GDSN PROVIDER USER GUIDE

StiboSystems

STEP Trailblazer 8.2

Table of Contents

StiboSystems

Introduction

About This Guide

This guide describes STEP GDSN Provider including how to setup data pools and use data pools to send data to different data recipients.

We recommend that you familiarize yourself with the key concepts of this guide.

The guide assumes that:

- Users have a basic knowledge of STEP.
- Users have a AS2 server setup with an In hotfolder and Out hotfolder to be able to send and receive data.
- Users and trading partners have adopted the GS1 GTIN, GLN, Global Data Dictionary (GDD) and Global Product Classification (GPC) standards.
- Users have a STEP GDSN Provider license or the STEP GDSN Provider license version 2.

If you need information on other STEP components, see the online help or the specific user guides.

GDSN Provider

The Global Data Synchronization Network (GDSN) is an internet-based global network and global registry that enables secure and continuous data synchronization between trading partners. This connection is made via a network of interoperable GDSN-certified data pools.

Within the GDSN network, trade items are identified using target markets and a unique combination of the GS1 Identification Keys called Global Trade Item Numbers (GTIN) and Global Location Numbers (GLN).

GDSN enables trading partners to synchronize data. Any changes to the data pool made by one company are automatically available to all of its trading partners.

GS1 Global Registry®

GS1 describes the GS1 Global Registry as the GDSN information directory that details who has subscribed to trade items or party data, guarantees the uniqueness of the registered items and parties, and ensures that all data pools in the network are complying with a standards-based set of validation rules.

How GDSN Works with Data Pools

GS1-certified data pools are electronic catalogs of standardized item data. They serve as a source and/or a recipient of master data.

The GDSN works together with data pools in the following way:

1. The data provider selects a source data pool and the data recipient selects a recipient data pool as a single point of entry to the GDSN.
2. The data provider registers product and company information in its source data pool. This information is also registered in the GS1 Global Registry.
3. The data provider agrees with a data recipient to synchronize data from the provider to the recipient. The provider then makes a publish request to the data pool, so that relevant registered items are sent to the recipient.
4. The data recipient makes a subscription request (Catalog Item Subscription -CIS).
5. If the subscription criteria match items that are registered in the GS1 Global Registry, the recipient's data pool is notified using a Catalog Item Notification (CIN) message, and then the synchronization takes place. Data is published from the data provider's data pool to the recipient's data pool.
6. After receiving the data, a Catalog Item Confirmation message (CIC) is sent from the data recipient to the data provider.

Common Standards

Standard	Description
GTIN	A Global Trade Item Number is a global identification number that can be used by a company to uniquely identify trade items. Trade items are defined as products or services.
GLN	The Global Location Number is a unique 13-digit identification number. The GLN can be used to identify a company's physical location and to identify corporate entities as well as a company's legal and functional entities. Each data provider and each data recipient has their own unique GLN that is used when publishing and subscribing for data.
GPC	To ensure that products are classified correctly and uniformly, GDSN uses GS1 Global Product Classification (GPC), a system that gives data providers and data recipients a common language for grouping products in the same way everywhere in the world.

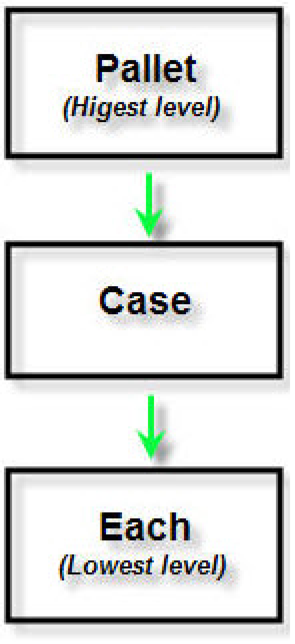
Key Terminology

Term	Definition
CIS	Catalog Item Subscription. Subscription message sent by a data recipient to establish a request for data.
CIN	Catalog Item Notification. Notification message used to transmit new or updated item information from the data provider to a data recipient. The CIN includes the requested product data.
CIC	Catalog Item Confirmation. Confirmation message sent to the data provider by a data recipient.
Synclist	Synchronization List. List that includes all synchronized catalog items (GTIN, GLN, TM). Keeps track of where data has been notified.

About Trade Item Hierarchies

The trade item hierarchy - or the packaging hierarchy - describes the relationship between trade items that contain other trade items, and it describes on which level in the hierarchy each item fits in. A trade item could, for example, belong to one of the following levels: base unit (Each), case, and pallet. Regardless of how many levels are in a hierarchy, the final level must be a base unit.

A parent item is an item that contains lower level trade items (children) in a packaging hierarchy. A child item is an item with a higher level trade item (parent) in a packaging hierarchy. A child item can have multiple parents, and it can therefore be included in many packaging hierarchies.



For detailed information about GDSN, search the web.

GDSN Component Model Overview

The GDSN Provider uses a component model to define the objects, references and attributes of the component. These elements define the configuration of the GDSN Provider component, specify how STEP communicates with the GDSN, and store the status of the products within the GDSN.

There are two options for configuring the GDSN Component Model. It can be configured manually or through the **Easy setup of GDSN Component Model** dialog. Configuring the component model through the use of the dialog is recommended as most settings are related to the internal workings of the GDSN solution and can be set automatically. For more information about setting up the GDSN component model, see the **Setting Up the GDSN Component Model** section.

Primary Elements of the GDSN Component Model

The following section describes the primary elements of the component model. For a detailed list of all elements of the component model, see the **GDSN Provider Component Model Elements** section.

GDSN Product

GDSN Products define the object types of the products and packaging objects that can be registered and published to the GDSN. GDSN products must have a GTIN attribute. The product object types must be configured prior to setting up the GDSN component model.

Target Market Object Type

A GDSN product is registered to a target market. The target market defines which STEP contexts to use when product data are extracted for that particular market. The context is determined by a reference from the target market to the context. The target market has an attribute that contains the code of the target market. This code must follow the naming scheme for target markets defined by the GDSN data pool. Target markets are defined per data pool and all target markets for a data pool have the same parent object in STEP.

Recipient

A GDSN product is published to a data recipient. The recipient has a GLN attribute for the GLN of the data recipient. Recipients are defined per data pool and all recipients of a data pool have the same parent object in STEP.

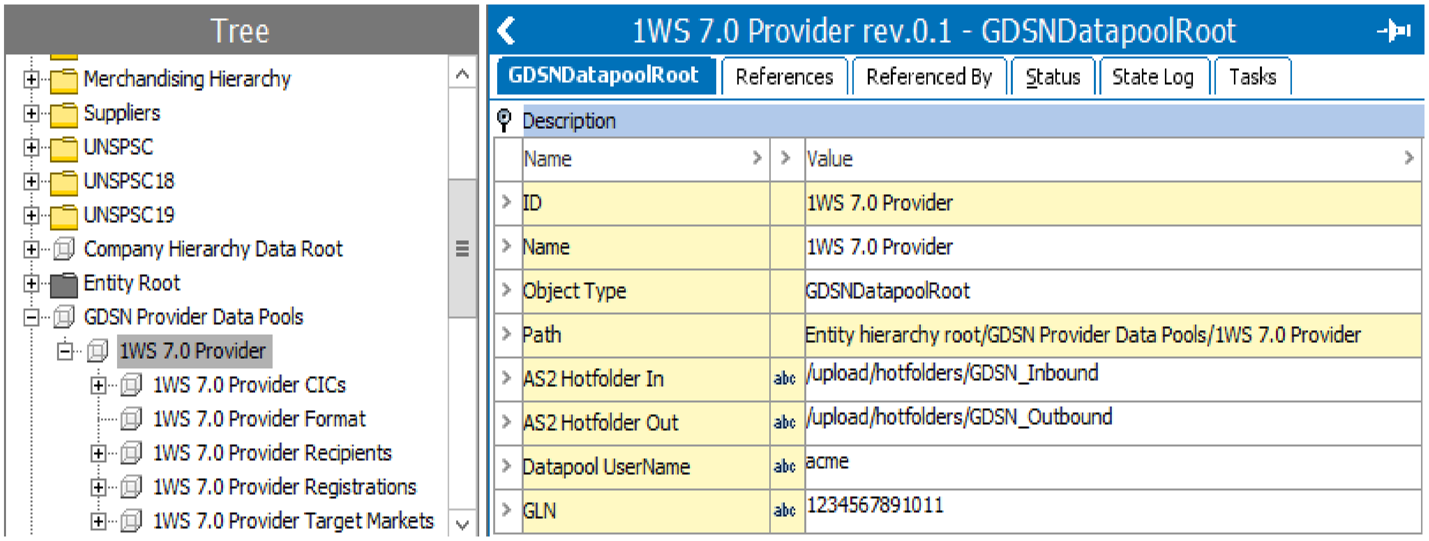
Data pool

The Data pool object represents the source GDSN data pool where the GDSN products are registered. The data pool has a GLN attribute that holds the GLN of the data pool. The data pool defines the format through a reference to a format object. The data pool has an attribute for the GDSN data pool GLN and for the GDSN data pool user name.

Note: The GDSN data pool user name is not a STEP user name.

Communication between STEP and the GDSN data pool takes place through hotfolders. You configure the in and out hotfolders on the data pool in two attributes: GDSN AS2 hotfolder In and GDSN AS2 hotfolder Out.

The following screenshot shows the expected children of a data pool entity node. The GDSN Provider component will not function correctly if these objects do not exist. The default configuration setup creates these entities as children of the data pool.



Name	Value
ID	1WS 7.0 Provider
Name	1WS 7.0 Provider
Object Type	GDSNDatapoolRoot
Path	Entity hierarchy root/GDSN Provider Data Pools/1WS 7.0 Provider
AS2 Hotfolder In	/upload/hotfolders/GDSN_Inbound
AS2 Hotfolder Out	/upload/hotfolders/GDSN_Outbound
Datapool UserName	acme
GLN	1234567891011

Format

This is the data pool format. The GDSN Provider is format independent, which means that all GDSN data pool specific formats are configured individually. The GDSN data pool configurations are stored in a Format object. The Format object can contain format specifications, for example, for 1 World Sync. You can create any number of Format objects.

The format defines how the exported STEPXML of a GDSN product is translated to the XML format expected by the relevant GDSN data pool. The format also defines how the XML messages received from the GDSN data pool are translated into messages that the GDSN Provider understands. The format supplies an XML schema for the XML format the GDSN data pool understands. The schema is stored in an asset that is linked to the format through a reference.

CIC (Catalog Item Confirmation)

The CIC object type is used to create objects that can run in a workflow that handles CIC messages. The CIC object contains a reference to the recipient that returns the CIC message and to the registration. You can place part of the CIC message in attributes on the CIC object. CIC objects are specified per data pool and all CIC objects have the same parent object in STEP.

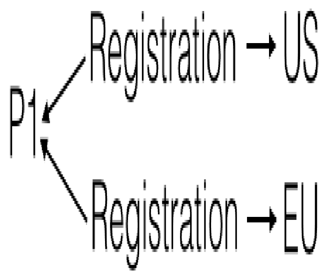
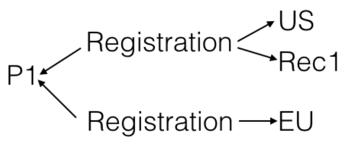
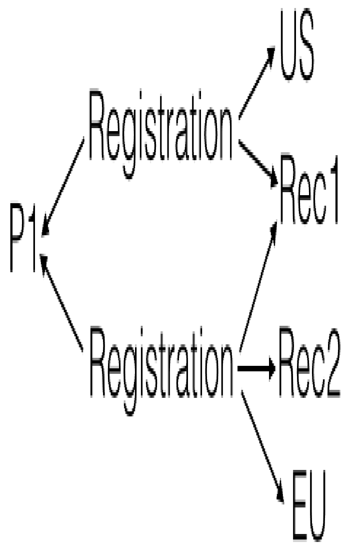
The CIC object stores the CIC status that is returned to a recipient when a product needs to be corrected. When data needs to be corrected, a CIC workflow is started that handles the correction. A CIC object is only created when a CIC workflow is started. If a returned CIC message does not require any data provider actions, no CIC object is created.

Status

The component model stores the registration and publishing status of the products in GDSN. Once a product is registered for a given target market, a registration entity is created with a reference to the product and the target market. The registration entity holds the status of the product registration of the given target market.

