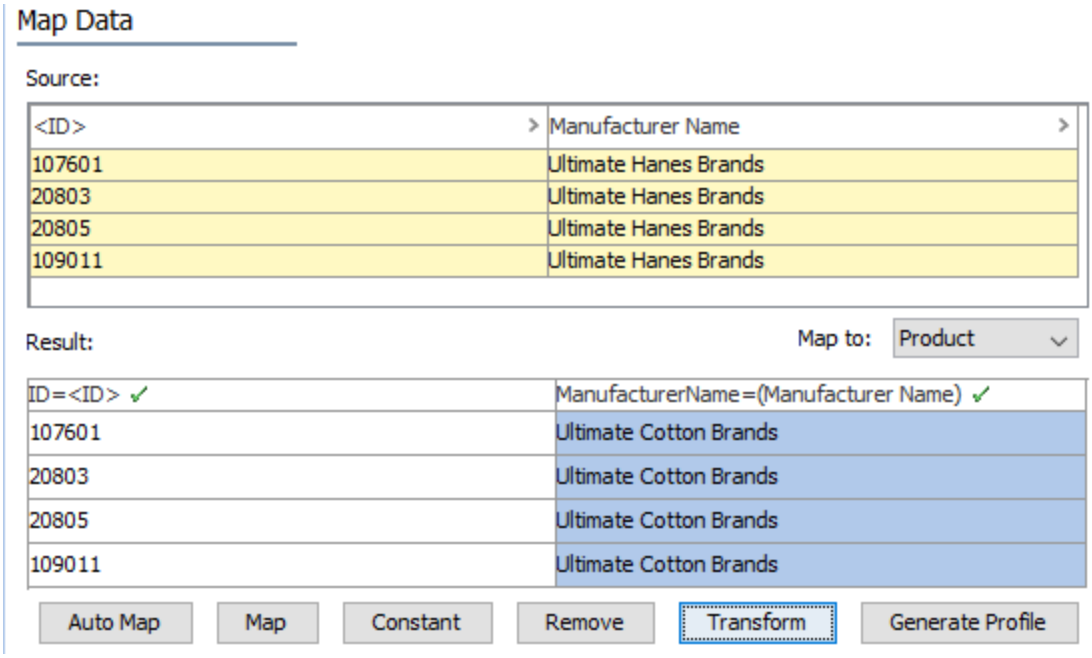
Click or tab out of the parameter to complete the configuration. The Preview pane shows the results of the transformation.



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.

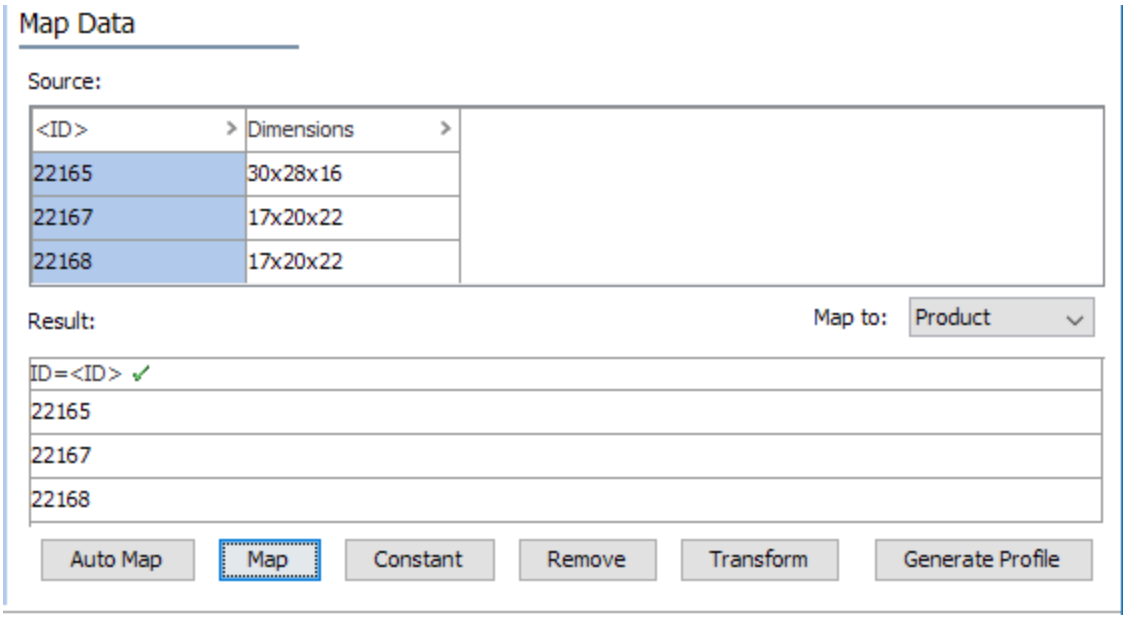


Split and Extract Data Example

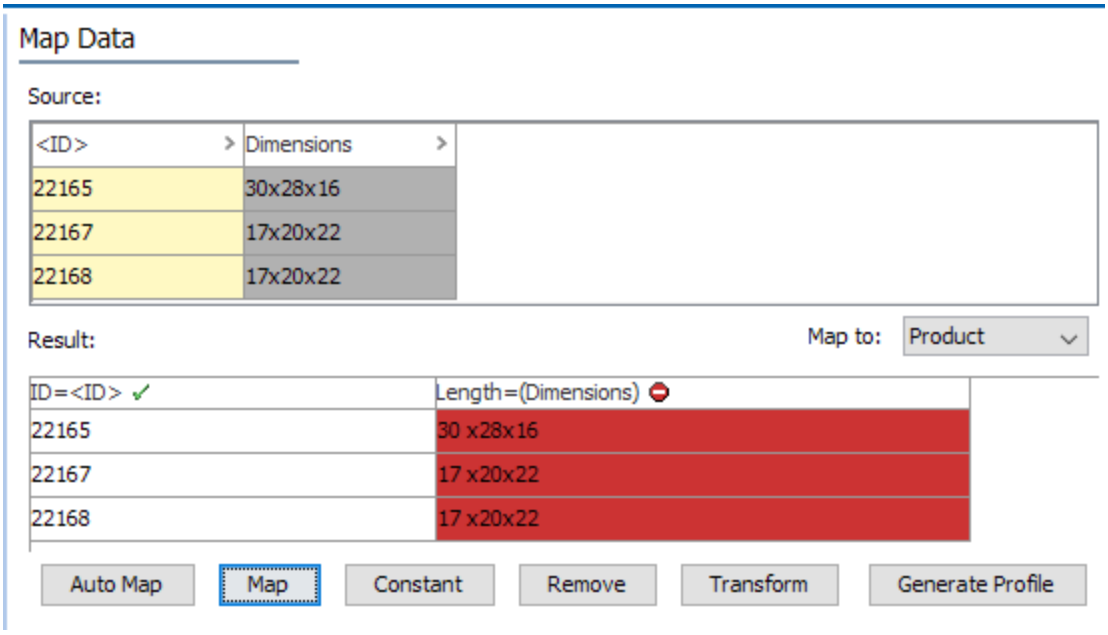
Transformations can improve the quality of the data you want to import. For general information about transformations, see the **Inbound Map Data - Transform** section.

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.

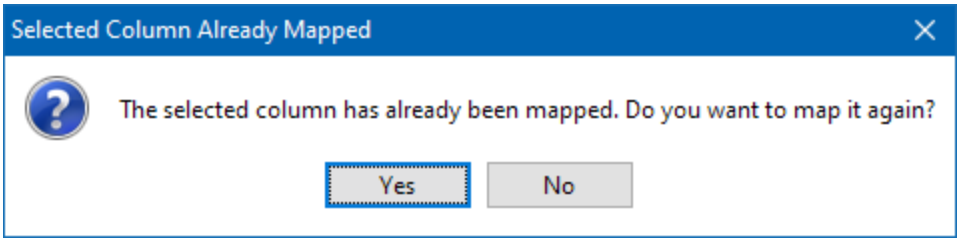
1. Map the product ID in the import tool. Then select the column that holds the dimensional data and begin to map it to the desired three separate attributes in STEP.



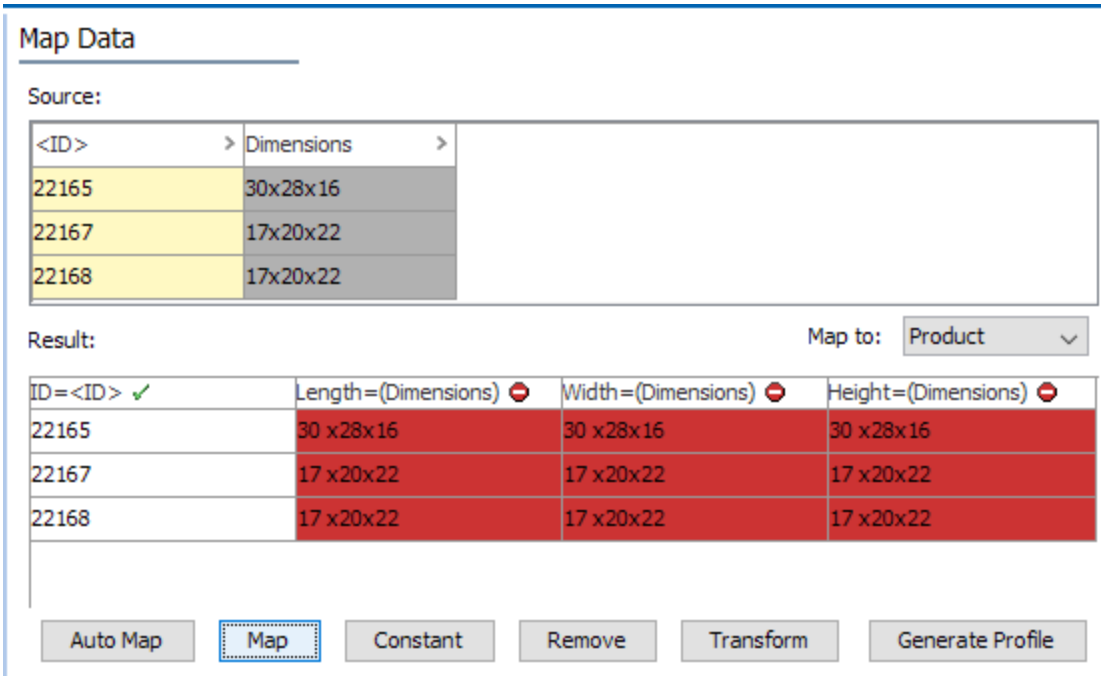
2. Manually map the column to the first attribute that will be populated with some part of the data. In this case, the fist group of numbers are mapped to the attribute 'Length.' Notice that the Results field shows it in red- this is expected until proper transformation is set.



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**.

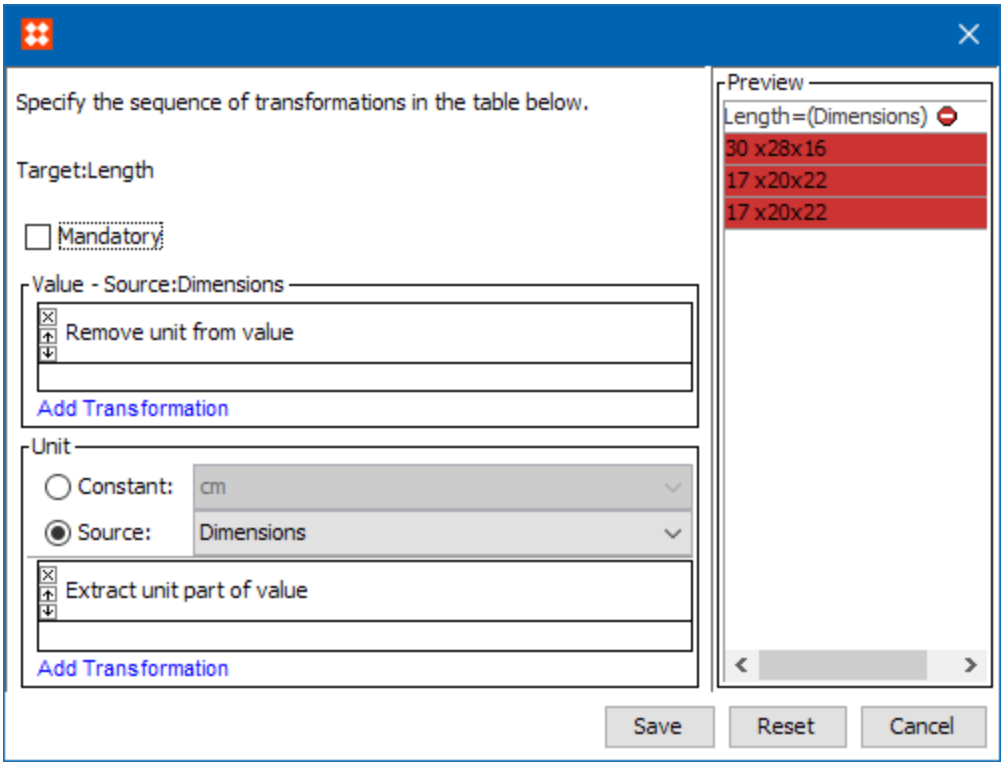


In this example, the same column has been mapped to three different attributes: Length, Width, and Height.

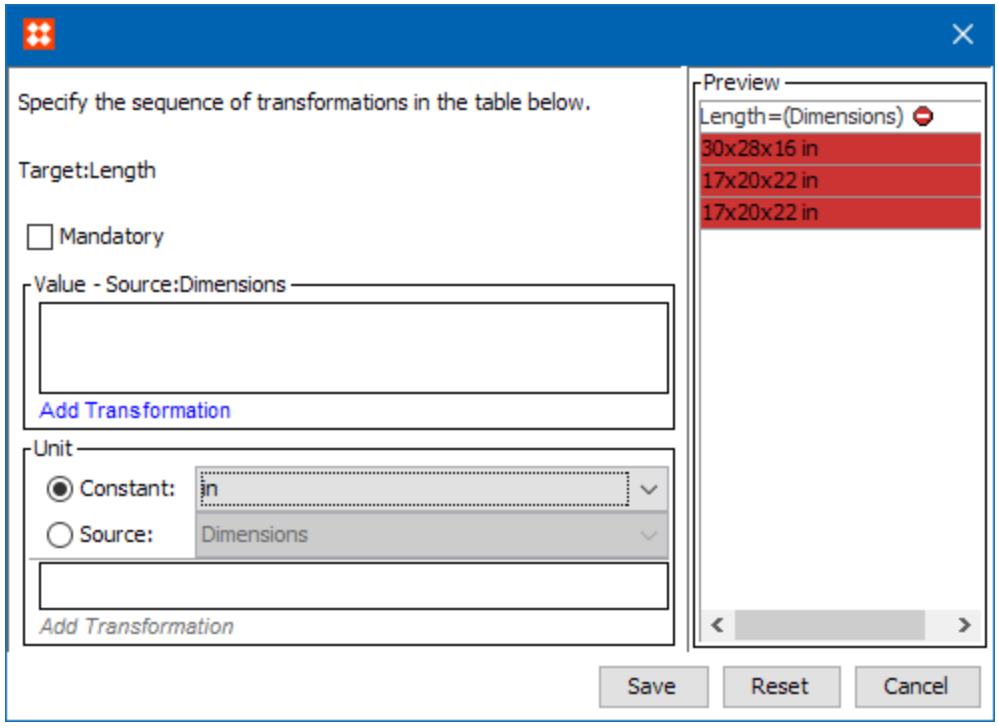


Note that in this particular case, each of the three attributes mapped to have a default unit in centimeters, 'cm.'

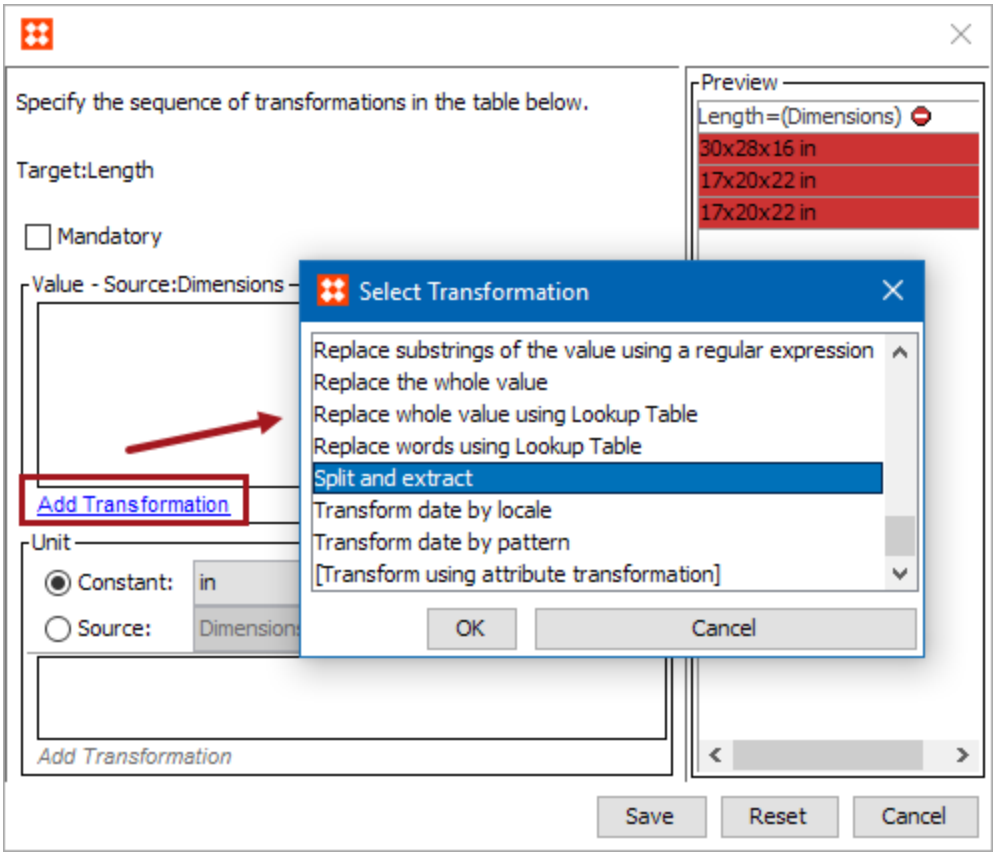
4. In the **Results** pane, select the first attribute that you want to perform a transformation on. In this example that is 'Length.' Click **Transform**. The following window appears.



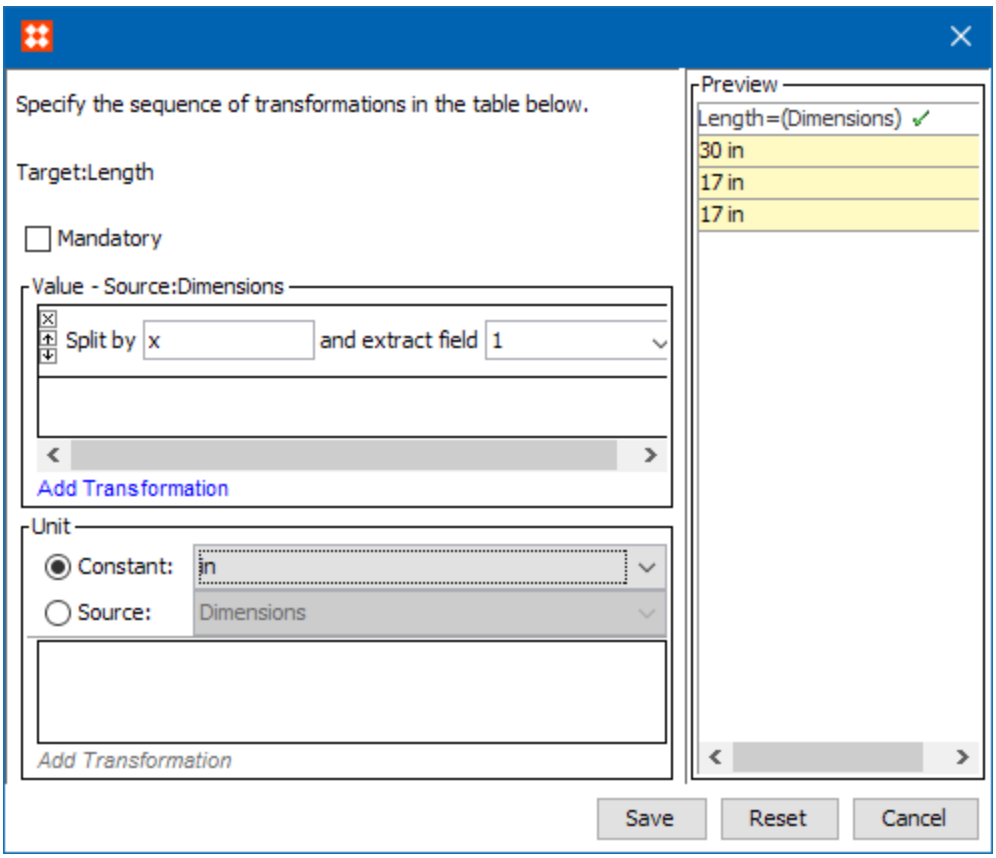
5. In the Unit field, select **Constant**, and choose the desired measurement from the dropdown if there are choices. In this case inches, 'in' are chosen. In the Value-Source: Dimensions field, remove the transformation **Remove unit from value**.



6. Under the Value-Source: Dimensions field, click **Add Transformation**. A list of available transformation functions appears. Select the **Split and extract** transformation.

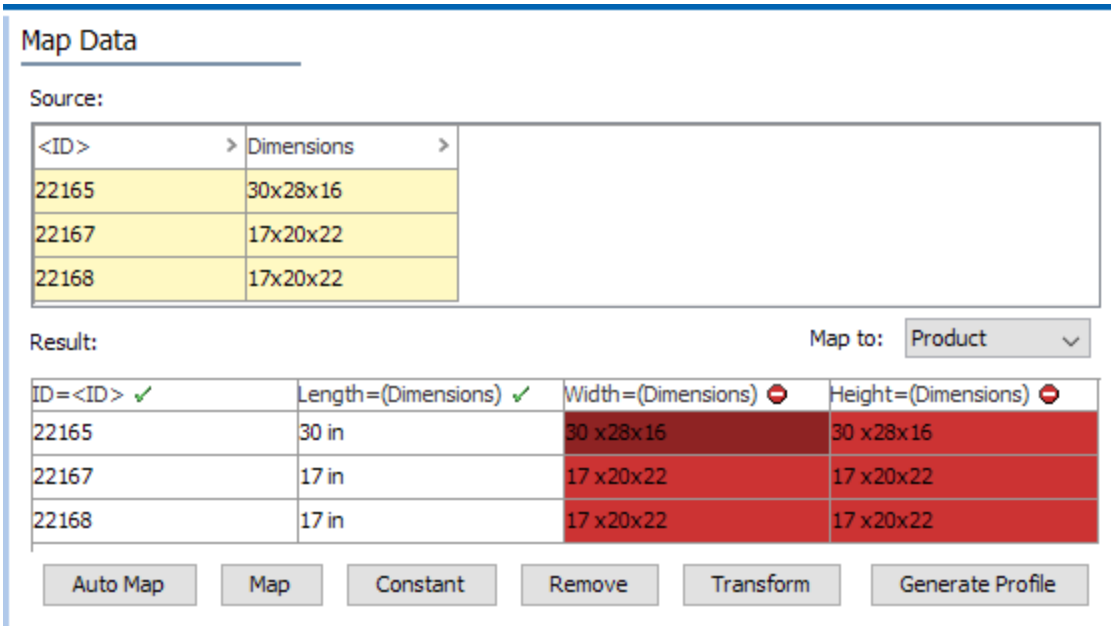


7. In the **Split and extract** transformation, specify how the data in the original file's dimensional data column should be split. In this example, you want to split the data by looking at the information between the " x " (include the spaces), and extracting the data before the first occurrence of 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**.

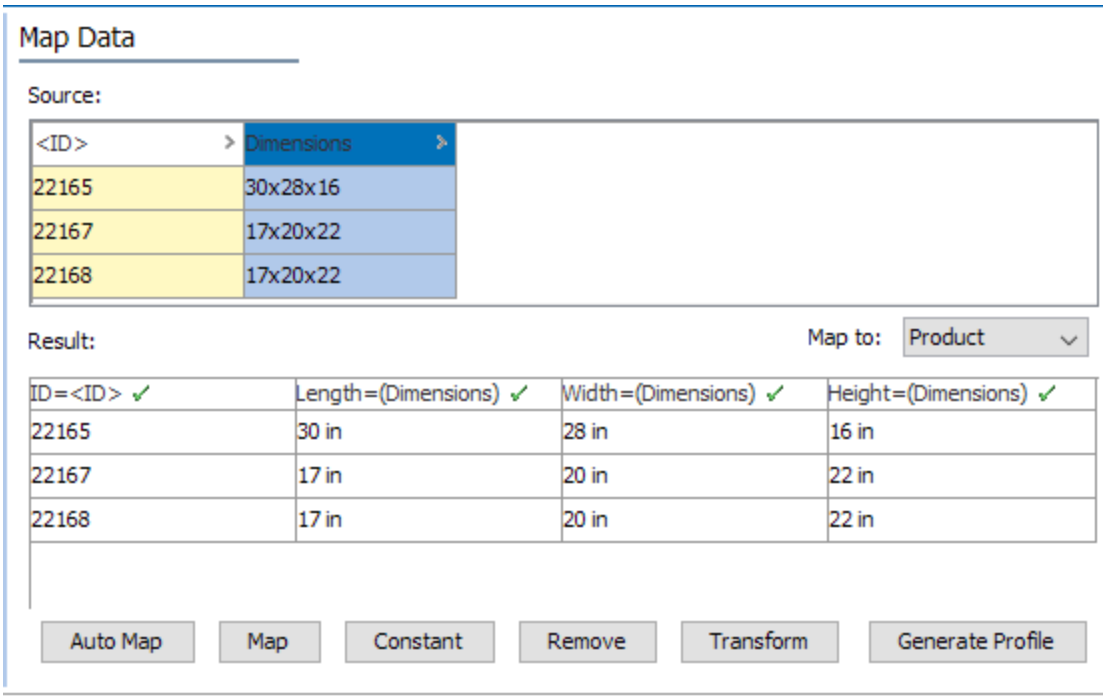


Note: If the 'Mandatory' option is checked, it will skip the empty values, and those objects will not be updated / imported into STEP.

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, see the **Inbound Map Data - Transform** section.

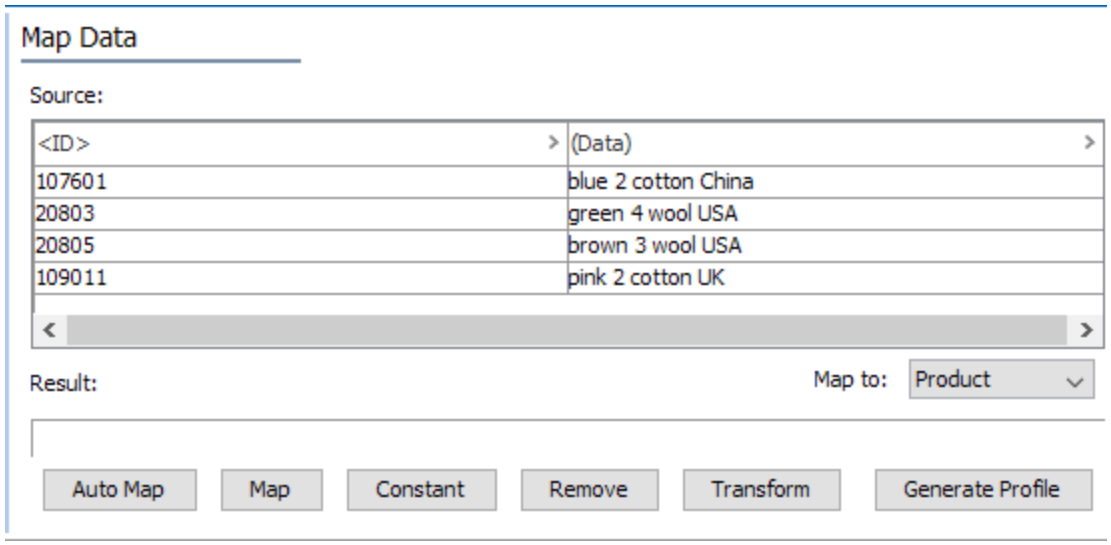
This example describes how to extract and clean information from a legacy system, or groups of information stored in one column, and load it into separate attributes in STEP. If information is more technical or cryptic, see the last section of this topic on how to convert cryptic data.

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.

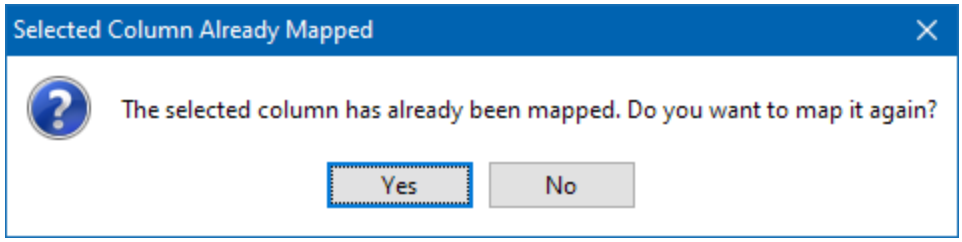


A single import column contains four pieces of data that are to be stored separately in STEP. Using transformations, separate values can be created for the attributes. In this case it is: Primary Color, Hat Size, Material, and Country of Origin. For the first source row, the object should have Color = blue, Hat Size = 2, Material = Cotton, and Country of Origin = CN for China.

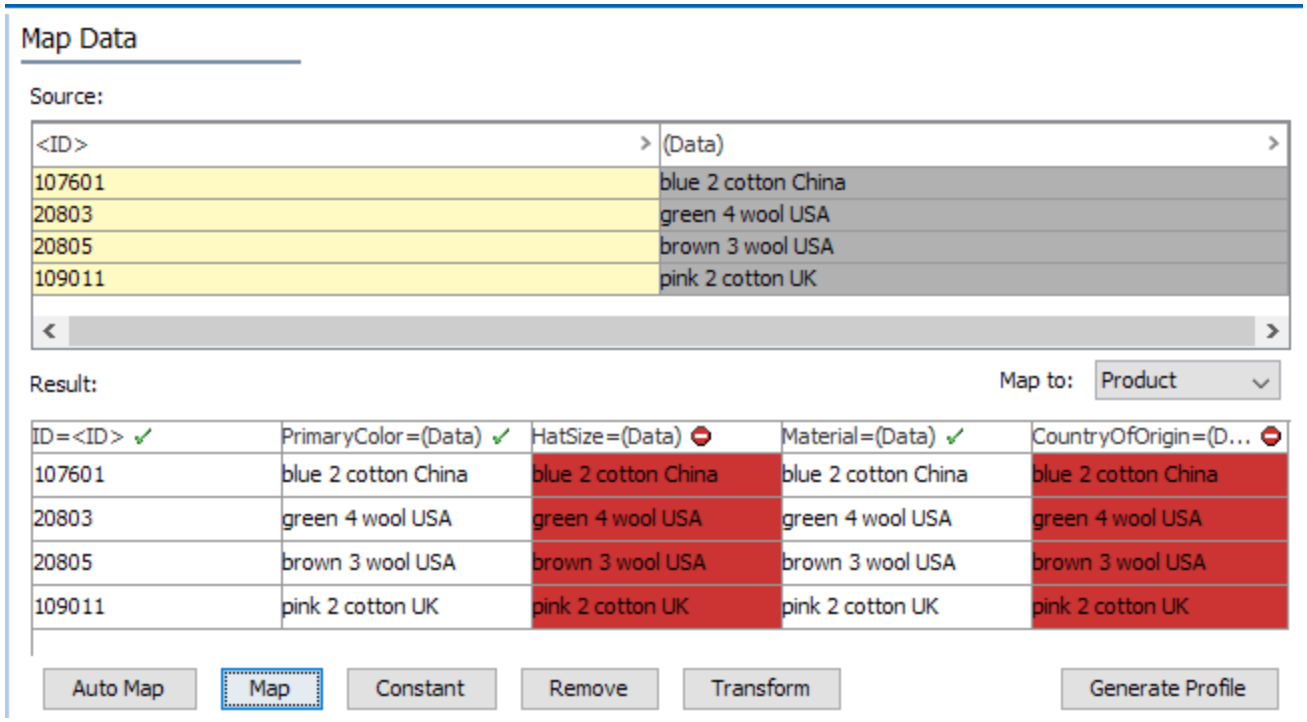
1. In the 'Map Data' step of an import wizard, map the object's ID. Then, map the column that holds the data to the four needed attributes. This means that the same column will be mapped 4 times. For more on how to map data, see the **Import Manager - Map Data** topic or the **IIEP - Configure Processing 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.

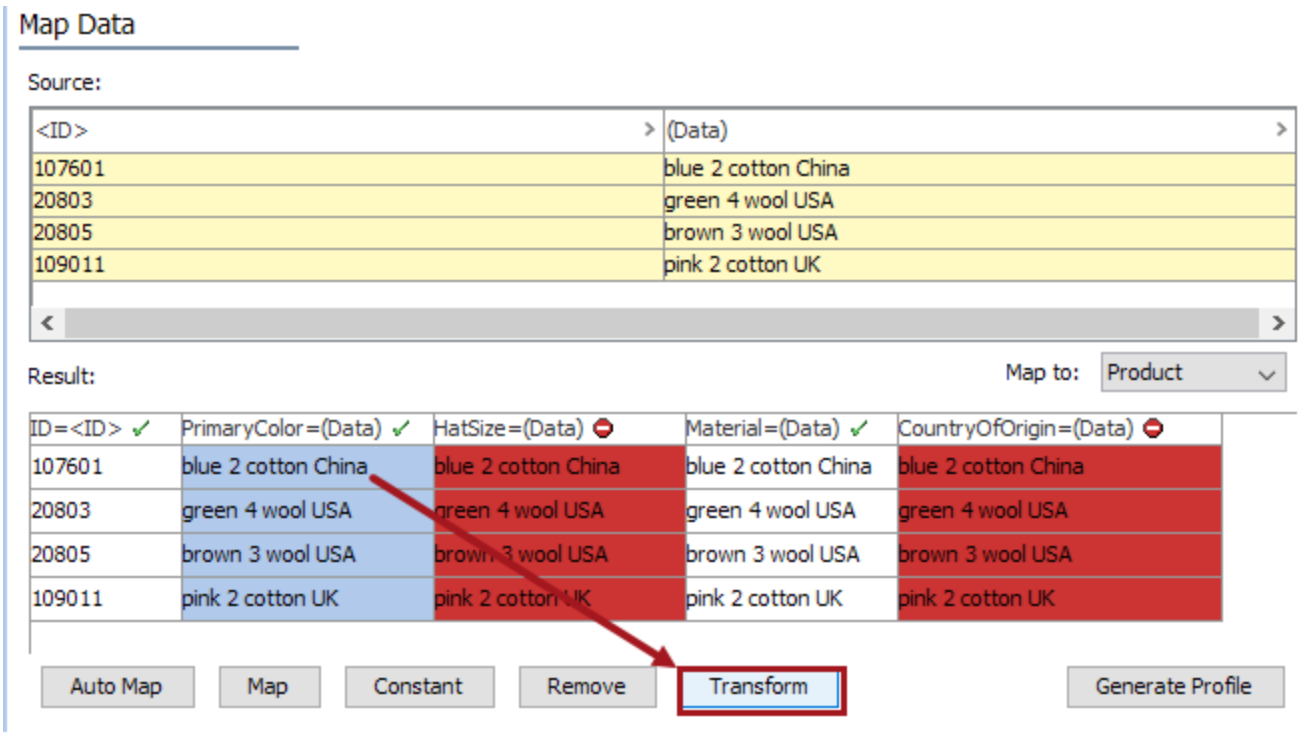


Applying transformations

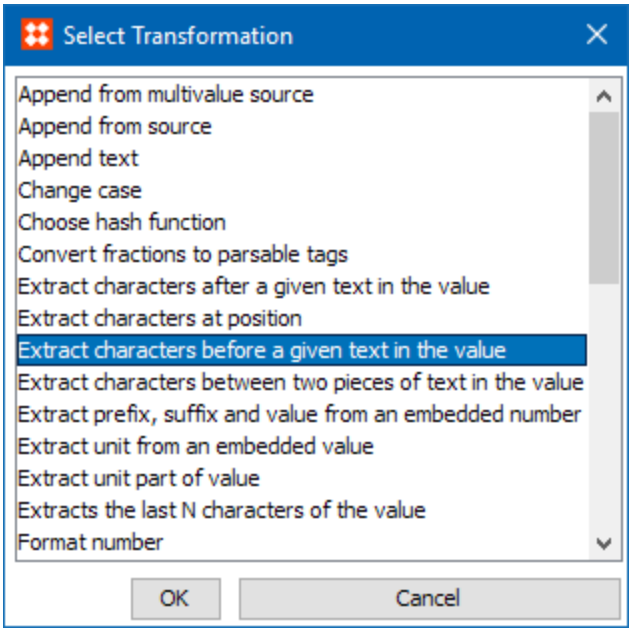
Below are examples of how transformations can be used for each of the columns to clarify the data.

For more on all available transformations, see 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**.

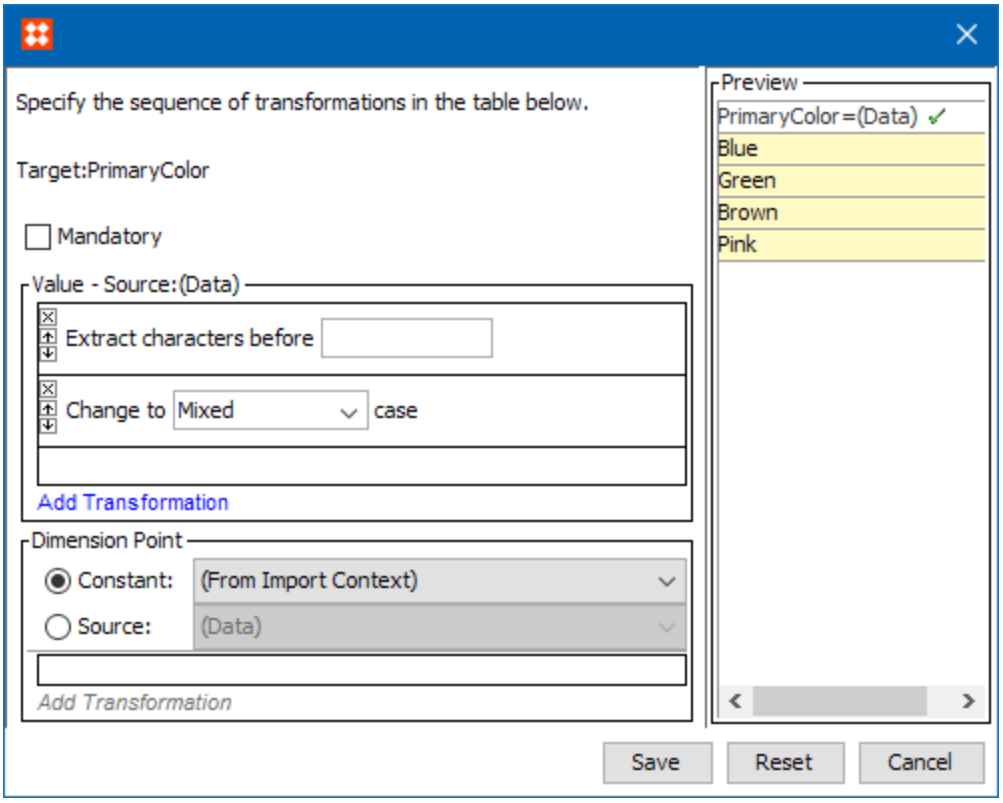


2. In the Transformation window, click on the blue **Add Transformation** link, to open the list of available transformations.



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

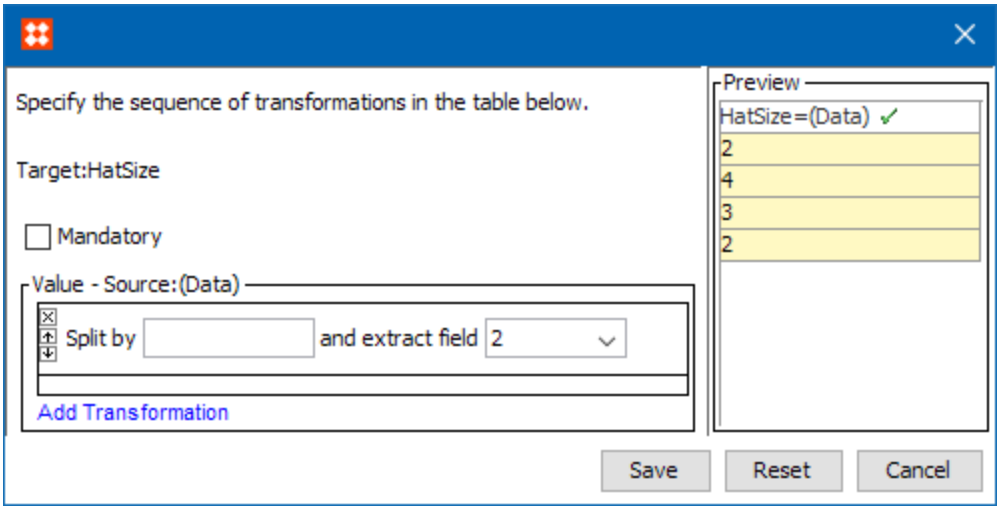
in the picture. Also, the 'Change case' transformation was added. 'Mixed' was chosen from the dropdown so that the first letter of the colors are capitalized and the rest appear lower case.



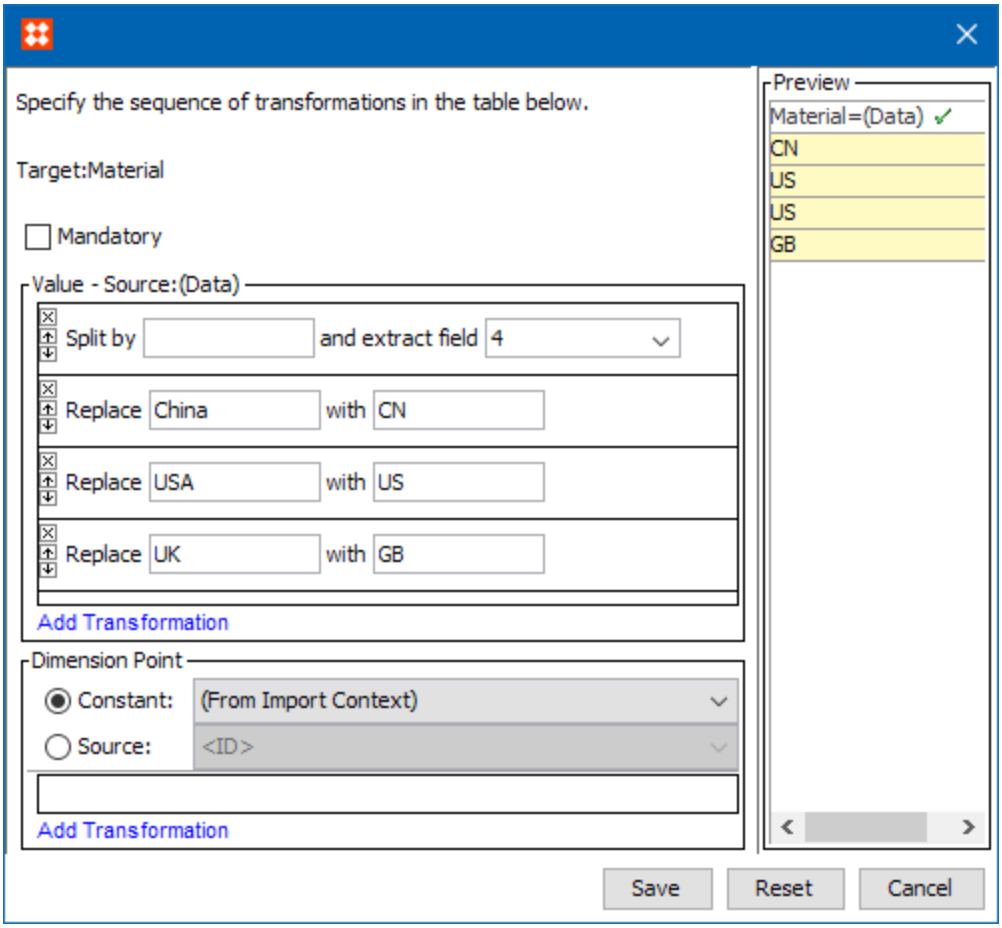
It is important to note that transitions are evaluated from the top down. Careful consideration needs to go into the order of the transformations applied. Each time a new transformation needs to be applied, **Add Transformation** needs to be clicked.

Note: If the 'Mandatory' option is checked, it will skip the empty values, and those objects will not be imported.

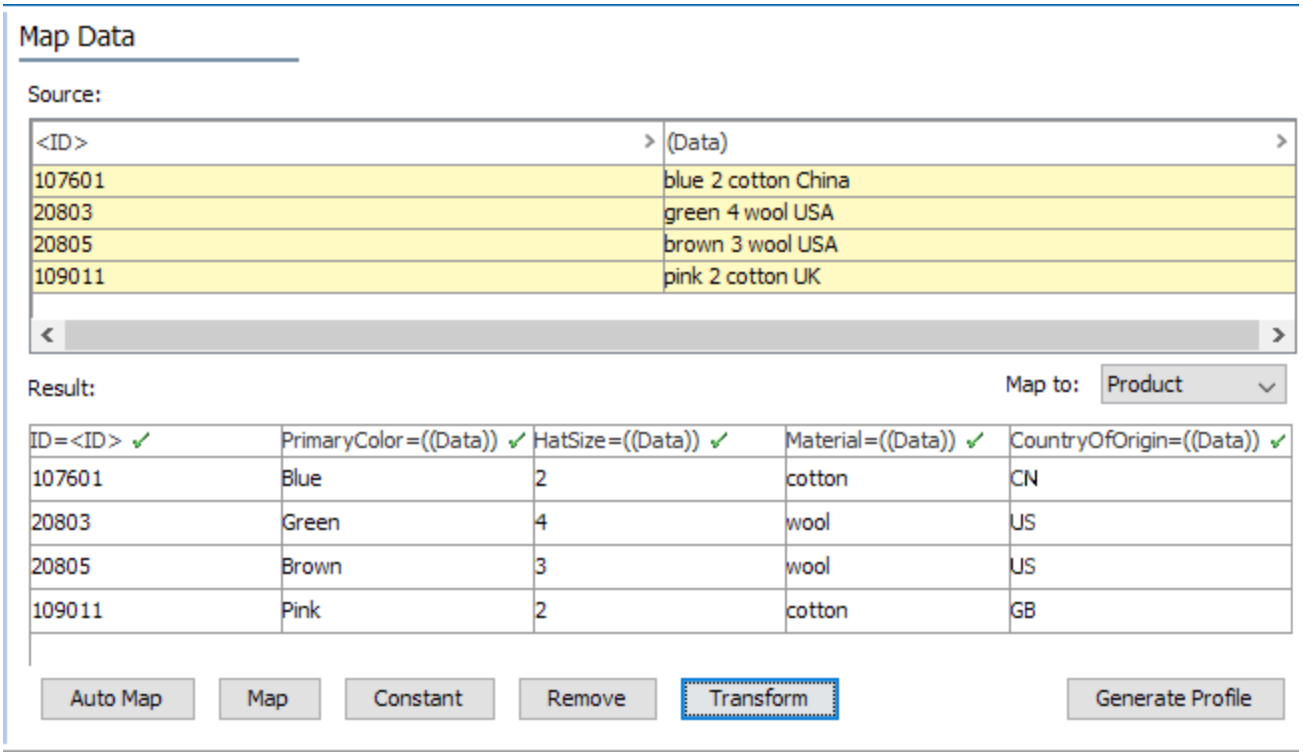
4. Continue with transforming the next column. For example, in the 'HatSize=(Data)' column, the 'Split and extract' transformation can be selected to show the correct numbers. In the first field, a space, which can not be seen in the picture, was added. In the second field, the number 2 was selected 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. Note that you will need to 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

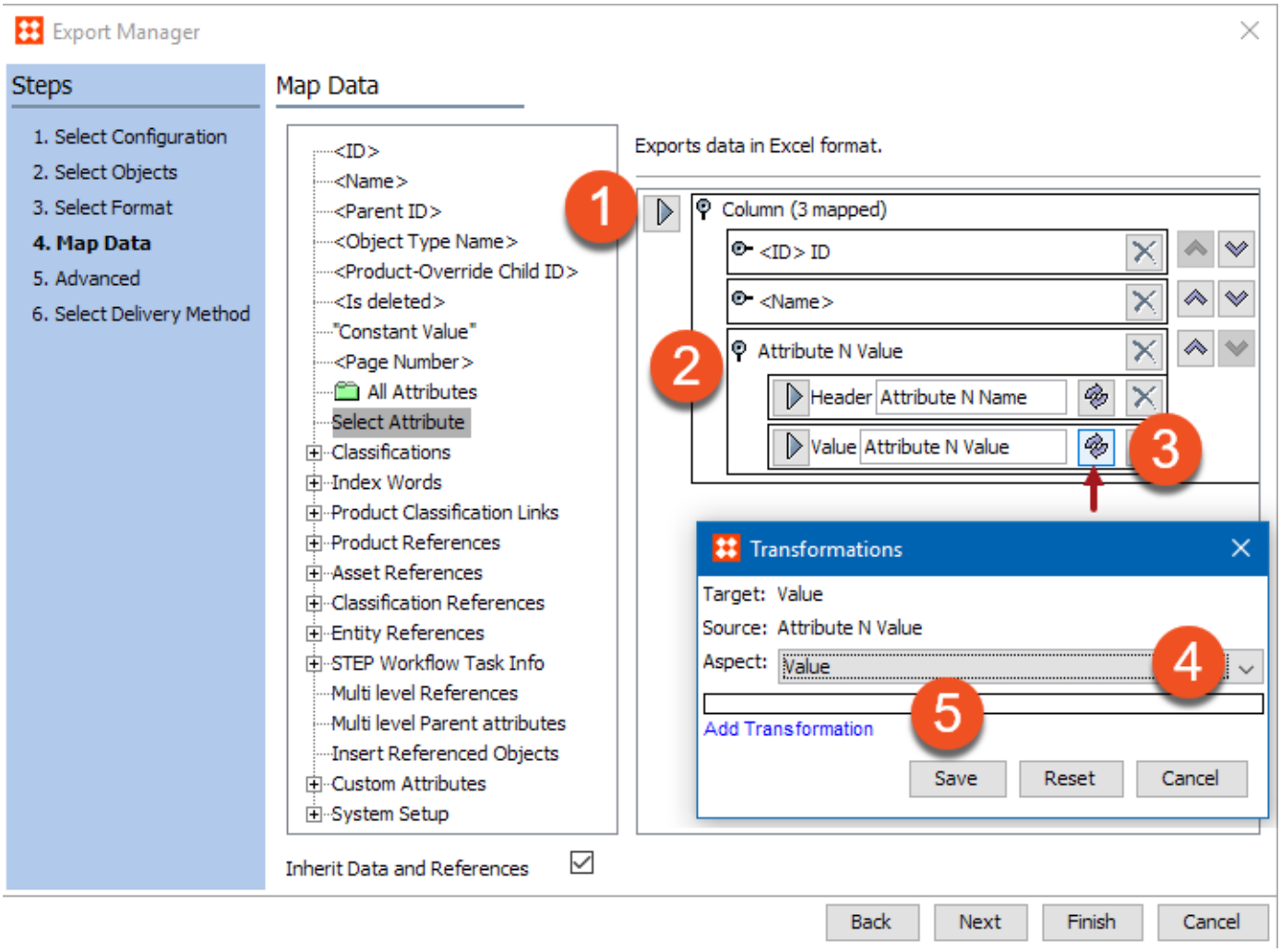
- Date localization and transformation works on attributes with these validation base types: Date (DD-MON-YYYY), 'ISO Date' (YYYY-MM-DD), and 'ISO Date and Time' (YYYY-MM-DD HH24:MI:SS).
- The steps for import and export must be followed carefully or the process may fail.

For information regarding converting attributes from Date type to 'ISO Date' type, see the **Maintaining Attributes** section of the **System Setup / Super User Guide** documentation.

Export Transformations

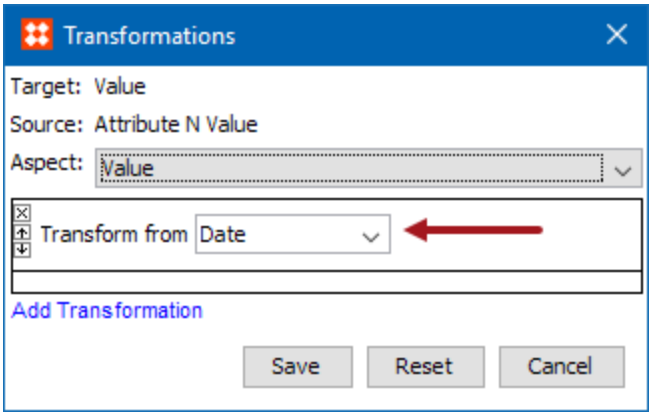
The steps for exporting data using date transformations should be followed very carefully. If not, the export may fail and the entire process will need to be restarted.

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, see **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 you wish to transform. For 'Attribute N' values, you will see a Header row and a Value row.
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 / Transform date by pattern. Follow the directions for the applicable selection and then continue on to the next step in these directions.



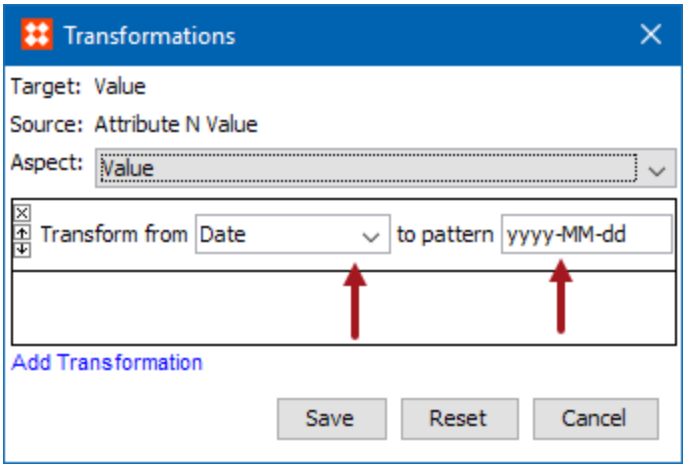
Transform date by locale

- 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, see **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, see **Export Manager - Select Delivery Method**.
11. Make selections on the Save Export Configuration screen, if desired. Click OK. For details, see **Running a Data Export**.
12. The 'Starting process Exporting' dialog will appear. For details, see **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.

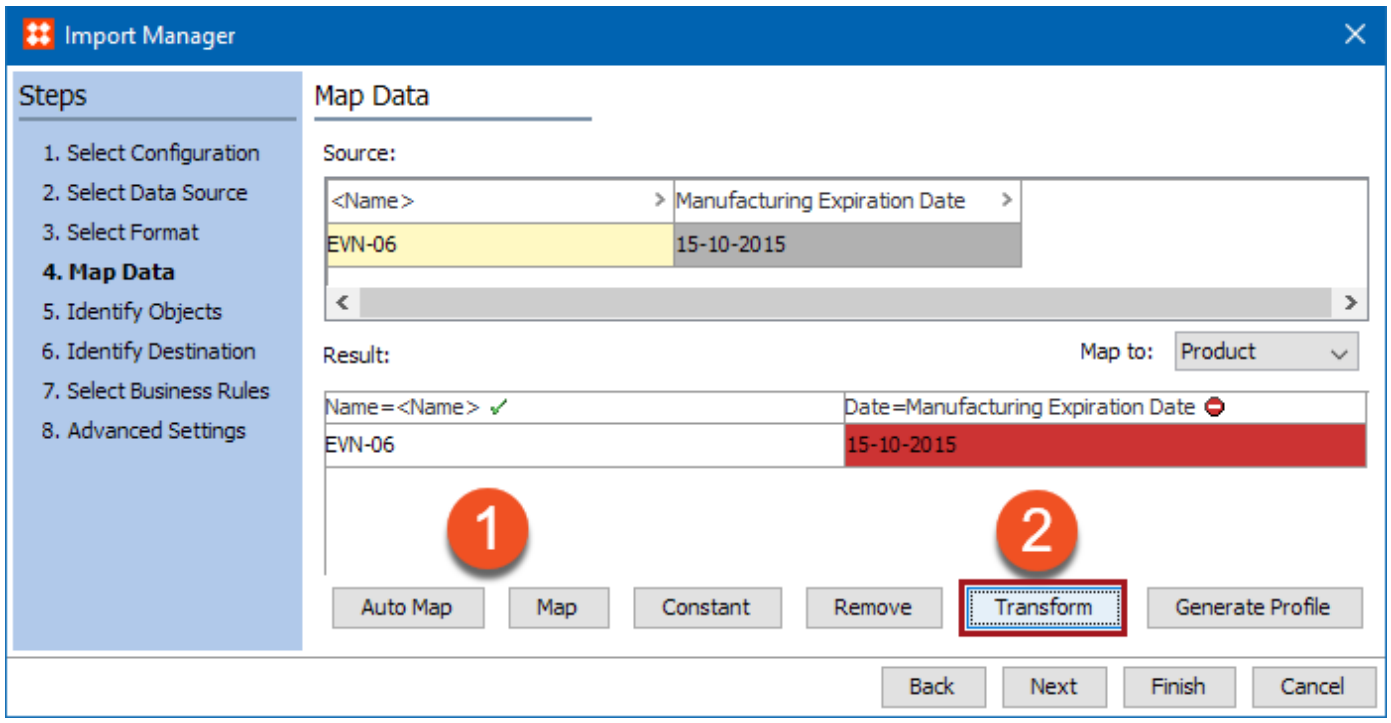
Import Transformations

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.

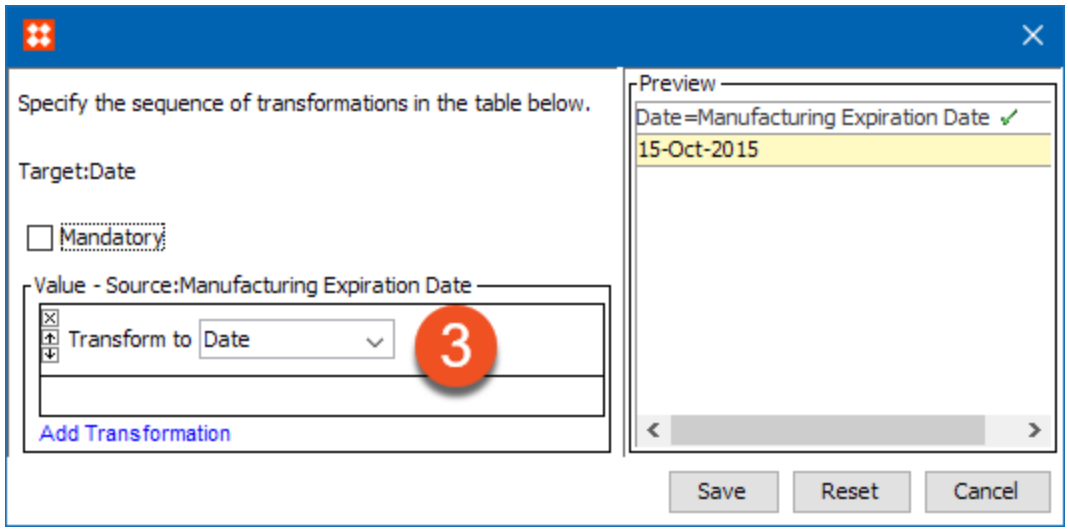
Important: Begin by making sure the date fields in the input file do not have any custom formatting. The best results will occur when importing dates that are in text format. Users should convert the date fields to text format prior to starting the import process.

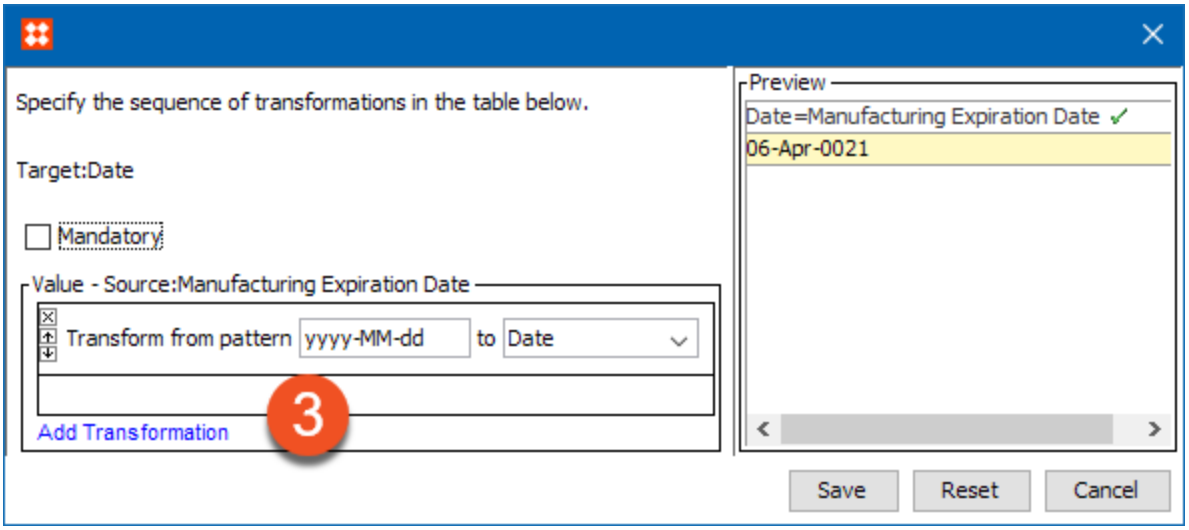
---

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 / selecting or manually typing in (or pasting) the file location. For details, see **Import Manager - Select Data Source**.
3. Make a format selection in the Select Format step. For details, see **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, see **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 / Transform date by pattern. Follow the directions for the applicable selection and then continue on with the remaining steps.



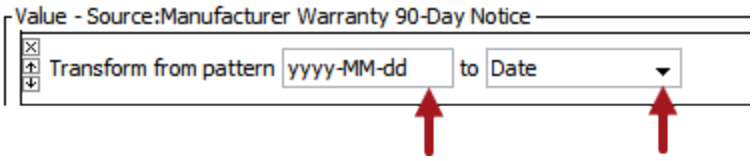


Transform date by locale

- 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.

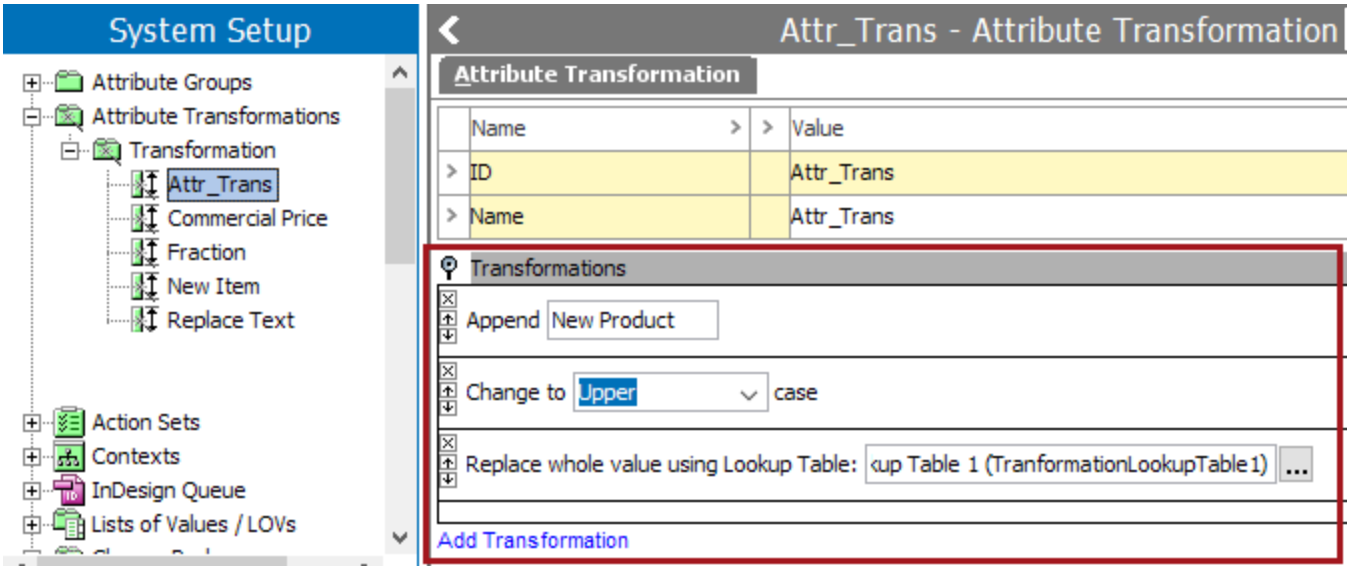
8. Select Next to move to the Identifying Objects step. For details, see **Import Manager - Identify Objects**. If you are importing an ID field with the objects, the data will be automatically matched up with existing data. If you are importing data without an ID field, click a Source column and then 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, see **Running a Data Import**.
12. The 'Starting process Importing' dialog will appear. Choose 'Go to process.' For details, see **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, the user will see an 'import completed' message 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

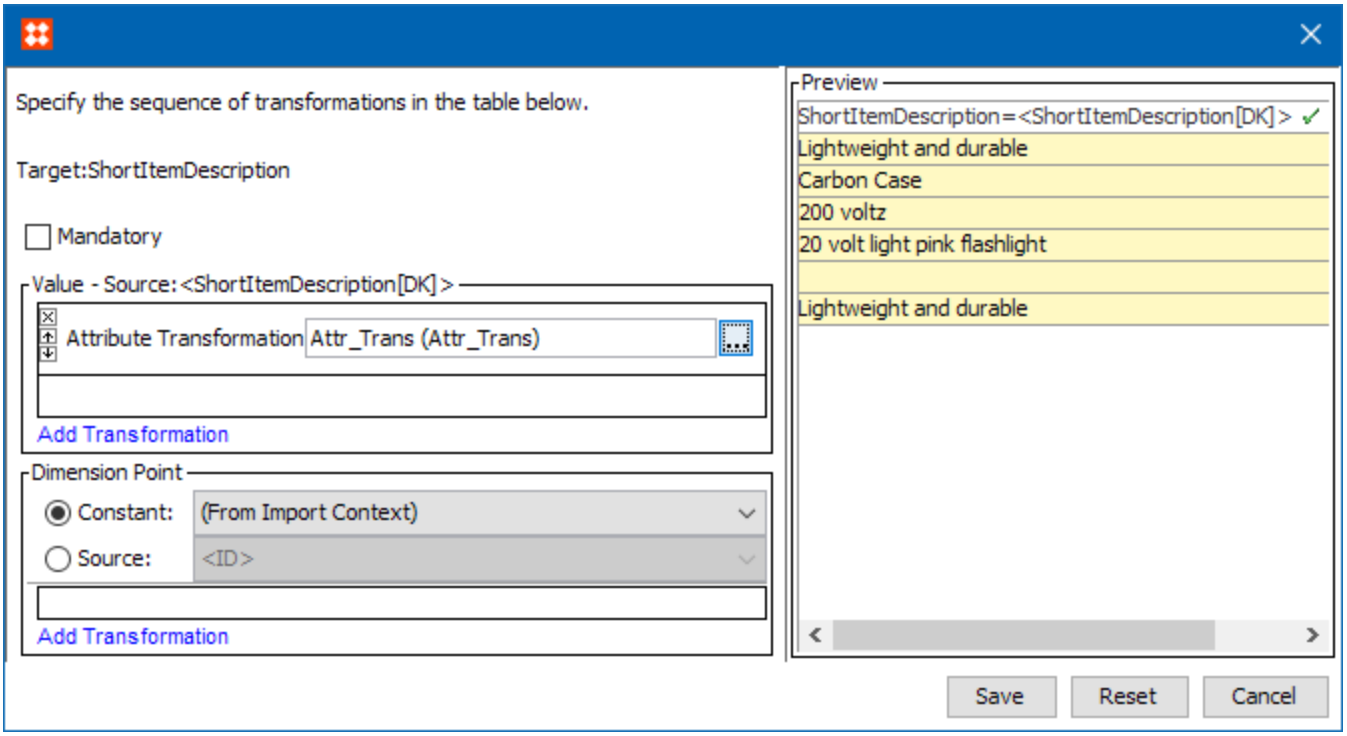
Transformations can improve the quality of the data you want to import. For general information about transformations, see the **Inbound Map Data - Transform** section.

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

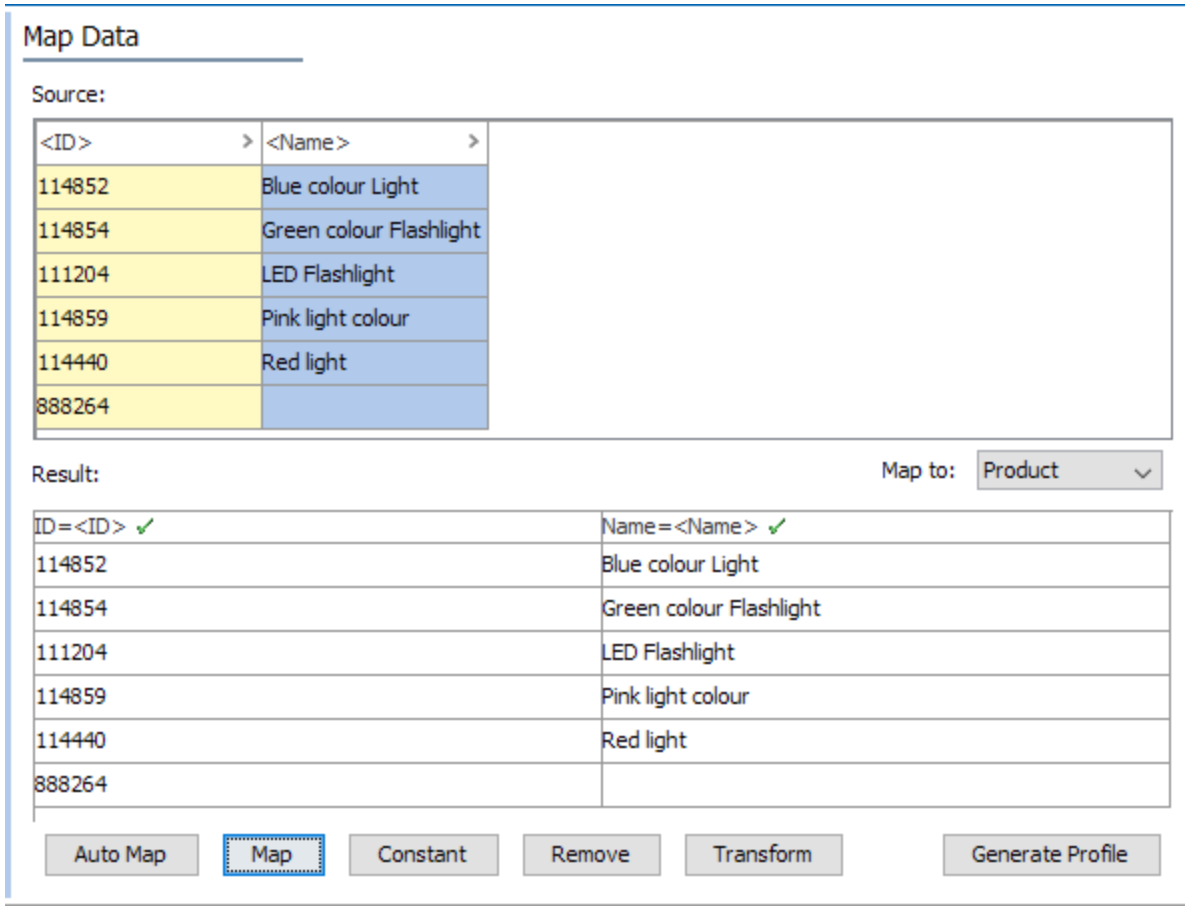
Transformations can improve the quality of the data you want to import. For general information about transformations, see the **Inbound Map Data - Transform** section.

Transformation Lookup Tables can be used to transform words and entire values when importing tabular-based data. For information on lookup tables, see the **Transformation Lookup Tables** topic in **Resource Materials** online help.

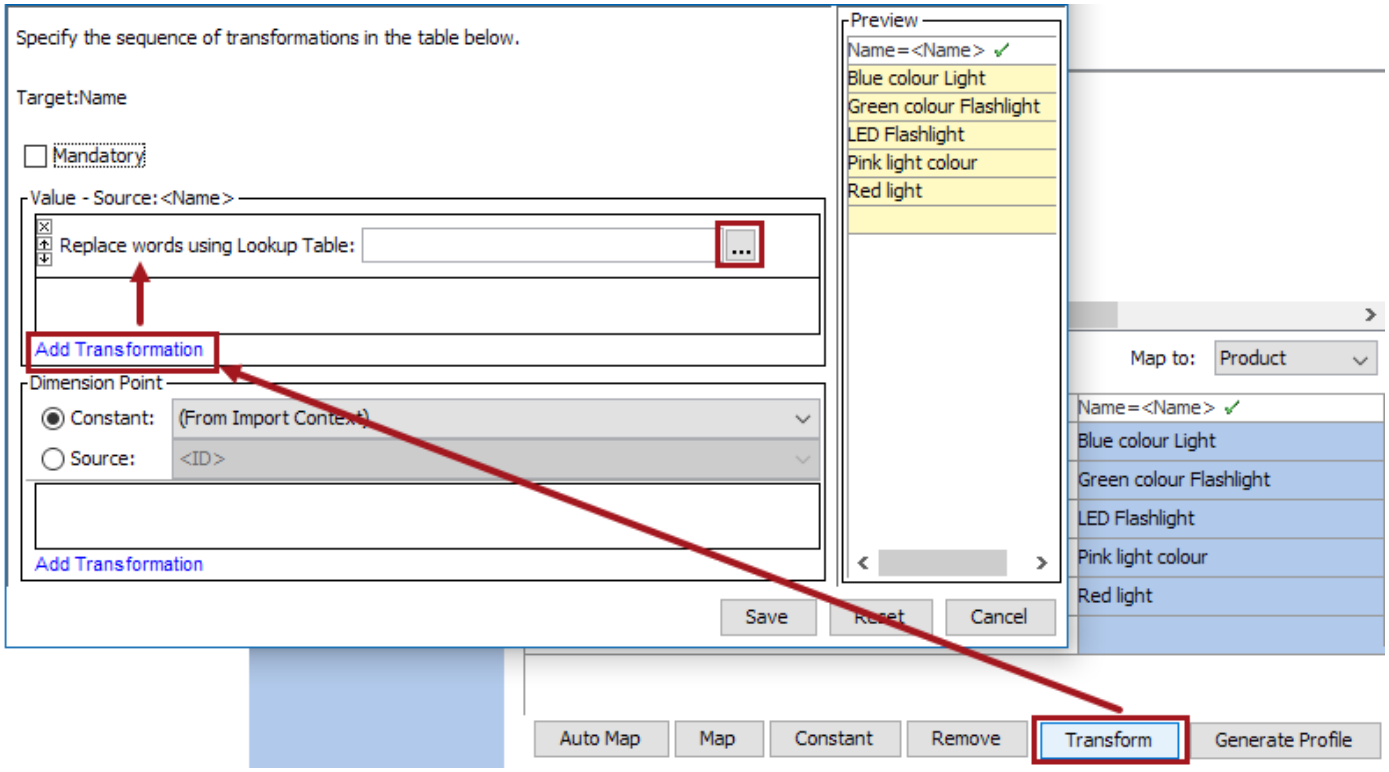
Note: Transformations that use Lookup Tables can be saved as an Import Manager configuration. To use a saved Lookup Table in a configuration, the referenced Transformation Lookup Table must not be deleted and must have content in the context of the import.

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.

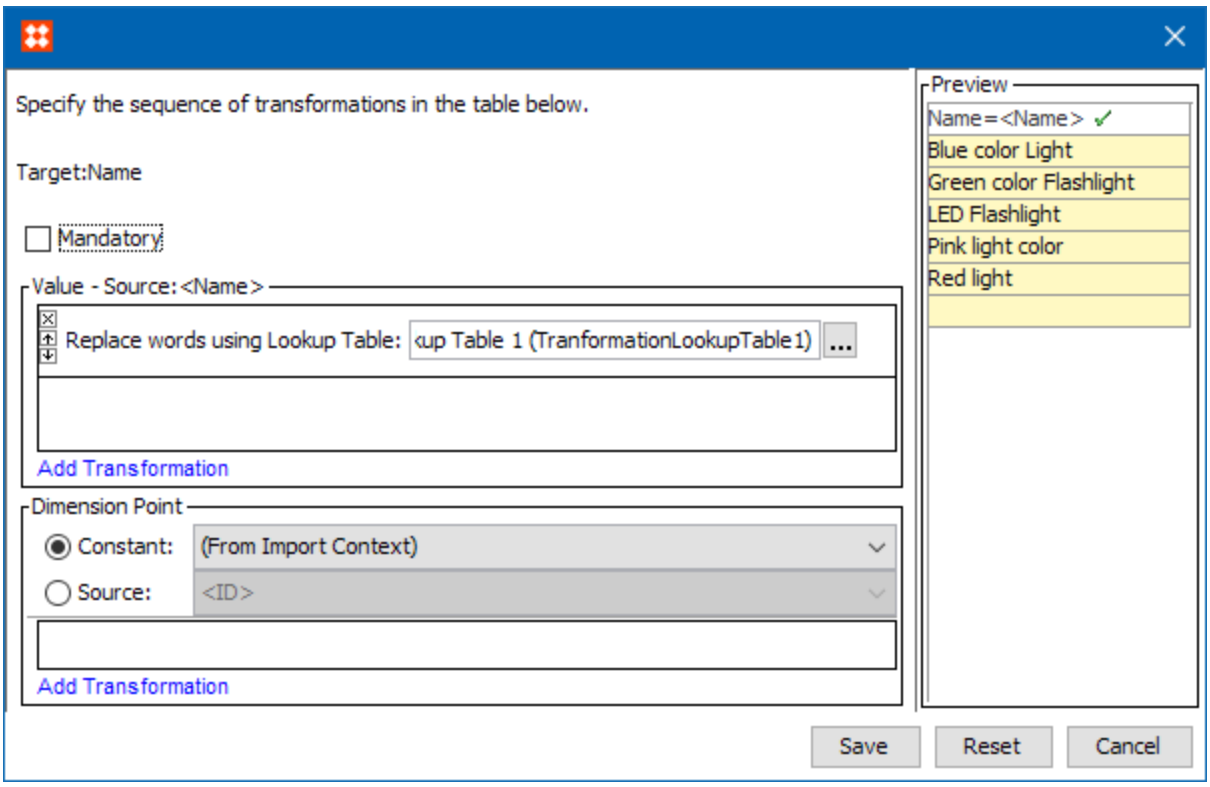
1. In the **Preview**, before any transformations are added to the Results field, the data looks like the table below:



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.

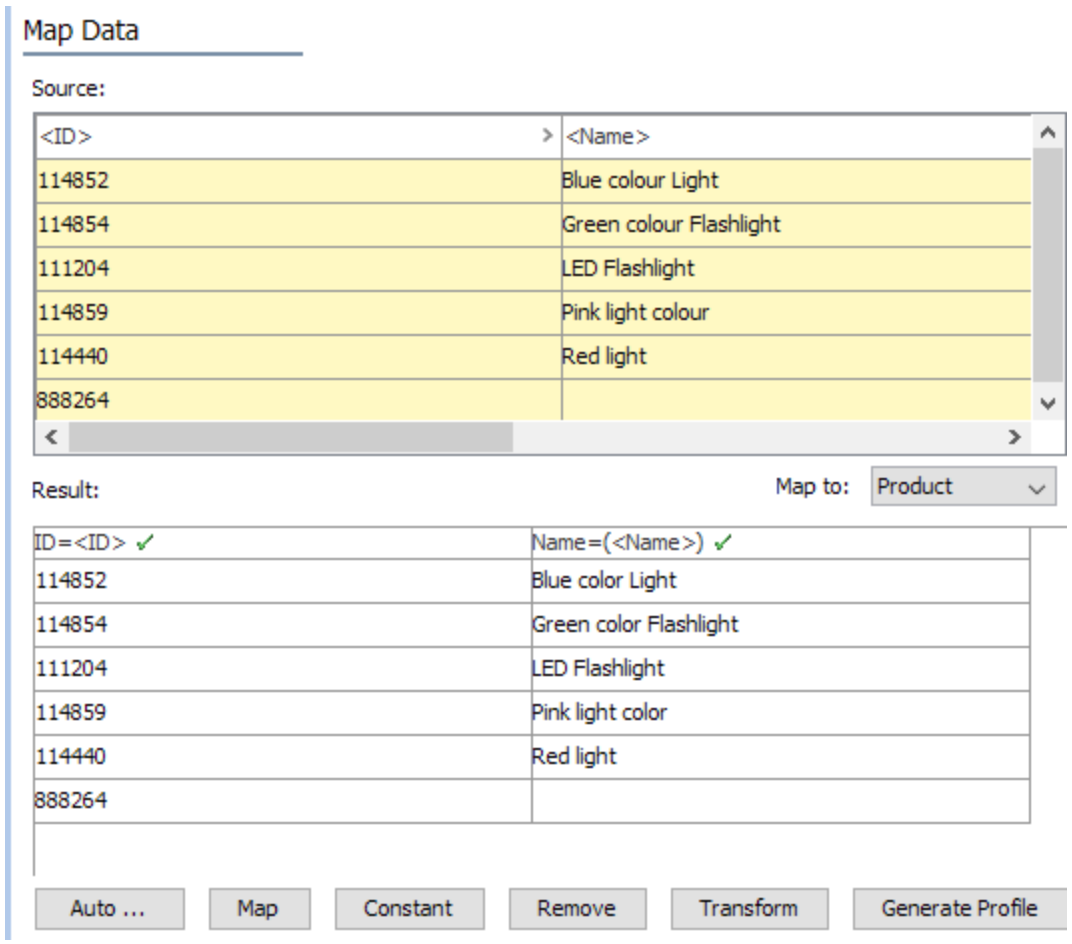


3. When properly added, the transformation changes the input text to the proper format.



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.


Map Data
Source:
<ID>
<Name>
114852
Blue colour Light
114854
Green colour Flashlight
111204
LED Flashlight
114859
Pink light colour
114440
Red light
888264
Result:
Map to:
Product
ID=<ID>
Name=(<Name>)
114852
Blue color Light
114854
Green color Flashlight
111204
LED Flashlight
114859
Pink light color
114440
Red light
888264
Auto ...
Map
Constant
Remove
Transform
Generate Profile

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 Data Export** within the **System Setup / Super User Guide** 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

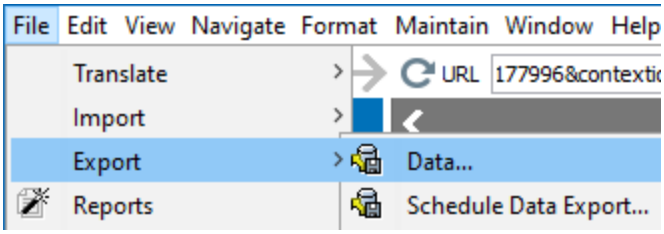
1. Select the appropriate STEP context. All data is exported using the currently selected context and the current workspace, typically Main.

   Note: When exporting products, see 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.

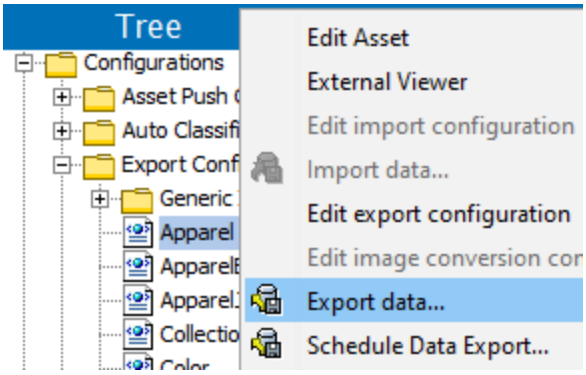
   - **From the File Menu**: Click the File menu > Export > **Data**.

     The wizard opens in Step 1 and allows you to select an existing configuration or create a new one.

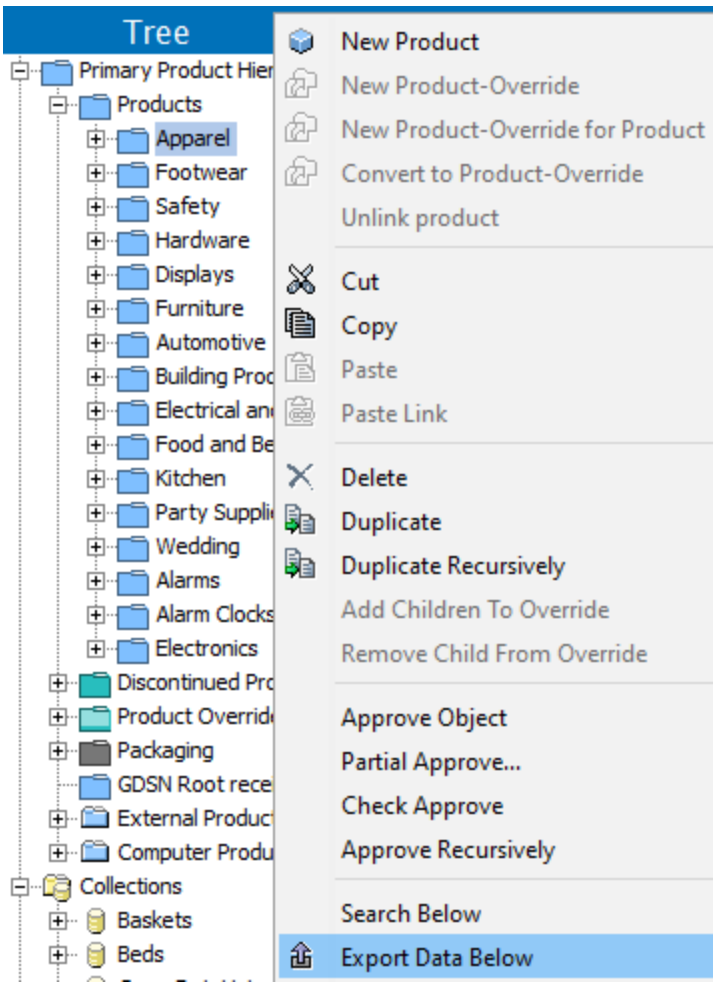
   - **From an Export Configuration**: On the Tree, select an export configuration, right-click, and then select **Export Data**.

     The wizard opens in Step 2 with the configuration and objects included in the configuration already selected.

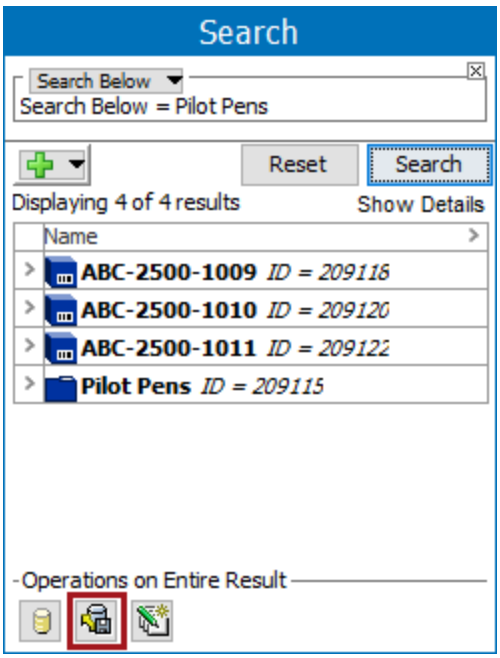
   - **From an Object**: On the Tree, select a classification, product group, publication, or entity, right-click, and then select **Export Data Below**.

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.



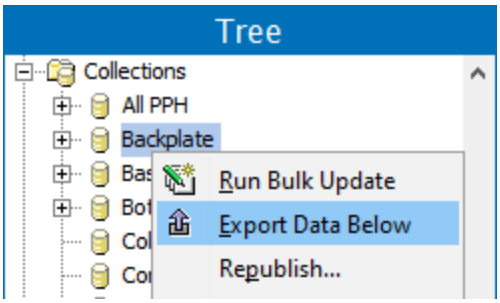
- **From a Search Operation**: On the Search tab, perform a search with results displayed. In the **Operations on Entire Result** section, click the export button.

  The wizard opens in Step 2 with 'Product selection made by search URL' displayed.



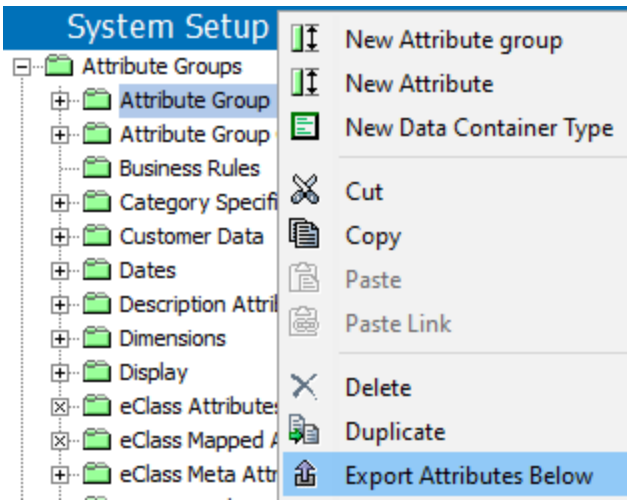
- **From a Collection**: On the Tree, select a collection, right-click, and then select **Export Data**. All object in the collection are included in the export.

  The wizard opens in Step 3 ready to choose a format.

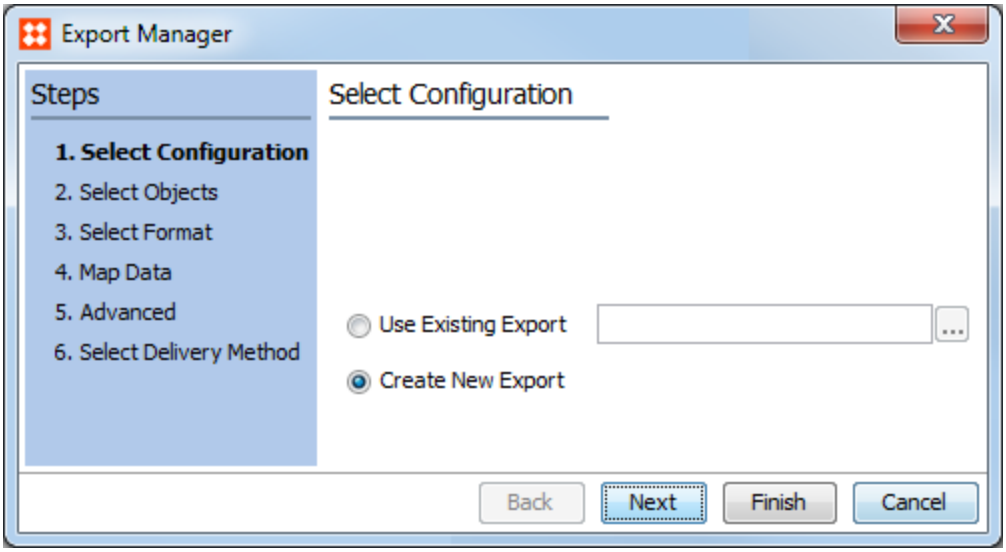


- **From an Attribute Group**: On System Setup, select an attribute group, right-click and then select **Export Attributes Below**.

  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

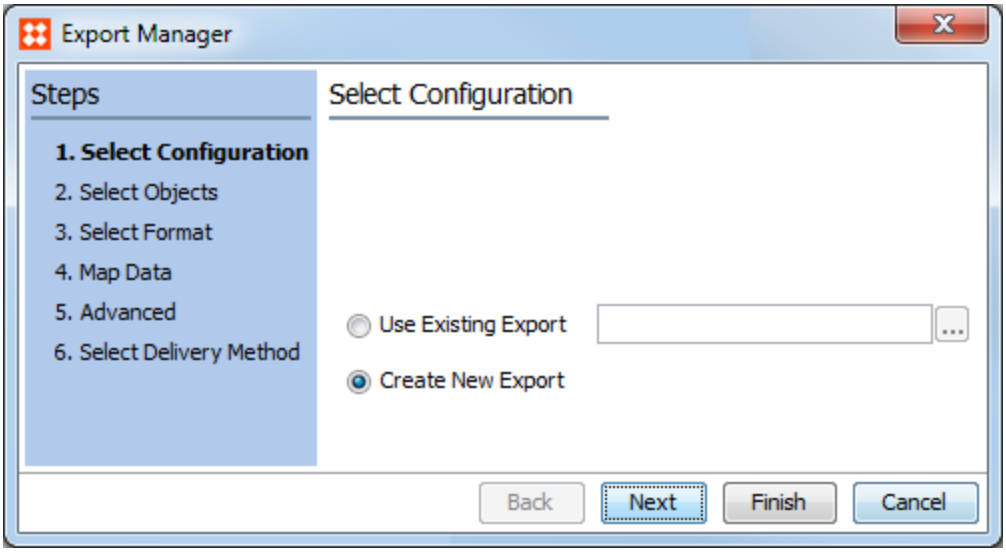
When exporting products, it is common setup to first review exactly the products and attributes that you want to extract. It is important to choose the exported attributes carefully since there is no distinction in the output between a missing (or blank) attribute value and a value that is empty because 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.

Determine the following information about your required export enables you to accurately set the wizard parameters:

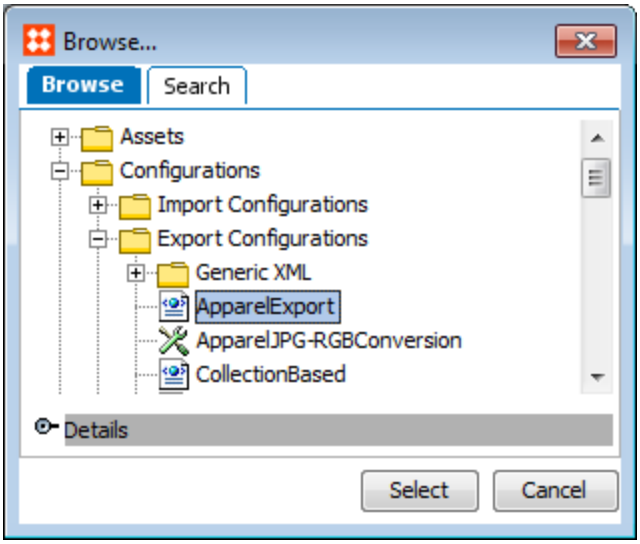
- If the attributes you want to extract are global, and for example, relevant for nearly all products, you can select products from disparate product lines, since the attributes would be valid for all those products.
- If the product lines you want to extract do not have common attributes, it can be useful to extract multiple files with separate sets of attributes.
- If your goal is to create an XLS file 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 Wizard, it is easier to select the appropriate attributes on the Data Map step.

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



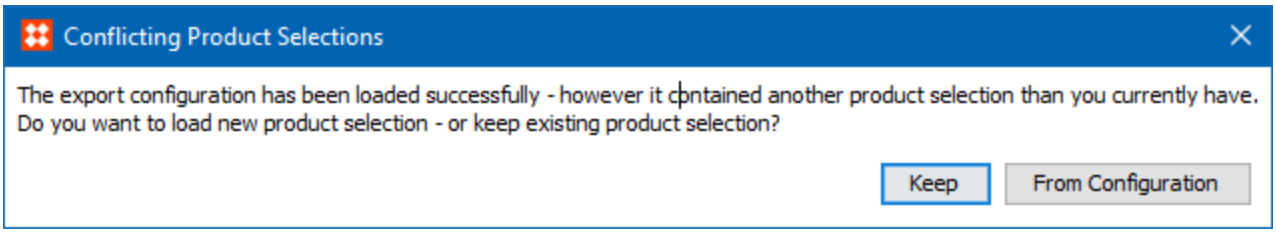
1. Select a radio button to use an existing export configuration, or create a new one. The following options are available when using an existing configuration:
   - To use an export configuration previously saved, select the **Use Existing Export** radio button, click the ellipsis button (...) , and either Browse or Search to locate the configuration in STEP. For more information on saved configurations, see 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

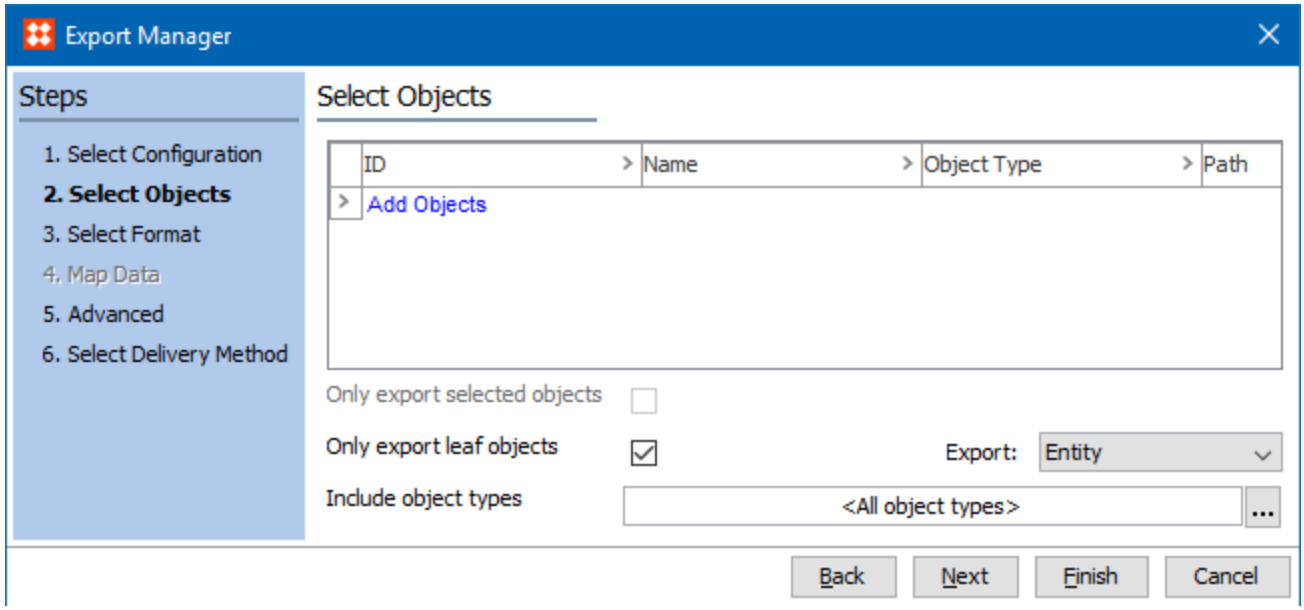
then save the modified configuration in the final step of the wizard.

- 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**.

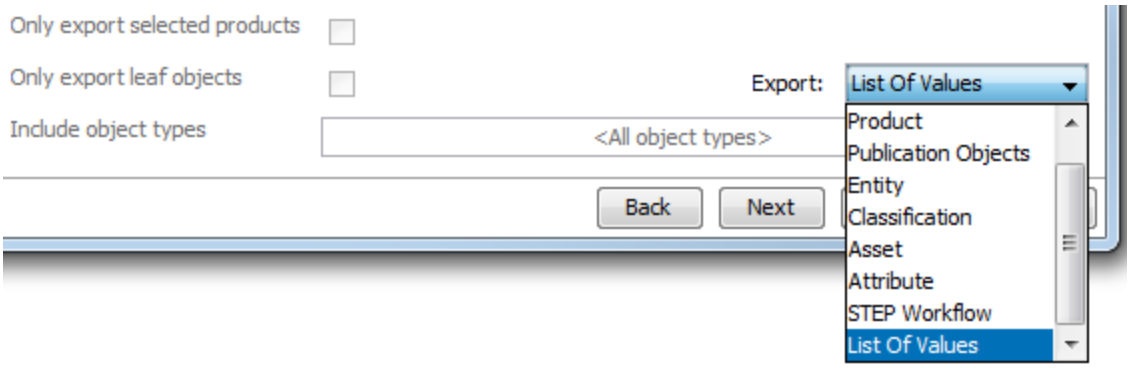
Export Manager - Select Objects



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, see **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 to export products, which results in exporting all products linked in below the selected classification.

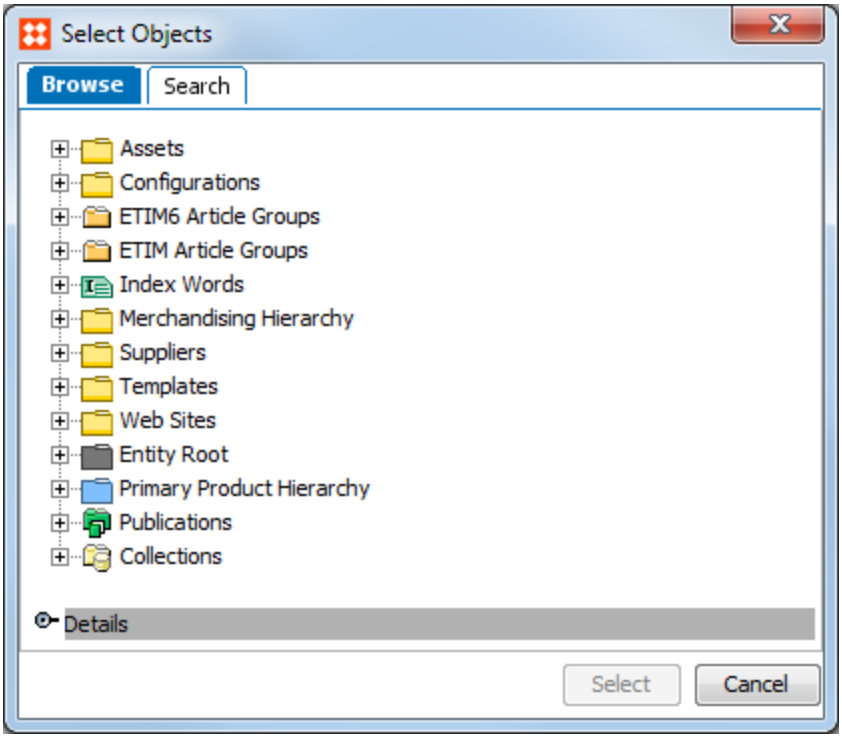


Note: Selecting an option from the Export dropdown impacts formats that require mapping. When the Map Data step is disabled, for example with STEPXML, this selection has no effect.

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, see 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 note 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.

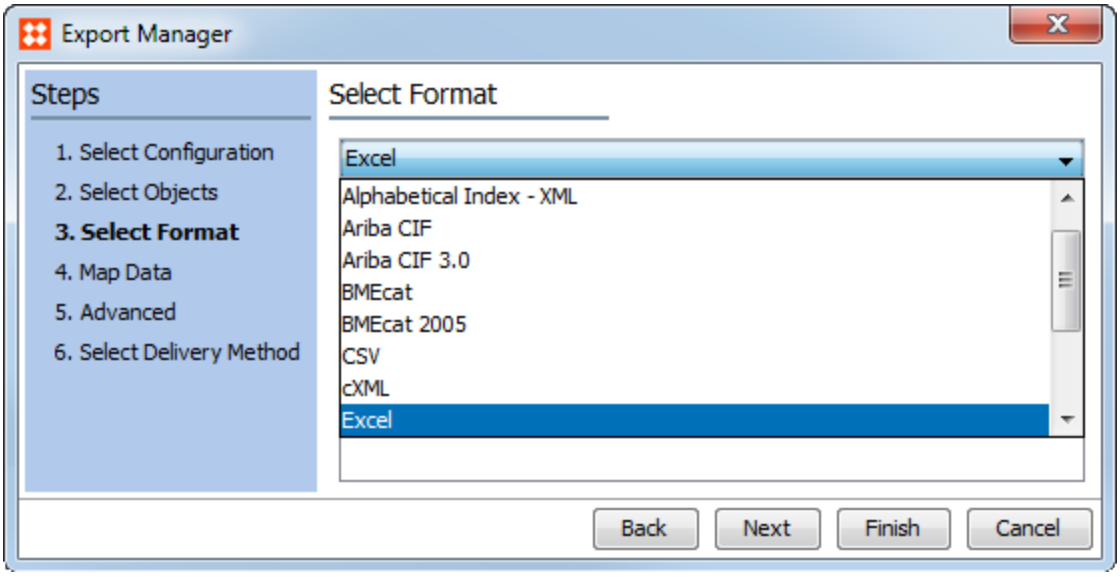


4. Click **Select**. The selected nodes appear in the **Select Objects** window.
5. Select **Only export selected objects** to export only the root nodes, and not the objects that live below the selected node. This option is only available when 'Only export leaf objects' is not selected. If available 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**.

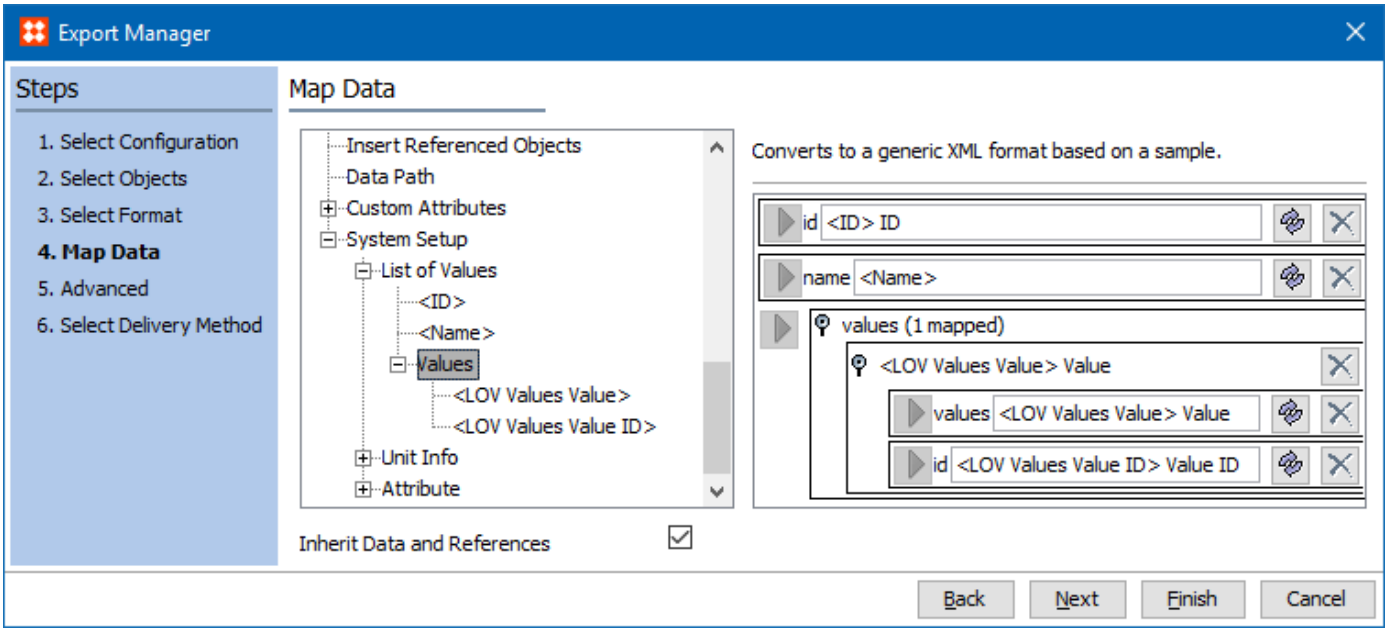
Export Manager - Select Format



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
   - CSV Format
   - cXML Format
   - Excel Format
   - Excel Custom Template
   - Excel Smartsheet Format
   - FAB-DIS Format
   - Flatplanner Format
   - Generic XML Format
   - IDoc MATMAS05 Format
   - Publication Excel Format
   - STEPXML 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



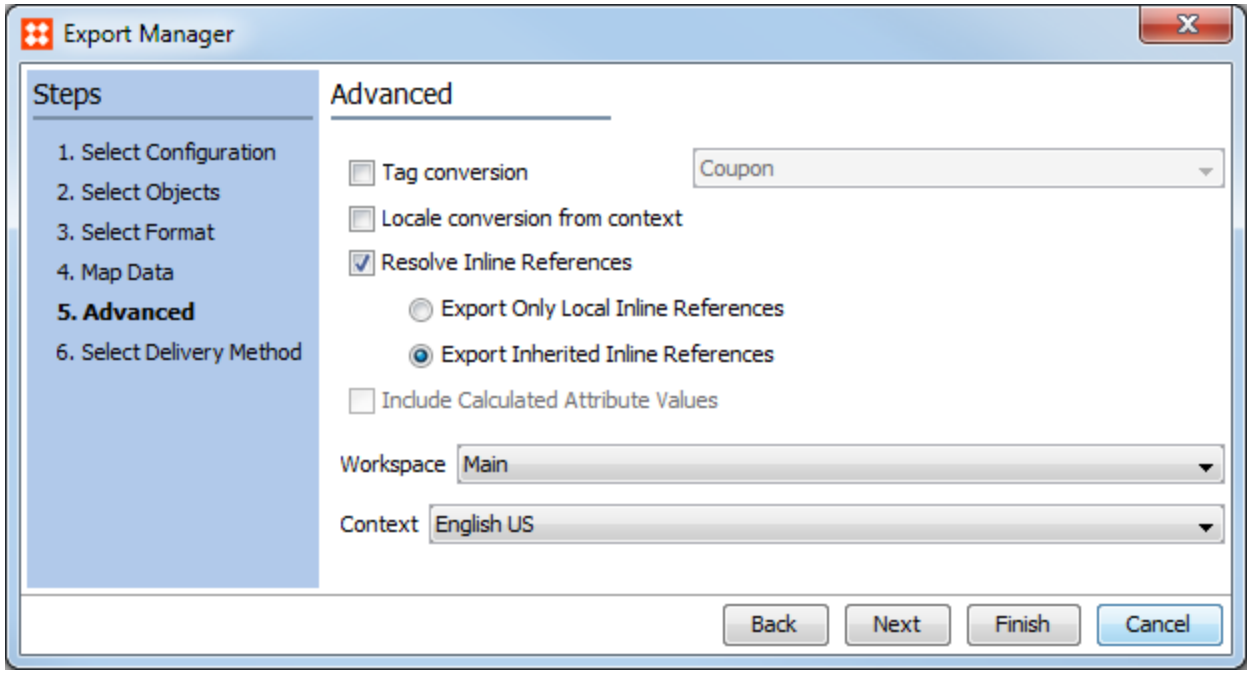
Generally, you will create mapping rules by selecting a data source (which indicates what you want to extract for each record) and creating a mapping target (which indicates where it belongs in the output).

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, see **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 see the **Tags** topic, and for the available tag conversion formats, see the **Configuring Tag Output Formatting** topic in the **System Setup / Super User Guide** 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.

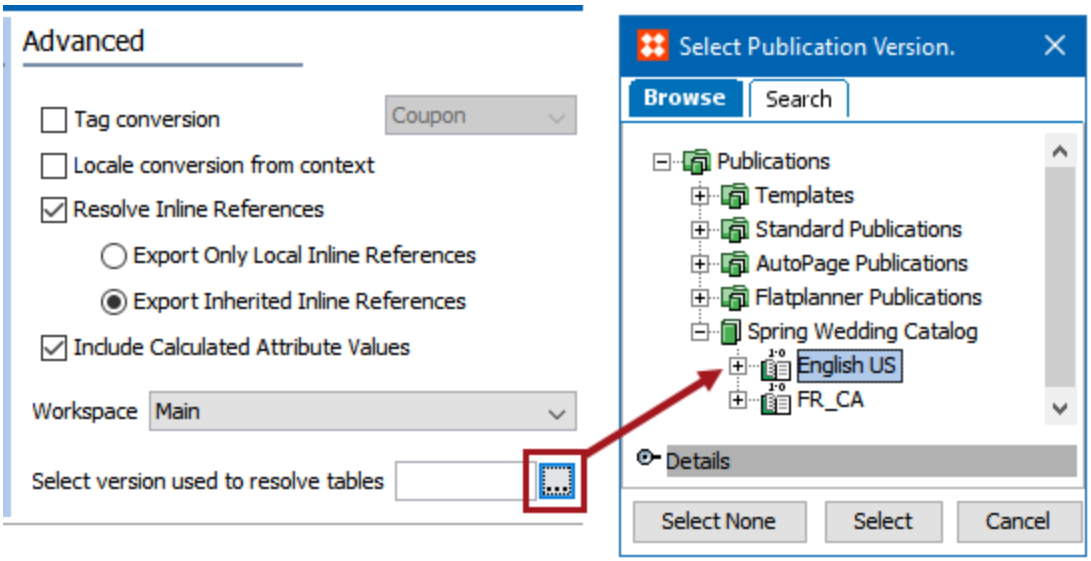
- **Include Calculated Attribute Values**: This option is only enabled (and is checked by default) when calculated attributes have been mapped. When checked, calculated attribute values are resolved upon export.

  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, see **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, see the **Calculated Attributes** topic in **System Setup / Super User Guide** documentation.

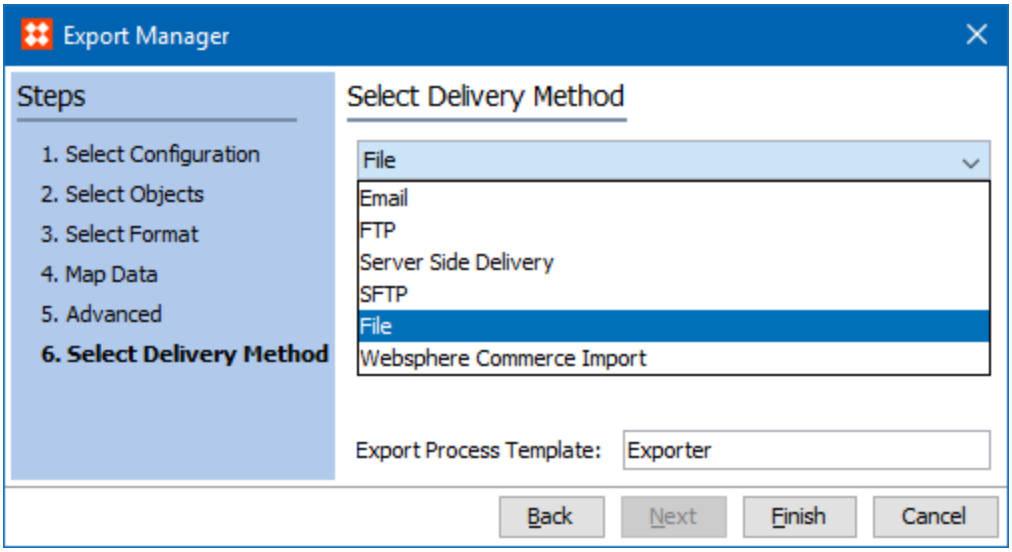
Important: If many complex calculated attribute values (traversing hierarchies and/or references) are used, consider if they should be exported, since they can impact performance. If required, consider scheduling for non-peak times. Simple calculations are not detrimental to an export, regardless of 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, see the **Export Data for Selected** Contexts section of the **STEPXML Outbound Parameters** documentation.
- **Select version used to resolve tables**: This setting is only available if the format is STEPXML and 'Include Tables' is set to Yes. 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). If a publication version is not selected, the version last used in STEP Workbench is used. For more information, see 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, see **OIEP Delivery Methods**.

1. In the **File Delivery** list, select one of the following export options:
   - **Email** delivers files as an email attachment and allows a zipped delivery.
   - **File** delivers a file via a background process and notifies when the file is ready for download.
   - **FTP** delivers files using file transfer protocol (FTP).
   - **Server Side Delivery** exports a file to a folder located on the STEP system application server.
   - **JDBC** exports data to a Relational database management system (RDBMS).
   - **SFTP** delivers a file using the Secure File Transfer Protocol (SFTP).
   - **Websphere Commerce Import** delivers a STEPXML file to a Websphere Commerce server.
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**.

Email Delivery Method

The email delivery option is very straight forward and, like 'Copy to directory', it allows the delivery to be zipped. It is typically used to test the export output, or to deliver a small export to one or more persons as a one-time occurrence.

This delivery method is available in Export Manager and in OIEPs. For information on a parameter, hover over the parameter to display help text.

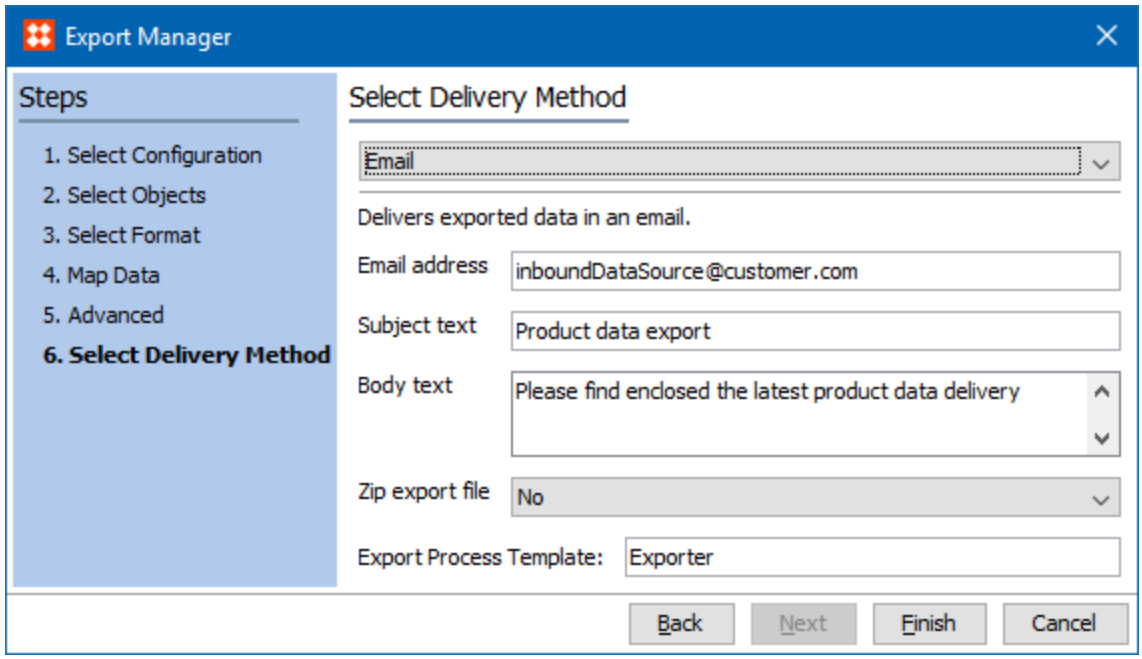
Prerequisites

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, see 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.
3. For the **Subject text** parameter, enter the text that will display for the email subject line.
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.

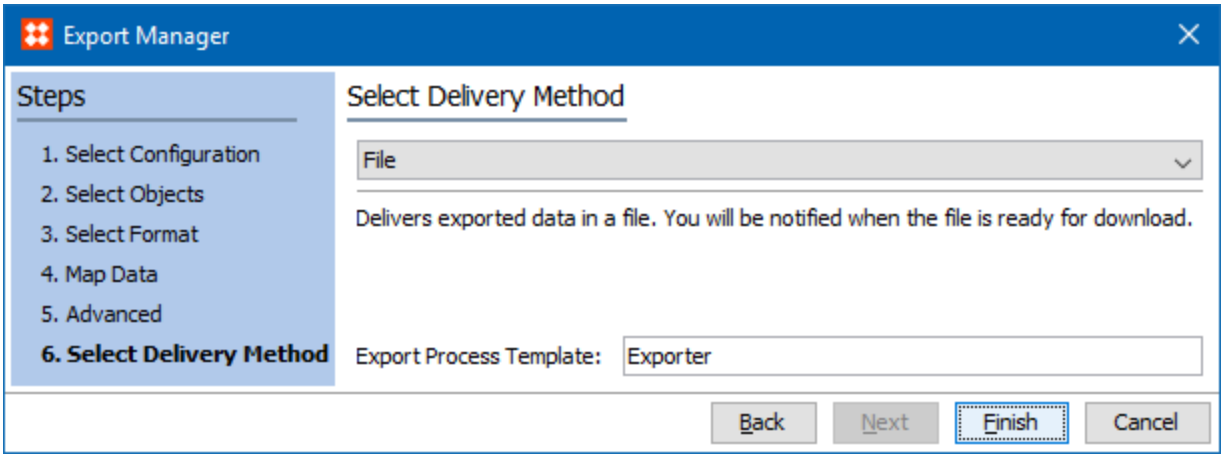
6. For the **Export Process Template** parameter, leave the default setting, which is required to start the background process for the export.
7. Click **Finish** to display the Save Export Configuration window as defined in **Running a Data Export**.

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

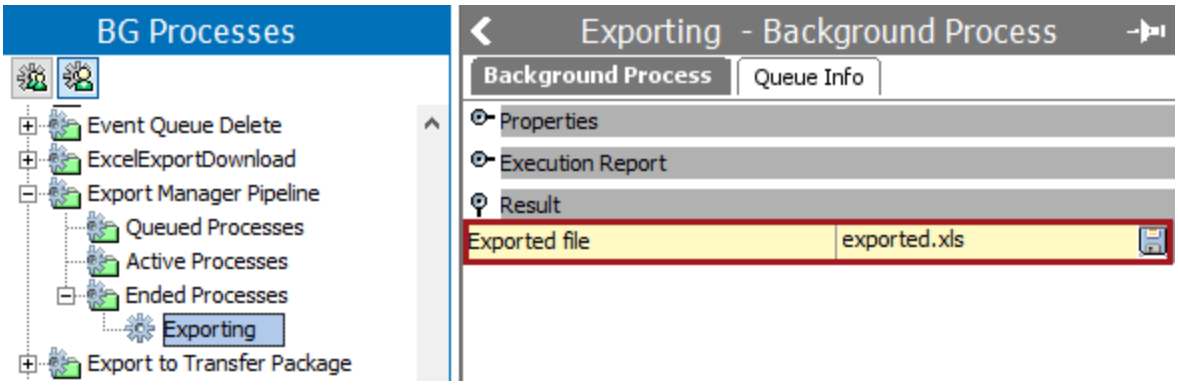
For information on a parameter, hover over the parameter label to display help text.



1. Select **File** from the dropdown.
2. 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**.

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**.



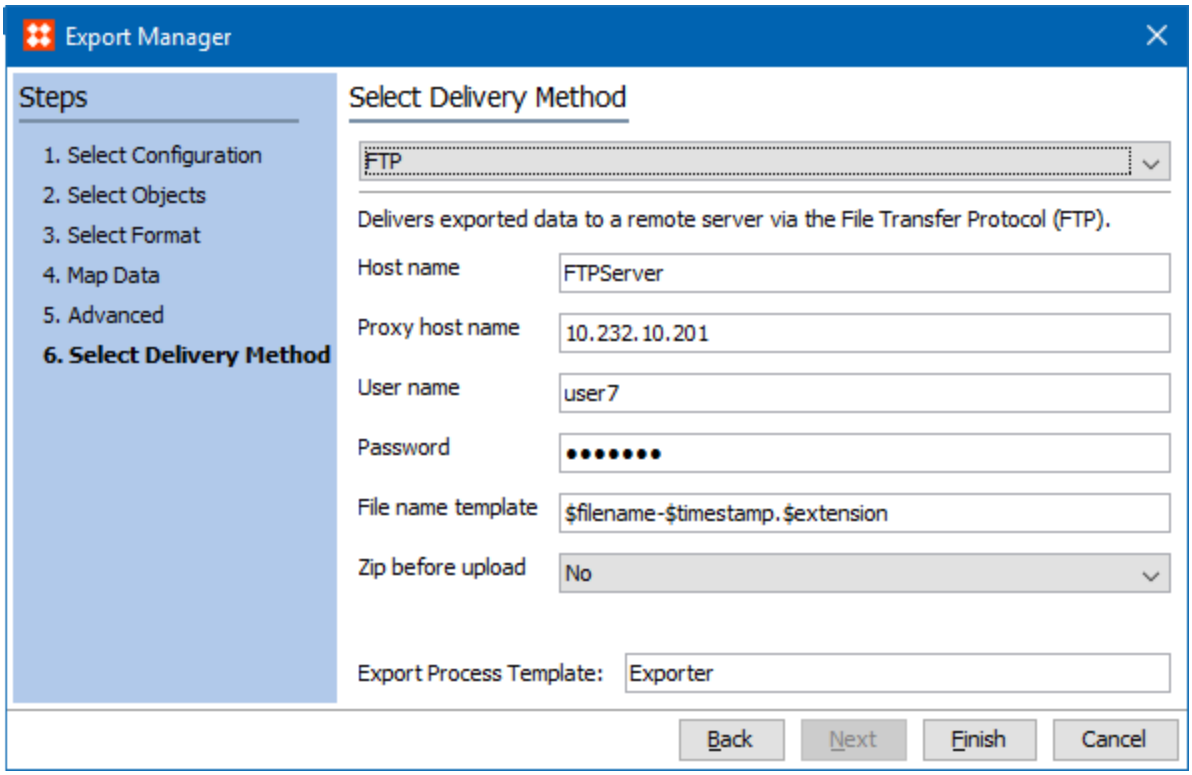
FTP Delivery Method

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. FTP is also available in OIEP.

The preferred method for FTP delivery is the SFTP secure delivery method. For information on the SFTP delivery method, see **SFTP Delivery Method**.

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, static text, or use a combination. The default template is $filename-$timestamp.$extension and creates a file name like 'GenericXML-1490813266535.xml' based on the settings of the OIEP. 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. For example, 20161511094709625 is used for an export on 15 NOV 2016 at 9:47:09:625 AM.

  Alternatively, the following options can be used to generate a readable date and can include static characters to separate the timestamp elements. Individual elements can be ordered as required, as shown below.

  $timestamp(ddMMyyyy) displays day, month, and year and is output '15112016' for 15 NOV 2016.

  $timestamp(yyyy-MM-dd_HH.mm.ss) displays year, month, day, hour, minutes, and seconds 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 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 Endpoint (OIEP).

For more information on the OIEP option, see **JDBC Delivery Method** in the **Outbound Integration Endpoints** documentation.

Prerequisites

Important: For complete setup requirements, see 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.'

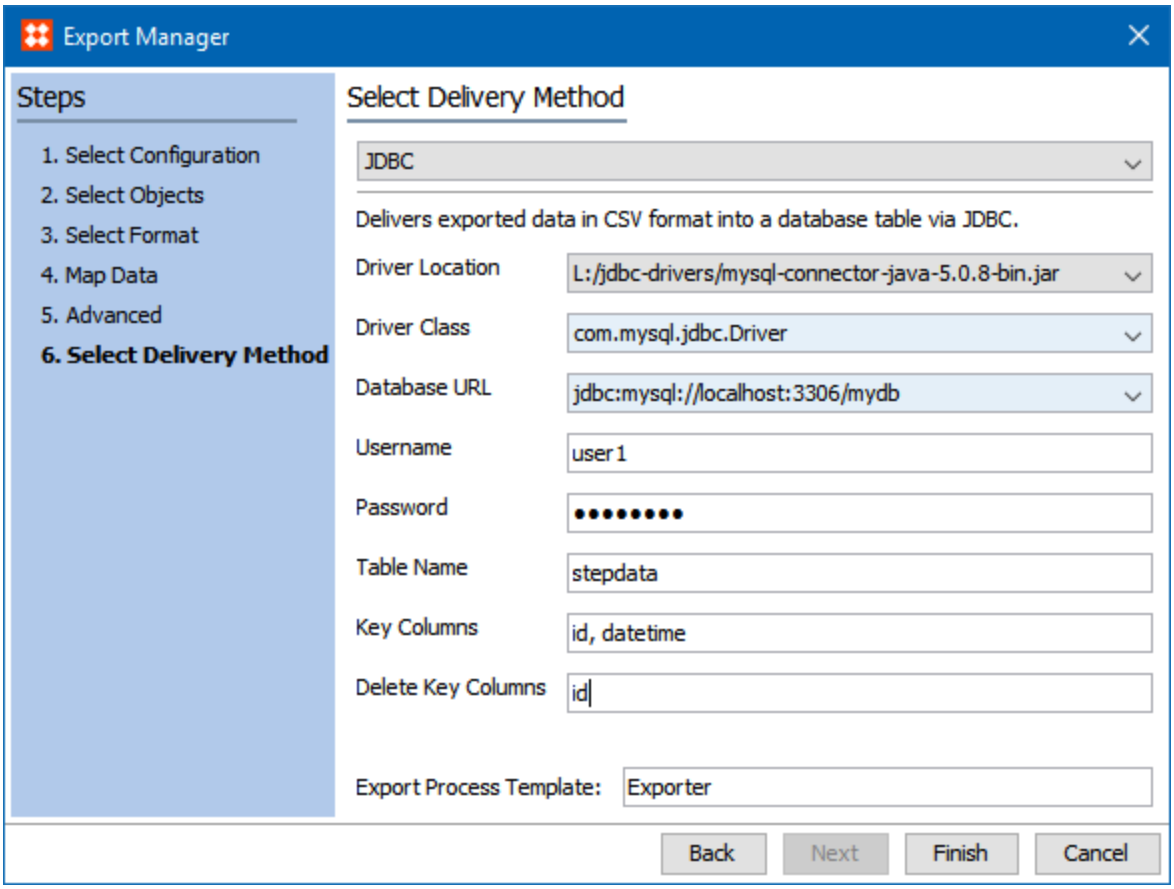
3. Prior to configuration, clicking the **Database URL** dropdown parameter displays the required property name. Provide a selection for the dropdown parameter via the sharedconfig.properties file on the STEP application server using the case-sensitive **JDBCDeliveryPlugin.URL.[n]** property.

