STIBO SYSTEMS
MASTER DATA MANAGEMENT

USER GUIDE

Configuration Management

Release 11.0-MP4 (September 21, 2022)

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 workflows, Web UIs, integration endpoints (IEPs), and business rules can be exported. 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 (e.g., users may not export, edit the comments, and re-import).

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>
```

More information on exporting configuration definitions as comments is available as follows:

- For workflow definitions, refer to the **Exporting Workflow Definitions** topic in the **Workflows** documentation.
- For Web UI definitions, refer to the **Exporting Web UI Definitions as Comments** topic of the **Web User Interfaces** documentation.
- For inbound integration endpoint definitions, refer to the **Exporting Inbound Integration Endpoint Definitions as Comments** topic in the **Data Exchange** documentation.
- For outbound integration endpoint definitions, refer to the **Exporting Outbound Integration Endpoint Definitions as Comments** topic in the **Data Exchange** documentation.
- For business rule definitions, refer to the **Exporting Business Rule Definitions as Comments** topic of the **Business Rules** documentation.

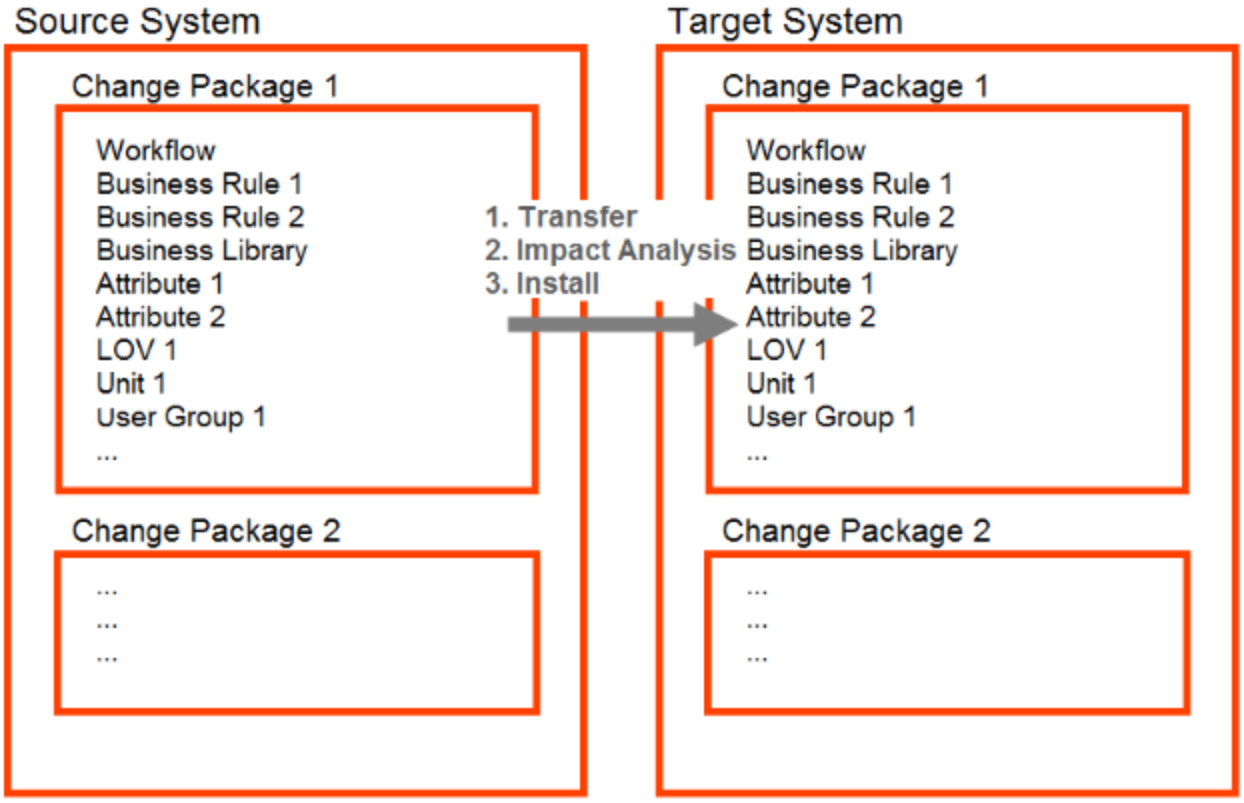
Note: The content of the comment field is not part of the STEPXML XSD and therefore Stibo Systems reserves the right to change the format of the output content at any time.

Change Packages

A change package allows for an easy way to prepare, process 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

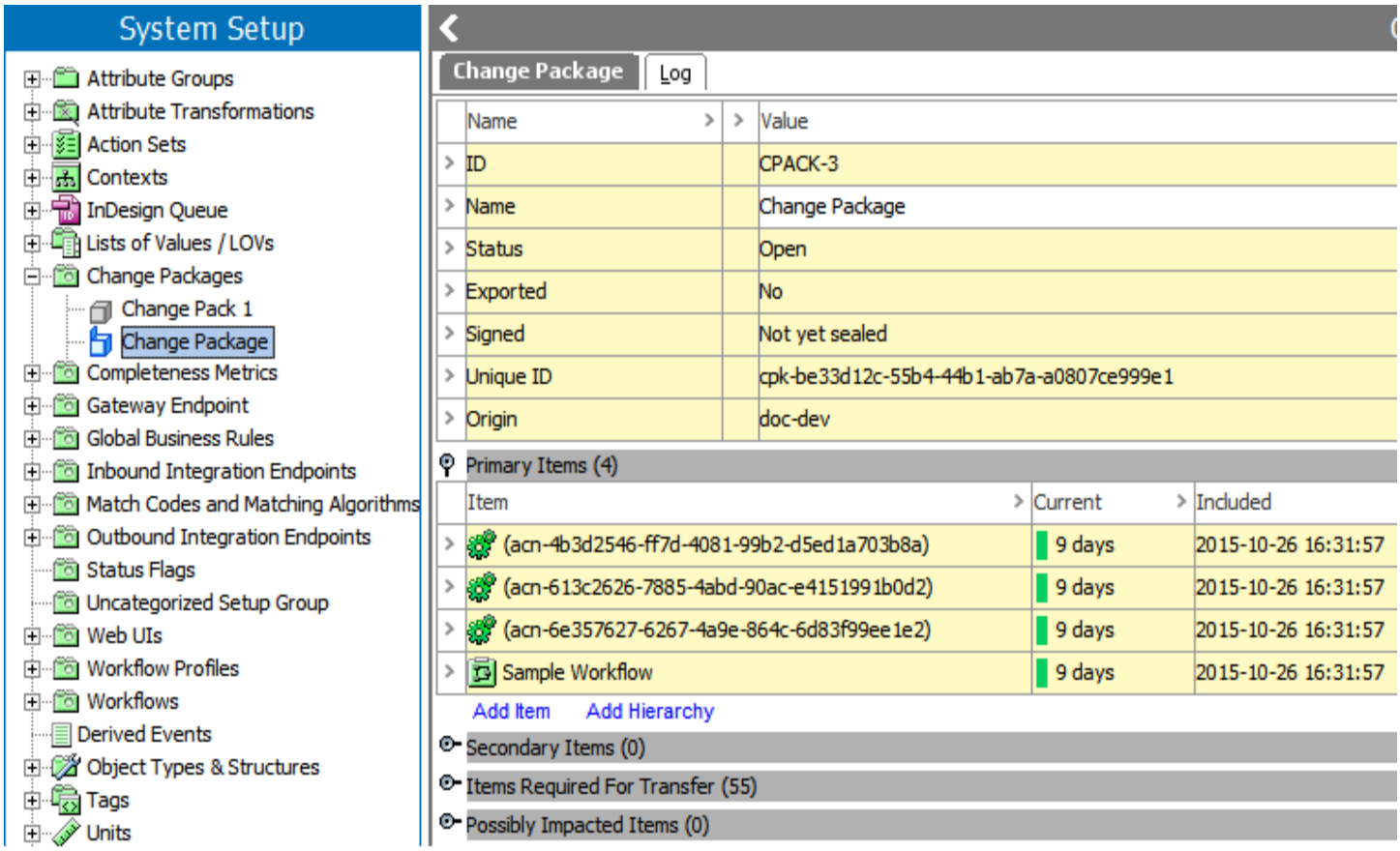
The overall flow of change packages is shown below:



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 applied. The user may then choose to remove or install the change package. If installed successfully, the configurations contained in the change package will be loaded to the system and available for immediate use.

Change Package Objects

Change packages are system setup objects that act as containers to house a set of configurations.



Note: Change packages can include the following STEP objects: Action sets, Asset Importer, Attributes, Attribute groups, Attribute transformations, Business Rules, Classification Product Link Types, Completeness Metric, Contexts, D&B, Dimensions, Dimension Points, Derived Events, Event processors, Event Queues, Gateway Endpoints, Integration endpoints, Link Types, List of Values, Match Codes, Matching Algorithms, Object Types and Structures, PIM Tables, Reference Types, Setup Entities, Setup Groups, Status Flags, Sufficiency configurations, Unique keys, Units, Unit groups, Users and User Groups, Web UI configurations, Workflows, and XSLT Style sheets.

Change Package Icons and Statuses

Icon	Status	Description
	Open	• Change package is in an editable state, therefore not final or ready for export yet. Packages are open when created and when re-opened from a sealed state • Can have an impact report run only if previously sealed • Cannot be exported • Items can be added and removed • Packages can be deleted and the items within them are left on the system unchanged
	Sealed	• Change package is locked for editing and ready for export • Can have an impact report run

Icon	Status	Description
		• Can be exported • Items cannot be added or removed • Packages can be deleted and the items within them are left on the system unchanged
	Dormant	• Change package has been imported from another system but not yet installed • Can have an impact report run • Can be exported • Items cannot be added or removed • Packages can be deleted and there is no impact to the items within them. If the package has been previously installed, the items within the package are left on the system unchanged. If the package has not yet been installed, the items within it are no longer available for installation.

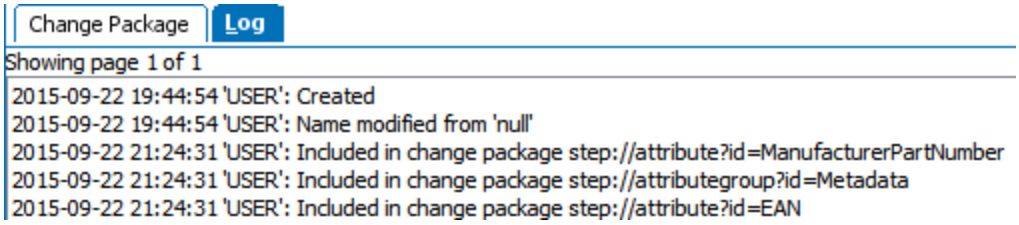
Change Package Tab

Flipper	Description
	Provides basic information about the change package, including the Status, whether or not the change package has been exported, whether or not the package has been sealed, and where the package is originally from (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 will be present with links to these processes.
Primary Items	Displays a list of objects that have been directly added to the change packages, as well as providing the interface for adding and removing items from the change package. Users can select to add a single object (Add Item), or an object and all of its child objects (Add Hierarchy). 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	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. 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	System generated list of objects that are required for the change package due to interactions with the selected objects. This list can only be edited by 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 objects are included in the change package as a means of ensuring that the primary and secondary items are successfully

Flipper	Description
	transferred, and will be created and/or updated on the target system when the change package is installed.
Possibly Impacted Items	System generated list of items that might be affected by the transfer of the change package on the new 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. They will only be modified if they already exist on the target system.

Log Tab

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

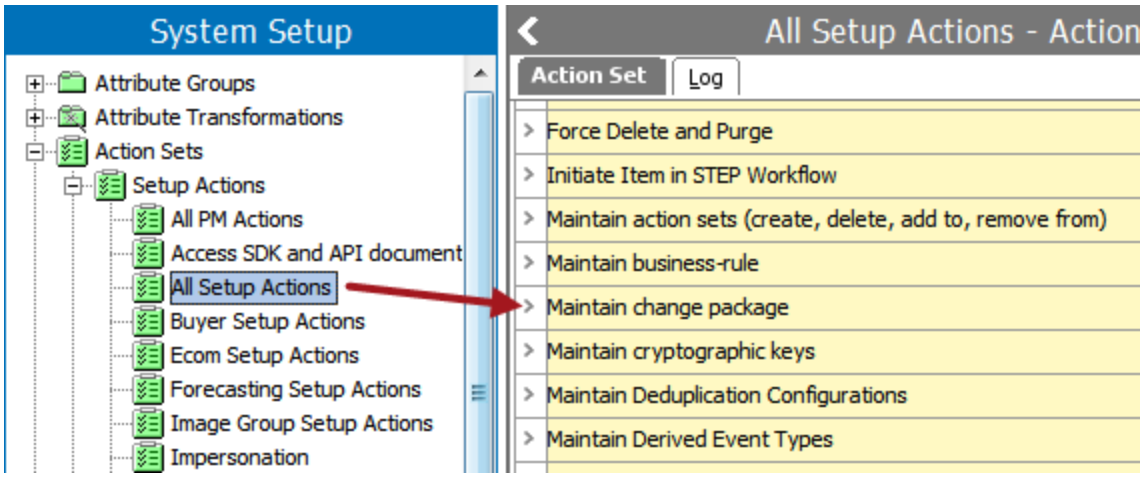


In addition, when an item is added to a change package or removed from a change package, the log of the item itself is also updated accordingly.

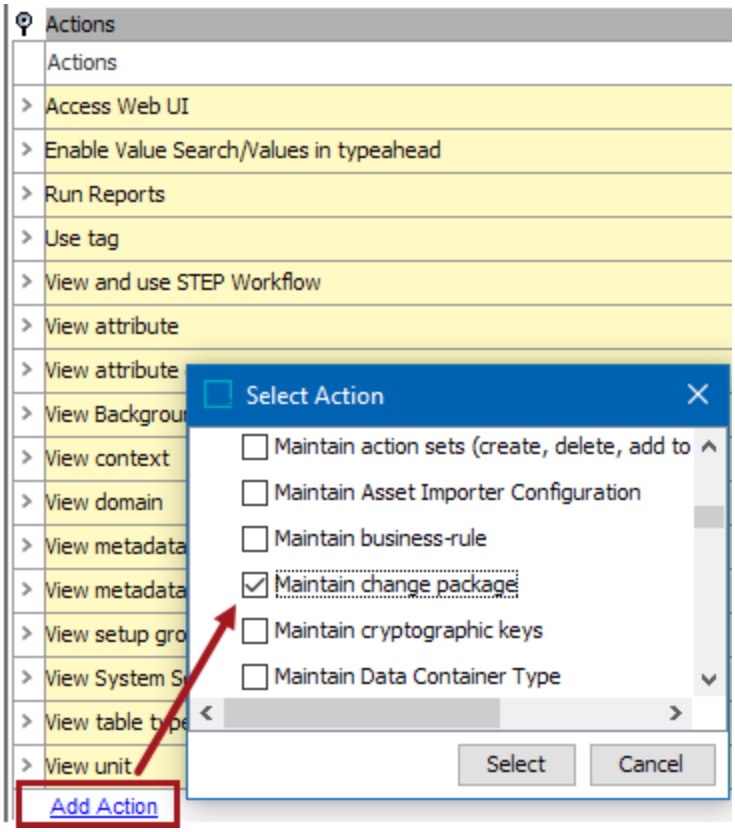
Change Package Privileges

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.

As an administrator, to make sure 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**.

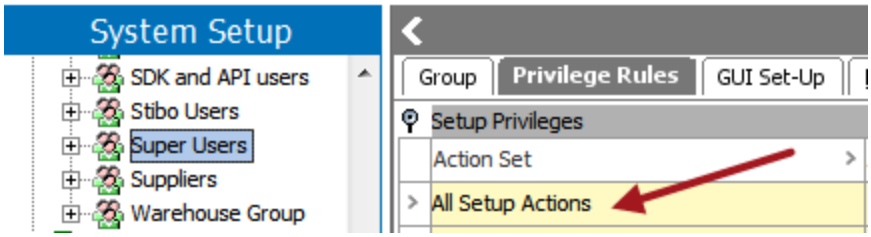


In the list to the right, if the **Maintain change package** action does not exist, scroll to the bottom of the list and select **Add Action**. This will populate a list of actions that are available to add to this set.



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 in order to enable full use of change package functionality.

Once it has been verified that the **All Setup Actions** action set does in fact include *all* setup actions, any users requiring access to change package functionality must be 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 within the **System Setup** documentation.

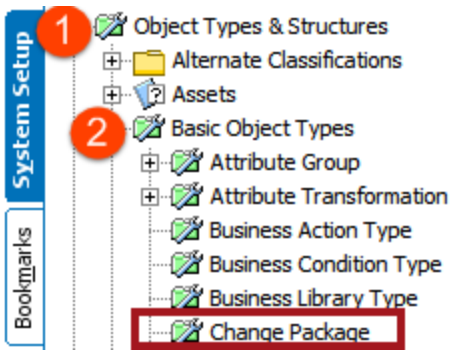
For information on including users in a change package, refer to the **Editing a Change Package** topic.

Initial Setup for Change Packages

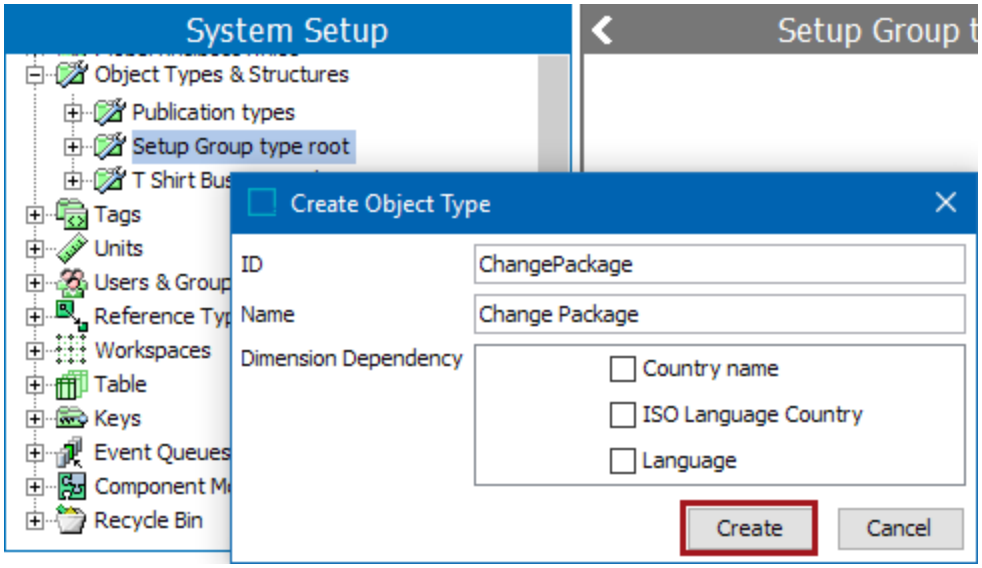
In order to create a Change Package that will process and migrate STEP configuration changes, the basic change package configuration must first be in place as defined below.

Change Packages functionality was introduced in version 7.4.

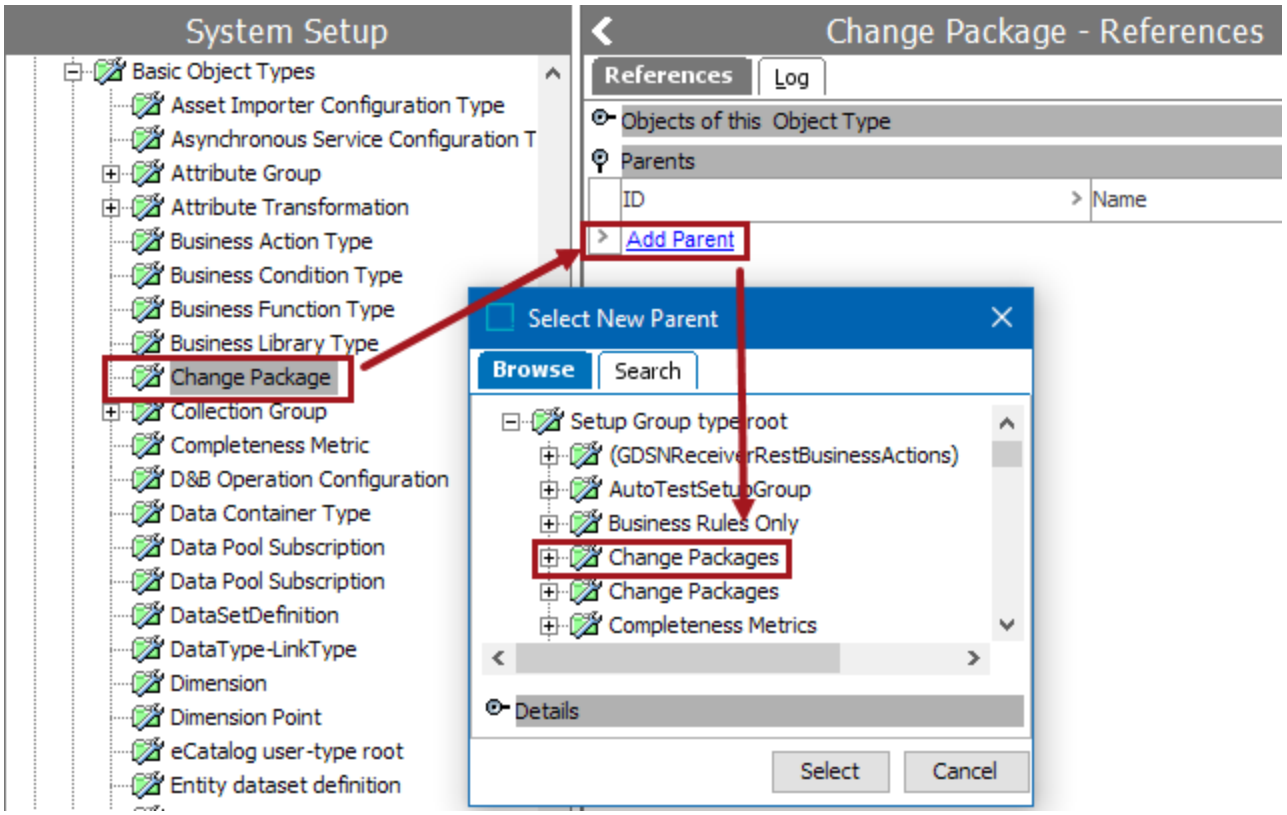
1. On System Setup, select **Object Types and Structures** > **Basic Object Types**, and verify **Change Package** exists.



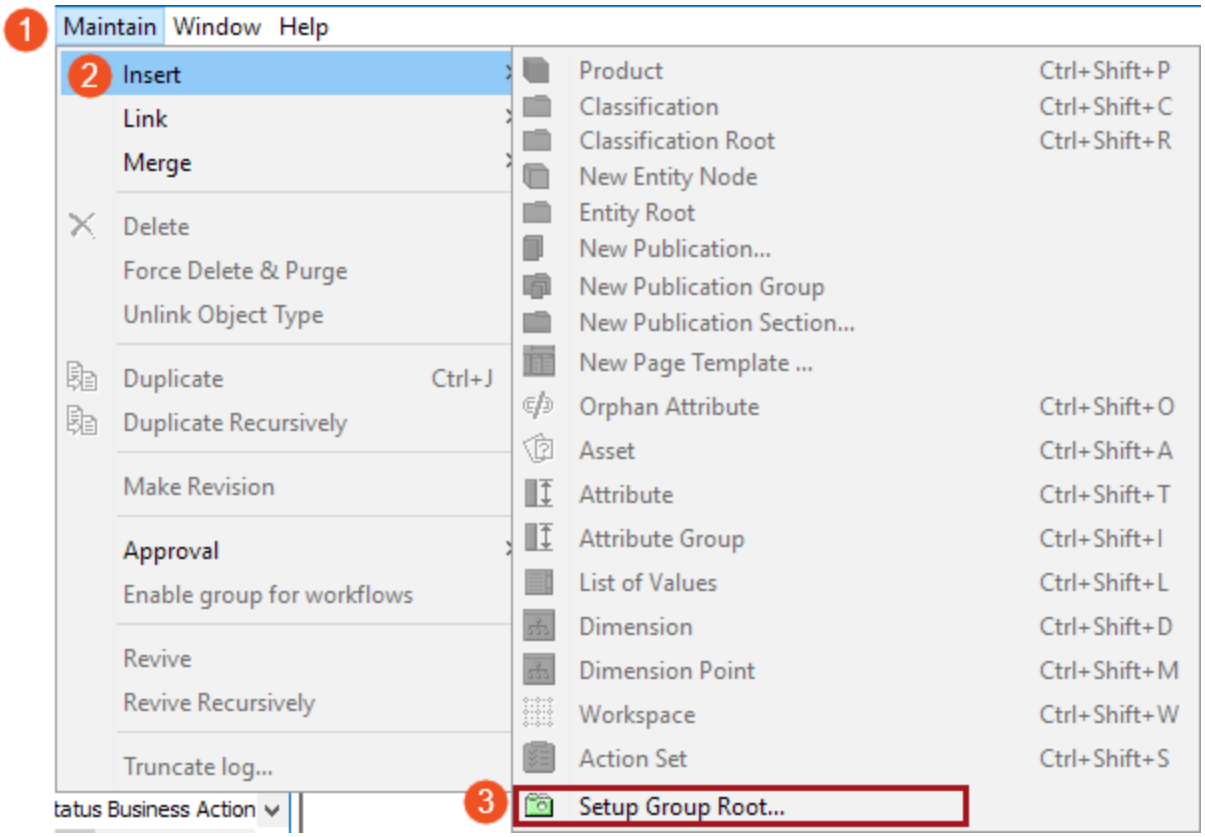
2. Under **Setup Group type root** right-click to create a **New Object Type**.



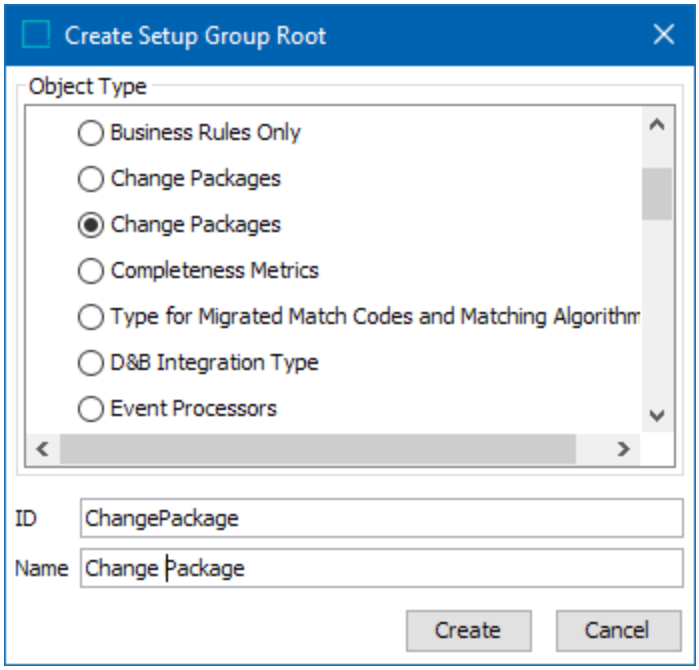
3. Under **Basic Object Types** node, return to the Change Package object.
4. On the References tab, click **Add Parent** link. Select the folder you created for change packages under **Setup Group type root** and click the Select button.



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.

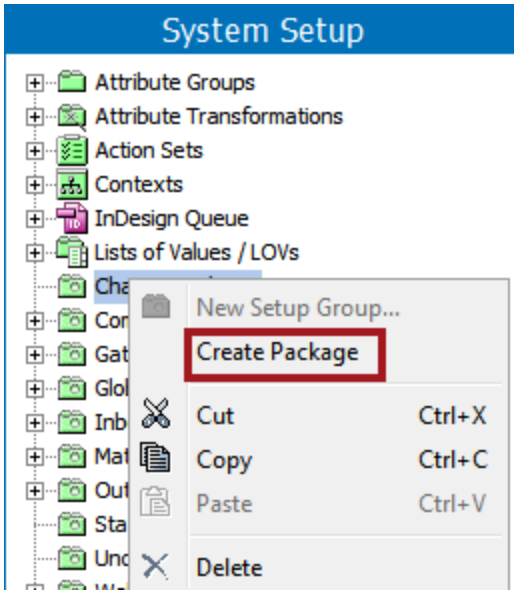


Refer to the **Creating a Change Package** topic for details on the next step.

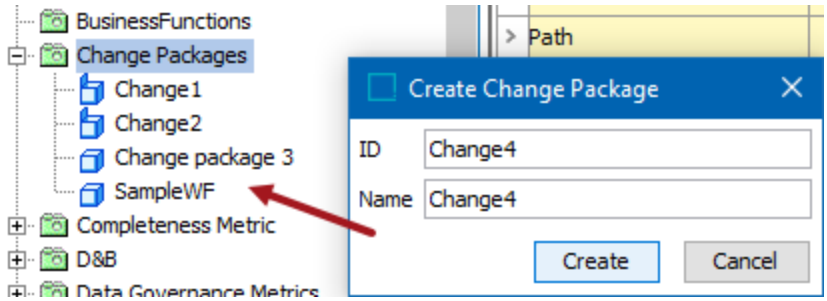
Creating a Change Package

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 of this guide.

Editing a Change Package

A change package serves as a container to store a set of system configurations for migration to another system. Therefore, once a change package has been created, it is considered empty until one or more items have been added to it. When objects have been added, the system then tracks whether or not subsequent changes occur on those items. Information on the change package informs the user of whether or not an item in the change package is up to date when compared to the current system configurations. Users then have the option to resolve discrepancies. Details for working with open change packages are described below.

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

Users in Change Packages

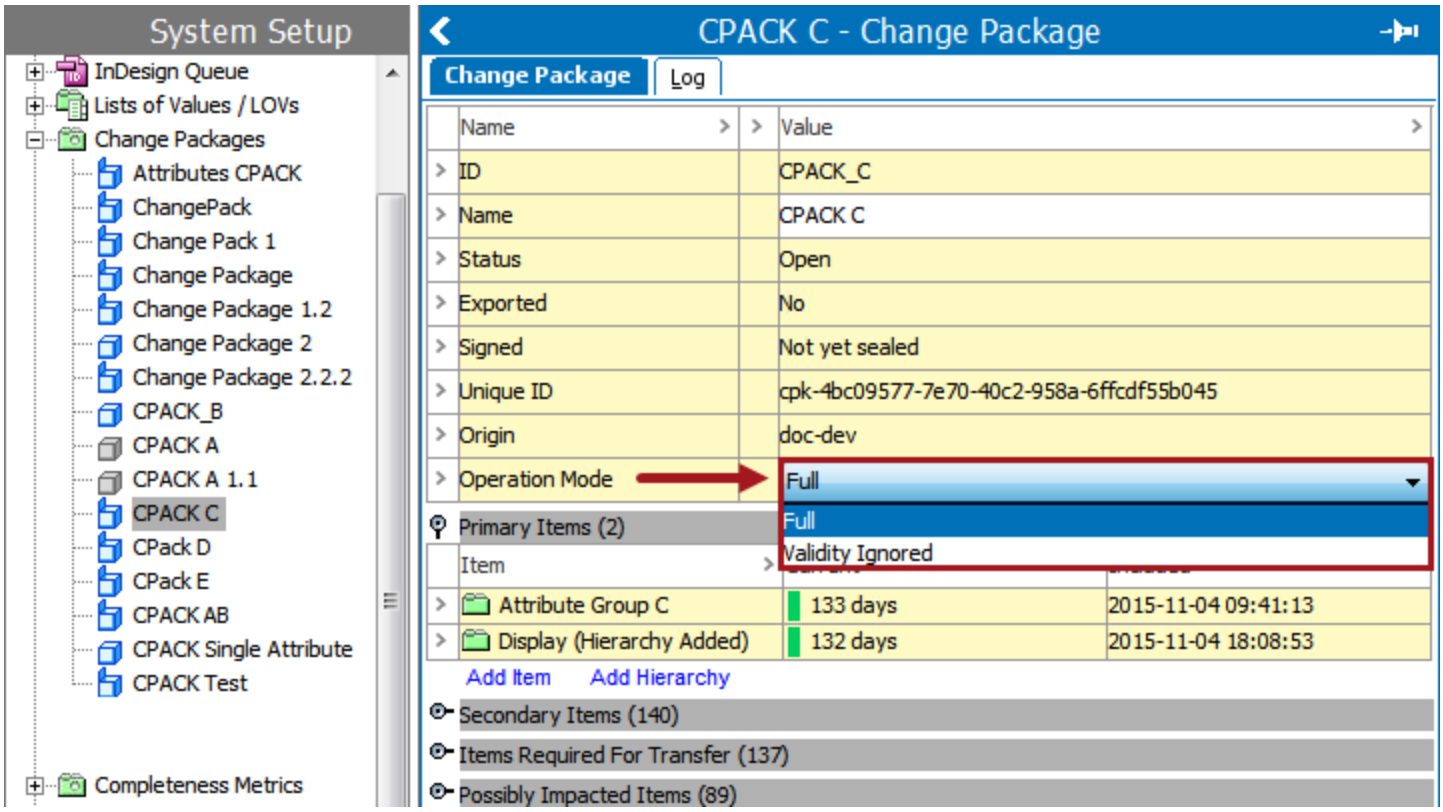
When users and/or user groups are added to a change package, passwords for the users are not included. Since creating a new user requires a password, new users cannot be created via STEPXML import. However, changes to existing users can be imported.

> **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.

Importing a change package that includes users is only successful for users that already exist in the target (receiving) system. Importing users that do not already exist results in the message: 'The tag <user> requires the attribute Password. The user with ID <id> was skipped.'

Set the Operation Mode

The Operation Mode determines how the dependency analysis will function for a given change package.

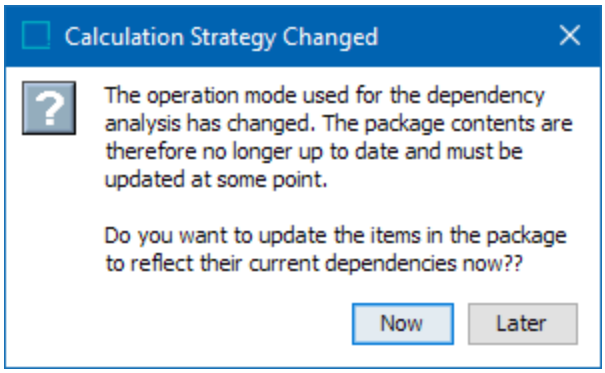


Set the Operation Mode as follows:

- **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 option 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, whereas they would be in 'Full' mode.

> **Important:** Use caution when running a change package in **Validity Ignored** operation mode. With this mode, installing the change package on a target can have an uncertain outcome. For example, if business rules are being moved from one system to another via the change package, any binds associated with those business rules are ignored, which can result in the business rules having a larger effect than intended on the receiving system. The same holds true for attribute validations.

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.



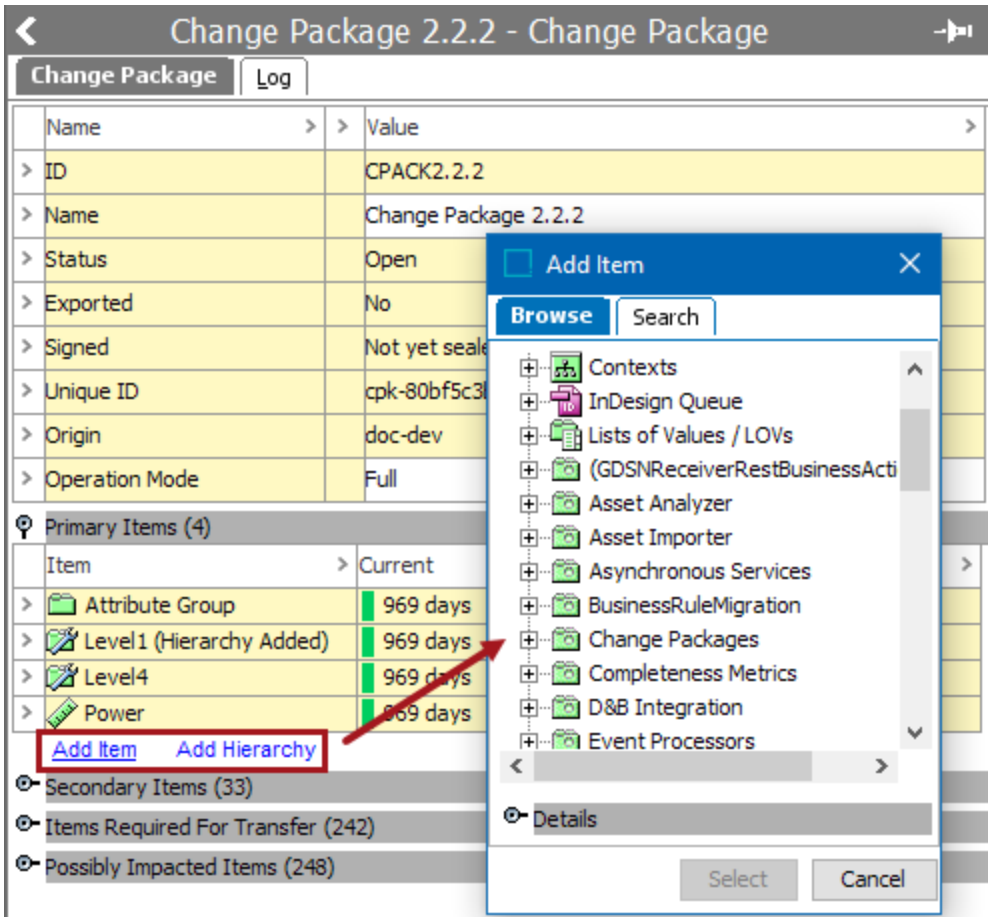
Add Items to a Change Package

You can select any number of objects to be added to the package using the **Add Item** and **Add Hierarchy** links under the Primary Items flipper as defined below.

1. On the Add Item dialog, select an item and click the **Select** button.

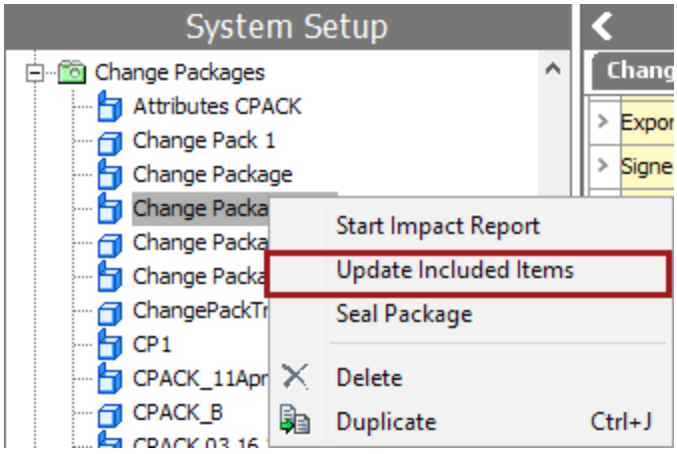
   - **Add Item** adds a single primary item to a change package. For example, selection of an attribute group will add the attribute group only and no child attributes of the group.
   - **Add Hierarchy** adds an object and *all* children of that object (direct children and beyond). The selected object is added as a primary object, and all children are added as secondary objects. For example, selection of an attribute group will add the attribute group as a primary item, and all children of the group as secondary items. If the group includes other attribute groups, those groups and their attributes will also be included as secondary items.

   Note: Selecting an invalid object type reports the issue in red text at the bottom of the Add dialog and the Select button is not enabled.



Added items are displayed in the Primary Items list.

2. Right-click on the change package and select 'Update Included Items' to ensure an accurate report of the package dependencies.

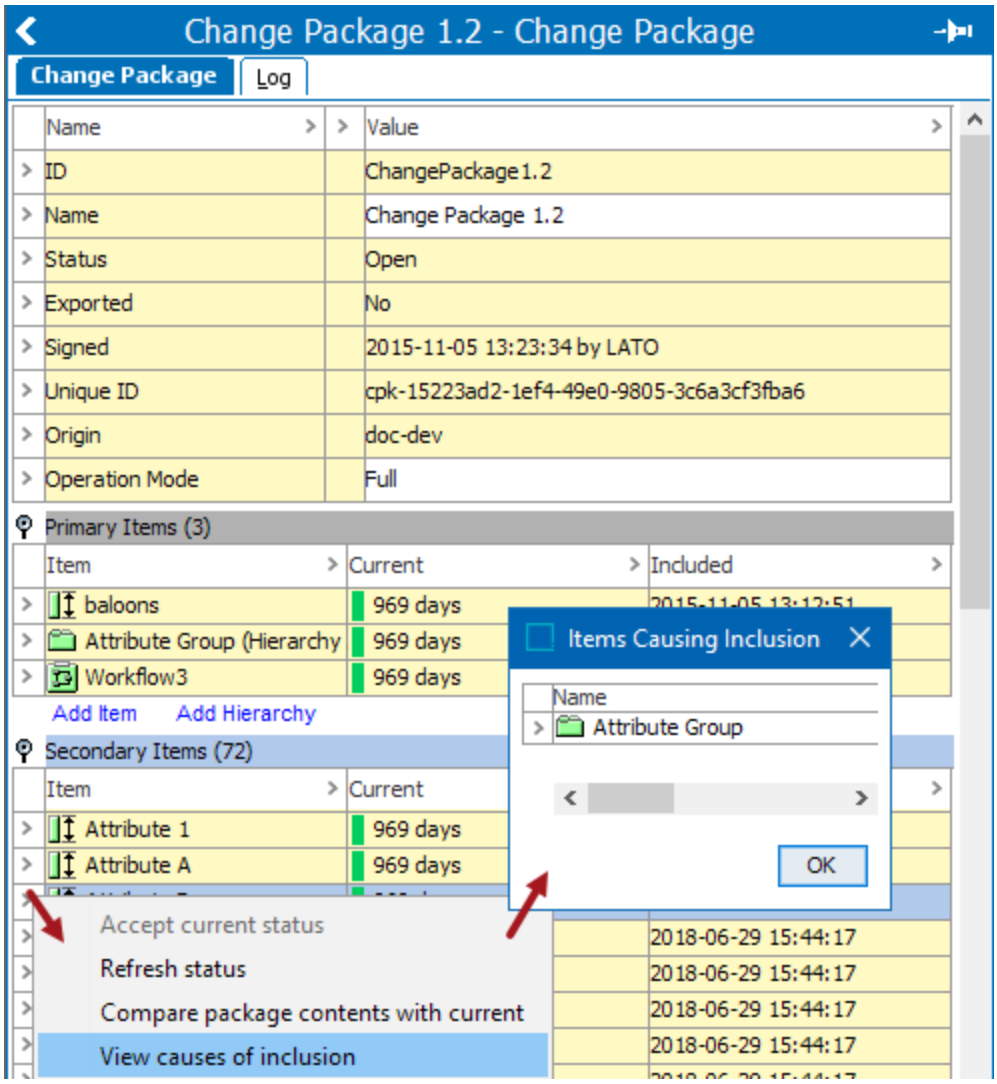


> **Note:** In order to allow for easy addition of primary objects, full dependency calculations are only applied on demand rather than running a potentially complex analysis for each individual addition or removal of an object.

3. 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 cause of inclusion' option.



Ignore Auto-selected Objects

Users can ignore items listed in the **Items Required For Transfer** and **Possibly Impacted Items** sections.

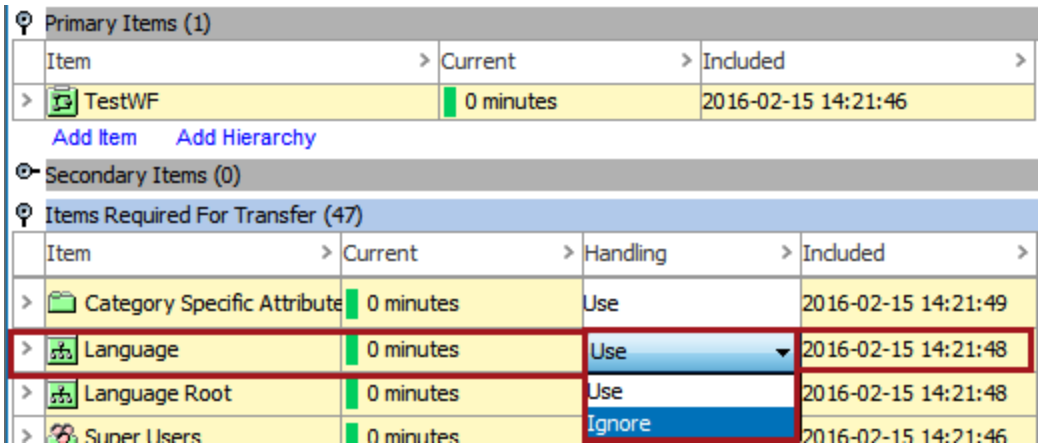
Note: Ignoring an item does not necessarily mean that the number of included items in the change package will diminish.

For either section, in the **Handling** column for the unneeded item select Ignore. However, the results of ignoring items are different based on the section, 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 over 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.

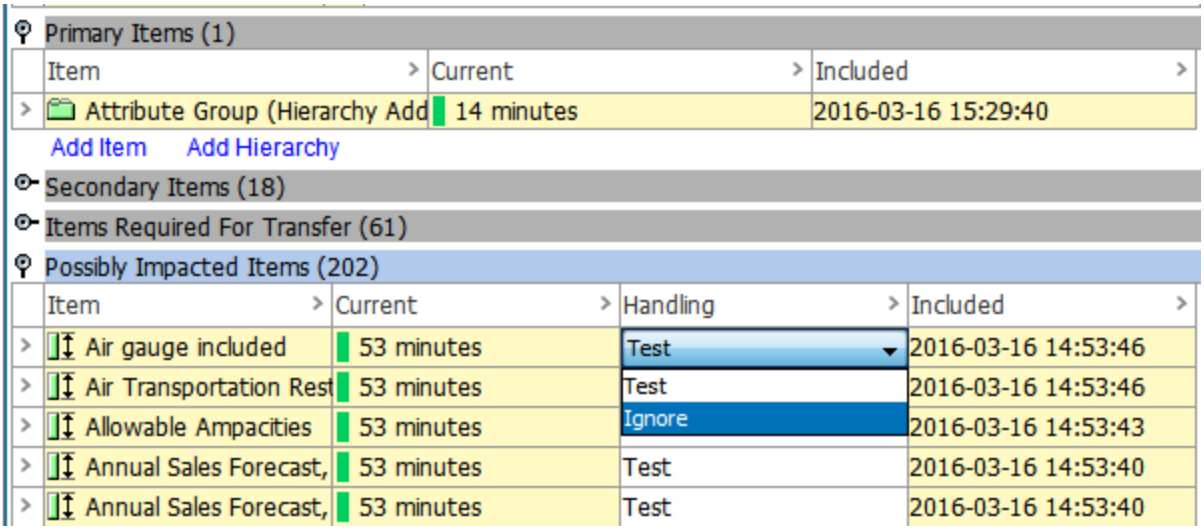
  The **Use** setting means the object transfers over to the receiving system.



- In the **Possibly Impacted Items** section the **Test** and **Ignore** settings provide communication between administrators creating change packages and those that are deploying them on target systems.

  The **Test** setting tells the installer that there are potential impacts to this object and it should be tested accordingly.

  The **Ignore** setting indicates that the package creator is confident in the outcome of the deployment and additional testing on the specified object is not needed. The possibly impacted items are not reported. For more information on impact reports, refer to the **Analyzing and Installing Change Packages** topic.

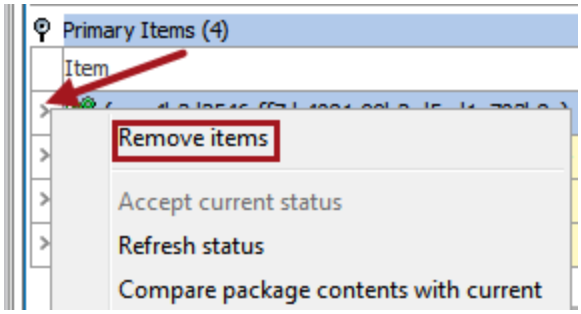


Important: When an item is deleted from the 'Primary Items' folder and then added back, the system does not retain items initially Ignored in either the 'Items Required for Transfer' or 'Possibly Impacted Items' sections. The items are added back with the re-added **Primary Item**, and must be manually set to Ignore again.

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 selecting **Remove items**.

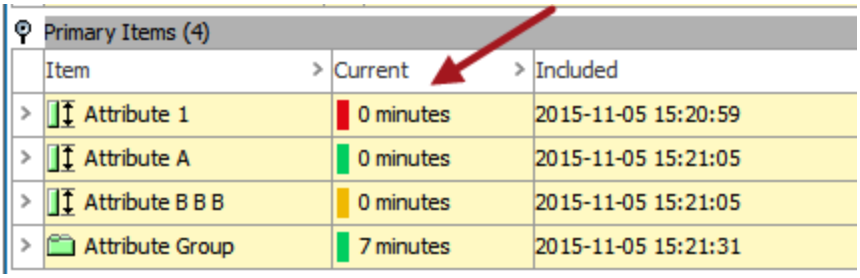


3. Right-click the change package and select the 'Update Included Items' option. This ensures an accurate report of the dependencies in the package. Full dependency calculations are only applied on demand rather than running a potentially complex analysis for each individual removal of an object.

Status and Discrepancies in Change Package Items

When an item is placed into 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.

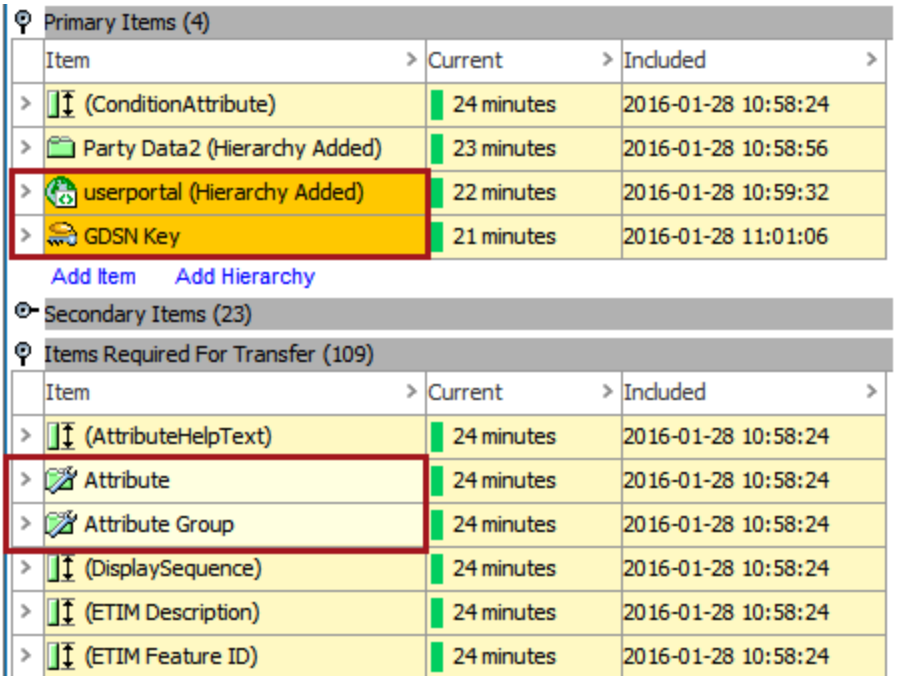
Each item in a change package has a color indicator and a notation of how long it has been since the object in the change package has been compared to the current system state.



- A **Green** indicator means the object reflected 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.

Additionally, some objects are treated differently in regards to how they are tracked. This is indicated by the background color of the objects in the Items column.