If the product is later registered to another target market, a new registration entity is created that refers to the product and the new target market. Two registration entity objects then exist for the product: one for the first target market and one for the second target market.

If a product is later published to one or more recipients in a target market, a reference is created from the registration entity that references the target market to each of the recipient entities. The status of the product published to the recipient is stored on the reference from the registration to the recipient. This makes it possible to record, for example, that one recipient has accepted the published product while another recipient has sent a CIC message requesting a correction of an attribute.

Product registered in two markets	Published to recipient Rec1 in US market	Published to recipients Rec1 and Rec2 in EU market
 $$\begin{tikzcd} & \text{Registration} \arrow[ld] \arrow[r] & \text{US} \\ \text{P1} & & \\ & \text{Registration} \arrow[lu] \arrow[r] & \text{EU} \end{tikzcd}$$	 $$\begin{tikzcd} & \text{Registration} \arrow[ld] \arrow[r] \arrow[rd] & \text{US} \\ \text{P1} & & \text{Rec1} \\ & \text{Registration} \arrow[lu] \arrow[r] & \text{EU} \end{tikzcd}$$	 $$\begin{tikzcd} & & \text{US} \\ & \text{Registration} \arrow[ld] \arrow[ru] \arrow[rd] & \\ \text{P1} & & \text{Rec1} \\ & \text{Registration} \arrow[lu] \arrow[ru] \arrow[r] \arrow[rd] & \text{Rec2} \\ & & \text{EU} \end{tikzcd}$$

Package hierarchy

The component model uses product to product references to model package hierarchy relationships. Parent items have a product to product reference to the children in the package hierarchy. Each level in the package hierarchy, for example, a Pallet to Case or a Case to an Each, can use the same or different product to product reference types. You can therefore configure the component model to use one product to product reference type for the

Pallet to Case relationship and another product to product reference type for the Case to Each relationship. An attribute on a package hierarchy reference type specifies the quantity of the next level in the package hierarchy. Another attribute on the reference holds the registration status.

Setting Up the GDSN Component Model

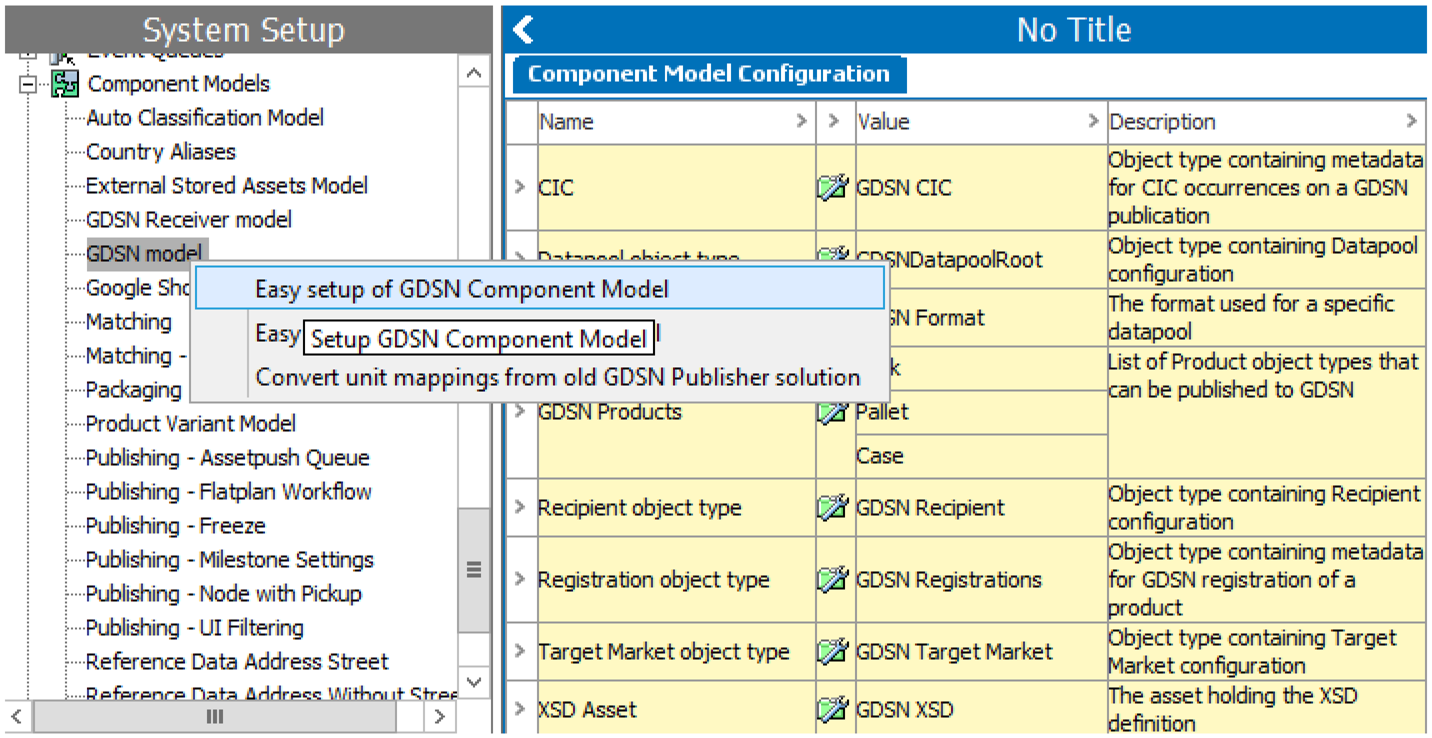
The GDSN component model currently consists of multiple aspects, and they all have to be configured. Many of the aspects are dependent on each other and related to the internal workings of the GDSN solution. Most of them, however, can be configured automatically.

We therefore recommend that you use the **Easy setup of GDSN Component Model** dialog to configure the component model. When you use the dialog you specify settings for four aspects and the remaining aspects are then specified automatically.

Prior to setting up the component model, the product object types for the packaging hierarchy must be configured. This is done by creating a pallet object type, a case object type, and a pack, or each object type but these can be named whatever the user requires. You will notice in the directions below that in addition to selecting the product object types, you are asked to select a GTIN attribute, an Information Provider GLN attribute, and a child count attribute. The GDSN Component Model will provide these attributes for selection but in order to do so, follow steps one and two and select **'OK'**. At this point the component model will create the attributes and you can start at step one below and the attributes will be available for selection.

Note: If you choose to configure the GTIN and IPGLN attributes manually, there are pre-configured GTIN and IPGLN attributes that will validate the format of the GTIN to determine if it is correct. This will prevent the entry of a GTIN (UPC, EAN, etc.) or GLN from being sent to the data pool with an incorrect check digit or length of value. Also, these attributes must be description attributes for the component model to work. For information about these attributes, see 'Validation Rules' section of the 'Attributes' topic.

1. In **System Setup**, expand **Component Models**.
2. Right-click **GDSN Model**, and then choose **Easy setup of GDSN Component Model**.


System Setup

Event Queues
Component Models
- Auto Classification Model
- Country Aliases
- External Stored Assets Model
- GDSN Receiver model
- GDSN model
- Google Sho
- Matching
- Matching -
- Packaging
- Product Variant Model
- Publishing - Assetpush Queue
- Publishing - Flatplan Workflow
- Publishing - Freeze
- Publishing - Milestone Settings
- Publishing - Node with Pickup
- Publishing - UI Filtering
- Reference Data Address Street
- Reference Data Address Without Stree

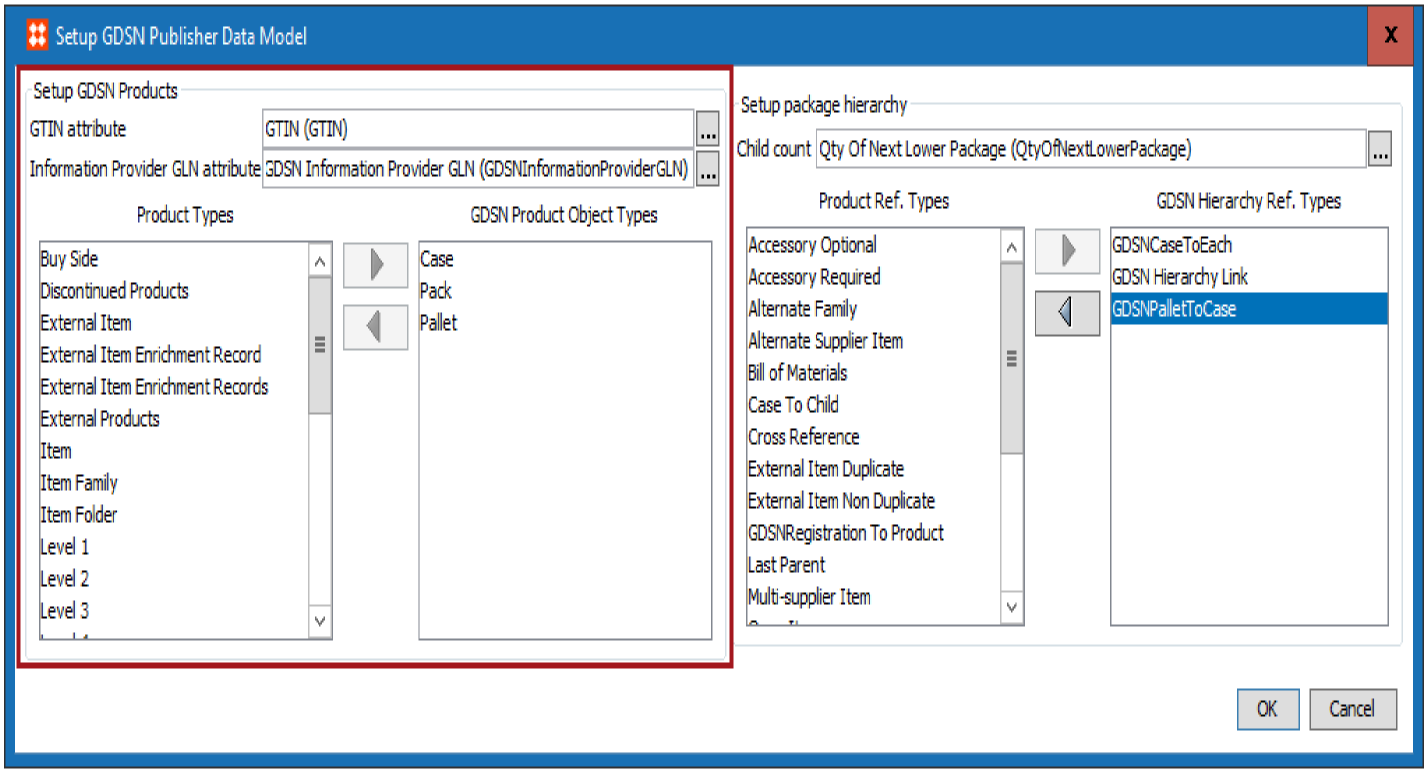
Easy setup of GDSN Component Model
Easy Setup GDSN Component Model
Convert unit mappings from old GDSN Publisher solution

No Title

Component Model Configuration

Name	Value	Description
CIC	GDSN CIC	Object type containing metadata for CIC occurrences on a GDSN publication
Datapool object type	GDSNDatapoolRoot	Object type containing Datapool configuration
	N Format	The format used for a specific datapool
	k	List of Product object types that can be published to GDSN
GDSN Products	Pallet	
	Case	
Recipient object type	GDSN Recipient	Object type containing Recipient configuration
Registration object type	GDSN Registrations	Object type containing metadata for GDSN registration of a product
Target Market object type	GDSN Target Market	Object type containing Target Market configuration
XSD Asset	GDSN XSD	The asset holding the XSD definition

3. In the **Setup GDSN Products** area, specify the following:
   - **GTIN attribute**: Select the attribute that you want to hold the Global Trade Item Number. Click the ellipsis button **(…)** to search or browse for the relevant attribute.
   - **Information Provider GLN attribute**: If the **Easy setup of GDSN Component Model** was used to create the IPGLN, this attribute can be found **GDSN System Attributes** attribute group.
   - **GDSN Product Object Types**: Select the product object types that can be published to the GDSN network.