   ```
   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.

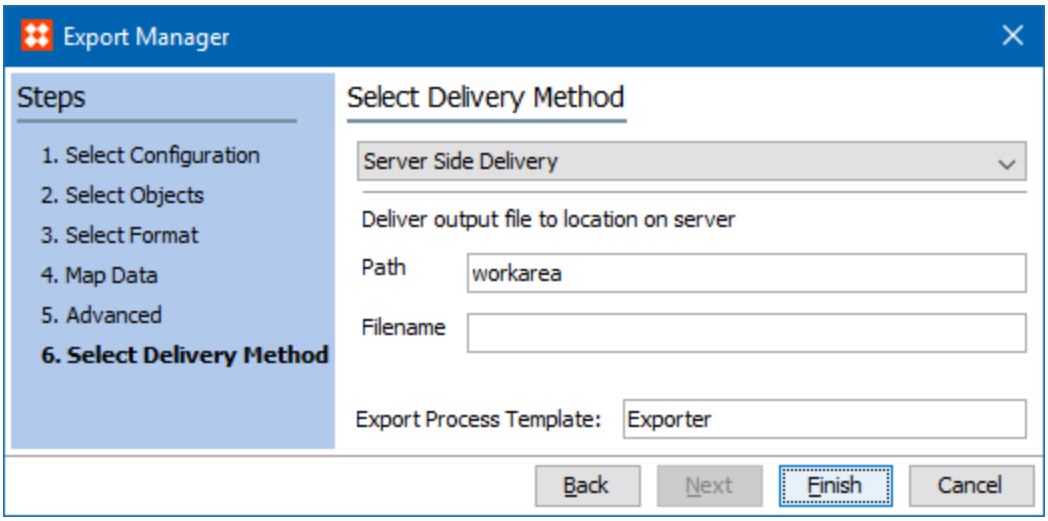
10. In **Export Process Template**, leave the default setting, which is required to start the background process for the export.
11. Click the **Finish** button to complete the wizard and begin the export as described in **Running a Data Export**.

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

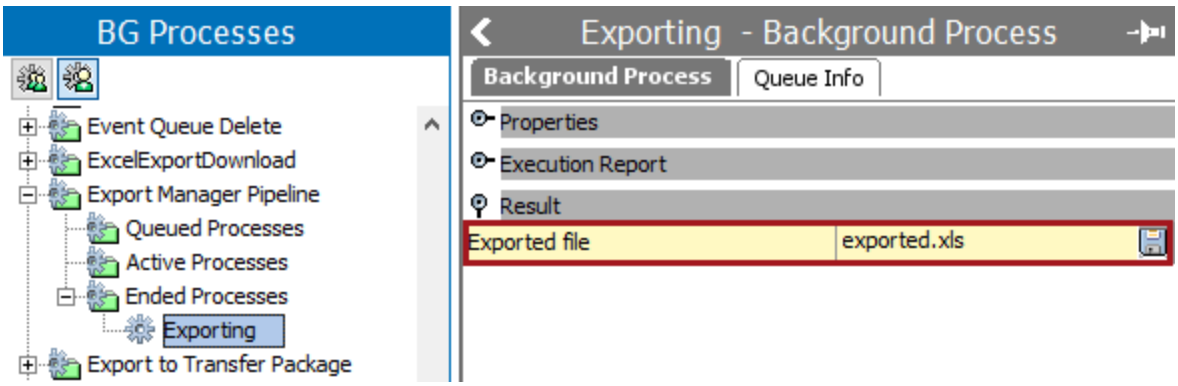
For information on a parameter, hover over the parameter label to display help text.



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

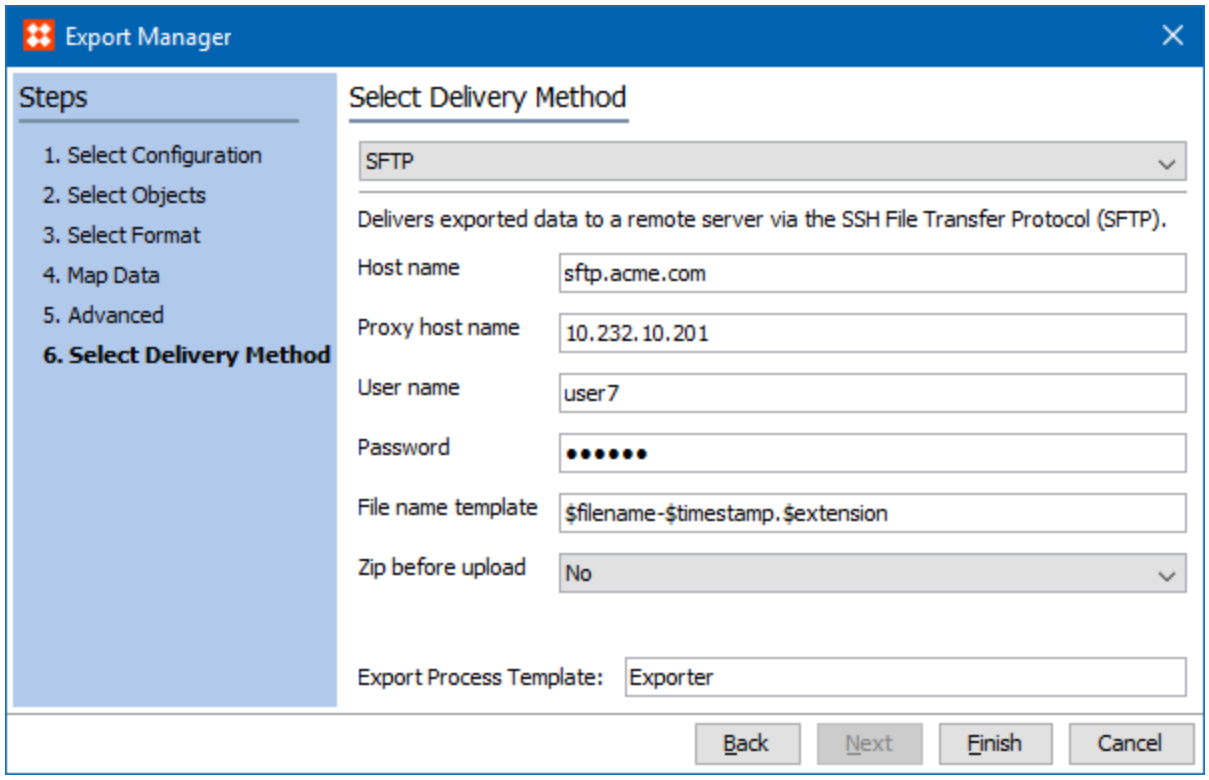
SFTP Delivery Method

The preferred method for 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.

SFTP is also available in OIEP. For information on the FTP delivery method, see **FTP Delivery Method**.

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.
6. In **File name template**, to create the file name used for the output, enter template variables, static text, or use a combination. The default template is $filename-$timestamp.$extension and creates a file name like 'GenericXML-1490813266535.xml' based on the settings of the OIEP. Each variable is described below:
   - **$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. For example, 20161511094709625 is used for an export on 15 NOV 2016 at 9:47:09:625 AM.

  Alternatively, the following options can be used to generate a readable date and can include static characters to separate the timestamp elements. Individual elements can be ordered as required, as shown below.

  $timestamp(ddMMyyyy) displays day, month, and year and is output '15112016' for 15 NOV 2016.

  $timestamp(yyyy-MM-dd_HH.mm.ss) displays year, month, day, hour, minutes, and seconds 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.

---

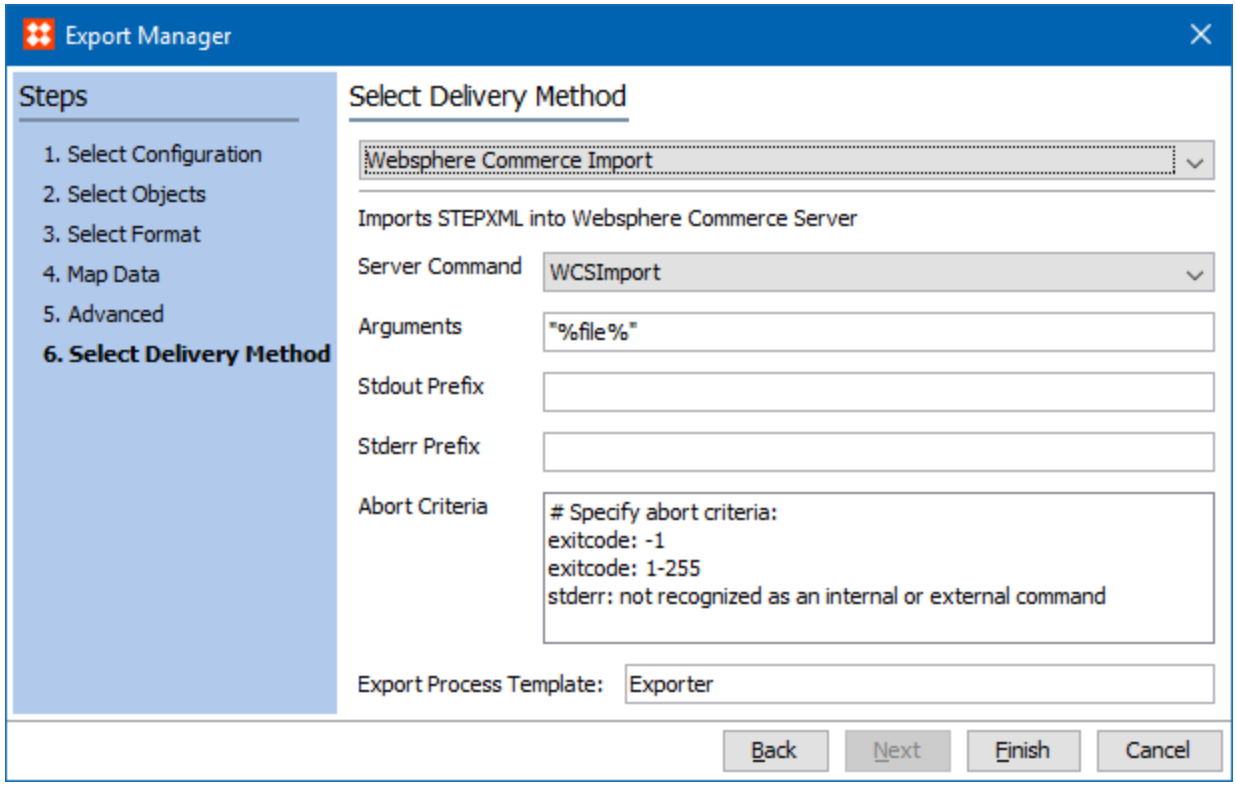
7. In **Zip before upload**,select an option from the dropdown to specify if the output file should be delivered in a .ZIP (compressed) file format.
   - **Yes** uses the File Name Template and the extension .ZIP. When a file is zipped, the File Name Template is used for the .ZIP file. The $filename 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**.

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**.

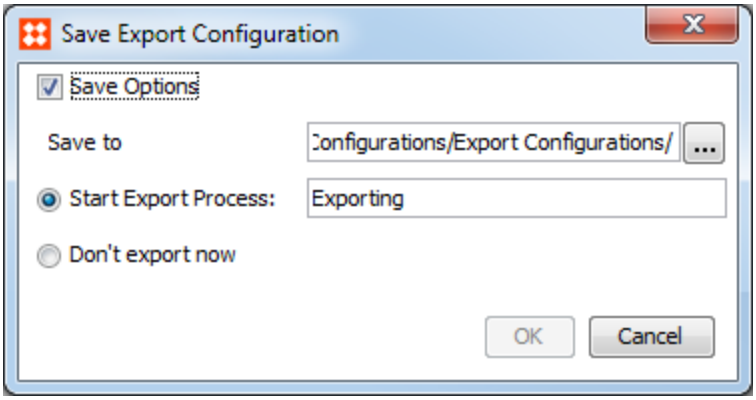
Running a Data Export

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. Whether you save your configuration or not, this dialog also allows you to start the export process.

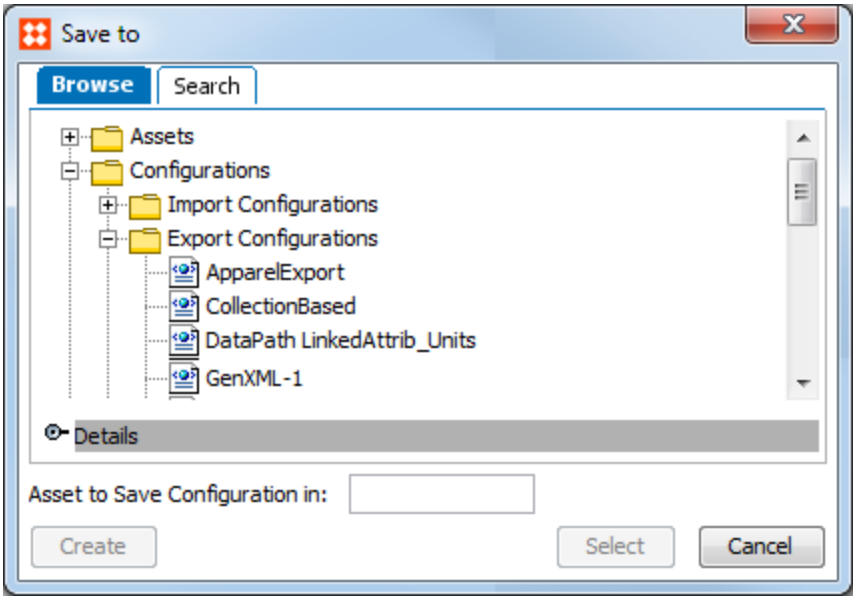
Save an Export Configuration

1. The **Save Options** checkbox is checked by default. Leave this setting if:
   - You made changes to an existing configuration and want to save those changes.
   - You want to save a new configuration.

   Uncheck the Save Options checkbox if you do not want to save this configuration.



2. When the Save Options checkbox is checked, click the ellipsis button (...) on the Save to field 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 or Search option to select the configuration, and click the **Select** button which is now enabled.



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, see **Monitoring a Data Export**.

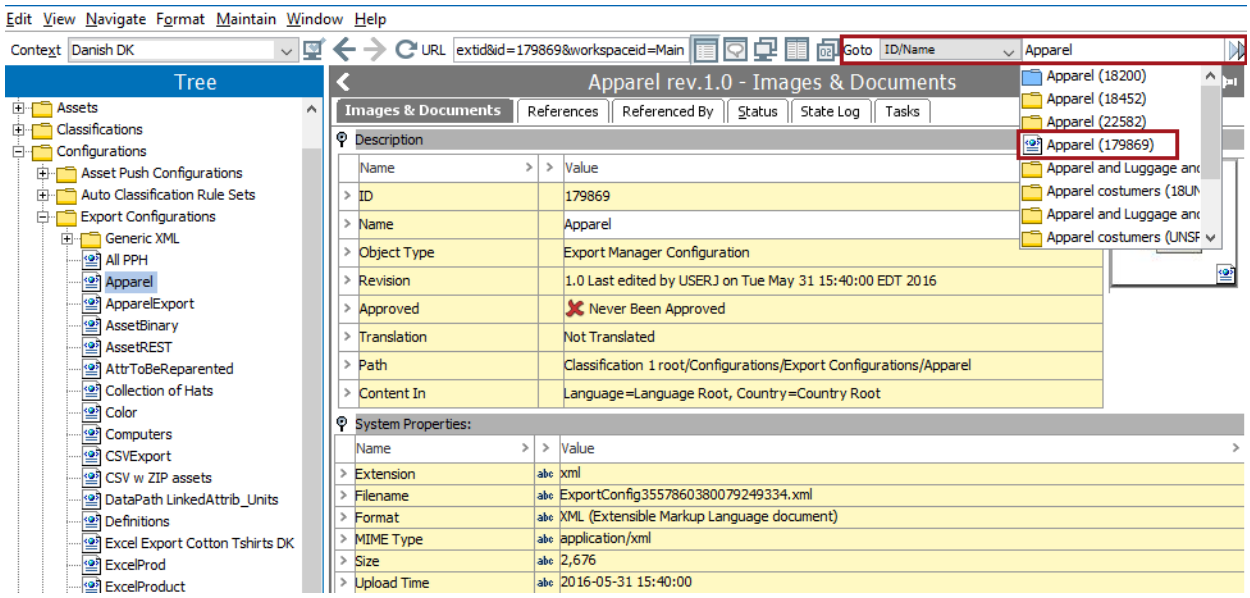
Maintaining a Saved Export Configuration

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 (see **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.

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

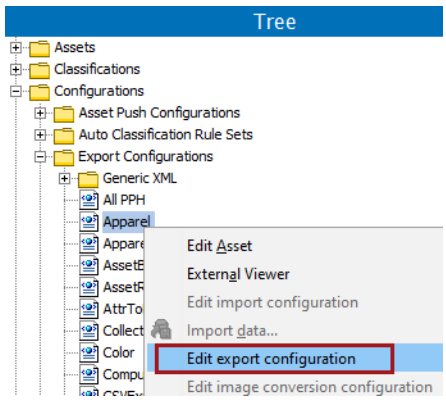
Use the following steps to apply updates to the configuration selected for editing.

- 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, see 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.

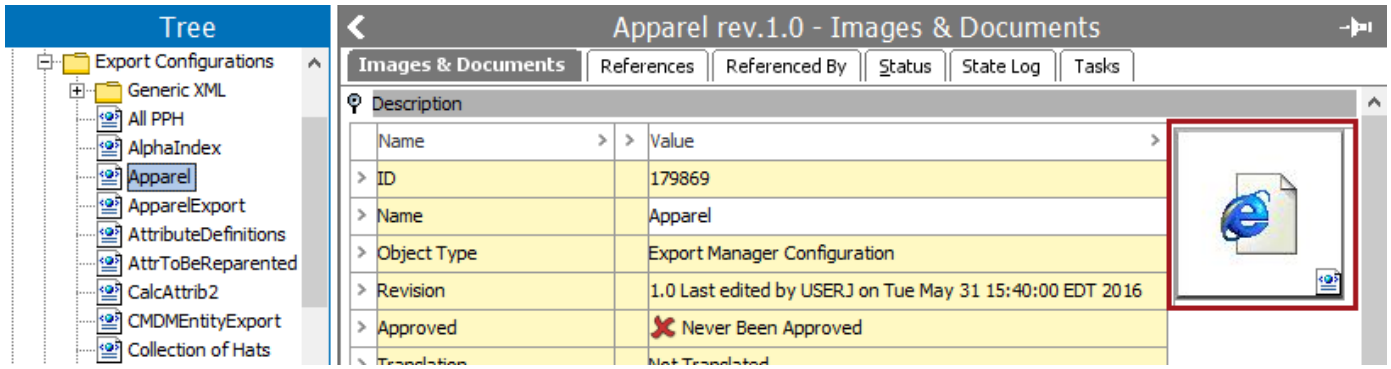
Edit export configuration via XML

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.

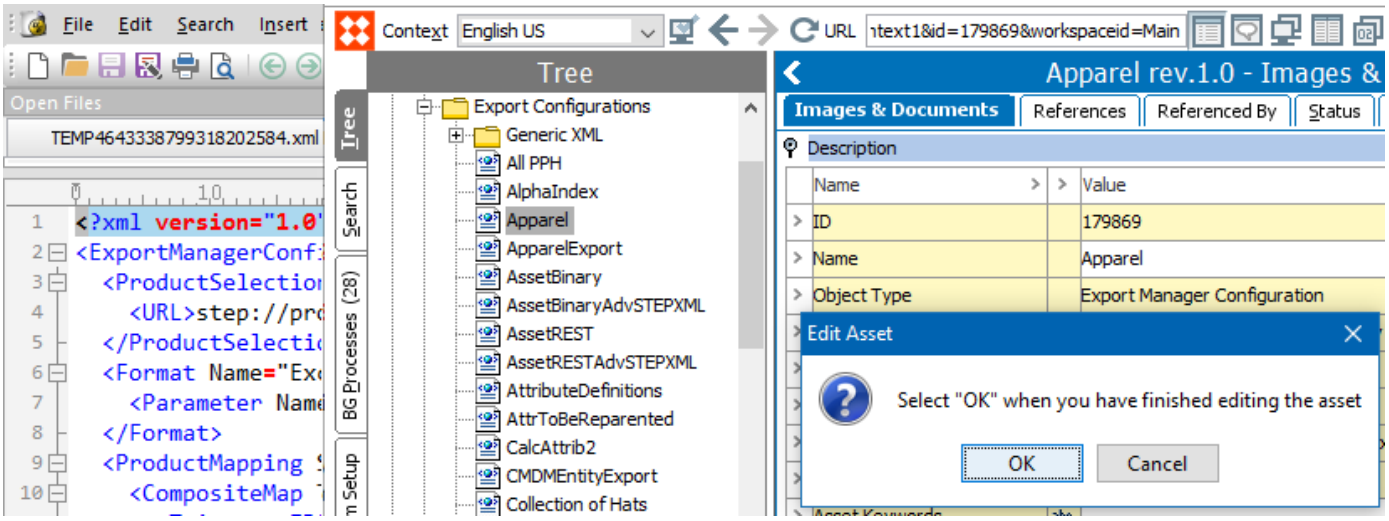
---

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.

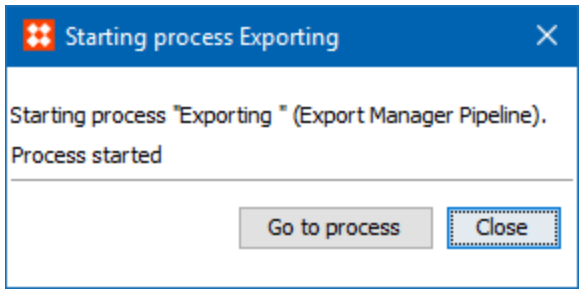
Important: Do not dismiss the Edit Asset dialog until updates to the XML file are complete.



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:



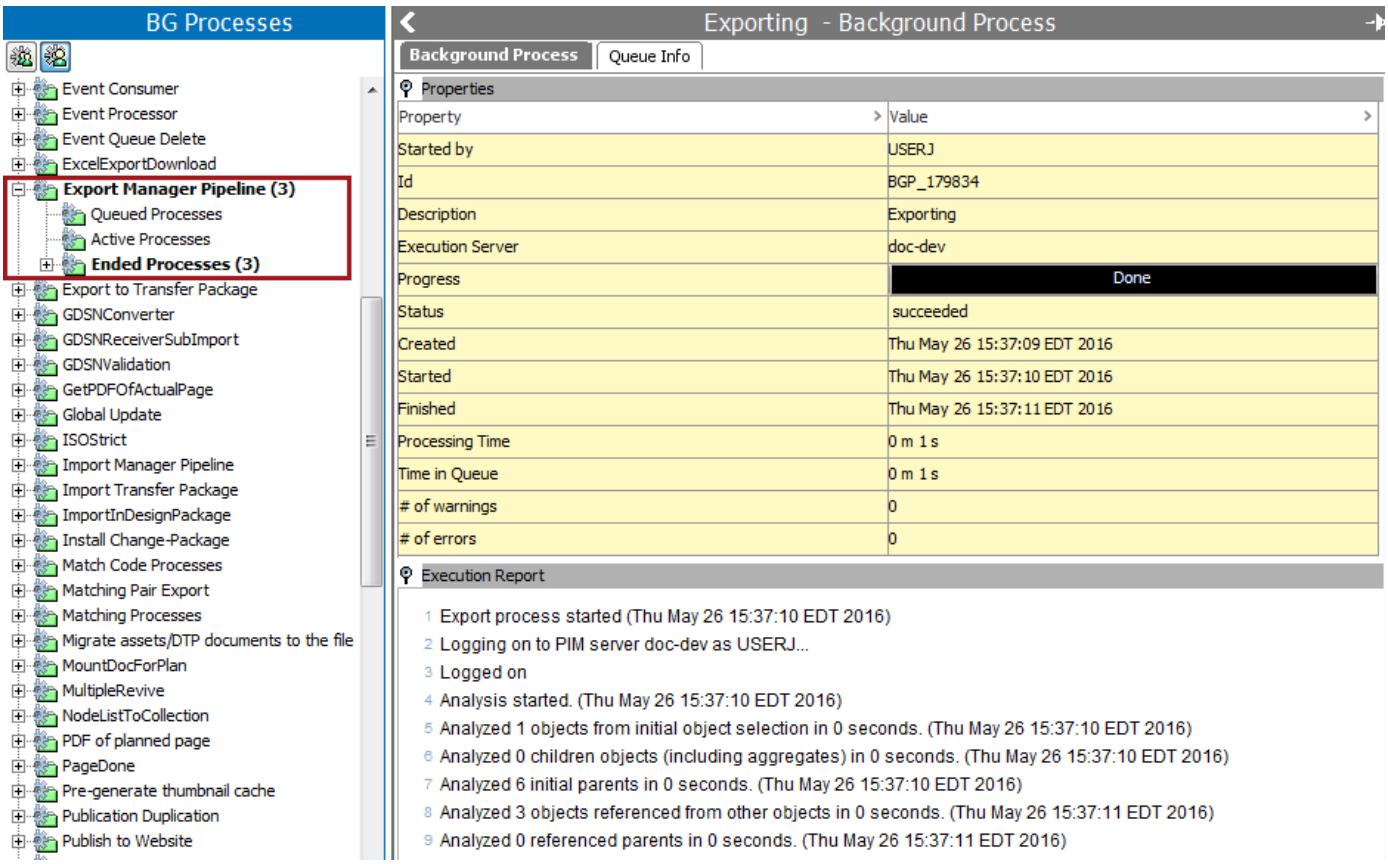
Monitoring a data export allows users to view the process' activity and status, take action based on the process' result, analyze the execution report for details, and truncate the execution report to eliminate unneeded information.

Monitoring the Export Process

There are two ways to monitor a particular data export process:

- In the **Starting Process** window (that appears when you launch the export), click the **Go To Process** button.
- On the **BG Processes** tab select the **Export Manager Pipeline** node. 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, look in **Ended Processes** folder.

The Properties flipper includes details about the export process.



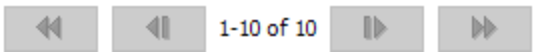
Execution Report

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:

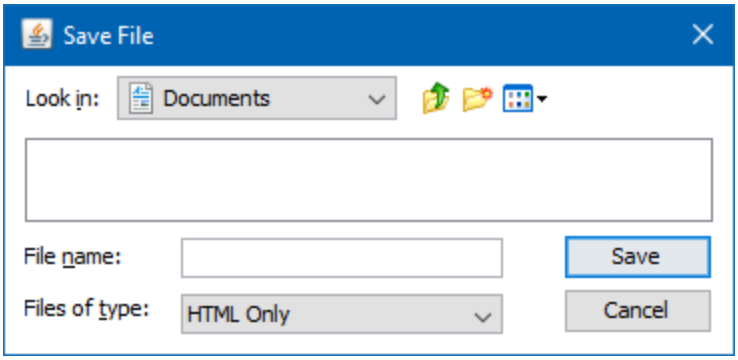


The buttons are enabled when until the log reaches over 100 lines.

- 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.



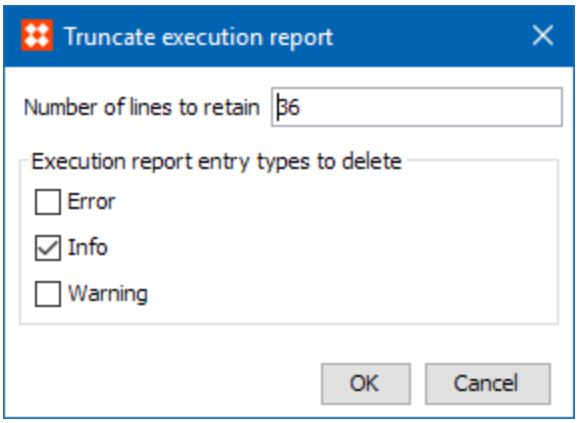
Note: Changing the default format manually may result in an error.

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.

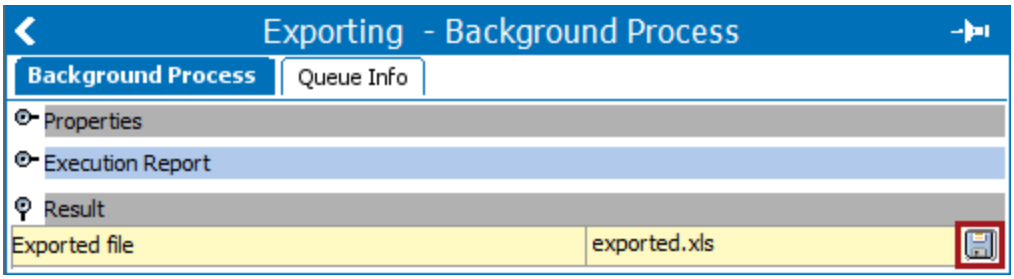
Result

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.

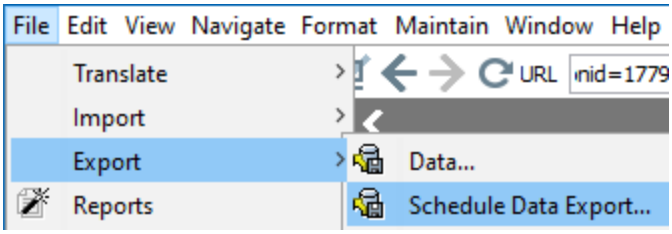
Scheduling a Data Export

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.

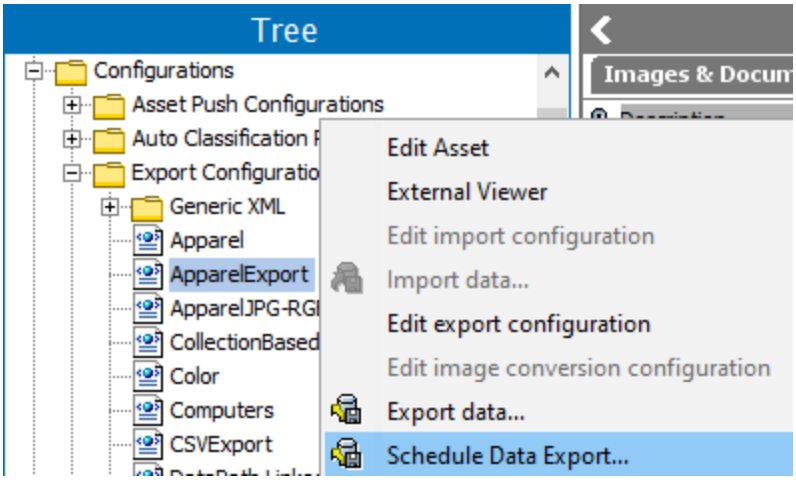
Important: Consider the time zone of the application server compared to that of the workbench (the client) where the schedule is created or viewed. When scheduling a job, the local time zone is displayed in the workbench, but the time zone of the server is used to run the background process. Although 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

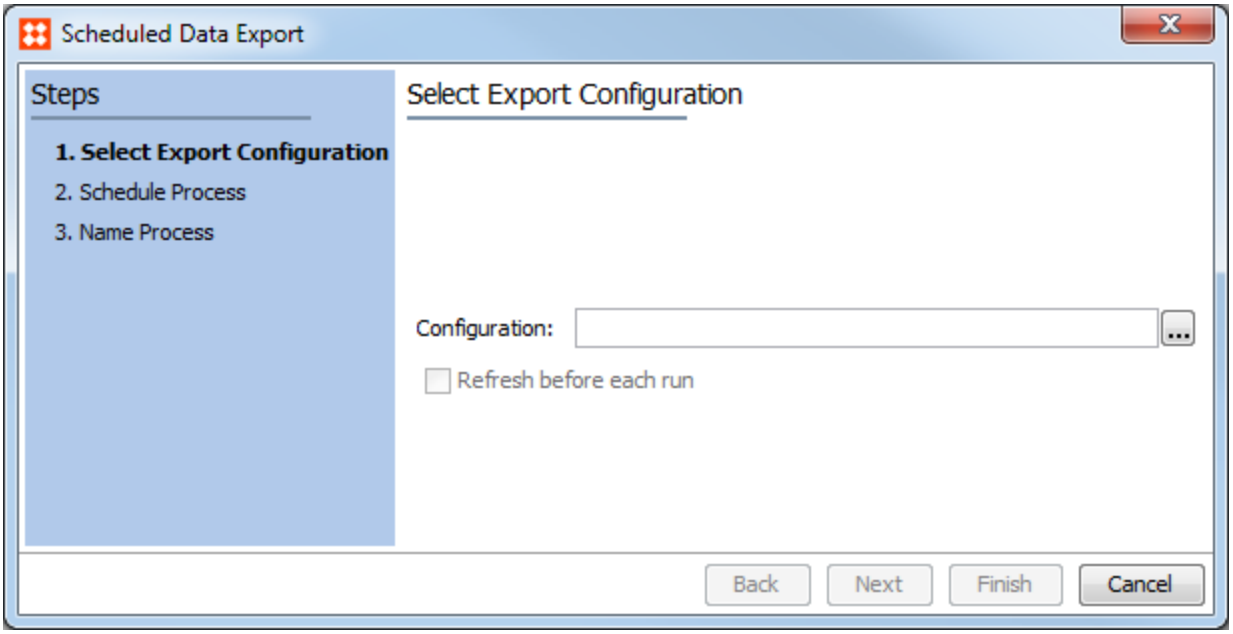
1. Select the appropriate context in STEP.
2. Verify the required Export Configuration already exists for the export you want to schedule. To create a new one, use the steps described in the **Creating a Data Export** documentation and then save the configuration as described in the **Running a Data 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**.



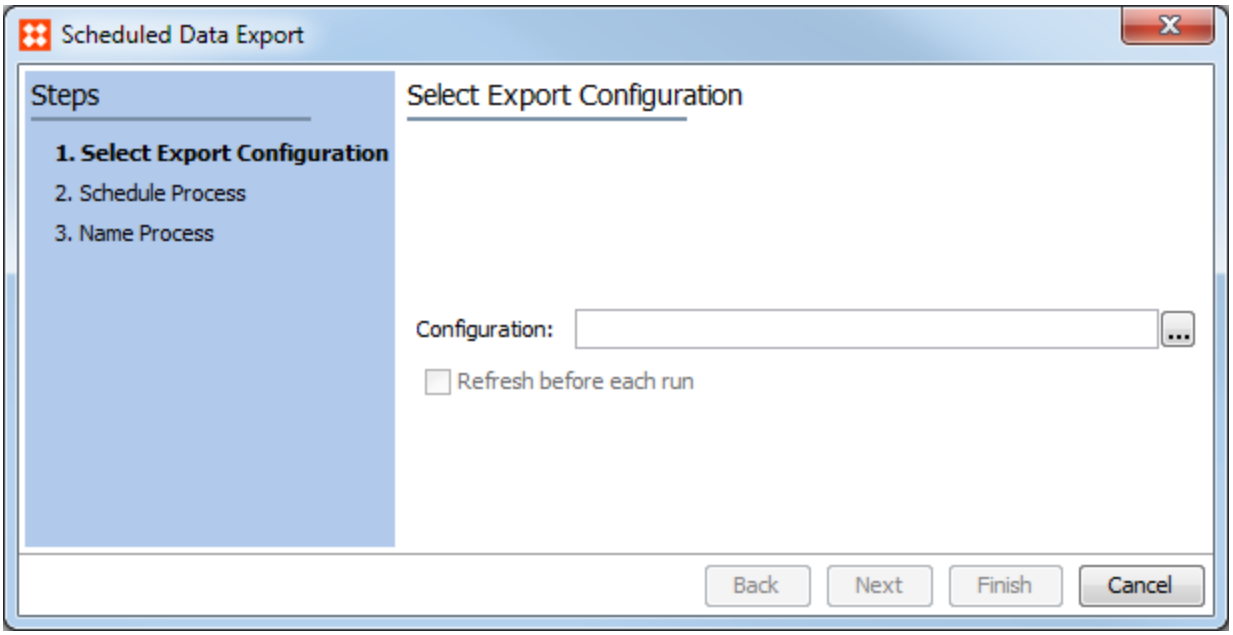
4. After selecting one of the above options to start the Schedule Data Export wizard, the following dialog is displayed and includes these steps:



- **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.

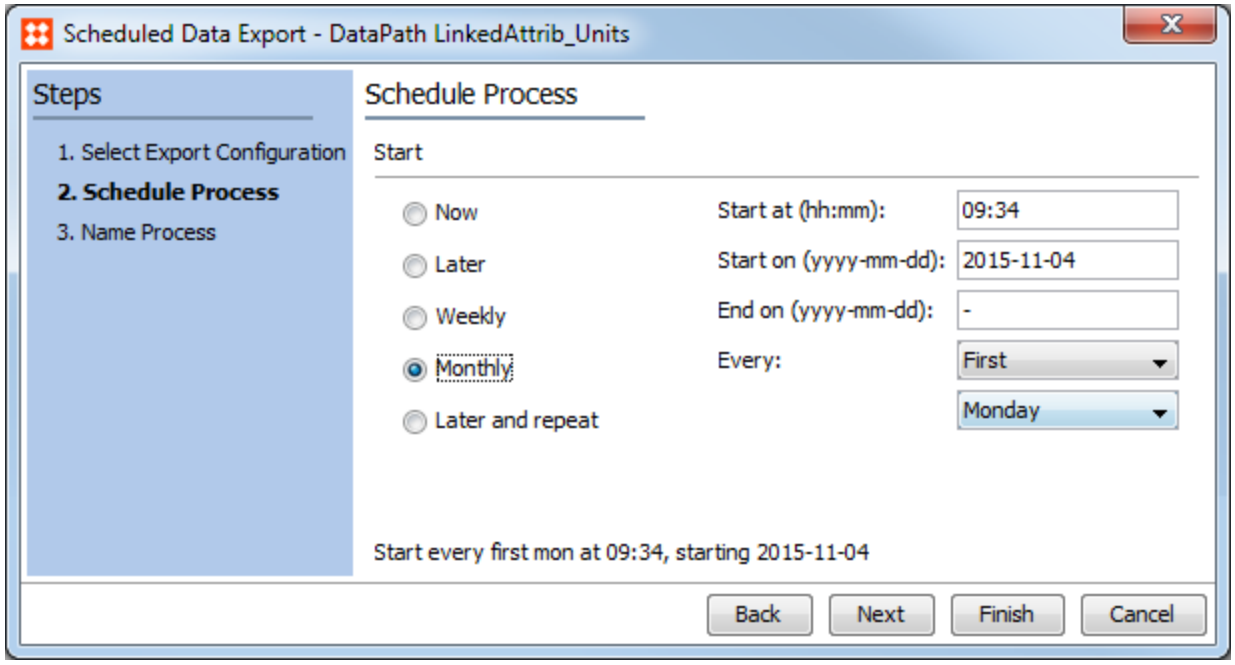
5. Manage scheduled data exports using the following information:
   - **Deleting a Scheduled Data Export** within the **System Setup / Super User Guide** documentation, removes a scheduled export process, which is required to run current parameters when the original configuration is modified.
   - **Scheduled Process Properties** within the **System Setup / Super User Guide** 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, see 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



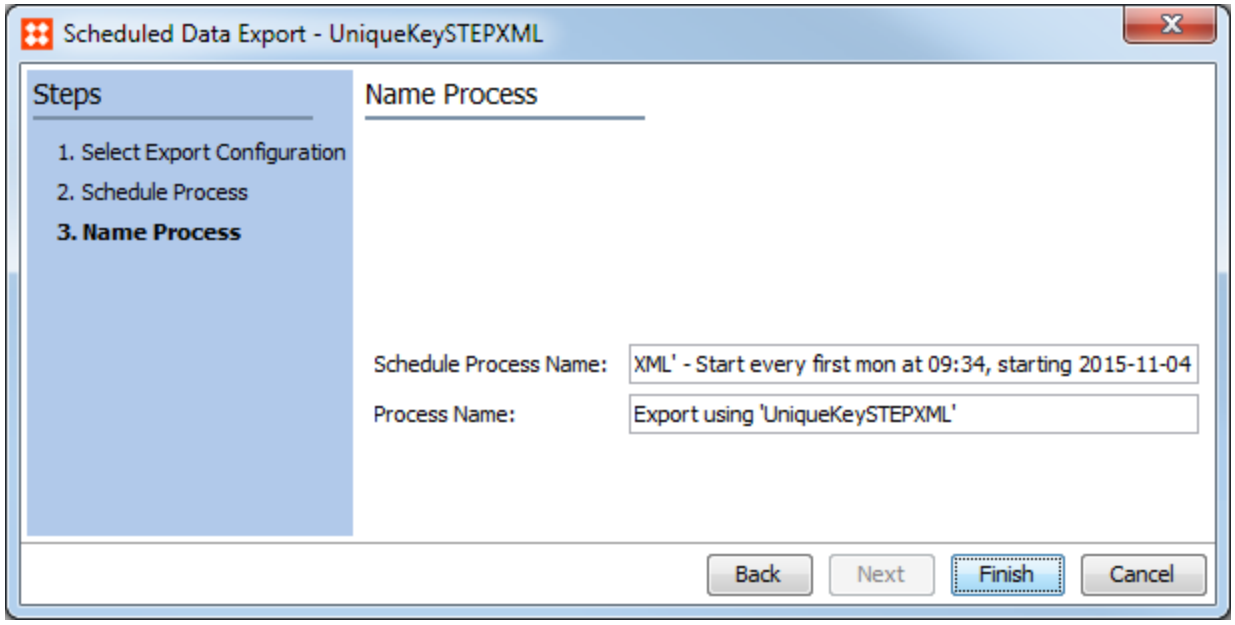
Important: Consider the time zone of the application server compared to that of the workbench (the client) where the schedule is created or viewed. When scheduling a job, the local time zone is displayed in the workbench, but the time zone of the server is used to run the background process. Although 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. See **Deleting a Scheduled Data Export** within the **System Setup / Super User Guide** 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, see the **Scheduled Process Properties** section within the **System Setup / Super User Guide** 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, click the **STEP API Documentation** button on the STEP 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:

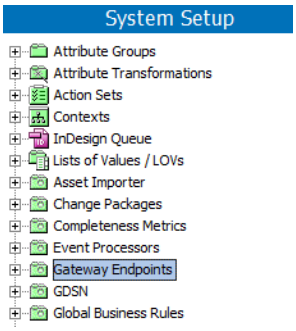
1. Maintain or modify the endpoint as described in **Maintaining a Gateway Integration Endpoint**.
2. Monitor the endpoint as described in **Monitoring a Gateway Integration Endpoint**.
3. Export gateway integration endpoint definition for comparison purposes in an external source control system in **Exporting Gateway Integration Endpoint Definitions as Comments**.

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, see the **Action Sets** section and the **Users and Groups** section in the **System Setup / Super User** documentation.

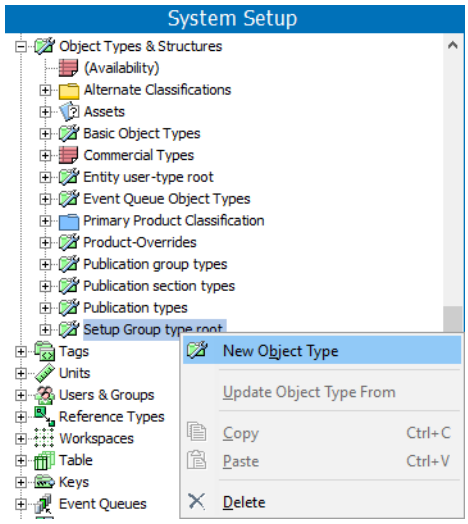
Review your System Setup tab to determine if one or more gateway integration endpoint nodes already exist. The name of the node on your system should include the word 'gateway' but is not required to match the one in the image below.



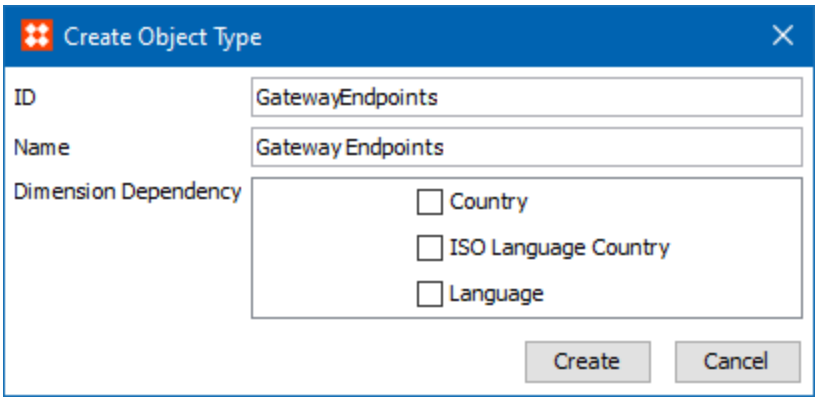
Once the setup has been completed, the steps in this section are only needed if additional levels of organization are desired.

Create the Gateway Setup Group

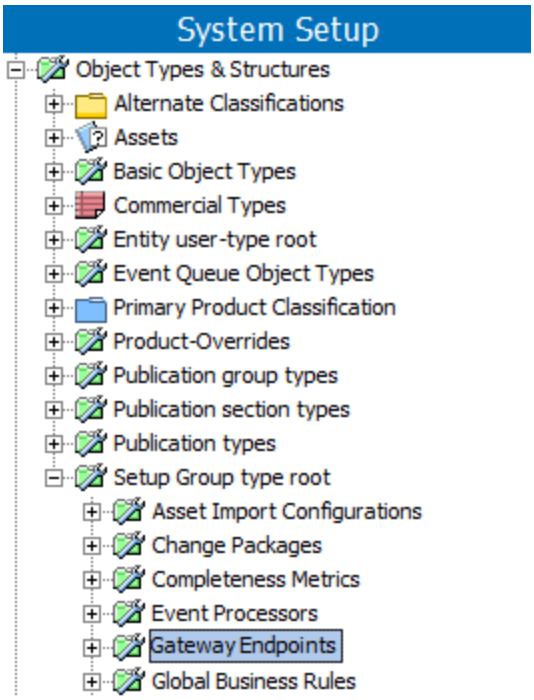
1. In System Setup, expand **Object Types & Structures**.
2. Right-click 'Setup Group type root', and choose **New Object Type**.



3. Enter an **ID** and a **Name**, select any required Dimension Dependencies, and click **Create**.

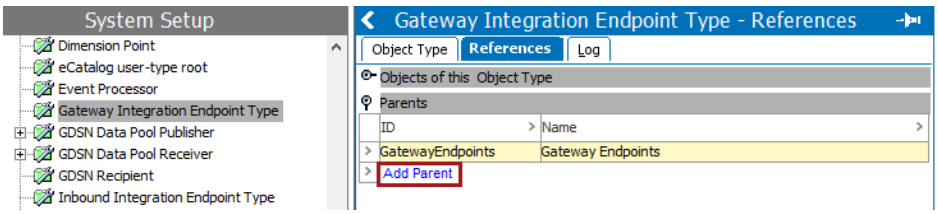


4. The new setup group appears in System Setup under 'Object Types & Structures' as a child in the **Setup Group type root**.

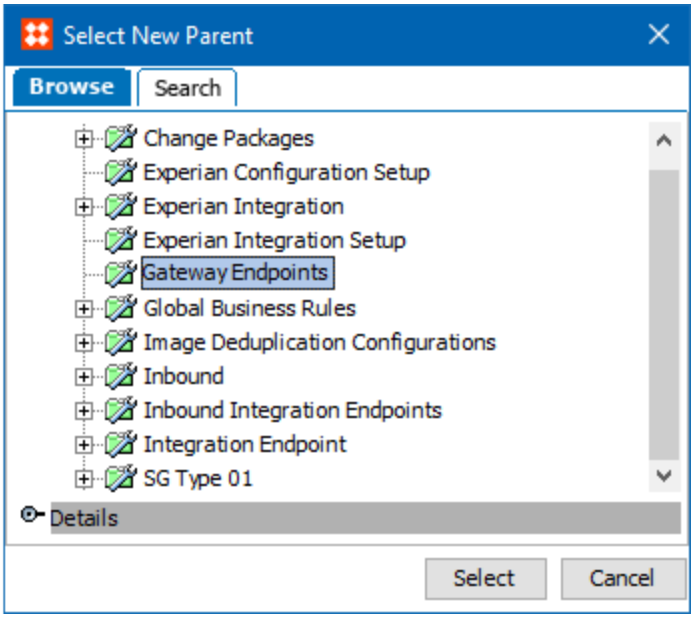


Link the Gateway Object Type to Setup Group

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.

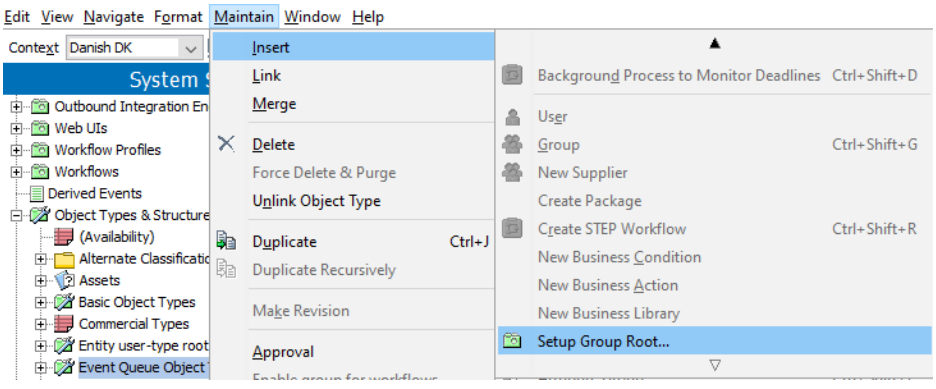


3. In the Select New Parent dialog, select the setup group you created, and click **Select** to make it a valid parent.

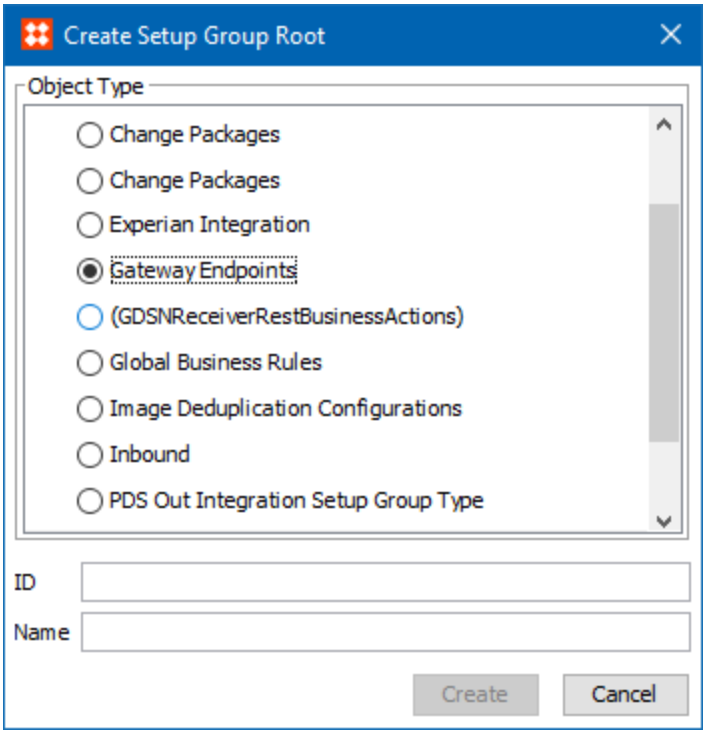


Create an Instance of the Gateway Endpoint Object

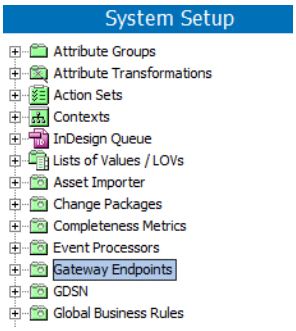
1. On the System Setup tab, select any object in the **System Setup** hierarchy to activate the following Maintain menu selection.
2. Click the Maintain menu, point to Insert, and select **Setup Group Root**.



3. In the Create Setup Group Root dialog, select the gateway object type, enter an **ID** and a **Name**, and click **Create**.



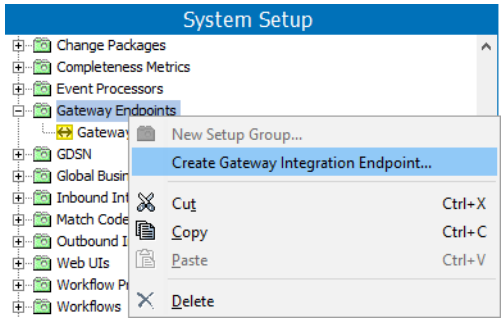
A setup group is created as a node in the System Setup hierarchy. Gateway integration endpoints can now be created under this new node.



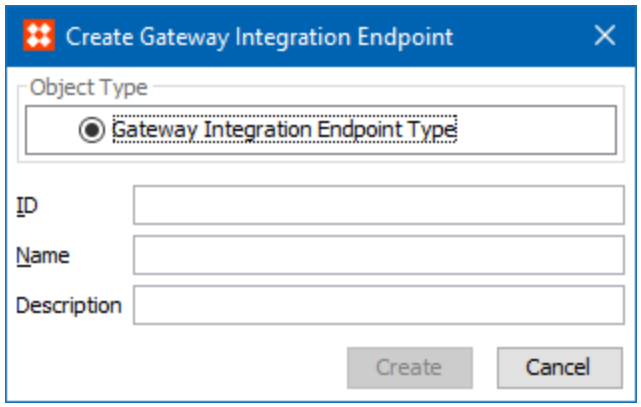
Creating a Gateway Integration Endpoint

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.

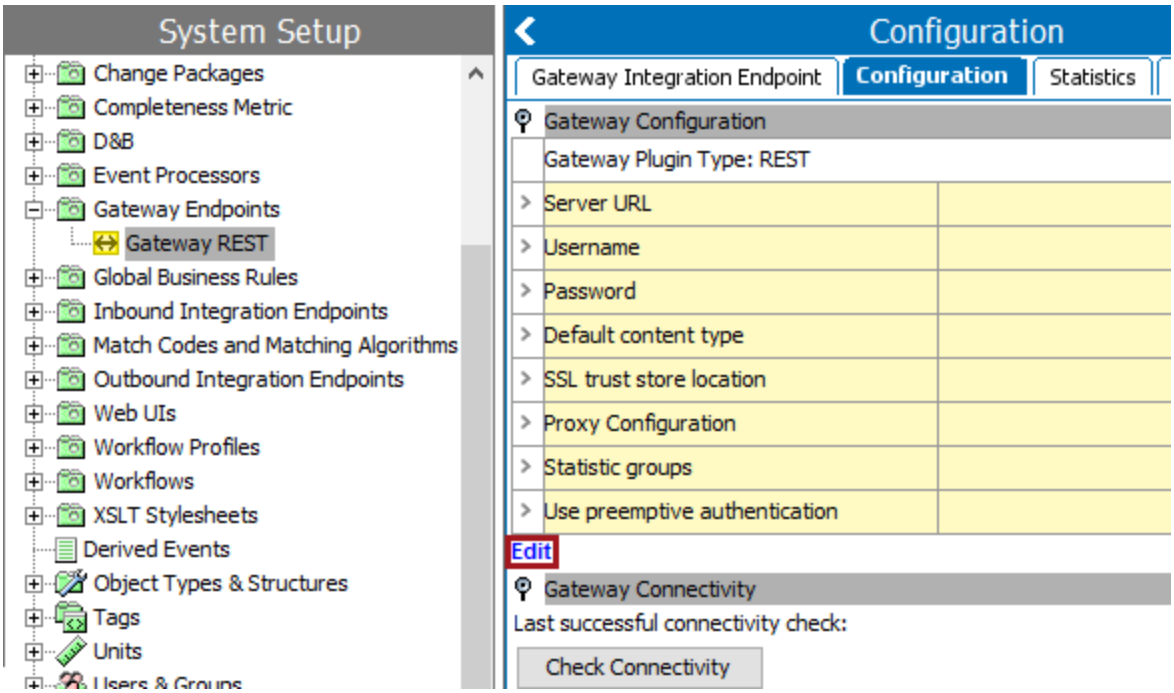


3. Click **Create** and then configure the endpoint. For more information, see the **Configuring a Gateway Integration Endpoint** documentation.

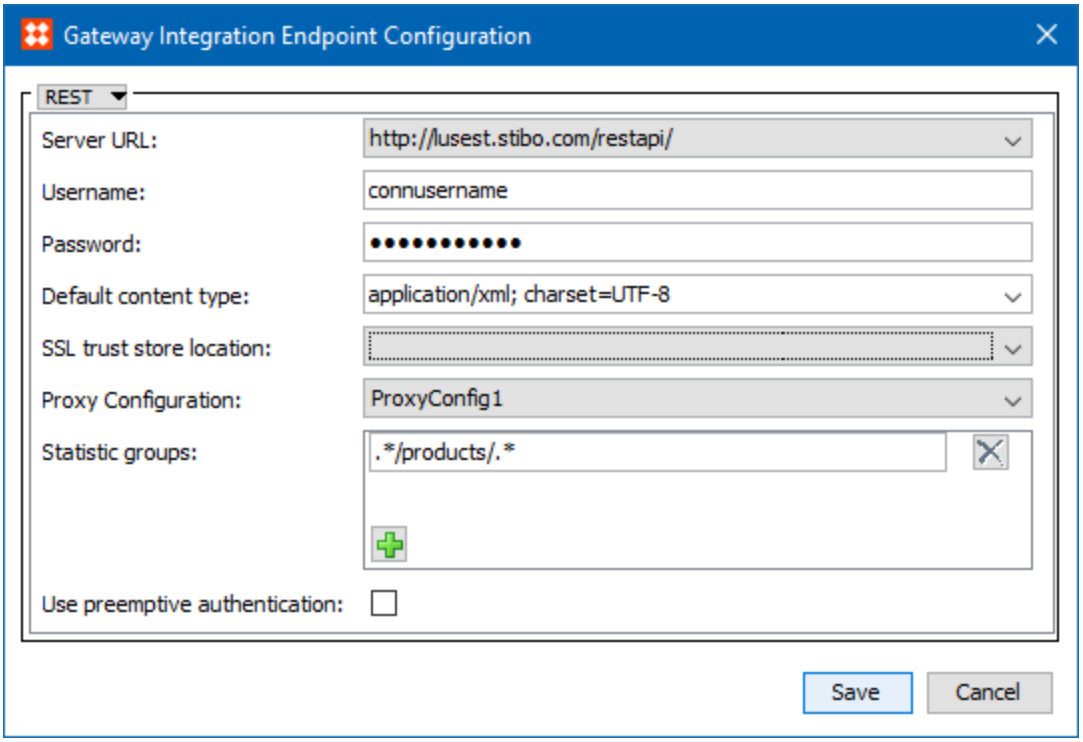
Configuring a Gateway Integration Endpoint

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, use the following parameters to specify which external system the gateway integration endpoint will access.



- **Server URL** - The URL of the server to be accessed. The available URLs are system dependent and must be configured in the sharedconfig.properties file using the 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
  ```

- **Username** - Login required to establish the connection.
- **Password** - Password required to establish the connection.
- **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. To display a trust store in the dropdown, first update the 'RESTGateway.SSLTrustStoreLocation' property in the sharedconfig.properties file. Contact Stibo Systems technical support for assistance in adding a trust store to your system.

  If no trust store locations are selected, the gateway connection is not configured to use SSL encryption and the connection will be established using a certificate signed by a recognized CA 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. The configurations that appear in this dropdown are populated from properties added into the sharedconfig.properties file using the dynamic property 'RESTGateway.ProxyConfiguration.' See the **Proxy Configuration Setup** section below for details.

- **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 **java.util.regex.Pattern** class. For more information, see the **Gateway Integration Endpoint Statistics** section or the **Regular Expression** topic in the **Resource Materials** online help.
- **Use preemptive authentication** - Optional. Check this box to allow privileged users to grant this gateway the ability to override the default usage of normal basic authentication and use preemptive authentication instead. With preemptive authentication, every time the gateway binding invokes a request (GET, PUT, POST, etc.), the authentication header containing the user name and password will be applied to the HTTP request.

4. Click **Save** to complete the configuration.
5. Create a business action to access the gateway endpoint. For details, see **Gateway Integration Endpoint Bind** in the **Resource Materials** online help.

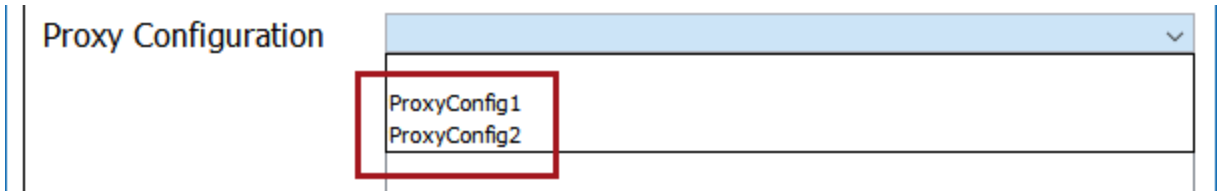
Proxy Configuration Setup

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 will appear in the dropdown as a selectable option, like so:

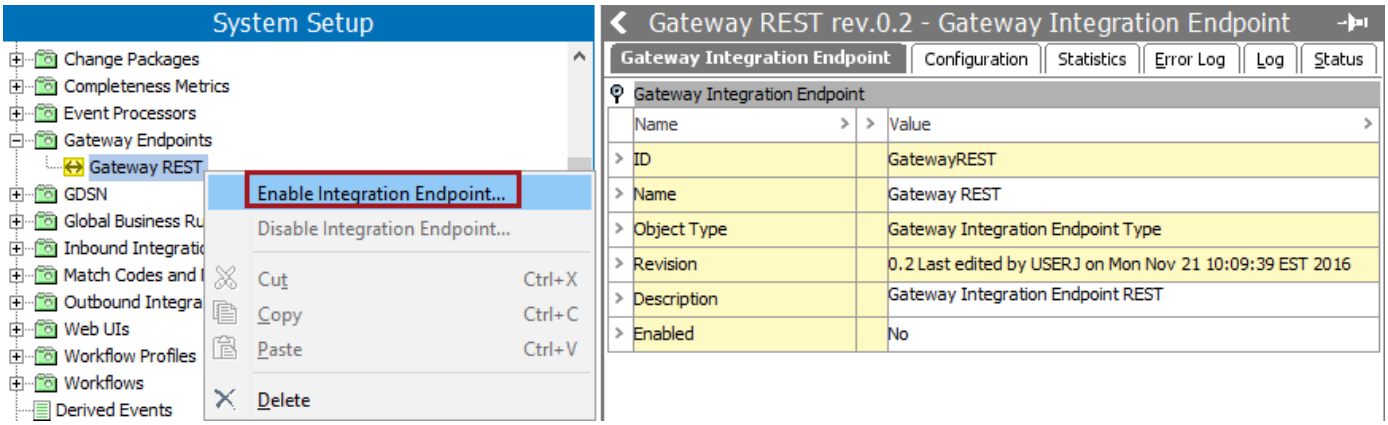


3. The IP of the proxy server being accessed
4. The port of the proxy server being accessed
5. The username for the proxy server being accessed (optional)
6. The password for the proxy server being accessed (optional)

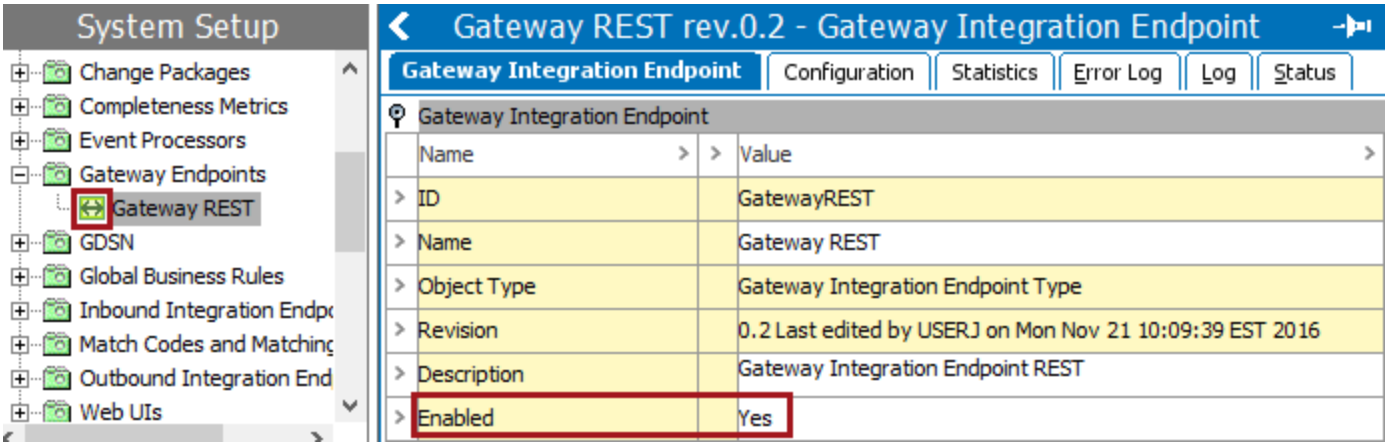
Running a Gateway Integration Endpoint

After configuring a gateway endpoint, it must be enabled and connected before it can run.

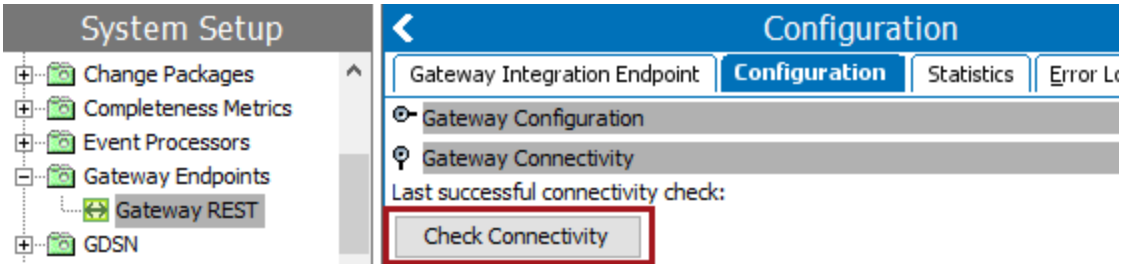
1. On System Setup expand the Gateway Endpoints node to display all existing gateway endpoints.
2. Select a disabled gateway endpoint (indicated by the yellow gateway icon meaning it has been disabled by user request), right-click, and select **Enable Integration Endpoint** from the menu.



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.'

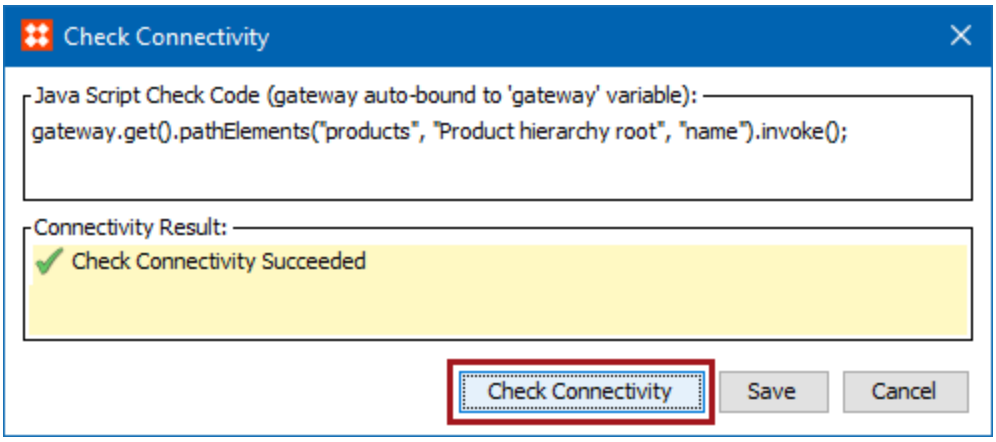


3. On the Configuration tab under the Gateway Connectivity flipper, click the **Check Connectivity** button.



4. In the **Check Connectivity** dialog, add JavaScript to access the external system.

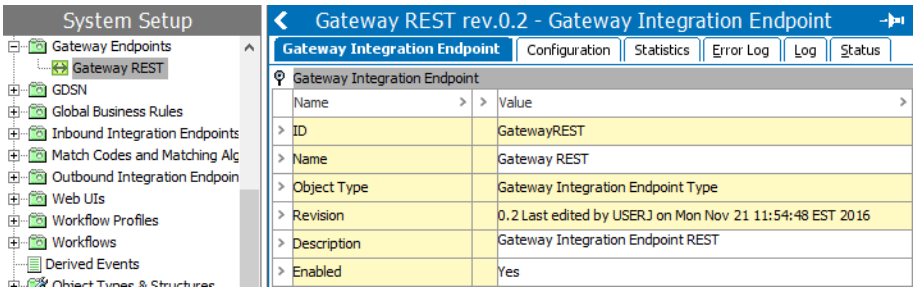
   In the following example, the JavaScript gets the name of the "Product Hierarchy root." For information about the JavaScript syntax, click the **STEP API Documentation** button on the STEP Start Page.



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, see the **Configuring a Gateway Integration Endpoint** section.
- **Statistics**: provides statistics about frequency and timings of external calls using the gateway. For more information, see the **Gateway Integration Endpoint Statistics** section.
- **Error Log**: shows errors reported from external calls using the gateway. For more information, see 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.

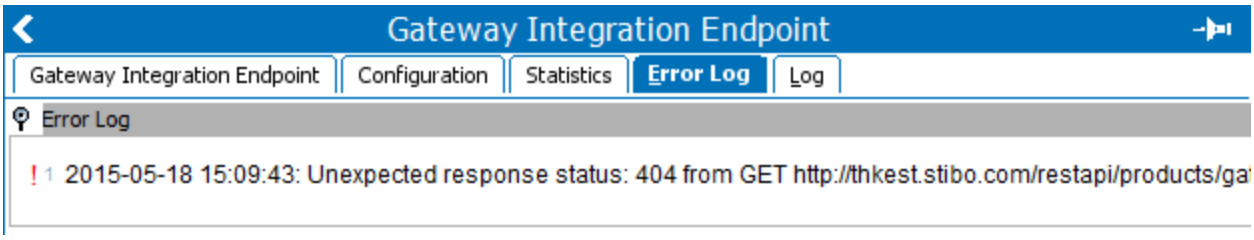
Monitoring a Gateway Integration Endpoint

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 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

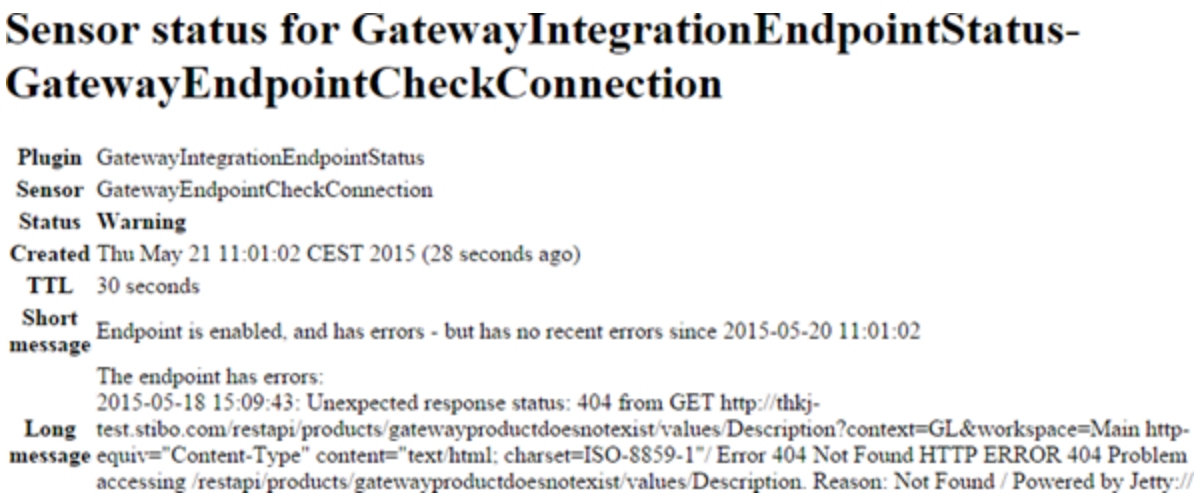
Use the following URL, replacing <appserver> with your own application server name, and replacing <Gateway-ID> with the STEP ID of the desired endpoint. The ID is displayed in the gateway integration endpoint editor. For more information, see **Maintaining a Gateway Integration Endpoint**.

```
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.'

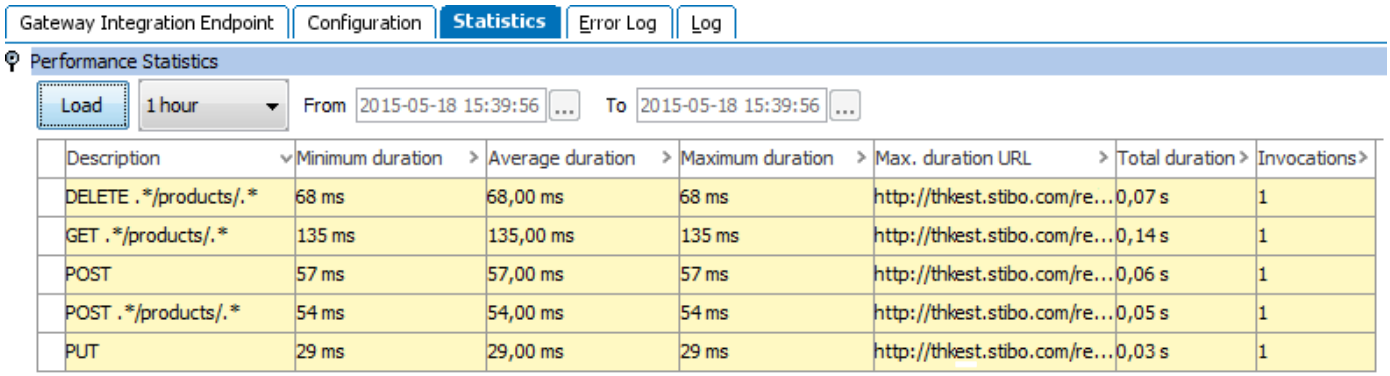


Sensor status for GatewayIntegrationEndpointStatus-GatewayEndpointCheckConnection

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, see the **Statistic groups** section of the **Configuring a Gateway Integration Endpoint** documentation or the **Regular Expression** topic in **Resource Materials** online help.

Exporting Gateway Integration Endpoint Definitions as Comments

Integration endpoint definitions can be exported as comments using Advanced STEPXML. These exports are intended to be used for submission to external source control systems for comparison purposes. Users can import them into source code repository systems where they can be compared from version to version. Editing and/or importing of these files is not supported (e.g., users may not export, edit the comments, and re-import in STEP).

To export an integration endpoint definition for external comparison, use Advanced STEPXML and set the DefinitionsAsComments tag to 'true', as shown in the following sample template:

```
<?xml version='1.0'?>
<STEP-ProductInformation DefinitionsAsComments="true">
<OutBoundIntegrationEndpoint ExportSize="All"/>
</STEP-ProductInformation>
```

An example of the output for an integration endpoint definition as comments is shown below:

```
<OutBoundIntegrationEndpoint...>
<!--Definition: [This will be the IEP definition. Removed for brevity. -->
[Remaining configuration is included]
</OutBoundIntegrationEndpoint>
```

Note: The content of the comment field is not part of the STEPXML XSD and therefore Stibo Systems reserves the right to change the format of the output content at any time.

Additionally, integration endpoints can be exported using STEPXML. For information, see the Configuration > **Include Integration Endpoints** section of the **STEPXML Outbound Parameters** documentation.

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 spot 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.

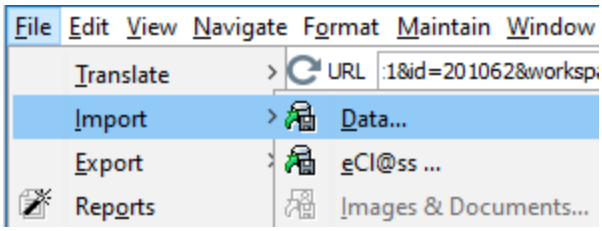
The user running the import should have the following setup privilege action sets: Create domain, Create value in hard domain, Create value in medium domain, Delete domain, Merge entire domain, and View domain. For more information, see **Action Sets** in the **System Setup / Super User Guide** documentation.

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

To import data using the Import Manager:

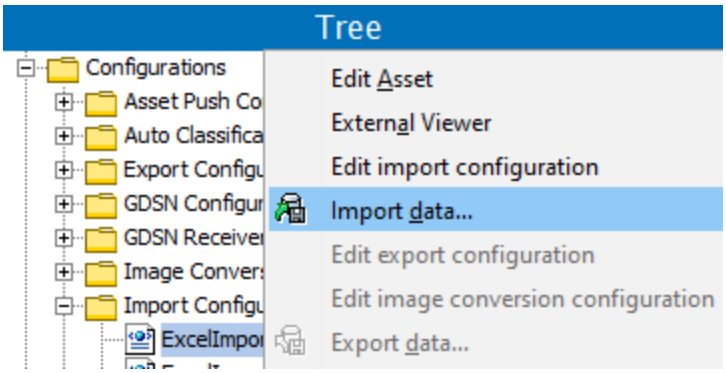
1. Verify that all attributes and reference / link types being imported already exist in STEP.
2. Use one of the following methods to launch the Import Manager wizard, noting that using an import configuration pre-loads the import file and settings.
   - **From the File Menu**: Click the File menu > Import > **Data**.

     The wizard opens in Step 1 and allows you to select an existing configuration or create a new one.

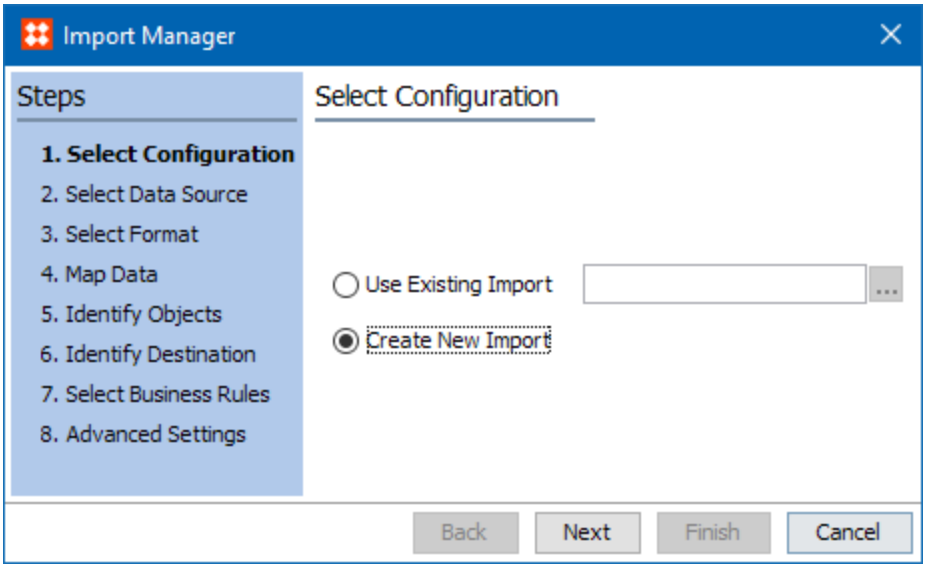
     

   - **From an Import Configuration**: On the Tree, select an import configuration, right-click, and then select **Import Data**.

     The wizard opens in Step 2 with the configuration and objects included in the configuration already selected.

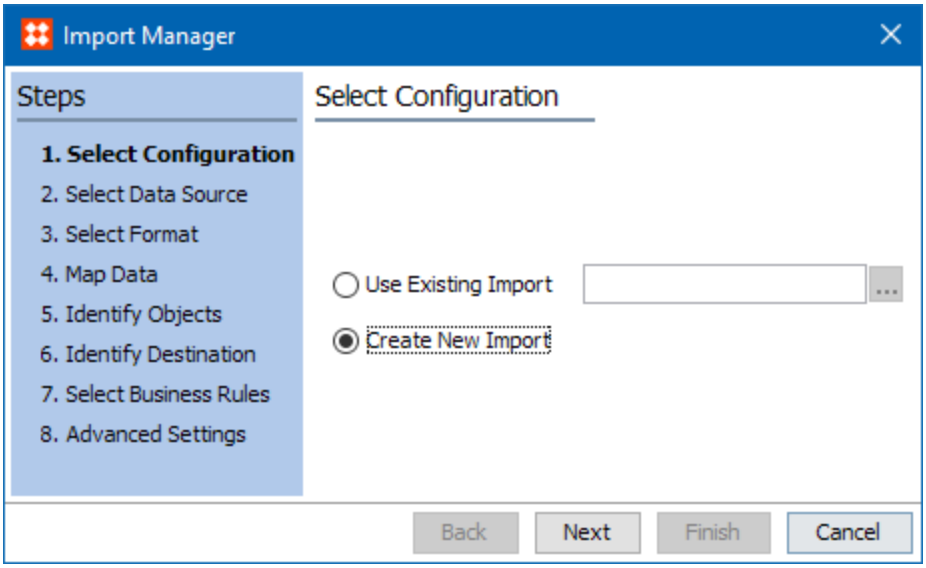
3. The Import Manager wizard displays and can involve the following steps:



- **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

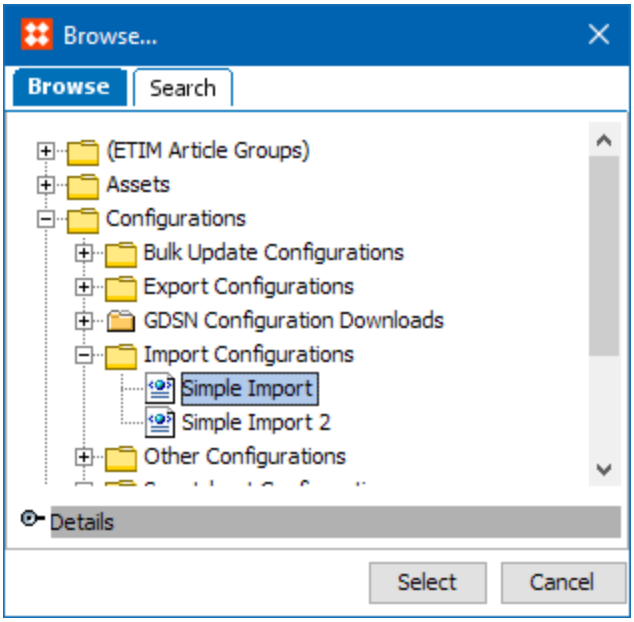


Once created, an import configuration can be saved as an asset in STEP for use with other import files. A configuration file contains an XML representation of the parameter settings in Import Manager.

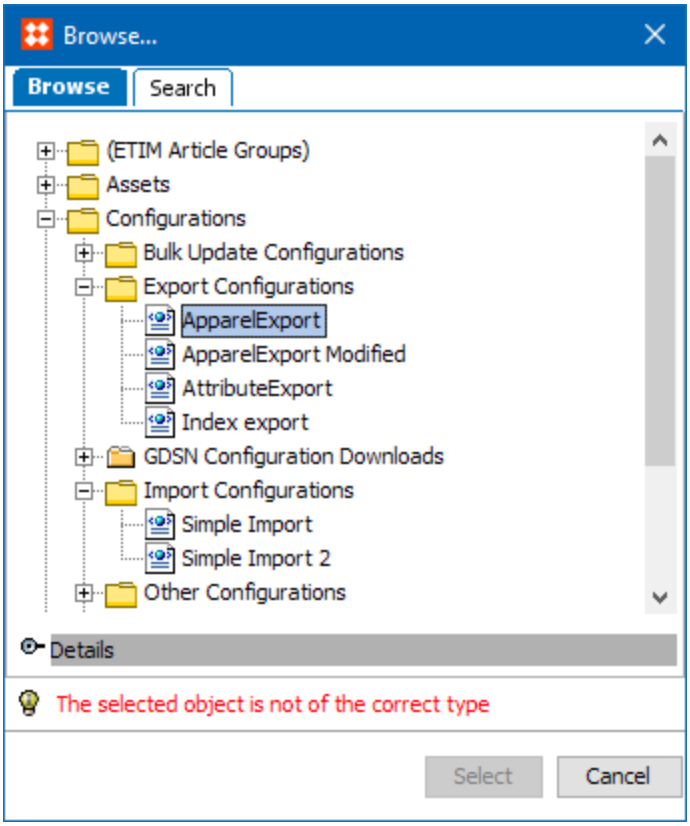
1. Select a configuration option:
   - **Use Existing Import** - allows you to select a previously saved configuration. Click the ellipsis button (...) to browse or search for the configuration in STEP.
   - To create a new configuration, Select **Create New Import**.
2. Click **Next** to display **Import 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.

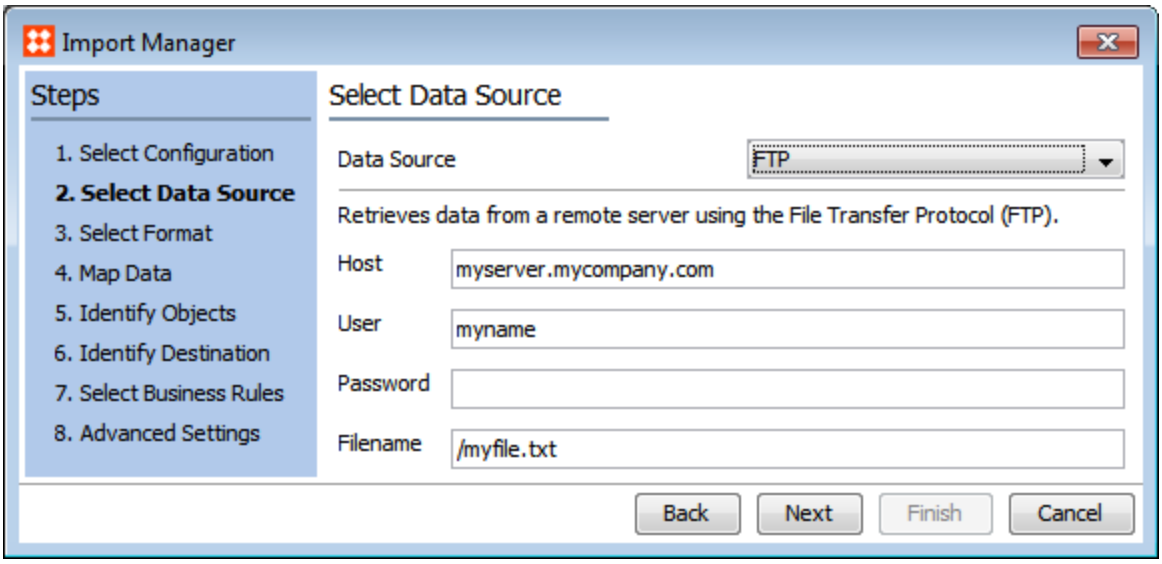


Import Manager - Select Data Source

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.

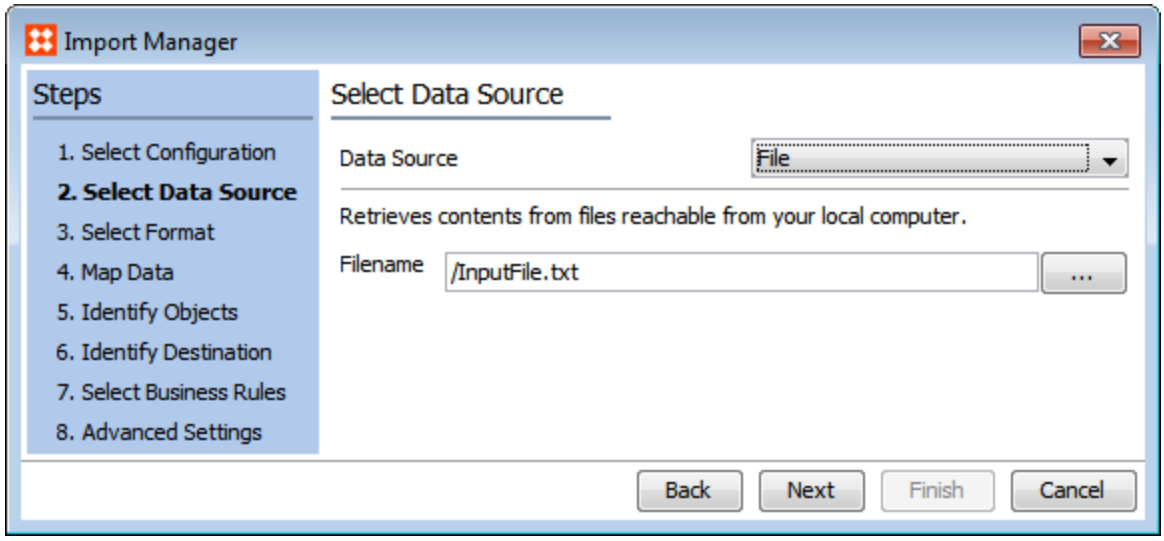
- **FTP** allows you to select a file from an FTP site.
- **File** allows you to select a file from your local hard drive or from your network.

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**.

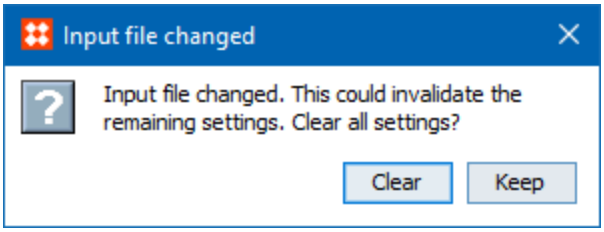
Using File



It is common practice to copy very large files to your local drive prior to importing.

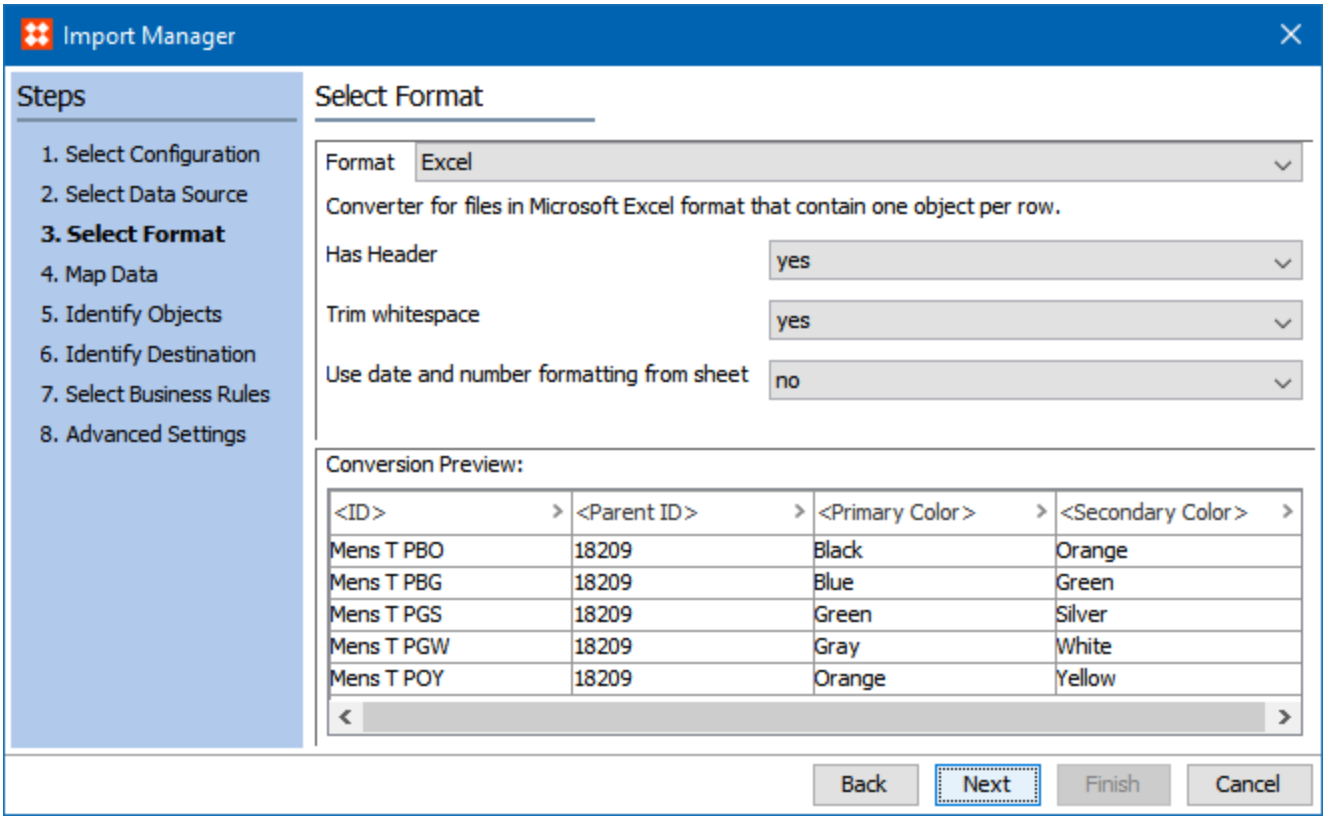
1. Select **File** from the Data Source list and click the ellipsis button (...) on the Filename field.
2. Locate the relevant file, and then click **Select File**, and then click **Next** to display **Import Manager - Select Format**.

   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.

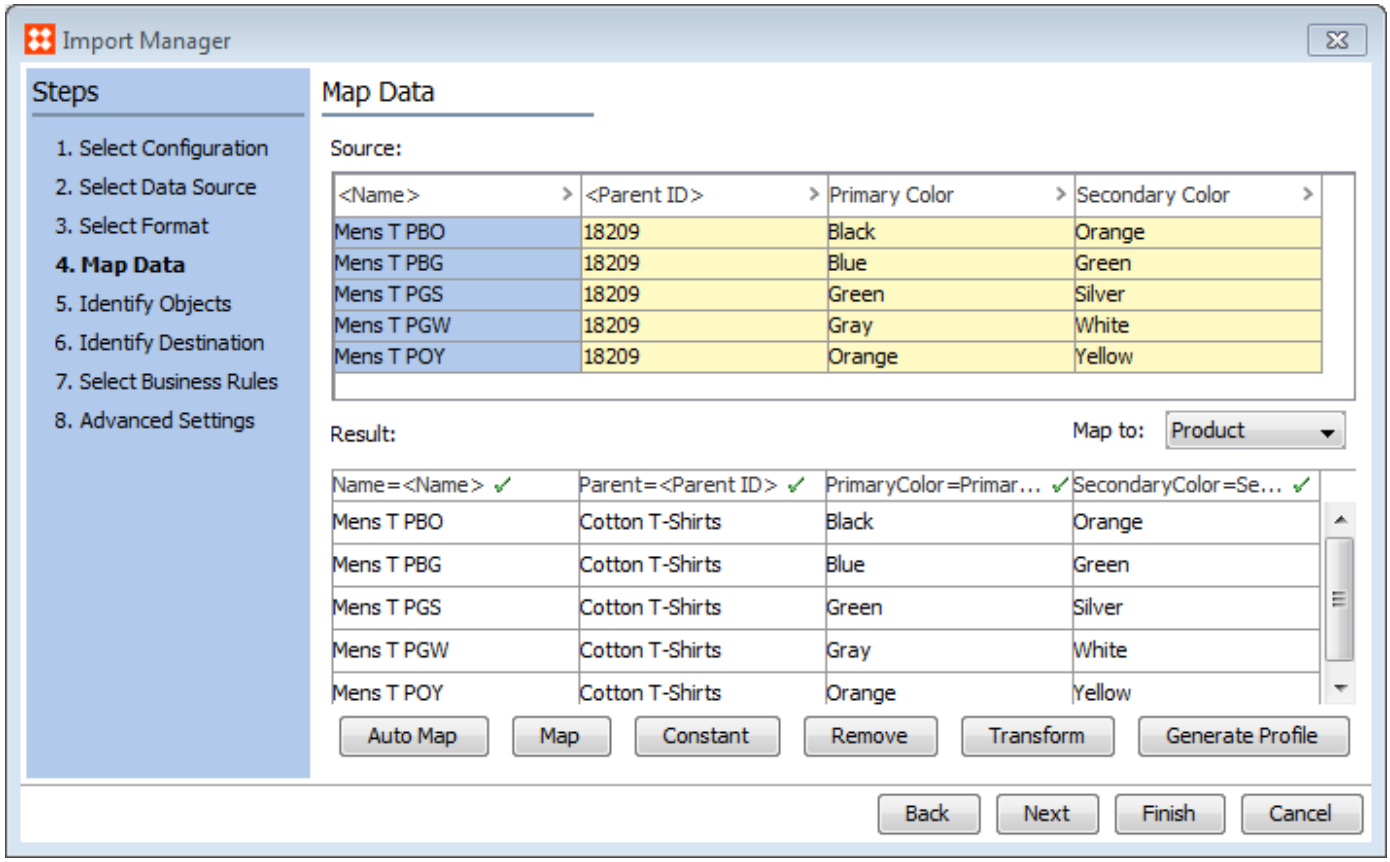
The following formats are available:

- Advanced STEPXML Format
- BMEcat Format
- BMEcat 2005 Format
- CSV Format
- eCl@ss 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



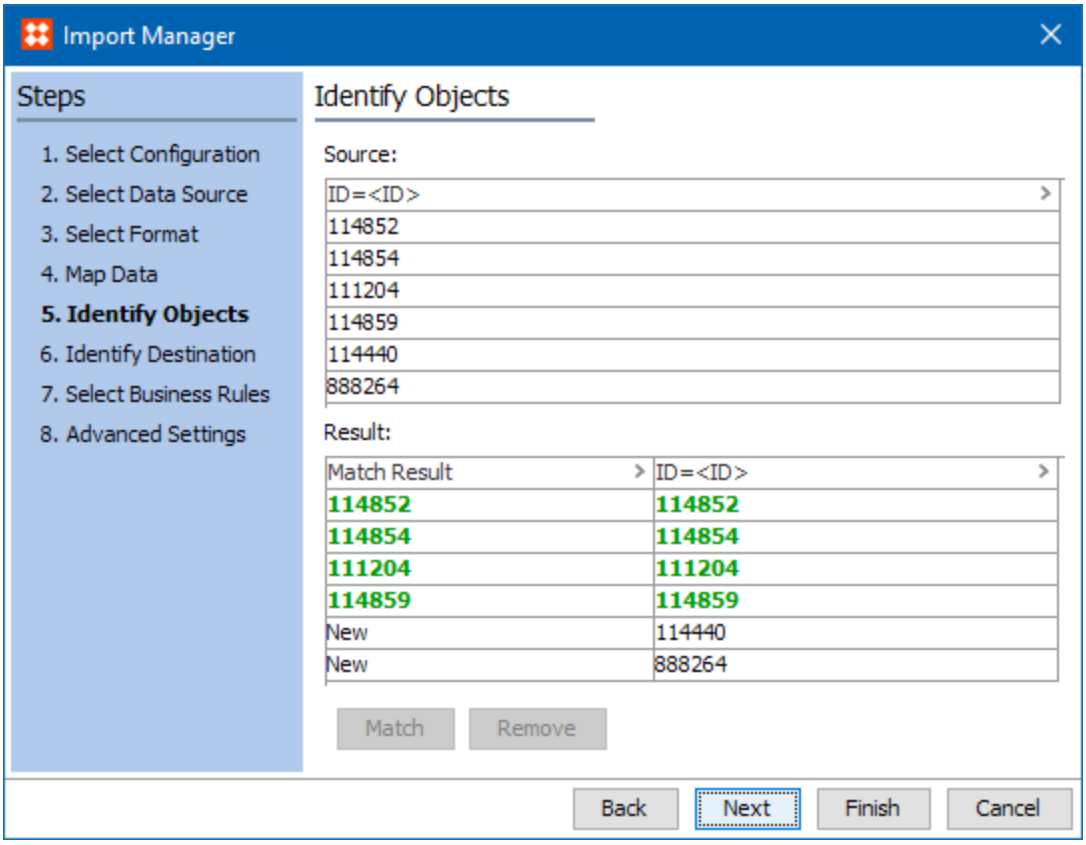
Mapping is required when importing some file types. Mapping relates the import data to the appropriate element in the STEP data model and specifies the desired object type.

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, see 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.

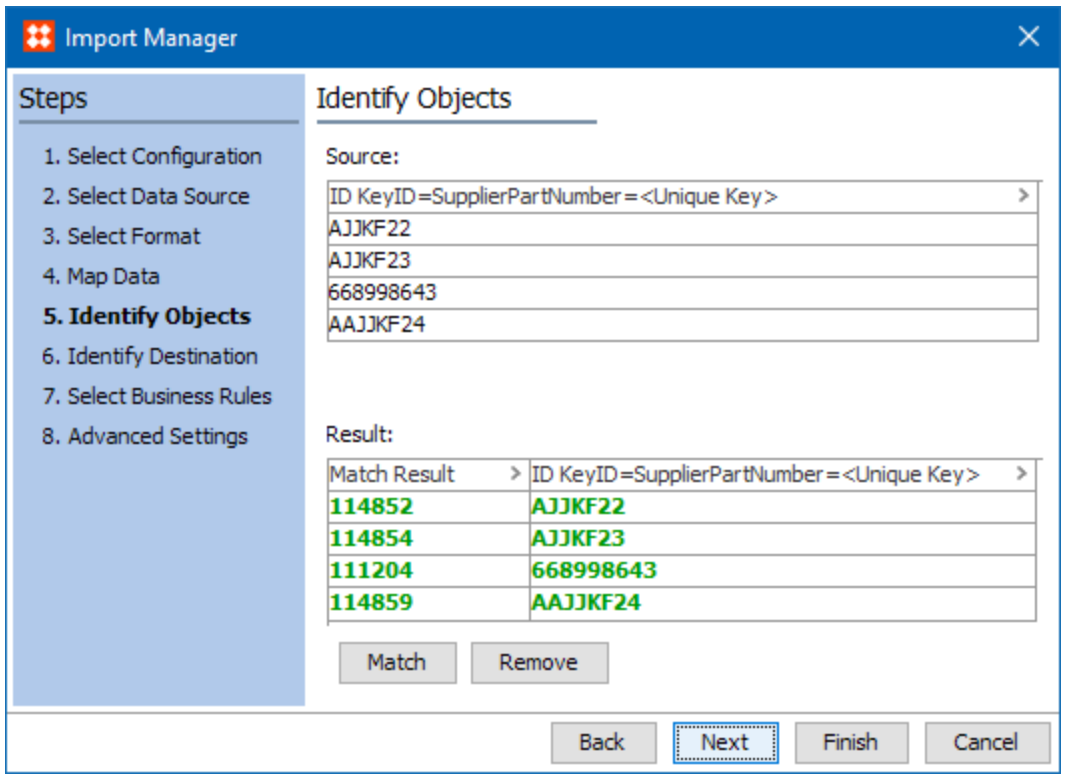
Click the Next button to display **Import Manager - Identify Destination**.

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.

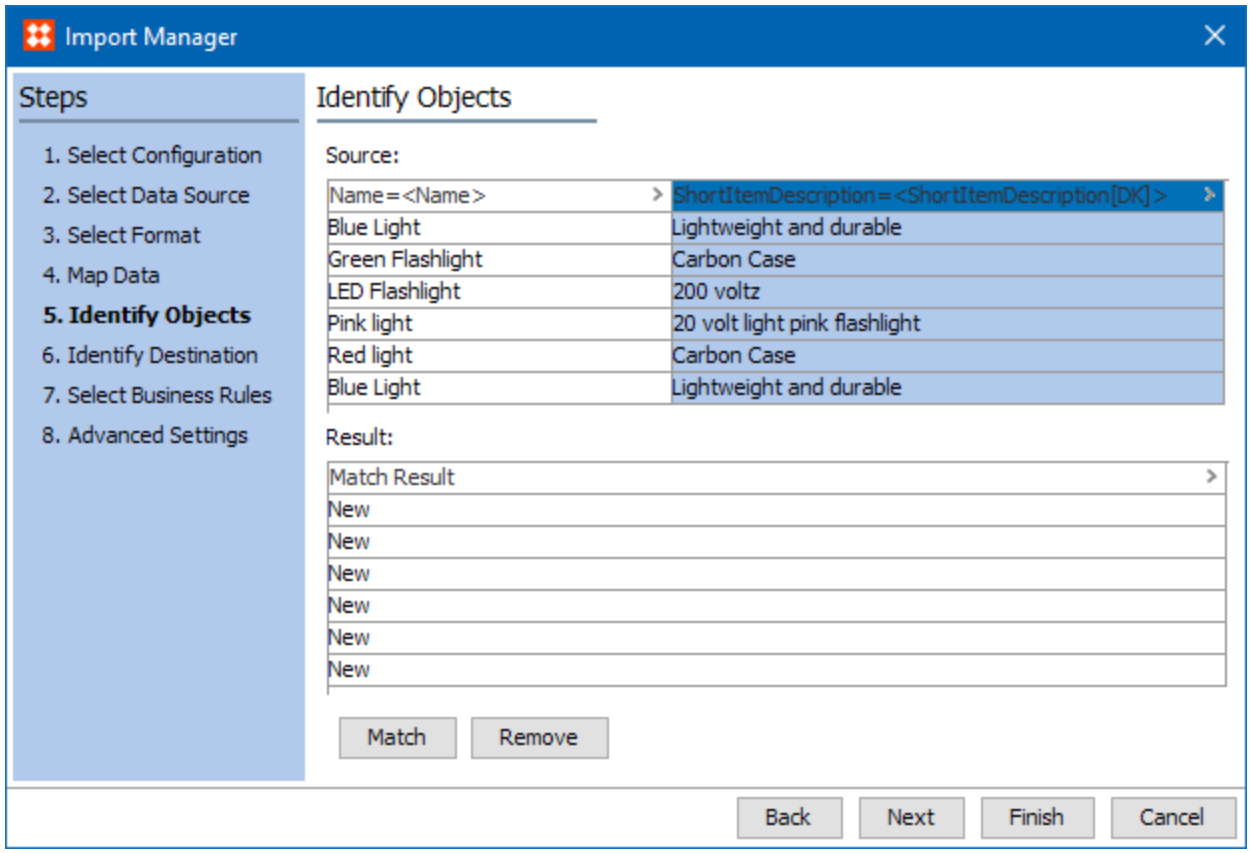
The following example describes how to identify products by attribute values. The product data is loaded from a file with no product IDs listed but contains the manufacturer's own part numbers.

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.

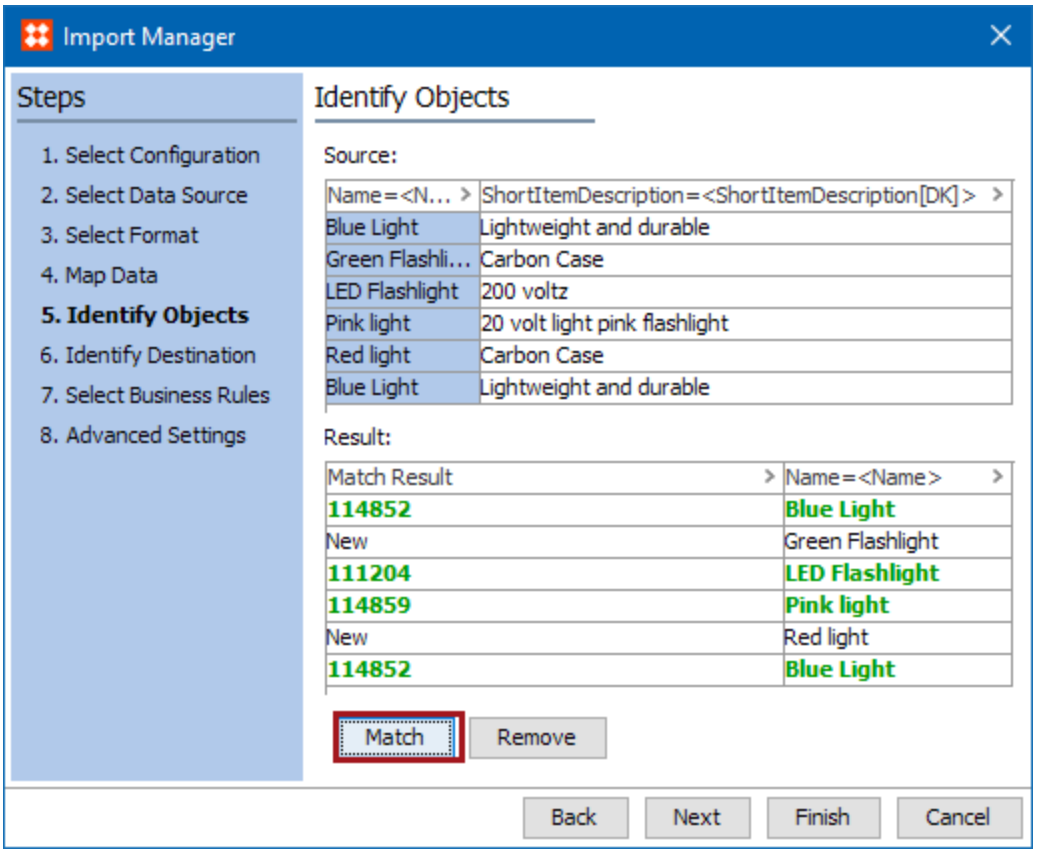
A file with two columns of data is imported to STEP. Both columns are mapped to attributes. The first one is mapped to the attribute 'Manufacturer Part Number', and the second is mapped to 'Description.'

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.

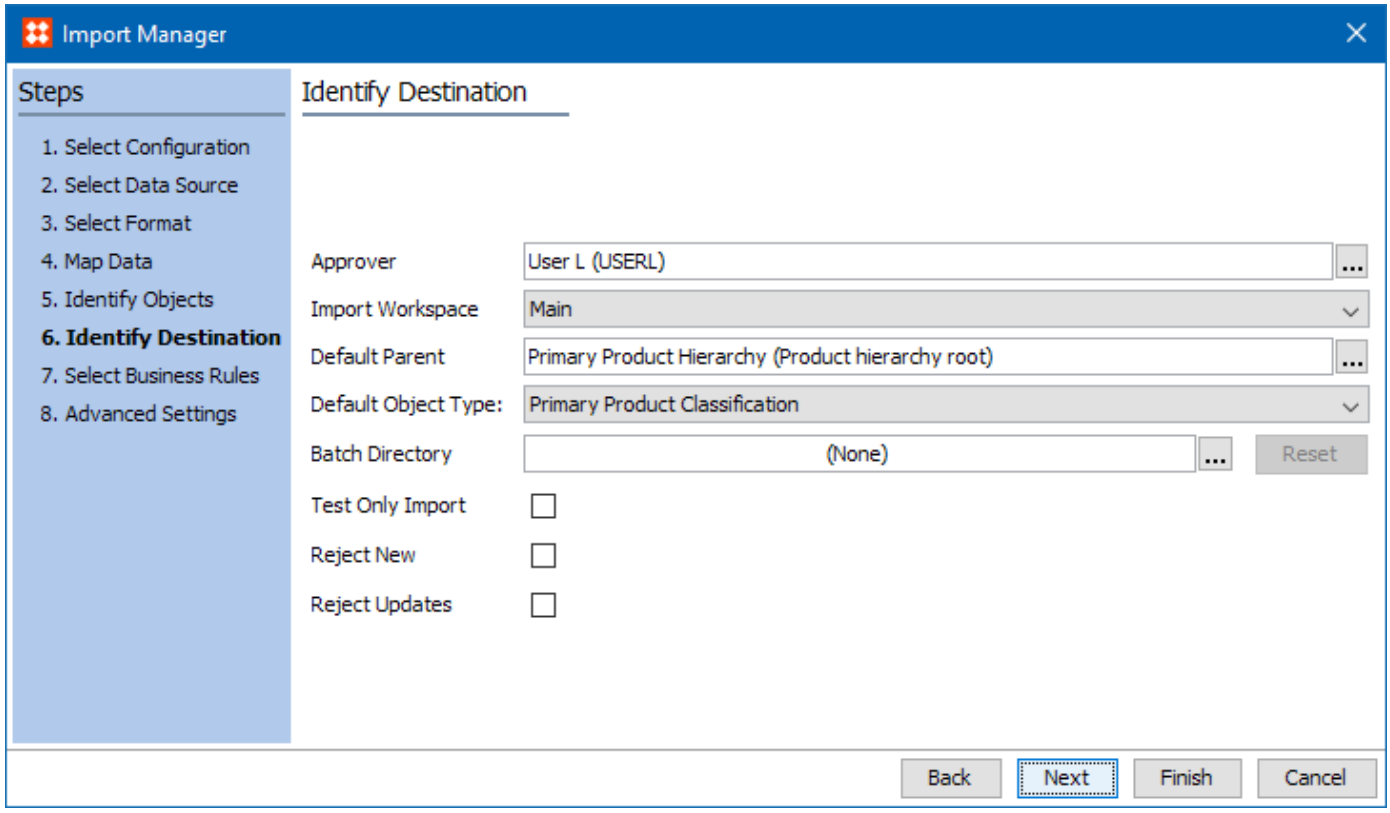


The first column is the ID of the product in STEP. The second column is the attribute value that matches that product.

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



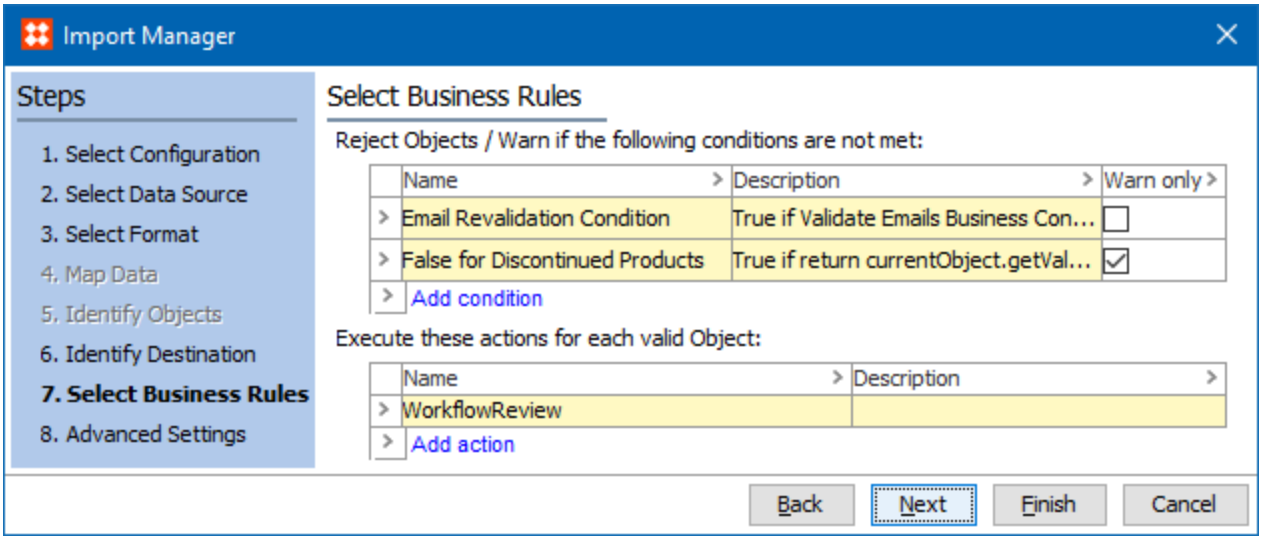
- **Approver** is used to specify the user name of the person that is allowed to approve the products after the import process is complete. When you click the ellipsis button (...), you can search or browse for the user. You are the default Approver. Normally, this is used in conjunction with a customized workflow. See also 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 Product** is used to specify where new products should be placed in STEP. **A parent is required to create new objects, entities, assets, and classifications in STEP.** Use this parameter if no column has been mapped to the Parent ID, or if the entry in the Parent ID column is invalid (for example, if the parent product listed in the input file does not exist in STEP). The import does not check all of the records in the load file. This means that if any of the records in your load file is a new product, and does not have a parent, it cannot be created in STEP. Similarly, even if you have mapped a column in the load file as the Parent ID, the import has not validated all those entries before this point. When parent data is missing in the import file, it can sometimes be derived via transformations or added 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, see 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.

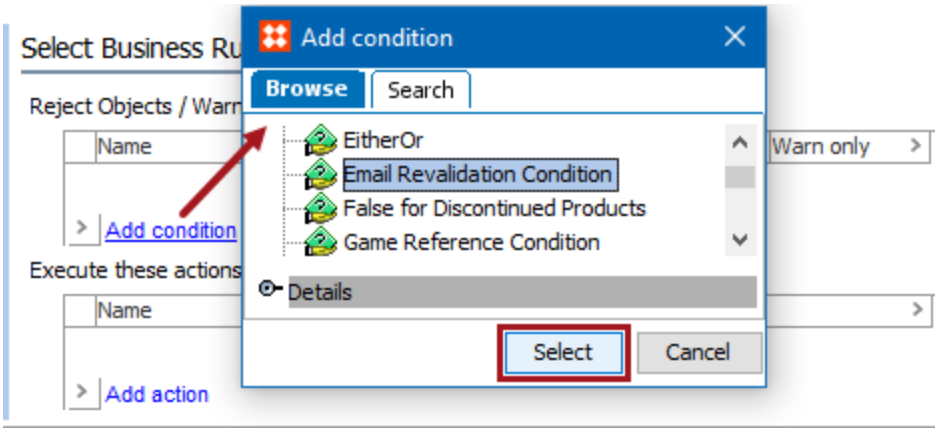
Click the **Next** button to display **Import Manager - Select Business Rules**.

Import Manager - Select Business Rules

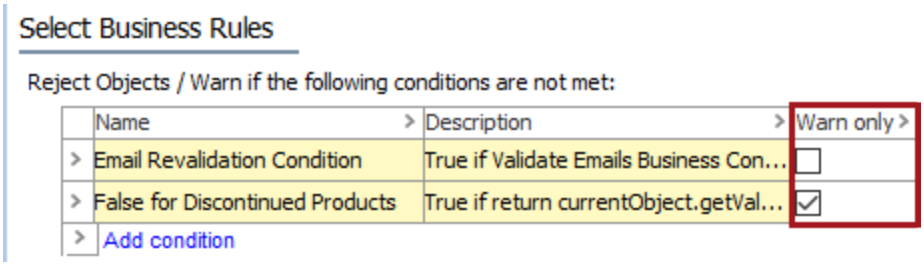


Import data is processed based on any selected business rules. All business rules that exist on your system are available for selection. First, selected business conditions are validated against objects being imported. Conditions that fail are reported as errors unless the 'Warn only' option is enabled, which reports failures as 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, see the **Business Rule Limitations** section below.

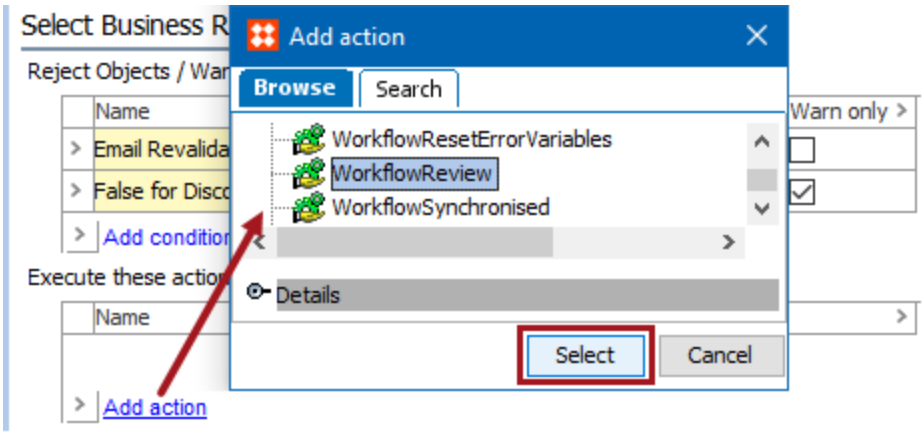
1. If necessary, click the **Add condition** link to add conditions. Then, use Browse or Search tabs to locate the condition, and click the **Select** button. Repeat until all the relevant conditions are added.



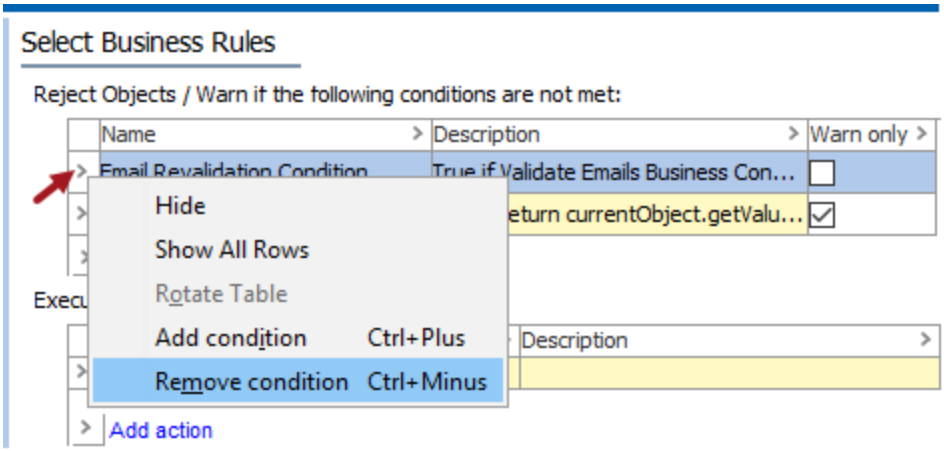
Once a condition is added, the 'Warn only' checkbox is displayed. By default, the option is not checked, meaning objects that fail the condition are reported as errors in the BGP execution report and are not imported. When checked, a warning is written to the BGP execution report when the condition fails and objects are imported. For more information, see the **Business Rule Limitations** section below.



2. If necessary, click the **Add action** link to add actions. Then, use Browse or Search tabs to locate the action, and click the **Select** button. Repeat until all the relevant actions are added.



3. If necessary, to remove a **condition**, select it, right-click, and then select **Remove condition**.



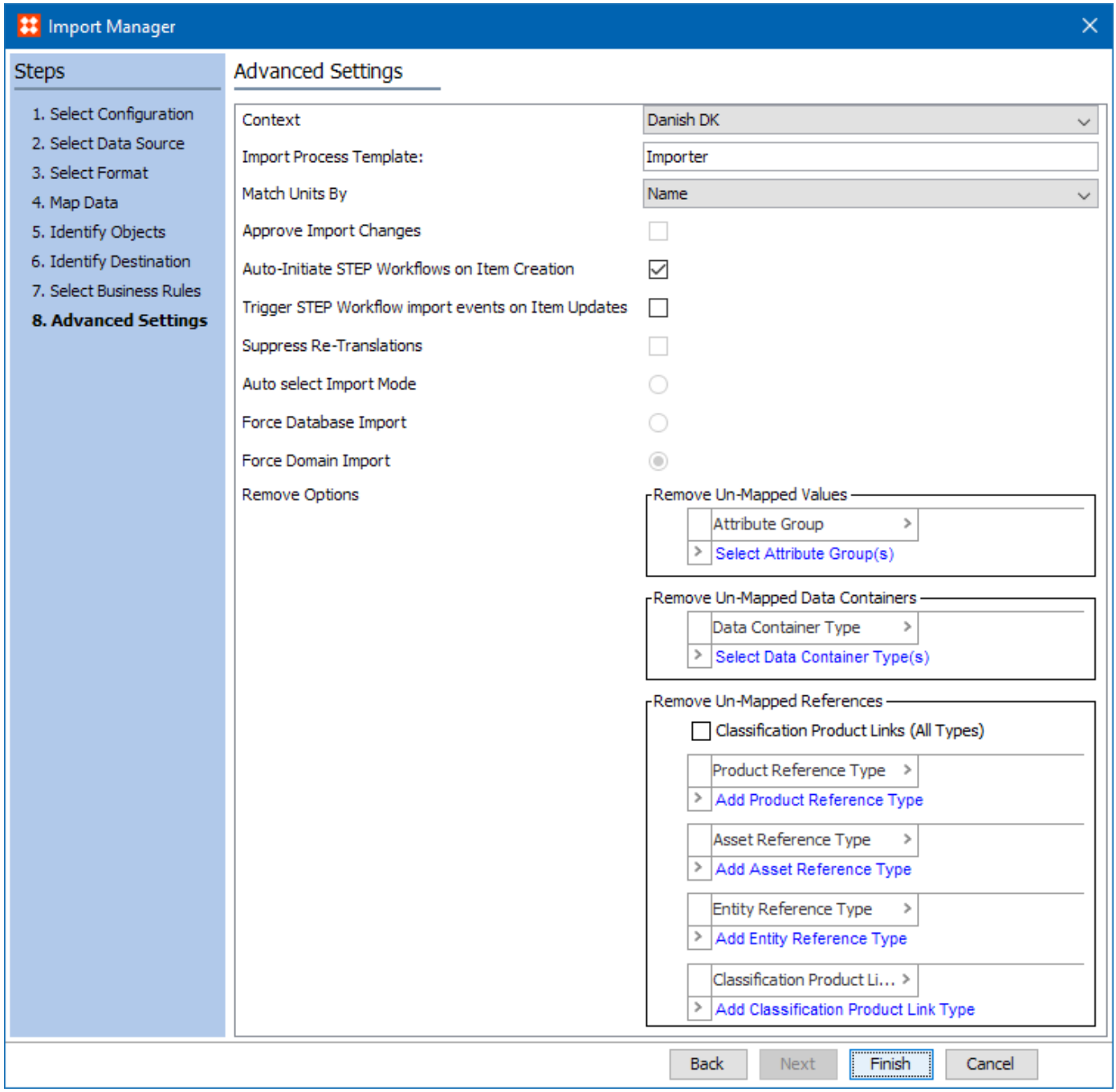
4. If necessary, to remove an **action**, select it, right-click, and then select **Remove action**.
5. Click the **Next** button to display **Import Manager - Advanced Settings** or click the **Finish** button to launch the import process as defined in **Running a Data Import**.

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.
- All selected business rules are run in this sequence: business conditions without 'Warn only' option enabled, business conditions with 'Warn only' option enabled, business actions.
- 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. 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 'Warn only' option can be used for onboarding / enriching objects from suppliers via the Excel Smartsheet format, business conditions set as 'Warn only' are reported 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, see the **Excel Smartsheet Format** topic in this guide.

For more information on business rules, see 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 previous step this option cannot be selected - instead it is possible to approve entire object from business rules.
- **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 Doman Import (explained below) is recommended.**

Force Domain Import - Access of the user who started the import to the hierarchies or attributes being modified is verified during the import. Generally, a more detailed execution report is delivered. If access is not allowed, the import fails and the error "Not privileged to change value ..." is included in the import execution report. Since this option checks all user privileges, it can run slowly and is only advisable for small import files.

Force Database Import - Access of the user who started the import to the hierarchies or attributes being modified is ignored, and the import succeeds. Typically this setting results in a faster import time and is useful for: very large imports, data migration, when working with externally maintained data, commercial data, or small changes to existing data. This mode is not beneficial when the import will create references.

Remove Options

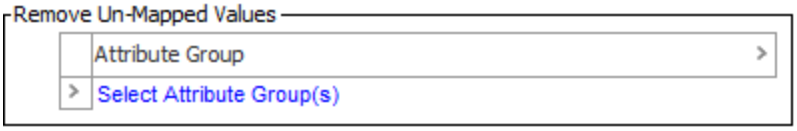
Setting any of the Remove Options means that any existing data for the selected option type will be removed on the objects included in the import file, and **only** the data of the selected option type for the objects in the import file will exist in STEP after the import. Essentially, the remove options 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.

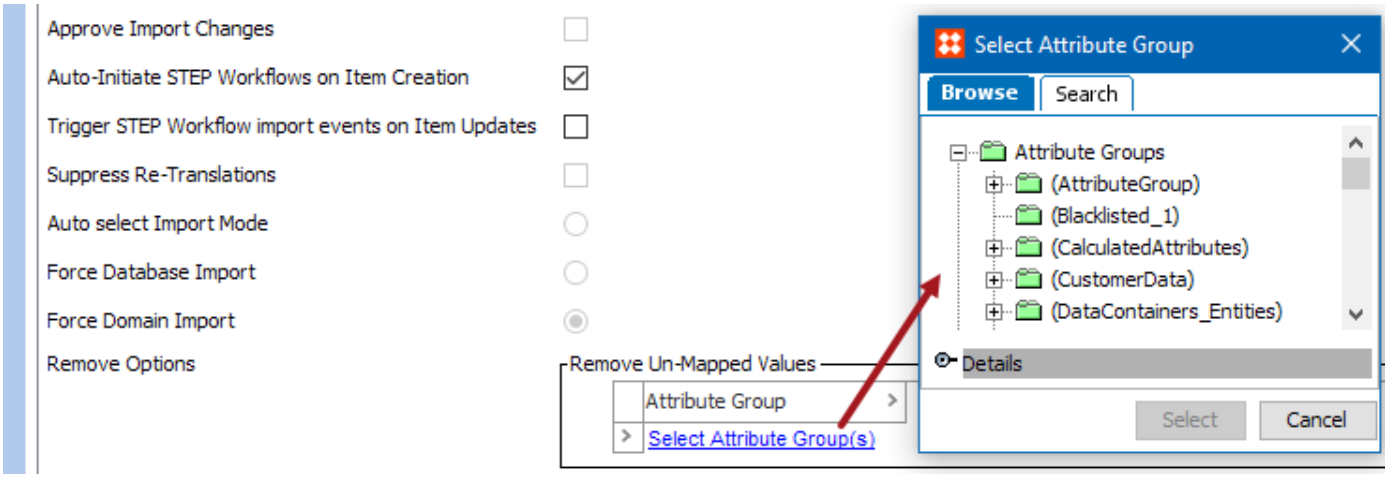
Remove Un-Mapped Values

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.



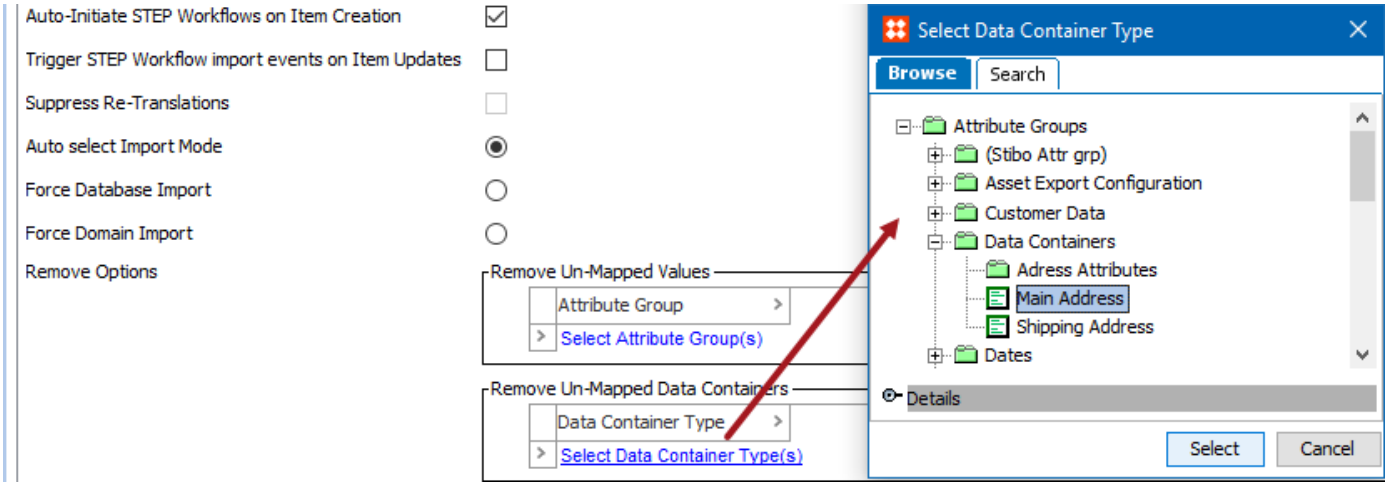
For example, you have 200 products listed in your load file, and they are all gloves of one type or another. You use 8 product attributes to define these gloves. Now you want to use just four attributes. Further, all of the original 8 attributes are in their own attribute group. You load the 200 products using just the 4 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 8 old attribute values for all 200 products, and it will show only the 4 attribute values present in the load file.

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.

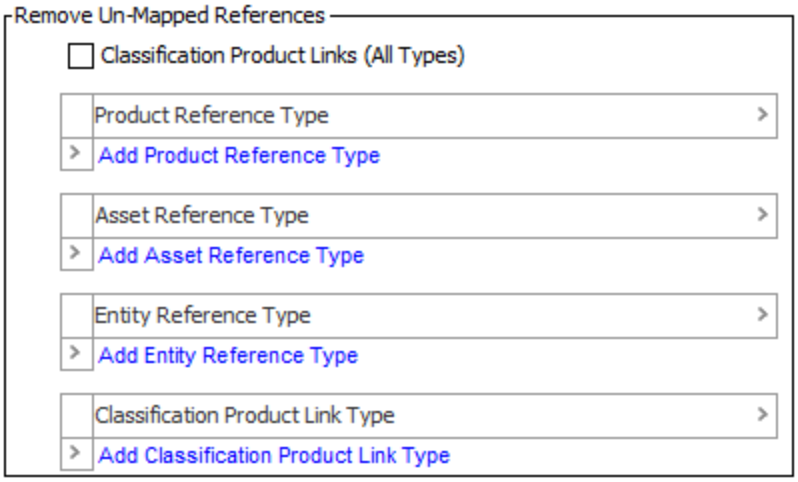


For example, you have a data container type that allows for multiple data containers (e.g., Shipping Address). Your warehouse usually ships to all the locations in Shipping Address, but now your warehouse has been designated to ship to addresses in a different data container type named Shipping Addresses - 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, see **Reference Types** in the **System Setup / Super User Guide** documentation.

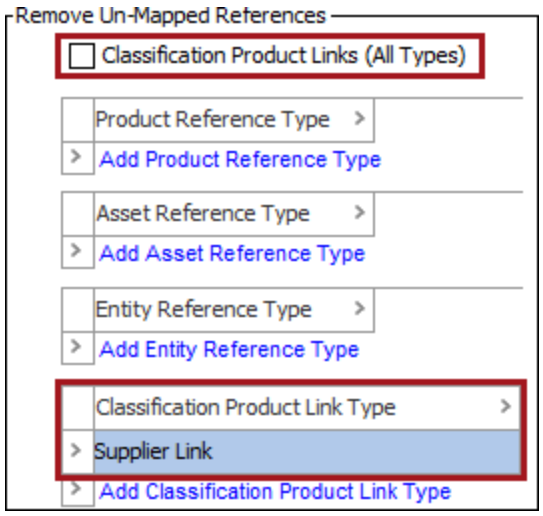


For details on how to add or remove references types, see the **Adding and Removing Un-Mapped References** section below.

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. When this option is displayed, the Classification Product Link Type option is not displayed.

- **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. When this option is displayed, the Classification Reference Type option is not displayed. For more information on classification product link types, see **Product to Classification Link Types** in the **System Setup / Super User Guide** documentation.

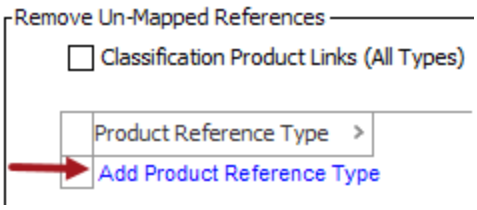
  For example, if you have products that are linked multiple times to different classifications, you can use this option to clean up product to classification links not specified by the import file. You therefore define certain products that are linked only to the classifications you list in your load 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.

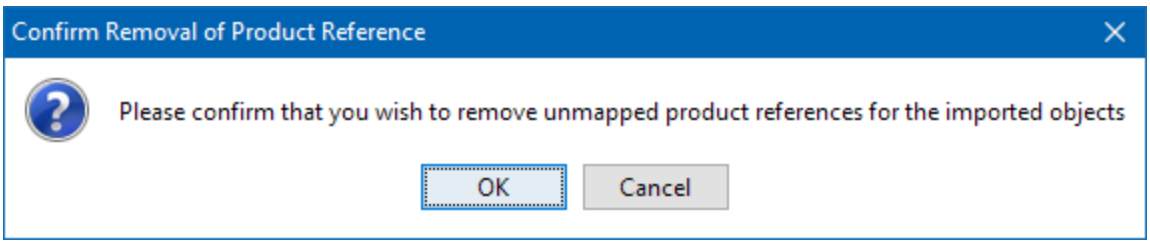
Adding and Removing Un-Mapped References

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.'

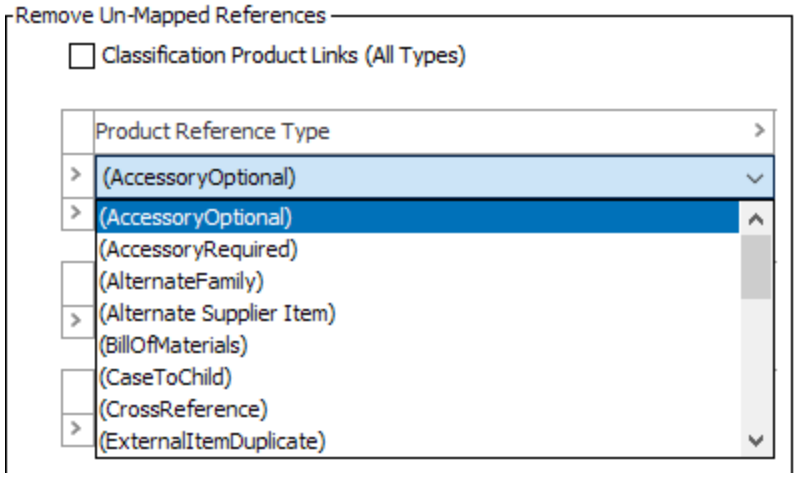
1. Click the option 'Add Product Reference Type.'



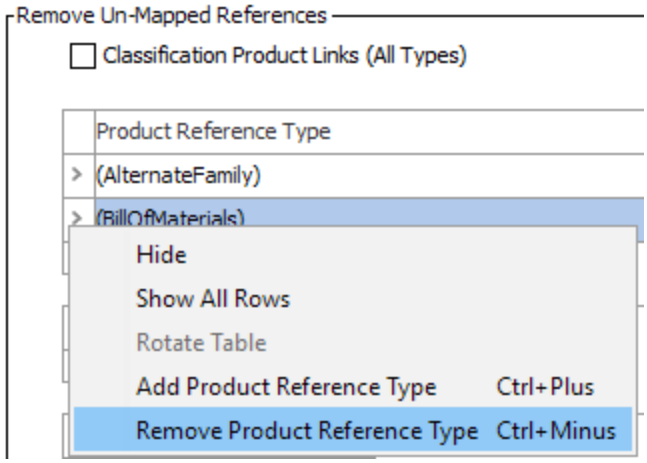
2. A dialog allows the user to confirm that they want to remove all unmapped Product references for the imported object.



3. Click OK, and then select the appropriate reference type from the dropdown list.



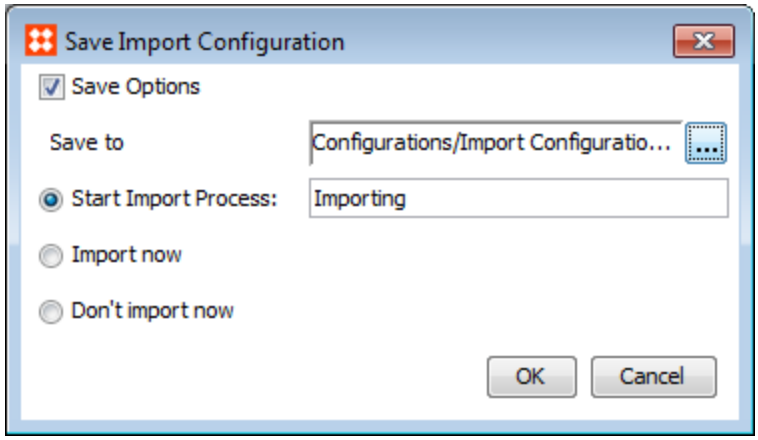
4. Right-click on the field of the Reference type will give the following options:
   - **Hide** - hides the selected reference type from the dropdown list.
   - **Show all Rows** - shows all the selected reference types.
   - **Add Reference Type** - adds an additional reference.
   - **Remove Reference Type** - removes the selected reference type.



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.

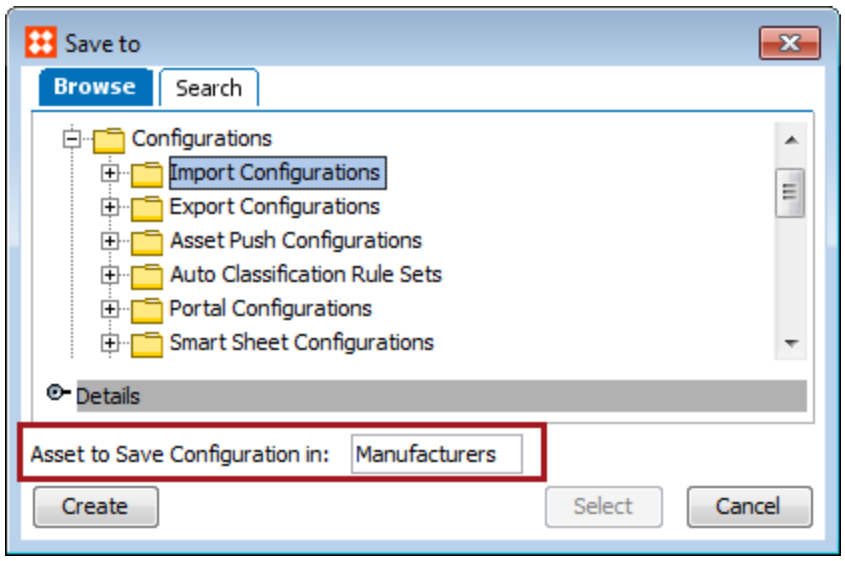
Saving a configuration allows you to modify it (instead of recreating it) during testing. Initially, use a small group of objects to test your settings. Refine the wizard parameters to resolve any issues and save the updated configuration.



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.

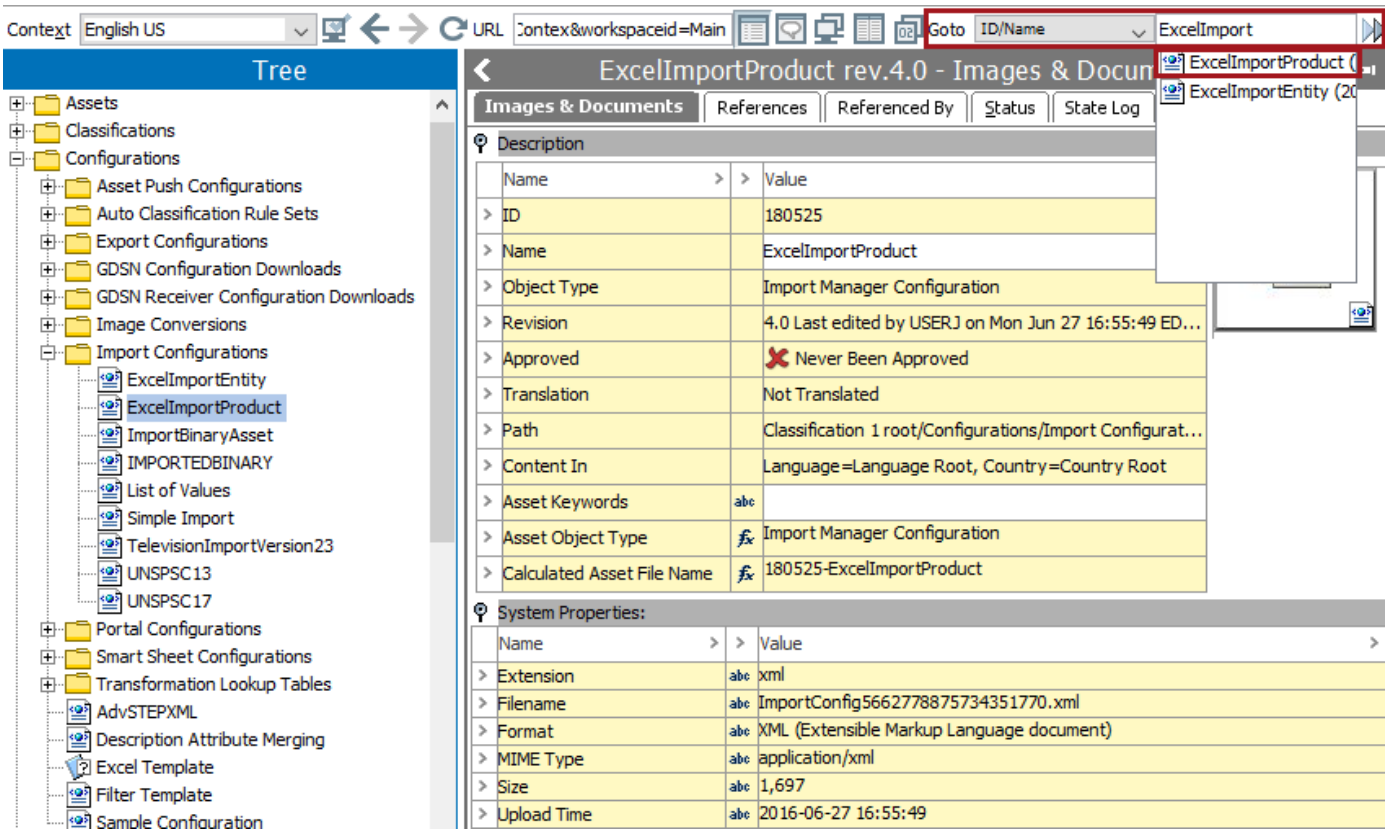
Maintaining a Saved Import Configuration

Completing the Import Manager wizard in workbench includes an option to save an import configuration (see **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.

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

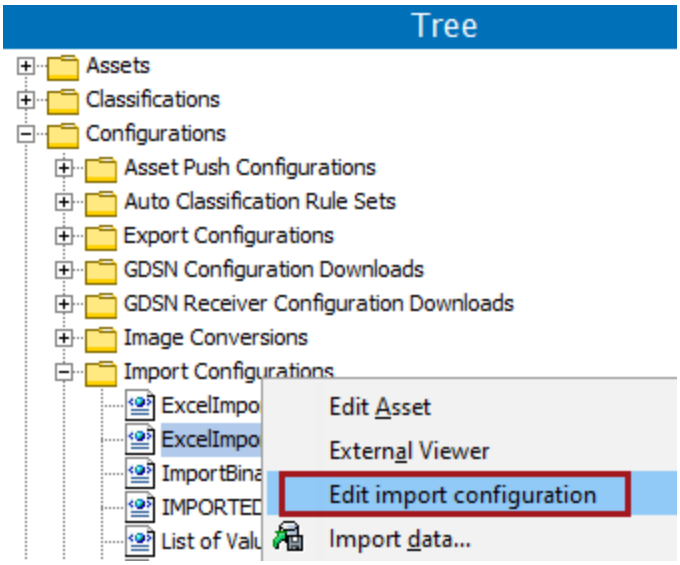
These editing options are available to update the selected configuration.

- 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, see the steps in the **Running a Data Import**

section.

- To change the settings of the existing configuration parameters, follow the steps below.

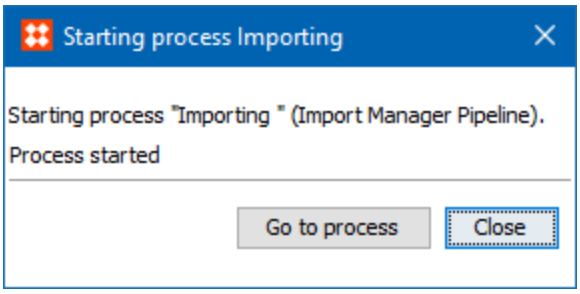
1. Select an import configuration, right-click and select **Edit import configuration** option.



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.

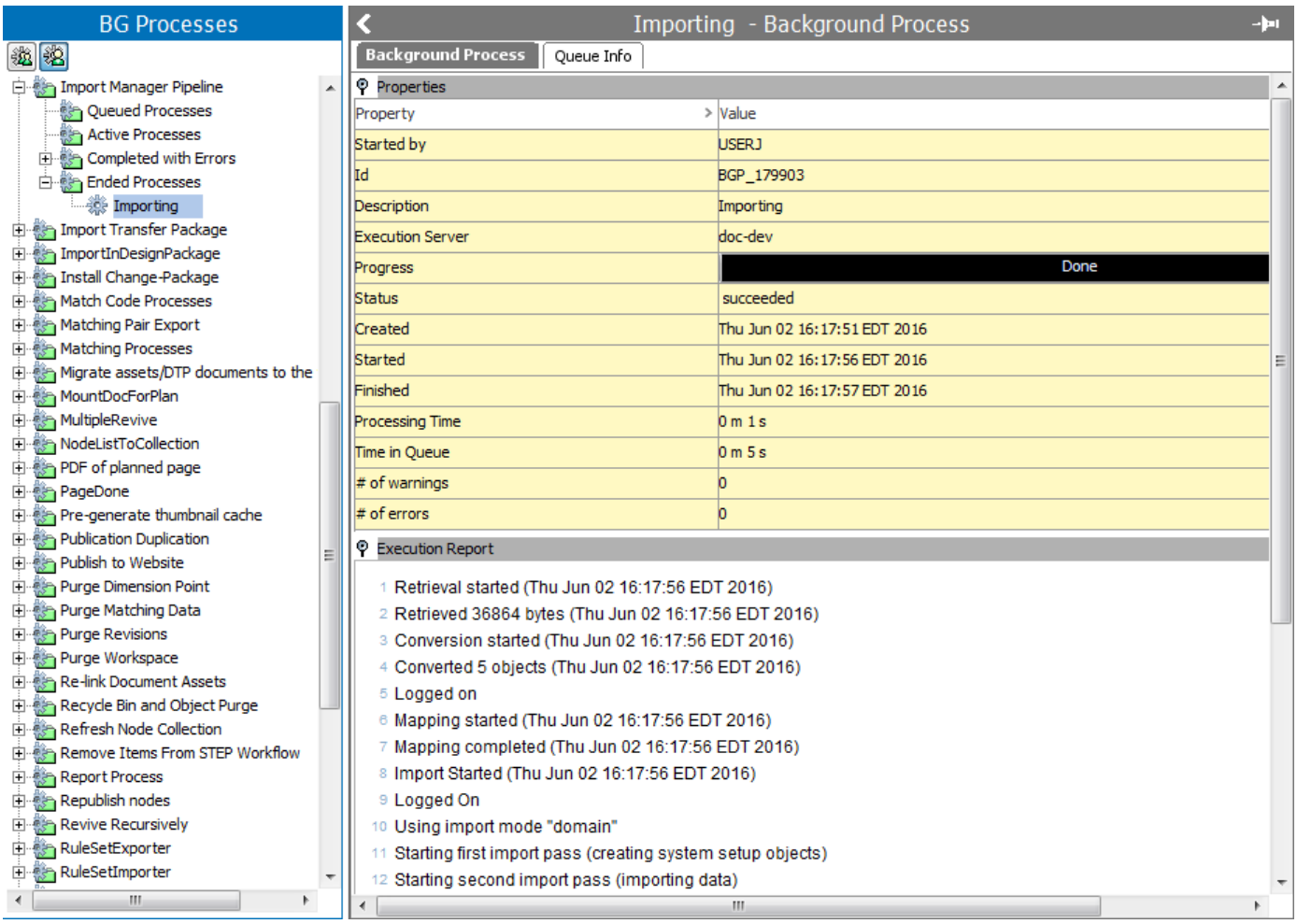
Monitoring the Import Process

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, see **Background Processes and Queues** in the **System Setup / Super User Guide** 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, see 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.

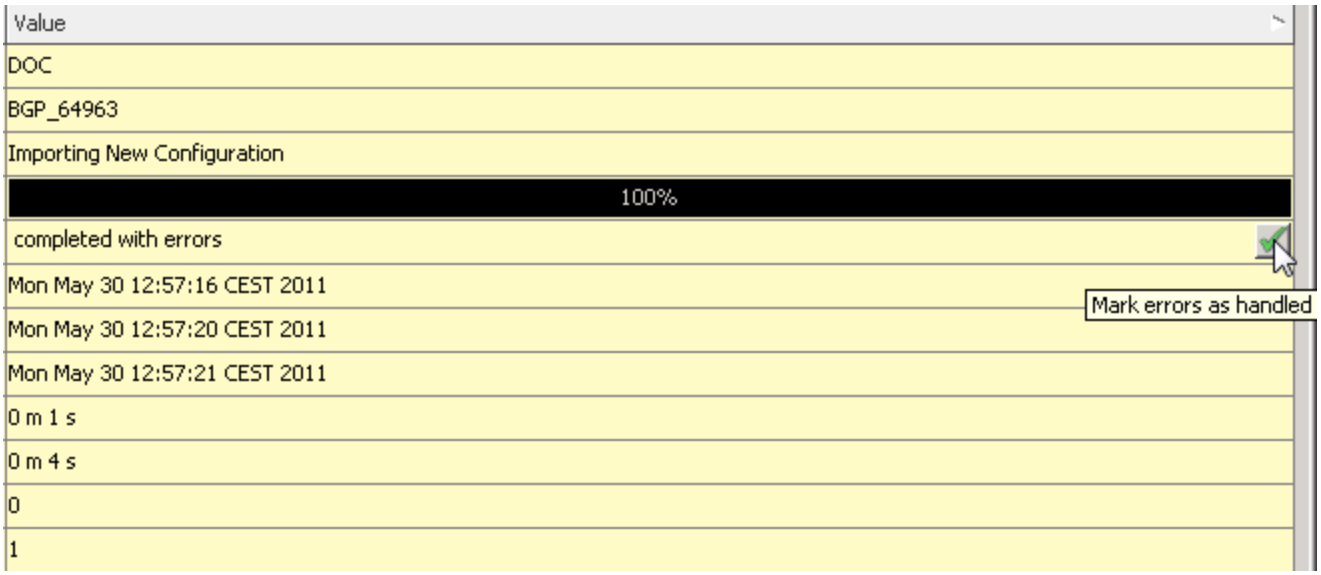
For Excel and CSV imports, the error file contains the rows from the original file with the defect data.

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.



Import Error Messages

The following table shows most of the different 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
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.
Product x contains more than one reference to asset y of Type z, reference data will not be	You cannot make a product-to-asset reference from the same source product to the same target asset more than

Error Message	Description
imported.	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 "see" 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.
No ContextID specified in STEPXML to import	Importing STEPXML without a Context ID.

Error Message	Description
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. See 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:

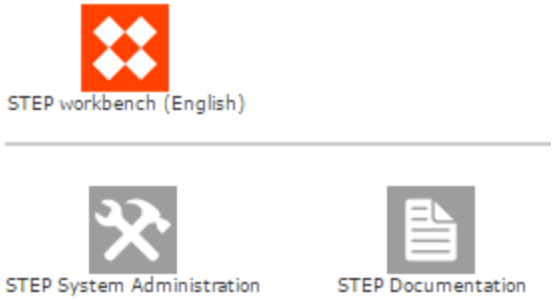
Description	Importing
Execution Server	doc-rel
Progress	100%
Status	completed with errors
Created	Sun Feb 05 06:49:19 EST 2017
Started	Sun Feb 05 06:49:20 EST 2017
Finished	NA
Processing Time	0 m 2 s
Time in Queue	0 m 1 s
# of warnings	0
# of errors	1

Execution Report

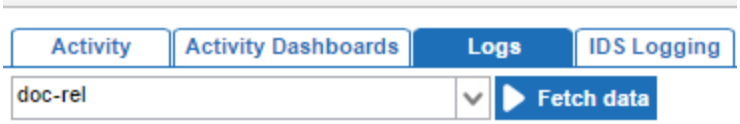
```
 1 Retrieval started (Sun Feb 05 06:49:21 EST 2017)
 2 Retrieved 8817 bytes (Sun Feb 05 06:49:21 EST 2017)
 3 Conversion started (Sun Feb 05 06:49:21 EST 2017)
 4 Converted 6 objects (Sun Feb 05 06:49:24 EST 2017)
 5 Logged on
 6 Mapping started (Sun Feb 05 06:49:24 EST 2017)
 7 Mapping completed (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:

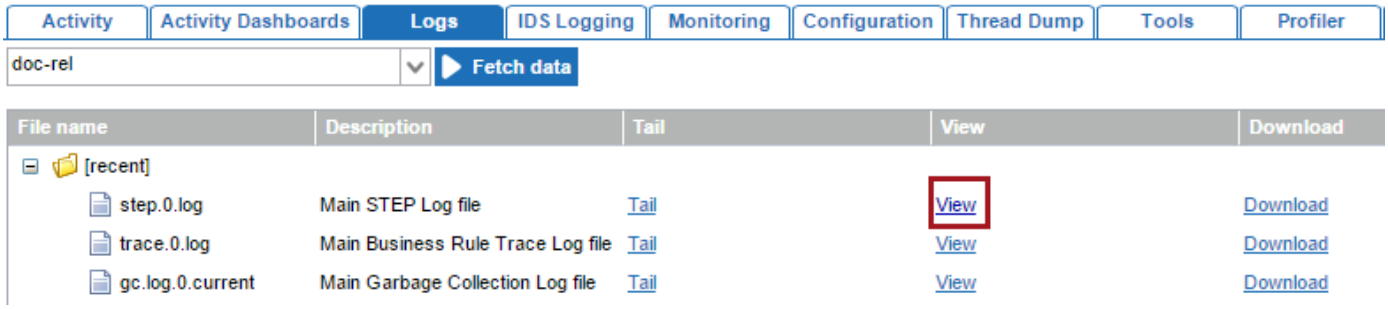
1. Launch the STEP Start Page
2. Click on the System Administration link



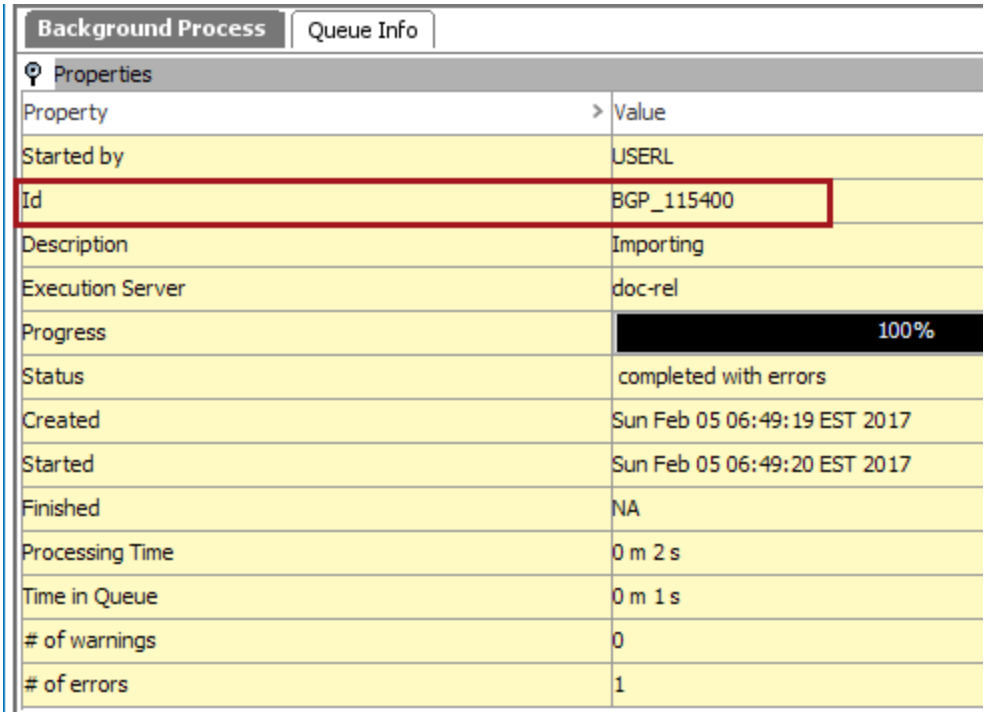
3. Once logged in, click on the Logs Tab
4. Select the Server in the dropdown list and click on 'Fetch Data'



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

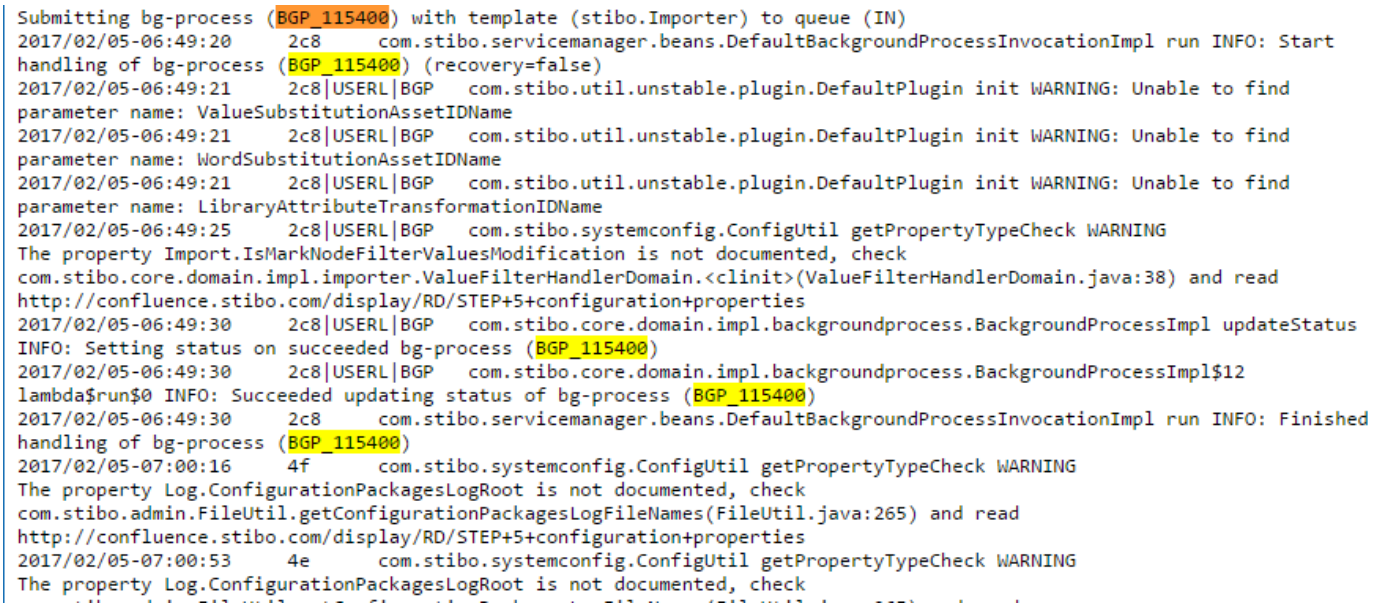


Background Process | Queue Info

Properties

Property	Value
Started by	USERL
Id	BGP_115400
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

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, click the **STEP API Documentation** button on the STEP 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 queue settings for the endpoint as described in **IIEP - Configure Endpoint**.
6. In the wizard, if available, configure defined preprocessors for customer-specific solutions as described in **IIEP - Configure PreProcessor**.
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 postprocessors for customer-specific solutions as described in **IIEP - Configure PostProcessor**.
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 - Configure Error Reporter**.
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 inbound integration endpoint definition for comparison purposes in an external source control system in **Exporting Inbound Integration Endpoint Definitions as Comments**.
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:

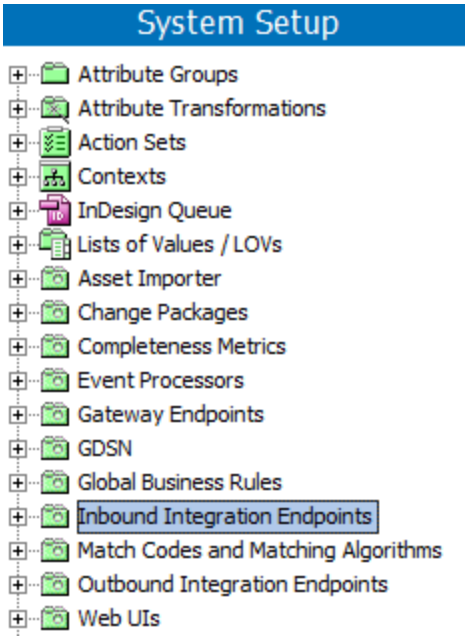
- An Inbound Integration Setup Group
- Linking between the IIEP Object Type and the Setup Group
- An instance of the IIEP Object

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, see the **Action Sets** section and the **Users and Groups** section in the **System Setup / Super User Guide** 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, see the **Working with Users** topic in the **System Setup / Super User Guide** documentation.

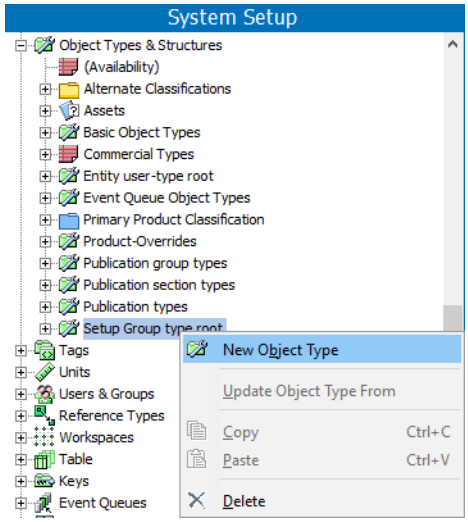
Review your System Setup tab to determine if one or more inbound integration endpoint nodes already exist. The name of the node on your system is not required to match the one in the image below.



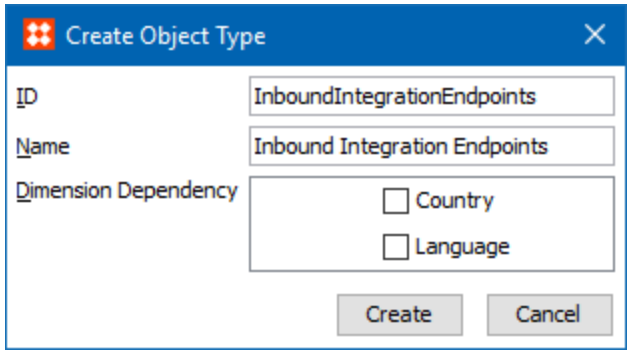
Once the setup has been completed, the steps in this section are only needed if additional levels of organization are desired.

Create the Inbound Integration Setup Group

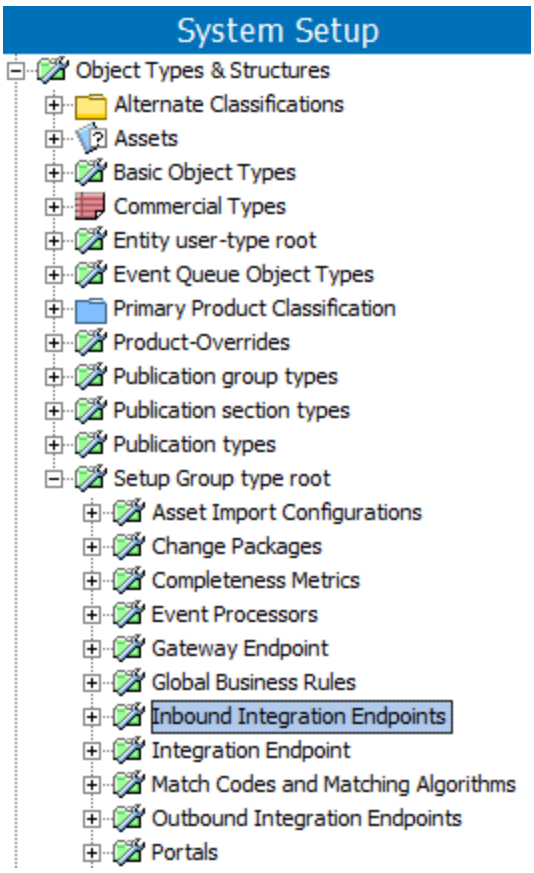
1. In System Setup, expand **Object Types & Structures**.
2. Right-click 'Setup Group type root', and choose **New Object Type**.



3. Enter an **ID** and a **Name**, select any required Dimension Dependencies based on your system, and click **Create**.

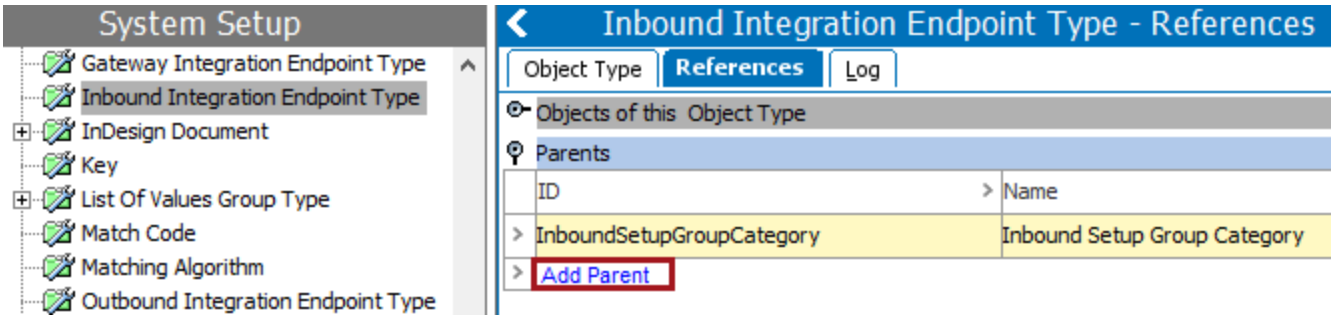


4. The new setup group appears in System Setup under 'Object Types & Structures' as a child in the **Setup Group type root**.

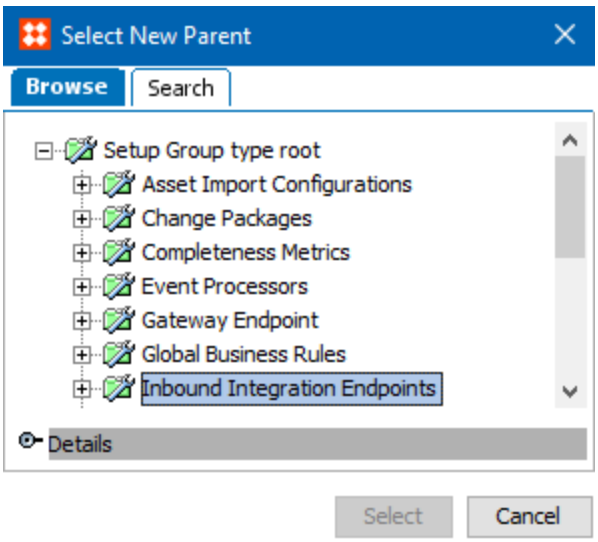


Link the IIEP Object Type to Setup Group

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.

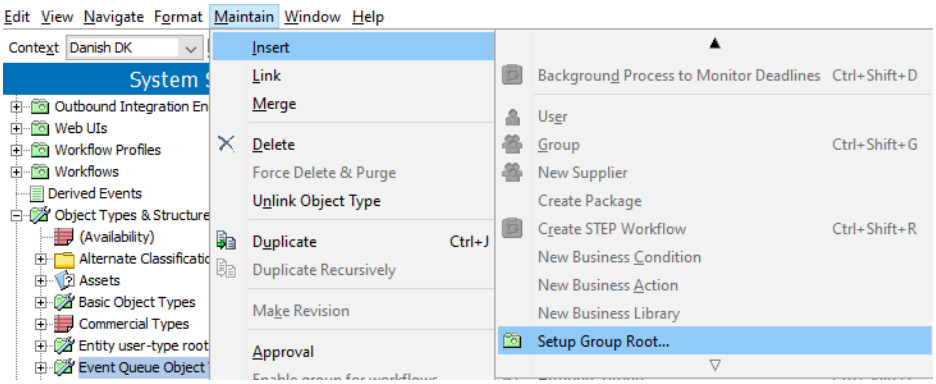


3. In the Select New Parent dialog, select the setup group you created, and click **Select** to make it a valid parent.

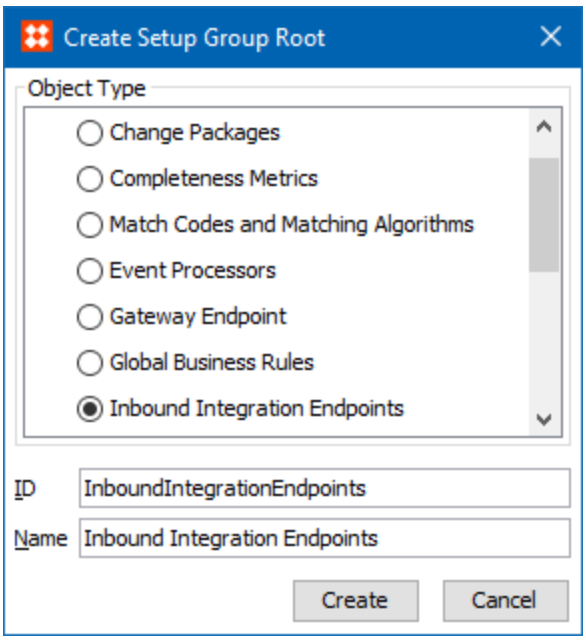


Create an Instance of the IIEP Object

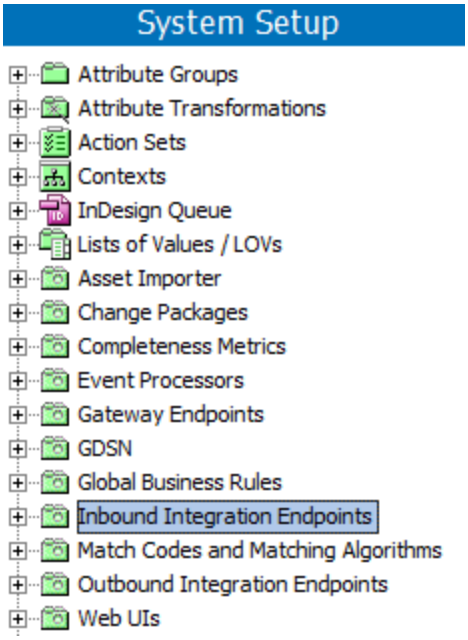
1. On the System Setup tab, select any object in the **System Setup** hierarchy to activate the following Maintain menu selection.
2. Click the Maintain menu, point to Insert, and select **Setup Group Root**.



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.

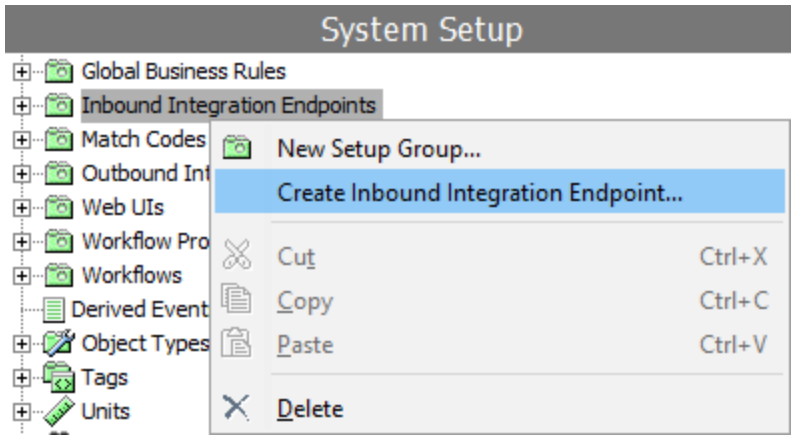


Creating an Inbound Integration Endpoint

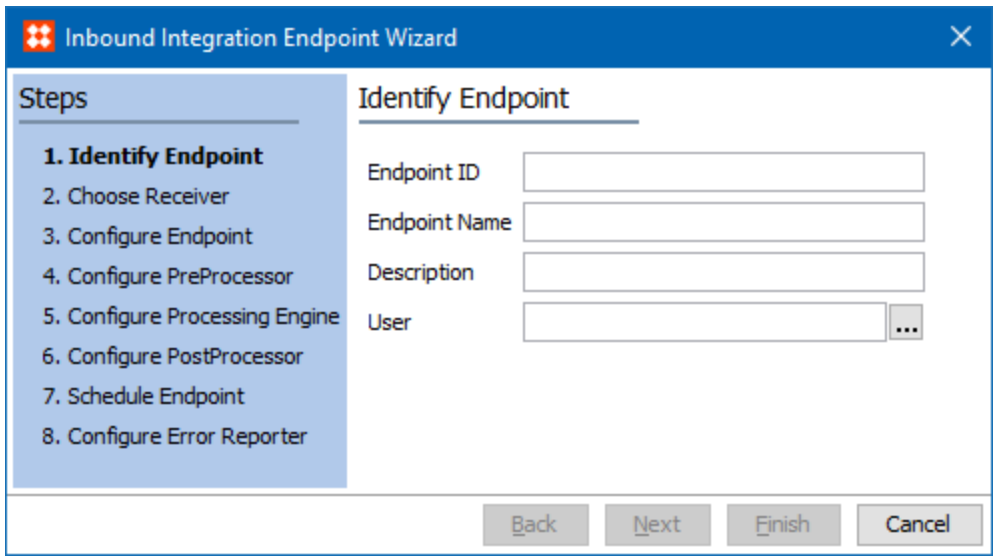
After creating a setup group for inbound integration endpoints, create an inbound integration endpoint to monitor and maintain integrations with systems that send data to STEP in the form of files or messages.

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:

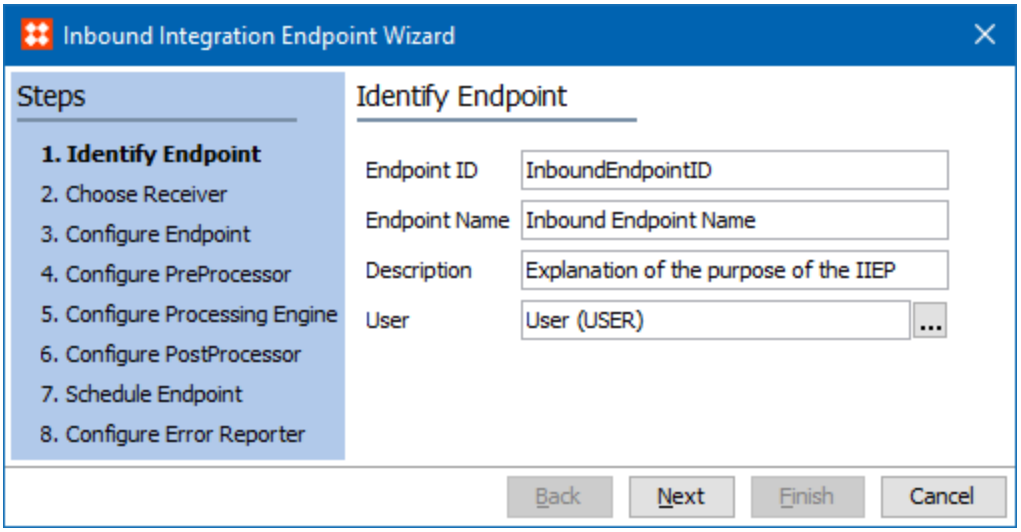


- **Identify Endpoint** allows you to specify the name and ID of the endpoint and identify the user whose privileges are applied.
- **Choose Receiver** allows you to specify the receiver of the data.

- **Configure Endpoint** allows you to specify processing, context and queue settings for the integration endpoint.
- **Configure PreProcessor**, when available, allows you to configure defined preprocessors for customer-specific solutions.
- **Configure Processing Engine** allows you to specify the data format and map the data, among other options.
- **Configure PostProcessor**, 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.
- **Configure Error Reporter** allows you to specify, select, and configure 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



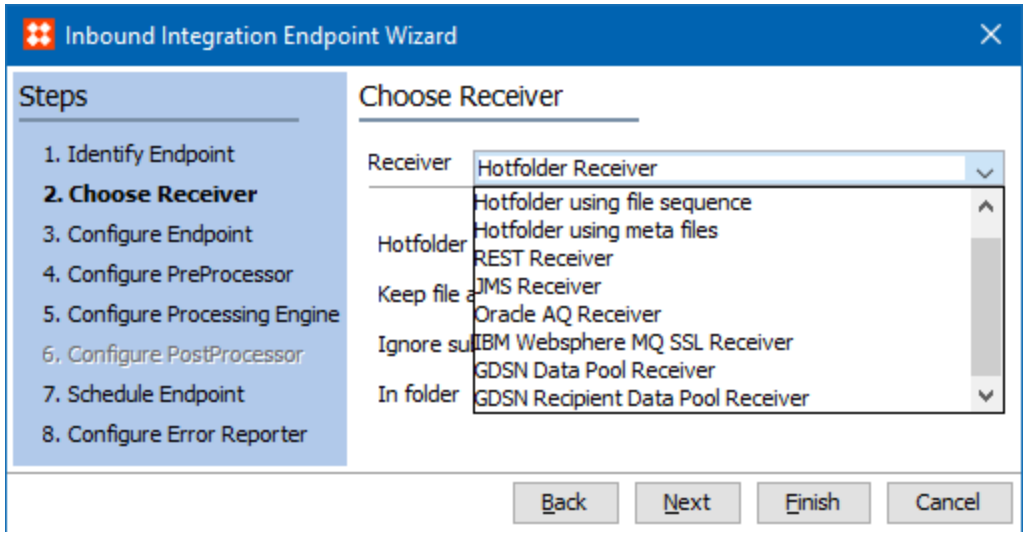
1. For **Endpoint ID**, enter an ID. Common setup is to use no spaces or punctuation.
2. For **Endpoint Name**, enter a name. Common setup is to repeat the ID with added spaces for readability.
3. For **Description**, enter an optional description for the IIEP.
4. For **User**, search or browse for a 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, see the **Working with Users** topic in the **System Setup / Super User Guide** documentation.

5. Click the **Next** button to display **IIEP - Choose Receiver**.

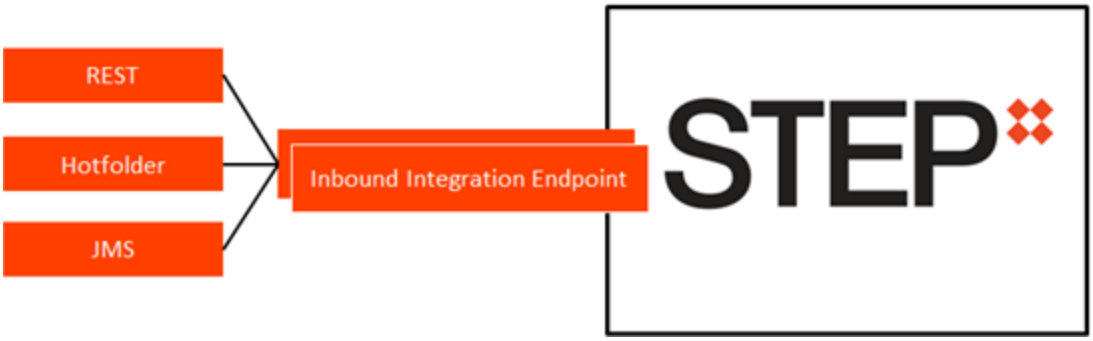
IIEP - Choose Receiver



1. Select the receiver to be used for the inbound endpoint. The available standard receivers are:
   - GDSN Data Pool Receiver (available with GDSN implementation)
   - GDSN Recipient Data Pool Receiver (available with GDSN implementation)
   - Hotfolder Receiver (default receiver and required for Asset Importer processing engine)
   - Hotfolder Receiver Using File Sequence
   - Hotfolder Receiver Using Meta Files
   - IBM Websphere MQ SSL Receiver
   - JMS Receiver
   - Dynamic JMS Receiver
   - Oracle AQ Receiver
   - Product Data Syndication Receiver
   - REST Receiver
   - Web UI File Loading Receiver
2. Click the **Next** button to display **IIEP - Configure Endpoint**.

IIEP 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.



Import Manager does not require a receiver method since the user chooses a single file for import. For more on Import Manager, see **Import Manager**.

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). See **Amazon SQS Receiver** documentation.
GDSN Data Pool Receiver	The GDSN solution must be implemented fully before the GDSN Data Pool Receiver method is available and functional. See **GDSN Provider** 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. See **GDSN Receiver** documentation.
Hotfolder Receiver	Enables setup of a standard data hotfolder, typically on the application server. See **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. See **Hotfolder Receiver Using File Sequence** documentation.

Method	Description
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. See **Hotfolder Receiver Using Meta Files** documentation.
IBM Websphere MQ SSL	IBM Websphere MQ SSL, uses MQ series Secure Sockets Layer, and enables data exchange across IBM and non-IBM platforms. See **IBM Websphere 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. See **JMS 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. See the **Dynamic JMS Receiver** documentation.
Oracle AQ Receiver	Oracle Advanced Queuing (Oracle AQ) enables messages to be exchanged between two systems. See **Oracle AQ Receiver** documentation.
Product Data Syndication Receiver	STEP receives data from the Product Data Syndication (PDS) platform via the default PDS Inbound Integration Endpoint and the API. See **Product Data Syndication Receiver** documentation.
REST Receiver	Representational State Transfer (REST) is a software architecture style used to design web services. See **REST Receiver** documentation.
Web UI File Loading Receiver	Receiver used for the Web UI 'File Loading Widget' for non-hotfolder based receiver types. See the **Web UI File Loading Receiver** documentation.

Amazon SQS Receiver

Using the Amazon SQS Receiver allows data to be imported into STEP from a FIFO Amazon SQS queue on AWS.

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.

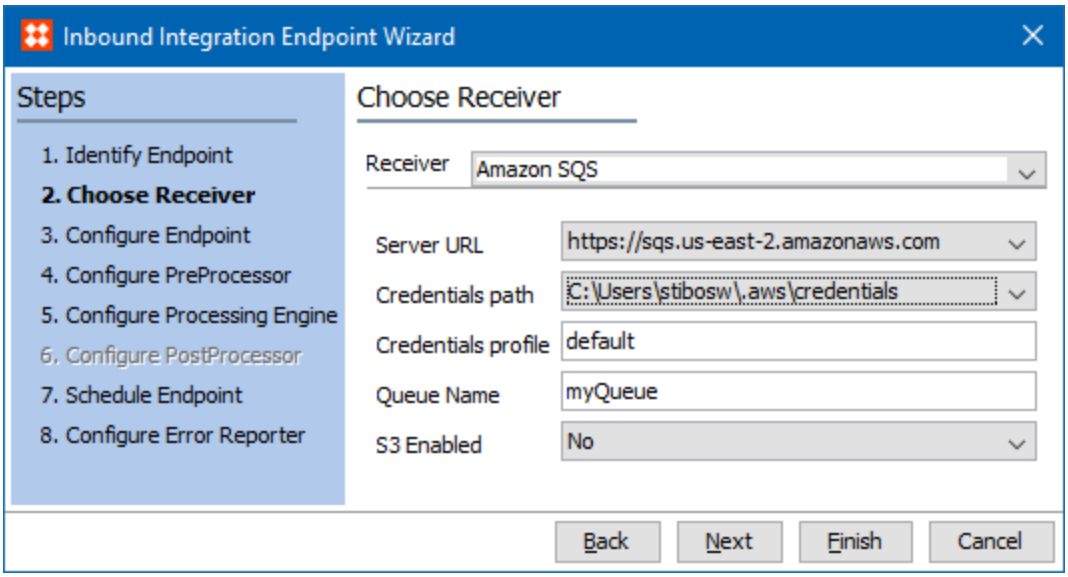
   ```
   SQSCredentialsPath=1=C:\\Users\\stibosw\\.aws\\credentials,2=C:\\Users\\stibosw
   \\.aws\\credentials123
   ```

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, see 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.

   When the system writing to the SQS queue uses the Amazon's S3 linking functionality, this option allows the receiver to read references to S3 items from SQS. This effectively allows larger messages to be received from the queue.

7. Click the **Next** button to continue with the Configure Endpoint step, or the **Finish** button to close the wizard.

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, see the **IIEP - Configure Asset Importer Processing Engine** documentation.

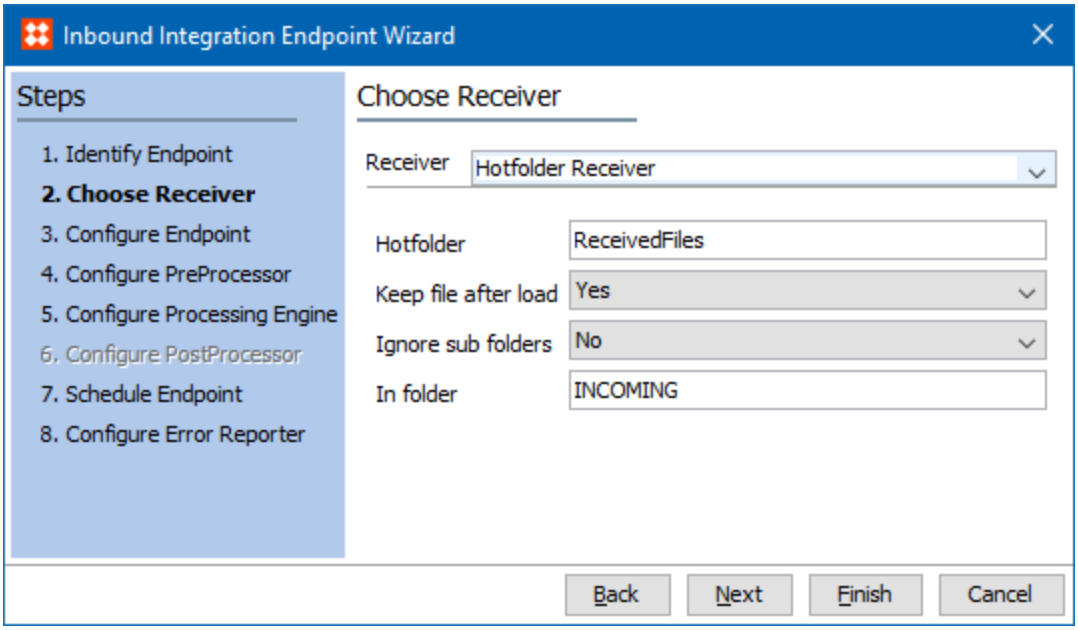
To control the order of the files to be imported, set up either a **Hotfolder Receiver Using Meta Files** or a **Hotfolder Receiver Using File Sequence** instead.

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 **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.
5. 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.
6. Click the **Next** button to continue with the Configure Endpoint step, or the **Finish** button to close the wizard.

Hotfolder Receiver Using File Sequence

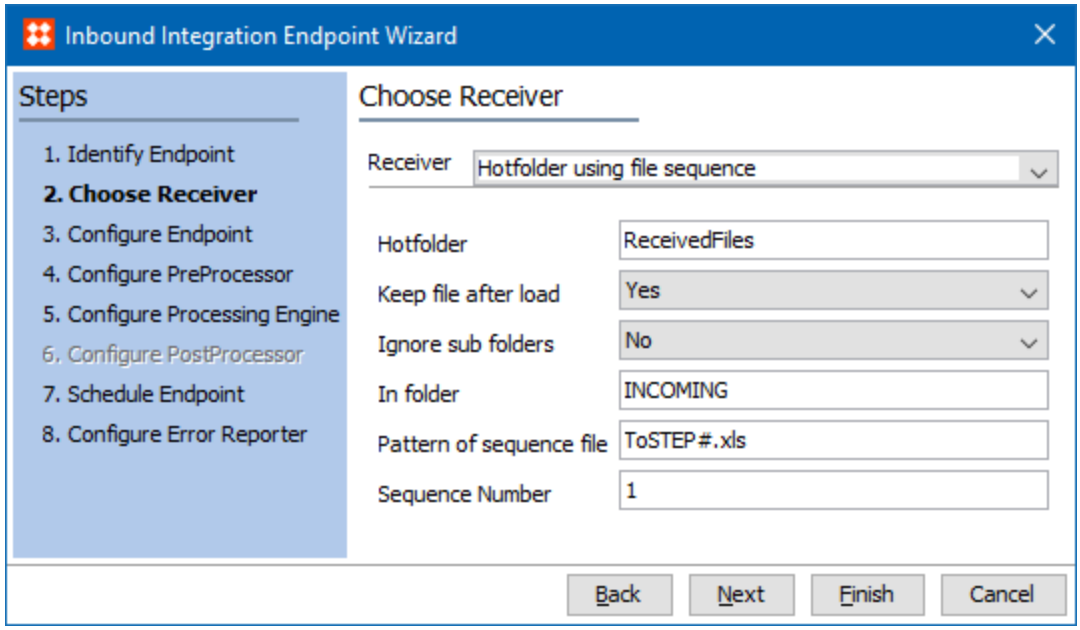
The 'Hotfolder using File Sequence' receiver enables setup of a standard data hotfolder where a sequence ID (included in the file name) determines the file processing order. Files are placed into this folder and are processed by the integration endpoint. Typically this is on the application server or 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 **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.

5. 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.
6. 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.
7. 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 see the number for the next file. With this option, sequences can only be skipped if you manually reconfigure the 'Sequence Number.'
8. 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.

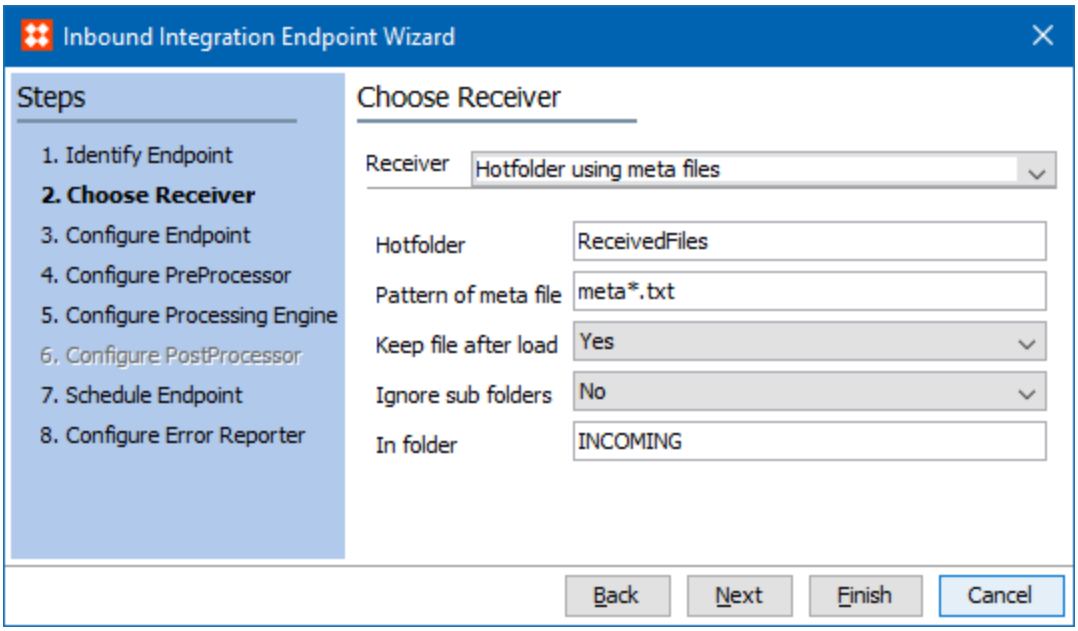
Prerequisites

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.

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 **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.
5. 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.
6. Click the **Next** button to continue with the Configure Endpoint step, or the **Finish** button to close the wizard.

IBM WebSphere MQ SSL Receiver

For information on connecting to IBM WebSphere MQ in a non-SSL way, see **JMS Receiver**.

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 **WSMQSSLProviderURL** property. The elements are needed for each URL in the format [host]:[port]/[channel]:
   - [host] = hostname or IP of the WebSphere MQ server
   - [port] = port number for the channel
   - [channel] = name of the channel
2. Prior to configuration, clicking the **Queue Manager** 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 **WSMQSSLQueueManager** property.
3. 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 **WSMQSSLQueue** property.
4. Prior to configuration, clicking the **Key Store** 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 **WSMQSSLKeyStoreLocation** property.

   This must be in jks format, with the personal certificate for the Queue Manager. To generate, consult the manual for IBM WebSphere MQ. The password for the key store must be configured in sharedconfig.properties using the case-sensitive **WSMQSSLKeyStorePassword** property.

5. Prior to configuration, clicking the **Trust Store** 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 **WSMQSSLTrustStoreLocation** property.

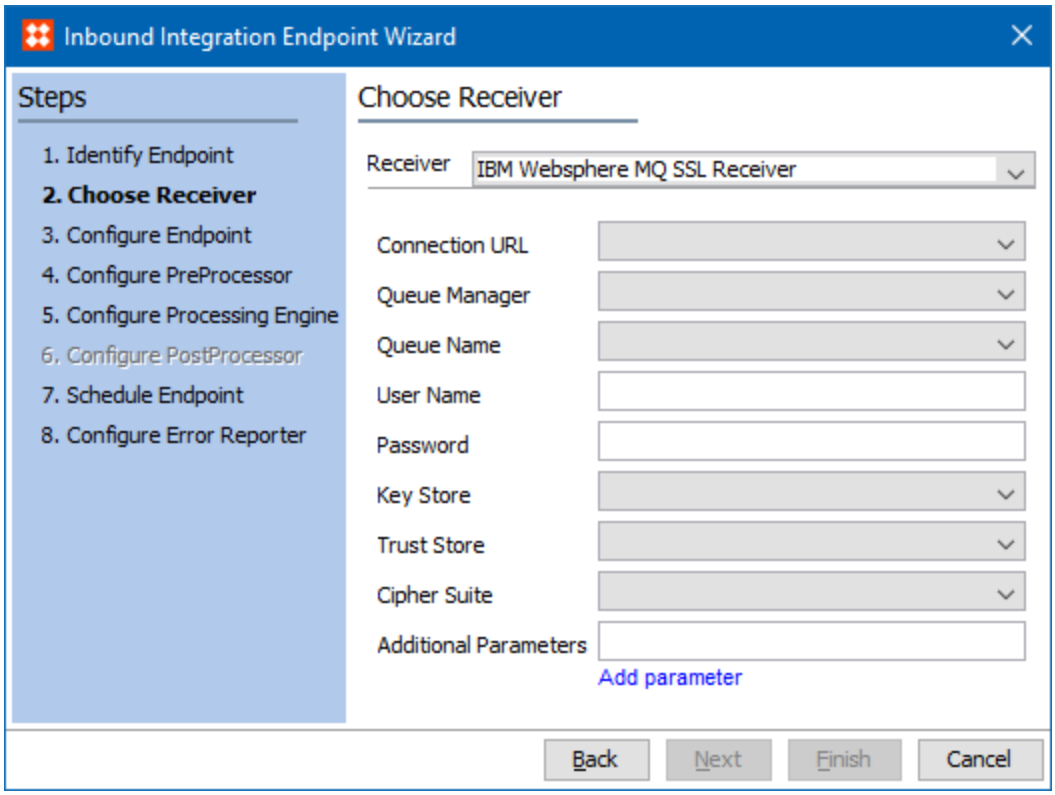
   The Trust Store can be the same file as the Key Store. To generate, consult the manual for IBM WebSphere MQ. The password for the trust store must be configured in sharedconfig.properties using the case-sensitive **WSMQSSLTrustStorePassword** property.

6. Prior to configuration, clicking the **Cipher Suite** 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 **WSMQSSLCipherSuite** property.

   Set this property to the same value as SSL CipherSuite in WebSphere MQ.

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 Websphere 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 that will be used to log onto the JMS provider.
6. For **Password**, if required, enter the password that will be used to log onto the JMS provider.
7. For **Key Store**, select the key store.
8. For **Trust Store**, select the trust store with the CA for the Queue Manager.
9. For **Cipher Suite**, since STEP is running on non-IBM jre, this must be the same value as configured in the Queue Manager.
10. 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 WebSphere MQ.
11. 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.

If additional JME support is required, contact your Stibo Systems account manager for customizations.

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:

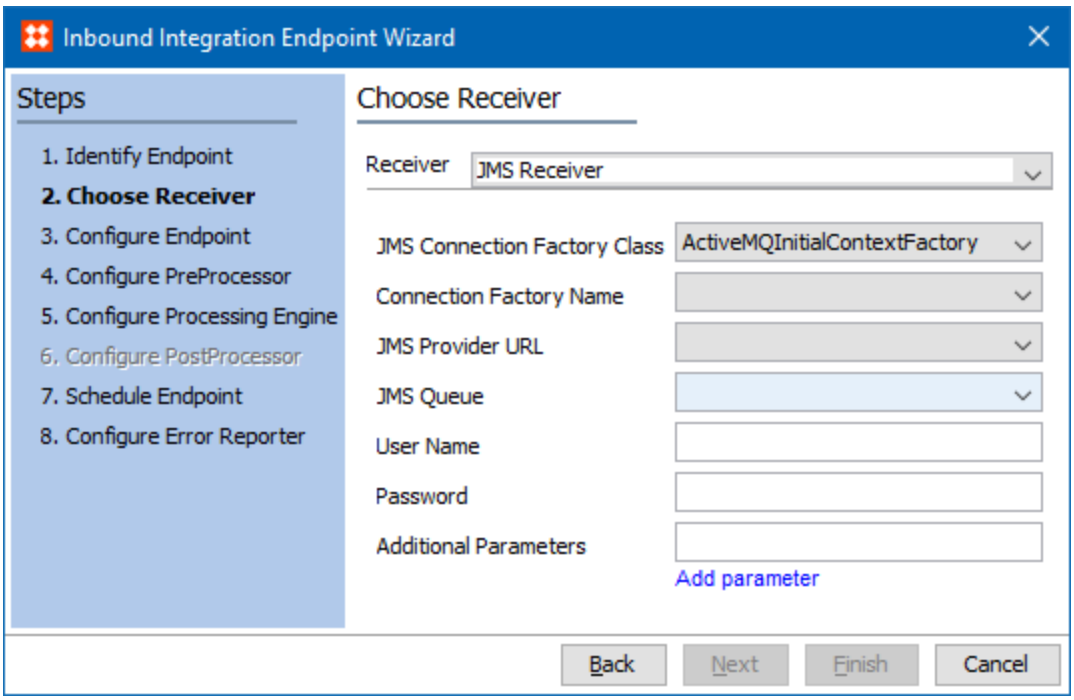
   ```
   JMSProviderURL=1=tcp://[IP address]:[Port, ActiveMQ default is 61616]
   ```

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.



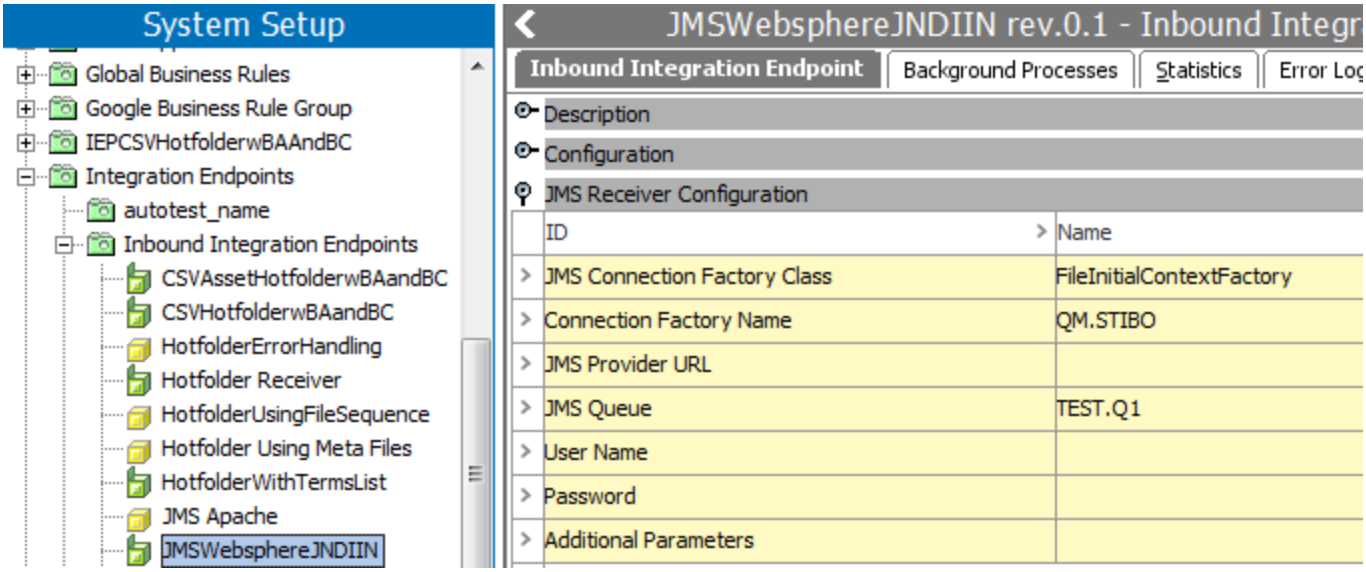
1. For **Receiver**, choose **JMS Receiver**.
2. For **JMS Connection Factory Class**, select one of the available JMS connection factory classes.
   - **ActiveMQInitialContextFactory**: Allows set up of a JMS receiver connecting to Apache ActiveMQ. For information, search the web.
   - **FileInitialContextFactory**: 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 manual 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

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.

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

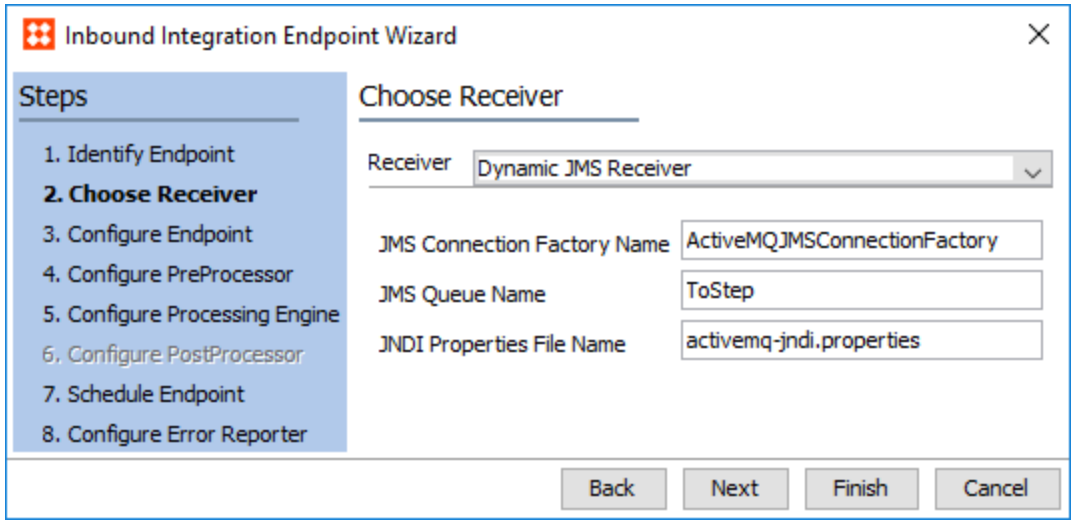
Changes to the properties file or any files found in the class path directory, as outlined below, are implemented when the server is restarted.

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. See 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.

Azure Message Bus

In the JNDI properties file, add properties to set the context factory and map queues, setting connectionfactory = SBCF as shown below:

```
connectionfactory.SBCF = failover://
(amqps://khpetest.servicebus.windows.net)?jms.prefetchPolicy.all=10&failover.maxRecon
nectAttempts=20
```

Register queues in JNDI using the form:

- queue.[jndi_name] = [physical_name]
- topic.[jndi_name] = [physical_name]
- queue.QUEUE = queue1

For more information on the Qpid JMS client library, see 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 and the queue have the same name).

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

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=10.0.42.126

ConnectionFactory/RefAddr/3/Type=host

ConnectionFactory/RefAddr/3/Encoding=String

ConnectionFactory/RefAddr/4/Content=pimconsumer

ConnectionFactory/RefAddr/4/Type=virtualHost

ConnectionFactory/RefAddr/4/Encoding=String

# rabbitmqQueue01 Queue

rabbitmqQueue01/ClassName=javax.jms.Queue

rabbitmqQueue01/FactoryName=com.rabbitmq.jms.admin.RMQObjectFactory

rabbitmqQueue01/RefAddr/0/Content=jms/Queue

rabbitmqQueue01/RefAddr/0/Type=name

rabbitmqQueue01/RefAddr/0/Encoding=String
```

```
rabbitmqQueue01/RefAddr/1/Content=javax.jms.Queue
rabbitmqQueue01/RefAddr/1/Type=type
rabbitmqQueue01/RefAddr/1/Encoding=String
rabbitmqQueue01/RefAddr/2/Content=com.rabbitmq.jms.admin.RMQObjectFactory
rabbitmqQueue01/RefAddr/2/Type=factory
rabbitmqQueue01/RefAddr/2/Encoding=String
rabbitmqQueue01/RefAddr/3/Content=rabbitmqQueue01
rabbitmqQueue01/RefAddr/3/Type=destinationName
rabbitmqQueue01/RefAddr/3/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
```

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:

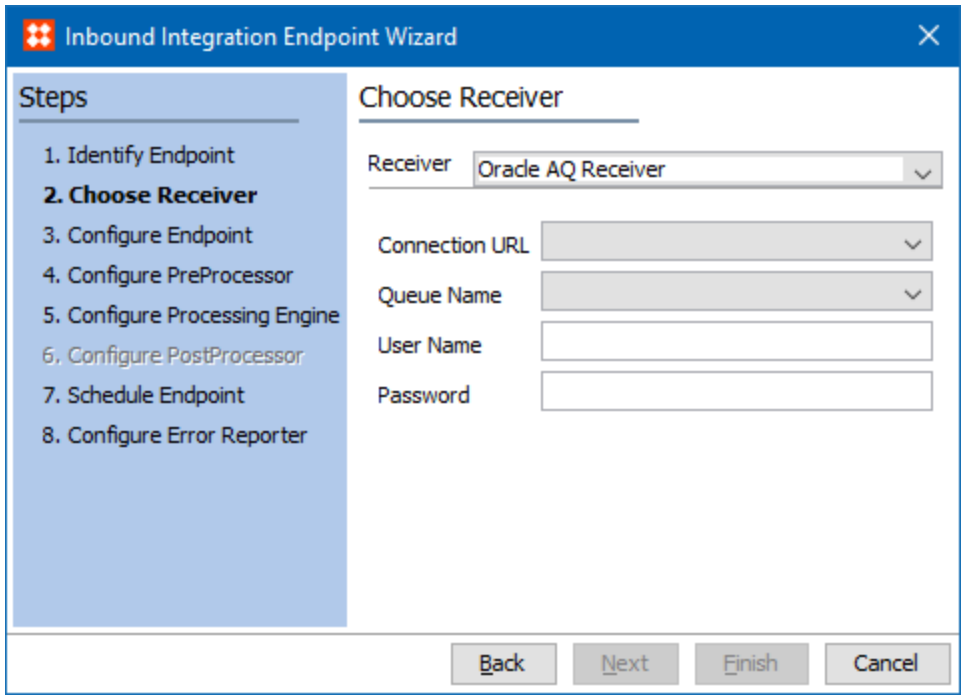
   ```
   OracleAQReceiverConnectionURL=1=ConnectionURL1,2=ConnectionURL2
   ```

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 Syndication Receiver

The Product Data Syndication Receiver option is only available in IIEPs. STEP receives data from the Product Data Syndication (PDS) platform via the default PDS Inbound Integration Endpoint and the API.

For additional information on PDS, see the **Product Data Syndication** 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. 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 **PDS.Url** property. For example, a single URL is displayed with the following property setting:

   ```
   PDS.Url=1=https://StiboPDS/
   ```

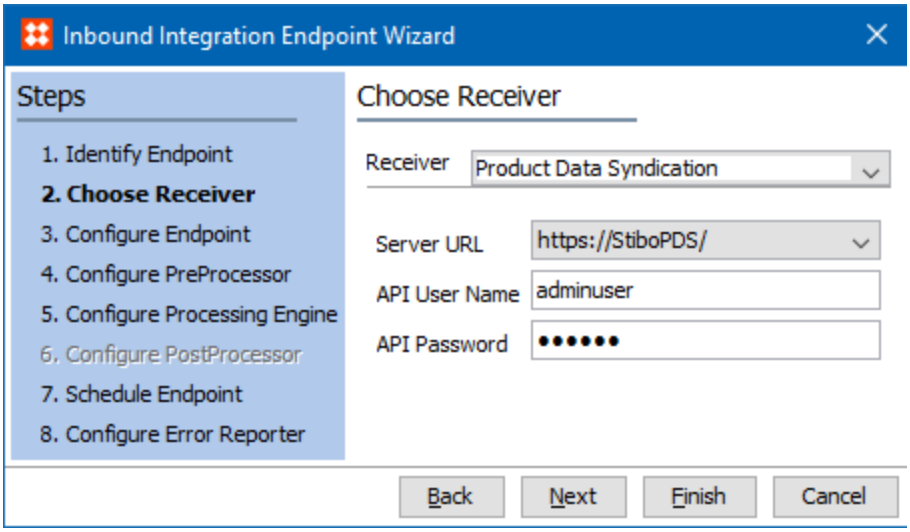
   If multiple URLs are required, separate the entries with a comma, as shown below:

   ```
   PDS.Url=1=https://StiboPDS1/,2=https://StiboPDS2/
   ```

2. Determine the API user name and password. This will be required on the Choose Receiver step of the wizard.
3. Perform additional configuration required in the IIEP as described in the **PDS 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 Syndication**.
2. For the **Server URL** parameter, select a URL from the dropdown.
3. For the **API User Name** parameter, type the user name with access to the API.
4. For the **API Password** parameter, type the password for the user name with access to the API.
5. Click the **Next** button to continue with the Configure Endpoint step, or the **Finish** button to close the wizard.

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 doesn't have to use XML to provide the response. You can find REST-based Web services that output the data in Command Separated Value (CSV), JavaScript Object Notation (JSON), and Really Simple Syndication (RSS). Supported formats include XML files using STEPXML or Generic XML.

If you want to learn more about REST, search for resources on the web.

For information about how to use the STEP REST API, click the **STEP API Documentation** button on the STEP 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.'

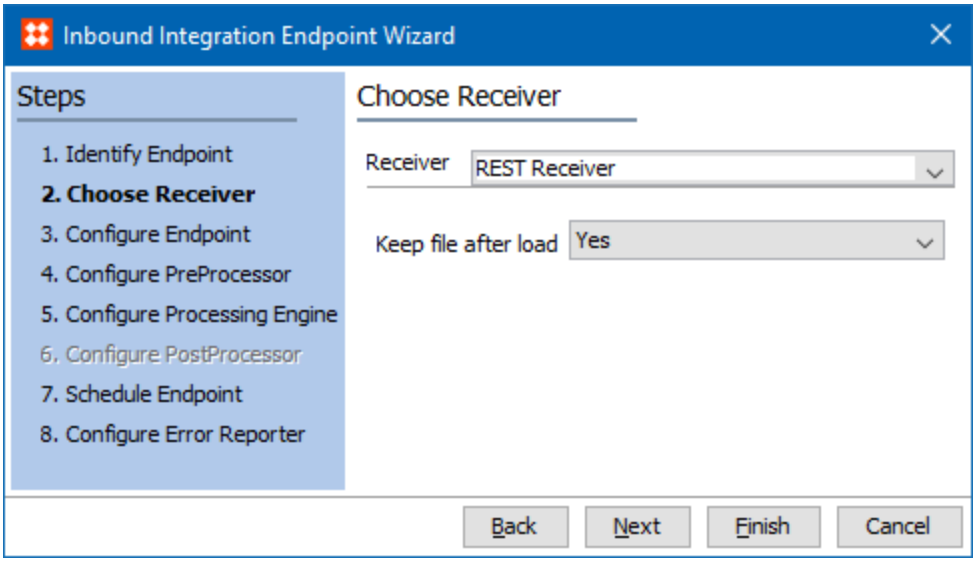
Notice that in the URL shown above, the context and workspace qualifiers have no significance, but they are required for the REST API.

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. Click the **Next** button to continue with the Configure Endpoint step, or the **Finish** button to close the wizard.

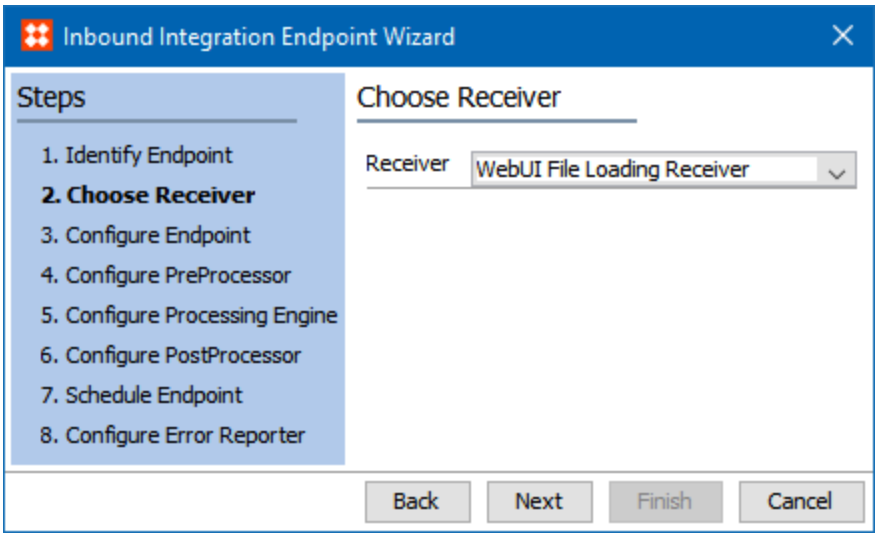
Web UI File Loading Receiver

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.

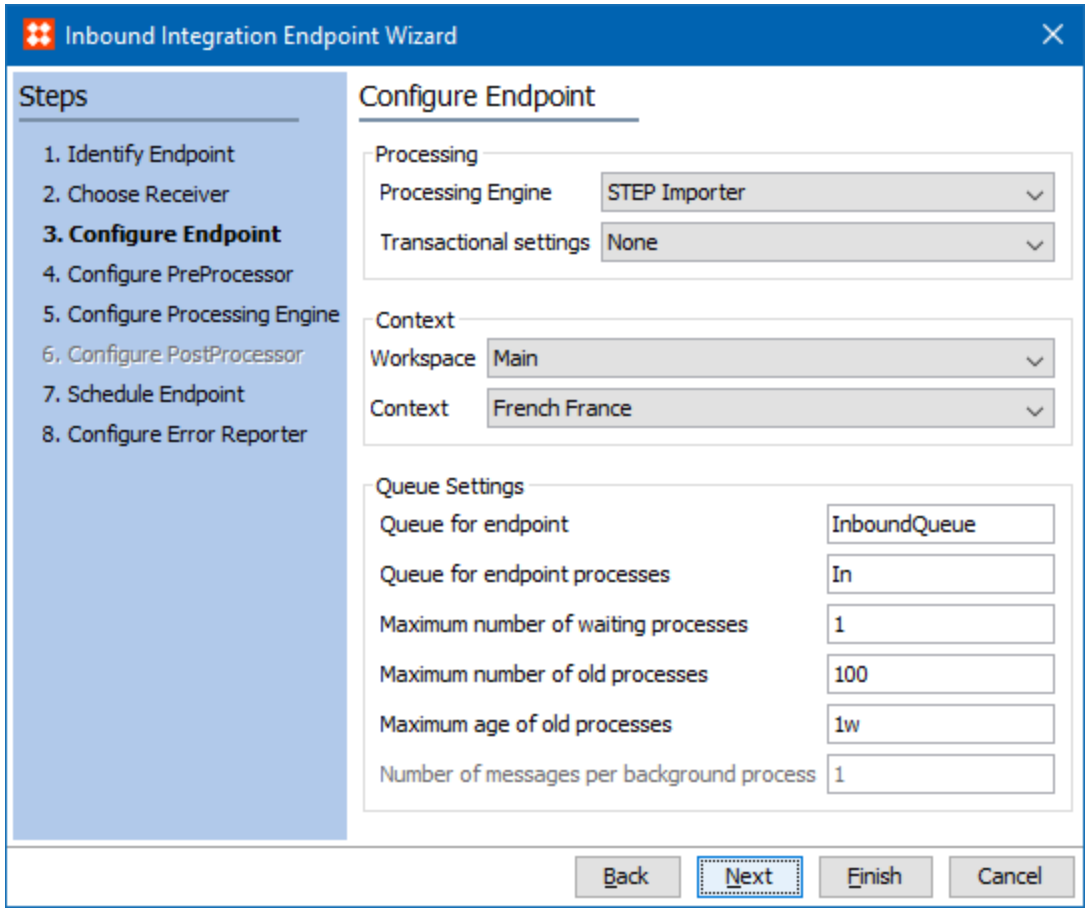
Configuration

Use the following parameters to configure the IIEP.



1. For **Receiver**, choose **WebUI 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 STEP 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 Syndication** - available when the 'productdatasyndication-integration' component is activated. No further configuration is required. For information, see the **Product Data Syndication** 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, see the **File Loading Widget** topic in the **Web User Interfaces** documentation.

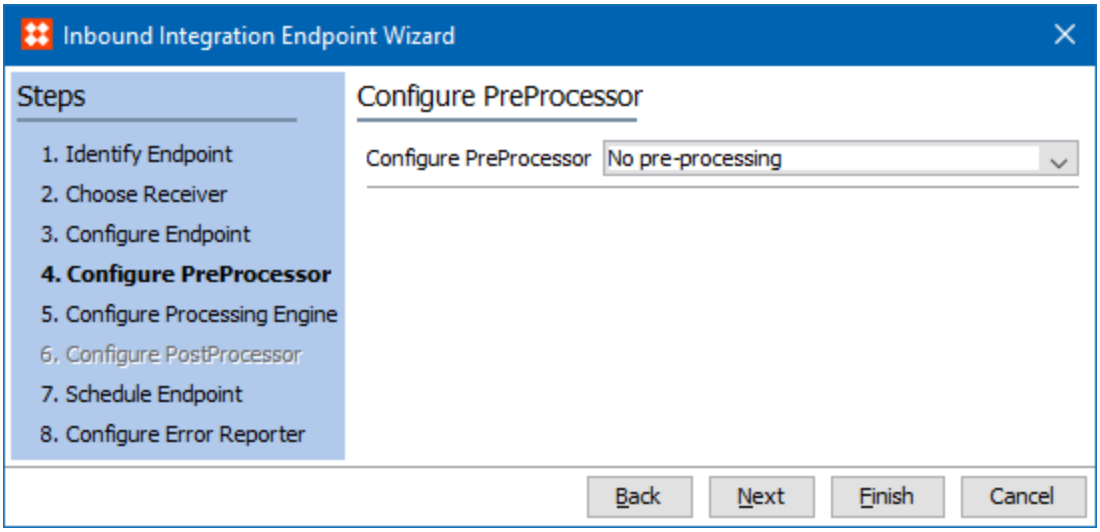
2. For **Transactional Settings** specify if the processes will be chain, strict, or none. For details on the effects of each option, see **Integration Endpoint Transactional Settings** documentation.
3. The queue size of a background process with strict or chain transactional setting must be 1. Only a transactional setting of None allows multiple background processes to work in parallel / concurrently. For more information, see **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.
   - For **Queue for endpoint**, enter a name for the queue that is used by the IIEP Background Process to poll the endpoint. 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. 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.

6. Click the **Next** button to display **IIEP - Configure PreProcessor** when available, or display **IIEP - Configure Processing Engine**.

IIEP - Configure PreProcessor



A preprocessor 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 preprocessor, contact Stibo Systems Support.

1. If necessary, choose a preprocessor and follow the steps in the linked topics for setup:

- **No pre-processing option** is selected by default.
- **XML Normalizer** preprocessor 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, see the **IIEP - Configure XML Normalizer PreProcessor** documentation.
- For details on the **XSLT** preprocessor, see **IIEP - Configure Transformation by XSLT PreProcessor** documentation.
- **Transformation by Import Configuration** can be used to transform Excel and CSV files to STEPXML on import.

2. Click the **Next** button to display **IIEP - Configure Processing Engine**.

IIEP - Configure Transformation by XSLT PreProcessor

The Transformation by XSLT preprocessor 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 preprocessor successfully, the incoming file or message size must be 50 MB or smaller.

Prerequisites

Before configuring the Transformation by XSLT Preprocessor, 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 Preprocessor** section below.

Initial Setup for XSLT Stylesheets

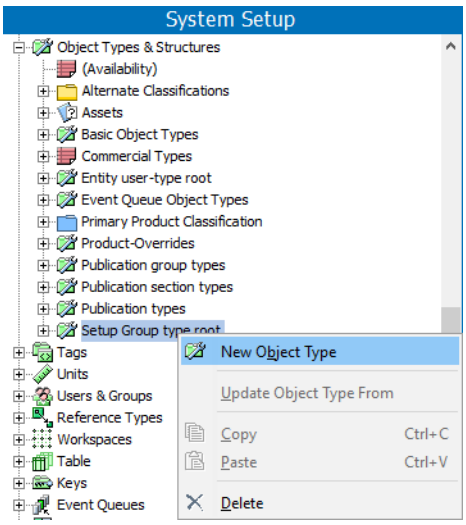
Creating an XSLT stylesheet setup group must only be performed once, unless additional levels of organization are desired.

Navigate to System Setup > Object Types & Structures > Setup Group type root, and determine if an entry exists for the XSLT Stylesheet.

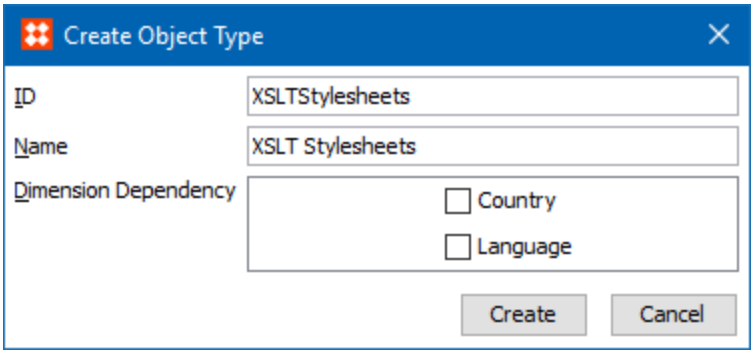
- If an XSLT Stylesheet setup group does not exist, continue with the steps below to create a setup group, link the XSLT object type, and create an instance of the XSLT object.
- If an XSLT Stylesheet setup group exists, but has no children, proceed to the **Link the XSLT Stylesheets object type to setup 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

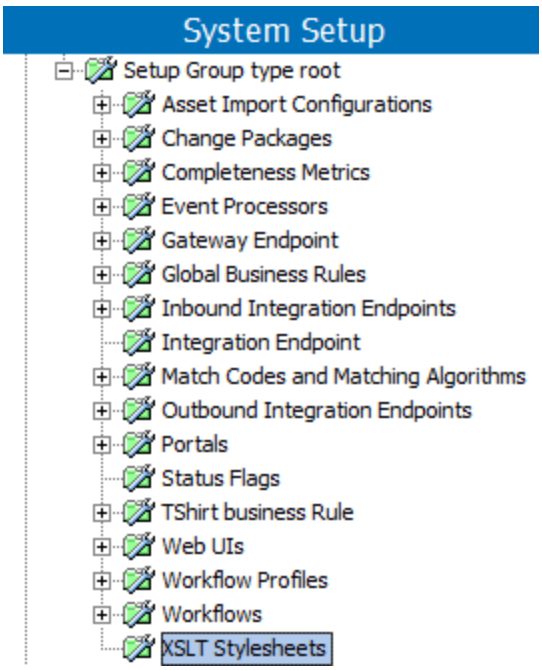
1. In System Setup, expand **Object Types & Structures**.
2. Right-click 'Setup Group type root', and choose **New Object Type**.



3. Enter an **ID** and a **Name**, select any required Dimension Dependencies, and click **Create**.

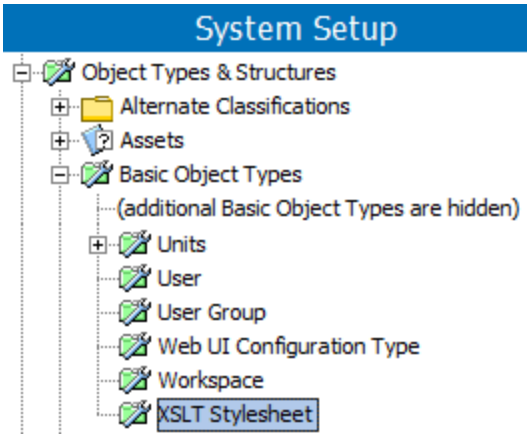


4. The new setup group appears in System Setup under 'Object Types & Structures' as a child in the **Setup Group type root**.

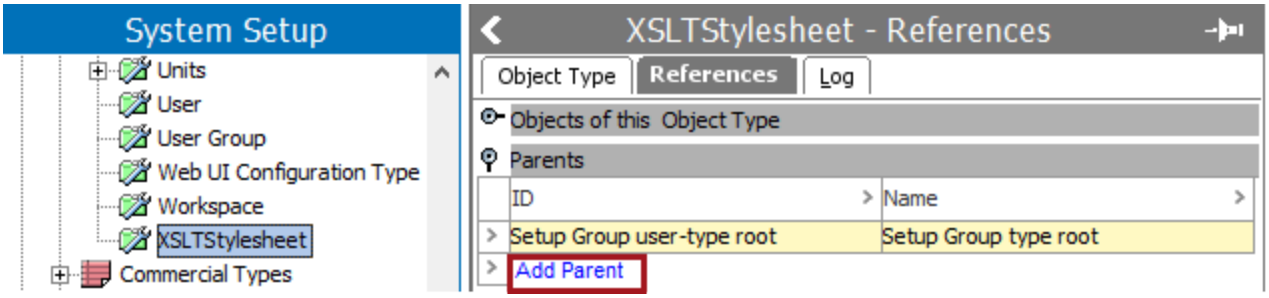


Link the XSLT stylesheet object type to the XSLT setup group

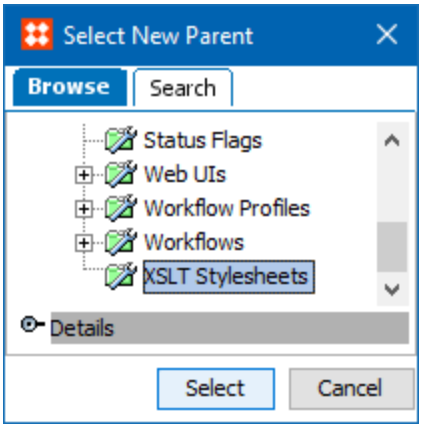
1. In Object Types & Structures > expand Basic Object Types > select **XSLT Stylesheet**.



2. On the References tab > Parents flipper > click the **Add Parent** link.



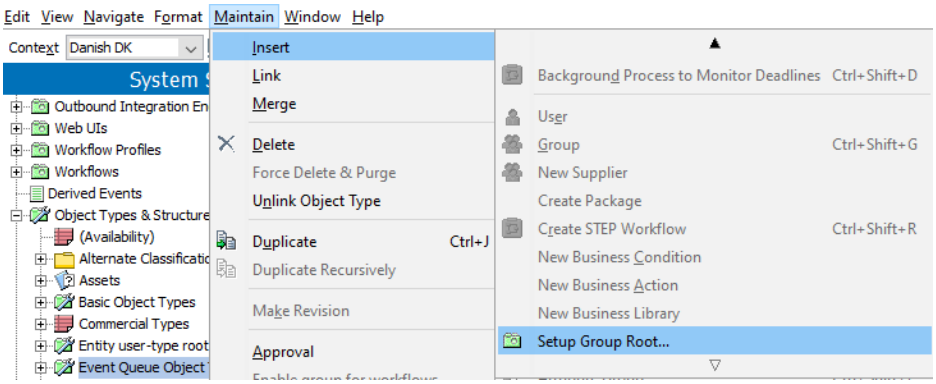
3. In the Select New Parent dialog, select the setup group type root you created, and click **Select** to make it a valid parent.



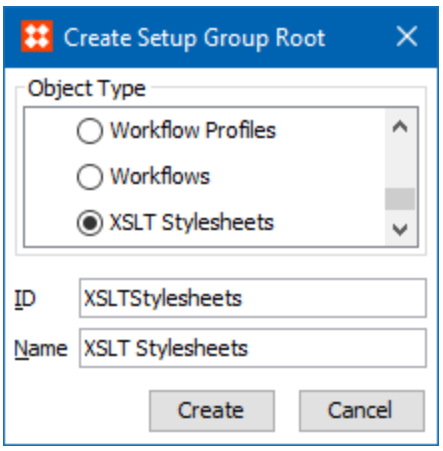
The Setup Group type root node now displays XSLT Stylesheets with a child of XSLT Stylesheet.

Create an instance of the XSLT Stylesheet object

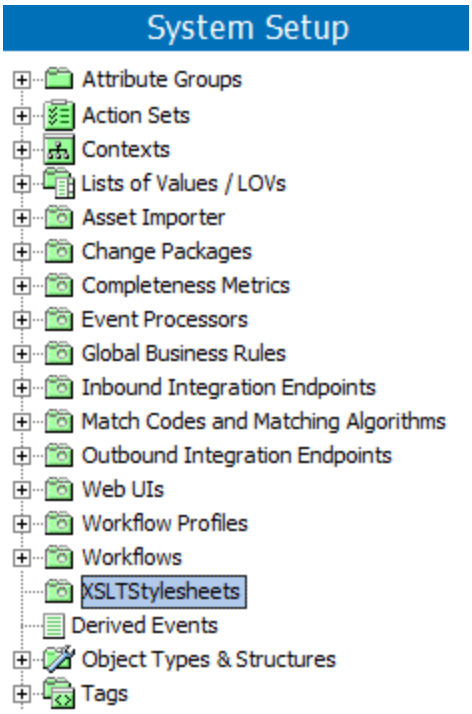
1. On the System Setup tab, select any object in the **System Setup** hierarchy to activate the following **Maintain** menu selection.
2. Click the Maintain menu, point to Insert, and select **Setup Group Root**.



3. In the Create Setup Group Root dialog, select the XSLT Stylesheets object type, enter an **ID** and a **Name**, and click **Create**.



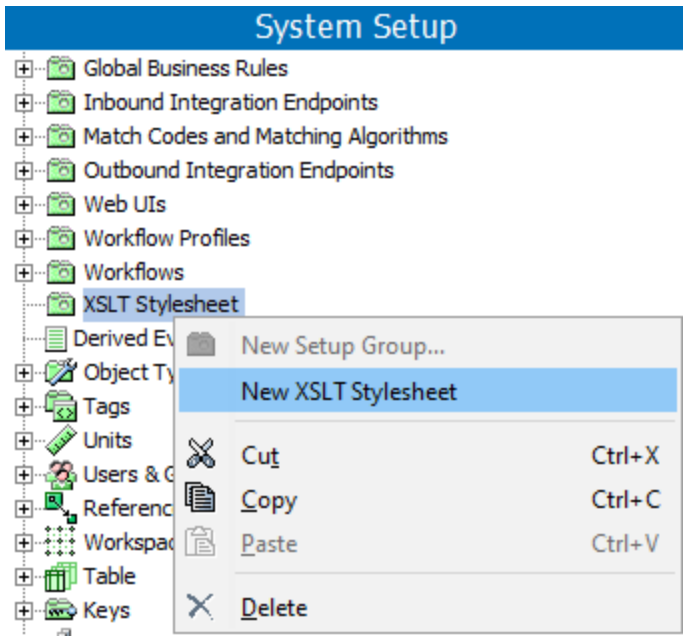
A setup group is created as a node in the System Setup hierarchy. XSLT Stylesheets can now be created under this new node.



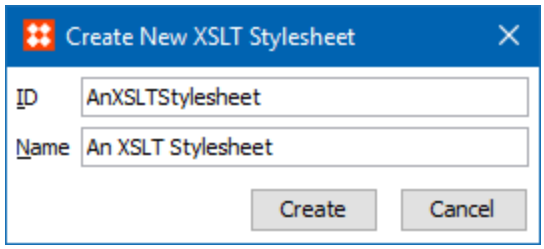
Create an XSLT Stylesheet Object

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 preprocessor.

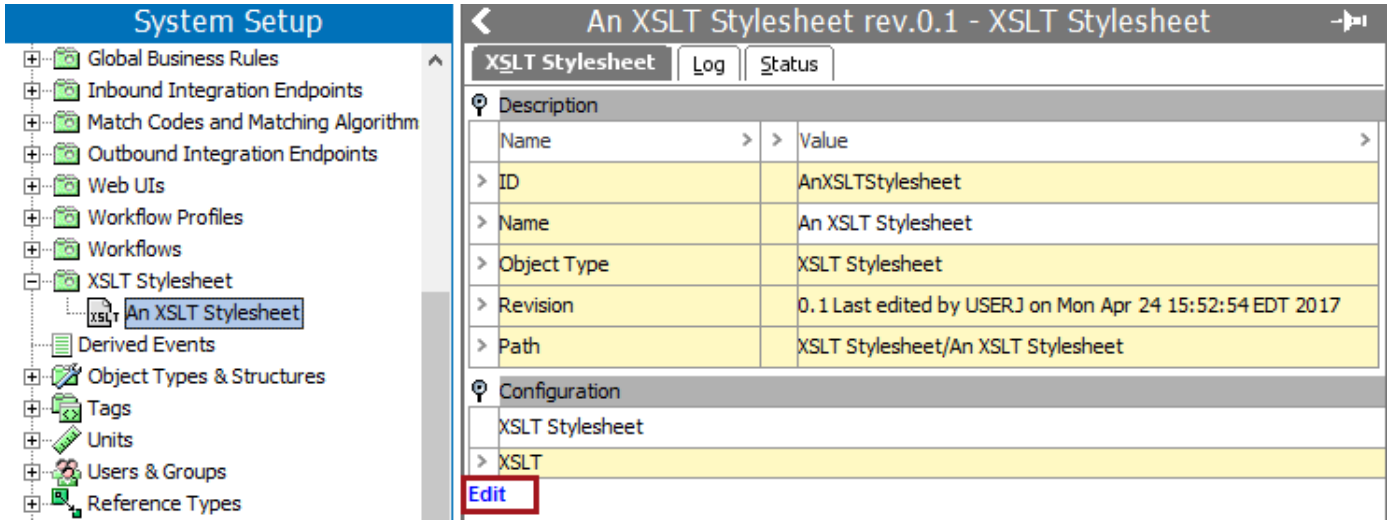
1. In System Setup, select your XSLT Stylesheet node, right-click, and choose **New XSLT Stylesheet**.



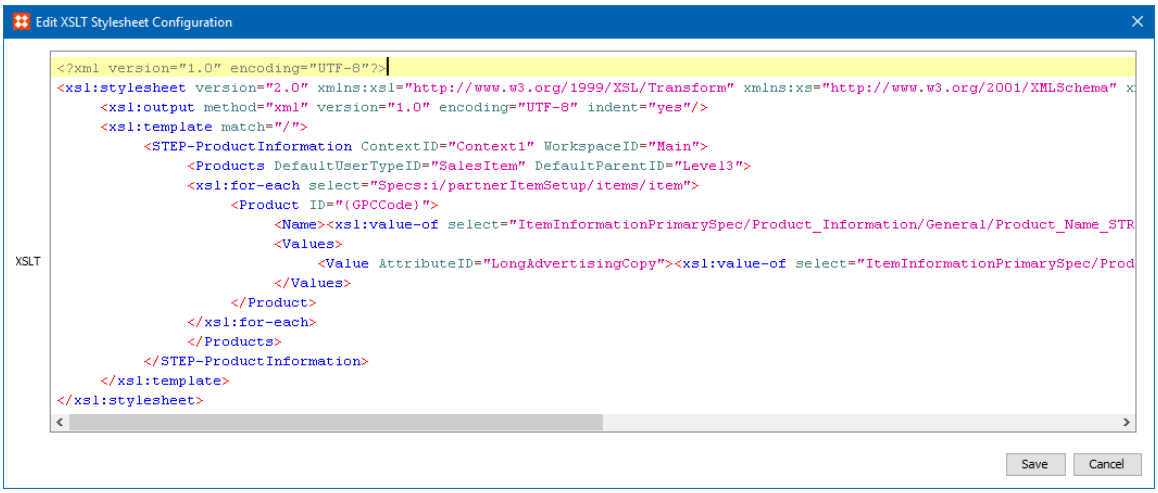
2. Enter an **ID** and a **Name**, and click **Create**.



3. Select the new stylesheet object that displays below the XSLT node and click the **Edit** link.



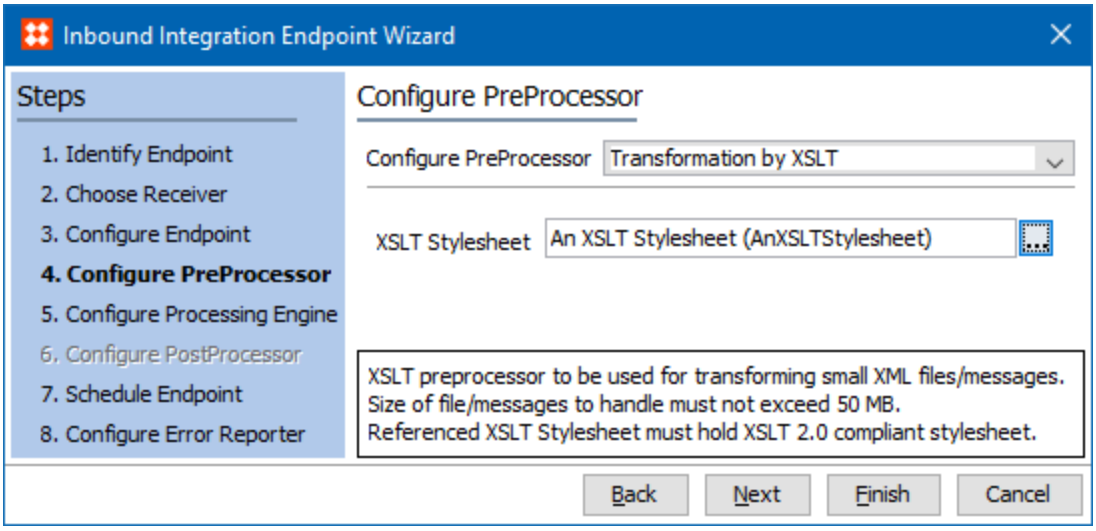
4. Type or copy your XSLT 2.0-compliant stylesheet into the text box, and click the **Save** button.



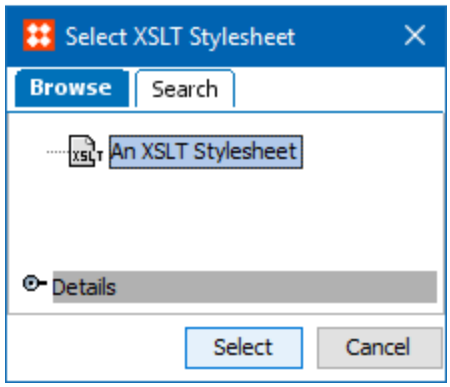
The XSLT stylesheet is now ready for selection in the IIEP wizard.

Configure the Transformation by XSLT Preprocessor

Once the prerequisite tasks have been completed, you are ready to configure the preprocessor in the IIEP wizard.



1. In the **Configure PreProcessor** 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 PreProcessor

While this preprocessor 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.

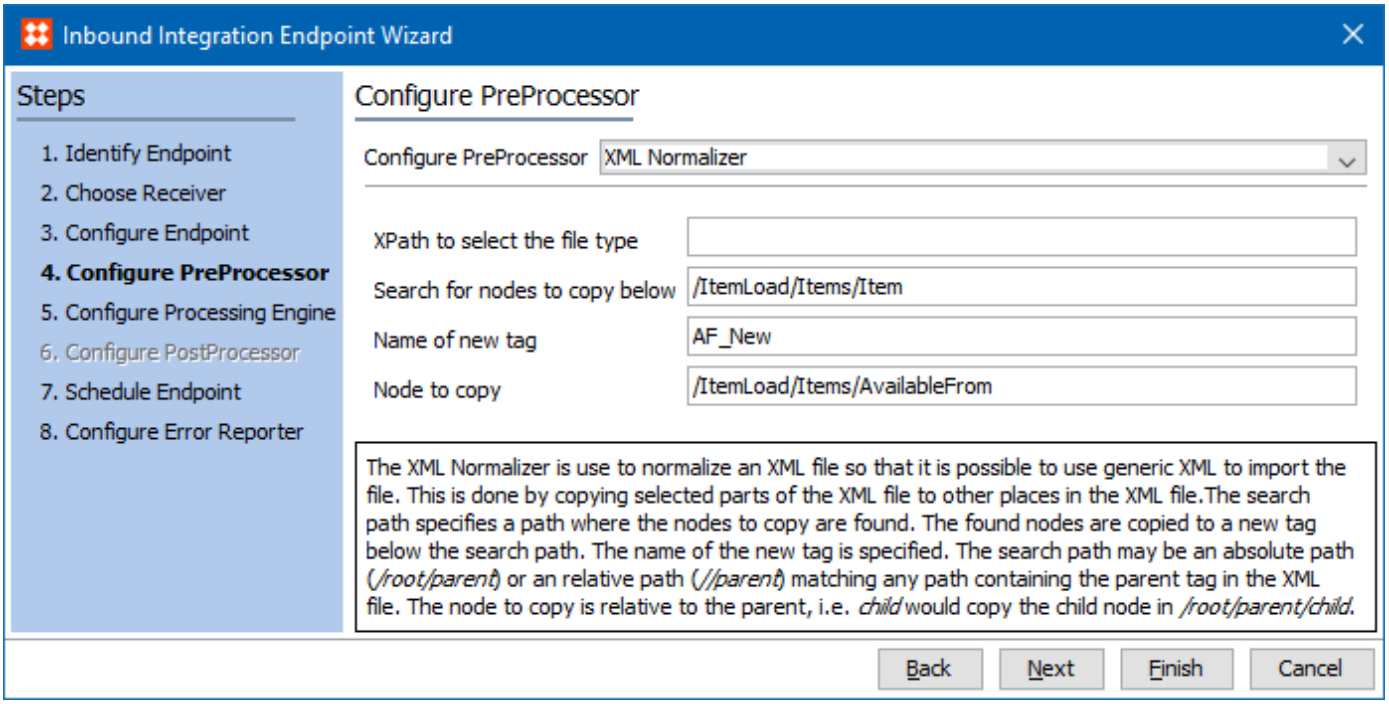
Important: For a successful import, verify that the node data is valid to exist in the new location.

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>
```

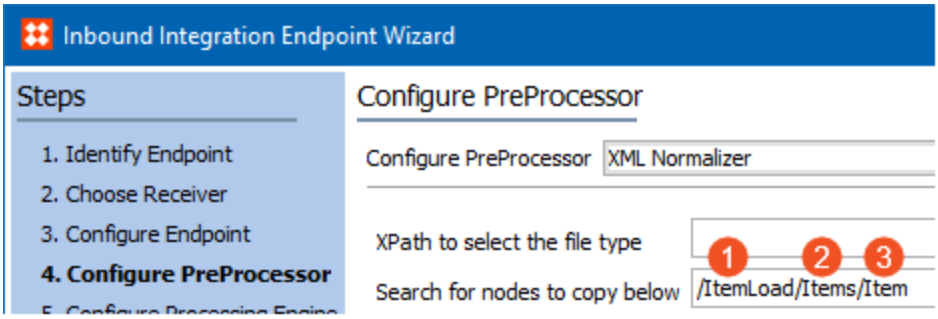
Configure and Use the XML Normalizer

1. In the **Configure PreProcessor** 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.

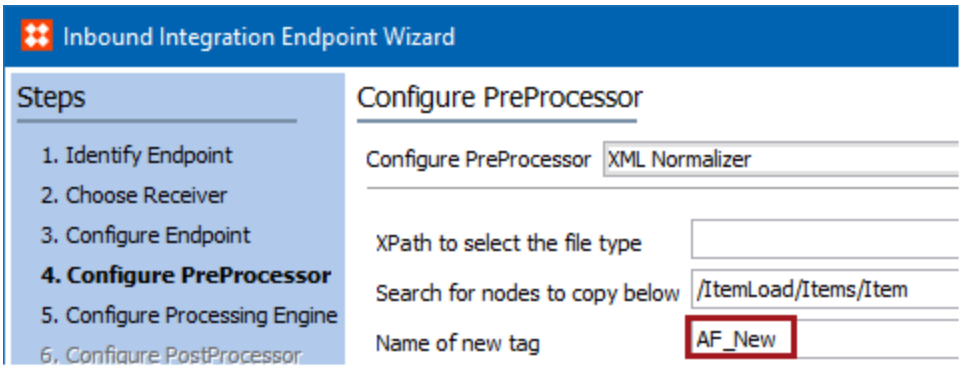
  Using the original example input file above, the path /ItemLoad/Items/Item indicates that the copied node will be placed within the 'Item' node.




```
<?xml version="1.0" encoding="utf-8"?>
1 <ItemLoad>
2  <Items>
    <AvailableFrom>2015-01-01 00:00:00</AvailableFrom></AF_New>
3   <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, see **IIEP - Configure STEP Importer Processing Engine** documentation.

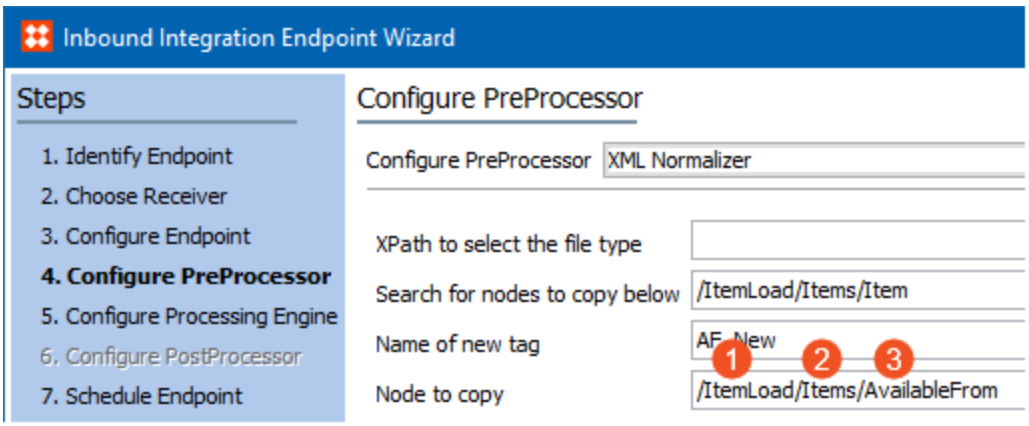
  In the wizard image, 'AF_New' is the tag that will be added as a prefix to the information being copied. The normalized XML file below shows the newly copied tag with the '<AF_New>' wrapper.




```
<?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"?>
1 <ItemLoad>
2   <Items>
3   <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, see 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, see 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>
```

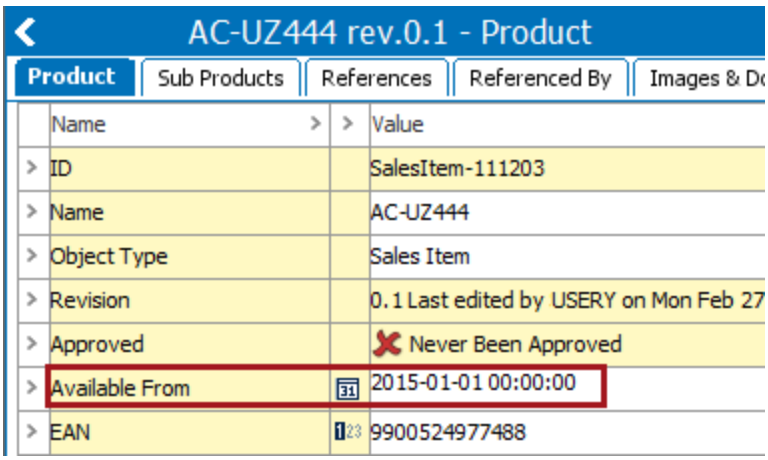
5. Click the **Next** button to display **IIEP - Configure Processing Engine**.

Preprocessor Results

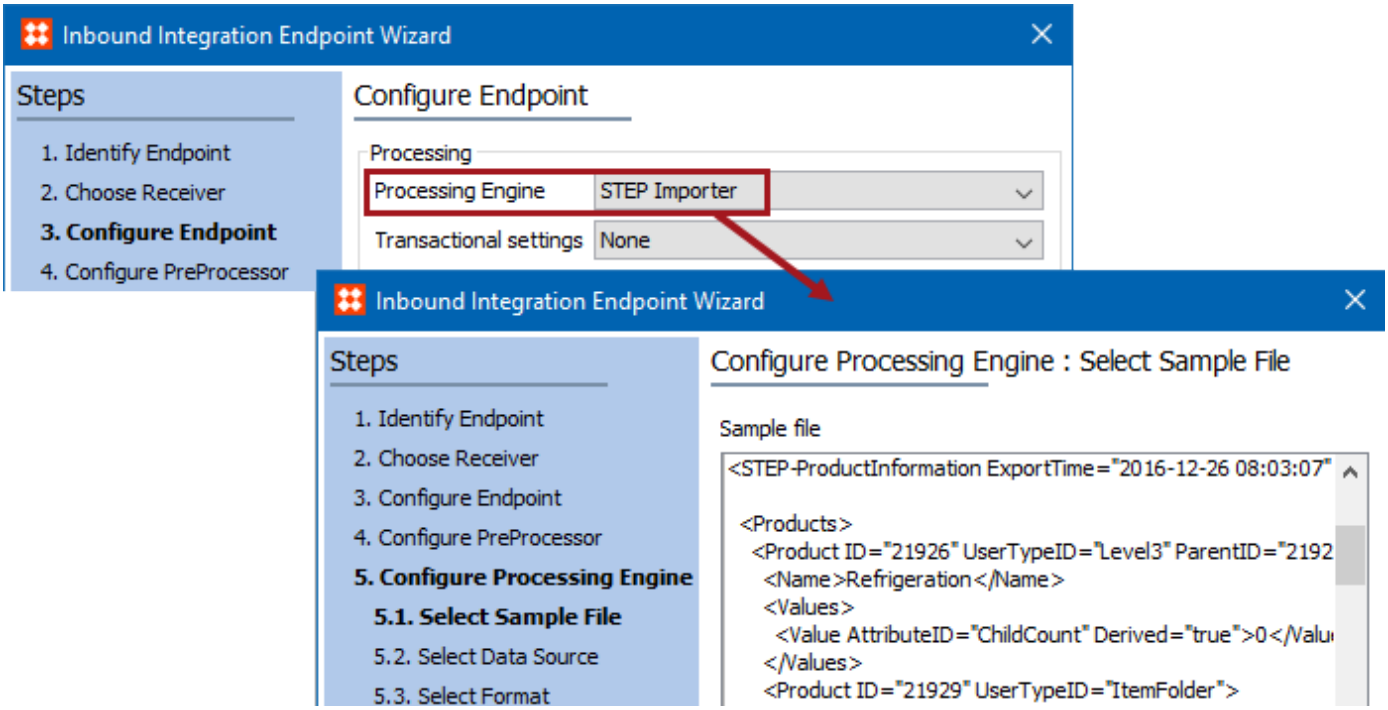
Continuing the original example, the input file is normalized by the preprocessor as follows:

```
<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 & Dc
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



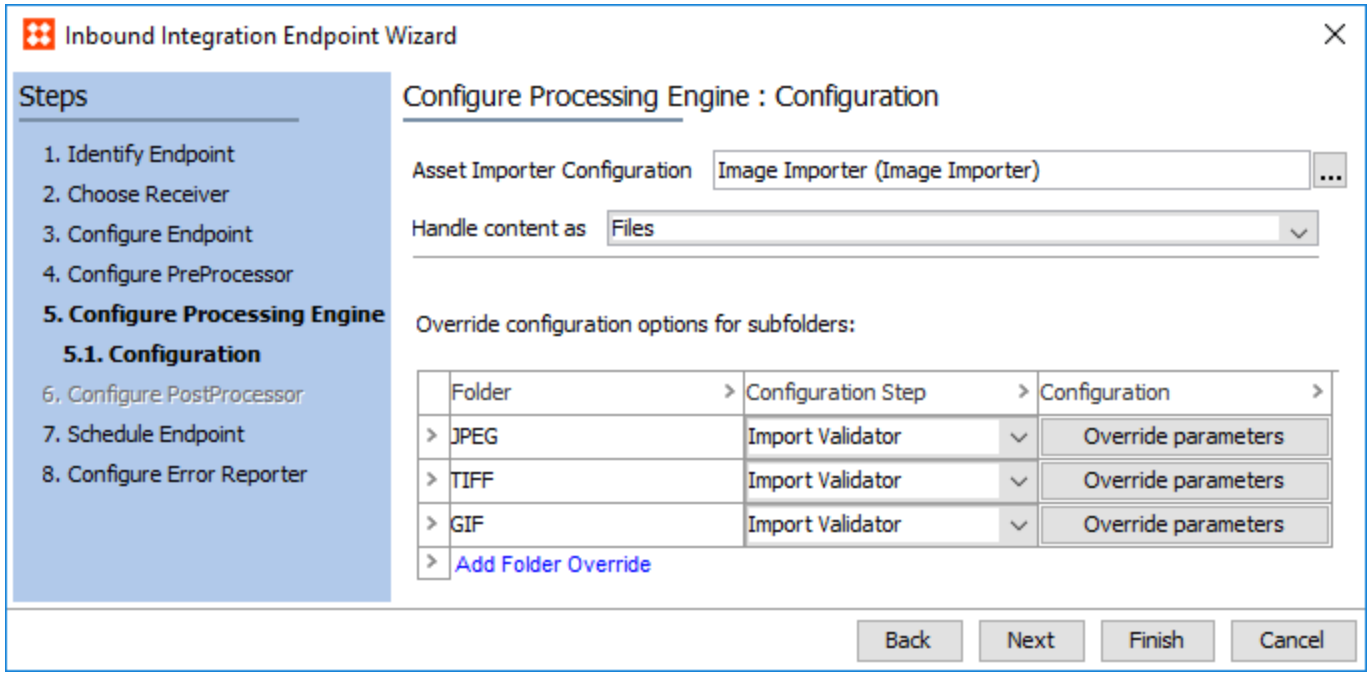
The Processing Engine is responsible for performing the actual import, and can be replaced via custom extension, for example, if some special format not supported in the standard STEP Importer is to be handled.

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 Syndication - no further configuration is required. For information, see the **Product Data Syndication** topic.
   - STEP Match and Merge Importer - only available when the Matching & Merging deduplication solution is implemented. For information, see the **IIEP - Configure Match and Merge Importer** topic.
2. Click the **Next** button to display **IIEP - Configure PostProcessor**when available, or display **IIEP - Schedule Endpoint**.

IIEP - Configure Asset Importer Processing Engine



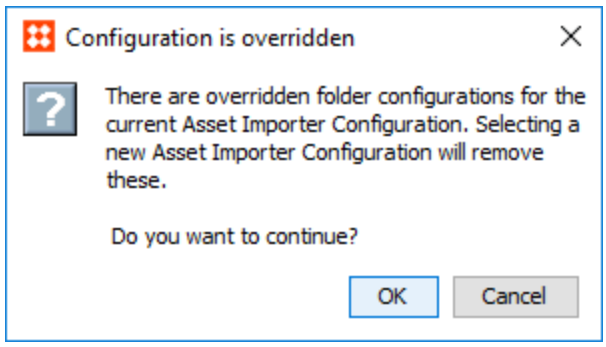
When using an Asset Importer endpoint, it is necessary to specify an asset importer configuration and define how the imported content will be received.

Prerequisites

The following settings are important when using this processing engine:

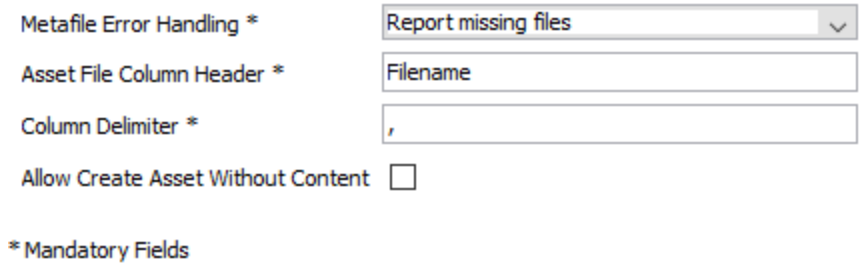
- On the Choose Receiver step, the 'Hotfolder Receiver' option must be selected as defined in the **IIEP - Choose Receiver** topic.
- PostProcessors are not available for the Asset Importer.
- If desired, folder overrides can also be configured, allowing users to replace certain aspects of the asset import configuration for sub folders within the top-level hot folder. For required configuration, see 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). See 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.

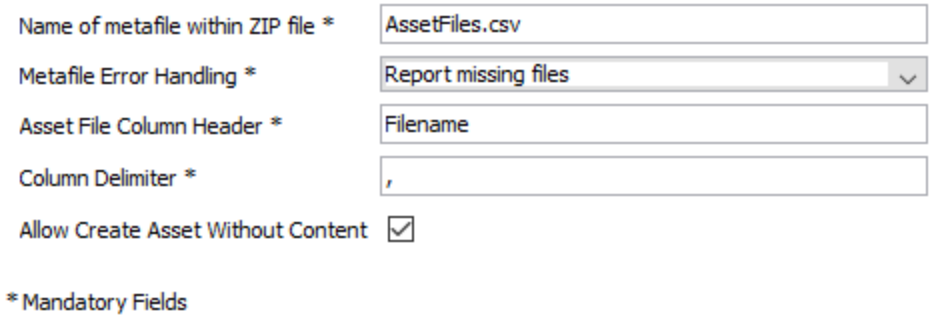
In the **Asset File Column Header** parameter, specify the column heading of the metafile that contains the asset information. This includes an asset name; an external URL; or a relative path if the corresponding asset is in a subfolder below the root hotfolder directory. For example, 'Supplier1/Asset.jpg' 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. See 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.

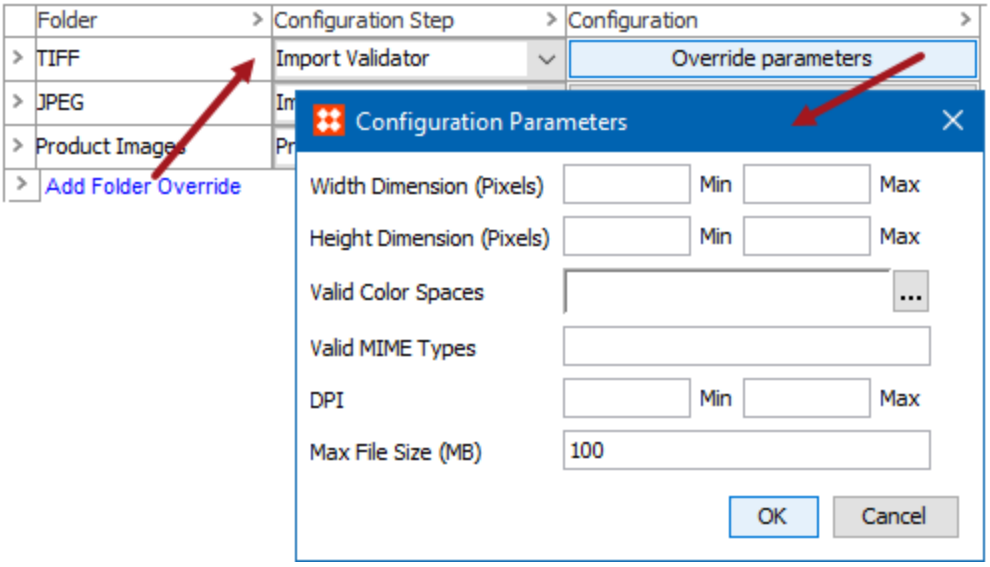
In the **Column Delimiter** parameter, specify which character should be used to separate each of the columns in the metadata file. The default value is a coma (,) due to backwards compatibility, but it is recommended to use either a semicolon or a pipe character, which is less likely to be included 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, see 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, see the **Product Linker** section of the **Asset Importer** documentation

Examples

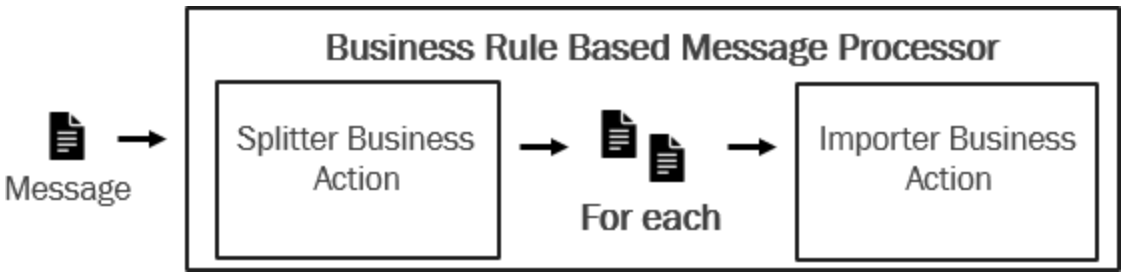
1. Including metadata info to import (Metadata Importer):
   - **Filename;Product;Reference;Title;Photographer;PhotoCategory**
   - 1234.jpg;Item-12345;PrimaryImage;Glacier Bay Alaska;Peter Parker;Landscapes
2. Using a relative path (Metafile Receiver)
   - **Filename;Product;Reference**
   - Supplier1/1234.jpg;Item-12345;PrimaryImage
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, see 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.

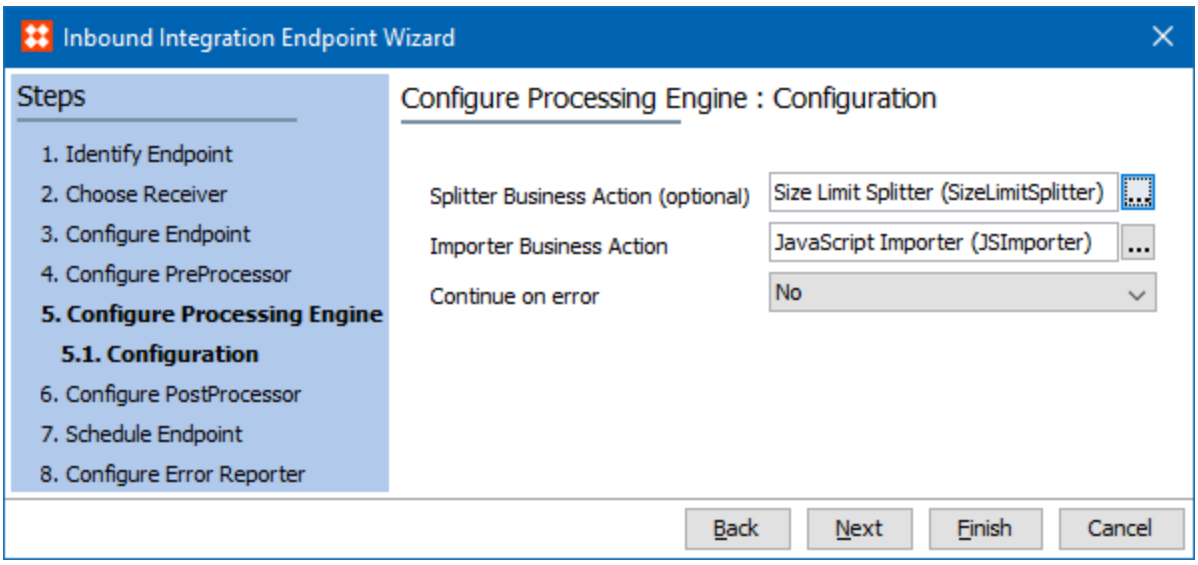
Important: Given the flexibility of this functionality, it is easy to write importer business action code that will not perform well. Generally, avoid long-running queries, extensive traversal of hierarchies, and business actions holding large data structures in memory. Imports via the Business Rule 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. Business actions used by this processor should be configured to be valid for all object types.
2. 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.



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, see 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, see 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 PostProcessor** when available, or **IIEP - Schedule Endpoint**.

Splitter Business Action Details and Examples

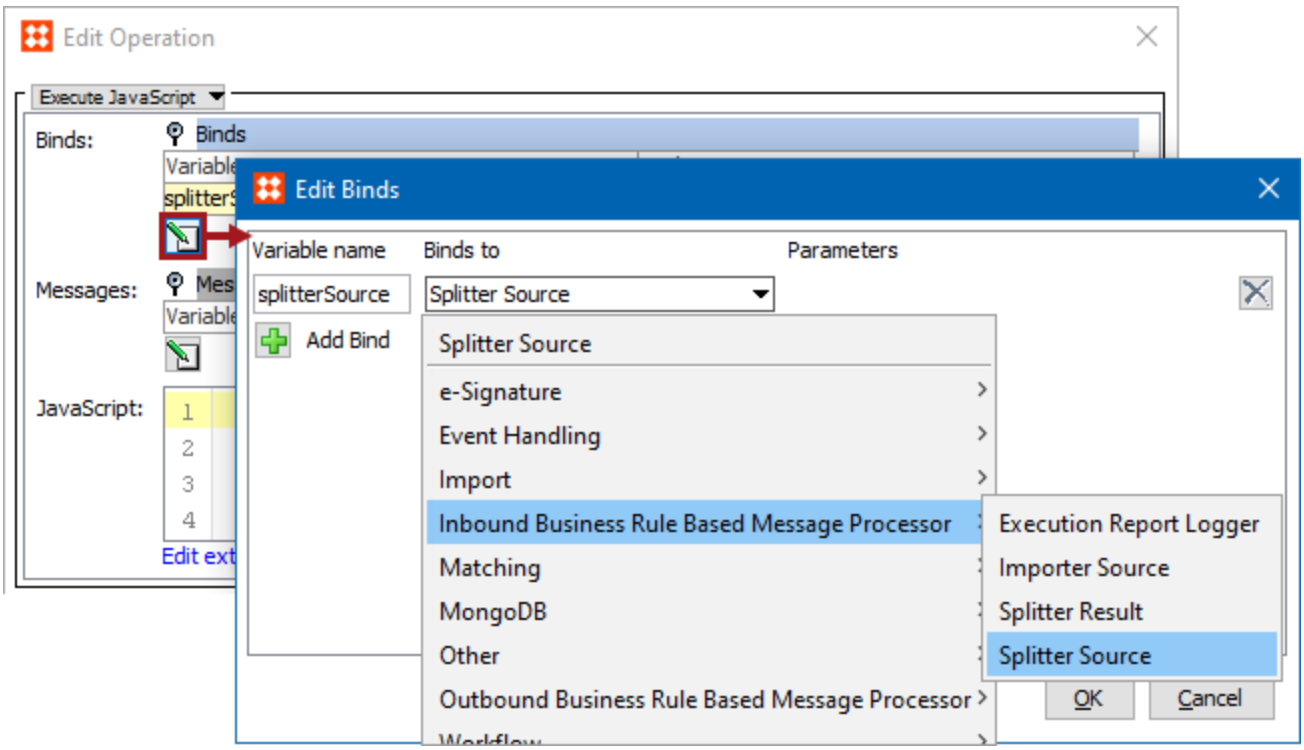
The **InboundBusinessProcessorSplitterSource** interface (for JavaScript business actions, available via the Splitter Source bind option) provides access to the inbound file / message for the splitter business action.

