STIBO SYSTEMS
MASTER DATA MANAGEMENT

USER GUIDE

Configuration Management

Release 2023.3 (11.3) – September 2023

Table of Contents

Configuration Management

Management of System Setup configurations across multiple systems can be a complex process. In addition to standard import / export functionality, STEP provides several tools to assist in configuration management. Each of these is described in the subsequent sections.

- Change Packages
- STEPXML Comparison Tool
- Version Control System Integration
- Maintaining Partial Data Sets on Lower Level DTAP Environments
- Transferring STEP Configuration and Sample Data

Export Configuration Definitions as Comments

Using Advanced STEPXML, configuration definitions for the following objects can be exported as comments:

- Bulk Update Configuration
- Business Rule
- Event Processor
- Event Queue
- Export Configuration
- Gateway Integration Endpoint
- Image Conversion Configuration
- Import Configuration
- Inbound Integration Endpoint
- Match Code
- Matching Algorithm
- System Setup objects that reference workflows and/or business rules, such as: Asset Importer, Elasticsearch Configurations, Metrics, Sufficiencies, Value Generators, XSLT Stylesheets, etc.
- Outbound Integration Endpoint
- Web UI Configuration
- STEP Workflow

- STEP Workflow Profile
- Transformation Lookup Table Configuration

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 import of these files is not supported (for example, users may not export, edit the comments, and re-import). For exporting business rules in an editable format, refer to the VCSI: Editable Business Rules Format topic.

Inclusion of configuration definitions as comments is accomplished by setting the DefinitionsAsComments tag to 'true' in an Advanced STEPXML template.

For example:

```
<?xml version='1.0'?>
<STEP-ProductInformation DefinitionsAsComments="true">
<STEPWorkflows ExportSize="All"/>
</STEP-ProductInformation>
```

Below is an example of a business rule definition output as comments:

```
0034   <BusinessRules>
0035     <BusinessRule ID="DTPWS" Scope="Global" Type="Action" RunPrivileged="false">
0036
0037 <!-- Definition:
0038 Action #1 (JavaScriptBusinessActionWithBinds):
0039 <config>
0040   <bindings>
0041     <binding alias="gateway" type="GatewayPlugin" contract="GatewayBinding">
0042       Saasaksqa1
0043     </binding>
0044   </bindings>
0045   <messages/>
0046   <javaScript>
0047 var soapEnvelope = "<soapenv:Envelope xmlns:soapenv=\"http://schemas.xmlsoap.org
0048                 "    <soapenv:Header/>\n" +
0049                 "    <soapenv:Body>\n" +
0050                 "       <urn:getWorkspaceList>\n" +
0051                 "          <in0>\n" +
0052                 "          </in0>\n" +
0053                 "       </urn:getWorkspaceList>\n" +
0054                 "    </soapenv:Body>\n" +
0055                 "</soapenv:Envelope>";
0056 var result = gateway.post().bodyContentType("text/xml").path("/IntegrationAPI/Wo
0057 logger.info("Result: " + result);
0058   </javaScript>
0059 </config> -->
0060       <SetupGroupLink SetupGroupID="SAASMTLSActions"/>
0061       <Name>DTPWS</Name>
0062       <OnApprove ApproveSetup="Never"/>
0063       <Configuration>H4sIAAAAAAAAAJVUwU4bMRA9N19h7QElqmrDrUqyQUBDm6pIiFBxiYSGzbB
0064       <ValidObjectTypes AllObjectTypesValid="true"/>
0065     </BusinessRule>
```

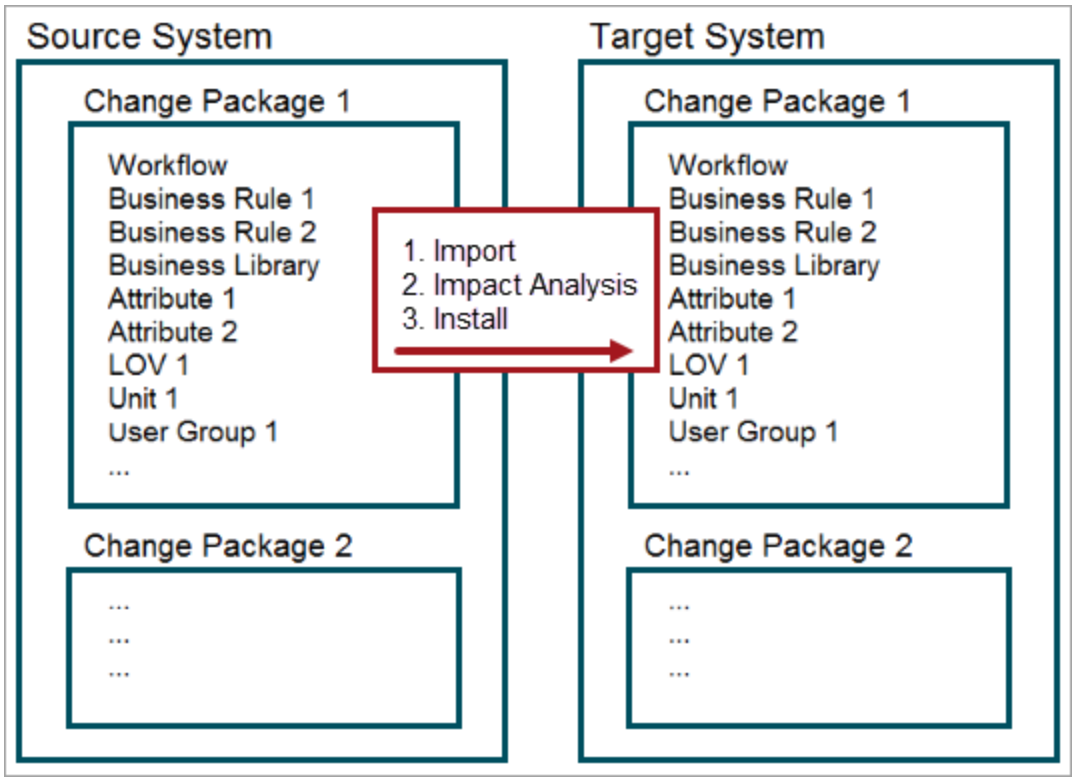
Note: The content of the comment field is not part of the STEPXML XSD and therefore Stibo Systems reserves the right to change the format of the output content at any time.

Change Packages

A change package is a way to group, analyze, and migrate STEP configuration changes between systems in a larger system landscape. Change packages are designed to:

- Minimize offline tracking of configuration changes
- Lessen the chance for introducing faulty configuration changes
- Assist system administrators with impact analysis to enable more informed decision making when deciding what to include for transfer

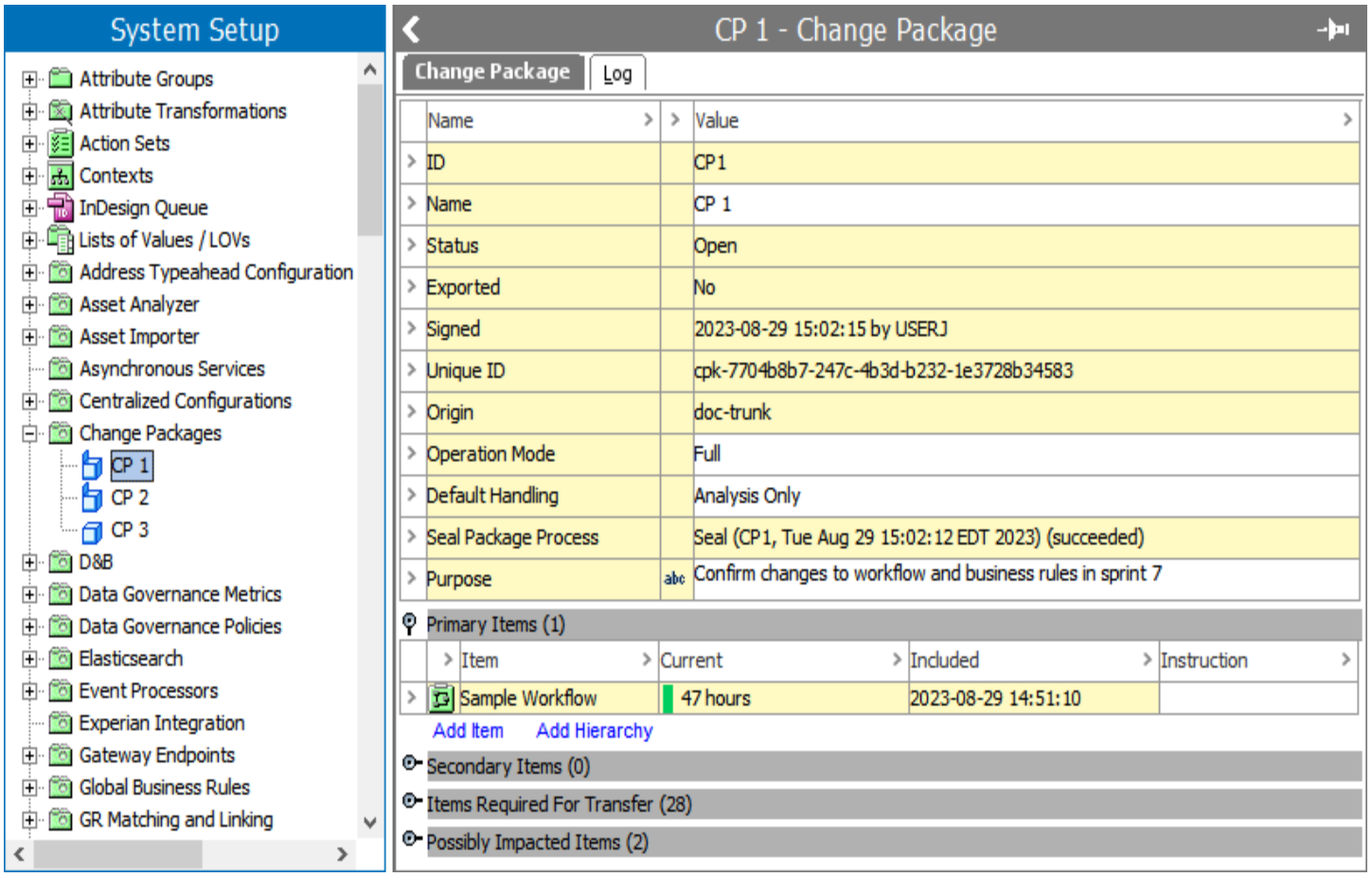
The overall flow of change packages is shown below:



Once created, a user may add or remove items from the package until they are satisfied with the contents. The change package can then be sealed and exported for loading to another system. Upon loading of the change package to the target system, an impact report can be run that helps identify areas that may need to be updated prior to installation and could be impacted upon installation. This provides an indication to the user of how successful the change package will be if installed. The user may then choose to iteratively update the change package in the source system, install the change package as-is, or ignore installation. If installed successfully, the configurations contained in the change package will be applied to the system and will be available for immediate use, unless the object requires manual configuration prior to usage.

Change Package Object

A change package object is a system setup object to house a set of configurations.



The objects that are supported by changes packages are defined in the Change Package Object Support topic.

Change Package Icons and Statuses

It may be necessary to reload or navigate between objects to refresh the change package status.

Icon	Status	Description
	Open	• Change package can be edited; therefore, it is not final or ready for export. Packages are open when created and when re-opened from a sealed state. • An impact report can be run only if it was previously sealed. • Cannot be exported.

Icon	Status	Description
		• Items can be added and removed.
	Sealed	• Change package is locked for editing and ready for export. • Can have an impact report run. • Can be exported. • Items cannot be added or removed. • Generates an event to support automated transfer between systems or external version control system. **Note:** To seal the change package, workspace revisable objects added from the Tree tab (such as, products, classifications, assets, and import / export configurations, etc.) must be approved. For information on workspace revisability, refer to the Generating Revisions topic in the System Setup documentation.
	Dormant	• Change package has been imported from another system but not yet installed. • Can have an impact report run and/or be installed. • Can be exported. • Items cannot be added or removed.

Note: Change packages can be deleted regardless of state and there is no impact to the items contained in the change package. If a change package has been exported to a file saved locally, then a deleted change package can be restored by importing the change package on the system.

Change Package Tab

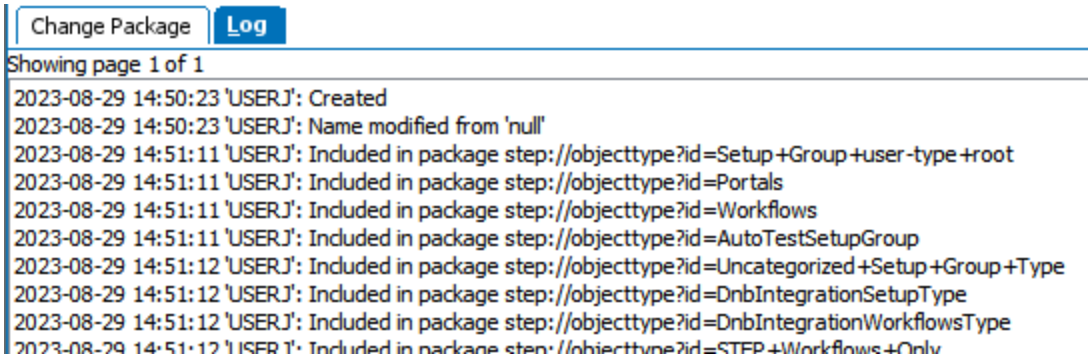
Flipper	Description
	Basic information about the change package including the Status (open, sealed, dormant), if the change package has been exported, if the package has been signed, and where the package originated (e.g., created on the current system or imported from another system). If the package has been sealed or an impact report has been run on the package, additional fields are present with links to these processes when relevant, otherwise these attributes are not shown until

Flipper	Description
	populated. Additional description attributes can be added by the user by adding them to the object type.
Primary Items	A list of objects that have been directly added to the change package, as well as allowing the user to add and remove items from the change package. Users can add a single object (Add Item) or add an object and all of its child objects (Add Hierarchy), when supported. For details, refer to the Editing a Change Package topic. Items in this list are part of the change package and will be created and/or updated on the target system when the change package is installed.
Secondary Items	A system-generated list of objects that are part of the change package due to the addition of a parent using the Add Hierarchy option. This list can only be edited by adding or removing the driving primary item, which is labeled with (Hierarchy Added) in its name. Items in this list are part of the change package and will be created and/or updated on the target system when the change package is installed.
Items Required For Transfer	A system-generated list of objects that may be required for the change package due to relationships with the selected objects. This list can only be edited by the system when adding or removing the driving primary item. These objects are essentially prerequisites for the transfer as the selected primary objects and/or the secondary objects have some dependency on them. The action taken is based on the Handling option setting: • Analyze and Install - The objects are included in the change package as a means of ensuring that the primary and secondary items are successfully transferred, and will be created and/or updated on the target system when the change package is installed. • Analyze Only - The item is only used for the Impact Analysis and not installed. Items with this setting can be promoted to the Primary area when the user decides a prerequisite should be installed. • Ignore - The items are not used for analysis or installation and are not included in the count shown on the flipper.
Possibly Impacted Items	A system-generated list of items that might be affected by the transfer of the change package on the target system. This list can only be edited by adding or removing the driving primary item. These objects are dependent in some way on the primary or secondary object, but are not required for configuration of those objects and are therefore not included in the change package installation, unless promoted to primary by the user.

Log Tab

The Log allows administrators to monitor the activity of the change package. This information, along with the data displayed directly on the change packages, provides detailed logging and tracking for comprehensive audit trails.

Logs for change packages include the item URLs for additions and deletions, as well as additional information, such as sealing, reopening, importing, analyzing, and installing.



Change Package Object Support

Change packages support objects in the tables below at the following levels:

- Full: You can add the object to a change package and it installs successfully.
- Semi: You can add the object to a change package and it can be installed, but manual action is required on the target system to complete the process. For example, enabling an endpoint after installation or updates to a component model or shared configuration properties. Objects that reference business rules or workflows do not have these associations included in the dependency analysis or impact analysis, so there is manual effort to confirm all necessary business rules are included in the Primary Items. Typically, these have an orange background color.
- Addition only: You can add the object to a change package will not be installed. This level provides a reminder of a 'to-do list' item with the option to add details for the installer in the Instruction parameter within the change package, describing the manual configuration. For example, when a scheduled export or bulk update is required, the BGP ID can be searched for in the Add Items window and selected. A BGP is not created on installation, however the Instruction can include details about the timing and frequency, business rule or export configuration and collection required, making an external document unnecessary.
- Unsupported: You cannot add the object to a change package. These can be addressed by adding an instruction for the installer via metadata on the change package itself when a custom attribute is added to the change package object type.

Full Support		
action sets	dimensions	setup entities
assets (individual)	dimension points	setup groups
asset importer	derived events (all, not individual)	status flags
attribute groups	entities (individual)	tags*
attributes	export configurations	transformation lookup tables
bulk update configurations	image conversion configurations	unit groups
business rules	import configurations	units
classification product links	list of values	user groups
classifications (individual)	object types (user-created,	users (service account / externally managed)
collection groups		

Full Support		
collections (searched-based) contexts (individual) data container types	not system owned) products (individual) reference types (individual by type)	value generators workflow profiles workflows

* The Tags node cannot be added to include all tags, but Tag Groups can be included. All tags in a group are included.

Semi Support	Notes
attribute transformations	Impact analysis not available.
asset importer	Ignored impact analysis.
asset push configuration	Manually configure pipeline XML files on application server if necessary.
collections (created from file)	Must recreate from file manually.
eCatalogs	Price / Terms List / Collections from lists must be recreated manually. Ignore impact analysis.
event queues	Enable, set to read events and verify event triggers.
GIEPs	Enable and test connectivity. Configuration property changes may be required.
IIEPs	Enable and verify linked business rules are included.
keys	Deactivate before installation when modifying an existing key and reactivate after installation in all cases.
match codes	Manually configure component model changes.
matching algorithm	Manually configure component model changes.

Semi Support	Notes
metrics	Verify linked business rules are included and Event Processor status.
OIEPs	Enable, set to read events and verify event triggers, if queue-based. Verify linked business rules are included.
sufficiencies	Verify linked business rules are included and Event Processor status.
users	Users created on target systems use temporary password. Reset manually if passwords are managed by STEP.
value generators	Impact analysis not available. CSVs for List Value Generators are not included in change packages. Import CSVs manually.
Web UI	Impact analysis not available. Verify linked business rules and workflows are included.
XSLT	Impact analysis not available.

Addition Only Support	Notes
background processes	Must search for 'BGP' or ID; only Scheduled BGPs are included in results.
columns / column groups	Use STEPXML instead.
rows / row groups	Use STEPXML instead.
tables / table groups	Use STEPXML instead.

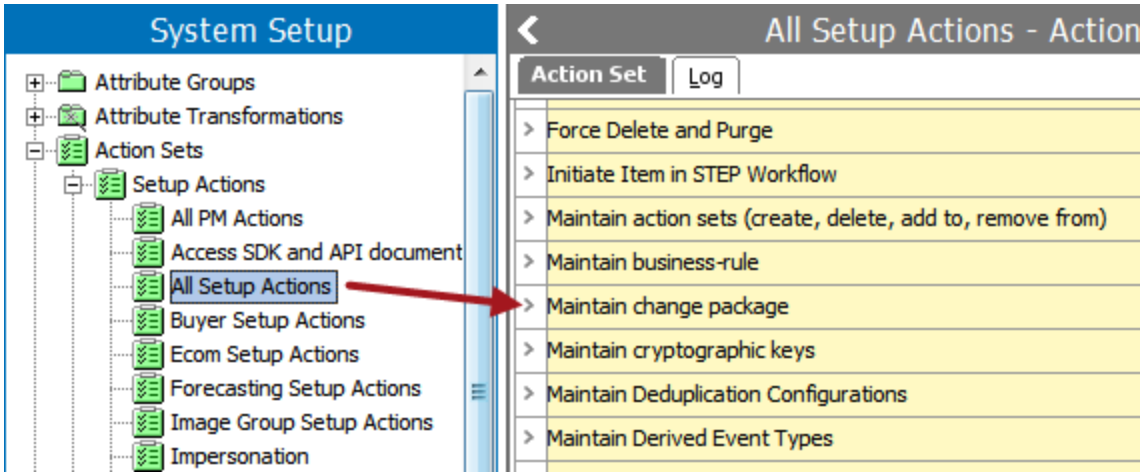
Unsupported	Notes
component model	

Unsupported	Notes
faceted search	
internal object types	System Setup object types: alternate classifications, assets, commercial terms, entity user-type root, etc.
print-related objects	Price lists, templates, Flatplans, colors, rules, etc.
product attribute / classification attribute link types	Under Reference Types.
standard configuration objects - System Setup	Top nodes: contexts, List of Values / LOVs, attribute groups, attribute transformations, Object Types & Structures, tags, units, etc.
standard configuration objects - Tree	Specific system default top nodes: collections, PPH, eCatalogs, index words. Other Tree nodes are supported.
system settings	Users & Groups properties, shared configuration properties.
translation configurations	

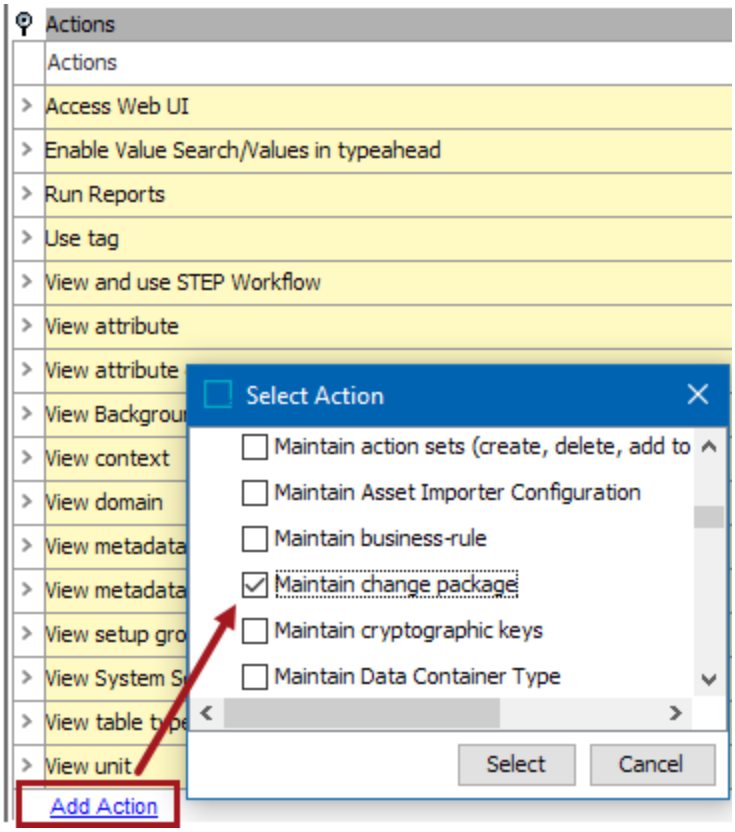
Change Package Privileges

In order to use change packages, the user must be part of a user group that has an unrestricted setup action set applied to it (e.g., has a Setup Privilege defined that includes the 'All Setup Actions' action set). This action set must include *all* setup actions, including the 'Maintain change package' action.

Note: As an administrator, ensure that the All Setup Actions action set has 'Maintain change package' in it, in System Setup go to Action Sets > Setup Actions > **All Setup Actions**.

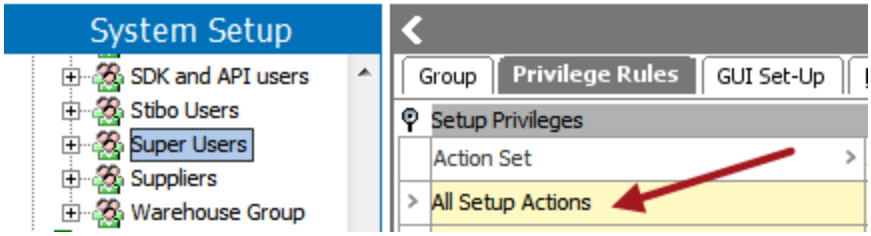


In the list to the right, if the **Maintain change package** action does not exist, scroll to the bottom of the list and click the **Add Action** option. A list of actions that are available to add to this set is displayed.



Note: In addition to the **Maintain change package** action, the **All Setup Actions** action set must contain *all* setup actions. If any actions are present in the **Add Action** pop up, these must also be added to the **All Setup Actions** action set to enable full use of change package functionality.

When the **All Setup Actions** action set includes *all* setup actions, add users requiring access to change package functionality as part of a user group that has the **All Setup Actions** privilege applied.



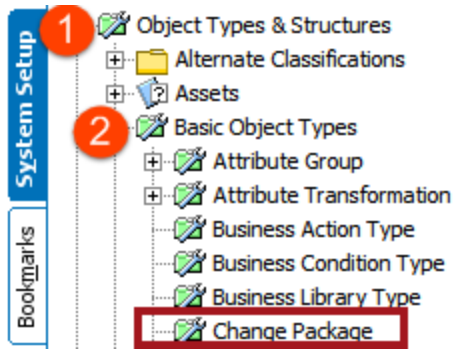
For more information, refer to the Users and Groups topic or the Action Sets topic, both in the System Setup documentation.

For information on including users in a change package, refer to the Editing a Change Package topic.

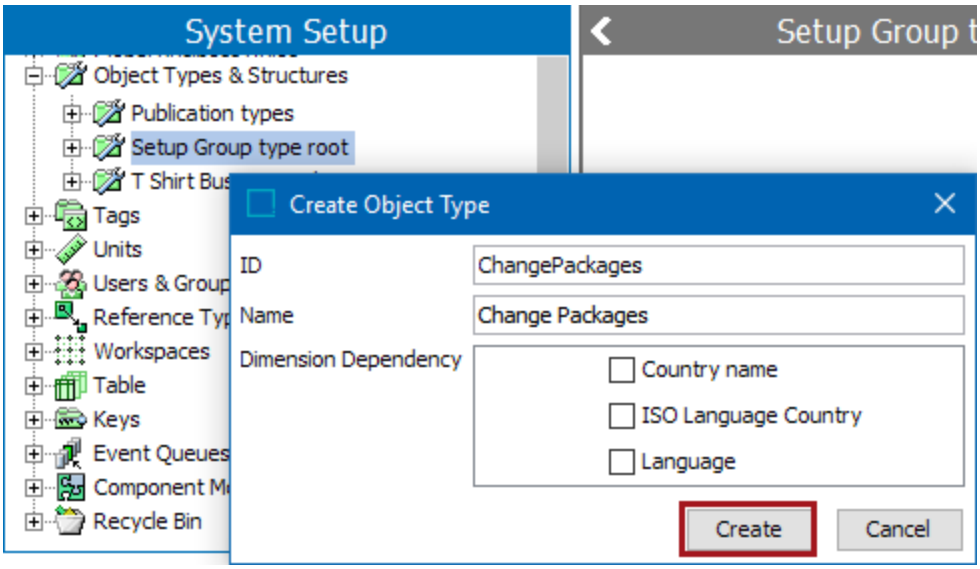
Initial Setup for Change Packages

To create a Change Package that will process and migrate STEP configuration changes, first verify the basic change package configuration exists as defined below.

1. On System Setup, select Object Types and Structures, navigate to Basic Object Types, and verify **Change Package** exists.

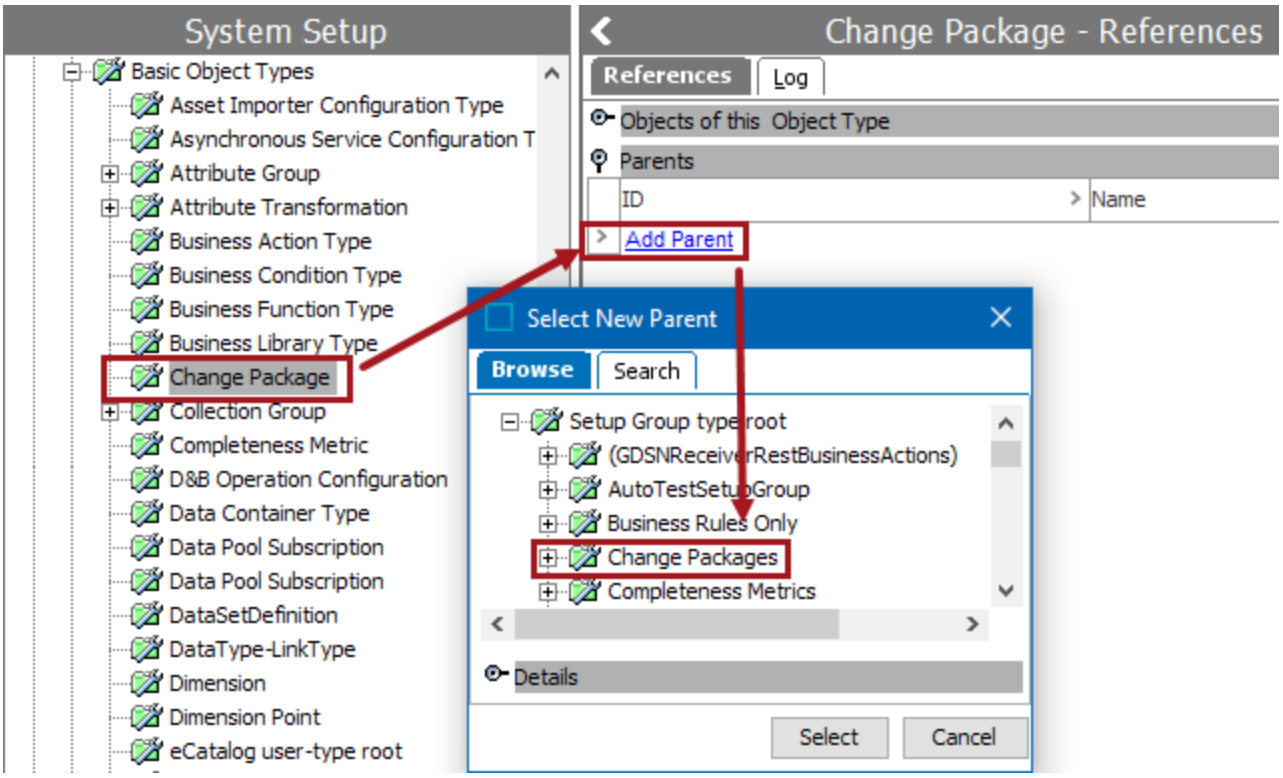


2. Under **Setup Group type root**, right-click to create a **New Object Type**.

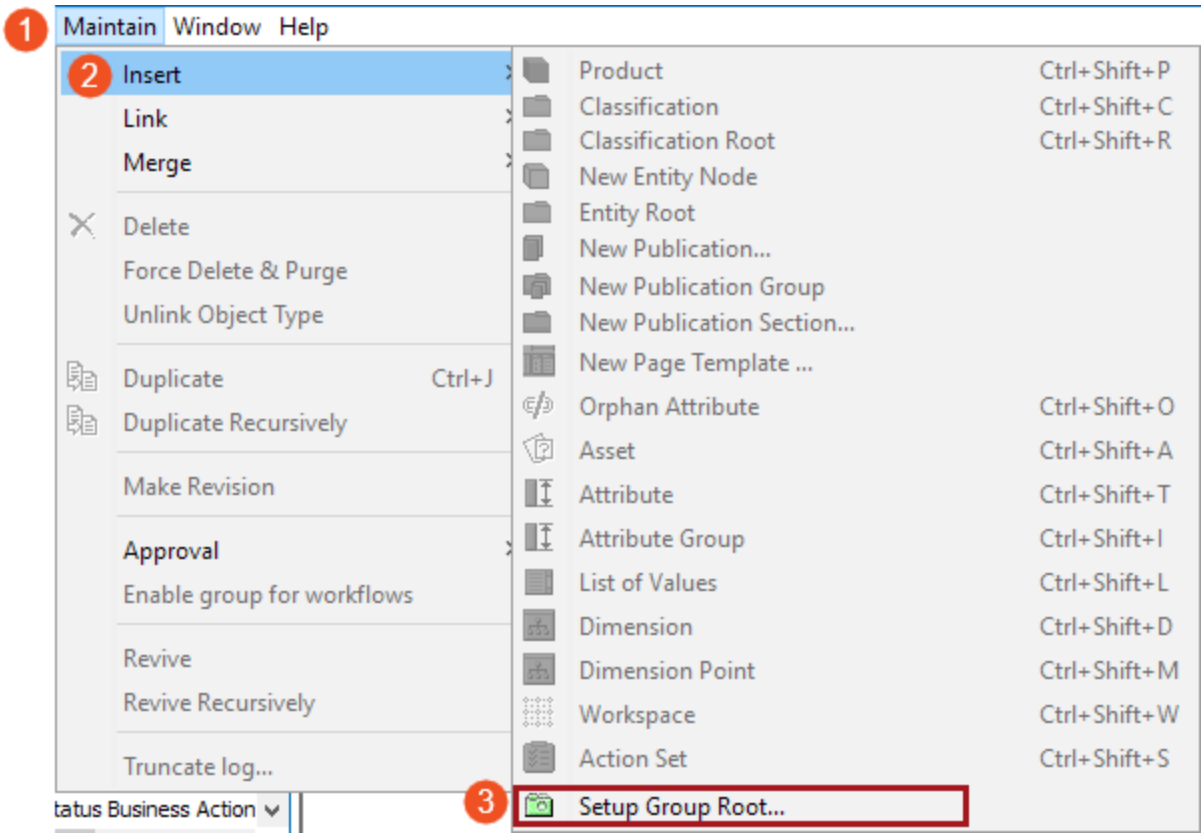


3. Under Basic Object Types node, return to the Change Package object.

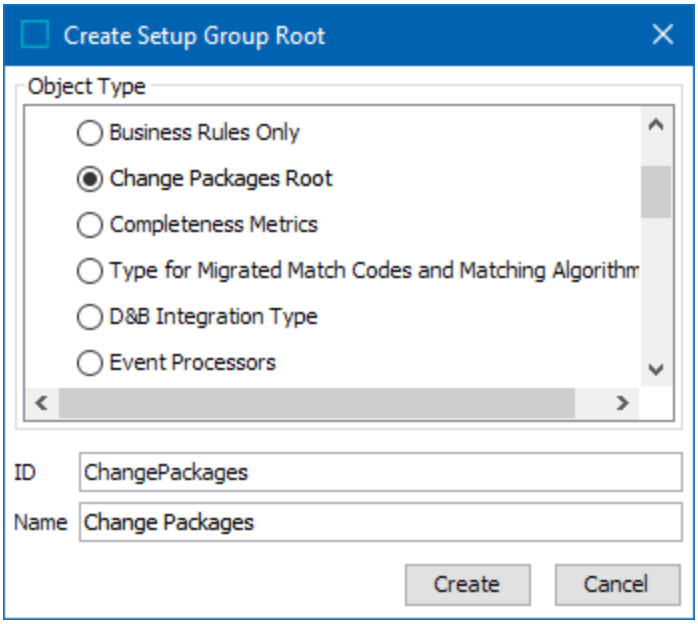
4. On the References tab, click **Add Parent** link. Select the folder you created for change packages under **Setup Group type root** and click the **Select** button.



5. On the Maintain menu, navigate to Insert, and select **Setup Group Root** to open the 'Create Setup Group Root' dialog.



6. Select the Change Packages object type that you created above, enter an ID and Name, and click the **Create** button. This creates a folder in System Setup where you can then create individual change packages.

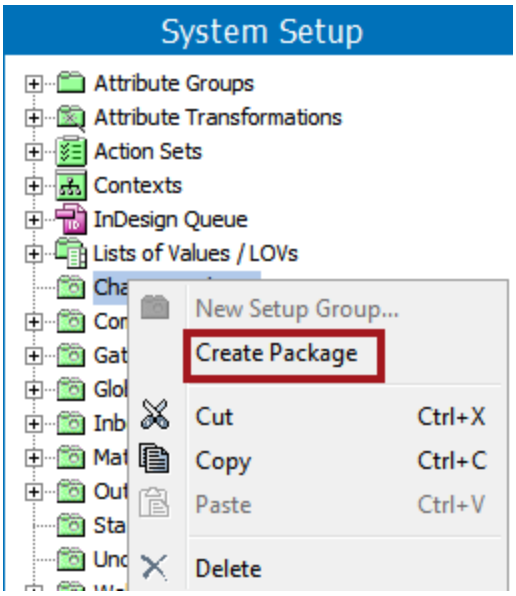


Refer to the Creating a Change Package topic for details on the next step.

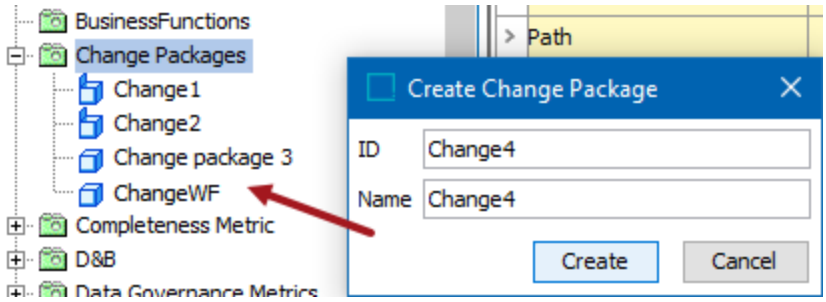
Creating a Change Package

To create a Change Package, you must complete the Initial Setup for Change Packages topic of this documentation.

1. On the System Setup tab, locate the folder established for holding change packages.
2. Right-click the folder and select **Create Package**.



3. In the Create Change Package dialog, assign an ID and a name to the package and click the **Create** button.



For details on adding items and working with change packages, refer to the Editing a Change Package topic

Editing a Change Package

A change package is a container to store a set of system configurations for migration to another system. When a change package has been created, it is empty until items have been added to it. When objects have been added, the system then tracks if subsequent changes occur on those items. Information on the change package informs the user if an item in the change package is up to date compared to the current system configuration using color coding. Users can then resolve discrepancies if desired. Details for working with open change packages are described below.

After editing a change package but before exporting it, you must finalize the package as defined in the Finalizing a Change Package topic.

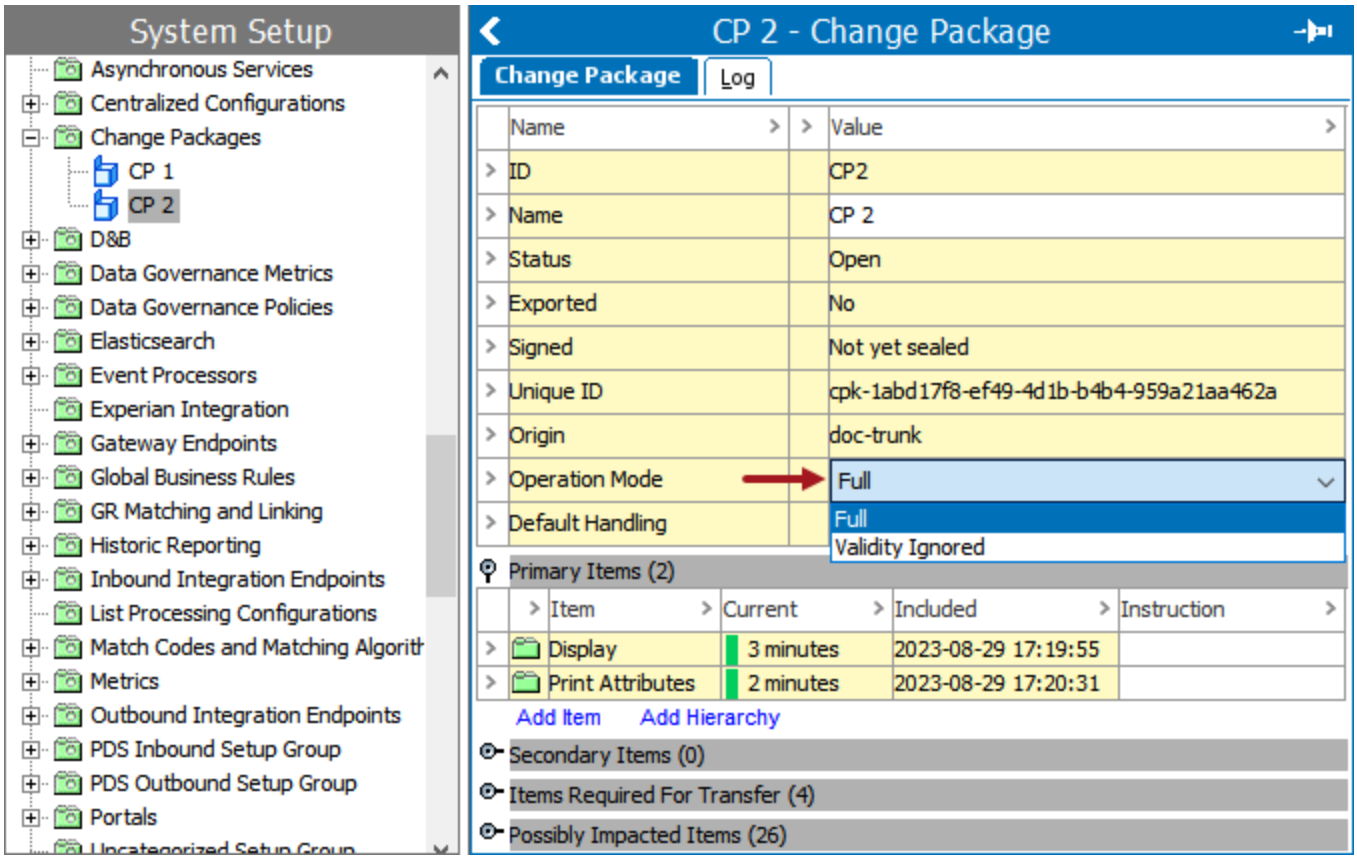
Users in Change Packages

When users and/or user groups are added to a change package, passwords for the users are not included. Since creating a new user requires a password, new users are created with a temporary password which must be reset manually when passwords are maintained in STEP. If the user is a service account that will never be used to log into STEP, or if users are managed by an external system, manually resetting the temporary passwords is not necessary. Changes to existing users can be made using change packages; but passwords can only be changed by the user or an administrator.

Note: When changing a user from one assigned group to another, the user is added to the new group, but must be manually removed from the original group.

Set the Operation Mode

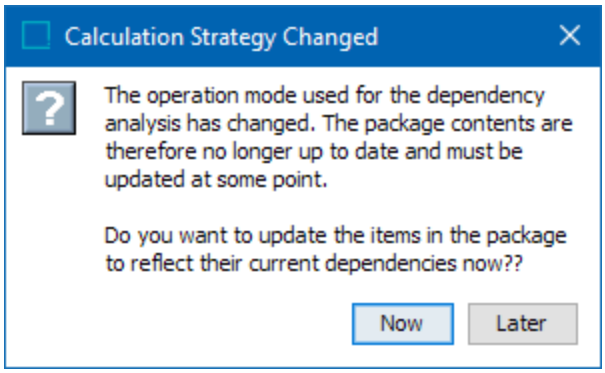
The Operation Mode determines how the dependency analysis functions for the change package.



Set the Operation Mode as follows:

- **Full**: This default setting means all objects that are manually added to the change package will, in turn, have all of their associated items included in the change package. This automated inclusion pulls in not only items that the object touches (references, workflows, etc.), but also objects touched by those items. For example, if an attribute is valid on two object types, each of those two object types is also added to the package.
- **Validity Ignored**: This setting means the change package ignores associations made as a result of valid attributes, object types, and reference types when the dependency analysis is made. For example, when the user adds an attribute in this mode, the object types and references on which the attribute is valid are *not* automatically added as they are in 'Full' mode.

Changing the operation mode displays a prompt to run the dependency analysis. The analysis can be run at the time of the prompt or at a later stage in the change package.



Add Items to a Change Package

Objects can be added to the package using drag-and-drop for most objects, or one of the links below the Primary Items flipper. Support for objects in change packages is defined in the Change Package Object Support topic.

For example, with drag-and-drop, add only the specific items selected (no children) which is the same functionality as using the **Add Item** link. With the **Add Hierarchy** link, add the selected item that supports hierarchies as a primary object and all of the children as secondary objects.