Setup GDSN Publisher Data Model

Setup GDSN Products

GTIN attribute: GTIN (GTIN)

Information Provider GLN attribute: GDSN Information Provider GLN (GDSNInformationProviderGLN)

Product Types: Buy Side, Discontinued Products, External Item, External Item Enrichment Record, External Item Enrichment Records, External Products, Item, Item Family, Item Folder, Level 1, Level 2, Level 3

GDSN Product Object Types: Case, Pack, Pallet

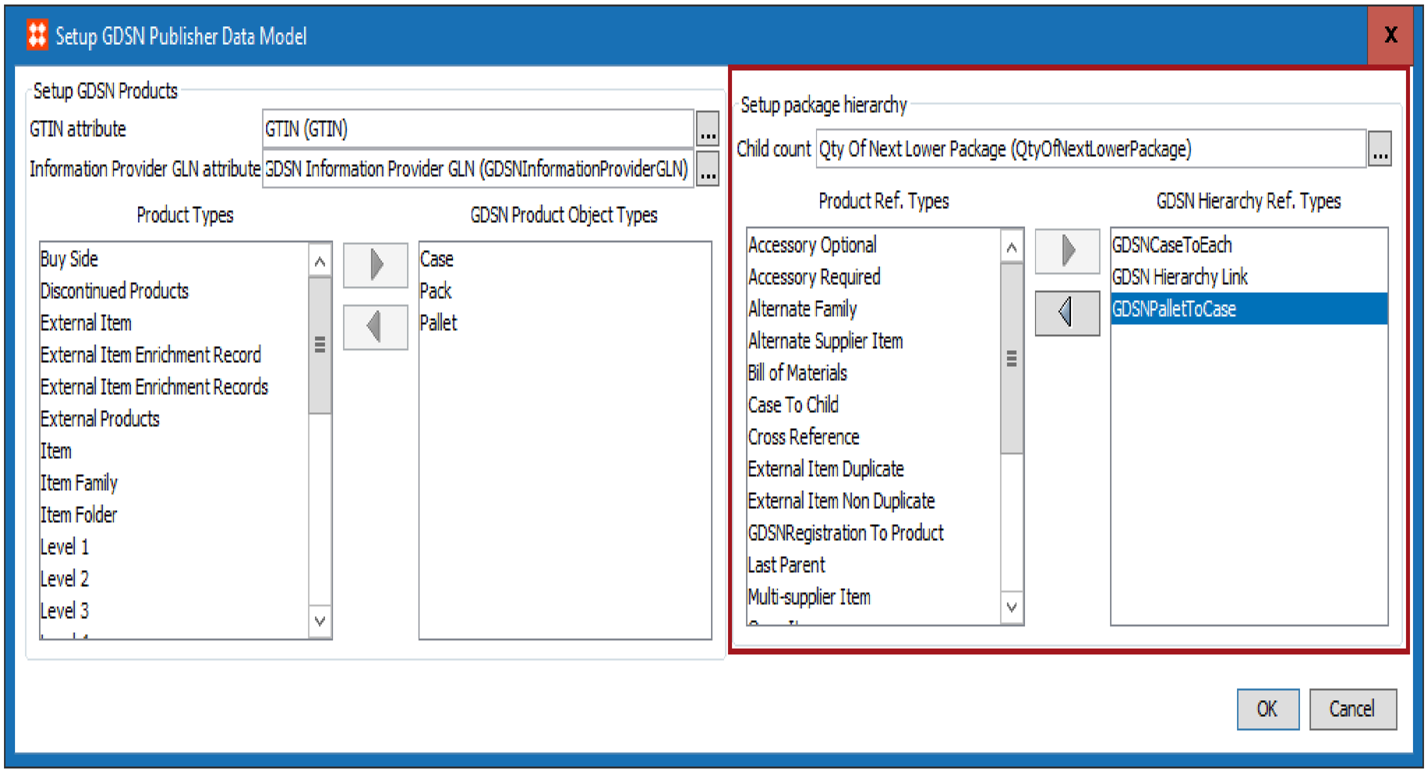
Setup package hierarchy

Child count: Qty Of Next Lower Package (QtyOfNextLowerPackage)

Product Ref. Types: Accessory Optional, Accessory Required, Alternate Family, Alternate Supplier Item, Bill of Materials, Case To Child, Cross Reference, External Item Duplicate, External Item Non Duplicate, GDSNRegistration To Product, Last Parent, Multi-supplier Item

GDSN Hierarchy Ref. Types: GDSNCaseToEach, GDSN Hierarchy Link, GDSNPalletToCase

4. In the **Setup Package Hierarchy** area, specify the following:
   - **Child Count**: Select the attribute that you want to hold the quantity of the next lower level trade item. This attribute applies to the package hierarchy link from a GDSN product to a child product in the package hierarchy. If the **Easy setup of GDSN Component Model** was used to create this attribute, the attribute can be found in the **Metadata** attribute group.
   - **GDSN Hierarchy Ref. types**: Select the product reference types to be used when the GDSN package hierarchy is created.


Setup GDSN Publisher Data Model

Setup GDSN Products

GTIN attribute: GTIN (GTIN)

Information Provider GLN attribute: GDSN Information Provider GLN (GDSNInformationProviderGLN)

Product Types: Buy Side, Discontinued Products, External Item, External Item Enrichment Record, External Item Enrichment Records, External Products, Item, Item Family, Item Folder, Level 1, Level 2, Level 3

GDSN Product Object Types: Case, Pack, Pallet

Setup package hierarchy

Child count: Qty Of Next Lower Package (QtyOfNextLowerPackage)

Product Ref. Types: Accessory Optional, Accessory Required, Alternate Family, Alternate Supplier Item, Bill of Materials, Case To Child, Cross Reference, External Item Duplicate, External Item Non Duplicate, GDSNRegistration To Product, Last Parent, Multi-supplier Item

GDSN Hierarchy Ref. Types: GDSNCaseToEach, GDSN Hierarchy Link, GDSNPalletToCase

The configuration of GDSN Product Object Types is mandatory, whereas the manual configuration of the other aspects described in the previous steps are optional. If you do not specify the settings manually, they are created automatically. If you choose an existing GTIN attribute, it is automatically made valid for all the selected GDSN product object types. Likewise, the package hierarchy child count is automatically made valid for all the package hierarchy link types.

The GDSN product object types and the package hierarchy links must fit together. A package hierarchy link must have a source or target type that is of one of the selected GDSN product object types. If the object types and links do not fit together, a message is displayed saying that the selected source type is not a GDSN product, and the OK button is disabled.

If you do not select a package hierarchy, a single package hierarchy link is created automatically. This reference type will then have all the selected GDSN product object types as valid sources and targets. However, it is a good practice to have different reference types for each level in the packaging hierarchy to avoid cyclical packaging hierarchies. Example: pallet to case and case to package would be different types.

Note: At this point, there are additional configurations that need to be made such as creating the data pools, creating Target Markets, recipients, and formats however it is highly suggested that you complete the '**Easy setup of GDSN provider data pool**' first.

GDSN Provider Component Model Elements

The following table describes the elements of the Provider component model.

Name	Type	Description	How it is used
CIC	Object type	Object type that contains meta data for CIC occurrences on a GDSN publication.	When a recipient sends a CIC message, a new entity object of this type is created.
Datapool object type	Object type	Object type that contains data pool configuration.	Objects of this type are created automatically. Specify an ID and name.
Format	Object type	The format used for a specific data pool.	Objects of this type are created automatically. ID and name are based on the data pool. Format objects are special because they include tabs for inbound and outbound configurations.
GDSN Products	Object type	List of product object types that can be published to GDSN	These are the products and packaging objects. Must be pre-configured and then selected in the **Setup GDSN Publisher Data Model** dialog.
Recipient object type	Object types	Object type that contains recipient configuration	This type is created automatically. You must create the entity objects when the component model has been setup. The recipient objects must be created in an entity folder as a child of the data pool entity object.
Registration object type	Object type	Object type that contains meta data for GDSN registration of a product	This type is created automatically. Objects of this type are created by the GDSN Provider component when a product is registered in GDSN.
Target Market object type	Object type	Object type that contains target market configuration	This type is created automatically. You create entity objects of this type to represent the target markets that a product can be registered to. Create the target markets object in an entity folder as a child of the data pool entity object.
XSD Asset	Object type	The asset that holds the XML schema definitions	This object type is used to store the schema that the generated XML is validated against before the XML is sent to the data pool.

Name	Type	Description	How it is used
As2In Folder	Attribute	Description attribute that contains the folder path for receiving data from data pool	The attribute points to a hotfolder where files from the GDSN network are picked up by the inbound integration endpoint.
AS2Out Folder	Attribute	Description attribute that contains the folder path for sending data to a data pool.	This attribute points to a hotfolder where XML files for the GDSN network are located. It is assumed that the files are picked up by the GDSN network using an AS2 server.
CIC status	Attribute	Description attribute that contains the CIC status on the reference that relates the registration to the recipient.	Contains the CIC status for a publication of a product to a given recipient.
Datapool GLN	Attribute	Description attribute that contains the GLN of the data pool.	The data pool itself also has a GLN. So, for example, the 1WorldSync data pool has a GLN. The data pool's GLN is part of the protocol within GDSN.
Datapool Username	Attribute	Description attribute that contains the user name of the provider to the data pool.	The user name that is used when sending and receiving data from the data pool. This is not a STEP user name.
Format Inbound Configuration	Attribute	The inbound integration configuration for this format.	This attribute contains the inbound configuration of the format. The inbound configuration is viewable in the **Inbound** tab on the format editor.
Format Outbound Configuration	Attribute	The outbound integration configuration for this format.	The attribute contains the outbound configuration of the format. The outbound configuration is viewable in the Outbound tab on the format editor.
GTIN	Attribute	Specification attribute that contains the GTIN	This attribute is the GTIN of the product or the packaging objects that are part of a package hierarchy.

Name	Type	Description	How it is used
My GLN	Attribute	Description attribute that contains the GLN for a registration.	Attribute that identifies a company's registering and publishing items (the provider GLN).
Package Hierarchy Reference Status	Attribute	Description attribute that contains the status of a packaging object's registration.	Meta data on the reference from a packaging object to the lower level item that shows the status of the registration.
Pending command	Attribute	Description attribute containing the pending command for the registration if another command needs to be run before the current command	This attribute must be valid for the registration object type. It is used to store the pending command used by the Business Actions 'Set PendingCommand' and 'Execute Pending Command'
Publish status	Attribute	Description attribute that contains the status of a registration to recipient reference.	Metadata on the reference from a registration to a recipient. The attribute contains the status of the publication of a product or a packaging object.
Quantity of next lower level package	Attribute	Description attribute that contains the quantity of the next lower level packages in this package.	This attribute specifies the number of lower level objects a given packaging object can contain. The lower level packaging object is given as the source of the packaging hierarchy reference.
Recipient GLN	Attribute	Description attribute that contains the GLN of a recipient	The GLN of the recipient of a publication.
Registration Status	Attribute	Description attribute that contains the status of a registration.	Meta-data on the Registration object that contains the status of the registration to a given target market.
Target Market Name	Attribute	Description attribute that contains the target market.	The name of the target market as defined by the data pool. If this attribute is not set, or if it uses name other than the one defined by the data pool, the registration to the target market will fail.

Name	Type	Description	How it is used
CIC to Recipient reference	Reference type	Reference type that relates the CIC to the recipient.	The reference is created when a CIC message is received from a recipient, and links the recipient to the CIC message.
CIC to Registration reference	Reference type	Reference type that relates the CIC to the registration.	The reference is created when a CIC message is received from a recipient, and links the CIC message to a registration and thereby to a product or packaging object and a target market.
Datapool to format reference	Reference type	Reference type that relates the data pool to its format definition.	The system will use this reference to find the format node that holds the data pool configuration.
Format to asset reference	Reference type	Reference type that relates the format to the asset that contains the XML schema asset.	The XML schema is used to validate the generated XML before the XML is sent to the data pool.
Package Hierarchy References	Reference type	The reference type that relates a packaging object to its child packaging object.	This reference type must be created manually before using the setup dialog. The references are used to build the packaging hierarchy.
Registration to Product Reference	Reference type	The reference type that relates a registration object to a packaging object.	The reference is created when a product or packaging object is registered in the GDSN network.
Registration to Recipient Reference	Reference type	The reference type that relates a registration object to a recipient.	The reference type is created when a product or packaging object is published to the GDSN network.
Registration to Target Market Reference	Reference type	The reference type that relates a registration object to a target market.	The reference is created when a product or a packaging object is registered in the GDSN network and shows the target market for a registration.
Target Market to Context Reference	Reference type	The reference type that relates a target market to a context.	The referenced context is used when data is exported for a GDSN registration. Values for attributes and so on are defined by the language settings of the context. Must be created when a new target market is created.