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 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

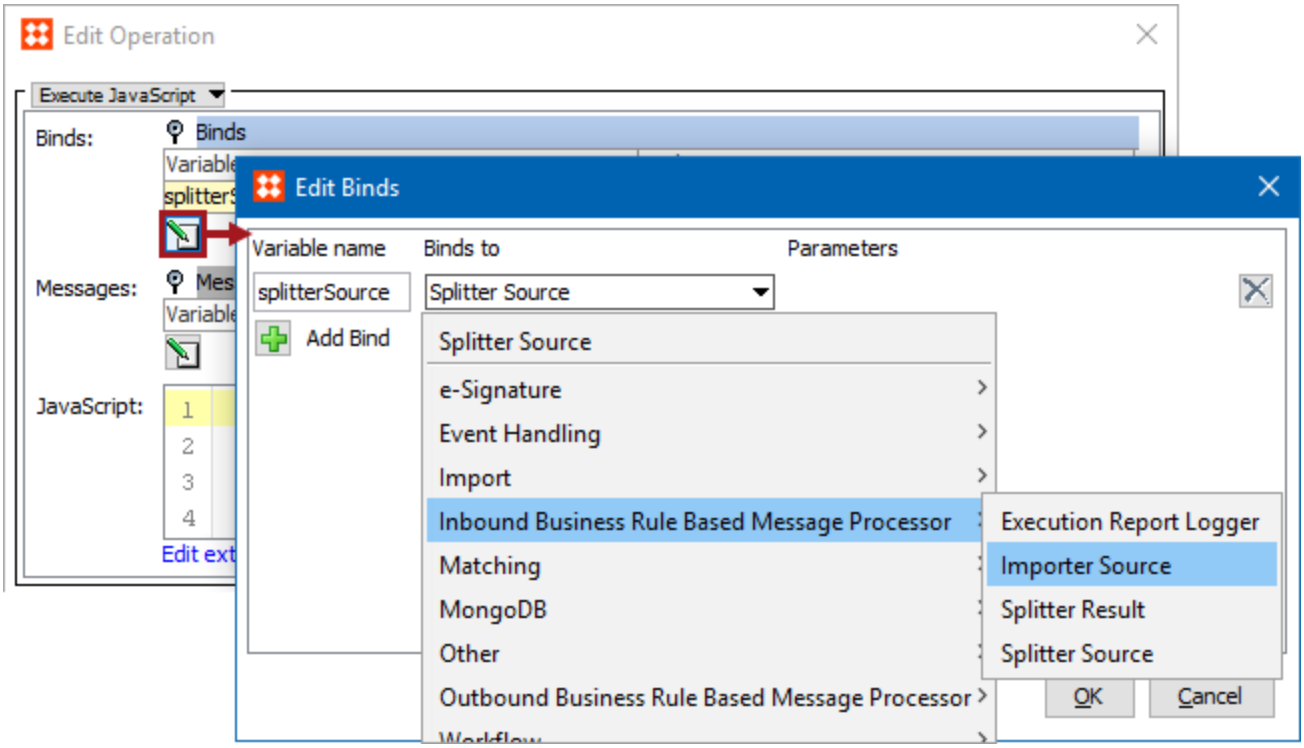
```
// Splitter source bound to splitterSource
// Splitter Result bound to splitterResult

var ite = splitterSource.lines();
while (ite.hasNext()) {
  inboundMessages.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 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"
}
```

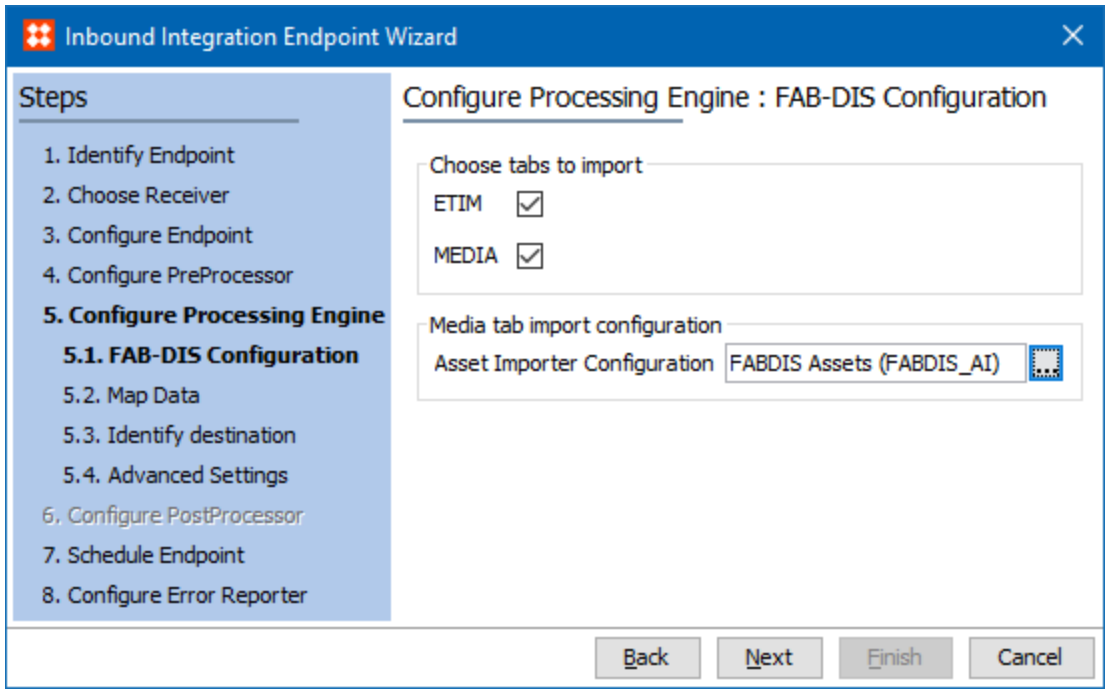
Sample script

```
// 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) {
```

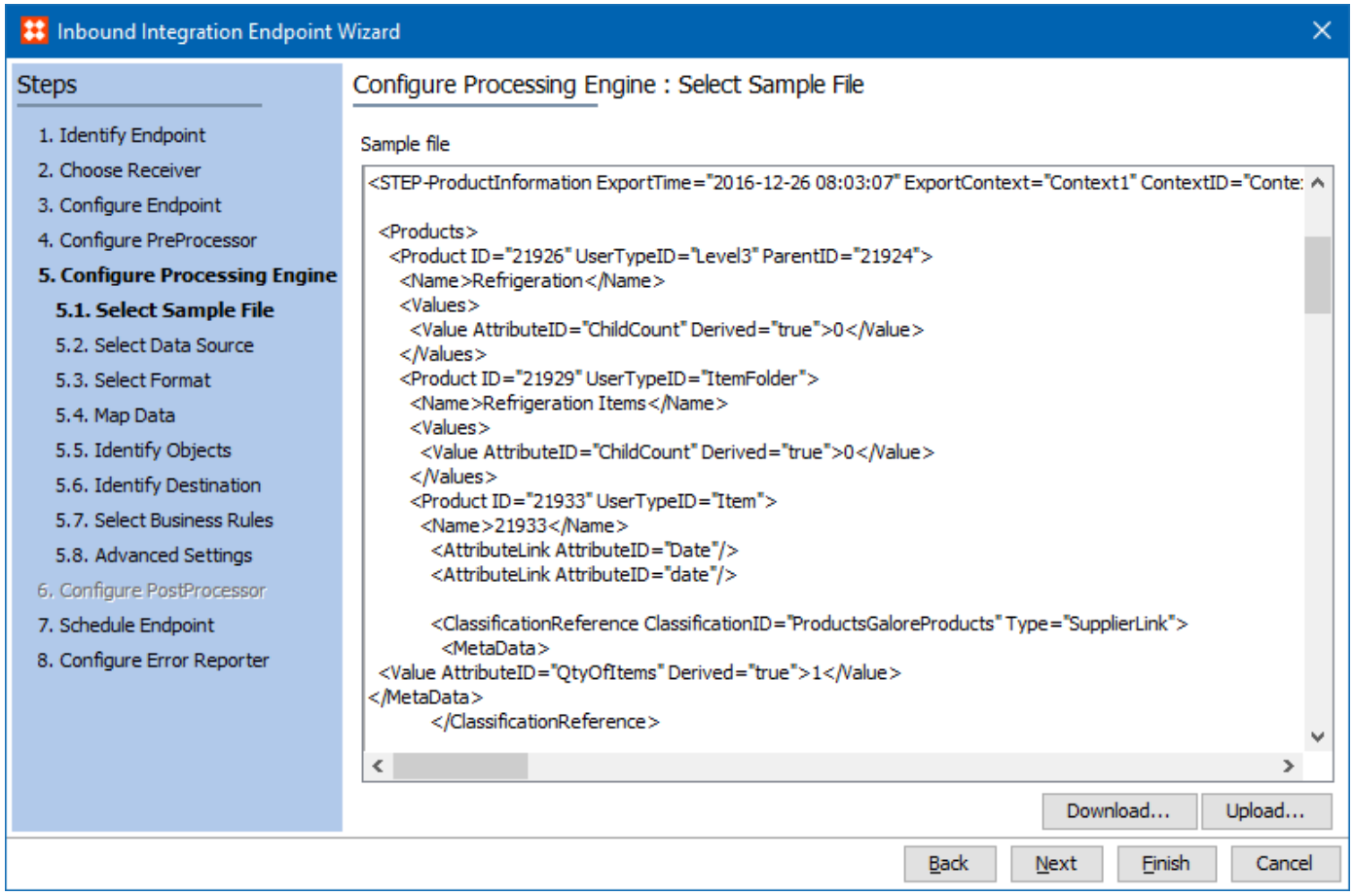
```
      throw new java.lang.RuntimeException("Could not create new product: " +
e.javaException.getMessage());
    }
    throw(e);
  }
  setValue(prod, "EAN", ean, true);
}
if (!prod.getParent().equals(category)) {
  try {
    prod.setParent(category);
  } catch (e) {
    if (e.javaException instanceof
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, see 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 STEP 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 preprocessor transforms the file to import, use a sample file that matches whatever format is produced by the preprocessor. If the sample file is Excel, no preview is shown.

- When using the **Transformation by XSLT** preprocessor, 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, see **IIEP - Configure Transformation by XSLT Preprocessor** documentation.
- When using the **XML Normalizer** preprocessor, verify that the sample file includes both the new tag, and the copied tag data in the correct location. For more information, see **IIEP - Configure XML Normalizer PreProcessor** 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, see topics in the **Commercial Data** section of the **STEP Publisher (InDesign)** 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.

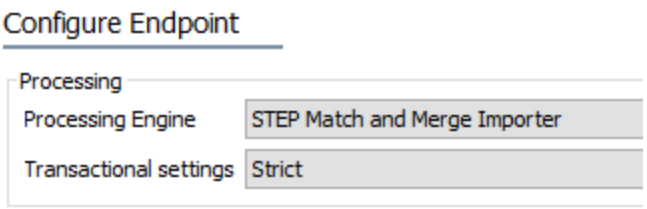
   When using the XML Normalizer preprocessor, verify that the template file includes both the new tag, and the copied tag data in the correct location. For more information, see **IIEP - Configure PreProcessor** documentation.

   - 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, see 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, see **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, see **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, see **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, see **Import Manager - Advanced Settings**.
10. Click the **Next** button to display **IIEP - Configure PostProcessor** when available, or display **IIEP - Schedule Endpoint**.

IIEP - Configure Match and Merge Importer

The STEP Match and Merge Importer plays an integral role in the Matching & Merging deduplication solution, and is responsible for importing, processing, and standardizing data, and ultimately matching that data with existing golden records. When using the STEP Match and Merge Importer in an IIEP, it is not possible to make use of a postprocessor.

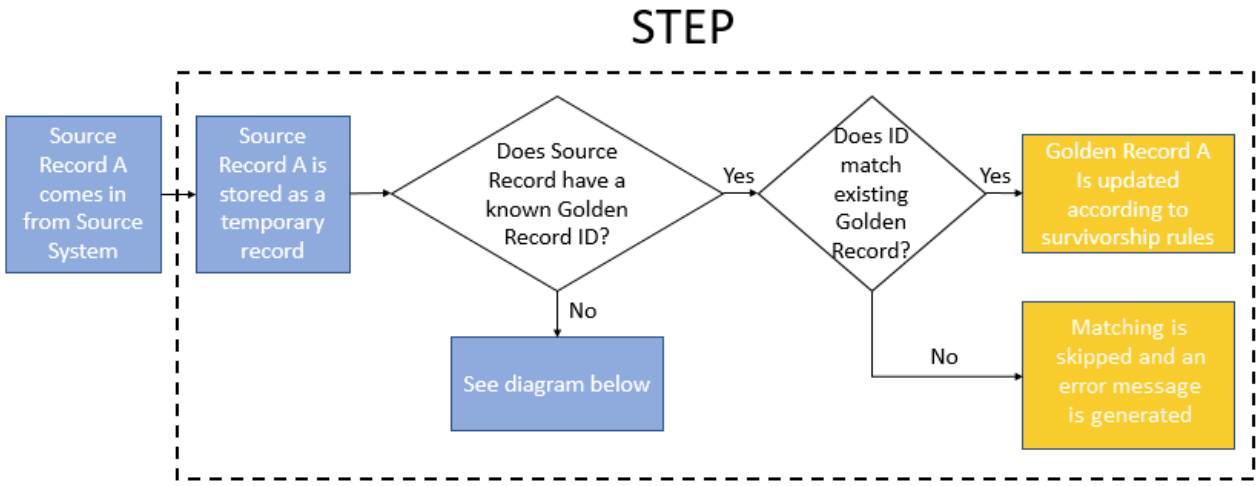


Match and Merge Importing Process and Logic

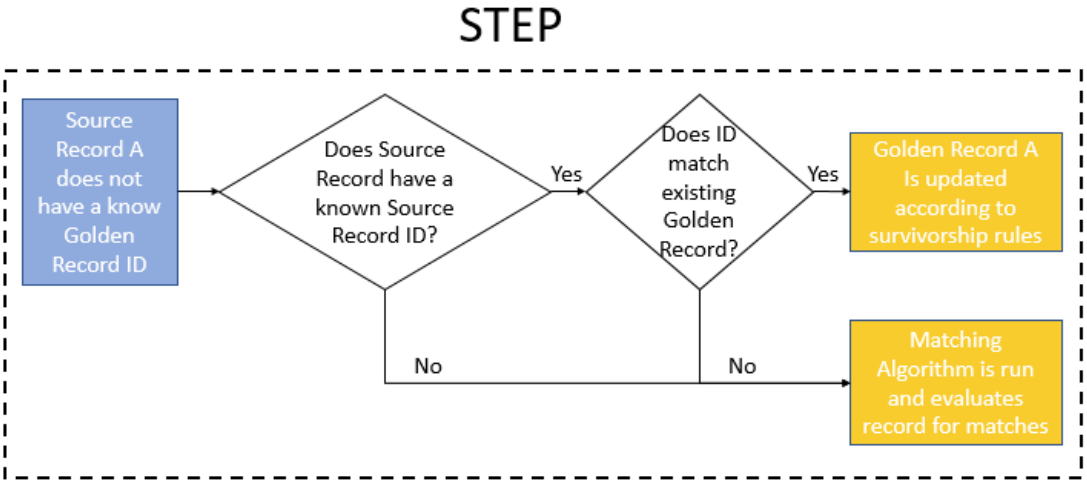
During an import, source records listed in the imported file are created, and treated as temporary STEP objects. Despite their temporary nature, business rules can act on them, and they can be matched via a matching algorithm. When being matched, survivorship rules determine which (if any) source values get promoted to existing golden records. After the import process is complete, these temporary objects are discarded.

Imported data is processed in the following order:

1. The IIEP checks the temporary source record to see if it has a known golden record ID. If it does have an ID, then the corresponding golden record in STEP is updated based on the survivorship rules configured in the matching algorithm. If no existing golden record shares the same ID, then the source record is ignored and is not imported to a golden record. A relevant error message is provided.



2. The IIEP checks the temporary source record to see if it has a known source record ID. If an existing active (or not deactivated) golden record is found that has the same source ID, survivorship rules are applied to promote source values to the golden record.



3. If neither a golden record ID or source record ID is found, the configured matching algorithm is invoked, and matches the temporary source records to existing golden records in STEP. If a source record matches one or more golden records above the auto threshold, then survivorship rules are applied to promote source values to the golden record that is matched with the highest score (if multiple golden records match above auto threshold, then golden records are not merged in this process). Those temporary source records that are matched below the auto threshold become their own golden records.

Note: Because the imported records are temporary STEP objects, it is not possible to query for children of said records during the import. Furthermore, the importer cannot generate match codes or run survivorship rules on children of the incoming records.

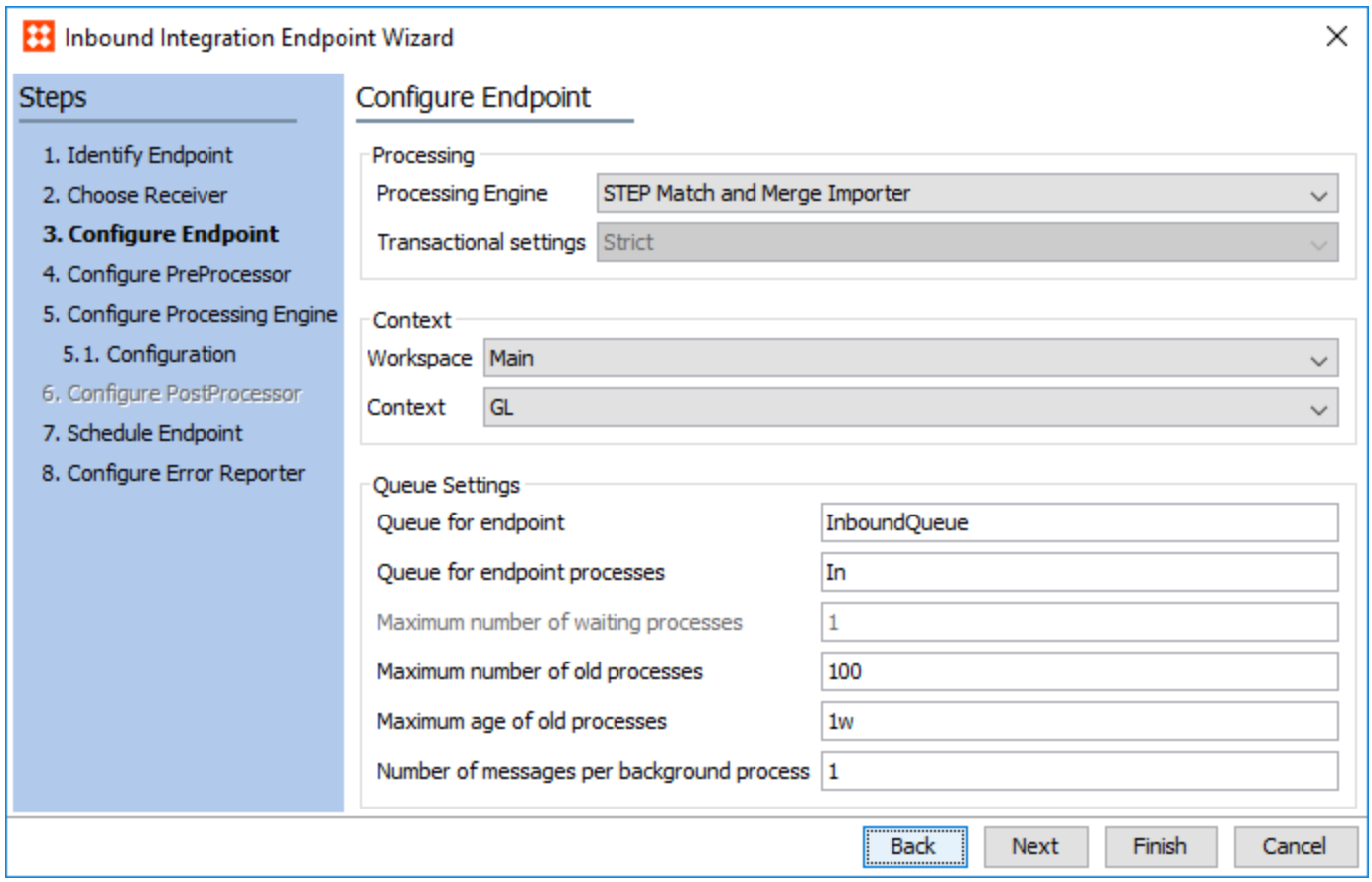
Configuration Wizard Steps

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.

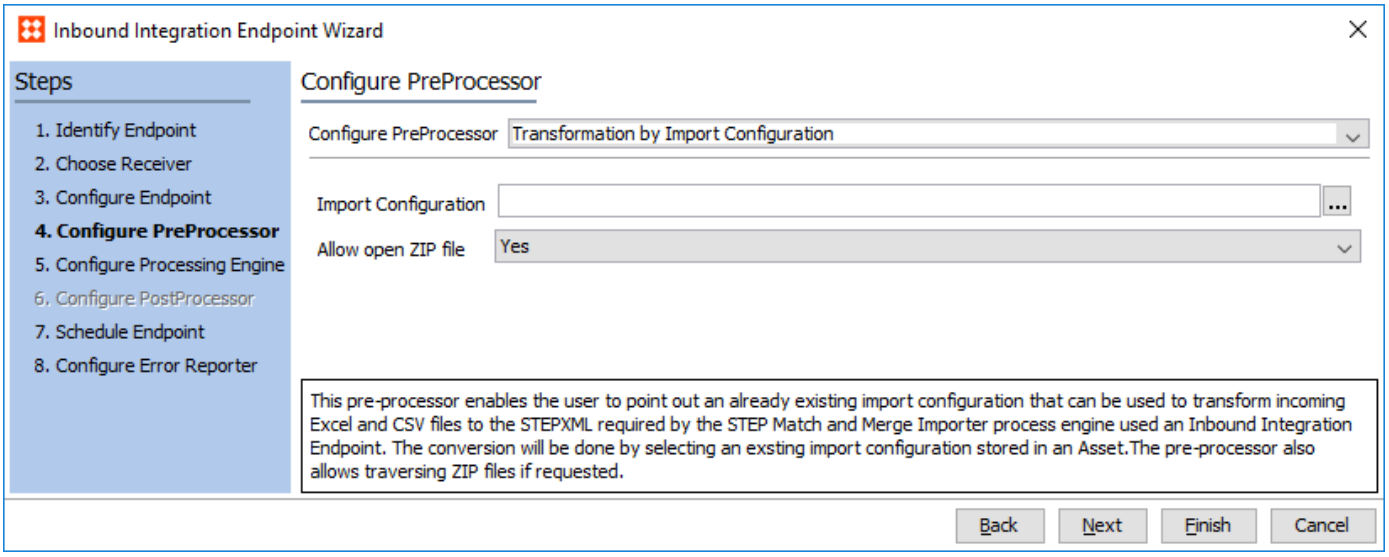
   For details on how the other parameters in this step may be configured, see the **IIEP - Configure Endpoint** documentation.



2. On the **Configure PreProcessor** 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 PreProcessor' dropdown, select 'Transformation by Import Configuration.'

     When making use of 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.

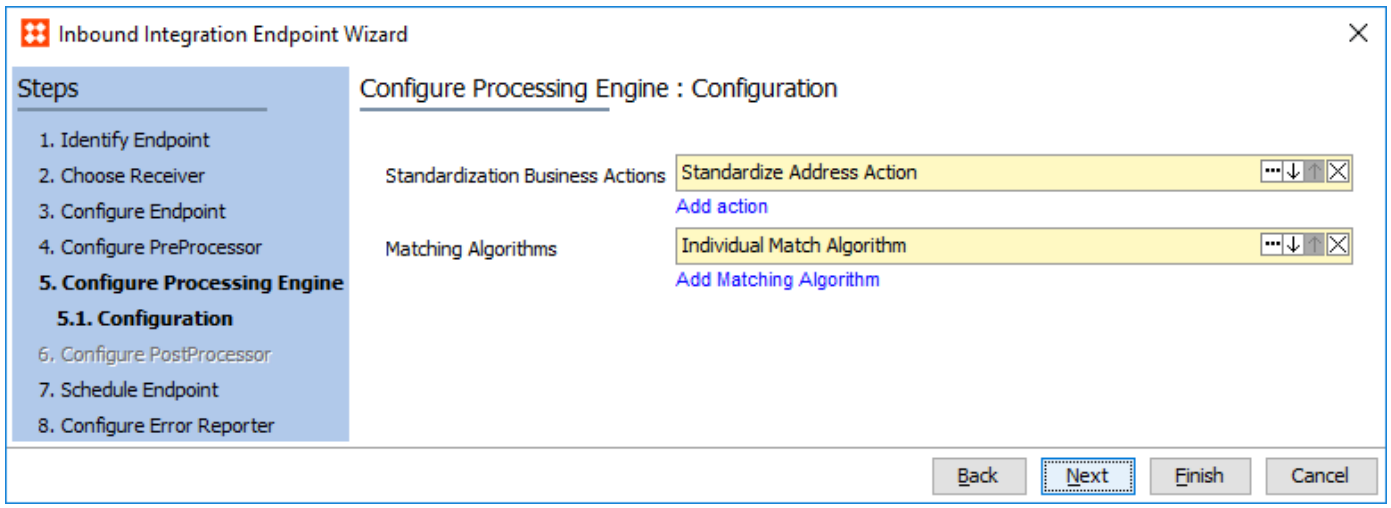
For more information on Excel / CSV imports, see the **Entity Reference via Source Record ID - Map Inbound** section of the **Data Exchange** documentation.

For more information on XML imports, see the **Source Record ID in STEPXML** section below.

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.
   - 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

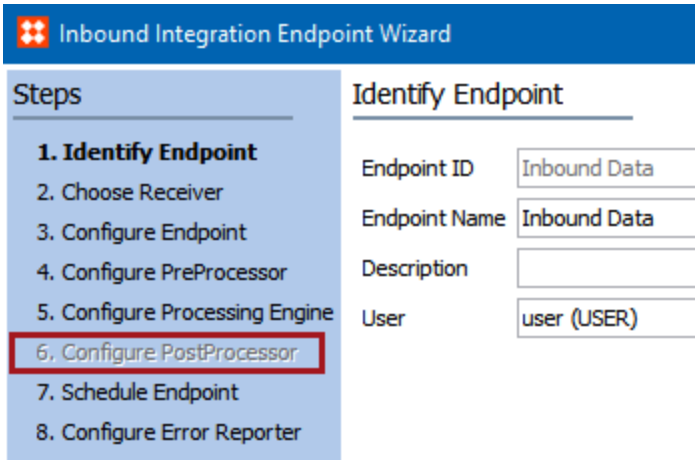
The Source Record ID of a given record is stored in the Source Record ID Attribute of that record. This attribute is a metadata attribute placed on the record's Source Relation Reference, which is a reference between the source record and the Source System in question.

See below for 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>
        </EntityCrossReference>
        <EntityCrossReference Type="CustomerToOrganization">
          <Entity>
            <EntityCrossReference EntityID="SAP_Test" Type="MergeSourceRelation">
              <MetaData>
```

```
                <Value AttributeID="SourceRecordID">SSB_00002</Value>
              </MetaData>
            </EntityCrossReference>
        </Entity>
    </EntityCrossReference>
</Entity>
```

IIEP - Configure PostProcessor

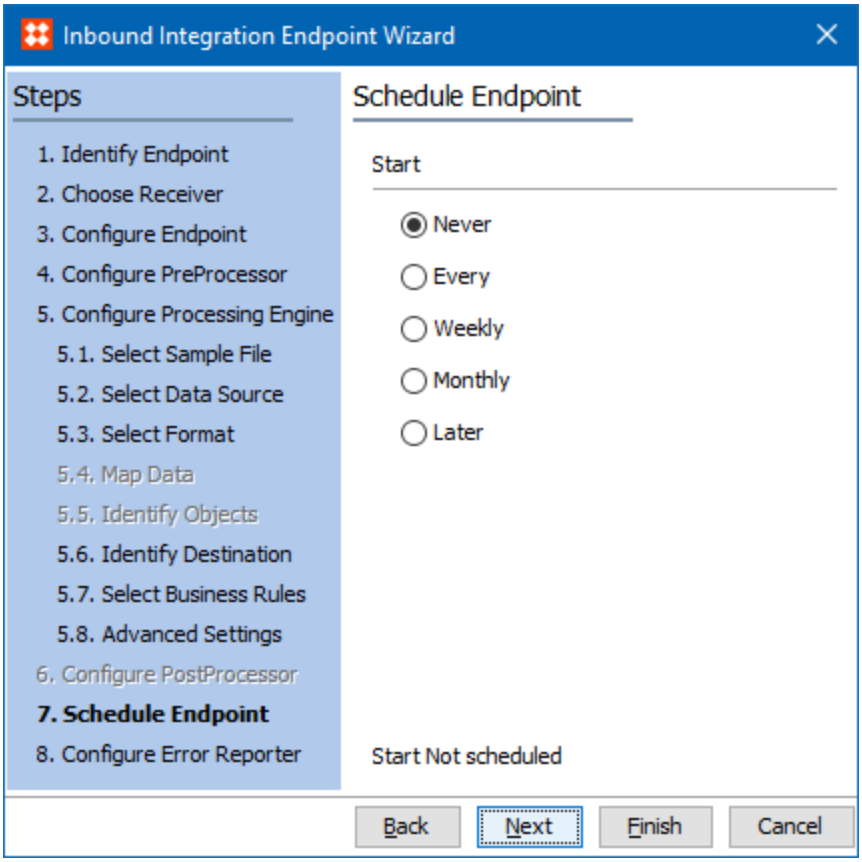


By default, STEP does not include any pre-configured post-processor and the wizard step is disabled. When created, a customized post-processor has access to import events (information about what has changed) and, based on the events, can trigger any required system change.

Typically, post-processor logic is implemented via business rules referenced from the processing engine configuration. For assistance in creating a post-processor, contact Stibo Systems Support.

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:

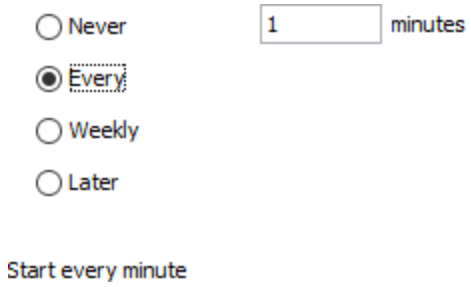
- IIEPs that do not use the REST Receiver method should typically be scheduled to poll the data source for new files / messages with regular intervals.
- IIEPs that do use the REST Receiver method can be invoked via a REST POST request. With this functionality, for example, a system delivering data to STEP could potentially invoke the IIEP after having uploaded a file to a Hotfolder. For more information, see 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.

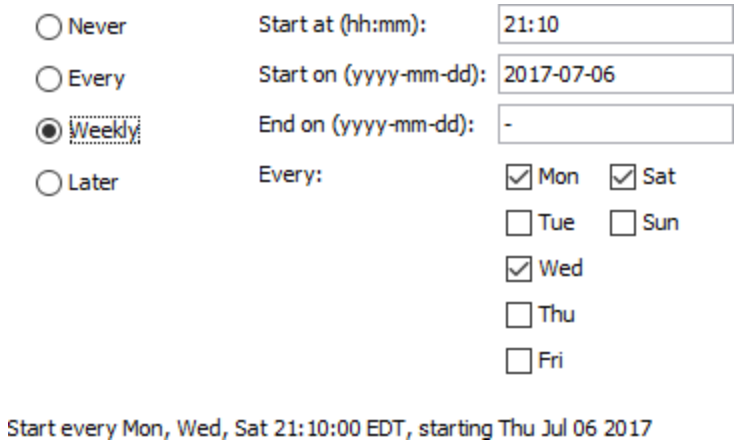
---

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.

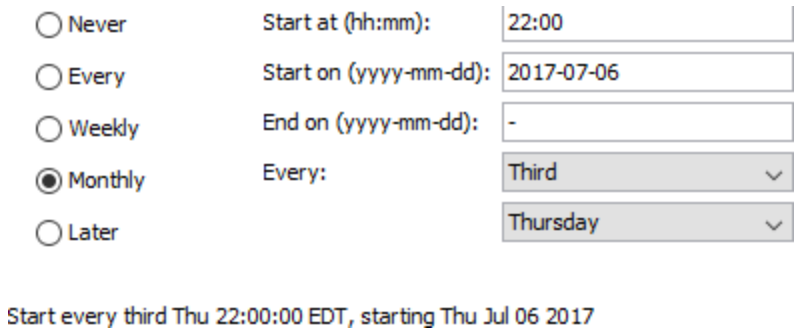
- **Every** - automatically run the endpoint repeatedly, every selected number of minutes. One (1) minute is the shortest interval allowed and is closest to real time. Enter the number of minutes in the text box. The selection is summarized at the bottom of the dialog.



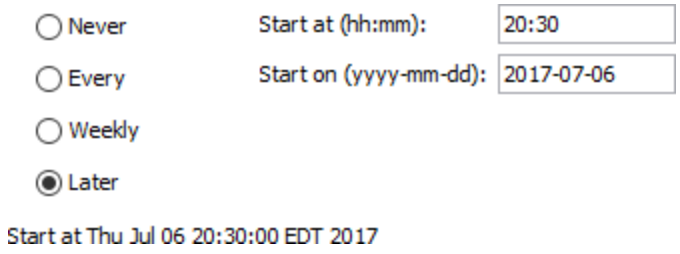
- **Weekly** - automatically run the endpoint repeatedly, based on the selected time, start and end dates, and days of the week. Use this option if a daily schedule is needed. The 'Start at' parameter determines the time of day that the endpoint will run. The 'Start on' parameter determines the date 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.

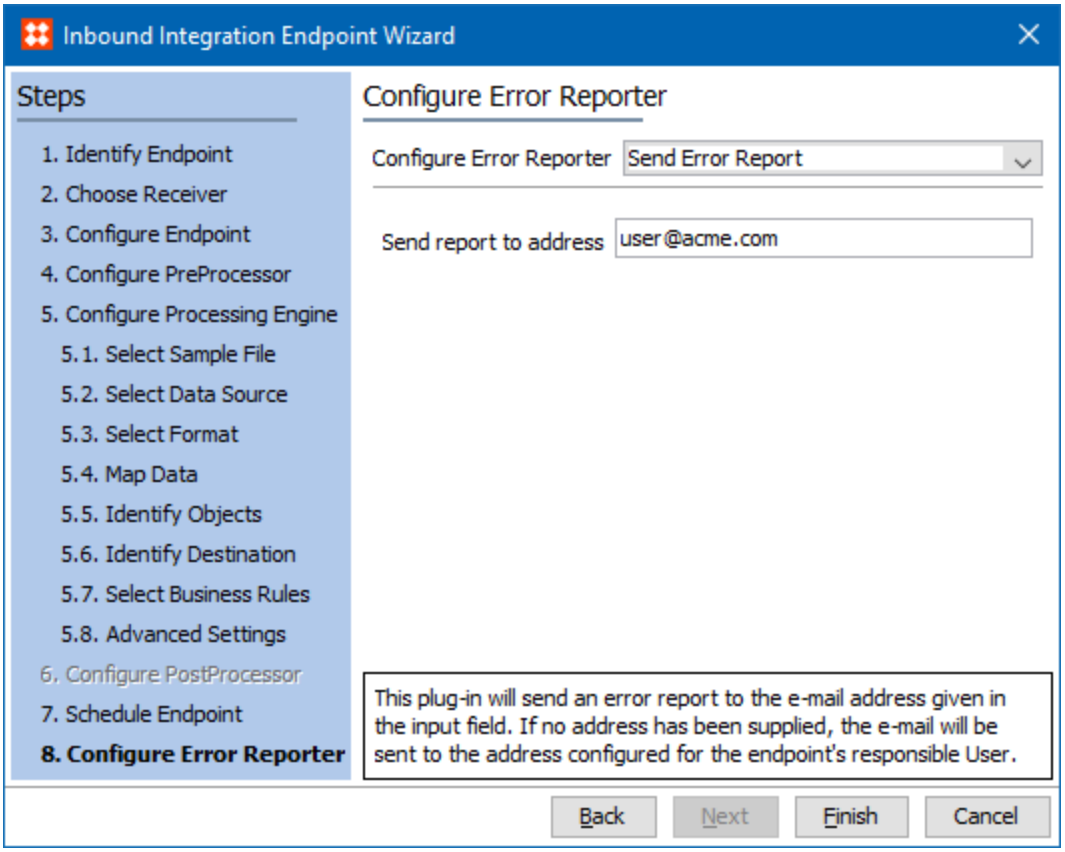


- **Later** - automatically run the endpoint only once, at the time and date specified. The selections are summarized at the bottom of the dialog.



2. Click the **Next** button to display **IIEP - Configure Error Reporter**.

IIEP - Configure Error Reporter



For additional information on IIEP errors, see the **Resolving a Failed Background Process** section of the **Handling Failed IIEP Background Processes** topic.

1. On **Configure Error Reporter**, select an option that is activated if an endpoint-related background process fails. The following options are standard with STEP:
   - **Send Error Report** emails information about the failed endpoint, the background process, the failed file, the failing process step, the cause of the error, and a copy of the file that triggered the error. Leaving the 'Send report to address' parameter blank means the error report is 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 is written to the failed Background Process Execution Log.
   - **Not Defined**, the error reporter is disabled, no notification by email is sent.
2. In **Send report to address**, enter the email address of recipient of the error report. Use a semicolon (;) to separate multiple email addresses.

   For information on configuring email from STEP, see the **Email from STEP** topic in the **Resource Materials** of online help.

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, see the **Running an Inbound Integration Endpoint** documentation.

Running an Inbound Integration Endpoint

After configuring an inbound endpoint, it must be enabled before it can run on schedule or be invoked manually.

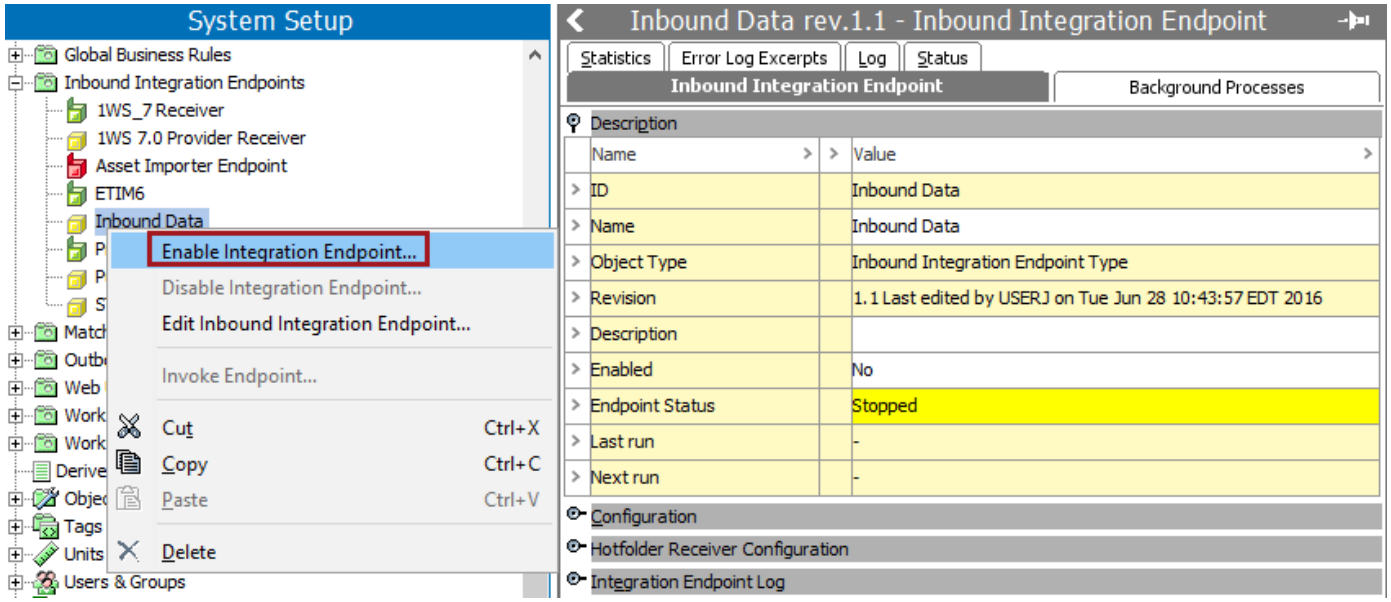
The following icons are used to indicate the status of the inbound integration endpoint. The endpoint status is displayed in the System Setup hierarchy and on the endpoint editor in the Description flipper.

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	An error occurred during processing that caused the integration endpoint background process to fail. No data will be imported until the endpoint has been manually resumed / reactivated. For more information, see **Handling Failed IIEP Background Processes** documentation.

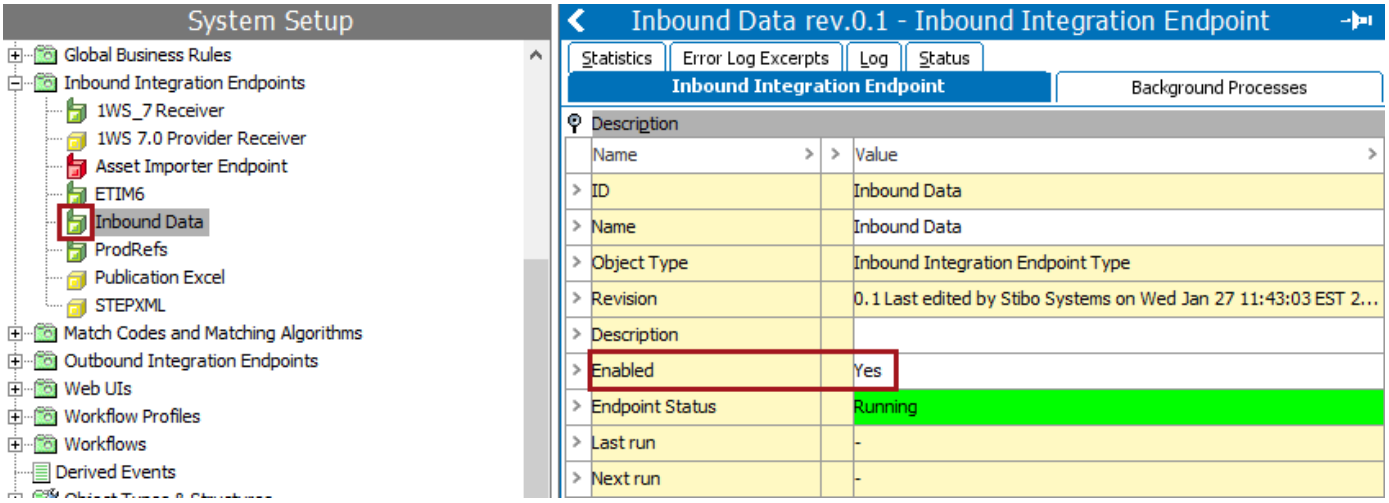
Enable and Invoke an Inbound Integration Endpoint

Once configured, use the following steps to run the endpoint.

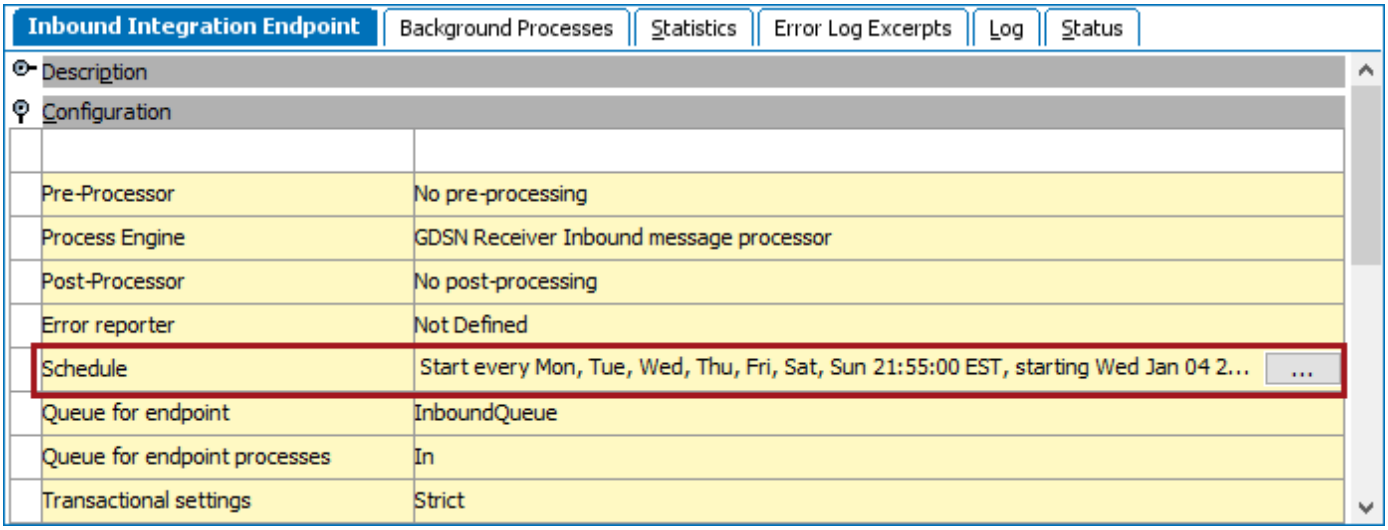
1. On System Setup expand the Inbound Integrations Endpoints node to display all existing inbound endpoints.
2. Select a disabled inbound endpoint or a failed endpoint, right-click, and select **Enable Integration Endpoint** from the menu.



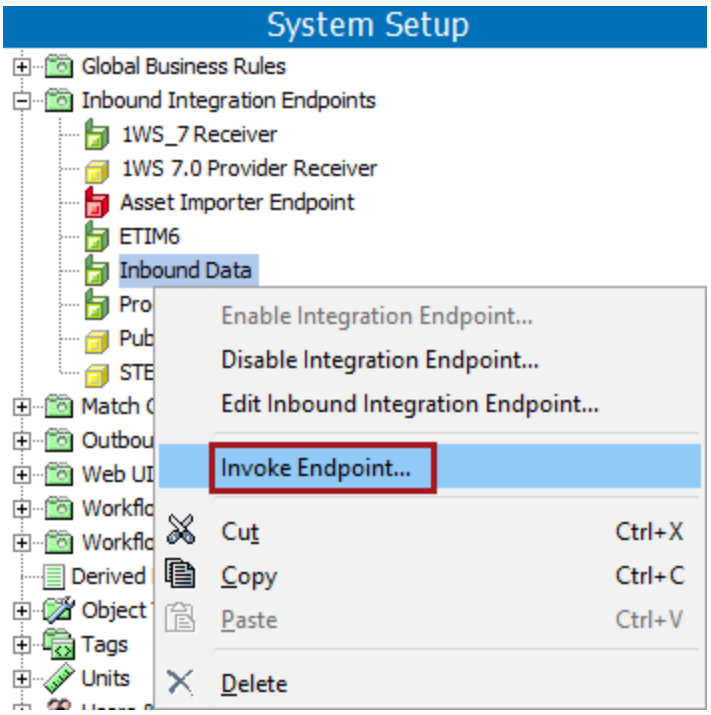
The tree navigator icon now shows enabled. On the Inbound Integration Endpoint tab, the Description flipper includes the 'Enabled' parameter, which now displays 'Yes.'



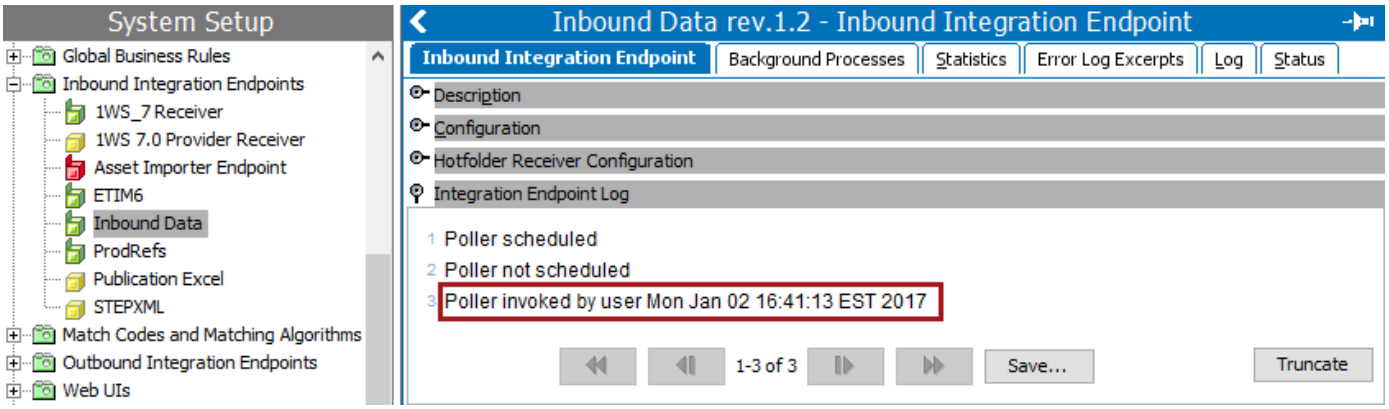
3. Open the Configuration flipper to view the Schedule parameter.
   - If scheduled, the Schedule parameter shows when the IIEP will be invoked automatically.



   - 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**.

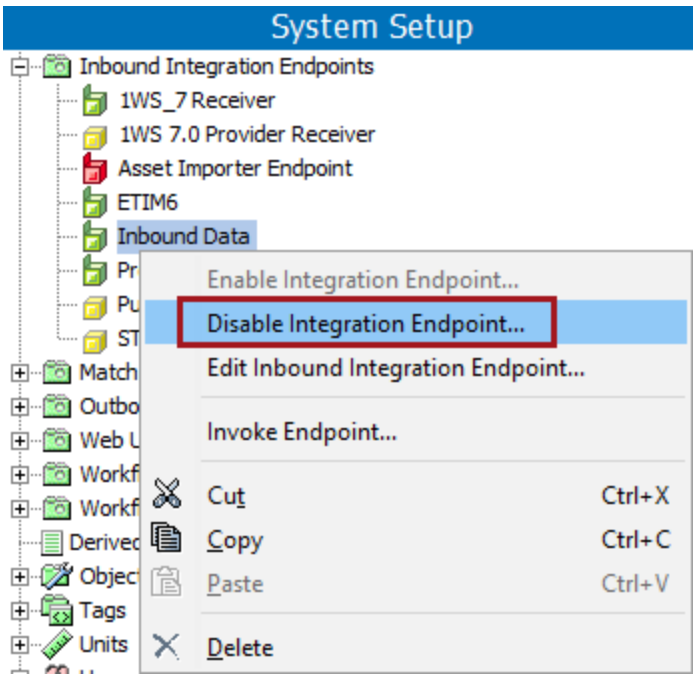


4. When invoked on schedule and manually, files or messages available in the location identified by the IIEP receiver configuration are processed. The Integration Endpoint Log flipper displays the time the endpoint was invoked. For scheduled IIEPs, the log includes a link to the background process that was started. For more information, see **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 ( ).

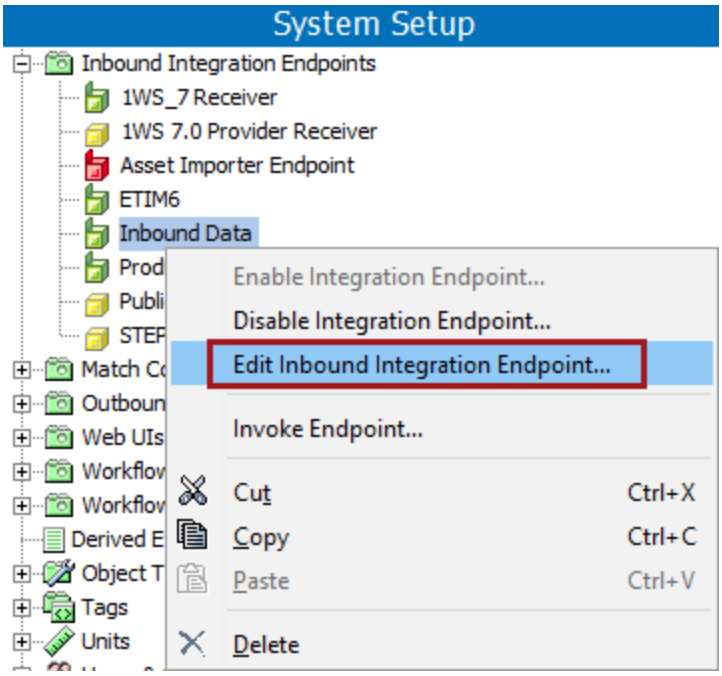


Once disabled, the inbound integration endpoint can be restarted using the Enable Integration Endpoint step defined above.

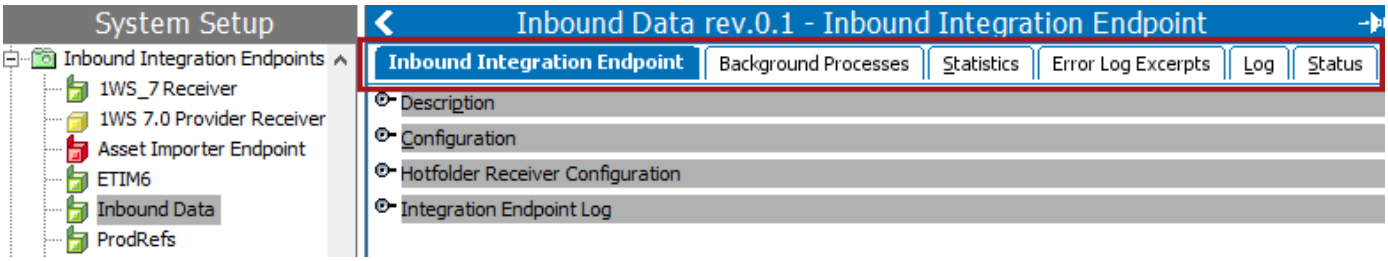
Additional information about restarting a failed endpoint, indicated by the failed icon ( ) can be found in **Handling Failed IIEP Background Processes** documentation.

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 Endpoint** option on the IIEP right-click menu.



Each tab on the editor is defined below and includes parameters to maintain the IIEP.



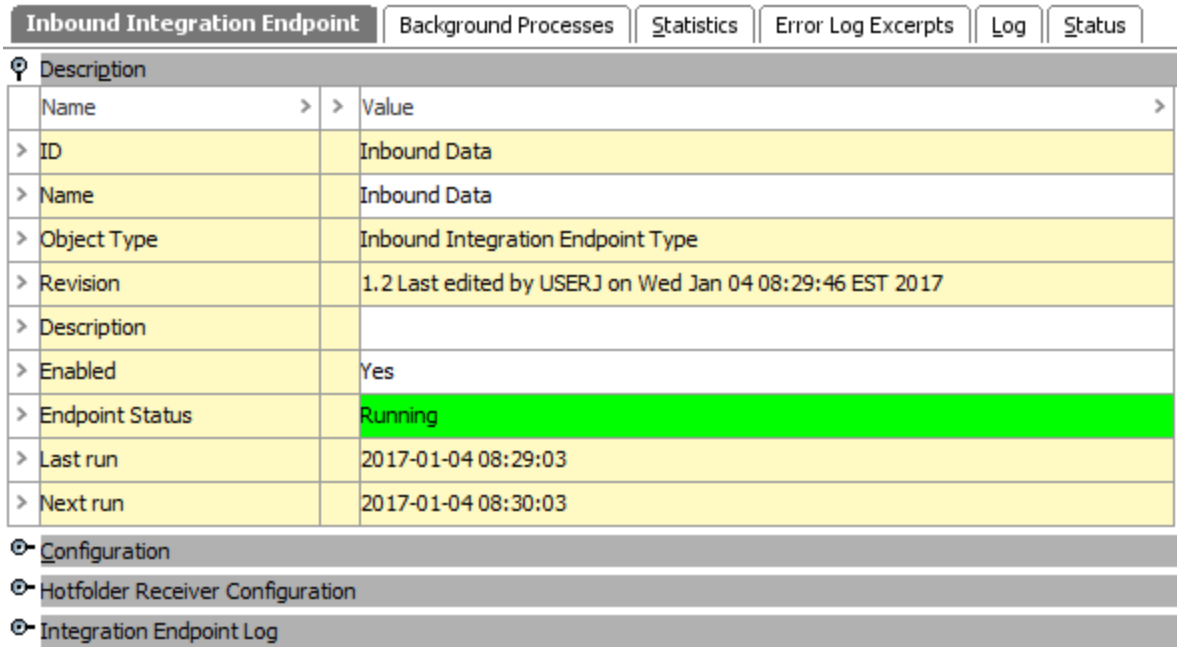
Inbound Integration Endpoint Tab

This tab holds basic information within the following flippers:

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, see **IIEP - Identify Endpoint** documentation.

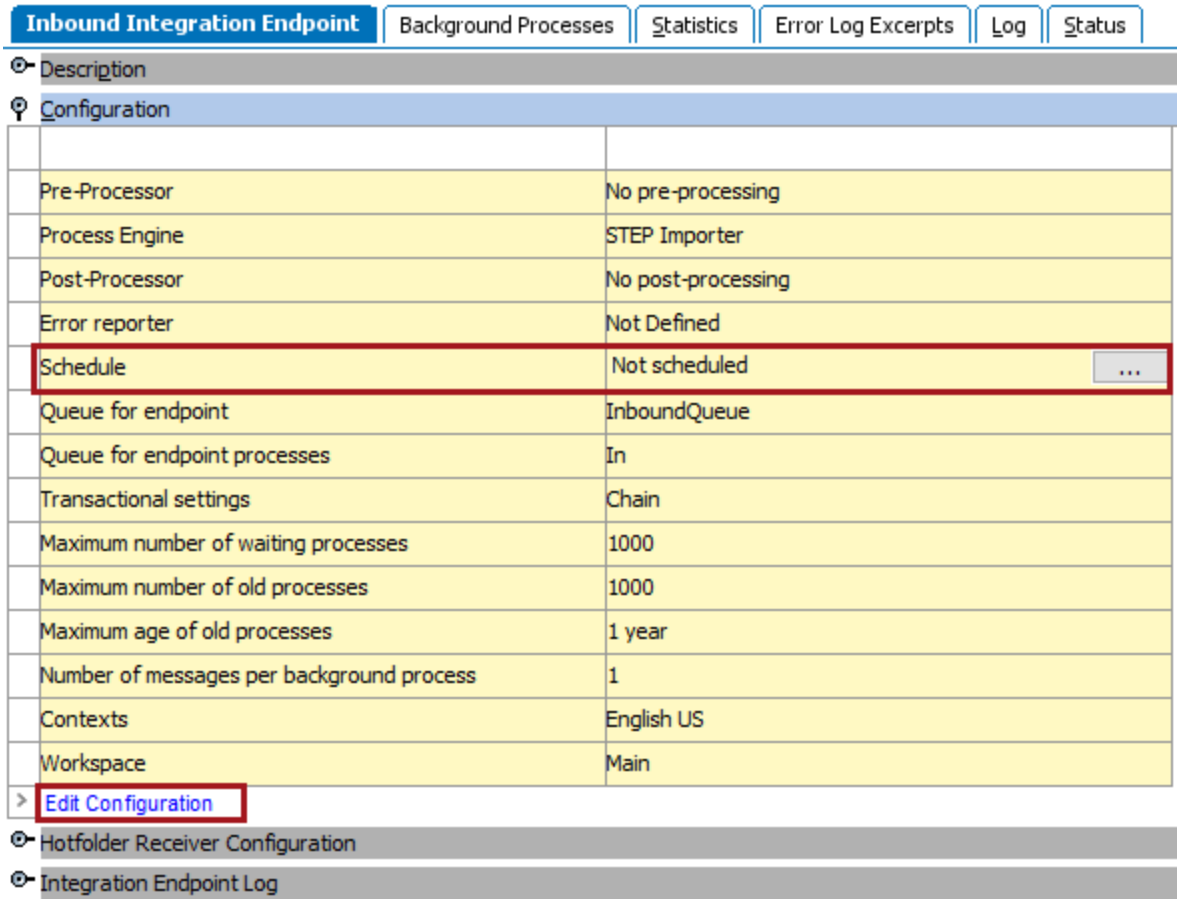
For information on the Endpoint Status, see **Running an Inbound Integration Endpoint** documentation.



Configuration Flipper

The Configuration flipper displays many of the settings determined through the wizard.

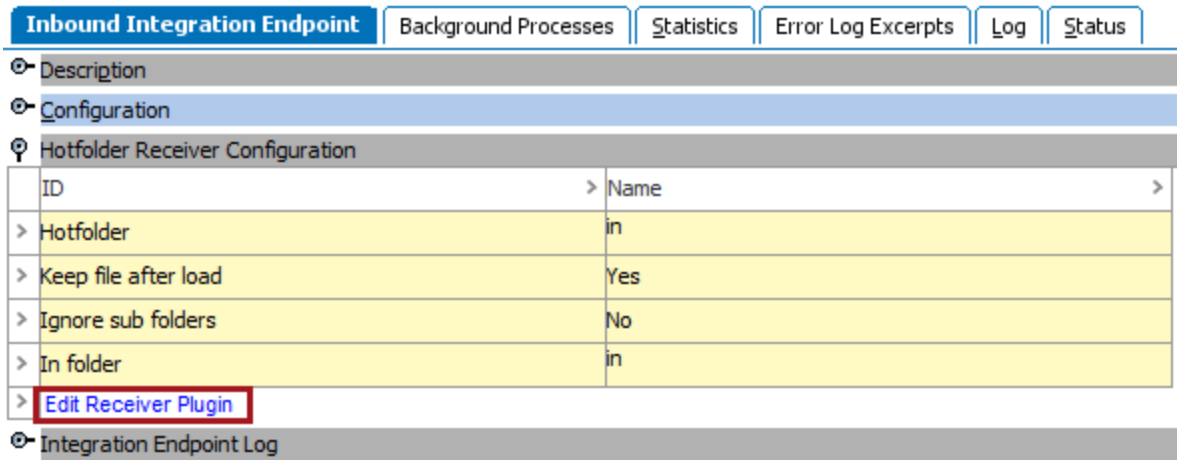
- The **Edit Configuration** link opens the wizard and allows you to modify the following parameters:
- For information on the Pre-Processor, see **IIEP - Configure PreProcessor**.
- For information on the Process Engine, see **IIEP - Configure Processing Engine**.
- For information on the Post-Processor, see **IIEP - Configure PostProcessor**.
- For information on the Error reporter, see **IIEP - Configure Error Reporter**.
- Changes to the schedule can be made easily using the Schedule parameter ellipsis button (...). This opens the wizard on the Schedule Endpoint step, see **IIEP - Schedule Endpoint**.
- For information on the Queue for endpoint, Queue for endpoint process, 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, see **IIEP - Configure Endpoint**.



Receiver Configuration Flipper

The Receiver Configuration flipper displays details about the receiver method selected for the IIEP. For more information about receiver methods, see **IIEP Receiver Methods**.

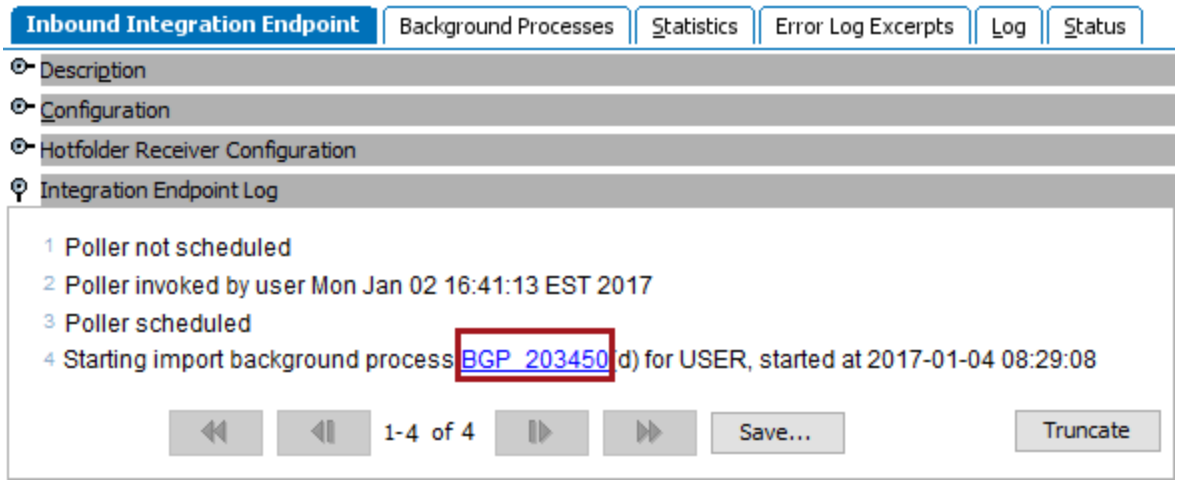
- The **Edit Receiver Plugin** link opens the wizard on the Choose Receiver step and allows you to modify the settings. For more information, see **IIEP - Choose Receiver**.



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.

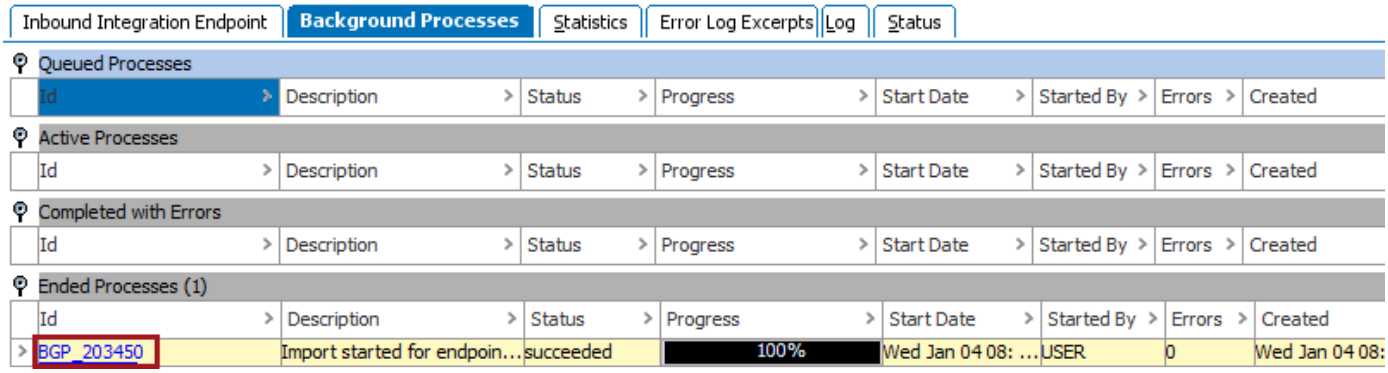


Background Processes Tab

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 Importer Background Processes.

- **Queued Processes** are background processes that are started in the state 'waiting.' The maximum number of background processes that can be started in the state waiting is specified in the integration endpoint configuration.
- **Active Processes** are background processes that are currently being processed. If data is processed as strict or chained, the endpoint only has one active background process at a time. Otherwise, it is possible to apply more background process queues to work concurrently. In this case the endpoint can have more active background processes at a time.
- **Failed Processes** are background processes that are no longer running. For more information, see **Handling Failed IIEP Background Processes**.
- **Completed with errors** have run, but an error has been detected.
- **Ended Processes** have completed successfully. The number of ended processes is determined by the configuration.

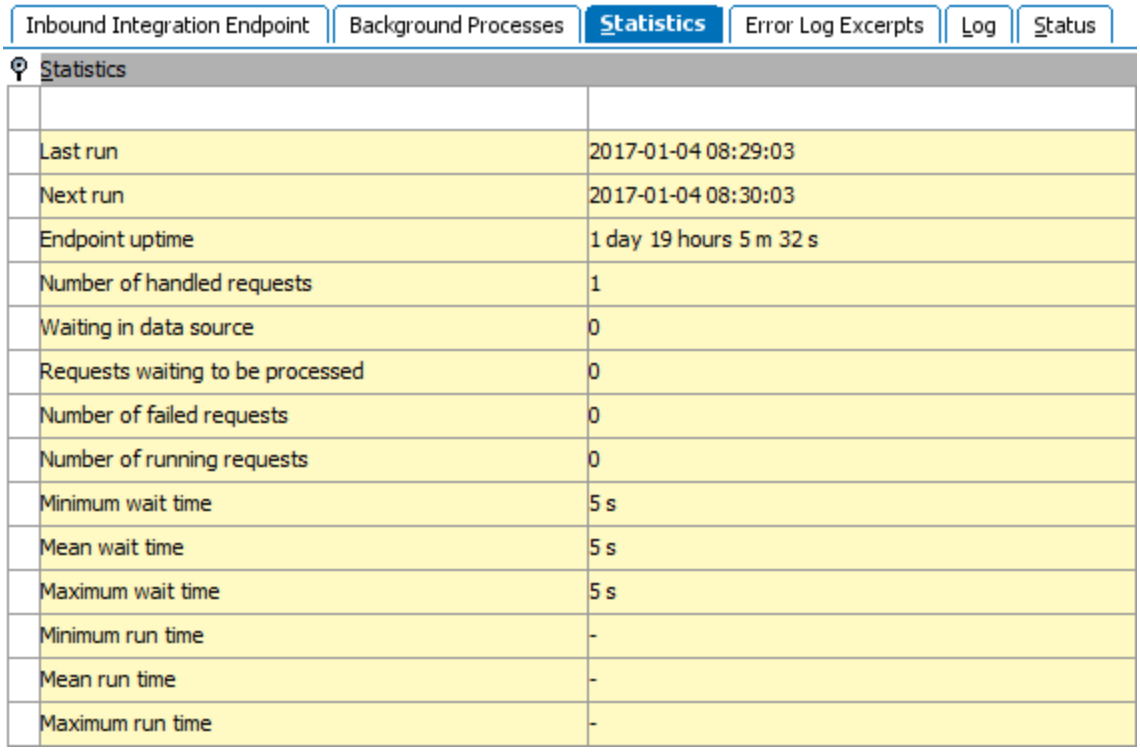
For more information, see the **Monitoring an IIEP via Background Process** section.



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.

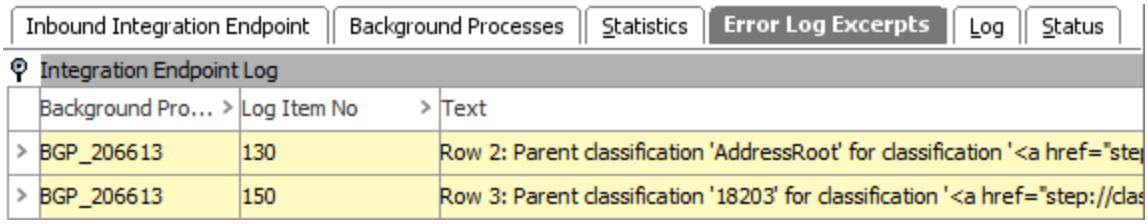
Note: Receivers can poll the data source to check for waiting files or messages more frequently than the IIEPs are invoked.



Error Log Excerpts Tab

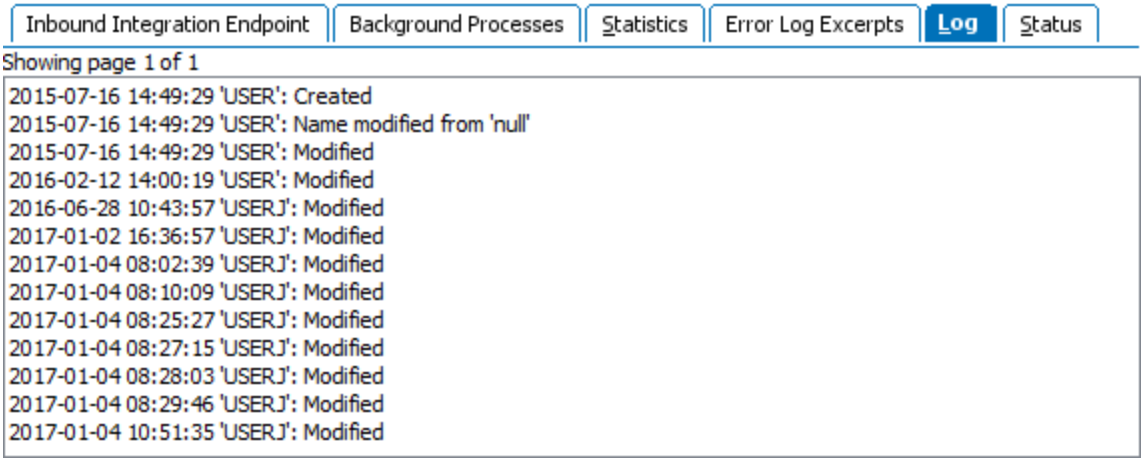
This tab shows data from the main Java log file related to failed background processes with Log Level > 'Info.' Click a hyperlink to a failed background process and correct the cause of the failed background process. For more

information, see **Handling Failed IIEP Background Processes**.



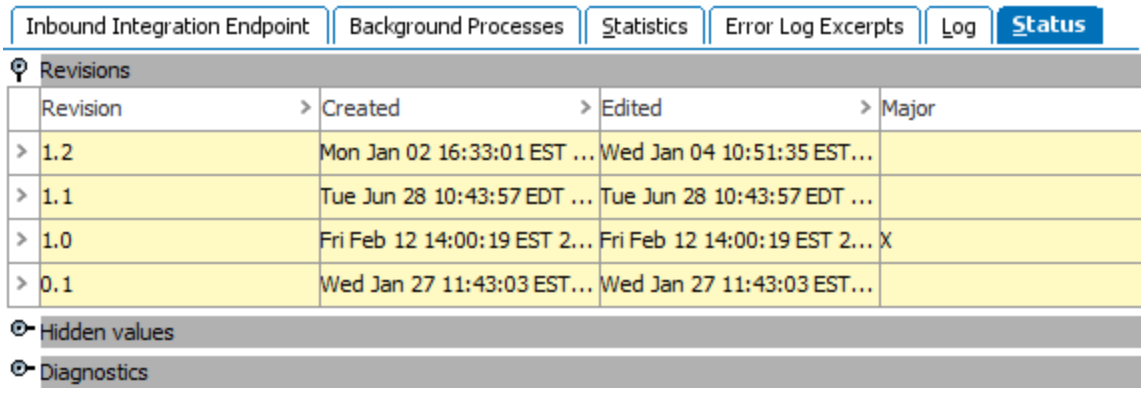
Log Tab

This tab provides information about changes to the IIEP configuration.



Status Tab

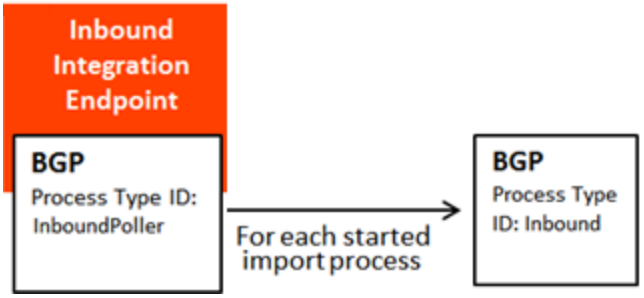
This tab provides information about revisions, hidden values and diagnostics.



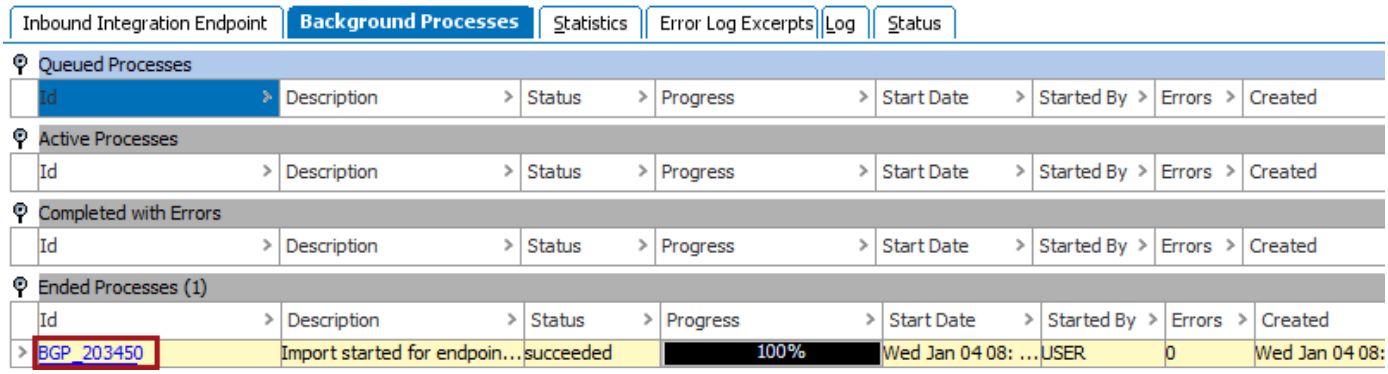
Monitoring an IIEP via Background Process

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, see **Maintaining an Inbound Integration Endpoint**.

When configuring an inbound integration endpoint that uses the STEP Importer 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.

Selecting the 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, see the **IIEP - Configure Endpoint** documentation.

Background Process Queue Size Property

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 different 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
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.
Product x contains more than one reference to asset y of Type z, reference data will not be	You cannot make a product-to-asset reference from the same source product to the same target asset more than

Error Message	Description
imported.	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 "see" 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.
No ContextID specified in STEPXML to import	Importing STEPXML without a Context ID.

Error Message	Description
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:

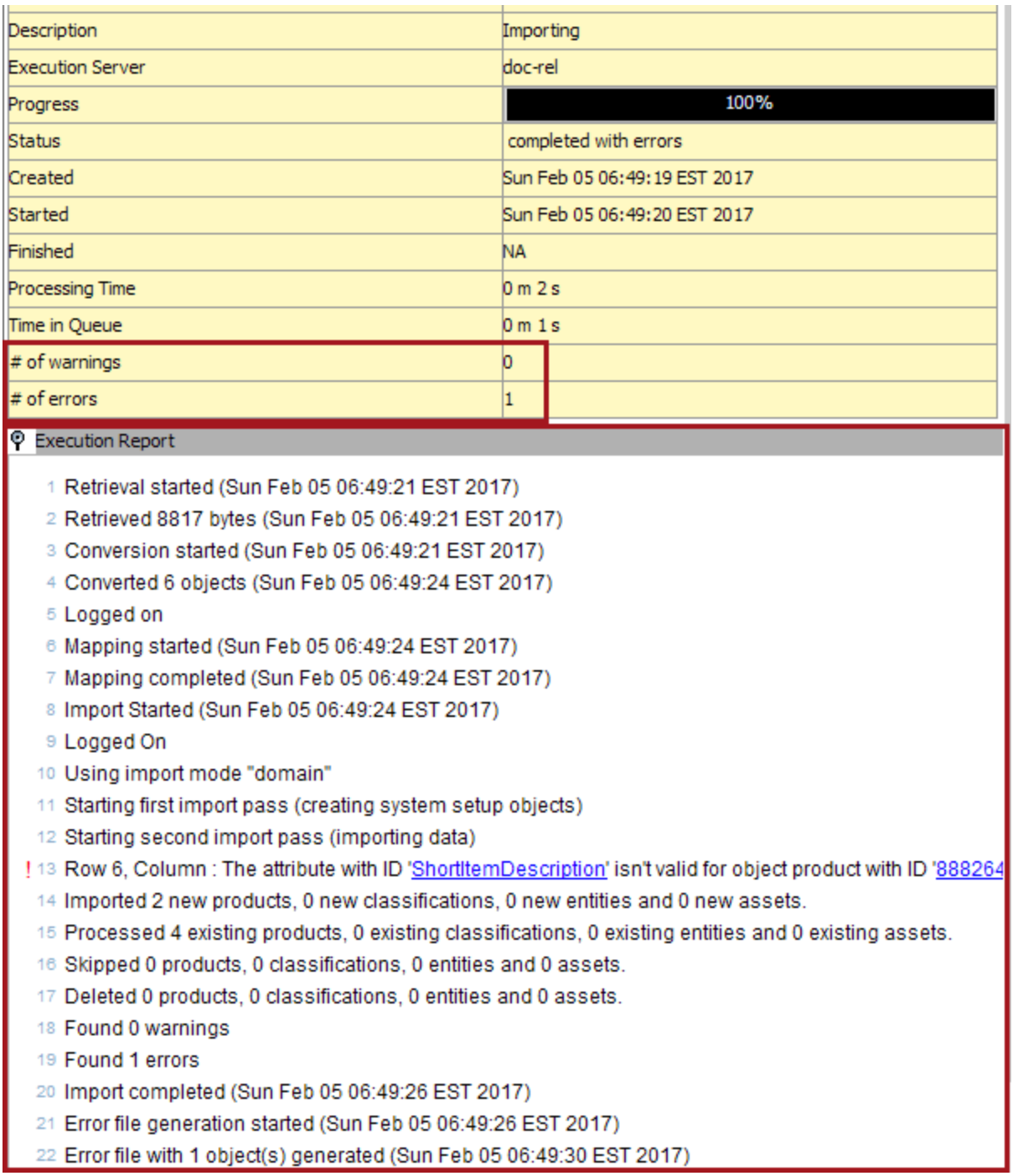
- Caught FatalDataStoreException at Mon Jun 06 13:10:16 CEST 2016: The transaction has been rolled back. See the nested exceptions for details on the errors that occurred., caused by SQLException: ORA-00001: unique constraint (STEPSYS.NODE_NODEID_UIX) violated {prepstmnt 523703057 INSERT INTO NODE

(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:



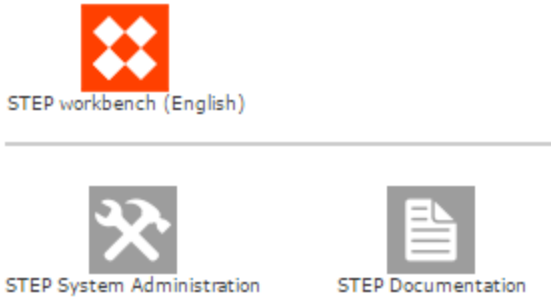
Description	Importing
Execution Server	doc-rel
Progress	100%
Status	completed with errors
Created	Sun Feb 05 06:49:19 EST 2017
Started	Sun Feb 05 06:49:20 EST 2017
Finished	NA
Processing Time	0 m 2 s
Time in Queue	0 m 1 s
# of warnings	0
# of errors	1

Execution Report

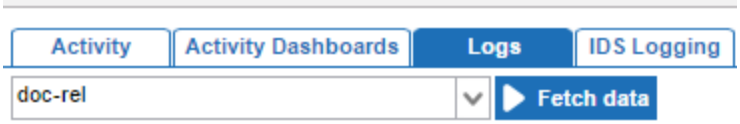
1 Retrieval started (Sun Feb 05 06:49:21 EST 2017)
2 Retrieved 8817 bytes (Sun Feb 05 06:49:21 EST 2017)
3 Conversion started (Sun Feb 05 06:49:21 EST 2017)
4 Converted 6 objects (Sun Feb 05 06:49:24 EST 2017)
5 Logged on
6 Mapping started (Sun Feb 05 06:49:24 EST 2017)
7 Mapping completed (Sun Feb 05 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:

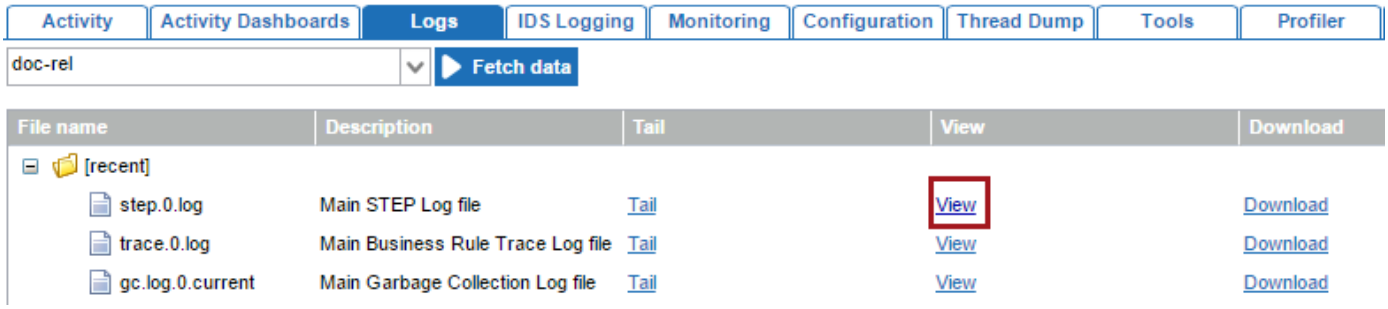
1. Launch the STEP Start Page
2. Click on the System Administration link



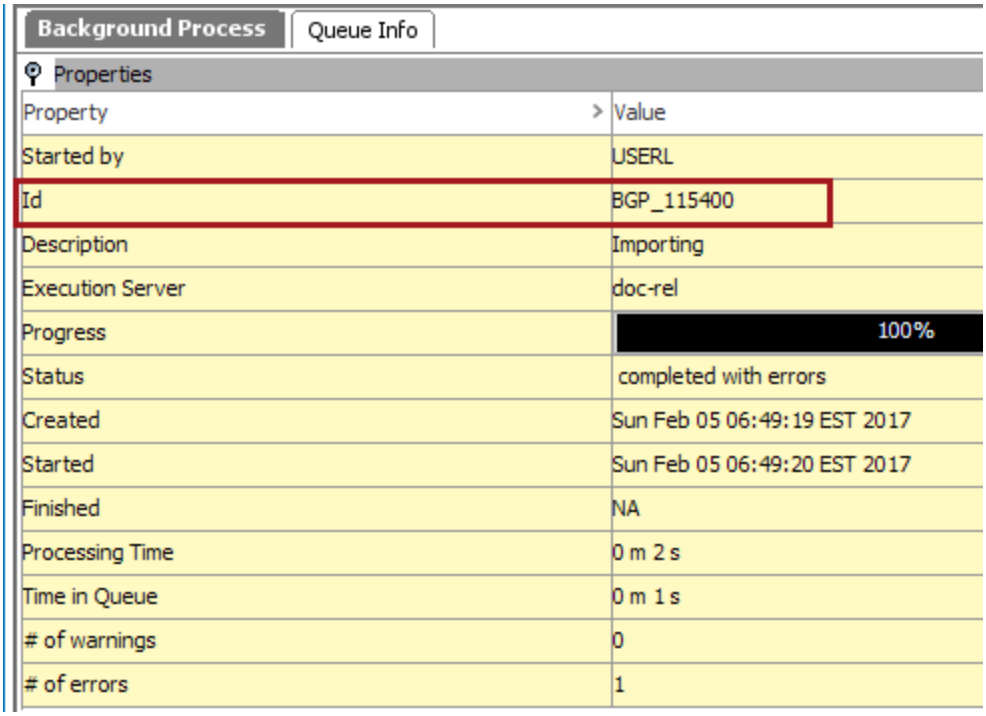
3. Once logged in, click on the Logs Tab
4. Select the Server in the dropdown list and click on 'Fetch Data'



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



Background Process | Queue Info

Properties

Property	Value
Started by	USERL
Id	BGP_115400
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

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 in **Exporting Inbound Integration Endpoint Definitions as Comments**.
- 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**.

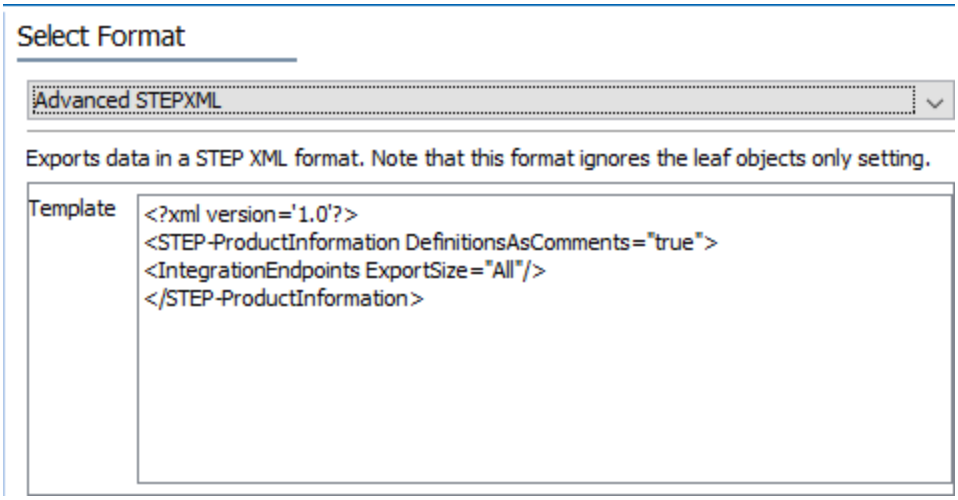
Exporting Inbound Integration Endpoint Definitions as Comments

Integration endpoint definitions can be exported as comments using Advanced STEPXML. These exports are intended to be used for submission to external source control systems for comparison purposes. Users can import them into source code repository systems where they can be compared from version to version. Editing and/or importing of these files is not supported (e.g., users may not export, edit the comments, and re-import in STEP).

To export integration endpoint definitions (both inbound and outbound) for external comparison, use Advanced STEPXML and set the DefinitionsAsComments tag to 'true', as shown below. This template exports event queues and configurations, outbound and inbound integrations and configurations.

On the Select Format step of the outbound tool (either Export Manager or OIEP), choose the Advanced STEPXML format, then copy and paste the following text into the Template field:

```
<?xml version='1.0'?>
<STEP-ProductInformation DefinitionsAsComments="true">
<IntegrationEndpoints ExportSize="All"/>
</STEP-ProductInformation>
```



An example of the output for an inbound integration endpoint definition as comments is shown below:

```
<InBoundIntegrationEndpoint...>
<!--Definition: [This will be the IEP definition. Removed for brevity.-->
[Remaining configuration is included]
</InBoundIntegrationEndpoint>
```

Note: The content of the comment field is not part of the STEPXML XSD and therefore Stibo Systems reserves the right to change the format of the output content at any time.

For more information, see the **STEP-ProductInformation Tag in STEPXML** section of the **Data Exchange** documentation.

Additionally, details about integration endpoint configuration can be exported using STEPXML. For information, see the Configuration > **Include Integration Endpoints** section of the **STEPXML Outbound Parameters** documentation.

Handling Failed IIEP Background Processes

You can reactivate a failed integration endpoint using Disable / Enable or Resume.

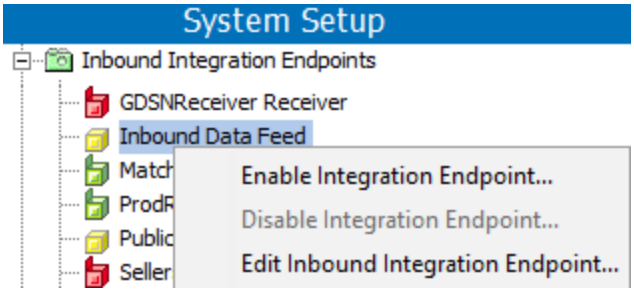
- If a background process fails and the processing of data is set to **strict**, you have to restart the endpoint before the endpoint is able to start another background process.
- If a background process fails and the processing of data is set to **chain**, the remaining background processes that belongs to the chain will also fail. You have to 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, see **Integration Endpoint Transactional Settings** documentation.

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**.



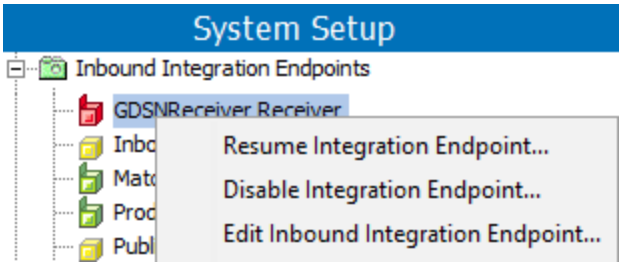
Resume a Background Process

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 IIEP, as described in the **IIEP - Configure Error Reporter** topic.

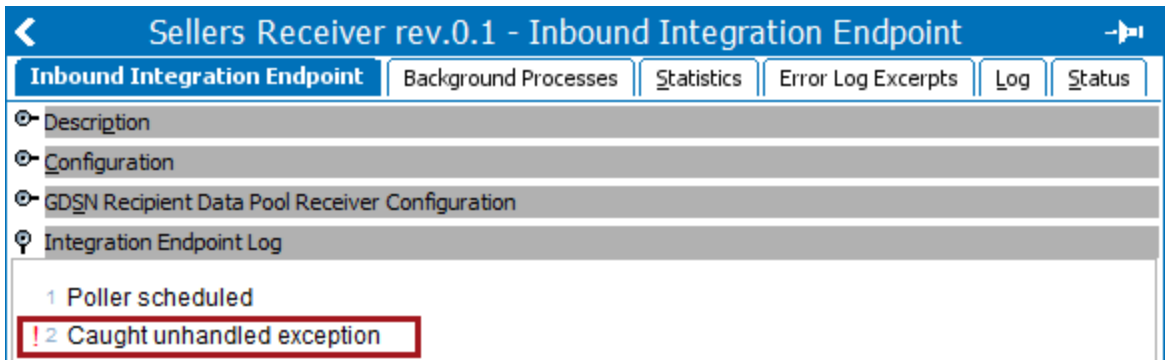
The two levels of failure reported in STEP are errors and warnings.

- 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.



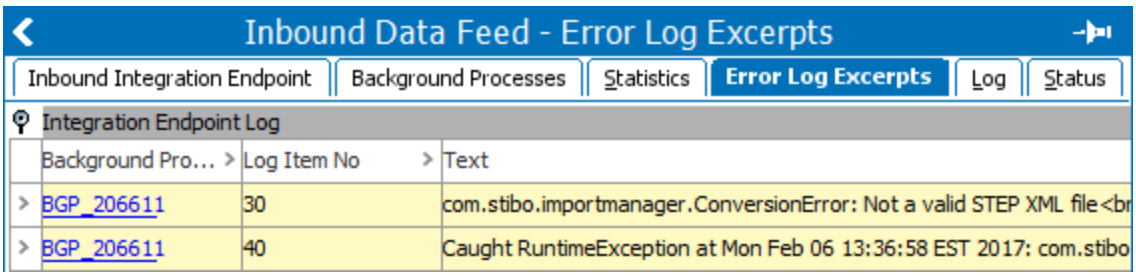
IIEP Fails After the BGP Starts

The Error Log Excerpts tab includes details about failures within the background process.

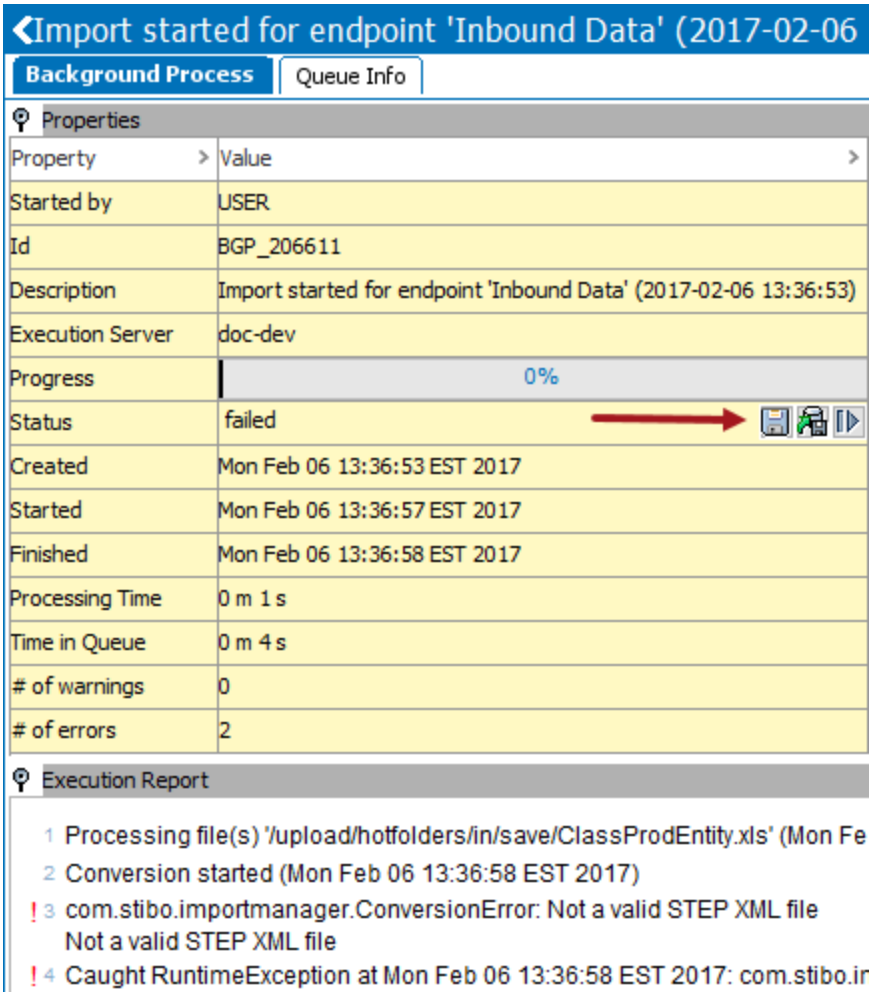
1. In **System Setup** select the relevant integration endpoint, and then click the **Error Log Excerpts** tab to display a list of failed background processes and the description of the failure.

   You can also see 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.

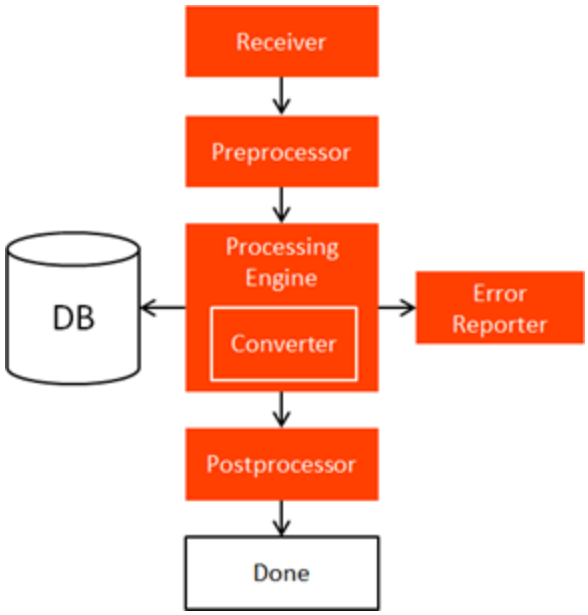


3. In the Background Process editor **Execution Report**, view the progress of the background process, and where it failed.
4. In the Properties area > **Value** column, click the Save button ( ) near the failed status, and save the file locally.

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 > Properties flipper > Status parameter Value column, click the Upload button ( ) near the failed status.
7. Restart the process in the Properties area > 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 **IIEP Receiver Methods** section.

Preprocessor

A preprocessor (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 preprocessors. For more information, see the **IIEP - Configure PreProcessor** 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, see the **IIEP - Configure Processing Engine** documentation.

Error Reporter

STEP includes a single Send Error Report option to report errors. An email can be sent to a specified address if errors occur when files or messages are handled by the processing engine. For more information, see the **IIEP - Configure Error Reporter** documentation.

Postprocessor

A customized postprocessor 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, see the **Events** documentation in the **System Setup**.

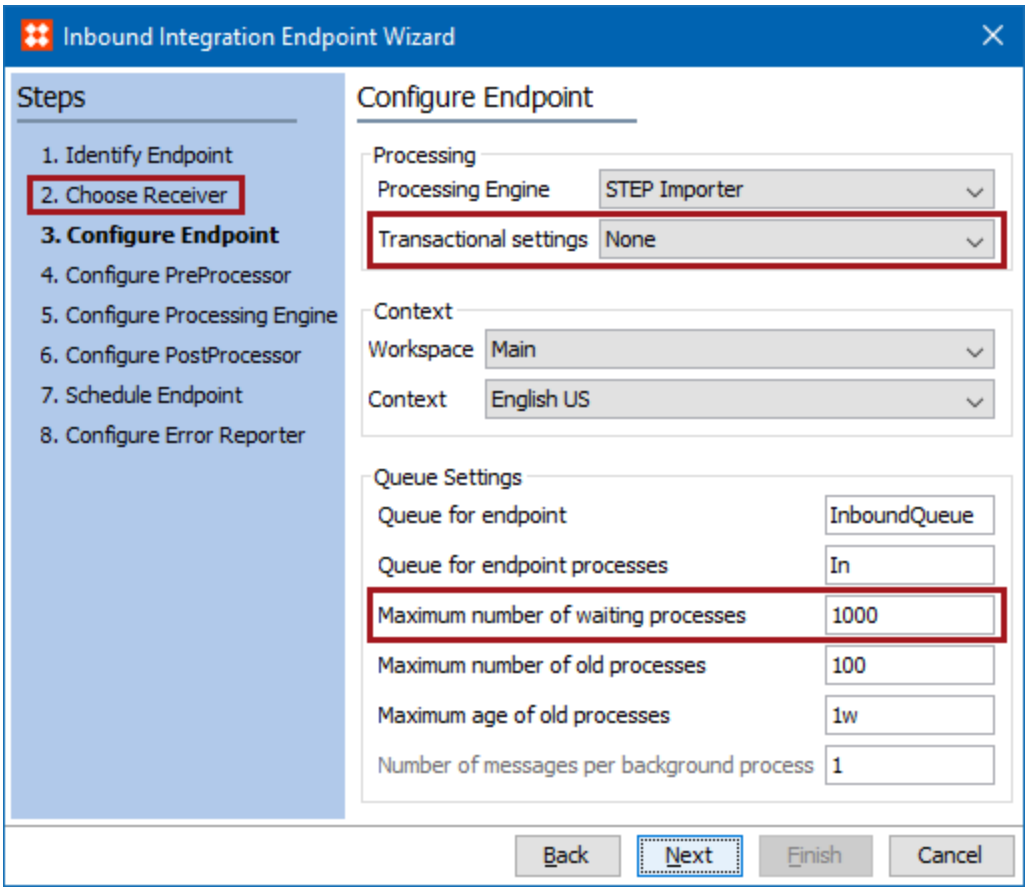
By default, STEP does not include any pre-configured post-processor, since postprocessor logic is also typically implemented via business rules referenced from the processing engine configuration.

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.

Inbound Integration Endpoints

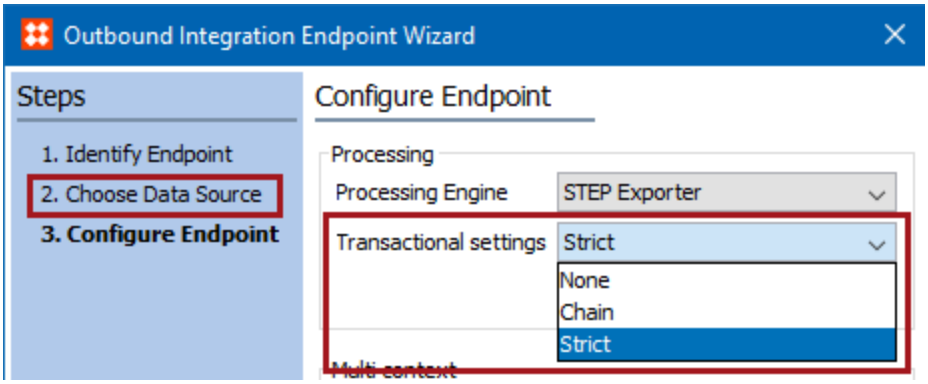
In the IIEP wizard, background process and error handling is managed by three (3) elements: the selected Receiver (on the Choose Receiver step), the 'Transactional settings' parameter, and the 'Maximum number of waiting processes' parameter (both on the Configure Endpoint step). These elements also work together to determine what additional options are available in the wizard, as defined below.



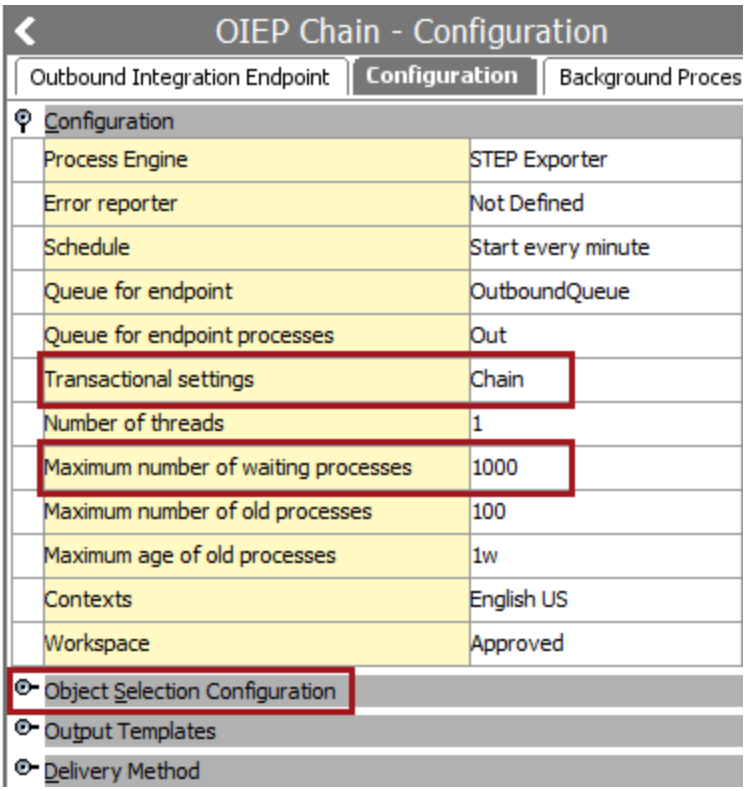
For more information, see the **IIEP - Configure Endpoint** documentation.

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, see the **OIEP - Select Objects - Configure Endpoint** topic.

For details on the 'Transactional settings' and 'Maximum number of waiting processes' parameters, see 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.

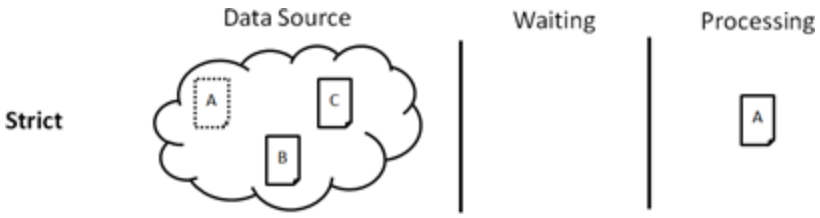
For more information on event-based OIEPs, see the **OIEP - Event-Based - Configure Endpoint** topic.

Transactional Settings Options

When available, there are three possible transactional settings: 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 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 common setup, unless there is a reason to use Chain or None (for assets).



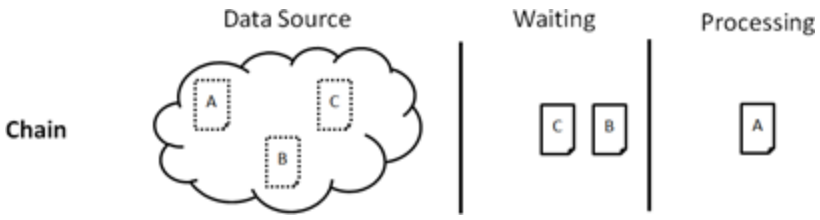
The queue size of a background process with strict transactional settings must be one (1). A strict transactional setting does not allow multiple background processes to work in parallel / concurrently regardless of the queue size. By definition, no new background processes can be generated while one 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 'chain' transactional setting processes data in batches of chained background processes. 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. Chain is allowed for all receiver methods.

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.

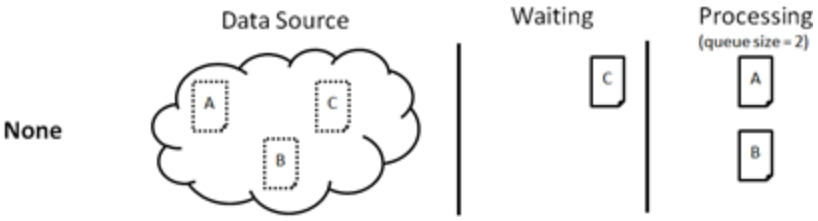


The queue size of a background process with chain transactional settings must be one (1). A chain transactional setting does not allow multiple background process to work in parallel / concurrently regardless of the queue size. By definition, no new background processes can be generated while one is active. The default 'Maximum number of waiting processes' is 1000.

Chain is allowed for all inbound receiver methods and for outbound select data endpoints.

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.

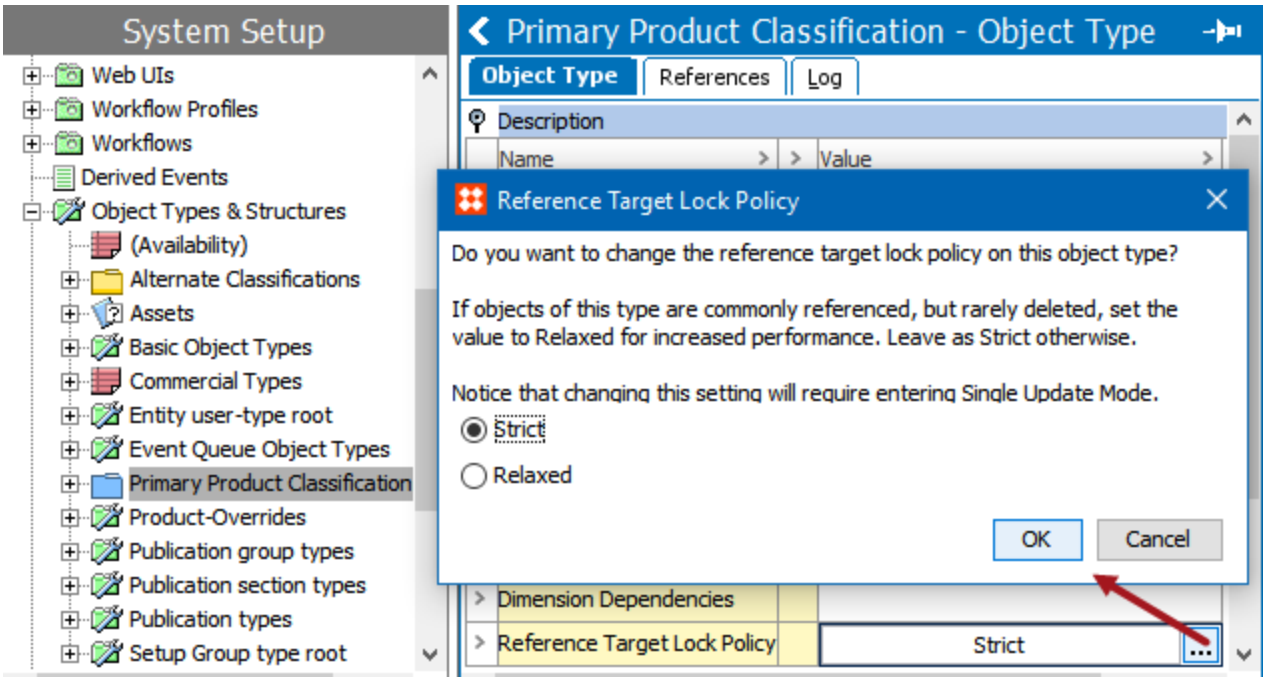
Reference Target Lock Policy on Object Types

The 'Reference Target Lock Policy' is used for the Assets, Entities, Classifications, and Products object types to manage how objects should be locked while they are being referenced. When a reference between two objects is being created in STEP, the target object is locked to ensure that it is not being deleted while the reference is being created, meaning only one process or user can edit it.

When a reference between two objects is being created in STEP, the target object will be 'locked' to ensure that it is not being deleted while the reference is being created. In most of these cases, deletion of the object is not a deletion from the database, but the deletion that occurs due to revision control when an insert of a new history entry occurs. This can be a major issue when 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. Although STEP continues to retry the import, this can cause inbound feeds to be negatively affected.

The 'Relaxed' setting on the property puts a less restrictive lock on the reference target objects being edited so that they can be updated concurrently by more than one process and/or user, with a full lock being employed only in the case that a deletion is attempted. STEP no longer locks the object. Instead the object type ID receives a shared 'user lock.' This means that multiple imports in parallel can reference the same objects without issues.

This setting, when used with object types that are frequently referenced but rarely deleted, will improve the performance and stability of parallel inbound imports, bulk updates, and users concurrently creating references to the same objects.



These Relaxed / Strict settings can be changed back and forth as needed, but changing the setting requires the system to enter Single Update Mode for a short time, only as long as it takes to update the setting. Keep in mind, however, that 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 have an associated 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, see the **Events** section of the **System Setup** documentation. For information about setting up an event processor, see **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 Reporter** 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 Forward, Rewind, Purge, and Republish**.
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 outbound integration endpoint definition for comparison purposes in an external source control system in **Exporting Outbound Integration Endpoint Definitions as Comments**.

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 Reporter** 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 outbound integration endpoint definitions for comparison purposes in an external source control system as described in **Exporting Outbound Integration Endpoint Definitions as Comments**.

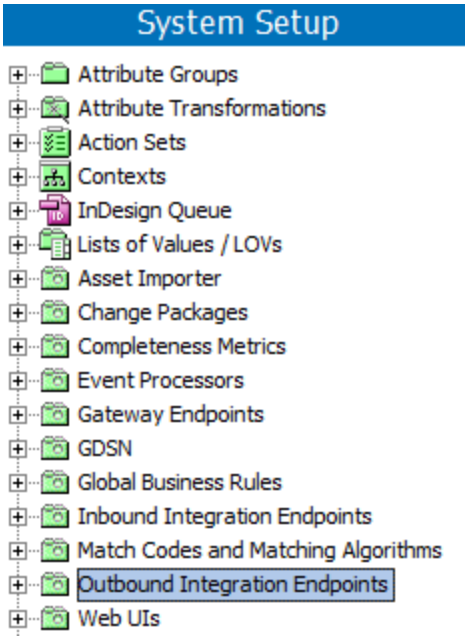
Initial Setup for an Outbound Integration Endpoint

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, see the **Action Sets** section and the **Users and Groups** section in the **System Setup / Super User Guide** documentation.

Note: Attempting to delete a user who is responsible for an OIEP will result in errors. For more information, see the **Working with Users** topic in the **System Setup / Super User Guide** documentation.

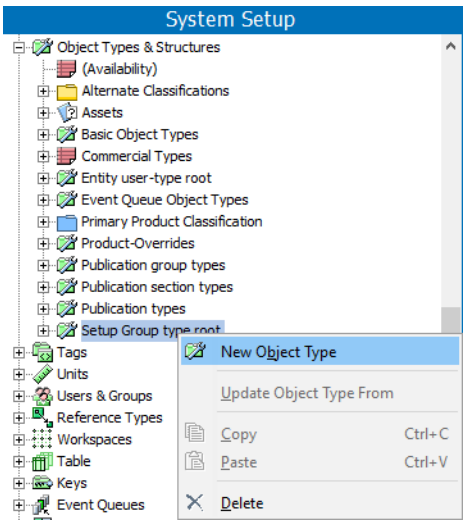
Review your System Setup tab to determine if one or more outbound integration endpoint nodes already exist. The name of the node on your system is not required to match the one in the image below.



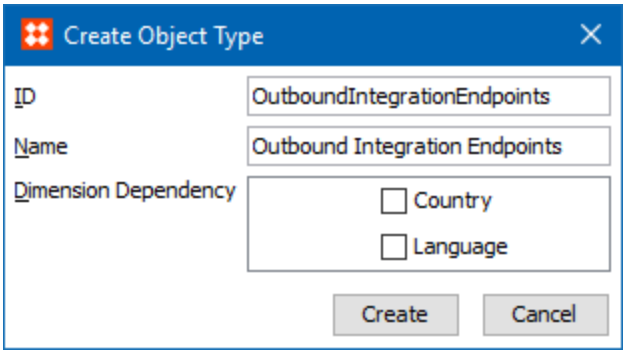
Once the setup has been completed, the steps in this section are only needed if additional levels of organization are desired.

Create the Outbound Integration Setup Group

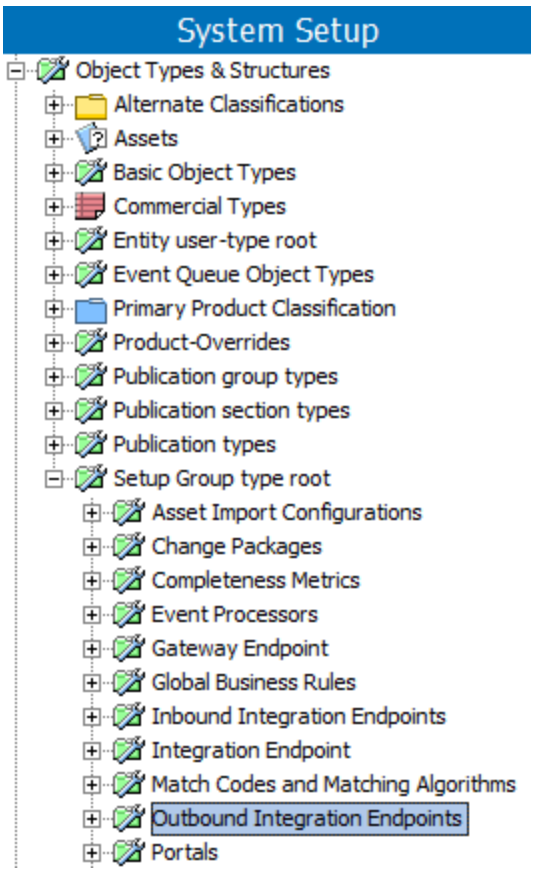
1. In System Setup, expand **Object Types & Structures**.
2. Right-click 'Setup Group type root', and choose **New Object Type**.



3. Enter an **ID** and a **Name**, select any required **Dimension Dependencies**, and click **Create**.

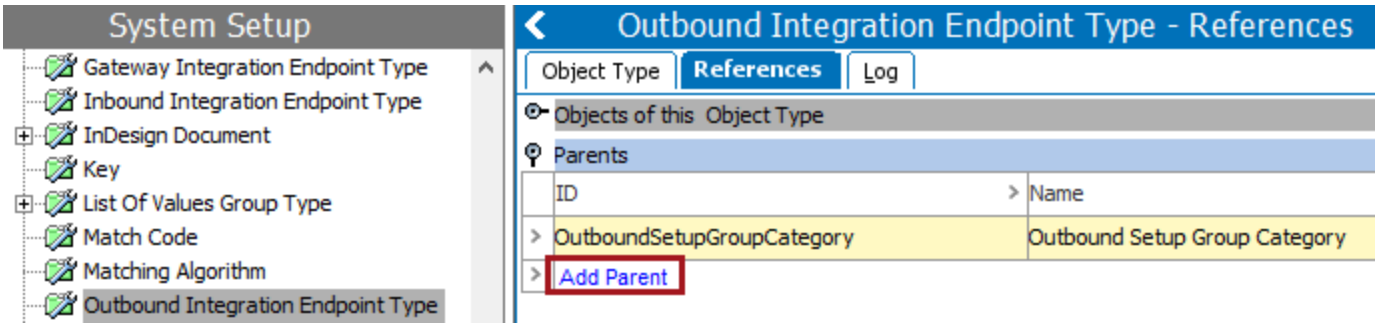


4. The new setup group appears in System Setup under 'Object Types & Structures' as a child in the **Setup Group type root**.

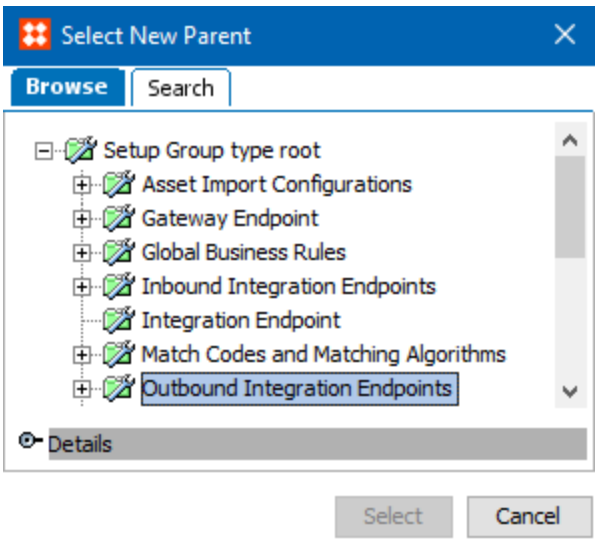


Link the OIEP Object Type to Setup Group

1. In Object Types & Structures > expand Basic Object Types > select **Outbound Integration Endpoint Type**.
2. On the References tab > Parents flipper > click the **Add Parent** link.

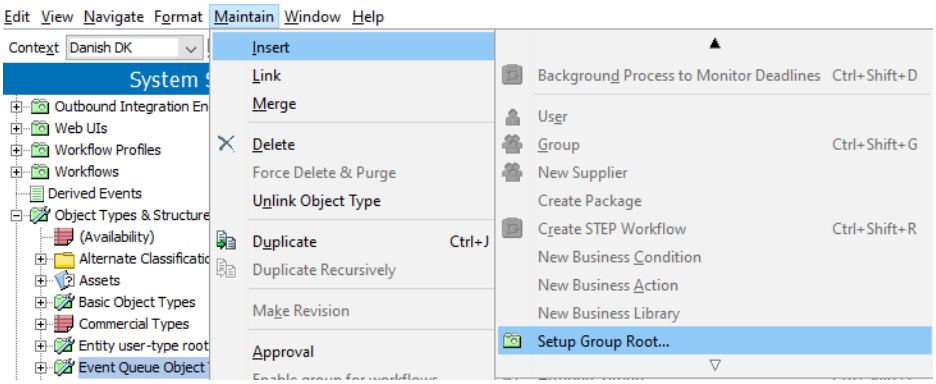


3. In the Select New Parent dialog, select the setup group you created, and click **Select** to make it a valid parent.

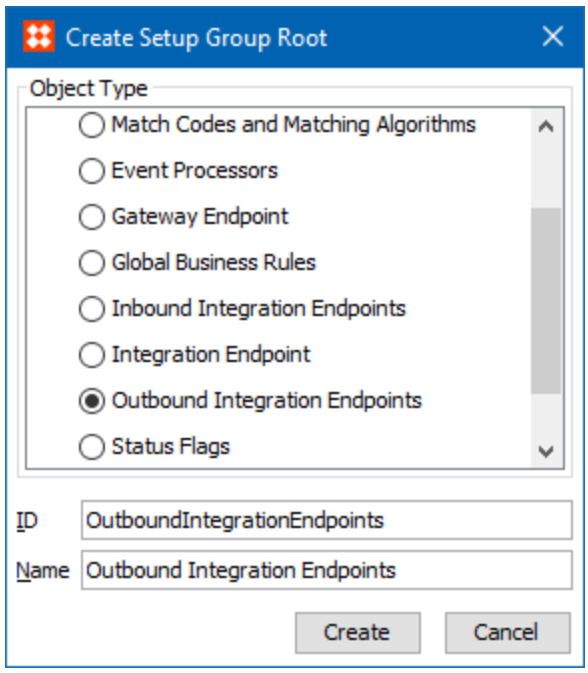


Create an Instance of the OIEP Object

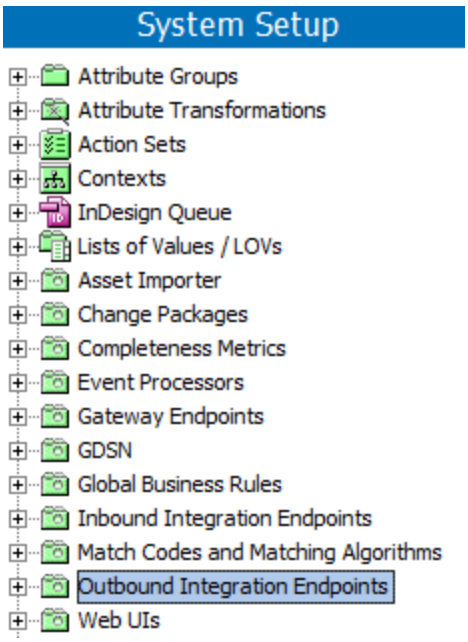
1. On the System Setup tab, select any object in the **System Setup** hierarchy to activate the following Maintain menu selection.
2. Click the Maintain menu, point to Insert, and select **Setup Group Root**.



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:

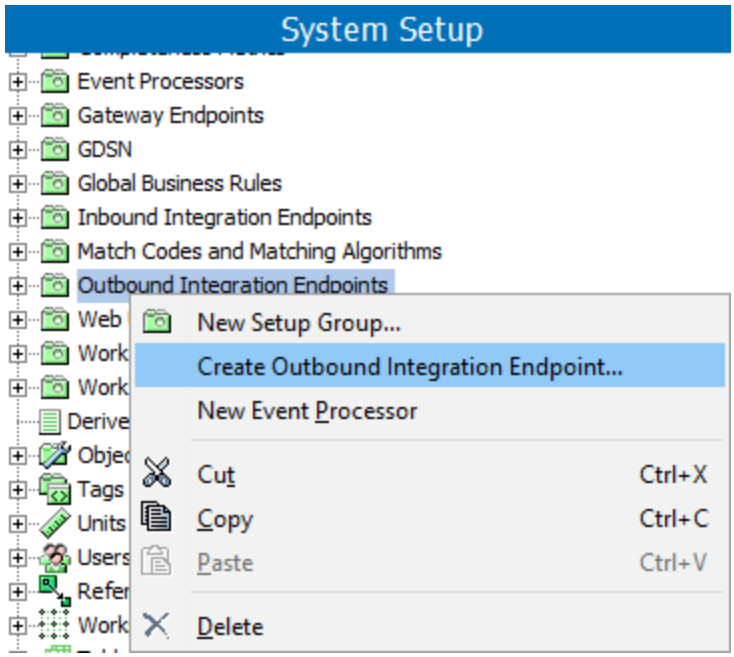
- To create an event-based OIEP that can monitor STEP for changes and export as needed, see **Creating an Event-Based Outbound Integration Endpoint**.
- To create an OIEP to send a static set of STEP data , see **Creating a Select Objects Outbound Integration Endpoint**.

Creating an Event-Based Outbound Integration Endpoint

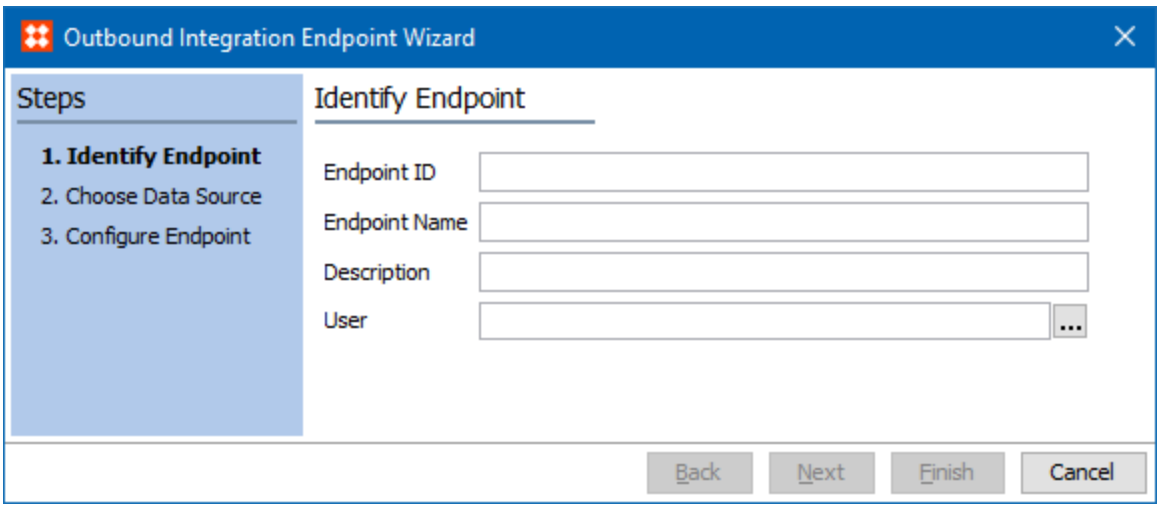
An event-based OIEP is used to monitor for approved data changes in STEP (based on the revisability setting of the changed data), and to send the changed data to one or more external systems.

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, see **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**.

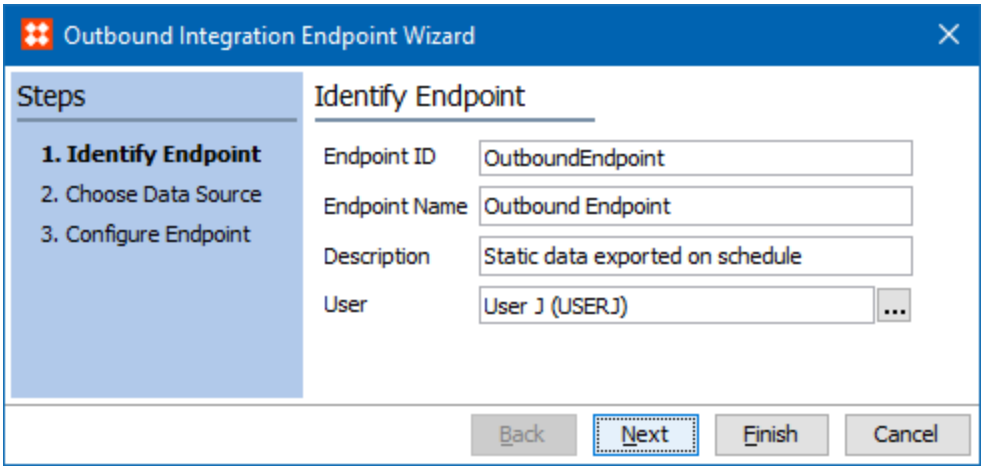


2. The Outbound Integration Endpoint wizard displays and involves the following steps:



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.

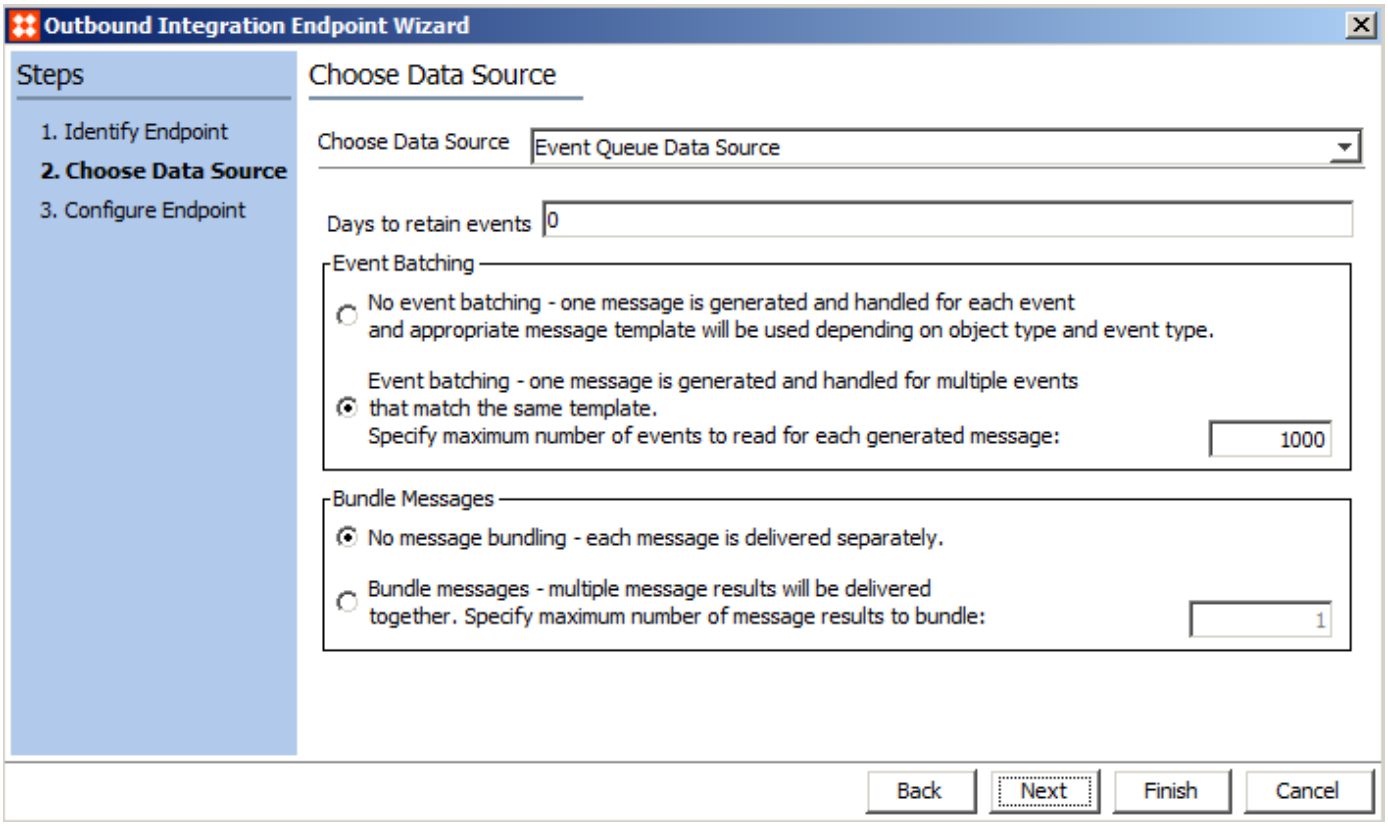
1. Enter an **ID**. Common setup is to use no spaces or punctuation.
2. Enter a **Name**. Common setup is to repeat the ID with added spaces for readability.
3. Enter an optional **Description**. It is good practice to add description for future reference.
4. Use the search or browse for a **User**. 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, see the **Working with Users** topic in the **System Setup / Super User Guide** documentation.

---

5. Click **Next** to display **OIEP - Event-Based - Choose Data Source**.

OIEP - Event-Based - Choose Data Source

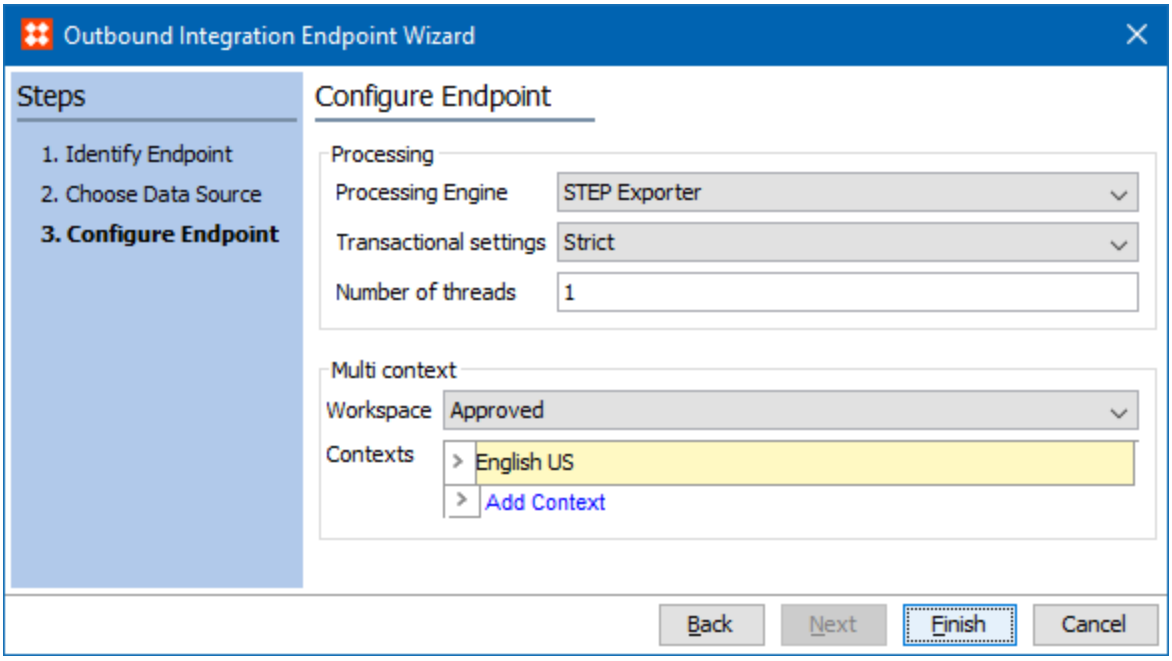


1. Choose the data source **Event Queue Data Source** for incremental exports where data should be published based on events. Creating an event-based OIEP automatically creates an event queue to process events.

   Note: A **Select Objects** data source requires a separate setup, for detailed steps, see 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, see **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, see the **OIEP - Configuration Flipper for Business Rule Based Message Processor** topic or click the **STEP API Documentation** button on the STEP Start Page.
   - If you need a customized processing engine, contact your Stibo Systems account manager or partner manager. For more information, see **Outbound Integration Endpoint Structure** documentation.
2. The **Transactional Settings** option must be set to 'Strict' for event-based OIEPs. For more information, see the **Integration Endpoint Transactional Settings** documentation.
3. The **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 number of threads, you must also consider hardware limitations. For more information, see **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.

5. Click **Finish** to complete the wizard.
6. Perform the manual configuration described in **OIEP - Event-Based - Manual Configuration**.

OIEP - Event-Based - Manual Configuration

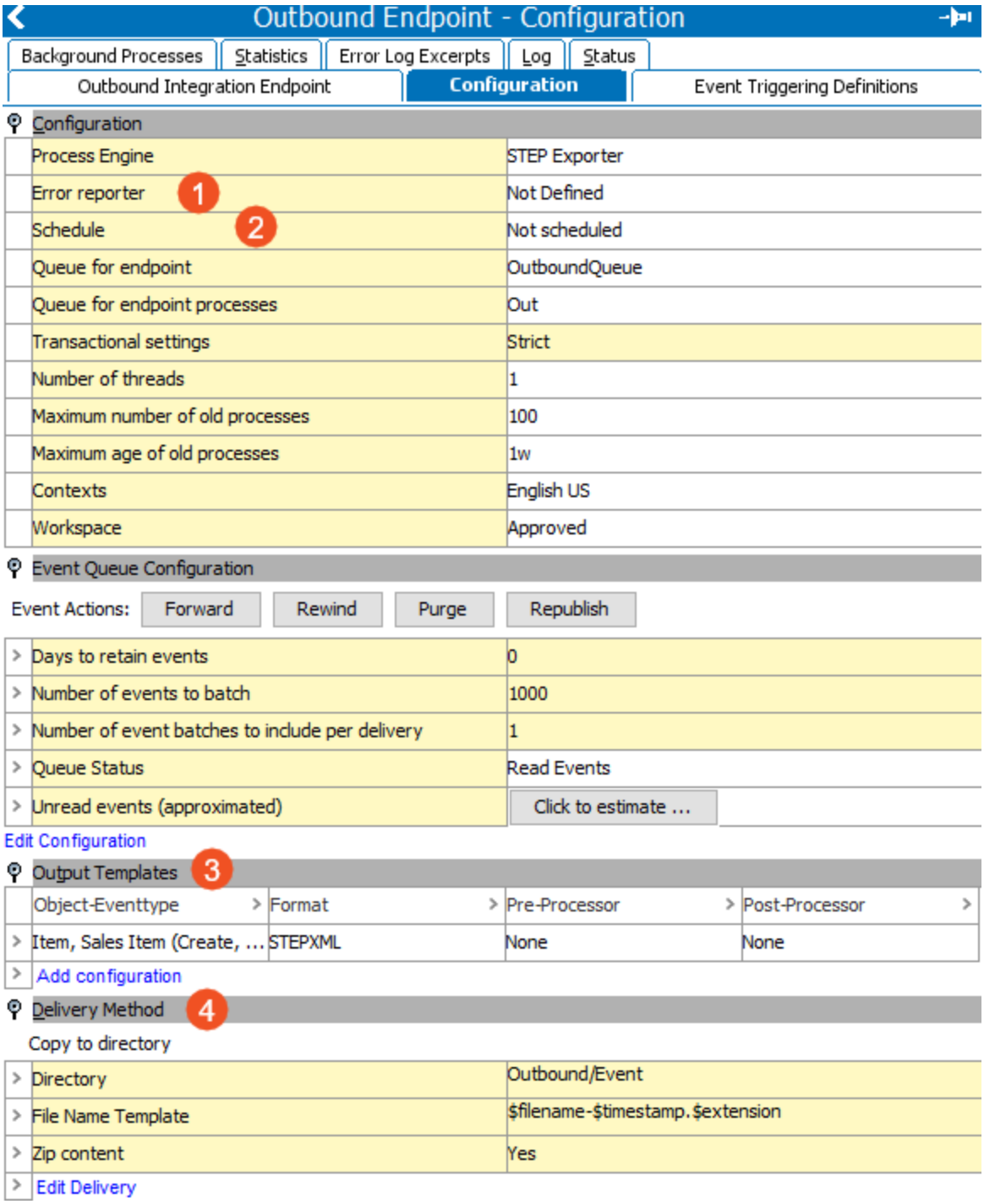
After completing the Outbound Integration Endpoint wizard for an event-based OIEP using the STEP Exporter process engine, the OIEP editor includes additional parameters that must be set manually before the OIEP can run.

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.

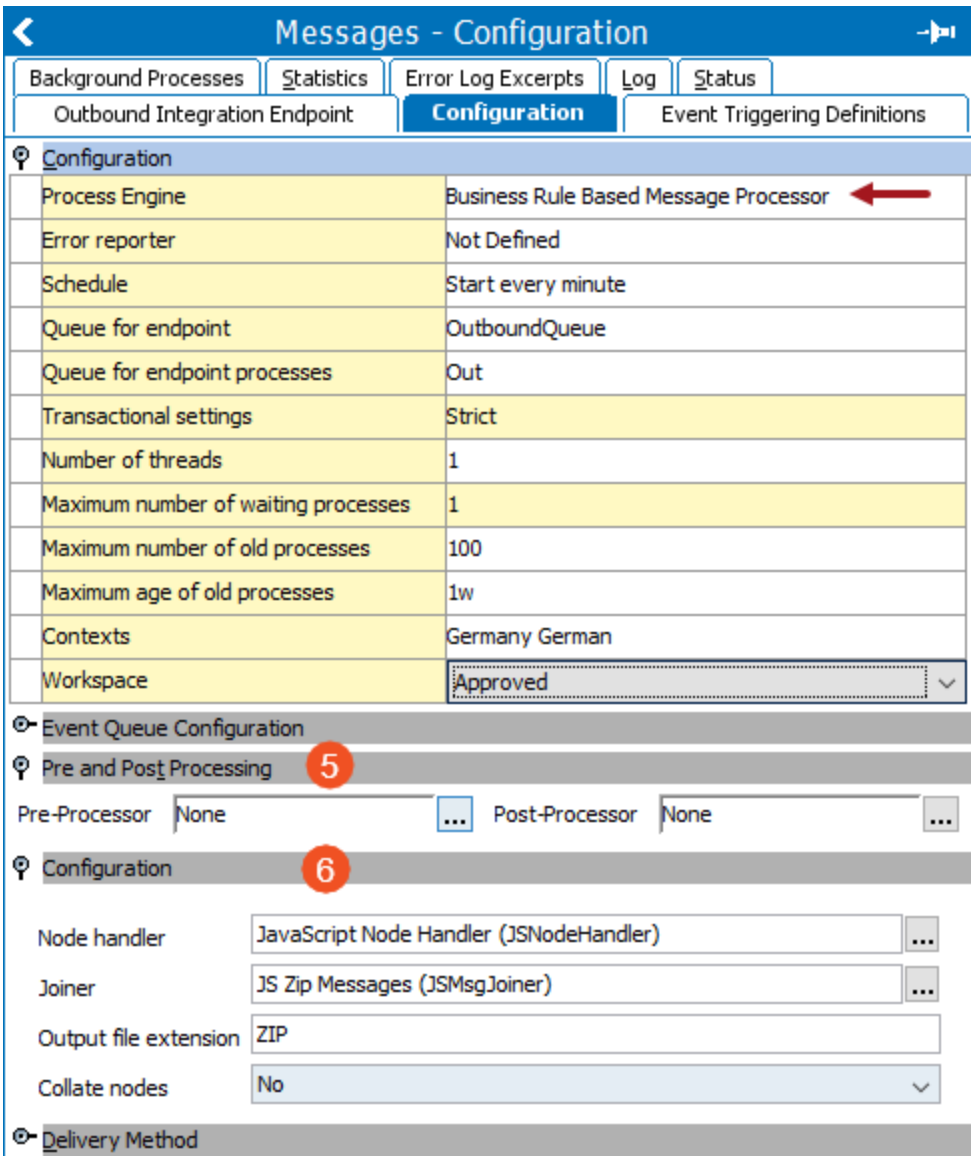
The configuration set from the wizard can also be modified manually using the Configuration tab.