Note: The Add Hierarchy functionality is not intended to move product or classification hierarchies; therefore, a limited number of objects support adding as a hierarchy, such as attribute groups, collection groups, collections, etc.

1. Choose a method to add items:

   - **Drag-and-drop** - lock the editor by clicking the pushpin button ( ) on the right side of the dialog, and then multi-select items or select an individual item from the System Setup or Tree tab and drag-and-drop them in the table header area between the Primary Items flipper and the Add Item link. When objects have been added, click the pushpin button once more to enable the editor.

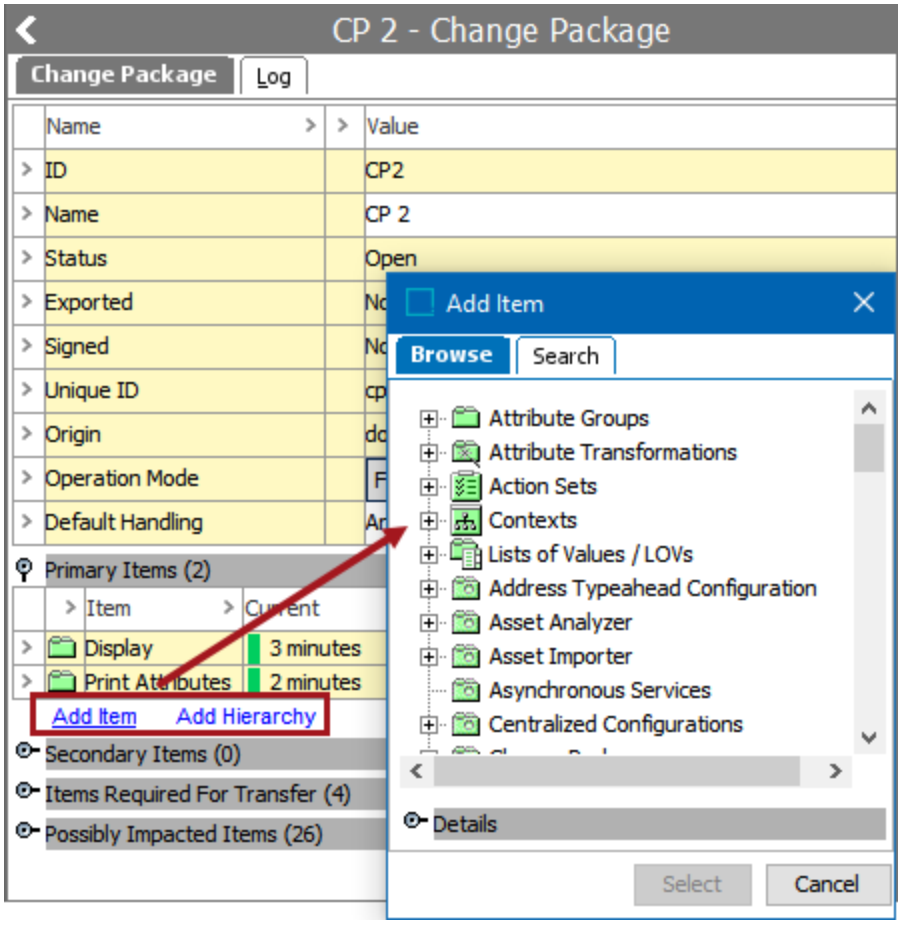
     Drag-and-drop is not supported for a group of items with their children populating the Secondary Items flipper; instead, use the Add Hierarchy link.

     Note: Attempting to drag-and-drop an invalid object is not allowed. On the dialogs, selecting an invalid object type reports the issue in red text at the bottom of the dialog and the Select button is not enabled.

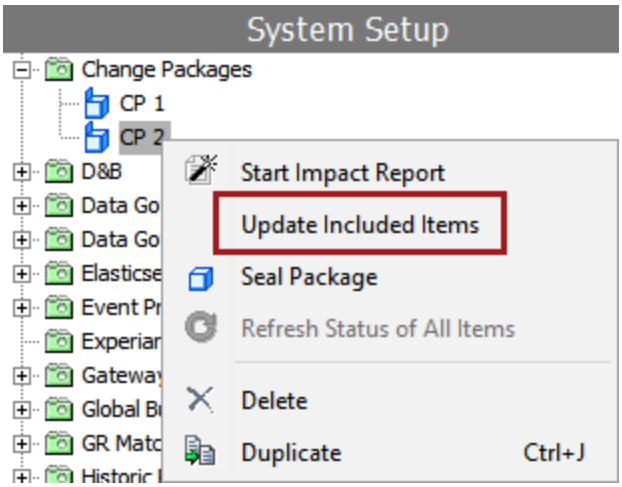
   - **Add Item** - click the link to open the Add Item dialog, select one or more primary items, and click the **Select** button.
   - **Add Hierarchy** - click link to open the Add Hierarchy dialog, select one or more primary items, and click the **Select** button.
   - **Promote to primary** – when default handling is set to Analysis Only, relevant objects added to the Items Required for Transfer and the Possibly Impacted Items flippers can be promoted to the Primary flipper for installation.

- **Default Handling** - Use Default Handling setting of 'Analysis and Installation' when adding items or hierarchies, to include all items added to Items Required for Transfer for installation.

> **Note:** When changing the Handling parameter for multiple items using copy / paste may take time depending on the scope of items pasted. Ideally, use the intended Default Handling before adding items to set the Handling appropriately when initially adding items.



2. The Primary Items flipper is updated with the valid objects added to the change package.
3. Right-click the change package and select **Update Included Items** to ensure an accurate report of the package dependencies.

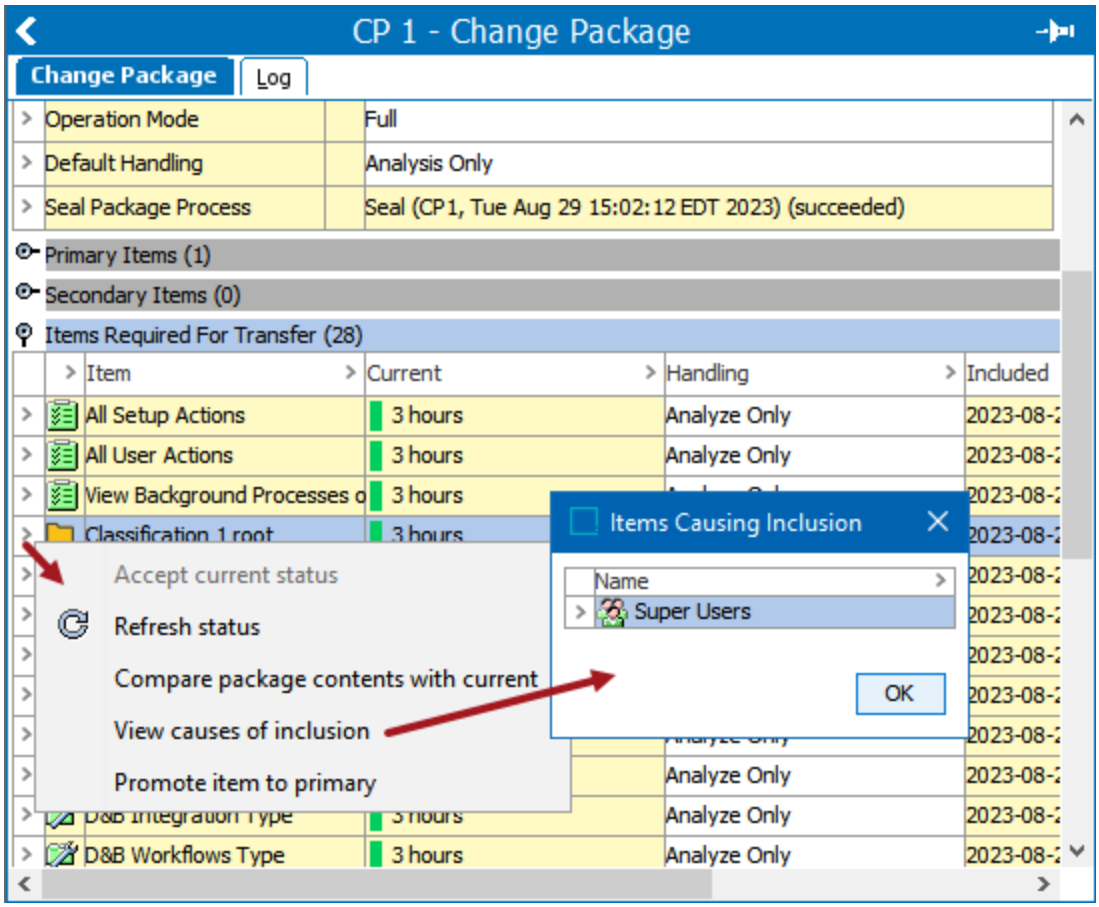


Note: To allow for the addition of primary objects, full dependency calculations are only applied on demand rather than running a potentially complex analysis for each individual addition or removal of an object.

For more information on the items in the Primary Items flipper, refer to the Status and Discrepancies in Change Package Items topic.

Reasons for Included Items

To understand the reason an item is included in a change package, right-click on the arrow next to the item and select the 'View causes of inclusion' option.



Ignore Auto-selected Objects

Users can ignore items listed in the **Items Required For Transfer** and **Possibly Impacted Items** flippers. When there are a lot of items to ignore, use copy / paste with multiselect prior to the paste operation to affect more than one item at a time. The dependency analysis may affect pasting and may not apply to all selected items at once. Additionally, the Default Handling option of 'Analysis Only' practically eliminates the need to manually set items to 'Ignore'.

Note: Ignoring an item does not necessarily mean that the number of included items in the change package is diminished.

When changing the Handling parameter for multiple items using copy / paste may take time depending on the scope of items pasted. Ideally, use the intended Default Handling before adding items to set the Handling appropriately when initially adding items.

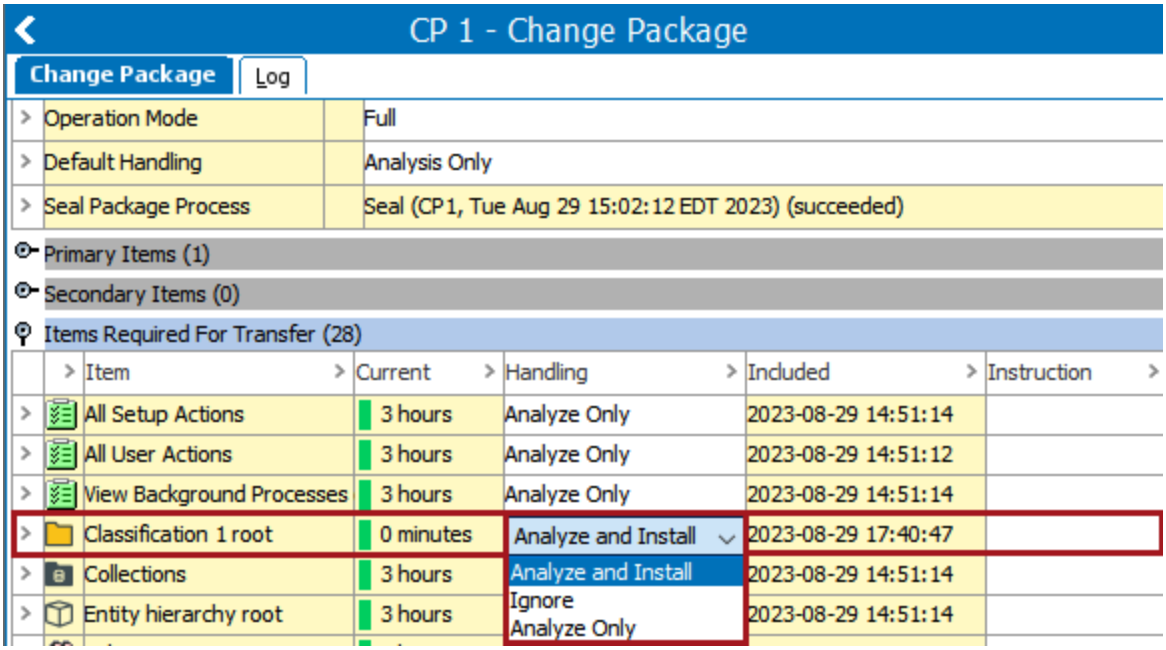
The result of ignoring items that are not needed in either section (via the **Handling** column) is different as defined below:

- In the **Items Required For Transfer** section:

  The **Ignore** setting means that while the items are still part of the package, when the package is transferred to the receiving system, these items will not be installed on the target system. In addition, these items are not evaluated or included in the impact report. This is especially useful if a user knows that a particular item is set up correctly on the receiving system and / or wants to isolate a particular set of objects for transfer without accounting for the full dependency analysis.

  The **Analyze and Install** setting means the object is used in the impact analysis and is installed on the target system.

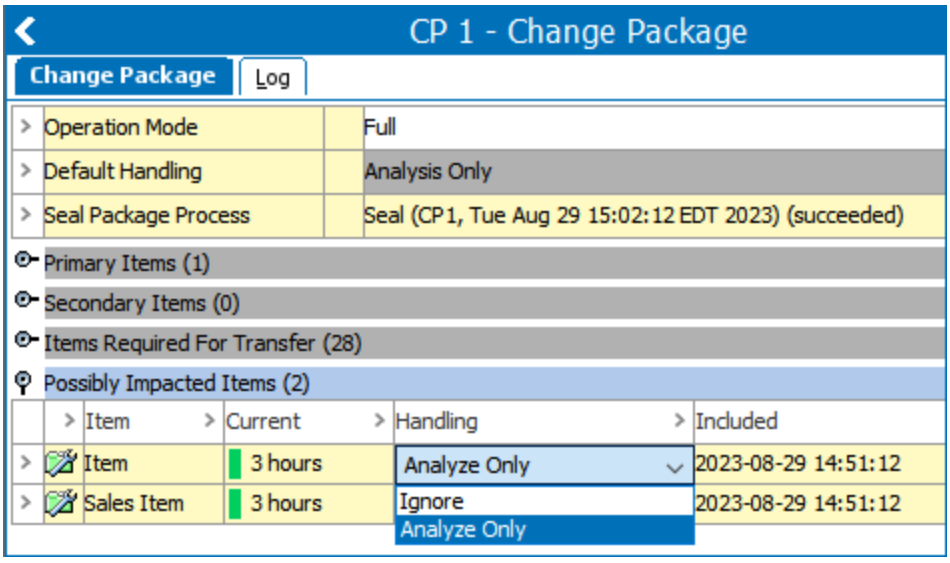
  The **Analyze Only** setting means that the object is used in the impact analysis and is not installed on the target system.



- In the **Possibly Impacted Items** section:

  The **Ignore** and **Analyze Only** settings define if the item is used in the impact analysis or not. In either case, the item is not installed unless it is explicitly promoted to the primary section by the user.

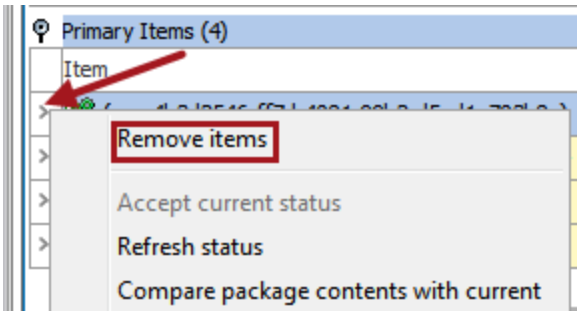
  For more information on impact reports, refer to the Analyzing and Installing Change Packages topic.



Remove Items from a Change Package

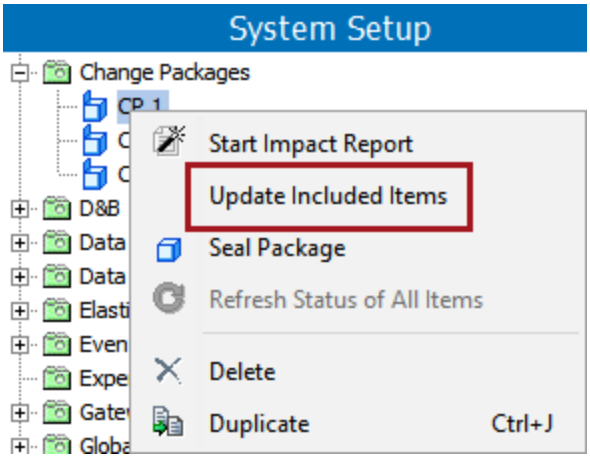
Only 'Primary Items' may be removed from a change package.

1. Verify the change package is open. Items may only be removed from an open package.
2. In the Primary Items flipper, click the row arrow on the item(s) and select **Remove items**.



3. Right-click the change package and select the **Update Included Items** option. This ensures an accurate

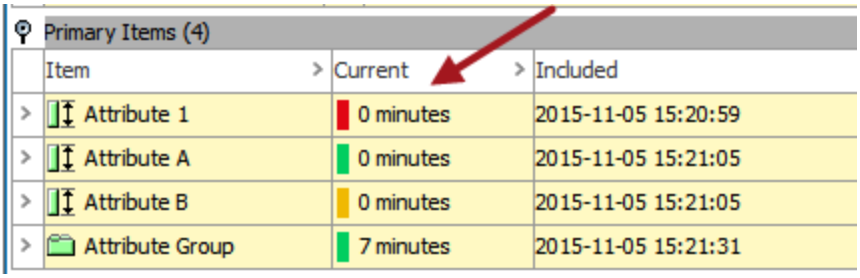
report of the dependencies in the package. Full dependency calculations are only applied on demand.



Status and Discrepancies in Change Package Items

When an item is added to a change package, the system tracks the details of the object from that point forward. If the selected object is changed, the change package notes a discrepancy between the stored version and the current version.

Each item in a change package has a color indicator and a notation of how long it has been since the object in the change package has been compared to the current system state.

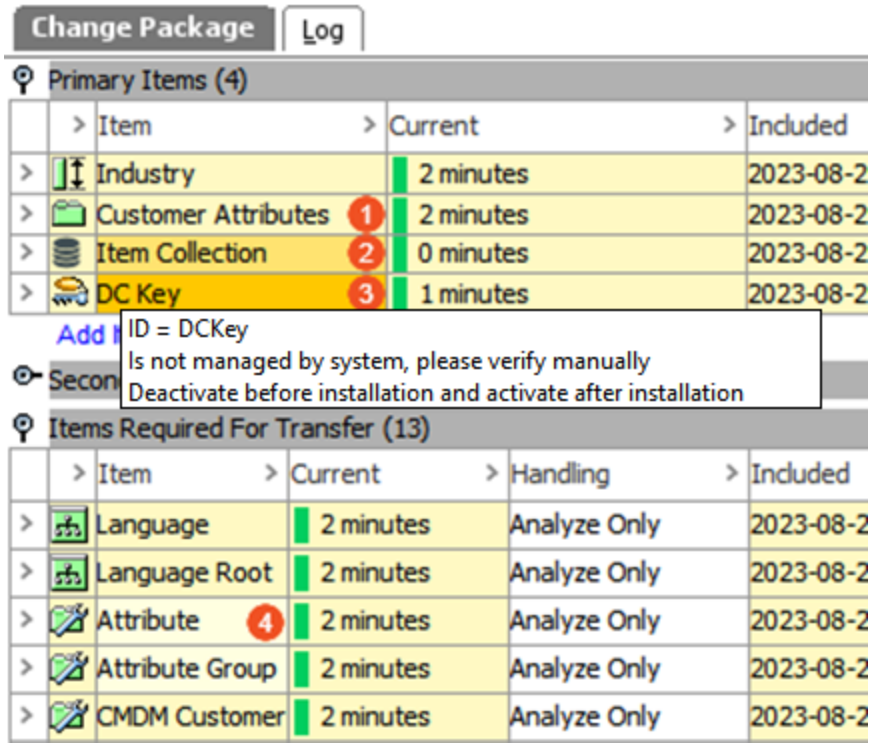


- A **Green** indicator means the object reflected the current status when it was last compared to the system.
- A **Yellow** indicator means that the object has been changed since it was added to the package, but that the change has been accepted.
- A **Red** indicator means that the object has changed since addition to the change pack and changes to this object have not yet been accepted.

The objects that are tracked differently are indicated by the background color of the objects in the Items column and are identified by the numbers in the following image. Hover over an object to view the help text for the object, as shown for the DC Key object in the image.

- Item 1 - the typical workbench yellow background indicates an object that is fully supported for analysis and installation
- Item 2 - the medium yellow background indicates there is not full support for installing the object on the target system. Manual actions are defined in the hover text and are added to the Instruction field.
- Item 3 - the orange background indicates that the objects need to be verified manually. These are objects that cannot have their contents analyzed or referenced objects identified. For example, a Web UI, Asset Importer, or IEP, is not parsed to identify any related business rules or workflows. Also, because they are not included in the Items Required for Transfer, and are not reported on in the impact report, they require manual operations before and/or after installation. Since many of these objects are installed correctly when all associated objects are included manually, they should be verified to ensure the intended functionality and if not, the change package must be revised in the source system.

- Item 4 - the lightest yellow background indicates that the object is part of the system's base configuration, and cannot be moved from one system to another via a change package.

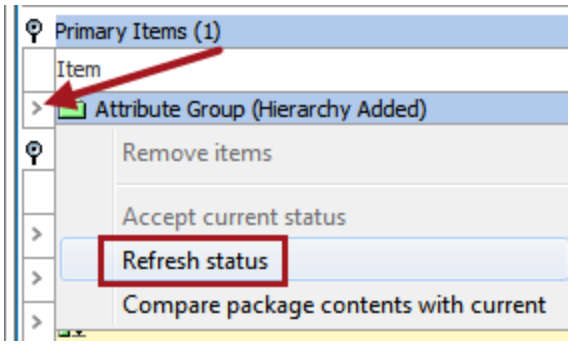


Important: At the time of sealing, the change package pulls the current system version of all objects included in the change package. Therefore, all objects have a green indicator upon sealing of the package. Following sealing, objects can still be refreshed and if a subsequent discrepancy arises, the object has a red indicator. However, the option to accept the change is not available while the package is sealed and an export of the change package includes all objects as they were at the time the package was sealed. Workspace-revisable objects must be current with their approvals prior to sealing the change package. When workspace-revisable objects are installed on a target system, they must be approved for the Current status to show green and also not have any other differences between systems.

Refresh Status

Refreshing an item sets the counter back to zero and updates the color indicator on the object.

To check the status of items in the change package, click on the row arrow in the item(s) and select **Refresh Status** or after a change package has been installed, right-click the change package and use the **Refresh Status of All Items** option to confirm the installation. For example, when installing a product node to support newly added attributes, the product requires approval in order to display a green indicator after the status is refreshed.

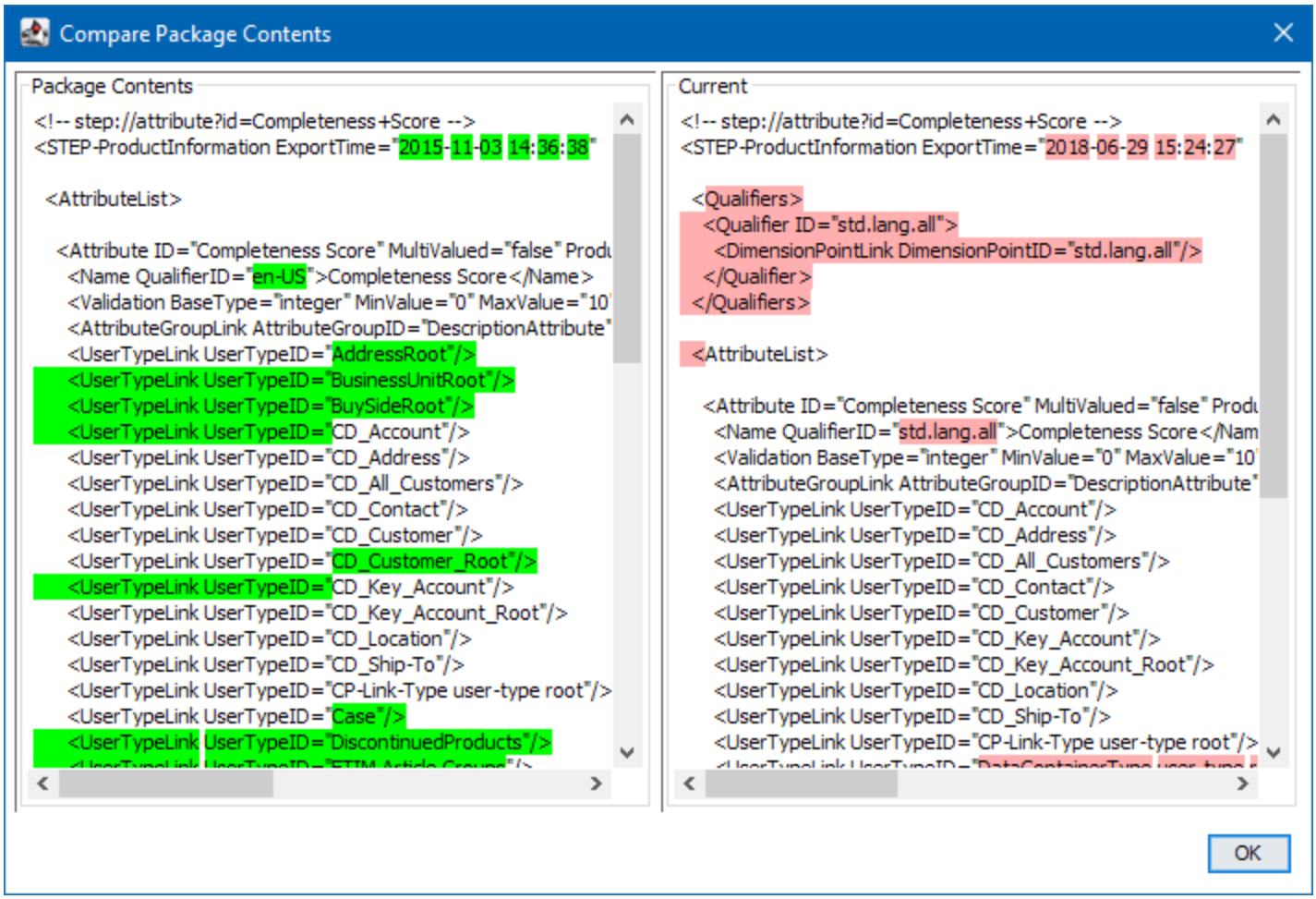


The refresh option is available on all change package objects, regardless of their current status or the status of the change package.

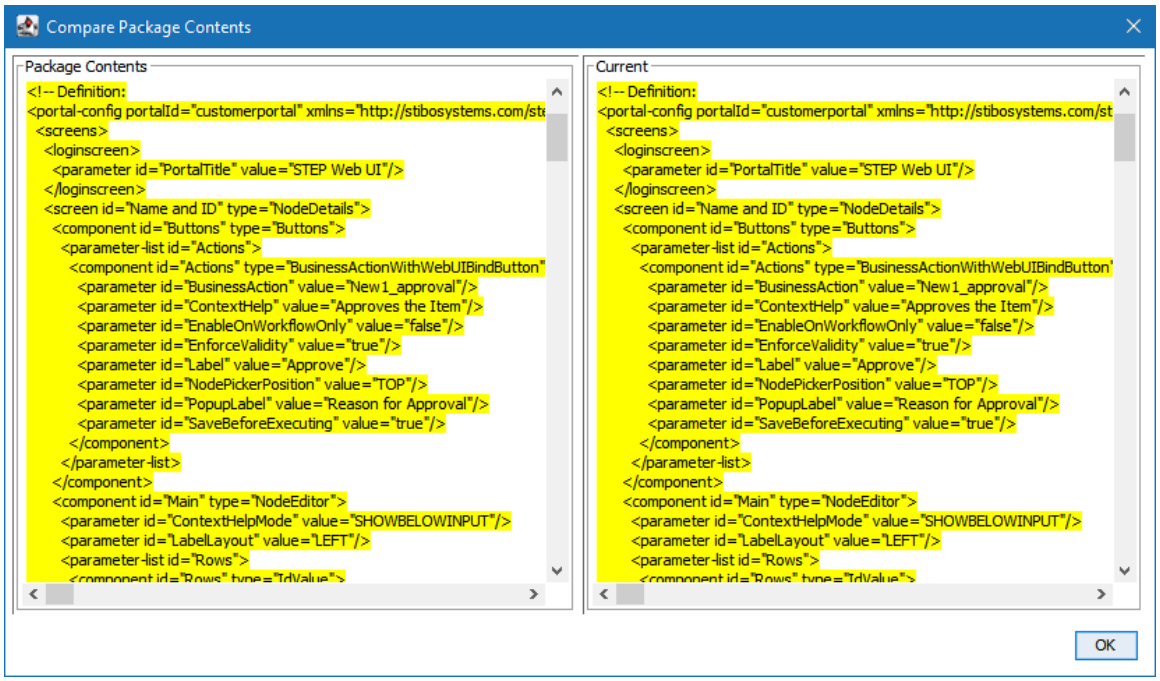
Compare Package Contents with Current

For a detailed comparison of a change package object(s) and the current system, right-click arrow on the object within the package and select the **Compare package contents with current** option.

When a change package is imported to a target system, but has not been installed, the Current window shows no content. Otherwise, differences are displayed.

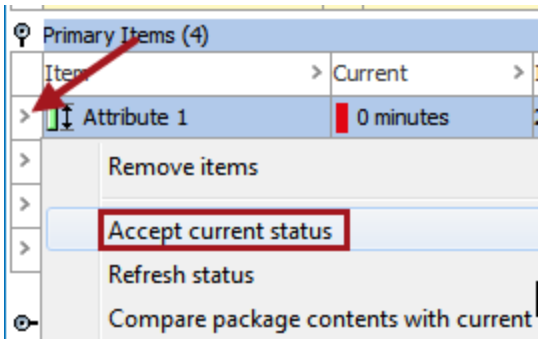


For large STEPXML files, like a Web UI, the comparison may become excessive, causing the comparison to be skipped. If this happens, the text is highlighted yellow (as shown below):



Accept Current Status

If an object has changed since being added to the package, it has a red color indicator and the **Accept Current Status** action is available on the row arrow right-click menu.



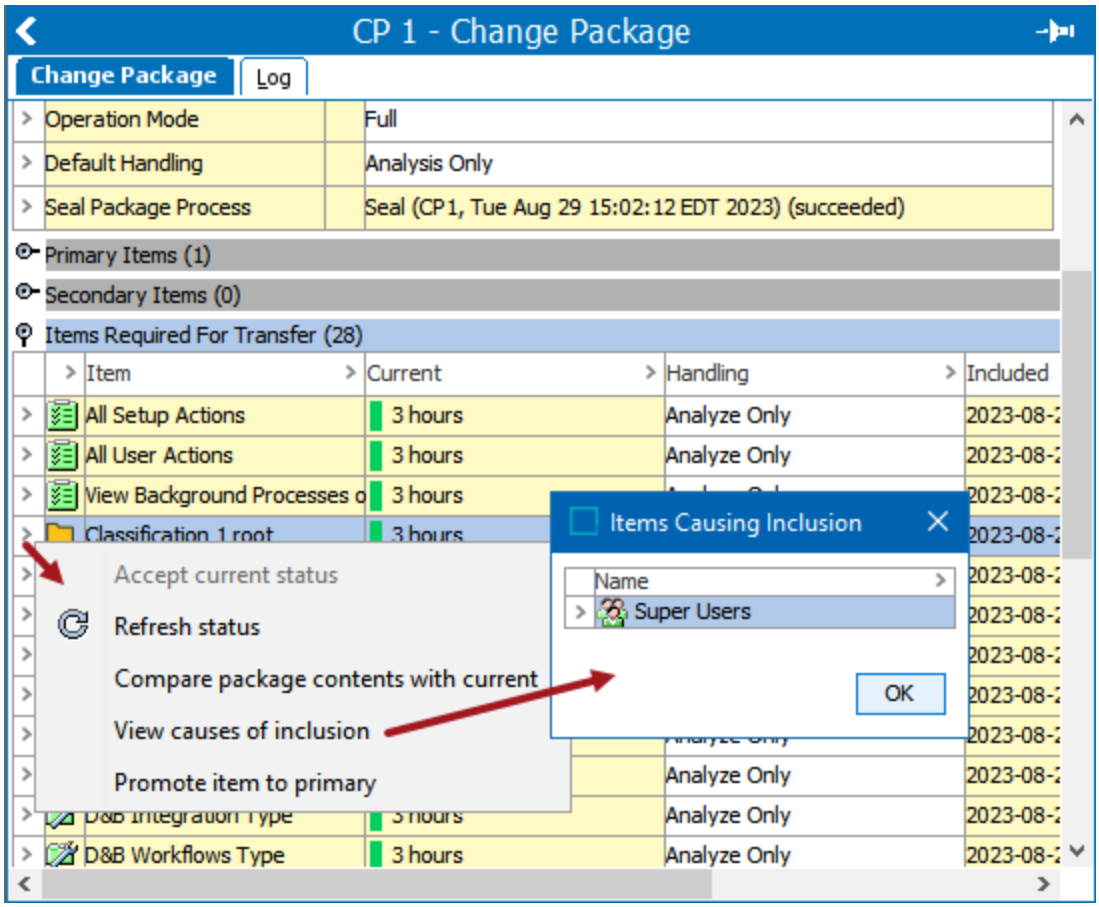
Accepting the current status of an item changes the color indicator to yellow. This means that the object has changed since its addition to the package, but that the change has been verified and the current object is accepted as part of the package.

Note: This option is only available for objects that are *not* up to date and are part of an open change package. If the package has been sealed this option is not available, regardless of object status.

View Causes of Inclusion

The 'View causes for inclusion' option is available for single or multi- selected items for the following flippers:

- Secondary Items
- Items Required for Transfer
- Possibly Impacted Items



Finalizing a Change Package

When the contents of a change package have been confirmed, it is sealed to indicate that no further edits can be made and that the package is ready for export.

Note: Sealing a change package generates an event that can be used to automate import on a target system or external repository. Refer to the OIEP for VCS Integration with Change Packages topic.

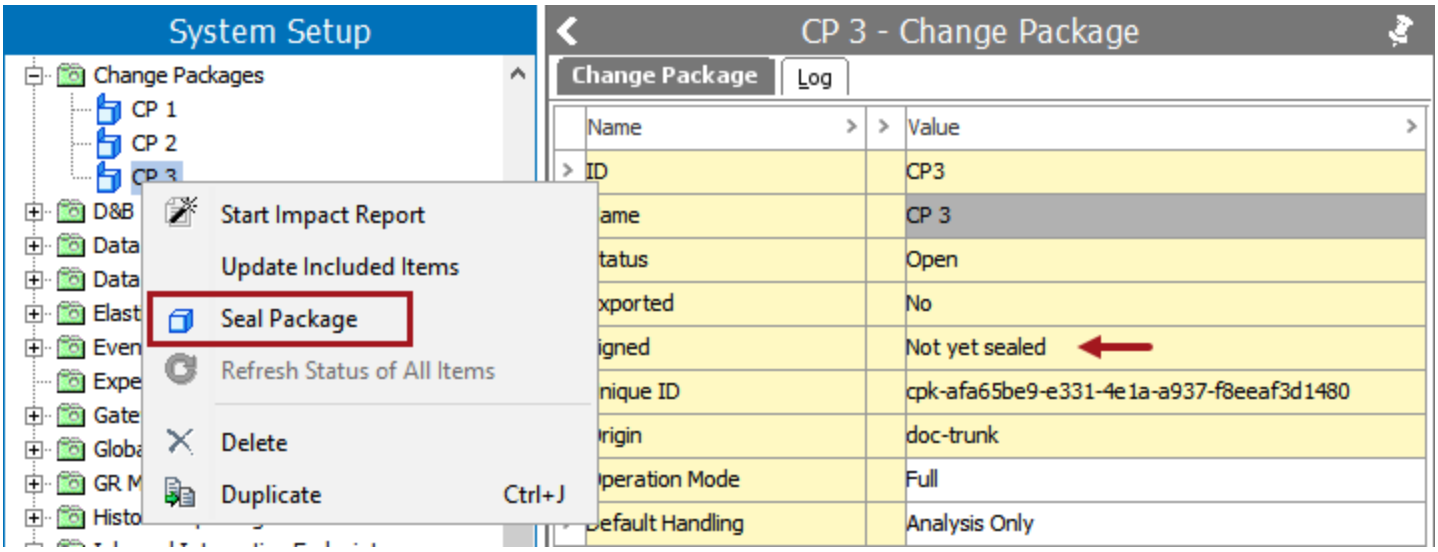
Seal a Change Package

Once a change package has been determined as ready for export, it must be sealed.

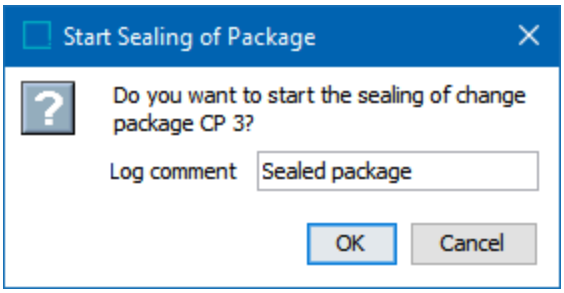
Important: Sealing a change package pulls the current system version of all objects included in the change package. Objects in the Tree that support workspace revisability must be approved prior to sealing.

Prior to a change package being sealed, it has a blue open box icon ( ) and the 'Signed' field is populated with 'Not yet sealed.'

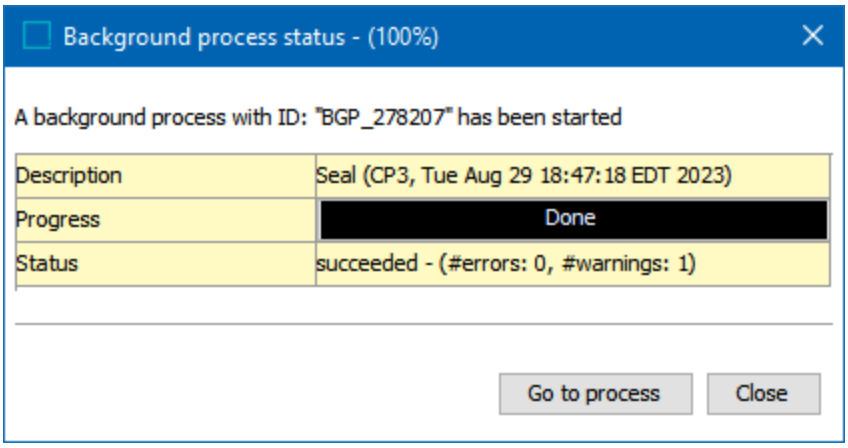
1. Identify an open change package.
2. Right-click the package and select the **Seal Package** option.



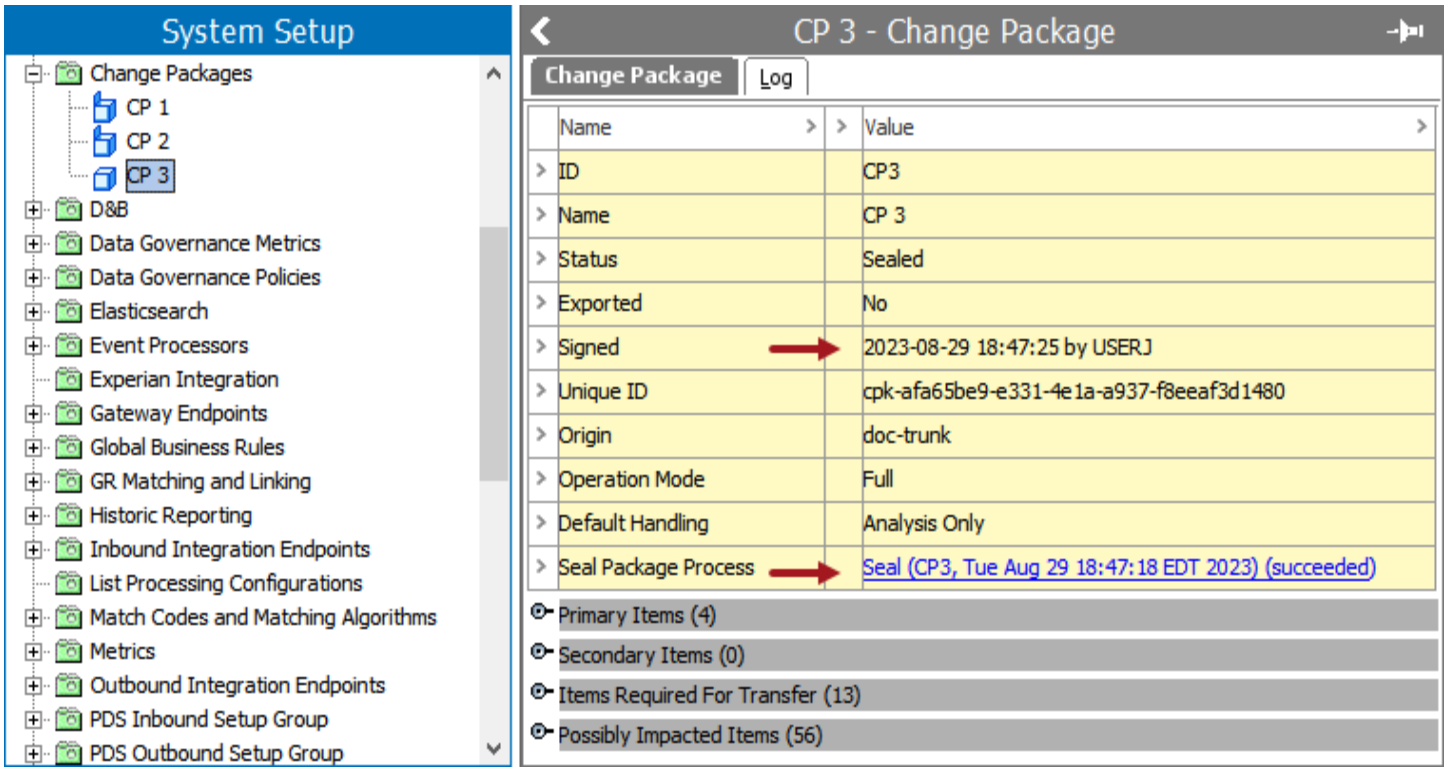
3. Add text to include in the Log, if desired, and click **OK** to start the background process.



4. Optionally, click the **Go to process** button to monitor the progress of the BGP or click **Close** to remain on the change package.



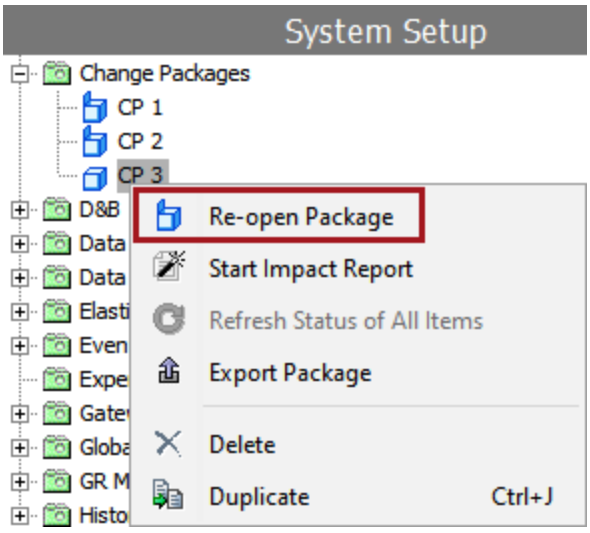
The sealed package displays a closed box icon ( ) and the Signed field indicates the date, time, and user responsible for the sealing. In addition, a link to the sealing background process is provided.



Modify a Sealed Change Package

Re-opening a sealed change package allows the user to edit the change package.

1. Right-click the change package and select the **Re-open Package** option.



2. Follow the steps in the Editing a Change Package topic to modify the change package.

For more information on the items in the Primary Items flipper, refer to the Status and Discrepancies in Change Package Items topic.