Setting Up GDSN Provider Data Pools

The **Setup GDSN Publisher Data Model** component model is a powerful tool that will provide the following:

- Message routing information for the incoming and outgoing message folders
- GDSN Workflows
- GDSN Business Rules
- Outbound and Inbound integration endpoints

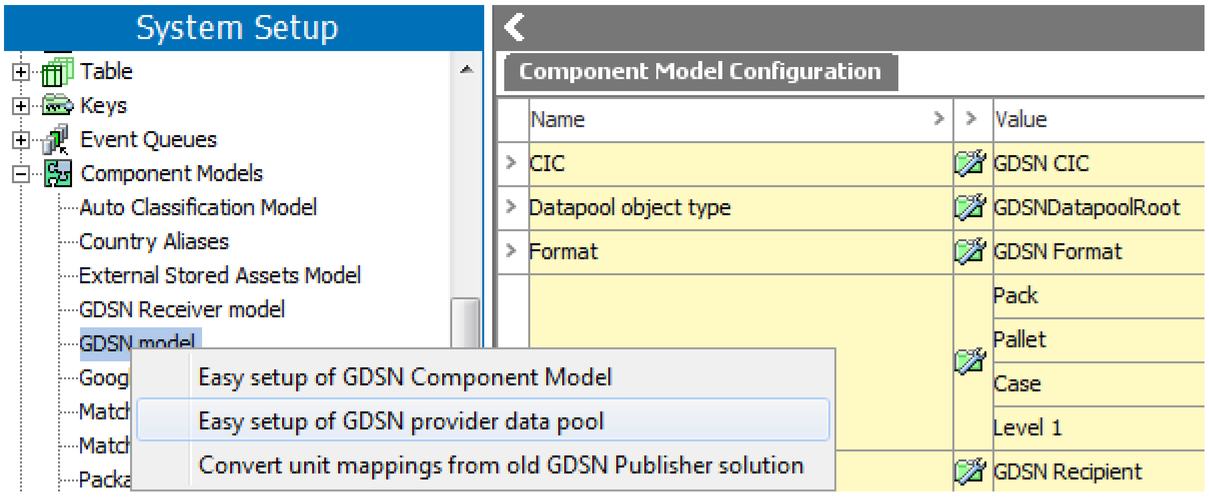
A STEP provider data pool handles the communication between the STEP system and the external data pool. The following requirements must be met for the STEP provider data pool to function correctly.

- An outbound integration endpoint to dispatch messages to a hot-folder.
- An inbound integration endpoint to pick up the incoming message from a hot-folder. Both endpoints will be configured with the data pool as receiver or delivery plug-in.
- The setup of an AS2 server is required to take the messages from the out-folder and transfer them to the external data pool and to deliver the external messages from the data pools in the in-folder. Setting up the AS2 server is beyond the scope of this documentation.

You can use the **Setup GDSN Publisher Data Model** dialog to set up the STEP data pool and the integration endpoints. The dialog contains information on how to communicate with the external data pool and is also used to group and store information related to the communication with the external data pool.

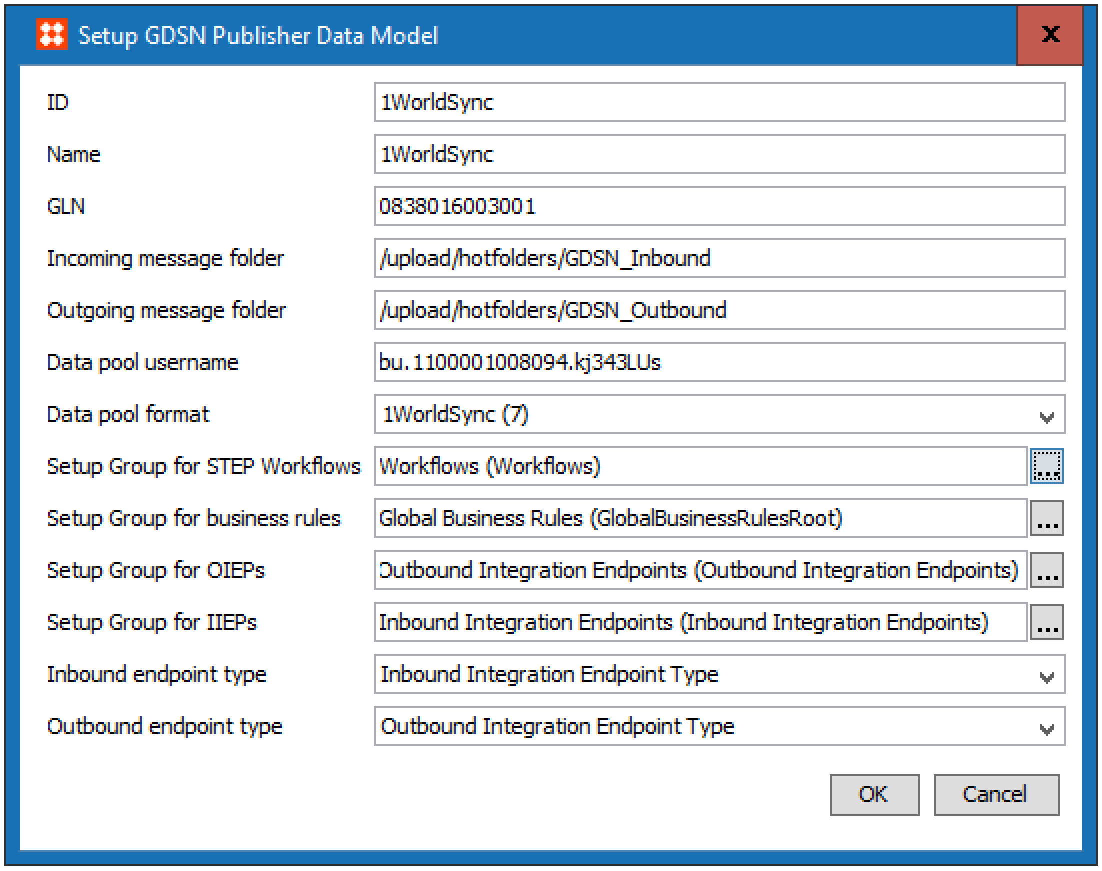
To Set Up The GDSN Data Pool

1. In **System Setup**, expand **Component Models**.
2. Right-click **GDSN Model**, and then choose **Easy setup of GDSN Provider Data Pool**.


System Setup
Table
Keys
Event Queues
Component Models
Auto Classification Model
Country Aliases
External Stored Assets Model
GDSN Receiver model
GDSN model
Goog
Matc
Matc
Packa
Easy setup of GDSN Component Model
Easy setup of GDSN provider data pool
Convert unit mappings from old GDSN Publisher solution
Component Model Configuration

Name	Value
CIC	GDSN CIC
Datapool object type	GDSNDatapoolRoot
Format	GDSN Format
	Pack
	Pallet
	Case
	Level 1
	GDSN Recipient

3. In the **Setup GDSN Publisher Data Model** dialog, enter information for your desired data pool:



ID	1WorldSync
Name	1WorldSync
GLN	0838016003001
Incoming message folder	/upload/hotfolders/GDSN_Inbound
Outgoing message folder	/upload/hotfolders/GDSN_Outbound
Data pool username	bu.1100001008094.kj343LUs
Data pool format	1WorldSync (7)
Setup Group for STEP Workflows	Workflows (Workflows)
Setup Group for business rules	Global Business Rules (GlobalBusinessRulesRoot)
Setup Group for OIEPs	Outbound Integration Endpoints (Outbound Integration Endpoints)
Setup Group for IIEPs	Inbound Integration Endpoints (Inbound Integration Endpoints)
Inbound endpoint type	Inbound Integration Endpoint Type
Outbound endpoint type	Outbound Integration Endpoint Type

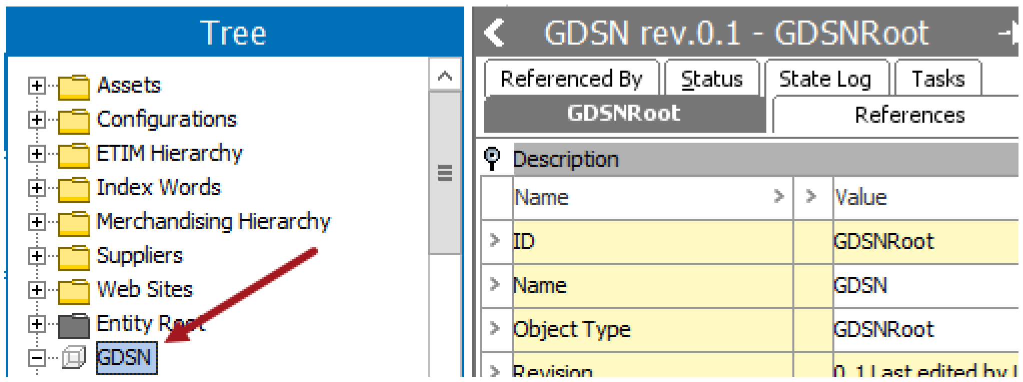
Field	Description
ID	The STEP ID of the STEP data pool object. This ID is also used as ID prefix for other STEP objects created under the data pool such as IIEPs, OIEPs, and grouping entities.
Name	The STEP name of the data pool
GLN	The Global Localization Number (GLN) of the data pool. For example, 838016003003 for 1WorldSync.
Incoming message folder	The folder messages from the external data pool are delivered to.
Outgoing message	The folder messages to be transferred to the external data pool are taken from.

Field	Description
folder	
Data pool user name	The user name / password to be included in the messages to the external data pool as part of the identification and authorization of the messages.
Data pool format	The format helps determine which templates to use for the messages sent to the external data pool. It can also contain information about the structure of the messages that the data pool can receive. This way, messages can be validated before being dispatched to the external data pool.
Setup Group for STEP Workflows	Depending on the selected data pool format, a STEP workflow that handles CICs (Catalog Item Confirmation) is created. The workflow is placed in the selected setup group. This field is automatically populated with any setup group that allows the creation of STEP workflows. If no setup group is found, a warning message is displayed before the setup dialog is displayed.
Setup Group for business rules	Depending on the selected data pool format, various business actions are created. (The business rules are used, for example, to change the status of an issue when an acknowledge message for a registration message is received). These business rules are placed in the selected setup group. This field is automatically populated with a random setup group that allows the creation of business rules. If no setup group is found, a warning message is displayed before the setup dialog is displayed.
Setup Group for the OIEPs	Setup group for the outbound integration endpoint used to generate messages to the external data pool. This field is automatically populated with a random setup group that allows the creation of outbound integration endpoints. If no setup group is found, a warning message is displayed before the setup dialog is displayed.
Setup Group for the IIEPs	Setup group for the inbound integration endpoint is used to pick up and process messages from the external data pool. This field is automatically populated with a random setup group that allows the creation of inbound integration endpoints. If no setup group is found, a warning message is displayed before the setup dialog is displayed.
Inbound endpoint type	Select the type of inbound integration endpoint (IIEP) you want to create. This field is automatically populated with a random outbound endpoint type that is valid for the inbound integration endpoints type.
Outbound endpoint type	Select the type of outbound integration endpoint (OIEP) you want to create. This field is automatically populated with a random outbound endpoint type that is valid for the outbound integration endpoints type.

After setting up the data pool object, configure the target markets and recipients, and complete the format of the outbound message template. For more information, see the **Data Pool Configuration** documentation.