Follow these steps to complete the manual configuration and set the event-based OIEP to run:

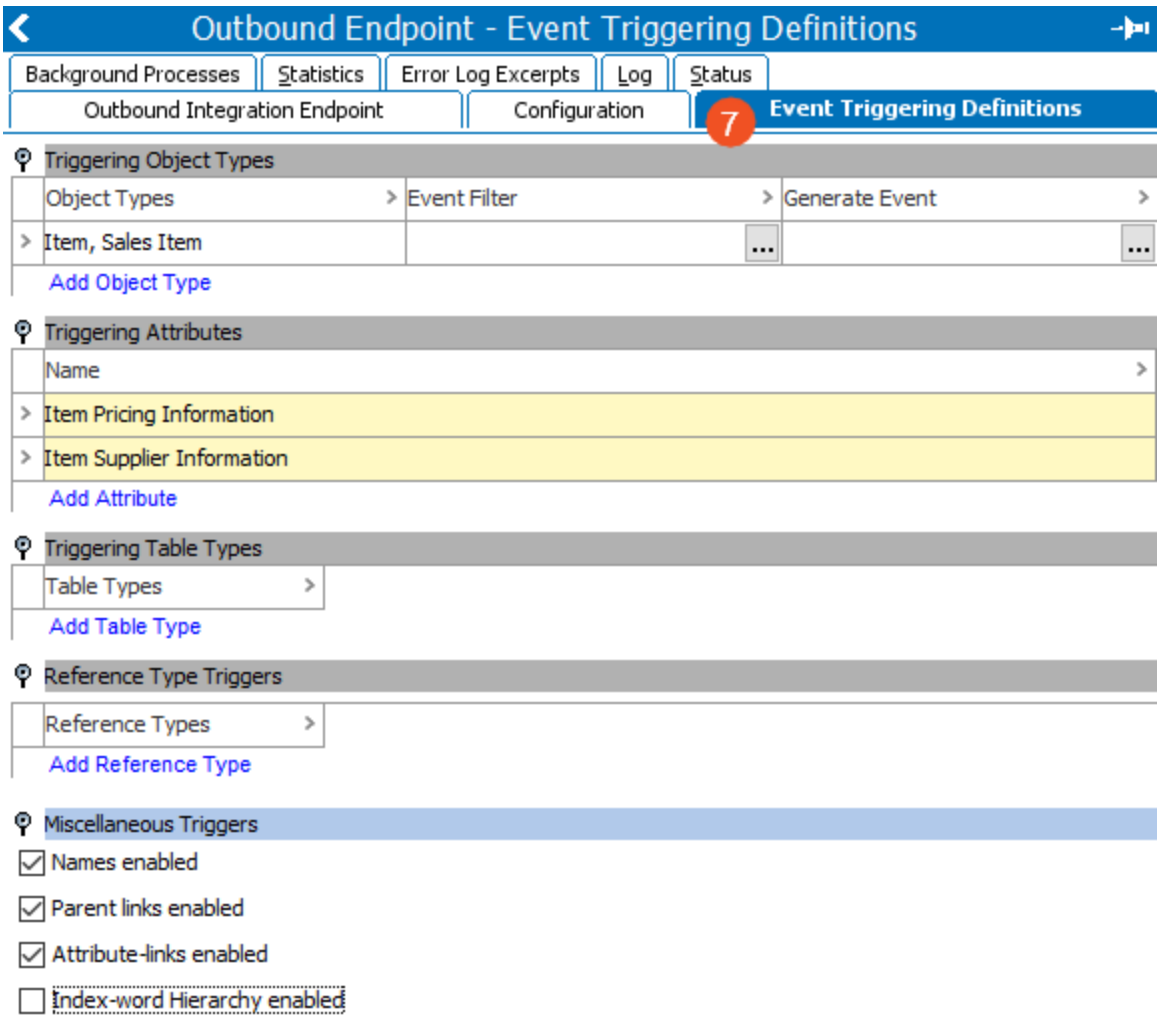
1. In the OIEP editor, configure to send an email if an endpoint-related background process fails as described in the **Error Reporter** section of **OIEP - Configuration Flipper**.
2. In the OIEP editor, determine when the OIEP runs as described in the **Schedule** section of **OIEP - Configuration 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**.



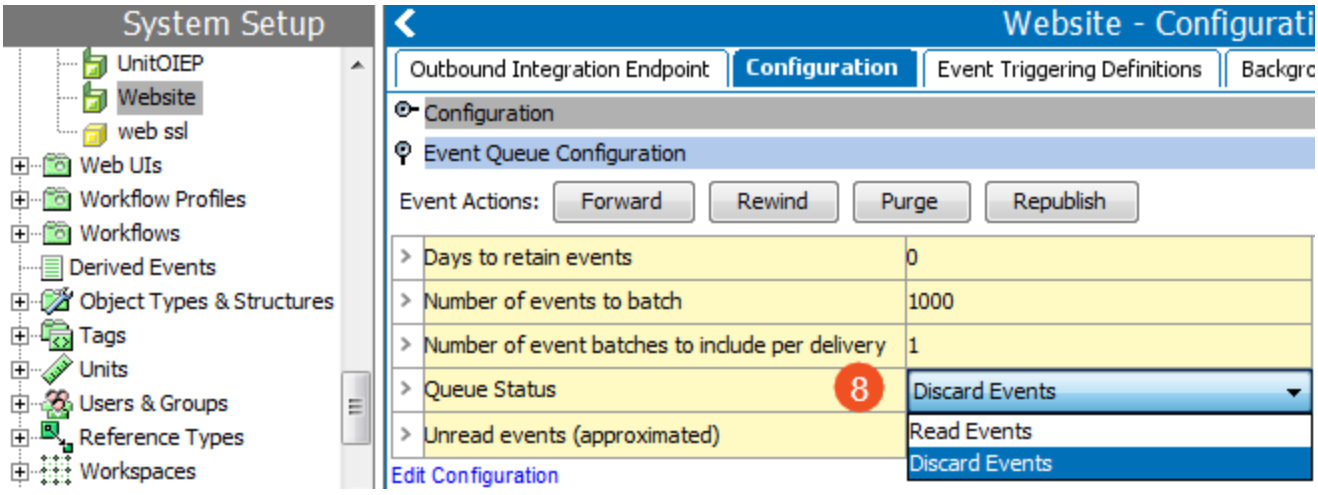
5. In the OIEP editor, when using the Business Rule Based Message Processor process engine, specify the pre- or post-processors (if any) as described in **OIEP - Pre and Post Processing Flipper**.
6. In the OIEP editor, when using the Business Rule Based Message Processor process engine, specify the business actions and settings as described in **OIEP - Configuration Flipper for Business Rule Based Message Processor**.



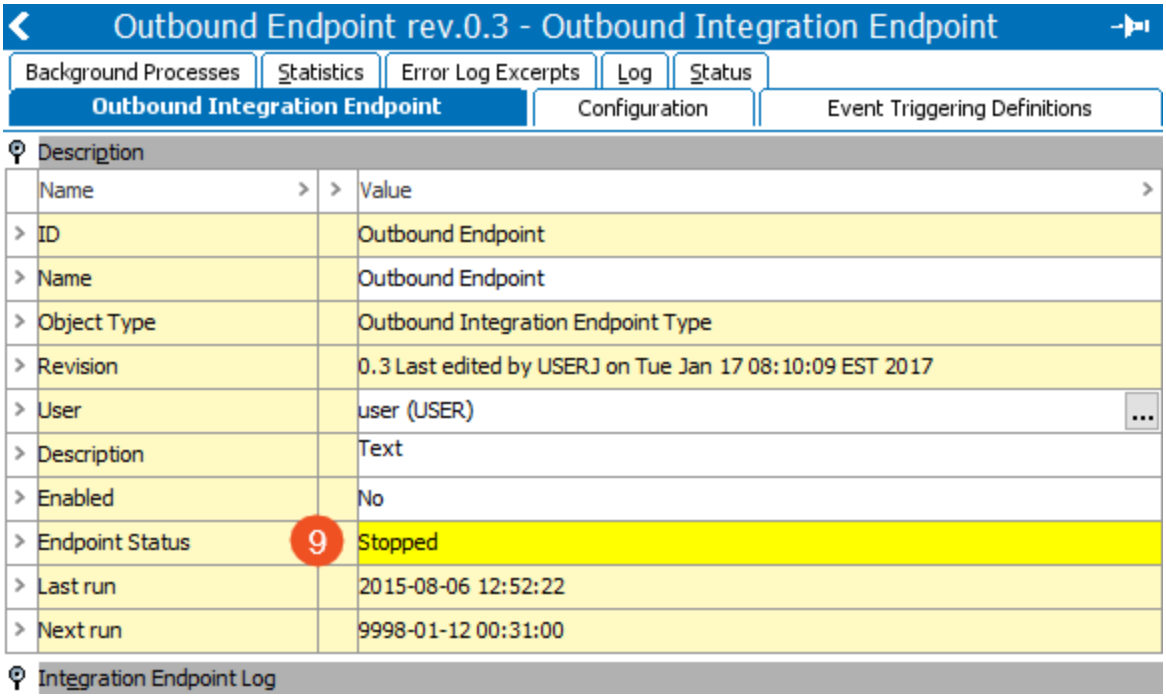
7. In the OIEP editor, define what data changes will cause the OIEP to run as described in the **OIEP - Event-Based - Event Triggering Definitions Tab**.



8. Set the Queue Status as described in **Event-Based OIEP Status and Queue Status**.

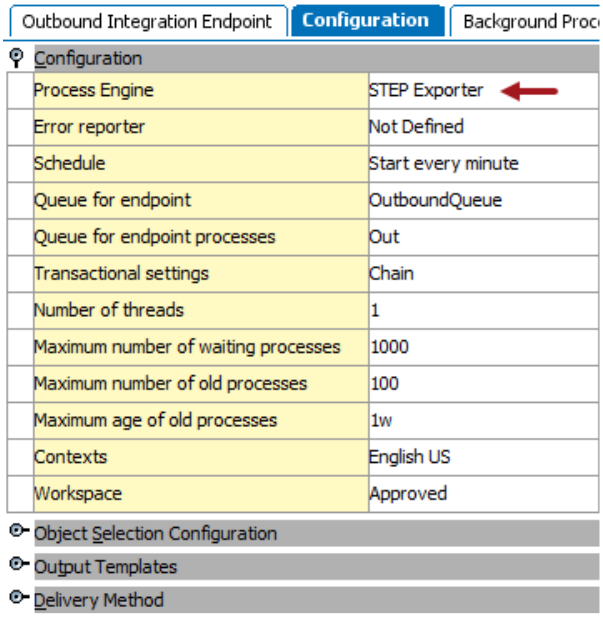


9. Enable the endpoint and invoke it as described in **Running an Outbound Integration Endpoint**, paying particular attention to the **Prerequisites for Event-Based OIEPs** section.



OIEP - Configuration Flipper

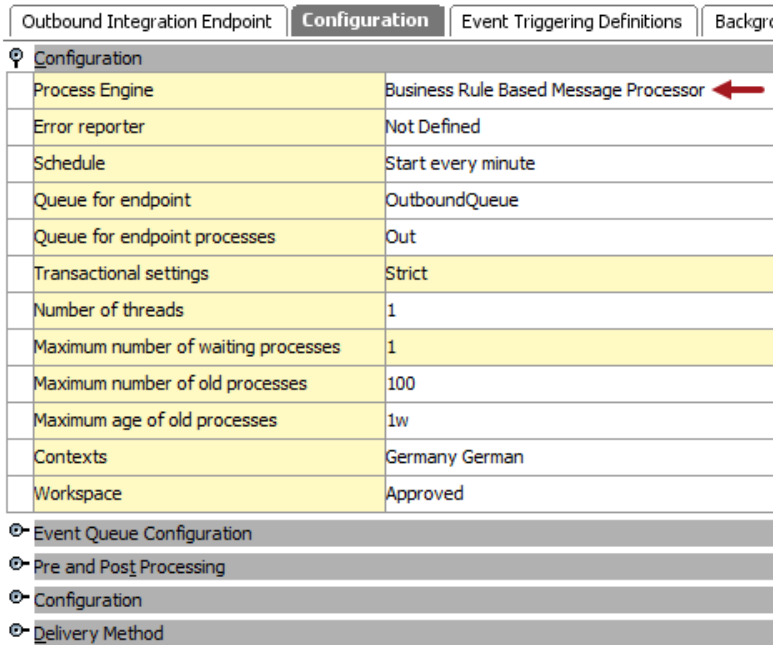
The following flippers are available when the STEP Exporter processor or the Business Rule Based Message processor is selected for an OIEP. For both processors, the Configuration flipper includes the same parameters for both Event-Based and Select Objects endpoints. Each parameter is described below.



Outbound Integration Endpoint | Configuration | Background Proc

Configuration	
Process Engine	STEP Exporter
Error reporter	Not Defined
Schedule	Start every minute
Queue for endpoint	OutboundQueue
Queue for endpoint processes	Out
Transactional settings	Chain
Number of threads	1
Maximum number of waiting processes	1000
Maximum number of old processes	100
Maximum age of old processes	1w
Contexts	English US
Workspace	Approved

Object Selection Configuration
Output Templates
Delivery Method



Outbound Integration Endpoint | Configuration | Event Triggering Definitions | Backgr

Configuration	
Process Engine	Business Rule Based Message Processor
Error reporter	Not Defined
Schedule	Start every minute
Queue for endpoint	OutboundQueue
Queue for endpoint processes	Out
Transactional settings	Strict
Number of threads	1
Maximum number of waiting processes	1
Maximum number of old processes	100
Maximum age of old processes	1w
Contexts	Germany German
Workspace	Approved

Event Queue Configuration
Pre and Post Processing
Configuration
Delivery Method

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, see **OIEP - Event-Based - Configure Endpoint** documentation or **OIEP - Select Objects - Configure Endpoint** documentation.

Error Reporter

The 'Error reporter' parameter is not included in the wizard, but it can be configured to send an error report email if a background process fails. The email contains information about the failed endpoint, including the background process, failed file, failing process step, cause of the error, and a copy of the file that triggered the error. A standard STEP system (without custom extension) includes only the 'Send Error Report' option.

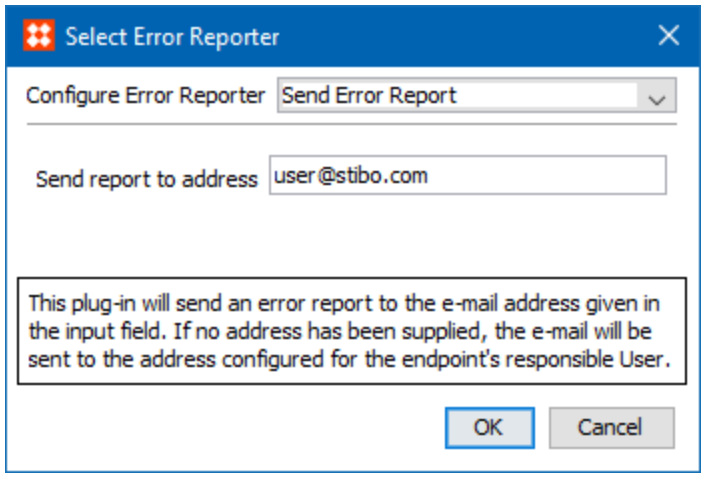
For information on configuring email from STEP, see the **Email from STEP** topic in the **Resource Materials** of online help.

By default, error reports are sent when either errors or warnings are encountered. To disable error reports when only warnings are reported, update the sharedconfig.properties file on the app server to include the following property:

```
Integration.Endpoint.SendEmailErrorReportOnWarning=false
```

Configure the Error Report Email

1. On the relevant OIEP > Configuration tab > Configuration flipper > click into the **Error Report** parameter to display the ellipsis button (...).
2. Click the ellipsis button (...) to display the **Select Error Reporter** dialog.



3. In Configure Error Reporter field, select **Send Error Report**.
4. In **Send report to address** field, enter one or more email addresses for the error report recipient(s). Use a semicolon-separated list to add more than one email address.

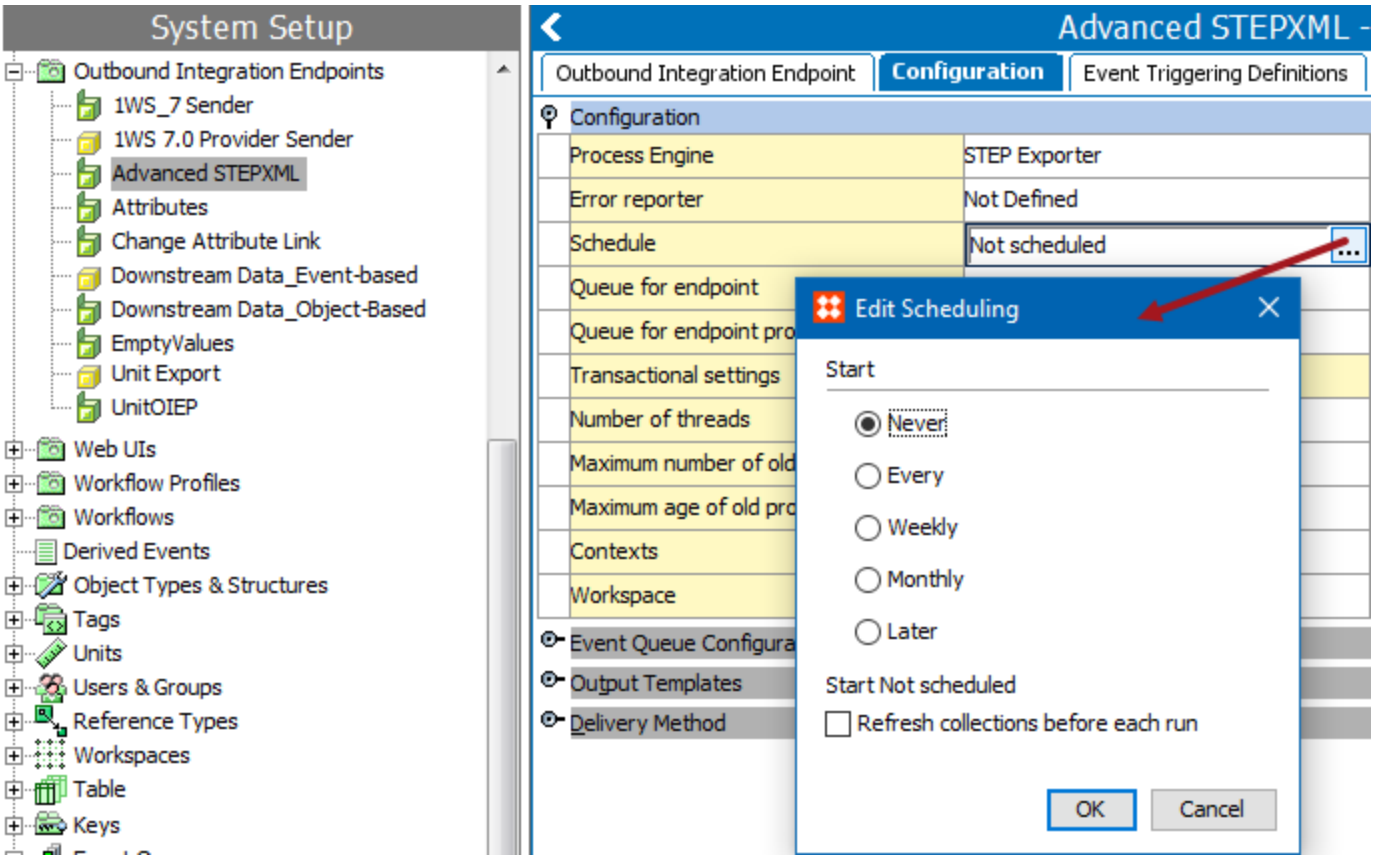
Note: If no email address is entered, the error report will be sent to the email address of the user who created the integration endpoint. If no email is defined for this user, or if no mail server is defined in the configuration, the error report will be written to the failed Background Process Execution Log.

Schedule

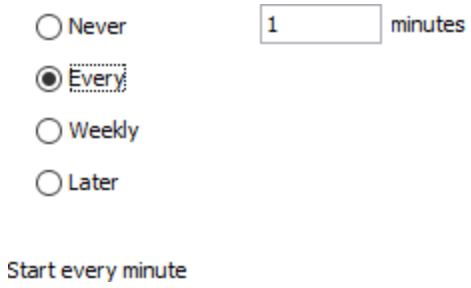
The schedule parameter is not included in the wizard, but it can be configured to repeatedly run the OIEP. For Select Objects OIEPs, the background processes runs as scheduled, regardless of changes to the data. For Event-Based OIEPs, the background process runs as scheduled but only when events are 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.

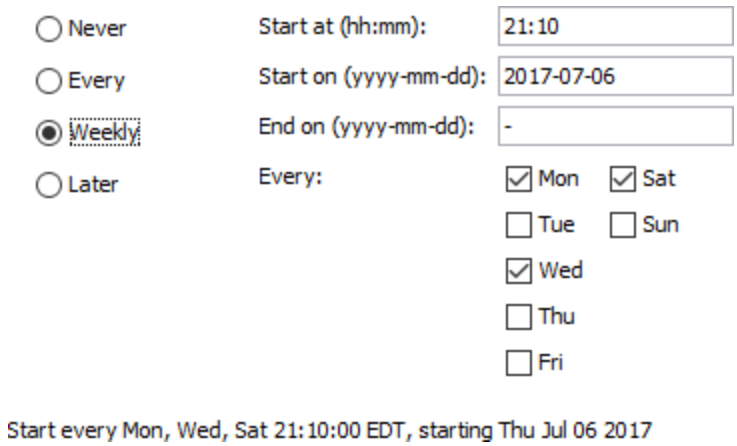
1. On the relevant OIEP > Configuration tab > Configuration flipper > click into the **Schedule** parameter to display the ellipsis button (...).
2. Click the ellipsis button (...), and then select one of the following options:



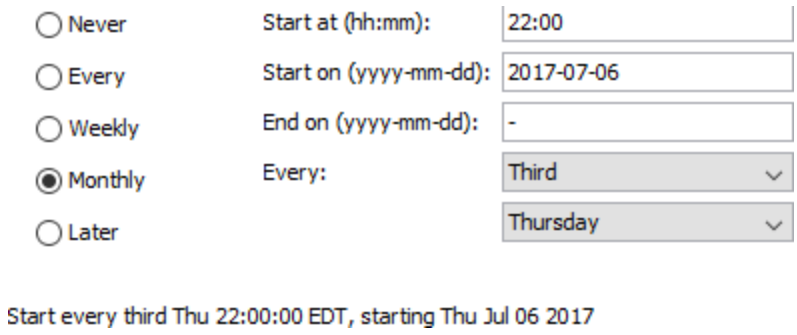
- **Never** - invoke the endpoint manually, no additional parameters are required, and no schedule is applied. This is the default setting and should be used while testing your endpoint.
- **Every** - automatically run the endpoint repeatedly, every selected number of minutes. One (1) minute is the shortest 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.



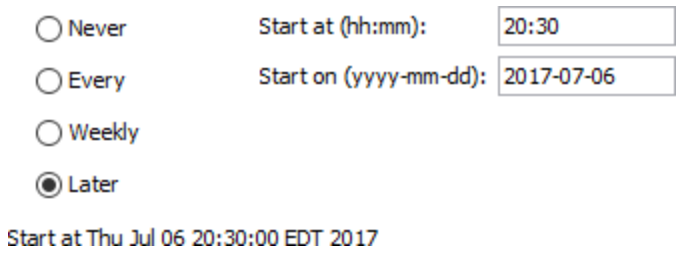
- **Weekly** - automatically run the endpoint repeatedly, based on the selected time, start and end dates, and days of the week. Use this option if a daily schedule is needed. The 'Start at' parameter determines the time of day that the endpoint will run. The 'Start on' parameter determines the date 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.



2. Select **Refresh collections before each run** if the outbound integration endpoint contains collections that should be automatically refreshed before each run.

Queue for Endpoint

'Queue for endpoint' is the name for the queue that is used by the OIEP Background Process to poll the endpoint. 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 OIEP, for example, including the OIEP ID in the name.

Queue for Endpoint Processes

'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 OIEP 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 detail, see the **Integration Endpoint Transactional Settings** topic.

Number of Threads

Although '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), 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. For more information, see **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.

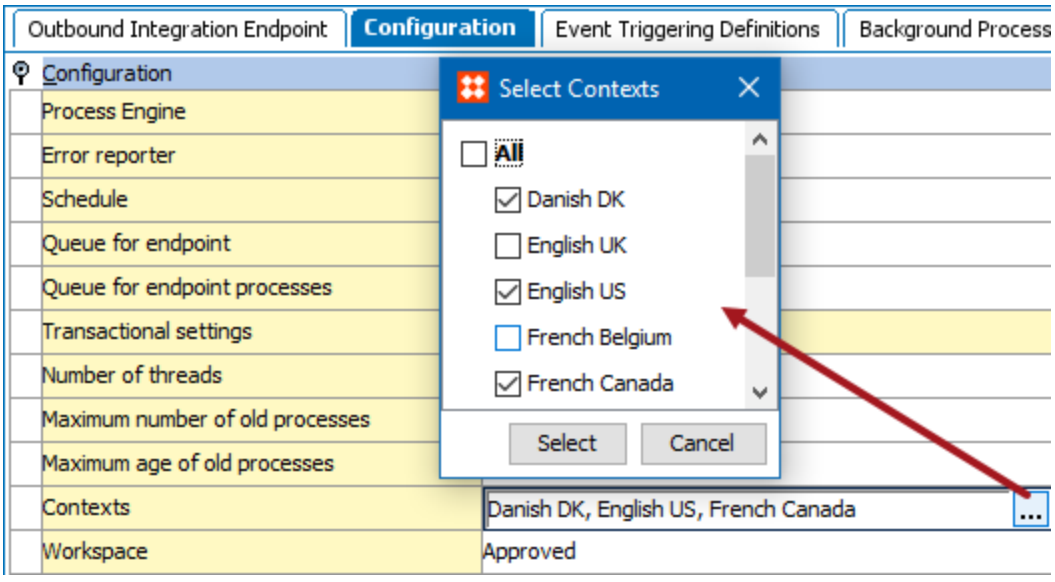
By default, the maximum age of old processes is set to one year.

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, see 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



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.

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.

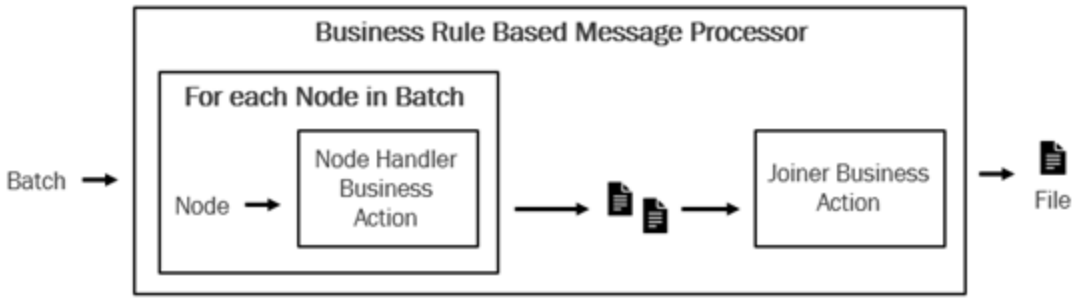
OIEP - Configuration Flipper for Business Rule Based Message Processor

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, see 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.



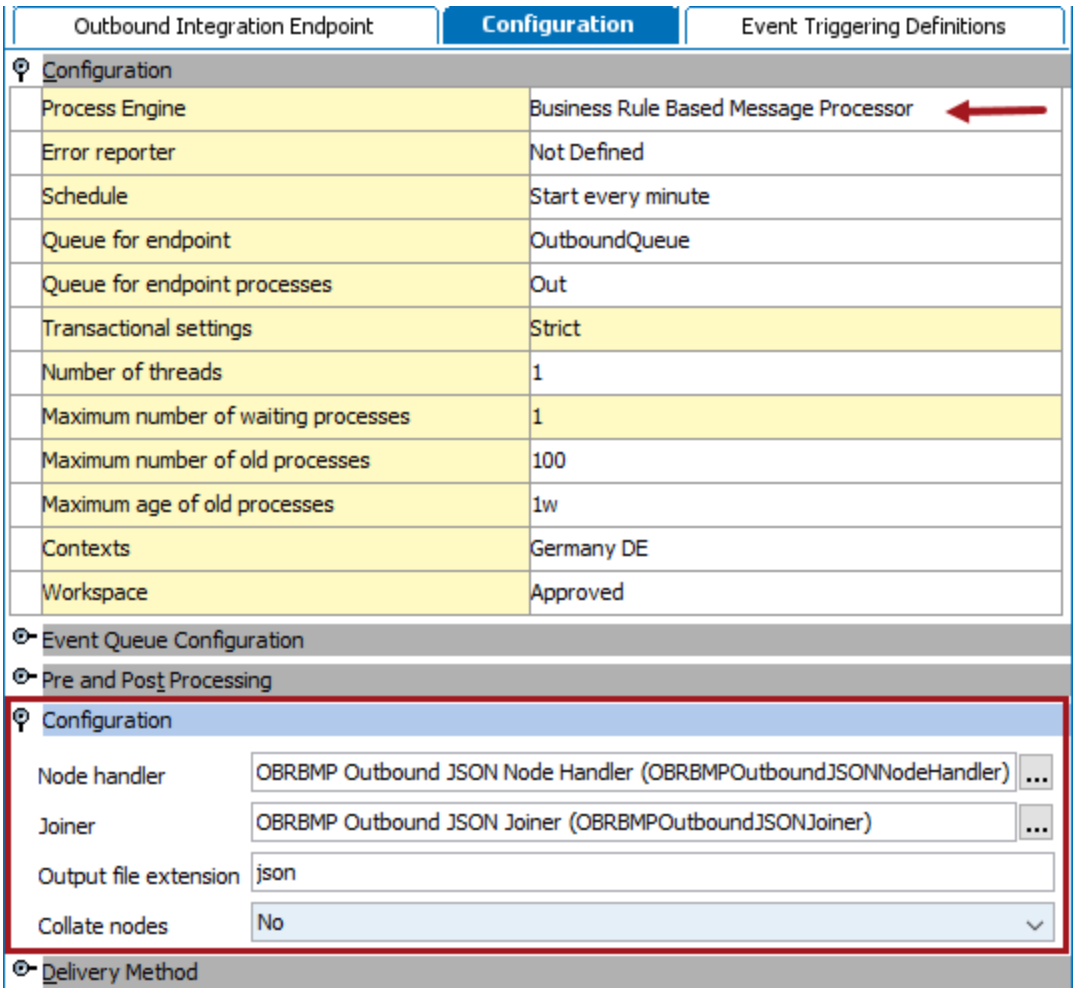
Prerequisites

1. Business actions used by this processor should be configured to be valid for all object types.
2. 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, see 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.

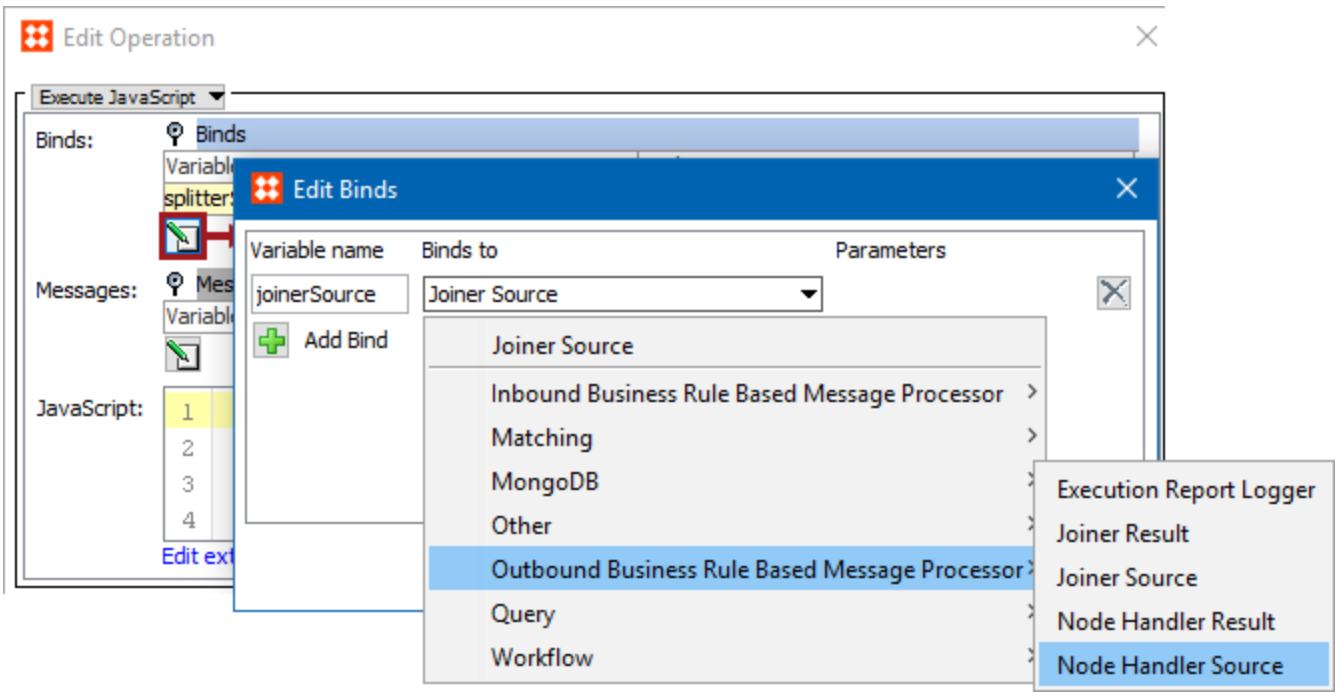
The **OutboundBusinessProcessorNodeHandlerSource** interface (for JavaScript business actions, the interface is available via the Node Handler Source bind option) provides the business action with access to the batch element. The interface has the following methods for accessing node / event information:

- **`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 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));
  }
}
```

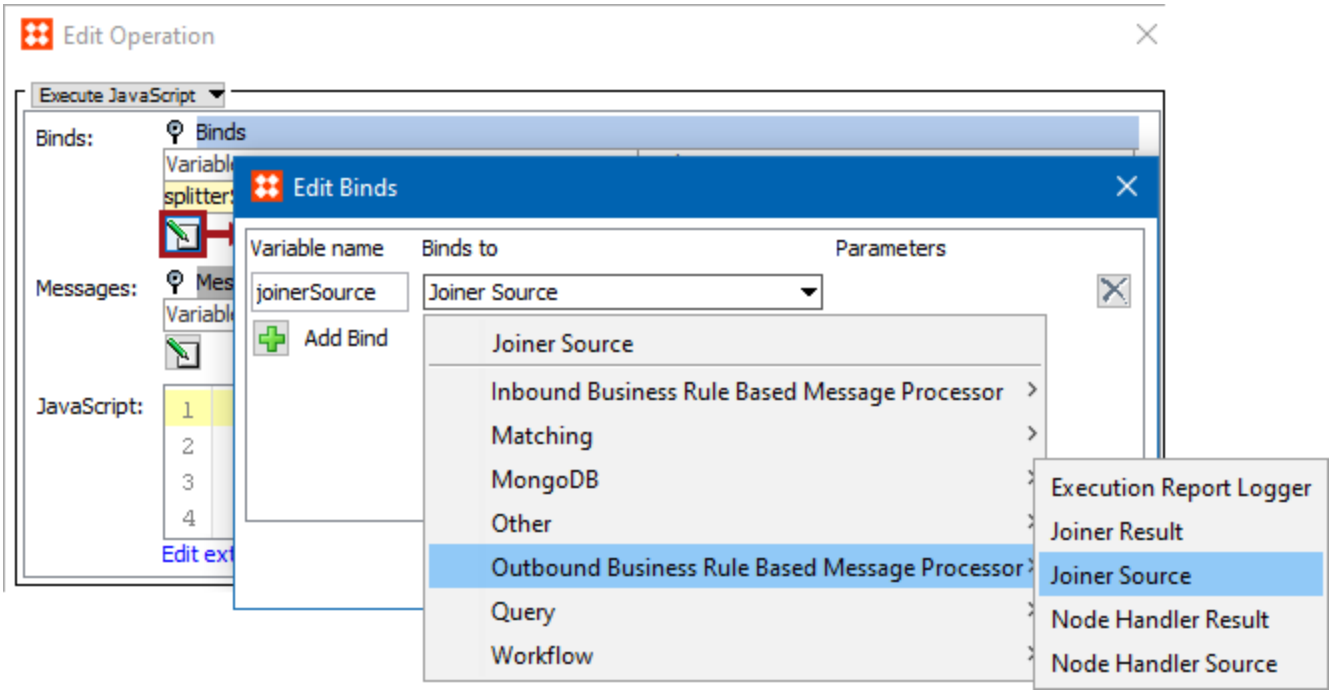
Joiner Business Action Details and Examples

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 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 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. See 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 preprocessor and post-processor, based on availability on your system

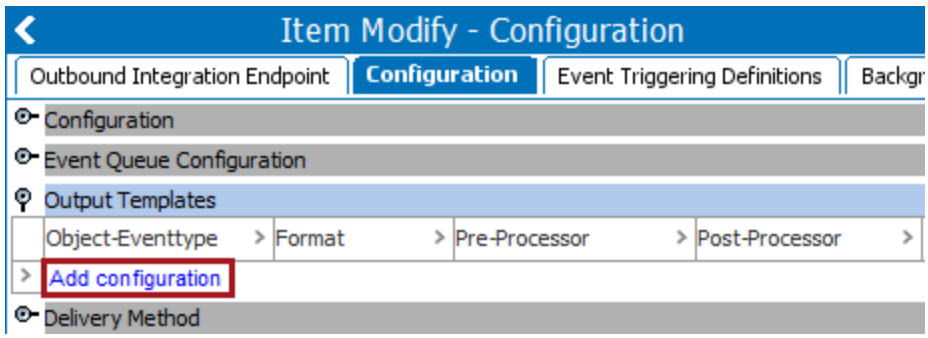
---

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, see **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.

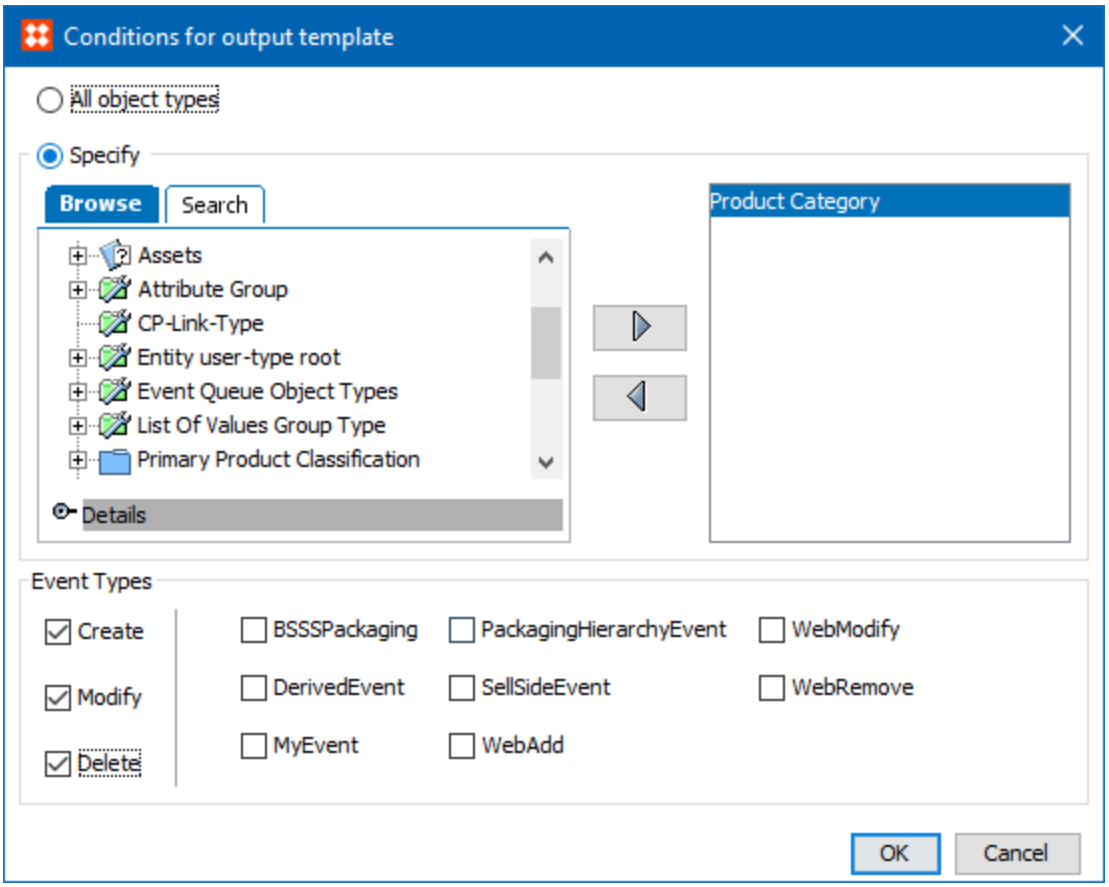
1. In System Setup, select the relevant OIEP > Configuration tab > Output Templates flipper > click the **Add configuration** link to display the 'Conditions for output template' dialog.



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, see the **Derived Events** section of the **System Setup \ Super User Guide** documentation.

  When using the Mongo delivery method, specific settings are required to deliver delete events. For details, see the Delivering Delete Events section of the **Mongo Delivery Method** 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

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.

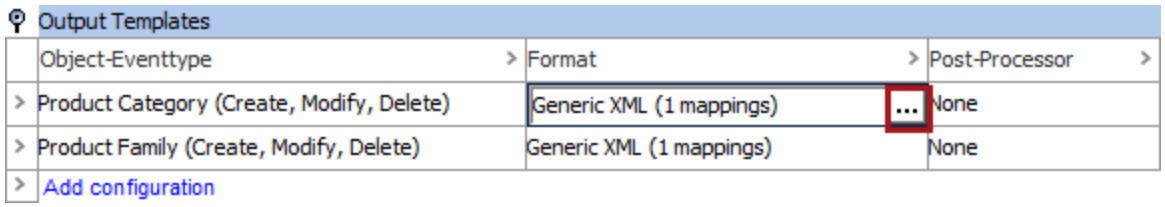
Limitations of Multiple Output Templates

- 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

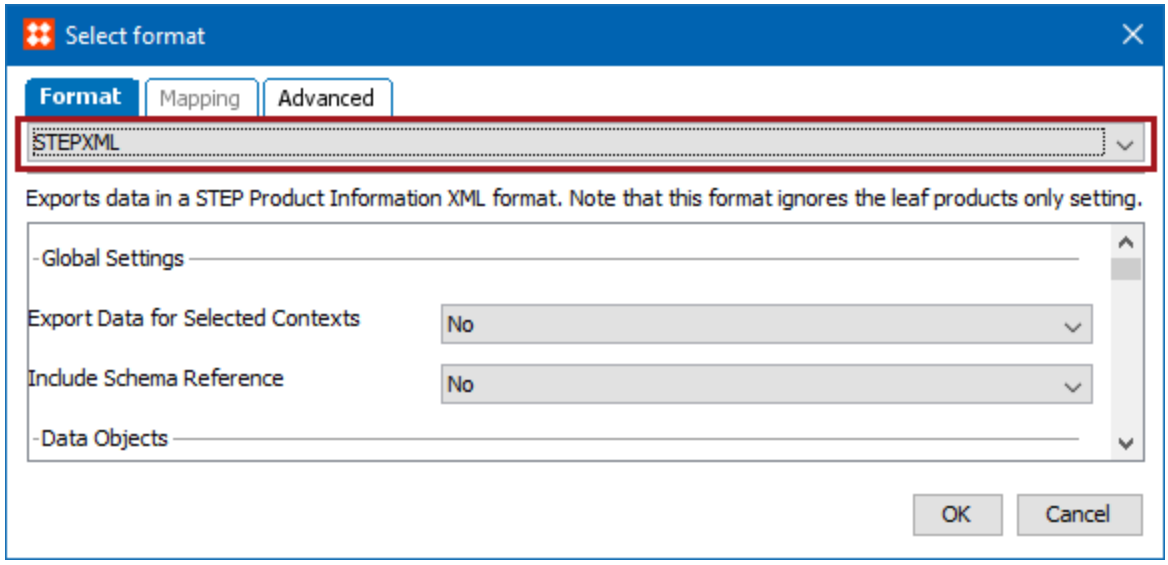
The format selection determines how the STEP data is formatted (XML or tabular).

1. Click into the format column to display the ellipsis button (...). Click the ellipsis button (...) to display the 'Select format' dialog.



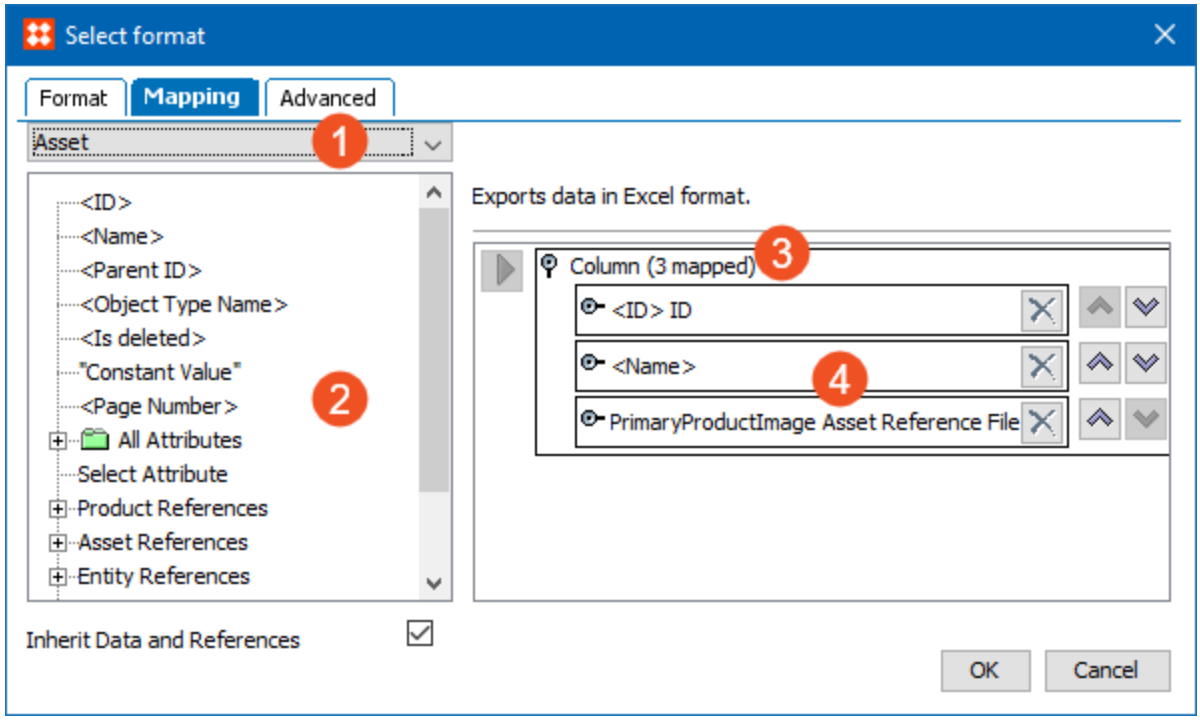
2. In the **Select format** window, on the **Format** tab, select the desired format from the dropdown. The options are the same as in the Export Manager, based on your system licenses. Use the following links for additional information:
   - Advanced STEPXML Format
   - Alphabetical Index - XML Format
   - 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
   - IDoc MATMAS05 Format
   - Publication Excel Format

- STEPXML Format
- xCBL Format



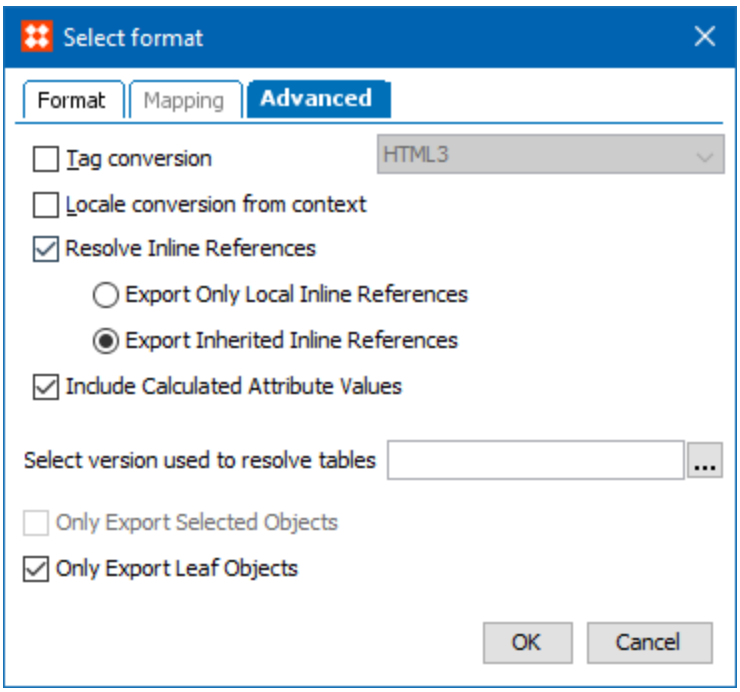
3. When available, specify how to map data to the export format, including the elements identified in the image below: object super type dropdown, data source, mapping target, and mapping rules.

   For more information about the object super type dropdown, see the **Export Manager - Select Objects** topic. For more information about the other elements, see 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, see the **Tags** topic in **System Setup / Super User Guide** 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, see **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, see **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, see the **Calculated Attributes** topic in **System Setup / Super User Guide** 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 only available if the format is STEPXML and 'Include Tables' is set to Yes. 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). If a publication version is not selected, the version last used in STEP Workbench is used. For more information, see 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, see **Export Manager - Advanced** documentation.

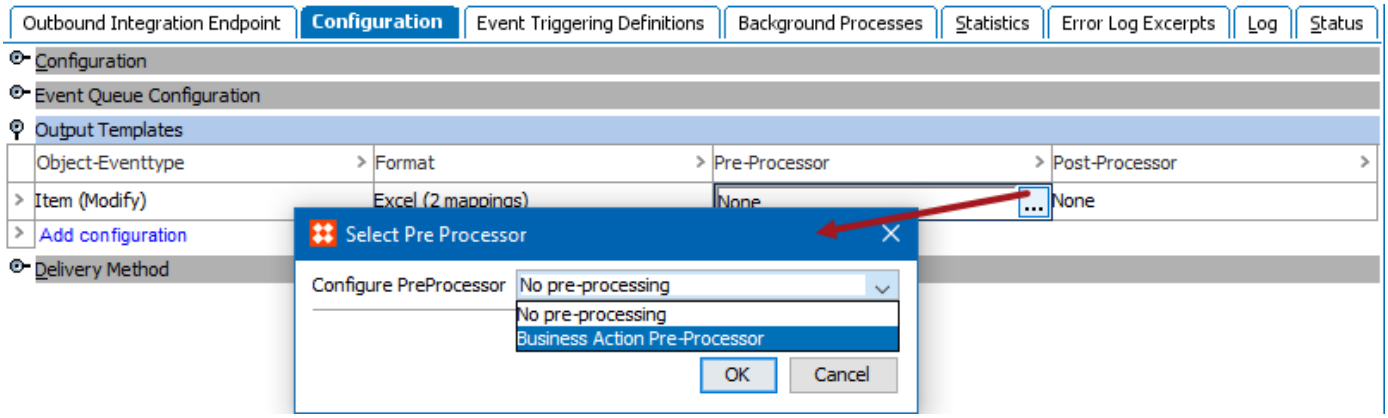
Configure the Pre-processor and Post-processor

Preprocessor and postprocessor options are available depending on your system configuration.

Pre-processor

For an event-based OIEP, the Business Action Pre-Processor can be applied before exporting events from a batch in an event queue. Applying the preprocessor 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

preprocessor for each of the output templates. For more information, see **OIEP - Pre-Processor - Business Action**.

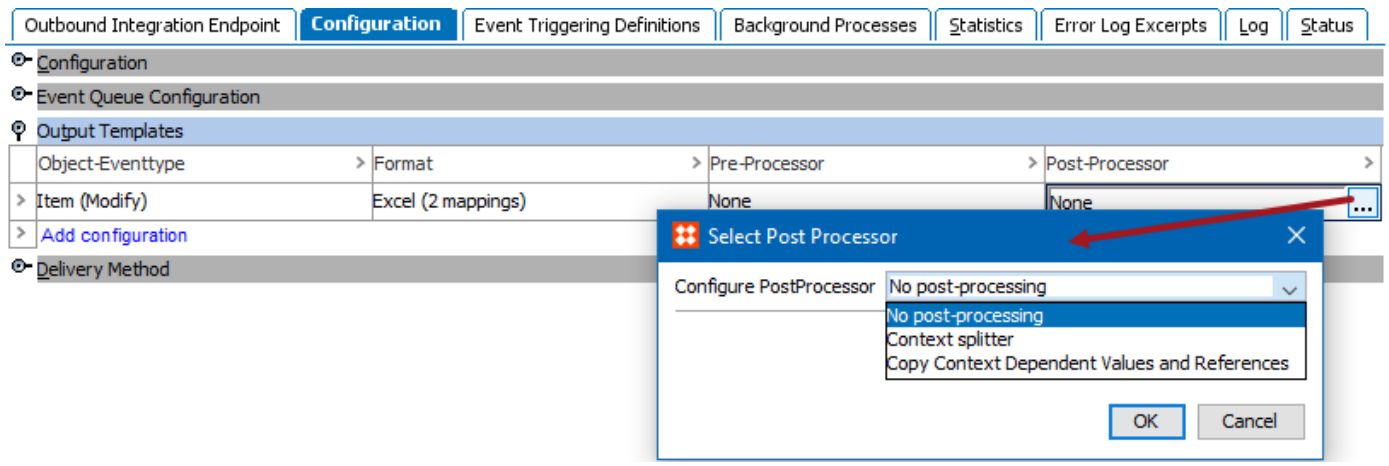


A Triggering Object Type event filter or Generate Event option can be used in place of a preprocessor. For more information, see the **OIEP - Event-Based - Event Triggering Definitions Tab** documentation.

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, see 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, see the **OIEP - Post-Processor - Context Splitter** documentation.

  Set **Copy inherited product values** parameter to **Yes** to copy and save inherited product values to the child product; or set to **No** to use inherited product values in the export, but leave the child product unmodified.

  When using any 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 in the wizard **OIEP - Configure Endpoint**.

  Set **Copy inherited product values** to **Yes** to copy and save inherited product values to the child product; set to **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 Context IDs 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.

  For more information, see the **OIEP - Post-Processor - Copy Context Dependent Values and References** section.

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:

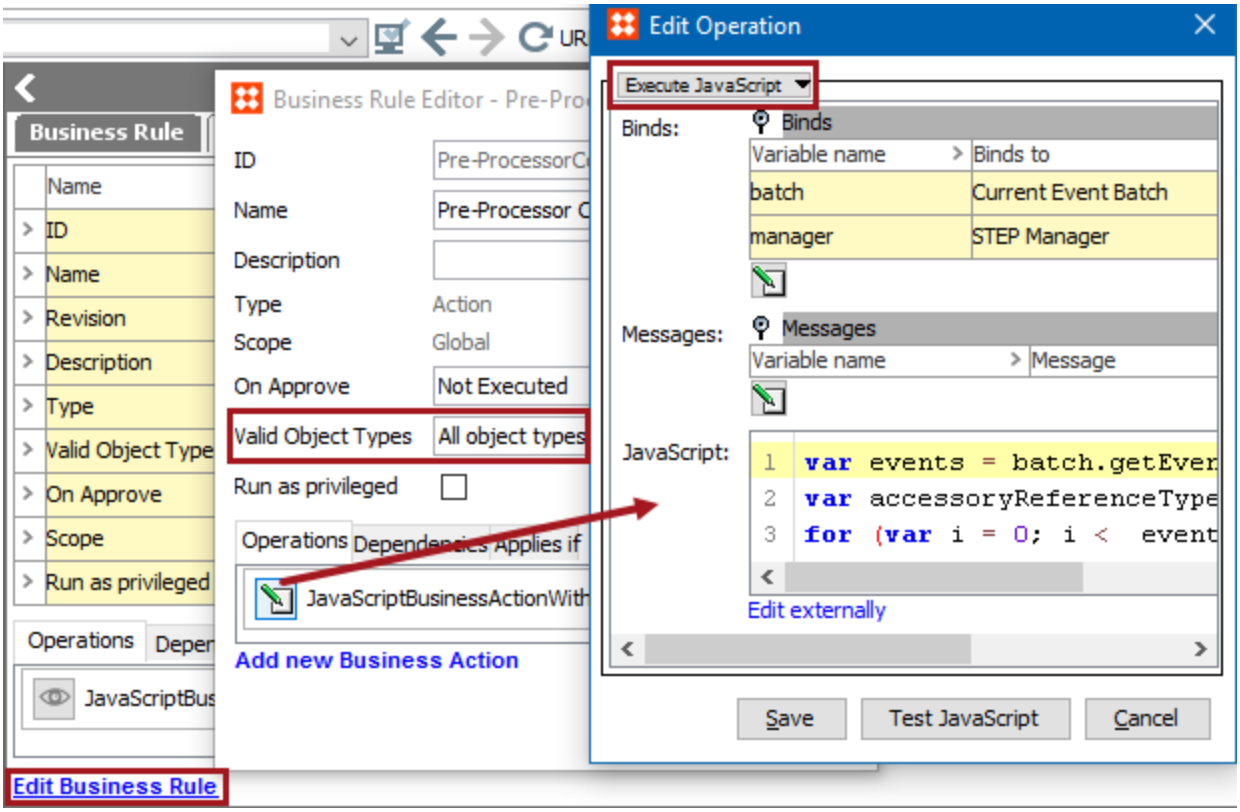
1. A business action - in this example, it will add referenced products to the products that are contained in the message.
2. An event-based OIEP configured with the business action - in this example, it is configured to deliver messages that inform a downstream system about changes to a product.

Business Action Configuration

Create a business action that is valid for 'All Object Types' (or specifically for the object types being handled by the event queue). For more information on creating business rules, see the **Specifying a Business Action Operation** topic.

For a full JavaScript code example, see the online version of this topic.

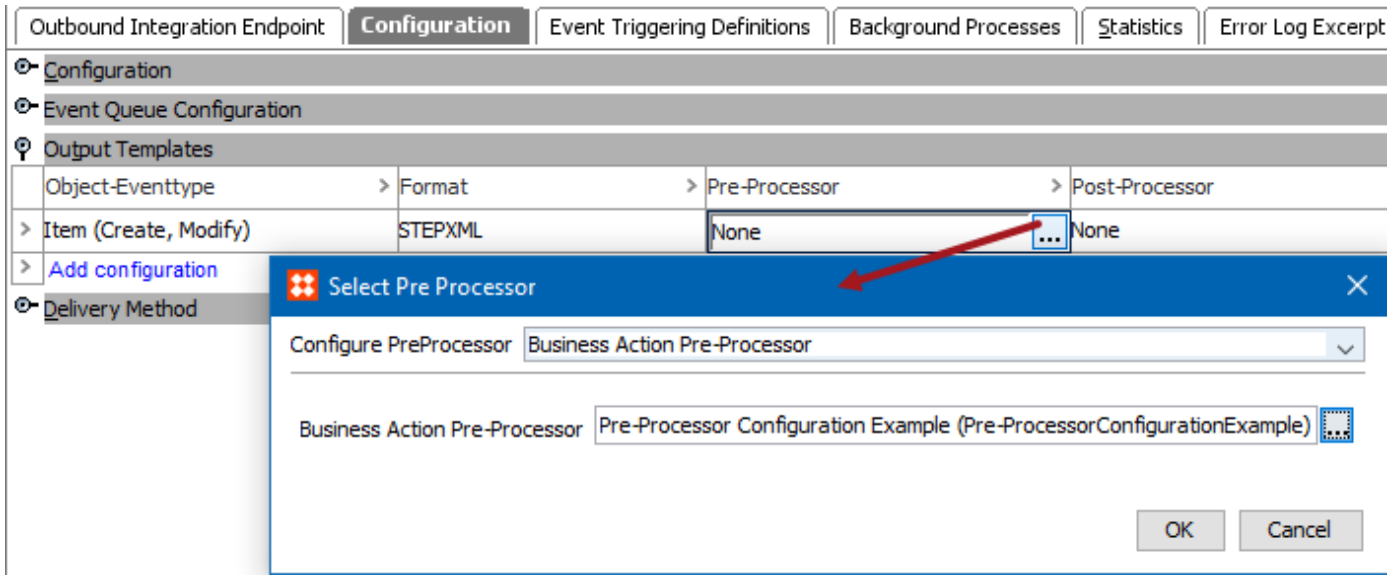
The business rule is configured in System Setup > Global Business Rules as displayed below:



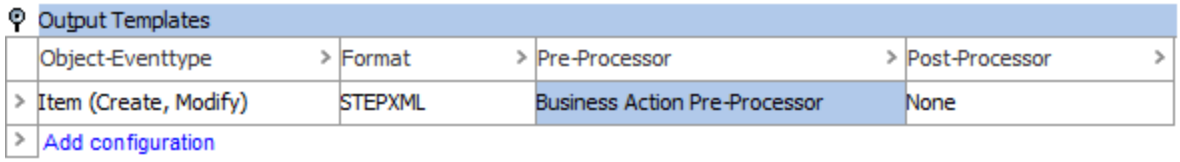
OIEP Configuration

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, see 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 Procesor dialog.
4. In **Configure PreProcessor**, 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.



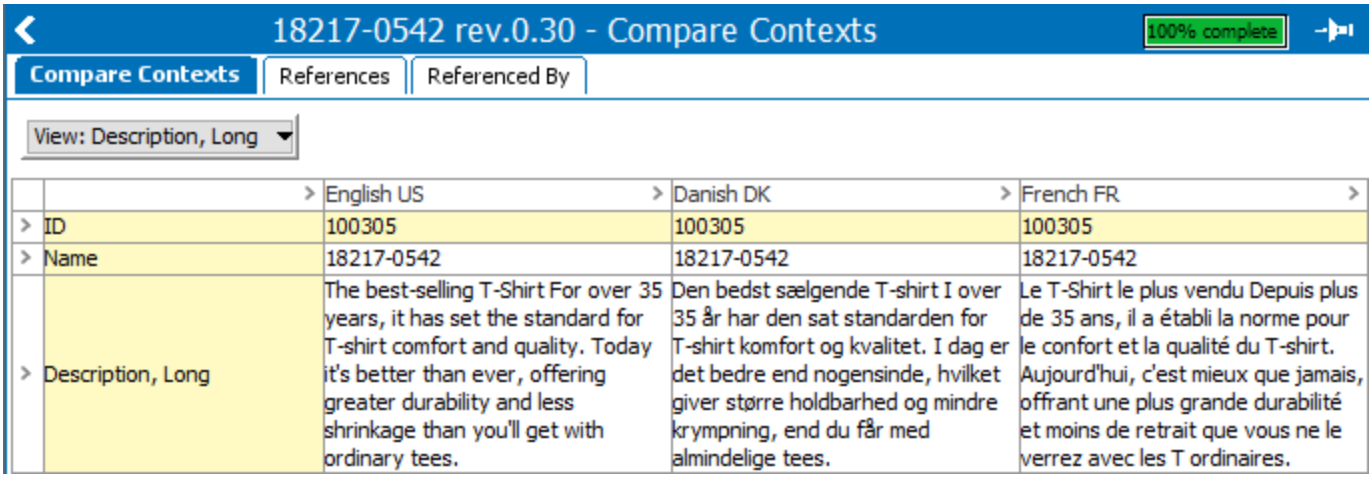
OIEP - Post-Processor - Context Splitter

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.

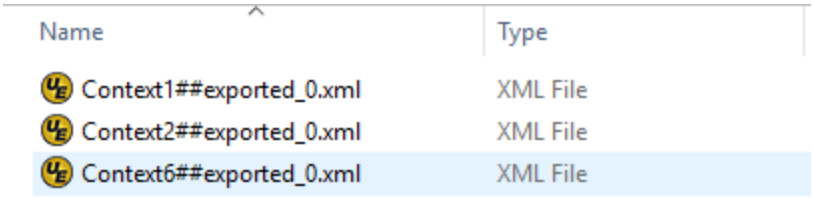
The product has a language-dependent attribute, 'Description, Long', with values defined in three contexts: English US, Danish DK, and French FR.



	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
```

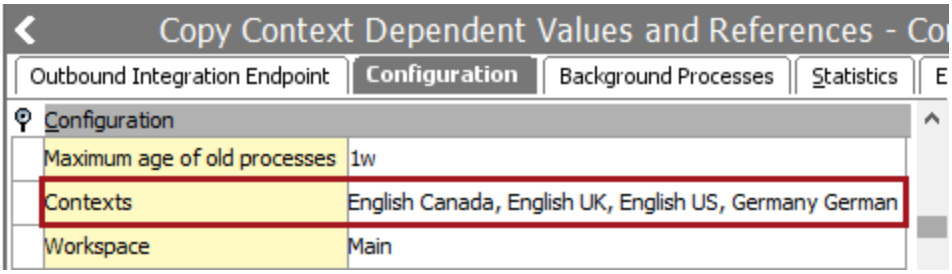
XML Output Example - Standard Cross-Context Export

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, see the **OIEP - Post-Processor - Context Splitter** topic in the **Outbound Integration Endpoints** documentation.

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.

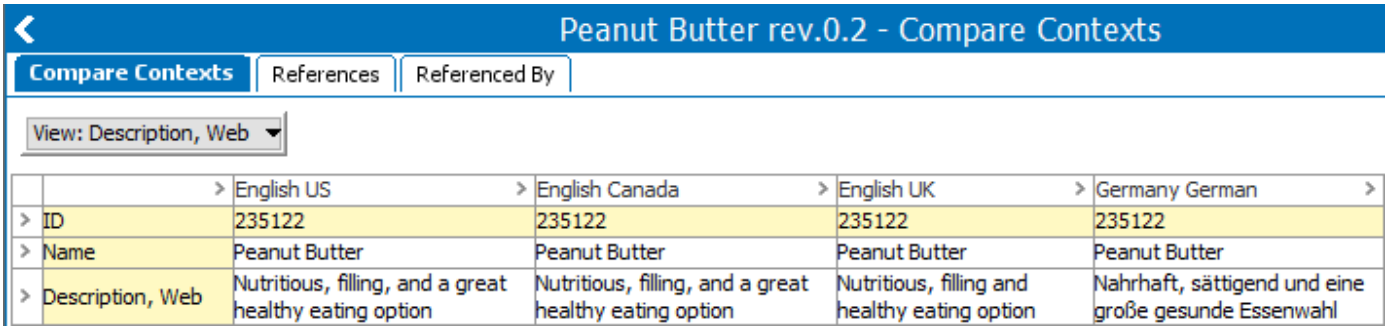
When the <ContextID> XML tag is used in combination with the <QualifierID> XML tag, it is possible for the downstream system to determine the inheritance relationship between a value (or reference) and a context. The <QualifierID> XML tag is useful when a service tries to re-import processed data into 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>
```

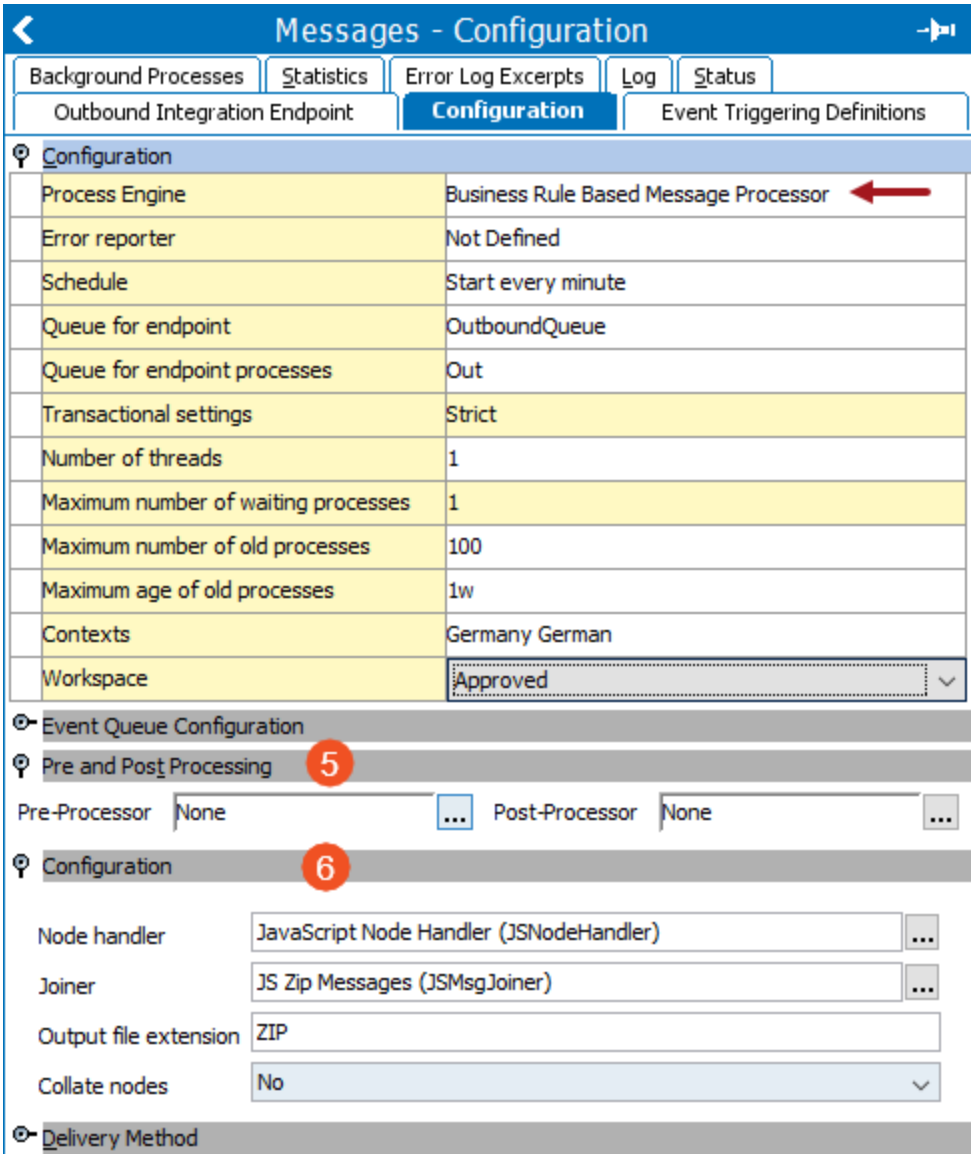
The <ContextID> tag has been added to the export along with the <QualifierID> tag, and each value—including inherited values—is explicitly exported. This means that the 'English Canada' (Context3) values are included in the export, even though it inherits from the master context. If the value is overwritten 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 Methods** topic.

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.

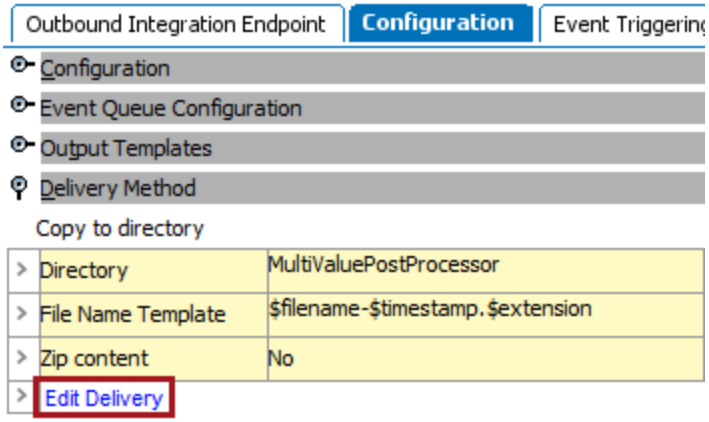
For details, see the **Configure the Pre-processor and Post-processor** section of the following topics:

- **OIEP - Select Objects - Output Templates Flipper**
- **OIEP - Event-Based - Output Templates Flipper**

OIEP - Delivery Method Flipper

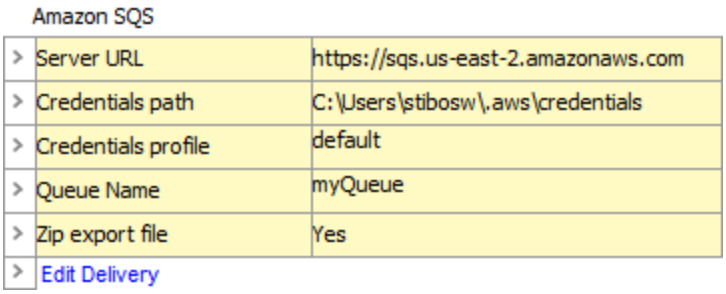
The Delivery Method flipper in an OIEP includes the same parameters for both Event-Based and Select Objects endpoints. The delivery method controls how the output is delivered to an external system.

Open the Delivery Method flipper to display the selected delivery method. As shown below, the parameters vary based on the selected delivery method. Each is described in the **OIEP Delivery Methods** topic.



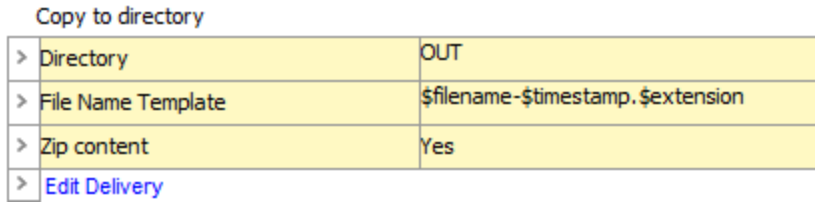
Amazon SQS

For details on this delivery method, see **Amazon SQS Delivery Method**.



Copy to Directory

For details on this delivery method, see **Copy to Directory Delivery Method**.



Deploy

For details on this delivery method, see **Deploy Delivery Method**.

Deploy

Extract to directory	workarea
Copy to file	$tmpdir/zip-$timestamp.$extension
Edit Delivery	

Email

For details on this delivery method, see **Email Delivery Method**.

Email

Email	OutboundData@customer.com
Subject text	Product data export
Body text	Please find enclosed the latest product data delivery
Zip export file	Yes
Edit Delivery	

FTP

For details on this delivery method, see **FTP Delivery Method**.

FTP

Hostname	FTPDeliveryHostServerName
Username	OIEPUser
Password	xxxxxxxx
File Name Template	$filename-$timestamp.$extension
Zip before upload	Yes
Edit Delivery	

IBM Websphere MQ SSL Delivery

For details on this delivery method, see **IBM Websphere MQ SSL Delivery Method**.

IBM Websphere MQ SSL Delivery

Connection URL	
Queue Manager	
Queue Name	
Binary Payload	No
User Name	
Password	
Key Store	
Trust Store	
Cipher Suite	
Additional Parameters	

Edit Delivery

JDBC

For details on this delivery method, see **JDBC Delivery Method**.

JDBC

Driver Location	L:/jdbc-drivers/mysql-connector-java-5.0.8-bin.jar
Driver Class	com.mysql.jdbc.Driver
Database URL	jdbc:mysql://localhost:3306/mydb
Username	root
Password	xxxx
Table Name	stepdata
Key Columns	id,datetime
Delete Key Columns	id

Edit Delivery

JMS

For details on this delivery method, see **JMS Delivery Method**.

JMS Delivery

JMS Connection Factory Class	ActiveMQInitialContextFactory
Connection Factory Name	ConnectionFactory
JMS Provider URL	tcp://ATTCM3S9:61616 =
JMS Queue	JMSQueueForSTEP
Binary Payload	No
User Name	OIEPUser
Password	xxxxxxxxx
Additional Parameters	

Edit Delivery

Dynamic JMS

For details on this delivery method, see the **Dynamic JMS Delivery Method**.

Dynamic JMS Delivery

JMS Connection Factory Name	JMSConnectionFactory
JMS Queue Name	JMS1
JNDI Properties File Name	jndi.properties
Binary Load	No

Edit Delivery

Mongo

For details on this delivery method, see **Mongo Delivery Method**.

Mongo delivery

Server(s)	mongohost1:27017,mongohost1:27018,mongohost1:27019
Raw DB prefix	raw
User name	userauth
Password	xxxxxxxxxxxx
Cluster Enabled Configuration	Yes
Use SSL to encrypt network traffic	No
Key store location	
Trust store location	
Actions	WorkflowReview

Edit Delivery

Oracle AQ

For details on this delivery method, see **Oracle AQ Delivery Method**.

Oracle AQ Delivery

Connection URL	jdbc:oracle:thin:@//server:1521/step
Queue Name	OracleAQRecQueue
Binary Payload	No
User Name	OIEPUser
Password	xxxxxxxx

Edit Delivery

Product Data Syndication

For details on this delivery method, see **Product Data Syndication Delivery Method**.

Product Data Syndication

Server URL	https://StiboPDS/
API User Name	user
API Password	xxxxxxxx

Edit Delivery

REST

For details on this delivery method, see **REST Delivery Method**.

REST

URL	http://www.Customer.com/sqlrest
User name	OIEPUser
Password	xxxxxxxxxx
Zip content	Yes

Edit Delivery

REST Direct

For details on this delivery method, see **REST Direct Delivery Method**.

REST Direct

URL	http://www.CustomerDirect.com/sqlrestdirect
User name	OIEPUser
Password	xxxxxxxxxx
Zip content	Yes
Header	
Footer	

Edit Delivery

SFTP

For details on this delivery method, see **SFTP Delivery Method**.

SFTP

Hostname	SFTPDeliveryHostServerName
Username	OIEPUser
Password	xxxxxxxxx
File Name Template	$filename-$timestamp.$extension
Zip before upload	Yes

Edit Delivery

Wiki

For details on this delivery method, see **Wiki Delivery Method**.

WikiDelivery

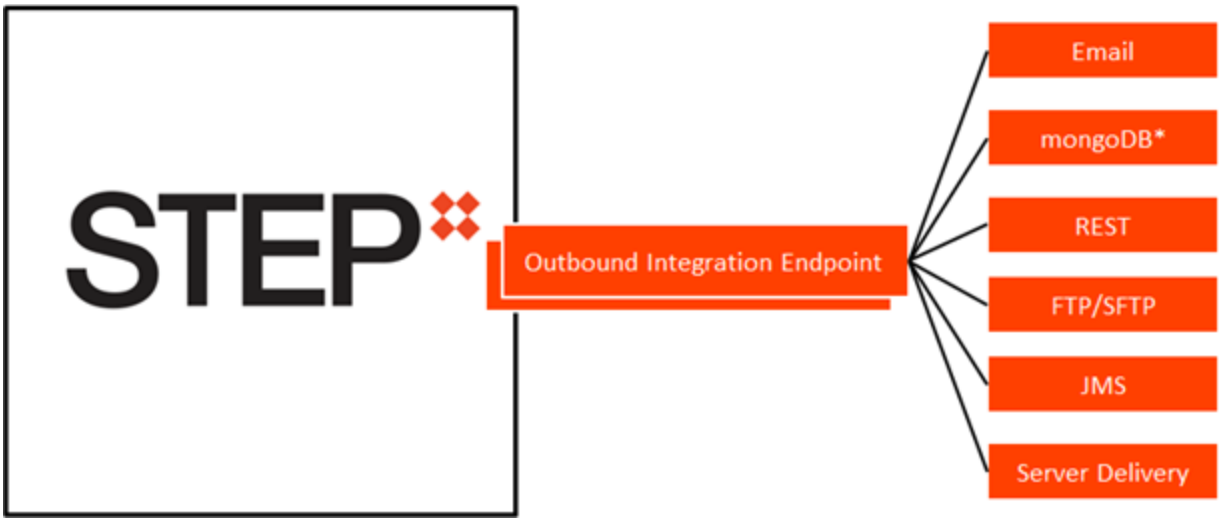
WikiMainSpace	STEP/
WikisName	wikis/xwiki/
WikiPassword	admin
WikiRestSpace	rest/
WikiUrl	http://vp487.vps.ohs.ca/
WikiUsername	admin

Edit Delivery

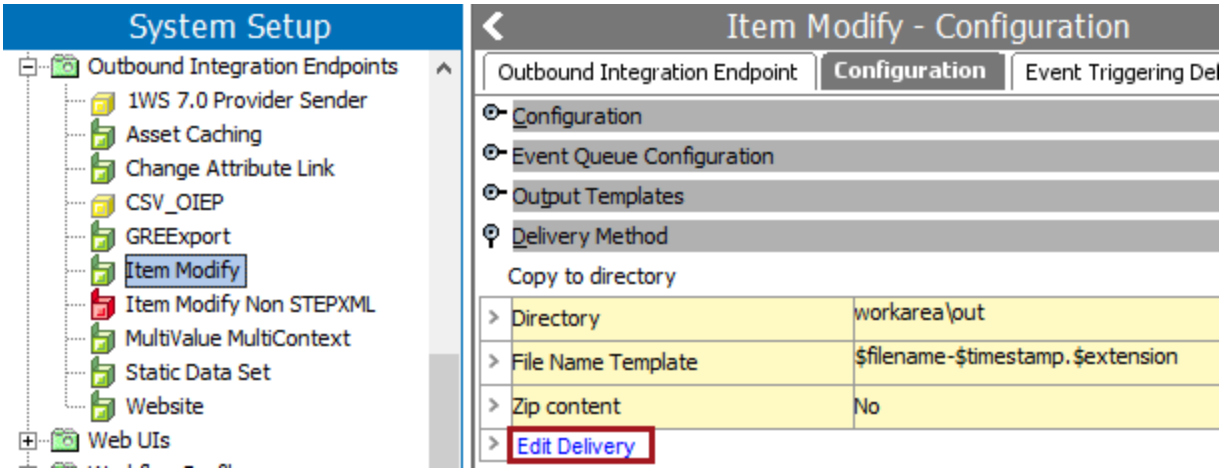
OIEP Delivery Methods

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, see **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.



In an OIEP, the delivery method is set in the editor on the Configuration tab, under the Delivery Method flipper.



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). See **Amazon SQS 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. See **Copy to Directory Delivery Method** documentation.
Deploy	Delivers files to a directory on the application server. Does not allow the delivery to be zipped. See **Deploy Delivery Method** documentation.
Email	Delivers files as an email attachment and allows a zipped delivery. See **Email Delivery Method** documentation.
FTP	Delivers files using file transfer protocol (FTP). See **FTP Delivery Method** documentation.
GDSN Data Pool	The GDSN solution must be implemented fully before the GDSN Data Pool method is available and functional. See **GDSN Provider** documentation.
GDSN Receiver Data Pool	The GDSN solution must be implemented fully before the GDSN Receiver Data Pool method is available and functional. See **GDSN Receiver** documentation.
IBM Websphere MQ SSL	IBM Websphere MQ SSL, uses MQ series Secure Sockets Layer, and enables data exchange across IBM and non-IBM platforms. See **IBM Websphere 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. See **JDBC Delivery Method** documentation.
JMS	The standard JMS delivery options (Apache Active MQ, Websphere MQ, and Oracle AQ) can be used to deliver messages to a JMS receiver without further customizations. See **JMS Delivery Method** documentation.
Dynamic JMS	The standard Dynamic JMS delivery options (Apache Active MQ, Websphere 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. See the **Dynamic JMS 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

Method	Description
	systems. See **Mongo Delivery Method** documentation.
Oracle AQ	Oracle Advanced Queuing (Oracle AQ) enables messages to be exchanged between two systems. See **Oracle AQ Delivery Method** documentation.
Product Data Syndication	Product Data Syndication sends data to the Product Data Syndication (PDS) platform via the default PDS Outbound Integration Endpoint and the API. See **Product Data Syndication Delivery Method** documentation.
REST	The REST Delivery Method delivers a call-back URL to the REST service and does not include actual STEP data. See **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. See **REST Direct Delivery Method** documentation.
SFTP	Delivers a file using the Secure File Transfer Protocol (SFTP). See **SFTP Delivery Method** documentation.
Wiki	Delivers metadata to the metadata wiki platform. See **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.

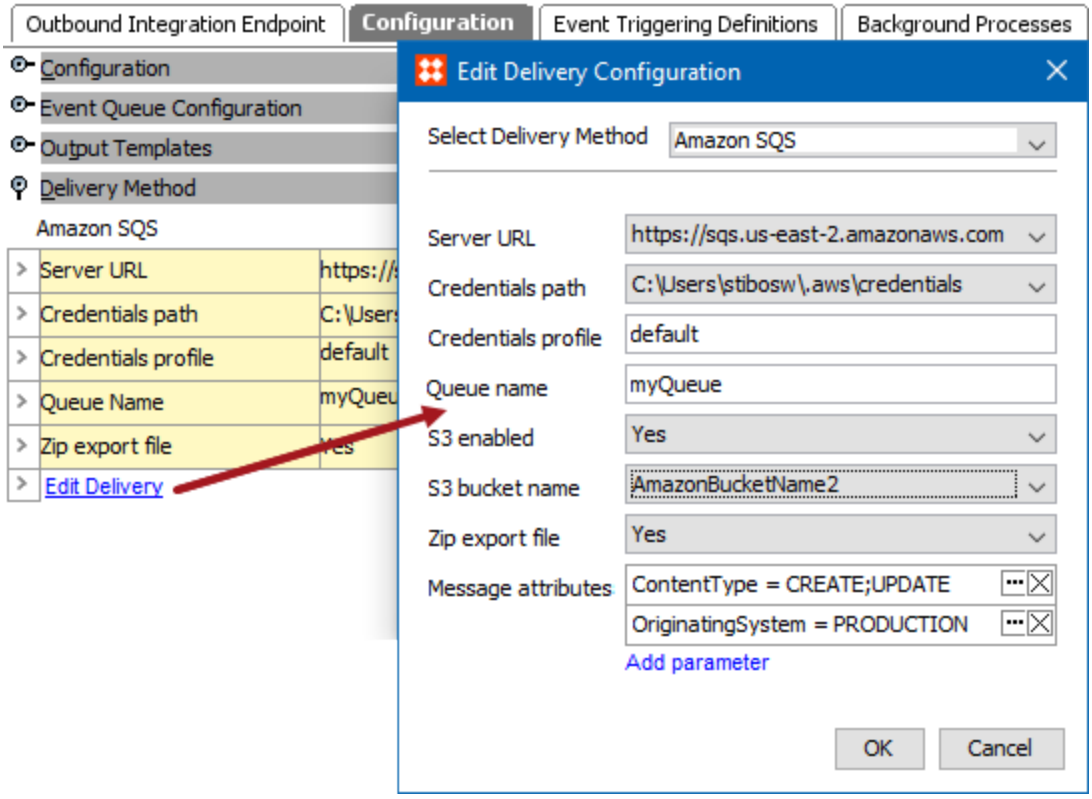
6. If necessary, establish the Amazon S3 storage account and create the required buckets. For details, see Amazon S3 on the web.
7. If message attributes are required, create up to seven (7) string Message Attributes on the Amazon website. The message attribute names in Amazon correspond to the Key parameters 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.

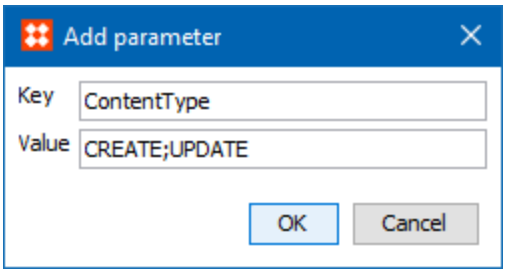


1. For **Select Delivery Method**, choose **Amazon SQS**.
2. In **Server URL**, select the URL that points to the SQS server where the data will be delivered.
3. In **Credentials path**, select the path to the desired SQS credentials file.
4. In **Credentials profile**, enter the profile name included 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 (see 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.

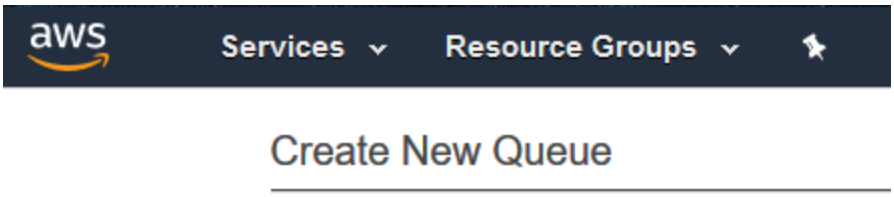


10. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

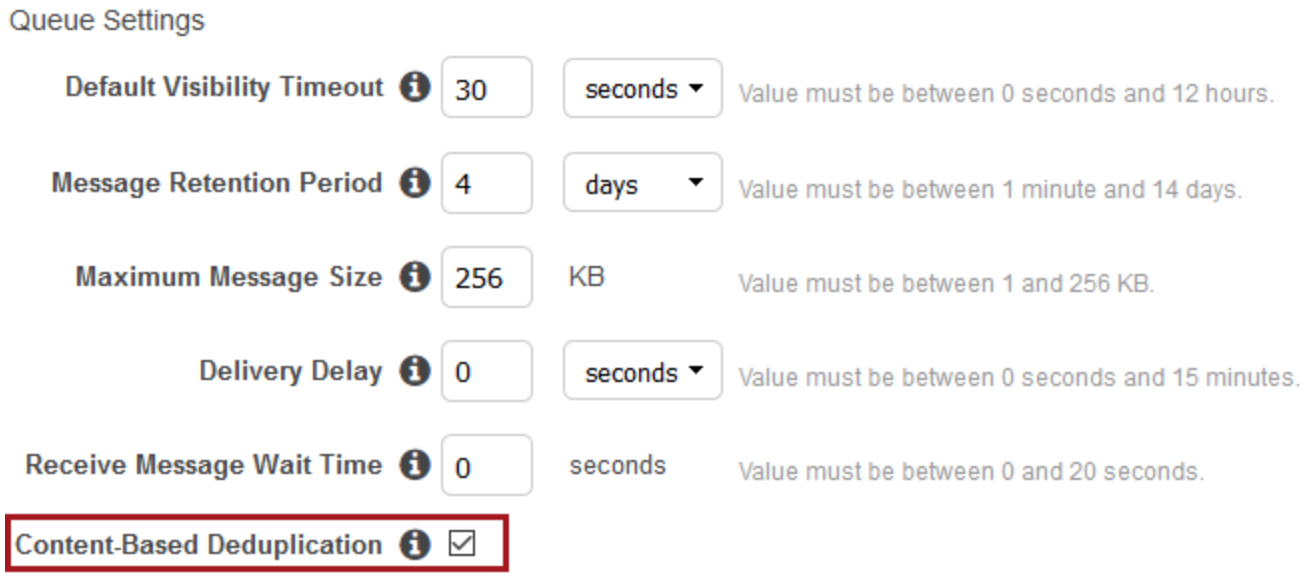
Amazon SQS Queue Configuration

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.



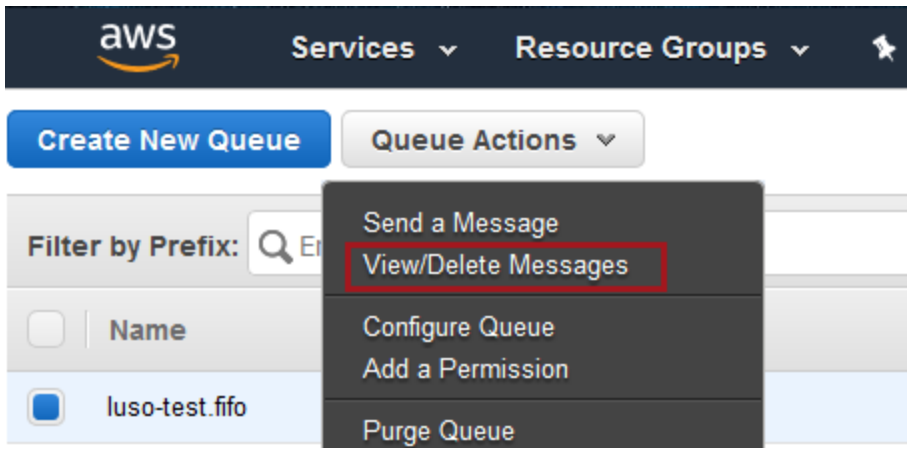
4. For **Queue Name**, type a queue name including the '**.fifo**' extension.
5. For the **type of queue** section, select **FIFO Queue**.
6. Click the **Configure Queue** button at the bottom of the page to display the default Queue Attributes parameters.
7. Check the **Content-Based Deduplication** parameter box to enable the queue.



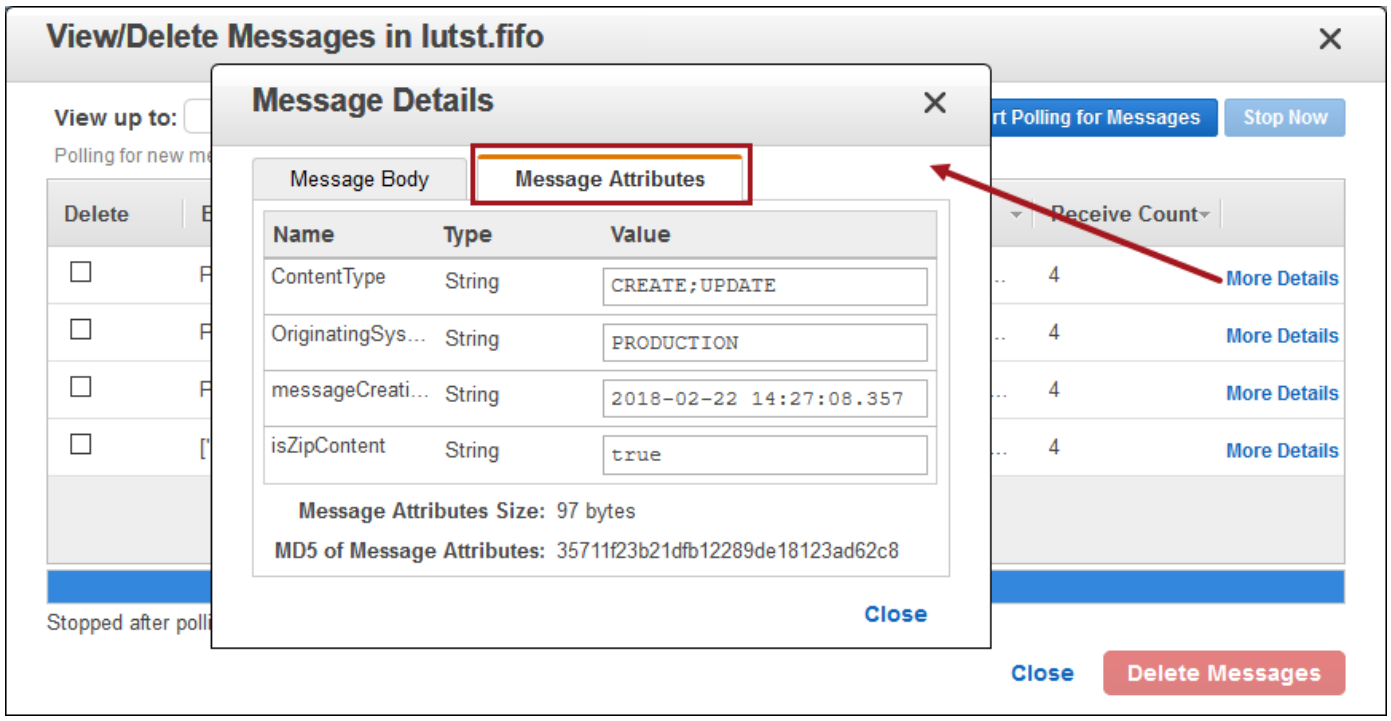
8. Click the **Create Queue** button. The new queue is displayed in the list.

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.

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.

For more information, see the **Deploy Delivery Method** documentation.

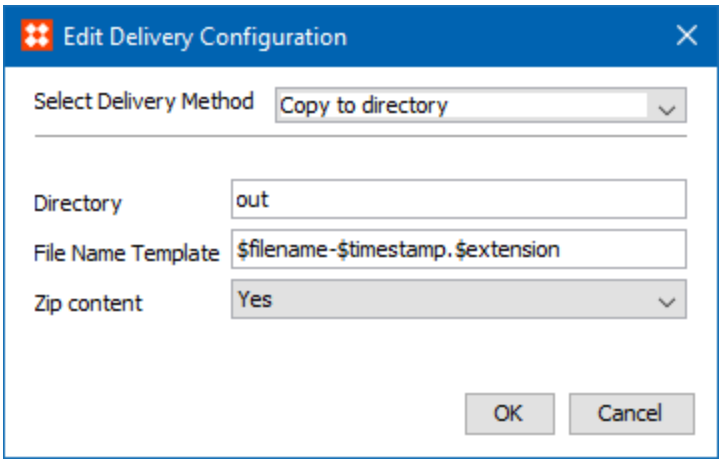
Prerequisites

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**, enter the file naming template to be used for the deliveries. Each element of the template (indicated by the $ character) is replaced with text as defined below. The template allows you to format the exported file name as required by external systems. The default template is: $filename-$timestamp.$extension. Note that the hyphen (-) and the period (.) are actual characters that are used to build the complete name of the exported file.

- **$filename** For event-based OIEPs, this variable is replaced with 'csv' for CSV format, 'excel' for Excel format, and the event ID for STEPXML format exports. For Select Object OIEPs, $filename is replaced with 'exported.' Text can be added before or after the $filename template, and $filename can also be omitted altogether. When exporting multiple contexts, '##' is used as a separator within the filename to distinguish the context name for each output.
- **$timestamp** This variable is replaced with the time in milliseconds between January 1, 1970, and the time when the file is uploaded, which allows for a unique identifier. For example, 1479230247017 is used for an export on 15 NOV 2016.

  Alternatively, the following options can be used to generate a readable date and can include static characters to separate the timestamp elements. Individual elements can be ordered as required, as shown below.

  $timestamp(ddMMyyyy) displays day, month, and year and is output '15112016' for 15 NOV 2016.

  $timestamp(yyyy-MM-dd_HH.mm.ss) displays year, month, day, hour, minutes, and seconds 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 the File Name Template and the extension .ZIP.
   - **No** uses the File Name Template 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.

For more information, see the **Copy to Directory Delivery Method** documentation.

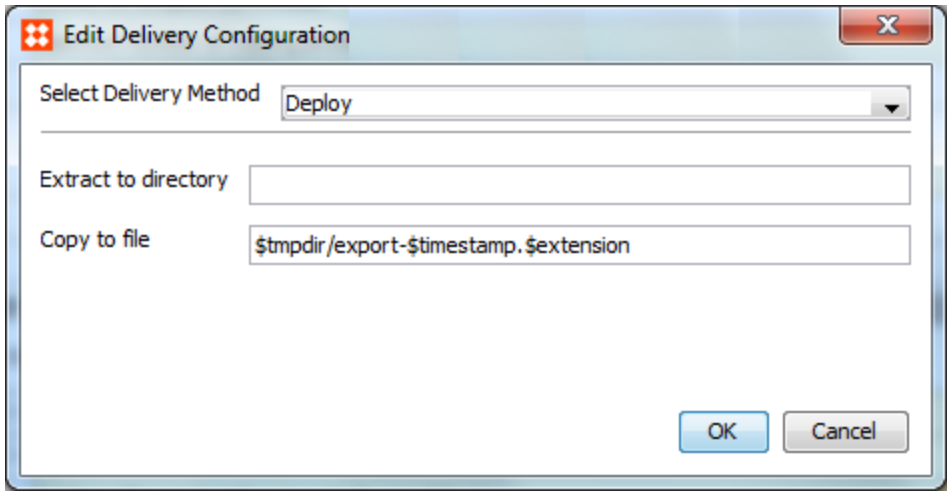
Prerequisites

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 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 milliseconds between January 1, 1970, and the time when the file is uploaded, which allows for a unique identifier. For example, 1479230247017 is used for an export on 15 NOV 2016.

  Alternatively, the following options can be used to generate a readable date and can include static characters to separate the timestamp elements. Individual elements can be ordered as required, as shown below.

  $timestamp(ddMMyyyy) displays day, month, and year and is output '15112016' for 15 NOV 2016.

  $timestamp(yyyy-MM-dd_HH.mm.ss) displays year, month, day, hour, minutes, and seconds 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.

Email Delivery Method

The email delivery option is very straight forward and, like 'Copy to directory', it allows the delivery to be zipped. It is typically used to test the export output, or to deliver a small export to one or more persons as a one-time occurrence.

This delivery method is available in Export Manager and in OIEPs. For information on a parameter, hover over the parameter to display help text.

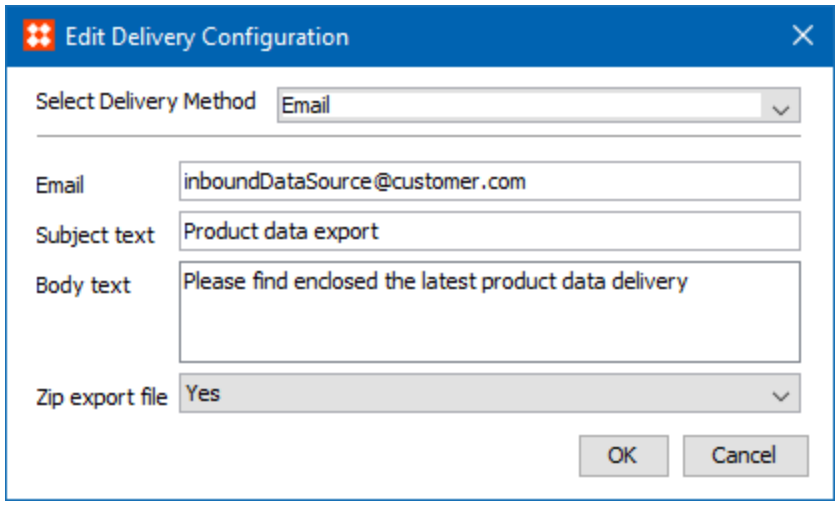
Prerequisites

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.

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.
3. For the **Subject text** parameter, enter the text that will display for the email subject line.
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 exported file.
   - **No** does not zip the exported file.
6. 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 also available in Export Manager.

The preferred method for FTP delivery is the SFTP secure delivery method. For information on the OIEP SFTP delivery method, see **SFTP Delivery Method**.

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]
```

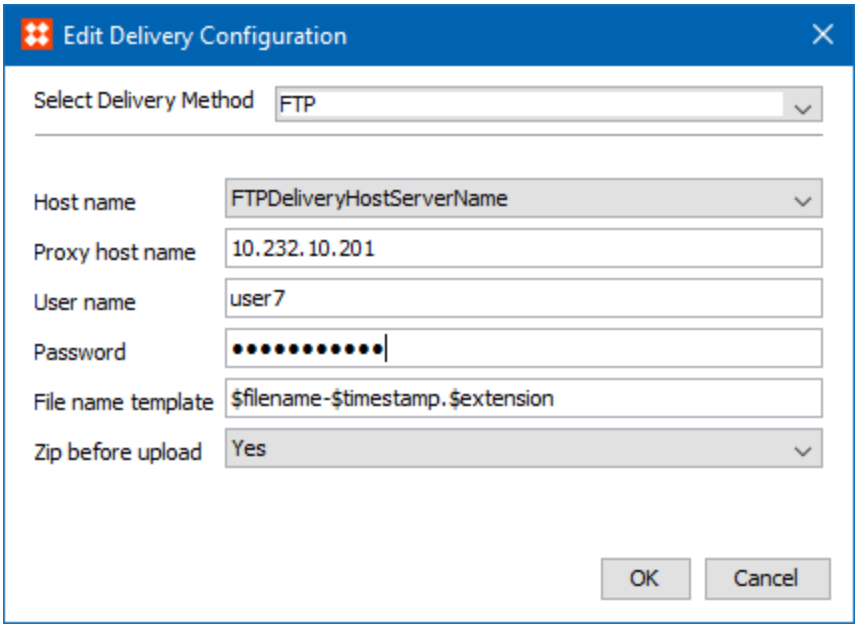
Using the host name shown in the image below, the entry in the properties file would be:

```
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, static text, or use a combination. The default template is $filename-$timestamp.$extension and creates a file name like 'GenericXML-1490813266535.xml' based on the settings of the OIEP. Each variable is described below:
   - **$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. For select object OIEPs, $filename is replaced with 'exported.'
   - **$timestamp** This variable is replaced with the time in milliseconds between January 1, 1970, and the time when the file is uploaded, which allows for a unique identifier. For example, 1479230247017 is used for an export on 15 NOV 2016.

     Alternatively, the following options can be used to generate a readable date and can include static characters to separate the timestamp elements. Individual elements can be ordered as required, as shown below.

     $timestamp(ddMMyyyy) displays day, month, and year and is output '15112016' for 15 NOV 2016.

     $timestamp(yyyy-MM-dd_HH.mm.ss) displays year, month, day, hour, minutes, and seconds 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 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. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

IBM Websphere MQ SSL Delivery Method

For information on connecting to IBM Websphere MQ in a non-SSL way, see **JMS Delivery Method**. This delivery option is only available in OIEPs.

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 **WSMQSSLProviderURL** property. Use the [host]:[port]/[channel] format as defined below:
   - [host] = hostname or IP of the Websphere MQ server
   - [port] = port number for the channel
   - [channel] = name of the channel
2. Prior to configuration, clicking the **Queue Manager** 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 **WSMQSSLQueueManager** property.
3. 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 **WSMQSSLQueue** property.
4. Prior to configuration, clicking the **Key Store** 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 **WSMQSSLKeyStoreLocation** property.

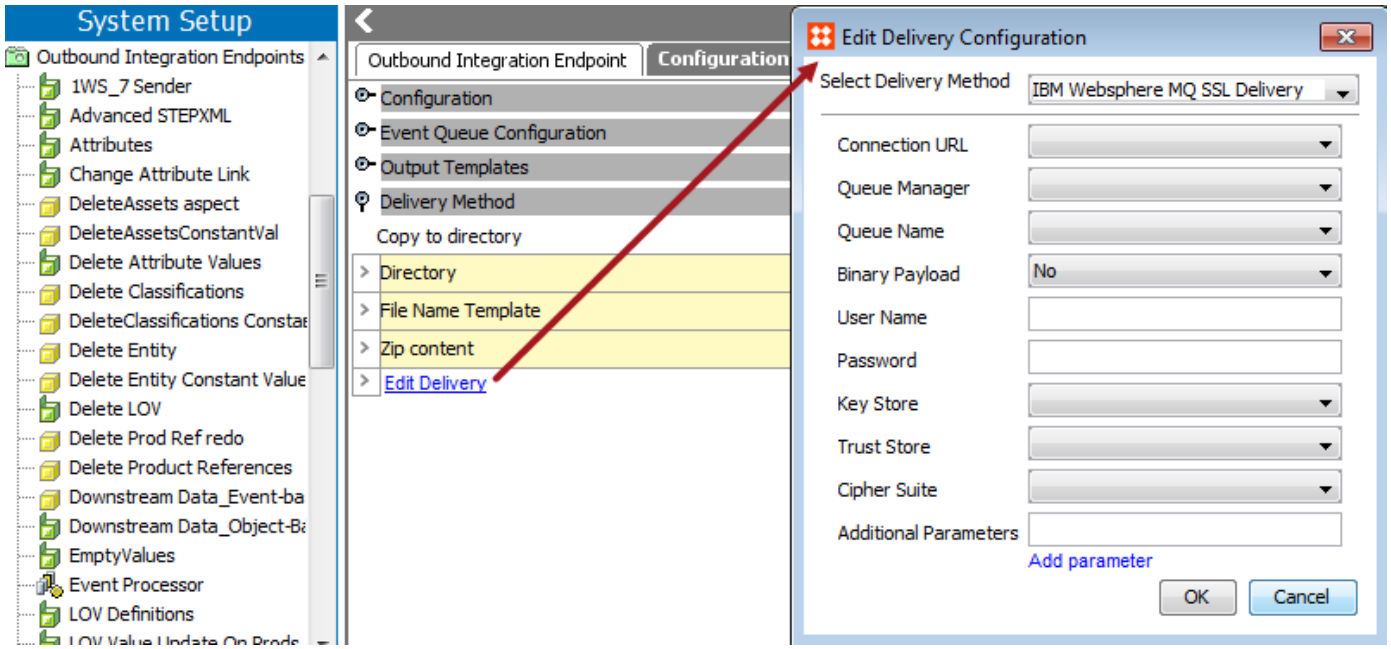
   Use the jks format with the personal certificate for the Queue Manager. To generate this, consult the manual for IBM Websphere MQ. The password for the key store must be configured in sharedconfig.properties file using the **WSMQSSLKeyStorePassword** property.
5. Prior to configuration, clicking the **Trust Store** 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 **WSMQSSLTrustStoreLocation** property.

   This can be the same file as key store. To generate this, consult the manual for IBM Websphere MQ. The password for the trust store must be configured in sharedconfig.properties or config.properties using **WSMQSSLTrustStorePassword**.
6. Prior to configuration, clicking the **Cipher Suite** 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 **WSMQSSLCipher Suite** property. STEP is running on non-IBM JRE, so this must be the same value as configured in the Queue Manager.

Configuration

For information on a parameter, hover over the parameter field to display help text.

1. In the **Select Delivery Method** field, choose **IBM Websphere 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 that will be used to log onto the JMS provider.
7. In **Password**, if required, enter the password that will be used to log onto the JMS provider.
8. In **Key Store**, select the keystore in jks format, with the personal certificate for the Queue Manager.
9. In **Trust Store**, select the trust store with the CA for the Queue Manager.
10. In **Cipher Suite**, set to the same value as SSL CipherSuite in Websphere MQ.
11. If **Additional Parameters** are required, click Add parameter, then enter the Key and Value. For possible keys and values, consult the manual for IBM Websphere MQ.
12. 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 Endpoint (OIEP).

For more information on the Export Manager option, see **JDBC Delivery Method** in the **Export Manager** documentation.

Prerequisites

Important: For complete setup requirements, see 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.

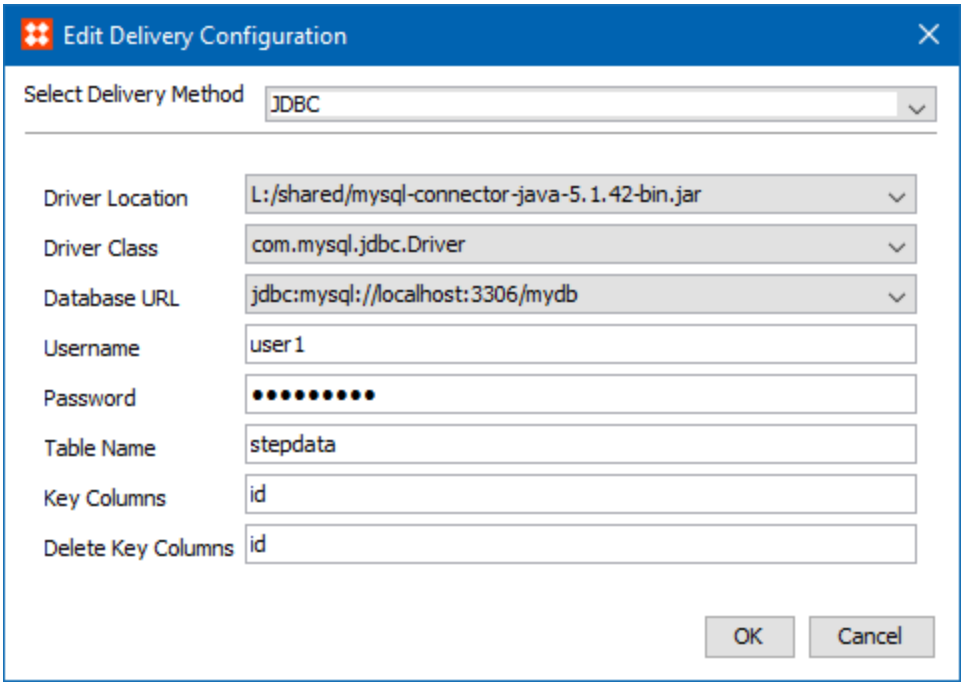
   ```
   JDBCDeliveryPlugin.URL.1 = jdbc:mysql://localhost:3306/mydb
   ```

   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. 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.

Prerequisites

Changes to the properties file, outlined below, are implemented when the server is restarted.

See 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':

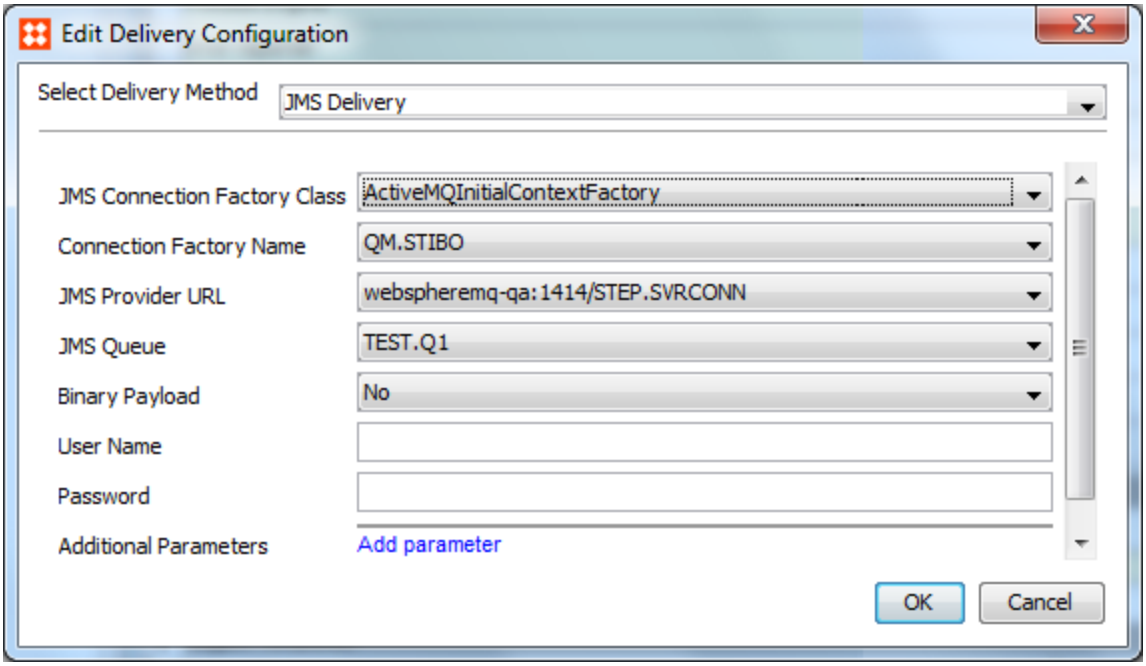
   ```
   JMSProviderURL=qa=http://step-qa,stage=http://step-stage
   ```

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 no longer 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.

Sharedconfig.properties File Examples

Each of these properties has a key/value pair as a value:

```
JMSConnectionFactoryName=FirstFactoryName=MyFactoryName

JMSProviderURL=ProviderURL=https://myproviderurl.com/url

JMSQueue=Queue=Myqueuename
```

Each of the properties can contain more than one value in a comma separated list:

```
JMSConnectionFactoryName=FirstFactoryName=MyFactoryName1,SecondFactoryName=MyFa
ctoryName2

JMSProviderURL=FirsProviderURL=https://myproviderurl.com/url,SecondProviderURL=
https://myproviderurl2.com/url

JMSQueue=Queue1=Firstqueuename,Queue2=Secondqueuename2
```

Azure Message Bus

In the JNDI properties file, add properties to set the context factory and map queues, setting connectionfactory = SBCF as shown below:

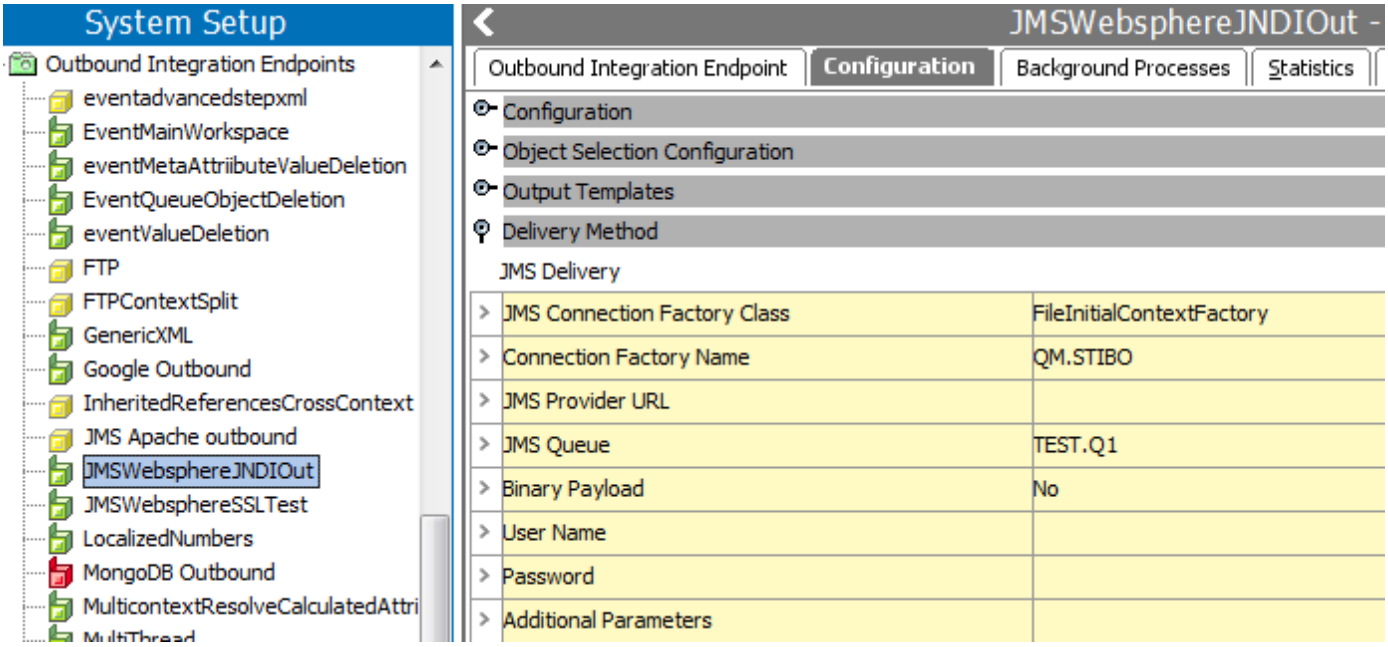
```
connectionfactory.SBCF = failover://
(amqps://khpetest.servicebus.windows.net)?jms.prefetchPolicy.all=10&failover.maxRecon
nectAttempts=20
```

Register queues in JNDI using the form:

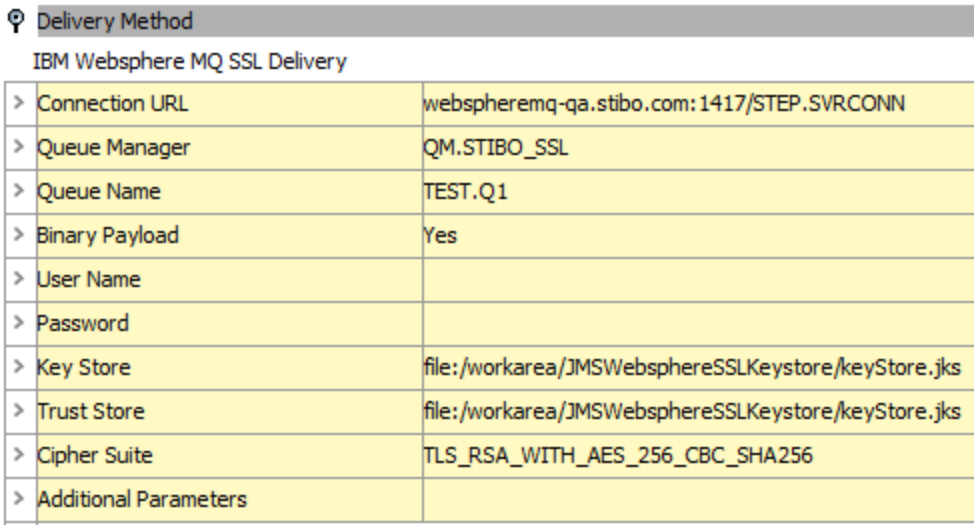
- queue.[jndi_name] = [physical_name]
- topic.[jndi_name] = [physical_name]
- queue.QUEUE = queue1

For more information on the Qpid JMS client library, see https://github.com/MicrosoftDocs/azure-docs/blob/master/articles/service-bus-messaging/service-bus-java-how-to-use-jms-api-amqp.md

FileInitialContextFactory JMS Delivery Configuration Example



JMS Websphere Delivery Using SSL Configuration Example



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	

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.

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.

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.

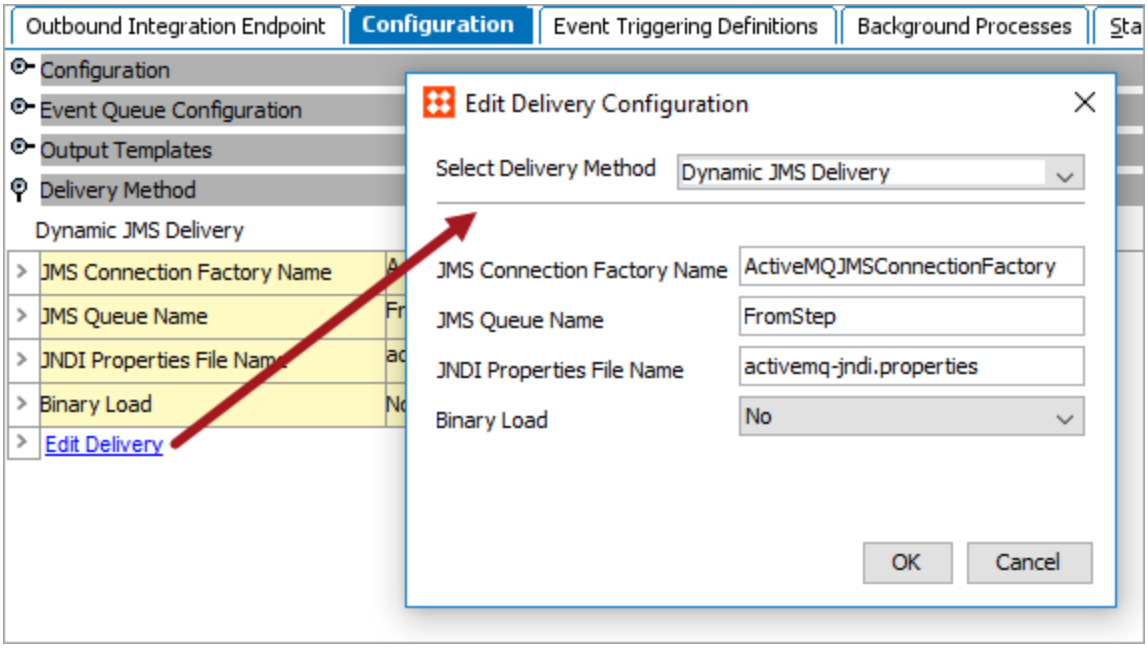
To enable the Dynamic JMS Delivery option, the case-sensitive configuration property 'JMS.Classpath' must be set in sharedconfig.properties on the STEP application server and should point to an existing directory accessible from all application servers. Client libraries and JNDI files (Java Naming and Directory Interface) must be placed in this directory. See 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 connectionfactory and queue in the OIEP delivery method.

Azure Message Bus

In the JNDI properties file, add properties to set the context factory and map queues, setting connectionfactory = SBCF as shown below:

```
connectionfactory.SBCF = failover://
(amqps://khpetest.servicebus.windows.net)?jms.prefetchPolicy.all=10&failover.maxRecon
nectAttempts=20
```

Register queues in JNDI using the form:

- queue.[jndi_name] = [physical_name]
- topic.[jndi_name] = [physical_name]
- queue.QUEUE = queue1

For more information on the Qpid JMS client library, see 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 and the queue have the same name).

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'

Apache RabbitMQ 1.11.1

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=10.0.42.126
ConnectionFactory/RefAddr/3/Type=host
ConnectionFactory/RefAddr/3/Encoding=String
ConnectionFactory/RefAddr/4/Content=pimconsumer
ConnectionFactory/RefAddr/4/Type=virtualHost
ConnectionFactory/RefAddr/4/Encoding=String
# rabbitmqQueue01 Queue
rabbitmqQueue01/ClassName=javax.jms.Queue
rabbitmqQueue01/FactoryName=com.rabbitmq.jms.admin.RMQObjectFactory
rabbitmqQueue01/RefAddr/0/Content=jms/Queue
rabbitmqQueue01/RefAddr/0/Type=name
rabbitmqQueue01/RefAddr/0/Encoding=String
rabbitmqQueue01/RefAddr/1/Content=javax.jms.Queue
rabbitmqQueue01/RefAddr/1/Type=type
rabbitmqQueue01/RefAddr/1/Encoding=String
rabbitmqQueue01/RefAddr/2/Content=com.rabbitmq.jms.admin.RMQObjectFactory
rabbitmqQueue01/RefAddr/2/Type=factory
rabbitmqQueue01/RefAddr/2/Encoding=String
rabbitmqQueue01/RefAddr/3/Content=rabbitmqQueue01
rabbitmqQueue01/RefAddr/3/Type=destinationName
rabbitmqQueue01/RefAddr/3/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
```

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, see https://docs.mongodb.com/manual/ or the **MongoDB Adapter Setup Quick Guides** documentation.

For more information on the Mongo Delivery Method, see the **Mongo Delivery Method Elements** documentation.

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, see 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 first configuration entry must use number 1, and each additional entry must be numbered sequentially (MongoDB.Server.Configuration.1, MongoDB.Server.Configuration.2, MongoDB.Server.Configuration.3, etc.). 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.

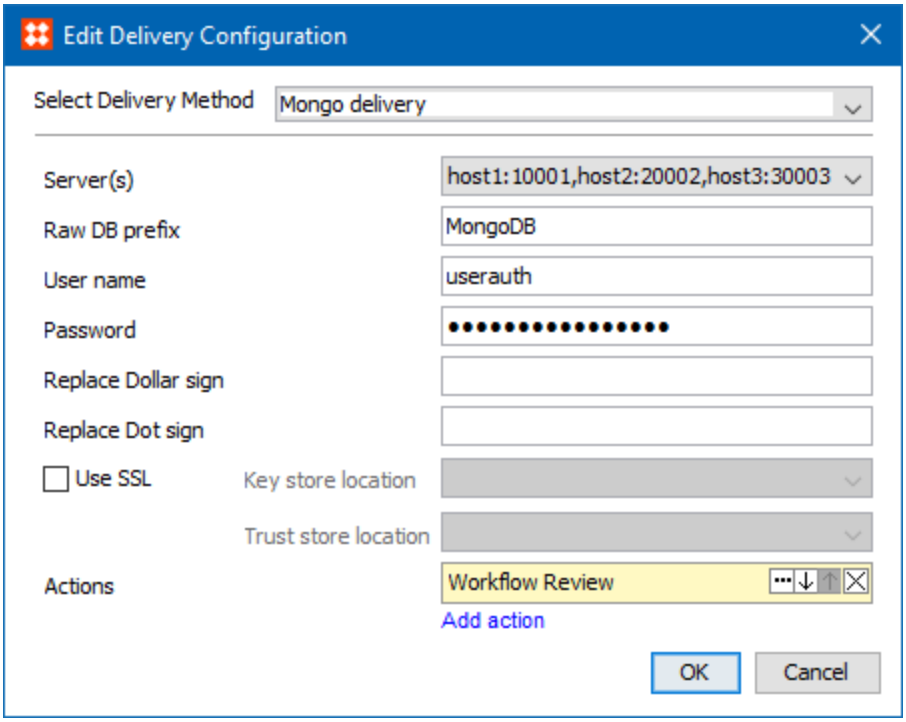
   The password for the key store is configured in sharedconfig.properties on the STEP application server, using the case-sensitive **MongoDB.SSLTrustStorePassword** property. For example:

   ```
   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**.



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:

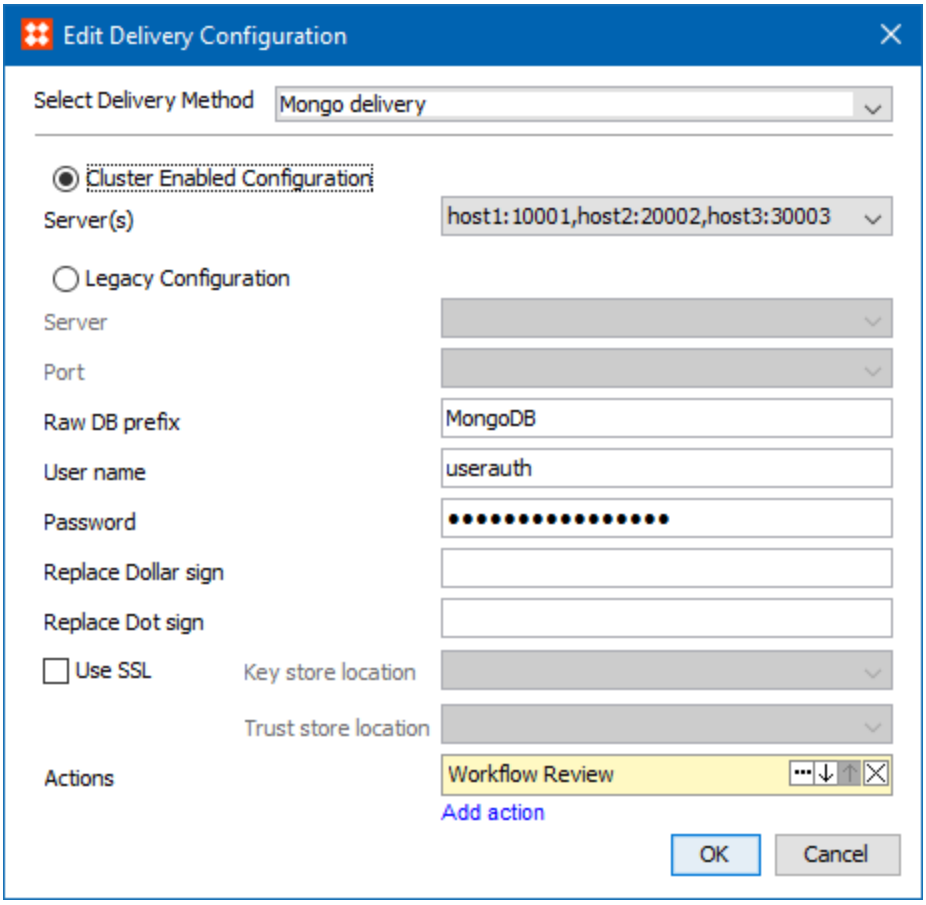
1. The OIEP is event-based, verified by viewing that the Configuration tab includes the Event Queue Configuration flipper. For more information, see the **OIEP - Event-Based - Event Queue Configuration Flipper** topic.
2. In the Event Queue Configuration flipper, the Queue Status parameter is set to Read Events.
3. In the Output Templates flipper, the Object-Eventtype column includes 'Delete' as an event type. For more information, see 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, see the **Event-Based OIEP Triggered by Deleting Products, Classifications, and Assets** section of the **Delete Objects in STEPXML** topic.

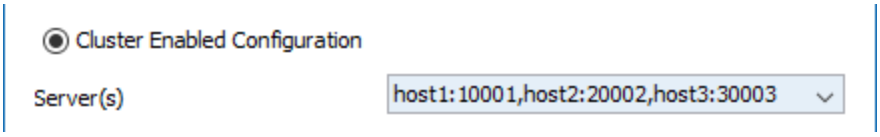
Legacy Configuration

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.

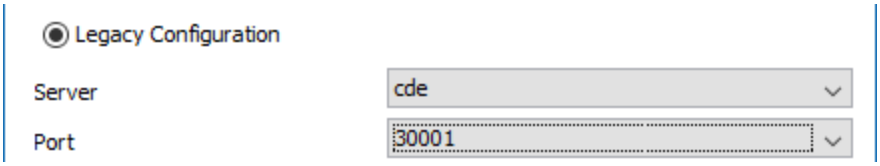
Note: Upgrading the sharedconfig.properties file to modify all configurations using the legacy properties to the cluster enabled property, and then removing the legacy properties completely, results in the radio button options being removed from the dialog, as shown in the image at the top of this topic.



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:

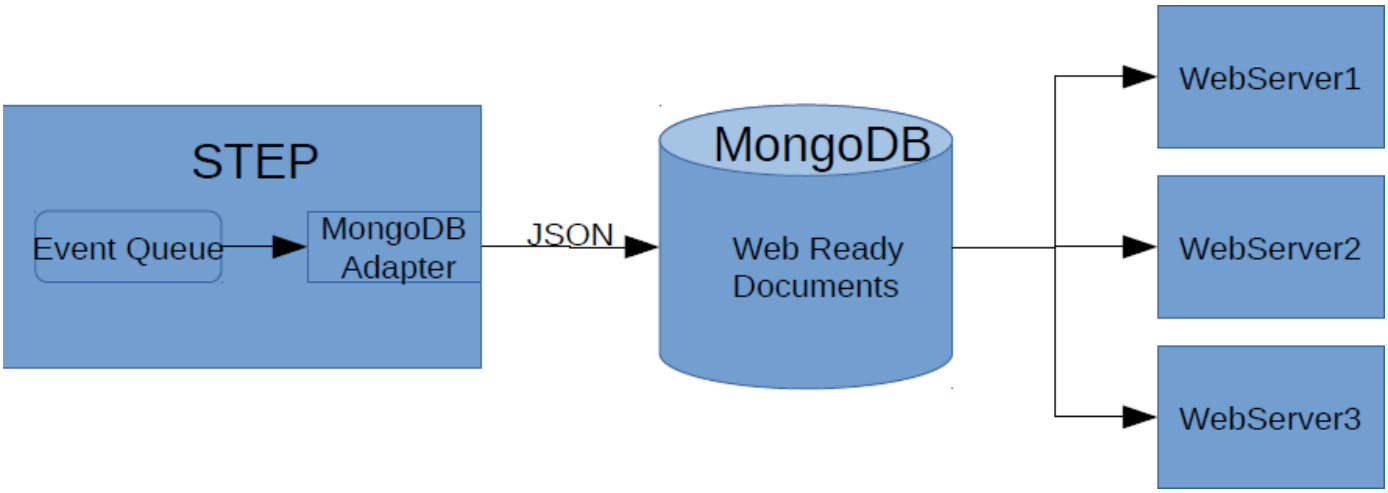
```
MongoDB.Server=1=vps124887.vps.ovh.ca,2=cde,3=rty

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.

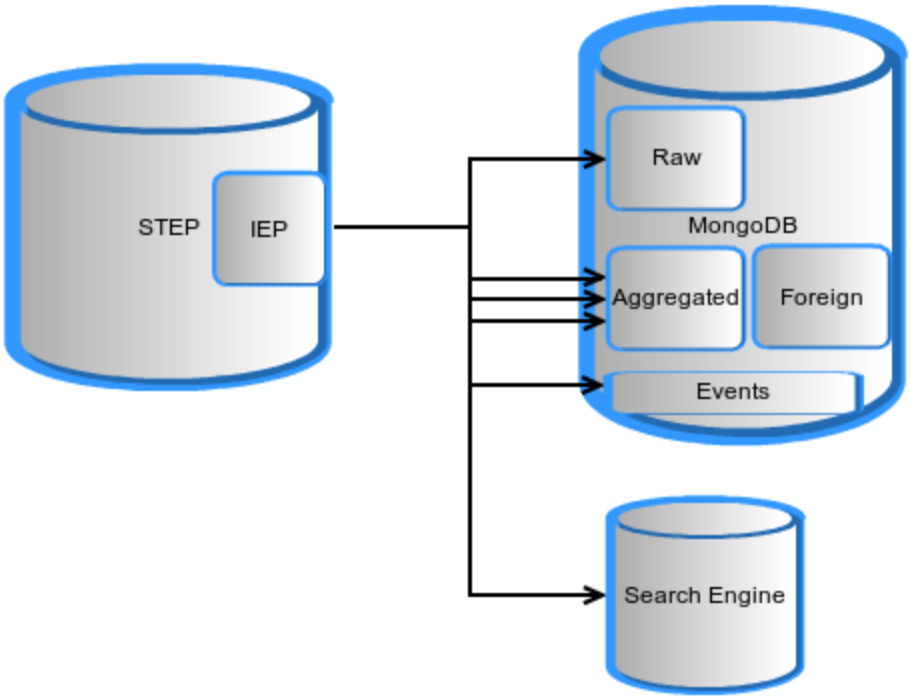
The Mongo adapter converts STEPXML to JSON and keeps a set of 'raw' collections in MongoDB that are in sync with STEP. Additionally, it allows you to add JavaScript triggers to maintain aggregated collections.

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:



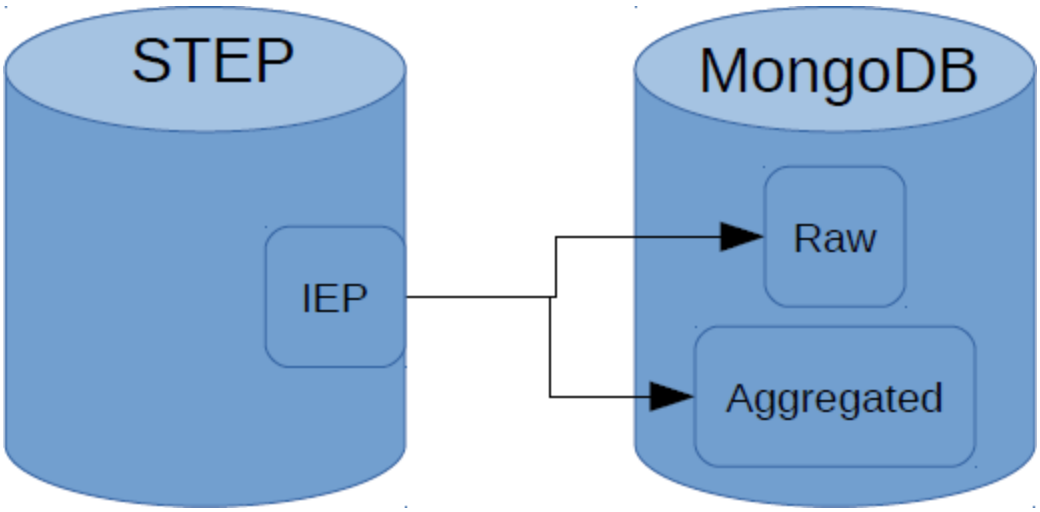
The STEP Mongo Adapter ensures that the raw collections are a one-to-one mirror of data in STEP. The adapter does not embed nodes found through references or child associations. These nodes are stored separately in the raw collections.