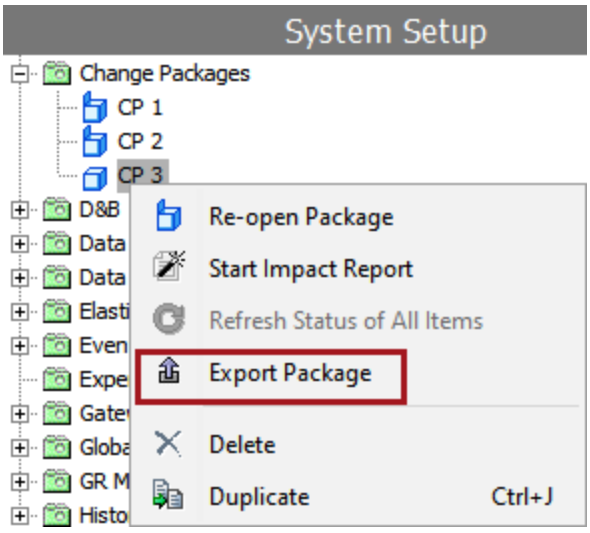
Export a Change Package

Change packages can be manually exported using the standard Export Manager functionality, and subsequently imported to target systems using the Import Manager. Alternately, events on an OIEP using REST Direct can be used to automate the import to a target system with a IIEP using REST Receiver.

For information on automating the export and import, refer to the OIEP for VCS Integration with Change Packages topic.

Manual Export

1. Verify the change package is sealed or dormant, as defined in the Change Packages topic.
2. Right-click the package and select the **Export Package** option.



3. On the Export Manager dialog, select a delivery method and finish the export, as defined in the Export Manager - Select Delivery Method topic of the Data Exchange documentation.

Analyzing and Installing Change Packages

The main purpose of a change package is to transfer configurations between systems or to a version control system. Once a change package has been sealed and exported from a source system, it is expected that it will then be imported to a target system. Upon import, the change package can be analyzed against the target system data set, and subsequently installed if desired.

Automated System to System Integration

To automate the integration, sealing a change package on the source system triggers an event, which can be linked to automated import on a target system using REST services for delivery and receiver methods. This eliminates the need for a user to export, download, name, and import a file, and to migrate it to the next system in a chain of IEPs across various STEP systems that are part of a Development, Testing, Acceptance, and Production (DTAP) environment. Additionally, or instead of direct system to system integration, change packages can automatically be exported to a version control system, like GitHub, GitLab or Bitbucket. For details on configuration for integration endpoints, refer to the OIEP for VCS Integration with Change Packages topic and the Change Package Git Delivery Method in OIEP section of the Integration Endpoint Options for VCS Integration topic.

Importing a Change Package

Change packages are exported as encoded STEPXML files and are imported using the Import Manager or an IIEP with the REST Receiver, both are defined in the Data Exchange documentation.

Note: You must create the setup group for change packages manually on the target system before you can import a change package. For more on creating the setup group, refer to the Initial Setup for Change Packages topic. After initial setup, if child setup groups are desired below the Change Packages top node, move these setup groups to target systems using a change package or recreate them manually with the same ID before transferring a new change package to the target systems.

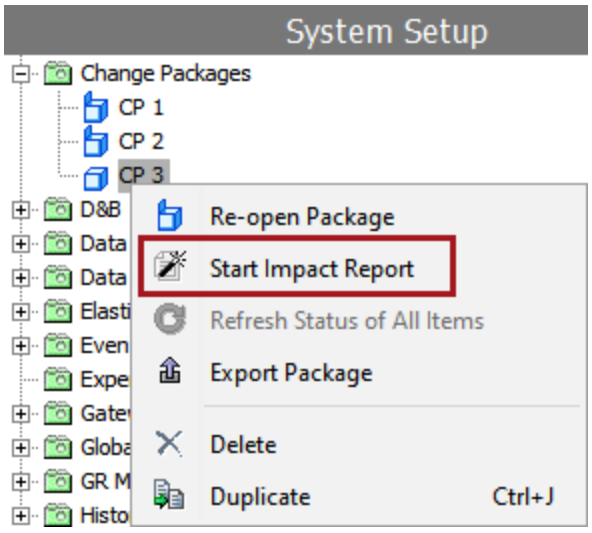
Upon import, the new change package is found in the same location on the System Setup tab as it existed on the source system. Imported change packages have a status of 'Dormant' and a gray icon: 

At this point, the *contents* of the change package have not yet been applied. Only the change package itself has been imported. No system configurations will be updated, and the status remains dormant until the change package is installed.

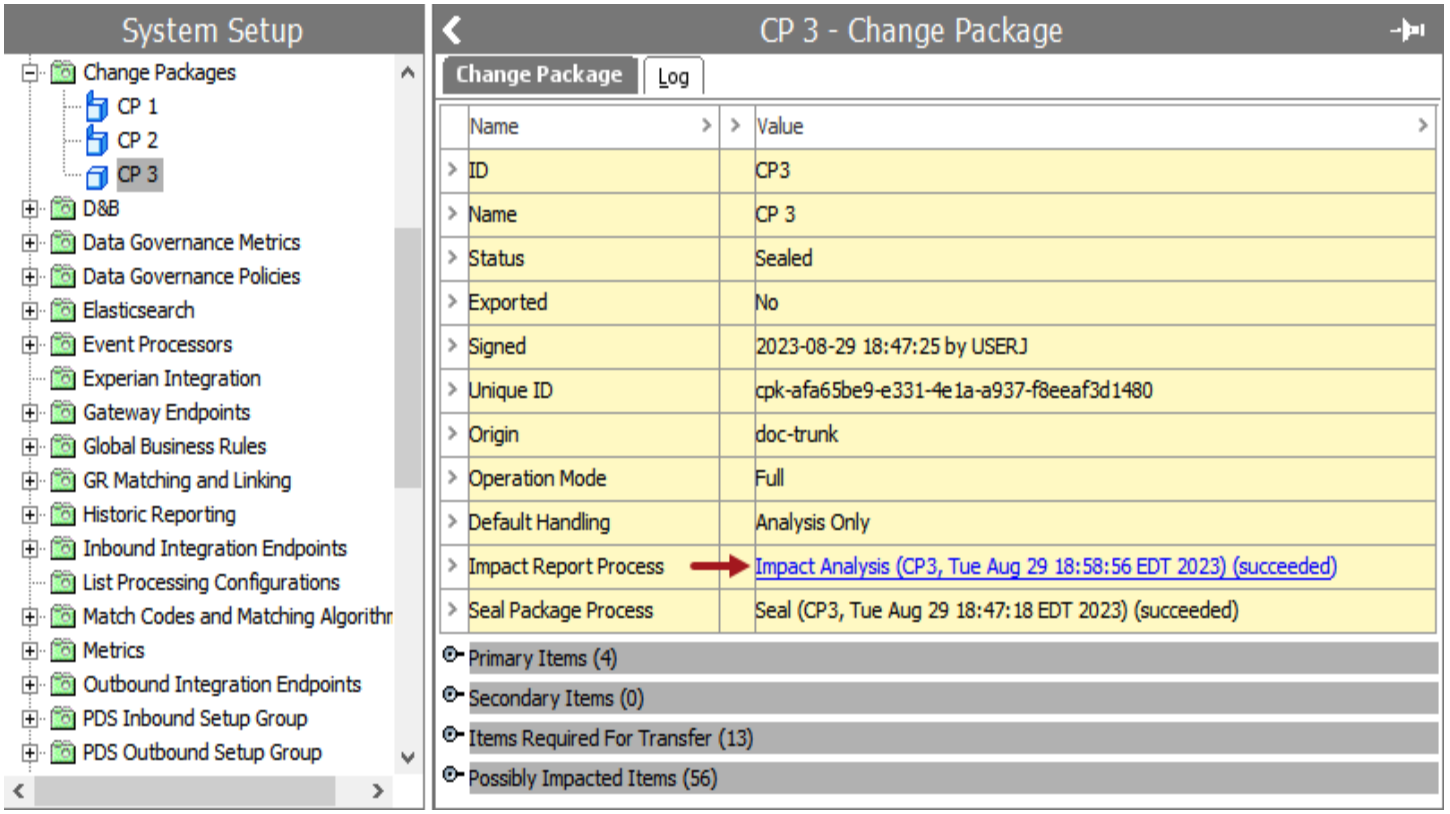
Analyzing a Change Package

Once a change package has been imported, you can run an impact report. The impact report provides the user with information they can use to assess if the change package should be installed, and what the system impacts are likely to be upon installation.

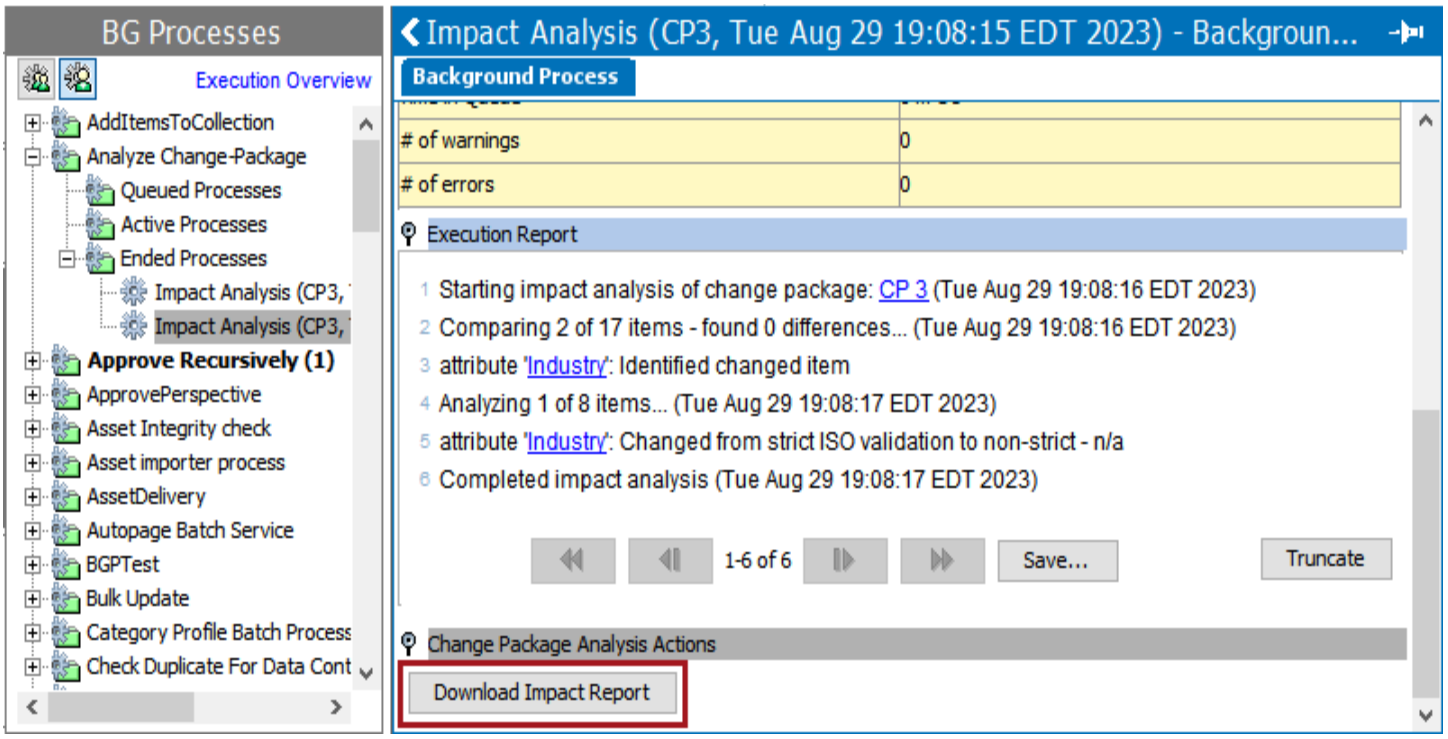
To run an impact report, right-click on the Change Package and select **Start Impact Report**.



The impact report runs as a background process, which is then accessible on the **BG Processes** tab under **Analyze Change Package**. A link to the background process is also provided on the change package object.



The contents of the report can be viewed directly in the execution report.



The impact report can also be downloaded via the **Download Impact Report** button for viewing offline (e.g., in Excel).

	A	B	C	D	E	F	G	H	I
1	Origin	Message Type	Inclusion Type	URL	Object Type	Object ID	Message	Current Status	Status Time
2	Detecti	IdentifiedChar	Change	step://attril	attribute	Industry	Identified ch	Out of sync	8/29/2023 19:08
3	Impact	PropertyMism	Change	step://attril	attribute	Industry	Changed fro	Out of sync	8/29/2023 19:08

Analyze the impact report to determine if the change package should be installed and if any changes should be made on the target system prior to installation. If system changes occur, it may be useful to re-run the impact report.

Note: Not all objects are supported in the impact report. Refer to the Change Package Object Support topic for details.

Using the default 'Analysis Only' Handling option prevents objects in the Items Required for Transfer list and the Possibly Impacted Items list used in the impact report from being installed. When the impact report identifies a missing object that is relevant to the feature being migrated, it is recommended to re-open the change package on the source system and either promote the item or add a necessary item that has not been identified, like a referenced business rule or workflow on an IEP or in a Web UI. If system changes occur, it may be useful to run the impact report again after it is re-imported.

If the change package should not be installed, right-click on the change package and select **Delete** to completely remove it from the system, without the option for restoration.

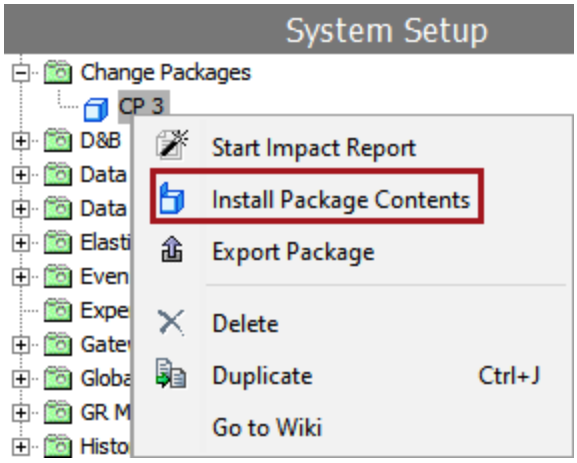
Note: A deleted change package can only be accessed by importing the package again.

Installing a Change Package Manually

Installation of the change package means that all objects within the change package in the Primary and Secondary flippers are added to the system, while the installation of items in the Items Required for Transfer list depend on the Handling option used for the item. Possibly Impacted Items are only analyzed, not installed, unless promoted to the Primary flipper, which removes the item from the Possibly Impacted Items list. If objects in the change package existed previously on the system, they are updated to reflect the contents of the package.

Refer to the Change Package Object Support topic for considerations when installing a change package.

When the impact report has been reviewed and the change package is determined acceptable, import it on the target system. Select the imported change package, right-click, and select **Install Package Contents**. Review the execution report for potential warnings or errors.



STEPXML Comparison Tool

Important: The STEPXML Comparison Tool has been superseded by the change packages functionality available in Stibo Systems Enterprise Platform (STEP) Workbench and may be removed in a future release. It is recommended that users transition to using change packages, which are described in the Change Packages section of the Configuration Management documentation.

STEP has a tool for comparing system setup on different instances of STEP. The comparison tool requires an XML file to be exported from the source system and the target system.

This can be used to identify:

- Configuration that is different
- Compare collections, bulk update configurations and export / import configurations
- Configuration that only exists on the source system
- Configuration that only exists on the target system to identify what needs to be deleted
- Configuration that is identical

Once the differences have been identified the system compare tool can then be used to do the following:

- Generate an XML file of only the differences to add to another STEP system.
  This data can then be imported to the target system
- Generate an XML of all the differences and import onto the target system

The comparison tool should not be used to migrate assets, products, classifications and entities from one system to another. It should only be used to compare two STEP systems and from this comparison generate STEPXML to move this configuration from a source system to a target system

The tool only adds / modifies configuration on target systems it does not delete items that should not exist on the target system. It does not make updates that require user input; as explained later in the document.

Refer to the following topics for more information:

- STEPXML Comparison Tool Prerequisites
- STEPXML Comparison Tool Limitations
- STEPXML Comparison Tool Scenarios
- Using the STEPXML Comparison Tool

STEPXML Comparison Tool Prerequisites

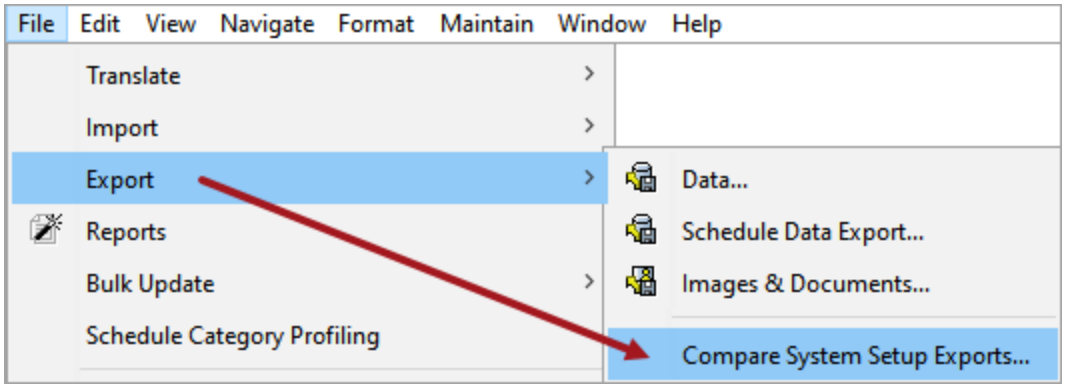
Typically, this type of work requires a Super User to carry out the task. The user would require in-depth knowledge of STEPXML, System Setup, and how to use export and import manager.

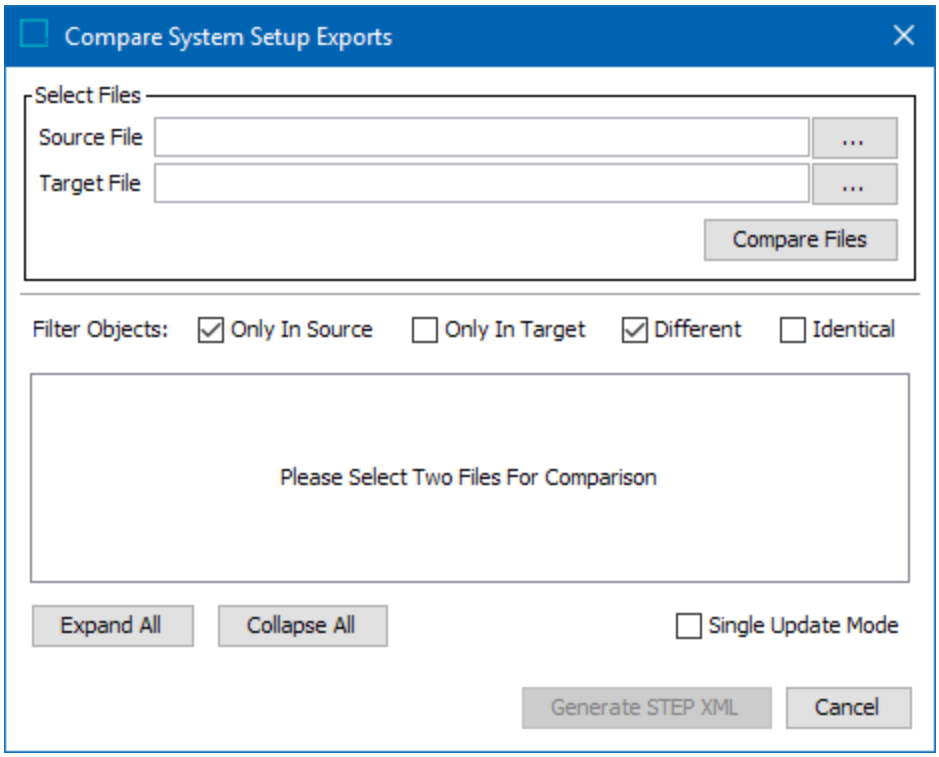
Accessing the System Compare Tool

To access the STEP System Administration data, the user must have Super Users privileges, i.e., be an administrator of the system. The user will require all the privileges necessary to make the updates the imported XML requires.

Important: A lot of memory is required to run this tool, therefore the STEPXML Comparison Tool has been superseded by the change packages functionality and may be removed in a future release. It is recommended that users transition to using change packages, which are described in the Change Packages section of the Configuration Management documentation.

Go to File > Export > Compare System Setup Exports... to display the **Compare System Setup Exports** dialog.





Moving configuration without using the STEP comparison tool

You can move configuration from one STEP instance to another without using the comparison tool. This process only adds / modifies configuration loaded onto a target system. To then identify what is different from the source and target machines, use the comparison tool.

Move all configurations from one STEP system to another

- Back-up target system
- Export XML from Source system excluding assets, classifications, products, and entities
- Run a Cross Context export if configuration is stored in more dimension points, i.e., LOVs, attribute names, etc.
- Import onto target system
- Check execution report for errors and resolve

Configuration that can be moved from one STEP system to another

Before a user can use the comparison tool the STEP export manager needs to be used to export STEPXML from the source and target systems.

When configuring the export, the following will need to be selected:

- Add the parent node for configuration files being which need to be moved from the source to the target
  If this is not done the import will fail on import the configuration files as the folder it resides in will not exist
- Select the STEP configurations that need to be exported

All parameters available in the Export Manager are defined in the STEPXML Outbound Parameters topic in the Data Exchange documentation. The Global Settings, Data Objects, and Publishing options are not valid for the STEPXML Comparison Tool.

Only the Configuration parameters in Export Manager for the STEPXML format are available for comparison with this tool as shown in the following section.

Configuration

Configuration Parameter	Description
Include Action Sets	Select **No** or **Yes** to control output of a list of all actions applied to each action set. For more information, refer to the Action Sets topic in the System Setup documentation.
Include Asset Push Event Queues	Asset push event queues are exported as follows: • Select **All** to output all asset push event queues. • Select **Selected** to output all asset push event queues based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no asset push event queues. For more information, refer to the Creating and Maintaining Asset Push Event Queues topic in the Digital Assets documentation.
Include Asset Push Configurations	Asset push configurations are exported as follows: • Select **All** to output all asset push configurations. • Select **Selected** to output all asset push configurations based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no asset push configurations. For more information, refer to the Asset Push topic in the Digital Assets documentation.
Include Attributes	For attributes, the validation base types, dimension dependencies applied units, applied LOV hierarchy filters, calculated templates, and so on, are exported as follows: • Select **All** to output all attributes, including fundamental system specific attributes

Configuration Parameter	Description
	(having an ID that starts with **stibo.** or **asset.**). • Select **Selected** or **Minimum** to output attributes based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no attributes. For more information, refer to the Attributes topic in the System Setup documentation.
Include Attribute Groups	For attribute groups, the view definitions and settings applied to attribute groups are exported as follows: • Select **All** to output all attribute groups. • Select **Selected** or **Minimum** to output attribute groups based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in Data Exchange documentation. • Select **None** to output no attribute groups. For more information, refer to the Attribute Groups topic in the System Setup documentation.
Include Attribute Transformations	Select **No** or **Yes** to control output of attribute transformations and their configurations. For more information, refer to the Attribute Transformations topic in the System Setup documentation.
Include Bulk Update Configurations	Assets saved as bulk update configurations are exported as follows: • Select **Yes** to output all bulk update configurations. • Select **Minimum** to output bulk update configurations based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **No** to output no bulk update configurations. For more information, refer to the Bulk Updates topic.
Include Business Rules (Global) and Libraries	For global business rules and libraries, the ID, name, links, on approve setting, dependencies, valid object types, and templates are exported as follows: • Select **All** to output all global business rules and libraries. • Select **Selected** or **Referenced** to output attributes based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation.

Configuration Parameter	Description
	• Select **None** to output no global business rules or libraries. Templates are exported in Base64 and can only be imported without modifications into another STEP system. For more information, refer to the Business Rules topic.
Include Collection Definitions	Select **No** or **Yes** to control output of collections groups and collections, including search URLs. The exported file will not contain products, classifications, or assets included in a collection. For more information, refer to the Collections topic in the Getting Started documentation.
Include Component Models	Select **No** or **Yes** to control output of details of component models, including ID, name, object types and links. The Component Model node in System Setup displays all available component models on your system. For more information, refer to the Component Models topic in the System Setup documentation.
Include Contexts	Select **No** or **Yes** to control output of dimensions, dimension points, contexts, applied dimension points, and locale settings. For more information, refer to the Contexts topic in the System Setup documentation.
Include Context Qualifiers	Select **No** or **Yes** to control output of a list of qualifiers used in exported data. When used in combination with export of product, classification, or asset data, controls qualifier IDs and dimension point IDs. For more information, refer to the Contexts topic in the System Setup documentation.
Include Data Container Definitions	Data container types are exported as follows: • Select **All** to output all data container types. • Select **Selected** or **Minimum** to output data container types based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no data container types. For more information, refer to the Data Containers topic in the System Setup documentation.

Configuration Parameter	Description
Include Derived Event Types	Select **No** or **Yes** to control output of objects created in System Setup as Derived Event Type. For more information, refer to the Derived Events topic in the System Setup documentation.
Include eCatalogs	Select **No** or **Yes** to control output of eCatalog configurations and price lists. Does not include product selection lists. For more information, refer to the eCatalogs topic.
Include Event Processors	Control output of event processor information, including ID, name, links, and configuration. • Select **All** to output all event processors. • Select **Minimum** to output event processors based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no event processors. The configurations are exported in Base64 and can only be imported without modifications into another STEP system. For more information, refer to the Event Processors topic in the System Setup documentation.
Include Event Queues	Select **No** or **Yes** to control output of event queues and applied message templates, output formatting, and triggering definitions. For more information, refer to the Event Queues topic in the System Setup documentation.
Include Export Configurations	Assets saved as export configurations are exported as follows: • Select **All** to output all export configurations. • Select **Minimum** to output export configurations based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no export configurations. For more information, refer to the Maintaining a Saved Export Configuration topic in the Data Exchange documentation.

Configuration Parameter	Description
Include Image Conversion Configurations	Image conversion configurations are exported as follows: • Select **All** to output all image conversion configurations. • Select **Selected** or **Minimum** to output image conversion configurations based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in Data Exchange documentation. • Select **None** to output no image conversion configurations. For more information, refer to the Image Conversion Configuration topic in the Digital Assets documentation.
Include Import Configurations	Assets saved as import configurations are exported as follows: • Select **All** to output all import configurations. • Select **Minimum** to output import configurations based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no import configurations. For more information, refer to the Maintaining a Saved Export Configuration topic in the Data Exchange documentation.
Include Integration Endpoints	Control output of integration endpoint ID, name, and configurations. The configurations are exported in Base64 and can only be imported without modifications into another STEP system. • Select **All** to output all integration endpoints. • Select **Selected** to output only integration endpoints in the selected hierarchy based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no integration endpoints. For more information, refer to the inbound and outbound integration endpoint topics in the Data Exchange documentation.
Include Key Definitions	Select **No** or **Yes** to control output of definitions of objects created as keys. For more information, refer to the Unique Keys topic in the System Setup documentation.
Include Link, Reference and	**Important:** Use caution when handling reference types and object types with an ID that start with **stibo.** since they are fundamental objects.

Configuration Parameter	Description
Object Types	User-created object types created below the 'Setup Group type root' node and system-specific object types, link types, and system specific reference types are exported as follows: • Select **All** to output all edge, reference, and object types. • Select **Selected** or **Minimum** to output only link, reference, and object types in the selected hierarchy based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no link, reference, and object types. **Note:** Edge types are identified with either 'PA' (product to attribute link type) or 'CA' (classification to attribute link type). For more information, refer to the Reference and Link Types topic in the System Setup documentation.
Include List of 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 topic in the Data Exchange documentation. • Select **None** to output no LOV definitions or values. For more information, refer to the List of Values (LOVs) topic in the System Setup documentation
Include Match Codes	Select **No** or **Yes** to control output of match code information, including ID, name, links, setup group, valid entity hierarchies, and valid object types. For more information, refer to the Match Codes topic.
Include Matching Algorithms	Control output of matching algorithms, including ID, name, links, and configuration. • Select **All** to output all matching algorithms. • Select **Selected** to output matching algorithms based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no matching algorithms.

Configuration Parameter	Description
	The configurations are exported in Base64 and can only be imported without modifications into another STEP system. For more information, refer to Configuring Matching Algorithms topic.
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 topic in the Data Exchange 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 topic in the Data Exchange documentation. • Select **None** to output no setup groups. To export user-created object types created below the 'Setup Group type root' node, refer to the **Include Link, Reference and Object Types** configuration parameter above. For more information, refer to the Setup Groups topic in the System Setup documentation.
Include Status Flags	Workflow status flags are exported as follows: • Select **All** to output all workflow status flags. • Select **Selected** to output workflow status flags based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no workflow status flags. For information, refer to the Status Flags topic in the Workflows documentation.
Include System Settings	Select **No** or **Yes** to control output of the following default settings defined on the Users and Groups root node: • Image & Document Settings flipper > Dimension Dependencies

Configuration Parameter	Description
	• 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, refer to the System Settings topic in the System Setup documentation.
Include Table Types	Row type definitions, column type definitions, and table type definitions, including applied transformations are exported as follows: • Select **All** to output all table types. • Select **Minimum** to output table types based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no table types. For more information, refer to the Tables documentation.
Include Tags	Select **No** or **Yes** to control output of style tags, character tags, special characters, footnotes, and hyperlinks, including information about rendering, short cuts, and applied output formatting. For more information, refer to the Tags topic in the System Setup documentation.
Include Transformation Lookup Tables	Assets saved as transformation lookup table configurations are exported as follows: • Select **All** to output all transformation lookup table configurations. • Select **Minimum** to output transformation lookup table configurations based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no transformation lookup table configurations. For information, refer to the Transformation Lookup Tables topic in the Resource Materials online help.

Configuration Parameter	Description
Include Translation Configurations	Saved translation configurations are exported as follows: • Select **All** to export all saved translation configurations. • Select **Selected** to export selected saved translation configurations based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to export no saved translation configurations. For information on saving translation configurations for data objects, refer to the Starting a Structured Translation topic; for information on saving translation configurations for setup objects, refer to the Structured Translation for Setup Objects topic, both in the Translations documentation.
Include Units	Unit groups and units, including applied meta attributes, values, and base unit conversions are exported as follows: • Select **All** to output all unit groups and units. • Select **Selected** or **Minimum** to output unit groups and units based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no unit groups or units. For more information, refer to the Units topic in the System Setup documentation.
Include Users and User Groups	Control output of all user groups and users, including information about applied privileges, meta attributes, restricted GUI setup, etc. User passwords are not included in the output. Since creating a new user requires a password, new users cannot be created via STEPXML import. However, changes to existing users can be imported. **Note:** When changing a user from one assigned group to another, the user is added to the new group, but must be manually removed from the original group. • Select **All** to output all users or user groups. The Select Objects step allows you to limit the users and/or groups after setting the Export parameter to All, use the Add Objects link to define the objects to export. • Select **Selected** to output users or user groups based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no users or user groups.

Configuration Parameter	Description
	For more information, refer to the Users and Groups topic in the System Setup documentation.
Include Web UI Configurations	Control output of Web UI configurations in the exported file. • Select **All** to output all Web UI configurations. • Select **Selected** to output Web UI configurations based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no Web UI configurations. For more information, refer to the Managing Web UI Configurations topic in the Web User Interfaces documentation.
Include Workflows	Workflow information, including ID, name, links, valid object types, and configuration are exported as outlined below. The configurations are exported in Base64 and can only be imported without modifications into another STEP system. • Select **All** to output all workflows. • Select **Selected** to output workflows based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no workflows. For more information, refer to the Workflows documentation.
Include Workflow Profiles	Workflow profiles (including information such as number of exceeded deadlines, how long the tasks have been assigned to the assignees, throughput for the month versus the last six months, etc.) are exported as follows: • Select **All** to output all workflow profiles. • Select **Selected** to output workflow profiles based on the explanation in the Minimum, Referenced, and Selected in STEPXML topic in the Data Exchange documentation. • Select **None** to output no workflow profiles. For more information, refer to the Monitoring Workflows topic in the Workflows documentation.

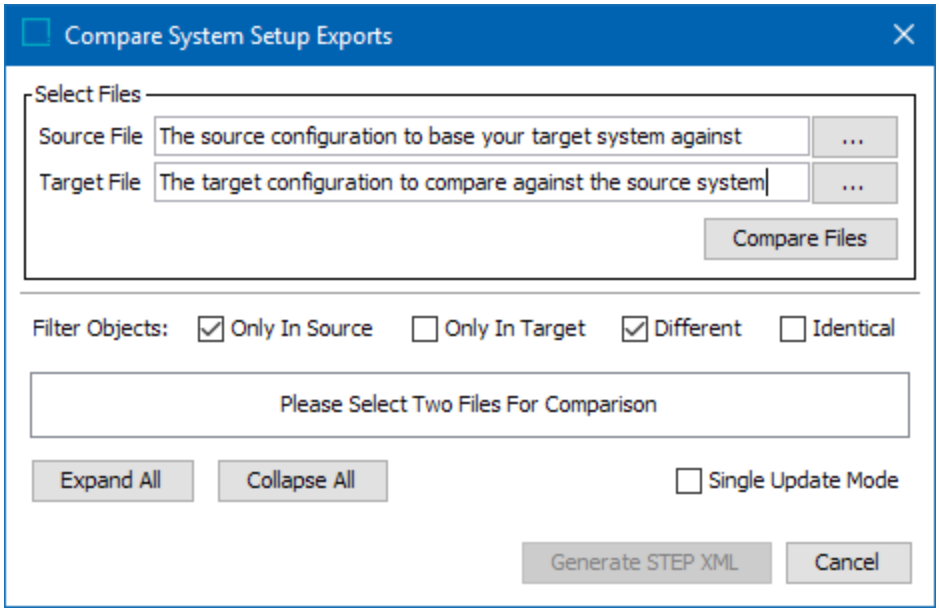
Using the STEPXML Comparison Tool

The information below outlines how you can use the comparison tool to compare a source and target system.

Select Source and Target

To use the comparison tool:

1. From the File menu, click Export, and choose Compare System Setup Exports.
2. Click the ellipsis button (...) and select the exported XML file from the source system.
3. Click the ellipsis button (...) and select the exported XML file from the target system.



Select Filter Options

Once the XML files are selected, the user selects how the comparison tool should filter the differences.

- Only in Source – Filter on configurations that only exist on the source system.
- Only in Target – Filter on configurations that only exist on the target system.
- Different – Filter on what is different between the source and target system.
- Identical – Filter on what is identical between the source and target system.

- Compare Files – Once selected the relevant configuration will be highlighted depending on the filter options selected above.

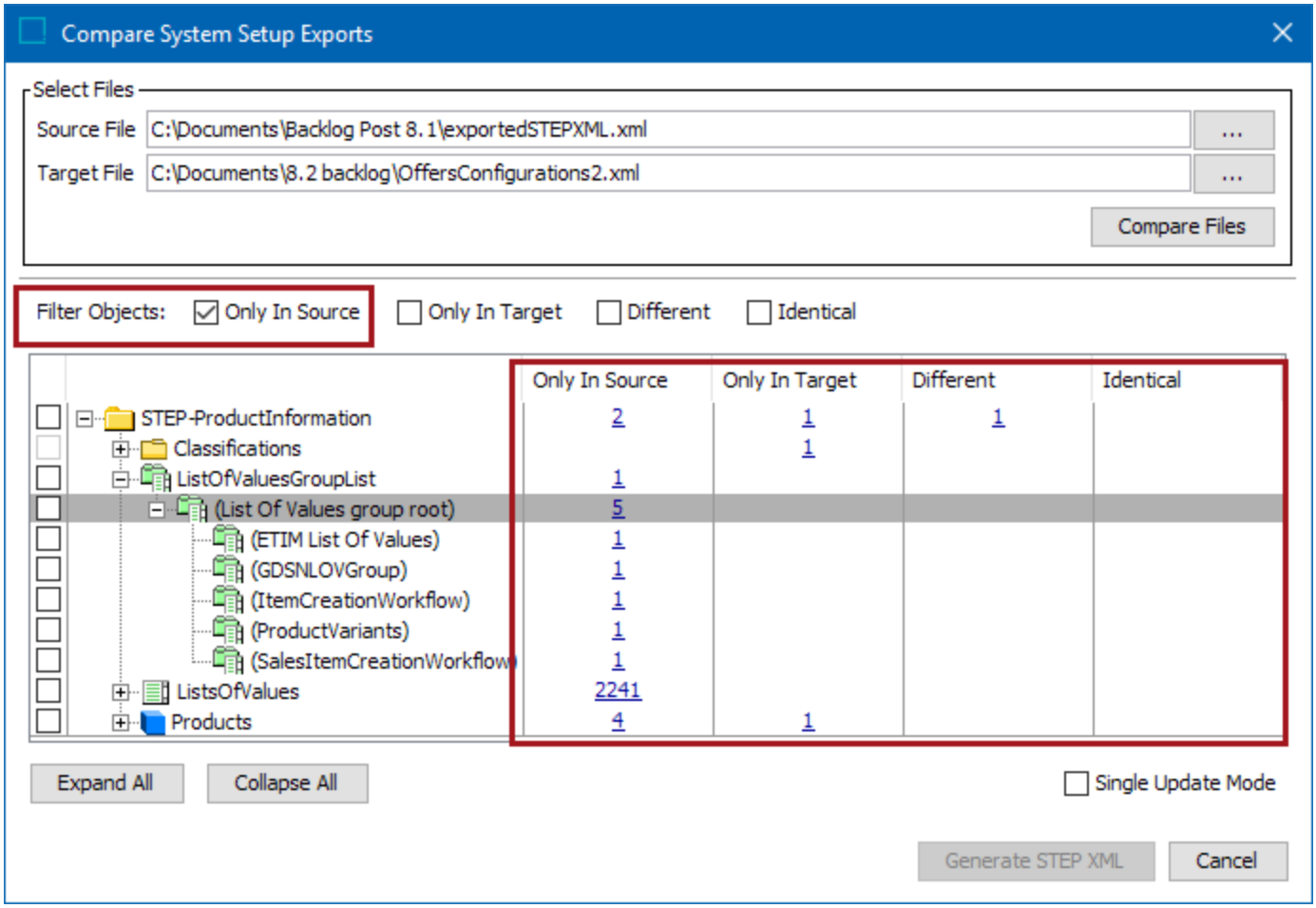
If you need to change the filter options selected, change them and click Compare Files to update

View Differences

Below is an example of how to filter what is only in the source XML file.

The comparison tool gives an overview of the following:

- Only In Source
- Only in Target
- Different
- Identical



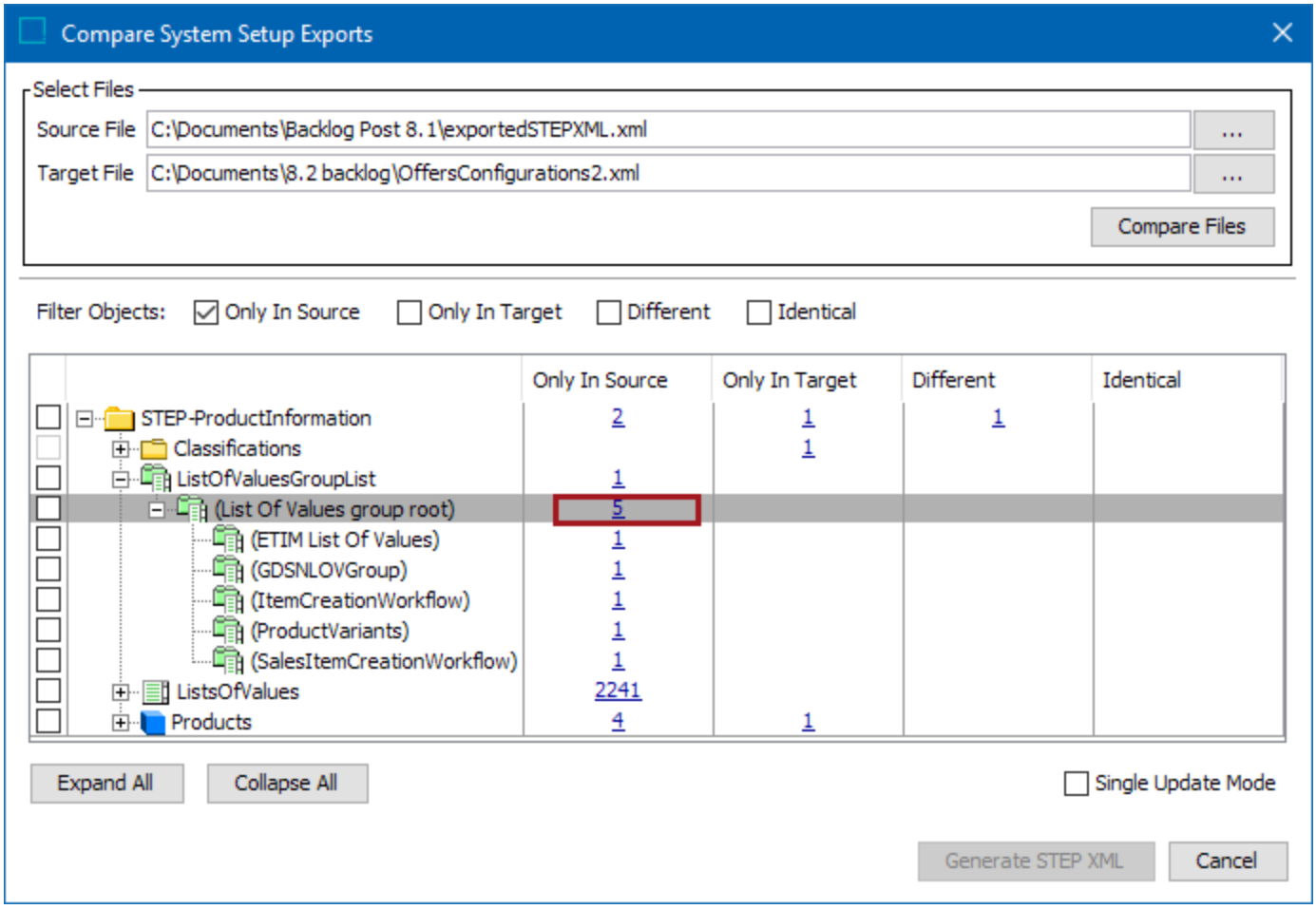
In this example, the user has selected to filter on 'Only In Source.' When the user opens the attribute list configuration, which indicates one attribute, only the attributes in the source system display.

Note: If you select the Expand All option the user is shown all configurations related to the filter selected i.e., Only In Source.

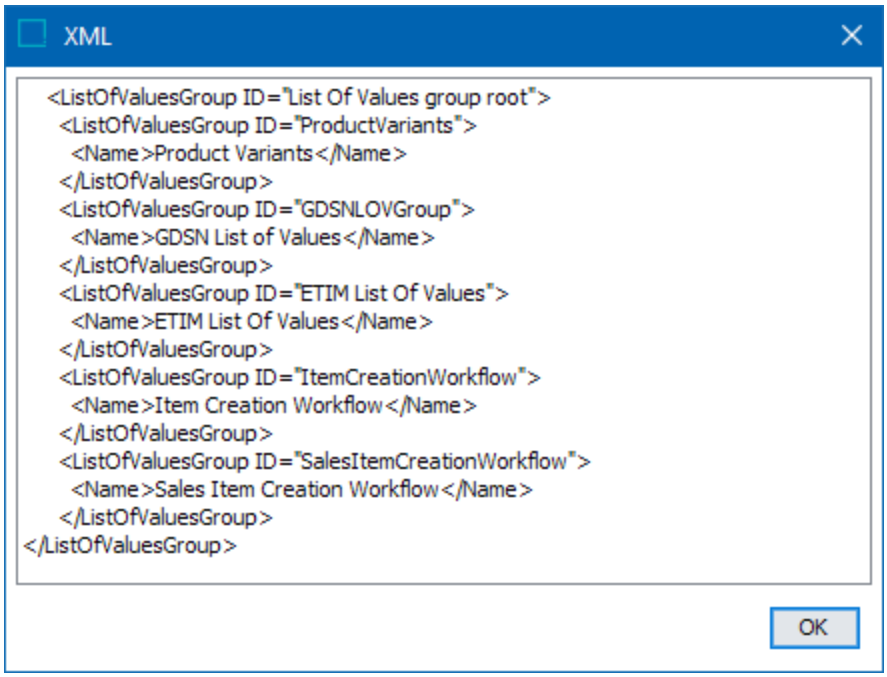
XML file differences

Within the comparison tool users can view the differences between the XML files from source and target systems.