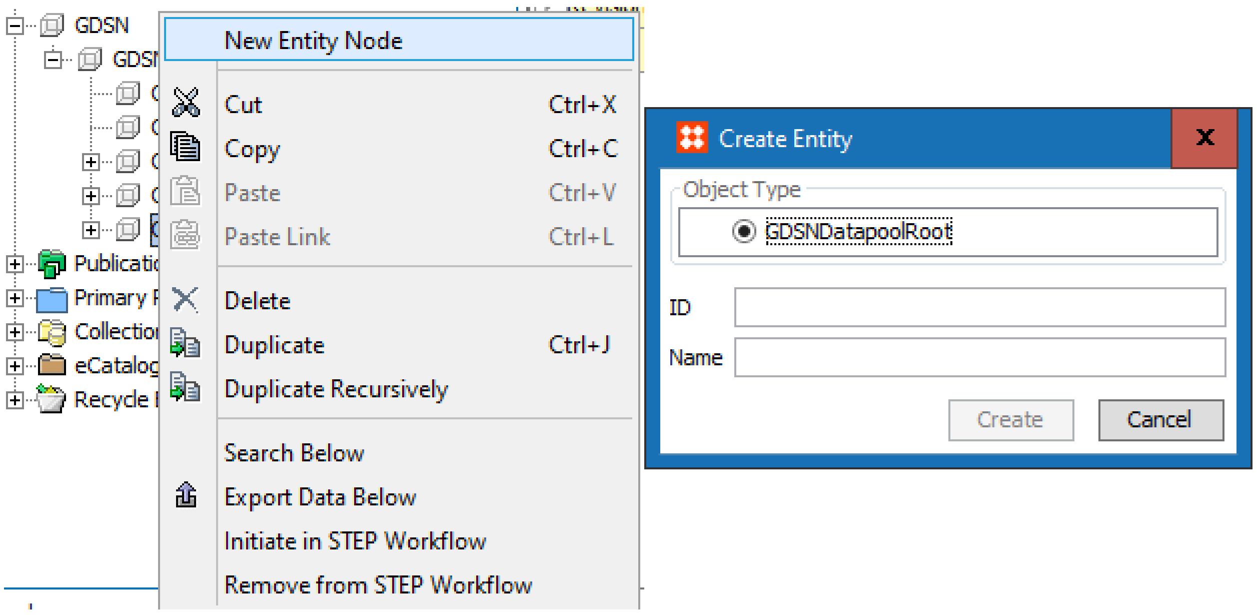
Data Pool Configuration

After the **Easy setup of GDSN Component Model** and the **Easy setup of GDSN provider data pool** component models have been configured, the data pool object needs to be configured for the data pool(s). This will require creating target markets, formatting the XML mappings, and creating the recipients that will be receiving the data. This begins from the GDSN root node in the **Tree** as shown in the image below.



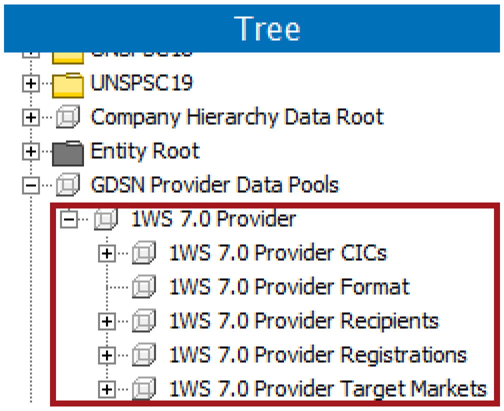
Name	Value
ID	GDSNRoot
Name	GDSN
Object Type	GDSNRoot
Revision	0.1 Last edited by

From this root node, one or many data pools can be created by right-clicking on the GDSN root node and selecting **New Entity Node** The **Create Entity** dialog box will appear for the 'GDSNDatapoolRoot' object type and an ID and name for the data pool are entered.



A data pool has the following five grouping entities, each of which need to be created. After the entities have been created, your data pool root should look similar to the image below.

- **CIC Group:** Contains information about the CICs (Catalog Information Confirmations) the recipients have sent back.
- **Format**: Contains the configuration of the format of the outbound messages.
- **Recipients Group:** Contains information about all the recipients used by this data pool.
- **Registrations Group:** Contains information about which products are registered with the data pool - and for which target markets.
- **Target Markets Group:** Contains information about all the target markets used by this data pool.


Tree
UNSPSC19
Company Hierarchy Data Root
Entity Root
GDSN Provider Data Pools
1WS 7.0 Provider
1WS 7.0 Provider CICs
1WS 7.0 Provider Format
1WS 7.0 Provider Recipients
1WS 7.0 Provider Registrations
1WS 7.0 Provider Target Markets

Of these five entities, three need to be set up. Once created, the CICs object type and the Datapool Registrations object type are ready for use. These are the root nodes and the CICs and Registrations are created automatically as messages are sent back and forth. However, there is still some come configuration required for the GDSN Format, GDSN Recipients, and GDSN Target Market object types.

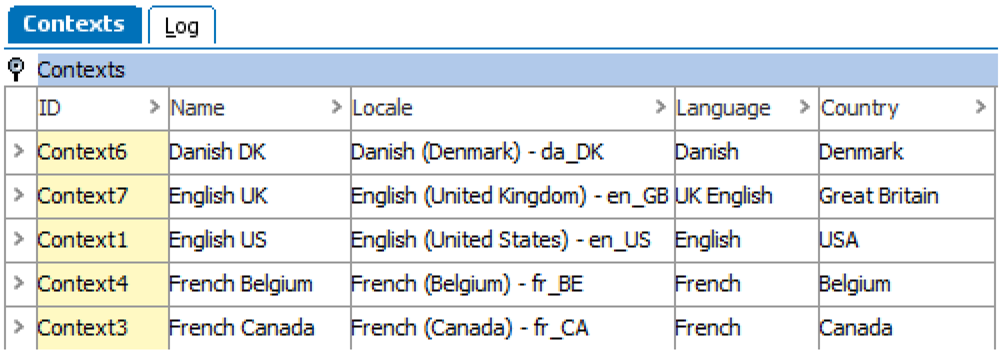
Note: For information related to setting up target markets, please see **Creating GDSN Target Markets**.

Note: For information related to setting up recipients, please see **Creating the GDSN Recipients**.

Note: For Information related to formatting the outbound GDSN messages, please see **Configuring the Outbound Message Format**.

Creating GDSN Provider Target Markets

This section explains the steps to creating the GDSN Provider Target Markets. However it is critical to set up the dimensions and contexts correctly prior to creating the Target Markets as one of the steps involves selecting the GDSN Context. An example of how the contexts can be set up is provided below but for more information regarding dimensions and contexts, see the **Contexts** section of the **System Setup** / **Super User Guide** documentation.

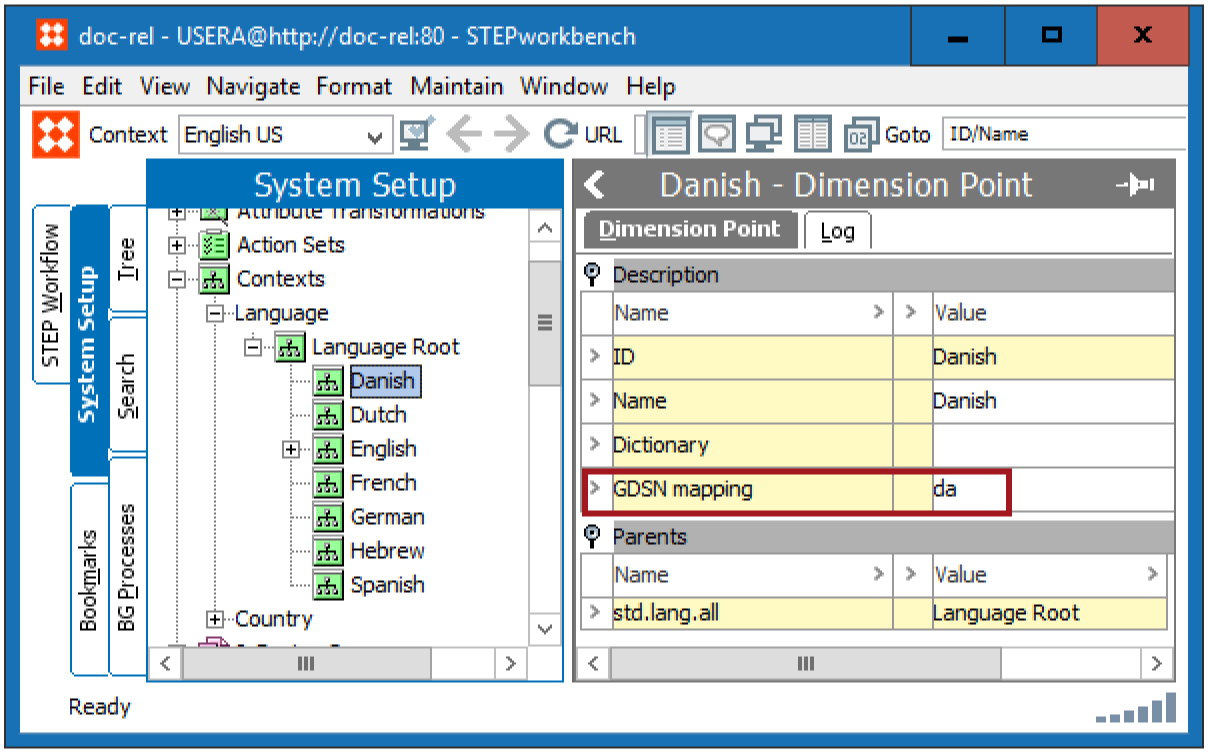


Contexts | Log

Contexts

ID	Name	Locale	Language	Country
Context6	Danish DK	Danish (Denmark) - da_DK	Danish	Denmark
Context7	English UK	English (United Kingdom) - en_GB	UK English	Great Britain
Context1	English US	English (United States) - en_US	English	USA
Context4	French Belgium	French (Belgium) - fr_BE	French	Belgium
Context3	French Canada	French (Canada) - fr_CA	French	Canada

When the GDSN Component Model was set up, it added an attribute to the language dimension point named **GDSN mapping**. This is where the ISO language code should be populated and will work in concert with the Target Market objects to map the language code in the outbound message.