- A **Dark Yellow** highlight indicates that the objects needs to be verified manually. These are objects that cannot have their contents or details tracked, such as a Web UI, and are not reported on in the impact report.
- A **Light Yellow** highlight 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.

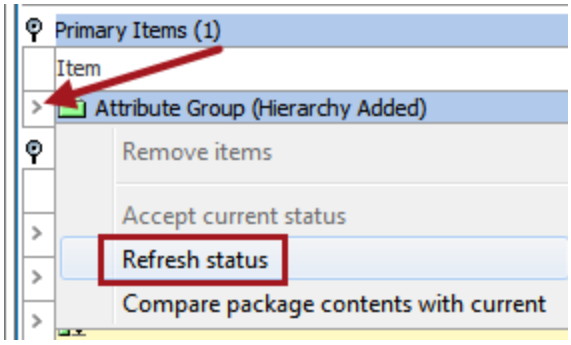


Important: At the time of sealing, the change package pulls the current system version of all objects included in the change package. Therefore, all objects will have a green indicator upon sealing of the package. Following sealing, objects can still be refreshed and if a subsequent discrepancy arises, the object will have a red indicator. However, the option to accept the change will not be available as the package has been sealed and an export of the change package will include all objects as they were at the time the package was sealed.

Refresh Status

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

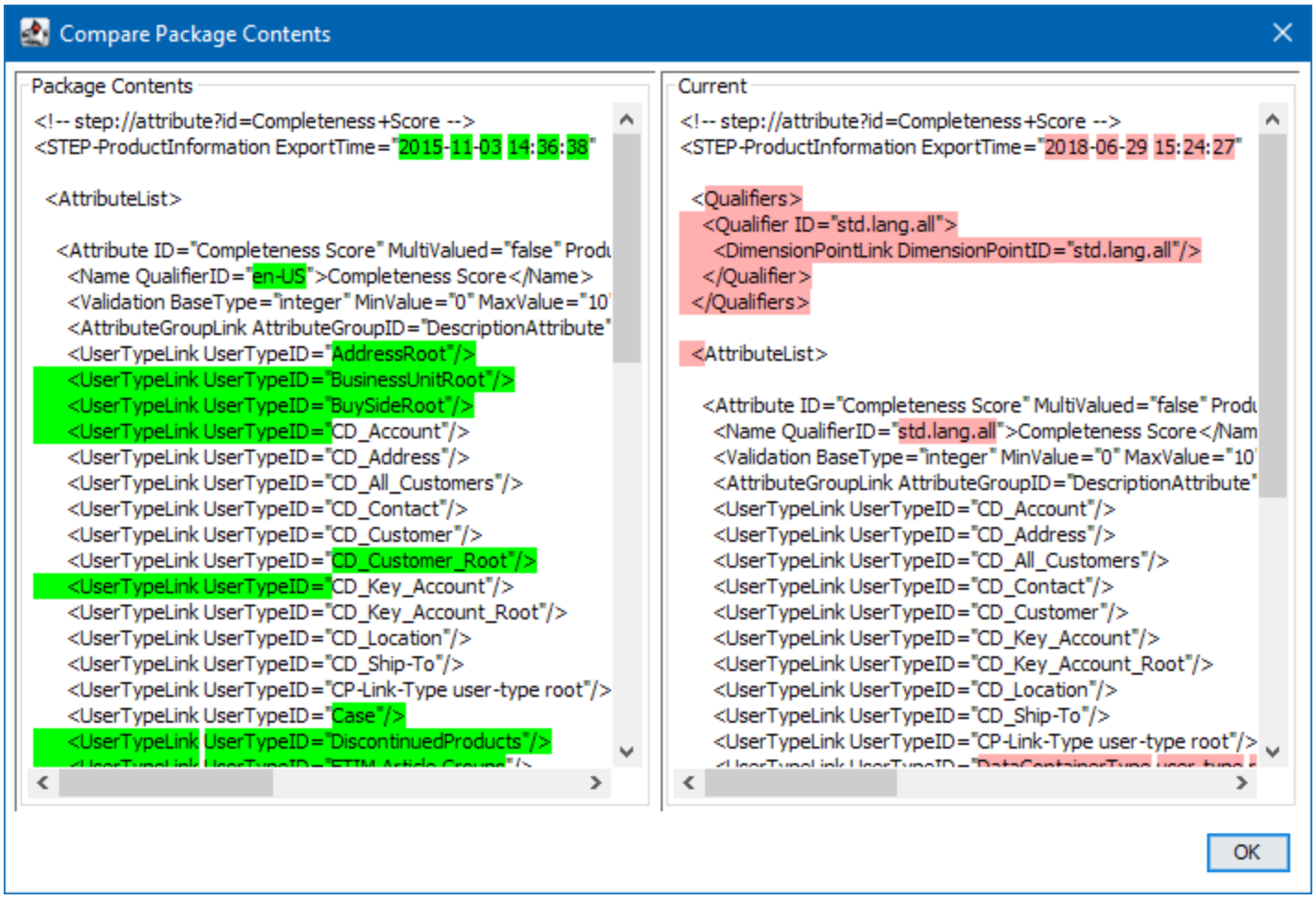
To check the status of items in the change package, click on the row arrow in the item(s) and select **Refresh Status**.



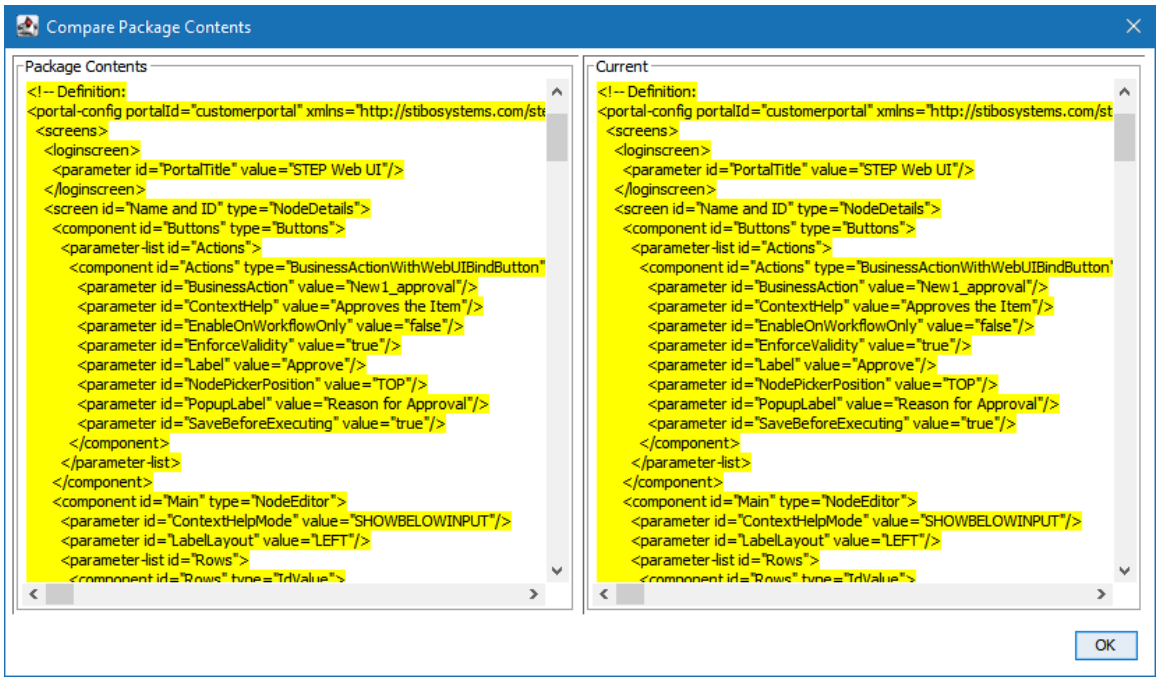
The refresh option is available on all change package objects, regardless of their current status or the status of the change package.

Compare Package Contents with Current

For a detailed comparison of a change package object (or objects) and the current system, right-click the package and select the 'Compare package contents with current' option.

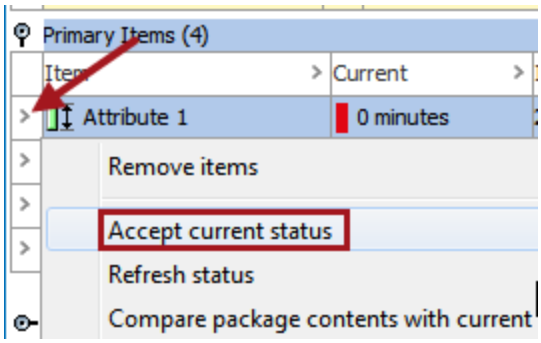


In the case of large STEPXML files, the amount of memory needed to do the comparison may become excessive, causing the comparison to be skipped. If this happens, the text is highlighted yellow (as shown below):



Accept Current Status

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



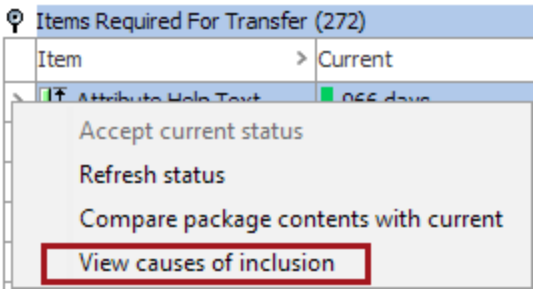
Accepting the current status of an item turns the color indicator yellow. This informs the user that the object has changed since addition to the package, but that the change has been verified and the current object is accepted as part of the package.

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

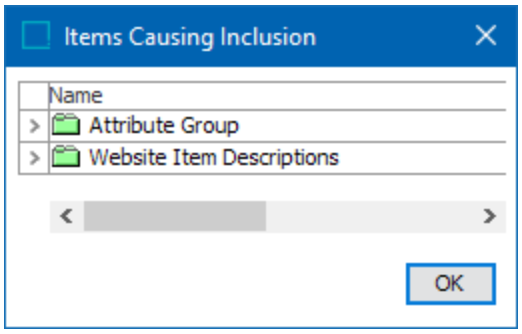
The 'View causes for inclusion' option is visible on items for the following flippers:

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



Note: You can multi-select items in the change package and run the 'View cause for inclusion.'

Select the 'View causes for inclusion' action to launch the 'Items Causing Inclusion' dialog. This shows the item (s) that is / are the cause for inclusion on the given item selected in the change package.



Note: You cannot click the object in the dialog to view the object itself.

Finalizing a Change Package

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

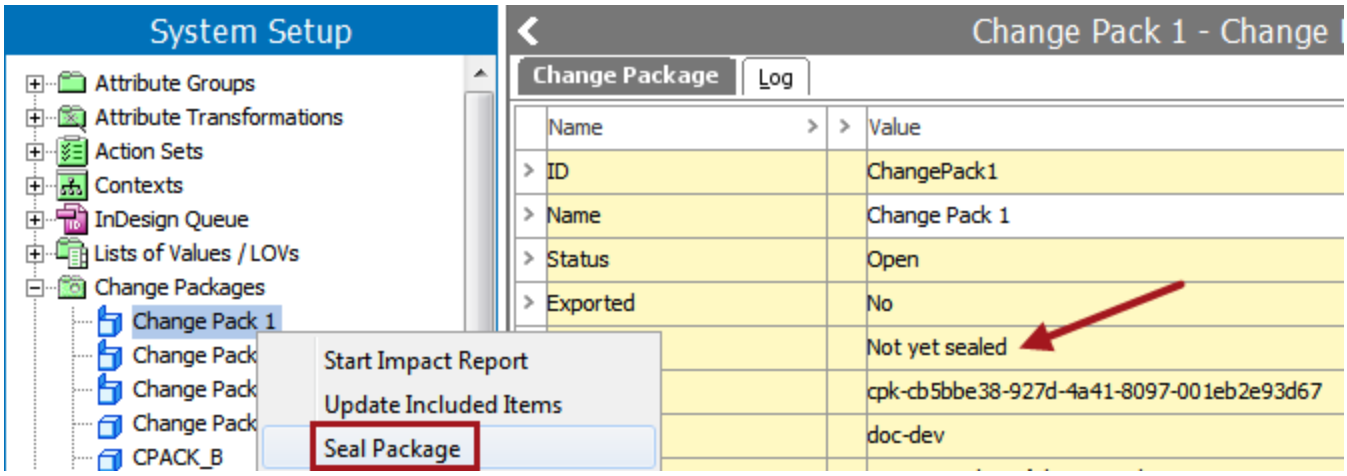
Seal a Change Package

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

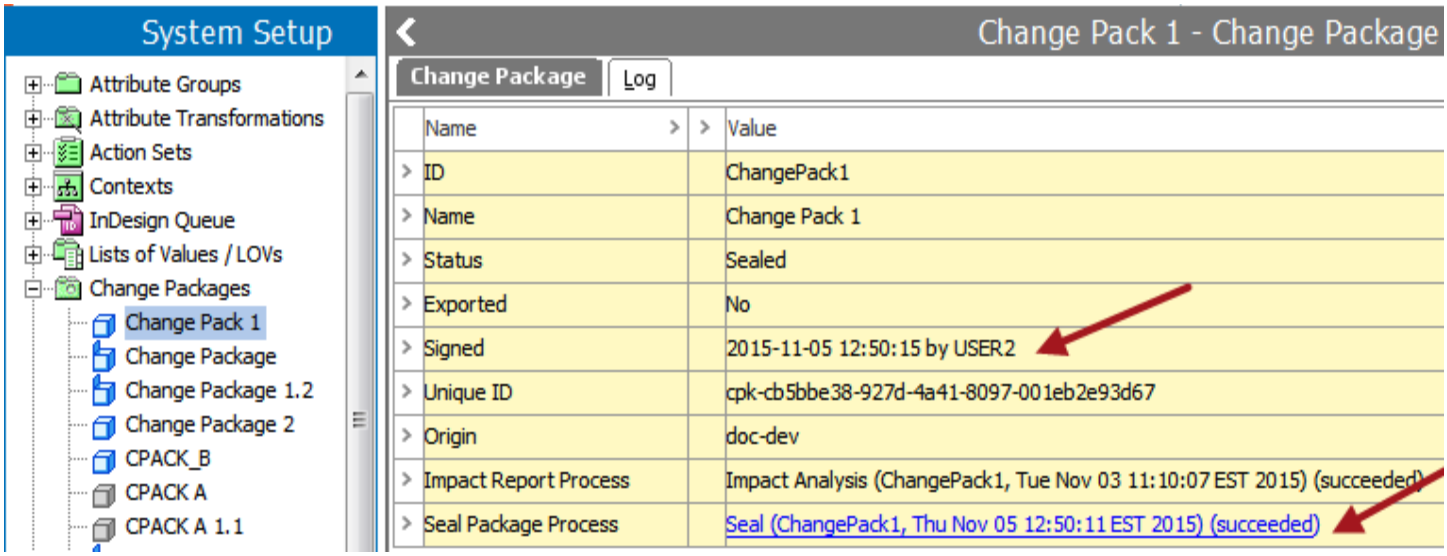
Important: Sealing a change package pulls the current system version of all objects included in the change package.

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. Right-click the package and select the **Seal Package** option.



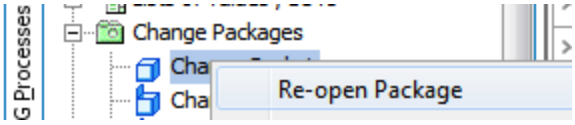
The sealed package displays a closed box icon ( ) and the Signed field indicates the date, time, and user responsible for the sealing. In addition, a link to the sealing background process is provided.



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.



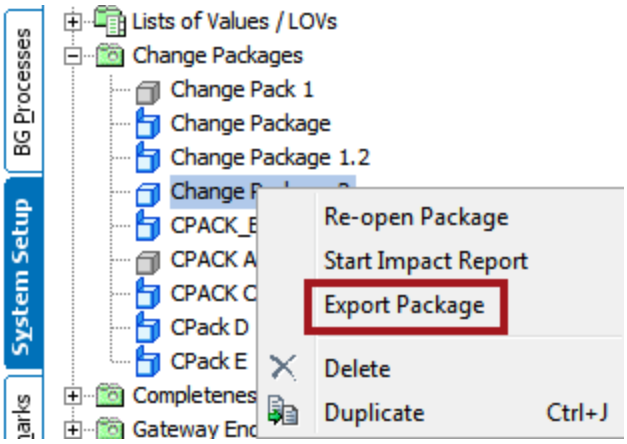
2. Follow the steps in the **Editing a Change Package** topic to modify the change package.

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 are exported using the standard Export Manager functionality, and can be imported to target systems using the Import Manager.

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

As the main purpose of a change package is to transfer configurations between systems, 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.

Importing a Change Package

Change packages are exported as encoded STEPXML files and are therefore easily imported using the Import Manager. For more information, refer to the **Import Manager** documentation.

> **Note:** It is required to 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.

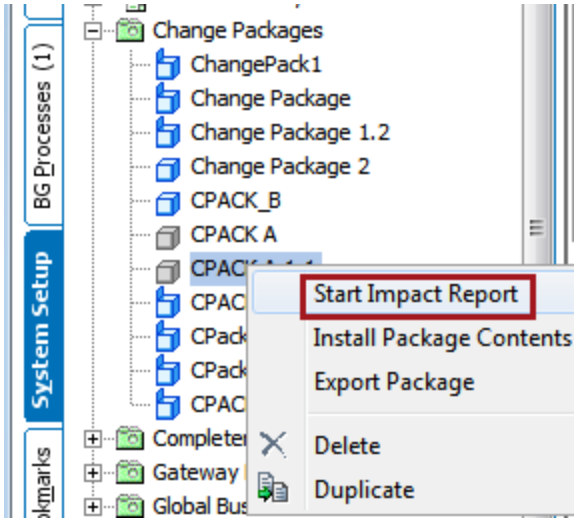
Upon import, the new change package is found in the same location on the System Setup tab as it existed on the source system. Imported change packages have a status of 'Dormant' and a gray icon:

Note that the *contents* of the change package have not yet been applied at this point. Only the change package itself has been imported, and no system configurations will be updated unless the change package is installed.

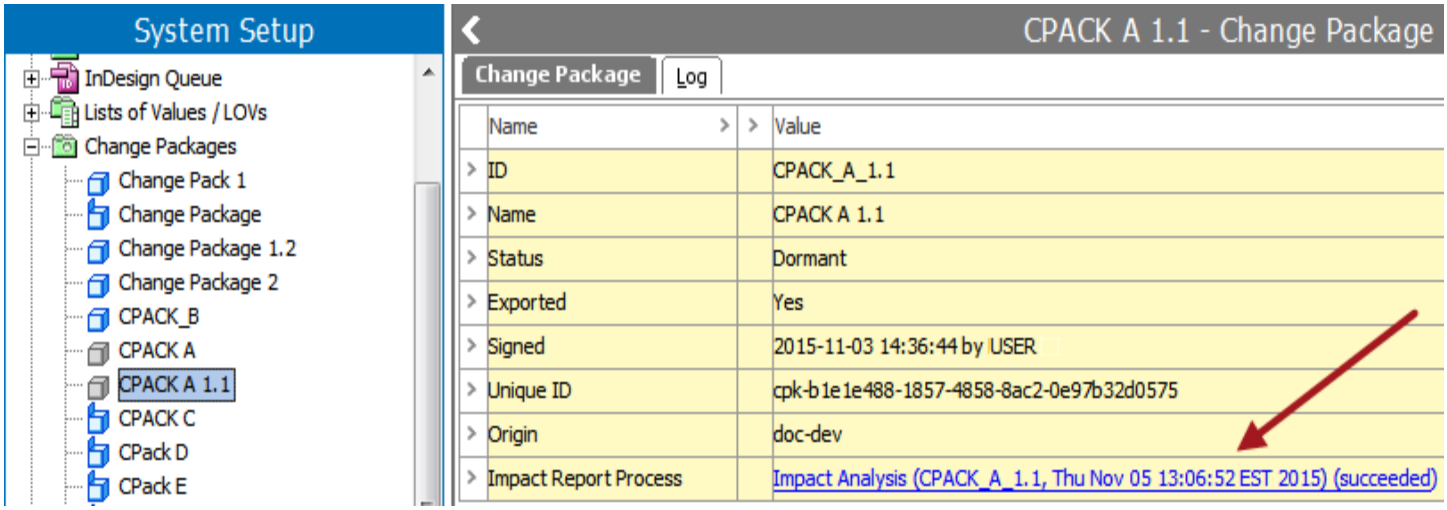
Analyzing a Change Package

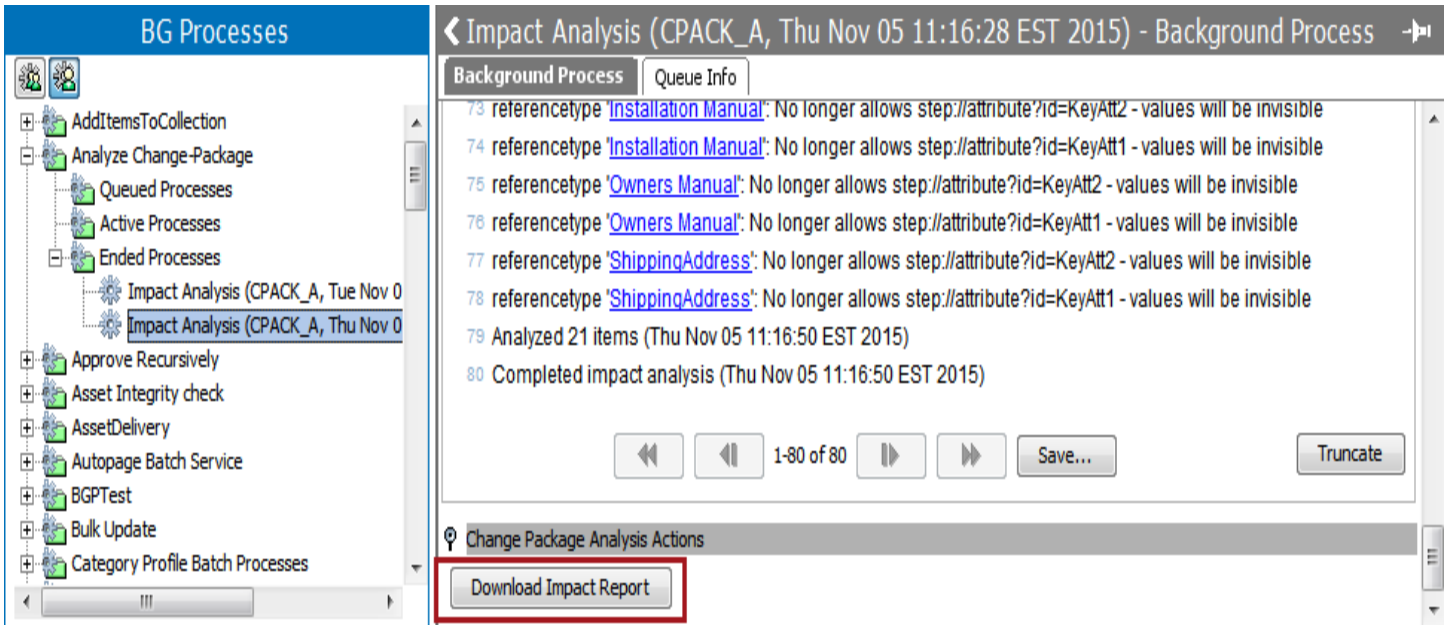
Once a change package has been imported, an impact report can be run. The impact report provides the user with a variety of information they can use to assess whether or not the change package should be installed, and what the system impacts are likely to be upon installation.

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



The impact report is run as a background process, which is then accessible on the **BG Processes** tab under **Analyze Change Package**. The contents of the report can be viewed directly in the execution report, or can be downloaded for viewing offline (e.g., in Excel). A link to the background process is also provided on the change package object.





A	B	C	D	E	F	G	H	I
Origin	Message Type	Inclusion Type	URL	Object Type	Object ID	Message	Current Status	Status Time
Detection	IdentifiedChangedItem	Precondition	step://cplinktype?id=WebsiteLink	cplinktype	WebsiteLink	Identified changed item	Out of sync	11/5/2015 11:16
Detection	IdentifiedChangedItem	Precondition	step://cplinktype?id=SupplierLink	cplinktype	SupplierLink	Identified changed item	Out of sync	11/5/2015 11:16
Detection	IdentifiedNewItem	Derived	step://attribute?id=Size	attribute	Size	Identified new item	Out of sync	11/3/2015 14:39
Detection	IdentifiedChangedItem	Precondition	step://referencetype?id=MSDS	referencetype	MSDS	Identified changed item	Out of sync	11/5/2015 11:16
Impact	MissingAttribute	Precondition	step://cplinktype?id=MerchandisingLink	cplinktype	MerchandisingLink	No longer allows step://attribute?id=KeyAtt2 - values will be invisible	Out of sync	11/5/2015 11:16
Impact	MissingAttribute	Precondition	step://cplinktype?id=MerchandisingLink	cplinktype	MerchandisingLink	No longer allows step://attribute?id=KeyAtt1 - values will be invisible	Out of sync	11/5/2015 11:16
Impact	MissingAttribute	Precondition	step://referencetype?id=PrimaryProductImage	referencetype	PrimaryProductImage	No longer allows step://attribute?id=KeyAtt2 - values will be invisible	Out of sync	11/5/2015 11:16
Impact	PropertyMismatch	Precondition	step://cplinktype?id=Classifications	cplinktype	Classifications	Changed from externally maintained to internally md. - may enter single	Out of sync	11/5/2015 11:16

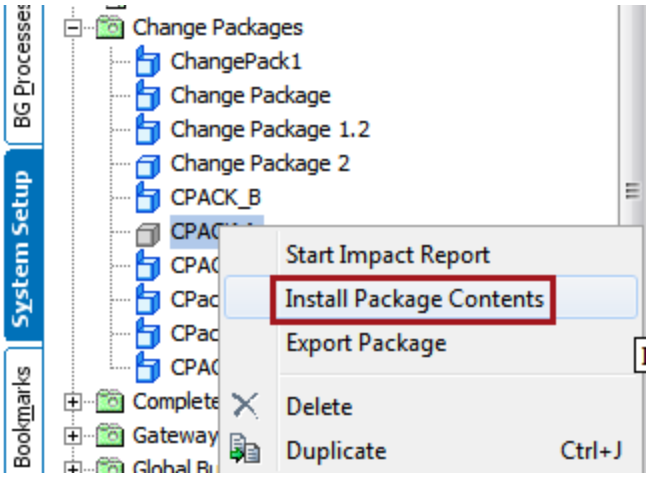
The impact report should be analyzed to determine whether or not the change package should be installed, and if any changes should be carried out on the target system prior to installation. If system changes occur, it may be useful to re-run the impact report.

If it is determined that the change package should not be installed, it can be removed from the system by right-clicking on the change package and selecting **Delete**.

Note: Deleting a change package removes it from the system entirely. It is not available in the Recycle Bin and can only be accessed again via re-import of the package.

Installing a Change Package

When the impact report has been reviewed and the change package determined acceptable, it is installed by right-clicking and selecting **Install Package Contents**.



Installation of the change package means that all objects within the change package are added to the system. If objects in the change package existed previously on the system, they will be updated to reflect the contents of the package.

Considerations

- The deletion of system configurations is not supported. Import of change packages supports configuration additions only.
- Event queues and IEPs are imported as disabled and must be manually enabled.

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 will only add / modify configuration on target systems it will not delete what should not exist on the target system. It will not make updates that require user input; this is explained in the document in more detail 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

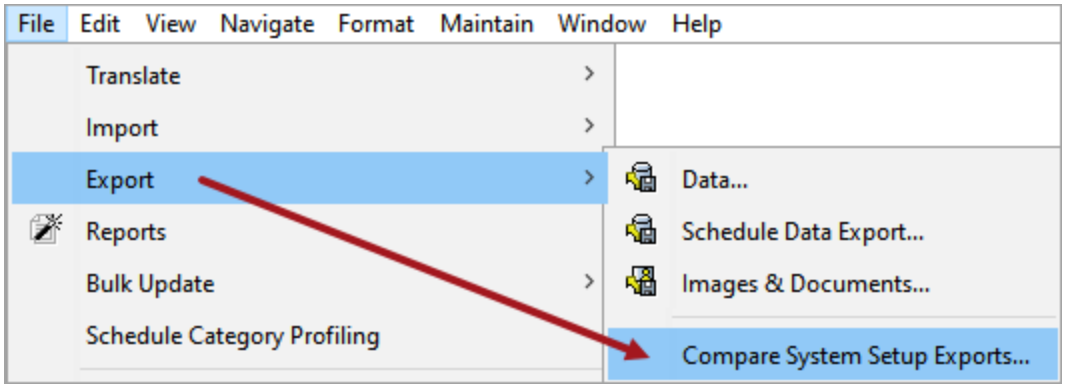
Typically this type of work requires a Super User to carry out the task. The user would require in depth knowledge of STEPXML, System Setup, and how to use export and import manager.

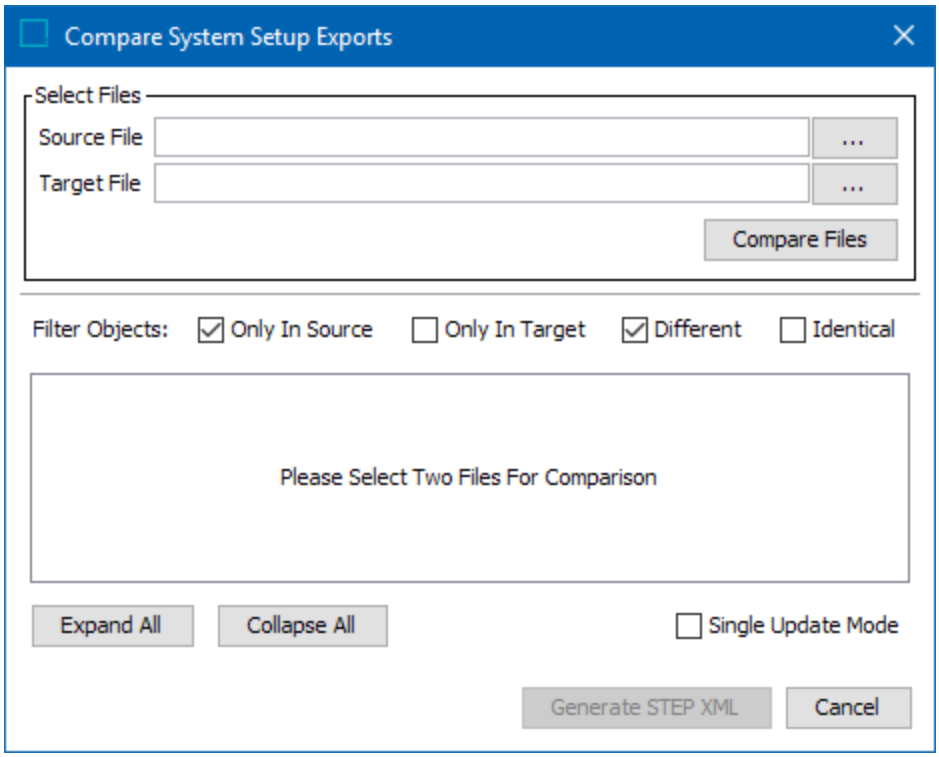
Accessing the System Compare Tool

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

It is possible to move configuration from one STEP instance to another without using the comparison tool. This process will only add / modify configuration loaded onto a target system. If you then need to identify what is different from the source and target machines the comparison tool will need to be used.

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

What configuration can be moved from one 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 **Action Sets** in **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** documentation. • Select **None** to output no asset push event queues.
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** documentation. • Select **None** to output no asset push configurations.
Include Attributes	For attributes, the validation base types, dimension dependencies applied units, applied LOV hierarchy filters, calculated templates, and so on, are exported as follows: • Select **All** to output all attributes, including fundamental system specific attributes (having an ID that starts with **stibo.** or **asset.**). • Select **Selected** or **Minimum** to output attributes based on the explanation in the **Minimum, Referenced, and Selected in STEPXML** documentation. • Select **None** to output no attributes. For more information, refer to **Attributes** in **System Setup** documentation.
Include Attribute Groups	For attribute groups, the view definitions and settings applied to attribute groups are exported as follows:

Configuration Parameter	Description
	• Select **All** to output all attribute groups. • Select **Selected** or **Minimum** to output attribute groups based on the explanation in the **Minimum, Referenced, and Selected in STEPXML** documentation. • Select **None** to output no attribute groups. For more information, refer to **Attribute Groups** in **System Setup** documentation.
Include Attribute Transformations	Select **No** or **Yes** to control output of attribute transformations and their configurations. For more information, refer to **Attribute Transformations** in **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** documentation. • Select **No** to output no bulk update configurations. For more information, refer to **Bulk Updates** documentation.
Include Business Rules (Global) and Libraries	For global business rules and libraries, the ID, name, links, on approve setting, dependencies, valid object types, and templates are exported as follows: • Select **All** to output all global business rules and libraries. • Select **Selected** or **Referenced** to output attributes based on the explanation in the **Minimum, Referenced, and Selected in STEPXML** documentation. • Select **None** to output no global business rules or libraries. Templates are exported in Base64 and can only be imported without modifications into another STEP system. For more information, refer to **Business Rules** documentation.
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 **Collections** in the **Getting Started** documentation.
Include	Select **No** or **Yes** to control output of details of component models, including ID, name,

Configuration Parameter	Description
Component Models	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 **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 **Contexts** in **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 **Contexts** in **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** documentation. • Select **None** to output no data container types. For more information, refer to **Data Containers** in **System Setup** documentation.
Include Derived Event Types	Select **No** or **Yes** to control output of objects created in System Setup as Derived Event Type. For more information, refer to **Derived Events** in **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 **eCatalogs**.
Include Event Processors	Control output of event processor information, including ID, name, links, and configuration. • Select **All** to output all event processors. • Select **Minimum** to output event processors based on the explanation in the **Minimum, Referenced, and Selected in STEPXML** documentation.

Configuration Parameter	Description
	• Select **None** to output no event processors. For more information, refer to **Event Processors** in **System Setup** documentation. The configurations are exported in Base64 and can only be imported without modifications into another STEP system.
Include Event Queues	Select **No** or **Yes** to control output of event queues and applied message templates, output formatting, and triggering definitions.
Include Export Configurations	Assets saved as export configurations are exported as follows: • Select **All** to output all export configurations. • Select **Minimum** to output export configurations based on the explanation in the **Minimum, Referenced, and Selected in STEPXML** documentation. • Select **None** to output no export configurations. For more information, refer to **Maintaining a Saved Export Configuration** in **Data Exchange**.
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** documentation. • Select **None** to output no image conversion configurations.