If there are differences for each of the filters between the source and target systems users can view them by selecting the hyperlink for each difference



If you select the hyperlink highlighted in screenshot above, the STEPXML displays for the List Of Values group that only exists in the source system as follows:

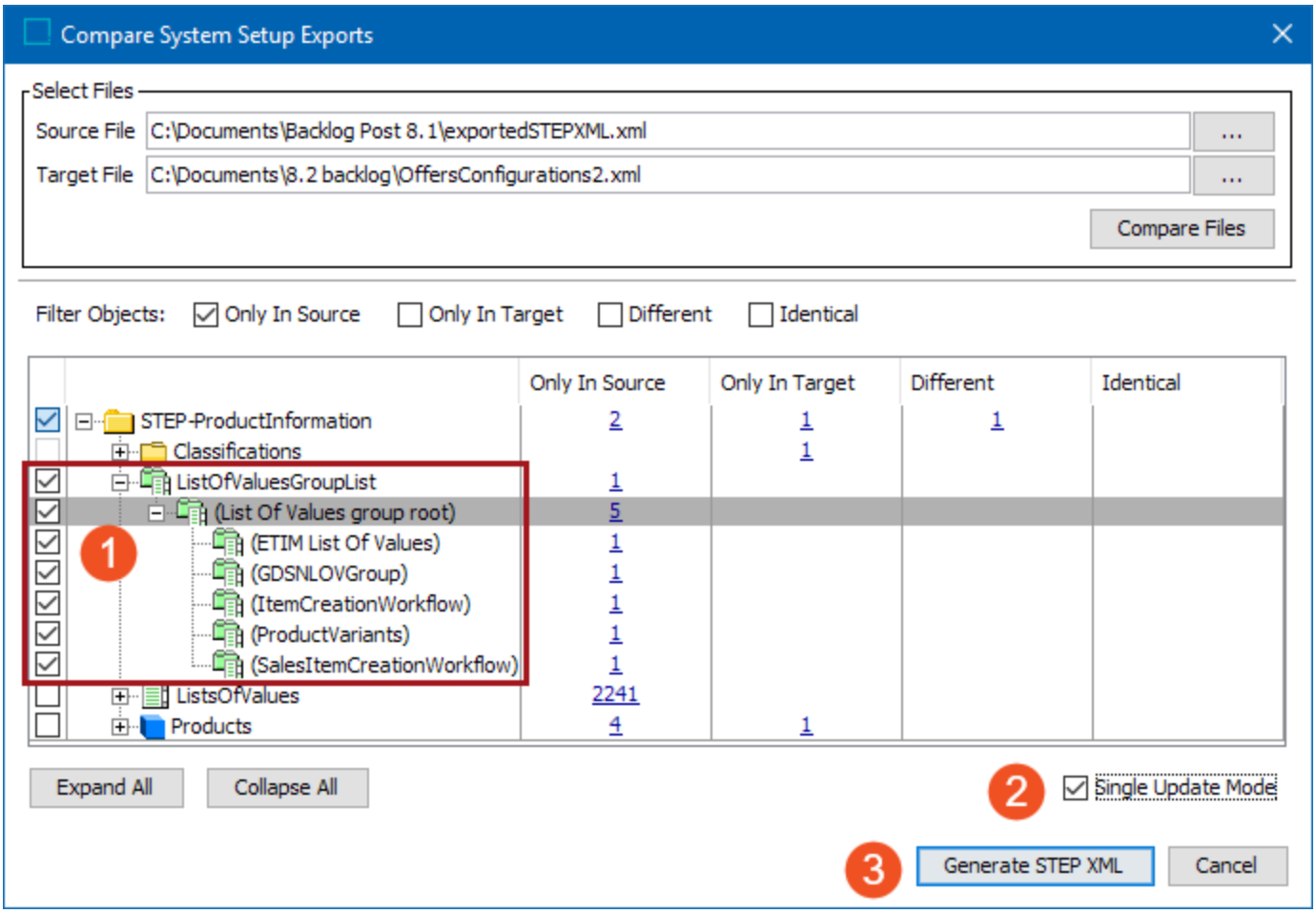


Generate STEPXML

Finally, the user selects the check boxes to identify the configuration required to be exported and then generates an XML file with configuration differences based on this selection.

1. Select the configuration to be moved to the target system. In this case it is the List Of Values Group List.
2. Check Single Update Mode option.

   This option must be checked if the generated STEPXML should make updates that require single update mode. Refer to the Single-Update Mode topic in the System Setup documentation for examples of operations that require Single Update Mode.
3. Generate STEPXML by clicking the Generate STEPXML button.



4. Save the XML file.

STEPXML Comparison Tool Scenarios

The following examples highlight how the STEPXML Comparison Tool can be used for running and loading STEPXML generated via the comparison tool.

It is advisable to run the STEP comparison tool when no one is using the system. The XML being loaded may require 'Single-Update Mode' and entering Single-Update Mode means users temporarily only have read-only access to the system.

If you load XML, which requires Single-Update Mode, and it cannot enter this state due to an active process on the server, the import enters a 'wait' state and then enters 'Single-Update Mode' when there are no active processes.

If the XML being loaded is not set to go into Single-Update Mode when imported, the process highlights that Single-Update Mode is required but was not successful.

Configurations for Exporting the Data

When doing the first export where you select the configurations you require to be exported, it is advisable to save a configuration file. As there are a number of configurations, you could miss a vital configuration if a user manually sets this each time they do a configuration export.

Scenario 1 - In this scenario we need to identify what is different between our source and target systems and update the target with the necessary updates.

Checking what is different between system to generate STEPXML to update target system:

- Back-up target system
- Export XML from Source system excluding Assets, Classifications, Entities and Products
- Run a Cross Context export if configuration is stored in more dimension points i.e., LOVs, attribute names etc.
- Export XML from Target system as above
- The compare tool will highlight what is on the Source system only and what is different
- Generate STEPXML tool
- Load into target system
- Check execution report for errors and resolve
- Use compare tool to identify what is different or only on the target system to remove or update

Scenario 2 - In this scenario we need to identify what only exists on the target system which will have to be manually removed or updated.

Removing configuration from a target system:

- Back-up target system
- Export XML from Source system excluding Assets, Classifications, Entities and Products
- Export XML from Target system as above
- The compare tool will highlight what is on the Target system only
- STEP user will need to manually remove the specific configurations from the target system

Scenario 3 - In this scenario we need to compare the system only.

Compare configurations to identify if the source and target systems match each other:

- Export XML from Source system excluding Assets, Classifications, Entities and Products
- Run a Cross Context export if configuration is stored in more dimension points i.e., LOVs, attribute names etc
- Export XML from Target system as above
- The compare tool will highlight what is not identical

Scenario 4 - In this scenario, use the compare tool to generate XML for specific object types. For example, to move two product types from source system to the target, choose to compare the same file and decide the objects to generate XML for.

Generating valid STEPXML:

- Export XML from Source system excluding Assets, Classifications, Entities and Products
- Run a Cross Context export if configuration is stored in more dimension points i.e., LOVs, attribute names etc
- Re-use the source XML in the target
- The compare tool will highlight what is identical and you can choose to view the XML via the hyperlinks for the appropriate objects

Considerations for STEPXML Imports

When using the STEPXML Comparison Tool, review the following considerations.

Single-Update Mode

- Configuration updates can require STEP to go into 'Single-Update Mode'. The updates that require Single-Update Mode via the comparison tool are listed the STEPXML Comparison Tool Limitations topic
- Change attribute to / from being free text searchable - Yes
- Change attribute to / from being multi valued - Yes
- Activate / Deactivate unique keys - Yes but only the configuration
- Change reference type to / from being multi valued

- Modify or Move classification-product link types
  (e.g., move from one type to another type)
- Change classification-product link type to / from being multi valued
- Remove child object type in product, classification, entity, or publication object type hierarchy

Removing valid object types from an attribute

Issues that can occur when removing validity:

- Remove object types as being valid for an attribute are not allowed when data exists for applicable products. A warning is included in the execution report.
- Users must manually insert the XML tag OnlyAllowValidUserTypes='true' in the STEP-ProductInformation tag.
- Attribute changes from Text to Number validation may not be allowed if there is data within the system for products that do not conform to Number validation.

Preparation required in target system for a successful import

Issues that could occur during STEPXML import:

- New users cannot be created by the import since a password is required.
- Event queues and consumers are created with a disabled status in the target system.
- If the configuration being loaded relies on an object that does not exist within STEP a warning is displayed in the execution report. Required nodes must exist to receive data being loaded.

  For example:

  - Valid object types must exist when loading attributes
  - Dimension points must exist when loading linked contexts
  - Referenced objects must exist when loading privilege rules
  - Referenced user groups must exist when loading Stateflows
  - Classification nodes must exist to hold bulk update configurations, export configurations, import configurations, transformation lookup tables, and Web UI configurations
  - Collections must exist for product selections based on collections

STEPXML Comparison Tool Limitations

The comparison tool can be used to create STEPXML to modify and add relevant configuration. It will not delete or modify configuration that is already in use within STEP. It will also not make updates that require user input.

Listed below are the configurations within STEP that cannot be updated using this tool:

- Cannot change a list of values (LOV) with an auto-generated ID.
- If attributes have been merged on the source system, the redundant attribute on the target system must be removed manually.
- If list of values (LOV) have been merged on the source system, the redundant values need to be removed / merged on the target system.
- Cannot swap attribute IDs.
- Cannot change an attribute from internal to externally maintained as this requires user input; i.e., where to take the values from Main / Approved workspace.
- Cannot remove dimension dependencies as this requires user input to determine which values to keep after removing the dependency.
- Cannot remove Workspaces. This is a manual task.
- Cannot change an attribute to have LOV validation or to not use LOV validation.
- Cannot change reference types to / from being externally maintained as this requires user input.
- Cannot change classification-product link type to / from being externally maintained as this requires user input.
- Cannot change 'Owns Product Links' setting on classification object type.
- Cannot change Revisability of an entity object type.

Version Control System Integration

To support Version Control System Integration (VCSI), a set of outbound integration endpoint options allow STEP to publish system configurations to a branch in Git, an external Version Control System (VCS) (refer to: https://git-scm.com). Using an inbound integration endpoint (IIEP), files from a Git branch can be combined, enriched with processing instructions, and imported on a target system as a STEPXML file.

This functionality allows for easy comparison of configurations across systems in a Development, Test, Acceptance, and Production (DTAP) environment and is meant to aid customers who need to transfer configuration changes between different systems and/or ensure systems are in sync configuration-wise. Users can choose to configure a collection to group changes or a change package.

Note: For on-premises systems, the 'configuration-management' add-on component must be activated to enable VCSI.

Configuration / Data

In STEP, the distinction between configuration and data is not always clear. While most of the objects and settings that can be found in the workbench System Setup tab are clearly configuration, the Tree tab holds both data and configuration. For example, objects like import, export, and bulk update configurations are configuration, but in addition, classification hierarchies, upper levels in the product and entity hierarchies, and entity structures used for modeling reference data, are widely regarded as configuration.

For the functionality presented in the related topics and in the topics of the Version Control System Integration (VCSI) documentation, there is no distinction between 'configuration' and 'data.' Instead, as described in the OIEP for VCS Integration with Git Delivery or the OIEP for VCS Integration with Change Packages topic, this can be configured. However, the functionality is not designed to handle vast amounts of data objects (e.g., SKUs).

Additional Information

Refer to the following topics for more information:

- Integration Endpoint Options for VCS Integration
- OIEP for VCS Integration with Change Packages
- OIEP for VCS Integration with Git Delivery
- IIEP for VCS Integration
- VCSI: Editable Business Rules Format
- VCSI: Example Setups
- VCSI: Considerations and Limitations

Integration Endpoint Options for VCS Integration

The suite for Version Control System Integration (VCSI) consists of these options:

- STEPXML Splitter Post-processor in Outbound Integration Endpoint (OIEP)
- Git Delivery Method or Change Package Git Delivery Method in Outbound Integration Endpoint (OIEP)
- STEPXML Joiner Pre-processor in Inbound Integration Endpoint (IIEP)
- Invoke OIEP Post-processor Inbound Integration Endpoint (IIEP)

Each is described in their own section below.

Additionally, within this topic, is information regarding an editable business rule format. JavaScript-based business rules can be created, maintained, and unit tested outside of STEP. This allows customers and partners to govern the lifecycle of business rules in a standard source code control system such as Git, and from there, deploy appropriate versions of the business rules to the various STEP systems that are part of a Development, Testing, Acceptance and Production (DTAP) environment.

Grouping Changes

Options available for grouping configuration objects include select objects, typically using a collection or event-driven with change packages upon sealing. Initial setup requirements are different for each option.

Change packages are recommended when working on projects in an iterative development process, where smaller changes are grouped separately for the complete configuration. However, a change package can also be used to group a more complete configuration definition.

STEPXML Splitter Post-processor in OIEP

Use of the STEPXML Splitter is determined by the grouping option selected:

- For the Change Package Git Delivery method, the STEPXML Splitter is integrated into the delivery method, so the STEPXML Splitter post-processor must not be added to the OIEP.

  For change packages, all exports use the Flattened split mode and export definitions as comments. The user can decide to also export business rules converted to an editable format on the delivery method dialog, which replaces the STEPXML files that include comments with editable JSON files for business rules.

- For a collection used to group changes and an Advanced STEPXML template configured for various configurations that do not include the Change Package object, the STEPXML Splitter is required on the OIEP.

The STEPXML Splitter post-processor can take any STEPXML file produced by the STEP Exporter processing engine as input and splits the file into multiple valid STEPXML files and/or editable business rule format files that are then passed to the configured delivery method. Generally, the splitter produces one file per STEP object and further normalizes the content so that elements for which the sequence has no significance in STEP are output in

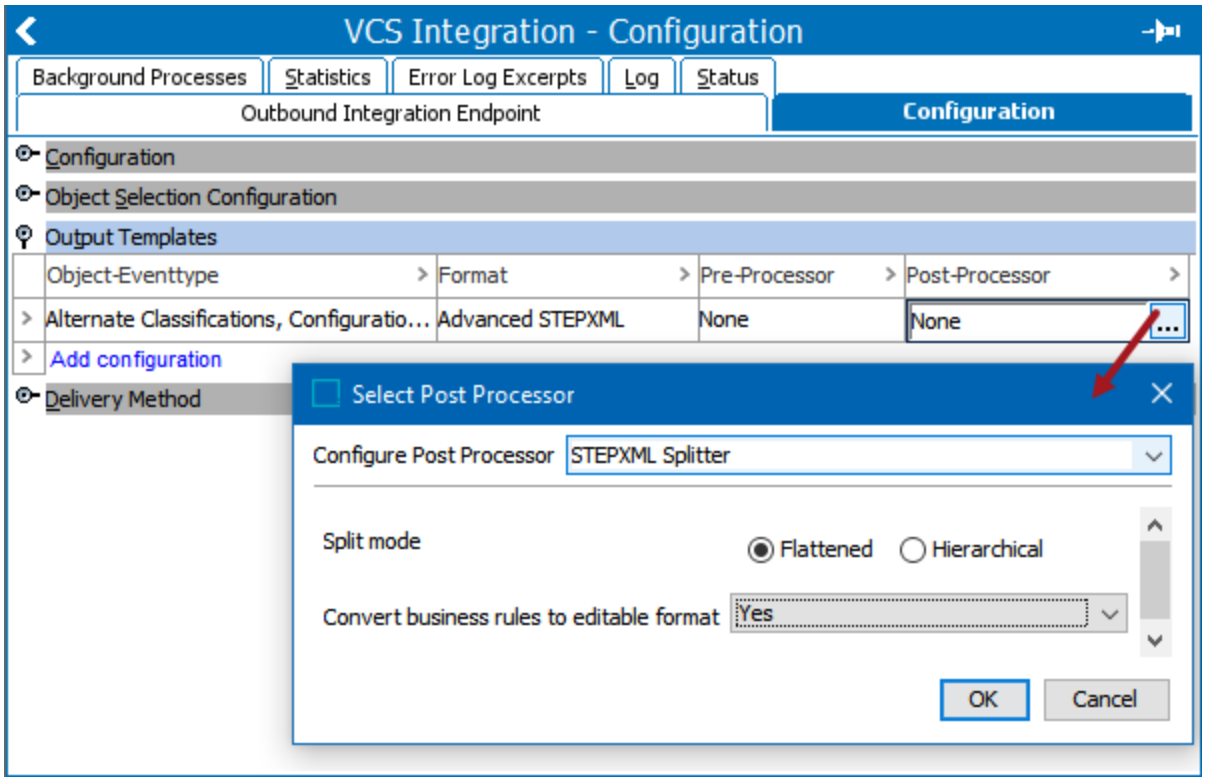
the same order every time. Non-object configurations (e.g., derived events) and system settings are output in a single file.

Splitting and normalizing makes it easier to compare configurations outside of STEP in a VCS like Git. Further, it makes it easy to selectively choose specific configuration items to be imported on another system.

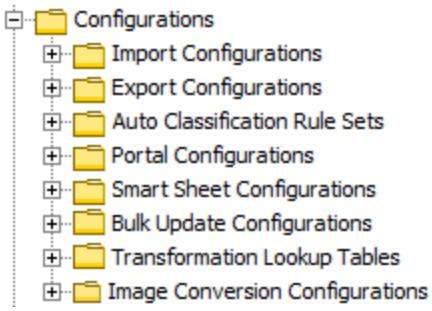
The STEPXML Splitter post processor has the configuration options explained below.

Split mode

Split mode defaults to ‘Flattened’ but also allows 'Hierarchical'. The parameter affects how to represent STEP objects in the produced split files when they are typically exported in a nested structure.



To illustrate the difference, in the example shown below, the following classification hierarchy is being exported:



In both modes, one file will be created per classification object.

In **Flattened** mode, upper levels are omitted, and each file contains exactly one 'Classification' element with parent identifier information is included in this topic in online help.

In **Hierarchical** mode, each of the leaf classifications (e.g., Import Configurations) is nested inside the element representing the 'Configurations' classification, which is stripped of all but ID, object type ID, and parent ID information is included in this topic in online help.

Generally, it is recommended to use the default 'Flattened' mode while the 'Hierarchical' mode should only be used if the full hierarchy path must be present in each file.

Note: The 'Hierarchical' example from above can be imported on a system where the classification with ID 'ConfigurationsRoot' is not present as it will be created during import. Importing the 'Flattened' example on such a system, however, results in an error.

Convert business rules to editable format

This option determines how business rules (conditions, actions, functions, and libraries) are exported.

- If set to **No**, the business rules are exported as STEPXML files.
- If set to **Yes**, the rules are exported in the editable *.js format described in this topic.

  When exporting editable business rules, set this option to 'Yes.' The business rules in the STEPXML that are fed to the post-processor are converted to the editable format and represented in a single *.js file instead of being represented in a STEPXML file. For details, refer to the VCSI: Editable Business Rules Format topic.

Change Package Git Delivery Method in OIEP

The Change Package Git Delivery method allows integration with popular repositories, supporting the HTTPS (token-based) or the SSH (file-based) access methods for GitHub, GitLab, and Bitbucket.

The Change Package Git Delivery method delivers files produced by the OIEP processing engine using an integrated STEPXML Splitter to deliver multiple files to a branch in a remote Git repository. Refer to https://git-scm.com for more information about Git.

The change package is represented below the specified branch within a configurable directory structure. At the end of the directory structure, change packages grouping files by the Primary and Secondary flippers, then by type of object with XML or JSON files named by object type and the ID of the object.

Note: The remote repository cannot be empty. At least one branch with one file must exist (e.g., the 'main' branch and the README.md file).

Prerequisites

Important: For on-premises systems, this feature requires the configuration-management component.

Changes to shared configuration properties on the STEP application server are required to populate the dropdown list delivery options. Changing these properties does not require a system to be restarted.

Prior to configuring the Change Package Git Delivery method, add the desired case-sensitive properties described below.

Note: In the properties mentioned below, use a sequential set of variable integers in the property names, for example, 1, 2, 3. Property values are not read correctly if the sequence is not sequential, for example 1, 3, 4.

1. For both on-prem and Stibo System SaaS systems, the values for the **Remote Git Repository URI** parameter defines the URI for the remote repository and are read from the case-sensitive `ChangePackageGitDelivery.RemoteRepoUri.[integer]` property. For example:

```
ChangePackageGitDelivery.RemoteRepoUri.1=https://gitlab.com/john-smith/step-conf.git
ChangePackageGitDelivery.RemoteRepoUri.2=git@bitbucket.org:john-smith/smithrepo.git
```

   Note: For internally hosted Git setups, ensure that any firewall access rules, IP white-listing, or ports (tcp/22 or similar) are configured to allow the STEP server(s) to access the needed Git repository.

2. The values for the **Git Branch** parameter define the name of the branch to which the delivery is published, and are read from the case-sensitive `ChangePackageGitDelivery.Branch.[integer]` property. For example:

```
ChangePackageGitDelivery.Branch.1=step-dev-1
ChangePackageGitDelivery.Branch.2=step-q
```

3. The values for the **Repository User Name** parameter defines the remote repository username and supports delivery via HTTPS and PAT or SSH with a key file accessible to the application server (discussed later). Values for this parameter are read from the case-sensitive `ChangePackageGitDelivery.RemoteRepoUsername.[integer]` property. For example:

```
ChangePackageGitDelivery.RemoteRepoUsername.1=john.smith
ChangePackageGitDelivery.RemoteRepoUsername.2=julie.baker
```

Using SSH to Connect

To connect via SSH, the **Path to Private Key When Using SSH** parameter defines the remote repository RSA SSH private key generated using the old OpenSSH format and ed25519. The generated key file must be accessible to the application server through shared storage or SFTP.

Important: PuTTYgen (.ppk) SSH keys are not supported by STEP. Using an unsupported key type results in an 'invalid private key' error.

Follow these steps to generate a valid private SSH key:

1. Use ssh -V to check the version on your system and then generate a key.
   - With OpenSSH versions prior to 7.8, use the following command to generate the new OpenSSH format key:

     ```
     ssh-keygen -t rsa -b 4096 -C <comment> -f <keyfile_name>
     ```

     For example:

     ```
     ssh-keygen -t rsa -b 4096 -C john.smith@acme.com -f git_rsa4096_key
     ```

   - With OpenSSH versions 7.8+, generated keys default to the new OpenSSH format. To generate keys in the old format, use the following command:

     ```
     ssh-keygen -t rsa -b 4096 -C john.smith@acme.com -m PEM -f git_rsa4096_key
     ```

     Important: While the general approach and commands are the same with later versions of OpenSHH, the `-m PEM` argument shown above is needed with versions 7.8+.

2. Add the public version of the SSH key (e.g., git_rsa4096_key.pub) to your GitLab or Bitbucket account. Failure to add the public key to your account results in a 'Not authorized' error.
   - For GitHub accounts, refer to the GitHub Limitations and Configuration section below.
   - For Bitbucket accounts, refer to the Bitbucket Limitations and Configuration section below.
   - For a general overview of supported authentication methods, refer to the Supported Authentication Methods (per Git Service) section below.

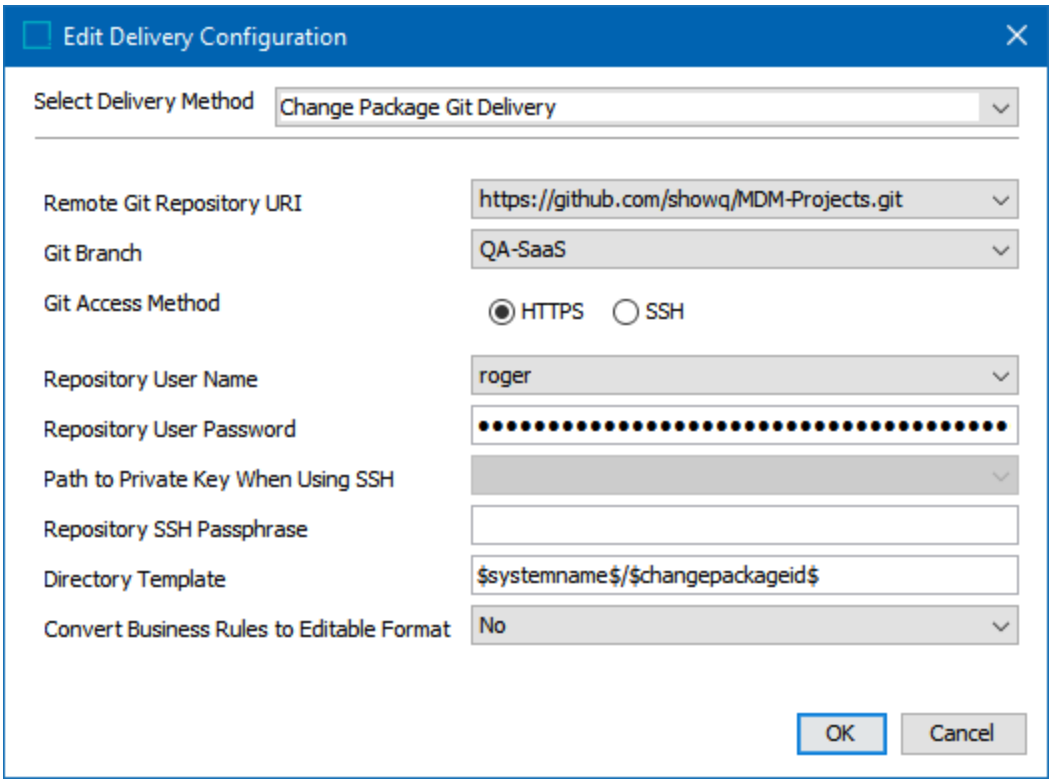
     Values for the **Path to Private Key When Using SSH** parameter are read from the case-sensitive `GetDelivery.SshPrivateKeyUri.[integer]` property. These entries must contain the full path to the private SSH key file. For example:

     ```
     GitDelivery.SshPrivateKeyUri.1=/home/stibosw/.ssh/git_rsa4096_key
     GitDelivery.SshPrivateKeyUri.2=/workarea/.ssh/gitlab_rsa4096_enc_key
     GitDelivery.SshPrivateKeyUri.3=/upload/.ssh/bitbucket_rsa4096_key
     ```

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 **Change Package Git Delivery**.



3. In **Remote Git Repository URI**, select a URI from the dropdown for the relevant remote repository.
4. In **Git Branch**, select relevant branch in the associated remote repository.
5. In **Git Access Method**, select HTTPS or SSH and provide additional values as required.
   - For **HTTPS**:
     - For the **Repository User Name** parameter, select an option from the dropdown.
     - For the **Repository User Password** parameter, add the personal access token generated from your repository developer tools.
   - For **SSH**:
     - For the **Path to Private Key When Using SSH** parameter, select the file path to the private key from the dropdown.
     - For the **Repository SSH Passphrase** parameter, add the passphrase.

     Note: Author name and email for the Git commit are sourced automatically from the user object associated with the sealing event on the change package and does not require explicit configuration in the delivery method. However, to support this functionality, email address values must exist for users that seal change packages that are integrated with Git.

6. In **Directory Template**, indicate where to store files on the repository using a standardized format. While editing the Directory Template, hover over the field to display the available macros.

   Use static text, macros, and the forward slash character as a separator to automatically create directories and organize change packages by:

   - `$systemname$/$changepackageid$` - default, creates a directory under the branch for each system name where a change package is sent from and sub directories with the ID of the change package.
   - `$systemname$` - system name, where the change package is sealed
   - `$changepackageorigin$` - origin, where the change package was initially created
   - `$changepackageuniqueid$` - system-defined unique ID
   - `$changepackageid$` - user-defined change package ID
   - `$attribute:attributeid$` - For additional flexibility, custom description attributes that are externally maintained and not dimension dependent, can be added to the change package object and referenced in the template to create subfolders based on values you define. When a custom attribute is used in the directory template, values must be present and LOV validation base type is recommended.

     Note: a-z, A-Z, 0-9,-, and _ are supported characters for static text. Other characters are replaced with an underscore.

7. In **Convert Business Rules to Editable Format**, select an option:

   - **Yes** - default, sends editable JSON files of business rules (actions, conditions, functions, and libraries) to the repository. The JSON files can be imported manually or with other files combined in a ZIP file that is sent to an IIEP using the STEPXML Joiner preprocessor.
   - **No** - exports STEPXML files that do not provide editable JavaScript access but can be imported as-is after export and include definitions as comments.

8. On the **Edit Delivery Configuration** dialog, click the **OK** button to save the delivery method.

Git Delivery Method in OIEP

The Git Delivery method delivers files produced by the outbound integration endpoint (OIEP) processing engine or a configured post-processor to a branch in a remote Git repository. Refer to https://git-scm.com for more information about Git.

When using this delivery method on an on-premises STEP system, the delivery to Git can be performed via a configured local directory accessible from all application servers in the cluster (refer to the Local git repository URI section below). With this setup, the delivery method first checks if the local delivery directory is Git enabled.

If the local delivery directory is Git enabled, the following operations are performed:

1. Git fetch
2. Git checkout (of configured branch)

3. Git hard reset
4. Git pull (if branch exists in remote)

If the local delivery directory is not Git enabled, the following operations are performed:

1. Git clone
2. Git checkout (of configured branch)

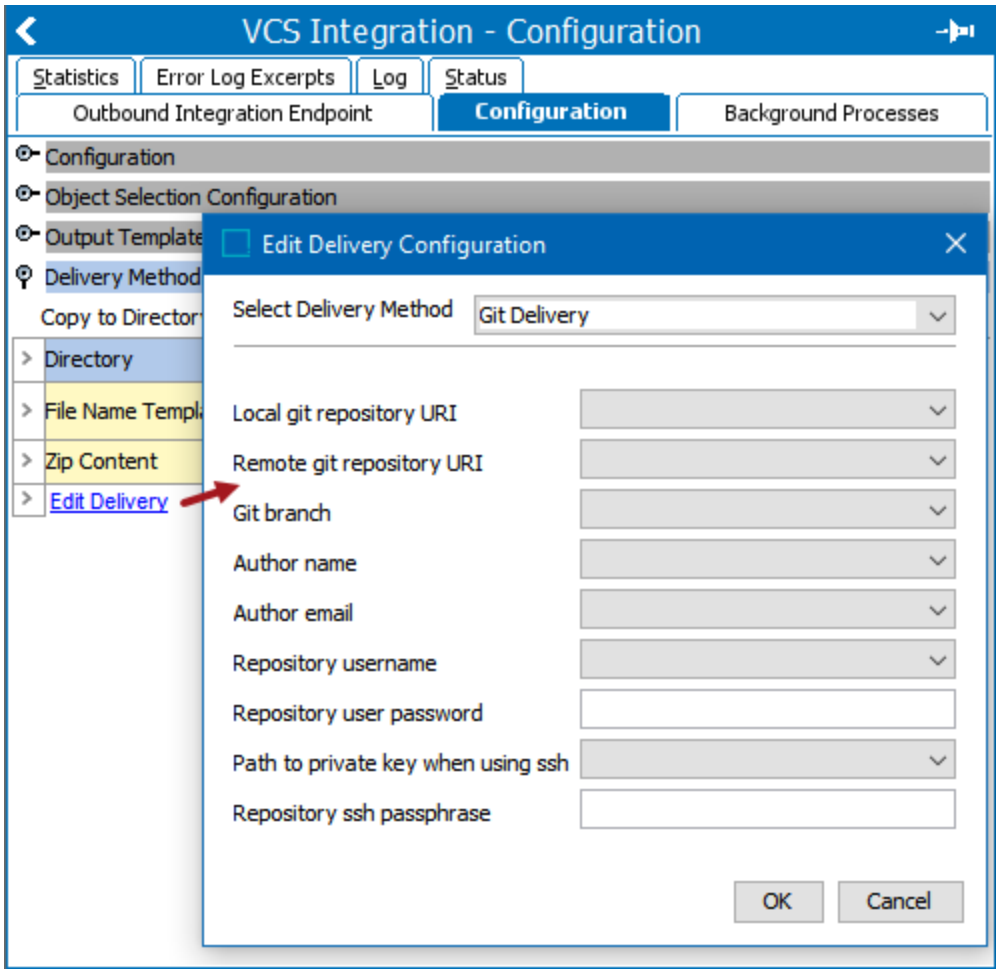
On cloud-based STEP systems, a temporary local directory is always used and the second approach (clone and checkout) is used.

Note: The remote repository cannot be empty. At least one branch with one file must exist (e.g., the 'main' branch and the README.md file).

The locally checked-out branch is now in sync with the remote branch, and the following operations are performed:

1. Files produced by the OIEP are written to the local directory
2. Files present in the local directory but not in the delivery are deleted
3. Git stage
4. Git commit
5. Git push

Configure the Git Delivery option using the following parameters:



Note: In the properties mentioned below, use a sequential set of variable integers in the property names, for example, 1, 2, 3. Property values are not read correctly if the sequence is not sequential, for example 1, 3, 4.

- **Local git repository URI** - URI for the local directory via which configuration files are synchronized.

  For on-prem systems, values for this parameter are read from the case-sensitive `GitDelivery.LocalRepoUri.[integer]` sharedconfig.properties file entries.

  Example:

  ```
  GitDelivery.LocalRepoUri.1=/workarea/conf-attributes
  .LocalRepoGitDeliveryUri.2=/workarea/conf-all
  ```

  allows selection between the values '/workarea/conf-attributes' and '/workarea/conf-all' for the parameter in the workbench.

  For cloud STEP systems, do not set the sharedconfig.properties file entry and leave this parameter blank.

- **Remote git repository URI** - URI for the remote repository.

For both on-prem and cloud STEP systems, values for this parameter are read from the case-sensitive `GitDelivery.RemoteRepoUri.[integer]` sharedconfig.properties file entries.

Examples:

```
GitDelivery.RemoteRepoUri.1=https://gitlab.com/john-smith/step-conf.git
GitDelivery.RemoteRepoUri.2=git@bitbucket.org:john-smith/smithrepo.git
```

Note: For internally hosted Git setups, you must ensure that any firewall access rules, IP white-listing, or ports (tcp/22 or similar) have been configured to allow the STEP server(s) to access the needed Git repository.

- **Git branch** - Name of the branch to which the delivery is published.

  Values for this parameter are read from the case-sensitive `GitDelivery.Branch.[integer]` sharedconfig.properties file entries.

  Examples:

  ```
  GitDelivery.Branch.1=step-dev-1
  GitDelivery.Branch.2=step-qa
  ```

- **Author name** - Author name for the Git commit.

  Values for this parameter are read from the case-sensitive `GitDelivery.AuthorName.[integer]` sharedconfig.properties file entries.

  Examples:

  ```
  GitDelivery.AuthorName.1=John Smith
  ```

- **Author email** - Author email for the Git commit.

  Values for this parameter are read from the case-sensitive `GitDelivery.AuthorEmail.[integer]` sharedconfig.properties file entries.

  Examples:

  ```
  GitDelivery.AuthorEmail.1=john.smith@acme.com
  ```

- **Repository username** - The remote repository username supports delivery via HTTPS and PAT (discussed later).

  Values for this parameter are read from the case-sensitive `GitDelivery. RemoteRepoUsername.[integer]` sharedconfig.properties entries.

  Example:

  ```
  GitDelivery.RemoteRepoUsername.1=john.smith
  ```

Note: The 'Repository username' and 'Repository user password' parameters and properties are only used for HTTPS-based and PAT-based (discussed later) connections. For SSH-based connections, leave these properties blank since they are ignored. SSH uses only the 'Path to private key when using ssh' and 'Repository ssh passphrase' (if applicable) parameters and properties.

- **Repository user password** - The remote repository password supports delivery via HTTPS and PAT (discussed later.) Password must be entered directly when configuring the delivery method for these authentication methods.
- **Path to private key when using ssh** - The remote repository RSA SSH private key generated using the old OpenSSH format.

Important: PuTTYgen (.ppk) SSH keys are not supported by STEP. Using an unsupported key type results in an 'invalid private key' error.

1. Use `ssh -v` to check the version on your system and then generate a key.
   - With OpenSSH versions prior to 7.8, use the following command to generate the new OpenSSH format key:

     ```
     ssh-keygen -t rsa -b 4096 -C <comment> -f <keyfile_name>
     ```

     Example:

     ```
     ssh-keygen -t rsa -b 4096 -C john.smith@acme.com -f git_rsa4096_key
     ```

   - With OpenSSH versions 7.8+, generated keys default to the new OpenSSH format. To generate keys in the old format, use the following command:

     ```
     ssh-keygen -t rsa -b 4096 -C john.smith@acme.com -m PEM -f git_rsa4096_key
     ```

     Important: While the general approach and commands are the same with later versions of OpenSHH, the `-m PEM` argument shown above is needed with versions 7.8+.

2. Add the public version of the SSH key (e.g., git_rsa4096_key.pub) to your GitLab or Bitbucket account. Failure to add the public key to your account results in a 'Not authorized' error.
   - For GitHub accounts, refer to the GitHub Limitations and Configuration section below.
   - For Bitbucket accounts, refer to the Bitbucket Limitations and Configuration section below.
   - For a general overview of supported authentication methods, refer to the Supported Authentication Methods (per Git Service) section below.

Values for this parameter are read from the case-sensitive `GetDelivery.SshPrivateKeyUri.[integer]` sharedconfig.properties entries. These entries must contain the full path to the private SSH key file.

Example:

```
GitDelivery.SshPrivateKeyUri.1=/home/stibosw/.ssh/git_rsa4096_key
GitDelivery.SshPrivateKeyUri.2=/workarea/.ssh/gitlab_rsa4096_enc_key
GitDelivery.SshPrivateKeyUri.3=/upload/.ssh/bitbucket_rsa4096_key
```

- **Repository ssh passphrase** - The remote repository SSH passphrase. The passphrase must be entered directly when configuring the delivery method.

Supported Authentication Methods (per Git Service)

	Protocols		
Git Services	**HTTPS**	**SSH**	**PAT***
GitHub	✗	✓	✓
GitLab	✓	✓	✓
Bitbucket	✗	✓	✓

*PAT = Personal Access Token / App Password

GitHub Limitations and Configuration

The following limitations apply when using GitHub with STEP:

Unlike other Git services, GitHub is more restrictive in the level of security placed on accessing its repositories, specifically:

- Basic authentication for HTTPS connections is not supported.
- RSA SSH keys that use the sha-1 signature algorithm are not supported.

Therefore, the available authentication methods for using the Change Package Git Delivery method or Git Delivery Method with a GitHub account are with a Personal Access Token (PAT) or SSH with a private key accessible to the application server.

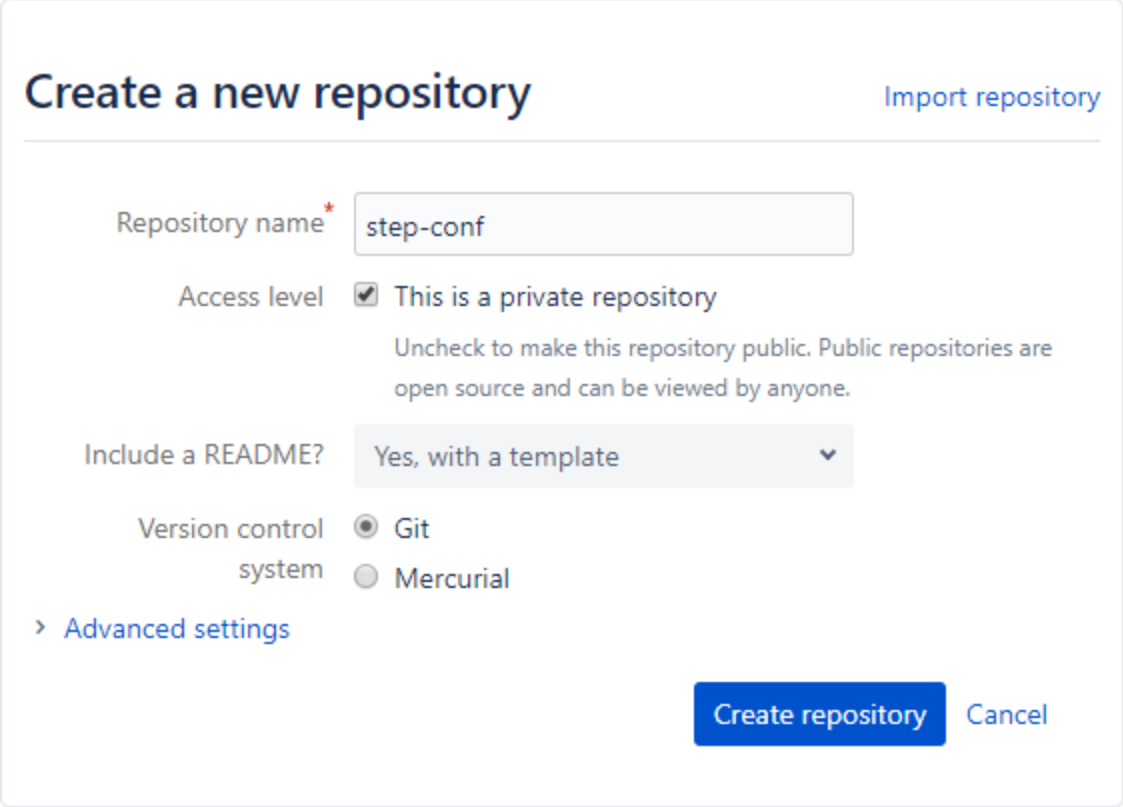
Bitbucket Limitations and Configuration

Bitbucket does not support using Basic authentication for HTTPS connections. This means that apart from SSH keys, the only other usable authentication method is app passwords.

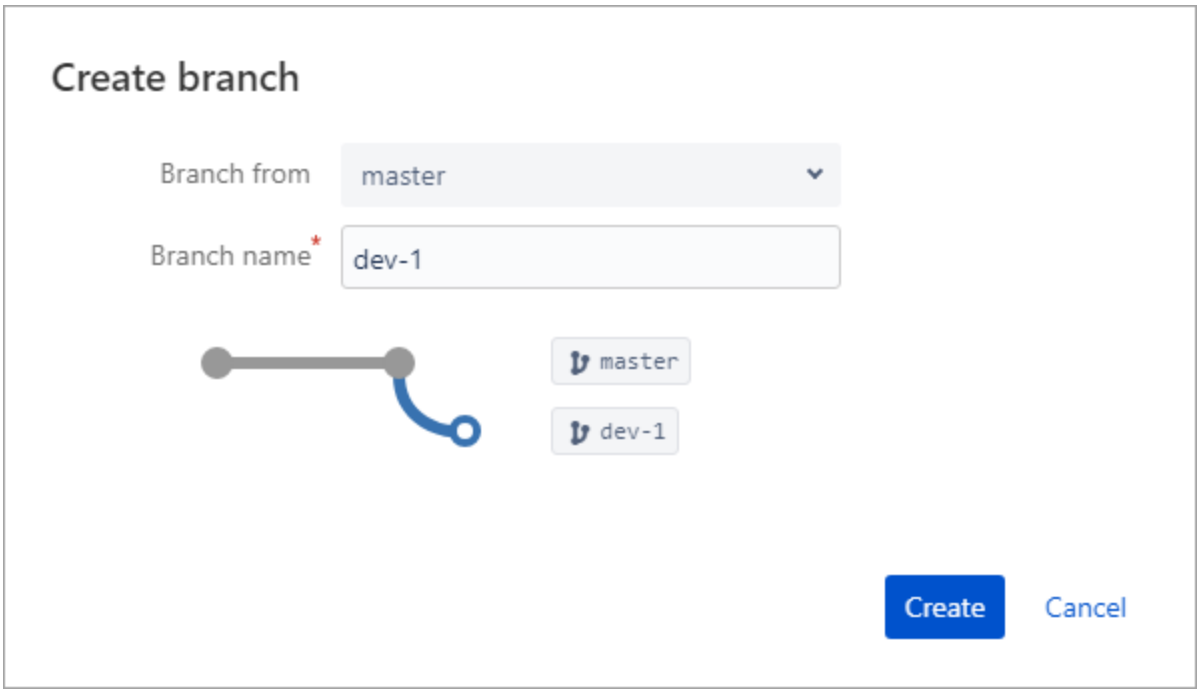
Remote Setup Example

Below is an example of how a remote Git repository using Bitbucket (https://bitbucket.org) can be configured to work with the Git delivery method.