Danish - Dimension Point

Dimension Point | Log

Description

Name	Value
ID	Danish
Name	Danish
Dictionary	
GDSN mapping	da

Parents

Name	Value
std.lang.all	Language Root

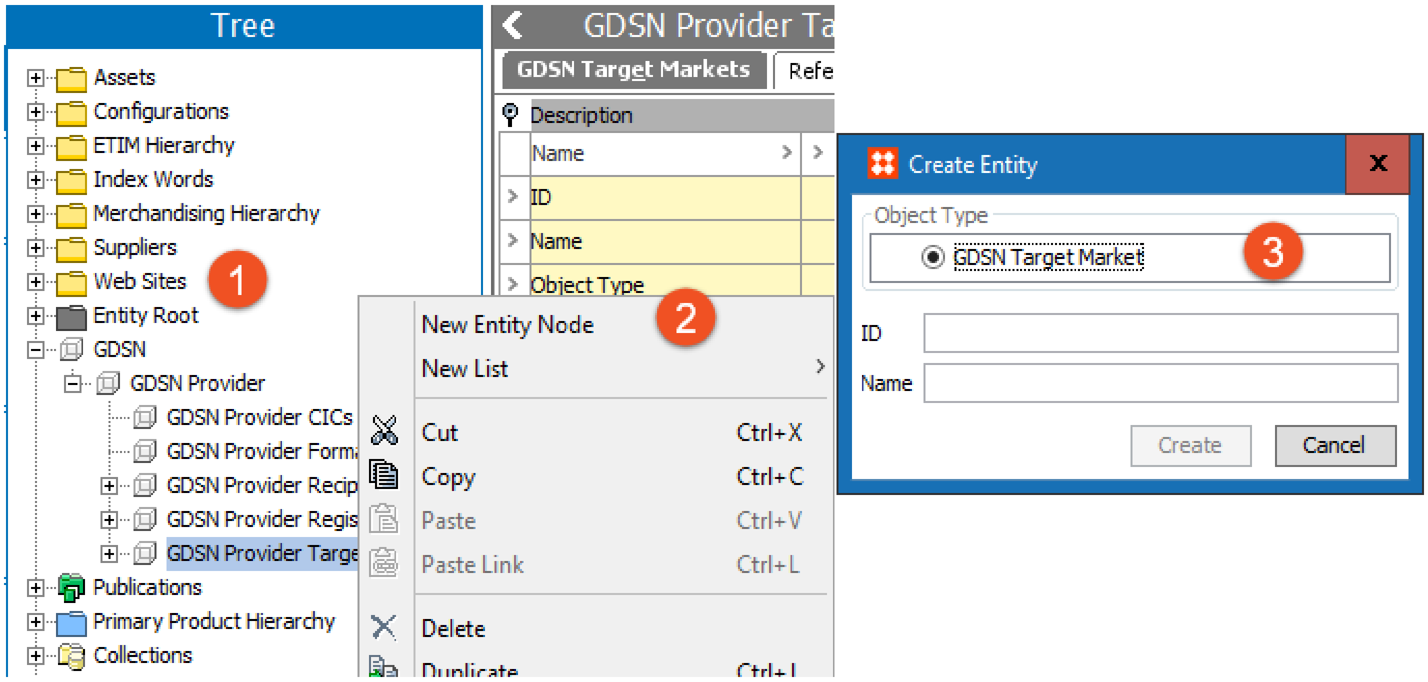
Create GDSN Target Markets

1. In the **Tree**, expand the relevant data pool, and then select the target market entity.

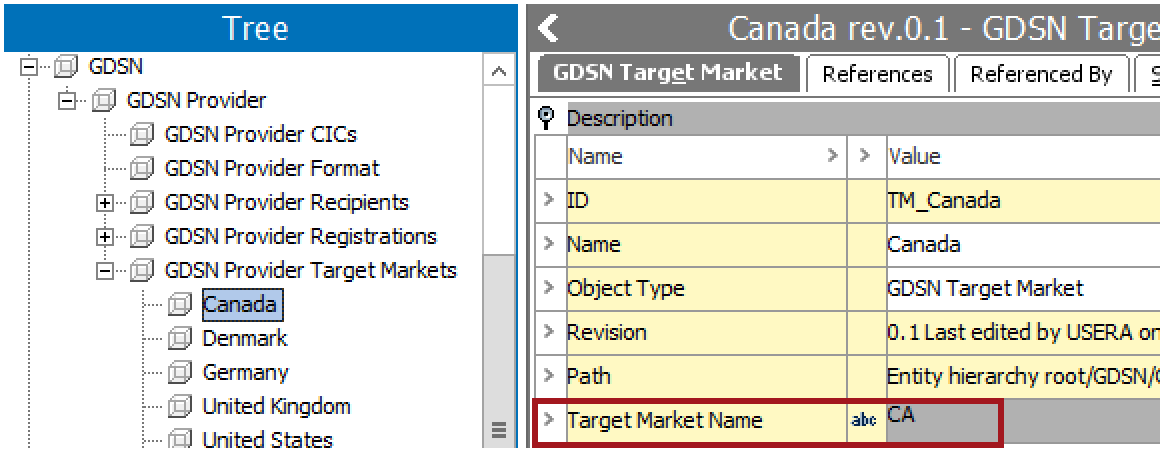
   You must create all target markets for the data pool in this folder, and the object type must be the target market type configured in the component model. This is the only object type that is valid below the target market folder.

2. Right-click the **Target Markets** entity, and then select **New Entity Note**.
3. Enter an **ID** and a **Name**, and then click **Create**. The name should be what you want your users to see when they select a Target Market in the Web UI.

Important: The ID value is the value that will get mapped and sent to the GDSN



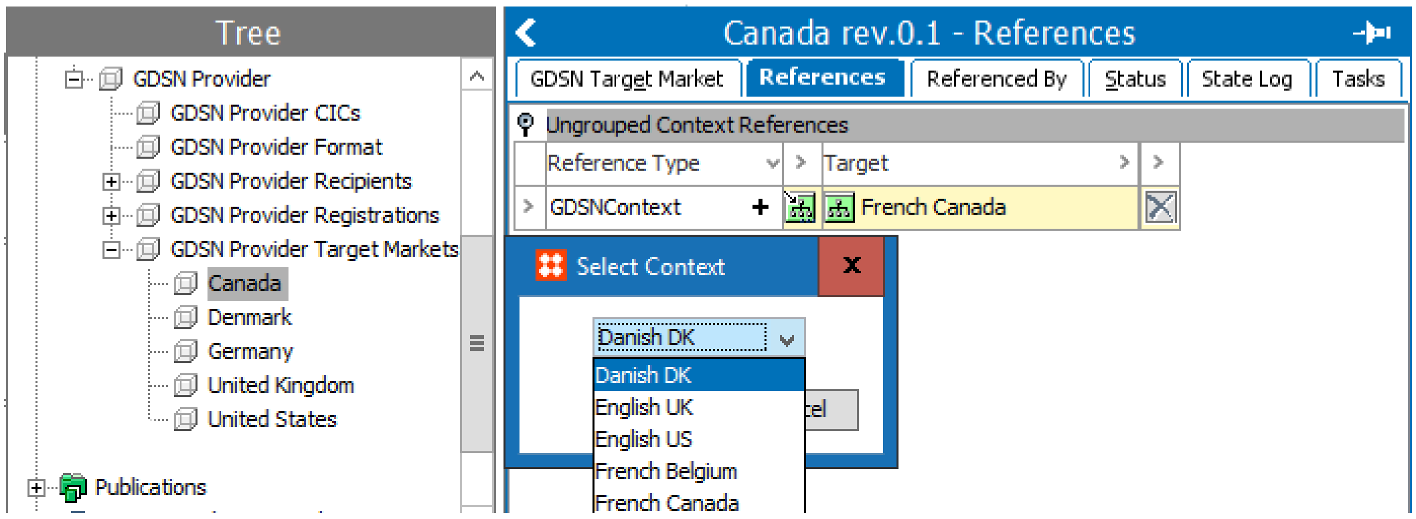
4. In the **Tree** select the entity you just created, populate the GDSN attribute **Target Market Name** with the ISO code meeting your data pool's requirements for the Target Market. This example shows a 1WorldSync setup and is in upper case letters.



Name		Value
ID		TM_Canada
Name		Canada
Object Type		GDSN Target Market
Revision		0.1 Last edited by USERA on
Path		Entity hierarchy root/GDSN/
Target Market Name	abc	CA

5. On the **Reference** tab in the **GDSNContext** field, link to the contexts to be used when messages are sent out for this target market. The Select Context dialog box will appear when the plus sign is selected.

Note: To publish to multiple languages within a single target market, more than one context reference can be made, but all contexts for a single target market must be defined to use the same Country Dimension. Example: A Context for Canada with English language and a Context for Canada with French language.



Reference Type	Target
GDSNContext	French Canada

Continue this process until all of your Target Markets are set up.

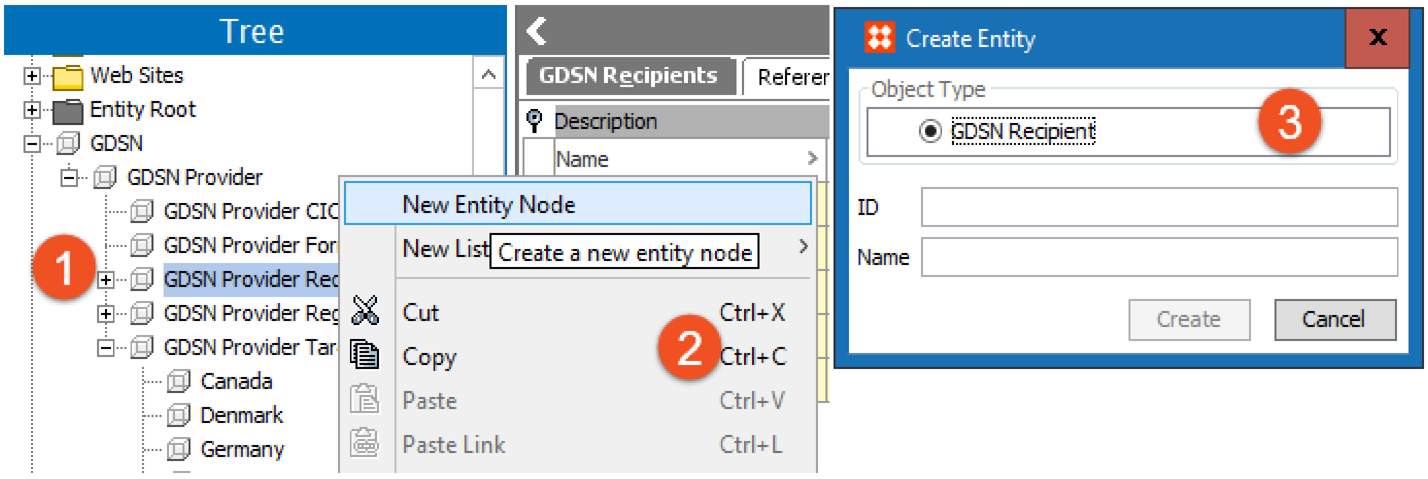
Creating the GDSN Recipients

Use the following steps to create GDSN Recipients.

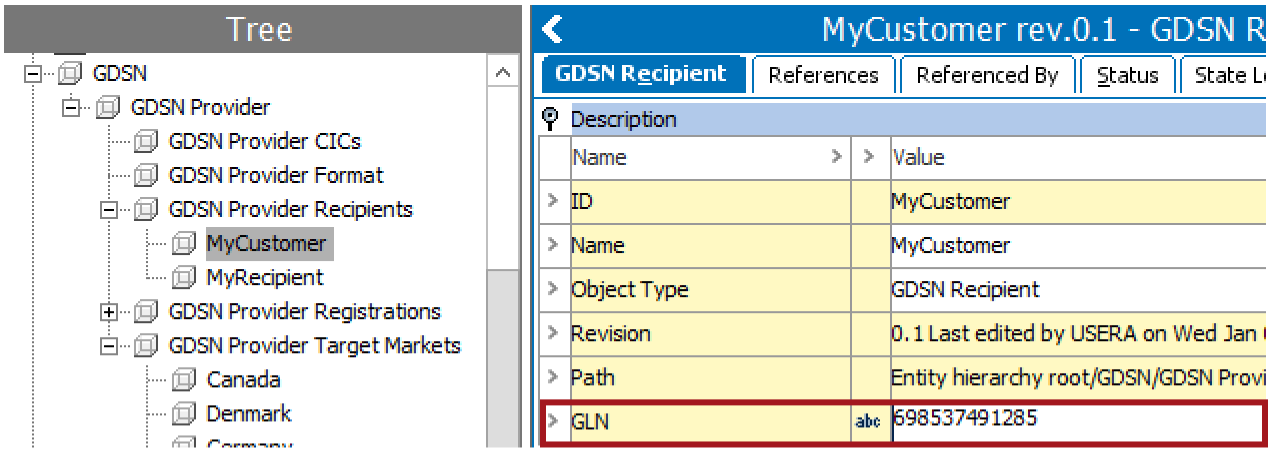
1. In the **Tree**, expand the relevant data pool, and select the recipients entity.

   You must create all recipients for the data pool in this folder, and the object type must be the recipient type configured in the component model. This is the only object type that is valid below the recipient folder.

2. Right-click the **Recipients** entity, and then select **New Entity Note**.
3. Enter an **ID** and a **Name**, and then click **Create**.



4. In the **Tree** select the entity you just created and enter your recipient's GLN in the **GLN** attribute field.



Name	Value
ID	MyCustomer
Name	MyCustomer
Object Type	GDSN Recipient
Revision	0.1 Last edited by USERA on Wed Jan
Path	Entity hierarchy root/GDSN/GDSN Provi
GLN	698537491285

Continue this process until all of your Recipients are set up.