Include Import Configurations	Assets saved as import configurations are exported as follows: • Select **All** to output all import configurations. • Select **Minimum** to output import configurations based on the explanation in the **Minimum, Referenced, and Selected in STEPXML** documentation. • Select **None** to output no import configurations. For more information, refer to **Maintaining a Saved Import Configuration** in **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.

Configuration Parameter	Description
	• 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** documentation. • Select **None** to output no integration endpoints. For more information, refer to the integration endpoint topics in **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** in **System Setup** documentation.
Include Link, Reference and Object Types	**Important:** Use caution when handling reference types and object types with an ID that start with **stibo.** since they are fundamental objects. User-created object types created below the 'Setup Group type root' node and system-specific object types, link types, and system specific reference types are exported as follows: • Select **All** to output all edge, reference, and object types. • Select **Selected** or **Minimum** to output only link, reference, and object types in the selected hierarchy based on the explanation in the **Minimum, Referenced, and Selected in STEPXML** documentation. • Select **None** to output no link, reference, and object types. For more information, refer to **Reference and Link Types** in **System Setup** documentation. **Note:** Edge types are identified with either 'PA' (product to attribute link type) or 'CA' (classification to attribute link type).
Include List of Values	LOV definitions and values in LOVs, including applied meta attributes, validation base types, and applied units are exported as follows: • Select **All** to output all LOV definitions and values. • Select **Selected** or **Minimum** to output LOV definitions and values based on the explanation in the **Minimum, Referenced, and Selected in STEPXML** documentation. • Select **None** to output no LOV definitions or values.

Configuration Parameter	Description
	For more information, refer to **List of Values (LOVs)** in **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 **Matching, Linking, and Merging** documentation.
Include Matching Algorithms	Control output of matching algorithms, including ID, name, links, and configuration. For more information, refer to **Matching, Linking, and Merging** documentation. • Select **All** to output all matching algorithms. • Select **Selected** to output matching algorithms based on the explanation in the **Minimum, Referenced, and Selected in STEPXML** documentation. • Select **None** to output no matching algorithms. The configurations are exported in Base64 and can only be imported without modifications into another STEP system.
Include Setup Entities	Control output of setup entities including ID, name, links, and configuration. The configurations are exported in Base64 and can only be imported without modifications into another STEP system. • Select **All** to output all setup entities. • Select **Selected** to output setup entities based on the explanation in the **Minimum, Referenced, and Selected in STEPXML** documentation. • Select **None** to output no setup entities.
Include Setup Groups	Control output of setup groups that are defined in system setup to hold integration endpoints, Web UI configurations, STEP workflows, and business rules. • Select **All** to output all setup groups. • Select **Selected** to output setup groups based on the explanation in the **Minimum, Referenced, and Selected in STEPXML** documentation. • Select **None** to output no setup groups. 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 **Setup Groups** in **System Setup** documentation.
Include Status	Workflow status flags are exported as follows:

Configuration Parameter	Description
Flags	• 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** documentation. • Select **None** to output no workflow status flags. For information, refer to the **Status Flags** topic in **Workflows** documentation.
Include System Settings	Select **No** or **Yes** to control output of the following default settings defined on the Users and Groups root node: • Image & Document Settings flipper > Dimension Dependencies • Calculated Attribute Settings flipper > Dimension Dependencies • Product Information Manager Default Settings flipper • Flatplanner Default Settings flipper • DTP Default Settings flipper • GDSN Default Settings flipper • Terms List Settings flipper • WebServices Default Settings flipper • Web UI Default Settings flipper • Default Reference Type of Primary Image flipper • Table Default width and height flipper For more information, refer to **System Settings** in **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** documentation. • Select **None** to output no table types. For more information, refer to **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 **Tags** in **System Setup** documentation.

Configuration Parameter	Description
Include Transformation Lookup Tables	Assets saved as transformation lookup table configurations are exported as follows: • Select **All** to output all transformation lookup table configurations. • Select **Minimum** to output transformation lookup table configurations based on the explanation in the **Minimum, Referenced, and Selected in STEPXML** documentation. • Select **None** to output no transformation lookup table configurations. For information, refer to the **Transformation Lookup Tables** topic in **Resource Materials** online help.
Include Translation Configurations	Saved translation configurations are exported as follows: • Select **All** to export all saved translation configurations. • Select **Selected** to export selected saved translation configurations based on the explanation in the **Minimum, Referenced, and Selected in STEPXML** 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.
Include Units	Unit groups and units, including applied meta attributes, values, and base unit conversions are exported as follows: • Select **All** to output all unit groups and units. • Select **Selected** or **Minimum** to output unit groups and units based on the explanation in the **Minimum, Referenced, and Selected in STEPXML** documentation. • Select **None** to output no unit groups or units. For more information, refer to **Units** in **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.

Configuration Parameter	Description
	• 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** documentation. • Select **None** to output no users or user groups. For more information, refer to **Users and Groups** in **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** documentation. • Select **None** to output no Web UI configurations. For more information, refer to **Managing Web UI Configurations** in the **Web User Interfaces**.
Include Websites	Select **No** or **Yes** to control output of websites, with and without workspace. The legacy 'Web Publisher' used this option to publish data incrementally from STEP. OIEPs provide improved functionality for this same purpose. For more information, refer to the **Data Exchange** 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** documentation. • Select **None** to output no workflows. For more information, refer to **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.

Configuration Parameter	Description
	• Select **Selected** to output workflow profiles based on the explanation in the **Minimum, Referenced, and Selected in STEPXML** documentation. • Select **None** to output no workflow profiles. For more information, refer to **Monitoring Workflows** in **Workflows** documentation.

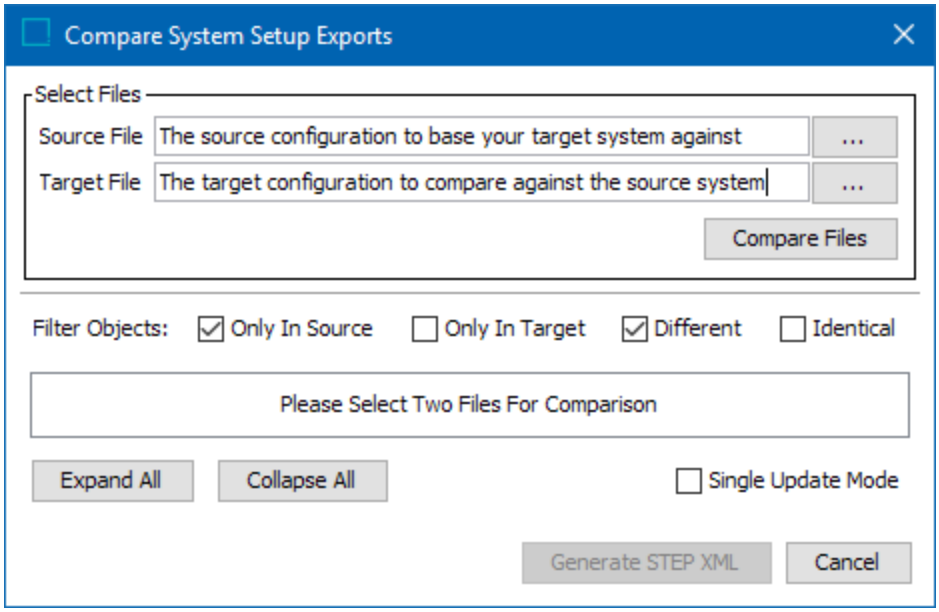
Using the STEPXML Comparison Tool

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

Select Source and Target configuration file

Once a user has logged into the comparison tool they will firstly need to select XML exported from the source system and XML exported from the target system as illustrated below:

1. Select the exported XML file from the source system
2. Select the exported XML file from the target system



Select how you want to filter the configurations

Once the XML files have been selected the user will need to select 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

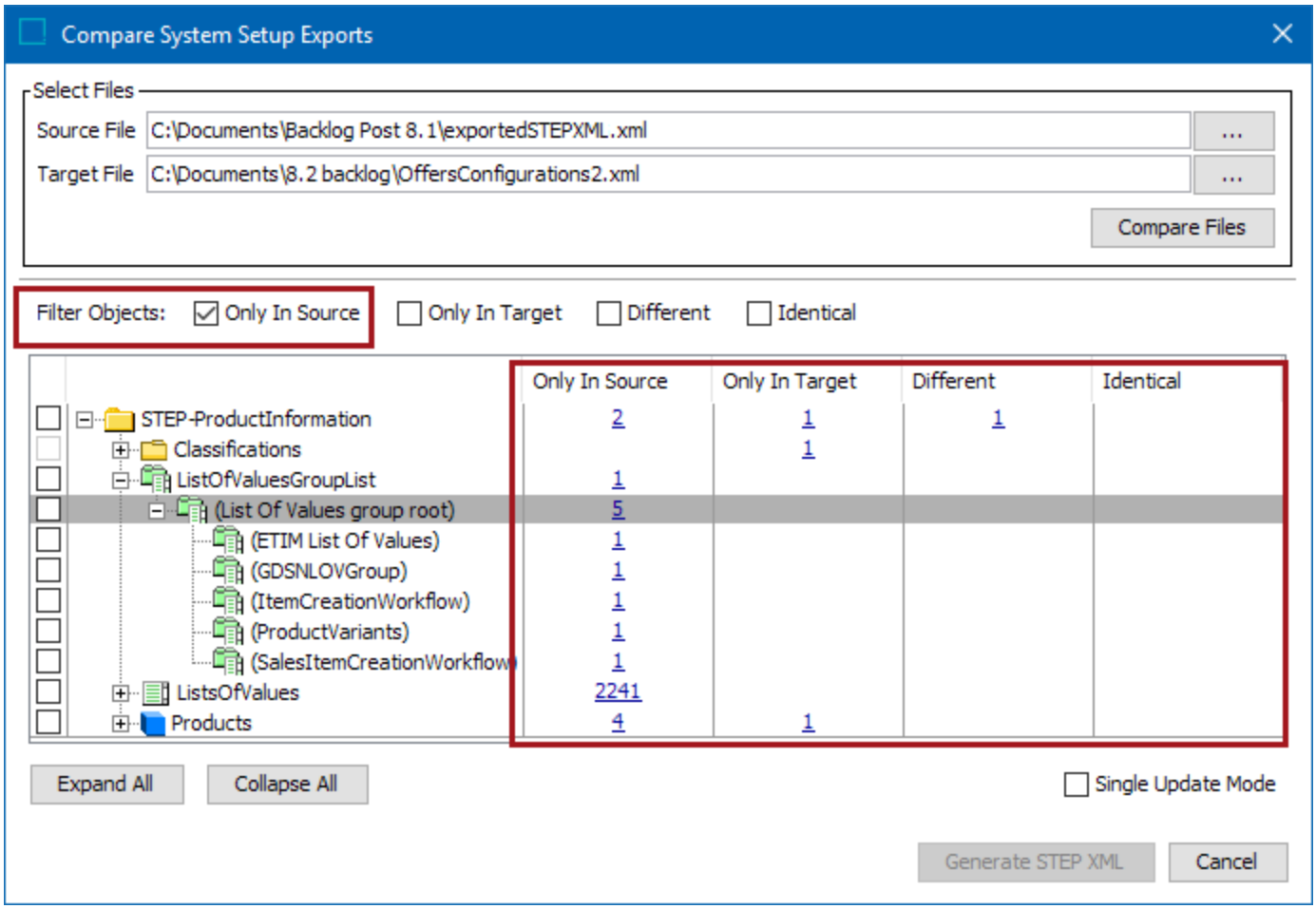
If you need to change the filter options selected, change them and click Compare Files to update

Viewing configuration differences when 'Only In Source' is selected for example

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

The comparison tool will give an overview of the following:

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



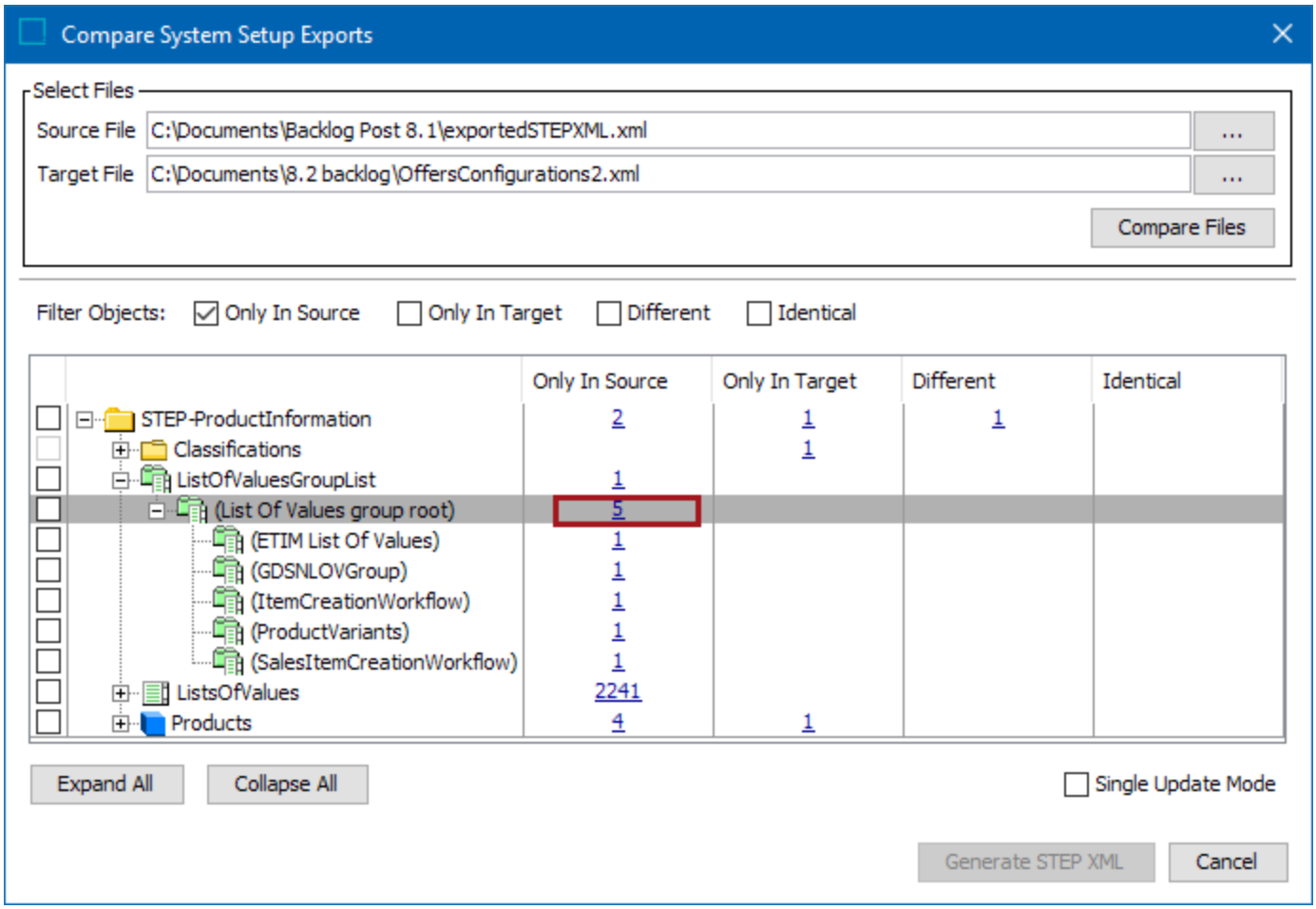
In this example the user has selected to filter on 'Only In Source', when the user opens the attribute list configuration which indicates one attribute the user is shown the attributes that appear in the source system only.

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

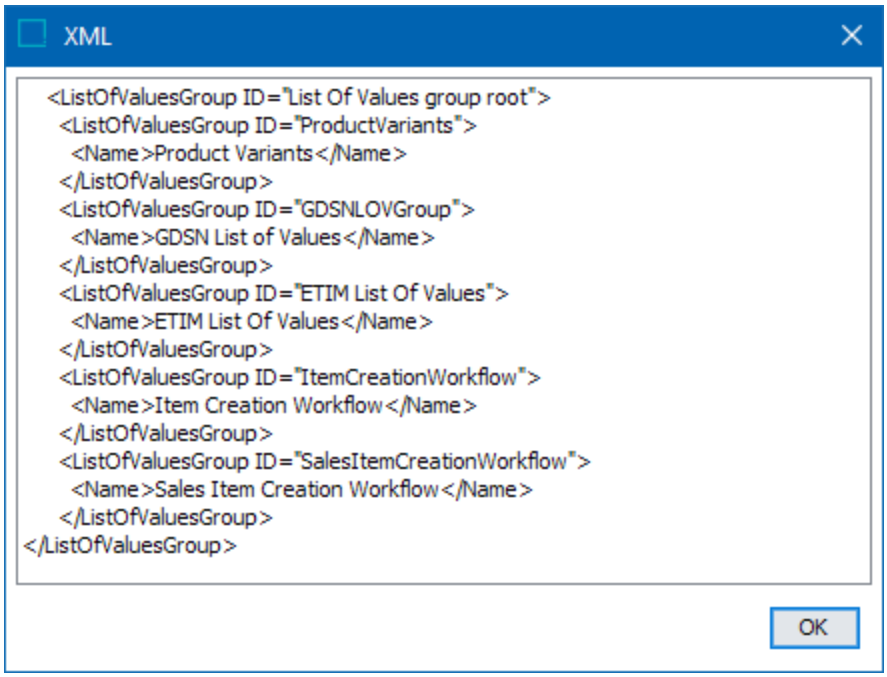
Viewing the differences in the comparison tool

Within the comparison tool users are able to view what the difference is 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 are able to view these differences by selecting the hyperlink for each difference



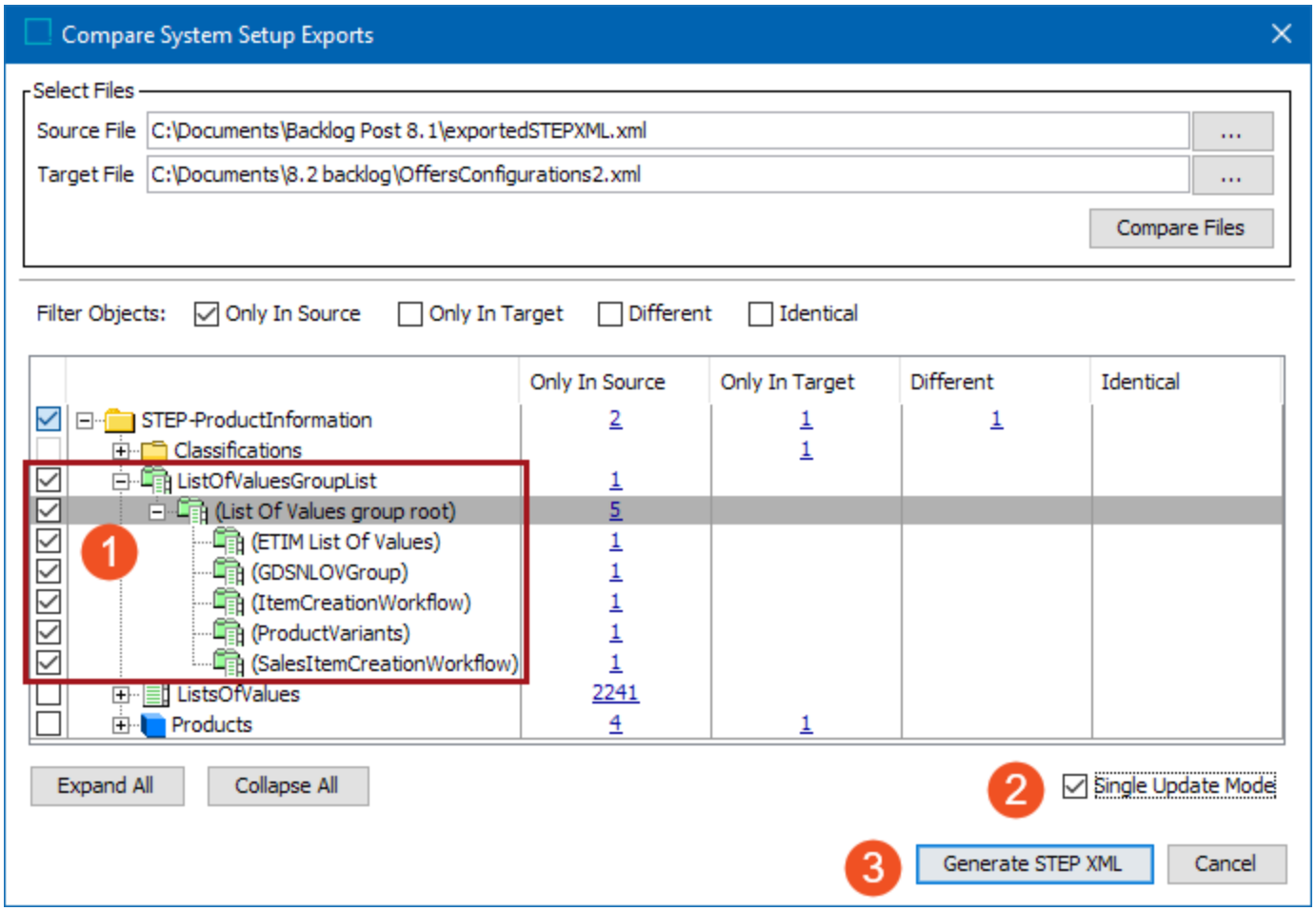
If you select the hyperlink highlighted in screenshot above you will be shown the STEPXML for the List Of Values group that only exists in the source system as follows:



Generate STEPXML with the configuration differences

Finally the user will need to select the check boxes to identify the configuration required to be exported and then generated an XML file based on this selection.

1. Select the configuration that needs to be moved to the target system. In this case it is the List Of Values Group List.
2. Check Single Update Mode option
3. If STEPXML generated needs to make updates that require single update mode this option must be checked. Examples of what puts a system into Single Update Mode are given later in the document
4. Generate STEPXML by hitting the Generate STEPXML button



5. Save XML file and name appropriately

STEPXML Comparison Tool Scenarios

The following examples highlight how the STEPXML Comparison Tool can be used.

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 only have read-only access to the system.

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

If the XML being loaded is not set to go in to Single-Update Mode when imported the process will highlight it is required to go into Single-Update Mode 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** section
- 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 value unless it has a non auto-generated ID
- If attributes have been merging on the source system the redundant attribute on the target system will need to be manually removed
- If List of values have been merged on the source system these redundant values will need to be removed / merged on the target system
- Swapping 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
- Remove Workspaces – manual task
- Cannot change an attribute to have LOV validation or not to 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

Via a set of outbound integration endpoint plugins, it is possible to configure STEP to publish the system configuration to a branch in Git, an external Version Control System (VCS) (refer to:https://git-scm.com). Using inbound integration endpoint plugins, 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.

Note: The 'configuration-management' add-on component must be activated to enable VCS.

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 STEP Workbench System Setup tab clearly are configuration, the Tree tab holds both data and configuration. For example, objects like import, export, and bulk update configurations are obviously 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 plugin topics / sections of the **Version Control System Integration** documentation, there is no preconceived notion of what is configuration and what is data. Instead, as described in the **Outbound Endpoint Configuration** topic, this can be configured. However, it should be stressed that the functionality is not designed to handle vast amounts of data objects (e.g., SKUs).

Information about VCS Integration

Additional information can be found in the following sections / topics:

- **Integration Endpoint Plugins**
- **Outbound Endpoint Configuration**
- **Inbound Endpoint Configuration**
- **Editable Business Rules Format**
- **VCS: Example Setups**
- **VCS: Considerations and Limitations**

Integration Endpoint Plugins

The plugin suite for VCS integrations consists of these plugins:

- Outbound Integration Endpoint 'STEPXML Splitter' Post-processor Plugin
- Outbound Integration Endpoint 'Git Delivery' Plugin
- Inbound Integration Endpoint 'STEPXML Joiner' Pre-processor Plugin
- Inbound Integration Endpoint 'Invoke OIEP' Post-processor Plugin

Each is described in their own section below.

Additionally, within this topic, you will also find information regarding an editable business rule format. JavaScript-based business rules can be created, maintained, and (unit) 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.

Outbound Integration Endpoint 'STEPXML Splitter' Post-processor Plugin

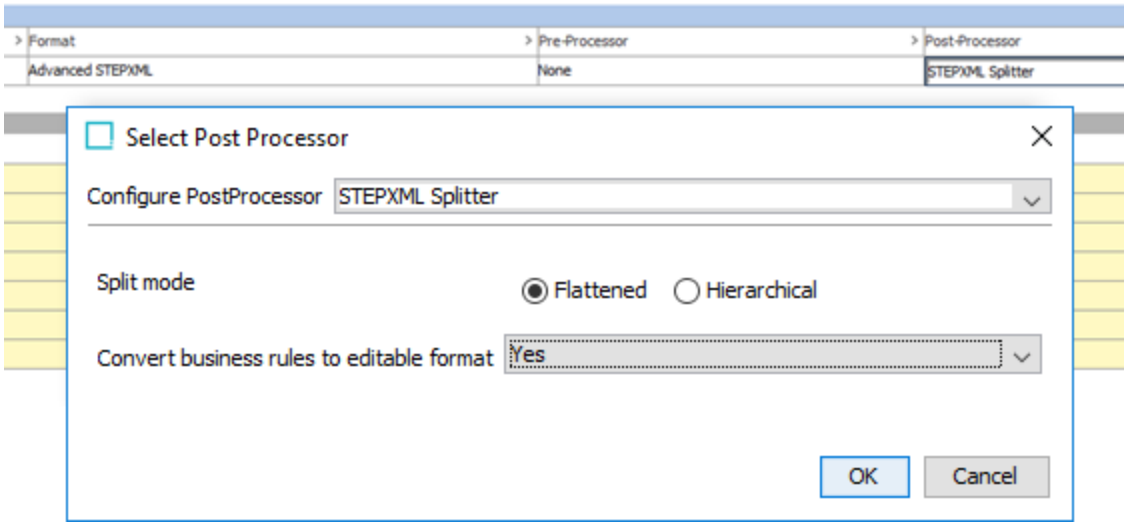
The 'STEPXML Splitter' post-processor can take any STEPXML file produced by the 'STEP Exporter' processing engine as input and will split the file into multiple valid STEPXML files / editable business rule format files that are then passed to the configured delivery plugin. Generally, the splitter produces 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.

The reasoning behind splitting and normalizing is that it 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.

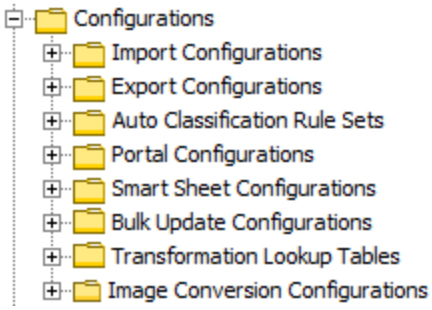
The ‘STEPXML Splitter’ plugin has two configuration options explained below.

Split mode

Split mode, that defaults to ‘Flattened’, with the other option being hierarchical, affects how STEP objects that normally are exported in a nested fashion are represented in the produced split files.



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



In both modes, one file will be created per classification object, but in ‘Hierarchical’ mode each of the leaf classifications, like e.g., 'Import Configurations', will be nested inside the element representing the 'Configurations' classification (this element will be stripped of all but ID, object type ID, and parent ID information) as shown below.