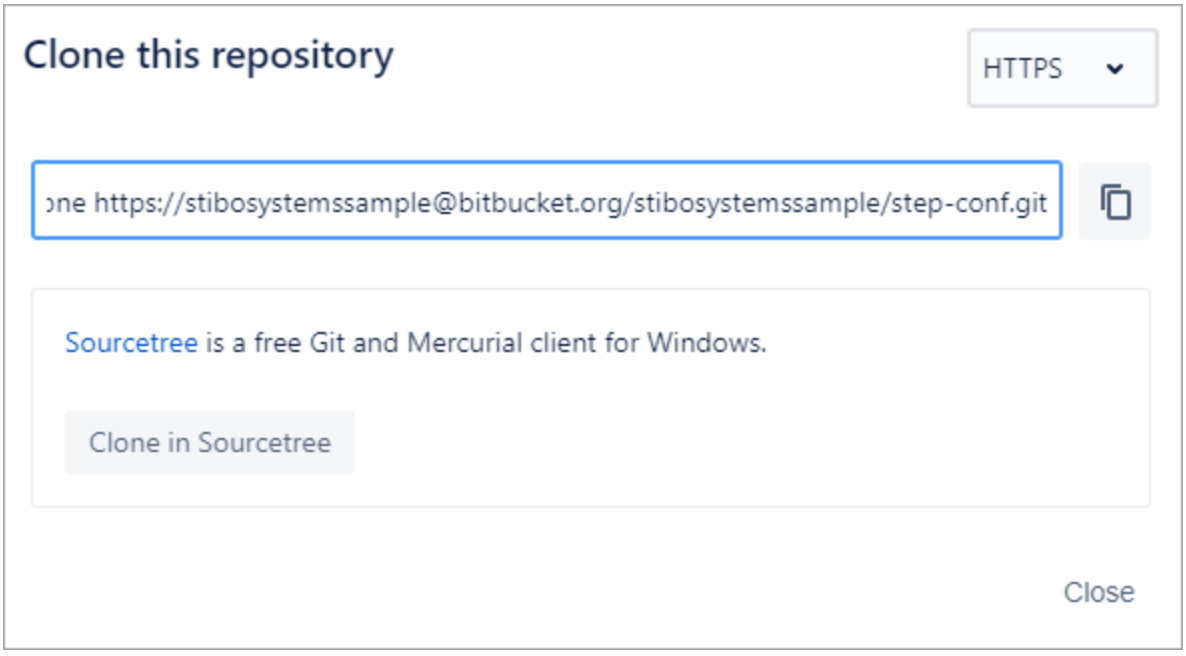
1. Create a new repository with a 'README' file via the Bitbucket web interface.



2. Create a system specific branch in the repository.



3. Get the 'Remote git repository URI' from, for example, the Bitbucket 'Clone' option.



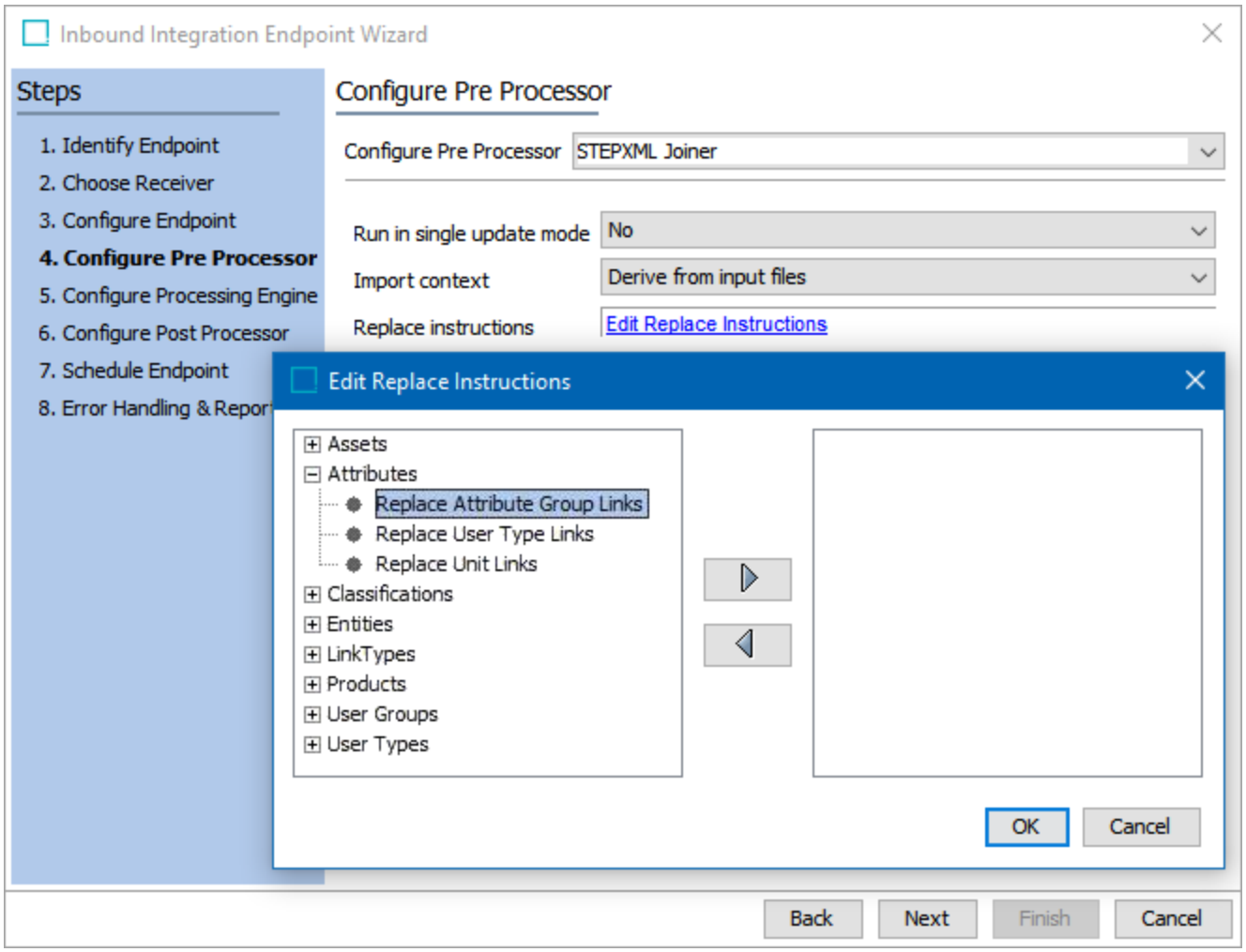
STEPXML Joiner Pre Processor in IIEP

The STEPXML Joiner inbound pre-processor is designed to be used to import configurations and settings previously exported via the outbound functionality (described above). The outbound functionality produces STEPXML files and potentially *.js files representing business rules, and change package metadata when using

the delivery method Change Package Git Delivery. The STEPXML Joiner option can take a .ZIP file containing any number of such files as input and then combines objects and settings in a single STEPXML and passes the combined file to the IIEP processing engine.

Various processing instructions for the combined STEPXML file can be included on the pre-processor. Processing instructions can be added via the UI (shown below) or can be included in a template STEPXML file within the .ZIP file to be processed by the endpoint.

Note: If a template file is provided in the .ZIP, the settings from this file override any UI configurations made in the UI. The template file must be named `ProcessingInstructions.xml` and only the ReplacementRules element should exist inside the STEP-ProductInformation element.



To configure the IIEP:

1. For **Configure PreProcessor**, select STEPXML Joiner.
2. For **Run in single update mode**, set to Yes or No as appropriate.

3. For **Import context**, select the context that will receive the imported data. When using the delivery method Change Package Git Delivery do not use option to 'Derive from input files' because they are not included when the change package is exported.
4. For **Replace instructions**, add replacement rules as needed.

   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, special processing instructions (replacement rules) are used to express that the properties that are not present in the import file are to be removed from the system as part of the import. Click the 'Edit Replace Instructions' link.

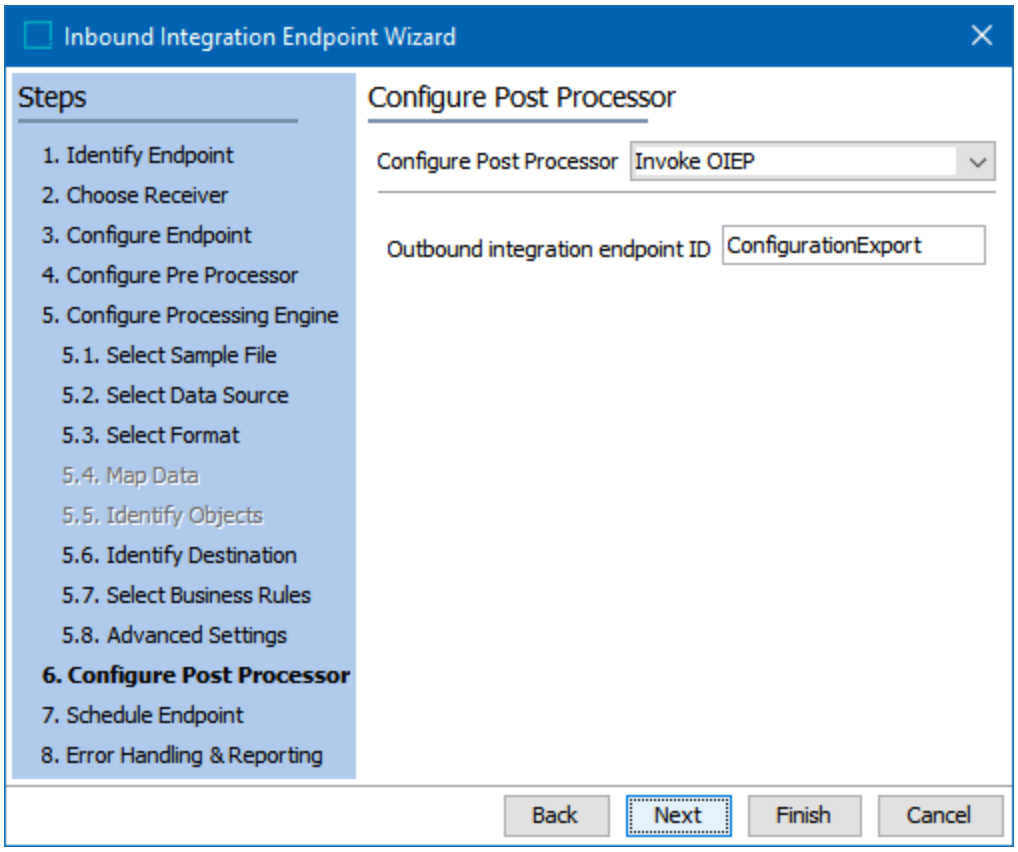
   If you are using replacement rules across contexts with suppressions, include all contexts in an export for best results. This ensures complete results are communicated between systems.

   For more information regarding ReplacementRules, refer to the ReplacementRules Tag in STEPXML topic in the Data Exchange documentation.
5. For the **Edit Replace Instructions** dialog, build the rules by selecting options on the left and using the arrow button to move the rule over to the right. Click OK to save changes before moving to the next step of the Inbound Integration Endpoint Wizard.

Invoke OIEP Post Processor in IIEP

The Invoke OIEP post processor allows for an OIEP to be invoked once the IIEP process has completed. This allows updating the representation of the system configuration in a remote Git branch immediately after the configuration has been imported, if desired.

As shown below, the option requires the ID of the OIEP to be invoked.



For more information, refer to these topics in the Version Control System Integration section:

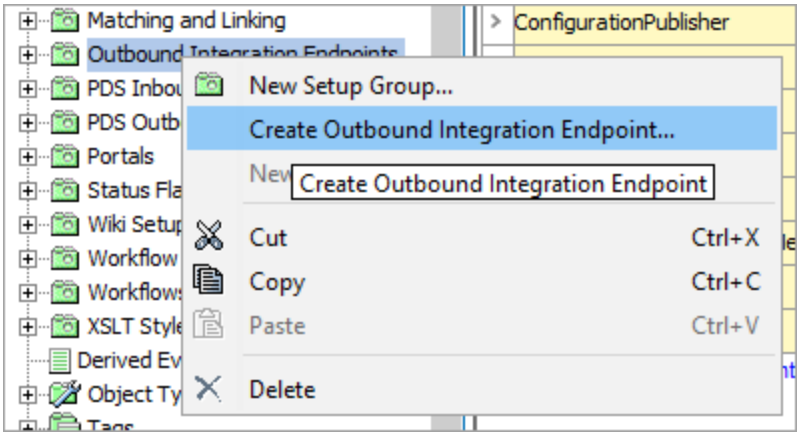
- VCSI: Editable Business Rules Format
- VCSI: Example Setups
- VCSI: Considerations and Limitations

OIEP for VCS Integration with Change Packages

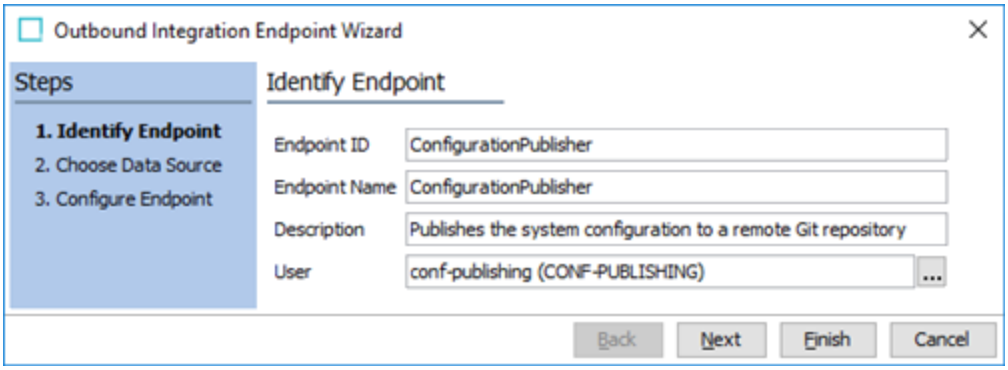
To support Version Control System Integration (VCSI), and integrations between STEP systems, this section describes how to configure an event-based outbound integration endpoint (OIEP) to be used for publishing the system configuration when using change packages and the delivery method: Change Package Git Delivery or REST Direct with a corresponding inbound integration endpoint (IIEP) using the REST Receiver on a target system.

Note: To compare the configurations from multiple STEP systems, the endpoint configurations should be identical except for the Directory Template, or they can be delivered in a separate branch, if desired.

1. For first-time setup of the Change Package Git Delivery option, it is recommended to create and use a test repository containing only a README.md file. This allows you to verify the STEP-specific Git behavior before enabling a production repository.
2. From System Setup, select a setup group configured to hold OIEPs, right-click and select **Create Outbound Integration Endpoint** to launch the Outbound Integration Endpoint Wizard.

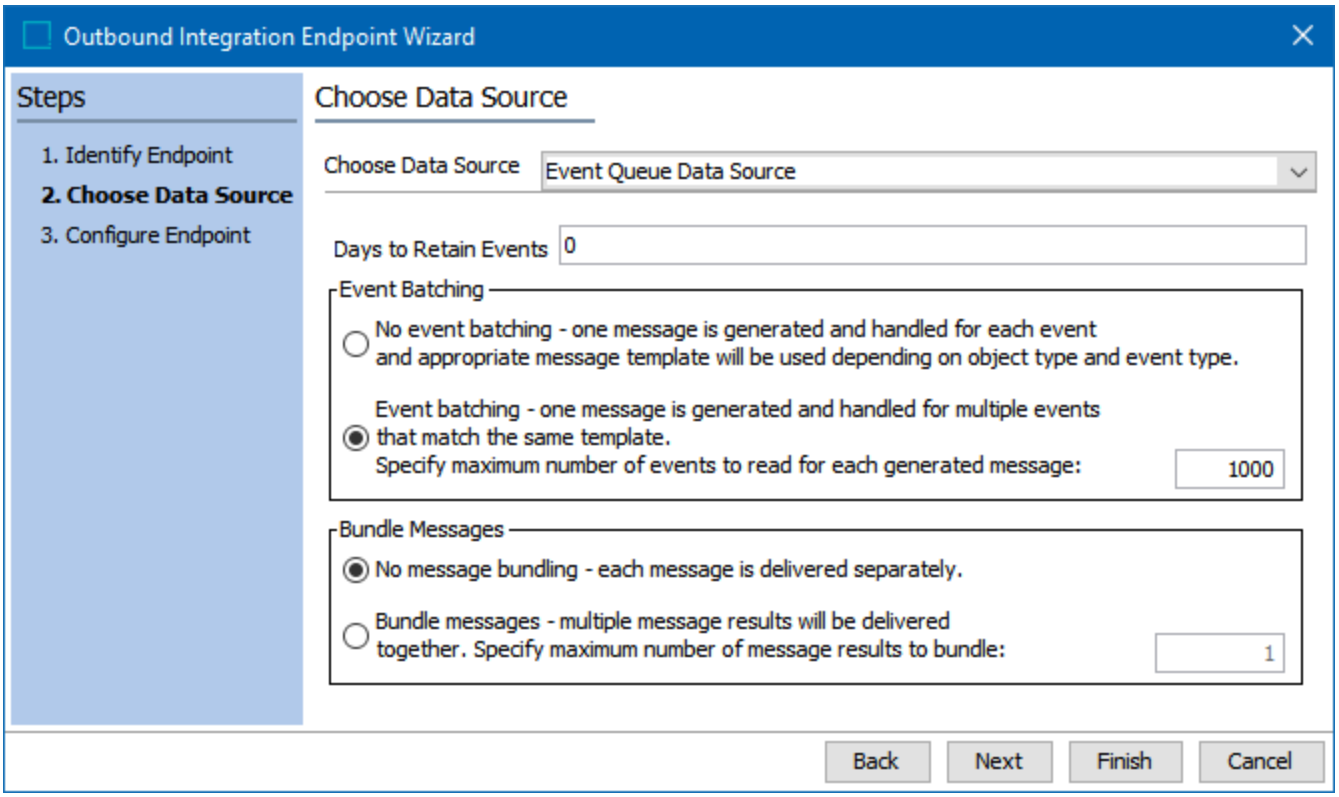


3. In the 'Identify Endpoint' step:



- For **Endpoint ID**, **Endpoint Name**, and **Description**, enter basic information.
- For **User**, select a system user who has view privileges to the configuration objects to be exported.

4. If the Change Package Git Delivery method is to be used, in the 'Choose Data Source' step:



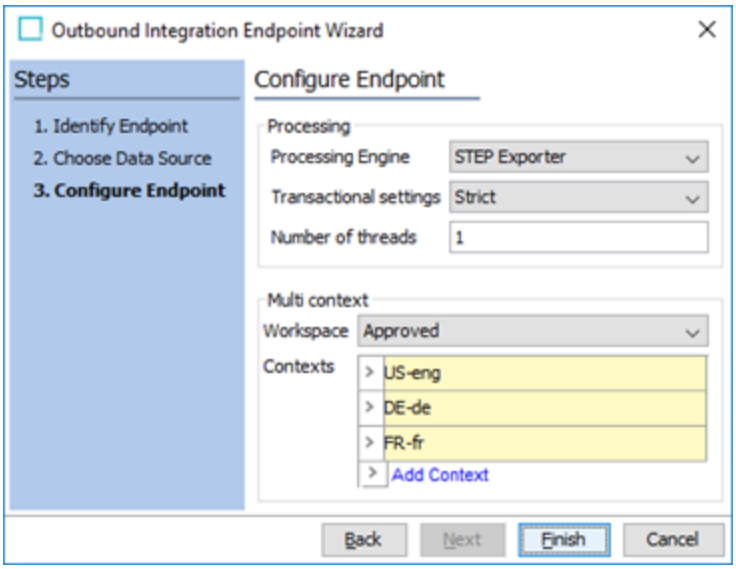
- For **Choose Data Source**, select the 'Event Queue Data Source' option.

> **Note:** Sealing a change package generates an event which can trigger the OIEP when the schedule is running.

- For **Event Batching**, select 'No event batching' to separately export change packages to the VCS.
- For **Bundle Messages**, select 'No message bundling' to separately deliver each message.

> **Note:** Volume of change packages is relatively low compared to other integrations, therefore event batching and message bundling are not necessary.

5. In the 'Configure Endpoint' step:



- For **Processing Engine**, select 'STEP Exporter'.

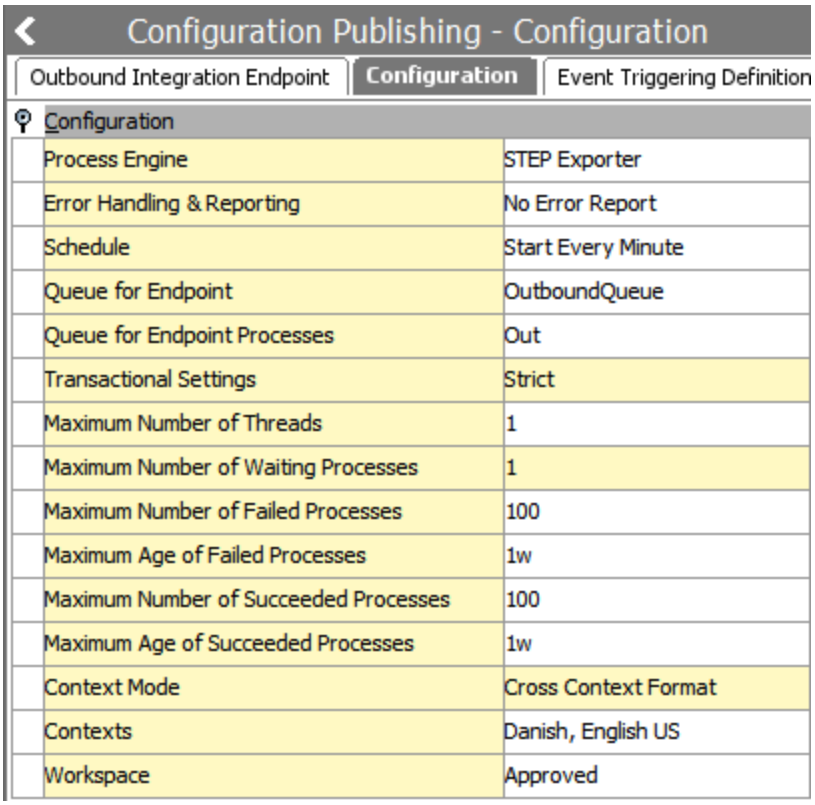
  For information on **Transactional settings** parameter, refer to the Integration Endpoint Transactional Settings topic and for the **Number of threads** parameter, refer to the Event-Based OIEP Multithreading Support topic, both in the Data Exchange documentation.

- For **Workspace**, select the 'Approved' since the approved version of objects are included with the change package when sealed.

- For **Contexts**, if configuration data that is dimension dependent is to be published, select all relevant contexts.

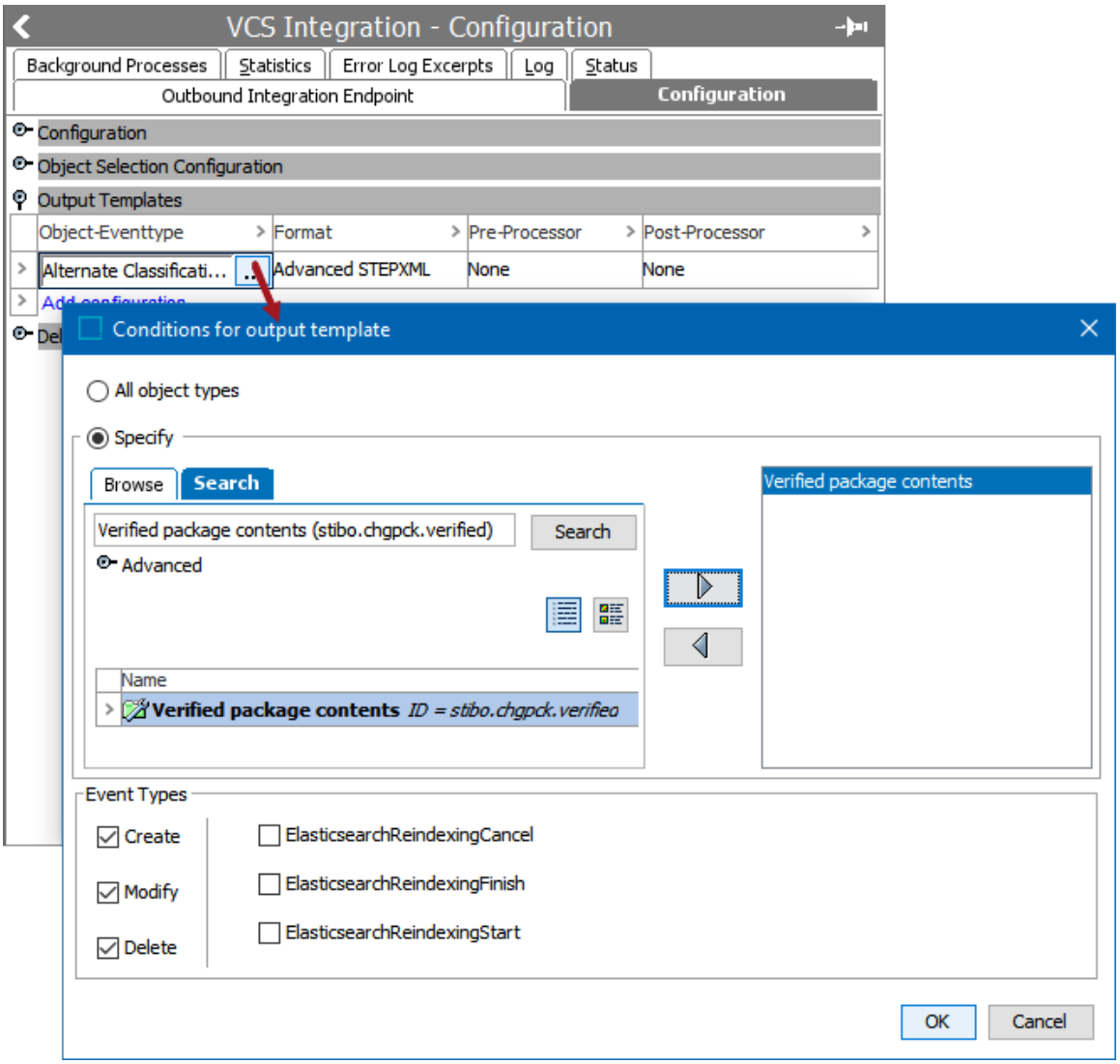
6. Click the **Finish** button to close the wizard.

7. On the 'Configuration' tab of the newly created endpoint, configure the schedule, queue, and process retention settings as desired. For more information, refer to the OIEP - Configuration Flipper topic in the Data Exchange documentation.

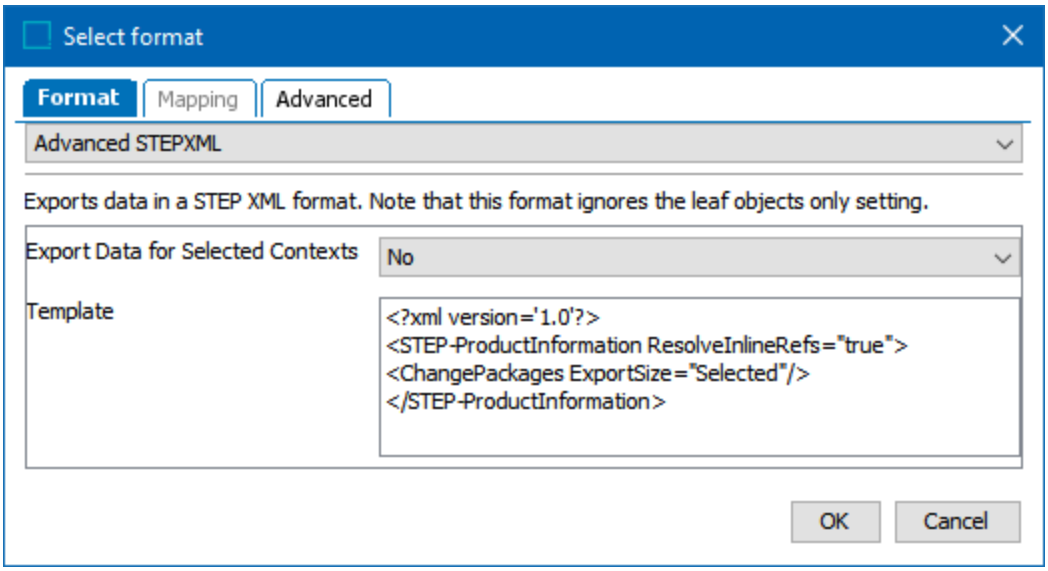


8. In the 'Output Templates' flipper, add a single configuration and search for 'Verified package contents.'

> **Note:** The Change Package Git Delivery method only works with an output template for 'Verified package content'. Since the Change Package object type has three states: Dormant, Open, and Sealed, and only Sealed change packages can be exported, using the 'Change Package' object does not work.



9. For 'Format', the VCSI Change Package option works with 'Advanced STEPXML' format, as defined below with the desired value for Export Data for Selected Contexts.



Add the following Advanced STEPXML template:

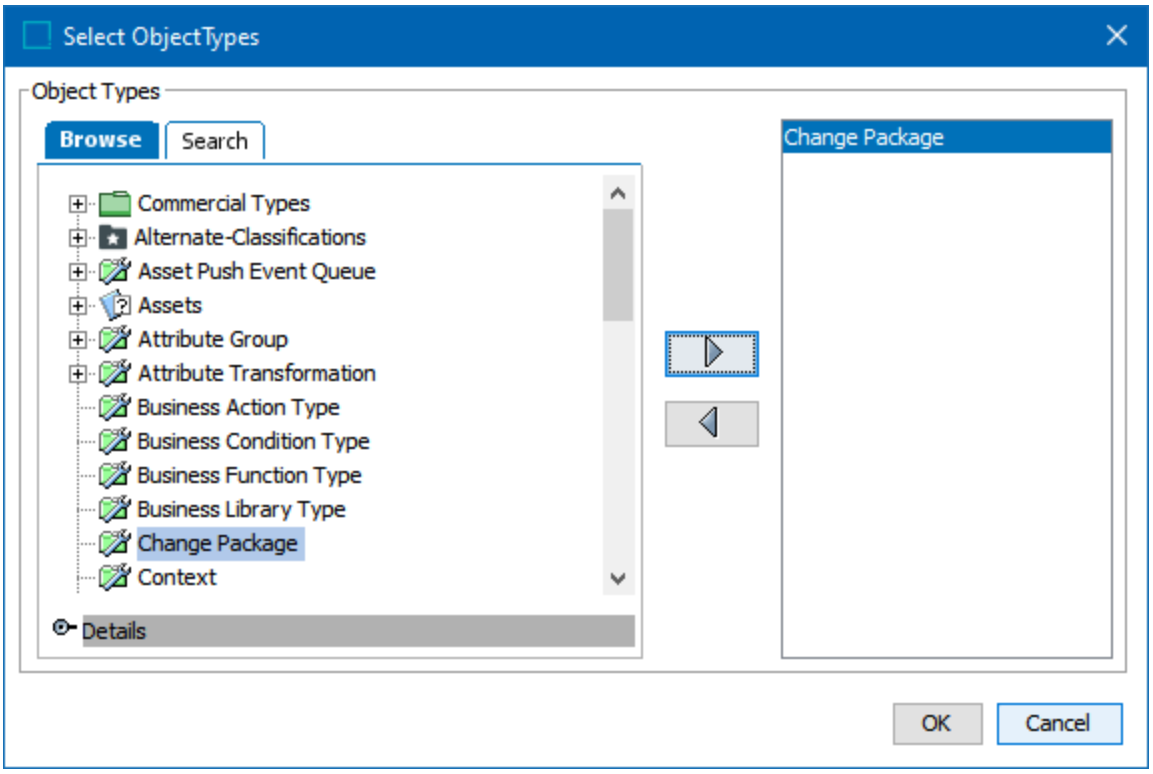
```
<?xml version='1.0'?>
<STEP-ProductInformation ResolveInlineRefs="true">
<ChangePackages ExportSize="Selected"/>
</STEP-ProductInformation>
```

Refer to the OIEP - Event-Based - Output Templates Flipper topic in the Data Exchange documentation.

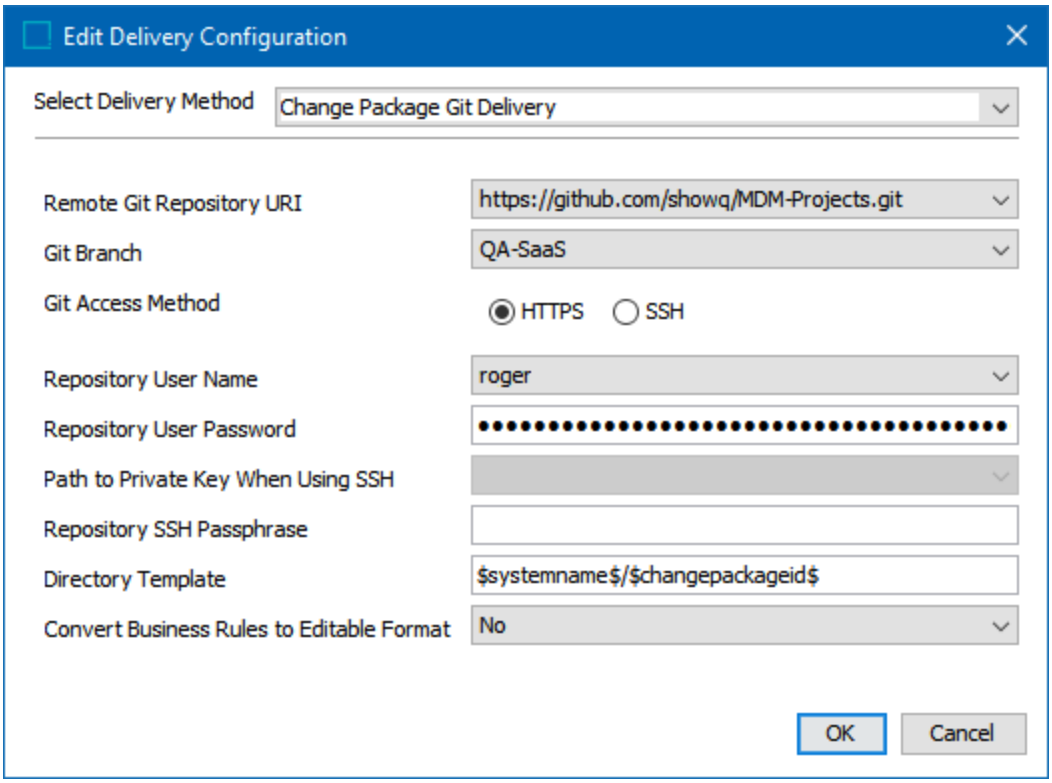
10. On the Event Triggering Definitions tab, set the following options:

    For **Triggering Object Types** flipper, add the 'Change Package Object Type.'

    Since sealing a change package triggers the event, no other event triggers are useful, however, an event filter or generator can be configured if desired.



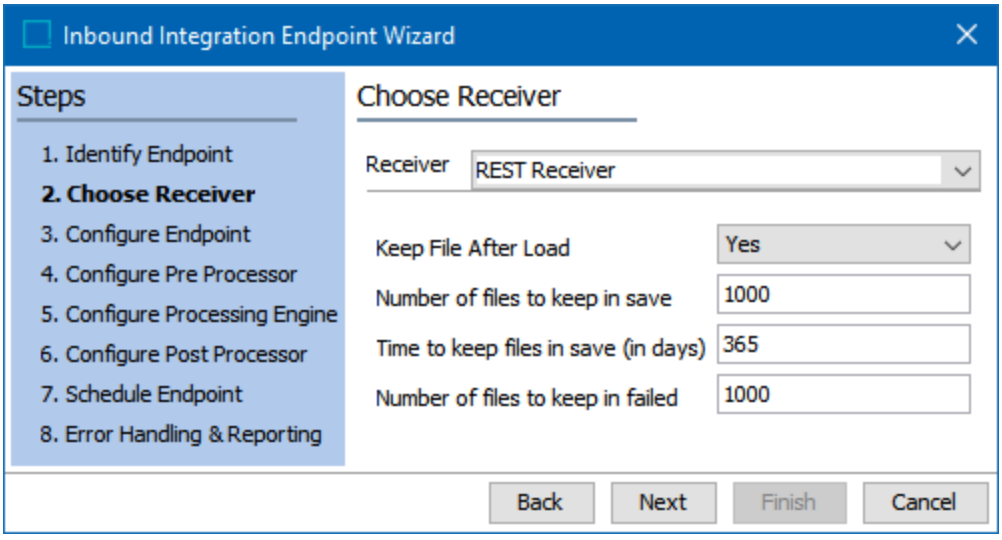
11. Configure the 'Change Package Git Delivery' delivery method. Refer to the 'Change Package Git Delivery Method for OIEP' heading in the Integration Endpoint Options for VCS Integration topic.



12. For integration between STEP systems, choose an integration option:

    - Configure SFTP, Email, or Copy to Directory delivery method with less setup to quickly access files locally.
    - Configure the 'REST Direct' delivery method after creating an IIEP on the target system using the REST Receiver, as defined in the next section.

Configure the 'REST Direct' delivery method with REST Receiver

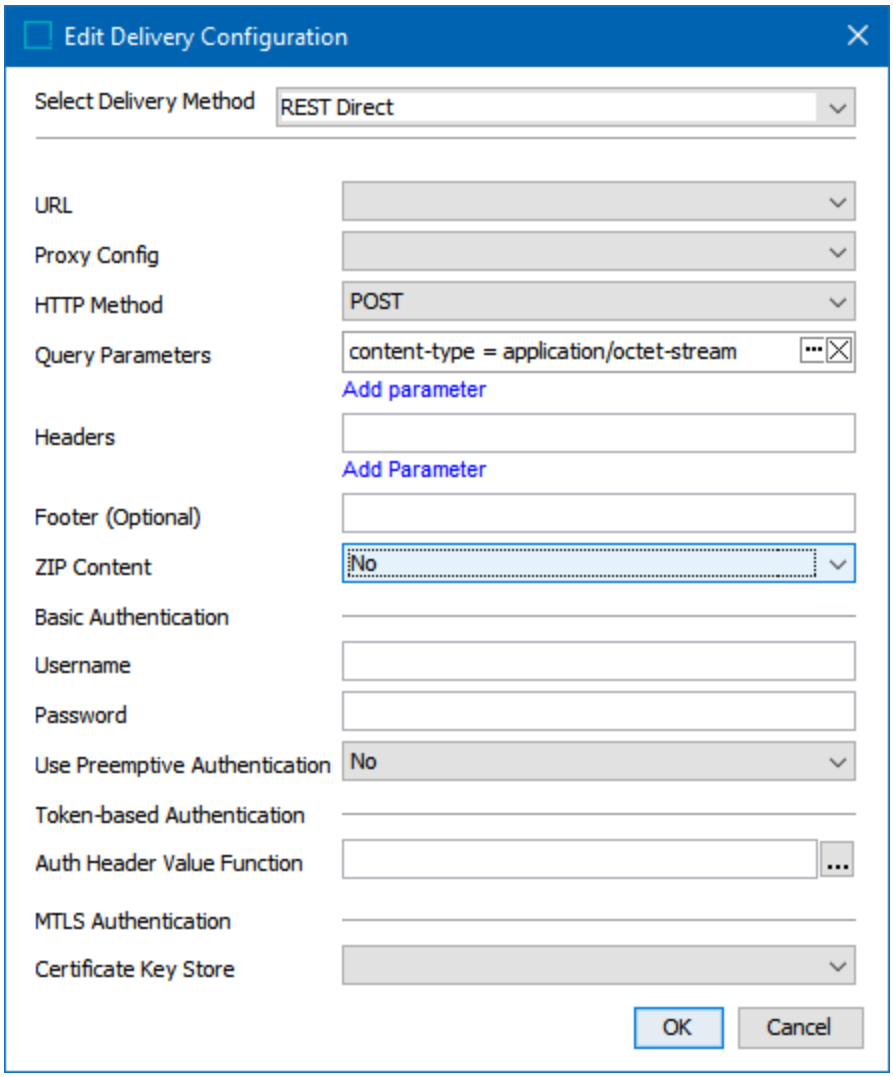


- Identify the ID of the IIEP.
- Configure the `RestDirectDeliveryURL` property to the URL required by REST API V2 to POST the upload and invoke IIEP, as demonstrated in this example.

```
https://[step-hostname]:[step-port]/restapiv2/[iiep-id]/upload-and-invoke?context=[context-id]&workspace=Main
```

Note: Replace the text between the brackets with relevant values from your environments.

- Configure an OIEP with a REST Direct delivery method as follows. Use the shown values as defaults, select an appropriate URL, and include the required authentication options for a user with sufficient privileges.

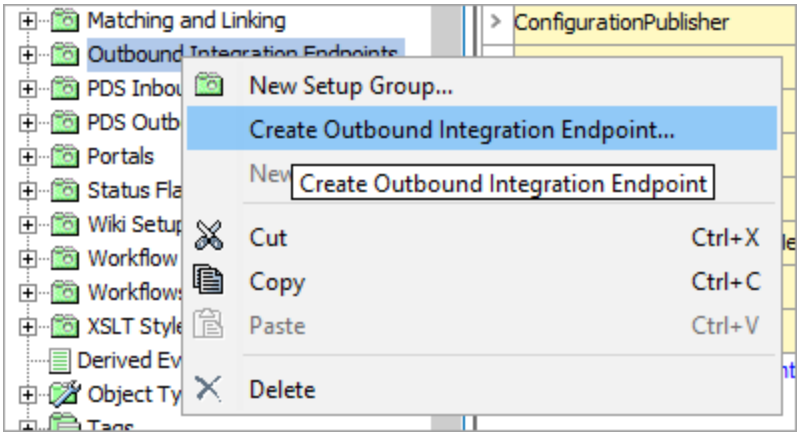


OIEP for VCS Integration with Git Delivery

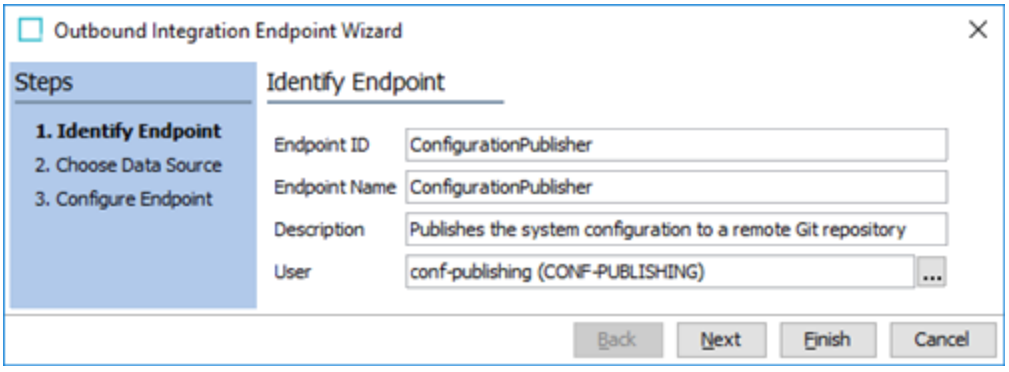
To support Version Control System Integration (VCSI), this section describes how to configure an outbound integration endpoint (OIEP) to be used for publishing the system configuration to a remote Git repository.