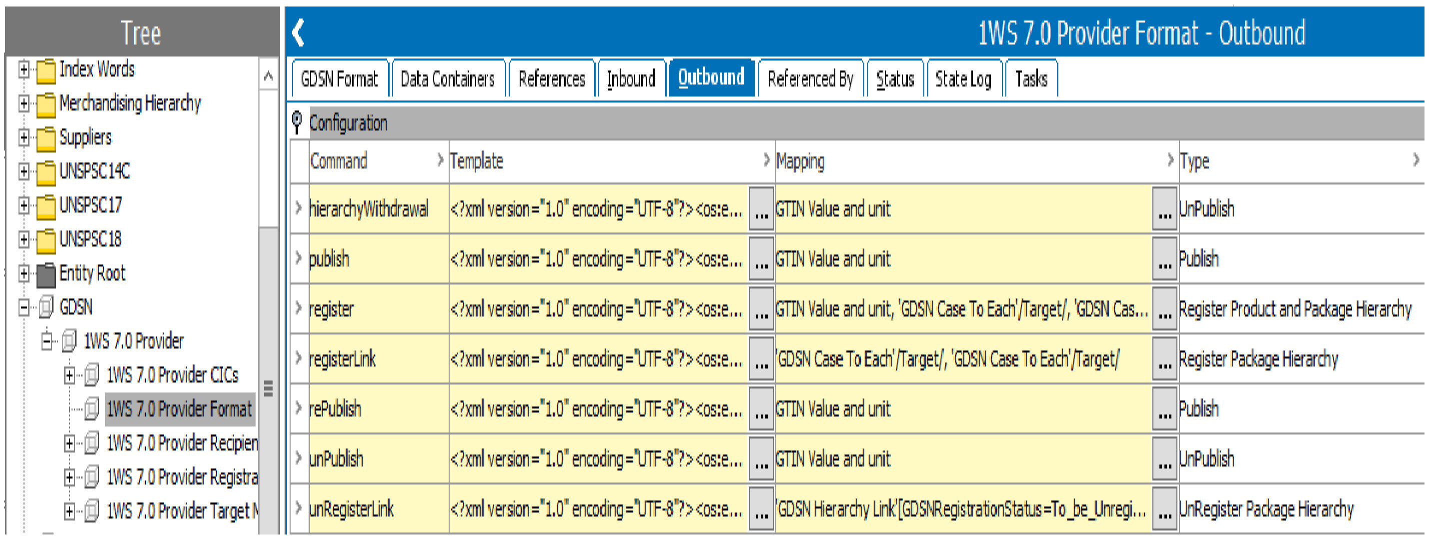
Configuring the Outbound Message Format

Before you can register products in the data pool, you must configure the format templates. The outbound format configuration defines the format that is used when messages are generated for a GDSN data pool. The configuration is a list of the message types that can be generated. Each message contains four components:

- A command
- A generic XML template
- Export mappings
- The message type

To Configure Format Templates

1. In the **Tree**, expand the relevant data pool, and then click the **Format** entity.
2. Click the **Outbound** tab. The tab has the following columns:



Command	Template	Mapping	Type
hierarchyWithdrawal	<?xml version="1.0" encoding="UTF-8"?><os:e...	GTIN Value and unit	UnPublish
publish	<?xml version="1.0" encoding="UTF-8"?><os:e...	GTIN Value and unit	Publish
register	<?xml version="1.0" encoding="UTF-8"?><os:e...	GTIN Value and unit, 'GDSN Case To Each'/Target/, 'GDSN Cas...	Register Product and Package Hierarchy
registerLink	<?xml version="1.0" encoding="UTF-8"?><os:e...	'GDSN Case To Each'/Target/, 'GDSN Case To Each'/Target/	Register Package Hierarchy
rePublish	<?xml version="1.0" encoding="UTF-8"?><os:e...	GTIN Value and unit	Publish
unPublish	<?xml version="1.0" encoding="UTF-8"?><os:e...	GTIN Value and unit	UnPublish
unRegisterLink	<?xml version="1.0" encoding="UTF-8"?><os:e...	'GDSN Hierarchy Link'[GDSNRegistrationStatus=To_be_Unregi...	UnRegister Package Hierarchy

- **Command**

  The command is a key that identifies the type of the message. The command key is stored in the configuration of the business actions that generate messages for GDSN, and it is used in the configuration of the Web UI action buttons. See the **GDSN Web UI Buttons** section of the **GDSN Provider** documentation.

- **Template**

  The generic XML follows the syntax of the generic XML export format (see **Generic XML Format** documentation). The GDSN Provider component defines some additional generic XML parameter tags:

  To edit the XML template, click the ellipsis button **(...)**. The **Create new template** dialog appears. The following parameters are available:

Parameter Tag	Description
<?Parameter MyGLN?>	The GLN the product is registered from. Configured in either the business action starting the endpoint or in the configuration of the register / publish buttons in the Web UI.
<?Parameter Username?>	The data pool user-name. Is configured directly on the data pool.
<?Parameter DataPoolGLN?>	The data pool GLN. Is configured directly on the data pool.
<?Parameter TargetMarket?>	The target market attribute value from the selected target market. Is configured on the target market entity.
<?Parameter RecipientGLN?>	The GLN of the selected recipient. Is configured on the recipient.
<?Parameter TimeStamp?>	A time stamp in the format: yyyy-MM-dd'T'HH:mm:ss
<?Parameter MessageID?>	A string generated from the message type name, a time stamp, and a random number, to be used as a message ID.
<?Parameter TargetMarket|{att_id}?>	The value of the Target Market attribute with ID att_id.
<?Parameter Recipient|{att_id}?>	The value of the Recipient attribute with ID att_id.

- **Mapping**

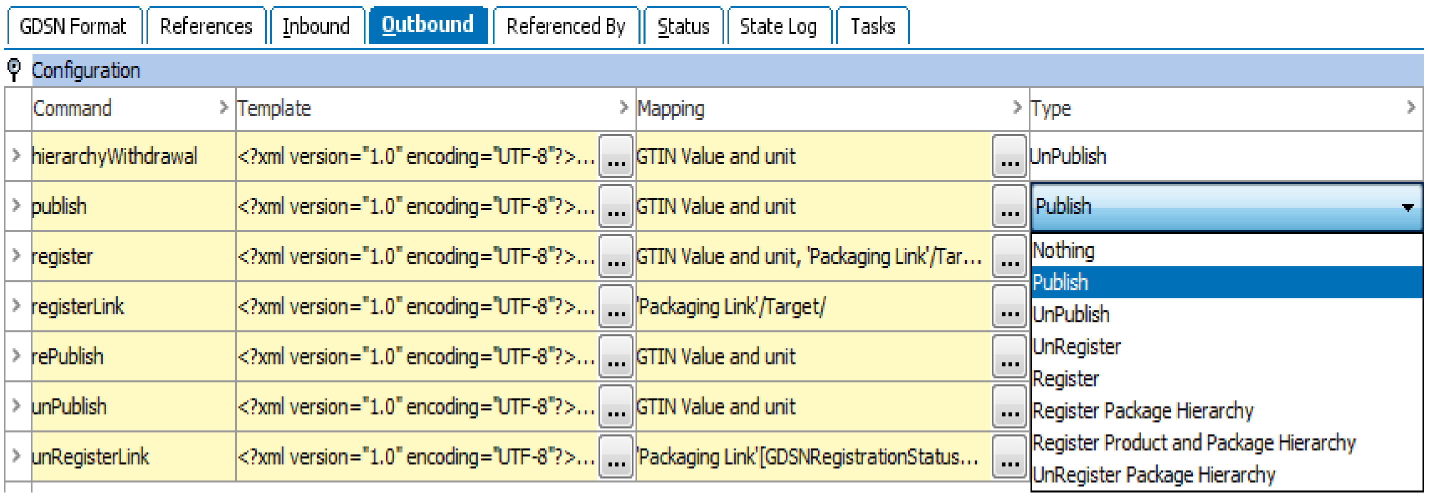
The export mappings define the mapping between the STEP data model and the targets defined by the generic XML template.

In the **Mapping** column, click the ellipsis button **(...)** , next to the relevant mapping.

For information about the Mappings dialog, see the **Generic XML Format** documentation.

- **Type**

The type of a message is used to update the status of the products sent to GDSN. If the message type is 'register', for example, you can update the status of the Registration to 'Registration Pending'. If 'nothing' is selected, the status is not changed.



Command	Template	Mapping	Type
hierarchyWithdrawal	<?xml version="1.0" encoding="UTF-8"?>...	GTIN Value and unit	UnPublish
publish	<?xml version="1.0" encoding="UTF-8"?>...	GTIN Value and unit	Publish
register	<?xml version="1.0" encoding="UTF-8"?>...	GTIN Value and unit, 'Packaging Link'/Tar...	
registerLink	<?xml version="1.0" encoding="UTF-8"?>...	'Packaging Link'/Target/	
rePublish	<?xml version="1.0" encoding="UTF-8"?>...	GTIN Value and unit	
unPublish	<?xml version="1.0" encoding="UTF-8"?>...	GTIN Value and unit	
unRegisterLink	<?xml version="1.0" encoding="UTF-8"?>...	'Packaging Link'[GDSNRegistrationStatus...	

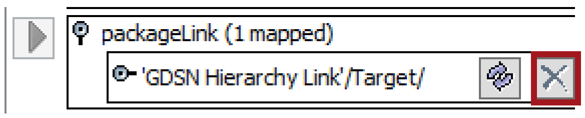
Nothing
Publish
UnPublish
UnRegister
Register
Register Package Hierarchy
Register Product and Package Hierarchy
UnRegister Package Hierarchy

The following message types are available:

Status	Description
Nothing	Used when no status updates are required for the outbound message
Publish	Sets the publish status to publish_pending
UnPublish	Sets the publish status for the product to unpublish_pending
UnRegister	Sets the registration status to Unregister_pending
Register	Sets the registration status to register pending
Register Package Hierarchy	Sets the registration status for all objects in the hierarchy to register_pending; it is assumed that the products are already registered
Register Product and Package Hierarchy	Sets the registration status on all products in the hierarchy to register_pending and the same for the objects.
Unregister Package Hierarchy	Sets the registration status of the hierarchy to Unregister_pending

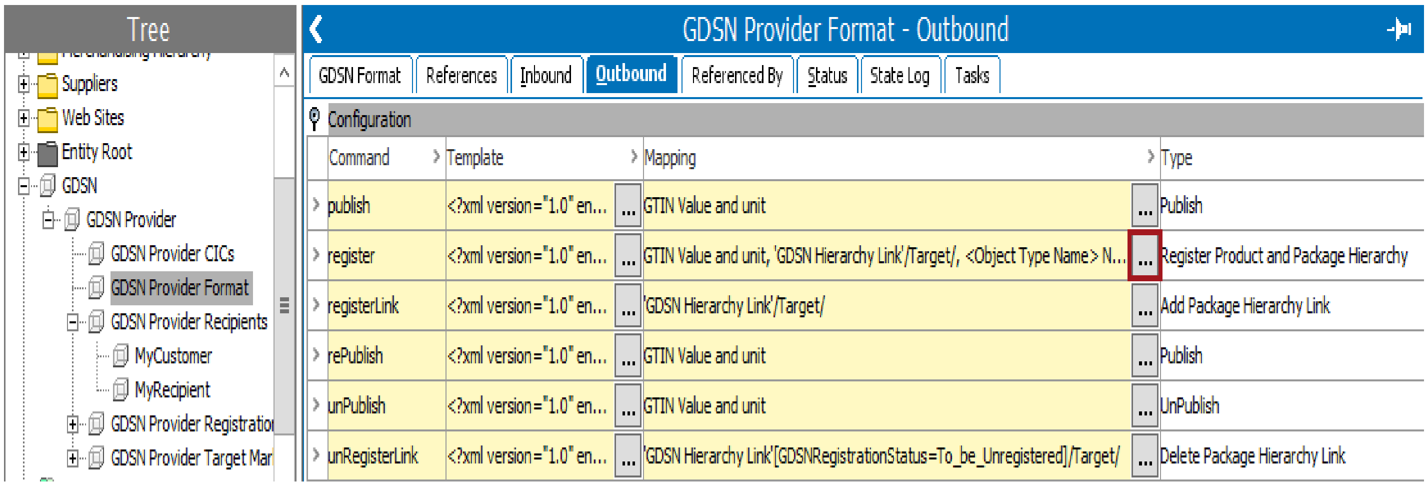
Configuring the GDSN Register Template

The register template represents the majority of the work involved in preparing the outbound message templates since it contains the most information about products. The STEP core solution will include a few basic elements of the register template. It will be project work to add each of the attributes to the template and to map them. Also, the register template may or may not include links between packaging levels and the quantities contained within them. The links section will be included in the out-of-box template and you will need to remove them if you manage them in dedicated messages as shown in the image below.



packageLink (1 mapped)
'GDSN Hierarchy Link'/Target/

All of the information in this section begins by selecting the selecting the register mapping button as highlighted below.