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
asset	Exported assets
classification	Exported classifications

Raw Collection	Purpose
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, see the MongoDB documentation on the web.

For an expanded example of the basic conversion using STEP Mongo Adapter, see 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, see the **Applies If Tab** section of the **Editing a Business Rule** 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, see: 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, see 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
   ```

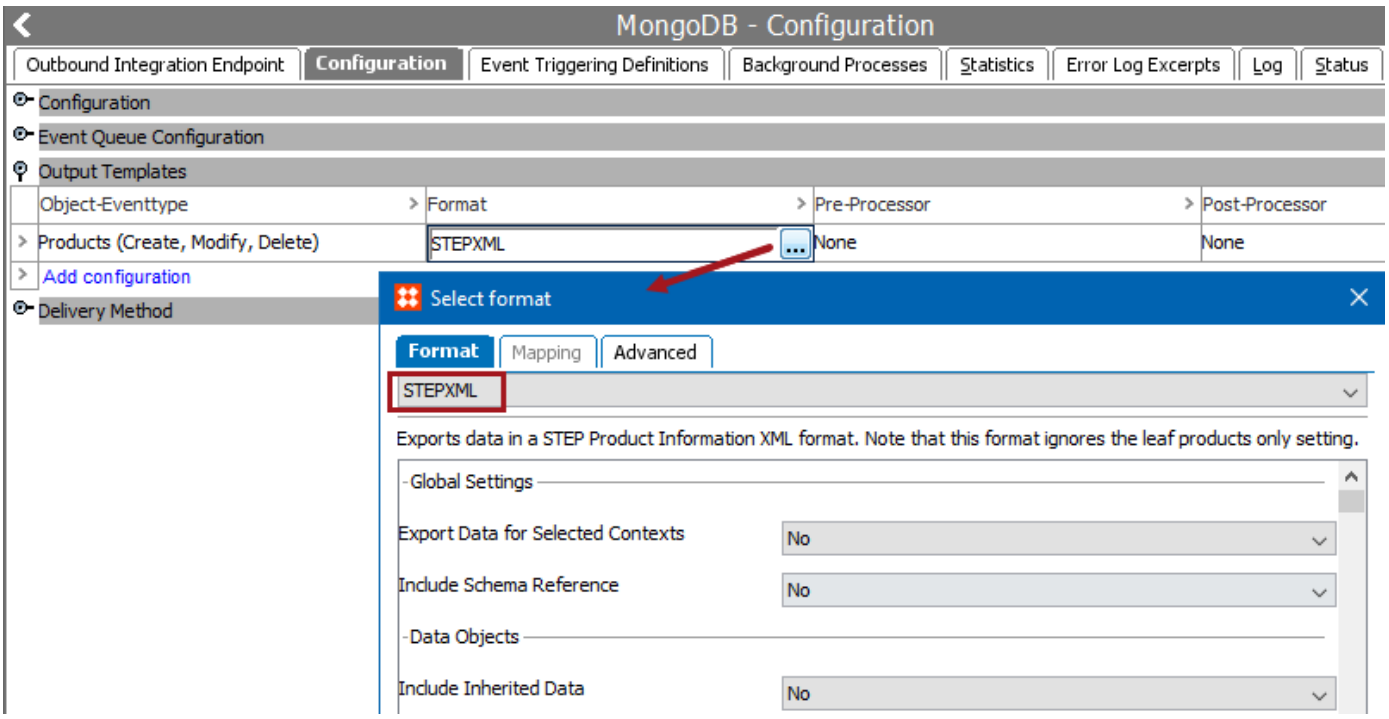
   For example, to configure the adapter to use a MongoDB instance running on either 'Server1' or 'Server2' using port '27712' on both servers, the properties would be:

   ```
   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.

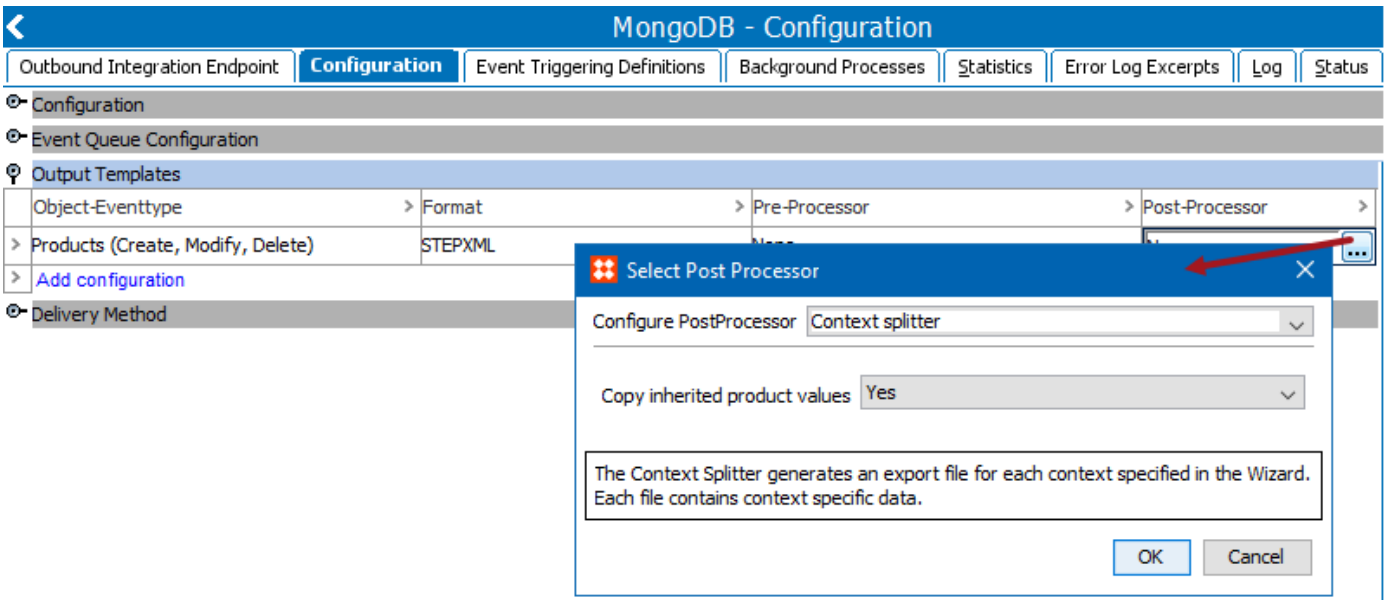
1. In System Setup, create an OIEP, selecting STEP Exporter as the Process Engine. For details on creating a new OIEP, see the **Creating an Event-Based Outbound Integration Endpoint** topic.
2. For the new OIEP, open Configuration > Output Templates > Add configuration > Format > click the ellipsis 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, see 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).

1. Start the Mongo database without authentication.
2. Log onto the database using the Mongo client.
3. 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:

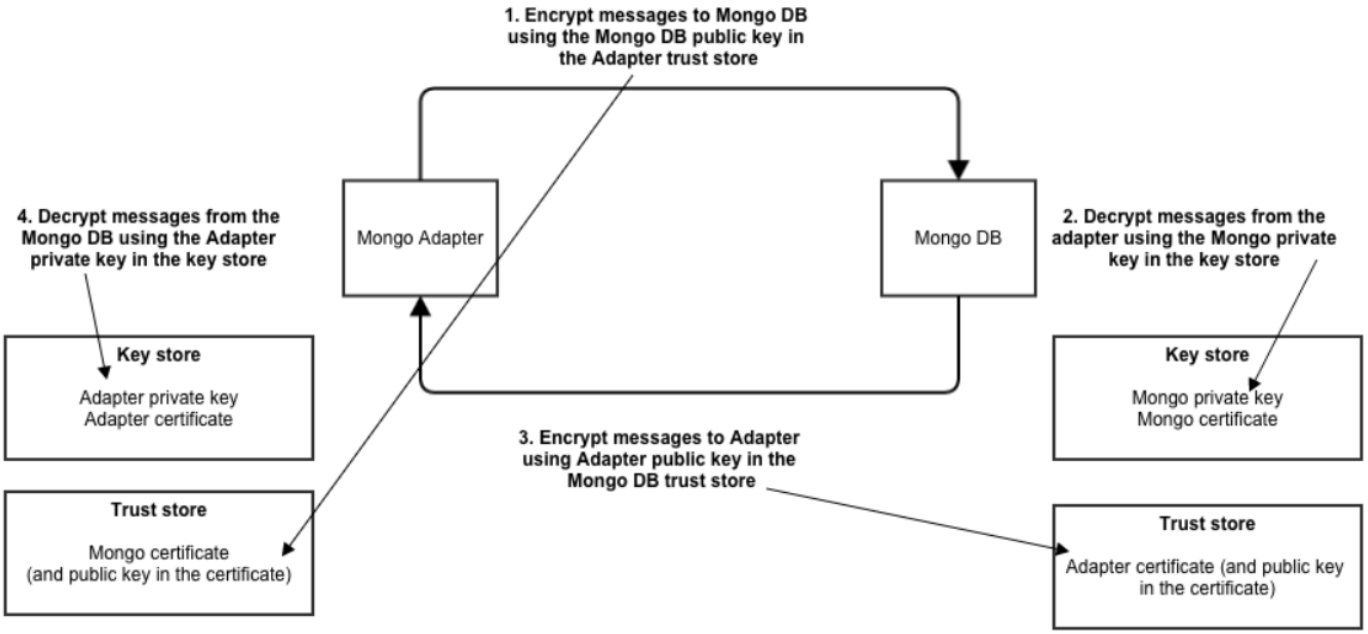
```
mongod --auth --config
```

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://docs.mongodb.com/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 datacontainer, 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 dir, 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": {
    "SpecificatonAttribute": {}
  },
  "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.

Prerequisites

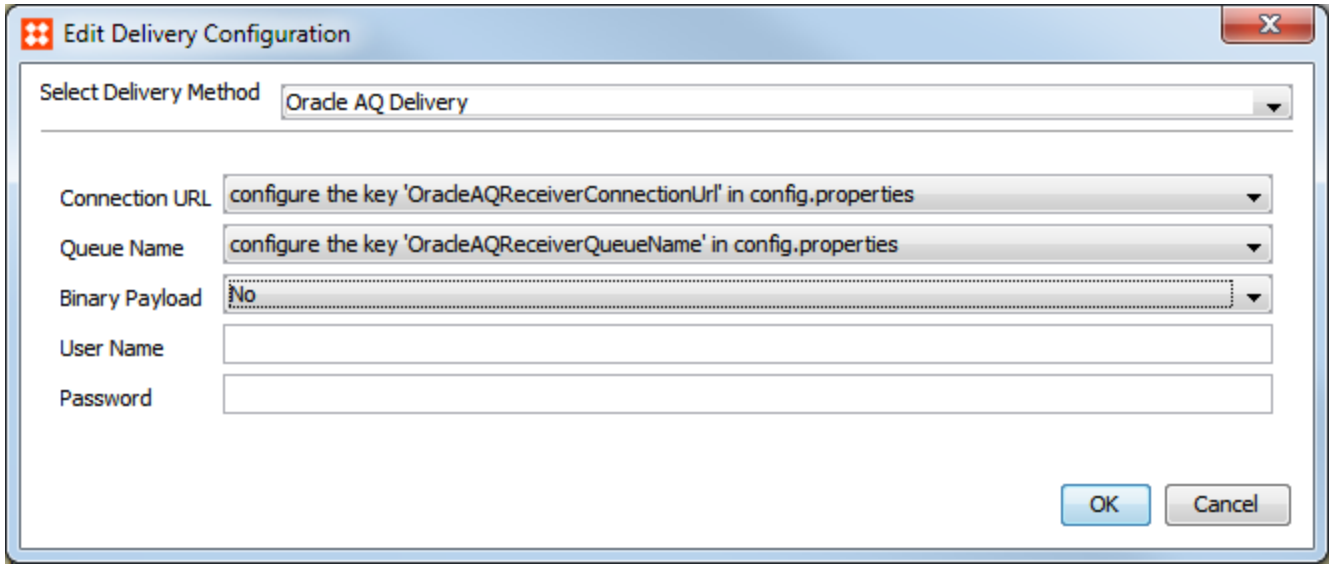
Changes to the properties file, outlined below, are implemented when the server is restarted.

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 Syndication Delivery Method

The Product Data Syndication delivery method is only available in OIEPs. STEP sends data to the Product Data Syndication (PDS) platform via the default PDS Outbound Integration Endpoint and the API.

For additional information on PDS, see the **Product Data Syndication** 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. 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 **PDS.Url** property. For example, a single URL is displayed with the following property setting:

   ```
   PDS.Url=1=https://StiboPDS/
   ```

   If multiple URLs are required, separate the entries with a comma, as shown below:

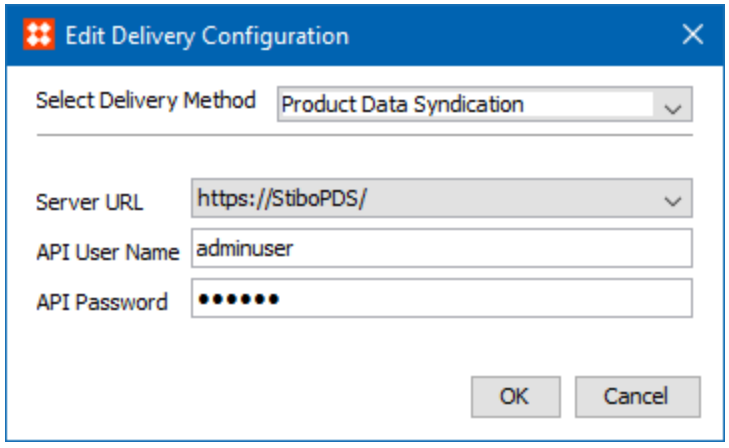
   ```
   PDS.Url=1=https://StiboPDS1/,2=https://StiboPDS2/
   ```

2. Determine the API user name and password. This will be required on the Edit Delivery Configuration dialog.
3. Perform additional configuration required in the OIEP as described in the **PDS Outbound Integration Endpoint Configuration** topic.

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 Syndication**.



3. For the **Server URL** parameter, select a URL from the dropdown.

4. For the **API User Name** parameter, type the user name with access to the API.
5. For the **API Password** parameter, type the password for the user name with access to the API.
6. On the 'Edit Delivery Configuration' dialog, click the **OK** button to save the delivery method.

REST Delivery Method

The REST Delivery Method delivers a call-back URL to the REST service and does not include actual 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.

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':

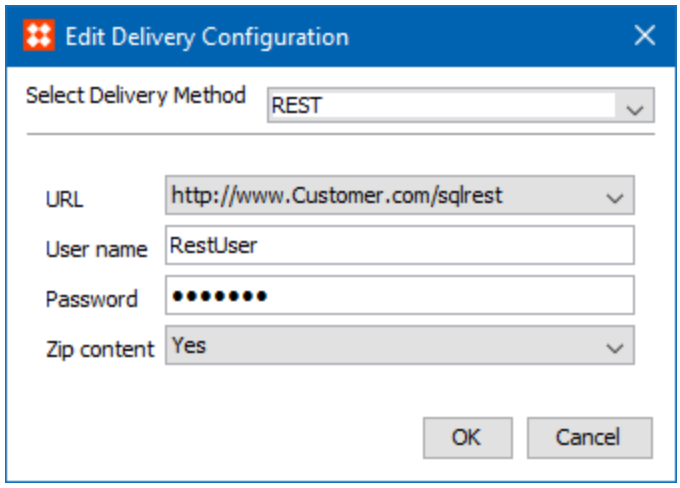
```
RESTDeliveryURL=qa=http://step-qa,stage=http://step-stage
```

Important: For information about how to use the REST API to upload files to REST, click the **STEP API Documentation** button on the STEP 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 **User Name**, enter the user name that will be used to log on to the REST endpoint.
4. In **Password**, enter the password that will be used to log on to the REST endpoint.
5. In **Zip content**, specify whether to zip the contents before upload.
6. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

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.

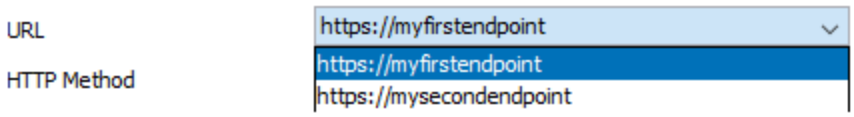
This delivery option is only available in OIEPs and if the delivery fails, the OIEP will be disabled. There is no special resilience handling.

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:

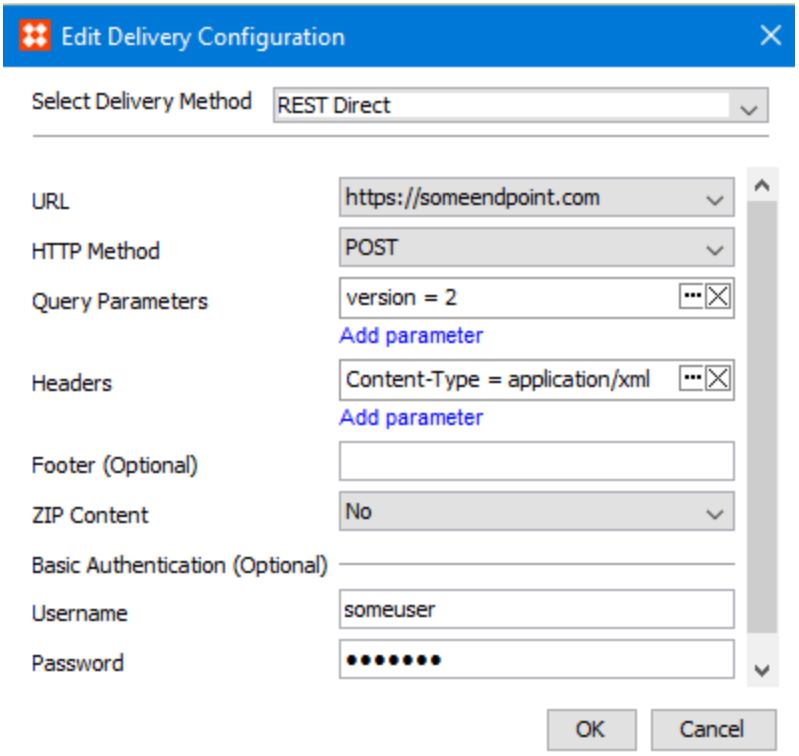
```
RestDirectDeliveryURL = 1=http://myfirstendpoint, 2=http://mysecondendpoint
```



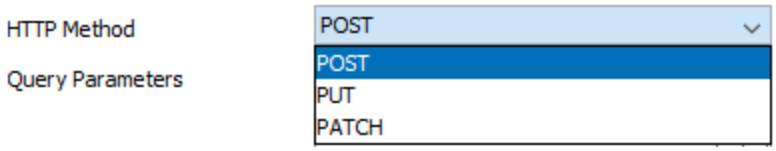
Configuration

For information on a parameter, hover over the parameter field to display help text.

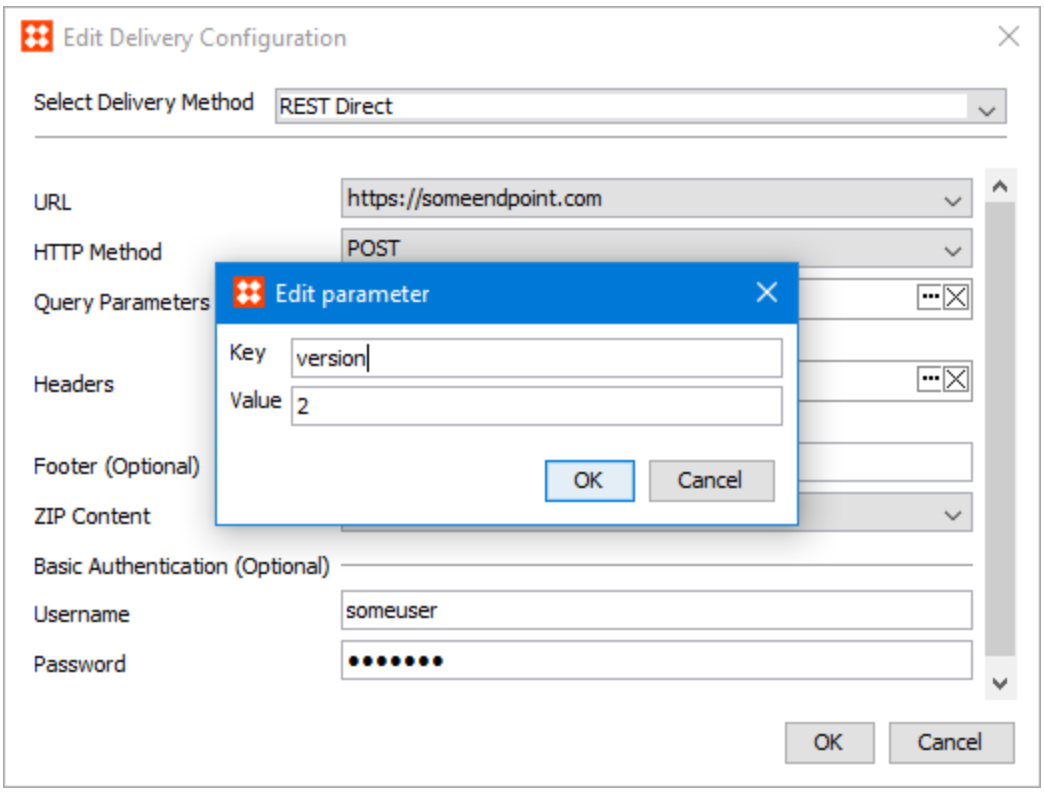
1. For **Select Delivery Method**, choose **REST Direct**.



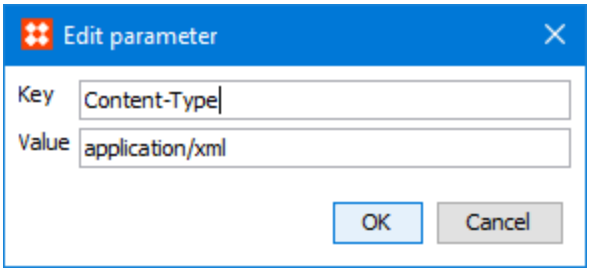
2. In **URL**, select the URL that points to the REST endpoint where the delivered data should be received.
3. In the **HTTP Method** option, use the dropdown to select a POST, PUT, or PATCH method.



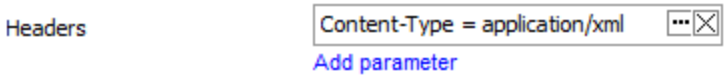
If required, in the **Query Parameters** field add any query parameters that are required for a successful REST transaction. You can click the ellipsis button (...) to add the 'Key' and 'Parameter' for the first parameter, or you can click on the **Add parameter** link to add the first and subsequent parameters.



4. For **Headers**, click the **Add Parameter** link and add a key and a value. This is part of the HTTP network protocol.



5. 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.

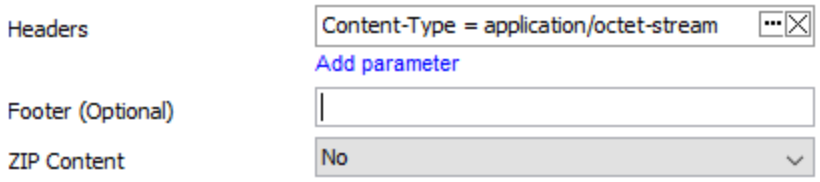


Note: Once a header is displayed, click the ellipsis button (...) to edit it or the X button to remove it.

For more information about headers, see the 'Additional Request Information' section at the end of this topic.

6. 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.



7. In **Zip Content**, specify whether to zip the contents before upload.
8. In **Username**, optionally enter the user name that will be used to log on to the REST endpoint.
9. In **Password**, optionally enter the password that will be used to log on to the REST endpoint.

10. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

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 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.

This delivery method is also available in Export Manager. For information on the OIEP FTP delivery method, see **FTP Delivery Method**.

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].

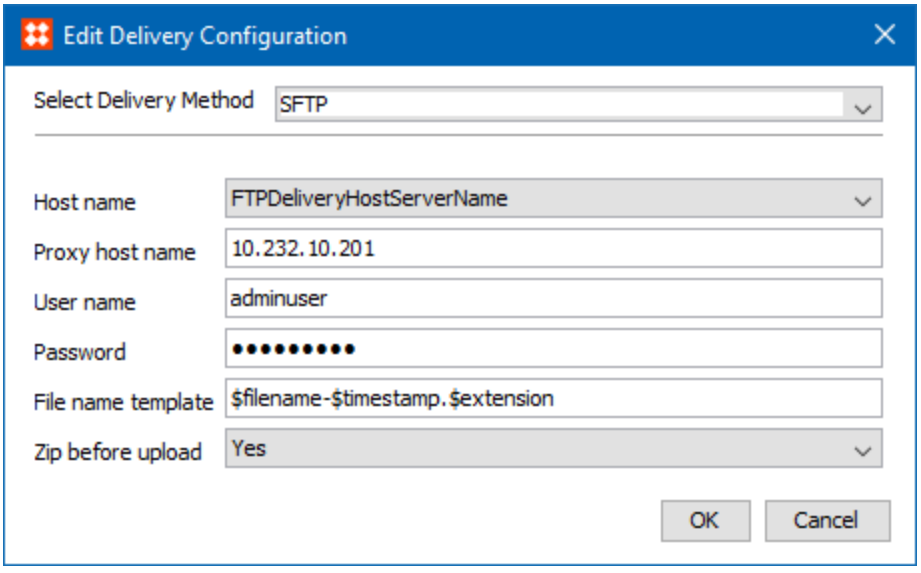
Using the host name shown in the image, the entry in the properties file would be:

```
FTPDeliveryHostName=1=FTPDeliveryHostServerName
```

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.
6. In **File name template**, to create the file name used for the output, enter template variables, static text, or use a combination. The default template is $filename-$timestamp.$extension and creates a file name like 'GenericXML-1490813266535.xml' based on the settings of the OIEP. Each variable is described below:
   - **$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 milliseconds between January 1, 1970, and the time when the file is uploaded, which allows for a unique identifier. For example, 1479230247017 is used for an export on 15 NOV 2016.

     Alternatively, the following options can be used to generate a readable date and can include static characters to separate the timestamp elements. Individual elements can be ordered as required, as shown below.

     $timestamp(ddMMyyyy) displays day, month, and year and is output '15112016' for 15 NOV 2016.

     $timestamp(yyyy-MM-dd_HH.mm.ss) displays year, month, day, hour, minutes, and seconds 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 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. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

Wiki Delivery Method

The XWiki 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.

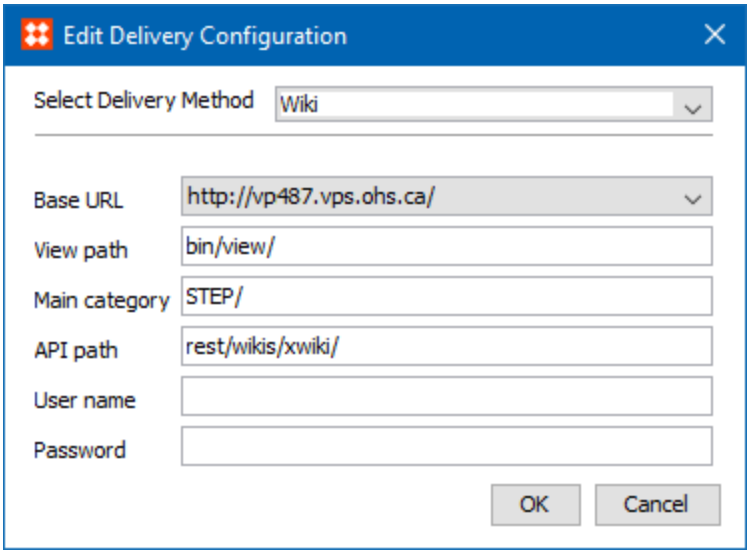
Configuration

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 XWiki 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 **XWiki**.



2. In **Base URL**, select the URL that points to the wiki that will be used for this STEP system. See 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.

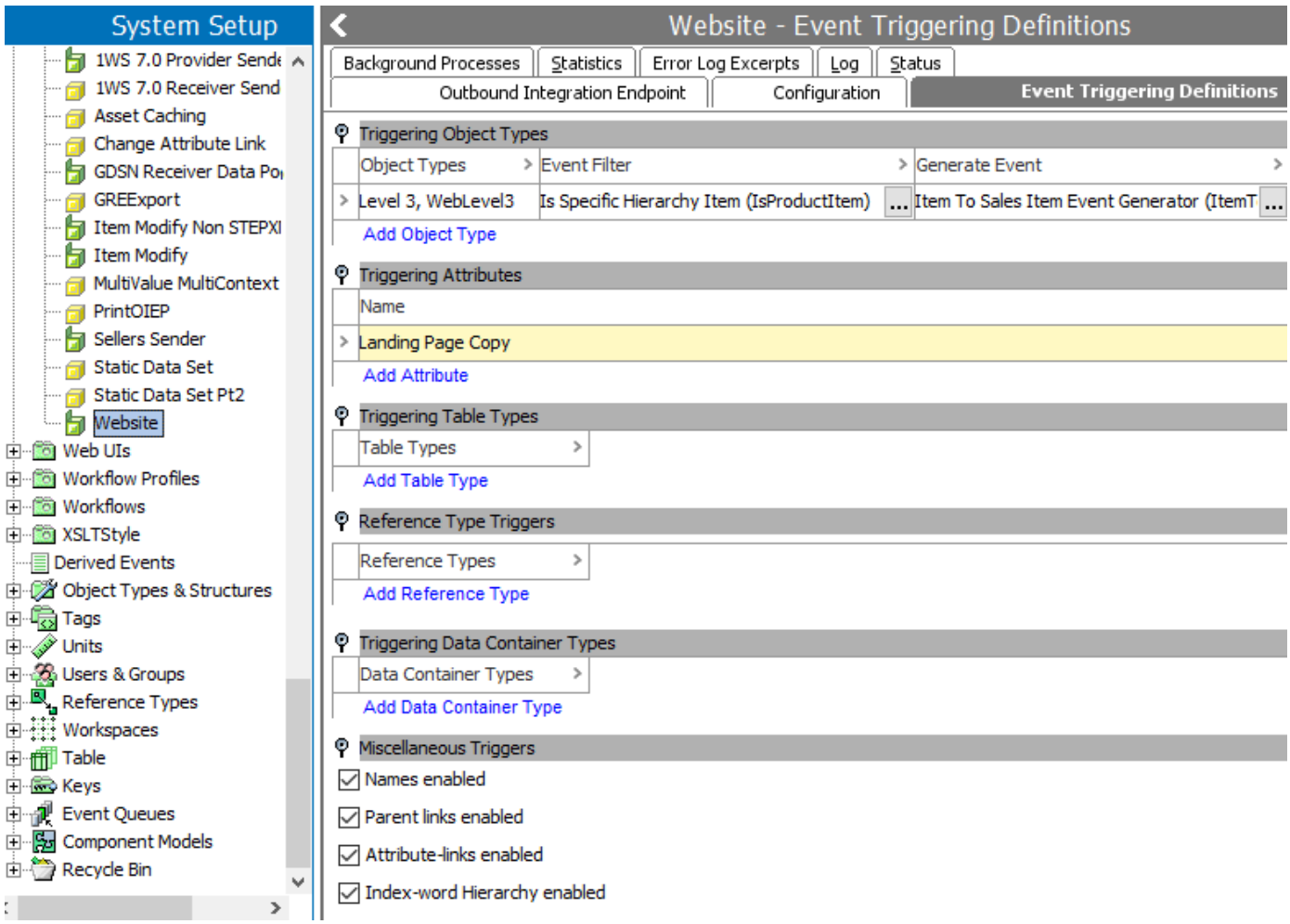
- Workspace revisable objects, such as a product, generate an event when data changes are approved.
- Global revisable objects, such as an attribute or an entity storing customer data, generate an event as soon as the data changes because approval is not available or required.

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.

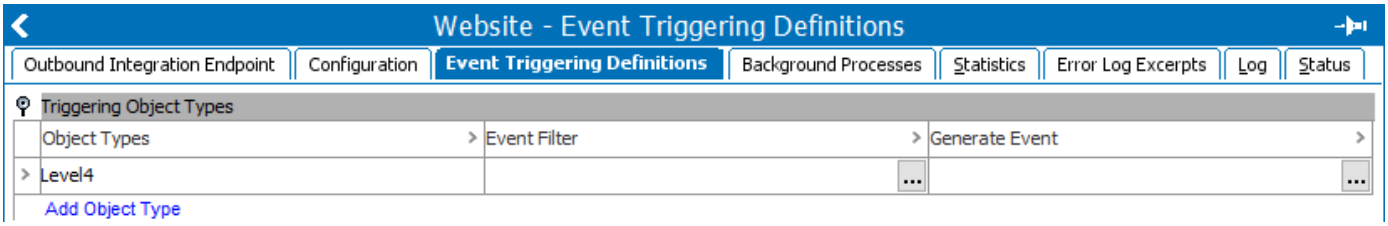
For details about event generation, see the **Events** section of the **System Setup / Super User Guide** documentation.

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, see 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, see **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 when setting up:

- 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

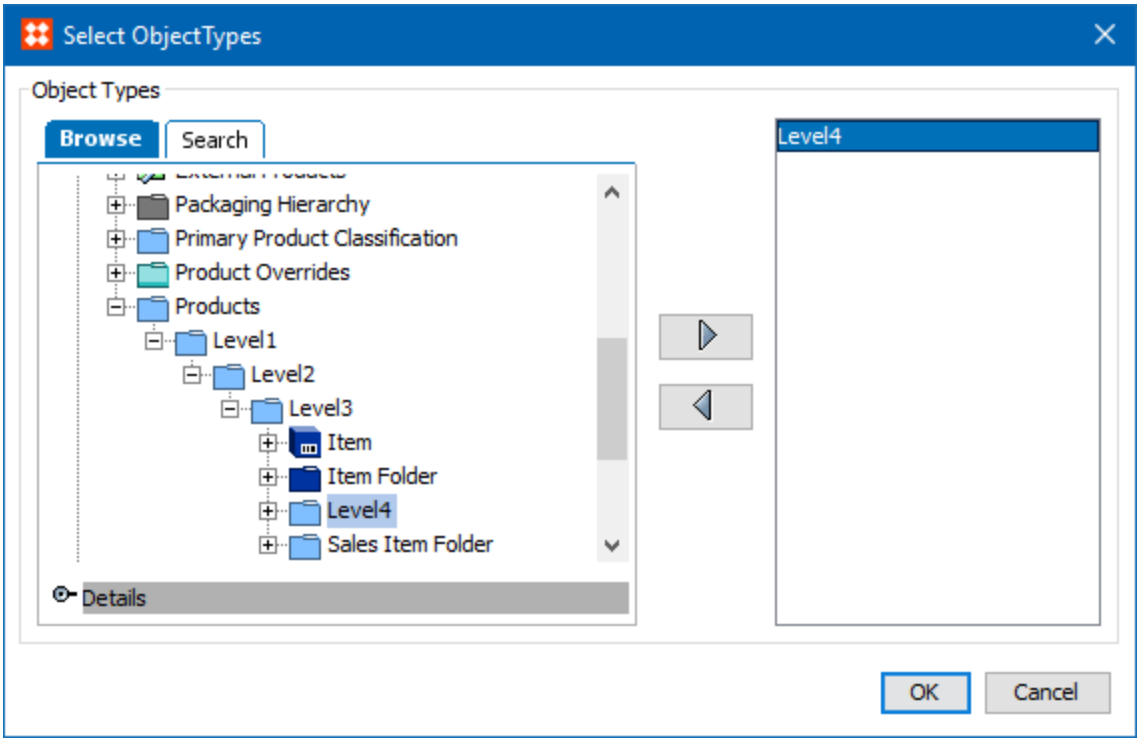
The Triggering Object Types section allows you to identify object type(s) that are important for this OIEP. The OIEP will listen for events on these object types in order to generate the desired output as defined by the output template configuration. When listening for events on changes to objects like 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, see 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.

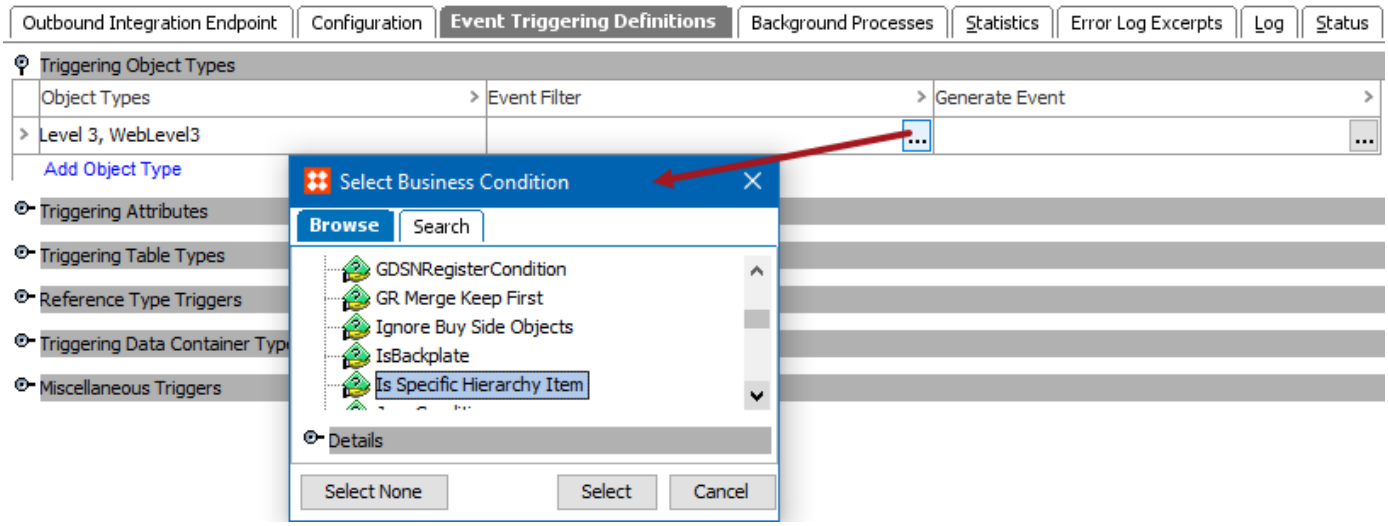
To select the object types:

- 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 approval (when approval is required), and can be used in place of a preprocessor. 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; if 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 you only want to output the children, you can use a business condition to filter out the parent. Then use a business action (see the Generate Event section below) to 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':

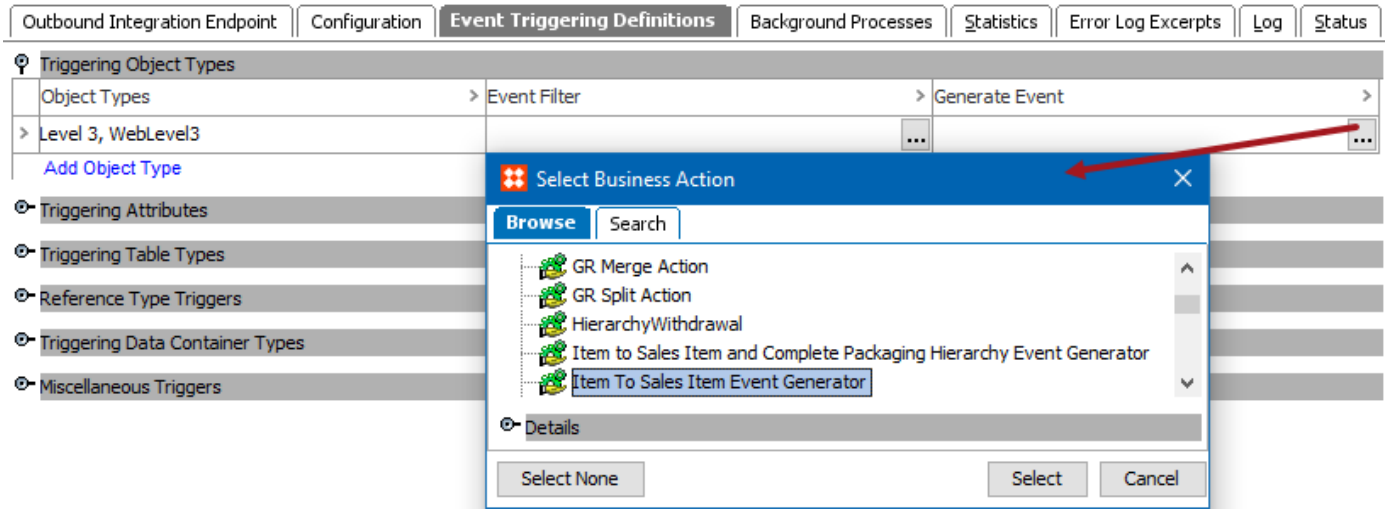
```
if (currentEventType instanceof
com.stibo.core.domain.eventqueue.DerivedEventType) {

// 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. See 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, see **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, see **Derived Events** of the **System Setup / Super User Guide** 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:

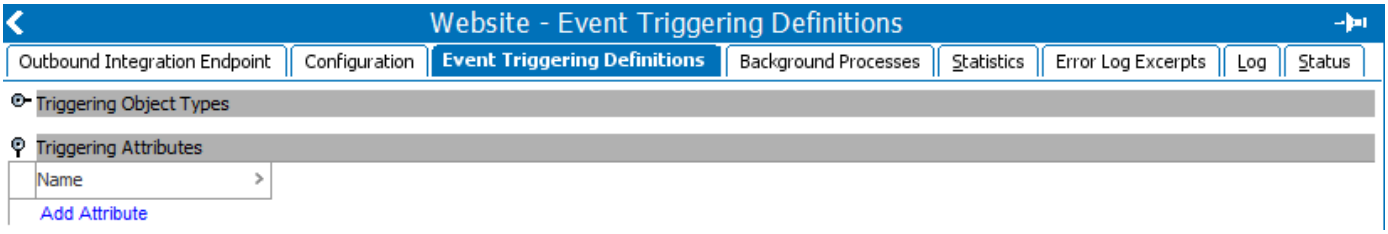
  ```
  currentEventQueue.queueDerivedEvent(webModify, currentObject);
  ```

- **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, see **Event-Based OIEP Example Business Rules for Derived Events**.

Triggering Attributes Flipper

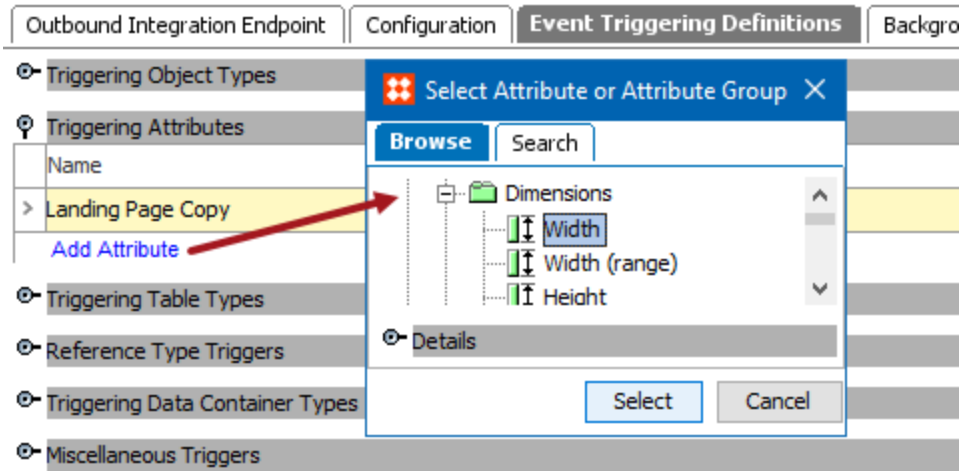


Triggering Attributes are used to filter the data being monitored by the OIEP in conjunction with the Triggering Object Types section. When a Triggering Attribute is defined, and that attribute's value changes for objects of the object types selected in the Triggering Object Types section, events are 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:

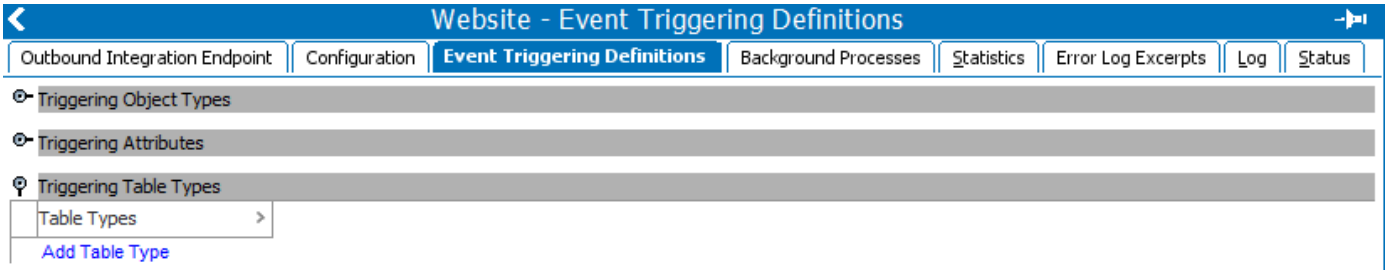
- Changes to values for externally maintained attributes and global revisable entities cause an event to be generated immediately.
- Changes to values for revisable attributes cause an event to be generated when the object holding the values is approved.

For more information, see the **Events** section of the **System Setup** documentation.

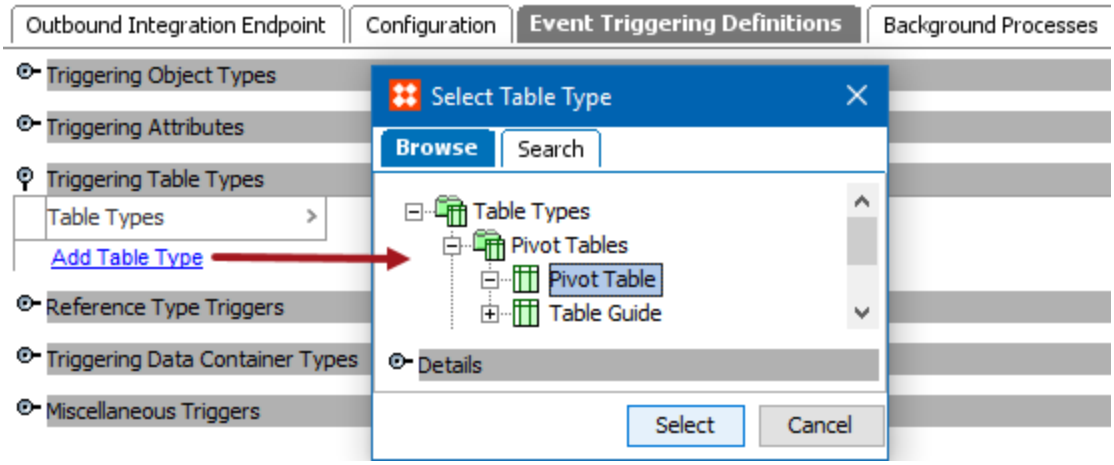


- Click **Add Attribute**, then browse or search for the relevant attribute(s) and/or attribute group(s).
- At least one attribute or attribute group must be selected, unless the integration handles ONLY System Setup data OR Create / Delete events.
- When an attribute or attribute group is selected as 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, see the **OIEP - Event-Based - Output Templates Flipper**.

Triggering Table Types Flipper

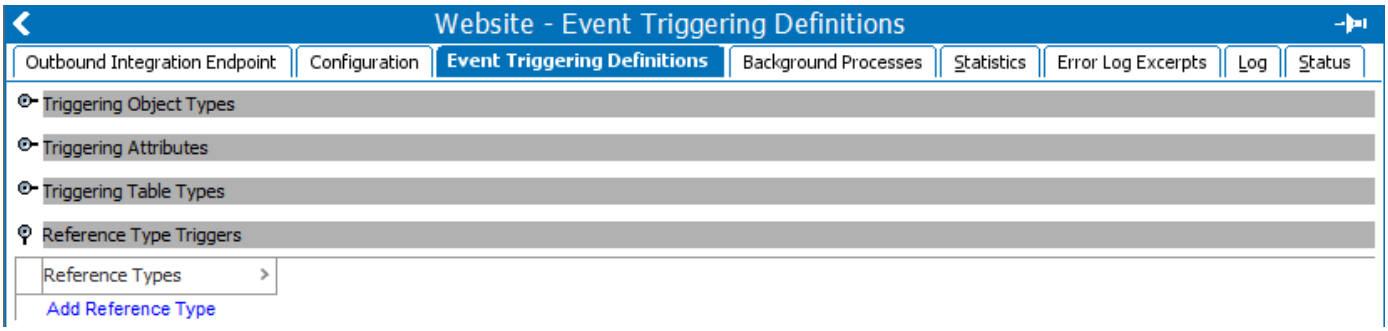


Allows you to specify (filter) table types for which you want events registered when tables owned by objects of the type listed in the Triggering Object Types section change. Common setup is to use tables and this setting only for print output.



- 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

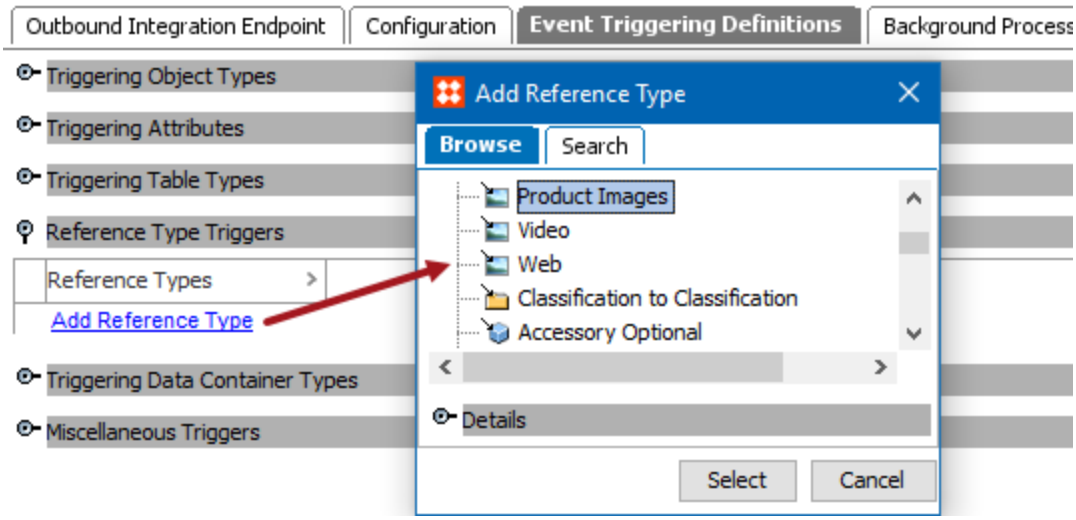


Use reference / link types to define (filter) which reference / link types to listen to for changes. This is a positive list so only the items listed are triggers. If functionality should vary by object type, a unique endpoint should be created for each object type.

Be aware that you can select both revisable and externally maintained reference / link types and events are generated as follows:

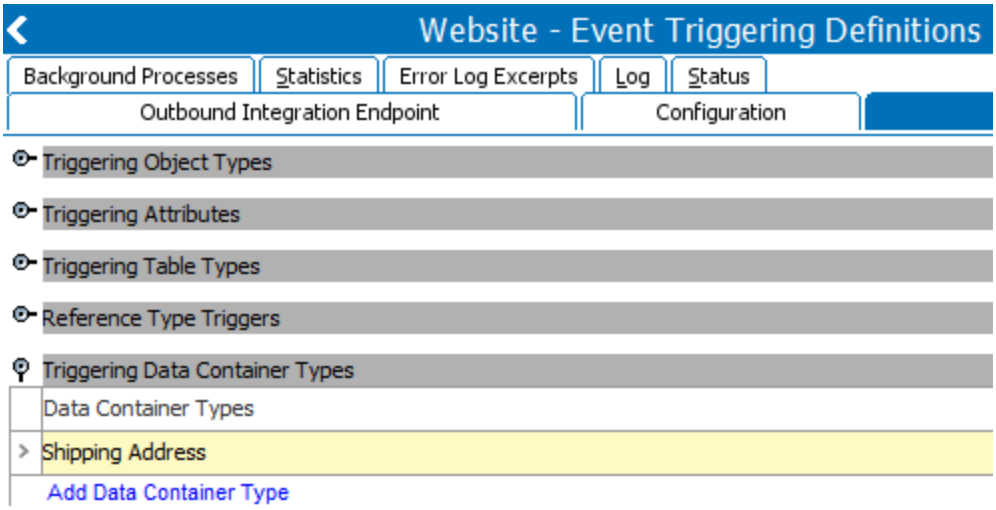
- Changes to values for externally maintained reference / link types cause an event to be generated immediately if an object type on which the reference is valid is selected as a triggering object type.
- Changes to values for revisable reference / link types cause an event to be generated when the object 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, see the **Events** section of the **System Setup** documentation.

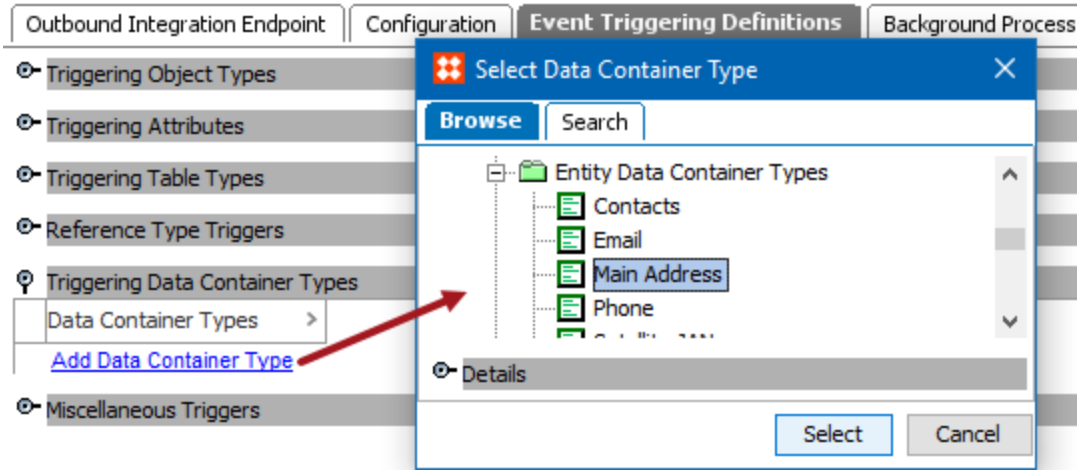


- Click **Add Reference Type**, then browse or search for the relevant reference type.

Triggering Data Container Types

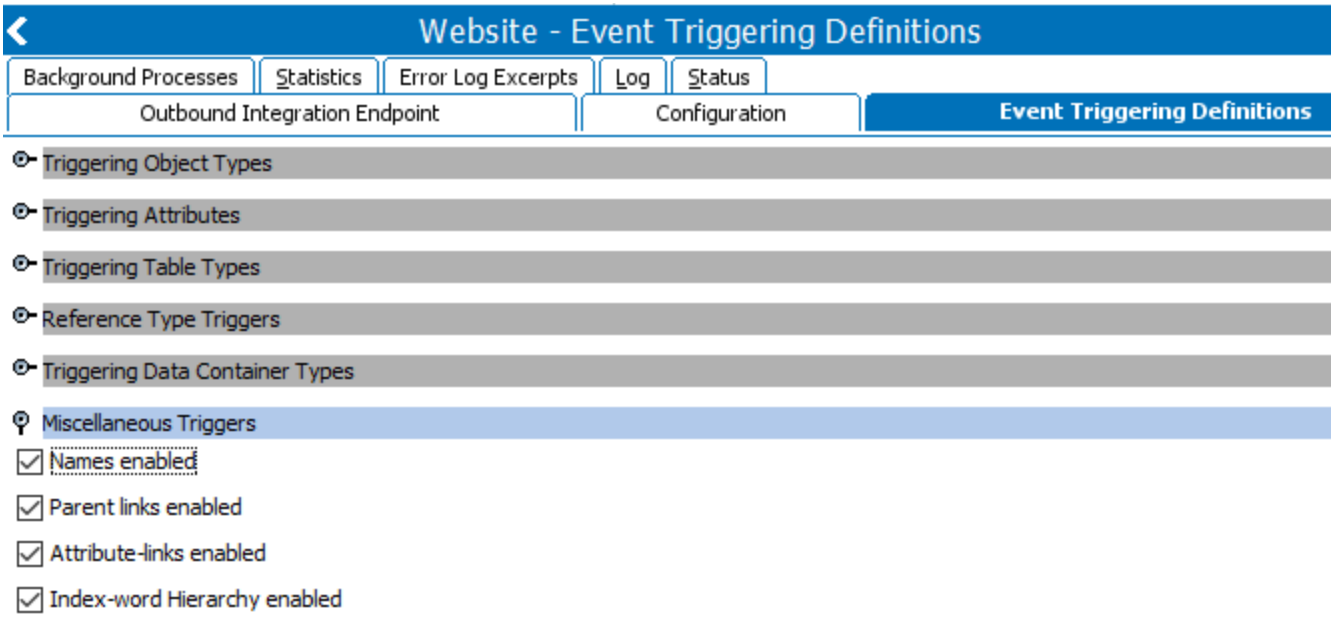


When monitoring changes on entity objects, specific data container types can be configured to trigger events upon creation / deletion of corresponding data containers. Any data container types omitted from the triggering definition will not trigger events.



- Click **Add Data Container Type**, then browse or search for the relevant data container type. Anytime 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 STEP Name of triggering object types for a global revisable object, or when a workspace revisable object is approved. An output template must be

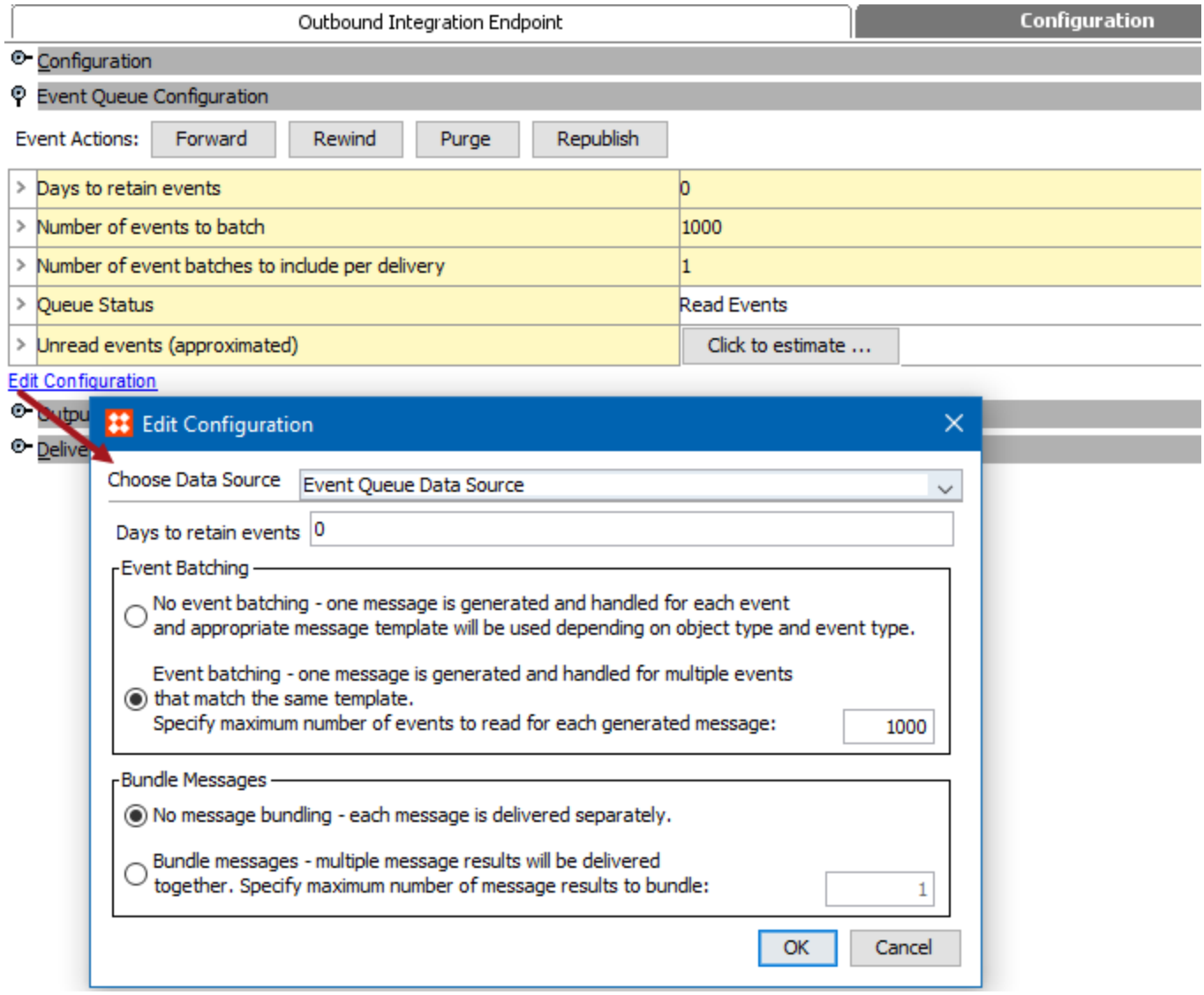
configured with a Modify event or the endpoint will skip the event since it cannot find a corresponding template.

- **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.

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, see **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, see **Event-Based OIEP Forward, Rewind, Purge, and Republish**.

- 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), see **Event-Based OIEP Event Batching**.
- For information on the **Queue Status** parameter, see **Event-Based OIEP Status and Queue Status**.
- For information on the **Unread events** parameter, see **Event-Based OIEP Queued Events**.

Creating a Select Objects Outbound Integration Endpoint

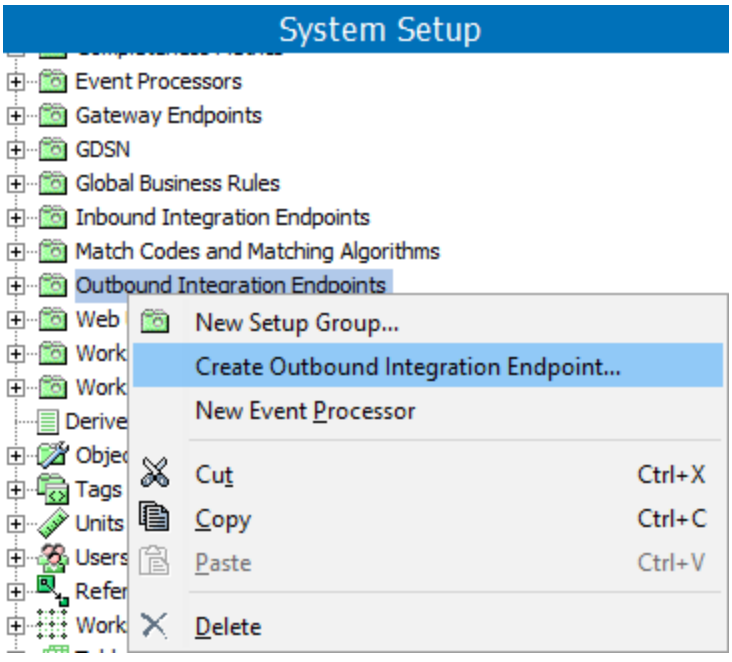
A Select Object OIEP publishes data non-incrementally from selected hierarchies (one or more root nodes) to external systems on scheduled intervals. This type of OIEP uses a static set of data and works exactly like a scheduled export created in Export Manager, with the differences being that the OIEP 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.

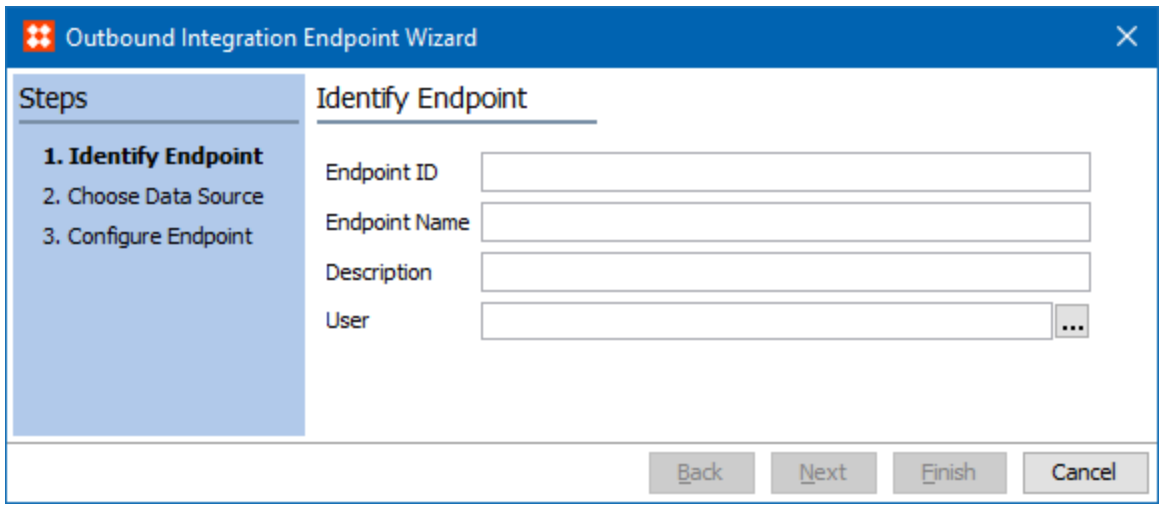
Select Object OIEP Setup

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, see **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**.

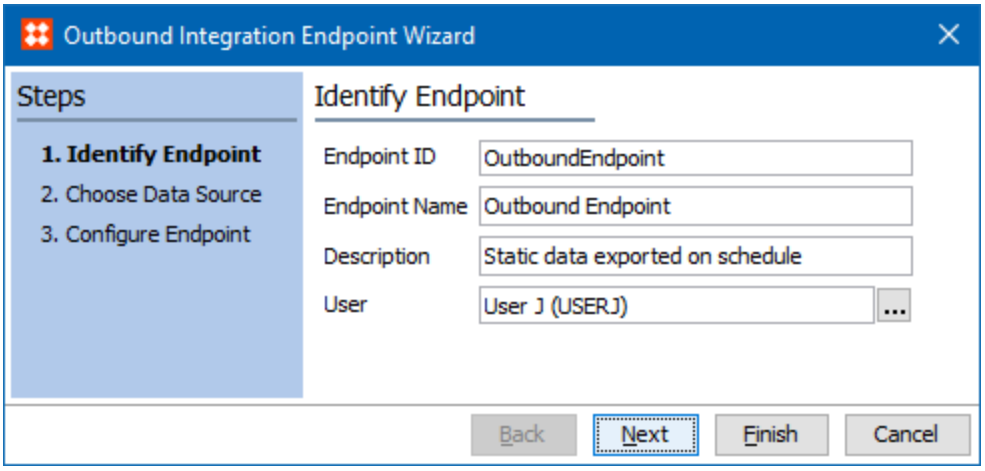


2. The Outbound Integration Endpoint wizard displays and involves the following steps:



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.

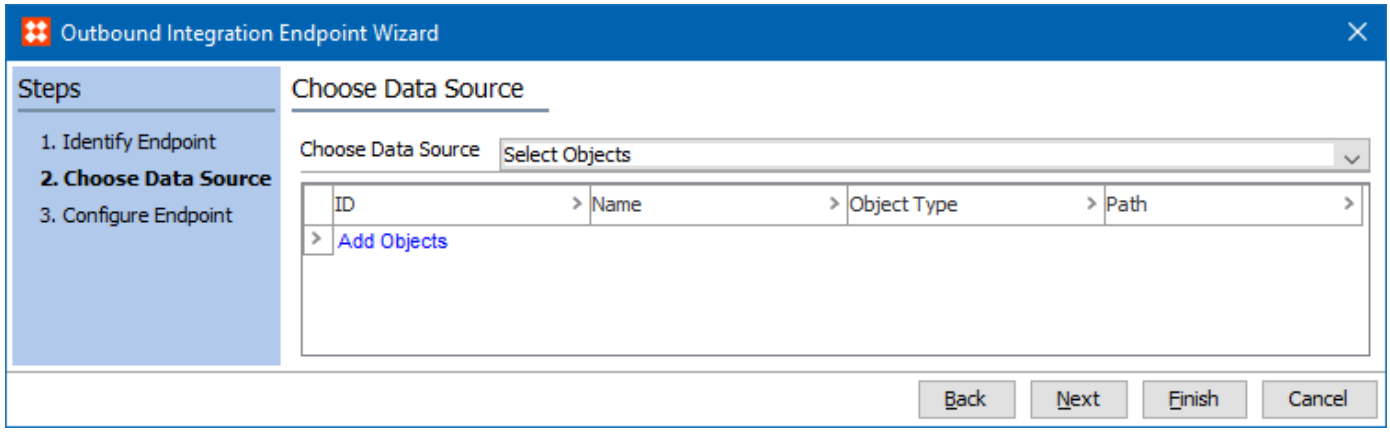
   Once the Endpoint ID, Endpoint Name, and User are supplied, the Next button is enabled.

Note: Attempting to delete a user who is selected in the User parameter for an OIEP will result in errors. For more information, see the **Working with Users** topic in the **System Setup / Super User Guide** documentation.

---

5. Click **Next** to display **OIEP - Select Objects - Choose Data Source**.

OIEP - Select Objects - Choose Data Source

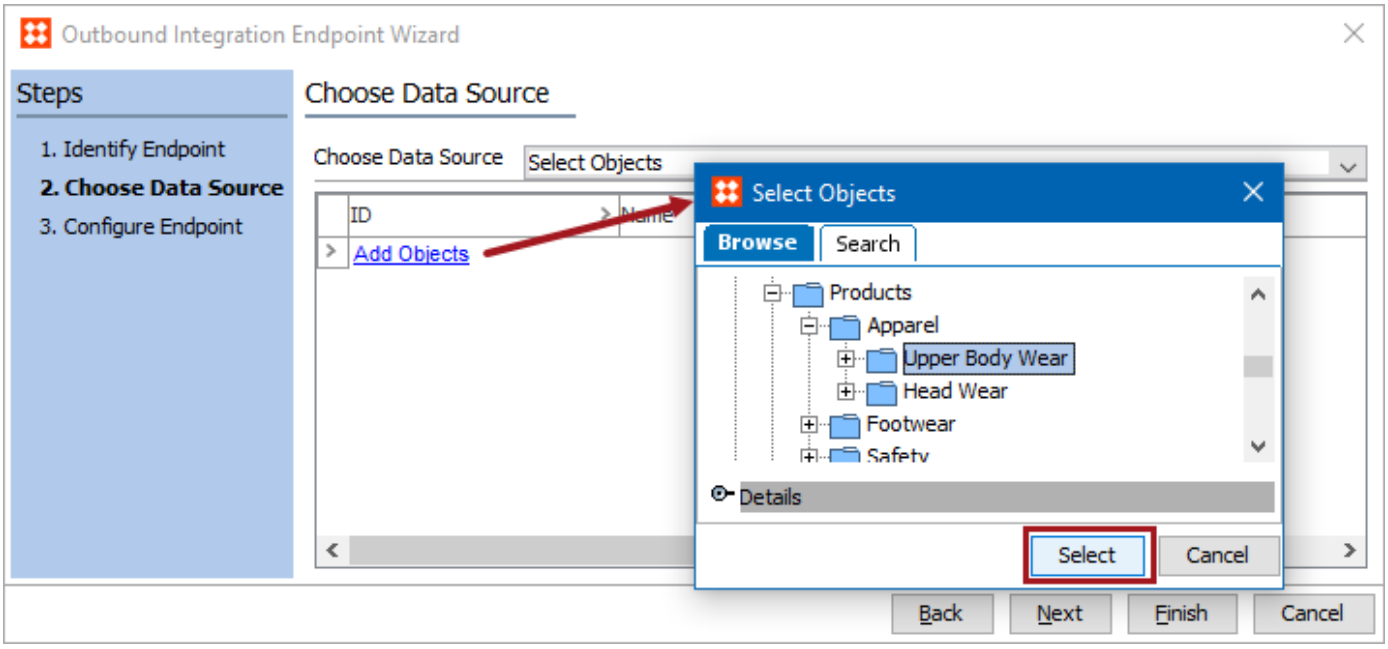


1. Choose the data source **Select Objects** for non-incremental exports where you want to publish data for specific hierarchies every time the endpoint is invoked.

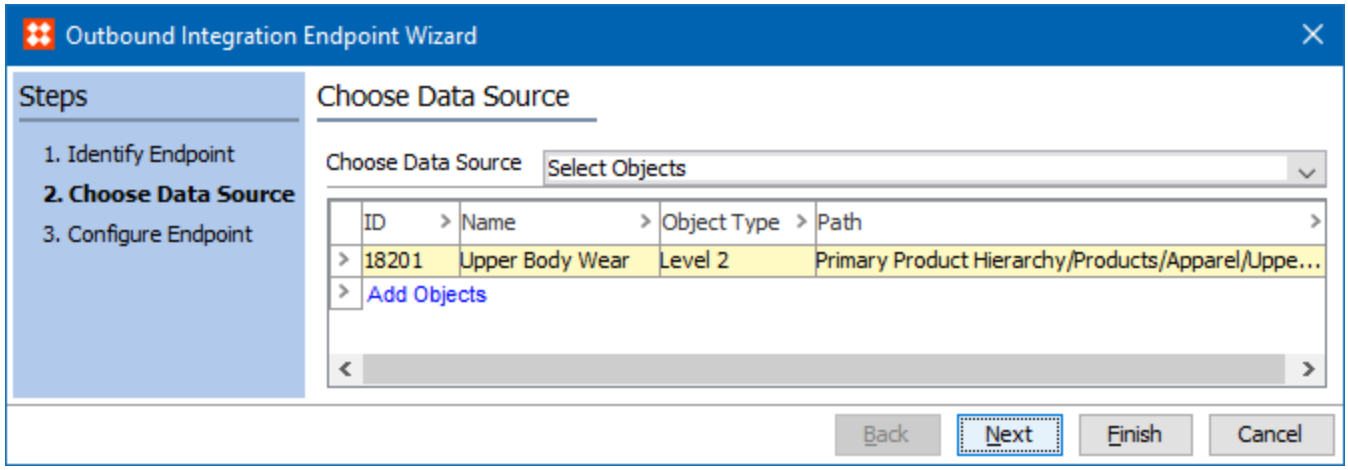
   Note: An **Event Queue Data Source** requires a separate setup. For detailed steps, see the **Creating an Outbound Integration Endpoint** documentation.

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.

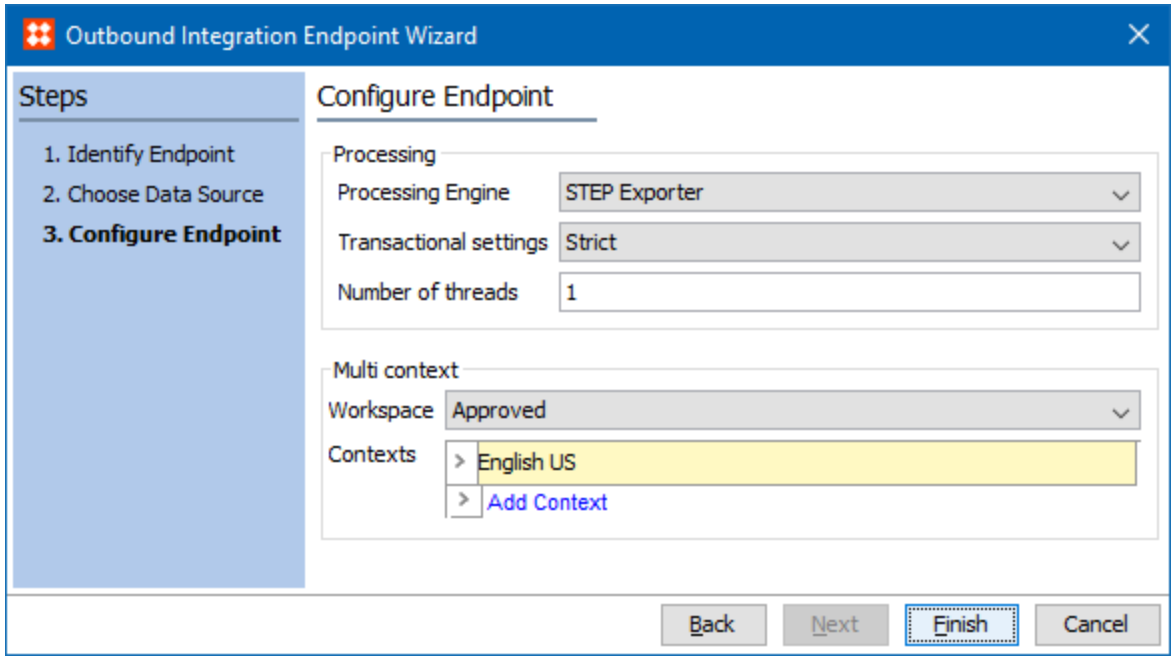


The selection(s) are displayed as a data source.

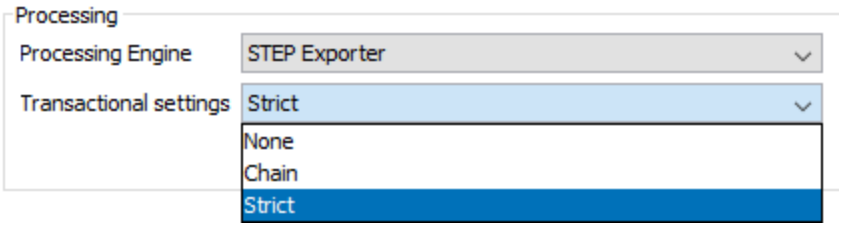


3. Click **Next** to display **OIEP - Select Objects - Configure Endpoint**.

OIEP - Select Objects - Configure Endpoint



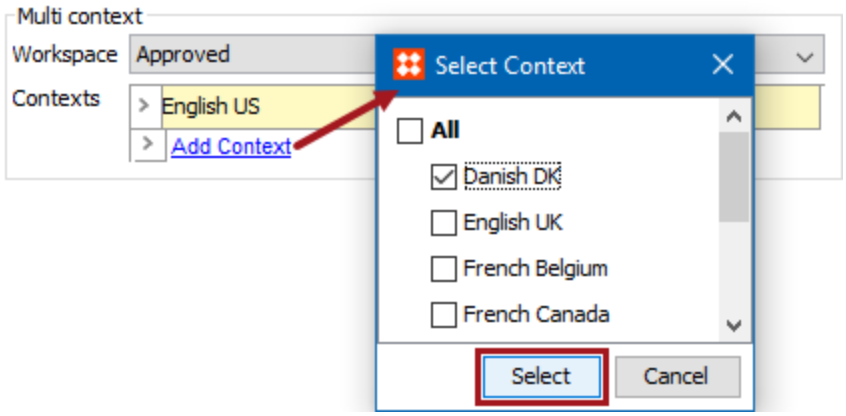
1. For the **Processing Engine** parameter, select an option from the dropdown:
   - **STEP Exporter** uses the same functionality as the Export Manager. This is the only processing engine option on a standard STEP system and performs the actual data export.
   - **Business Rule Based Message Processor** allows you to export STEP data using either JavaScript-based business actions, or Java business actions developed via the Extension API. For more information, see the **OIEP - Configuration Flipper for Business Rule Based Message Processor** topic or click the **STEP API Documentation** button on the STEP Start Page.
   - If you need a customized processing engine, contact your Stibo Systems account manager or partner manager. For more information, see **Outbound Integration Endpoint Structure** documentation.
2. The **Transactional Settings** option can be set for select objects OIEPs. For more information about Transactions Settings, see the **Integration Endpoint Transactional Settings** documentation.



3. The **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. 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**.

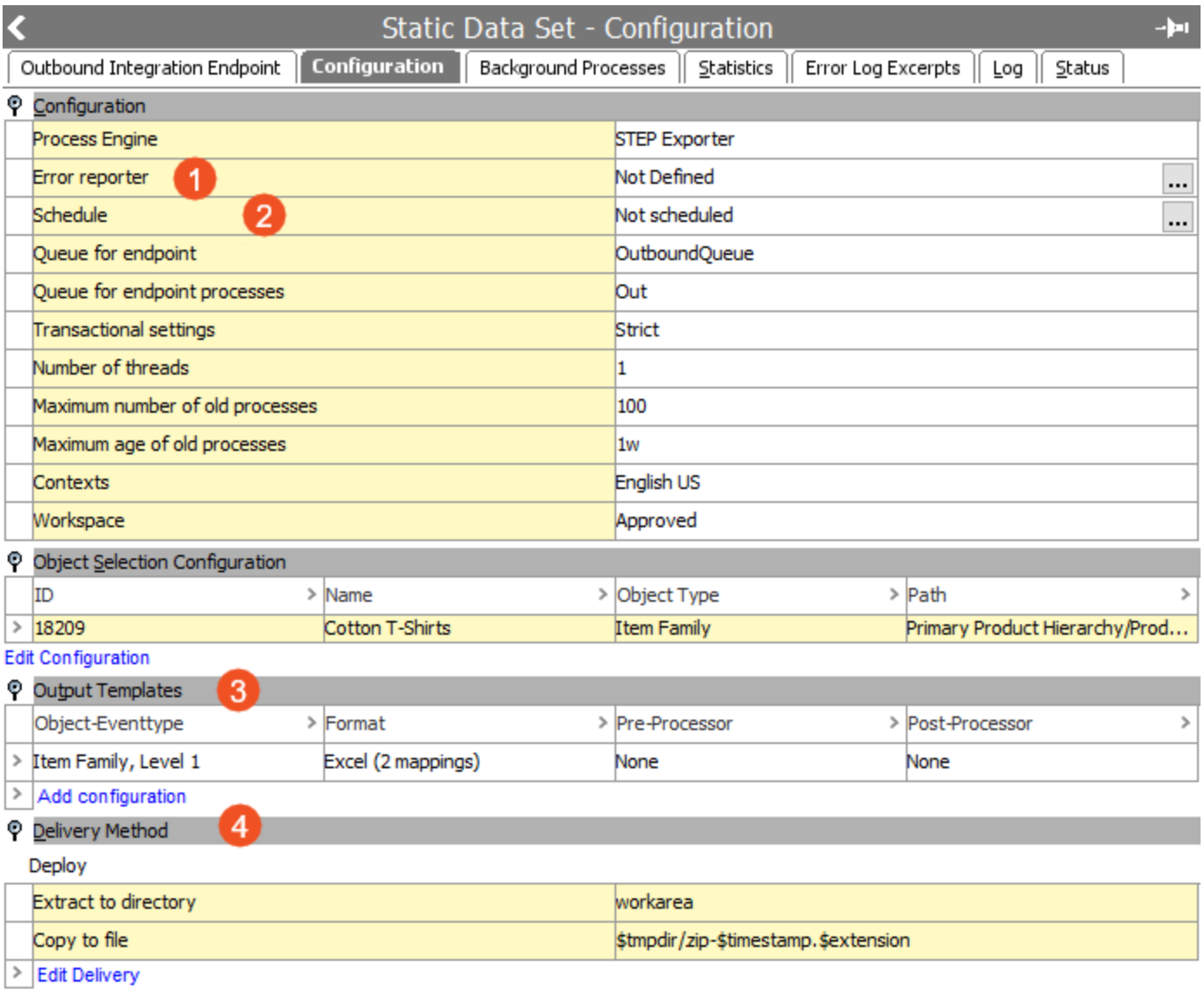
OIEP - Select Objects - Manual Configuration

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.

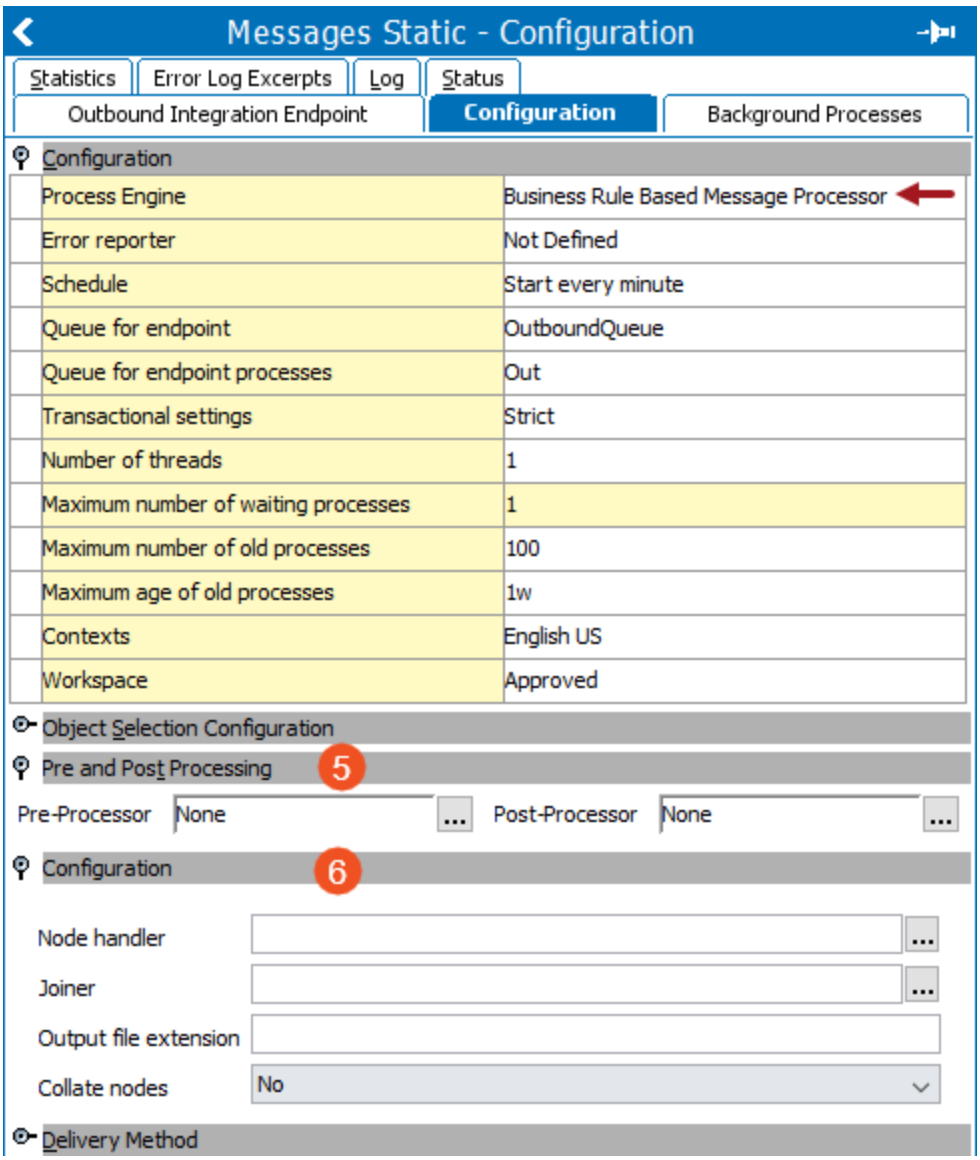
Note: In addition to the set up in this topic, additional manual configuration is required when using the Business Rule Based Message Processor, as defined in the **OIEP - Configuration Flipper for Business Rule Based Message Processor** topic.

To manually edit or review other parameter settings on an OIEP, see the **Maintaining an Outbound Integration Endpoint** documentation.

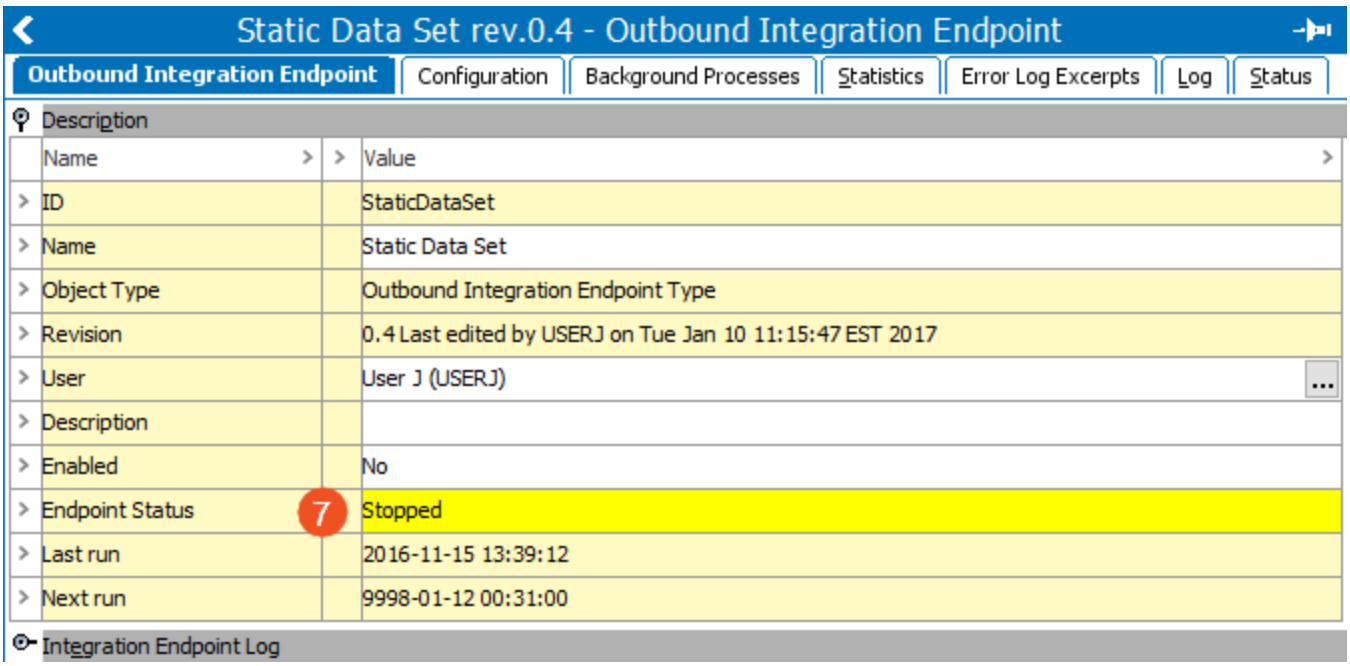
1. In the OIEP editor, configure to send an email if an endpoint-related background process fails as described in the **Error Reporter** section of the **OIEP - Configuration Flipper** documentation.
2. In the OIEP editor, determine when the OIEP runs as described in the **Schedule** section of the **OIEP - 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**.

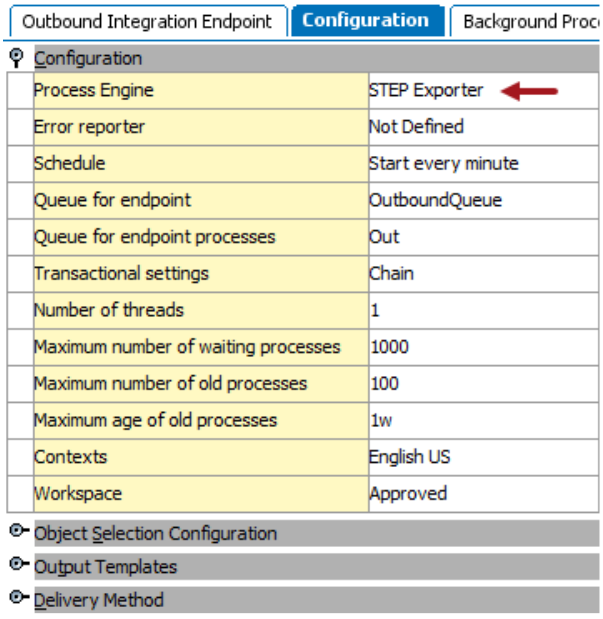


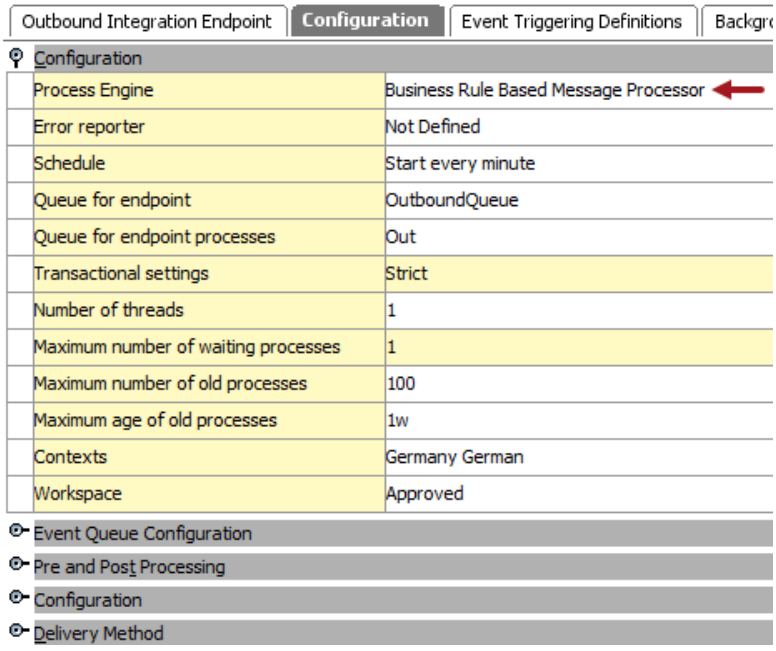
7. Enable the endpoint and invoke it as described in **Running an Outbound Integration Endpoint**.


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, see **OIEP - Event-Based - Configure Endpoint** documentation or **OIEP - Select Objects - Configure Endpoint** documentation.

Error Reporter

The 'Error reporter' parameter is not included in the wizard, but it can be configured to send an error report email if a background process fails. The email contains information about the failed endpoint, including the background process, failed file, failing process step, cause of the error, and a copy of the file that triggered the error. A standard STEP system (without custom extension) includes only the 'Send Error Report' option.

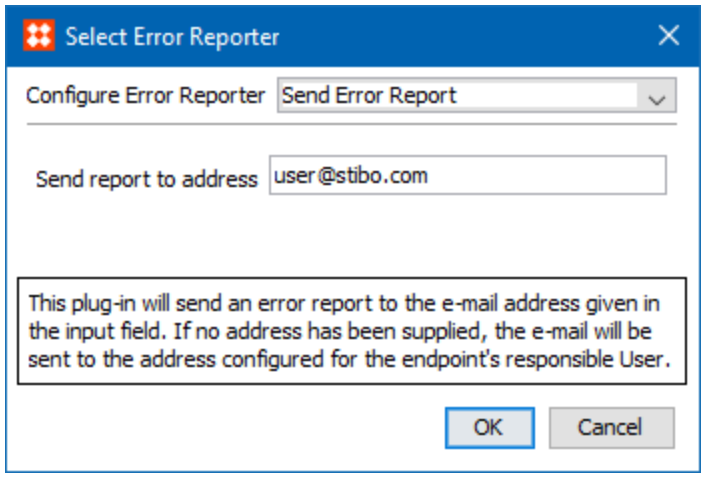
For information on configuring email from STEP, see the **Email from STEP** topic in the **Resource Materials** of online help.

By default, error reports are sent when either errors or warnings are encountered. To disable error reports when only warnings are reported, update the sharedconfig.properties file on the app server to include the following property:

```
Integration.Endpoint.SendEmailErrorReportOnWarning=false
```

Configure the Error Report Email

1. On the relevant OIEP > Configuration tab > Configuration flipper > click into the **Error Report** parameter to display the ellipsis button (...).
2. Click the ellipsis button (...) to display the **Select Error Reporter** dialog.



3. In Configure Error Reporter field, select **Send Error Report**.
4. In **Send report to address** field, enter one or more email addresses for the error report recipient(s). Use a semicolon-separated list to add more than one email address.

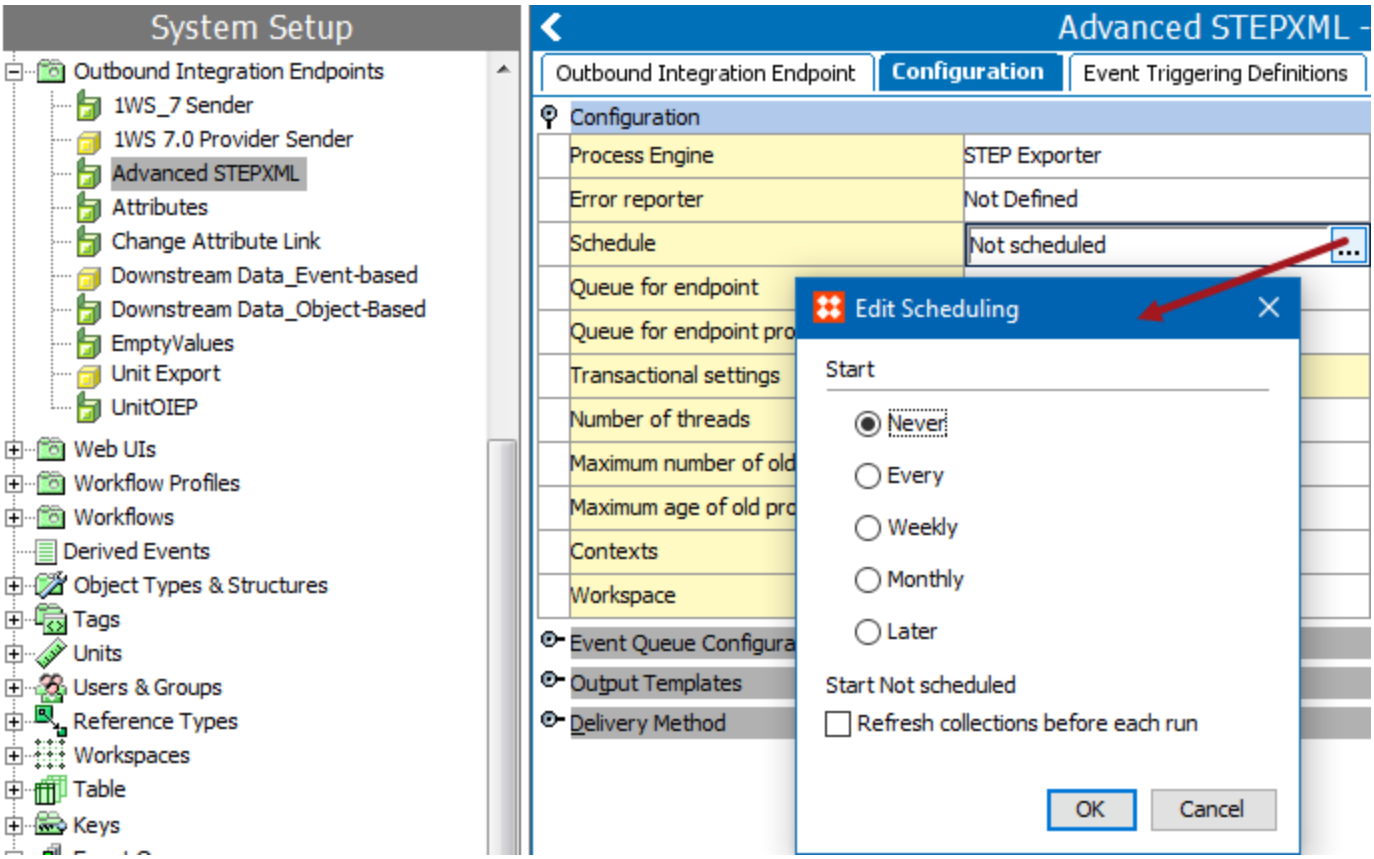
Note: If no email address is entered, the error report will be sent to the email address of the user who created the integration endpoint. If no email is defined for this user, or if no mail server is defined in the configuration, the error report will be written to the failed Background Process Execution Log.

Schedule

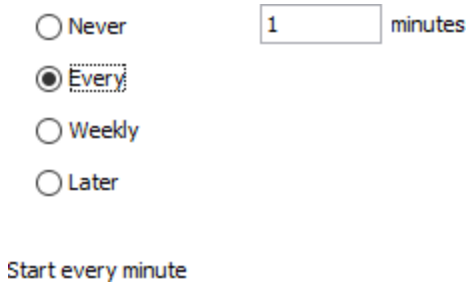
The schedule parameter is not included in the wizard, but it can be configured to repeatedly run the OIEP. For Select Objects OIEPs, the background processes runs as scheduled, regardless of changes to the data. For Event-Based OIEPs, the background process runs as scheduled but only when events are 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.

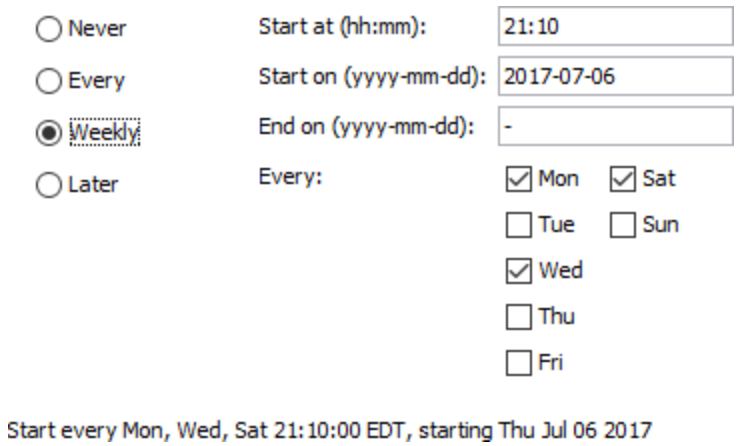
1. On the relevant OIEP > Configuration tab > Configuration flipper > click into the **Schedule** parameter to display the ellipsis button (...).
2. Click the ellipsis button (...), and then select one of the following options:



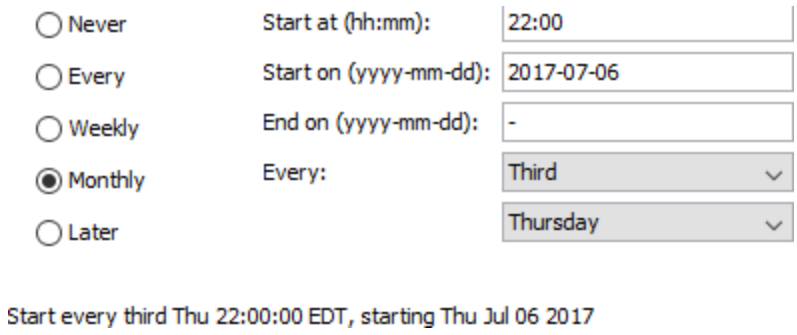
- **Never** - invoke the endpoint manually, no additional parameters are required, and no schedule is applied. This is the default setting and should be used while testing your endpoint.
- **Every** - automatically run the endpoint repeatedly, every selected number of minutes. One (1) minute is the shortest 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.



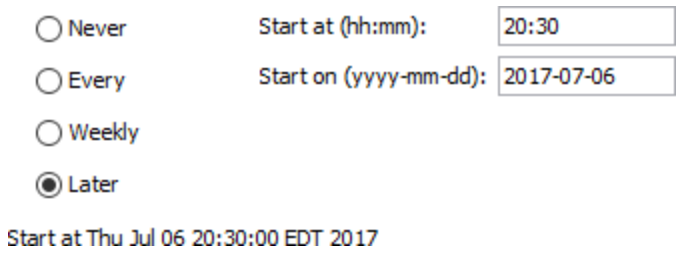
- **Weekly** - automatically run the endpoint repeatedly, based on the selected time, start and end dates, and days of the week. Use this option if a daily schedule is needed. The 'Start at' parameter determines the time of day that the endpoint will run. The 'Start on' parameter determines the date 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.



2. Select **Refresh collections before each run** if the outbound integration endpoint contains collections that should be automatically refreshed before each run.

Queue for Endpoint

'Queue for endpoint' is the name for the queue that is used by the OIEP Background Process to poll the endpoint. 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 OIEP, for example, including the OIEP ID in the name.

Queue for Endpoint Processes

'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 OIEP 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 detail, see the **Integration Endpoint Transactional Settings** topic.

Number of Threads

Although '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), 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. For more information, see **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.

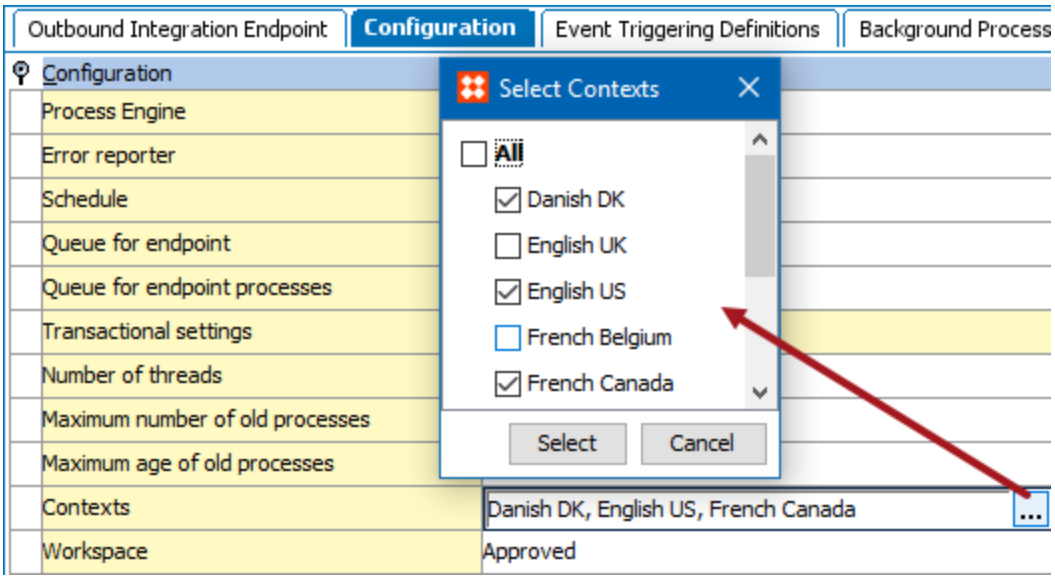
By default, the maximum age of old processes is set to one year.

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, see 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



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.

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.

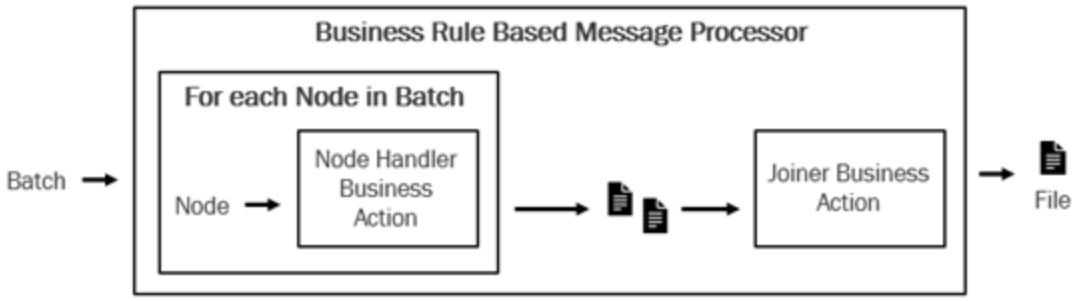
OIEP - Configuration Flipper for Business Rule Based Message Processor

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, see 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.



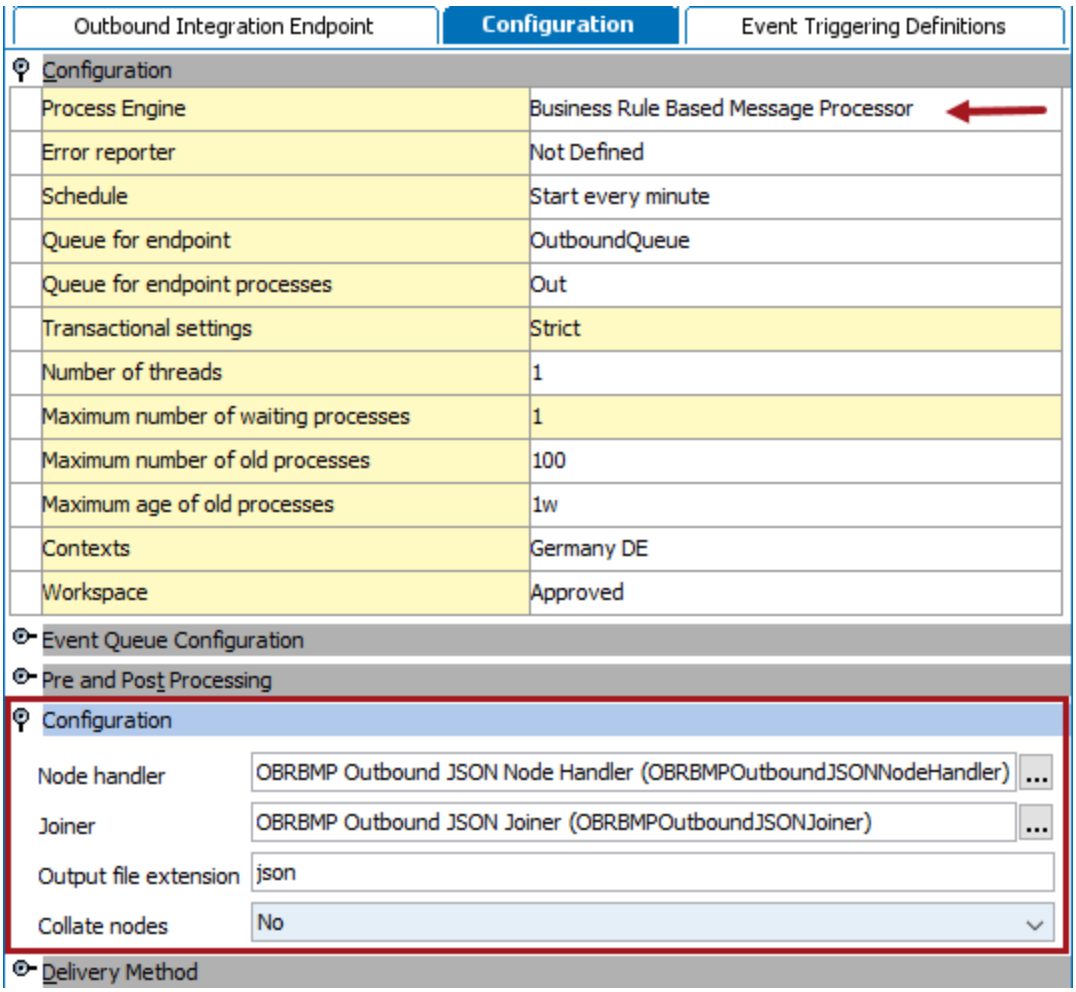
Prerequisites

1. Business actions used by this processor should be configured to be valid for all object types.
2. 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, see 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.

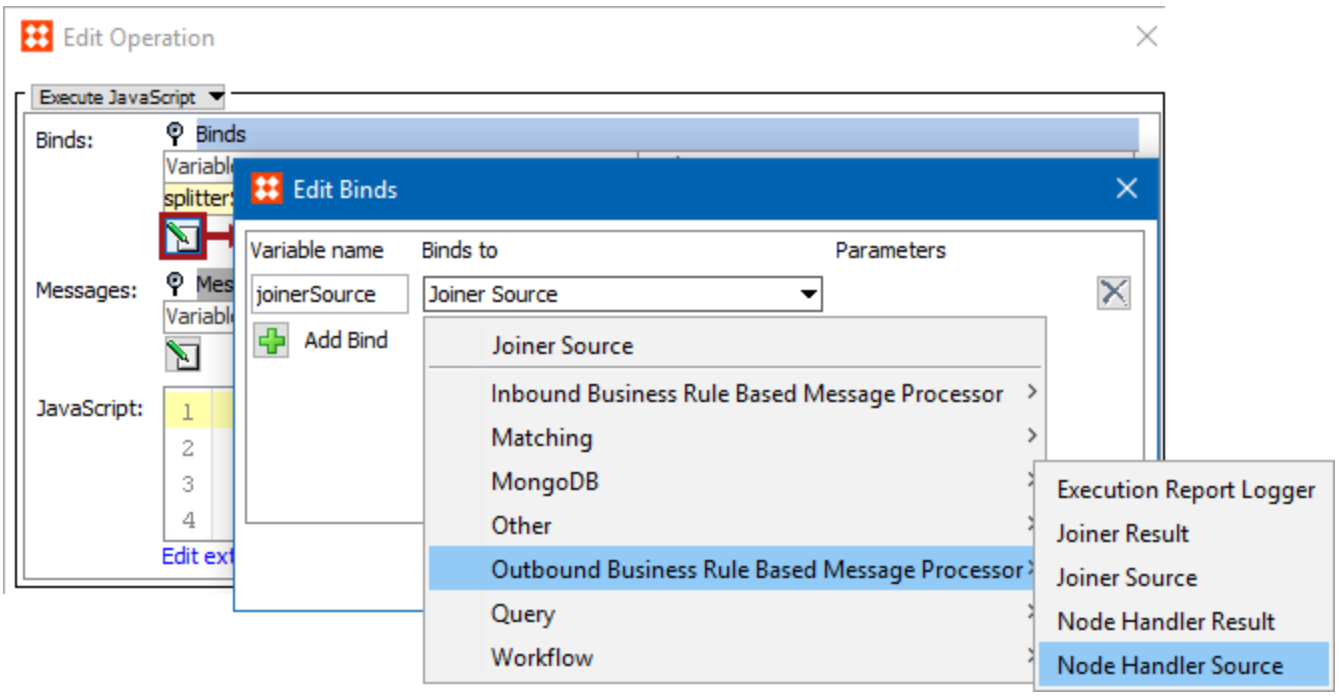
The **OutboundBusinessProcessorNodeHandlerSource** interface (for JavaScript business actions, the interface is available via the Node Handler Source bind option) provides the business action with access to the batch element. The interface has the following methods for accessing node / event information:

- **`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 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));
  }
}
```

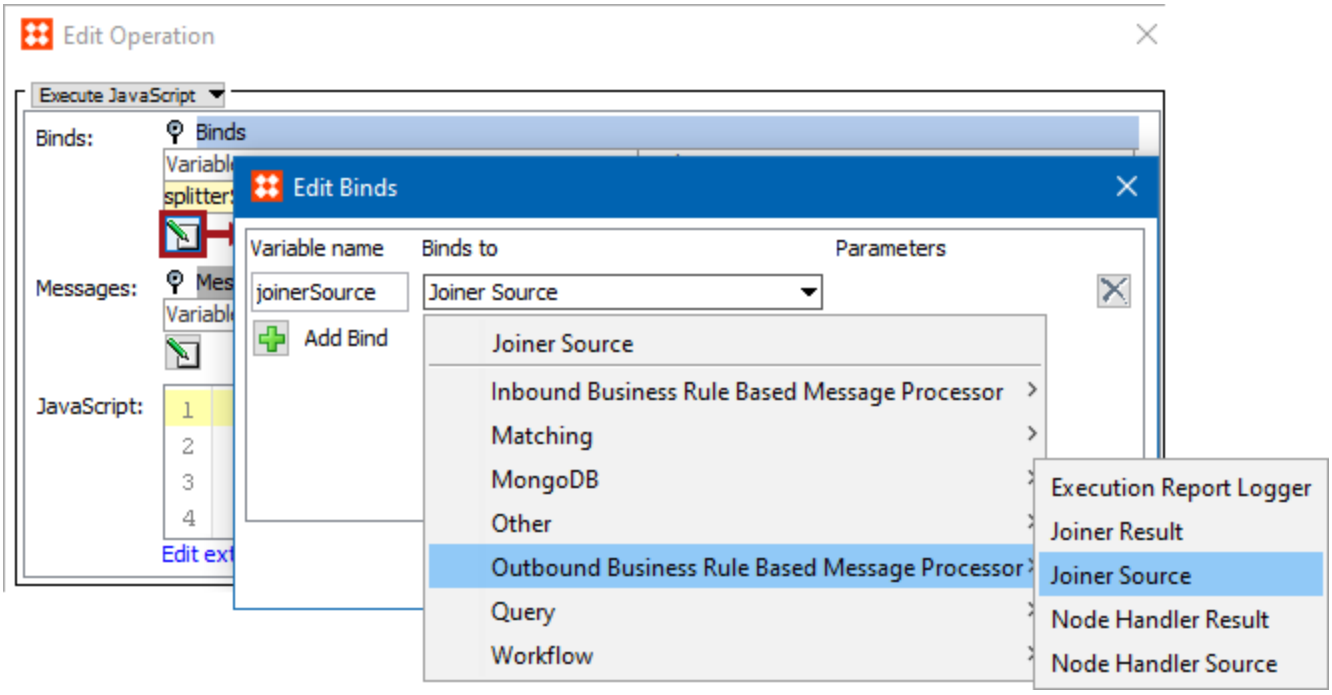
Joiner Business Action Details and Examples

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 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 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("]}}");
```

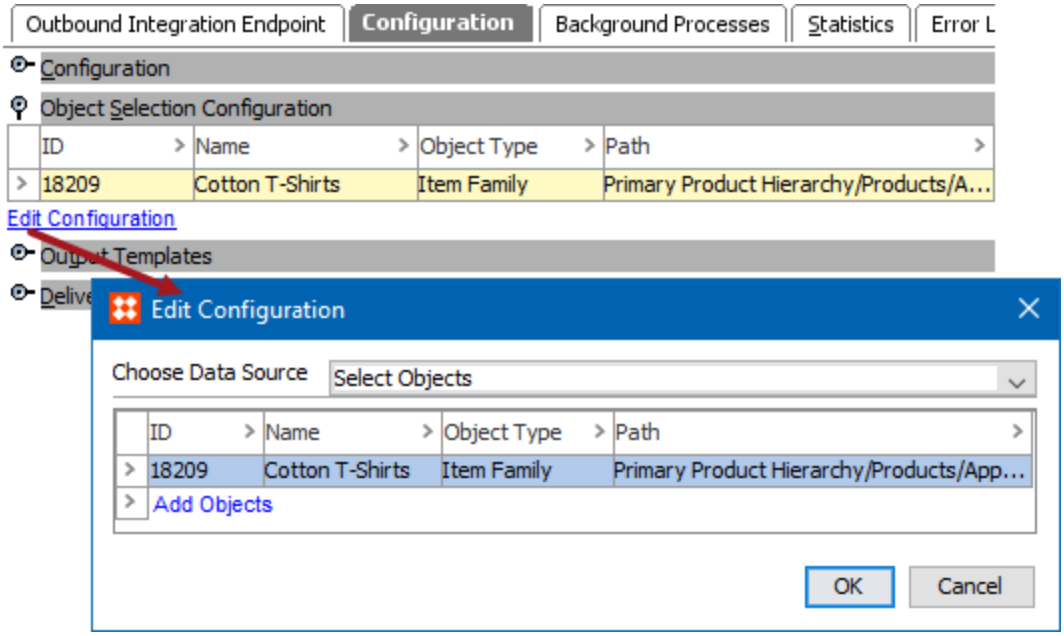
OIEP - Select Objects - Object Selection Configuration Flipper

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, see **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.

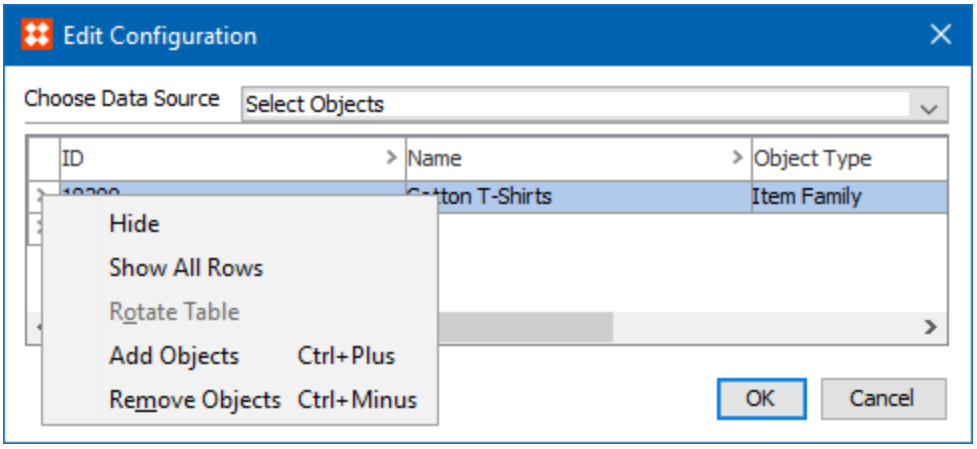


- Select an existing row and right-click to display additional options.

  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.



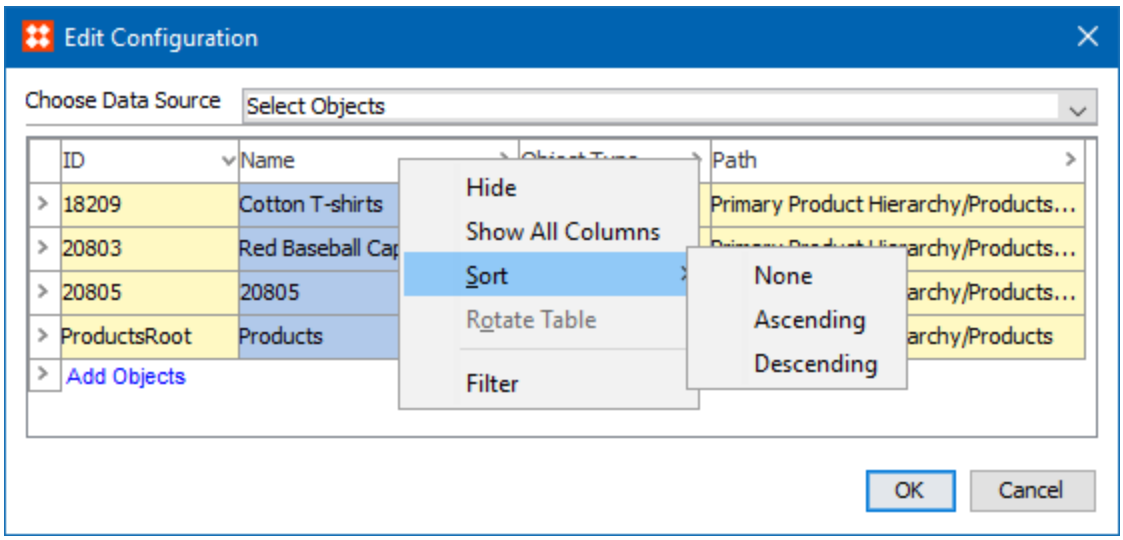
- Select an existing column and right-click to display additional options.

  Hide only affects the selected column.

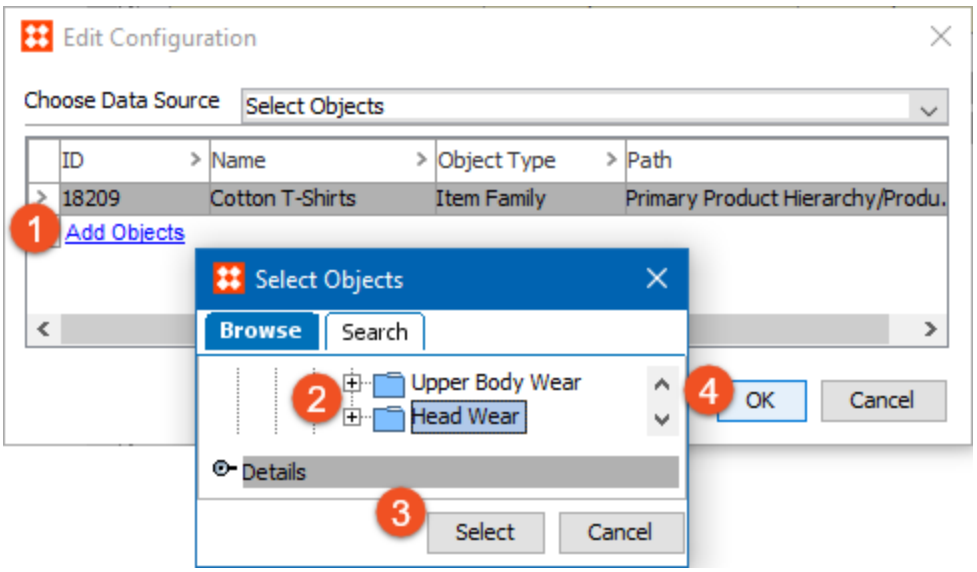
  Show All Columns affects all columns.

  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.

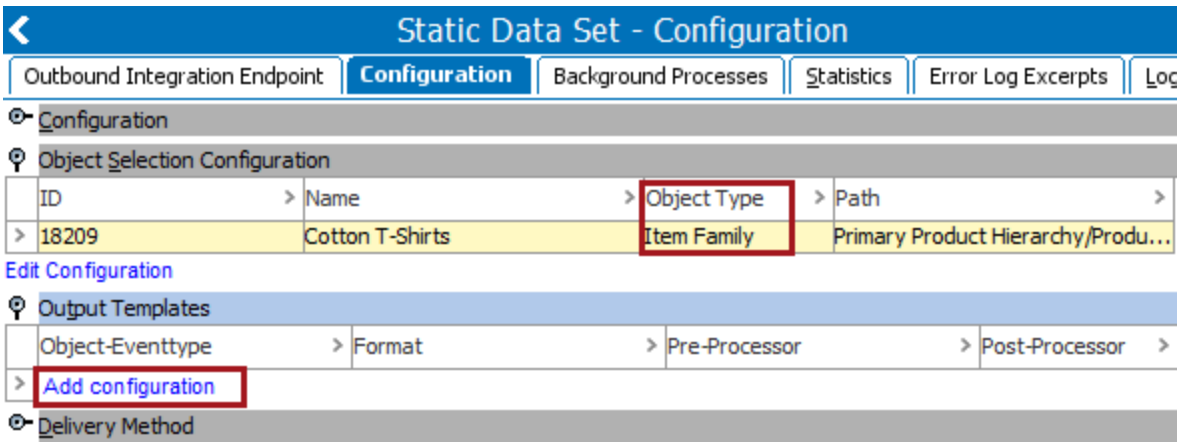
Configuring a select objects output template involves the following steps:

- Configure the object types
- Specify the format
- Configure preprocessor and postprocessor, based on availability on your system.

Configure the Object Types

The object type selection determines the STEP data that will be exported.

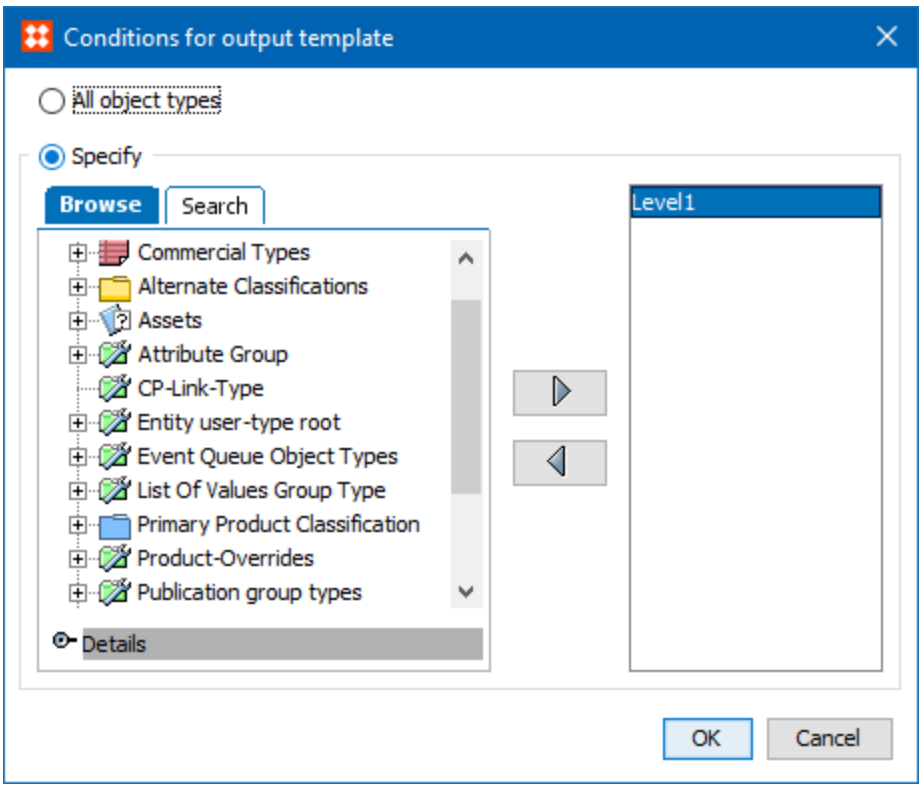
1. In System Setup, select the relevant OIEP > Configuration tab > Output Templates flipper > click the **Add configuration** link to display the 'Conditions for output template' dialog.



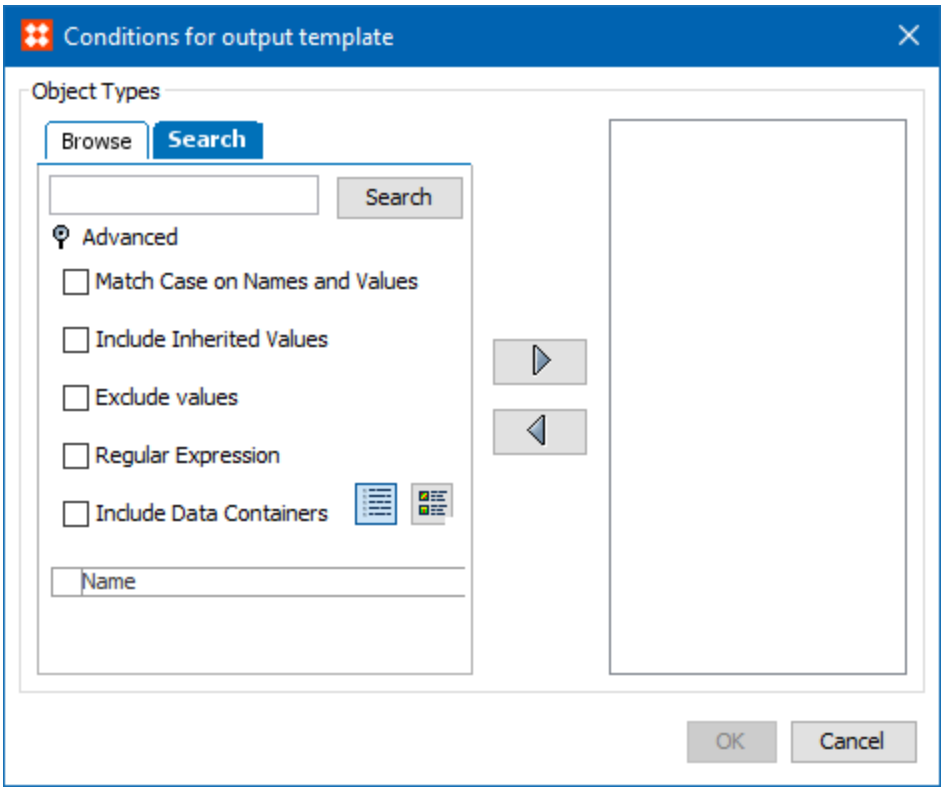
2. In the 'Conditions for output template' dialog, select 'All object types' or 'Specify' and choose the object types that correspond at the root level to the objects selected in the wizard **OIEP - Select Objects - Choose Data Source** step, which is displayed in the Object Selection Configuration flipper 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.

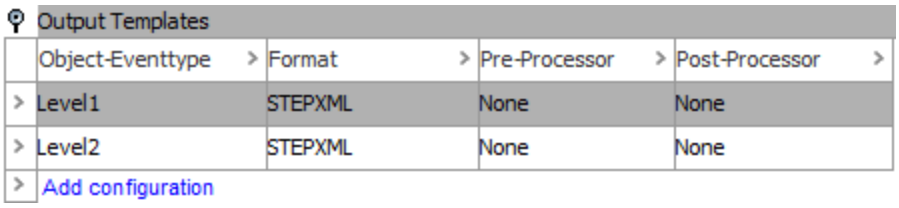


On the Search tab, open the Advanced flipper to display additional search options.



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.

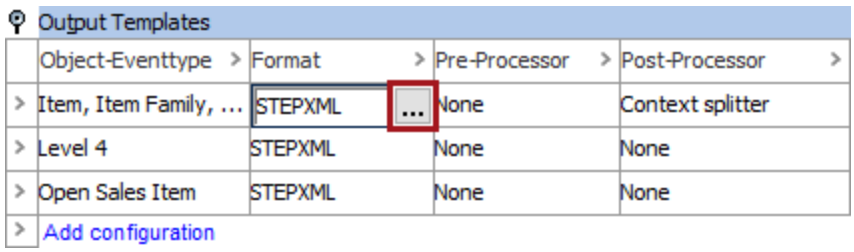
Limitations of Multiple Output Templates

- 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.

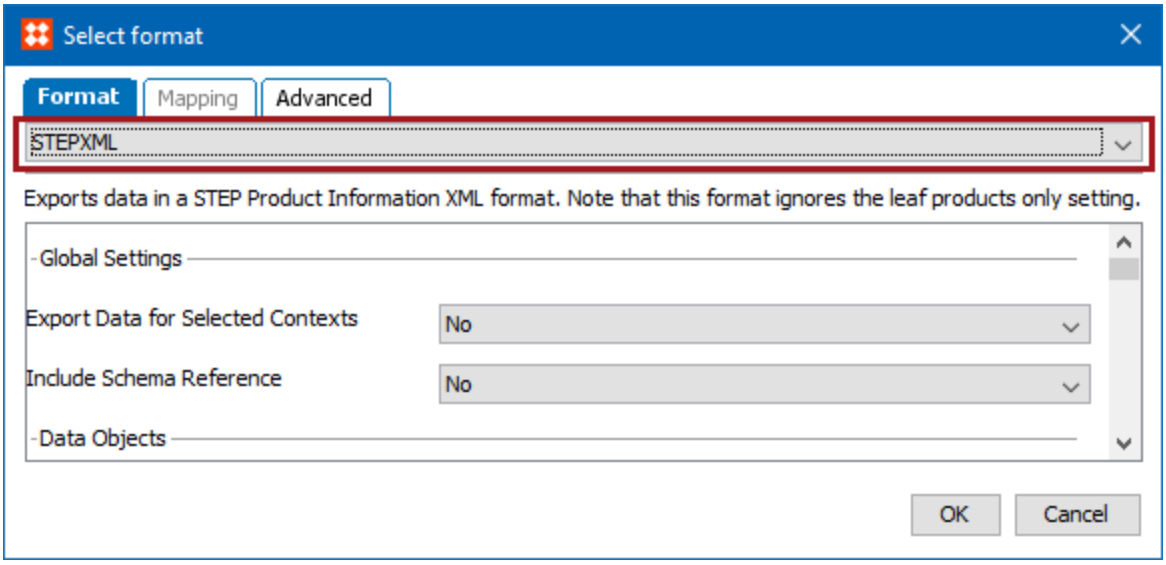
Configure the Format

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.

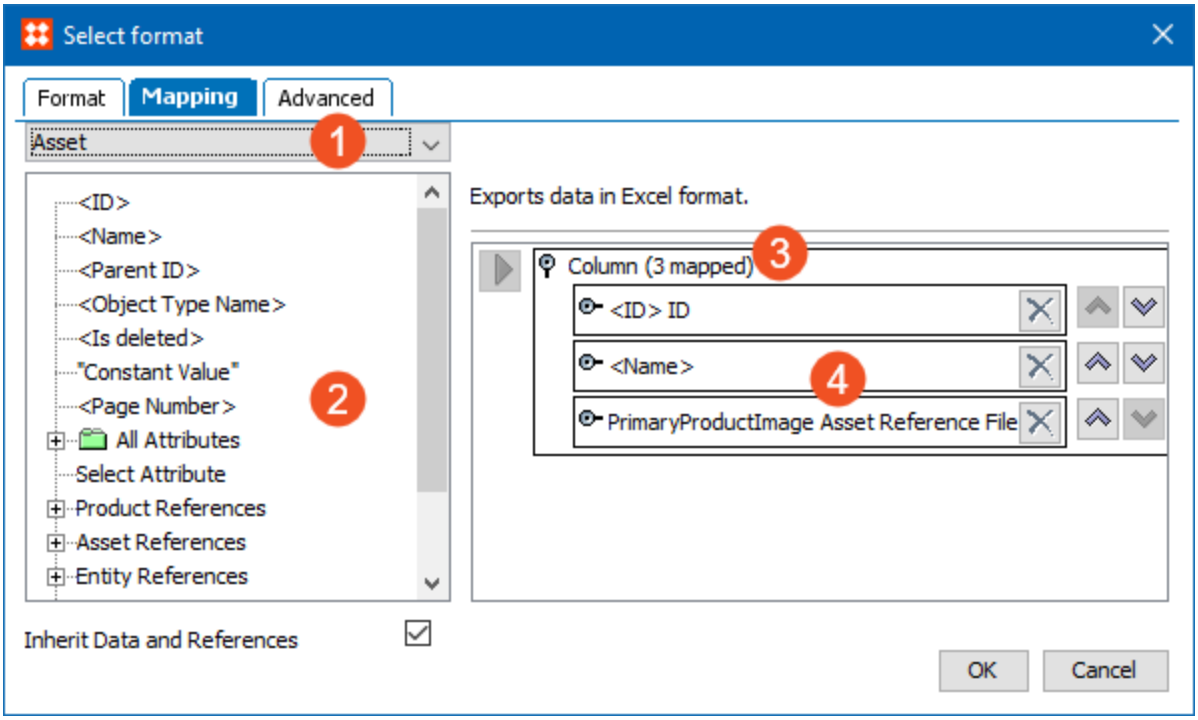


2. In the **Select format** window, on the **Format** tab, select the desired format from dropdown. These are the same options as in the Export Manager, based on your licenses. Use the following links for additional information:
   - Advanced STEPXML Format
   - Alphabetical Index - XML Format
   - Ariba 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
   - IDoc MATMAS05 Format
   - Publication Excel Format
   - STEPXML Format
   - xCBL Format



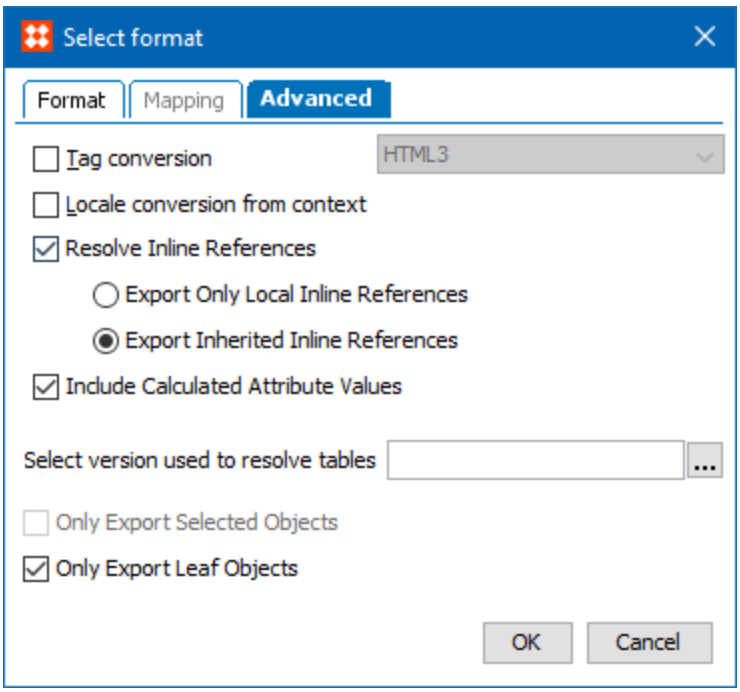
3. When available, specify how to map data to the export format, including the elements identified in the image below: object super type dropdown, data source, mapping target, and mapping rules.

   For more information about the object super type dropdown, see the **Export Manager - Select Objects** topic. For more information about the other elements, see 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, see **Inline References in Attribute Values** in the **Getting Started / User Guide**.

- **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, see **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, see the **Calculated Attributes** topic in **System Setup / Super User Guide** 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 only available if the format is STEPXML and 'Include Tables' is set to Yes. 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). If a publication version is not selected, the version last used in STEP Workbench is used. For more information, see 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, see **Export Manager - Advanced** documentation.

Configure the Pre-processor and Post-processor

Depending on your system setup, preprocessor and postprocessor options may be available.

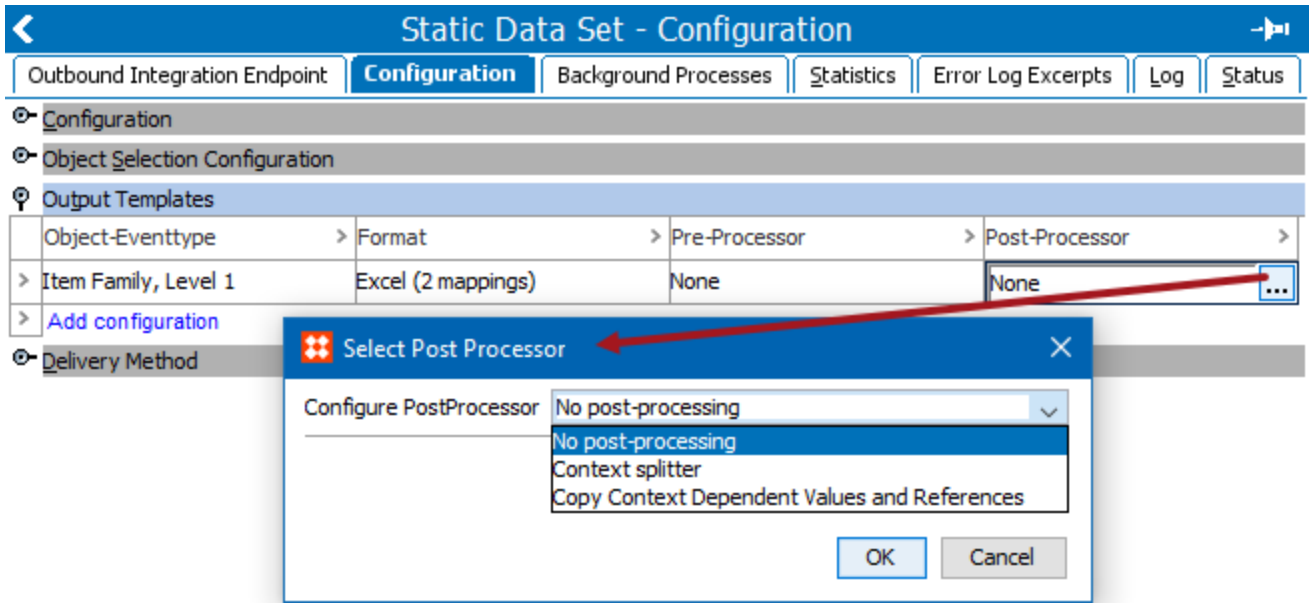
Pre-processor

Although a business action preprocessor 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 postprocessing 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**, 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, see 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, see 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.

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:

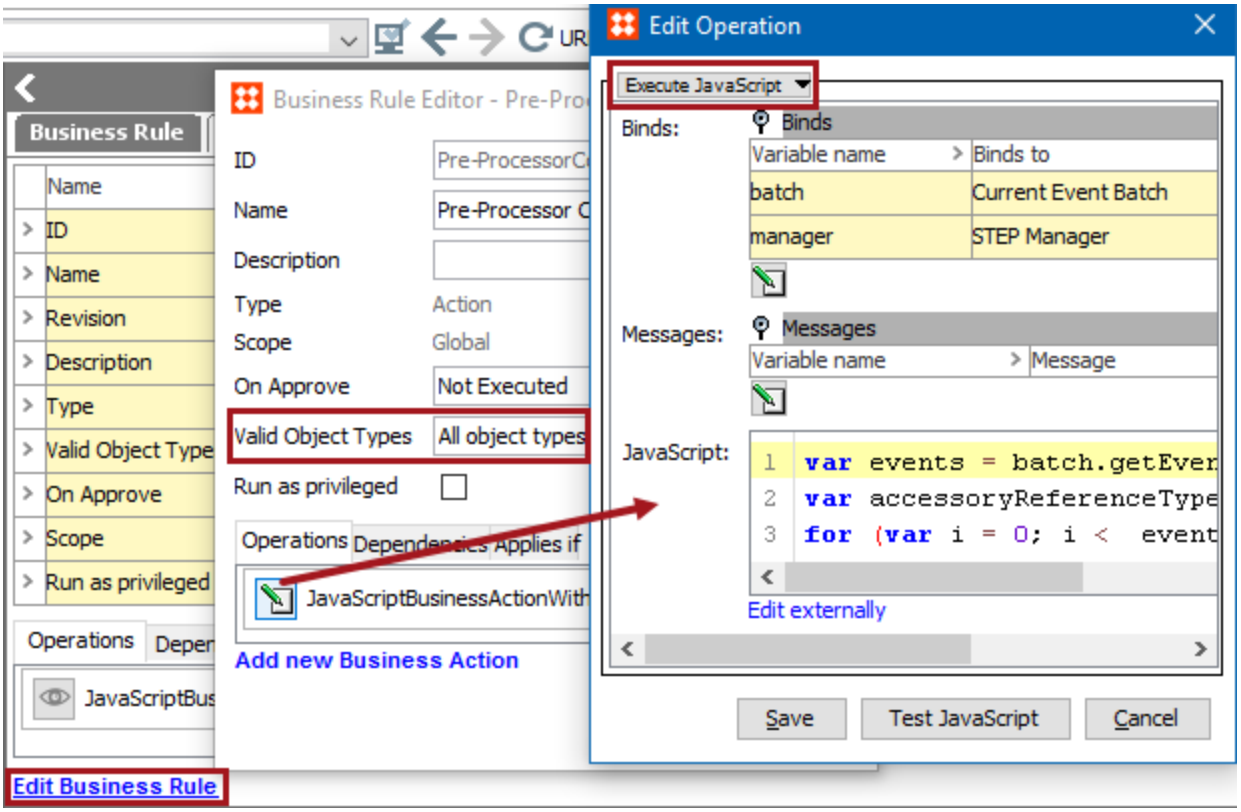
1. A business action - in this example, it will add referenced products to the products that are contained in the message.
2. An event-based OIEP configured with the business action - in this example, it is configured to deliver messages that inform a downstream system about changes to a product.

Business Action Configuration

Create a business action that is valid for 'All Object Types' (or specifically for the object types being handled by the event queue). For more information on creating business rules, see the **Specifying a Business Action Operation** topic.

For a full JavaScript code example, see the online version of this topic.

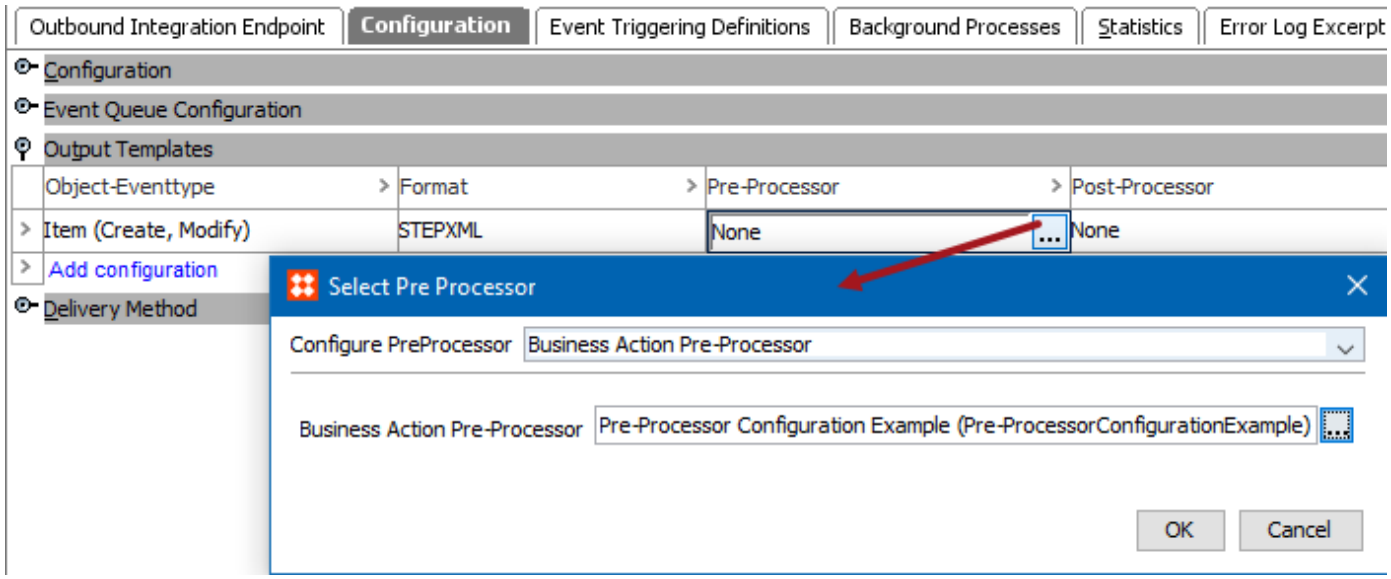
The business rule is configured in System Setup > Global Business Rules as displayed below:



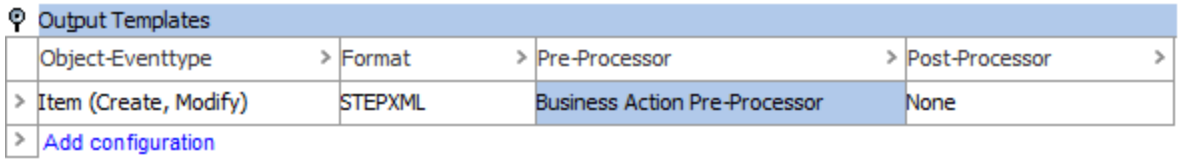
OIEP Configuration

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, see 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 Procesor dialog.
4. In **Configure PreProcessor**, 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.



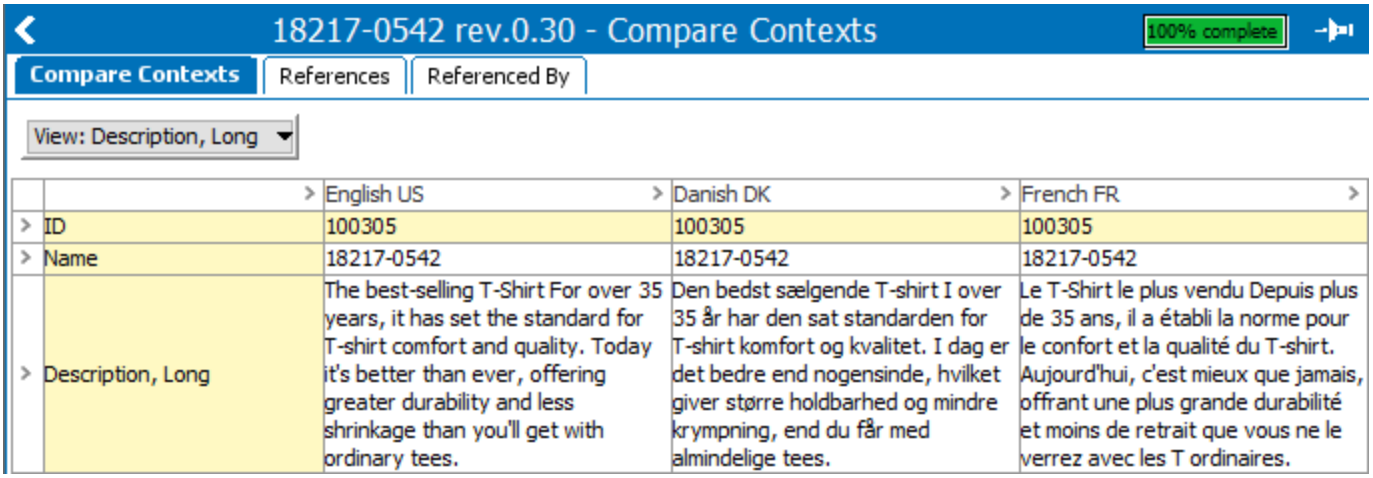
OIEP - Post-Processor - Context Splitter

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.

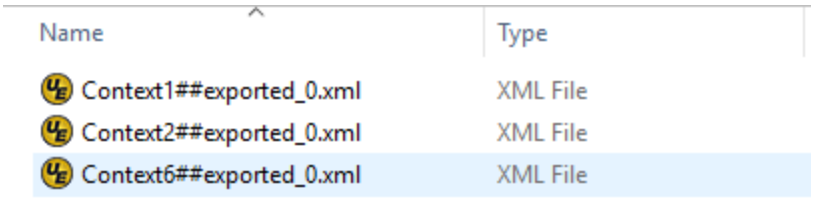
The product has a language-dependent attribute, 'Description, Long', with values defined in three contexts: English US, Danish DK, and French FR.



	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
```

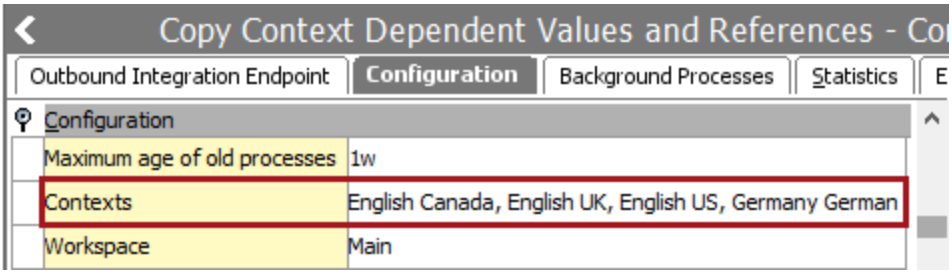
XML Output Example - Standard Cross-Context Export

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, see the **OIEP - Post-Processor - Context Splitter** topic in the **Outbound Integration Endpoints** documentation.