Note: To compare the configurations from multiple STEP systems, the endpoint configurations should be identical except for the Git branch 'Git Delivery' method information.

1. For first-time setup of the Git Delivery option, it is recommended to create and use a test repository containing only a README.md file. This allows you to verify the STEP-specific Git behavior before enabling a production repository.
2. From System Setup, select a setup group configured to hold OIEPs, right-click and select **Create Outbound Integration Endpoint** to launch the Outbound Integration Endpoint Wizard.

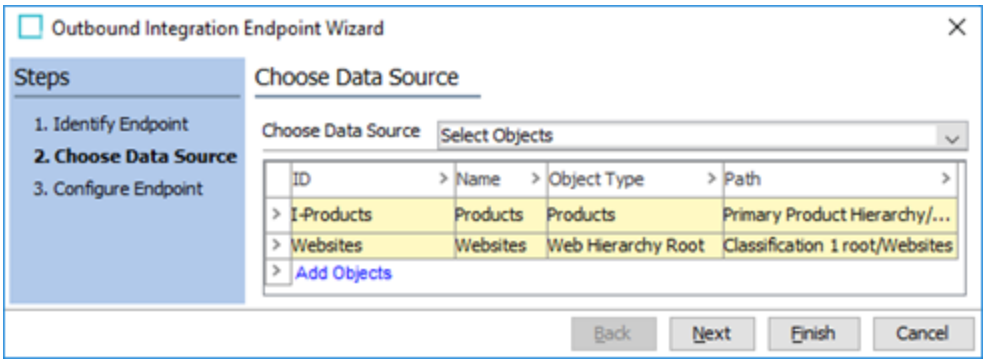


3. In the 'Identify Endpoint' step:



- For **Endpoint ID**, **Endpoint Name**, and **Description**, enter basic information.
- For **User**, select a system user who has view privileges to the configuration objects to be exported.

4. In the 'Choose Data Source' step:



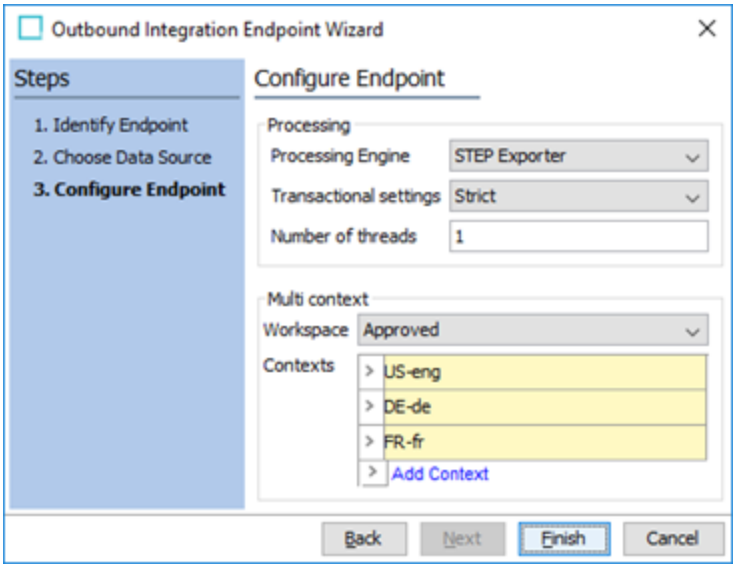
- For **Choose Data Source**, select the 'Select Objects' option.

> **Note:** All configurations and settings to be held in Git must be published each time the OIEP is invoked. Refer to the 'Git Delivery Method in OIEP' heading within the Integration Endpoint Options for VCS Integration topic.

- Click the **Add Objects** link and select the relevant root nodes for the types of data objects to be published.

> **Note:** Even if no product, entity, or classification objects are to be published, a 'dummy' selection must be made, namely, the root nodes.

5. In the 'Configure Endpoint' step:

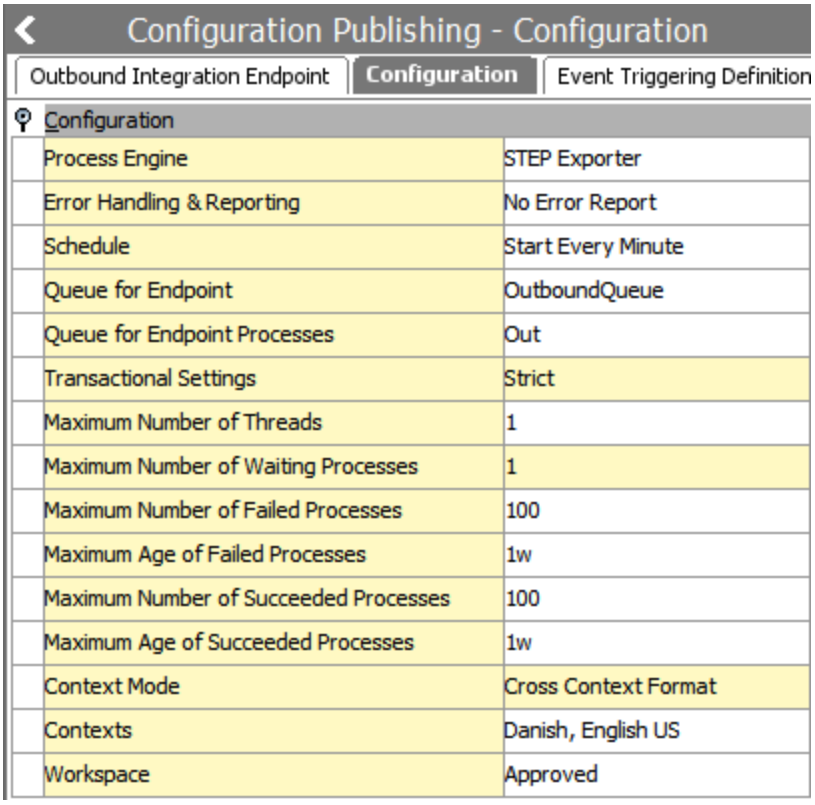


- For **Processing Engine**, select 'STEP Exporter'.

  For information on **Transactional settings** parameter, refer to the Integration Endpoint Transactional Settings topic and for the **Number of threads** parameter, refer to the Event-Based OIEP Multithreading Support topic, both in the Data Exchange documentation.

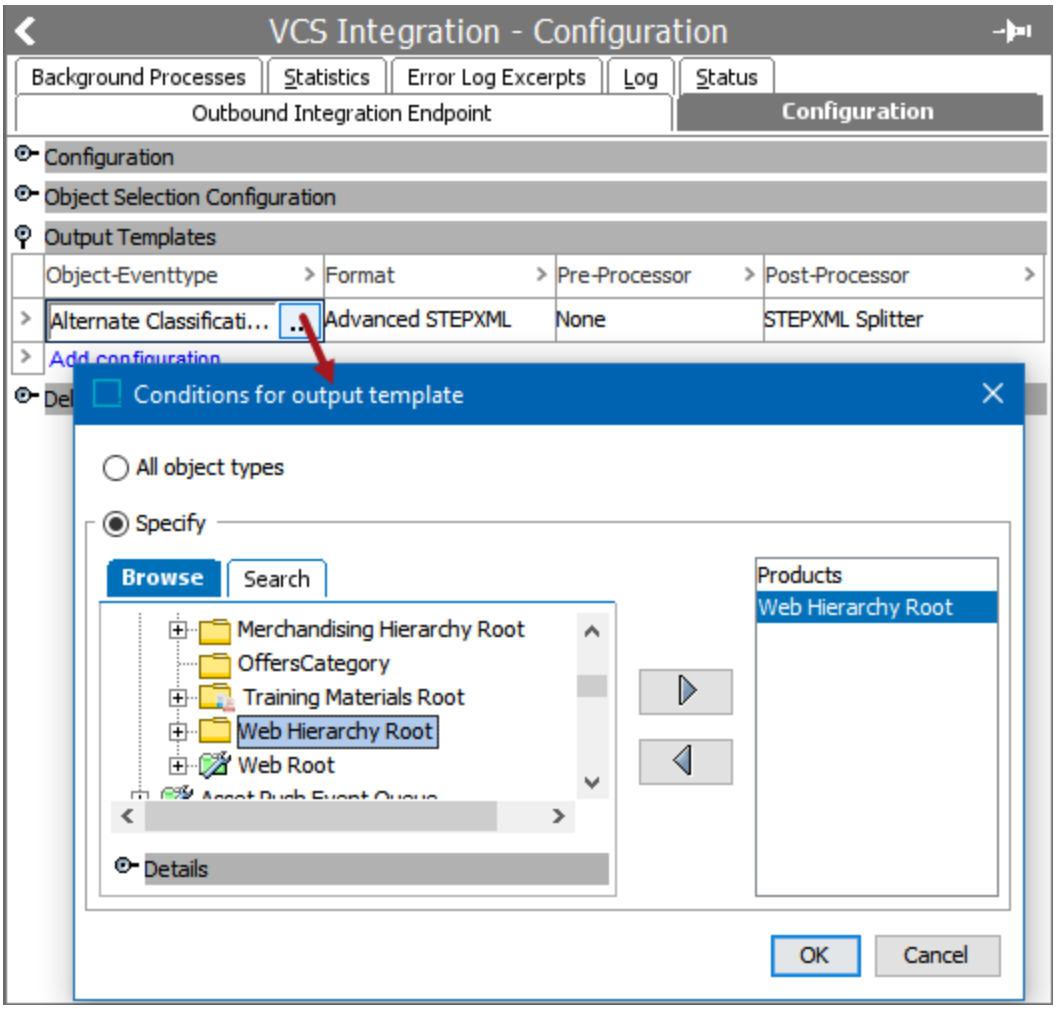
- For **Workspace**, most configuration objects are not workspace revised. If data objects like products, entities, and classifications are to be published, in most cases, it is the approved version of these objects that should be exported, so select the 'Approved' workspace.
- For **Contexts**, if configuration data that is dimension dependent is to be published, select all relevant contexts.

6. Click the **Finish** button to close the wizard.
7. On the 'Configuration' tab of the newly created endpoint, configure the schedule, queue, and process retention settings as desired. For more information, refer to the OIEP - Configuration Flipper topic in the Data Exchange documentation.



8. In the 'Output Templates' flipper, add a single configuration and select the object types of the nodes selected for publishing, including configurations with 'dummy' (root node) selections.

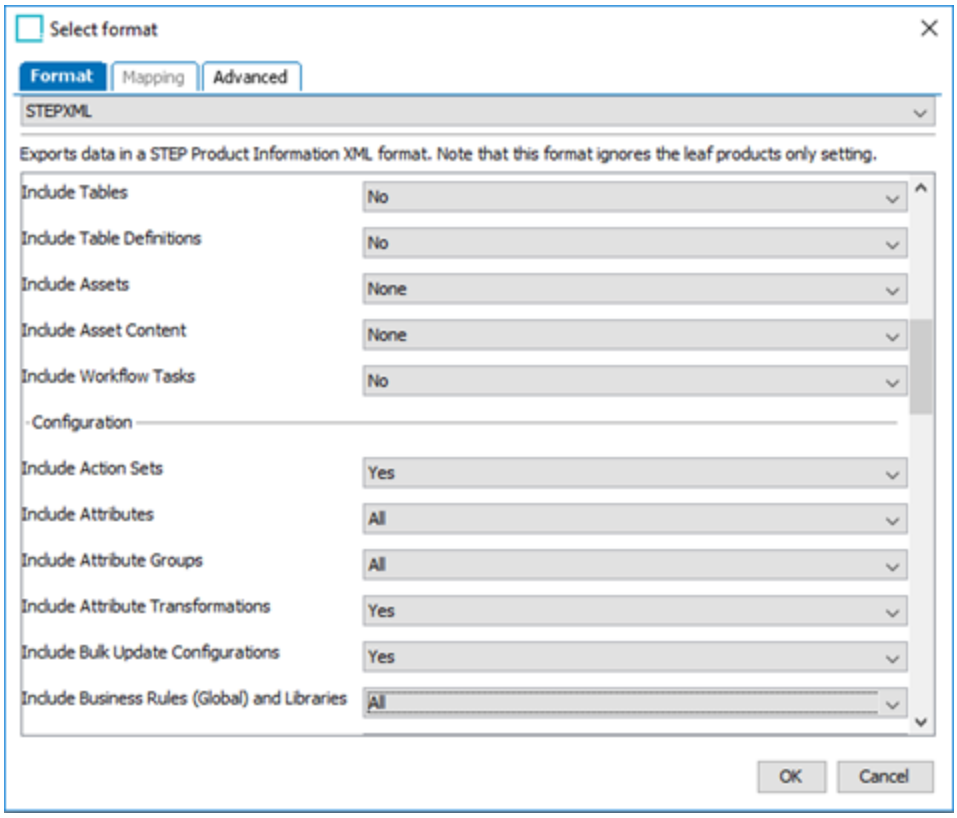
> **Note:** The Git delivery method only works with a single output template.



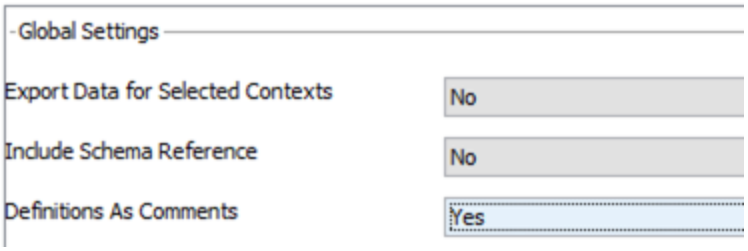
9. For 'Format', the VCSI options work with either the 'STEPXML' format or the 'Advanced STEPXML' format with object type filtering, as defined below.

   When using **STEPXML**:

   - Select 'Yes' or 'All' for each configuration type to publish.
   - Select 'Minimum' for products, entities, and classifications (if these are to be published).
   - Select 'No' or 'None' for all types that should not be published.

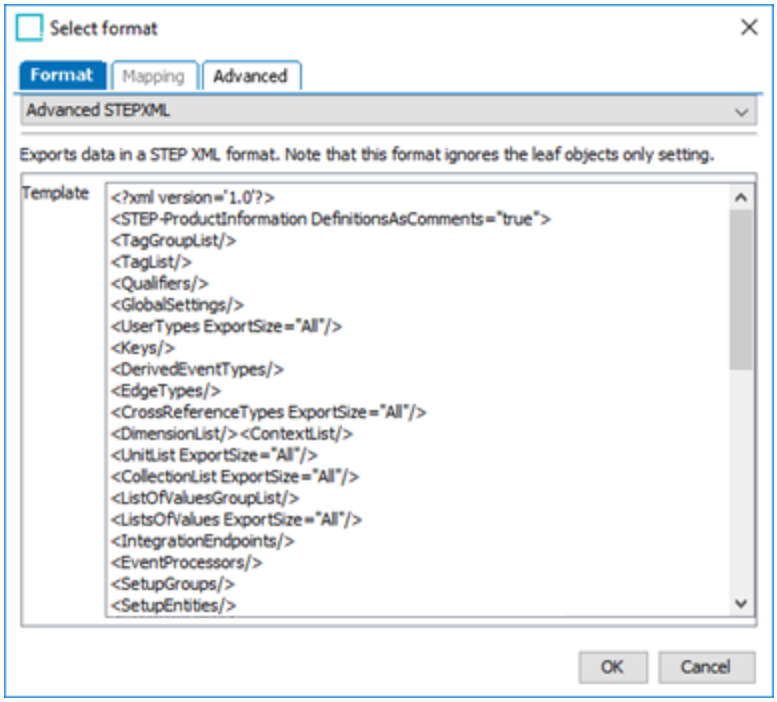


- To make obfuscated configurations (such as business rules) comparable outside STEP, on the 'Definitions As Comments' parameter, select 'Yes'.



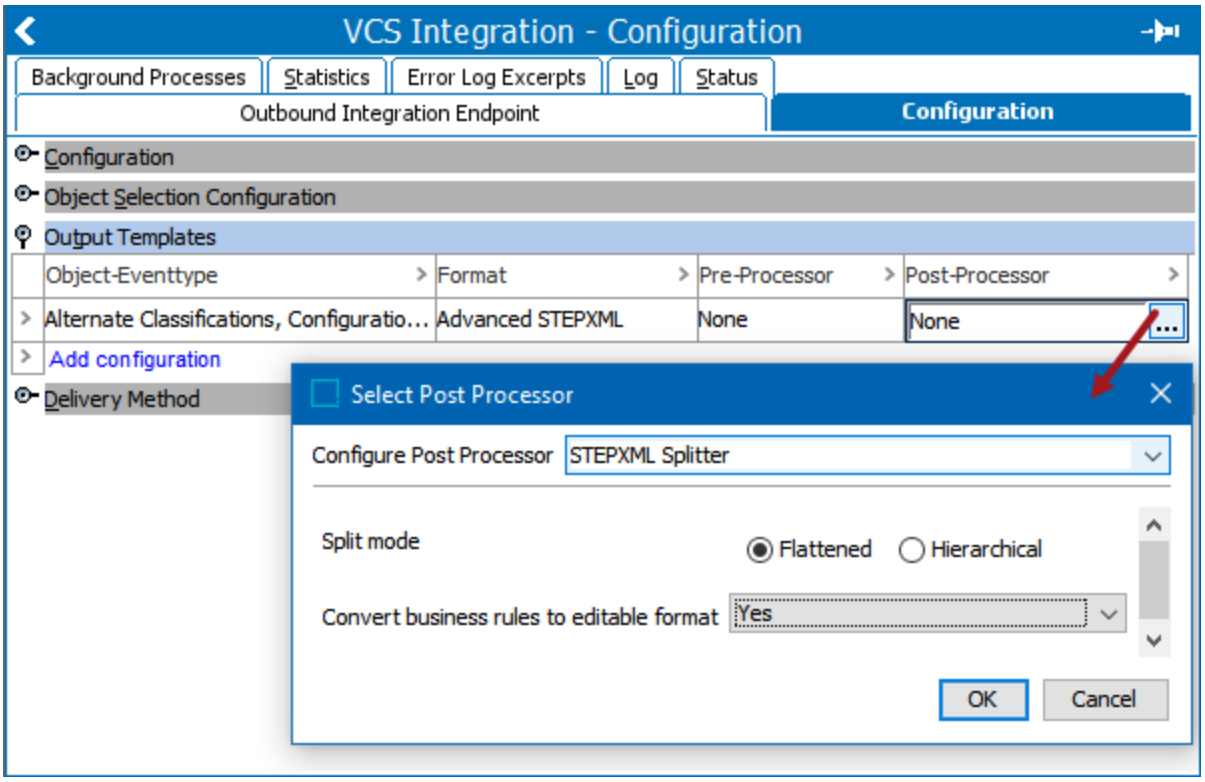
When using **Advanced STEPXML**, data objects are published and filtered by object type.

- Enter a template and set the STEP-ProductInformation tag 'DefinitionsAsComments' attribute to 'true'.

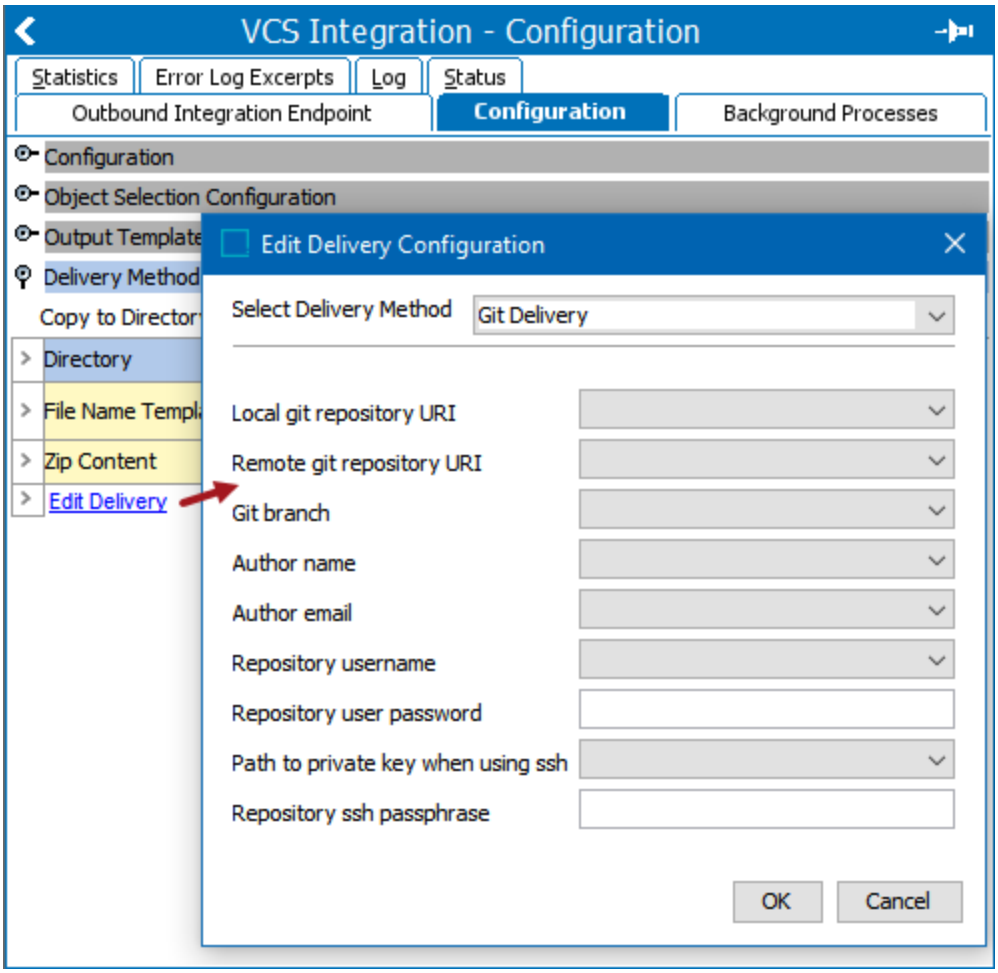


  A full example configuration with object type filtering is included in this topic in online help.

10. Configure the 'STEPXML Splitter' post-processor. For more information, refer to the 'STEPXML Splitter Post-processor for OIEP' heading in the Integration Endpoint Options for VCS Integration topic.



11. Configure the 'Git Delivery' delivery method. Refer to the 'Git Delivery Method for OIEP' heading in the Integration Endpoint Options for VCS Integration topic.


VCS Integration - Configuration
Statistics
Error Log Excerpts
Log
Status
Outbound Integration Endpoint
Configuration
Background Processes
Configuration
Object Selection Configuration
Output Template
Delivery Method
Copy to Director
Directory
File Name Templ
Zip Content
Edit Delivery
Edit Delivery Configuration
Select Delivery Method
Git Delivery
Local git repository URI
Remote git repository URI
Git branch
Author name
Author email
Repository username
Repository user password
Path to private key when using ssh
Repository ssh passphrase
OK
Cancel

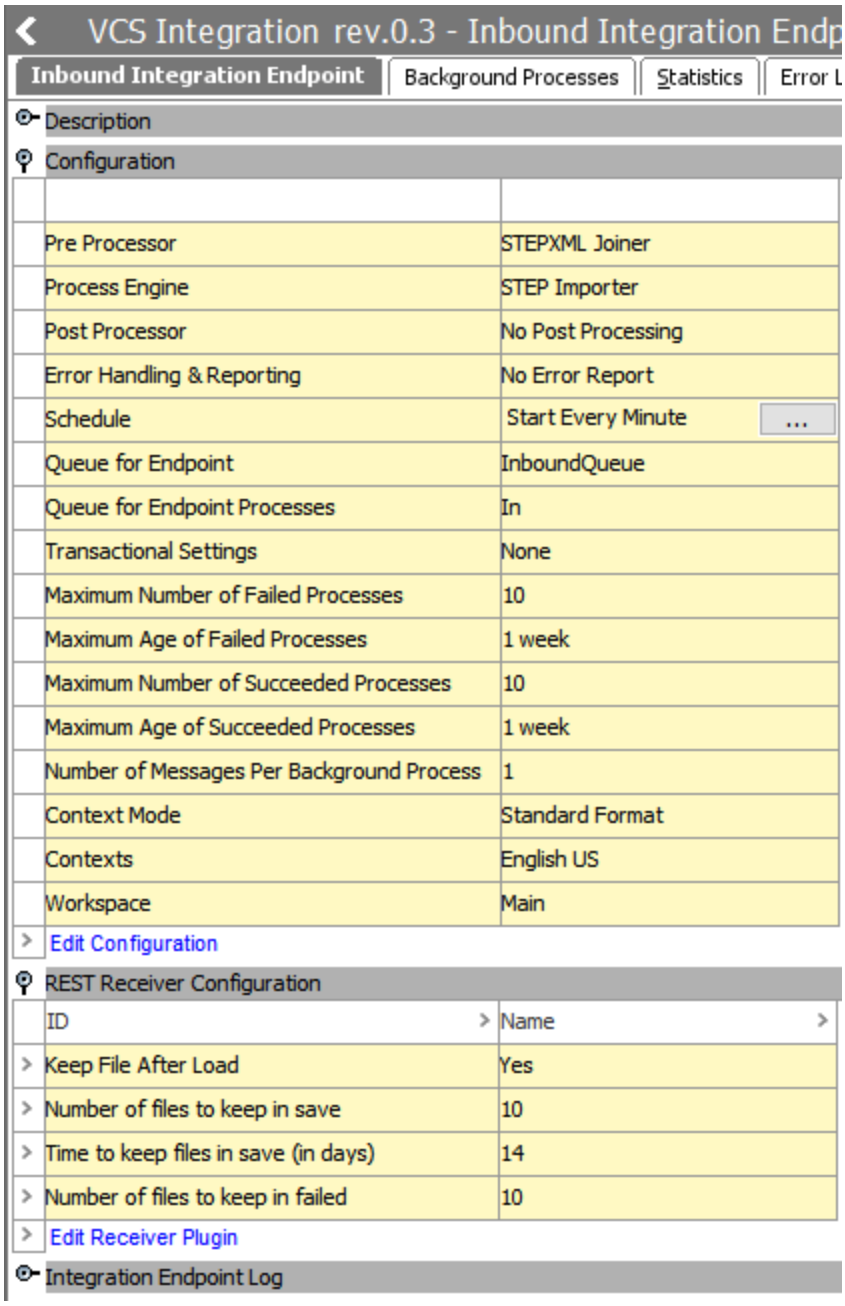
IIEP for VCS Integration

The inbound STEPXML Joiner pre-processor and the Invoke OIEP post-processor are configured via the Inbound Integration Endpoint Wizard. Open the wizard by selecting 'Create Inbound Integration Endpoint' in the context menu for a setup group configured to hold inbound integration endpoints (IIEPs).

For details, refer to the 'STEPXML Joiner Pre-processor for IIEP' section and the 'Invoke OIEP Post-processor for IIEP' section of the Integration Endpoint Options for VCS Integration topic.

The endpoint must be configured to import STEPXML (by providing any valid STEPXML file as a sample file) and can be configured with any receiver method capable of handling .ZIP files, for example, the 'Hotfolder Receiver' or the 'REST Receiver'. If workspace revisable objects like products and classifications are imported, set if import changes should be automatically approved.

Consider the example configuration in the image below:


VCS Integration rev.0.3 - Inbound Integration Endp
Inbound Integration Endpoint
Background Processes
Statistics
Error L
Description
Configuration
Pre Processor
STEPXML Joiner
Process Engine
STEP Importer
Post Processor
No Post Processing
Error Handling & Reporting
No Error Report
Schedule
Start Every Minute
...
Queue for Endpoint
InboundQueue
Queue for Endpoint Processes
In
Transactional Settings
None
Maximum Number of Failed Processes
10
Maximum Age of Failed Processes
1 week
Maximum Number of Succeeded Processes
10
Maximum Age of Succeeded Processes
1 week
Number of Messages Per Background Process
1
Context Mode
Standard Format
Contexts
English US
Workspace
Main
Edit Configuration
REST Receiver Configuration
ID
Name
Keep File After Load
Yes
Number of files to keep in save
10
Time to keep files in save (in days)
14
Number of files to keep in failed
10
Edit Receiver Plugin
Integration Endpoint Log

VCSI: Editable Business Rules Format

To support the Version Control System Integration (VCSI), JavaScript-based business rules can be created, maintained, and tested outside STEP. This allows customers and partners to govern the lifecycle of business rules in a standard source code control system such as Git, and from there, be able to deploy appropriate versions of the business rules to the various STEP systems that are part of a Development, Testing, Acceptance, and Production (DTAP) environment.

This topic describes the following in detail:

- Editable Business Rule Format
- Options for Export
- Options for Import
- REST Resources for Testing and Validation

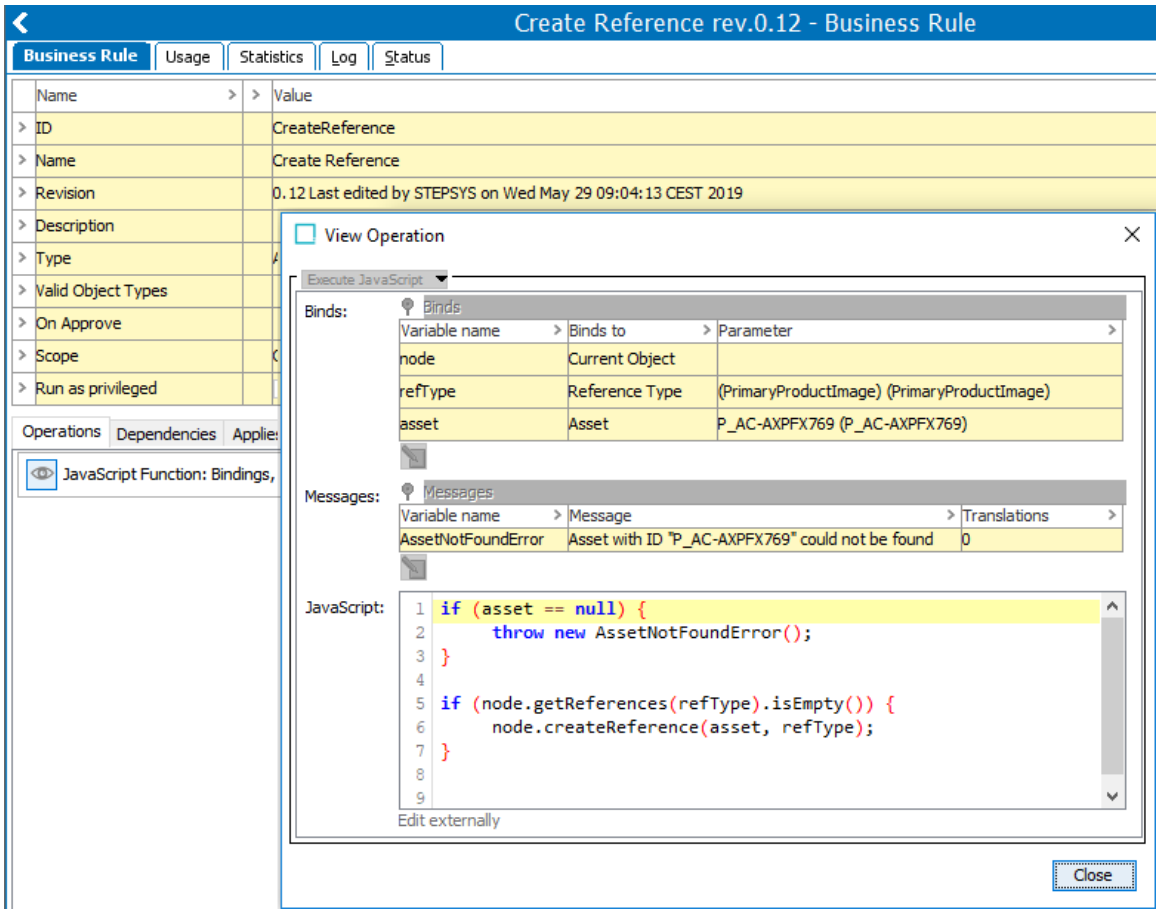
Note: For on-premises systems, this feature requires the 'configuration-management' add-on component.

Editable Business Rule Format

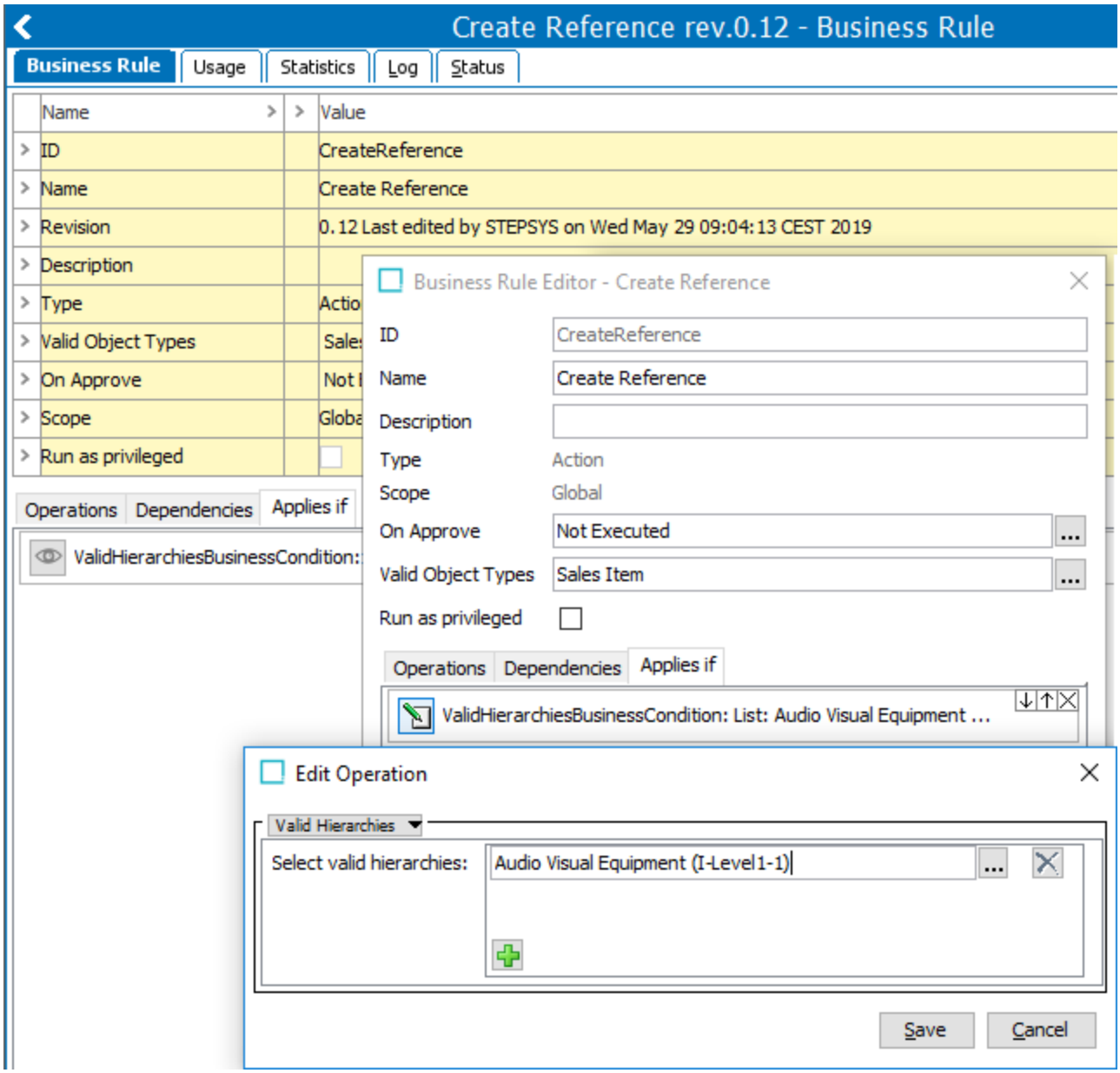
Business rules can be exported as *.js files that can be edited outside STEP and imported back into a STEP system, creating, or updating a business rule. This format is available for business conditions, actions, functions, and libraries of 'Global' scope created using the business rule format introduced with STEP 7.0. Each file represents a single business rule and contains all information necessary to create / update the rule on import.

In the *.js files, metadata and definitions of non-JavaScript operations and preconditions ('Applies if') is output in comment sections, while the JavaScript for the individual operations and preconditions are wrapped in functions with objects provided by the execution context as parameters (binds, messages, function input parameters, and referenced libraries).

For example, consider a simple business action with two operations - one JavaScript operation as shown below:



and also, the following non-JavaScript precondition:



When exported using the default settings, the business actions are represented in the generated file as follows:

Note: When exporting business rules in editable format using a change package, the export contextId and workspaceId are not populated. When editing the business rule, also specify the intended context and workspace for import.

```
/*===== export metadata =====
{
  "contextId" : "Context1",
  "workspaceId" : "Main"
```

```
}
*/
/*===== business rule definition =====
{
  "id" : "CreateReference",
  "type" : "BusinessAction",
  "setupGroups" : [ "Actions" ],
  "name" : "Create Reference",
  "description" : null,
  "scope" : "Global",
  "validObjectTypes" : [ "SalesItem" ],
  "allObjectTypesValid" : false,
  "runPrivileged" : false,
  "onApprove" : "Never",
  "dependencies" : [ ]
}
*/
/*===== business rule plugin definition =====
{
  "pluginId" : "JavaScriptBusinessActionWithBinds",
  "binds" : [ {
    "contract" : "CurrentObjectBindContract",
    "alias" : "node",
    "parameterClass" : "null",
    "value" : null,
    "description" : null
  }, {
    "contract" : "ReferenceTypeBindContract",
    "alias" : "refType",
    "parameterClass" : "com.stibo.core.domain.impl.ReferenceTypeImpl",
    "value" : "PrimaryProductImage",
    "description" : null
  }, {
    "contract" : "AssetBindContract",
    "alias" : "asset",
    "parameterClass" : "com.stibo.core.domain.impl.FrontAssetImpl$$Generated$$7",
    "value" : "P_AC-AXPFX769",
    "description" : null
  } ],
  "messages" : [ {
    "variable" : "AssetNotFoundError",
    "message" : "Asset with ID \"P_AC-AXPFX769\" could not be found",
    "translations" : [ ]
  } ],
  "pluginType" : "Operation"
}
*/
```

```
exports.operation1 = function (node, refType, asset, AssetNotFoundError) {
  if (asset == null) {
         throw new AssetNotFoundError();
  }

  if (node.getReferences(refType).isEmpty()) {
         node.createReference(asset, refType);
  }
}
/*===== business rule plugin definition =====
{
  "pluginId" : "ValidHierarchiesBusinessCondition",
  "parameters" : [ {
    "id" : "HierarchyRoots",
    "type" : "java.util.List",
    "values" : [ "step://product?id=I-Level1-1" ]
  } ],
  "pluginType" : "Precondition"
}
*/
```

The logic of the JavaScript operation is wrapped in a function. In the example, this function is exported in line with the Node.js module system convention. The case-sensitive property `ConfigurationManagement.BusinessRuleConverter.ExportFormat` in the sharedconfig.properties file on the application server can be used to change this. The valid values of this property are:

- 'NodeExport' (default; Node.js module system)
- 'EcmaScriptExport' (ECMAScript module system compliant format)
- 'NoExport' (functions not exported)

The format for business libraries differs as a library in STEP already holds a number of JavaScript functions that can be called from other business rules. To make these functions available to other modules, the functions are exported when the 'NodeExport' or 'EcmaScriptExport' settings are used.

For a JavaScript library example, refer to this topic in online help.

When exported with the `ConfigurationManagement.BusinessRuleConverter.ExportFormat` property set to the default 'NodeExport' value, the following is appended to the file, allowing you to require / import the functions from another Node.js module. Everything below, including the comment, is ignored when the library file is imported in STEP.

```
/*===== business library exports - this part will not be imported to STEP =====*/
exports.isProductBelow = isProductBelow
exports.isProduct = isProduct
```

Important: While it is possible in STEP to call functions in other business libraries from within a library function, this functionality is not supported when calling the exported library functions from another module.

To allow library functions that call functions in other referenced libraries to be executable outside STEP, these can be modified so that you can pass the library as a parameter. For example, assume that there is a library function like the one that follows:

```
//"lib" is alias for a referenced library with a function getUpc()
function setUpc(node, attributeId) {
  node.getValue(attributeId).setSimpleValue(lib.getUpc());
}
```

This function can be modified as shown below, allowing it to pass the library as a parameter when invoking the function outside STEP.

```
function setUpc(node, attributeId, passedLib) {
  if (lib == null) {
    lib = passedLib;
  }
  node.getValue(attributeId).setSimpleValue(lib.getUpc());
```

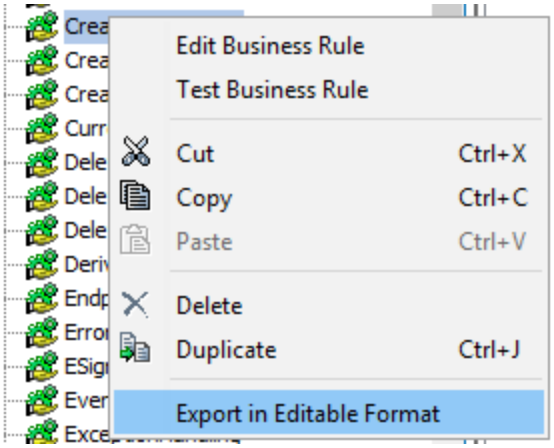
Note: Adding an extra optional parameter does not require that the JavaScript calling the function be modified.

Options for Export

Business rules can be exported to the editable format manually or via an outbound integration endpoint (OIEP).

Manual export

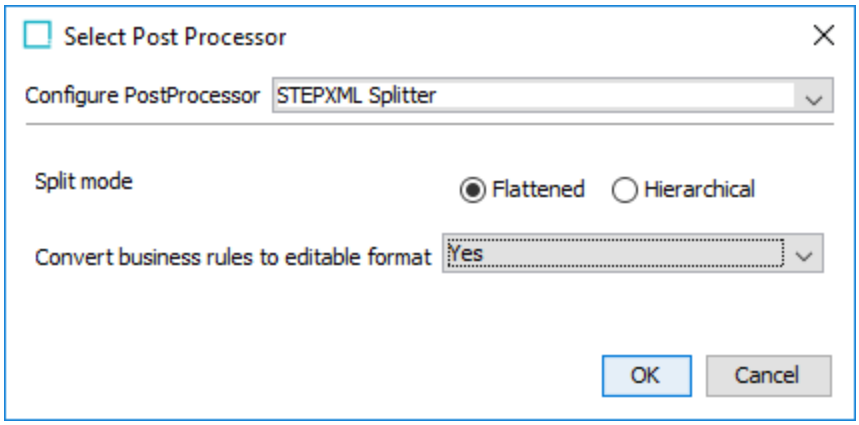
To manually export business rules individually, from a System Setup node that houses business rules, right-click and select the 'Export in Editable Format' option as shown below:



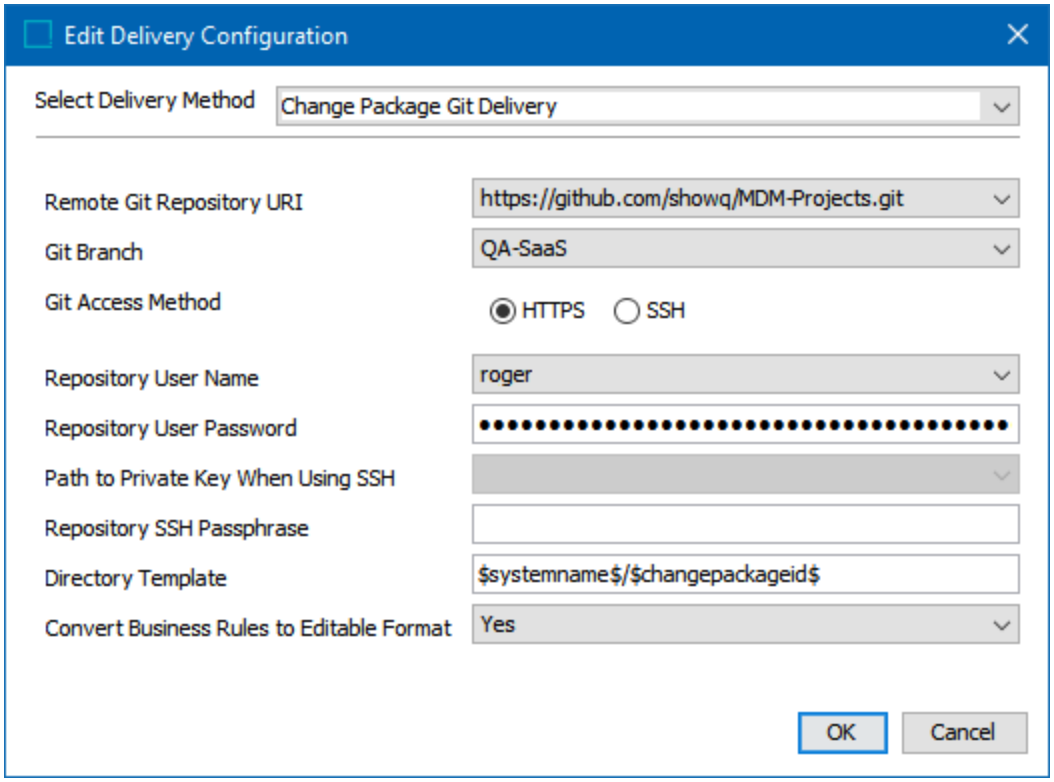
Outbound Integration Endpoint

When 'Convert business rules to editable format' is set to Yes, the postprocessor converts business rules in the STEPXML to the editable format and represents them in a single *.js file instead of representing them in a STEPXML file.