```
(…)
<Classifications>
  <Classification ID='ConfigurationsRoot' UserTypeID='ConfigurationsRoot'
ParentID='Classification 1 root'>
    <Classification ID='stibo.IMConfigFolder' UserTypeID='ImportConfigurationsRoot'>
      <Name>Import Configurations</Name>
      <MetaData>
        <Value AttributeID='Purpose' QualifierID='en-US'>Storage for import
configurations</Value>
      </MetaData>
    </Classification>
  </Classification>
</Classifications>
(…)
```

In 'Flattened' mode, upper levels will be omitted, and each file will contain exactly one 'Classification' element with parent identifier information as shown below.

```
(…)
<Classifications>
  <Classification ID='stibo.IMConfigFolder' UserTypeID='ImportConfigurationsRoot'
ParentID='ConfigurationsRoot'>
    <Name>Import Configurations</Name>
    <MetaData>
      <Value AttributeID='Purpose' QualifierID='en-US'>Storage for import
configurations</Value>
    </MetaData>
  </Classification>
</Classifications>
(…)
```

Generally, it is recommended to use the default 'Flattened' mode while the 'Hierarchical' mode only should be used if the full hierarchy path must be present in each file. Obviously, the 'Hierarchical' example from above can be imported on a system where the classification with ID 'ConfigurationsRoot' is not present (it will be created during import). Importing the 'Flattened' example on such a system will, on the other hand, result 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, the parameter 'Convert business rules to editable format' option should be set to 'Yes.' The business rules in the STEPXML that are fed to the post-processor will be 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 **Editable Business Rules Format** topic.

Outbound Integration Endpoint 'Git Delivery' Plugin

The 'Git Delivery' plugin delivers files produced by the outbound integration endpoint processing engine / 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 the plugin on an on-premise 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 plugin first checks if the local delivery directory is Git enabled. If this is the case, 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 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 will always be used and the second approach with clone and checkout will be used.

Note: The remote repository cannot be empty. At least one branch with one file must exist.

After this, the locally checked-out branch will be in sync with the remote branch, and the following operations are performed:

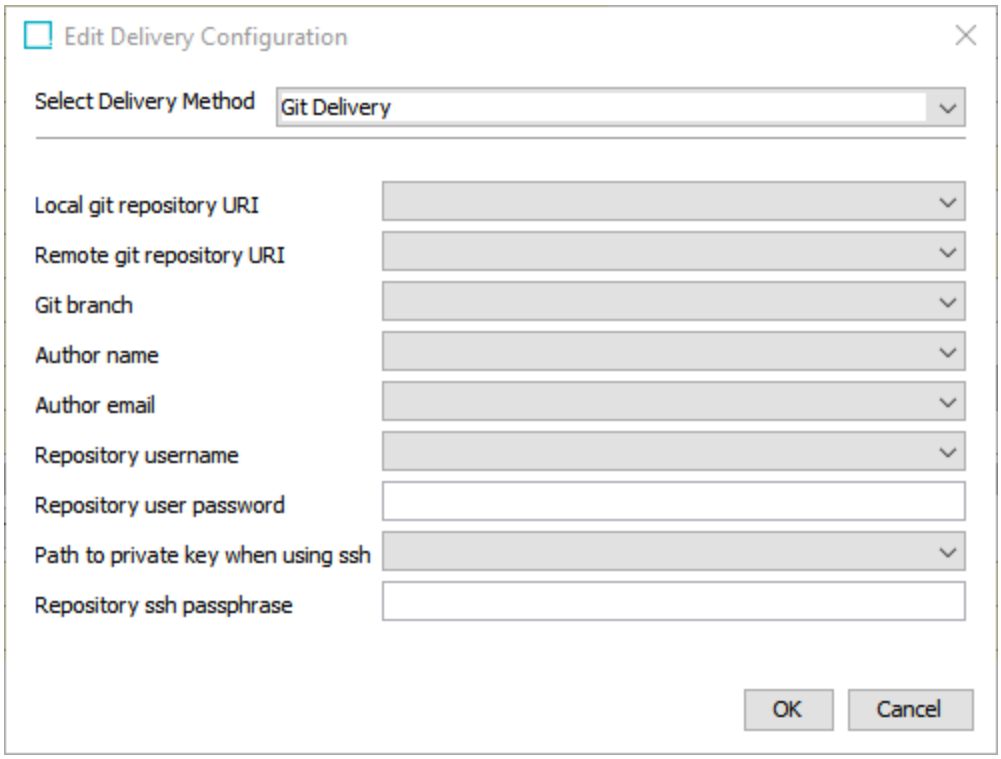
1. Files produced by the outbound integration endpoint are written to the local directory
2. Files in local directory but not in delivery are deleted
3. Git stage
4. Git commit
5. Git push

If the configured branch does not exist in the remote repository in advance, it will be created.

It is important to understand the implications of this functionality, namely that:

- The outbound integration endpoint must with each invocation produce files for all the configuration objects / settings to be represented in the configured Git branch. Exporting only a file for a single object will cause all other files in the Git branch to be deleted. Refer to the **Outbound Endpoint Configuration** topic for more information on how to configure an outbound integration endpoint using this delivery plugin.
- Only STEP should make changes in the configured Git branch. All other changes will be overwritten with the next STEP delivery.

The 'Git Delivery' plugin has the following parameters:



- **Local git repository URI** - URI for the local directory via which configuration files will be synchronized. Possible values for this parameter are read from 'GitDelivery.LocalRepoUri.[integer]' configuration property entries.

  Important: The 'GitDelivery.LocalRepoUri.[integer]' property should only be set for on-premise STEP systems. For cloud STEP systems, the property should not be set and the 'Local git repository URI' parameter should be left blank in the plugin configuration.

  Having, for example, the following entries in sharedconfig.properties:

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

  will make it possible to select between the values '/workarea/conf-attributes' and '/workarea/conf-all' for the parameter in the workbench.

  Note: For this property and the ones mentioned below, the sequence of the variable integers in the property names must be complete. E.g., 1, 2, 3. Property values will not be read correctly if the sequence is, for instance 1, 3, 4.

- **Remote git repository URI** - URI for the remote repository. As with the 'Local git repository URI' values that can be selected in the workbench are read from 'GitDelivery.RemoteRepoUri.[integer]' configuration property entries.

  Examples:

  ```
  GitDelivery.RemoteRepoUri.1=https://bitbucket.org/john-smith/step-conf.git

  GitDelivery.RemoteRepoUri.2=ssh://gituser@192.168.56.102/home/gituser/git/repo.git
  ```

- **Git branch** - Name of the branch to which the delivery is published. Values for this parameter are read from 'GitDelivery.Branch.[integer]' configuration property entries. Examples:

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

- **Author name** - Author name for the Git commit. Values for this parameter are read from 'GitDelivery.AuthorName.[integer]' configuration property entries. Example:

  ```
  GitDelivery.AuthorName.1=John Smith - STEP Dev1
  ```

- **Author email** - Author email for the Git commit. Values for this parameter are read from 'GitDelivery.AuthorEmail.[integer]' configuration property entries. Example:

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

- **Repository username** - The remote repository username. Values for this parameter are read from 'GitDelivery. RemoteRepoUsername.[integer]' configuration property entries. Example:

  ```
  GitDelivery.RemoteRepoUsername.1=john.smith@gmail.com
  ```

- **Repository user password** - The remote repository password. Password must be entered directly when configuring the plugin.
- **Path to private key when using ssh** - The remote repository private key when using Secure Shell Protocol (SSH). Values for this parameter are read from 'GetDelivery.SshPrivateKeyUri.[integer]' configuration property entries. The folder specified should contain the private SSH key. Example:

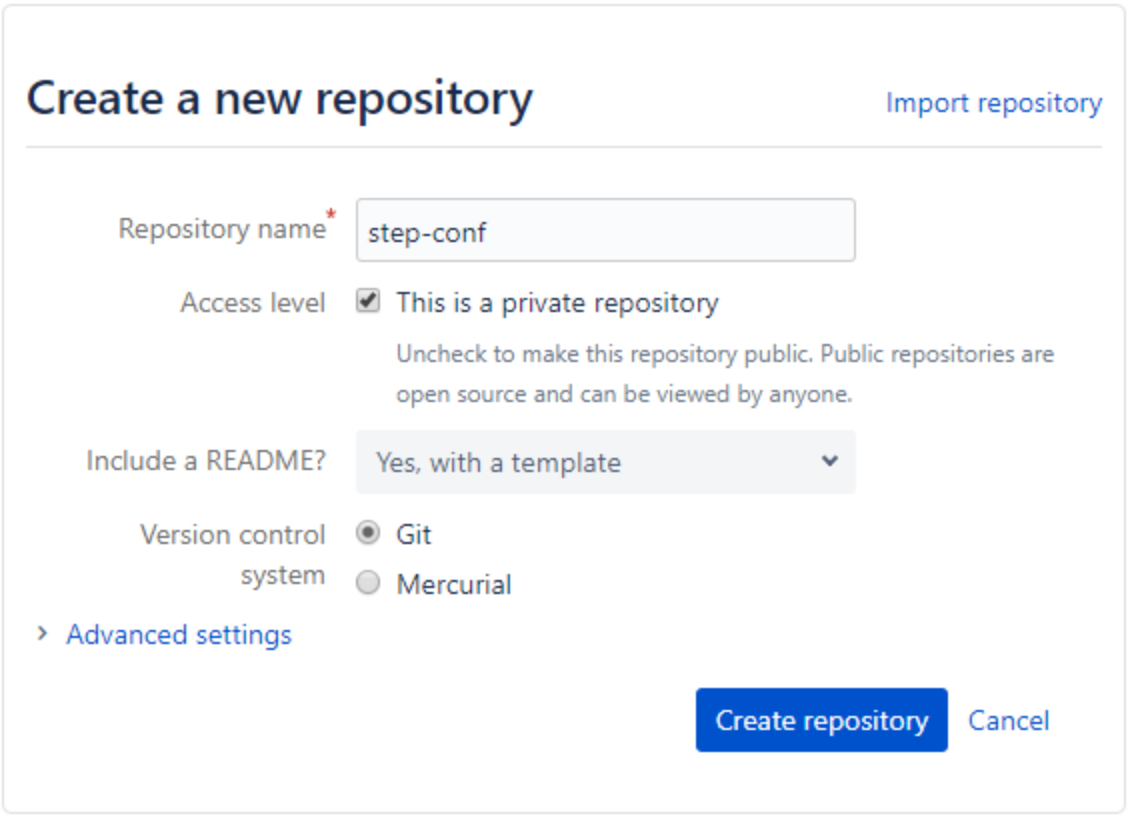
  ```
  GitDelivery.SshPrivateKeyUri.1=/home/stibosw/.ssh/id_rsa
  ```

- **Repository ssh passphrase** - The remote repository Secure Shell Protocol (SSH) password / passphrase. Passphrase must be entered directly when configuring the plugin.

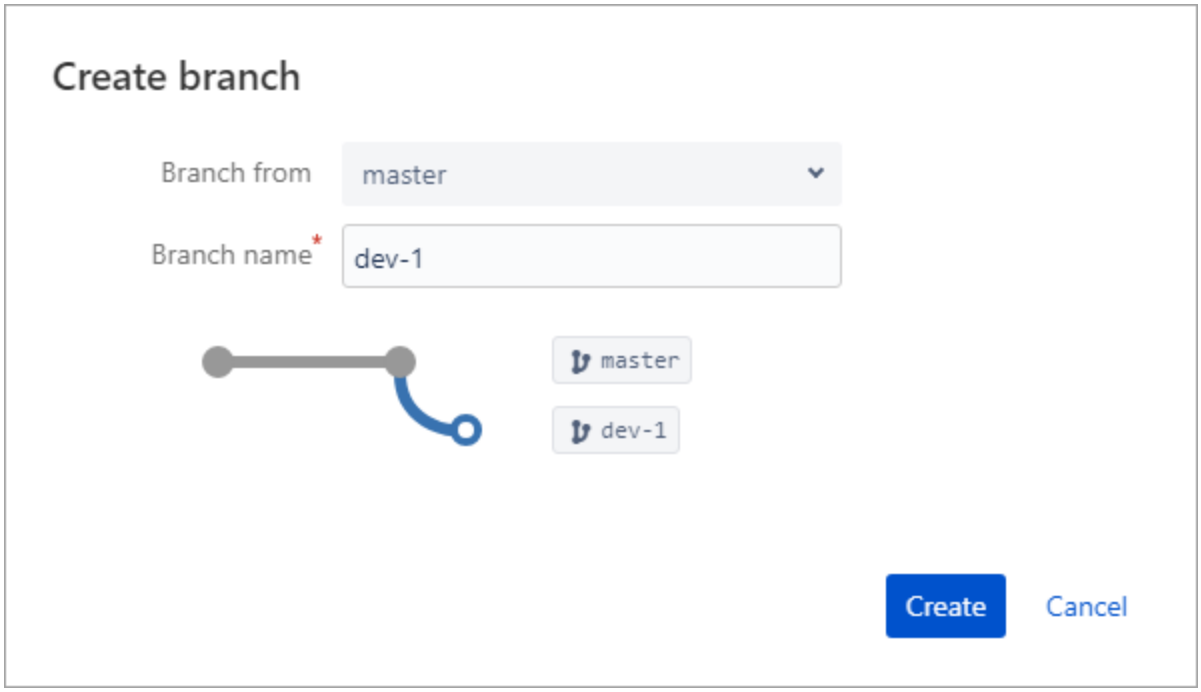
Remote Setup Example

This section contains an example of how a remote Git repository using Bitbucket (https://bitbucket.org) can be configured from scratch to work with the Git delivery plugin.

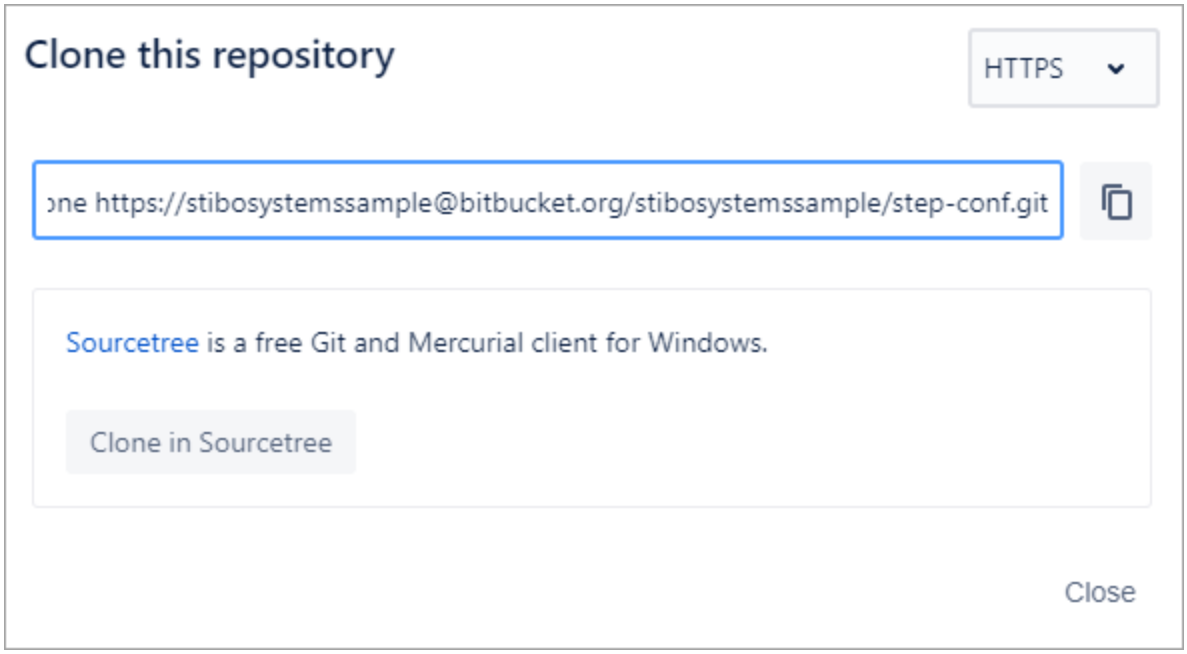
1. Create a new repository with a 'README' file via the Bitbucket web interface.



2. Create a system specific branch in the repository.



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



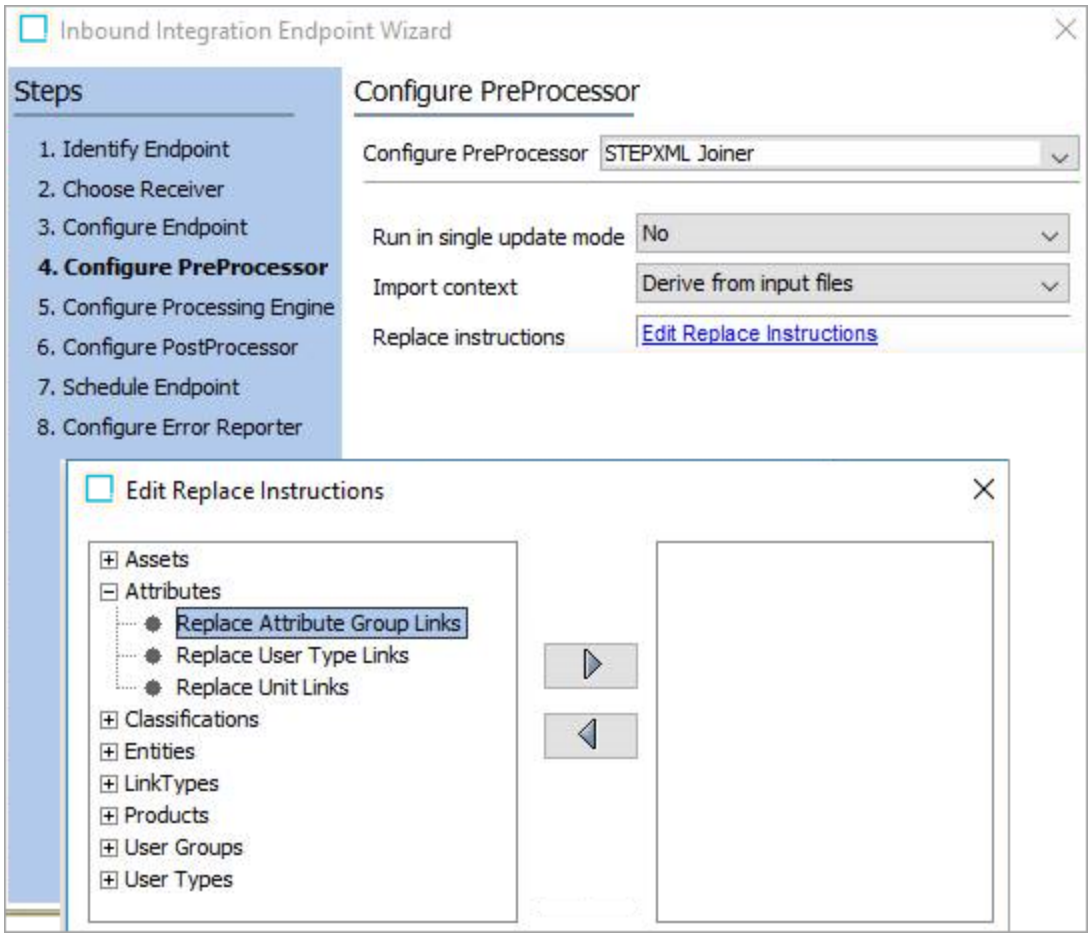
Inbound Integration Endpoint 'STEPXML Joiner' Pre-processor Plugin

The 'STEPXML Joiner' inbound pre-processor plugin has been designed to be used for getting configurations and settings exported via the outbound functionality, described above, imported on a system. The outbound functionality produces STEPXML files and potentially*.js files representing business rules. The 'STEPXML Joiner' plugin can take a .ZIP file containing any number of such files as input and will then combine these to a single STEPXML file with objects and settings appearing in the most sensible order and pass the combined file to the inbound integration endpoint processing engine.

The pre-processor plugin can add various processing instructions to the combined STEPXML file, and there are two options for specifying these instructions. Users can either use the UI to configure the processing instructions (refer to the screenshot below) or users can include a template STEPXML file in the .ZIP file fed to the endpoint.

To configure the UI:

1. Make a single update mode selection.
2. Make an import context selection. This is the context that will receive the imported data
3. For 'list properties' (multiple instances of the same XML element at the same level) such as 'Value' elements inside the 'Values' element for a product or 'TargetUserTypeLink' elements for a reference type definition, this means that special processing instructions must be used to express that properties not present in the import file must be removed from the system as part of the import. Add these replacement instructions by clicking 'Edit Replace Instructions'.
4. Within the 'Edit Replace Instructions' dialog, build the rules by selecting options on the left and using the arrow to move the rule over to the right box. Save changes before moving to the next step within the Inbound Integration Endpoint Wizard.

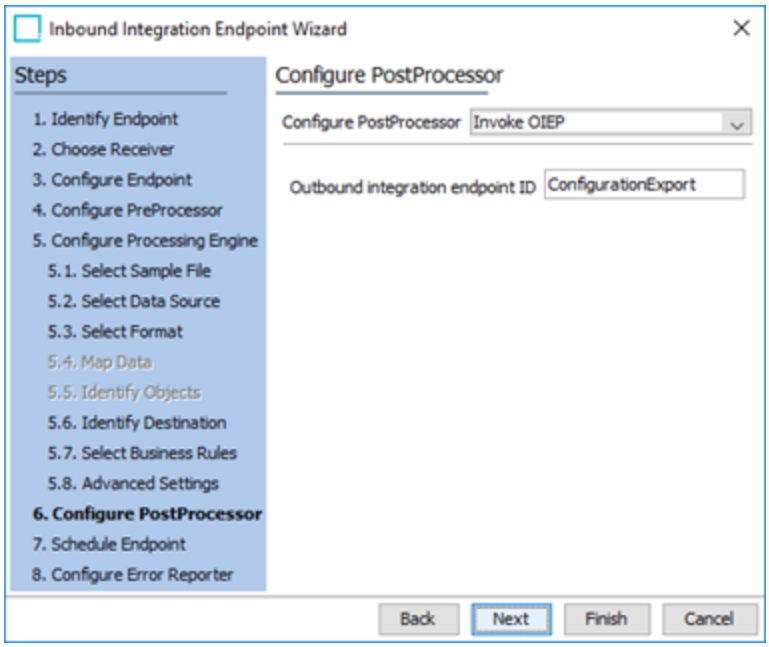


If a template file is provided in the .ZIP, the settings from this file overrule any UI configurations made in the UI. The template file must be named 'ProcessingInstructions.xml' and should, inside the STEP-ProductInformation element, only include the ReplacementRules element.

For more information regarding ReplacementRules, refer to the **ReplacementRules Tag in STEPXML** topic in the **STEPXML Tags and Examples** section of the **Data Exchange** documentation.

Inbound Integration Endpoint 'Invoke OIEP' Post-processor Plugin

The 'Invoke OIEP' post-processor plugin allows for an outbound integration endpoint to be invoked once the inbound import process has completed. This makes it possible to update the representation of the system configuration in a remote Git branch immediately after the configuration has been imported. As shown below, the plugin only has a single configuration parameter: the ID of the outbound integration endpoint to invoke.



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

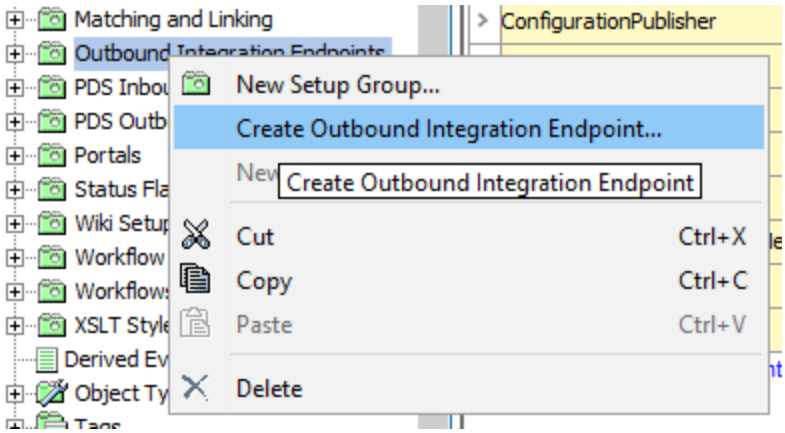
- **Editable Business Rules Format**
- **VCS: Example Setups**
- **VCS: Considerations and Limitations**

Outbound Endpoint Configuration

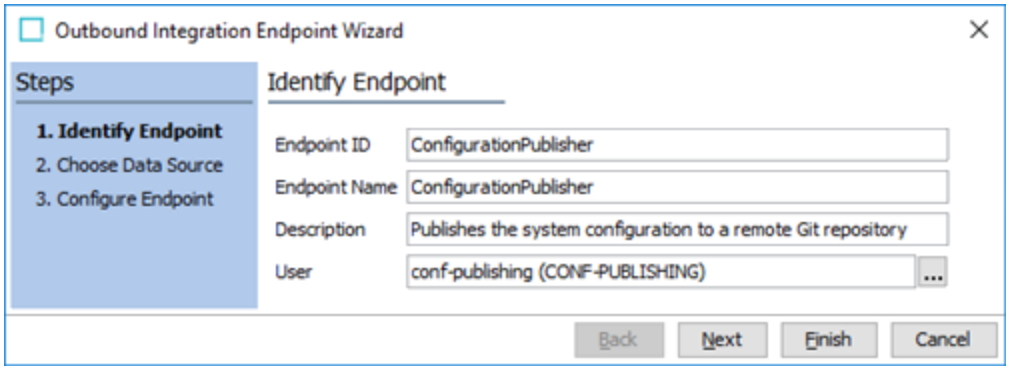
This section describes step-by-step how an outbound integration endpoint can be configured to be used for publishing the system configuration to a remote Git repository.

Note: If you want to compare the configurations from multiple systems, the endpoint configuration on the systems should be identical (except for the Git branch 'Git Delivery' plugin information).

1. Launch the 'Outbound Integration Endpoint Wizard' by selecting 'Create Outbound Integration Endpoint…' in the context menu for a setup group configured to hold outbound integration endpoints.

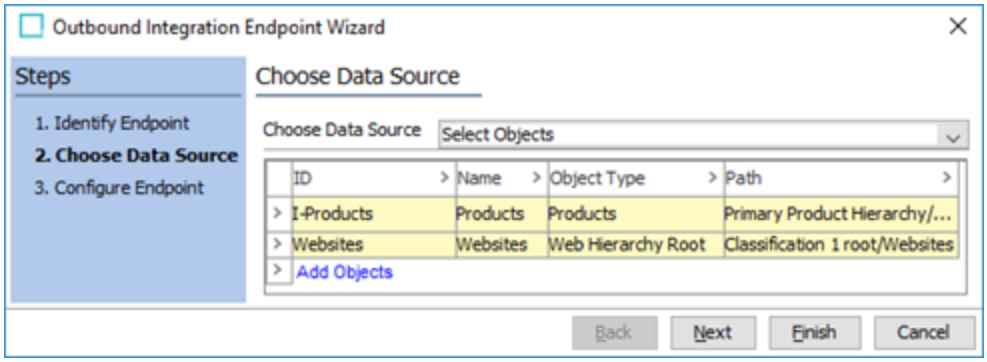


2. In the 'Identify Endpoint' step of the wizard, enter basic information. Make sure the endpoint is configured to run as a system user who has view privileges to the configuration objects to be exported.

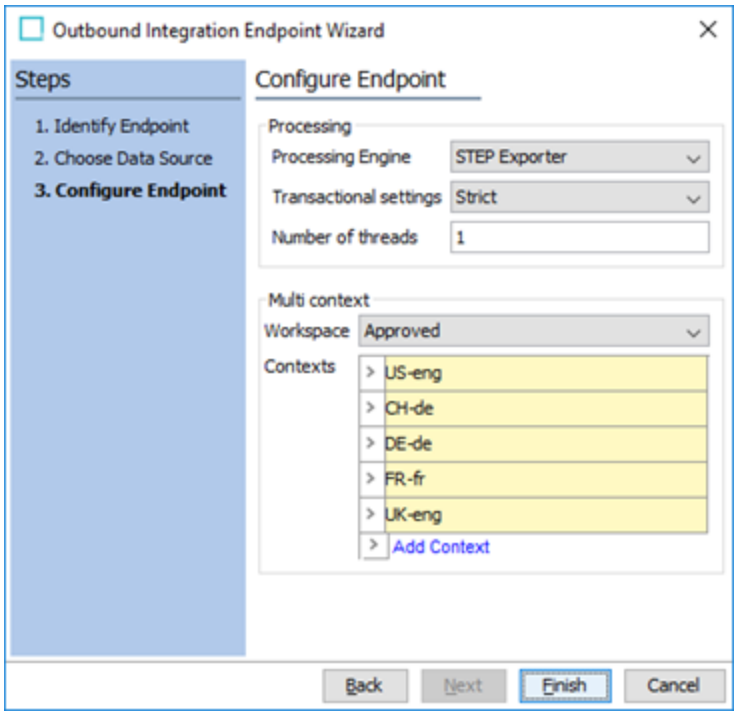


3. As stated in **Integration Endpoint Plugins** > **Outbound Integration Endpoint 'Git Delivery' Plugin** , all configurations and settings to be held in Git must be published each time the outbound integration endpoint is invoked. Therefore, the endpoint must in step 2 of the wizard be configured to use the 'Select Objects' data source option. **Even if no product, entity or classification objects are to be published, a 'dummy'**

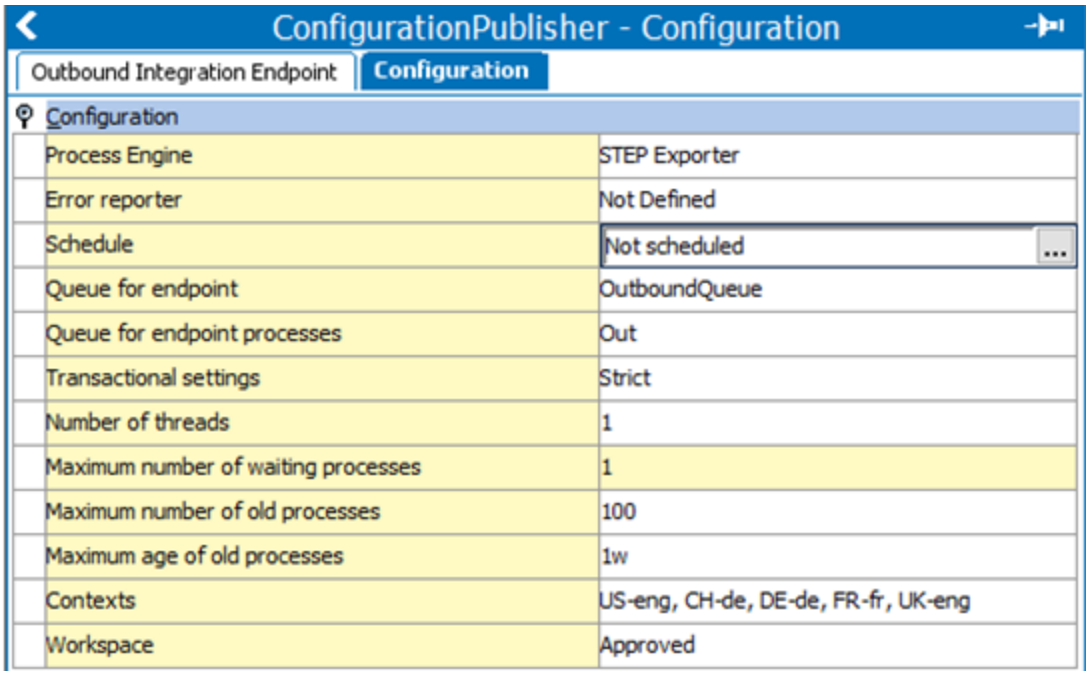
selection must be made. If such data objects are to be published, the relevant root nodes should be selected.



4. In the 'Configure Endpoint' step, select 'STEP Exporter' as the processing engine. As for the 'Workspace' parameter, 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 and for this, the 'Approved' workspace should be selected. If configuration data that is to be published is dimension dependent, all relevant contexts should be selected.

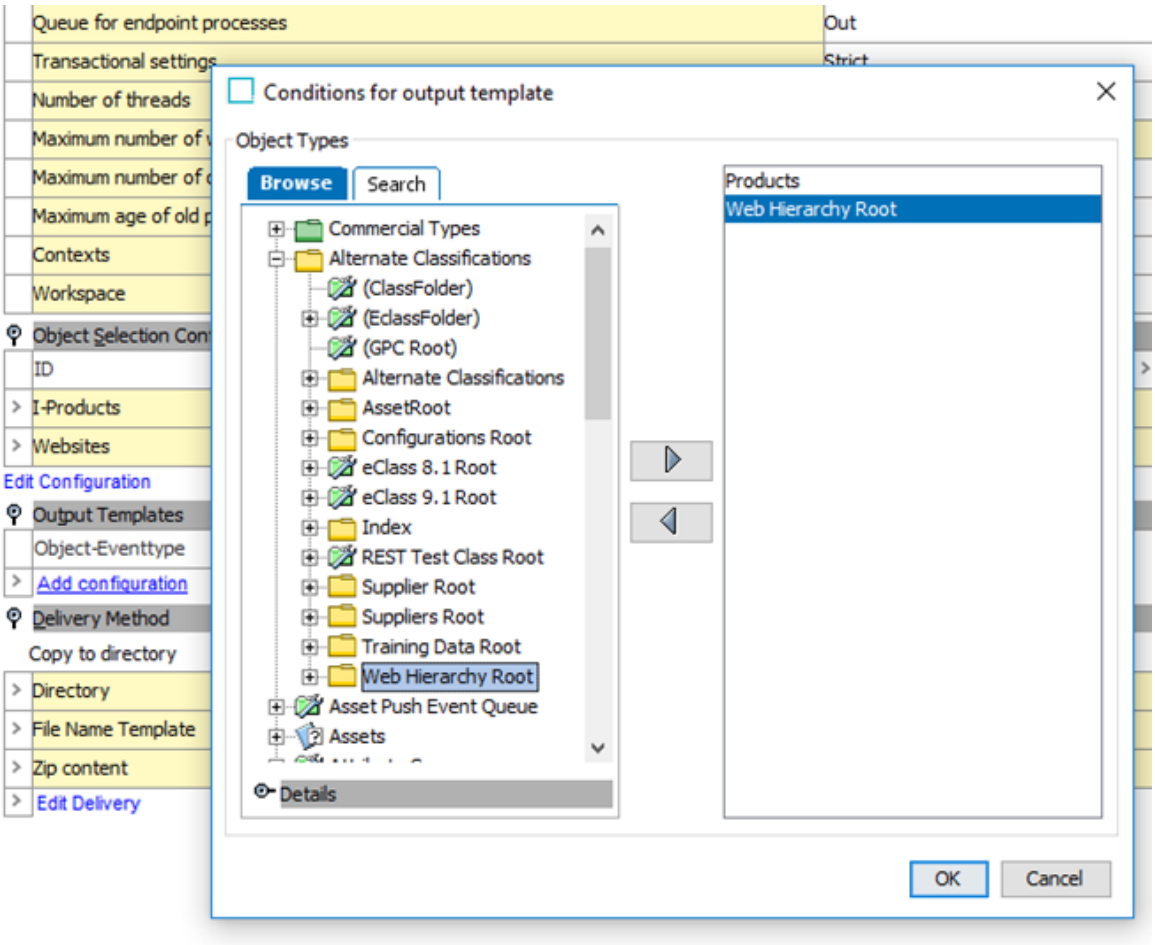


5. Once the wizard has been completed, on the 'Configuration' tab of the newly created endpoint, configure the schedule, queue, and process retention settings as desired.



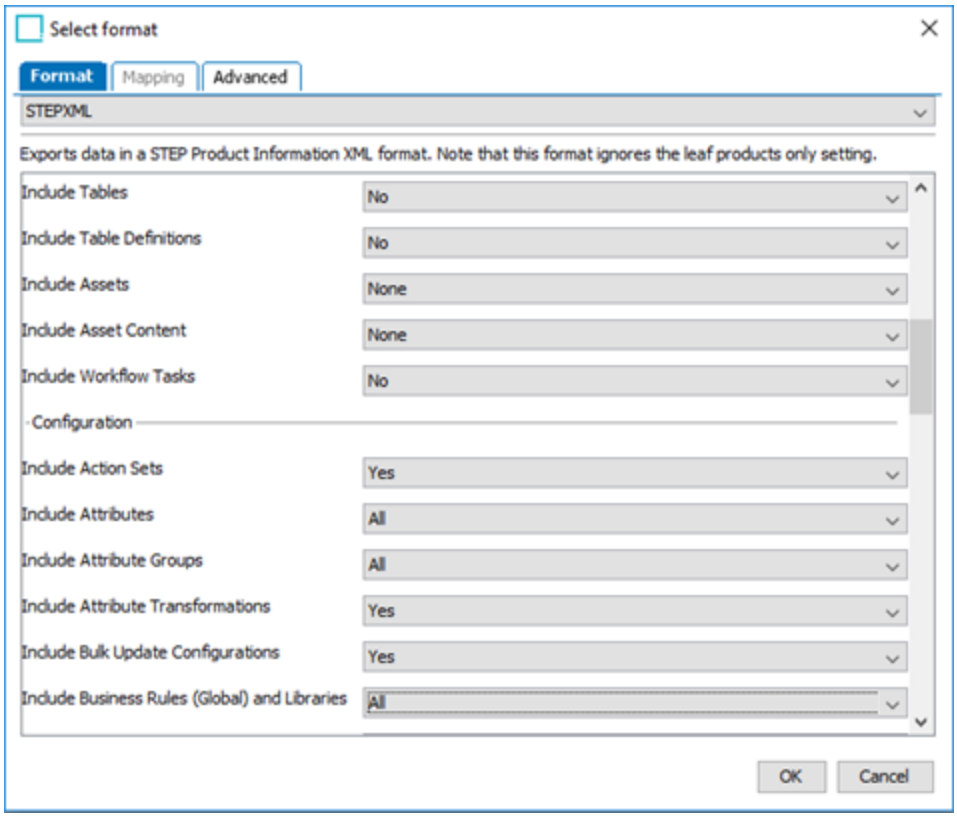
6. In the 'Output Templates' section, add a single configuration and select the object types of the nodes selected for publishing (must also be done for 'dummy' selections).

> **Note:** The Git delivery plugin will only work with a single output template.

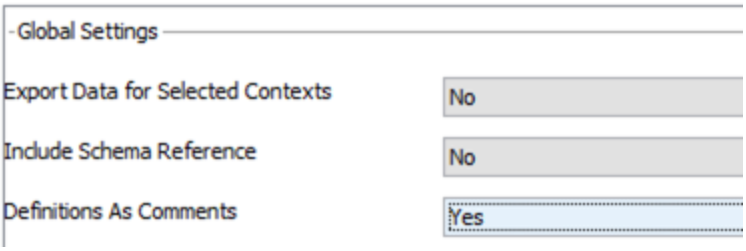


7. For 'Format', the Version Control System (VCS) integration plugins will work with either the 'STEPXML' or the 'Advanced STEPXML' format plugins. However, if data objects are to be published and the exported objects are to be filtered according to their object types, the 'Advanced STEPXML' option must be selected.

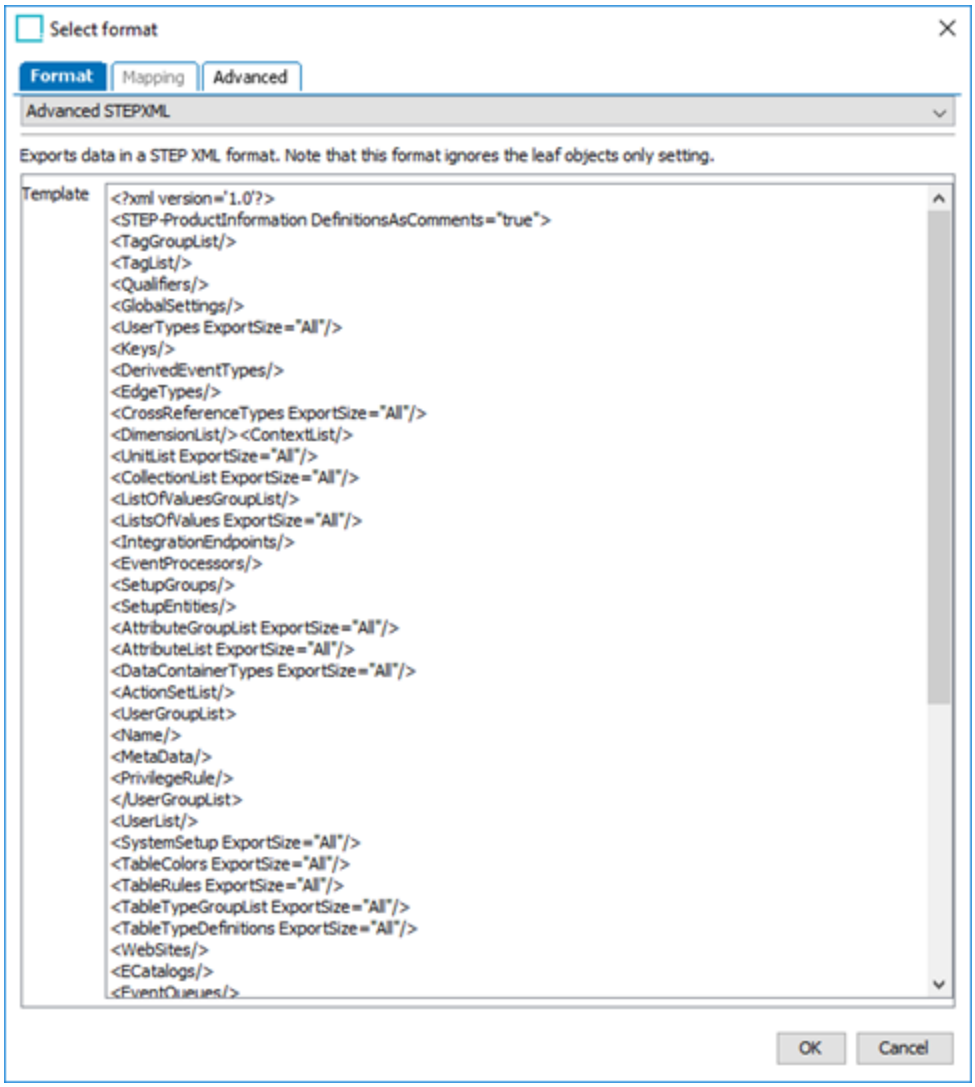
   If 'STEPXML' is selected, select 'Yes' / 'All' for the configuration types to export, 'Minimum' for products, entities, and classifications, if these are to be exported and 'No' / 'None' for all types that should not be published.



Further, to make obfuscated configurations (e.g., business rules) comparable outside STEP, select 'Yes' to export 'Definitions As Comments'.



If the 'Advanced STEPXML' option is selected, enter the appropriate template and make sure to set the STEP-ProductInformattion 'DefinitionsAsComments' attribute to 'true'.

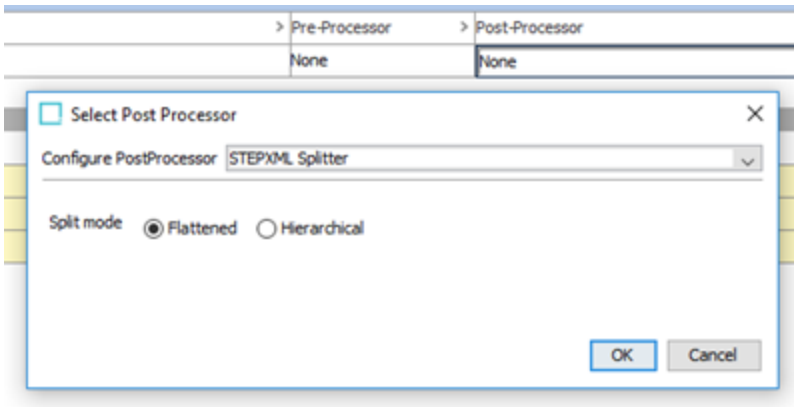


Full example configuration with product object type filtering is shown below:

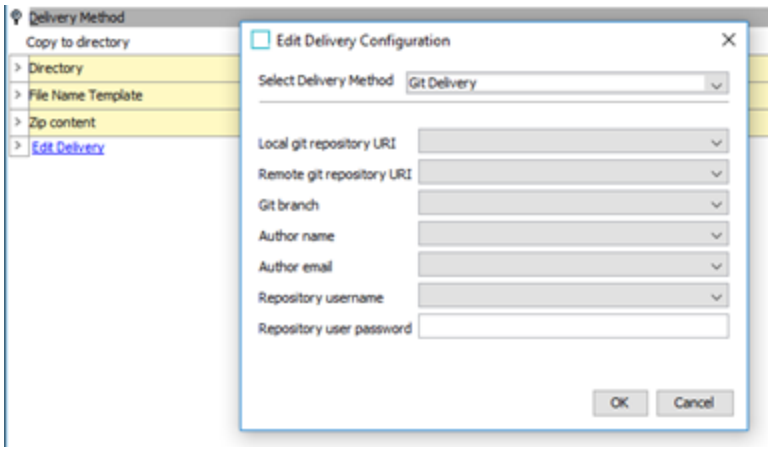
```
<?xml version='1.0'?>
<STEP-ProductInformation DefinitionsAsComments='true'>
  <TagGroupList/>
  <TagList/>
  <Qualifiers/>
  <GlobalSettings/>
  <UserTypes ExportSize='All'/>
  <Keys/>
  <DerivedEventTypes/>
  <EdgeTypes/>
  <CrossReferenceTypes ExportSize='All'/>
  <DimensionList/>
  <ContextList/>
  <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'/>
  <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/>
  <Products ExportSize='Minimum'>
    <FilterUserType ID='Level1'/>
    <FilterUserType ID='Level2'/>
    <FilterUserType ID='Level3'/>
    <Product/>
  </Products>
  <Classifications ExportSize='Minimum'/>
</STEP-ProductInformation>
```

8. Configure the 'STEPXML Splitter' post-processor. For more information, refer to **Integration Endpoint Plugins** > **Outbound Integration Endpoint 'STEPXML Splitter' Post-processor Plugin** for details.



9. Configure the 'Git Delivery' delivery plugin. Refer to section **Integration Endpoint Plugins** > **Outbound Integration Endpoint 'Git Delivery' Plugin** for details.

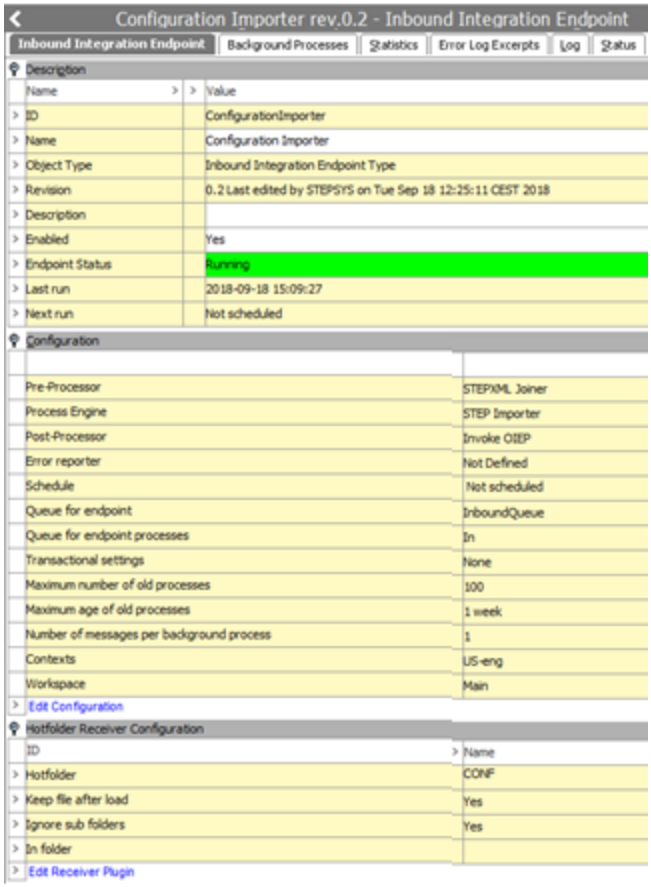


Inbound Endpoint Configuration

The inbound 'STEPXML Joiner' pre-processor plugin and the 'Invoke OIEP' post-processor plugin are configured via the 'Inbound Integration Endpoint Wizard' opened by selecting 'Create Inbound Integration Endpoint…' in the context menu for a setup group configured to hold inbound integration endpoints. For details, refer to the **Inbound Integration Endpoint 'STEPXML Joiner' Pre-processor Plugin** and the **Inbound Integration Endpoint 'Invoke OIEP' Post-processor Plugin** sections of the **Integration Endpoint Plugins** topic.

The endpoint must be configured to import STEPXML (provide any valid STEPXML file as a sample file) and can be configured with any receiver plugin capable of handling .ZIP files, for example, the 'Hotfolder Receiver' or the 'REST Receiver'. If workspace revisable objects like products and classifications are imported, it must be decided whether or not import changes automatically should be approved.

Example configuration shown below:



Editable Business Rules Format

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:

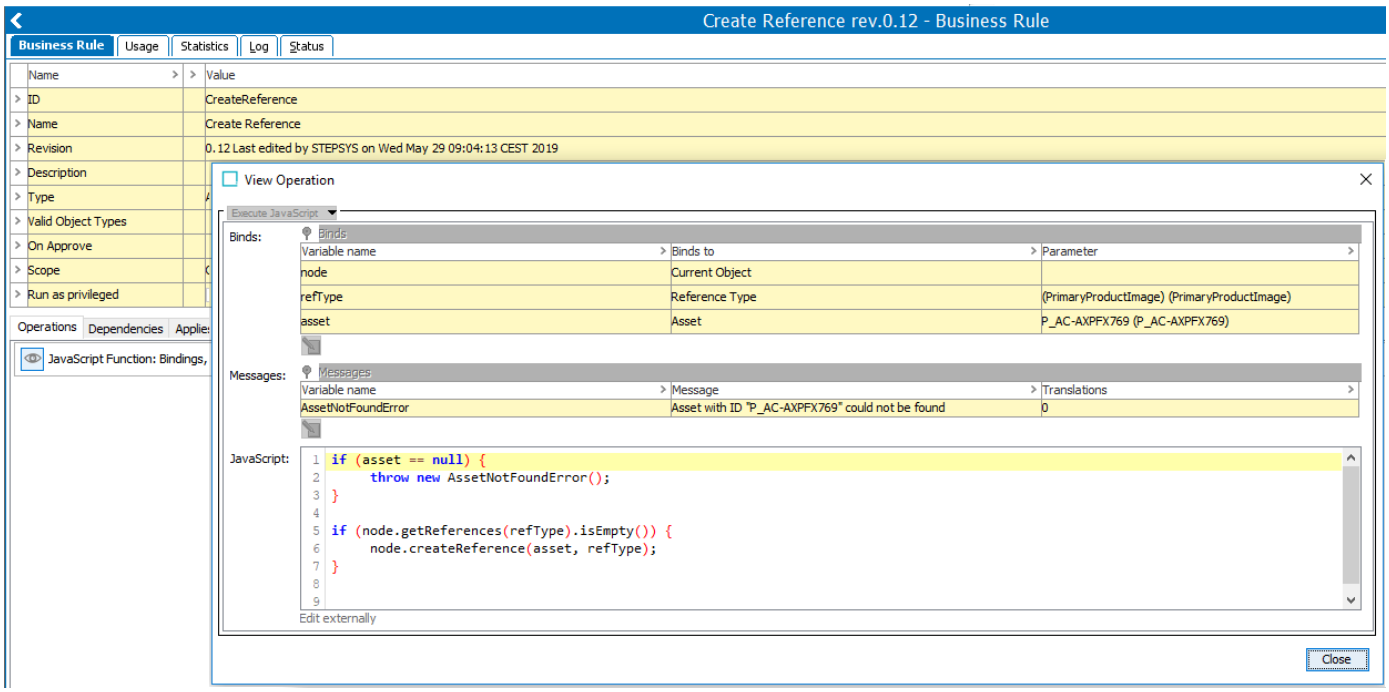
- Editable Business Rule Format
- Options for Export
- Options for Import
- REST Resources for Testing and Validation

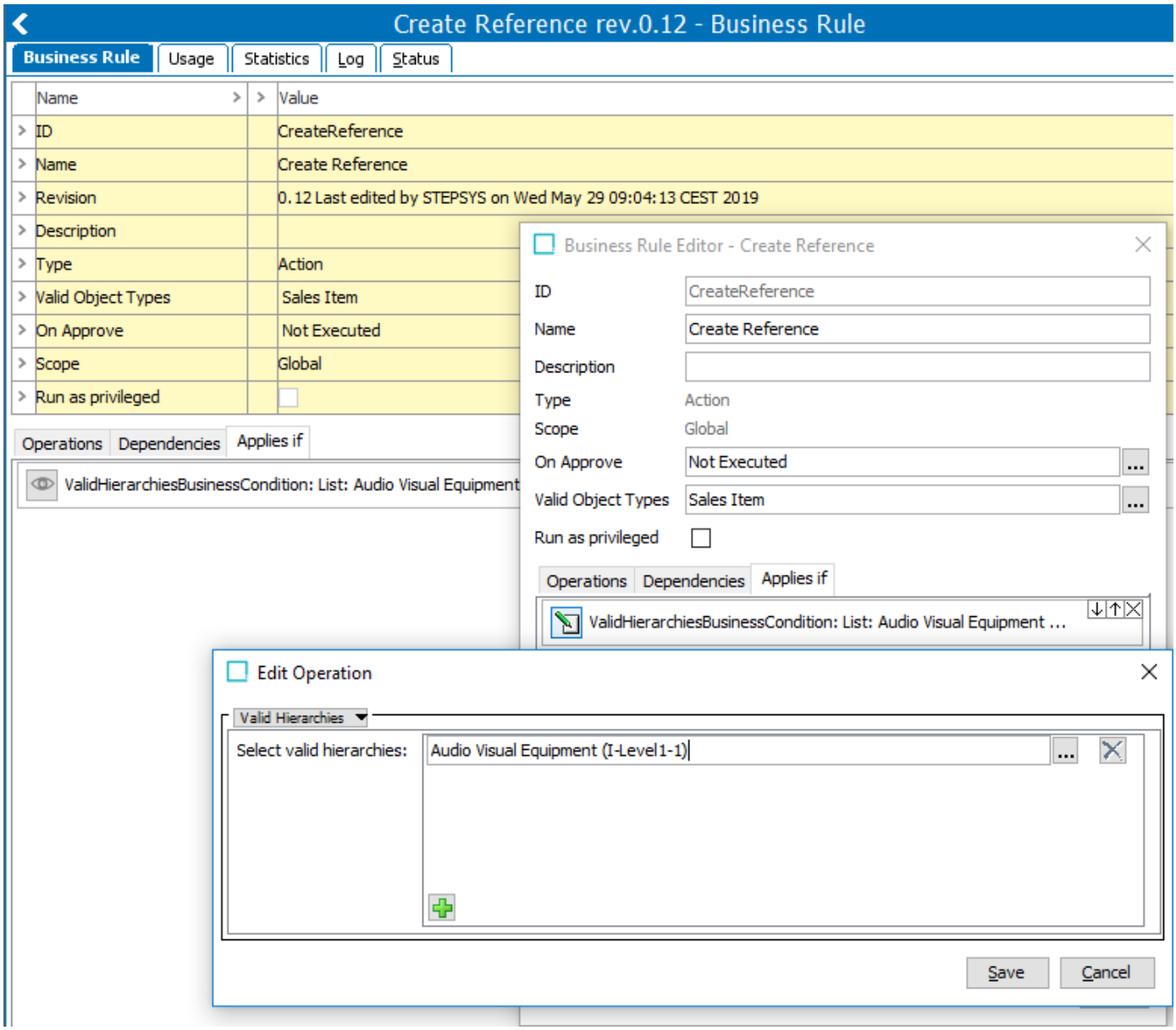
Note: The 'configuration-management' add-on component must be activated to enable the functionality described below.

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, assume there is a simple business action with one JavaScript operation and a non-JavaScript precondition as shown below:


Create Reference rev.0.12 - Business Rule
Business Rule
Usage
Statistics
Log
Status
Name
Value
ID
CreateReference
Name
Create Reference
Revision
0.12 Last edited by STEPSYS on Wed May 29 09:04:13 CEST 2019
Description
Type
Valid Object Types
On Approve
Scope
Run as privileged
Operations
Dependencies
JavaScript Function: Bindings,
View Operation
Execute JavaScript
Binds:
Binds
Variable name
Binds to
Parameter
node
Current Object
refType
Reference Type
(PrimaryProductImage) (PrimaryProductImage)
asset
Asset
P_AC-AXPFX769 (P_AC-AXPFX769)
Messages:
Messages
Variable name
Message
Translations
AssetNotFoundError
Asset with ID "P_AC-AXPFX769" could not be found
0
JavaScript:
1 if (asset == null) {
2 throw new AssetNotFoundError();
3 }
4
5 if (node.getReferences(refType).isEmpty()) {
6 node.createReference(asset, refType);
7 }
8
9
Edit externally
Close



When exported using the default settings, the business action will be represented as follows in the generated file:

```
/*===== 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"
}
*/
```

As mentioned above, the logic of the JavaScript operation is wrapped in a function. This function is, in the example, exported in line with the Node.js module system convention. The property 'ConfigurationManagement.BusinessRuleConverter.ExportFormat' in the sharedconfig.properties file 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 somewhat 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 example, assume a library has the following content:

```
function isProductBelow(prod, checkProdID) {
       if(!isProduct(prod)) throw "Function only works with Products";
       if(checkProdID == "Product hierarchy root") return true;
       if(prod.getID() == "Product hierarchy root") throw "The top level Product is never
below another Product.";
       var currentParentId;
       var currentProd = prod;
       while (true) {
               currentParentId = currentProd.getParent().getID();
               if(currentParentId == "Product hierarchy root") return false;
               else if (currentParentId == checkProdID) return true;
               else currentProd = currentProd.getParent();
       }
}

function isProduct(obj) {
```

```
        return obj instanceof com.stibo.core.domain.Product;
}
```

When exported with the sharedconfig.properties file 'ConfigurationManagement.BusinessRuleConverter.ExportFormat' property set to the default 'NodeExport' value, the following is appended to the file, making it possible 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 have library functions that call functions in other referenced libraries be executable outside STEP, these can be modified so that it is possible to 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 could be modified as shown below, making it possible 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());
```

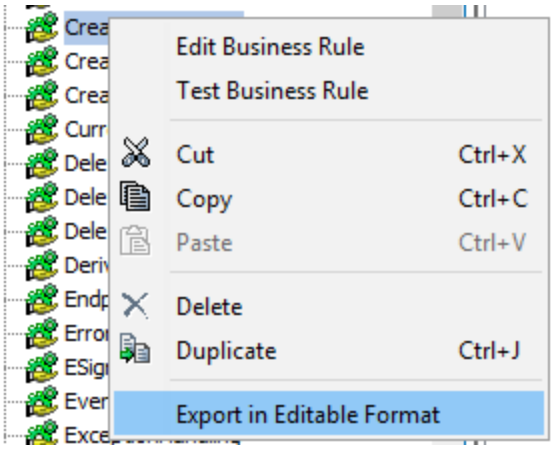
Note: Adding an extra optional parameter does not require that the JavaScript calling the function be modified.

Options for Export

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

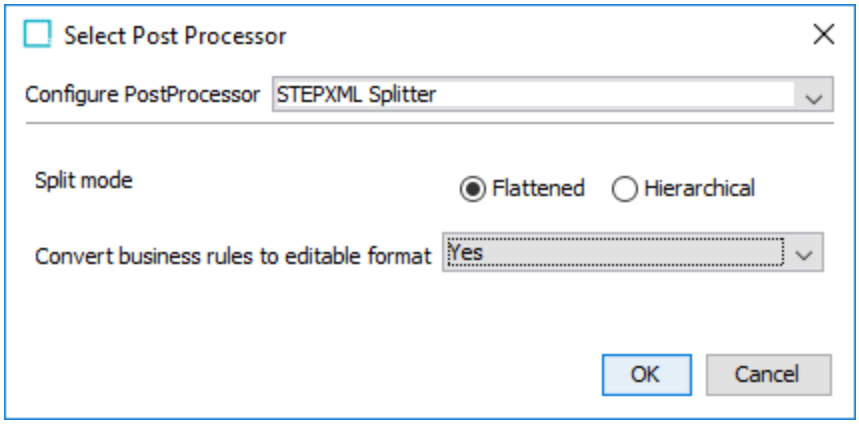
Manual export

To manually export business rules individually, use the 'Export in Editable Format' context menu option for business rules as shown below:



Outbound Integration Endpoint

When using the configuration management 'STEPXML Splitter' post-processor plugin for outbound integration endpoints, set the parameter 'Convert business rules to editable format' option (shown below).



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.

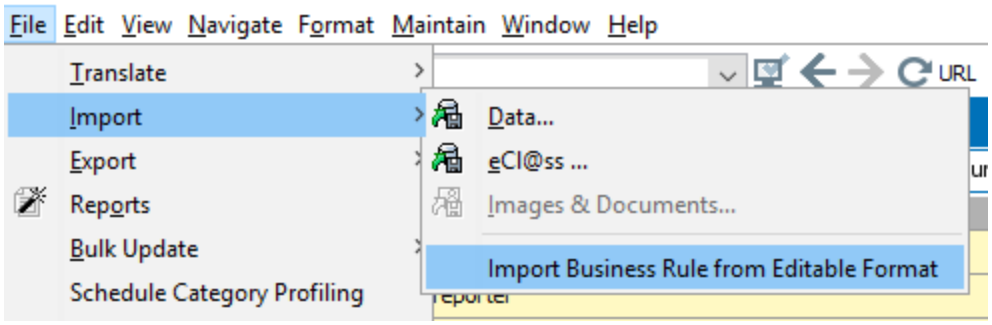
For details, refer to the **Outbound Integration Endpoint 'STEPXML Splitter' Post-processor Plugin** sections of the **Integration Endpoint Plugins** topic.

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, 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.

For details, refer to the **Inbound Integration Endpoint 'STEPXML Joiner' Pre-processor Plugin** section of the **Integration Endpoint Plugins** topic.

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 lets clients execute ECMAScript 5-compliant JavaScript on a running STEP server in a non-committing mode with access to a STEP Manager that again 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 would return "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 lets clients 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 plugin 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 whether or not 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' in the sharedconfig.properties file to be set to 'true' on systems to be used for tests and validation (defaults to 'false').

Note: There is an example step.js Node.js module that wraps the REST resources available from the STEP API Documentation page that can be used together with the documentation and examples provided.

Additional VCS information can be found in the following sections / topics:

- **Outbound Endpoint Configuration**
- **Inbound Endpoint Configuration**
- **VCS: Example Setups**
- **VCS: Considerations and Limitations**

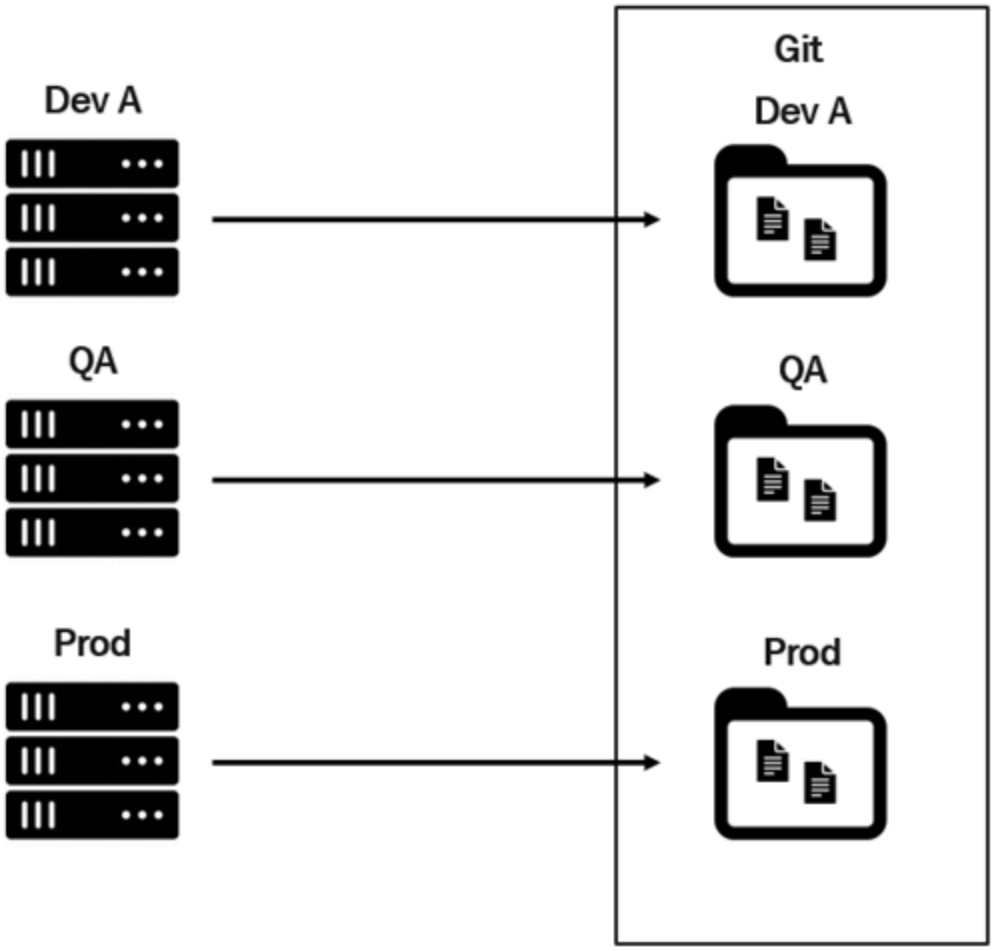
VCS: Example Setups

The plugins described in the **Integration Endpoint Plugins** topic can be used in several different scenarios and do not necessarily have to be used together. This section describes two different possible setups.

System Comparison

It is possible to use just the outbound integration endpoint plugins to have the configuration from each system in a 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 outbound integration endpoints remotely via REST (resource operation available in the STEP REST API V1).