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.

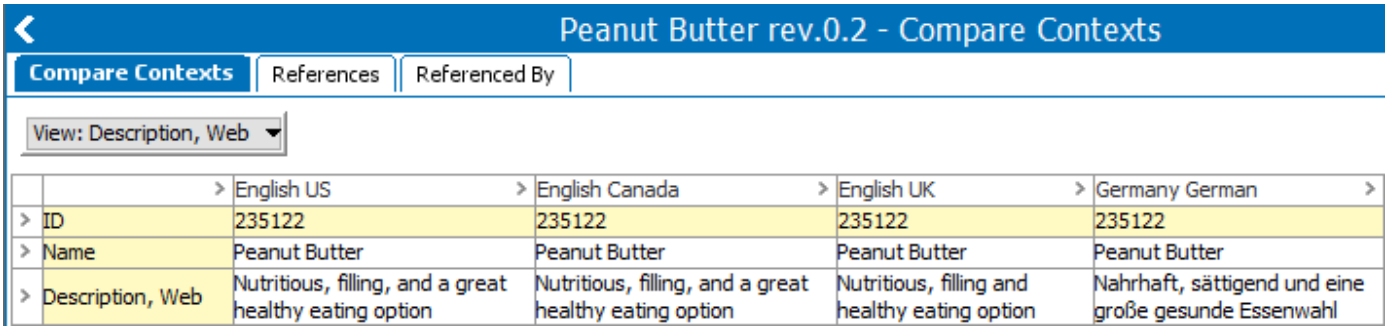
When the <ContextID> XML tag is used in combination with the <QualifierID> XML tag, it is possible for the downstream system to determine the inheritance relationship between a value (or reference) and a context. The <QualifierID> XML tag is useful when a service tries to re-import processed data into 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>
```

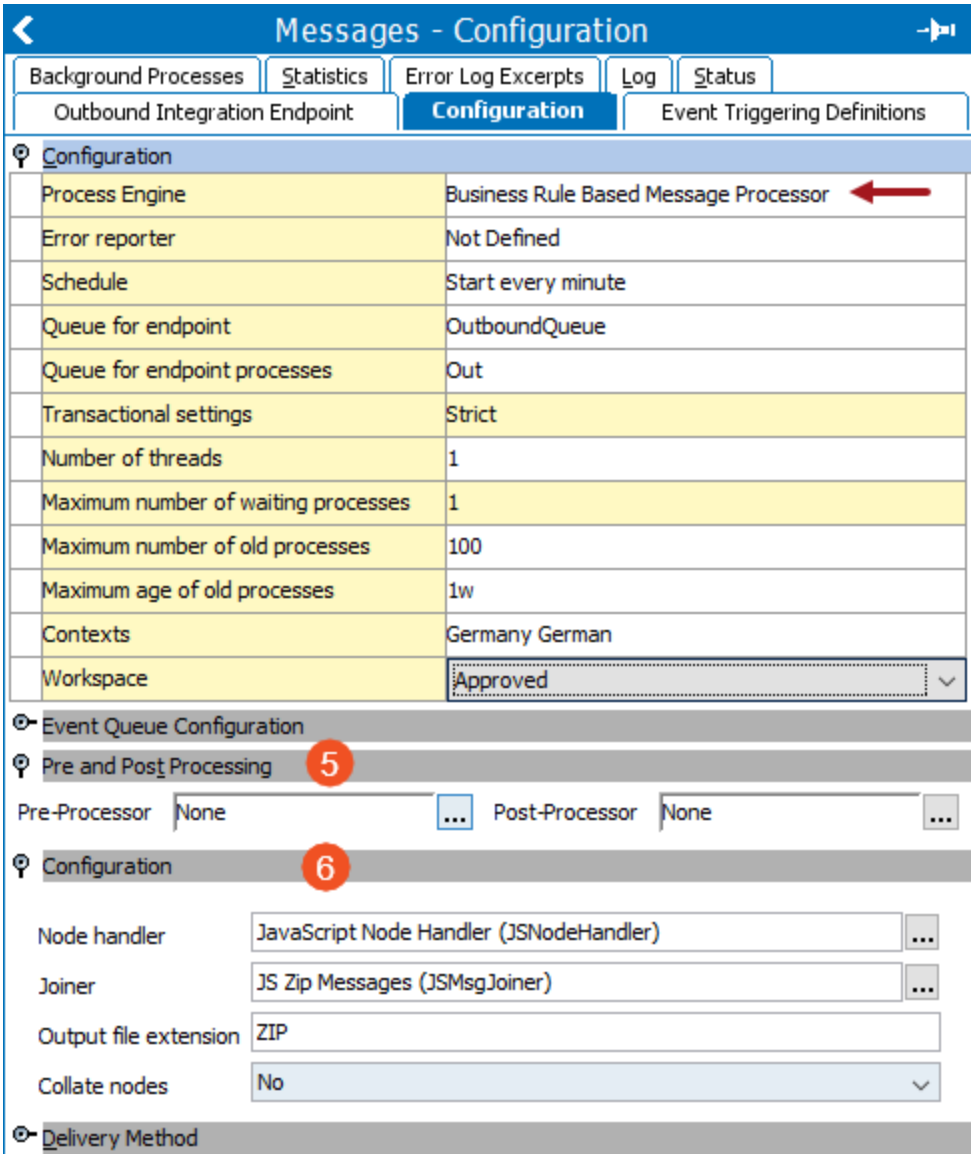
The <ContextID> tag has been added to the export along with the <QualifierID> tag, and each value—including inherited values—is explicitly exported. This means that the 'English Canada' (Context3) values are included in the export, even though it inherits from the master context. If the value is overwritten 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 Methods** topic.

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.

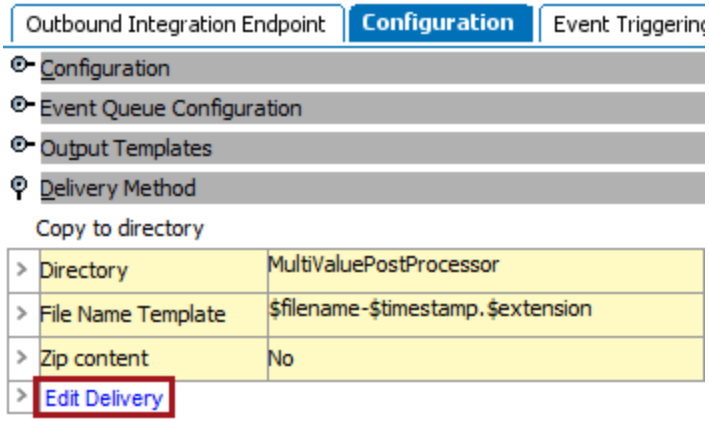
For details, see the **Configure the Pre-processor and Post-processor** section of the following topics:

- **OIEP - Select Objects - Output Templates Flipper**
- **OIEP - Event-Based - Output Templates Flipper**

OIEP - Delivery Method Flipper

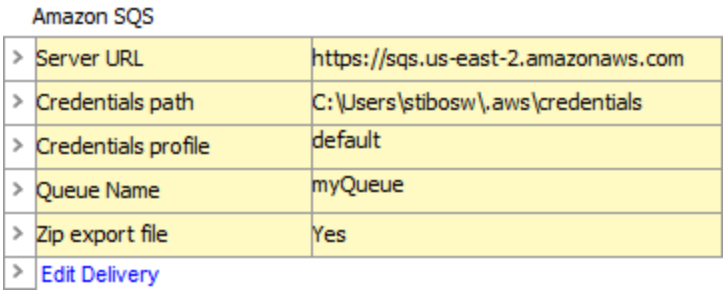
The Delivery Method flipper in an OIEP includes the same parameters for both Event-Based and Select Objects endpoints. The delivery method controls how the output is delivered to an external system.

Open the Delivery Method flipper to display the selected delivery method. As shown below, the parameters vary based on the selected delivery method. Each is described in the **OIEP Delivery Methods** topic.



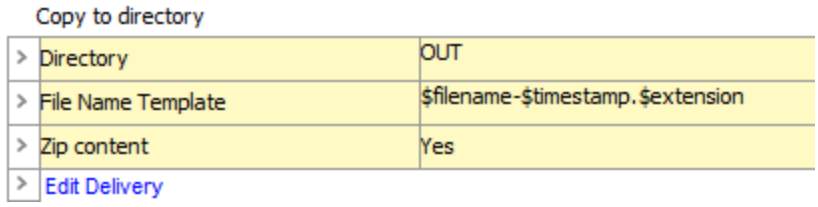
Amazon SQS

For details on this delivery method, see **Amazon SQS Delivery Method**.



Copy to Directory

For details on this delivery method, see **Copy to Directory Delivery Method**.



Deploy

For details on this delivery method, see **Deploy Delivery Method**.

Deploy

Extract to directory	workarea
Copy to file	$tmpdir/zip-$timestamp.$extension
Edit Delivery	

Email

For details on this delivery method, see **Email Delivery Method**.

Email

Email	OutboundData@customer.com
Subject text	Product data export
Body text	Please find enclosed the latest product data delivery
Zip export file	Yes
Edit Delivery	

FTP

For details on this delivery method, see **FTP Delivery Method**.

FTP

Hostname	FTPDeliveryHostServerName
Username	OIEPUser
Password	xxxxxxxx
File Name Template	$filename-$timestamp.$extension
Zip before upload	Yes
Edit Delivery	

IBM Websphere MQ SSL Delivery

For details on this delivery method, see **IBM Websphere MQ SSL Delivery Method**.

IBM Websphere MQ SSL Delivery

Connection URL	
Queue Manager	
Queue Name	
Binary Payload	No
User Name	
Password	
Key Store	
Trust Store	
Cipher Suite	
Additional Parameters	

Edit Delivery

JDBC

For details on this delivery method, see **JDBC Delivery Method**.

JDBC

Driver Location	L:/jdbc-drivers/mysql-connector-java-5.0.8-bin.jar
Driver Class	com.mysql.jdbc.Driver
Database URL	jdbc:mysql://localhost:3306/mydb
Username	root
Password	xxxx
Table Name	stepdata
Key Columns	id,datetime
Delete Key Columns	id

Edit Delivery

JMS

For details on this delivery method, see **JMS Delivery Method**.

JMS Delivery

JMS Connection Factory Class	ActiveMQInitialContextFactory
Connection Factory Name	ConnectionFactory
JMS Provider URL	tcp://ATTCM3S9:61616 =
JMS Queue	JMSQueueForSTEP
Binary Payload	No
User Name	OIEPUser
Password	xxxxxxxxx
Additional Parameters	

Edit Delivery

Dynamic JMS

For details on this delivery method, see the **Dynamic JMS Delivery Method**.

Dynamic JMS Delivery

JMS Connection Factory Name	JMSConnectionFactory
JMS Queue Name	JMS1
JNDI Properties File Name	jndi.properties
Binary Load	No

Edit Delivery

Mongo

For details on this delivery method, see **Mongo Delivery Method**.

Mongo delivery

Server(s)	mongohost1:27017,mongohost1:27018,mongohost1:27019
Raw DB prefix	raw
User name	userauth
Password	xxxxxxxxxxxx
Cluster Enabled Configuration	Yes
Use SSL to encrypt network traffic	No
Key store location	
Trust store location	
Actions	WorkflowReview

Edit Delivery

Oracle AQ

For details on this delivery method, see **Oracle AQ Delivery Method**.

Oracle AQ Delivery

Connection URL	jdbc:oracle:thin:@//server:1521/step
Queue Name	OracleAQRecQueue
Binary Payload	No
User Name	OIEPUser
Password	xxxxxxxx

Edit Delivery

Product Data Syndication

For details on this delivery method, see **Product Data Syndication Delivery Method**.

Product Data Syndication

Server URL	https://StiboPDS/
API User Name	user
API Password	xxxxxxxx

Edit Delivery

REST

For details on this delivery method, see **REST Delivery Method**.

REST

URL	http://www.Customer.com/sqlrest
User name	OIEPUser
Password	xxxxxxxxxx
Zip content	Yes

Edit Delivery

REST Direct

For details on this delivery method, see **REST Direct Delivery Method**.

REST Direct

URL	http://www.CustomerDirect.com/sqlrestdirect
User name	OIEPUser
Password	xxxxxxxxxx
Zip content	Yes
Header	
Footer	

Edit Delivery

SFTP

For details on this delivery method, see **SFTP Delivery Method**.

SFTP

Hostname	SFTPDeliveryHostServerName
Username	OIEPUser
Password	xxxxxxxxx
File Name Template	$filename-$timestamp.$extension
Zip before upload	Yes

Edit Delivery

Wiki

For details on this delivery method, see **Wiki Delivery Method**.

WikiDelivery

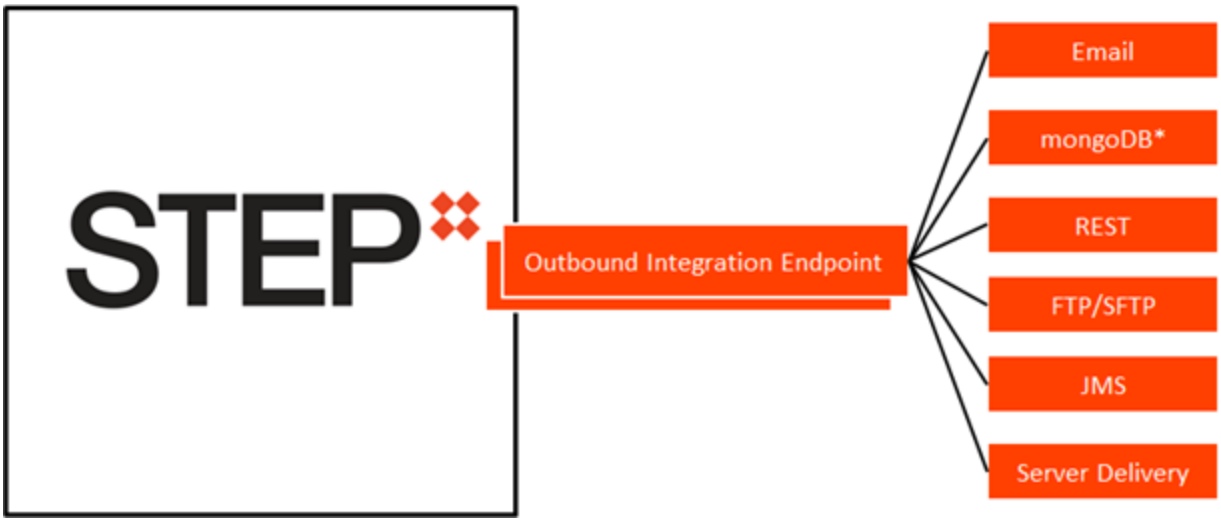
WikiMainSpace	STEP/
WikisName	wikis/xwiki/
WikiPassword	admin
WikiRestSpace	rest/
WikiUrl	http://vp487.vps.ohs.ca/
WikiUsername	admin

Edit Delivery

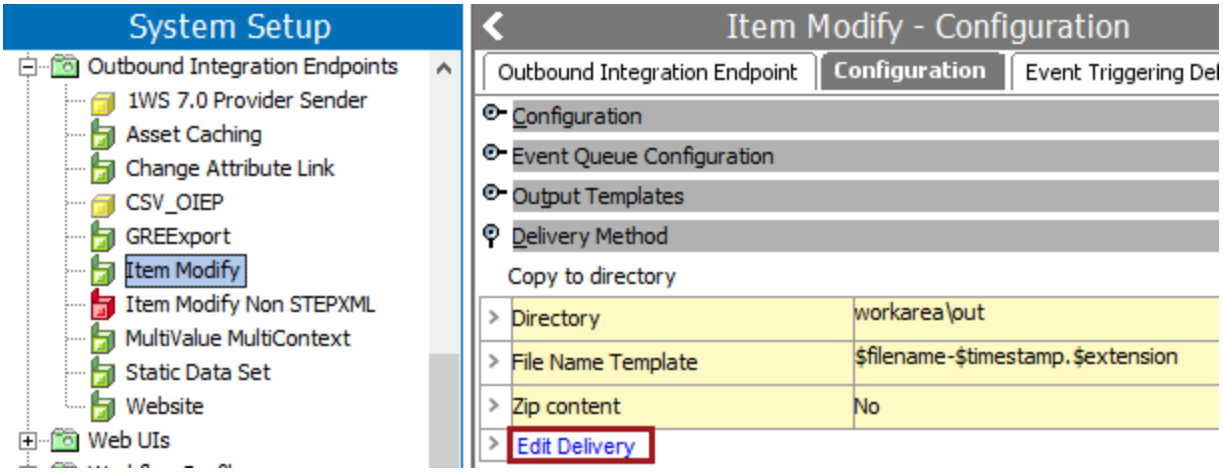
OIEP Delivery Methods

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, see **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.



In an OIEP, the delivery method is set in the editor on the Configuration tab, under the Delivery Method flipper.



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). See **Amazon SQS 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. See **Copy to Directory Delivery Method** documentation.
Deploy	Delivers files to a directory on the application server. Does not allow the delivery to be zipped. See **Deploy Delivery Method** documentation.
Email	Delivers files as an email attachment and allows a zipped delivery. See **Email Delivery Method** documentation.
FTP	Delivers files using file transfer protocol (FTP). See **FTP Delivery Method** documentation.
GDSN Data Pool	The GDSN solution must be implemented fully before the GDSN Data Pool method is available and functional. See **GDSN Provider** documentation.
GDSN Receiver Data Pool	The GDSN solution must be implemented fully before the GDSN Receiver Data Pool method is available and functional. See **GDSN Receiver** documentation.
IBM Websphere MQ SSL	IBM Websphere MQ SSL, uses MQ series Secure Sockets Layer, and enables data exchange across IBM and non-IBM platforms. See **IBM Websphere 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. See **JDBC Delivery Method** documentation.
JMS	The standard JMS delivery options (Apache Active MQ, Websphere MQ, and Oracle AQ) can be used to deliver messages to a JMS receiver without further customizations. See **JMS Delivery Method** documentation.
Dynamic JMS	The standard Dynamic JMS delivery options (Apache Active MQ, Websphere 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. See the **Dynamic JMS 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

Method	Description
	systems. See **Mongo Delivery Method** documentation.
Oracle AQ	Oracle Advanced Queuing (Oracle AQ) enables messages to be exchanged between two systems. See **Oracle AQ Delivery Method** documentation.
Product Data Syndication	Product Data Syndication sends data to the Product Data Syndication (PDS) platform via the default PDS Outbound Integration Endpoint and the API. See **Product Data Syndication Delivery Method** documentation.
REST	The REST Delivery Method delivers a call-back URL to the REST service and does not include actual STEP data. See **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. See **REST Direct Delivery Method** documentation.
SFTP	Delivers a file using the Secure File Transfer Protocol (SFTP). See **SFTP Delivery Method** documentation.
Wiki	Delivers metadata to the metadata wiki platform. See **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.

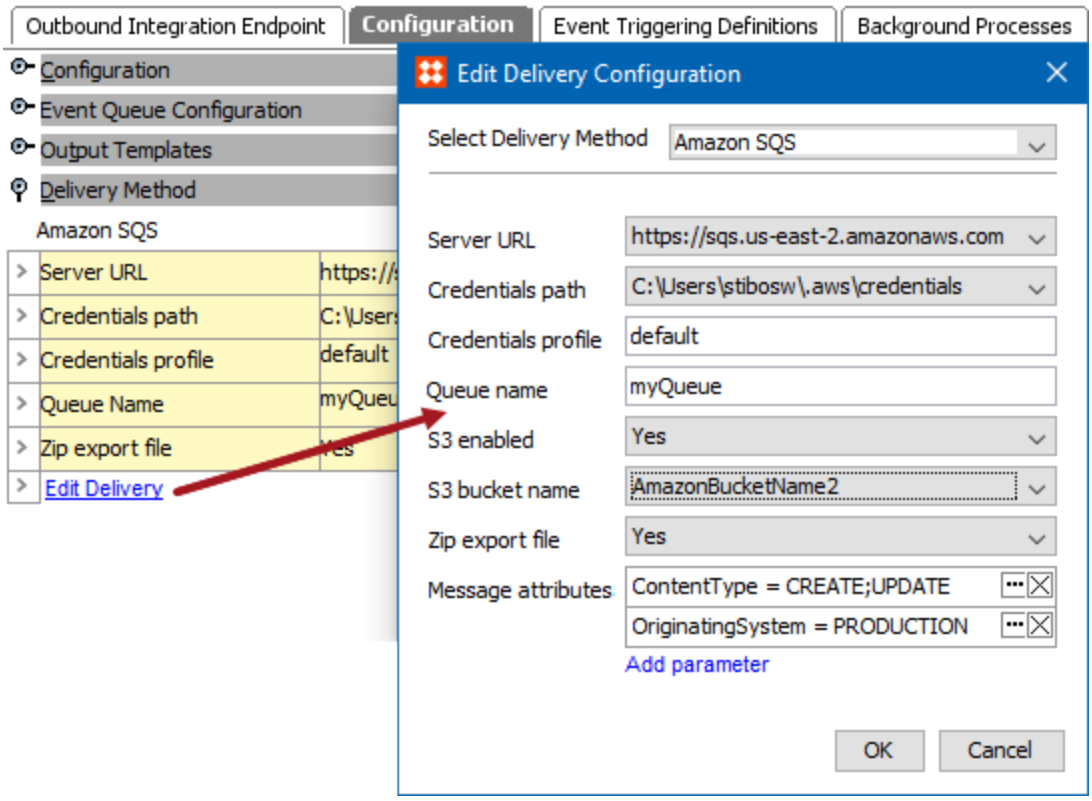
6. If necessary, establish the Amazon S3 storage account and create the required buckets. For details, see Amazon S3 on the web.
7. If message attributes are required, create up to seven (7) string Message Attributes on the Amazon website. The message attribute names in Amazon correspond to the Key parameters 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.

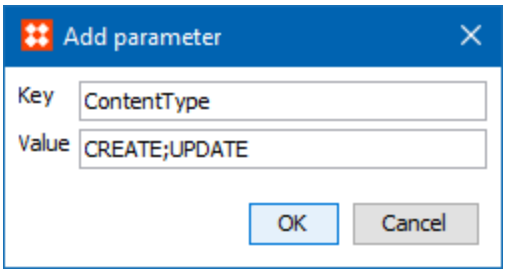


1. For **Select Delivery Method**, choose **Amazon SQS**.
2. In **Server URL**, select the URL that points to the SQS server where the data will be delivered.
3. In **Credentials path**, select the path to the desired SQS credentials file.
4. In **Credentials profile**, enter the profile name included 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 (see 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.

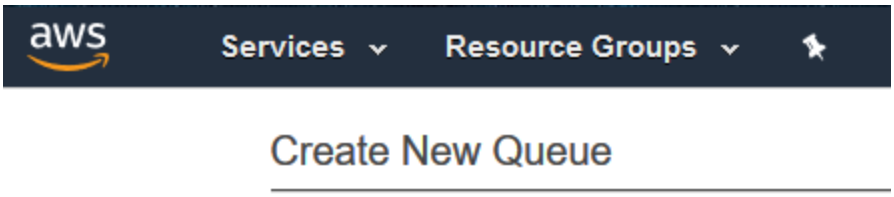


10. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

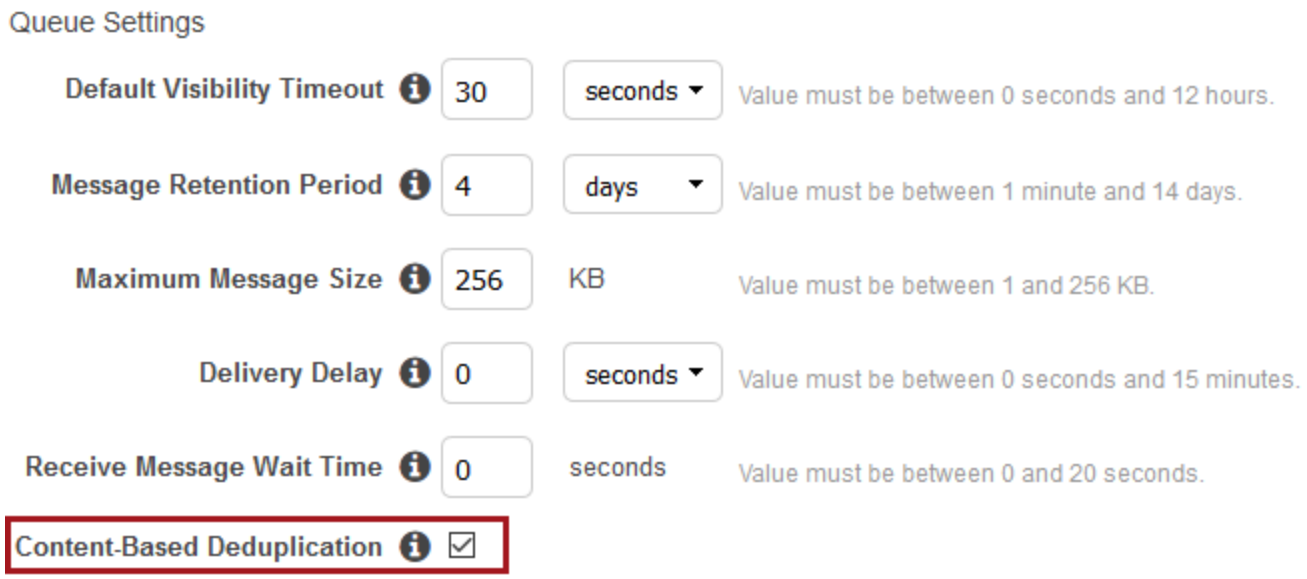
Amazon SQS Queue Configuration

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.



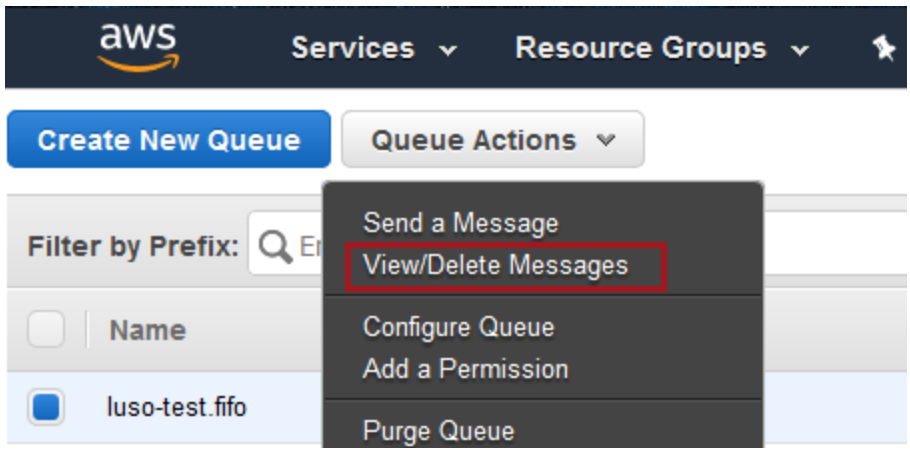
4. For **Queue Name**, type a queue name including the '**.fifo**' extension.
5. For the **type of queue** section, select **FIFO Queue**.
6. Click the **Configure Queue** button at the bottom of the page to display the default Queue Attributes parameters.
7. Check the **Content-Based Deduplication** parameter box to enable the queue.



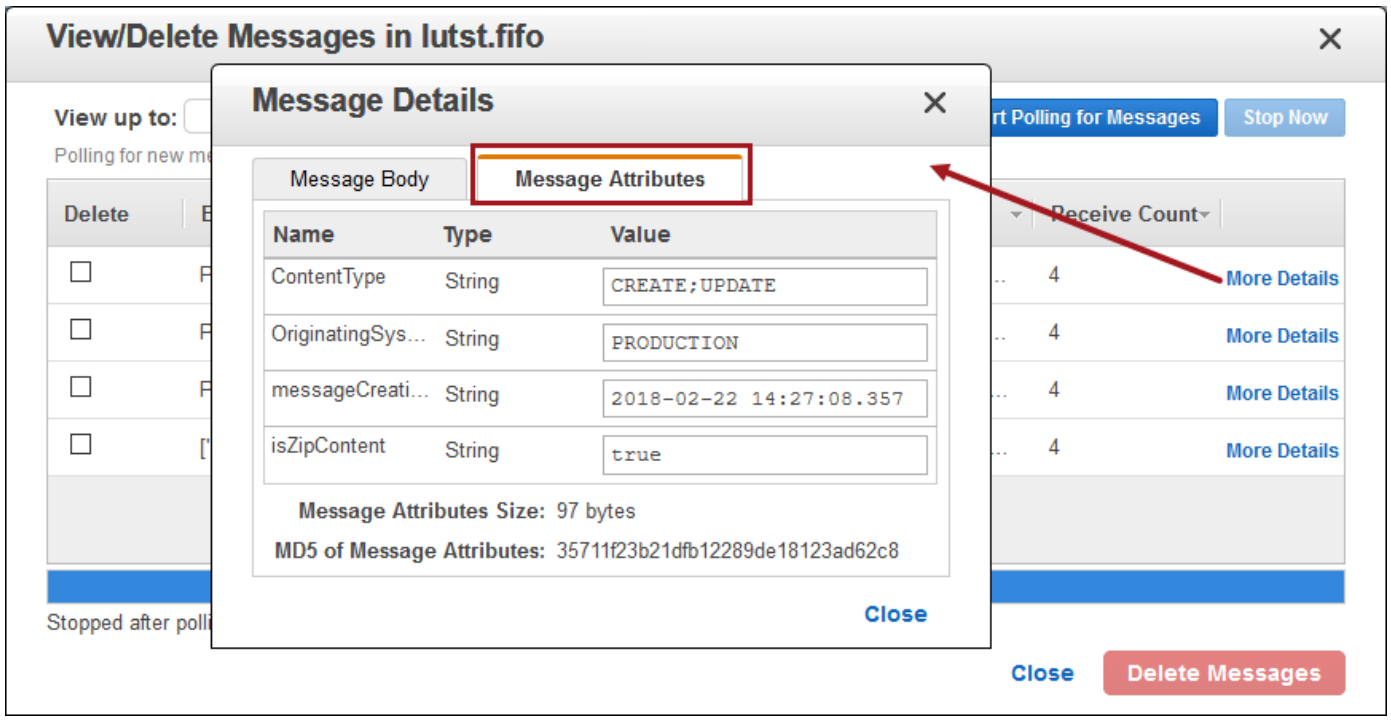
8. Click the **Create Queue** button. The new queue is displayed in the list.

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.

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.

For more information, see the **Deploy Delivery Method** documentation.

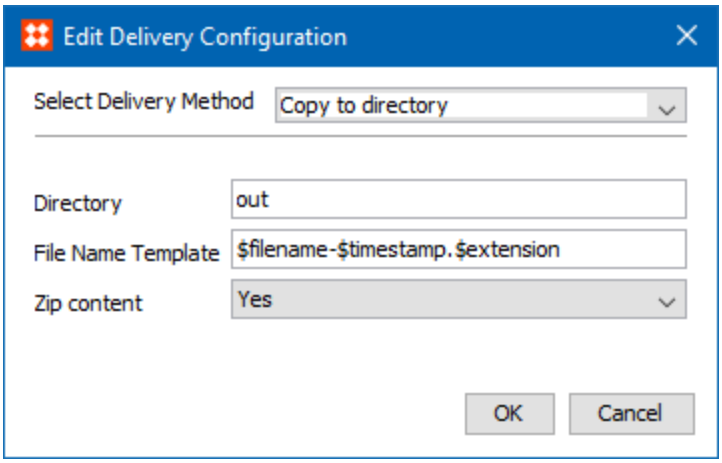
Prerequisites

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**, enter the file naming template to be used for the deliveries. Each element of the template (indicated by the $ character) is replaced with text as defined below. The template allows you to format the exported file name as required by external systems. The default template is: $filename-$timestamp.$extension. Note that the hyphen (-) and the period (.) are actual characters that are used to build the complete name of the exported file.

- **$filename** For event-based OIEPs, this variable is replaced with 'csv' for CSV format, 'excel' for Excel format, and the event ID for STEPXML format exports. For Select Object OIEPs, $filename is replaced with 'exported.' Text can be added before or after the $filename template, and $filename can also be omitted altogether. When exporting multiple contexts, '##' is used as a separator within the filename to distinguish the context name for each output.
- **$timestamp** This variable is replaced with the time in milliseconds between January 1, 1970, and the time when the file is uploaded, which allows for a unique identifier. For example, 1479230247017 is used for an export on 15 NOV 2016.

  Alternatively, the following options can be used to generate a readable date and can include static characters to separate the timestamp elements. Individual elements can be ordered as required, as shown below.

  $timestamp(ddMMyyyy) displays day, month, and year and is output '15112016' for 15 NOV 2016.

  $timestamp(yyyy-MM-dd_HH.mm.ss) displays year, month, day, hour, minutes, and seconds 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 the File Name Template and the extension .ZIP.
   - **No** uses the File Name Template 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.

For more information, see the **Copy to Directory Delivery Method** documentation.

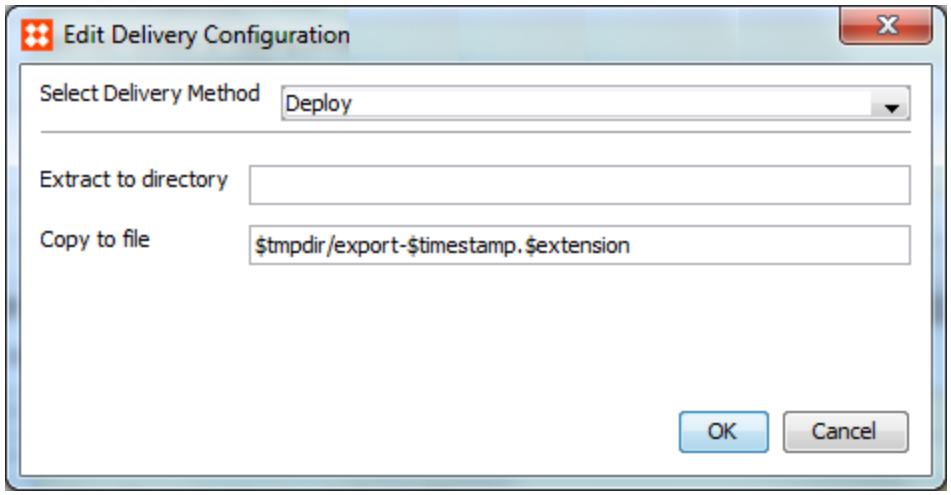
Prerequisites

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 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 milliseconds between January 1, 1970, and the time when the file is uploaded, which allows for a unique identifier. For example, 1479230247017 is used for an export on 15 NOV 2016.

  Alternatively, the following options can be used to generate a readable date and can include static characters to separate the timestamp elements. Individual elements can be ordered as required, as shown below.

  $timestamp(ddMMyyyy) displays day, month, and year and is output '15112016' for 15 NOV 2016.

  $timestamp(yyyy-MM-dd_HH.mm.ss) displays year, month, day, hour, minutes, and seconds 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.

Email Delivery Method

The email delivery option is very straight forward and, like 'Copy to directory', it allows the delivery to be zipped. It is typically used to test the export output, or to deliver a small export to one or more persons as a one-time occurrence.

This delivery method is available in Export Manager and in OIEPs. For information on a parameter, hover over the parameter to display help text.

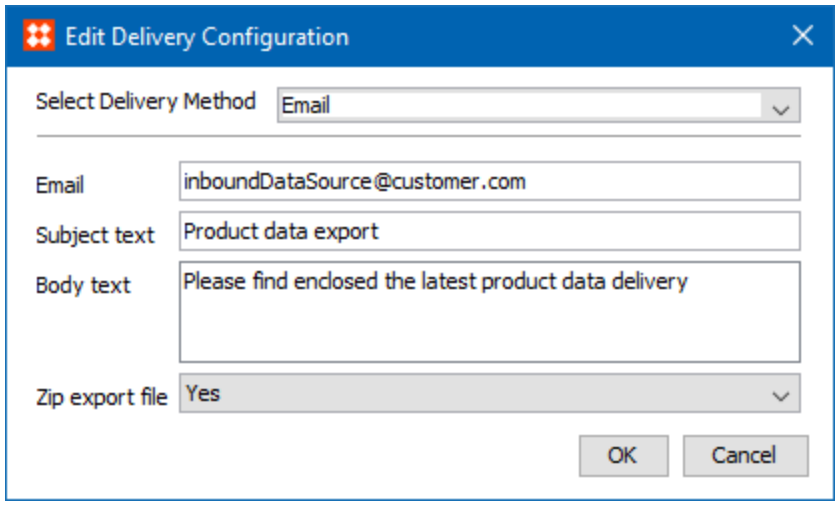
Prerequisites

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.

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.
3. For the **Subject text** parameter, enter the text that will display for the email subject line.
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 exported file.
   - **No** does not zip the exported file.
6. 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 also available in Export Manager.

The preferred method for FTP delivery is the SFTP secure delivery method. For information on the OIEP SFTP delivery method, see **SFTP Delivery Method**.

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]
```

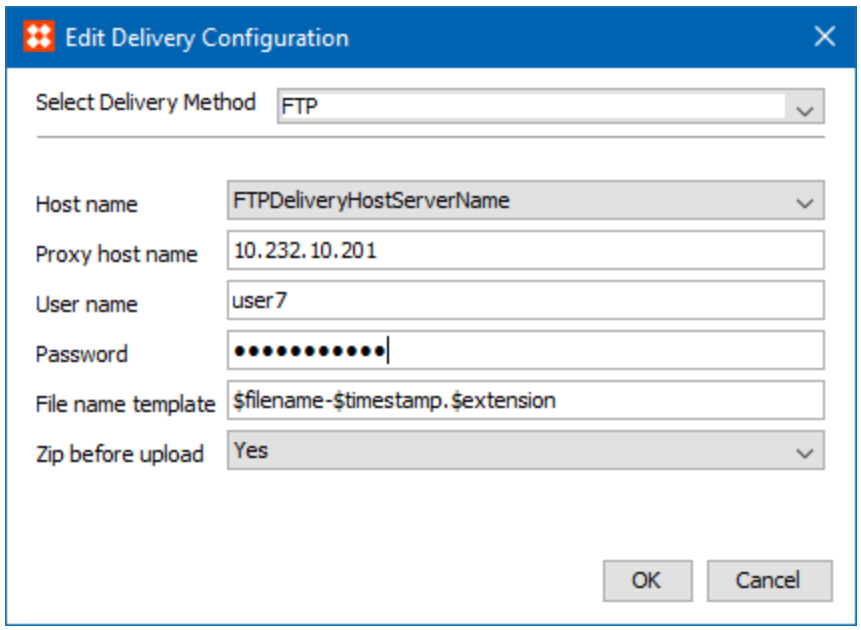
Using the host name shown in the image below, the entry in the properties file would be:

```
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, static text, or use a combination. The default template is $filename-$timestamp.$extension and creates a file name like 'GenericXML-1490813266535.xml' based on the settings of the OIEP. Each variable is described below:
   - **$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. For select object OIEPs, $filename is replaced with 'exported.'
   - **$timestamp** This variable is replaced with the time in milliseconds between January 1, 1970, and the time when the file is uploaded, which allows for a unique identifier. For example, 1479230247017 is used for an export on 15 NOV 2016.

     Alternatively, the following options can be used to generate a readable date and can include static characters to separate the timestamp elements. Individual elements can be ordered as required, as shown below.

     $timestamp(ddMMyyyy) displays day, month, and year and is output '15112016' for 15 NOV 2016.

     $timestamp(yyyy-MM-dd_HH.mm.ss) displays year, month, day, hour, minutes, and seconds 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 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. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

IBM Websphere MQ SSL Delivery Method

For information on connecting to IBM Websphere MQ in a non-SSL way, see **JMS Delivery Method**. This delivery option is only available in OIEPs.

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 **WSMQSSLProviderURL** property. Use the [host]:[port]/[channel] format as defined below:
   - [host] = hostname or IP of the Websphere MQ server
   - [port] = port number for the channel
   - [channel] = name of the channel
2. Prior to configuration, clicking the **Queue Manager** 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 **WSMQSSLQueueManager** property.
3. 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 **WSMQSSLQueue** property.
4. Prior to configuration, clicking the **Key Store** 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 **WSMQSSLKeyStoreLocation** property.

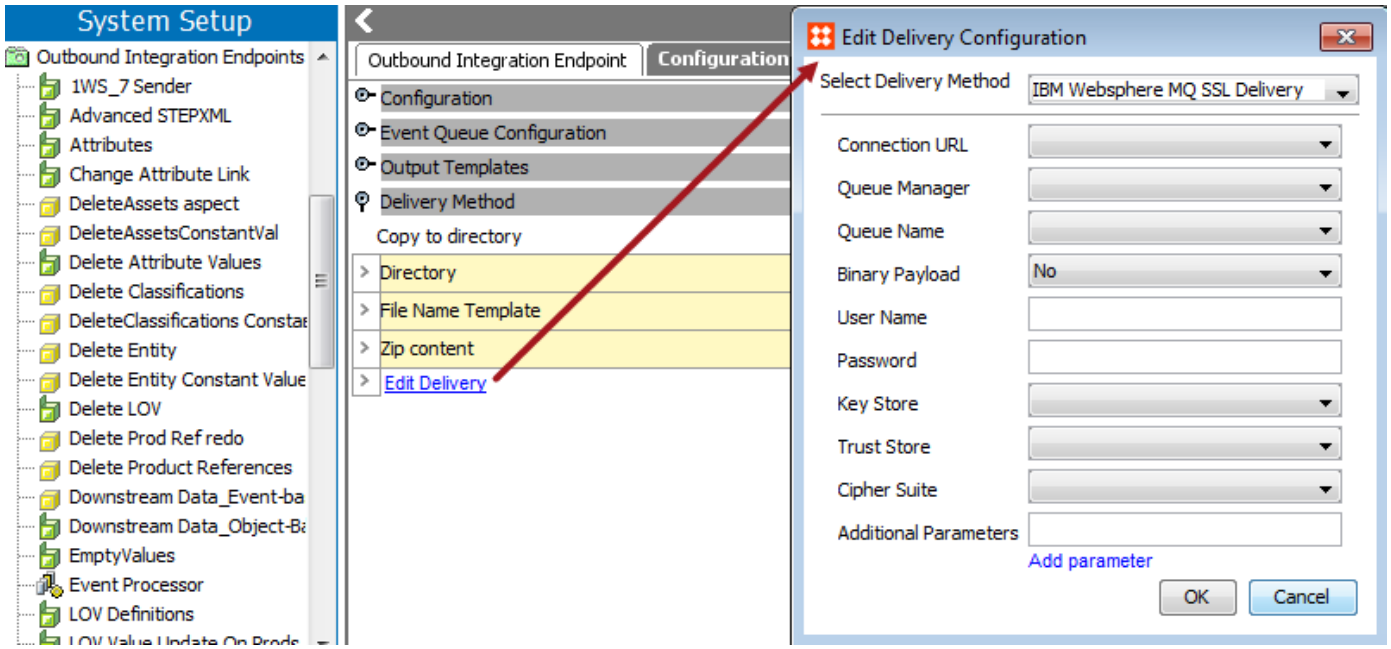
   Use the jks format with the personal certificate for the Queue Manager. To generate this, consult the manual for IBM Websphere MQ. The password for the key store must be configured in sharedconfig.properties file using the **WSMQSSLKeyStorePassword** property.
5. Prior to configuration, clicking the **Trust Store** 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 **WSMQSSLTrustStoreLocation** property.

   This can be the same file as key store. To generate this, consult the manual for IBM Websphere MQ. The password for the trust store must be configured in sharedconfig.properties or config.properties using **WSMQSSLTrustStorePassword**.
6. Prior to configuration, clicking the **Cipher Suite** 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 **WSMQSSLCipher Suite** property. STEP is running on non-IBM JRE, so this must be the same value as configured in the Queue Manager.

Configuration

For information on a parameter, hover over the parameter field to display help text.

1. In the **Select Delivery Method** field, choose **IBM Websphere 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 that will be used to log onto the JMS provider.
7. In **Password**, if required, enter the password that will be used to log onto the JMS provider.
8. In **Key Store**, select the keystore in jks format, with the personal certificate for the Queue Manager.
9. In **Trust Store**, select the trust store with the CA for the Queue Manager.
10. In **Cipher Suite**, set to the same value as SSL CipherSuite in Websphere MQ.
11. If **Additional Parameters** are required, click Add parameter, then enter the Key and Value. For possible keys and values, consult the manual for IBM Websphere MQ.
12. 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 Endpoint (OIEP).

For more information on the Export Manager option, see **JDBC Delivery Method** in the **Export Manager** documentation.

Prerequisites

Important: For complete setup requirements, see 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.

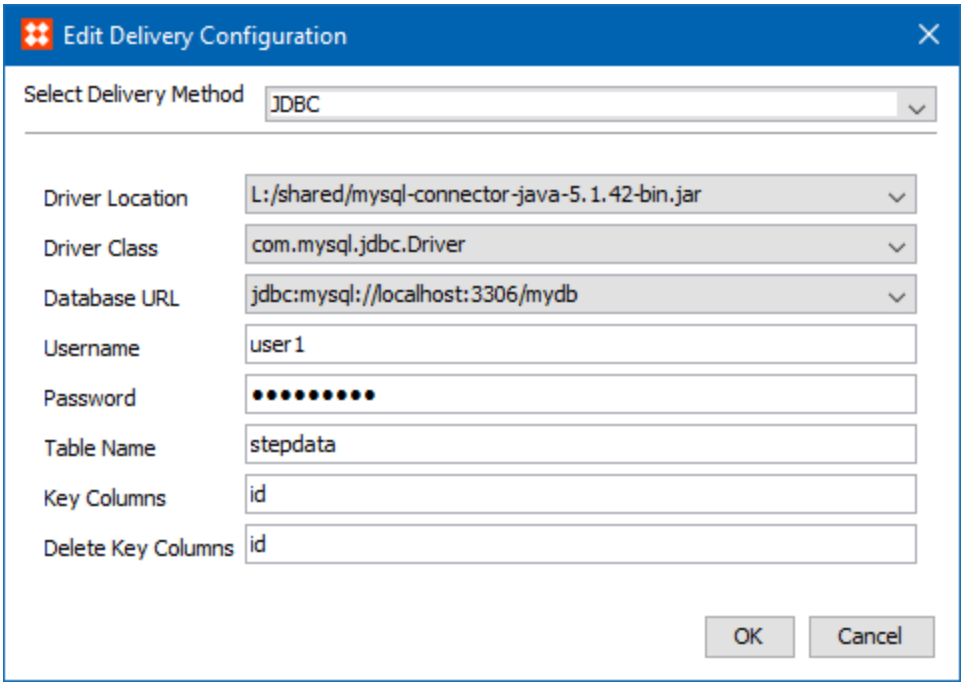
   ```
   JDBCDeliveryPlugin.URL.1 = jdbc:mysql://localhost:3306/mydb
   ```

   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. 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.

Prerequisites

Changes to the properties file, outlined below, are implemented when the server is restarted.

See 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':

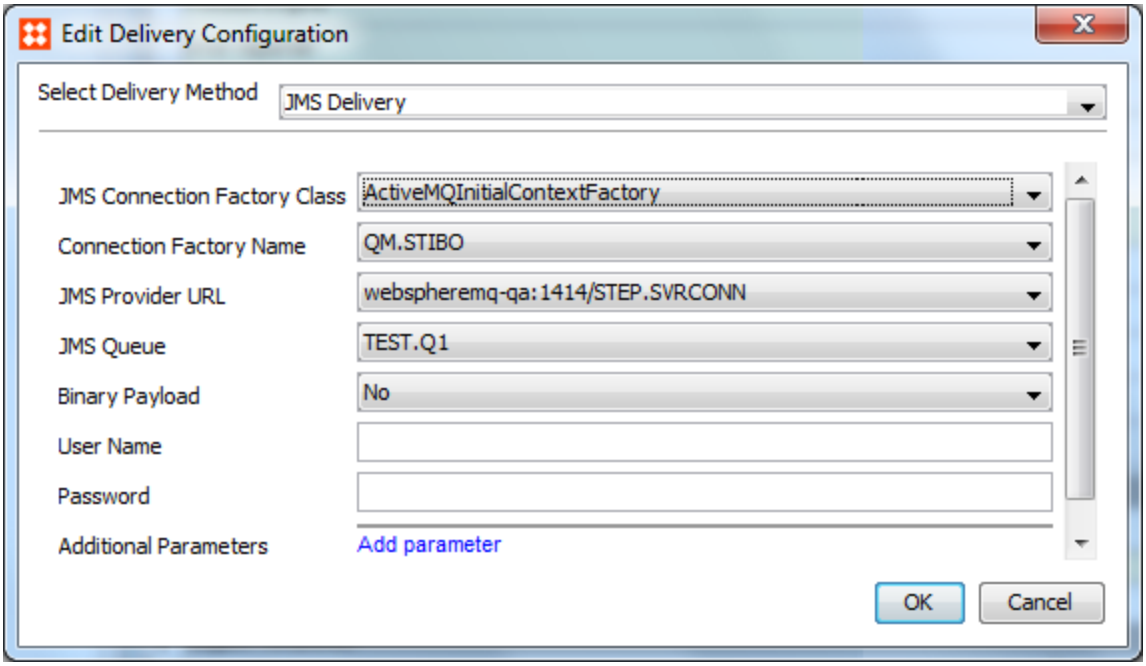
   ```
   JMSProviderURL=qa=http://step-qa,stage=http://step-stage
   ```

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 no longer 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.

Sharedconfig.properties File Examples

Each of these properties has a key/value pair as a value:

```
JMSConnectionFactoryName=FirstFactoryName=MyFactoryName

JMSProviderURL=ProviderURL=https://myproviderurl.com/url

JMSQueue=Queue=Myqueuename
```

Each of the properties can contain more than one value in a comma separated list:

```
JMSConnectionFactoryName=FirstFactoryName=MyFactoryName1,SecondFactoryName=MyFa
ctoryName2

JMSProviderURL=FirsProviderURL=https://myproviderurl.com/url,SecondProviderURL=
https://myproviderurl2.com/url

JMSQueue=Queue1=Firstqueuename,Queue2=Secondqueuename2
```

Azure Message Bus

In the JNDI properties file, add properties to set the context factory and map queues, setting connectionfactory = SBCF as shown below:

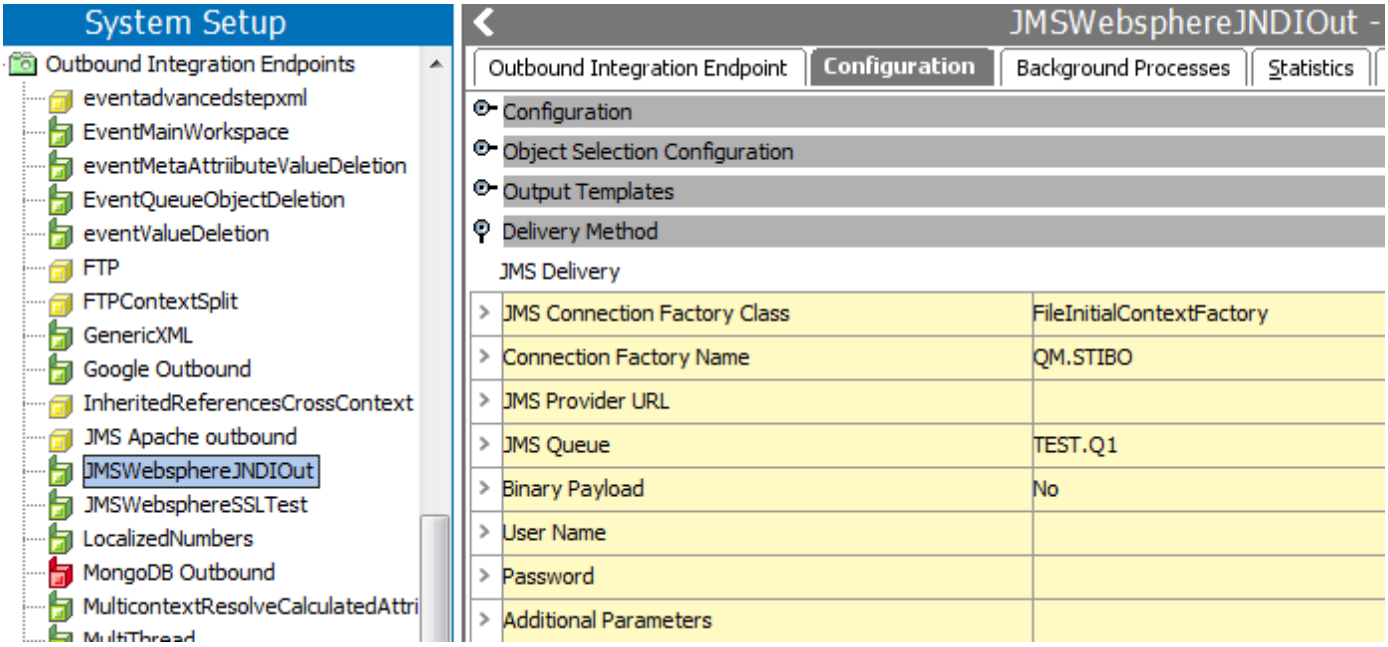
```
connectionfactory.SBCF = failover://
(amqps://khpetest.servicebus.windows.net)?jms.prefetchPolicy.all=10&failover.maxRecon
nectAttempts=20
```

Register queues in JNDI using the form:

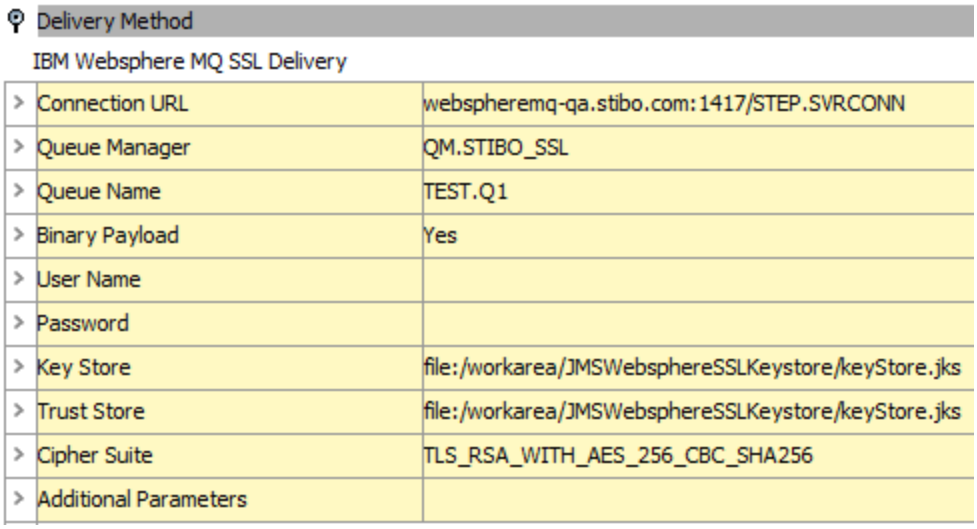
- queue.[jndi_name] = [physical_name]
- topic.[jndi_name] = [physical_name]
- queue.QUEUE = queue1

For more information on the Qpid JMS client library, see https://github.com/MicrosoftDocs/azure-docs/blob/master/articles/service-bus-messaging/service-bus-java-how-to-use-jms-api-amqp.md

FileInitialContextFactory JMS Delivery Configuration Example



JMS Websphere Delivery Using SSL Configuration Example



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	

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.

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.

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.

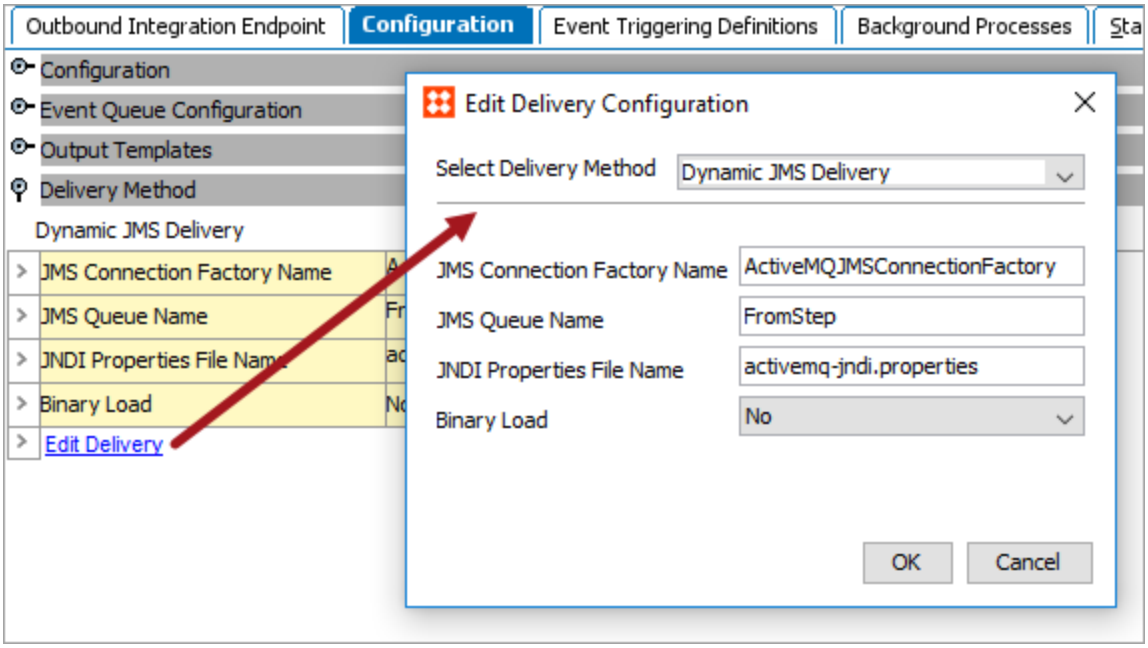
To enable the Dynamic JMS Delivery option, the case-sensitive configuration property 'JMS.Classpath' must be set in sharedconfig.properties on the STEP application server and should point to an existing directory accessible from all application servers. Client libraries and JNDI files (Java Naming and Directory Interface) must be placed in this directory. See 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 connectionfactory and queue in the OIEP delivery method.

Azure Message Bus

In the JNDI properties file, add properties to set the context factory and map queues, setting connectionfactory = SBCF as shown below:

```
connectionfactory.SBCF = failover://
(amqps://khpetest.servicebus.windows.net)?jms.prefetchPolicy.all=10&failover.maxReconnectAttempts=20
```

Register queues in JNDI using the form:

- queue.[jndi_name] = [physical_name]
- topic.[jndi_name] = [physical_name]
- queue.QUEUE = queue1

For more information on the Qpid JMS client library, see 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 and the queue have the same name).

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'

Apache RabbitMQ 1.11.1

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=10.0.42.126
ConnectionFactory/RefAddr/3/Type=host
ConnectionFactory/RefAddr/3/Encoding=String
ConnectionFactory/RefAddr/4/Content=pimconsumer
ConnectionFactory/RefAddr/4/Type=virtualHost
ConnectionFactory/RefAddr/4/Encoding=String
# rabbitmqQueue01 Queue
rabbitmqQueue01/ClassName=javax.jms.Queue
rabbitmqQueue01/FactoryName=com.rabbitmq.jms.admin.RMQObjectFactory
rabbitmqQueue01/RefAddr/0/Content=jms/Queue
rabbitmqQueue01/RefAddr/0/Type=name
rabbitmqQueue01/RefAddr/0/Encoding=String
rabbitmqQueue01/RefAddr/1/Content=javax.jms.Queue
rabbitmqQueue01/RefAddr/1/Type=type
rabbitmqQueue01/RefAddr/1/Encoding=String
rabbitmqQueue01/RefAddr/2/Content=com.rabbitmq.jms.admin.RMQObjectFactory
rabbitmqQueue01/RefAddr/2/Type=factory
rabbitmqQueue01/RefAddr/2/Encoding=String
rabbitmqQueue01/RefAddr/3/Content=rabbitmqQueue01
rabbitmqQueue01/RefAddr/3/Type=destinationName
rabbitmqQueue01/RefAddr/3/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
```

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, see https://docs.mongodb.com/manual/ or the **MongoDB Adapter Setup Quick Guides** documentation.

For more information on the Mongo Delivery Method, see the **Mongo Delivery Method Elements** documentation.

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, see 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 first configuration entry must use number 1, and each additional entry must be numbered sequentially (MongoDB.Server.Configuration.1, MongoDB.Server.Configuration.2, MongoDB.Server.Configuration.3, etc.). 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.

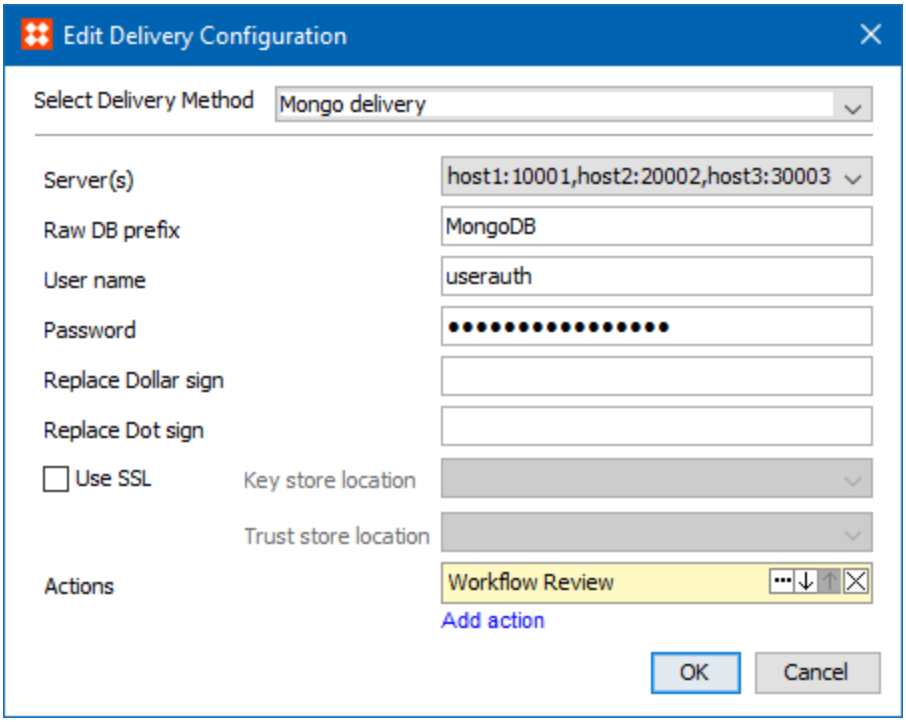
   The password for the key store is configured in sharedconfig.properties on the STEP application server, using the case-sensitive **MongoDB.SSLTrustStorePassword** property. For example:

   ```
   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**.



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:

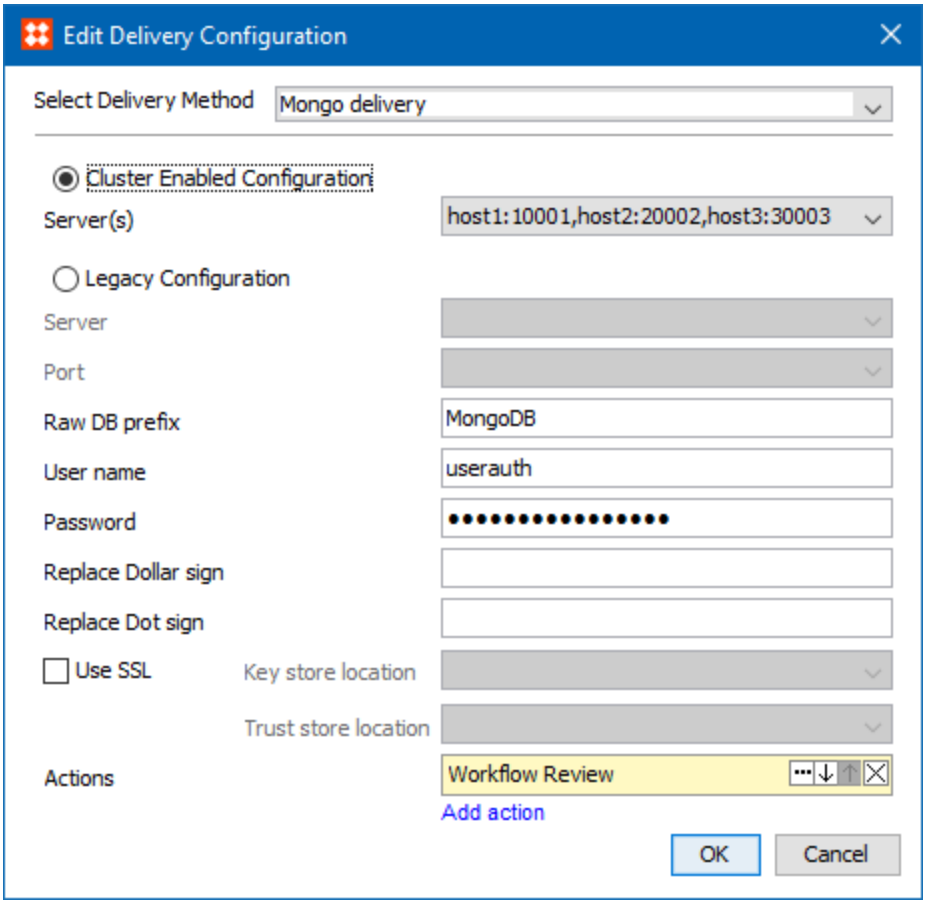
1. The OIEP is event-based, verified by viewing that the Configuration tab includes the Event Queue Configuration flipper. For more information, see the **OIEP - Event-Based - Event Queue Configuration Flipper** topic.
2. In the Event Queue Configuration flipper, the Queue Status parameter is set to Read Events.
3. In the Output Templates flipper, the Object-Eventtype column includes 'Delete' as an event type. For more information, see 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, see the **Event-Based OIEP Triggered by Deleting Products, Classifications, and Assets** section of the **Delete Objects in STEPXML** topic.

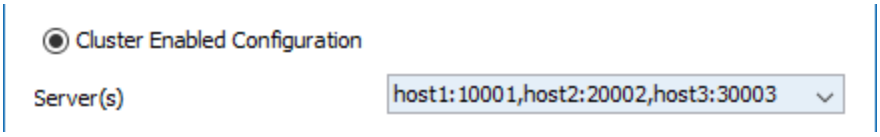
Legacy Configuration

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.

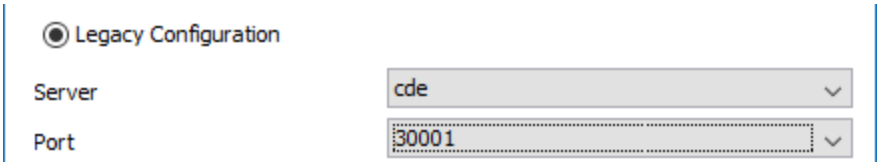
Note: Upgrading the sharedconfig.properties file to modify all configurations using the legacy properties to the cluster enabled property, and then removing the legacy properties completely, results in the radio button options being removed from the dialog, as shown in the image at the top of this topic.



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:

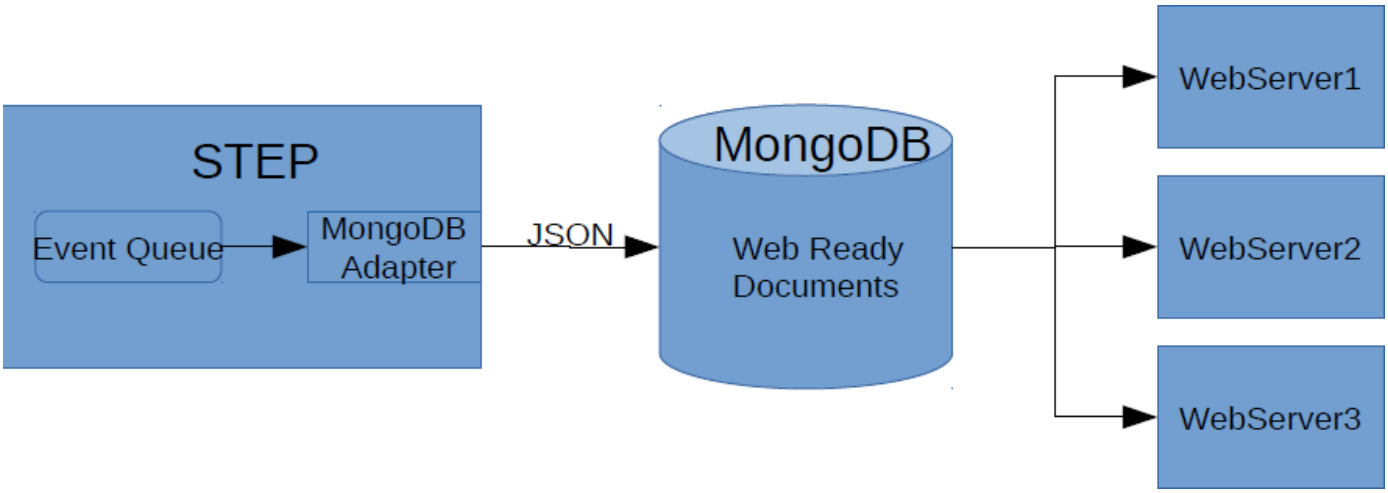
```
MongoDB.Server=1=vps124887.vps.ovh.ca,2=cde,3=rty

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.

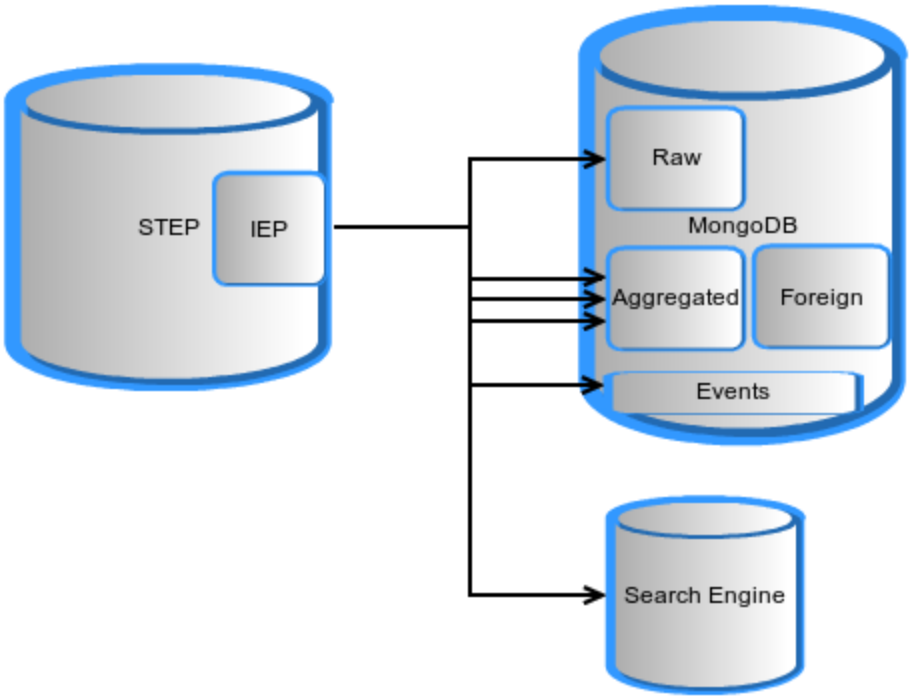
The Mongo adapter converts STEPXML to JSON and keeps a set of 'raw' collections in MongoDB that are in sync with STEP. Additionally, it allows you to add JavaScript triggers to maintain aggregated collections.

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:



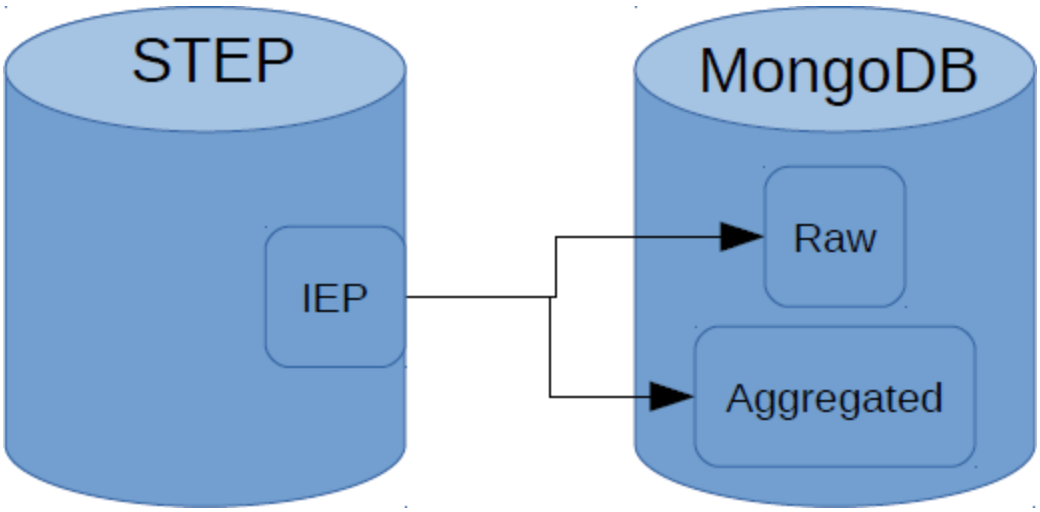
The STEP Mongo Adapter ensures that the raw collections are a one-to-one mirror of data in STEP. The adapter does not embed nodes found through references or child associations. These nodes are stored separately in the raw collections.

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
asset	Exported assets
classification	Exported classifications

Raw Collection	Purpose
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, see the MongoDB documentation on the web.

For an expanded example of the basic conversion using STEP Mongo Adapter, see 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, see the **Applies If Tab** section of the **Editing a Business Rule** 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, see: 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, see 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:

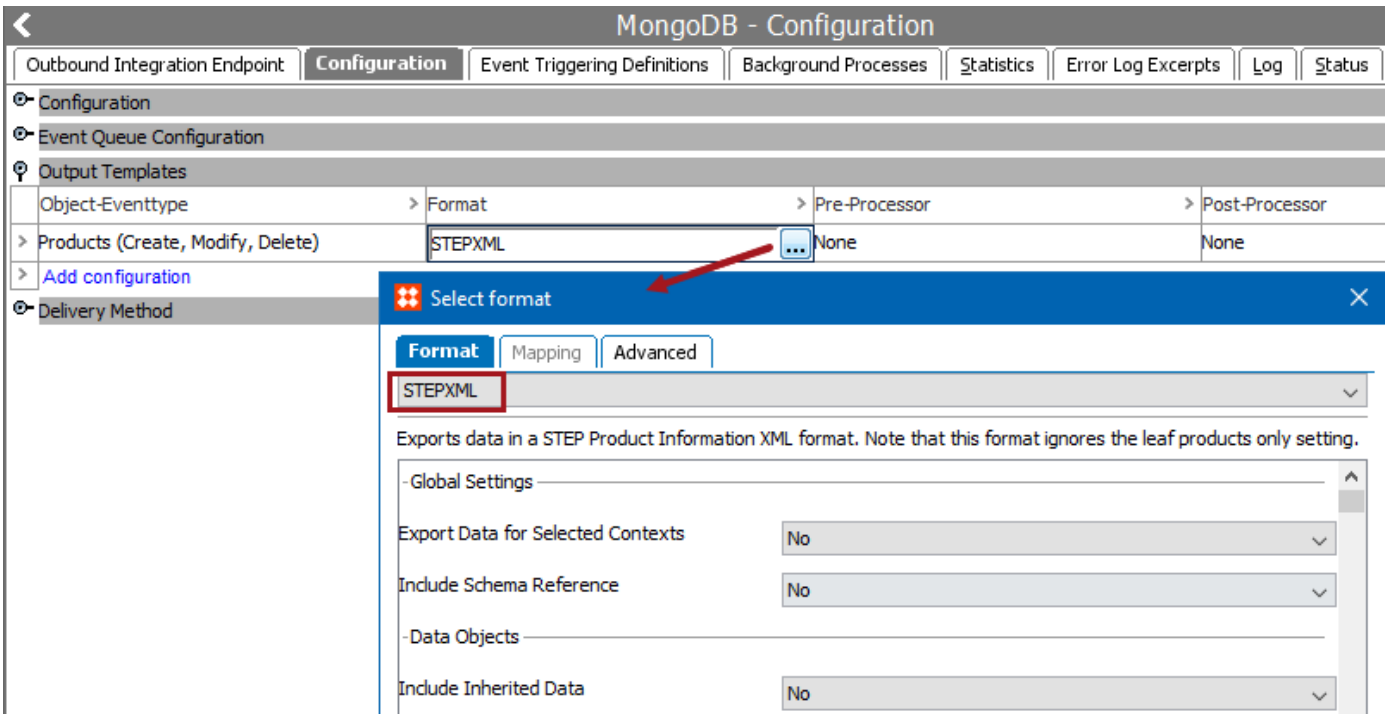
   ```
   MongoDB.Server = 1=Server1, 2=Server2
   ```

   ```
   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.

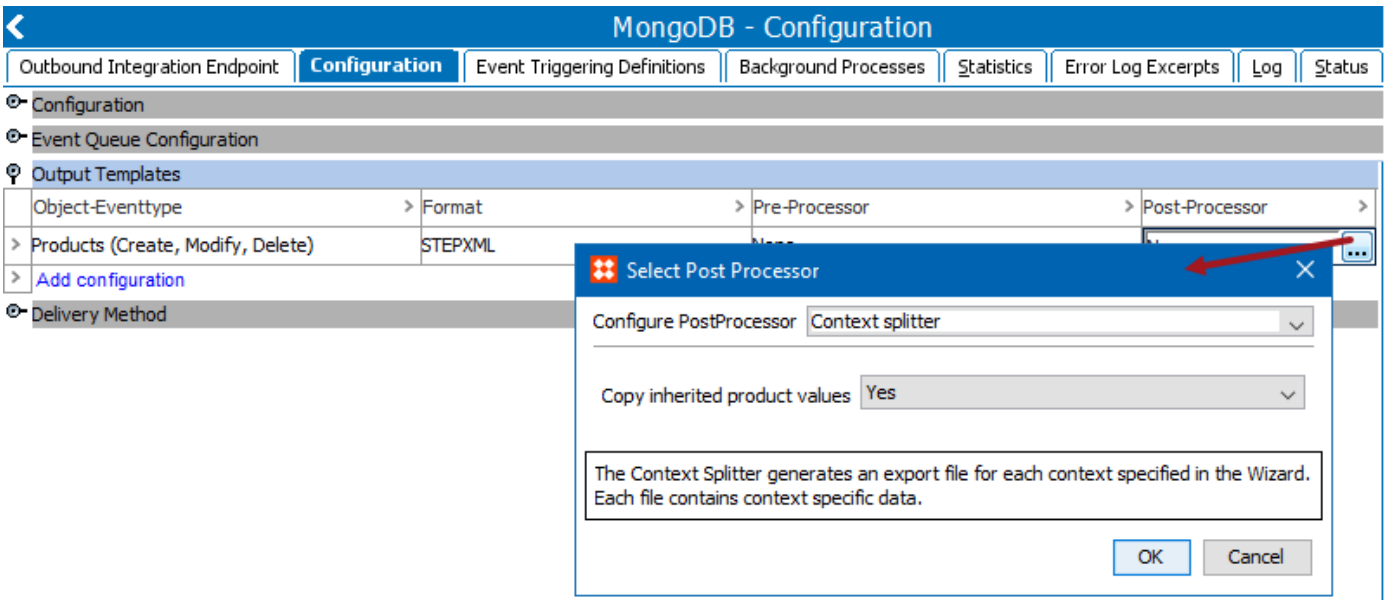
1. In System Setup, create an OIEP, selecting STEP Exporter as the Process Engine. For details on creating a new OIEP, see the **Creating an Event-Based Outbound Integration Endpoint** topic.
2. For the new OIEP, open Configuration > Output Templates > Add configuration > Format > click the ellipsis 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, see 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).

1. Start the Mongo database without authentication.
2. Log onto the database using the Mongo client.
3. 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:

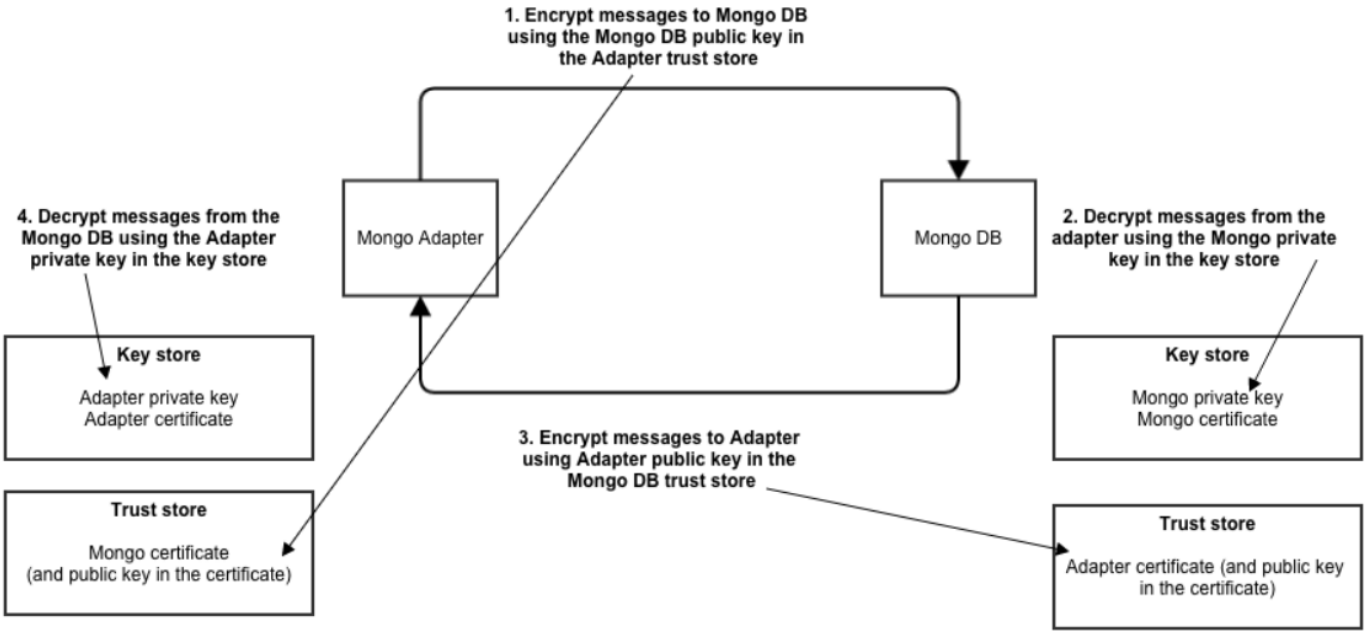
```
mongod --auth --config
```

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://docs.mongodb.com/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 datacontainer, 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 dir, 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": {
    "SpecificatonAttribute": {}
  },
  "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.

Prerequisites

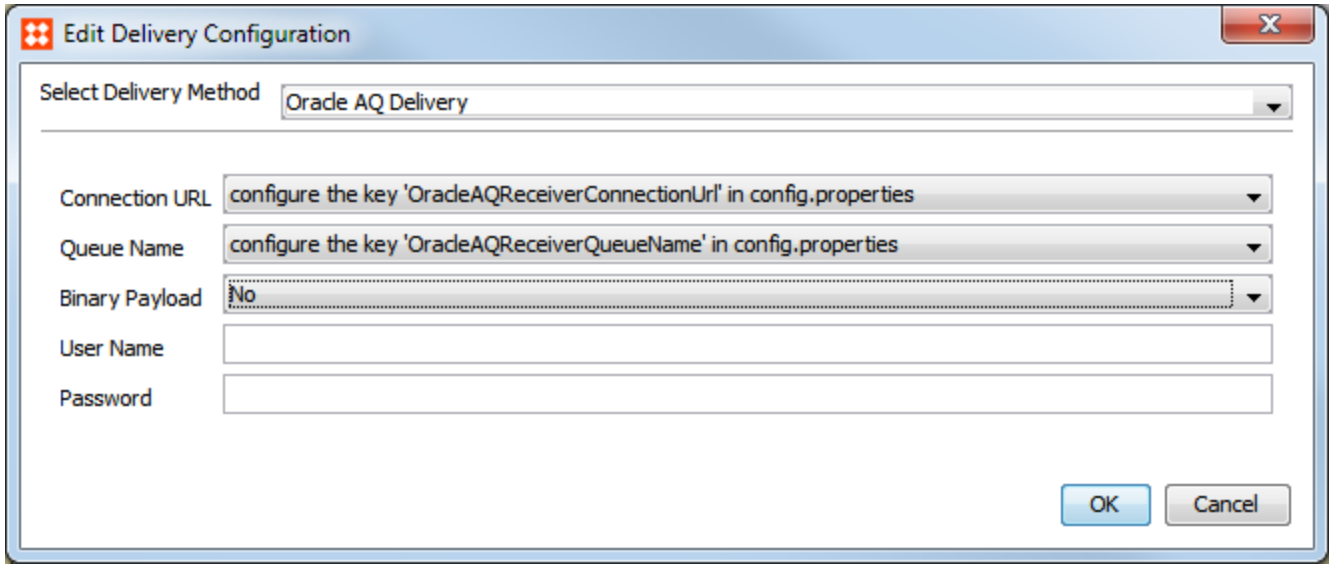
Changes to the properties file, outlined below, are implemented when the server is restarted.

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 Syndication Delivery Method

The Product Data Syndication delivery method is only available in OIEPs. STEP sends data to the Product Data Syndication (PDS) platform via the default PDS Outbound Integration Endpoint and the API.

For additional information on PDS, see the **Product Data Syndication** 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. 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 **PDS.Url** property. For example, a single URL is displayed with the following property setting:

   ```
   PDS.Url=1=https://StiboPDS/
   ```

   If multiple URLs are required, separate the entries with a comma, as shown below:

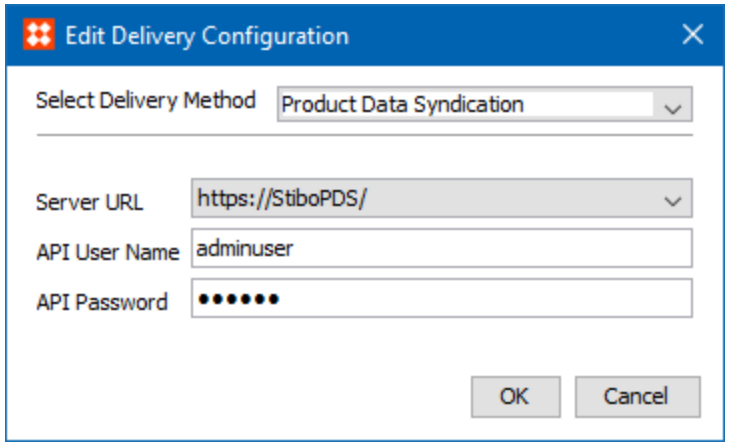
   ```
   PDS.Url=1=https://StiboPDS1/,2=https://StiboPDS2/
   ```

2. Determine the API user name and password. This will be required on the Edit Delivery Configuration dialog.
3. Perform additional configuration required in the OIEP as described in the **PDS Outbound Integration Endpoint Configuration** topic.

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 Syndication**.



3. For the **Server URL** parameter, select a URL from the dropdown.

4. For the **API User Name** parameter, type the user name with access to the API.
5. For the **API Password** parameter, type the password for the user name with access to the API.
6. On the 'Edit Delivery Configuration' dialog, click the **OK** button to save the delivery method.

REST Delivery Method

The REST Delivery Method delivers a call-back URL to the REST service and does not include actual 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.

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':

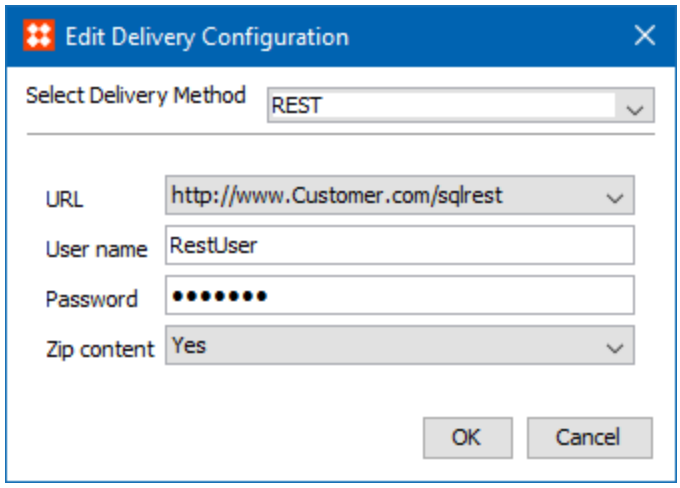
```
RESTDeliveryURL=qa=http://step-qa,stage=http://step-stage
```

Important: For information about how to use the REST API to upload files to REST, click the **STEP API Documentation** button on the STEP 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 **User Name**, enter the user name that will be used to log on to the REST endpoint.
4. In **Password**, enter the password that will be used to log on to the REST endpoint.
5. In **Zip content**, specify whether to zip the contents before upload.
6. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

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.

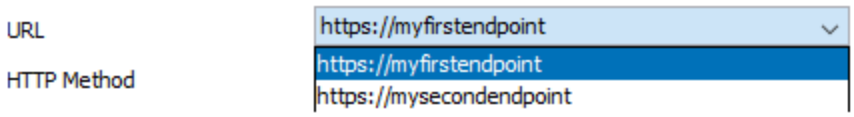
This delivery option is only available in OIEPs and if the delivery fails, the OIEP will be disabled. There is no special resilience handling.

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:

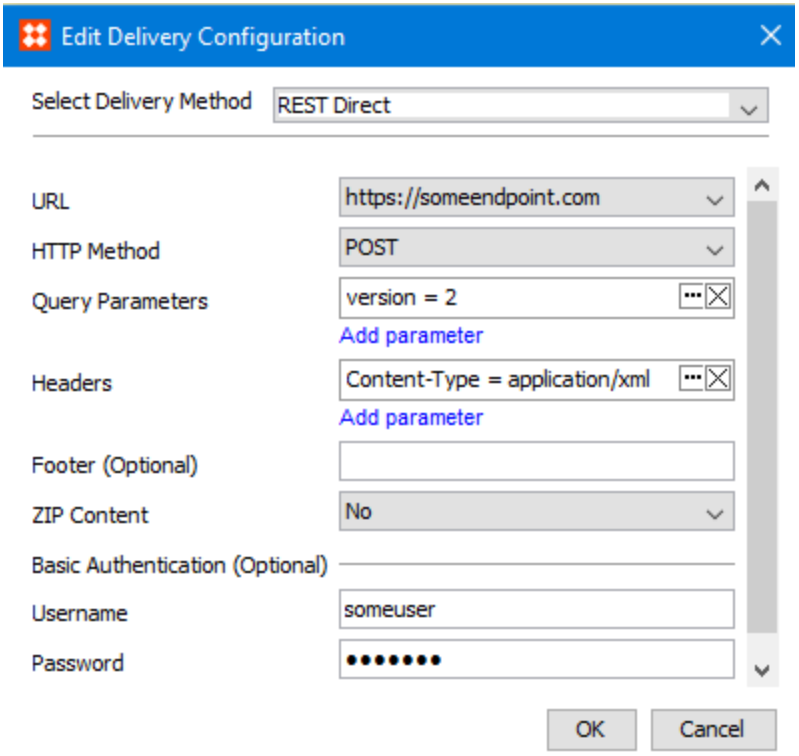
```
RestDirectDeliveryURL = 1=http://myfirstendpoint, 2=http://mysecondendpoint
```



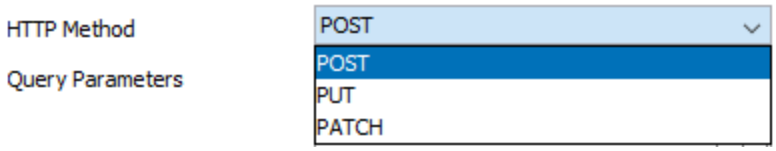
Configuration

For information on a parameter, hover over the parameter field to display help text.

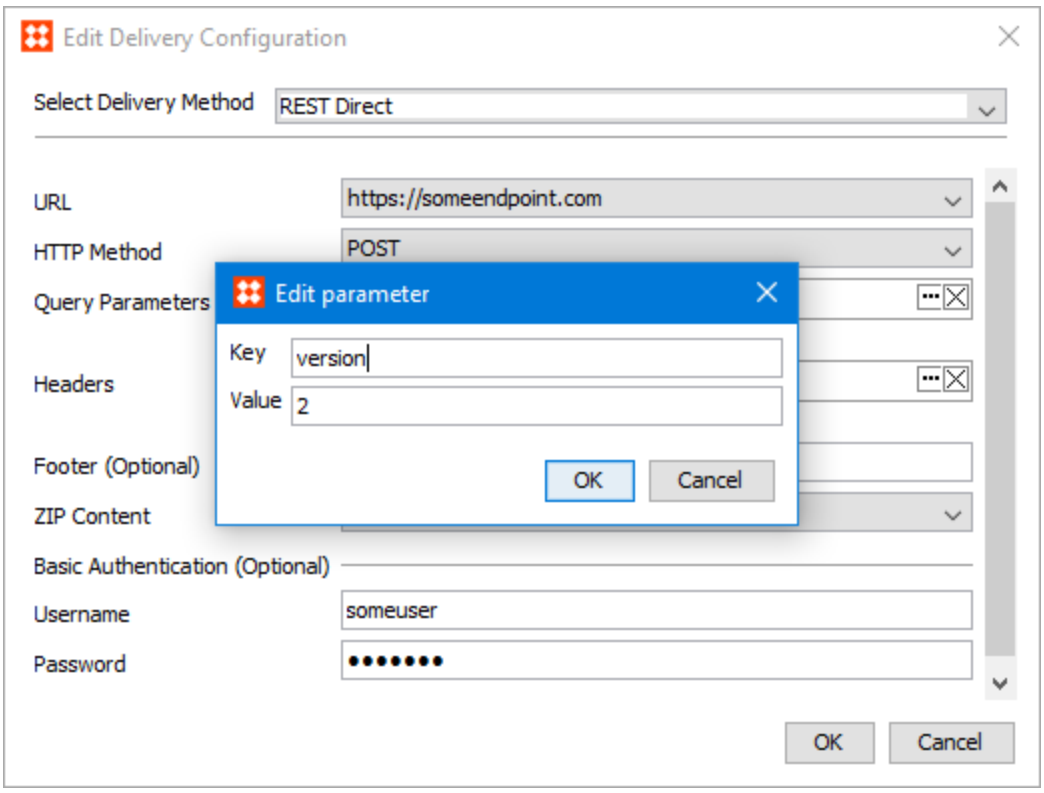
1. For **Select Delivery Method**, choose **REST Direct**.



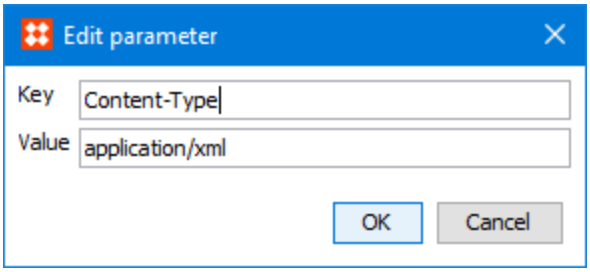
2. In **URL**, select the URL that points to the REST endpoint where the delivered data should be received.
3. In the **HTTP Method** option, use the dropdown to select a POST, PUT, or PATCH method.



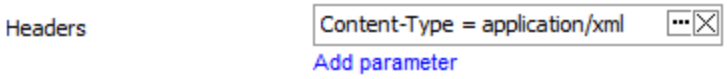
If required, in the **Query Parameters** field add any query parameters that are required for a successful REST transaction. You can click the ellipsis button (...) to add the 'Key' and 'Parameter' for the first parameter, or you can click on the **Add parameter** link to add the first and subsequent parameters.



4. For **Headers**, click the **Add Parameter** link and add a key and a value. This is part of the HTTP network protocol.



5. 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.

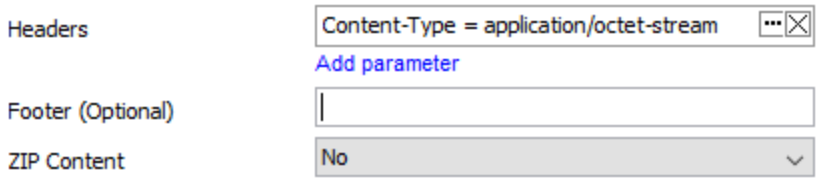


Note: Once a header is displayed, click the ellipsis button (...) to edit it or the X button to remove it.

For more information about headers, see the 'Additional Request Information' section at the end of this topic.

6. 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.



7. In **Zip Content**, specify whether to zip the contents before upload.
8. In **Username**, optionally enter the user name that will be used to log on to the REST endpoint.
9. In **Password**, optionally enter the password that will be used to log on to the REST endpoint.

10. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

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 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.

This delivery method is also available in Export Manager. For information on the OIEP FTP delivery method, see **FTP Delivery Method**.

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].

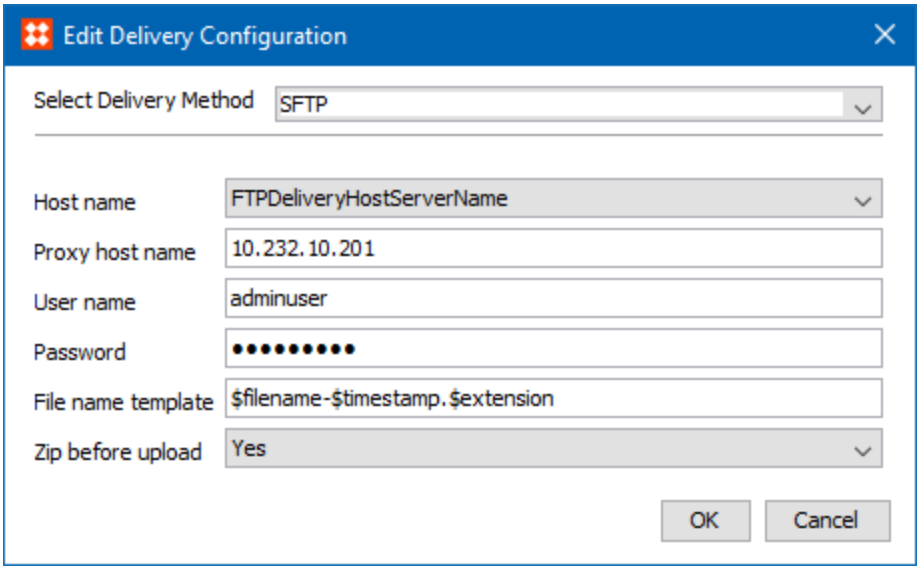
Using the host name shown in the image, the entry in the properties file would be:

```
FTPDeliveryHostName=1=FTPDeliveryHostServerName
```

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.
6. In **File name template**, to create the file name used for the output, enter template variables, static text, or use a combination. The default template is $filename-$timestamp.$extension and creates a file name like 'GenericXML-1490813266535.xml' based on the settings of the OIEP. Each variable is described below:
   - **$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 milliseconds between January 1, 1970, and the time when the file is uploaded, which allows for a unique identifier. For example, 1479230247017 is used for an export on 15 NOV 2016.

     Alternatively, the following options can be used to generate a readable date and can include static characters to separate the timestamp elements. Individual elements can be ordered as required, as shown below.

     $timestamp(ddMMyyyy) displays day, month, and year and is output '15112016' for 15 NOV 2016.

     $timestamp(yyyy-MM-dd_HH.mm.ss) displays year, month, day, hour, minutes, and seconds 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 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. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

Wiki Delivery Method

The XWiki 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.

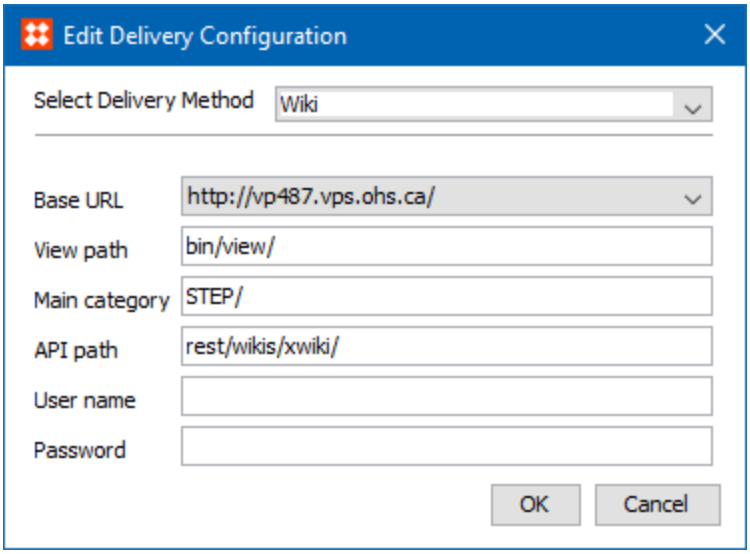
Configuration

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 XWiki 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 **XWiki**.



2. In **Base URL**, select the URL that points to the wiki that will be used for this STEP system. See 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 endpoint, it must be enabled before it can run on schedule or be invoked manually.

The following three 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 > 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	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, see **Handling Failed IIEP 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, keep the endpoint enabled but do not invoke the endpoint until the external system is ready to consume the data. In this way, data is collected from STEP and is then pushed to the external system only when invoked.

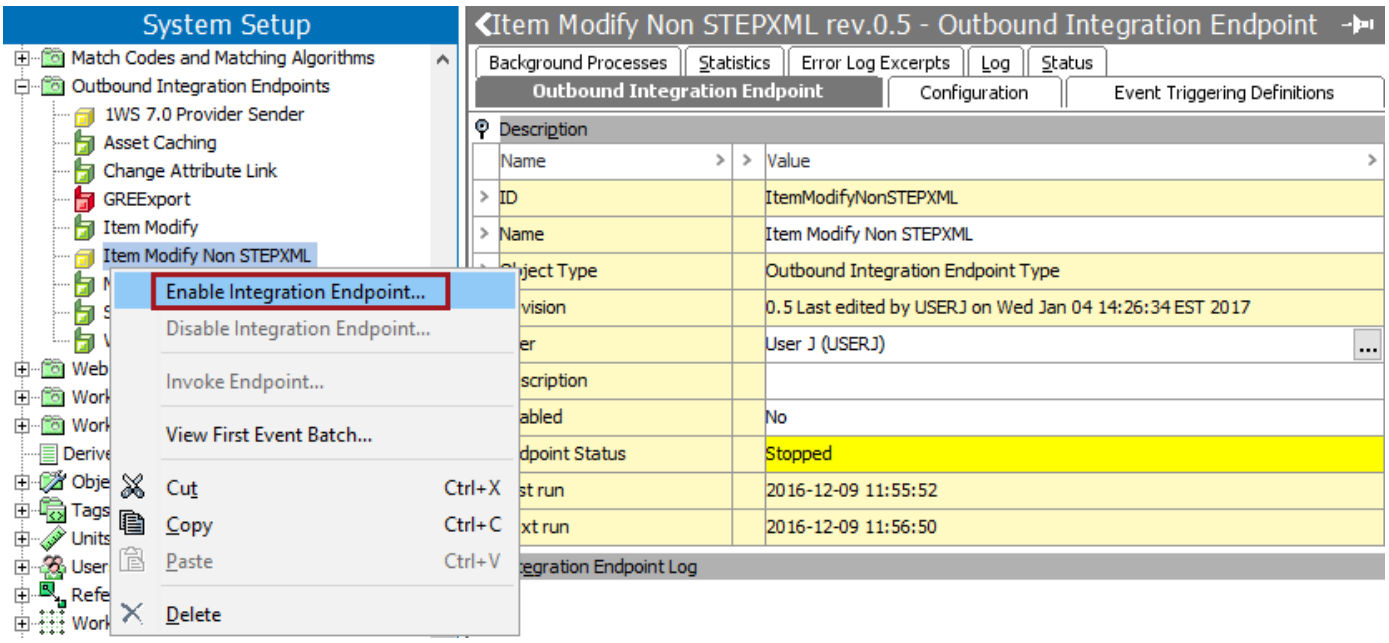
Prerequisites for Event-Based OIEPs

For an event-based OIEP, verify the Queue Status is set correctly. For information about how the Endpoint Status and Queue Status settings work together on event-based endpoints, see the **Event-Based OIEP Status and Queue Status** documentation.

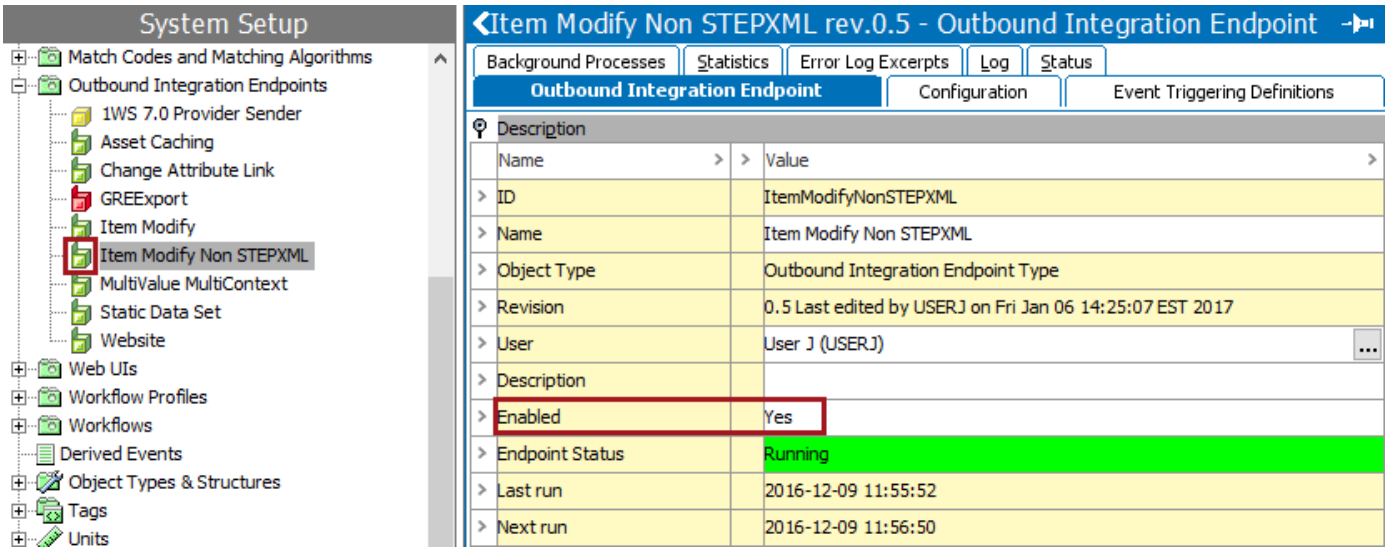
For information about setting up an event processor, see **Event Processors** in the **System Setup / Super User Guide**.

Required Setup for All OIEPs

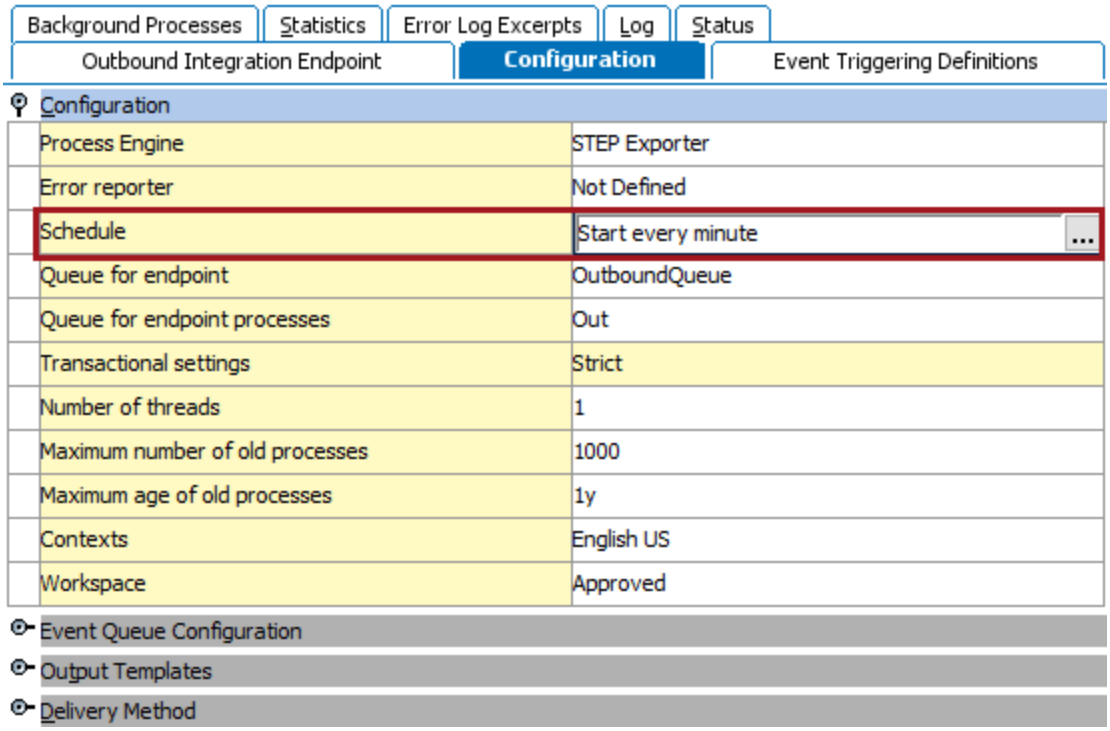
1. On System Setup expand the Outbound Integrations Endpoints node to display all existing outbound endpoints.
2. Select a disabled endpoint or a failed endpoint, right-click, and select **Enable Integration Endpoint** from the menu.



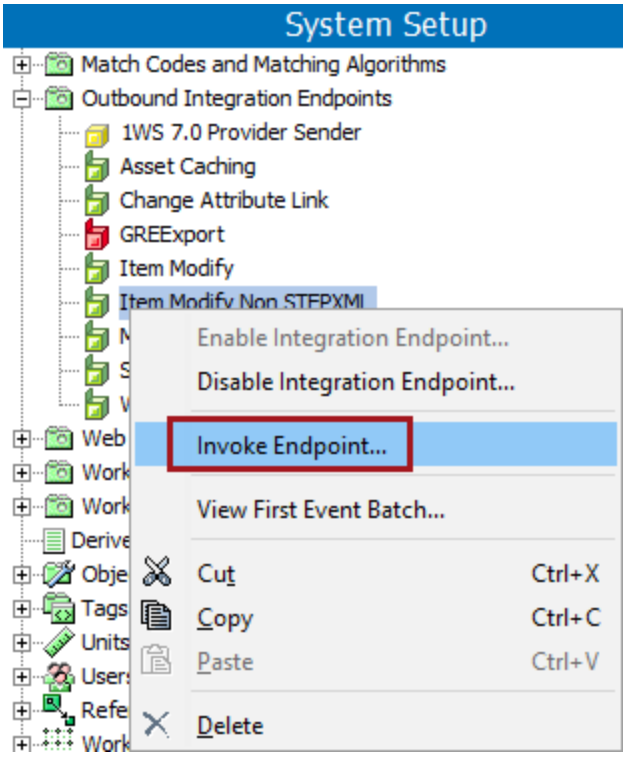
The tree navigator icon now shows enabled. On the Inbound Integration Endpoint tab, the Description flipper includes the 'Enabled' parameter, which now displays 'Yes.'



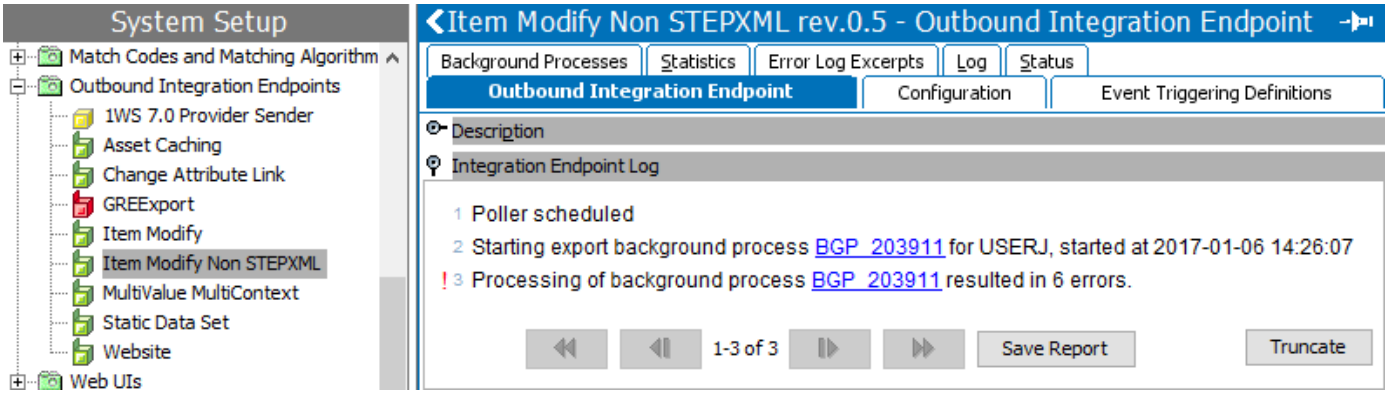
3. On the Configuration tab, open the Configuration flipper to view the Schedule parameter.
   - If scheduled, the Schedule parameter shows when the IIEP will be invoked automatically.



- If 'Not scheduled', or if the IIEP needs to be invoked on demand, right-click the enabled IIEP and click **Invoke Endpoint**.



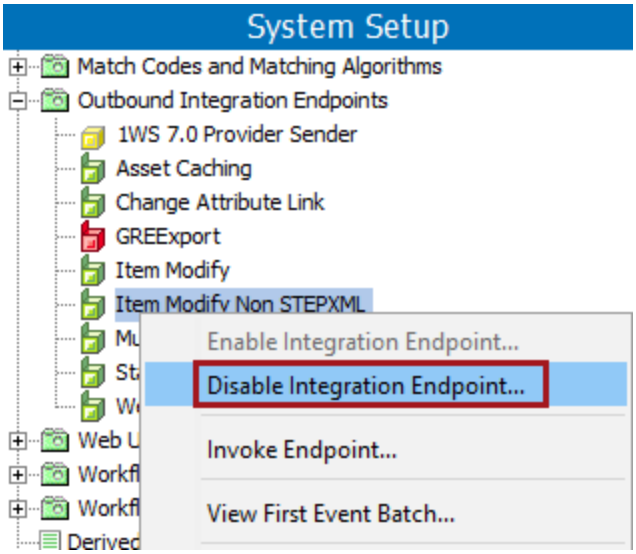
4. When invoked on schedule and manually, files or messages ready for export as identified by the OIEP configuration are processed. The Outbound Integration Endpoint tab Integration Endpoint Log flipper displays the time the endpoint was invoked. For scheduled OIEPs, the log includes a link to the background process that was started. For more information, see **Maintaining an Outbound Integration Endpoint**.



Disable an Outbound Integration Endpoint

When an OIEP 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.

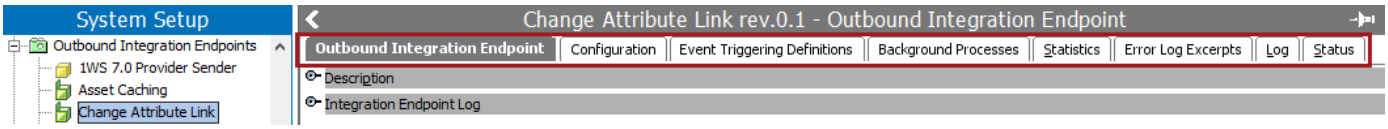


Once disabled, the outbound integration endpoint can be restarted using the Enable Integration Endpoint step defined above.

Additional information about restarting a failed endpoint, indicated by the failed icon ( ), can be found in 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, see the **Creating an Outbound Integration Endpoint** section.



Each tab on the editor is defined below and includes parameters to maintain the OIEP.

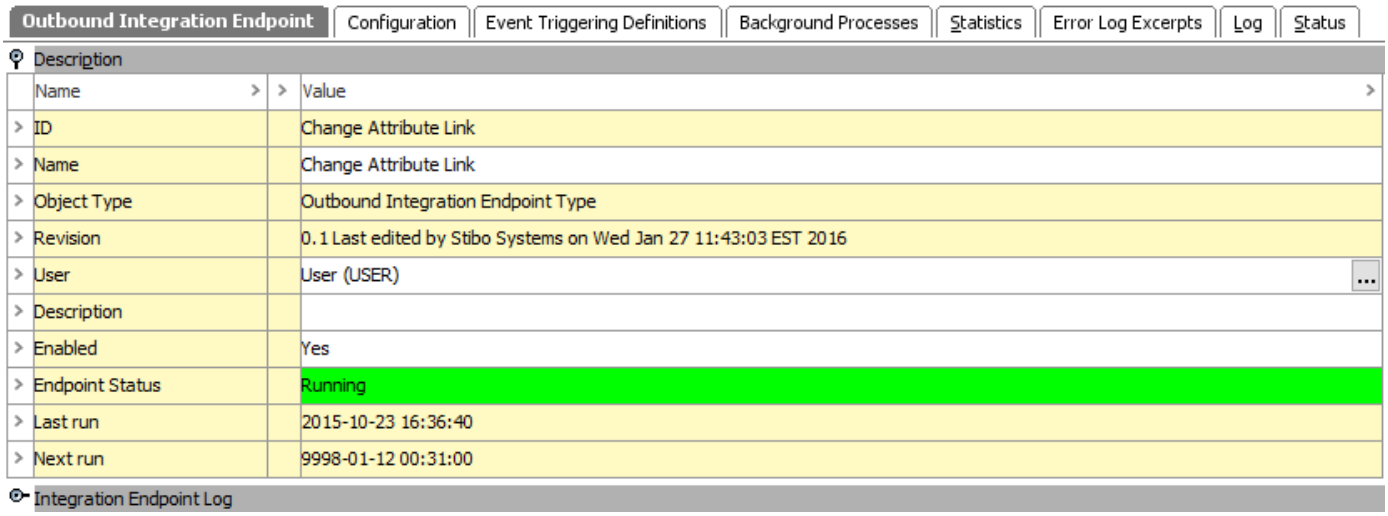
Outbound Integration Endpoint Tab

This tab holds basic information within the following flippers:

Description Flipper

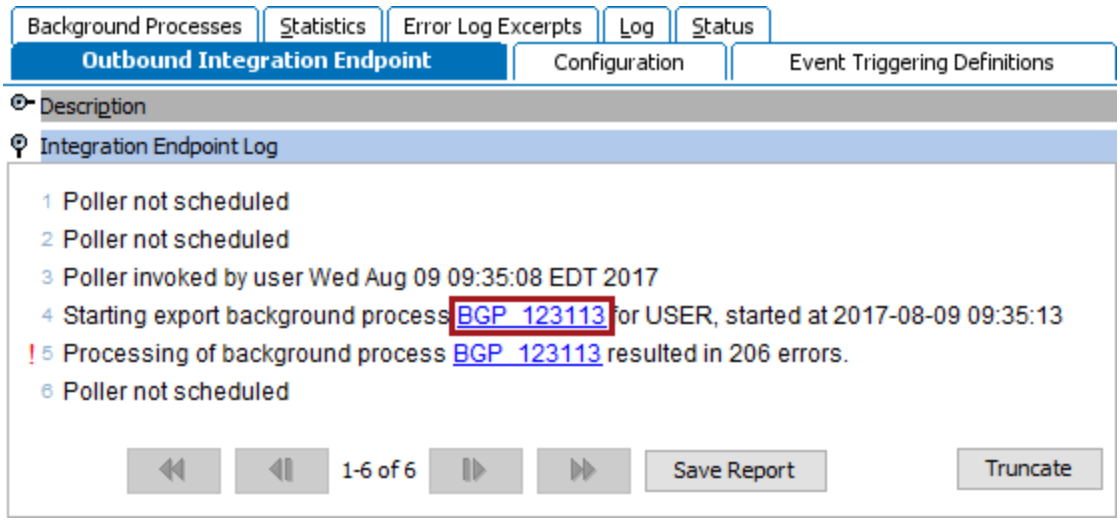
The Description flipper includes basic information to identify the OIEP. The name, user, description, and enabled parameters can be edited.

For information on the Endpoint Status, see **Running an Outbound Integration Endpoint** documentation.

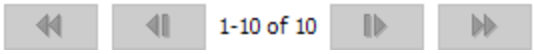


Integration Endpoint Log Flipper

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.

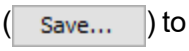


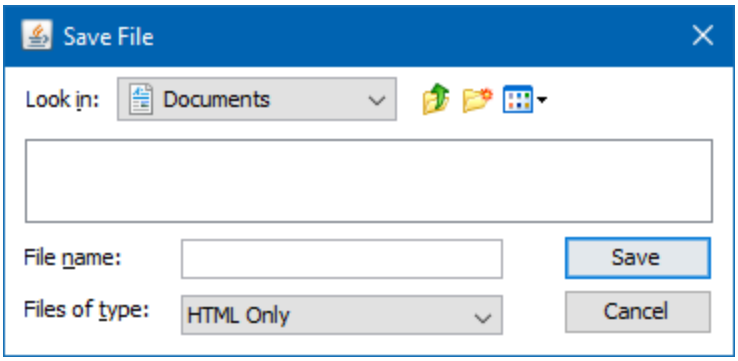
- **BGP link** - The background process opens the individual Importer Background Processes.
- **Navigation** - After the first entry, the Execution Report displays the following navigation buttons:



The buttons are enabled when until the log reaches over 100 lines.

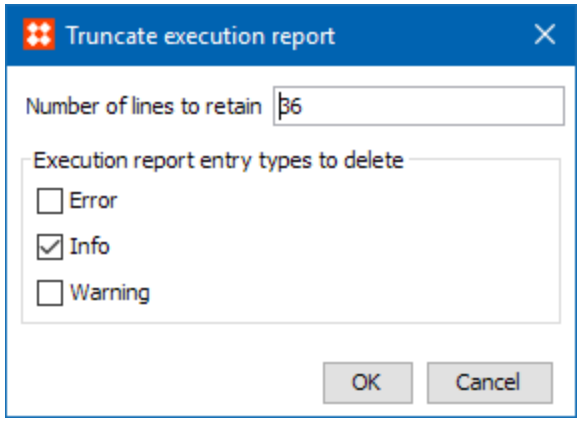
- Selecting the button will navigate back one page.
- Selecting the button will navigate to the beginning of the log.
- Selecting the button will navigate forward one page.
- Selecting the button will navigate to the end of the log.

- **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 become 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.

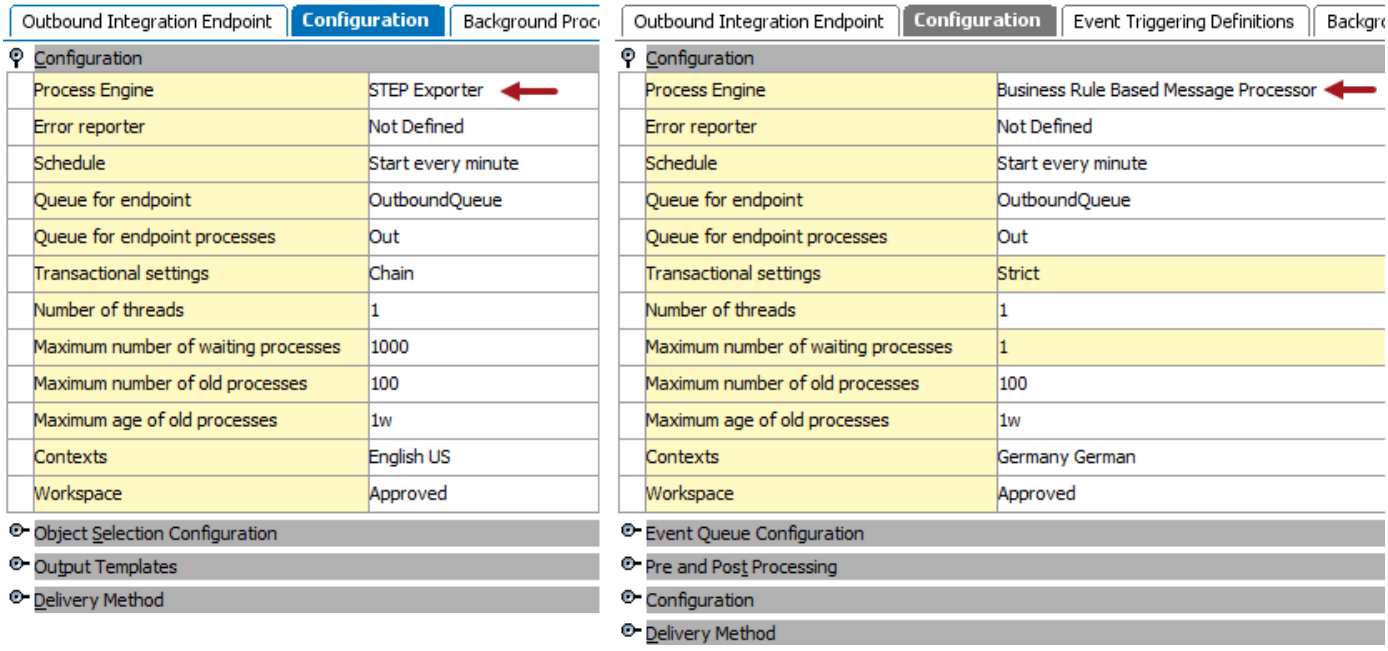
  - Click **OK** to permanently remove the selected number and types of execution report entries.

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.



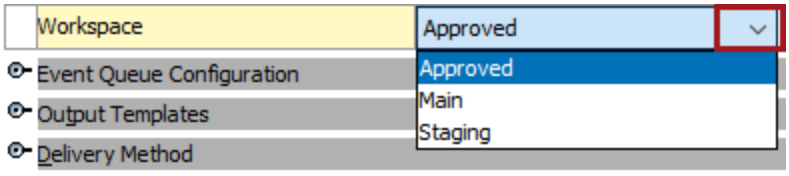
The **Transactional setting** parameter cannot be edited and does not have the same significance as for IIEPs. For information, see **OIEP - Event-Based - Configure Endpoint** or **OIEP - Select Objects - Configure Endpoint**.

Depending on the Process Enine selected, a second Configuration flipper can be displayed. For more information, see the

The following parameters can be edited within the OIEP editor as defined:

- Click the **Process engine** or **Workspace** parameter to display a dropdown button. Selections displayed depend on your system setup.

  For information on the Process engine or Workspace, see **OIEP - Event-Based - Configure Endpoint** or **OIEP - Select Objects - Configure Endpoint**.

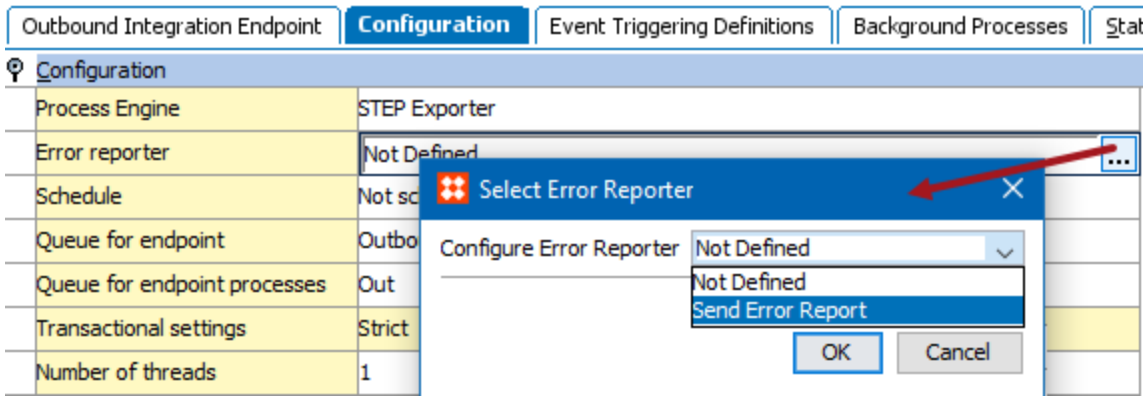


- Click the **Error reporter**, **Schedule**, or **Contexts** parameter to display an ellipsis button (...). This opens a dialog for editing.

  For information on the Error Reporter or Schedule, see the Error Reporter section of the **OIEP - Configuration Flipper**.

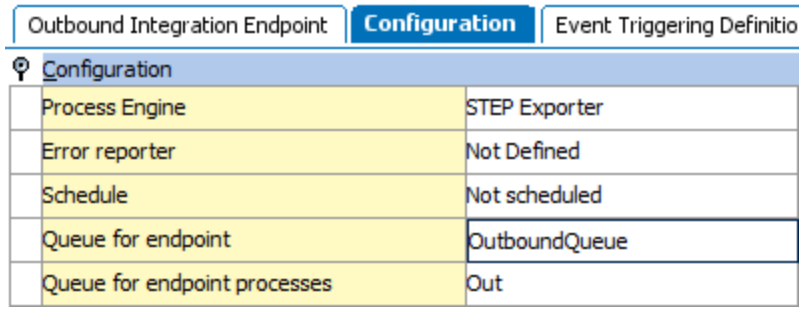
  For information on Contexts, see **OIEP - Event-Based - Configure Endpoint** or **OIEP - Select Objects -**

Configure Endpoint.



- Click the **Queue for endpoint**, **Queue for endpoint process**, **Number of threads**, **Maximum number of old processes**, or **Maximum age of old processes** parameter to edit the text with the field.

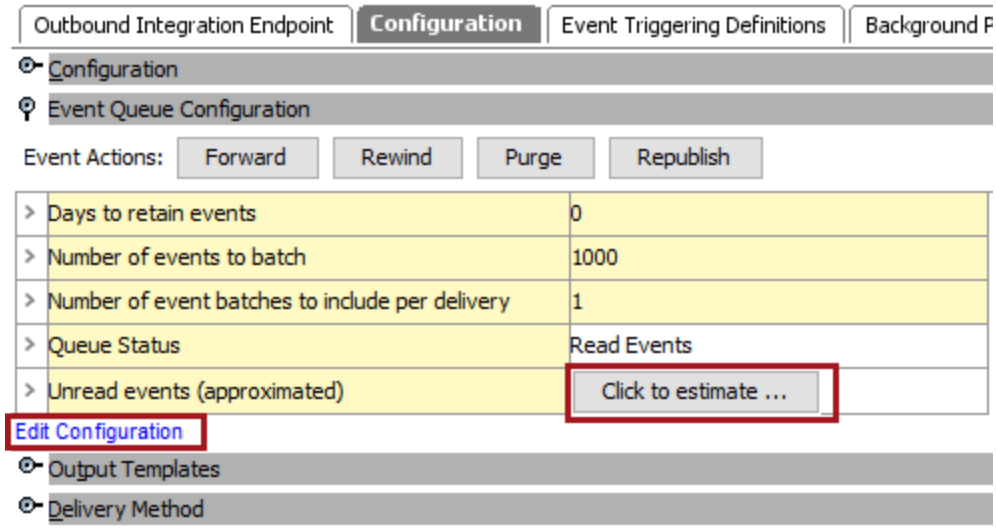
  For information on the Transactional settings, Number of threads, Contexts, and Workspace, see **OIEP - Event-Based - Configure Endpoint** or **OIEP - Select Objects - Configure Endpoint**.



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.

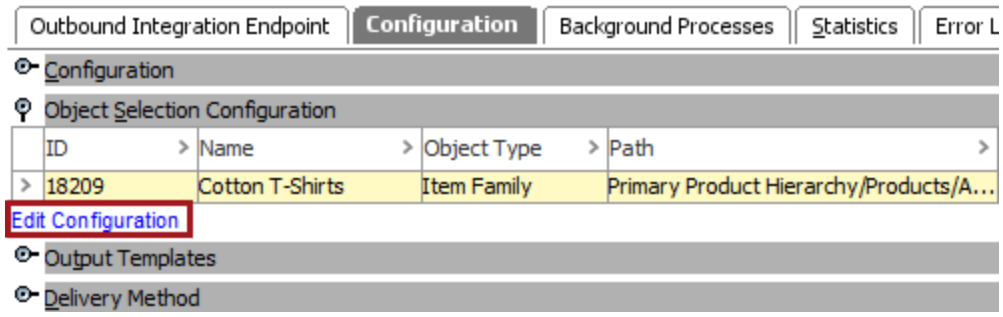
- The **Edit Configuration** link allows modification to the parameters displayed.
- The **Click to estimate** button displays a list of unread events.
- For more information on the Event Action buttons, see **Event-Based OIEP Forward, Rewind, Purge, and Republish**.



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, see **OIEP - Select Objects - Choose Data Source**.

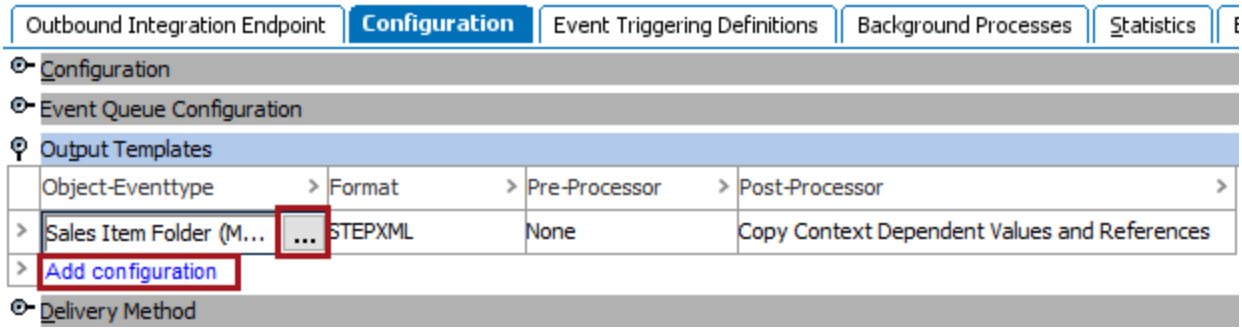
- The **Edit Configuration** link opens a dialog that allows modification to the objects selected for output. For details on the available options, see **OIEP - Select Objects - Object Selection Configuration Flipper**.



Output Templates Flipper

This flipper allows selection of object or event types, in addition to the format required for output. Preprocessors and postprocessors 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.

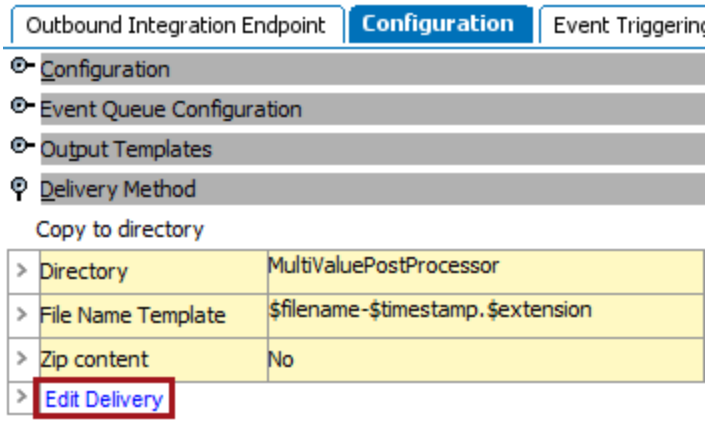


For details on the output templates, see **OIEP - Event-Based - Output Templates Flipper** or **OIEP - Select Objects - Output Templates Flipper**.

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, see **OIEP - Delivery Method Flipper**.

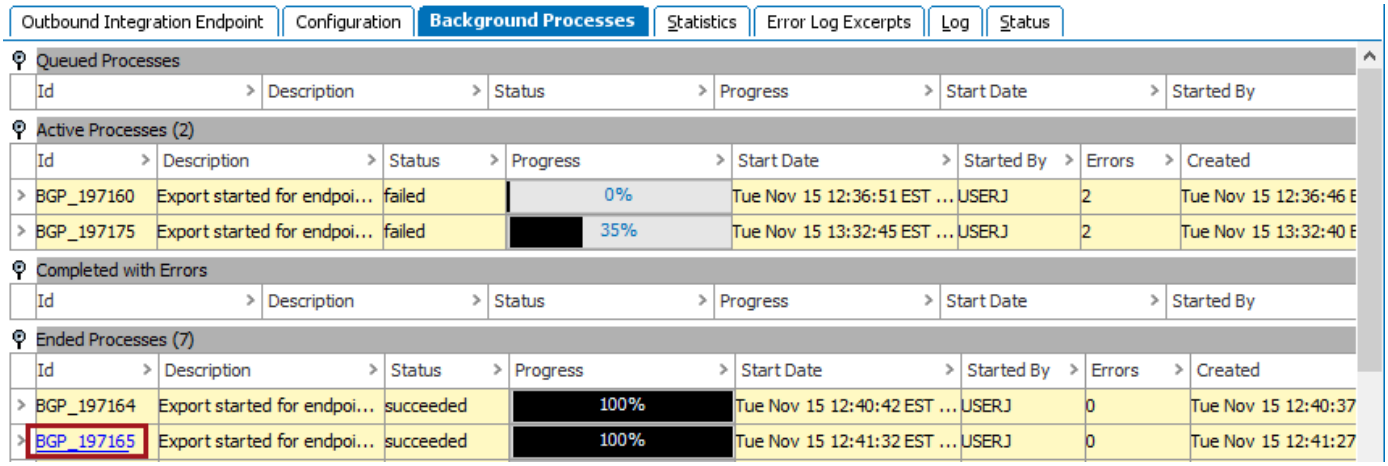
Background Processes Tab

This tab holds information about any related processes. Click the Background Process ID link to open the individual exporter background process.

- **Queued Processes** are background processes that are started in the state 'waiting.' The maximum number of background processes that can be started in the state waiting is specified in the integration endpoint configuration.
- **Active Processes** are background processes that are currently being processed. If data is processed as strict or chained, the endpoint only has one active background process at a time. Otherwise, it is possible to apply more background process queues to work concurrently. In this case the endpoint can have more active background processes at a time.

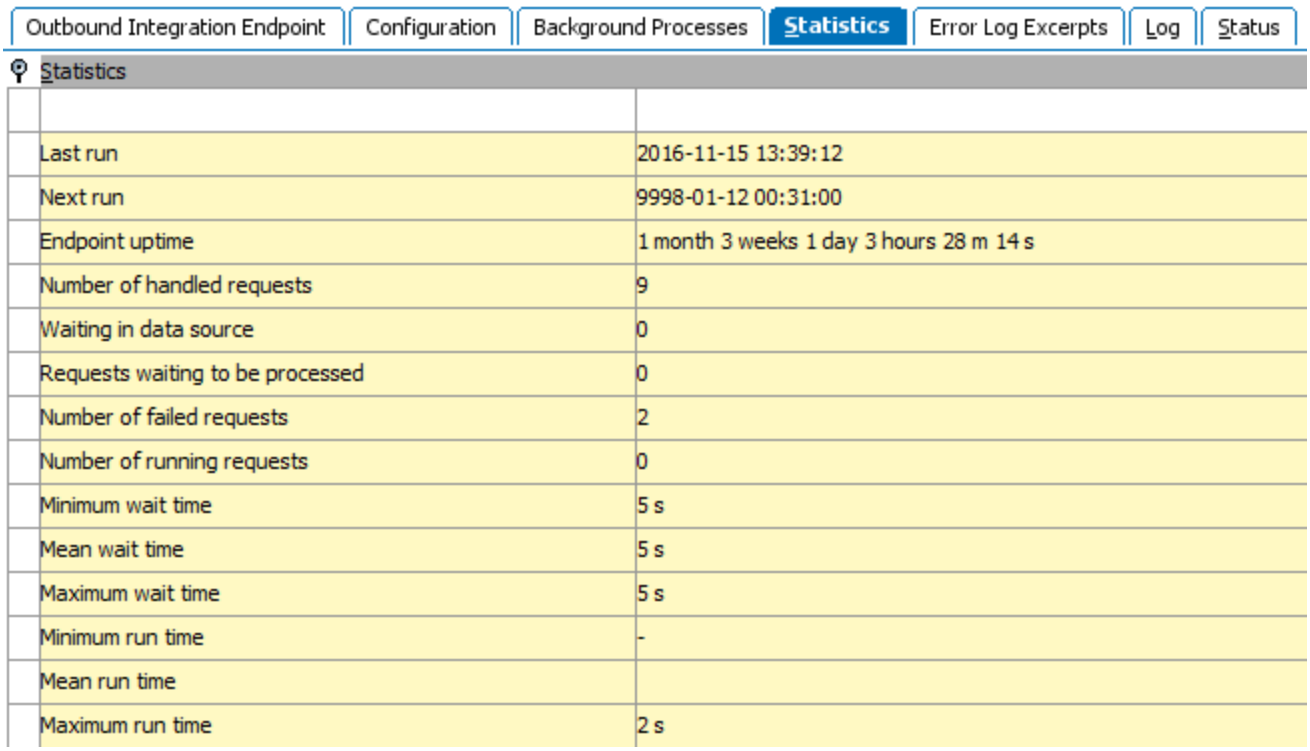
- **Failed Processes** are background processes that are no longer running. For more information, see **Handling Failed OIEP Background Processes**.
- **Completed with errors** have run, but an error has been detected.
- **Ended Processes** have completed successfully. The number of ended processes is determined by the configuration.

For more information, see the **Monitoring an OIEP via Background Processes** section.



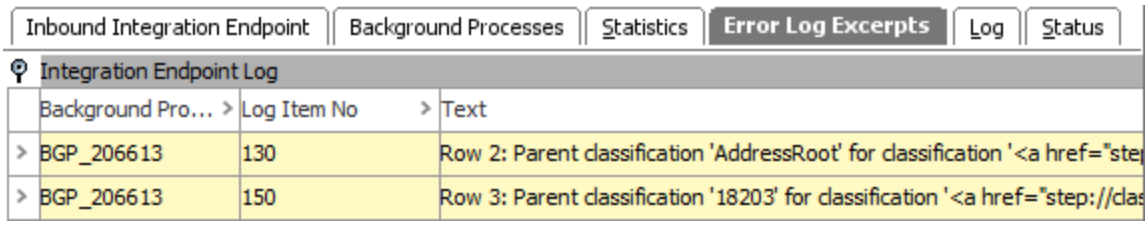
Statistics Tab

This tab provides statistics about the number of requests handled, run times, and wait times. The information cannot be edited nor exported.



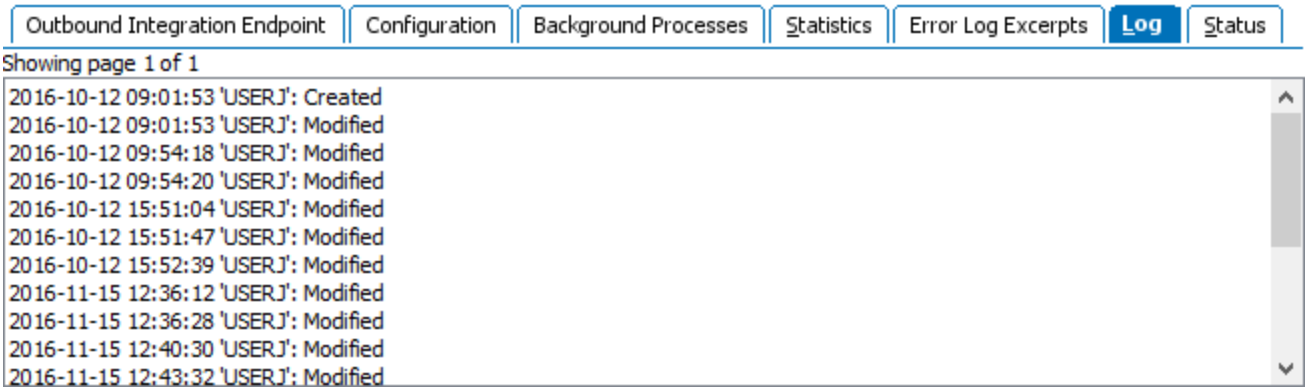
Error Log Excerpts Tab

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, see **Handling Failed OIEP Background Processes**.



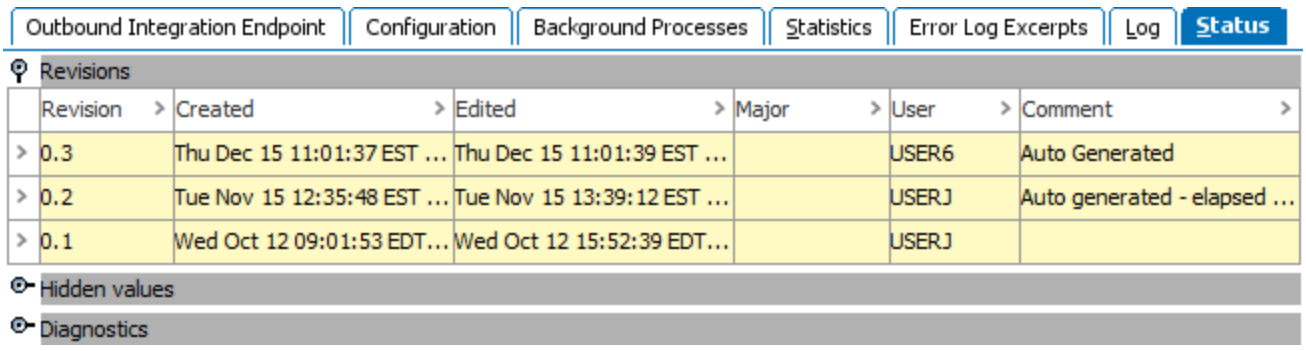
Log Tab

This tab provides information about changes to the OIEP configuration by a user, including the timestamp.

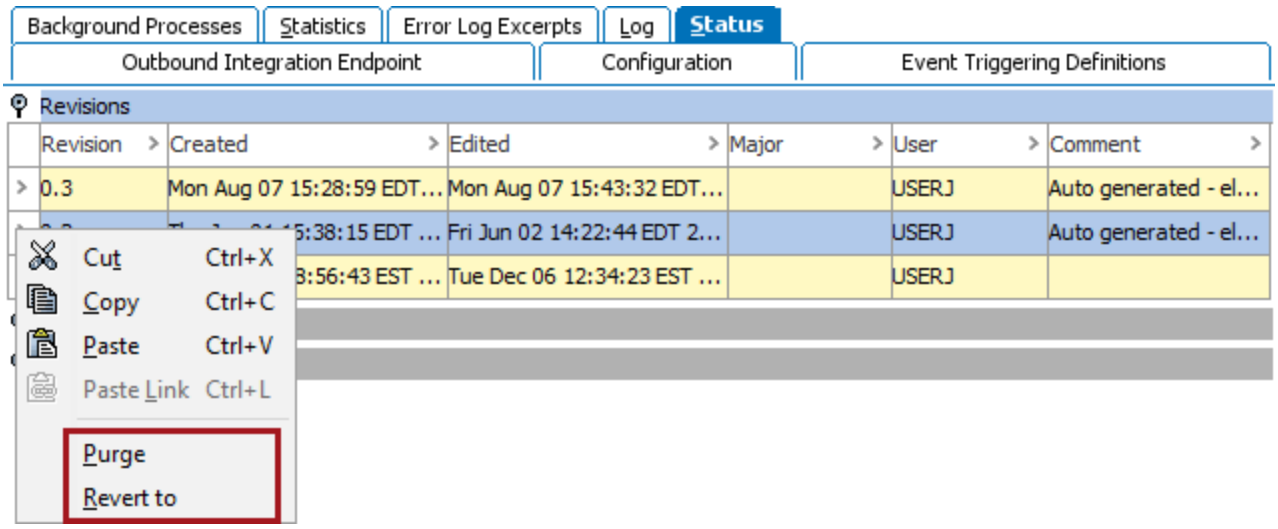


Status Tab

This tab provides information about revisions, hidden values and diagnostics.



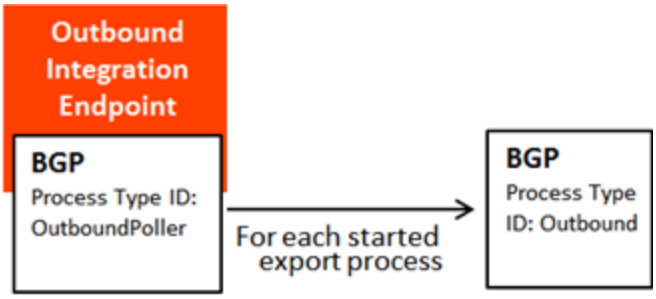
Right-click on the > to the left of a revision row to display a menu that allows you to revert to the changes for the selected revision or delete (purge) the selected revision.



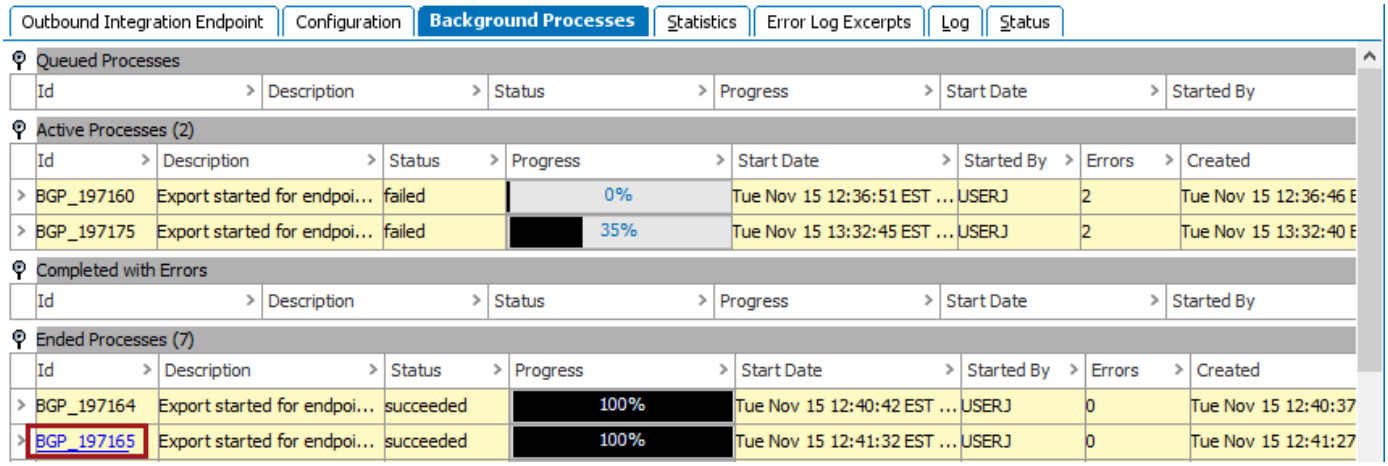
For more information on revisions, see the **Revisions** topic in **System Setup / Super User Guide** documentation.

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, see **Maintaining an Outbound Integration Endpoint**.



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.

Selecting the 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, see the **OIEP - Configuration Flipper** documentation.

Background Process Queue Size Property

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 in **Exporting Outbound Integration Endpoint Definitions as Comments**.
- 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**.

Exporting Outbound Integration Endpoint Definitions as Comments

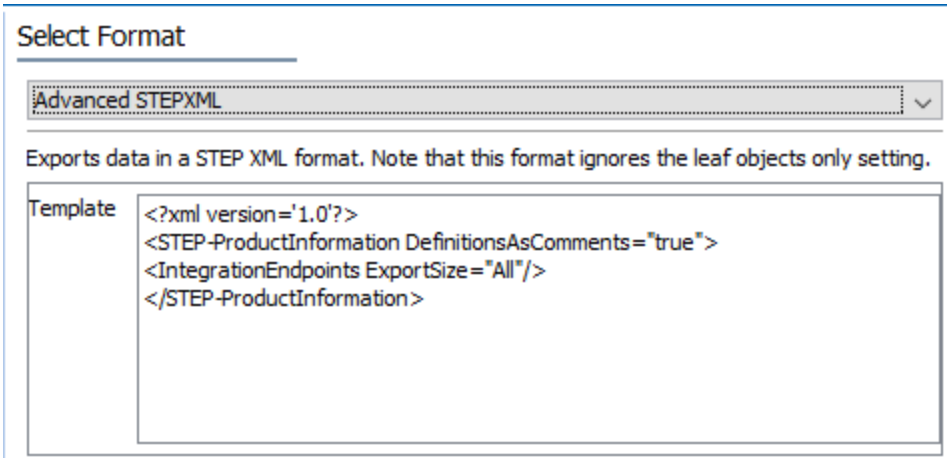
Integration endpoint definitions can be exported as comments using Advanced STEPXML. These exports are intended to be used for submission to external source control systems for comparison purposes. Users can import them into source code repository systems where they can be compared from version to version.

Editing and/or importing of these files is not supported (e.g., users may not export, edit the comments, and re-import in STEP). However, the exported file can be imported into another STEP system, which will result in creation of the same endpoint. For example, integration endpoints are typically built in a test or development system before being implemented in the production system. Once creation and testing is complete and satisfactory, the endpoint can be exported from the test environment and then imported into the production environment. This results in the same endpoint on both systems.

To export integration endpoint definitions (both inbound and outbound) for external comparison, use Advanced STEPXML and set the DefinitionsAsComments tag to 'true', as shown below.

On the Select Format step of the outbound tool (either Export Manager or OIEP), choose the Advanced STEPXML format, then copy and paste the following text into the Template field:

```
<?xml version='1.0'?>
<STEP-ProductInformation DefinitionsAsComments="true">
<IntegrationEndpoints ExportSize="All"/>
</STEP-ProductInformation>
```



An example of the output for an outbound integration endpoint definition as comments is shown below:

```
<OutBoundIntegrationEndpoint...>
<!--Definition: [This will be the IEP definition. Removed for brevity.-->
[Remaining configuration is included]
</OutBoundIntegrationEndpoint>
```

Note: The content of the comment field is not part of the STEPXML XSD and therefore Stibo Systems reserves the right to change the format of the output content at any time.

For more information, see the **STEP-ProductInformation Tag in STEPXML** section of the **Data Exchange** documentation.

Additionally, details about integration endpoint configuration can be exported using STEPXML. For information, see the Configuration > **Include Integration Endpoints** section of the **STEPXML Outbound Parameters** documentation.

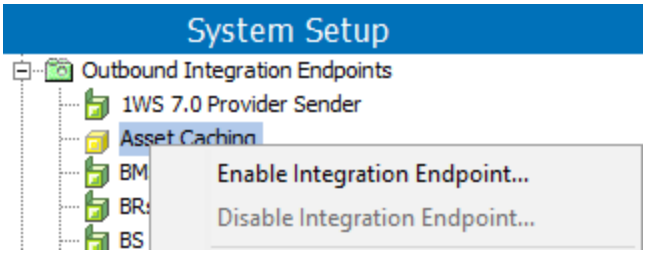
Handling Failed OIEP Background Processes

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**.



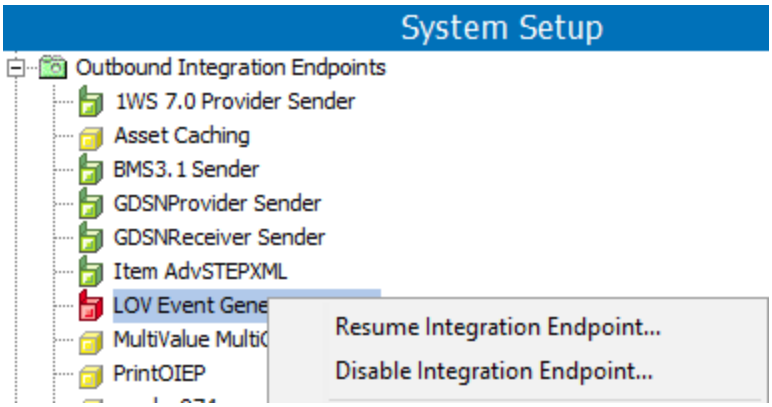
Resume a Background Process

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**.



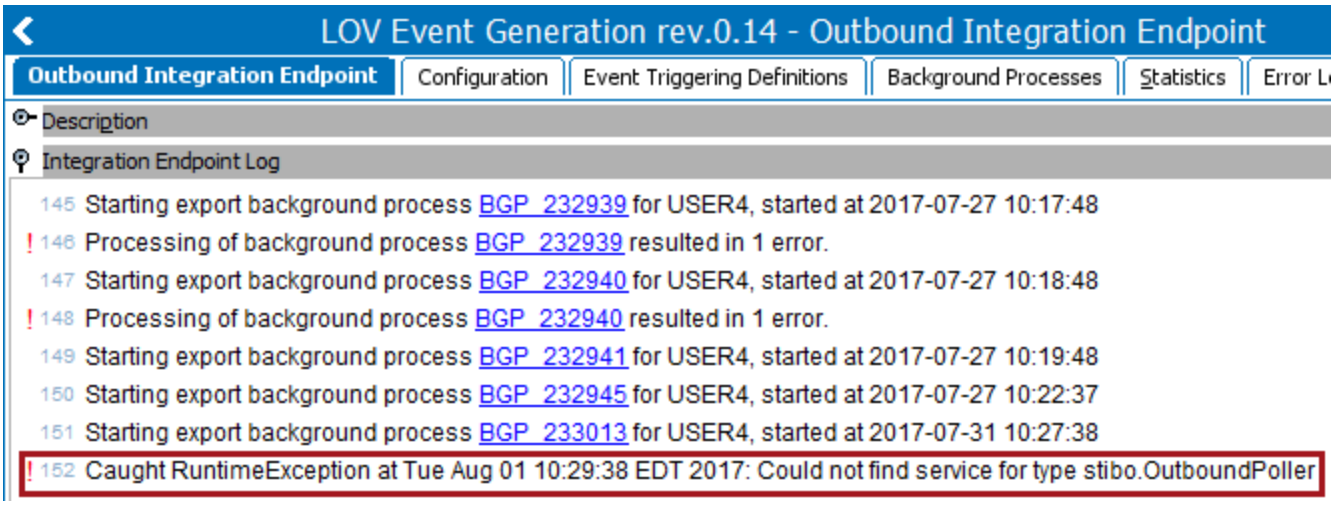
Resolving a Failed Background Process

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 Reporter** section of **OIEP - Configuration Flipper**.

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.



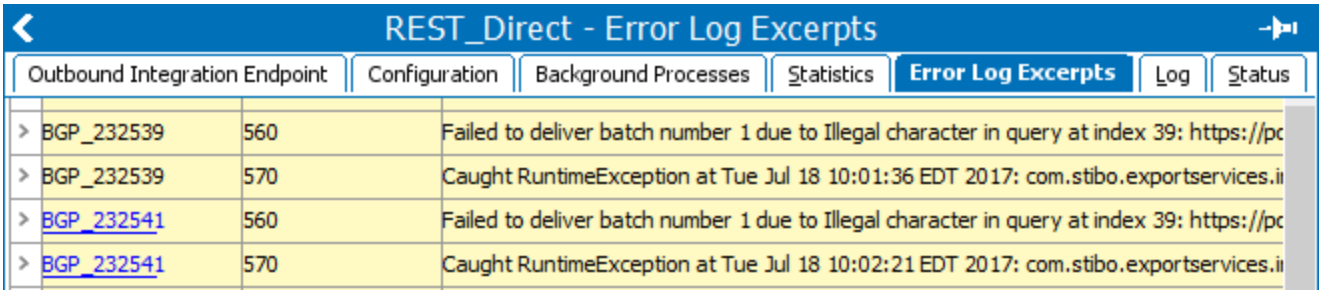
OIEP Fails After the BGP Starts

The Error Log Excerpts tab includes details about failures within the background process.

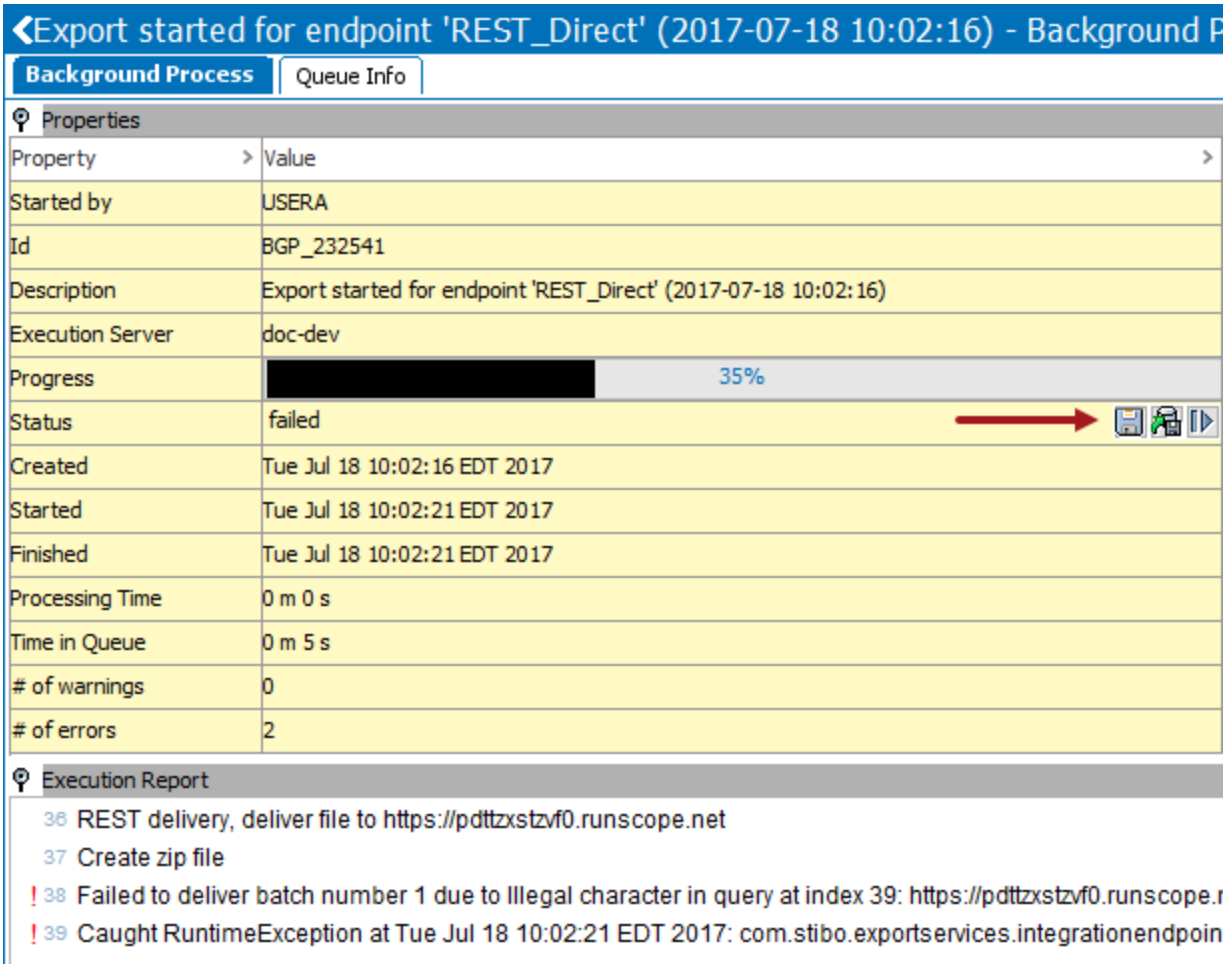
1. In **System Setup** select the relevant integration endpoint, and then click the **Error Log Excerpts** tab to display a list of failed background processes and the description of the failure.

   You can also see 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.



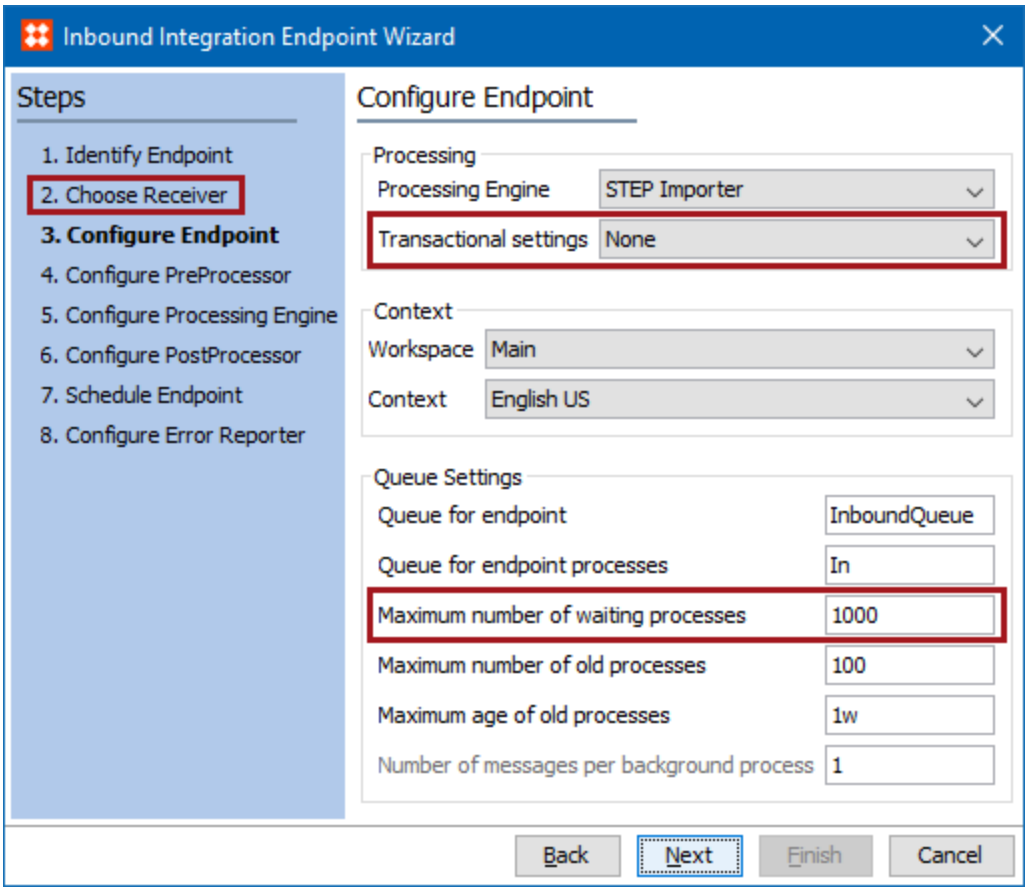
3. In the Background Process editor **Execution Report**, view the progress of the background process, and where it failed.
4. In the Properties flipper > Status parameter, if available, click the Save button ( ) in the Value column, near the failed status, to save the file locally.
5. Open the file 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 > Properties flipper > Status parameter Value column, click the Upload button ( ) near the failed status.
7. Restart the process in the Properties area > 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.

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.

Inbound Integration Endpoints

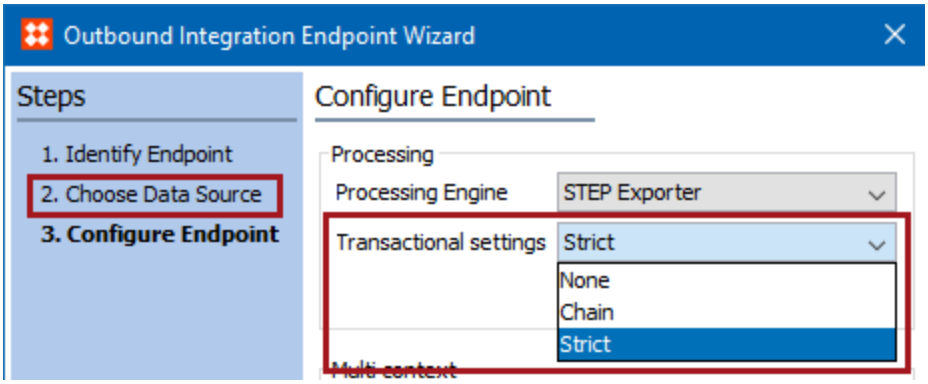
In the IIEP wizard, background process and error handling is managed by three (3) elements: the selected Receiver (on the Choose Receiver step), the 'Transactional settings' parameter, and the 'Maximum number of waiting processes' parameter (both on the Configure Endpoint step). These elements also work together to determine what additional options are available in the wizard, as defined below.



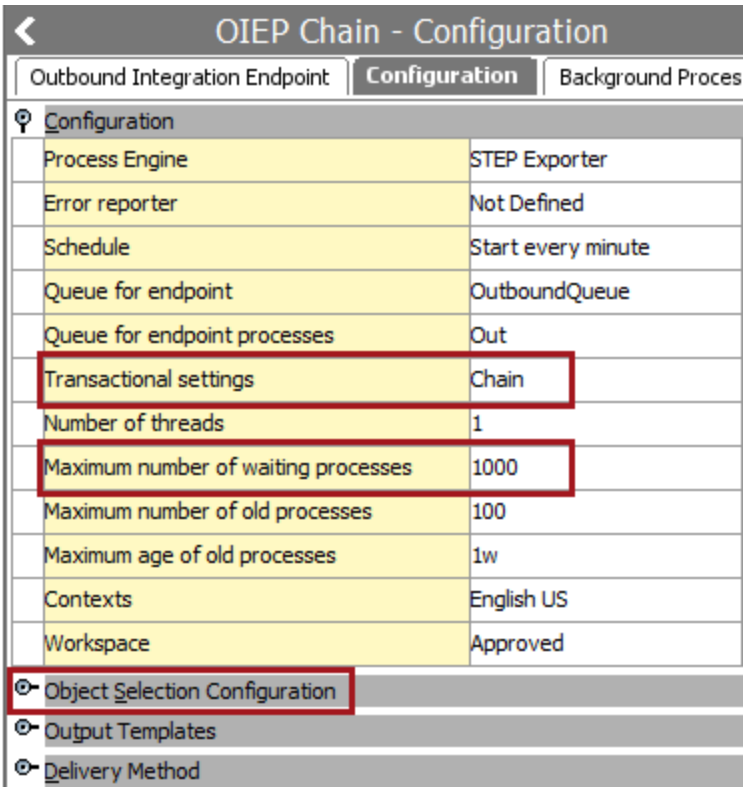
For more information, see the **IIEP - Configure Endpoint** documentation.

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, see the **OIEP - Select Objects - Configure Endpoint** topic.

For details on the 'Transactional settings' and 'Maximum number of waiting processes' parameters, see 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.

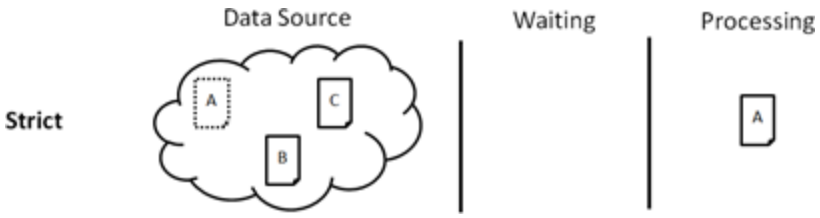
For more information on event-based OIEPs, see the **OIEP - Event-Based - Configure Endpoint** topic.

Transactional Settings Options

When available, there are three possible transactional settings: 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 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 common setup, unless there is a reason to use Chain or None (for assets).



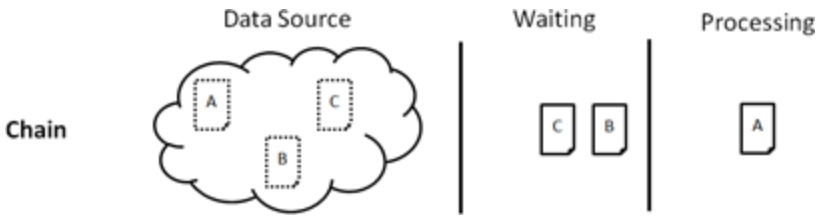
The queue size of a background process with strict transactional settings must be one (1). A strict transactional setting does not allow multiple background processes to work in parallel / concurrently regardless of the queue size. By definition, no new background processes can be generated while one 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 'chain' transactional setting processes data in batches of chained background processes. 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. Chain is allowed for all receiver methods.

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.

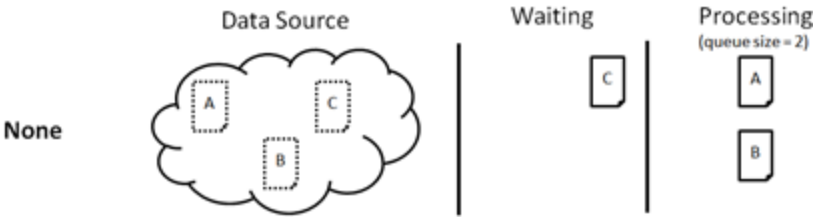


The queue size of a background process with chain transactional settings must be one (1). A chain transactional setting does not allow multiple background process to work in parallel / concurrently regardless of the queue size. By definition, no new background processes can be generated while one is active. The default 'Maximum number of waiting processes' is 1000.

Chain is allowed for all inbound receiver methods and for outbound select data endpoints.

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.



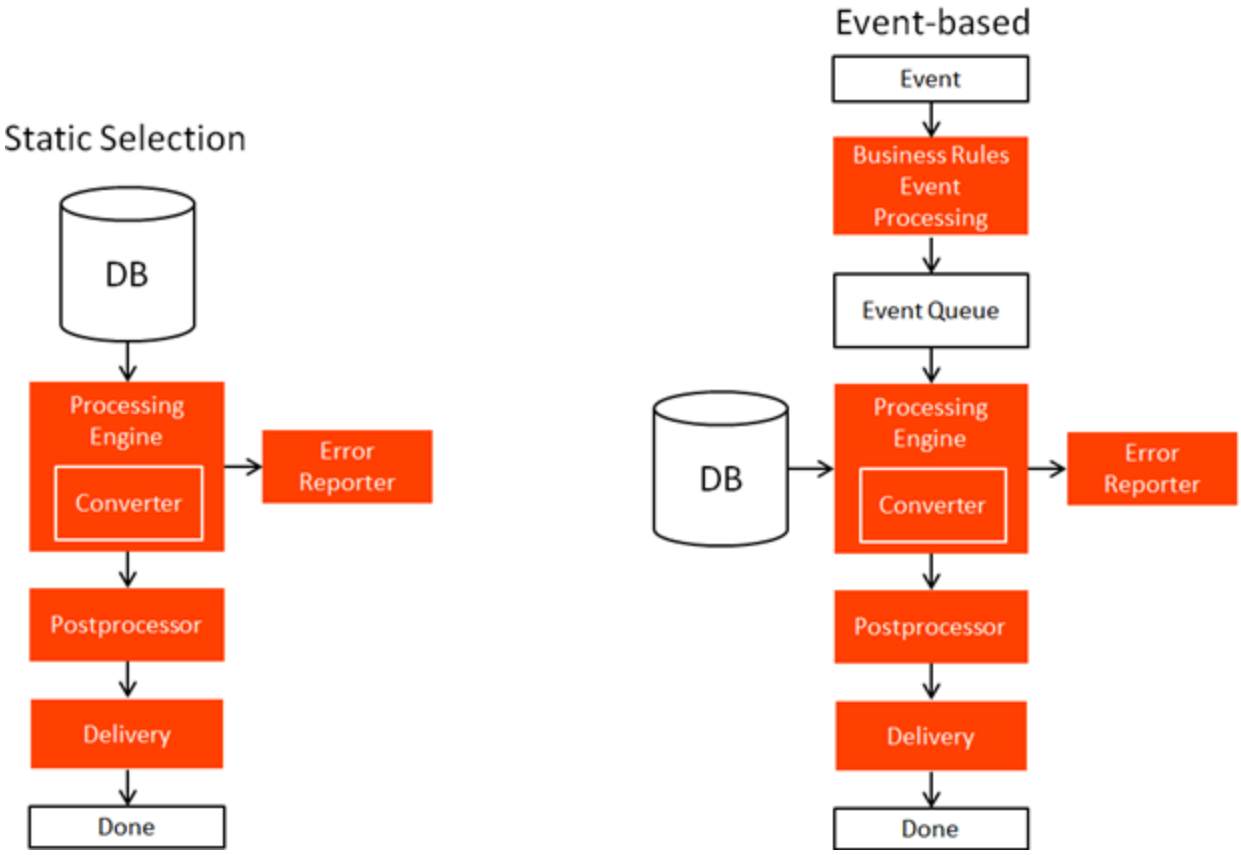
None is not allowed for the inbound 'Hotfolder Using File Sequence Receiver' or 'Hotfolder Using Meta Files Receiver' methods, but can be set for other inbound endpoint receivers. It can also be set for outbound select objects endpoints.

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 OIEP functionality has been created to allow for easy extensions / customizations and can be seen 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, see 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, see the **Configure Endpoint** documentation for an **Event-Based OIEP** or a **Select Objects OIEP**.

Error Reporter

STEP includes a single Send Error Report option to report errors. An email can be sent to a specified address if errors occur when files / messages are handled by the processing engine. For more information, see the **Error Reporter** 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, see 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, see 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 Forward, Rewind, Purge, and Republish**.
- Understand the multithreading functionality on an OIEP as described in **Event-Based OIEP Multithreading Support**.
- 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**.

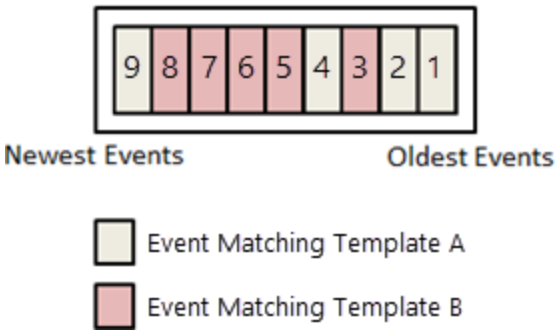
Event-Based OIEP Event Batching

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 strict 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, see **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, see the **Events** section of the **System Setup / Super User Guide** documentation.

Event-Based OIEP Examples

This section includes event-based OIEP configuration examples for semi-advanced to advanced cases:

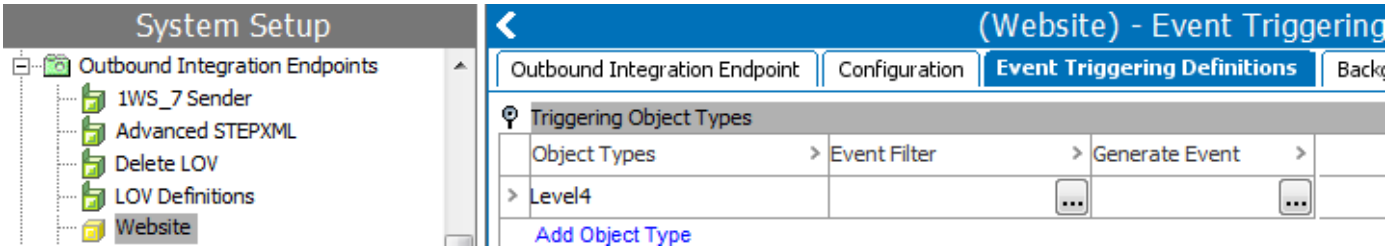
- Event-Based Example Business Rules for Derived Events
- Event-Based Example Publishing of Leaf Products with Inherited Data
- Event-Based Example Upward Inheritance
- Event-Based Example Products Linked to Classification

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.



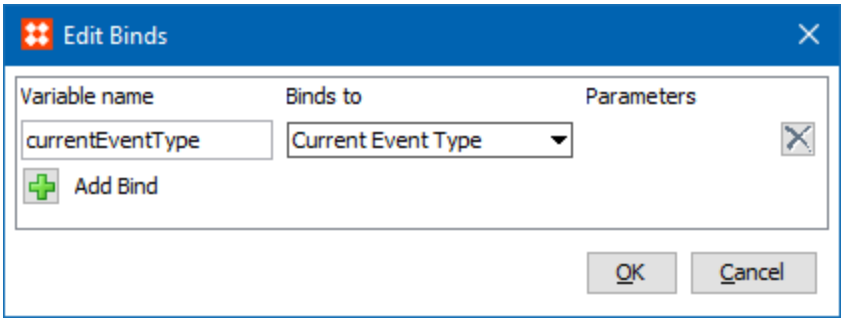
Important: The following scripts are only an example and should not be used as-is without thorough testing, including updating to match object and link types that exist on your system.

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.

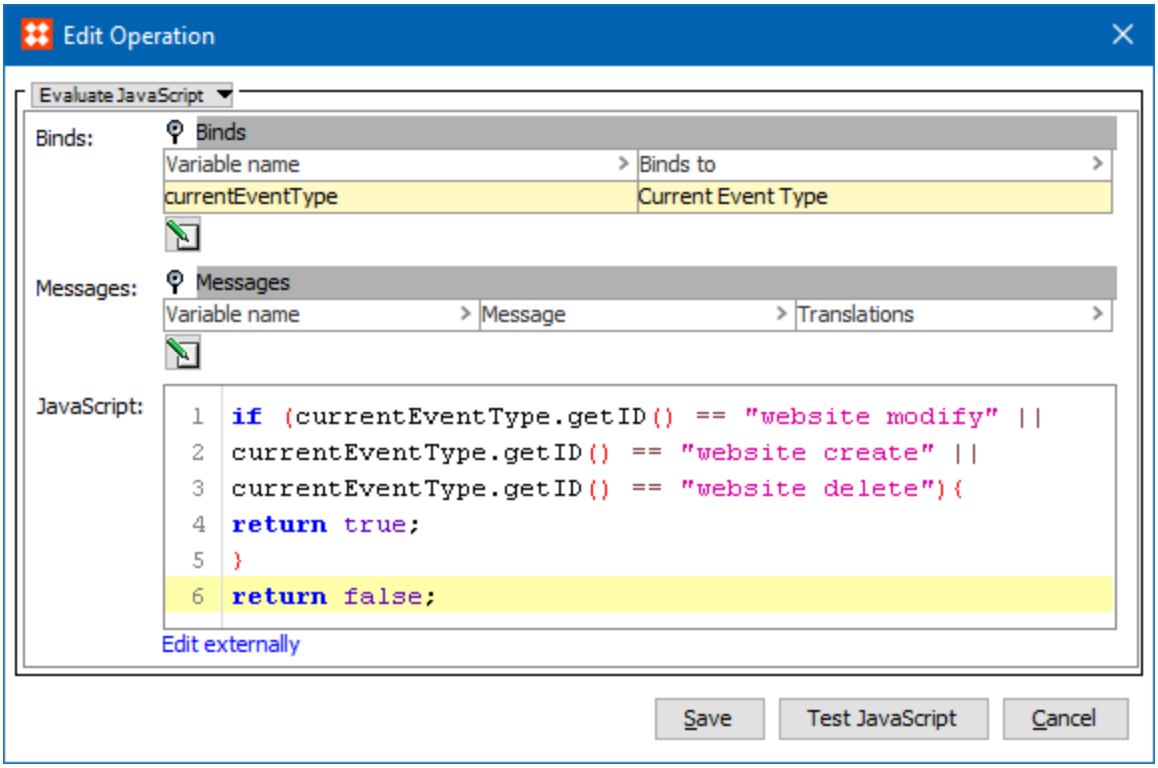
Event Filter - Using a bulk update or business action could generate Republish events, but they would be registered as core 'create' events. Instead, generate a derived event and then configure the relevant OIEP to discard all events that are not of this type.

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.



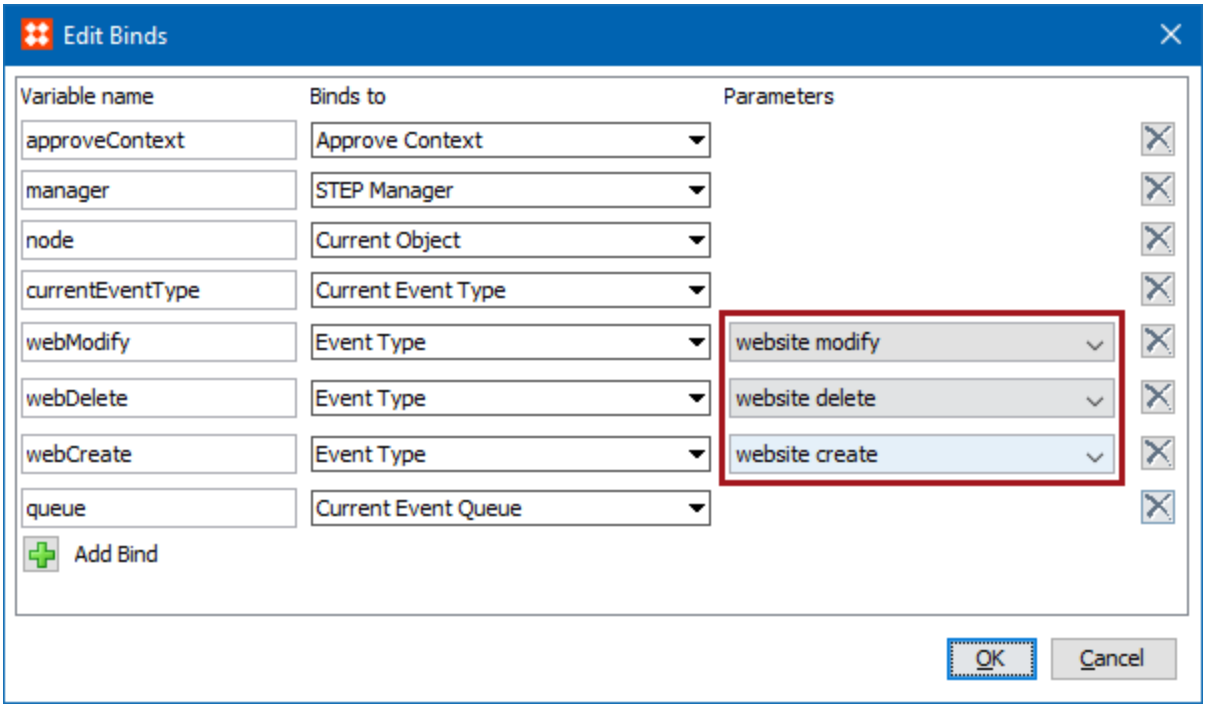
Use the following JavaScript code to identify the derived event types:

```
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, click the **STEP API Documentation** button on the STEP Start Page.

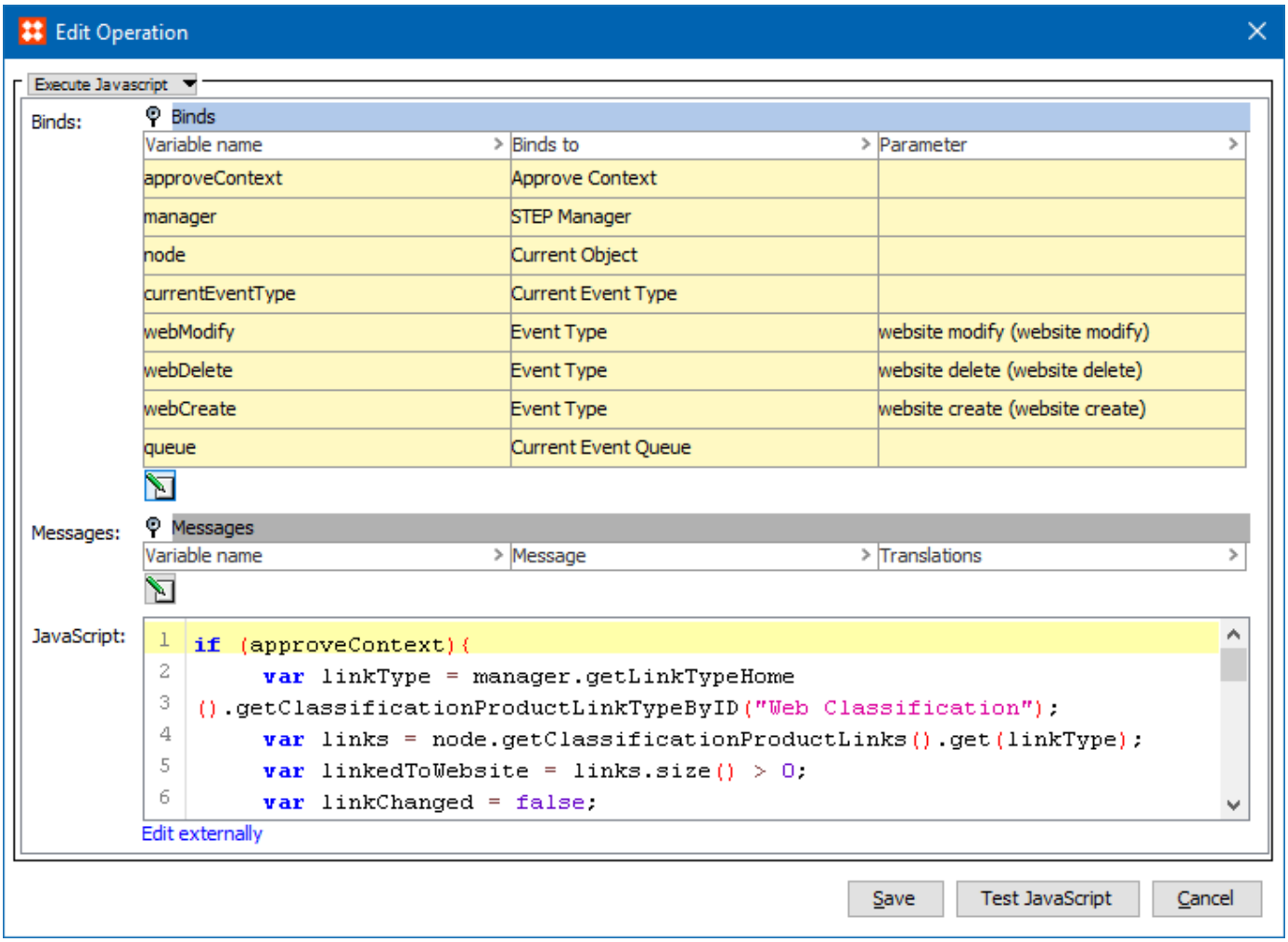
Create a new **business action** and make it valid for the appropriate object types. Use the **Execute JavaScript** operation and add the following binds, including binds for the derived events:



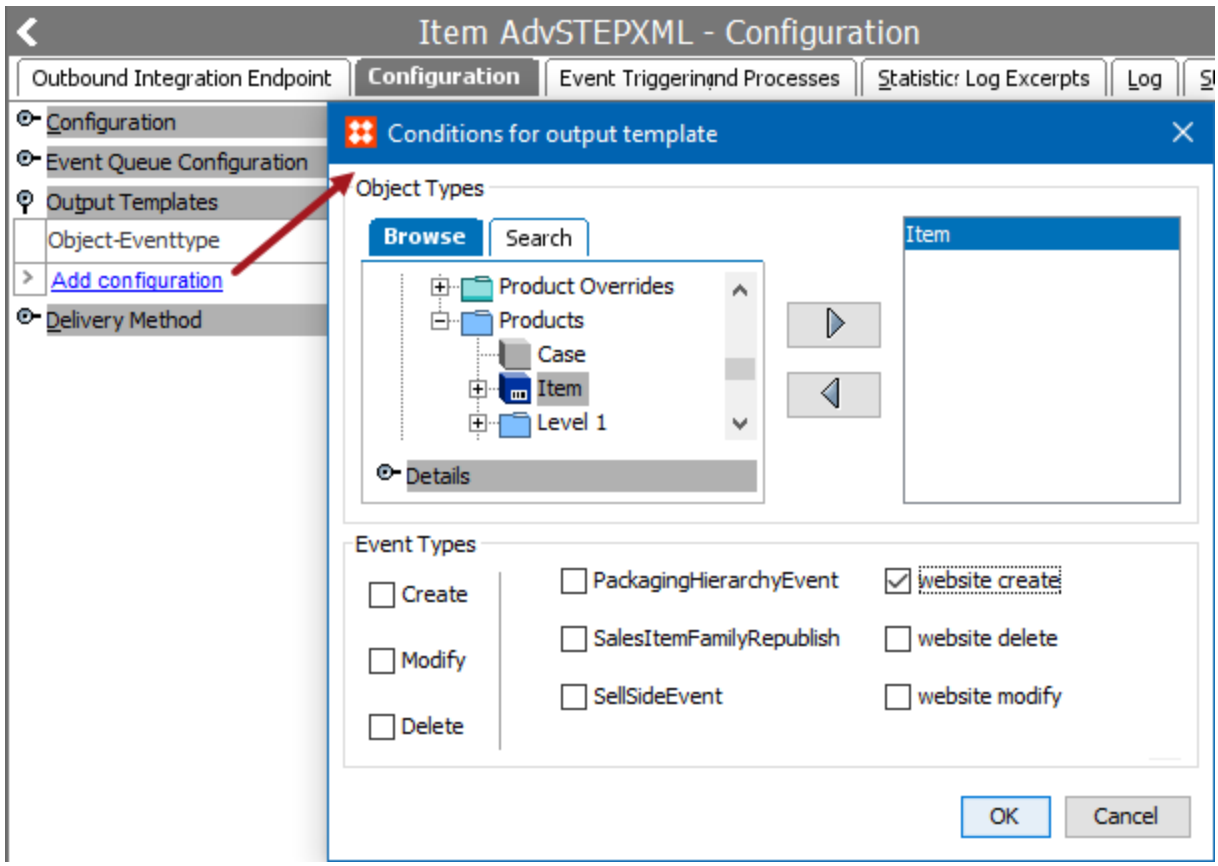
Use the following JavaScript code to create a derived event based on the approved change on the product:

```
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. See **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

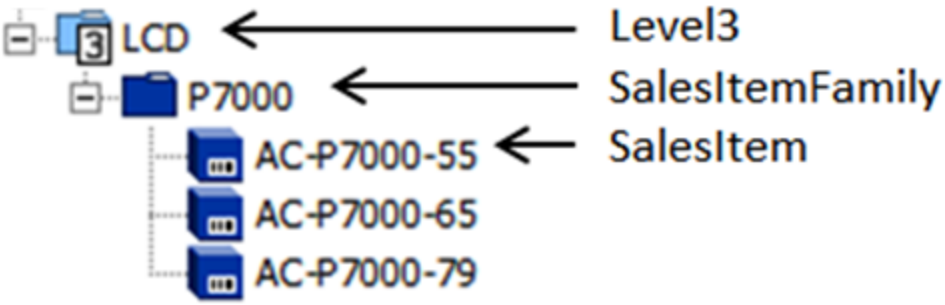
Problem: You only want to publish the leaf product level to a downstream system and the leaves inherit data from ancestors, so events will not be generated for the leaves when inherited data changes.

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

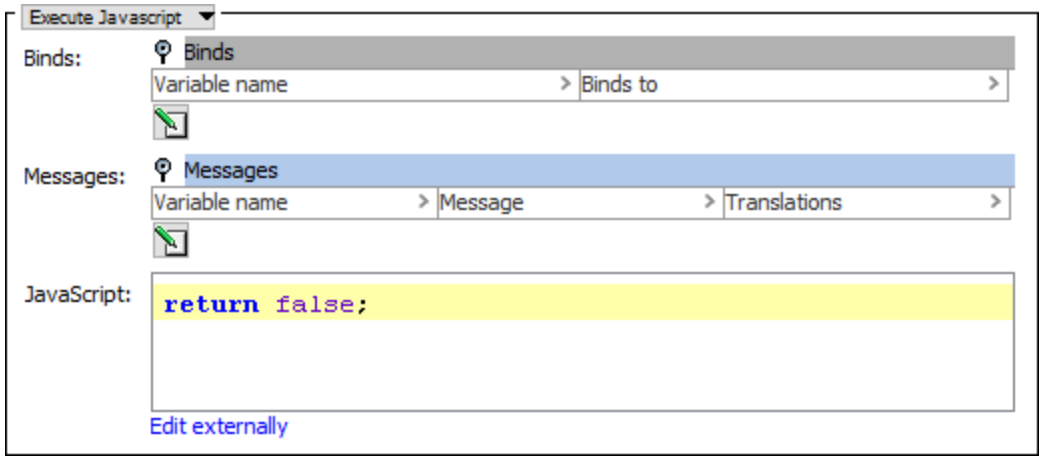
Object Types	Event Filter	Generate Event
Sales Item		
Level 3, Sales Item Family	LP Always False (LPAlwaysFalse)	LP Republish Sales Item Descendants (

Add Object Type

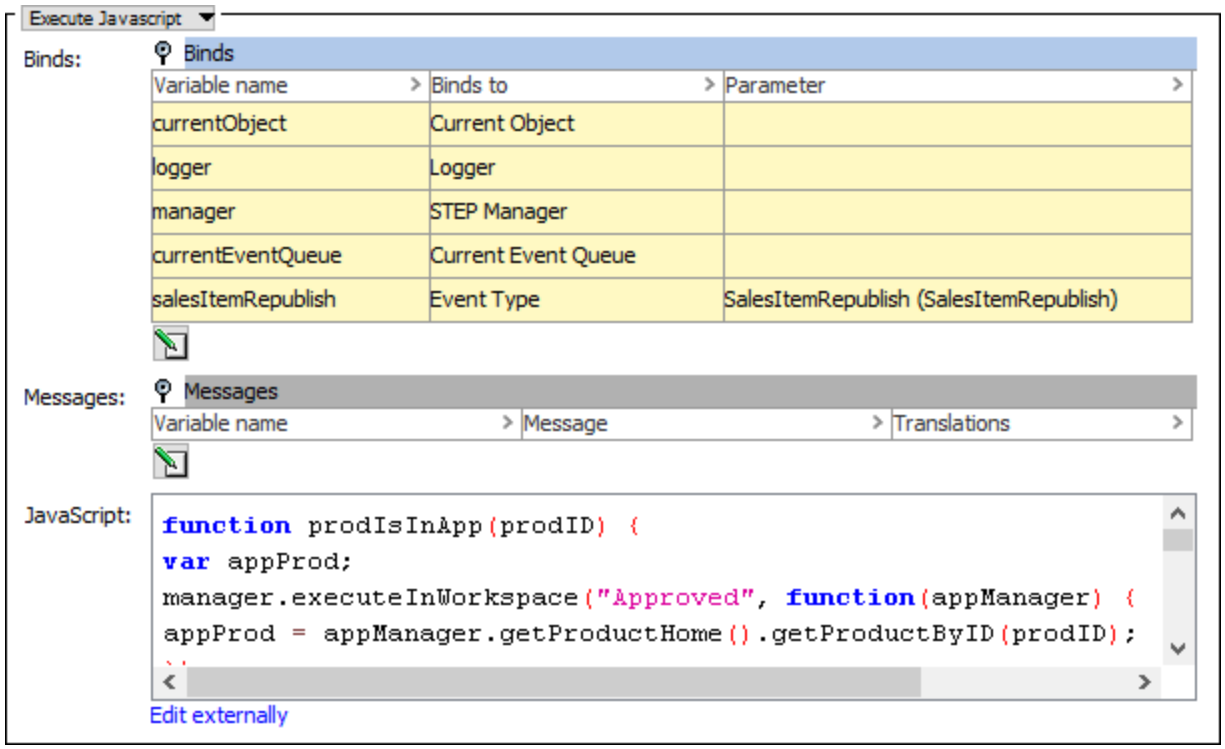
No filters or generators are used for Sales Items, so events generated for this object type always get registered in the queue.

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):



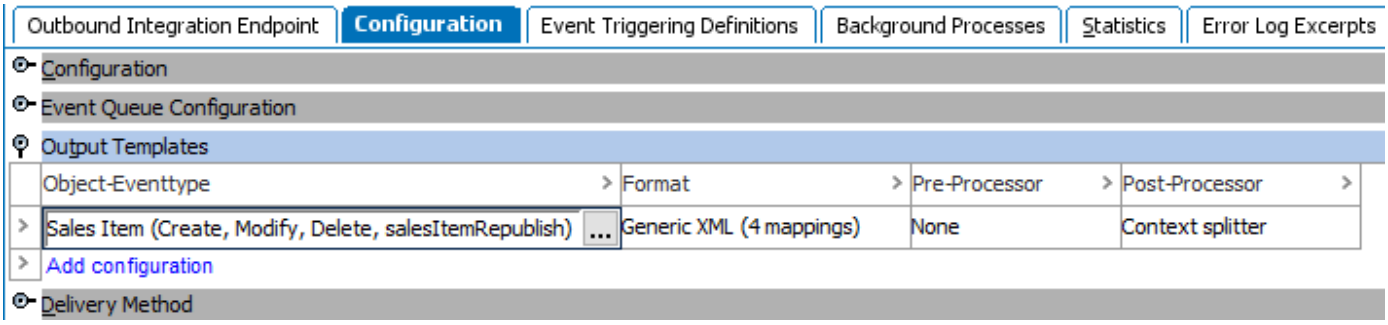
Notice the 'Event Type' bind to a derived event named 'salesItemRepublish.' This is the event that will be generated for descendant Sales Items and it must be defined in advance via the 'Derived Events' node in System Setup.

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, see 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.

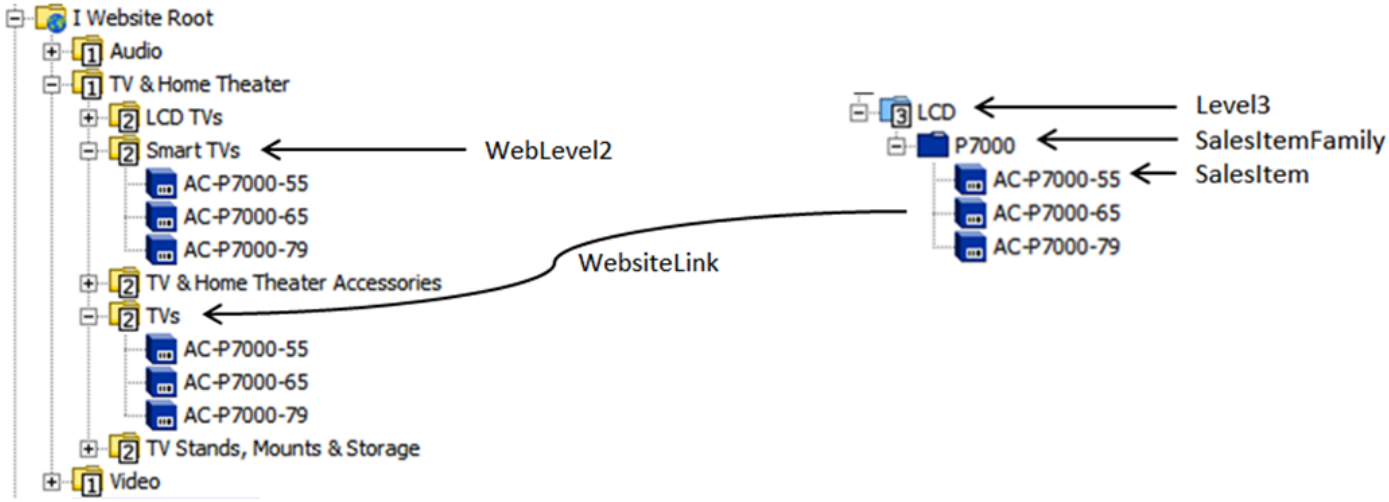


Event-Based Example Products Linked to Classification

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.

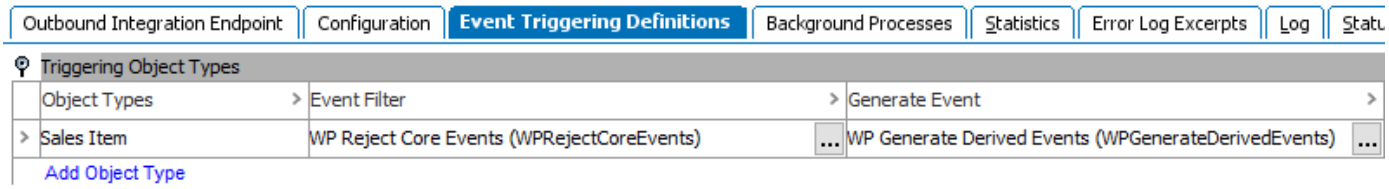
A better approach is to use three derived events with IDs 'webAdd', 'webModify', and 'webRemove' in an event-based OIEP. The conditions for triggering the different events are listed below:

- **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

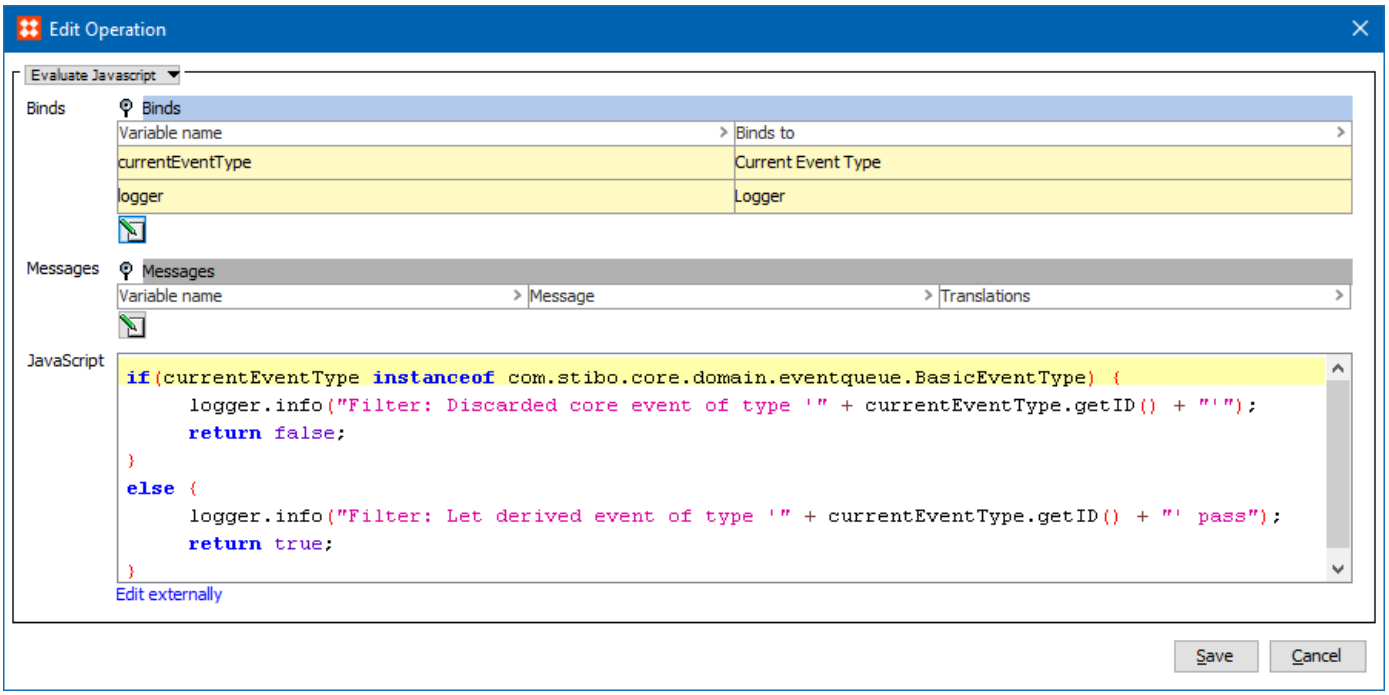
- **webRemove** - Earlier approved Sales Item where WebsiteLink has been removed is approved; Sales Item with WebsiteLink is approve deleted (core Delete Event)

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.

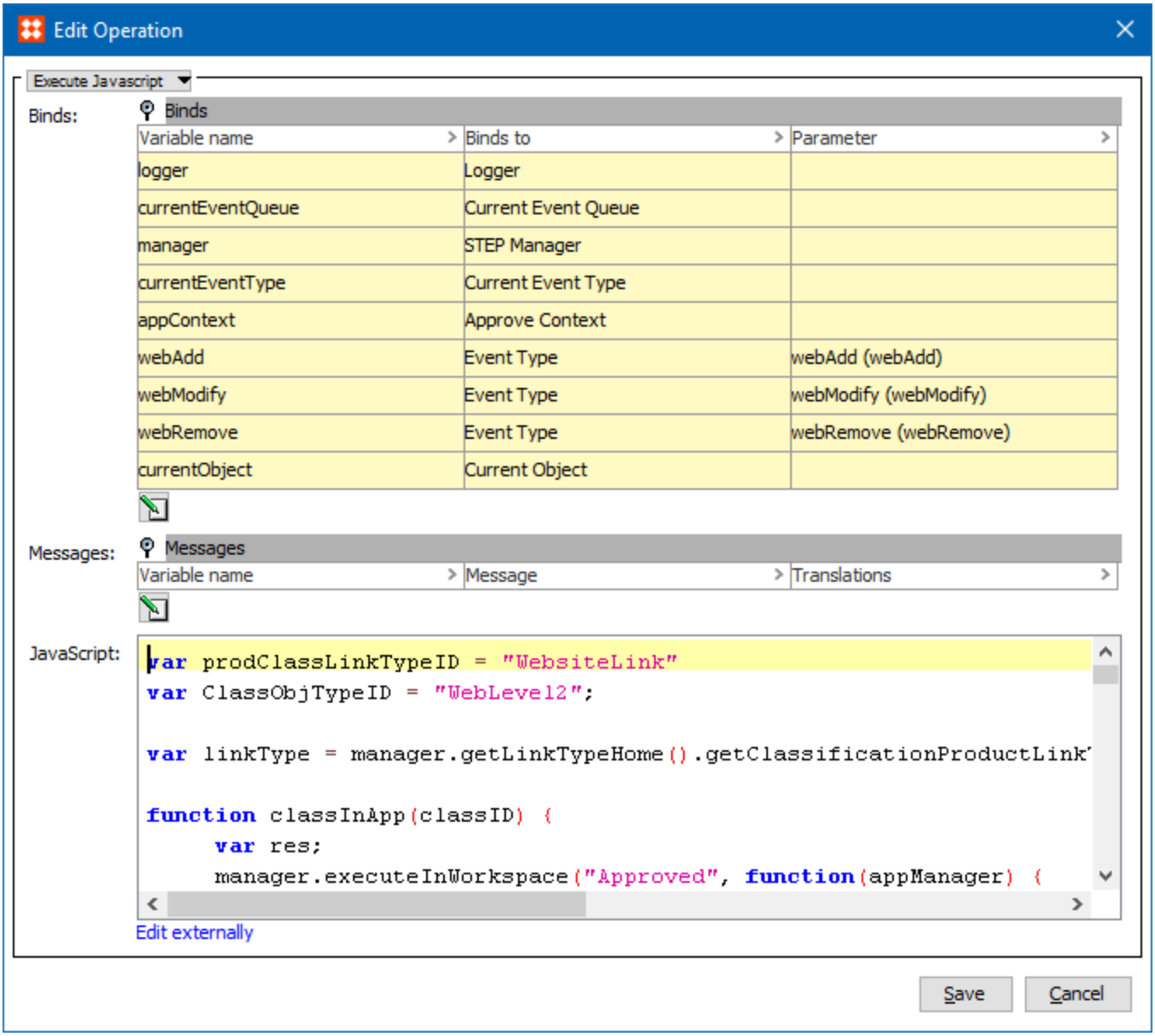


If logging is not required, the script could be:

```
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:

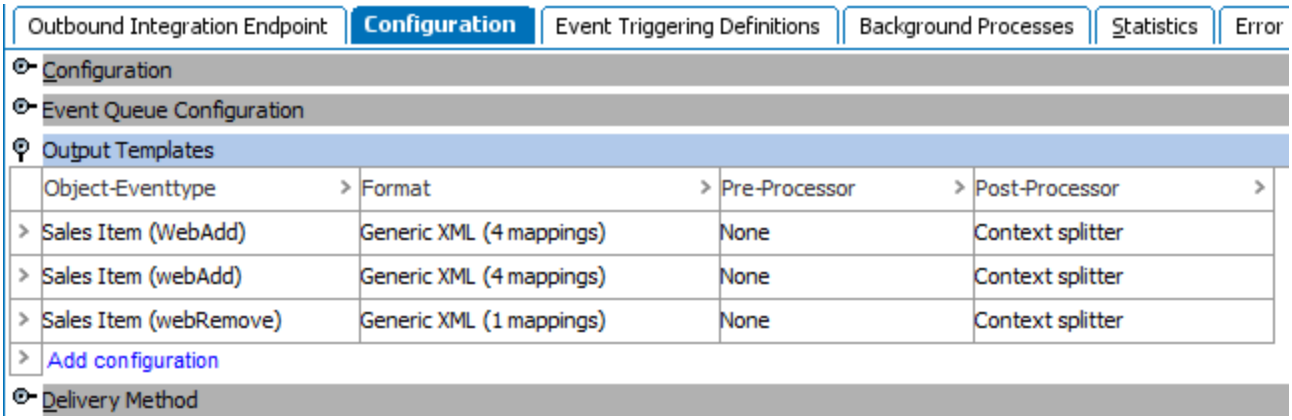


Keeping in mind that the following script is strictly to illustrate the solution, and can be difficult to read, this list outlines the intended logic:

- 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.

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.

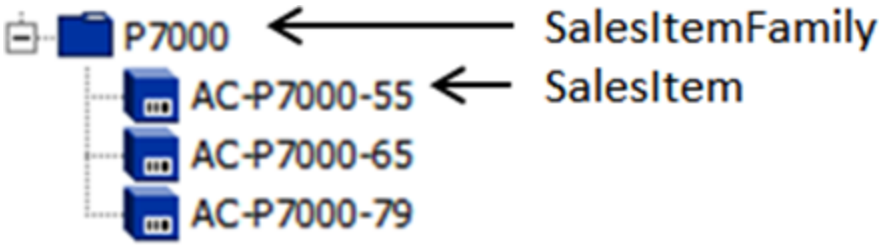
Solution:

In an event-based OIEP:

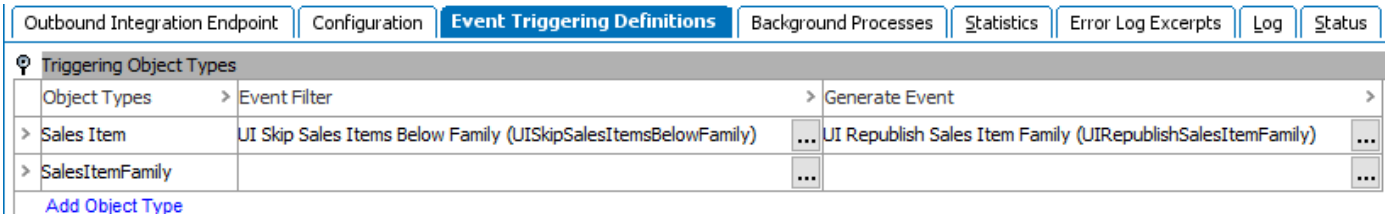
- If children get exported along with the parent, whenever an event is generated for a child with appropriate parent, catch it, discard it, and generate a derived event for the parent instead.
- If children are not published with parents, do not discard the children events.

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:



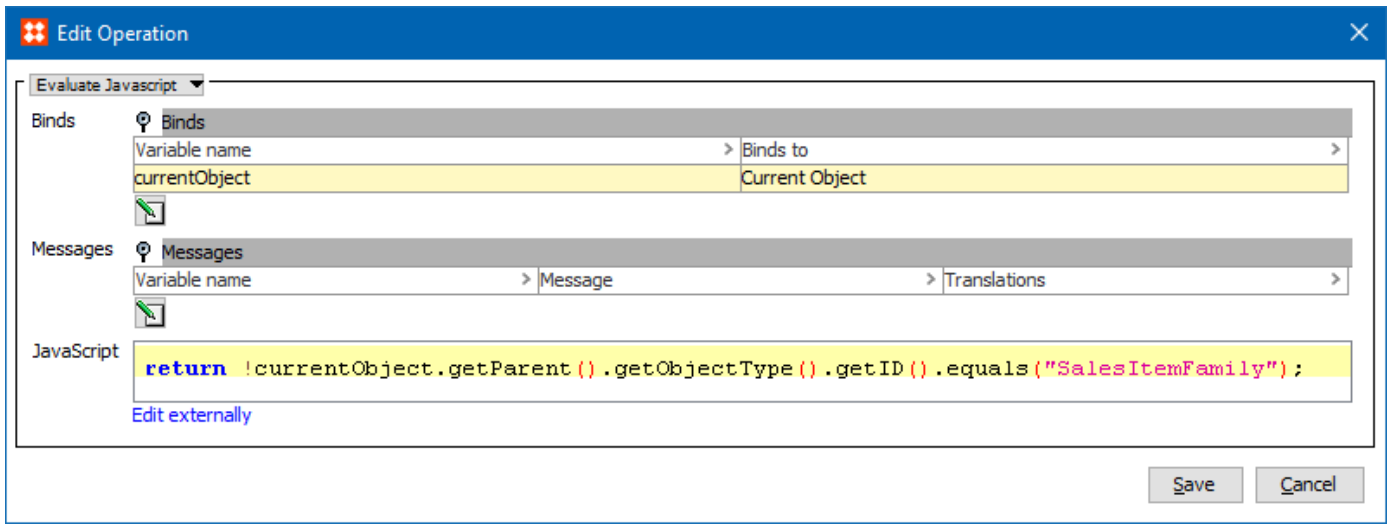
Use the following triggering object types setup:



No Filter or Generator is used for the Sales Item Family Object Type as all Events should pass for this type.

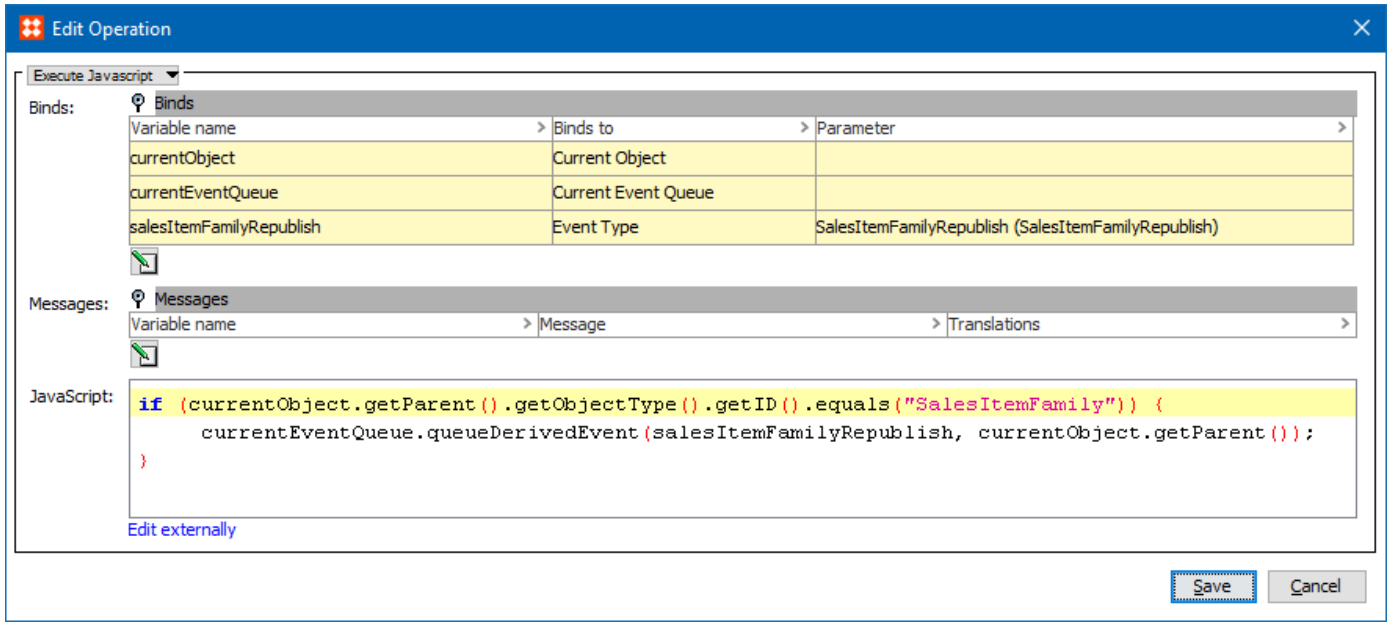
Use the following logic for the Sales Items:

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:



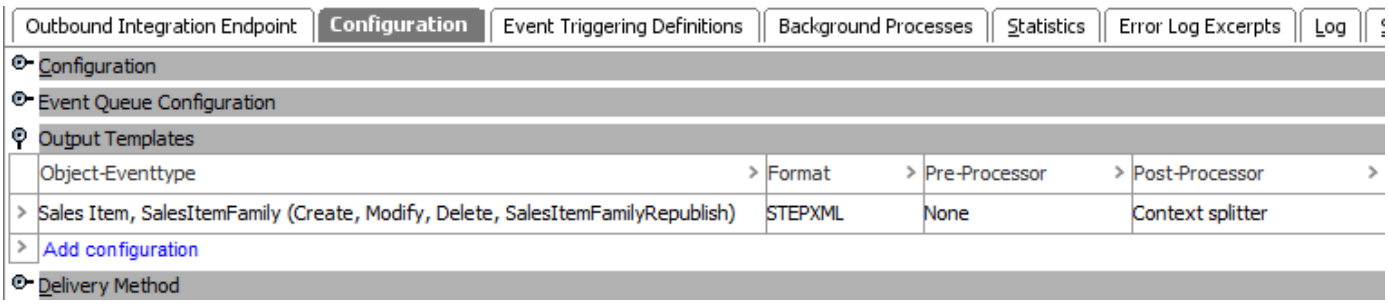
If Sales Items can only exist below Sales Item Family objects, instead an 'Always False' condition like the one used in **OIEP Example Publishing of Leaf Products with Inherited Data** can be used instead.

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.

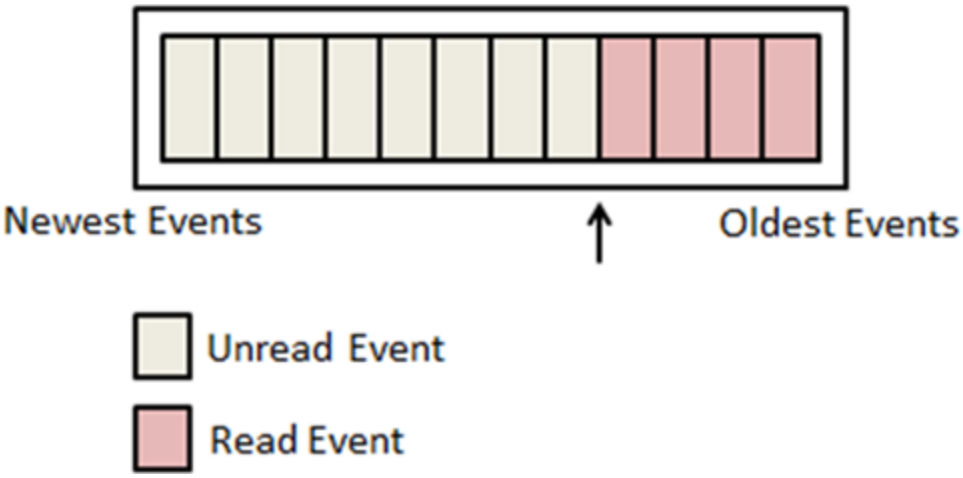

Outbound Integration Endpoint
Configuration
Event Triggering Definitions
Background Processes
Statistics
Error Log Excerpts
Log
Configuration
Event Queue Configuration
Output Templates
Object-Eventtype
Format
Pre-Processor
Post-Processor
Sales Item, SalesItemFamily (Create, Modify, Delete, SalesItemFamilyRepublish)
STEPXML
None
Context splitter
Add configuration
Delivery Method

Event-Based OIEP Forward, Rewind, Purge, and Republish

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', see the **OIEP - Event-Based - Event Queue Configuration Flipper** documentation.

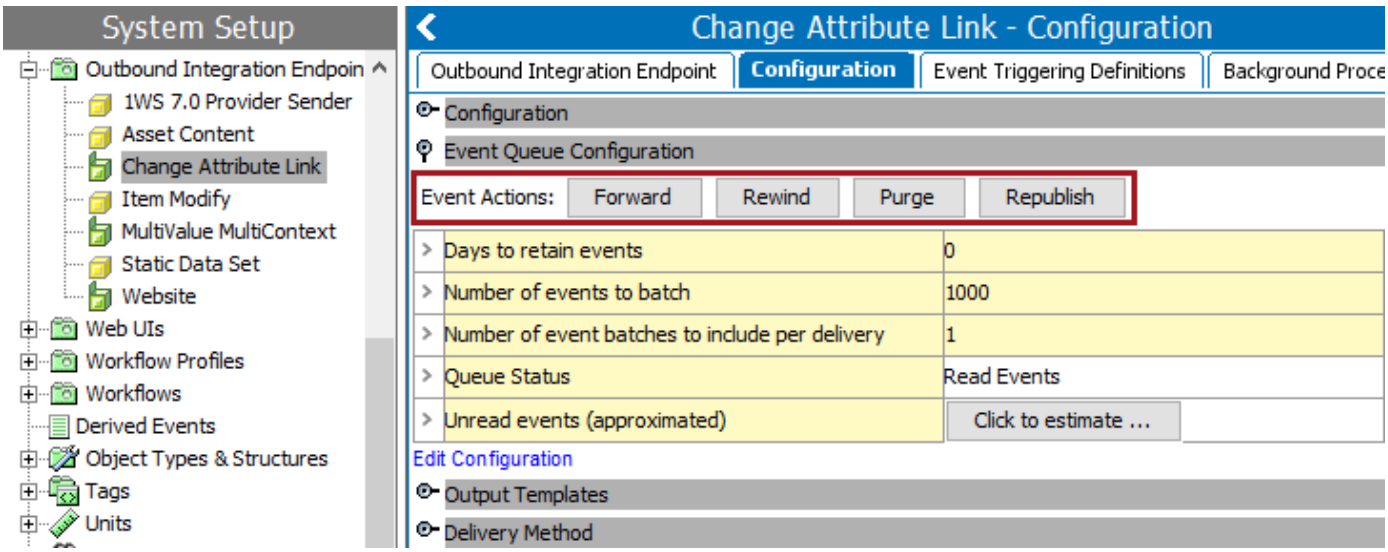
For example, if the queue is configured to retain events after they have been read, read events can be distinguished from unread events, as illustrated by the arrow in the following image.



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, see the **Event-Based OIEP Queued Events** 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.

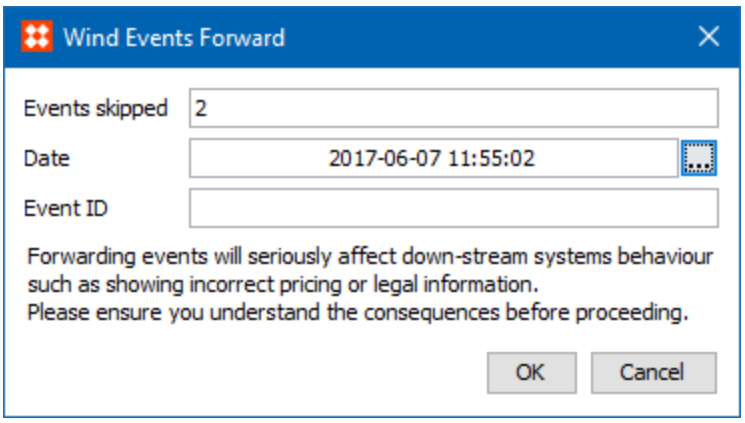
Important: Use caution when forwarding unread events as it may affect the downstream system's data, since the previous changes would be ignored by this activity.

---

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.



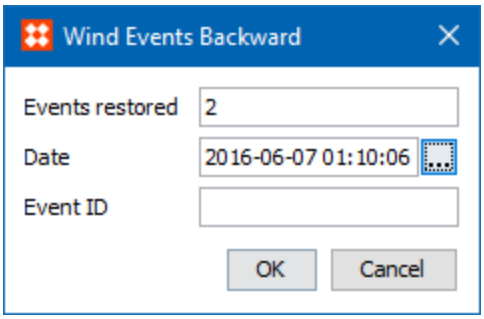
Rewind Events

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.

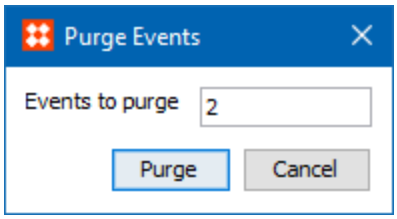


Purge Events

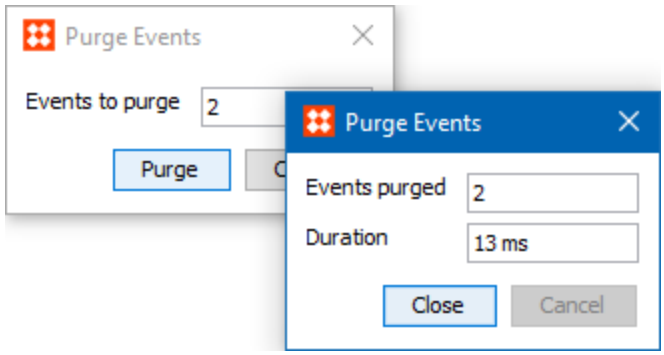
The Purge action is only available when the parameter 'Days to retain events' is set to a number greater than zero during creation or configuration of the OIEP and there are events in the queue and read events are older than the 'Days to retain events' parameter setting. This option is typically used 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.

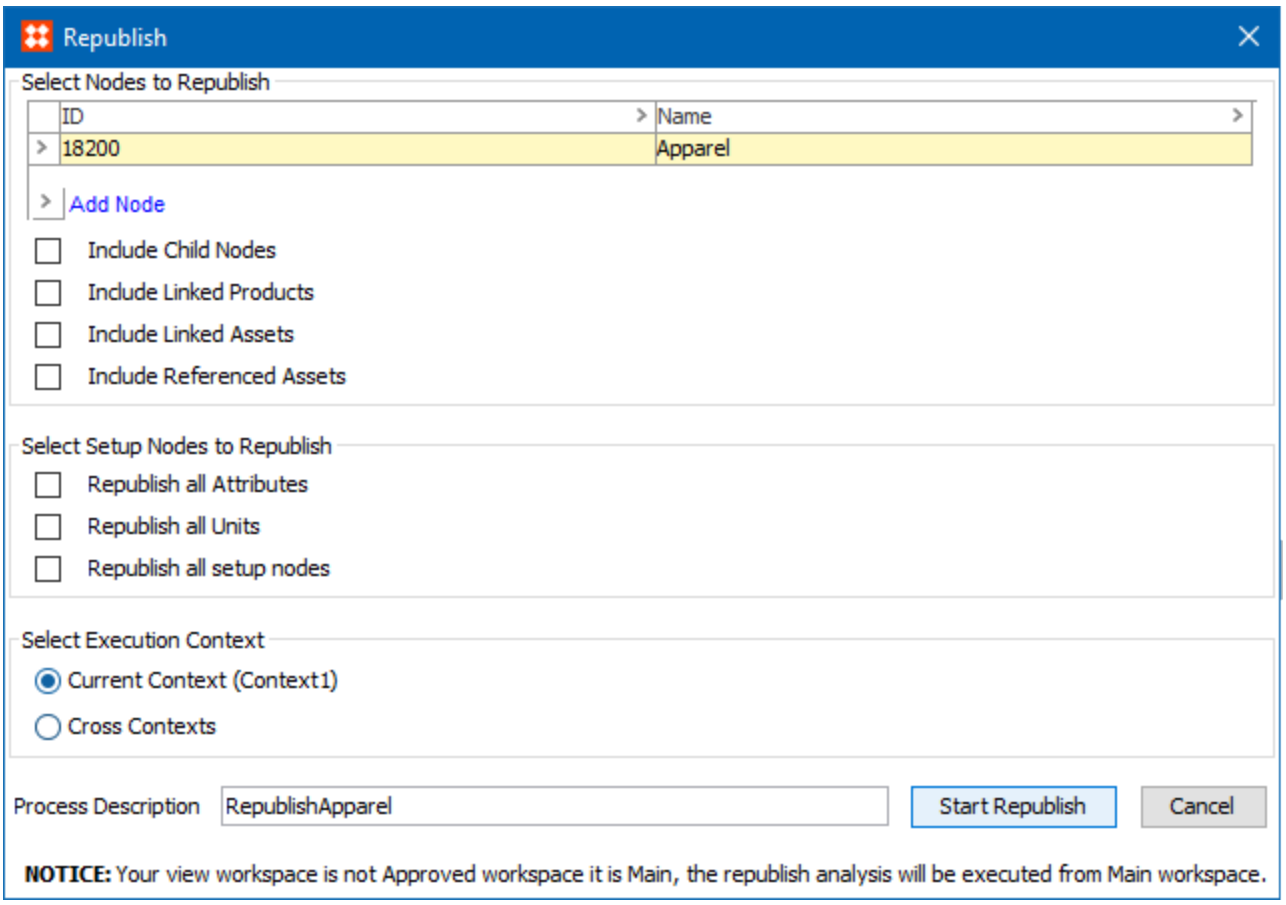


Note: When the number of read events in Purge Events shows 0, forward events to the current date and attempt to purge.

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, see the **Republish** section of the **Maintaining Collections** topic in the **Getting Started / User Guide** documentation.
- Via a business action using the Send Republish Event option, which is available from a business action operation when the 'matching' add-on component is activated. For more information, see 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, see the **Send Republish Event Operation** topic in the **Bulk Updates** documentation.

Event-Based OIEP Multithreading Support

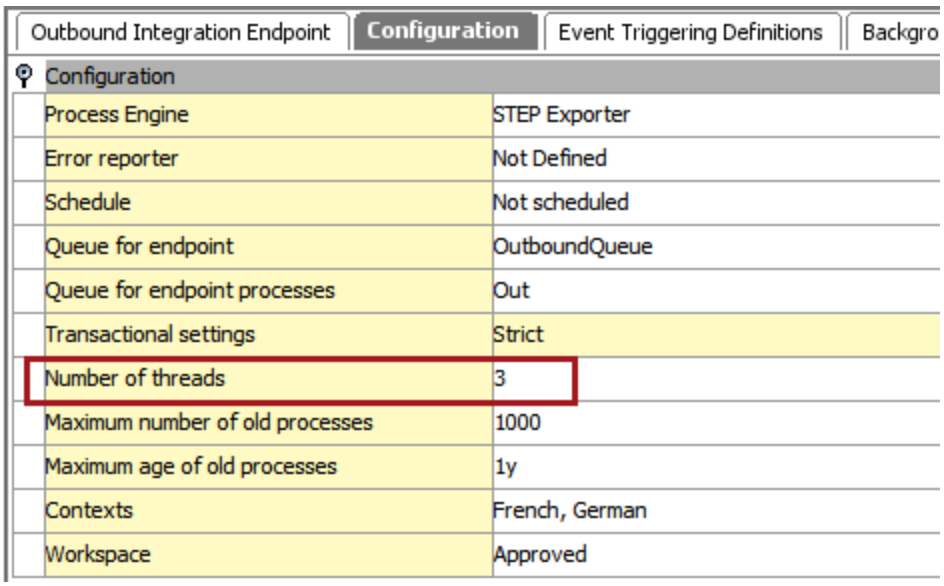
To increase performance when exporting from an Event-Based Outbound Integration Endpoint (OIEP) you can increase the number of threads for a given OIEP. Common setup is to use multithreading when there is a large amount of data going to a downstream system on a regular basis, and when the downstream 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

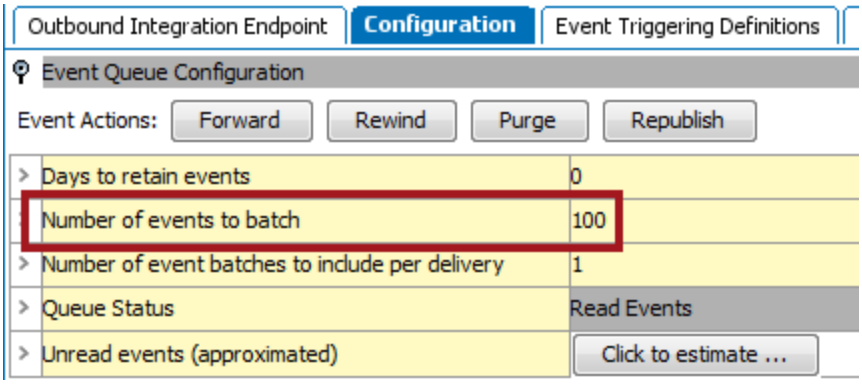
application server log: Number of threads are [x]. Number of batches per delivery will be set to 1 for integration endpoint [ID] as this is the only allowed value when multiple threads are used. Requested number of batches were [y].

When all threads are complete, the batch will still yield 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 Queued Events

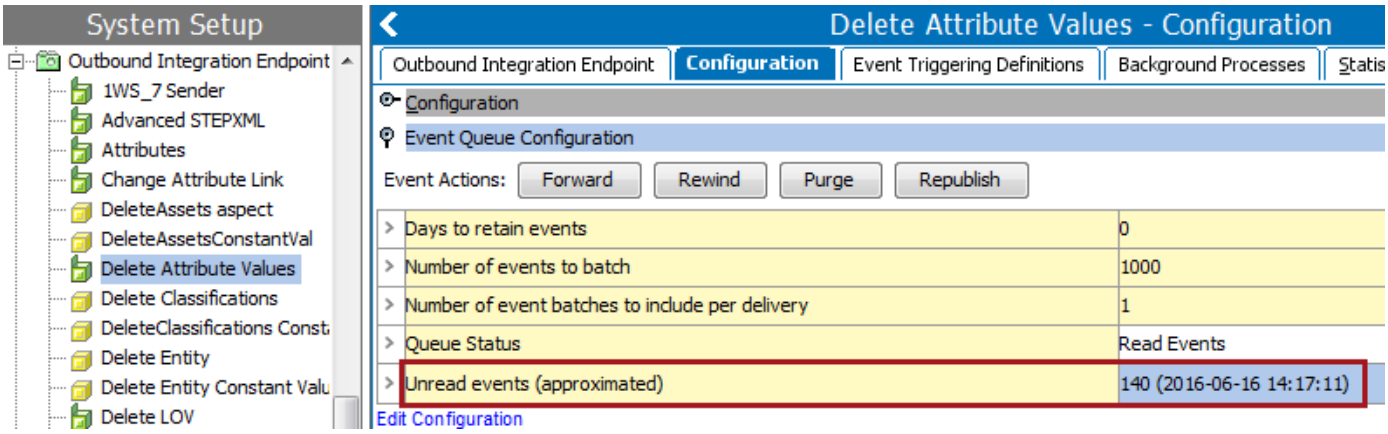
As events are found that meet the criteria set in the event-based OIEP's configuration and event triggering definitions, an estimate and details are available for review in System Setup.

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.

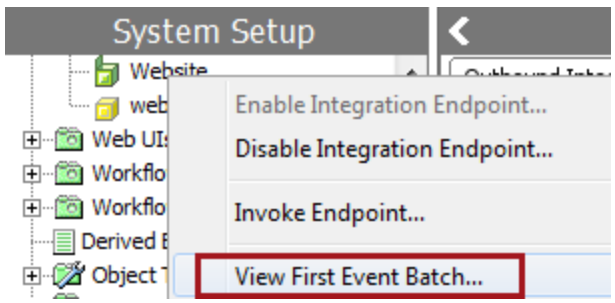


Using the forward, rewind, purge, or republish events options causes the number of unread events on the queue to change. For more information, see **Event-Based OIEP Forward, Rewind, Purge, and Republish**.

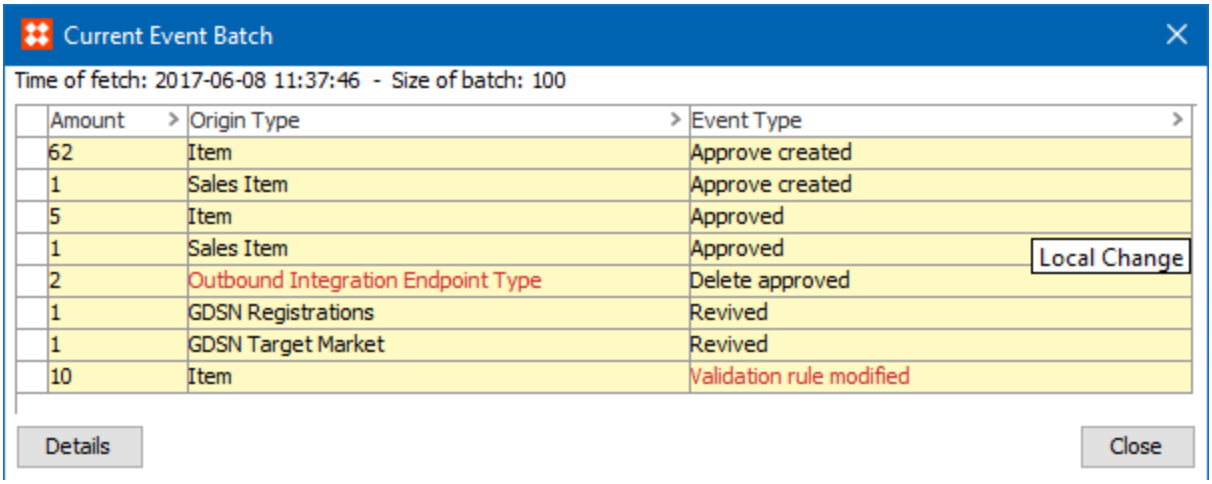
View First Event Batch

This option allows you to view more than just a count of the events.

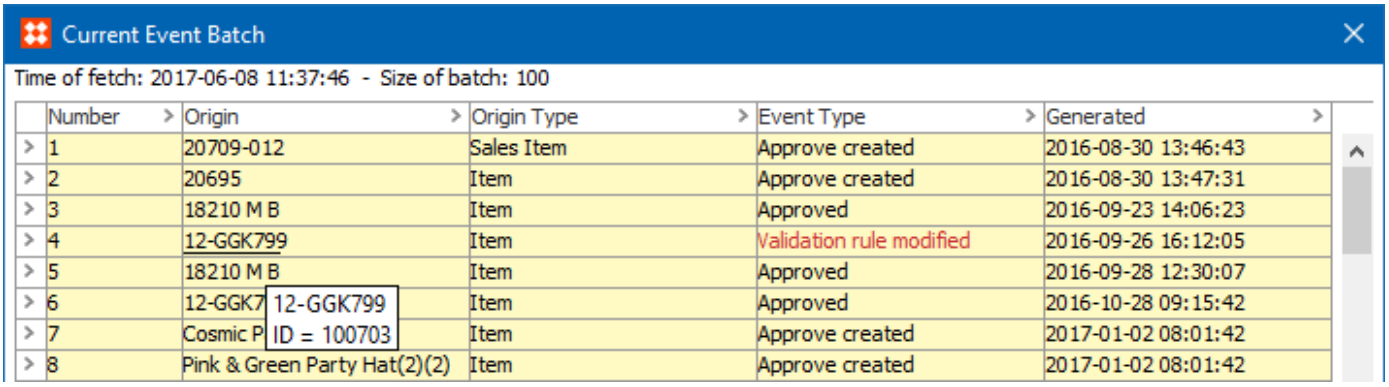
1. Right-click the OIEP and select View First Event Batch.



2. The Current Event Batch dialog displays the first 100 unread events on the queue. Derived events are represented by a number instead of by their ID.

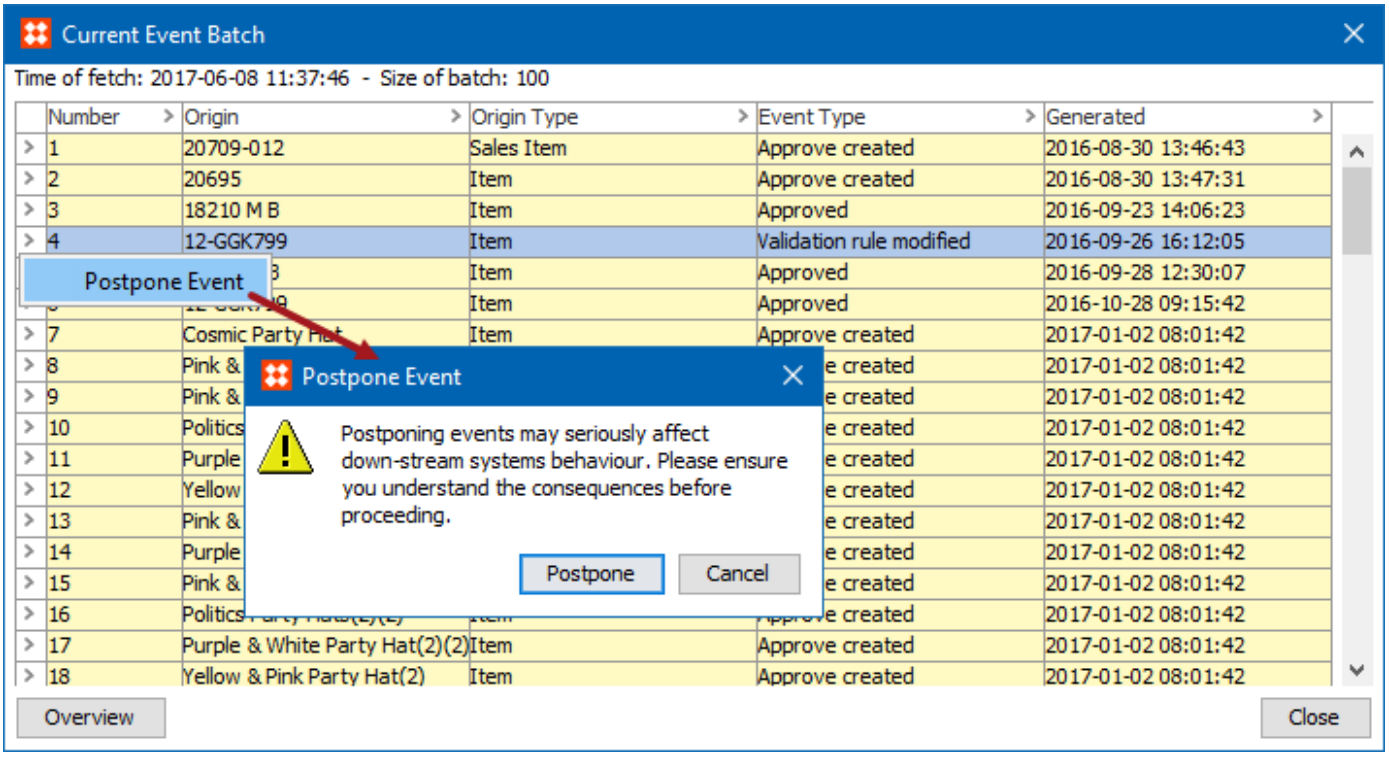


3. For more information about the events, click the **Details** button. The Origin column includes a link to the object being reported by the event.



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.

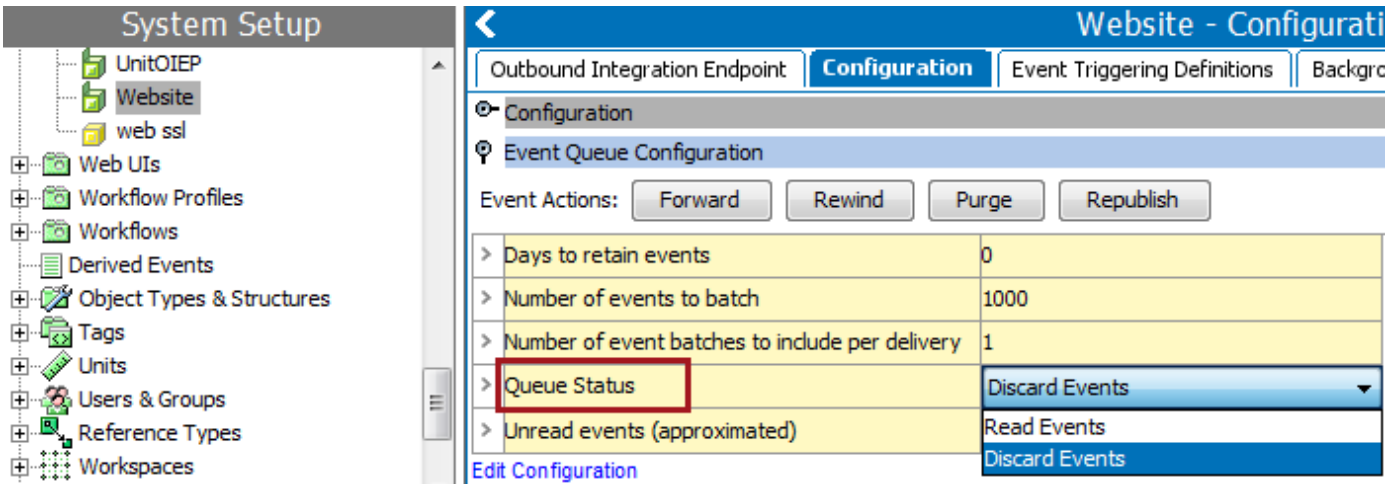
Common setup includes the following combination of queue status and endpoint status settings:

- **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, see the **Running an Outbound Integration Endpoint**.

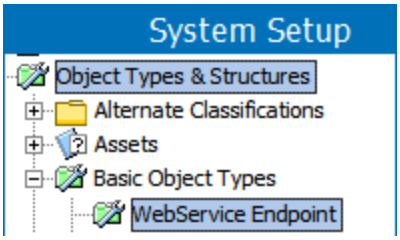
Web Service Endpoint

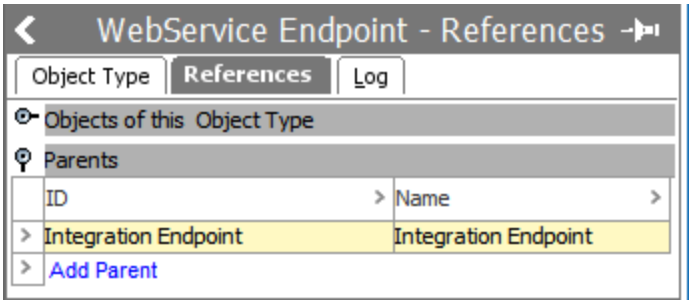
A Web Service Endpoint is a setup entity that can be used to parameterized STEP web services built to receive request data and transmit response data. One type of Web Service Endpoint exists in STEP: Matching and Merging Web Service Endpoint. For more information, see the **Matching and Merging Web Service Type** topic in the **Matching, Linking, and Merging** documentation.

Note: Web Services allow third-party systems to integrate with STEP data.

Prerequisites

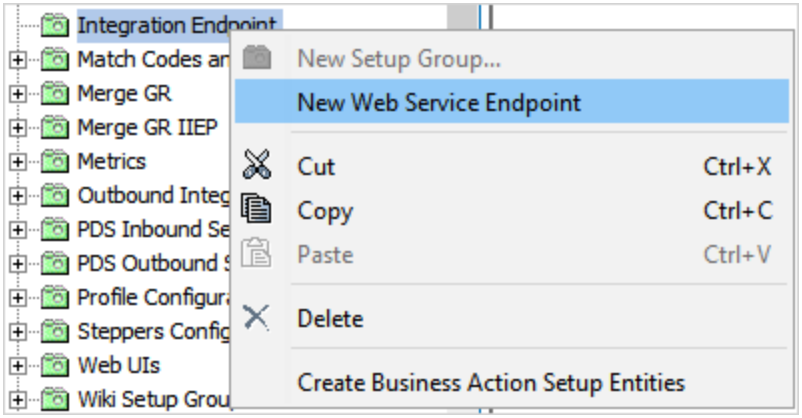
Before configuring and using a Web Service Endpoint, users must create an entity-object type and associate it with a desired parent node.



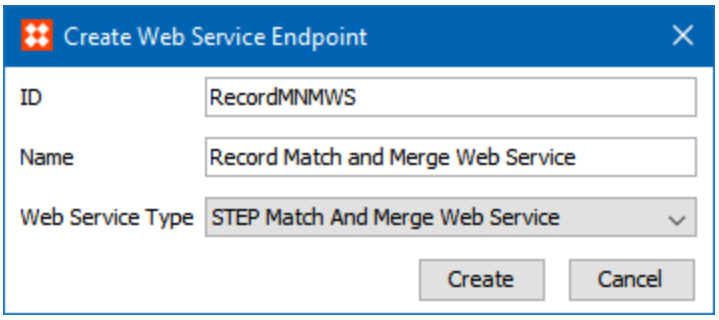


This requires the appropriate permissions to create system objects. For more information, see the **Creating an Object Type** topic in the **System Setup / Super User Guide** documentation.

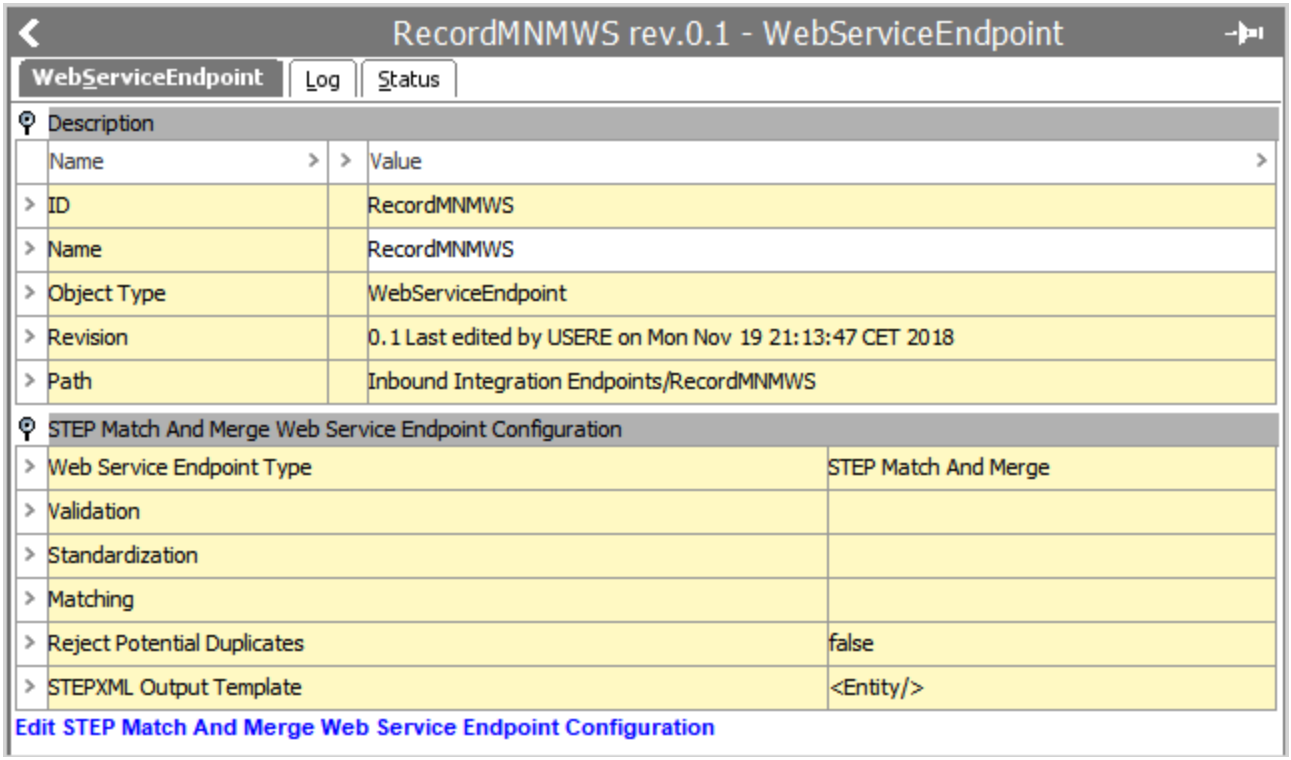
Once the entity object is created, create a Web Service Endpoint in the valid parent node by right-clicking on the node and selecting 'New Web Service Endpoint.'



A new Web Service Endpoint dialog will display requesting an ID, name, and the web service type.



An empty web service will be generated.


RecordMNMWS rev.0.1 - WebServiceEndpoint

WebServiceEndpoint | Log | Status

Description

Name	Value
ID	RecordMNMWS
Name	RecordMNMWS
Object Type	WebServiceEndpoint
Revision	0.1 Last edited by USERE on Mon Nov 19 21:13:47 CET 2018
Path	Inbound Integration Endpoints/RecordMNMWS

STEP Match And Merge Web Service Endpoint Configuration

Web Service Endpoint Type	STEP Match And Merge
Validation	
Standardization	
Matching	
Reject Potential Duplicates	false
STEPXML Output Template	<Entity/>

Edit STEP Match And Merge Web Service Endpoint Configuration