As shown below, set the 'Convert business rules to editable format' parameter to Yes when using the configuration management STEPXML Splitter post-processor for OIEPs:



or when using the Change Package Git Delivery method on an OIEP, where the STEPXML splitter is integrated:



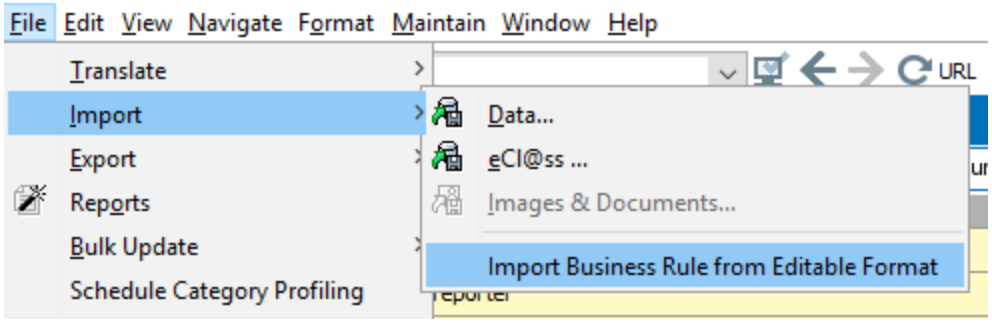
For details, refer to the 'STEPXML Splitter Post-processor in OIEP' section of the Integration Endpoint Options for VCS Integration topic.

Options for Import

Business rules can be imported manually or via an inbound integration endpoint (IIEP).

Manual import

To import a single business rule manually, from the File menu, select Import and then Import Business Rule from Editable Format, as shown below.



Inbound Integration Endpoint

For importing multiple business rules via an inbound integration endpoint (IIEP), the configuration management STEPXML Joiner preprocessor can be used. The preprocesser accepts a .ZIP file containing STEPXML files as input as well as business rule *.js files that the preprocessor will convert to STEPXML and merge into the STEPXML file delivered to the import part of the processing.

Note: When exporting business rules in editable format using Change Package Git Delivery, the export contextId and workspaceId are not populated. When editing the business rule, also specify the intended context and workspace for import.

For details, refer to the 'STEPXML Joiner Pre-processor for IIEP' section of the Integration Endpoint Options for VCS Integration topic.

Note: When importing a business rule, all valid object types included in the import will be added to the existing list of valid object types for that rule.

REST Resources for Testing and Validation

Available REST resource operations allow:

- Testing JavaScript on a running STEP server.
- Validating the syntax of a business rule in the editable format on a STEP server.

The REST resource for testing JavaScript is available at `http(s)://[step-hostname]:[step-port]/configuration-management/test-javascript?context=[context-id]&workspace=[workspace-id]` and allows clients to execute ECMAScript 5-compliant JavaScript on a running STEP server in a non-committing mode with access to a STEP Manager that gives access to the standard STEP Scripting API.

As an example, POSTing the function shown below to `https://[step server]/configuration-management/test-javascript?context=Context1&workspace=Main` returns "Context1":

```
function getContextId(manager) {
       return manager.getCurrentContext().getID();
}
getContextId(manager);
```

The resource for validating a business rule definition in the editable format is available at `http(s)://[step-hostname]:[step-port]/configuration-management/validate-business-rule`. The resource allows clients to POST a complete business rule definition and validates the business rule in these steps:

1. Model validation - validates the overall structure and determines if the business rule metadata is correct (syntax check only).
2. Domain validation - validates existence of the operation and precondition option and checks if the correct parameters have been supplied (values are not checked).
3. Conversion validation - validates if the business rule definition can successfully be converted to STEPXML.

The resource returns a Boolean indicating if the business rule is valid and includes a list of any encountered errors.

Example response:

```
{
    "valid": false,
    "errors": [
        "'businessRuleDefinition.id': may not be null"
    ]
}
```

These REST resources:

- Use basic authentication and the user invoking the resources must have a privilege that includes the 'Test JavaScript' setup action.
- Require the property `ConfigurationManagement.TestJavascript.Enabled` (which defaults to 'false') in the sharedconfig.properties file to be set to 'true' on systems to be used for tests and validation.

Note: Documentation and an example step.js Node.js module that wraps the REST resources and can be used together is available from the STEP API documentation accessible at [system]/sdk or from the system Start Page.

Additional VCSI information can be found in the following topics:

- OIEP for VCS Integration with Change Packages
- OIEP for VCS Integration with Git Delivery
- IIEP for VCS Integration
- VCSI: Example Setups
- VCSI: Considerations and Limitations

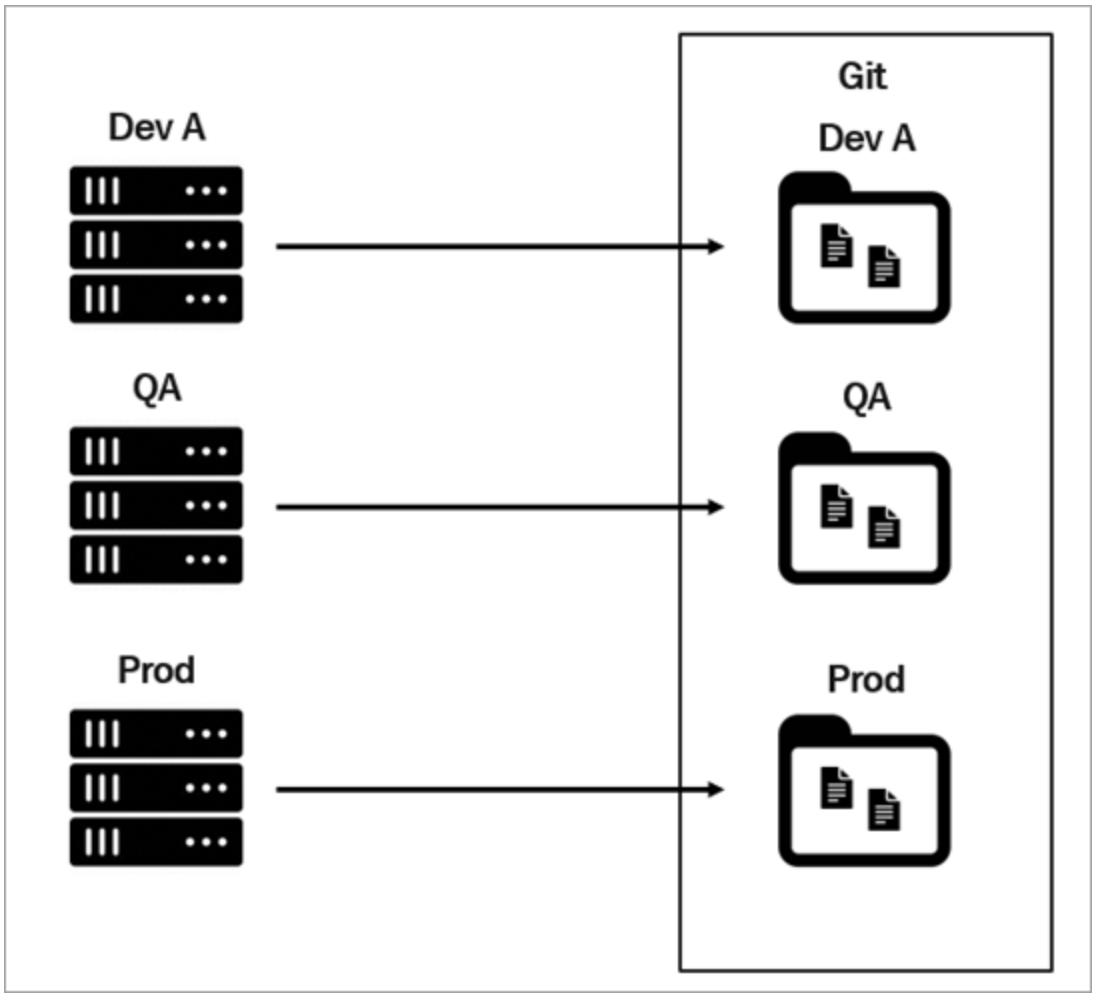
VCSI: Example Setups

The options described in the Integration Endpoint Options for VCS Integration topic can be used in several different scenarios and do not necessarily have to be used together. This section describes two potential setups.

System Comparison

You can use the OIEP options to have the configuration from each system in a Development, Testing, Acceptance, and Production (DTAP) environment published to different branches in a remote Git repository, allowing for easy manual comparison of configurations using the 'diff' tools Git offers.

Systems can publish their configurations with scheduled intervals or on demand either via workbench or by invoking the OIEPs remotely via REST using the resource operation available in the STEP REST API (accessible from the STEP API Documentation, at [system]/sdk or from the Start Page).

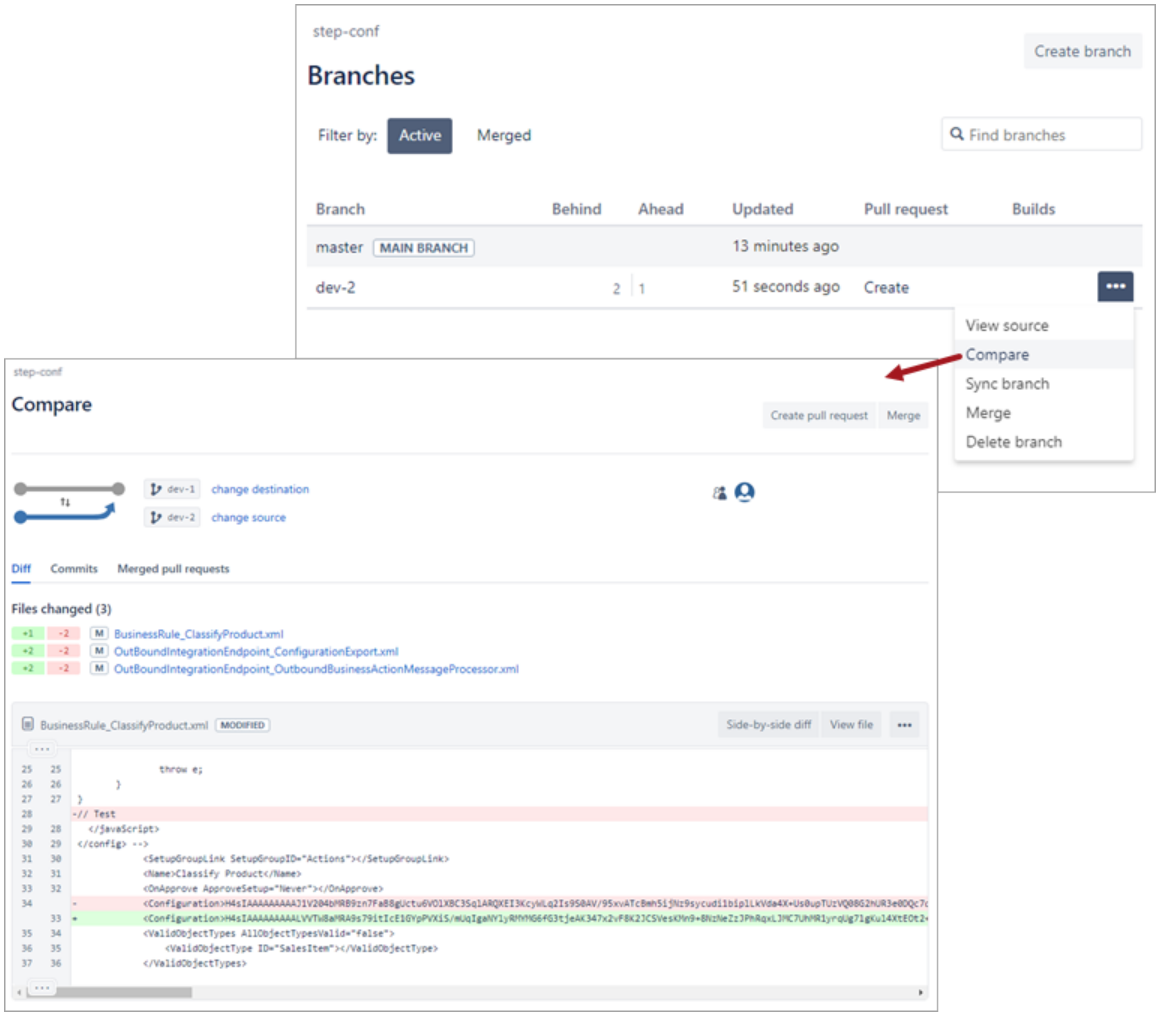


Such a setup can be used to ensure systems are in sync or only have expected differences. If differences are found, STEPXML files held in the Git branches can manually be imported one-by-one on a system that needs to be updated, or alternately, multiple files can be zipped and supplied to an IIEP configured to use the 'STEPXML Joiner' pre-processor described in the 'STEPXML Joiner Pre-processor for IIEP' section of the Integration Endpoint Options for VCS Integration topic.

Note: The VCS integration functionality offers no automatic dependency handling, which means that it is the responsibility of the user transferring files to ensure that all files necessary to create / update configuration objects are included and, given that the 'STEPXML Joiner' pre-processor is not used, that files are imported in the correct order.

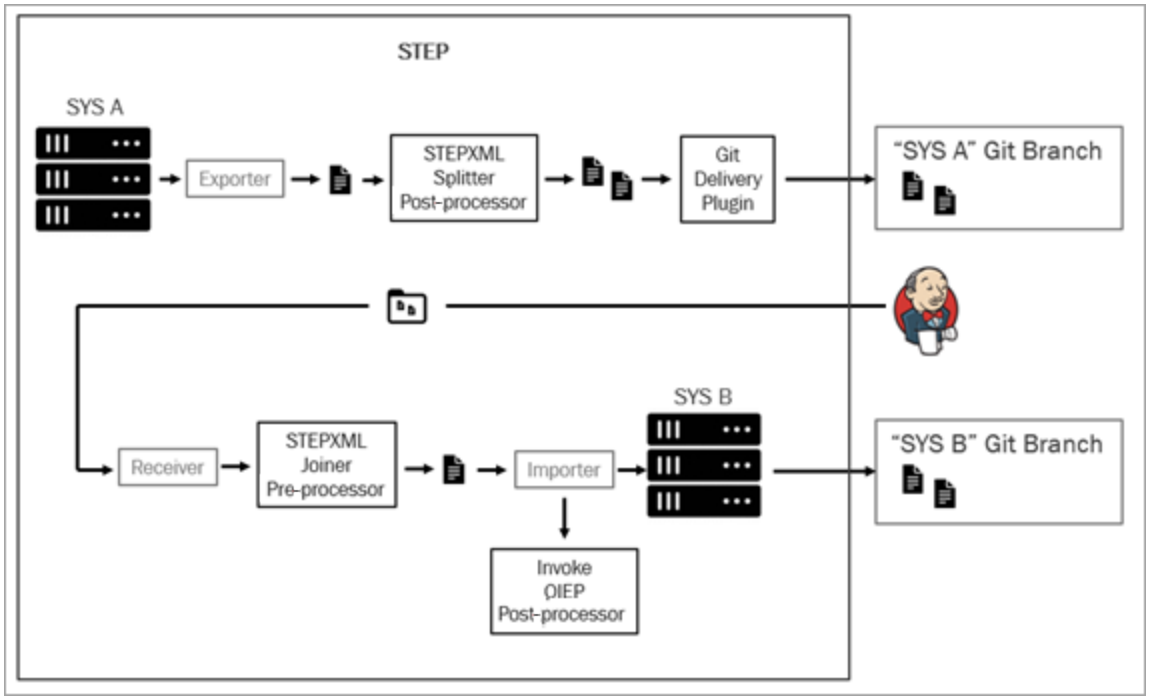
With this setup, a tool like Jenkins https://jenkins.io/ could be used to monitor the branches for changes. With a monitoring tool, diff reports can be sent to users via email.

Branch comparisons can be made using the Git command line tool (for example, refer to https://git-scm.com/docs/git-diff), or, if a tool like Bitbucket is used, via a web interface as shown below.



Semi-Automated System Synchronization

With a tool like Jenkins configured to monitor branches for changes, instead of sending diff reports to users, a Jenkins job could be used to automatically keep systems in sync. The diagram below illustrates a setup where changes on STEP system 'SYS A' are automatically deployed on another STEP system 'SYS B'.



For this to work, a Jenkins job must monitor the 'SYS A' branch for changes, and when such changes are identified, the job must compare the 'SYS A' branch with the 'SYS B' branch. The job then produces a .ZIP file containing files from the 'SYS A' branch that differ, and passes the .ZIP file to an IIEP on 'SYS B' configured to use the 'STEPXML Joiner' and 'Invoke OIEP' options.

An example Jenkins job 'Build' shell script ($gituser, $gitpassword and $sysbpassword, $sysbuser defined via 'Username and password (separated)' bindings) is included in this topic in online help.

Important: As GitHub and Bitbucket do not support Basic authentication (that is, using a user name and password in the URL), replace the value of `$gitpassword` with a Personal Access Token (in the case of GitHub) or an App Password (in the case of Bitbucket.)

Via REST, the Jenkins job could also invoke the IIEP on 'SYS B' and monitor the import process, notifying human users if errors occur. Alternately, the IIEP could be scheduled to run frequently, and an error reporter could be used to notify users about errors.

Note: For a setup like this, be aware that the 'STEP Importer' processing engine cannot handle all updates. Refer to the VCSI: Considerations and Limitations topic for more information.

VCSI: Considerations and Limitations

The Version Control System Integration (VCSI) functionality is only limited per standard STEP functionality, meaning that not all configurations can be exported / expressed in STEPXML, and not all changes can be applied via the STEP Importer processing engine. The functionality works for settings stored in the STEP database rather than files in the application server file system.

Known configurations / settings that cannot be exported / expressed in STEPXML:

- Web UI user configurable views
- Web UI user defined searches
- Web UI custom icons
- Scheduled background processes
- Workbench bookmarks

Known import limitations:

- Deletions can only be performed for products, entities, classifications, and assets. STEPXML for deleting such objects in a target system must be produced by a configured Jenkins job, or a job in a similar tool, upon identifying objects present in the target system and not present in the source system.
- A number of update operations for configuration objects cannot be carried out if there is data in the system conflicting with the change.
- A number of updates require single update mode.
- In some cases, workflow definitions cannot be updated if there are tasks for objects in the flow.

Maintaining Partial Data Sets on Lower Level DTAP Environments

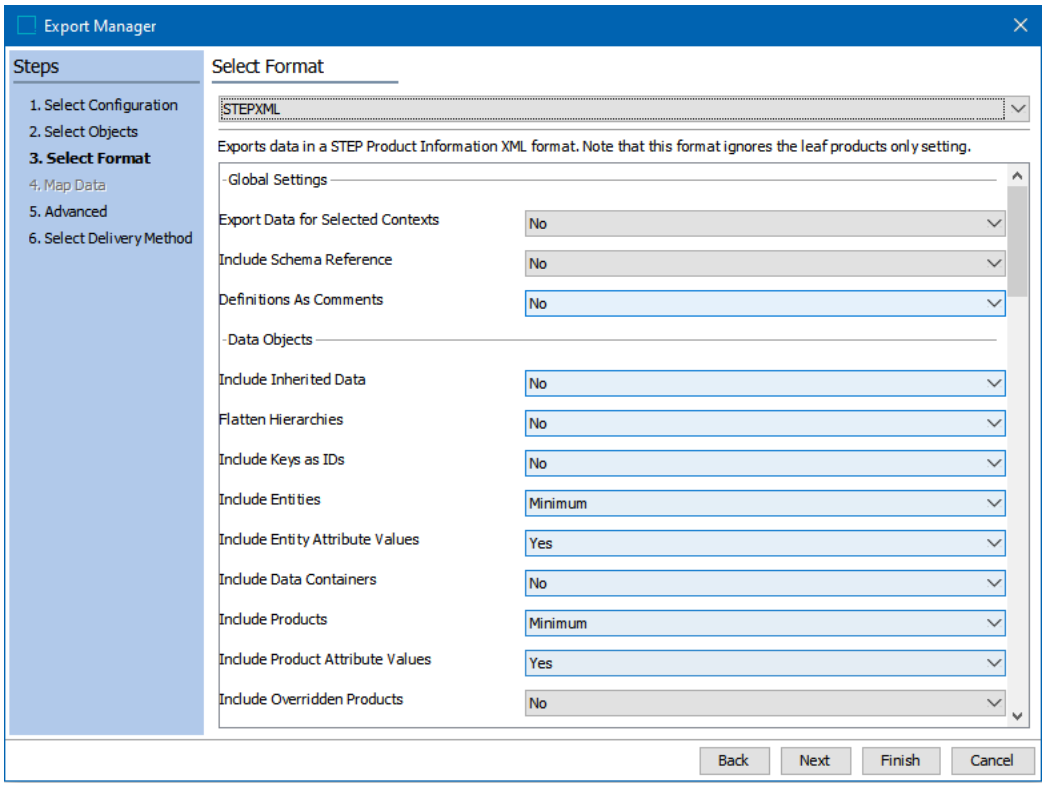
This section explains how you can keep Dev / QA / Sandbox systems up to date since these systems typically need all of the System Setup but only a small, representative subset of the data from your production environment. Since you only transfer a subset of the data, this makes it much faster and easier to keep these systems up to date and also reduces the hardware requirements for these systems.

Important: Oracle Data Pump exports and imports cannot be used to maintain partial data sets since this technology does not allow, for example, only exporting certain hierarchies or data from certain STEP contexts. Further, Oracle Data Pump imports overwrite any data created or modified in the target environment, which is often not desired. Instead, the recommendation is to use STEPXML for transferring data.

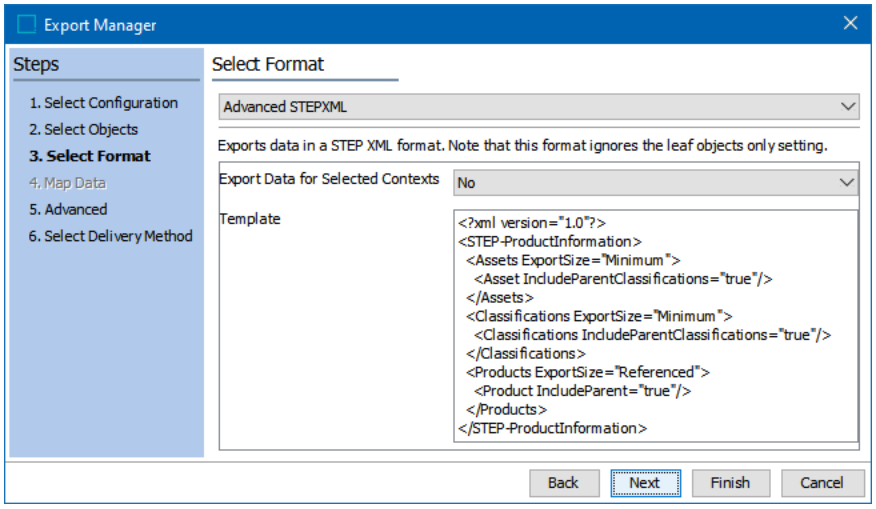
STEPXML Export Basics

STEPXML can be exported from a STEP system via the Export Manager using either the 'STEPXML' or the 'Advanced STEPXML' format. Both options produce the same format but differ in how you configure the export, i.e., how you decide what data should be included in the exported file. The Advanced STEPXML format makes use of an XML 'output template' sometimes also referred to as a 'recorder file' while the STEPXML format allows for the export to be configured via a UI with a large number of drop-down menus with the selections that allow for behind-the-scenes mapping to an output template. The Advanced STEPXML option is generally harder to work with but offers greater flexibility in configuring the export.

This is an Export Manager example with STEPXML:

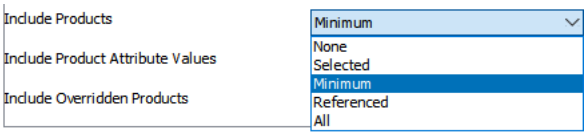


This is an Export Manager example with Advanced STEPXML:



Regardless which STEPXML format is used, a key concept to understand when exporting STEPXML is 'export size.' When exporting data, it is typically not feasible to individually select all the objects that should be exported. Instead, a few objects are selected and the export size then determines which objects should be included in addition to the selected objects.

With the STEPXML format, the export size is specified per super type via a dropdown as shown below:



With Advanced STEPXML, the export size is set via a super type-specific attribute as shown below:

```
<Products ExportSize="Minimum">
    <Product IncludeParent="true"/>
</Products>
```

The export sizes relevant in this context are described below.

Export Size: Selected

The 'Selected' export size, as the name suggests, indicates to the system that the data for the objects selected is to be included in the export.

When using the STEPXML format and selecting the option for products, classifications, and entities, additional objects will be included. Namely, all descendants of the selected object, and for classifications, all classification objects above the selected ones in the hierarchy. This is because choosing 'Selected' for products, classifications, and entities causes the output template shown below to be used for the export. For classifications, this template specifies that ancestors should be included (IncludeParent="true"), and since no detailed specification is given for the 'Classification' element (no nested elements specifying exactly what should be exported for a classification), descendants are also included. For products and entities, the presence of 'Product' and 'Entity' elements inside the outer 'Product' and 'Entity' elements similarly causes descendants to be included in the export.

```
<?xml version="1.0" encoding="utf-8"?>
<STEP-ProductInformation ResolveInlineRefs="true">
  <Classifications ExportSize="Selected">
    <Classification IncludeParent="true"/>
  </Classifications>
  <Products ExportSize="Selected">
    <Product>
      <Name/>
      <AttributeLink/>
      <DataContainerTypeLink/>
      <ClassificationReference/>
      <Product/>
      <ProductCrossReference/>
      <AssetCrossReference/>
      <EntityCrossReference/>
      <ClassificationCrossReference/>
      <Values/>
      <OverrideSubProduct/>
    </Product>
```

```
  </Products>
  <Entities ExportSize="Selected">
    <Entity>
      <Name/>
      <AttributeLink/>
      <ClassificationCrossReference/>
      <Entity/>
      <ProductCrossReference/>
      <AssetCrossReference/>
      <EntityCrossReference/>
      <ContextCrossReference/>
      <Values/>
    </Entity>
  </Entities>
</STEP-ProductInformation>
```

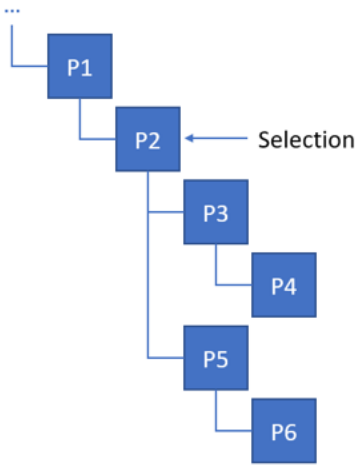
Export Size: All

The 'All' export size is straightforward in that it disregards the export selection and indicates to the system that all objects of a given super type are to be included in the export.

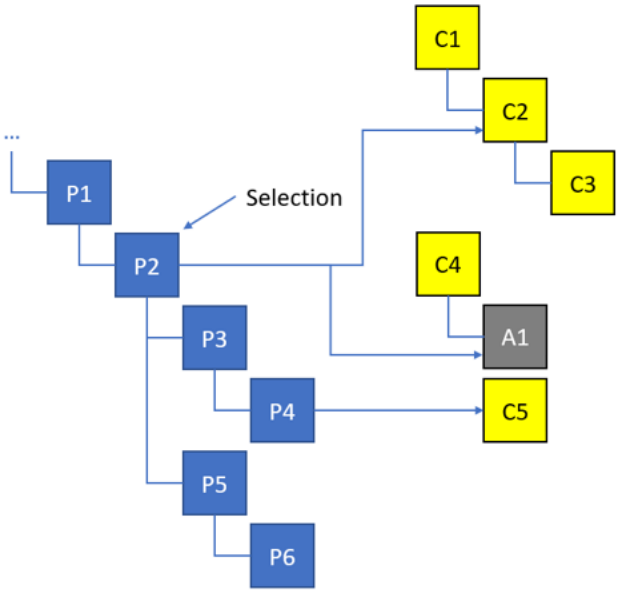
Export Size: Minimum

When focusing on a single super type (like products), the 'Minimum' export size works similarly to 'Selected' in that it indicates that the selected objects are to be exported, and dependent on the output template, includes descendants and ancestors.

For instance, with the example product hierarchy shown below, if P2 is selected for export and the export size 'Minimum' is used for products with the default output template, the exported file will contain data for P2, P3, P4, P5, and P6.



The difference between ‘Selected’ and ‘Minimum’ is that the ‘Minimum’ option works across super types. Assume you are working from a product hierarchy like the one above, but this time the selected product P2 is linked into a classification (C2) and further has a reference to an asset (A1) while the descendant product P4 is linked into the classification C5.



Running an export with just P2 selected and the ‘Minimum’ option specified only for products will cause the same product objects to be exported as in the example above. However, if ‘Minimum’ is also specified for classifications and assets and the default output template is used, the export will include asset A1 and classifications C1, C2, and C5, as well as the products. To summarize: the objects directly referenced / linked from the selection and its descendants will be included. In addition, the default template for classifications specifies that ancestors should be included; therefore, C1 is also included. The template is not applied recursively for non-selected objects when it comes to descendants, so C3 is not exported.

The ‘Minimum’ option can also be used to include configuration objects used by exported data in the exported file. As an example, the output template shown below will cause attributes, attribute groups, units, and lists of values (LOVs) relevant for the product selection to be included in the exported file.

```
<?xml version='1.0'?>
<STEP-ProductInformation>
  <AttributeList ExportSize="Minimum"/>
  <AttributeGroupList ExportSize='Minimum'/>
  <UnitList ExportSize='Minimum'/>
  <ListsOfValues ExportSize='Minimum'/>
  <Assets ExportSize="Minimum"/>
  <Classifications ExportSize="Minimum"/>
  <Products ExportSize="Minimum"/>
</STEP-ProductInformation>
```

To be more precise, if an export is run with P2 (from the example above) as the selection and the output template is used, in addition to data objects, the following configuration objects will be exported:

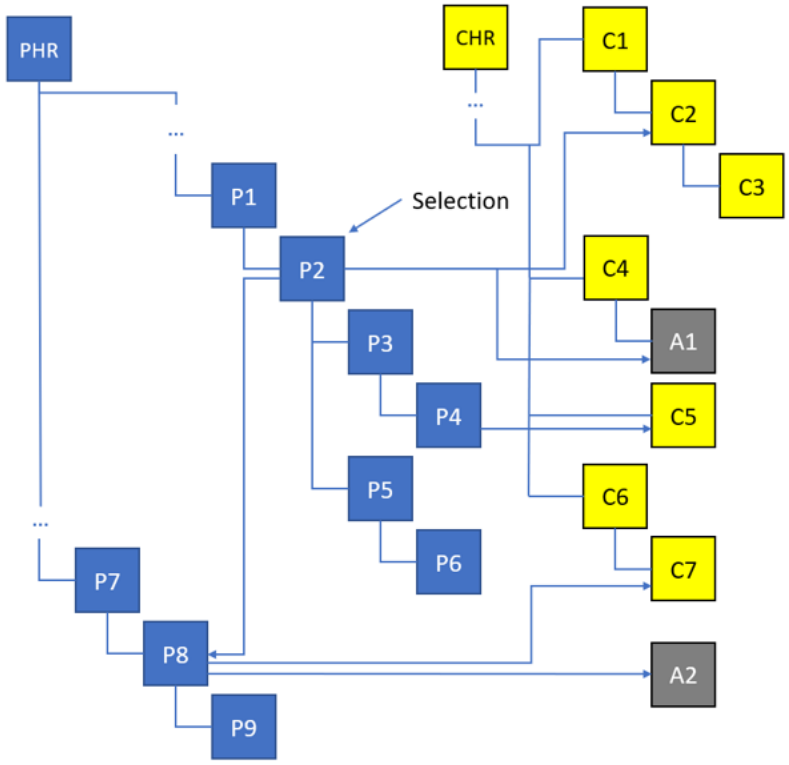
- All attributes used on the exported data objects
- All units used by the exported attributes
- All attribute groups that exported attributes are present in (and by default, all attribute groups up to the 'Attribute group root')
- All LOVs used by exported attributes

Note: The logic for including configuration objects is applied recursively. For instance, you will get definitions exported for attributes that are used for metadata on other configuration objects.

Export Size: Referenced

The 'Referenced' option is similar to 'Minimum,' but the option further prompts objects referenced from the selection or descendants to be exported.

Consider the following setup where the selection only contains product P2, which has a reference to product P8:



With this data, an export with the following output template:

```
<?xml version = "1.0" encoding = "utf-8"?>
<STEP-ProductInformation ResolveInlineRefs="true">
  <Assets ExportSize="Minimum"/>
  <Classifications ExportSize="Minimum"/>
```

```
  <Products ExportSize="Referenced"/>
</STEP-ProductInformation>
```

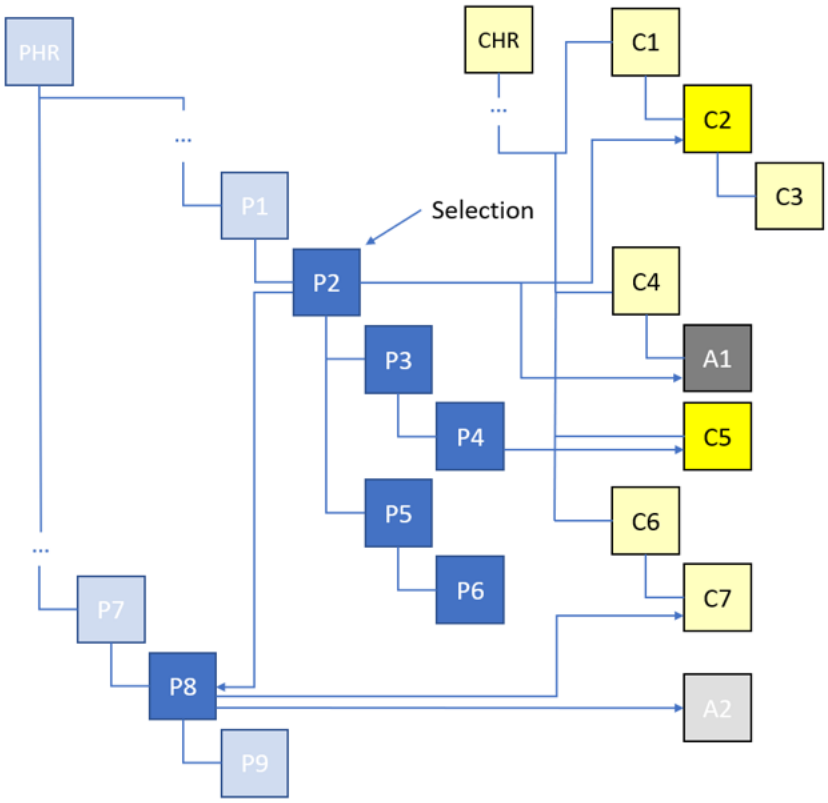
…produces the following objects in the export:

Products: P2, P3, P4, P5, P6, P8

Classifications: C2, C5

Assets: A1

To illustrate, in the diagram below, nodes that are not exported have been dimmed:



The logic is not applied recursively. In this example, the classification C7 that P8 is linked into and the asset A2 that the product references are not included in the export. Also P9, which is a child of P8, is not included.

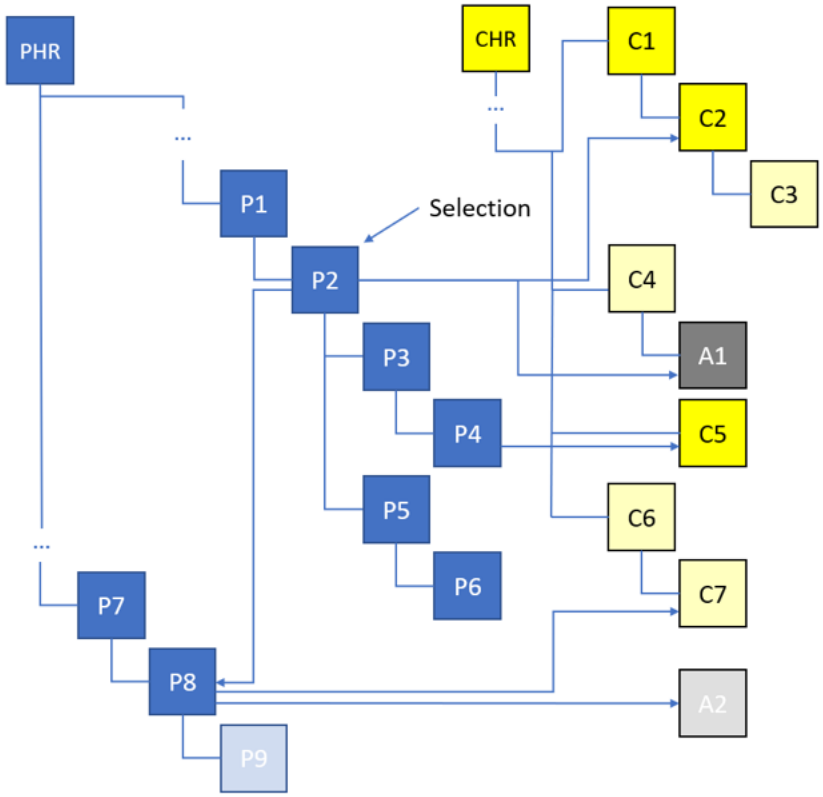
Including Ancestors

Reusing the last data example, importing the files exported with the settings described above results in errors on a system that does not have ancestors like P1, P7, C1 and C4 existing in the system in advance. However, when using the Advanced STEPXML format, the exporter can include ancestor objects up to the hierarchy roots ('PHR' and 'CHR' in the diagrams).

For products, classifications, and entities, include ancestors using the 'IncludeParent' attribute as shown in the output template example below:

```
<?xml version='1.0'?>
<STEP-ProductInformation>
  <Assets ExportSize="Minimum">
    <Asset/>
  </Assets>
  <Classifications ExportSize="Minimum">
    <Classification IncludeParent="true"/>
  </Classifications>
  <Products ExportSize="Referenced">
    <Product IncludeParent="true"/>
  </Products>
</STEP-ProductInformation>
```

With this output template, P1, P7, C1, and C4 and all ancestor nodes up to 'Product hierarchy root' ('PHR' in the diagram) and 'Classification 1 root' ('CHR' in the diagram), that both always exist on a STEP system, are included in the exported file. This allows importing the file on a system where the ancestor nodes do not exist in advance. This is illustrated below with non-exported nodes dimmed:



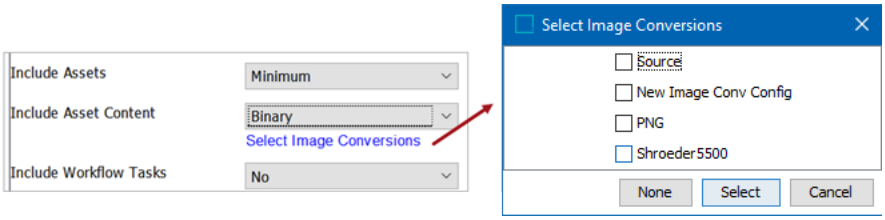
A similar attribute called 'IncludeParentClassifications' exists for assets. Setting this attribute to 'true' in an output template for the 'Asset' element will, when using the domain exporter (automatically enabled for In-Memory

systems), cause classifications that a selected or referenced asset is linked into to be included in the exported file. For example, using the hierarchy above, C4 and all ancestor classifications up to CHR will be included.

Asset Content

Asset content, i.e., the actual image and document files referenced from asset objects, is not represented in STEPXML per default. However, the data can be included using both the STEPXML and the Advanced STEPXML formats. Data can be imported again to create or update asset content.

With the STEPXML format, asset content can be included in the exported file as shown below:



With Advanced STEPXML, binary data can be included via the 'AssetContent' element as shown below:

```
<?xml version='1.0'?>
<STEP-ProductInformation>
  <Assets ExportSize="Minimum">
    <Asset>
      <AssetContent ExportType="Binary">
        <ImageConversionConfiguration ID="Source"/>
      </AssetContent>
      <!-- other asset specific instructions omitted ->
    </Asset>
  </Assets>
  <!-- other instructions omitted ->
</STEP-ProductInformation>
```

With Advanced STEPXML, once you start specifying sub elements for the super type specific elements like 'Asset' and 'Product,' you only export the data you have specified. For example, with the template above, you would not export asset names or values.

To export name, values, references, and classification links, use a template for assets as follows:

```
<Assets ExportSize="Minimum">
  <Asset>
    <Name/>
    <ClassificationReference/>
    <EntityCrossReference/>
    <Values/>
    <AssetContent ExportType="Binary">
      <ImageConversionConfiguration ID="Source"/>
```

```
    </AssetContent>
  </Asset>
</Assets>
```

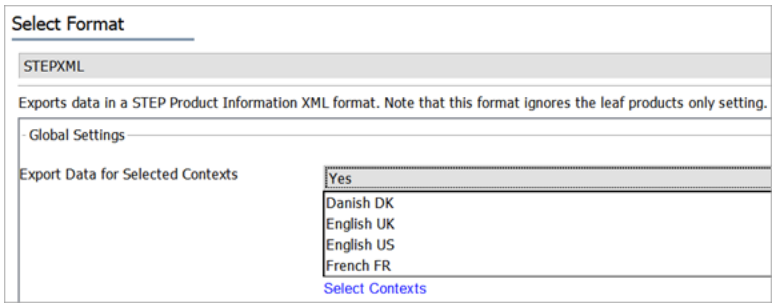
When exporting asset content, converted versions of the content cannot be imported – only the unconverted source (in the exported file, data for the unconverted source will be in an 'AssetBinaryContent' element for which the value of the 'ImageConversionConfiguration' 'ID' attribute is blank / empty string).

Important: Since including asset content can lead to very large files, seriously consider if asset content is required in the target systems.

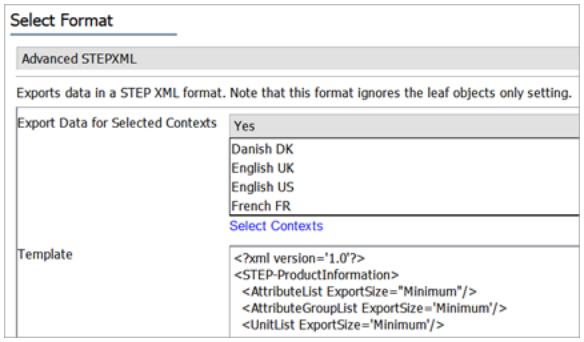
Cross Context Exports

A STEPXML file can contain data from multiple contexts, and you can manually export data for multiple contexts with the STEPXML format and the Advanced STEPXML format.