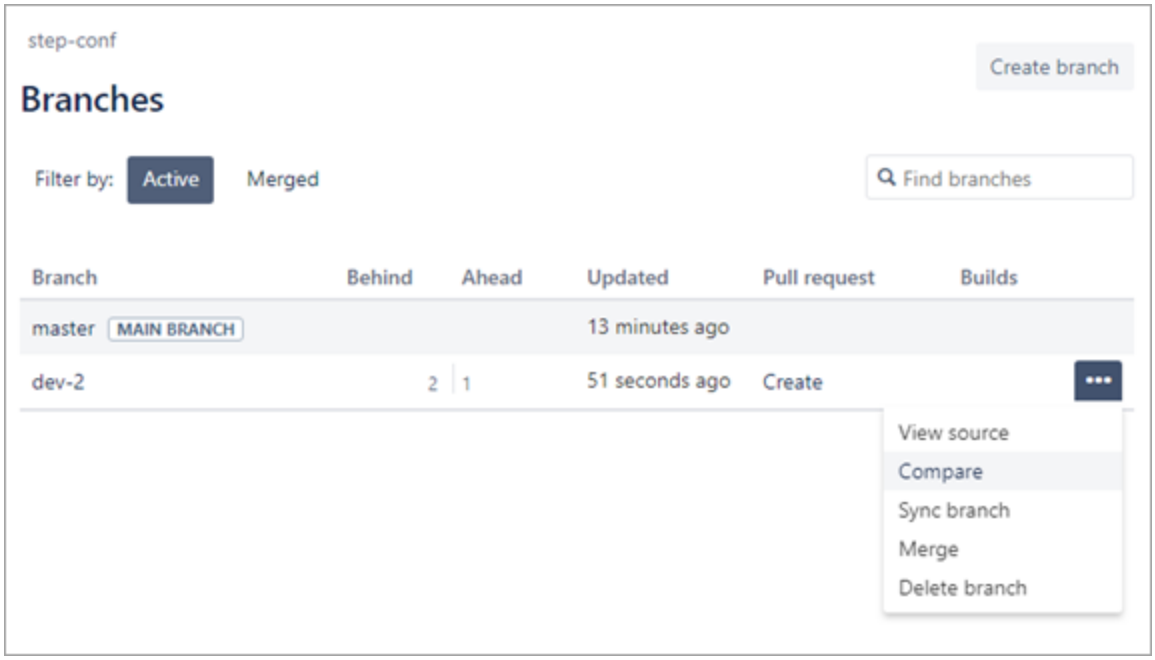


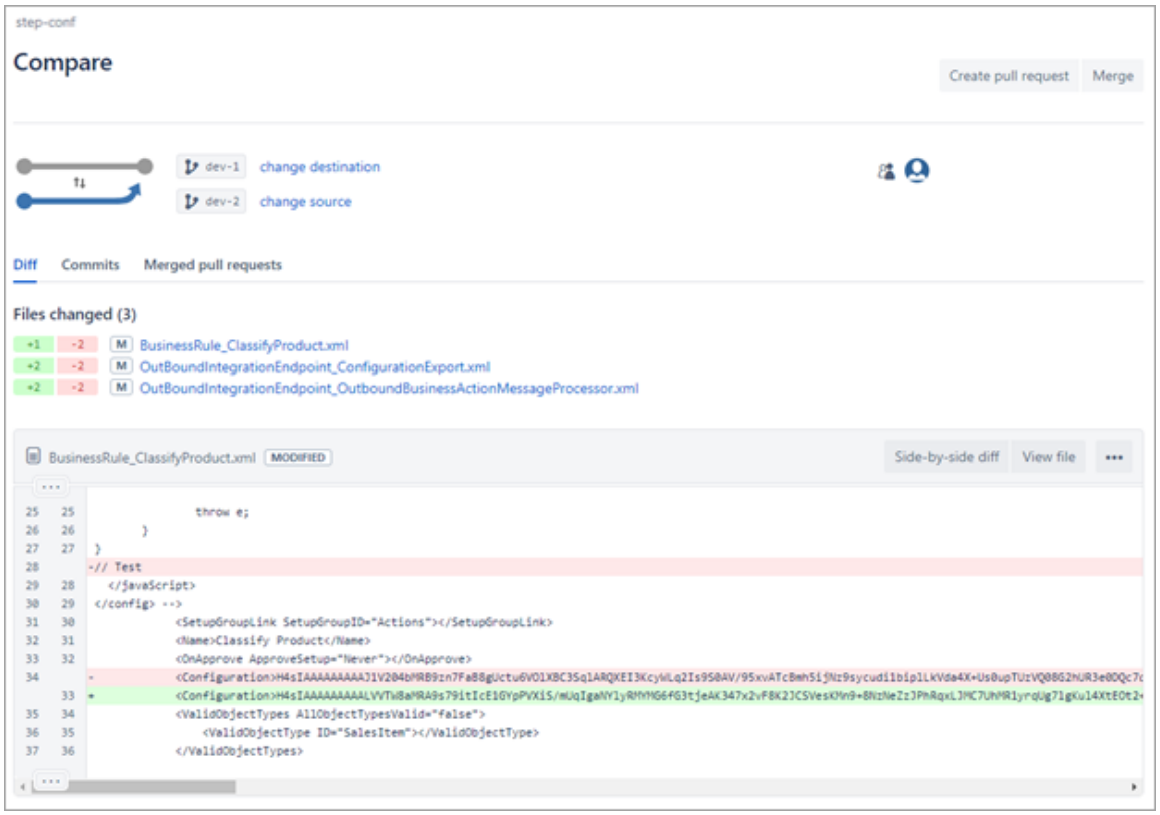
Such a setup can be used to ensure systems are in sync or only have expected differences. If differences are found, STEPXML files held in the Git branches can manually be imported one by one on a system that needs to be updated, or alternately, multiple files can be zipped and supplied to an inbound integration endpoint configured to use the 'STEPXML Joiner' pre-processor described in **Integration Endpoint Plugins** > **Inbound Integration Endpoint 'STEPXML Joiner' Pre-processor Plugin**.

Note: The VCS integration functionality offers no automatic dependency handling meaning 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.

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

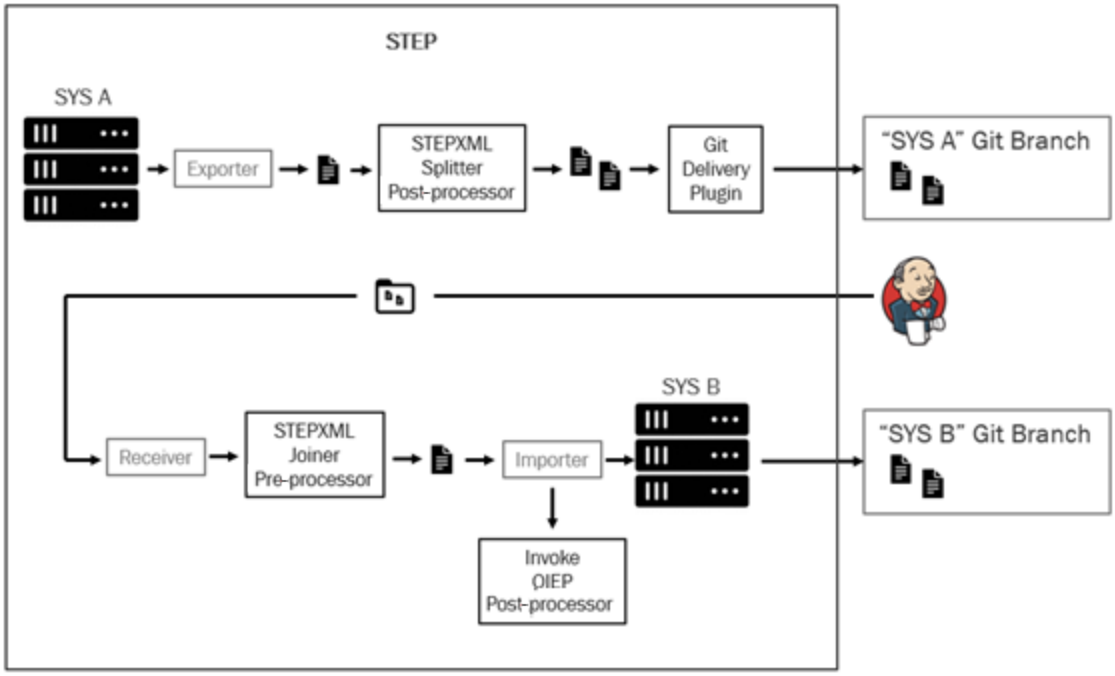
Branch comparisons can be made using the Git command line tool (refer to, for example, https://git-scm.com/docs/git-diff), or if a tool like Bitbucket is used, via a web interface as shown below.





Semi-automated System Synchronization

With a tool like Jenkins configured to monitor branches for changes, instead of just 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' automatically are 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, compare the branch with the 'SYS B' branch, produce a .ZIP file containing files from the 'SYS A' branch that differ, and then pass the .ZIP file to an inbound integration endpoint on 'SYS B' configured to use the 'STEPXML Joiner' and 'Invoke OIEP' plugins.

Example Jenkins job 'Build' shell script ($gituser, $gitpassword and $sysbpassword, $sysbuser defined via 'Username and password (separated)' bindings):

```
#!/bin/bash
timestamp=$(date +%s)
gitserver='bitbucket.org/john-smith/step-conf.git'

sysbstepserver='sys-b.domain.com'
hotfolder='/upload/hotfolders/ConfigurationManagement/in'


git clone https://$gituser:$gitpassword@$gitserver checkout_$timestamp
cd checkout_$timestamp
git checkout $sys-b
git checkout $sys-a
git pull
git diff -z $sys-b..sys-a --name-only --diff-filter=ACMRT | xargs -0 zip cpg_diff_
$timestamp.zip
sshpass -p $sysbuserpassword scp cpg_diff_$timestamp.zip
$sysbuser@$sysbstepserver:$hotfolder
```

The Jenkins job could further, via REST, invoke the inbound integration endpoint on 'SYS B' and monitor the import process, notifying human users if errors occur. Alternately, the inbound integration endpoint could be scheduled to run frequently, and an error reporter plugin 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 **VCS: Considerations and Limitations** topic for more information.

VCS: Considerations and Limitations

The Version Control System (VCS) integration 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 would, with the current solution, have to be produced by a configured Jenkins job, or a job in a similar tool, upon identifying objects present in the target system 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.
- Workflow definitions can, in some cases, not be updated if there are tasks for objects in the flow.

Maintaining Partial Data Sets on Lower Level DTAP Environments

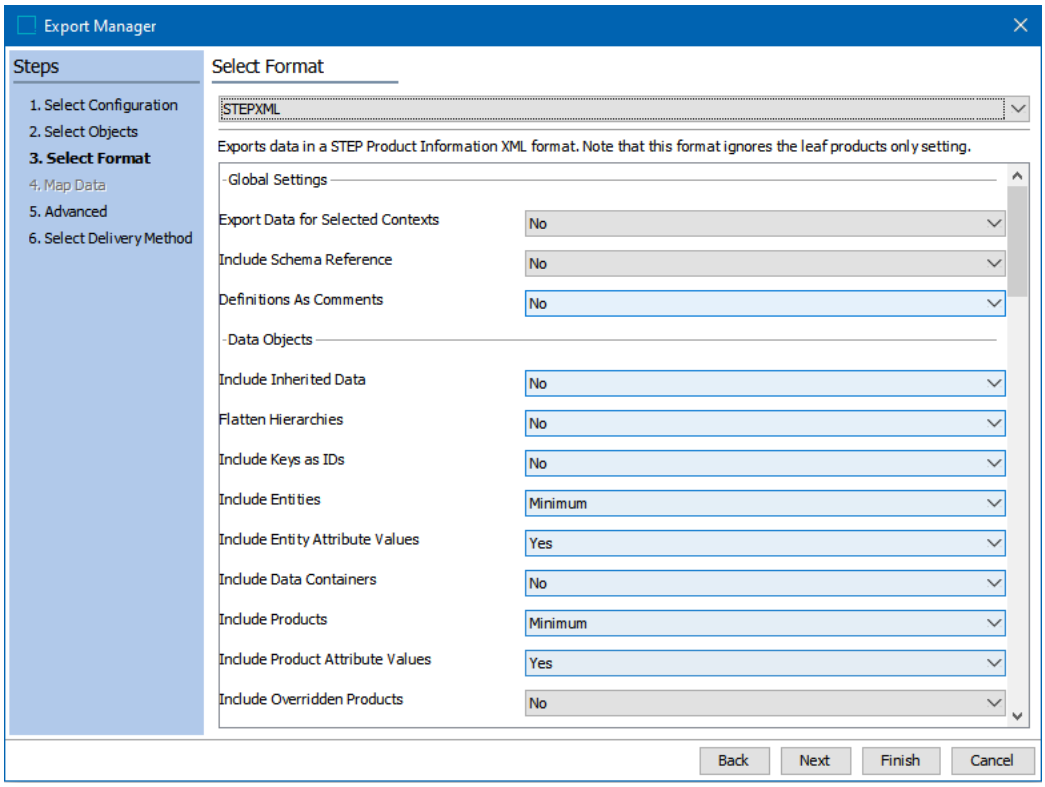
This section explains how you can keep Dev / QA / Sandbox systems up to date since these systems typically need all of the System Setup but only a small representative subset of the data from your production environment. Since you only transfer a subset of the data, this will make it much faster and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

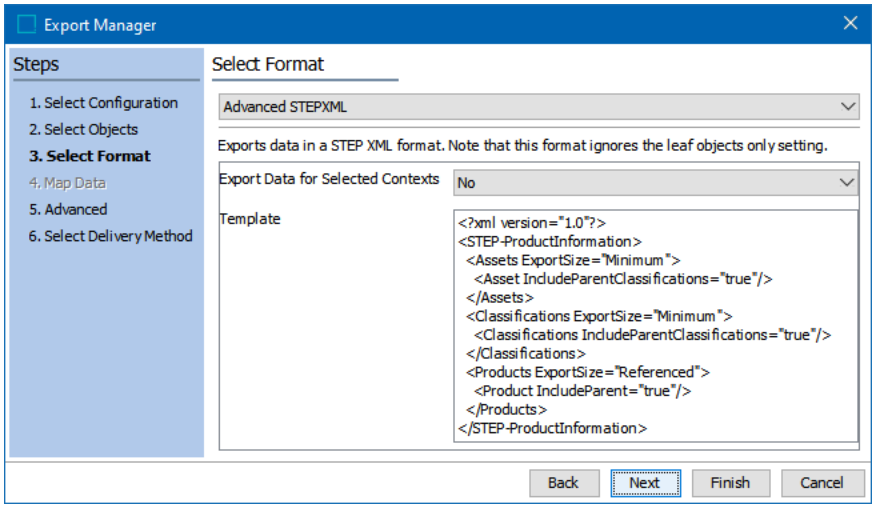
For such cases, Oracle Data Pump exports and imports cannot be used as it is not possible, with this technology, to do such things as only export certain hierarchies or data from certain STEP contexts. Further, Oracle Data Pump imports will overwrite any data created or modified in the target environment, which is often not desirable. Instead, the recommendation is to use STEPXML for transferring the 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 plugin. 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 plugin makes use of an XML 'output template' sometimes also referred to as a 'recorder file' while the STEPXML format plugin 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.

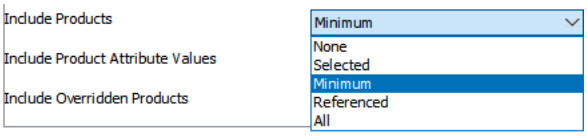
Export Manager examples shown below: STEPXML and Advanced STEPXML





Regardless of which STEPXML format plugin 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 plugin, 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 plugin 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>
```

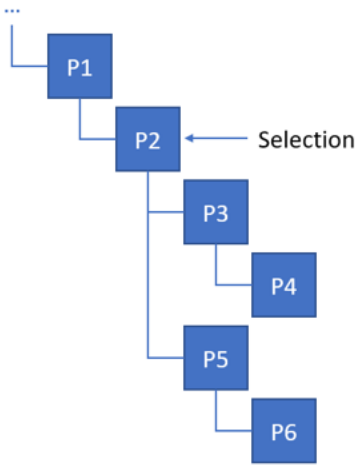
Export Size: All

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.

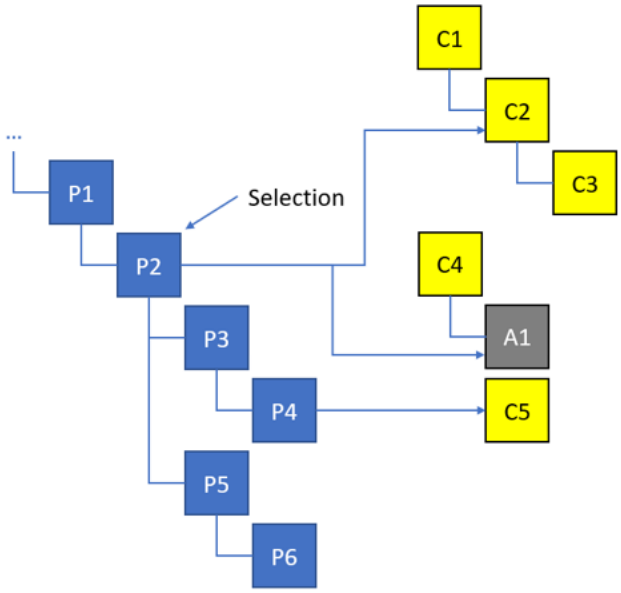
Export Size: Minimum

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:

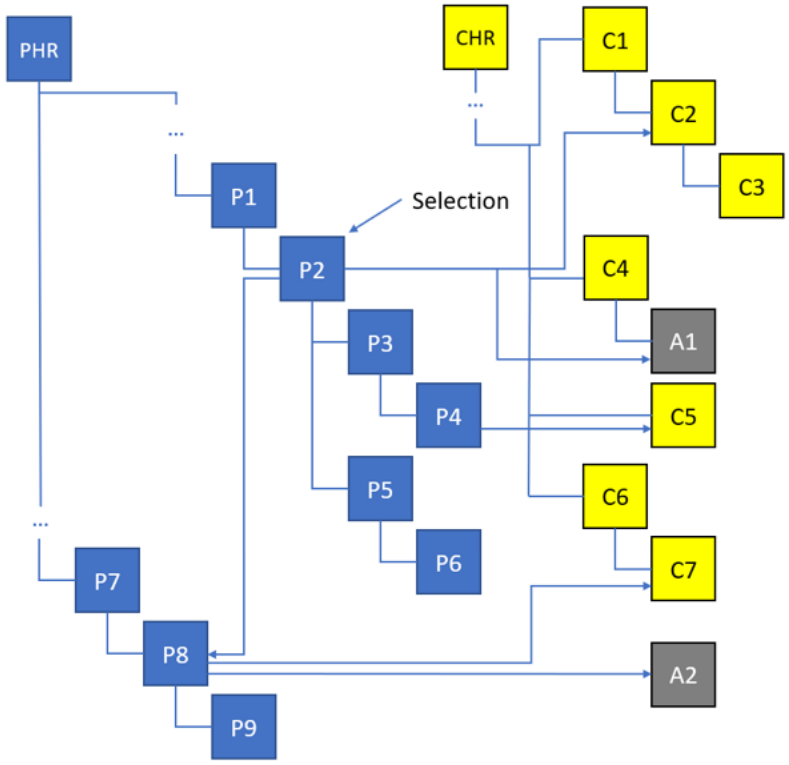
- All attributes used on the exported data objects
- All units used by the exported attributes
- All attribute groups that exported attributes are present in (and by default, all attribute groups up to the 'Attribute group root')
- All LOVs used by exported attributes

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>
```

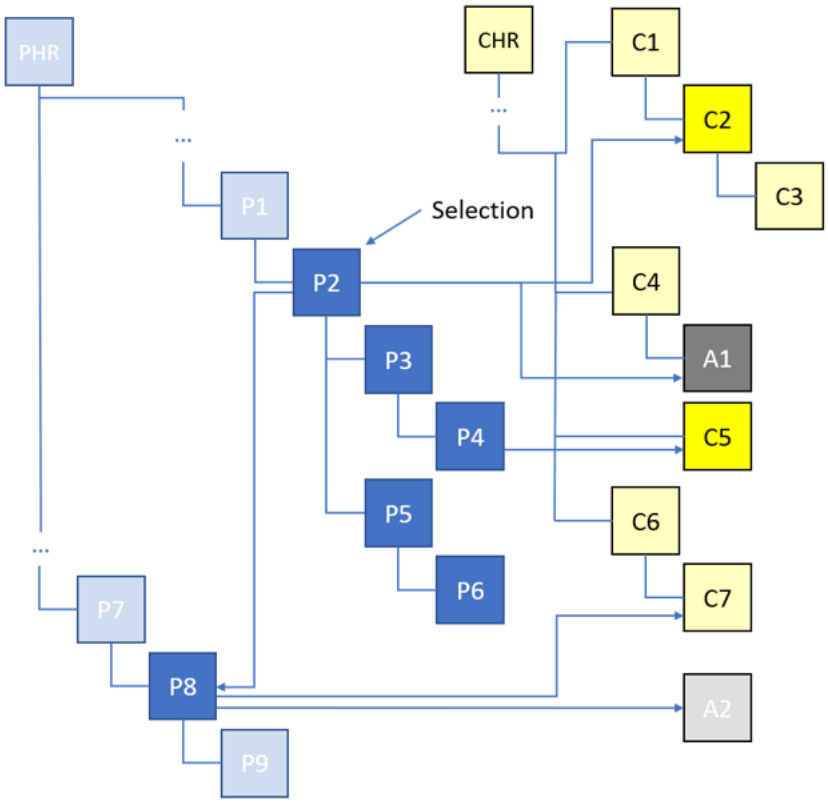
…will produce the following objects in the export:

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:



Notice again that the logic is not applied recursively. I.e., 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.

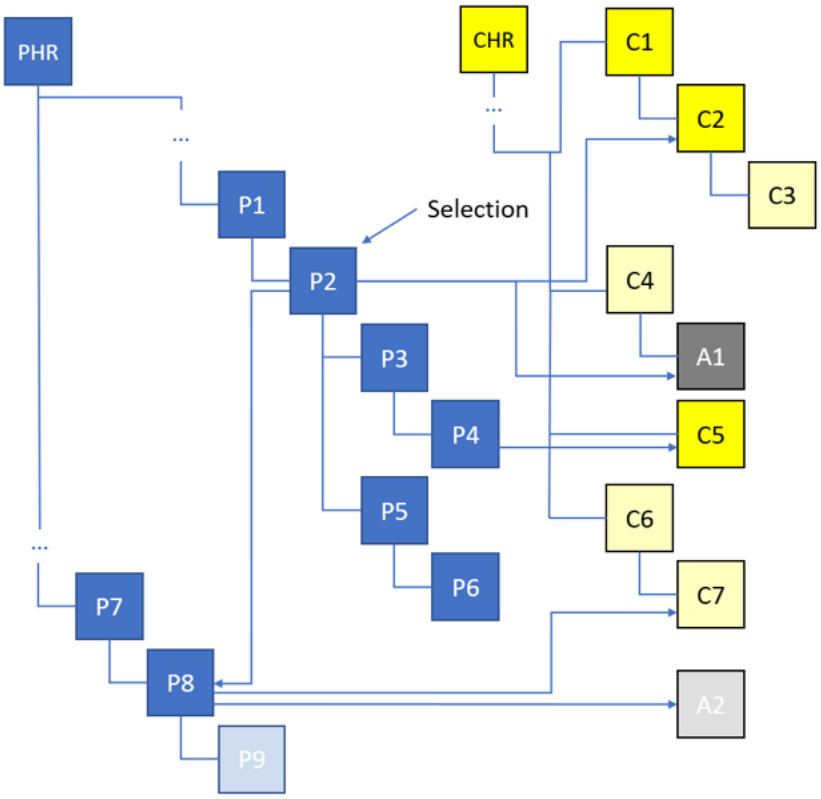
Including Ancestors

Reusing the last data example, it will not be possible to import files exported with the settings described above without 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 plugin, it is possible to specify that the exporter should include ancestor objects up to the hierarchy roots (‘PHR’ and ‘CHR’ in the diagrams).

For products, classifications, and entities, this is done 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, will be included in the exported file. This makes it possible to import the file on a system where the ancestor nodes do not exist in advance. This is illustrated below with non-exported nodes dimmed:

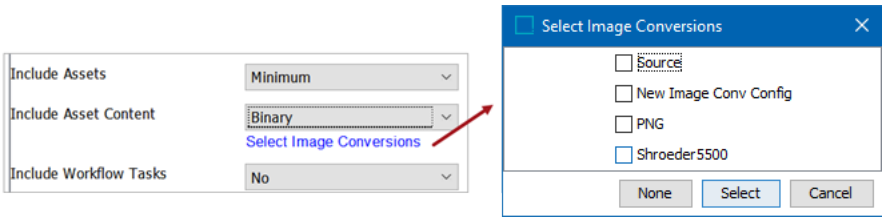


A similar attribute called 'IncludeParentClassifications' exists for assets. Setting this attribute to 'true' in an output template for the 'Asset' element will, when using the domain exporter (automatically enabled for In-Memory systems), cause classifications that a selected or referenced asset is 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. It is, however, possible to get the data included both when using the STEPXML and Advanced STEPXML format plugins. It is also possible to import the data again, creating or updating asset content.

With the STEPXML format plugin, 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>
```

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

To get name, values, references and classification links exported, the template for assets should look as follows:

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

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

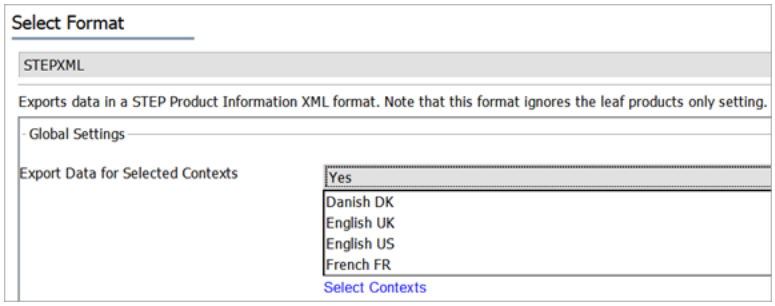
When exporting asset content, be aware that converted versions of the content cannot be imported – only the unconverted source (in the exported file, data for the unconverted source will be in an 'AssetBinaryContent' element for which the value of the 'ImageConversionConfiguration' 'ID' attribute is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

Important: Since including asset content can lead to very large files being created, seriously consider if asset content is strictly required in the systems that the data is being moved to.

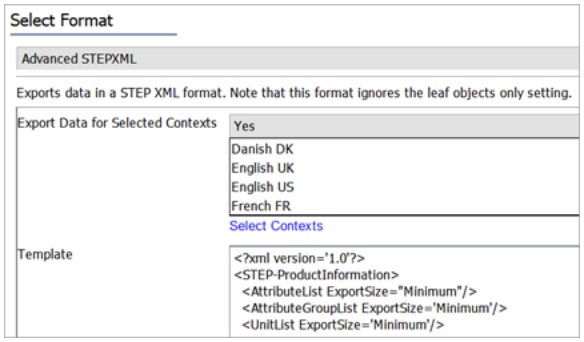
Cross Context Exports

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

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



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



Transferring Configuration and Data Between Systems

With the STEPXML export functionality described above, it is possible to 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, it is further possible to have ancestors to the exported nodes included, 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.

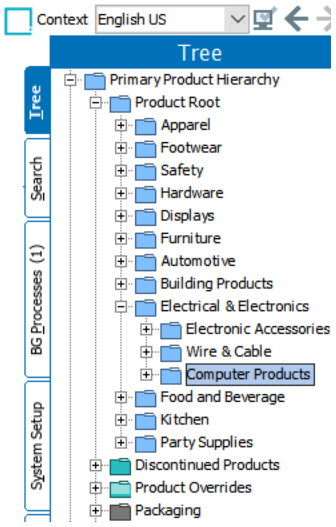
For configuration objects like attributes, it is also possible to use the 'Minimum' setting to avoid getting large amounts of attributes included that are not needed. If doing so, however, there is a high probability that attributes that are not directly used by the exported data but required for some 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 making it possible to easily compare 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.



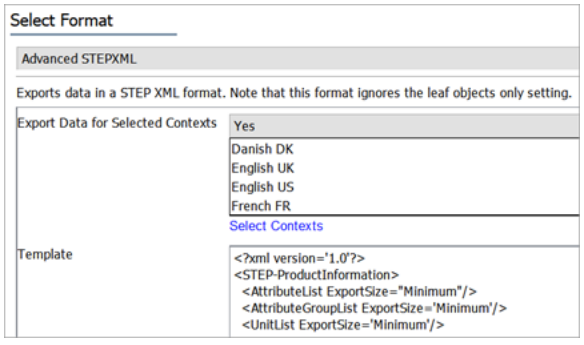
Consider the data hierarchy shown above. For this example, the goal is to export the 'Computer Products' category of products. And, in the exported file, include data for:

- 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.

For it to be possible 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 section of 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). Access documentation online access to these files.

System Configuration Export and Import

Important: The 'STEPXML Configuration Export' format plugin, 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 plugin, 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

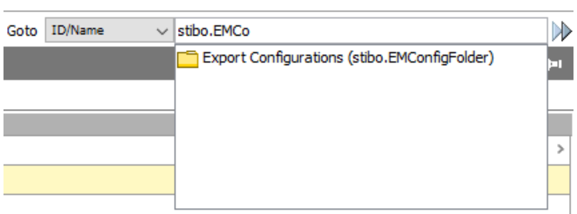
Follow the steps below to import the Export Configuration on the source system.

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.

```
<?xml version="1.0" encoding="utf-8"?>
<STEP-ProductInformation ContextID="Context1" WorkspaceID="Main">
  <ExportConfigurations>
    <ExportConfiguration ID="Configuration-Export">
      <Name>Configuration-Export</Name>
      <ClassificationReference ClassificationID="stibo.EMConfigFolder"/>
      <Configuration>H4sIAAAAAAAAAKVXbW/bOAz+319hGEWxG5Bm3acD2qbIkrQLLul6al
```

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

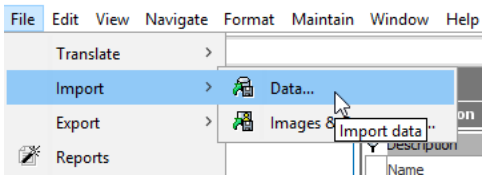
2. The Export Configuration will be imported below a Classification with ID 'stibo.EMConfigFolder.' This classification exists by default in all STEP systems; however, it could have been deleted. Check that the classification exists via the workbench **Goto** field as shown below.



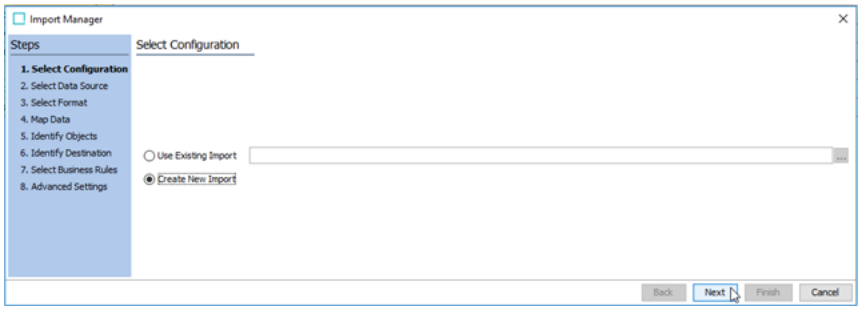
If the classification does not exist, open the 'Configuration-Export.xml' file in a text or XML editor, change the value for the 'ClassificationReference' element 'ClassificationID' attribute to the ID of the Classification to have the Export Configuration imported into and save the changes.

```
<?xml version="1.0" encoding="utf-8"?>
<STEP-ProductInformation ContextID="Context1" WorkspaceID="Main">
  <ExportConfigurations>
    <ExportConfiguration ID="Configuration-Export">
      <Name>Configuration-Export</Name>
      <ClassificationReference ClassificationID="stibo.EMConfigFolder"/>
      <Configuration>H4sIAAAAAAAAAKVXbW/bOAz+319hGEWxG5Bm3acD2qbIkrQLLul6a
    </ExportConfiguration>
```

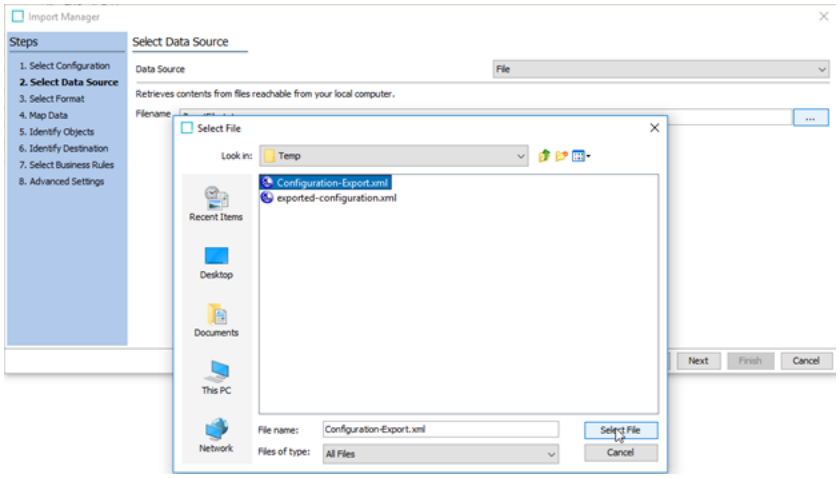
3. In the STEP Workbench, from the File menu select **Import** and then select **Data…** .



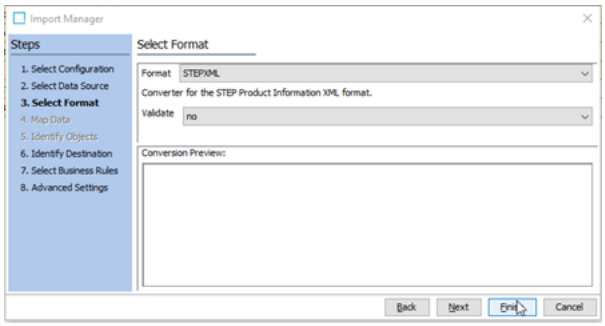
4. In the **Select Configuration** step of the Import Manager, click the **Next** button.



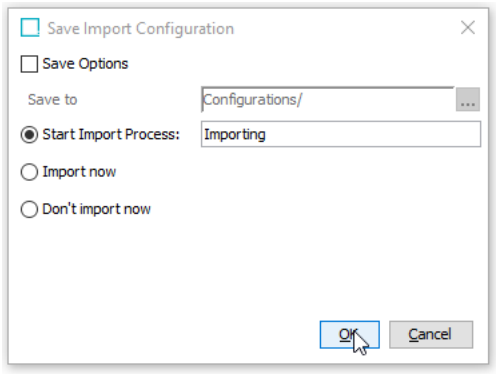
5. In the **Select Data Source** step of the Import Manager, click the ellipsis button (...), browse to where the 'Configuration-Export.xml' file is saved, select it and click the **Next** button.



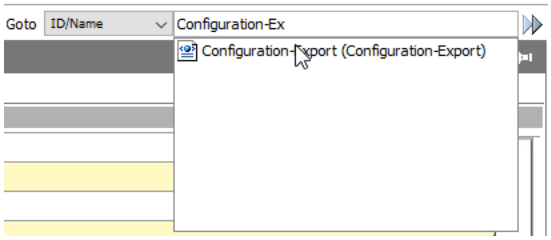
6. In the **Select Format** step of the Import Manager, STEP should automatically recognize the file as a STEPXML file. Confirm that Format is 'STEPXML' and click **Finish**.



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



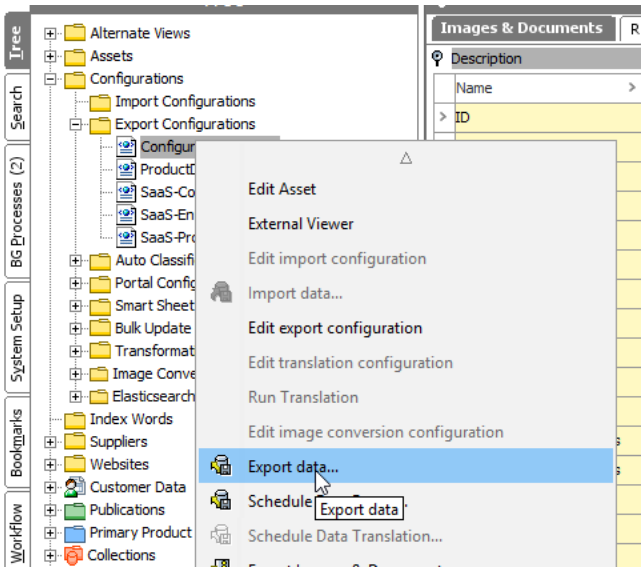
8. After the import process has finished, locate the 'Configuration-Export' Export Configuration. For example, you can use the STEP Workbench **Goto** field as shown below.



Using the Configuration-Export Export Configuration

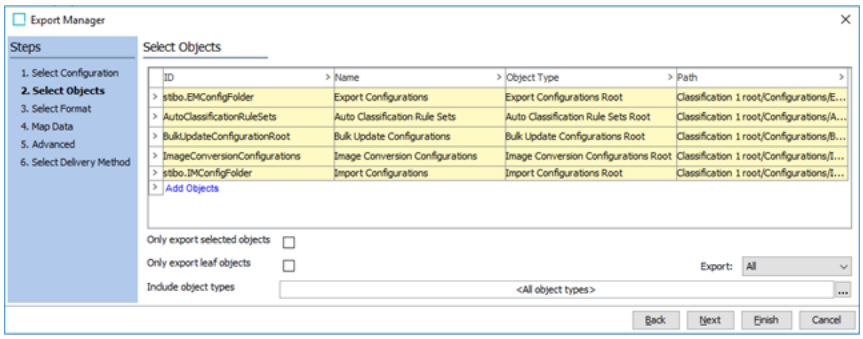
The 'Configuration-Export' export configuration must be modified so that the exported STEPXML file will include all objects required for the file to be imported on the target system without errors. Follow the steps below to configure and run the export.

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, it will make sense to launch an additional workbench instance so that it is possible to add objects in the Export Manager in one and browse the Tree and System Setup in another. The nodes that must be added are:

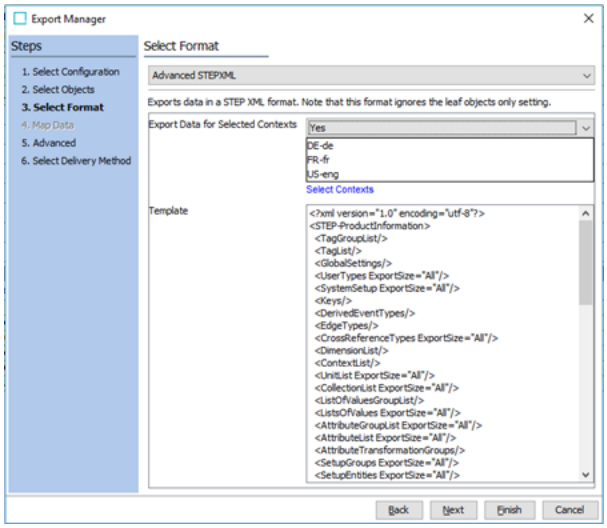
   - All Classifications holding Export Configurations (immediate parents)
   - All Classifications holding Import Configurations (immediate parents)
   - All Classifications holding Bulk Update Configurations (immediate parents)
   - All Classifications holding Image Conversion Configurations (immediate 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)



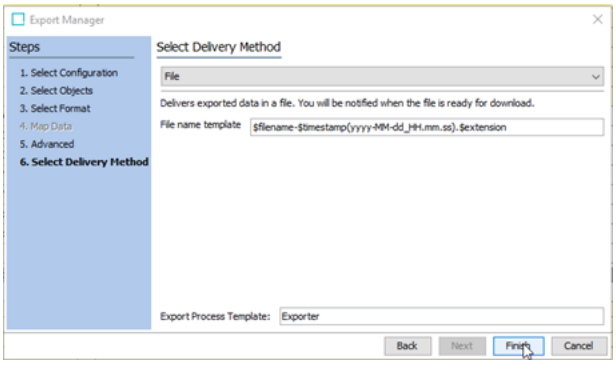
Note: With this export configuration, only the IDs and Names of these nodes and their ancestors up to the type specific root will be exported (i.e., no attribute values, references, etc.).

Click **Next** once all required nodes have been added.

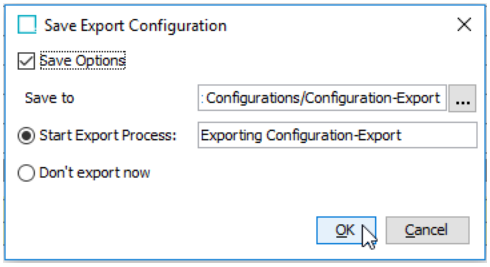
3. In the **Select Format** step of the Export Manager, under 'Export Data for Selected Contexts,' select the Contexts for which data should be exported. Notice that this is only relevant if the configuration objects and data nodes are dimension dependent (e.g., language-specific attribute names and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



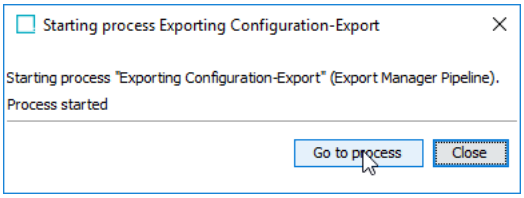
4. In the **Advanced** step of the Export Manager, make sure that **Workspace** is set to Main and click **Next**.
5. In the **Select Delivery Method** step, ensure that the 'File' option is selected and click **Finish**.



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.



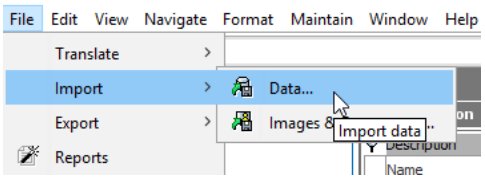
8. Once the export process has completed, click the save icon in the lower right corner of the background process editor to save the file to your local hard drive.

Importing the Configuration on the Target System

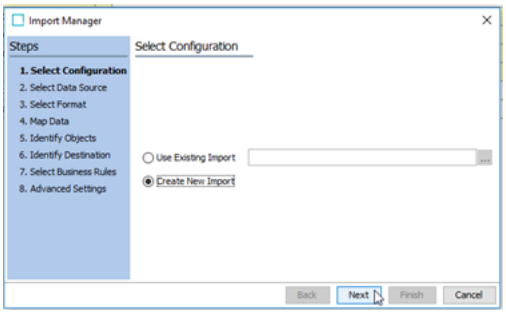
This section describes how to import the exported STEP configuration on the target system. Follow the steps below to import the configuration on the target system.

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.

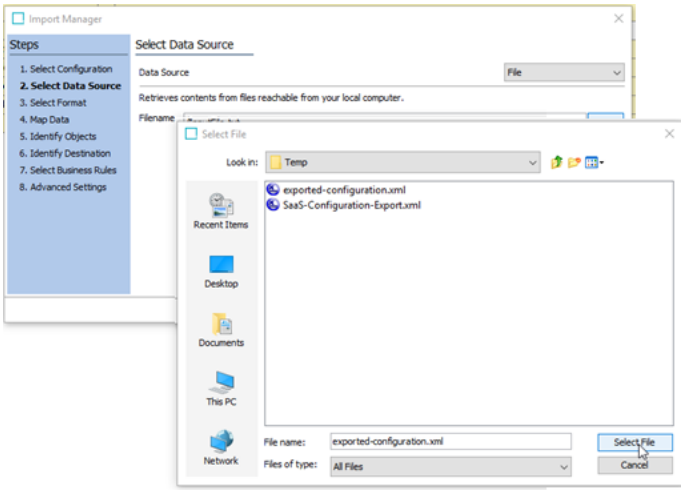
2. Launch the STEP Workbench for the target system, from the File menu select **Import** and then select **Data…** .



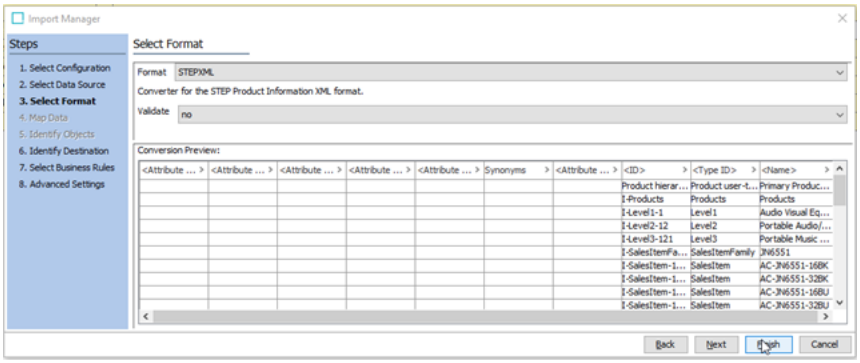
3. Click **Next** in the **Select Configuration** step of the Import Manager.



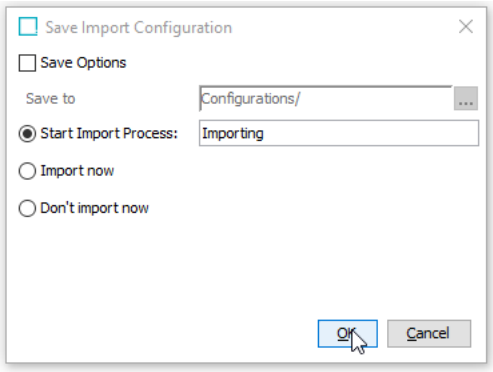
4. In the **Select Data Source** step of the Import Manager, click the ellipsis button (**...**), browse to where you have saved the STEPXML file exported from the source system, select it, and click **Next**.



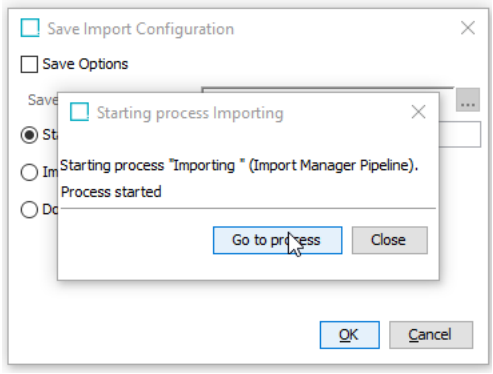
5. In the **Select Format** step of the Import Manager, STEP should automatically recognize the file as a STEPXML file. Confirm that Format is 'STEPXML' and click **Finish**.



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



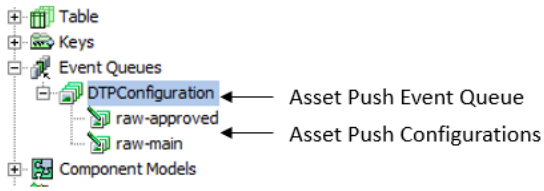
7. Click **Go to process** in the dialog that appears and inspect the execution report for warnings and errors.



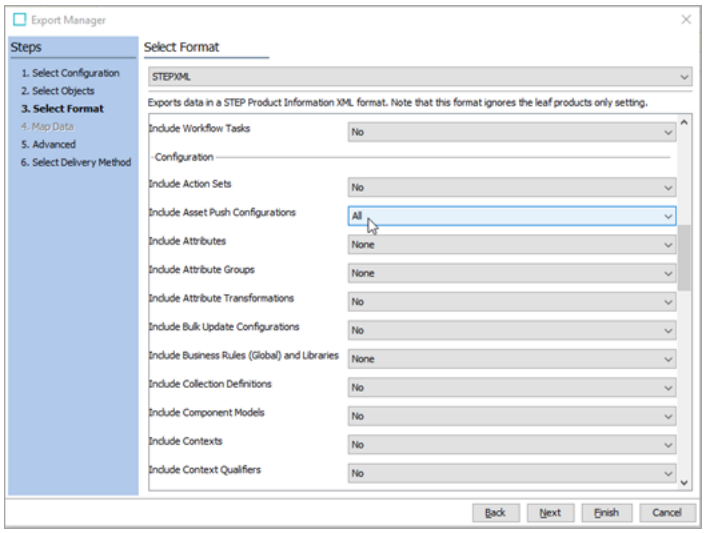
Taking Post Import Actions

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

- It is currently not possible to transfer Workflow Profiles via STEPXML. If Workflow Profiles have been defined in the source system, these must be re-created manually on the target system.
- Asset Push Configurations are deliberately not exported as it is not currently possible to transfer the parent Asset Push Event Queue type via STEP XML. The event queue(s) must be created manually on the target system after which the Asset Push Configurations can be transferred via STEPXML or recreated manually. The screenshot below shows an Asset Push Event Queue and two Asset Push Configurations in the workbench System Setup.



If Asset Push Configurations are to be transferred via STEPXML, these can be exported to STEPXML via the Export Manager and the STEPXML format plugin as shown below (blank selection, 'No,' or 'None' for everything except for 'Include Asset Push Configurations,' which must be set to 'All').



- User objects have deliberately been left out. Users maintained in STEP that are referenced from Integration Endpoints and Event Processors and used for M2M integrations must 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. 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.

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 exact 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 plugin 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

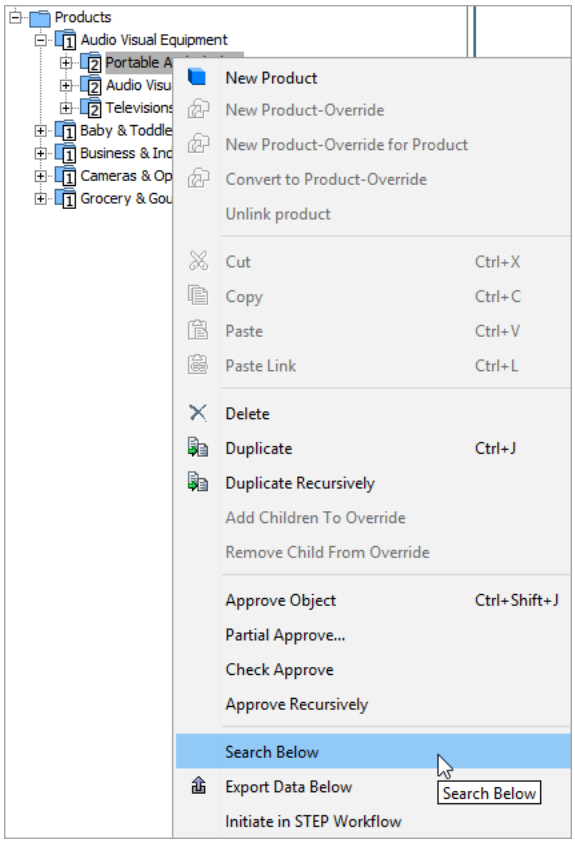
If the sample product data references entities, referenced entities and their ancestors can be included by extending the output template to be:

```
<?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>
```

Notice that 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.

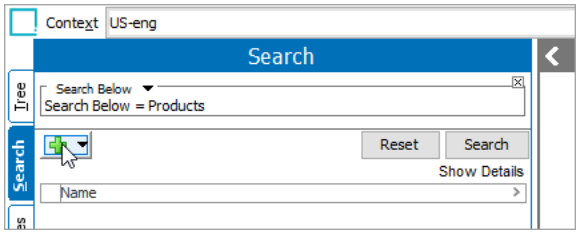
The first option is to select one or more category / family product nodes. With the template export configuration, all product nodes below the selections will be exported, so use the workbench **Search Below** feature to ensure that the number of product objects below the selections is not too large.



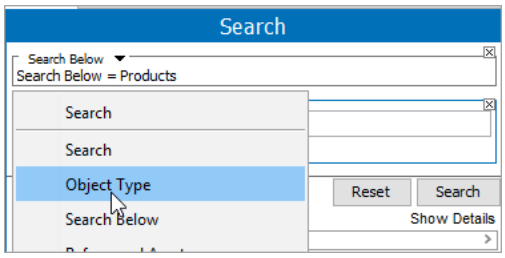
If a wider spread of product sample data is required, it is possible to select a category higher in the hierarchy that would contain too many product objects to be exported fully and sample products below that to use as the selection. Follow the steps below to create a Collection with the sample products.

1. Right-click the desired category in the STEP Workbench Tree and select **Search Below** from the context menu (as shown above).

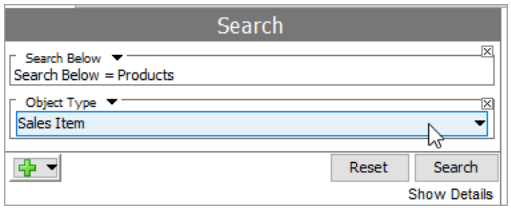
2. In the **Search** tab, click the button with the green plus sign to add another criteria to the search.



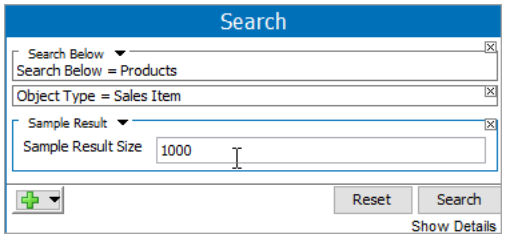
3. Click the search criteria type selector and choose **Object Type** from the menu.



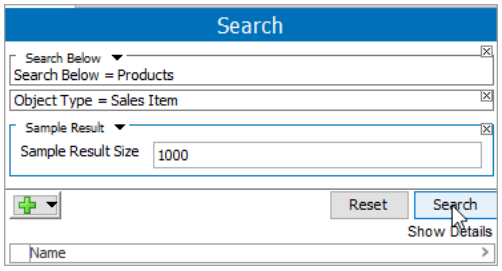
4. Select your leaf product object type.



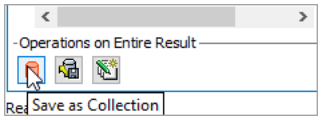
5. Add another criteria of the type **Sample Result**, and set the desired sample result size.

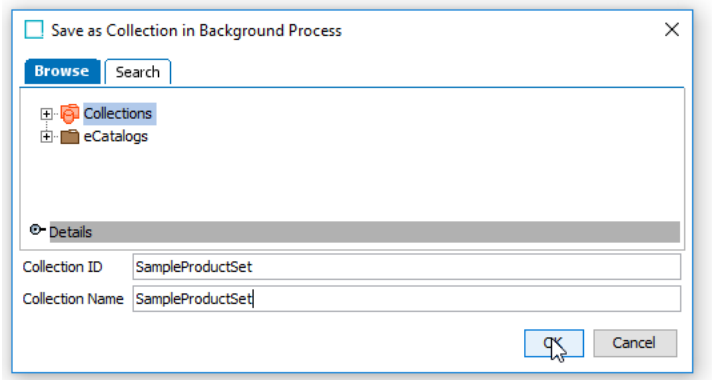


6. Execute the search.



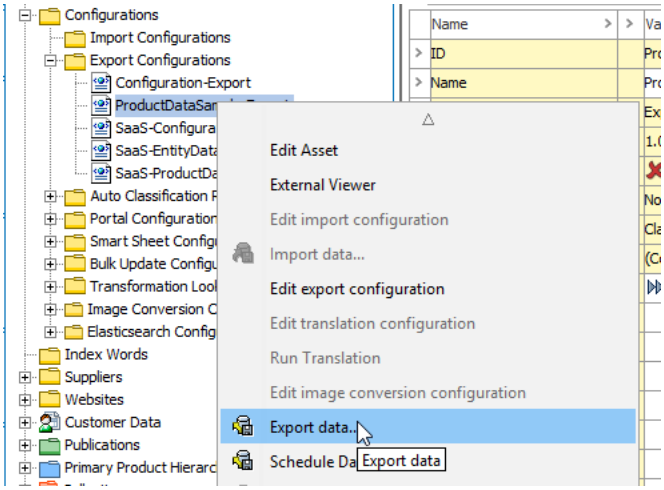
7. Click the **Save as Collection** button at the bottom of the search panel to store the result in a Collection.



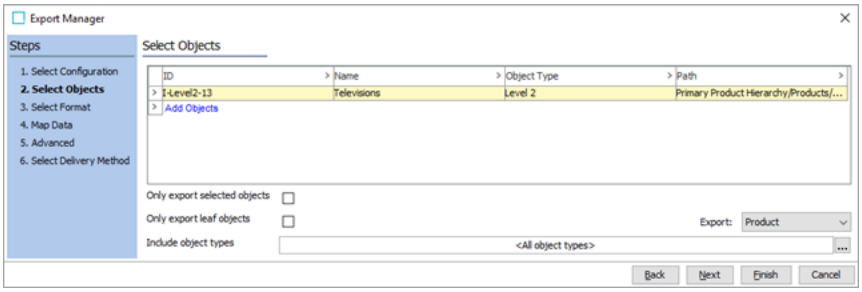


After having decided which approach to use (and having created the collection with sample data, if this is the desired approach), follow the steps below to export the sample data.

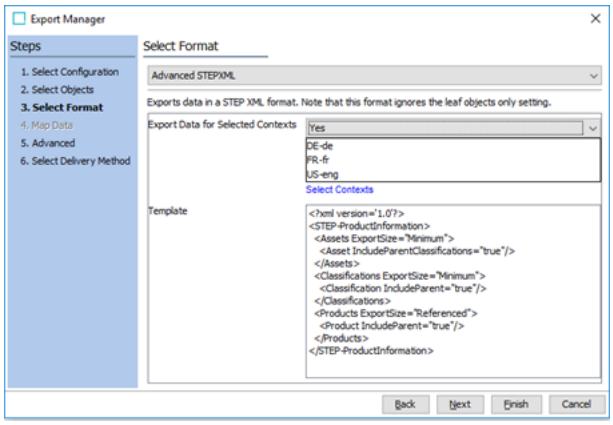
1. Locate the imported 'ProductDataSample-Export' export configuration in the STEP Workbench Tree, right-click it and select **Export data…**.



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**.

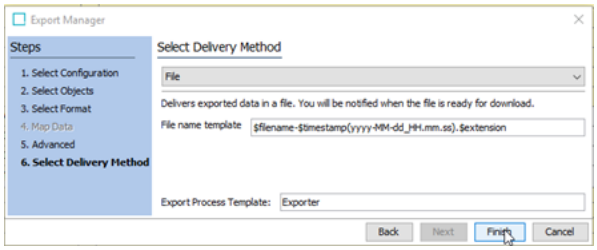


3. In the **Select Format** step, under **Export Data for Selected Contexts**, select the Contexts for which data should be exported if there is dimension dependent data. Click **Next** to continue.

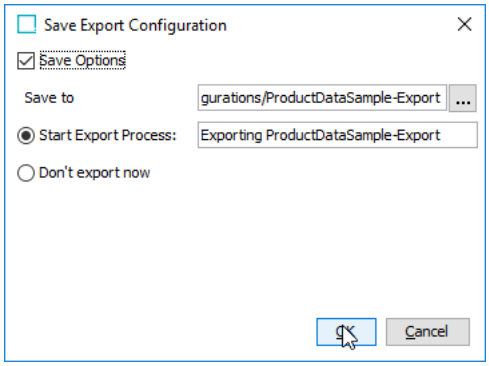


4. In the **Advanced** step, make sure that **Workspace** is set to Main and click **Next**. It might be relevant to export data from Approved, but this then requires that the produced STEPXML file is manually modified prior to being imported.

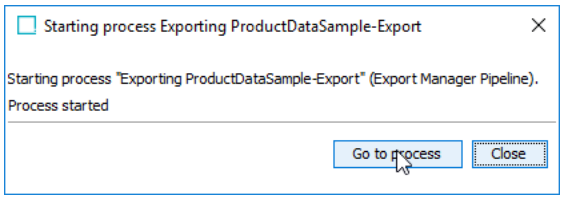
5. In the **Select Delivery Method** step of the Export Manager, ensure that the **File** option is selected and click **Finish**.



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. Once the export process has completed, click the save icon in the lower right corner of the background process editor to save the file to your local hard drive.

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 in this section of documentation.