The context selection using the STEPXML format is shown below:



The context selection using the Advanced STEPXML format is shown below:



Transferring Configuration and Data Between Systems

With the STEPXML export functionality described above, you can select specific category nodes and have all descendant nodes plus referenced and linked nodes across multiple contexts included in a single STEPXML file. If Advanced STEPXML is used, you can include ancestors to the exported nodes, thereby allowing for the data to be imported on an empty system.

Important: For the import to succeed, it is crucial that required configuration objects like object types, attributes, units, etc., are either included in the file or present in the target system prior to importing the data.

If the target system is empty, it will often make sense to include configuration objects in the data export file. The most straightforward approach is to use the export size 'All' for all configuration objects so that all attributes, units, object types, integration endpoints, Web UI configurations, etc. are included in the exported file.

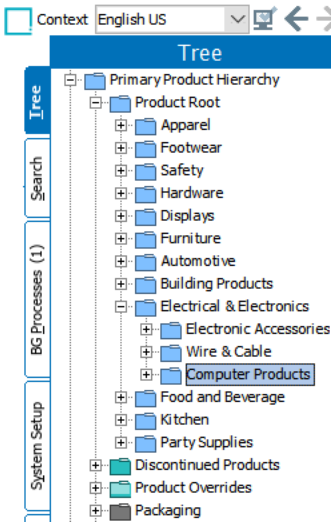
For configuration objects like attributes, you can use the 'Minimum' setting to avoid getting large amounts of attributes included that are not needed. If doing so, however, there is a high probability that attributes that are not directly used by the exported data but required for some other functionality, e.g., a business rule or a Web UI configuration, will be missing.

If the target system is not empty, it will make sense to use the Version Control System (VCS) integration functionality introduced with STEP version 9.1 to manage the configuration. For details about this functionality, refer to the Version Control System Integration section of the Configuration Management documentation. In brief, the functionality allows for system configurations to be pushed to a branch in a Git repository, thereby allowing comparison of the configuration across systems. The files present in Git will be valid STEPXML files that can be imported using the standard import manager or via inbound integration endpoint functionality also described in the Version Control System Integration section mentioned above.

With the desired configuration in place, data can be moved ad hoc using the functionality described above.

Example

This section illustrates how the export logic described can be used to export a sub section of a product hierarchy and have classification and asset dependencies included while ensuring that the exported file can be imported on a, data-wise, empty system.



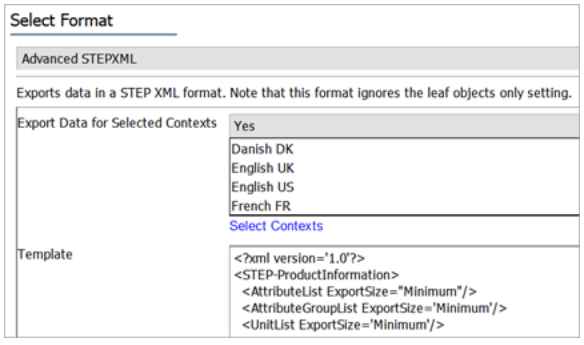
Consider the data hierarchy shown above. For this example, the goal is to export the 'Computer Products' category of products. And, in the exported file, include data for:

- All products below and including ‘Computer Products’
- All product ancestors for ‘Computer Products’ up to and including ‘Primary Product Hierarchy’
- All classifications that products below ‘Computer Products’ are linked into
- The ancestors for these classifications up to and including ‘Classification 1 root’ (hidden classification hierarchy root node)
- All assets referenced from products below ‘Computer Products’ including the binary asset content
- All classifications that these assets are placed in
- The ancestors for these classifications up to and including ‘Classification 1 root’

This goal can be achieved by selecting the ‘Computer Products’ product for export and using the following output template:

```
<?xml version='1.0'?>
<STEP-ProductInformation>
<Assets ExportSize="Minimum">
  <Asset IncludeParentClassifications="true">
    <Name/>
    <ClassificationReference/>
    <EntityCrossReference/>
    <Values/>
    <AssetContent ExportType="Binary">
      <ImageConversionConfiguration ID="Source"/>
    </AssetContent>
  </Asset>
</Assets>
<Classifications ExportSize="Minimum">
  <Classification IncludeParent="true"/>
</Classifications>
<Products ExportSize="Selected">
  <Product IncludeParent="true"/>
</Products>
</STEP-ProductInformation>
```

As described above, the exported file can be made to include data from multiple contexts by selecting the desired context in the export manager as shown below:



The export could further be modified to also include product objects from other categories referenced from products below ‘Computer Products,’ and their ancestors. This can be achieved by replacing ‘<Products ExportSize="Selected">‘ with ‘<Products ExportSize="Referenced">‘.

Note: The objects referenced from the reference targets will not be included in the export.

To import the generated file, definitions of attributes, object types, units, LOVs, etc., must either exist in the target system in advance or be included in the file. As mentioned above, the configuration can either be managed separately using the VCS integration functionality or the configuration objects can be included in the file by using the export size ‘All’ for all desired types. ‘Minimum’ can potentially also be used, e.g., for attributes, but as previously mentioned, the safest choice is to include all.

If you want to export a subset of data as described above and include all of the System Setup configurations, your output template would look similar to this one:

```
<?xml version='1.0'?>
<STEP-ProductInformation>
  <Assets ExportSize="Minimum">
    <Asset IncludeParentClassifications="true">
      <Name/>
      <ClassificationReference/>
      <EntityCrossReference/>
      <Values/>
      <AssetContent ExportType="Binary">
        <ImageConversionConfiguration ID="Source"/>
      </AssetContent>
    </Asset>
  </Assets>
  <Classifications ExportSize="Minimum">
    <Classification IncludeParent="true"/>
  </Classifications>
  <Products ExportSize="Selected">
    <Product IncludeParent="true"/>
  </Products>
  <TagGroupList/>
  <TagList/>
  <Qualifiers/>
  <GlobalSettings/>
  <UserTypes ExportSize="All"/>
  <Keys/>
  <DerivedEventTypes/>
  <EdgeTypes/>
  <CrossReferenceTypes ExportSize="All"/>
  <DimensionList/>
  <ContextList/>
  <UnitList ExportSize="All"/>
  <CollectionList ExportSize="All"/>
```

```
  <ListOfValuesGroupList/>
  <ListsOfValues ExportSize="All"/>
  <IntegrationEndpoints/>
  <EventProcessors/>
  <SetupGroups/>
  <SetupEntities/>
  <AttributeGroupList ExportSize="All"/>
  <AttributeList ExportSize="All"/>
  <DataContainerTypes ExportSize="All"/>
  <ActionSetList/>
  <UserGroupList/>
  <UserList/>
  <SystemSetup ExportSize="All"/>
  <TableColors ExportSize="All"/>
  <TableRules ExportSize="All"/>
  <TableTypeGroupList ExportSize="All"/>
  <TableTypeDefinitions ExportSize="All"/>
  <ECatalogs/>
  <EventQueues/>
  <STEPWorkflows ExportSize="All"/>
  <StatusFlags ExportSize="All"/>
  <BusinessLibraries ExportSize="All"/>
  <BusinessRules ExportSize="All"/>
  <MatchCodes/>
  <MatchingAlgorithms/>
  <PortalConfigurations ExportSize="All"/>
  <AttributeTransformationGroups/>
  <ImportConfigurations ExportSize="All"/>
  <ExportConfigurations ExportSize="All"/>
  <BulkUpdateConfigurations ExportSize="All"/>
  <TransformationLookupTableConfigurations ExportSize="All"/>
  <ComponentModels/>
</STEP-ProductInformation>
```

For reference, also refer to the Transferring STEP Configuration and Sample Data topic in this documentation.

Transferring STEP Configuration and Sample Data

This document describes how to transfer configuration and sample data from one STEP system (the source system) to another (the target system).

It is assumed that the target system is either empty or, configuration-wise, closely resembles the source system. Trying to update a target system that configuration-wise is far from the source system via STEPXML export / import will most likely lead to numerous errors during import.

It is recommended to transfer configuration and sample data separately, and the two tasks are covered in separate sections below.

Two template export configurations are supplied with this guide via the online help documentation: a 'Configuration-Export' export configuration for exporting the STEP configuration included in the 'Configuration-Export.xml' file and a 'ProductDataSample-Export' export configuration for exporting sample product data included in the 'SampleProductData-Export.xml' file (refer to the Sample Data: Export and Import section for details on how to export Customer MDM sample data). Refer to documentation online to access to these files.

System Configuration Export and Import

Important: The 'STEPXML Configuration Export' format, available with 10.1-MP3 and newer systems, significantly reduces the need for manual work when exporting the configuration from a STEP system. Refer to the STEPXML Configuration Export Format topic in the Data Formats section of the Data Exchange documentation. **When using this format, the steps described below in the sections 'Importing the Configuration-Export Export Configuration' and 'Using the Configuration-Export Export Configuration' can be skipped.**

This section describes how to export the configuration from one STEP system to STEPXML, producing a file that can be imported on an empty STEP system or a system with a very similar configuration without errors.

'Configuration,' as referenced in this document, is understood as the STEP system configuration stored in the STEP database accessible via the workbench System Setup tab, i.e., all object type definitions, attribute definitions, business rules, workflows, etc.

A template export configuration 'Configuration-Export' is supplied with this guide in the STEPXML file 'Configuration-Export.xml,' linked in the section above.

The following sections explain how to import, modify, and use the export configuration on the system from which the configuration should be exported (source system) and how to import the produced file on the target system.

Note: Prior to exporting and importing the STEP configuration, it is important to ensure that the same STEP software components (non-baseline components and extensions) installed on the source system are also

nstalled on the target system. Additionally, the same licenses must be enabled. Importing the configuration or features not available in the target system will lead to errors during import.

Importing the Configuration-Export Export Configuration

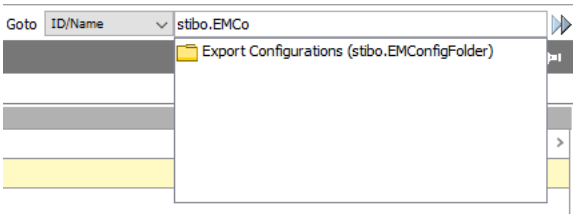
Follow the steps below to import the Export Configuration on the source system.

1. The Export Configuration will by default be imported in the STEP Context with the ID 'Context1' (a context existing by default in newer STEP systems). If you do not have a Context with this ID in the source system, open the 'Configuration-Export.xml' file in a text or XML editor, change the value for the 'STEP-ProductInformation' element 'ContextID' attribute to the ID of the Context to import into and save the changes.

```
01 <?xml version="1.0" encoding="utf-8"?>
02 <STEP-ProductInformation ContextID="Context1" WorkspaceID="Main">
03   <ExportConfigurations>
04     <ExportConfiguration ID="Configuration-Export">
05       <Name>Configuration-Export</Name>
06       <ClassificationReference ClassificationID="stibo.EMConfigFolder"/>
07       <Configuration>H4sIAAAAAAAAAKVXbW/bOAz+K319qBIkrQLLul6al
```

   All contexts in a STEP system are displayed in the Contexts editor of the System Setup tab in the workbench.

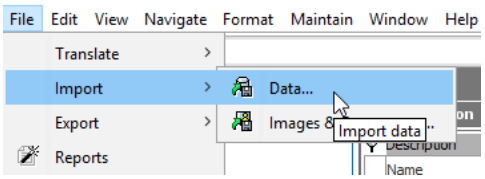
2. The Export Configuration will be imported below a Classification with ID 'stibo.EMConfigFolder.' This classification exists by default in all STEP systems; however, it could have been deleted. Check that the classification exists via the workbench **Goto** field as shown below.



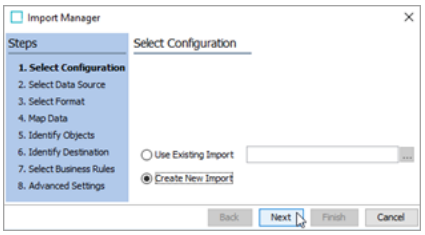
   If the classification does not exist, open the 'Configuration-Export.xml' file in a text or XML editor, change the value for the 'ClassificationReference' element 'ClassificationID' attribute to the ID of the Classification to have the Export Configuration imported into and save the changes.

```
01 <?xml version="1.0" encoding="utf-8"?>
02 <STEP-ProductInformation ContextID="Context1" WorkspaceID="Main">
03   <ExportConfigurations>
04     <ExportConfiguration ID="Configuration-Export">
05       <Name>Configuration-Export</Name>
06       <ClassificationReference ClassificationID="stibo.EMConfigFolder"/>
07       <Configuration>H4sIAAAAAAAAAKVXbW/bOAz+K319qBIkrQLLul6a
08     </ExportConfiguration>
```

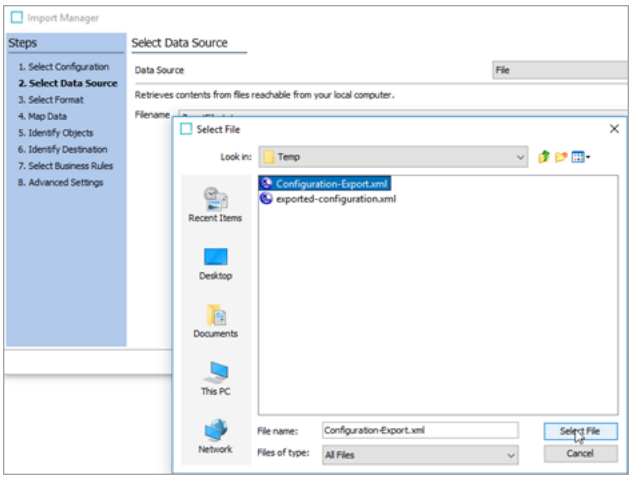
3. In the STEP Workbench, from the File menu select **Import** and then select **Data…** .



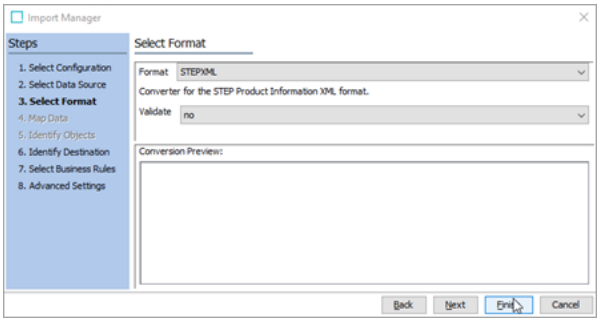
4. In the **Select Configuration** step of the Import Manager, click the **Next** button.



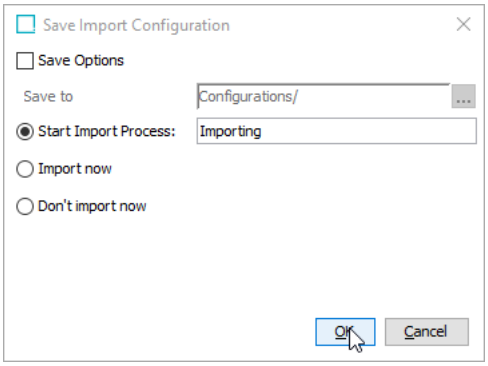
5. In the **Select Data Source** step of the Import Manager, click the ellipsis button (...), browse to where the 'Configuration-Export.xml' file is saved, select it and click the **Next** button.



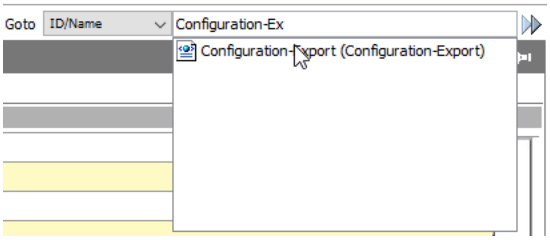
6. In the **Select Format** step of the Import Manager, STEP should automatically recognize the file as a STEPXML file. Confirm that Format is 'STEPXML' and click **Finish**.



7. In the **Save Import Configuration** dialog, uncheck **Save Options** if selected and click **OK** to start the import process.



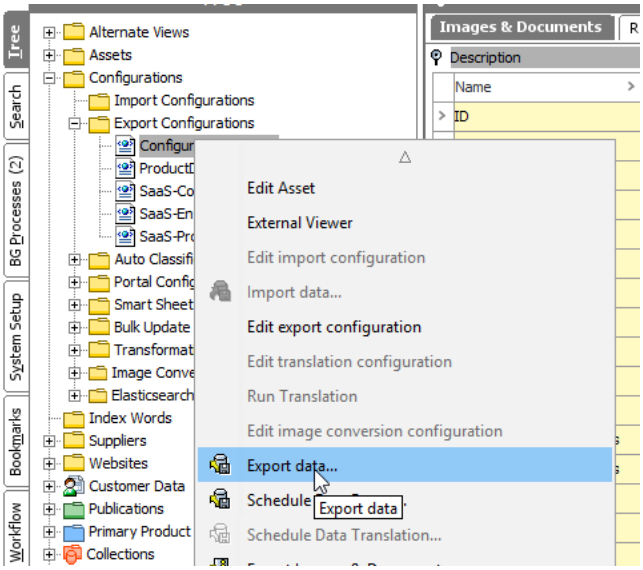
8. After the import process has finished, locate the 'Configuration-Export' Export Configuration. For example, you can use the STEP Workbench **Goto** field as shown below.



Using the Configuration-Export Export Configuration

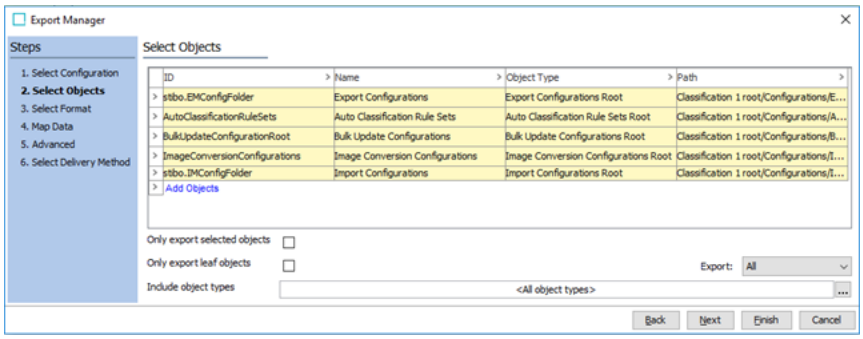
The 'Configuration-Export' export configuration must be modified so that the exported STEPXML file will include all objects required for the file to be imported on the target system without errors. Follow the steps below to configure and run the export.

1. Locate the 'Configuration-Export' file in the STEP Workbench Tree, right-click it, and select **Export data…**



2. In the **Select Objects** step of the Export Manager, data nodes (Classifications, Entities, and Products) that are required for the configuration to be successfully imported on the target system must be added. For this exercise, launch an additional workbench instance so that you can add objects in the Export Manager in one and browse the Tree and System Setup in another. The nodes that must be added are:

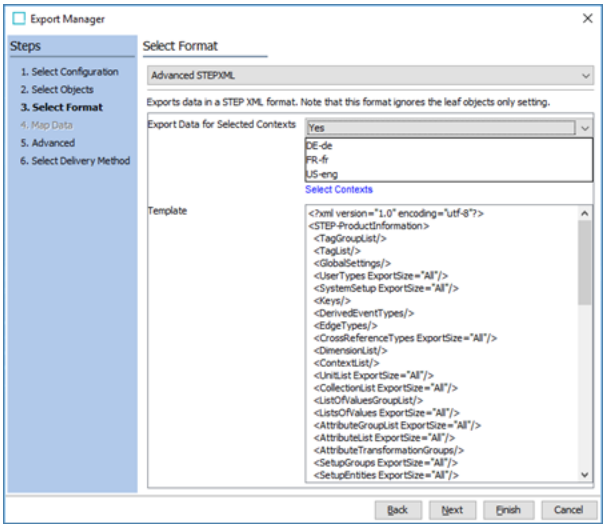
   - All Classifications holding Export Configurations (immediate parents)
   - All Classifications holding Import Configurations (immediate parents)
   - All Classifications holding Bulk Update Configurations (immediate parents)
   - All Classifications holding Image Conversion Configurations (immediate parents)
   - All Classifications holding Transformation Lookup Tables (immediate parents)
   - All Classifications / products / entities referenced by Privilege Rules on User Groups (it is not necessary to add the super type specific root nodes 'Product hierarchy root,' 'Classification 1 root,' 'Entity hierarchy root,' 'CollectionGroup root,' 'eCatalog root,' etc. that will exist in advance on the target system)
   - Entities and Products referenced from Match Codes (Category)
   - All Classifications referenced directly from Supplier User Groups (Supplier section on the Group tab for Supplier User Groups)



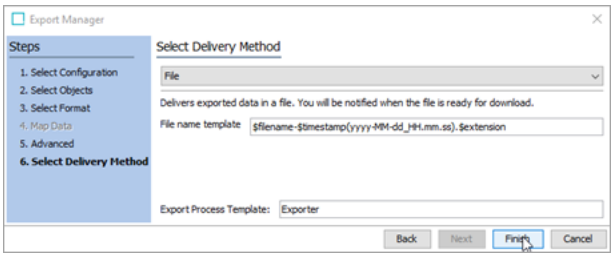
Note: With this export configuration, only the IDs and Names of these nodes and their ancestors up to the type specific root will be exported (i.e., no attribute values, references, etc.).

Click **Next** after adding all required nodes.

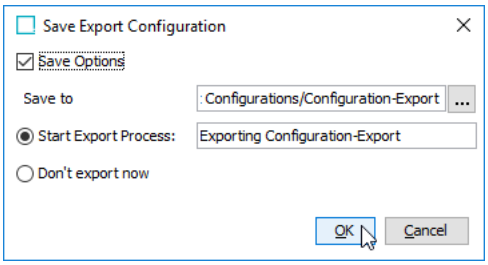
3. In the **Select Format** step of the Export Manager, under 'Export Data for Selected Contexts,' select the Contexts for which data should be exported. Notice that this is only relevant if the configuration objects and data nodes are dimension dependent (e.g., language-specific attribute names and LOV values). Click **Next** to continue.



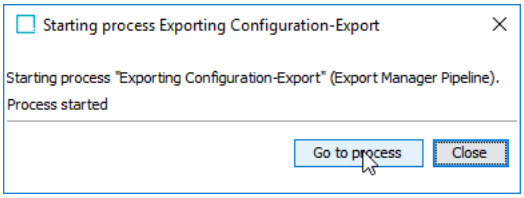
4. In the **Advanced** step of the Export Manager, make sure that **Workspace** is set to Main and click **Next**.
5. In the **Select Delivery Method** step, ensure that the 'File' option is selected and click **Finish**.



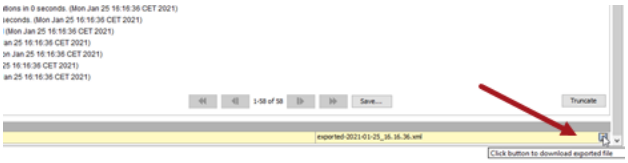
6. Select the **Save Options** checkbox to save the configuration changes for later use and click **OK** to start the export process.



7. Click **Go to process** in the dialog that appears.



8. When the export process has completed, click the save icon in the lower right corner of the background process editor to save the file locally.



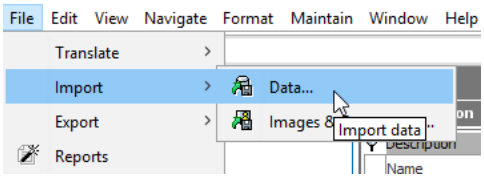
Importing the Configuration on the Target System

This section describes how to import the exported STEP configuration on the target system. Follow the steps below to import the configuration on the target system.

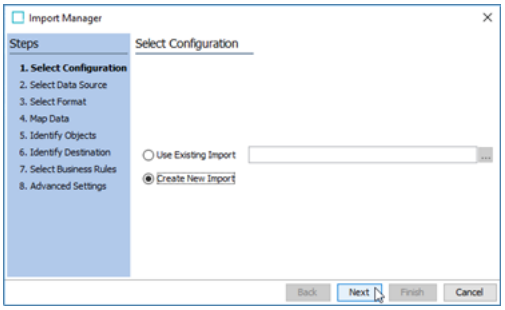
1. Assuming that the target system has not earlier been used, only a single Context with ID 'Context1' will exist in advance, and if the exported STEPXML file was not exported from a Context with the same ID, the STEPXML file must be modified. Open the exported STEPXML file in a text or XML editor. If the value for the

'STEP-ProductInformation' element 'ContextID' attribute is not 'Context1,' change it to be so and save the modified file.

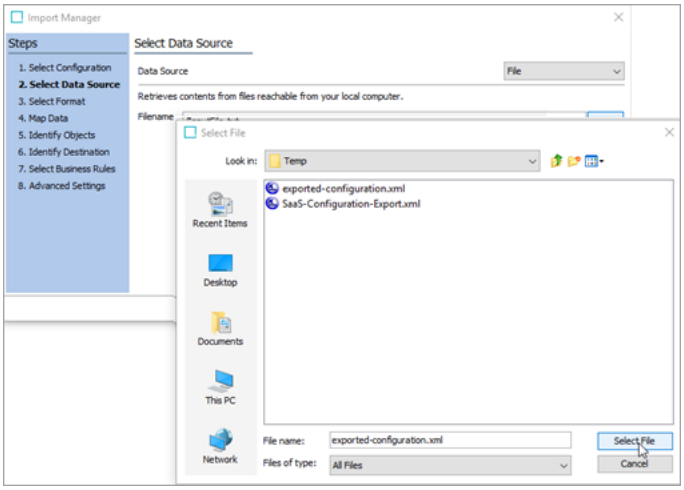
2. Launch the STEP Workbench for the target system, from the File menu select **Import** and then select **Data…** .



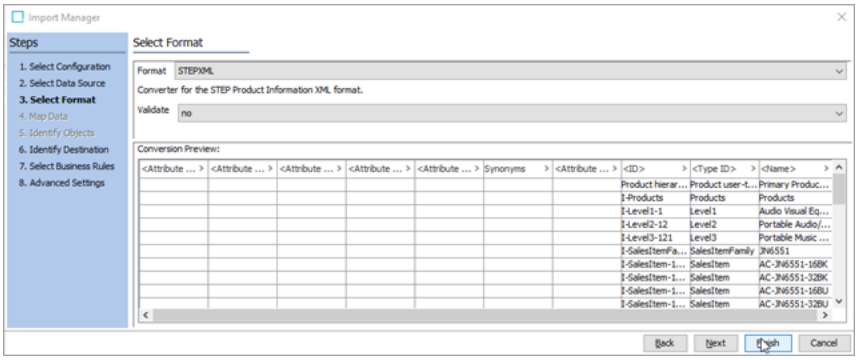
3. Click **Next** in the **Select Configuration** step of the Import Manager.



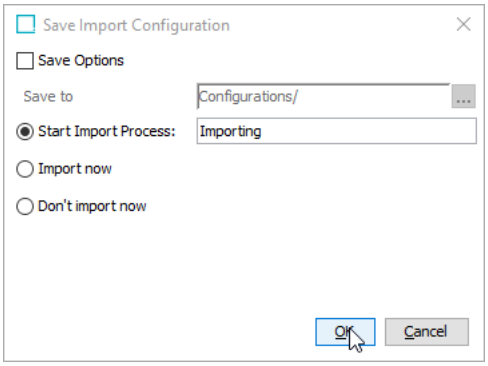
4. In the **Select Data Source** step of the Import Manager, click the ellipsis button (**...**), browse to where you have saved the STEPXML file exported from the source system, select it, and click **Next**.



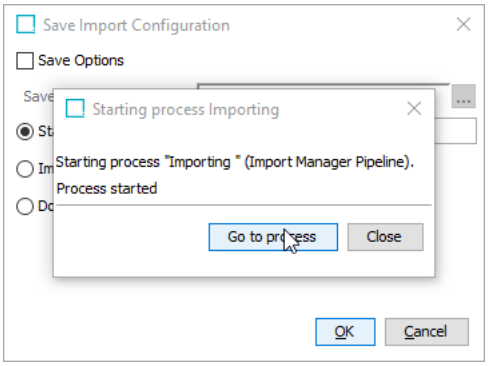
5. In the **Select Format** step of the Import Manager, STEP should automatically recognize the file as a STEPXML file. Confirm that Format is 'STEPXML' and click **Finish**.



6. In the **Save Import Configuration** window, uncheck **Save Options** if selected, and click **OK** to start the import process.



7. Click **Go to process** in the dialog that appears and inspect the execution report for warnings and errors.



Taking Post Import Actions

After the configuration has been imported, it may be necessary to perform a number of additional manual steps as described below.

- Configuration created by a CSV import file must be recreated in the target system. For example, collections created by a file or a value generator that uses an imported list.
- User objects maintained in STEP that are referenced from Integration Endpoints and Event Processors and used for M2M integrations can be created via import with a temporary password automatically assigned by the system, which must be reset by the user in Web UI or by an administrator. Users can also be created manually after import. It is expected that User objects representing human users will be automatically created in the target system via an IDP integration with user synchronization.
- Scheduled background processes must be recreated on the target system. This includes scheduled exports, scheduled data profiling and Workflow deadline monitoring.
- If the configuration has been imported on an empty target system, all integration endpoints, event processors, gateway integration endpoints, and keys are disabled and must be enabled manually. When importing changes to an existing key, the key must be disabled prior to installation.
- Prior to enabling integrations with external systems, check that the target system is configured to receive from / publish to the correct external system. Any passwords stored in these configurations must be re-entered manually.
- As data is imported in the Main workspace, data nodes must be approved to also be reflected in the Approved workspace.
- The deletion of system configurations or removal of associations is not supported.

Sample Data: Export and Import

This section describes how to export sample data from the source system and subsequently import it on the target system.

A template export configuration 'ProductDataSample-Export' is supplied with this guide in the STEPXML file 'ProductDataSample-Export.xml.' As the name indicates, the export configuration is meant to be used for exporting product sample data with related classifications and assets, meaning that it is not directly applicable for Customer MDM systems. A separate section 'Exporting Customer MDM Sample Data' describes how to export Customer MDM sample data.

Importing the ProductDataSample-Export Export Configuration

The 'ProductDataSample-Export' export configuration supplied in the STEPXML file 'ProductDataSample-Export.xml' can be imported on the source system following the same procedure used for importing the 'Configuration-Export' export configuration. Refer to the **Importing the Configuration-Export Export Configuration** section above.

Using the ProductDataSample-Export Export Configuration

The ProductDataSample-Export export configuration makes use of the Advanced STEPXML export format with the following instructions.

```
<?xml version='1.0'?>
<STEP-ProductInformation>
  <Assets ExportSize="Minimum">
    <Asset IncludeParentClassifications="true"/>
  </Assets>
  <Classifications ExportSize="Minimum">
    <Classification IncludeParent="true"/>
  </Classifications>
  <Products ExportSize="Referenced">
    <Product IncludeParent="true"/>
  </Products>
</STEP-ProductInformation>
```

When one or more products are selected as the root objects for the export, these instructions will cause the following objects to be exported:

- The selected product objects
- All ancestors for the selected products up to the product hierarchy root
- All descendants of the selected products
- All products referenced from the selection or descendants and the ancestors for these referenced products up to the product hierarchy root
- All classifications that the selected products and their descendants are linked into and the ancestors for these classifications up to the classification hierarchy root
- All assets referenced from the selected products and their descendants (asset objects, no content)
- Further, if the export is run on an In-Memory enabled system, all classifications that the referenced assets are linked into and their ancestors up to the classification hierarchy root

If the sample product data references entities, referenced entities and their ancestors can be included by extending the output template to be:

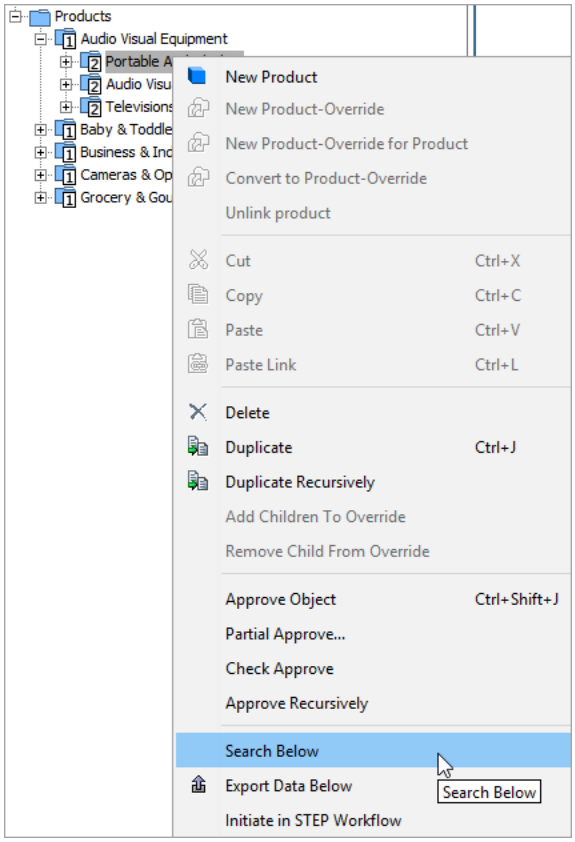
```
<?xml version='1.0'?>
<STEP-ProductInformation>
  <Assets ExportSize="Minimum">
    <Asset IncludeParentClassifications="true"/>
  </Assets>
  <Classifications ExportSize="Minimum">
    <Classification IncludeParent="true"/>
  </Classifications>
  <Entities ExportSize="Minimum">
    <Entity IncludeParent="true"/>
  </Entities>
  <Products ExportSize="Referenced">
    <Product IncludeParent="true"/>
```

```
  </Products>
</STEP-ProductInformation>
```

When importing a file produced with the ProductDataSample-Export export configuration, it is expected that the import process will report errors for missing reference and product classification link targets. Thus, while the template will cause referenced objects and their ancestors to be included in the export, the template will not be applied recursively to the referenced / linked objects. That is, the objects that these exported objects reference are not included in the produced file. As a result, these objects will be missing from the imported file.

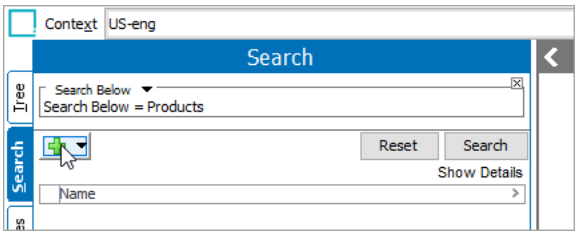
There are multiple different ways to make the initial sample product selection. This guide focuses on two.

The first option is to select one or more category / family product nodes. With the template export configuration, all product nodes below the selections will be exported, so use the workbench **Search Below** feature to ensure that the number of product objects below the selections is not too large.

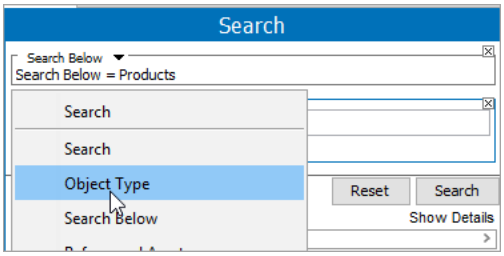


If a wider spread of product sample data is required, you can use a collection to select a category higher in the hierarchy which can contain too many product objects to be exported fully but instead use a sample below that level that to use as the selection. Follow the steps below to create a Collection with the sample products.

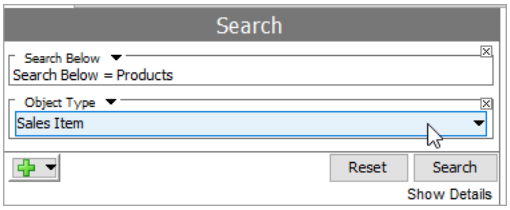
1. Right-click the desired category in the STEP Workbench Tree and select **Search Below** from the context menu (as shown above).
2. In the **Search** tab, click the button with the green plus sign to add another criteria to the search.



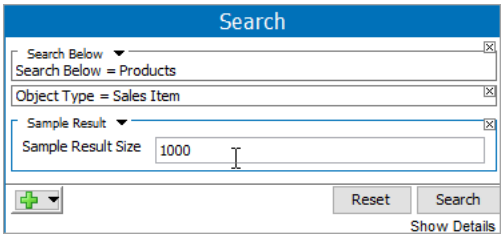
3. Click the search criteria type selector and choose **Object Type** from the menu.



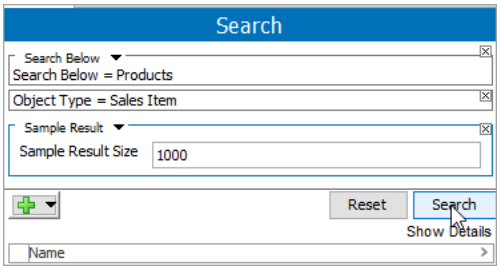
4. Select your leaf product object type.



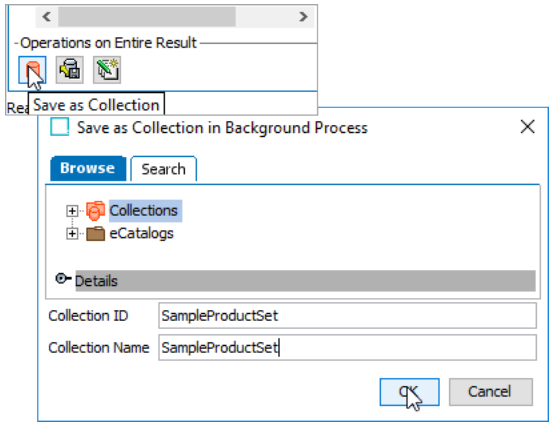
5. Add another criteria of the type **Sample Result**, and set the desired sample result size.



6. Execute the search.

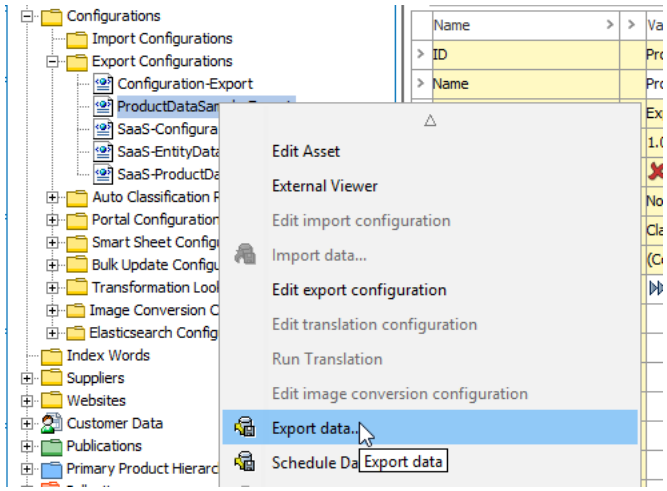


7. Click the **Save as Collection** button at the bottom of the search panel to store the result in a Collection.

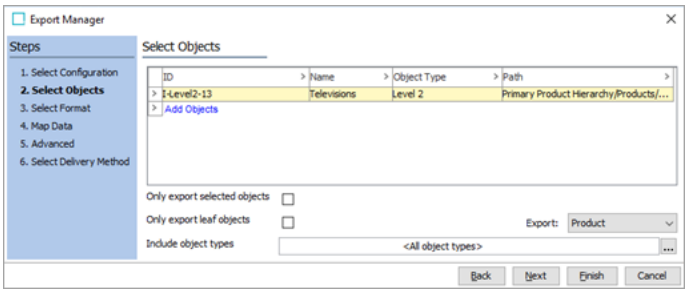


After deciding which approach to use (and creating the collection with sample data, if desired), follow the steps below to export the sample data.

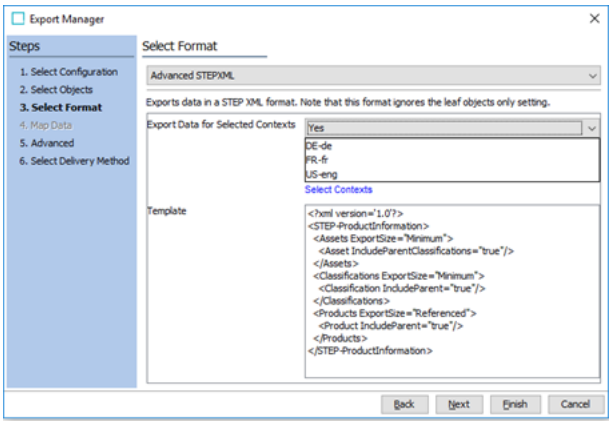
1. Locate the imported 'ProductDataSample-Export' export configuration in the STEP Workbench Tree, right-click it and select **Export data…**.



2. In the **Select Objects** step of the Export Manager, depending on which approach you have chosen, either select the root nodes for the sample product hierarchies you would like to export or select the Collection created earlier and click **Next**.

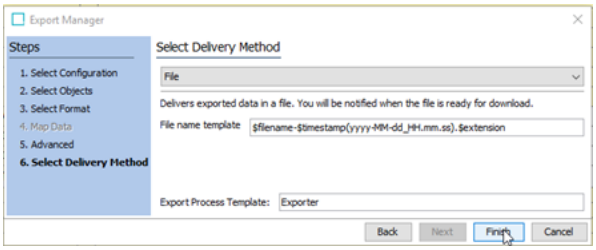


3. In the **Select Format** step, under **Export Data for Selected Contexts**, select the Contexts for which data should be exported if there is dimension dependent data. Click **Next** to continue.

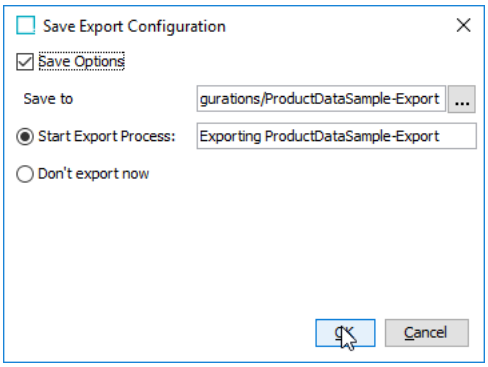


4. In the **Advanced** step, make sure that **Workspace** is set to Main and click **Next**. It might be relevant to export data from Approved, but this then requires that the produced STEPXML file is manually modified prior to being imported.

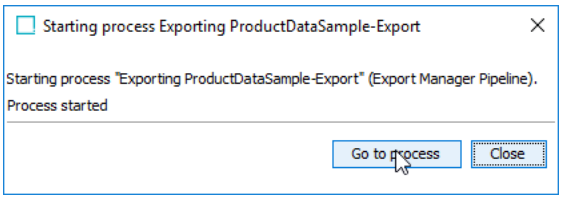
5. In the **Select Delivery Method** step of the Export Manager, ensure that the **File** option is selected and click **Finish**.



6. Select the **Save Options** checkbox to save the configuration changes for later use and click **OK** to start the export process.



7. Click **Go to process** in the dialog that appears.



8. When the export process has completed, click the save icon in the lower right corner of the background process editor to save the file locally.

Exporting Customer MDM Sample Data

If sample data instead needs to be exported from a Customer MDM system where data primarily is modeled using entities, use the same export configuration and replace the Advanced STEPXML instructions like the following.

```
<?xml version='1.0'?>
<STEP-ProductInformation>
  <Entities ExportSize="Referenced">
    <Entity IncludeParent="true"/>
  </Entities>
</STEP-ProductInformation>
```

Notice that, depending on the data model, it may be relevant to use a combination of these instructions and the product sample data instructions in the template (if the entities, for example, have referenced assets or reference classifications).

As Customer MDM setups often have large amounts of entities below the same parent node (e.g., all customer contacts below the same parent node), it is advanced to always use the sample search approach described above using a search that samples entities instead of products.

Importing the Sample Data on the Target System

The exported sample data can be imported on the target system in exactly the same way as the configuration. Refer to the **Importing the Configuration Export on the Target System** section above.

Note: The configuration must be imported before sample data.

For more information, also refer to the Maintaining Partial Data Sets on Lower Level DTAP Environments topic.