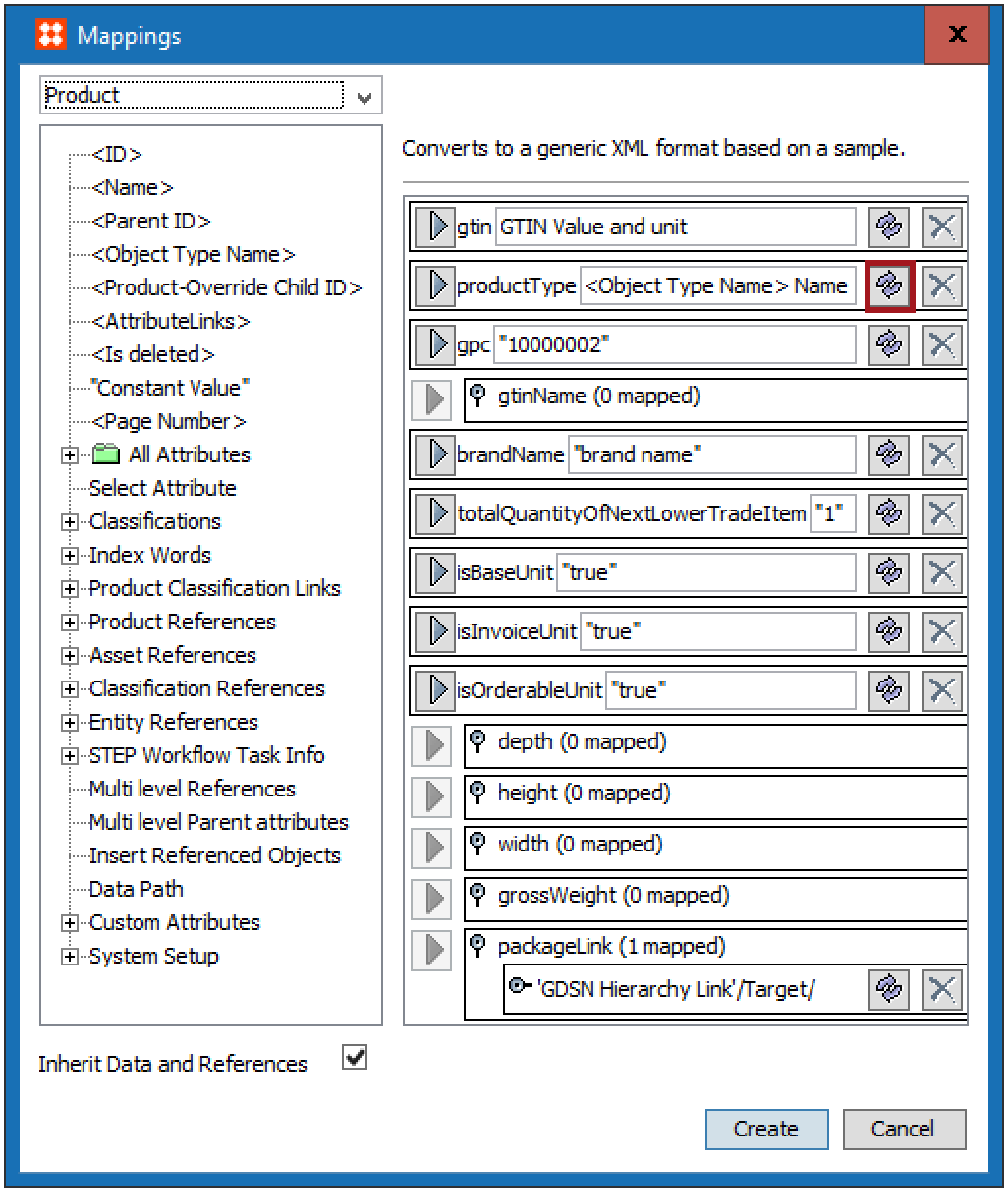
Command	Template	Mapping	Type
publish	<?xml version="1.0" en...	GTIN Value and unit	Publish
register	<?xml version="1.0" en...	GTIN Value and unit, 'GDSN Hierarchy Link'/Target/, <Object Type Name> N...	Register Product and Package Hierarchy
registerLink	<?xml version="1.0" en...	'GDSN Hierarchy Link'/Target/	Add Package Hierarchy Link
rePublish	<?xml version="1.0" en...	GTIN Value and unit	Publish
unPublish	<?xml version="1.0" en...	GTIN Value and unit	UnPublish
unRegisterLink	<?xml version="1.0" en...	'GDSN Hierarchy Link'[GDSNRegistrationStatus=To_be_Unregistered]/Target/	Delete Package Hierarchy Link

Mapping Data Into the Template

Object Types to <productType></productType>

If your model uses object types rather than an attribute value to define the product type, you will need to make transformations on the object type mappings in order to comply with the valid values that are allowable for productType. Follow the steps below to transform your object types to valid values.

1. After selecting the button, select the transformation button for **productType** highlighted in the image below.

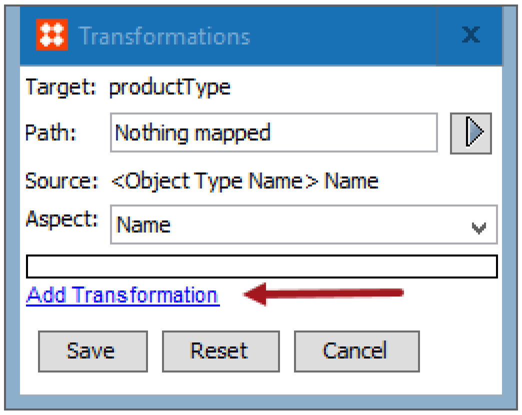


Converts to a generic XML format based on a sample.

gtin GTIN Value and unit
productType <Object Type Name> Name
gpc "10000002"
gtinName (0 mapped)
brandName "brand name"
totalQuantityOfNextLowerTradeItem "1"
isBaseUnit "true"
isInvoiceUnit "true"
isOrderableUnit "true"
depth (0 mapped)
height (0 mapped)
width (0 mapped)
grossWeight (0 mapped)
packageLink (1 mapped)
'GDSN Hierarchy Link'/Target/

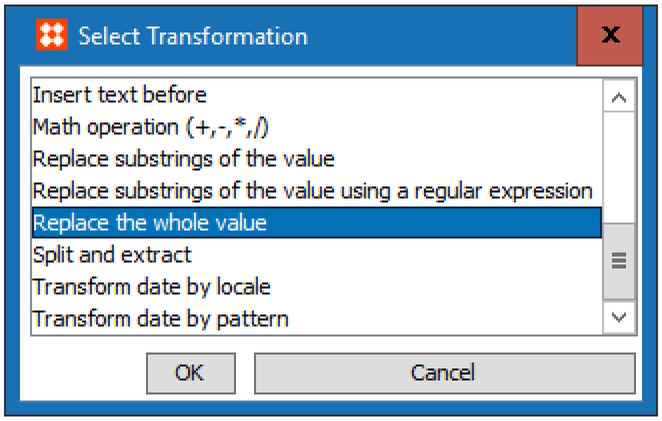
Inherit Data and References ✓

2. After the **Transformation** dialog box appears, select the **Add Transformation** link.



Target: productType
Path: Nothing mapped
Source: <Object Type Name> Name
Aspect: Name

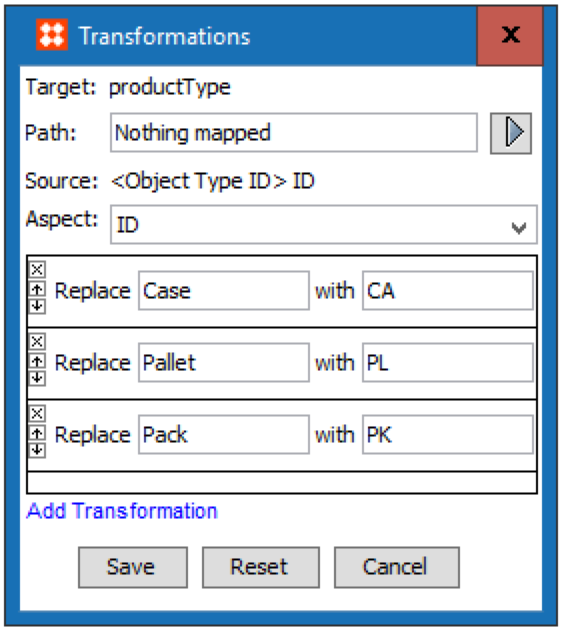
3. In the **Select Transformation** dialog box, find and select the **Replace the whole value** transformation.



Select Transformation

Insert text before
Math operation (+,-,*,/)
Replace substrings of the value
Replace substrings of the value using a regular expression
Replace the whole value
Split and extract
Transform date by locale
Transform date by pattern

4. From the **Transformations** dialog box, you can replace the object type ID or Name with the correlating GDSN qualifier for that object type. This is what gets mapped to be sent to the data pool. Since ID's cannot be changed to a different value, it is recommended that the ID value is replace with the appropriate qualifier. Continue to use the Add Transformation link until all of the product object types being sent to the data pool have the correct qualifier.



Transformations

Target: productType
Path: Nothing mapped
Source: <Object Type ID> ID
Aspect: ID

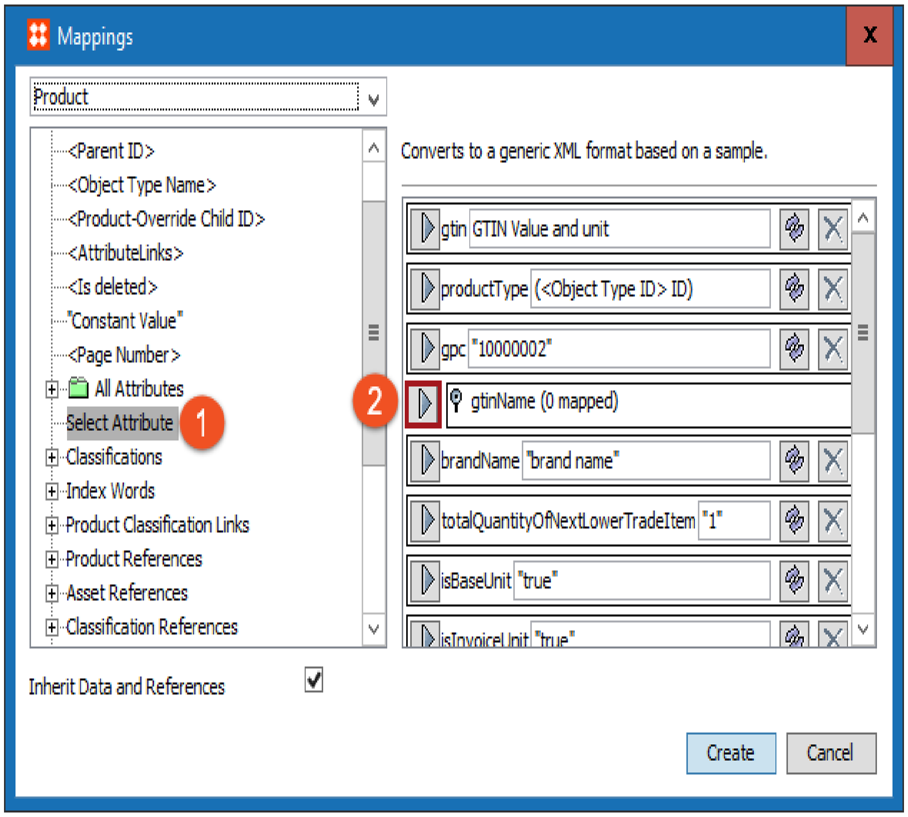
Replace Case with CA
Replace Pallet with PL
Replace Pack with PK

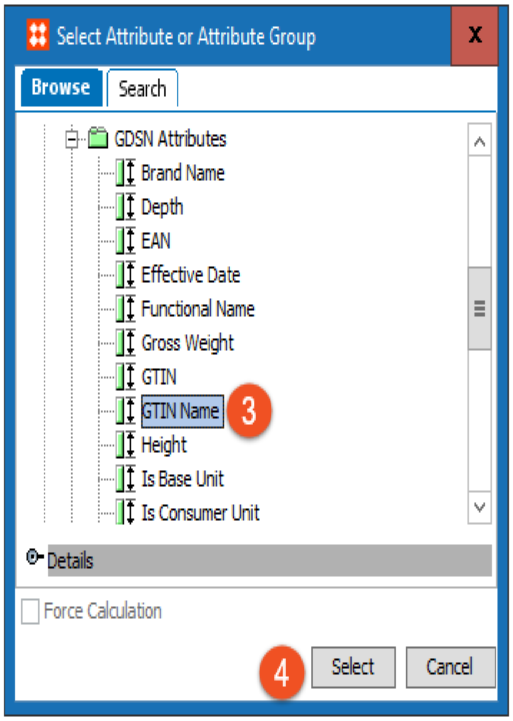
Normal Attributes

Attribute values that are found directly on the trade item are mapped simply by selecting 'Select Attribute' and clicking the right arrow by the attribute you're mapping, then choosing the desired attribute.

1. Select **Select Attribute** from the product mappings menu.
2. Select the mapping arrow.

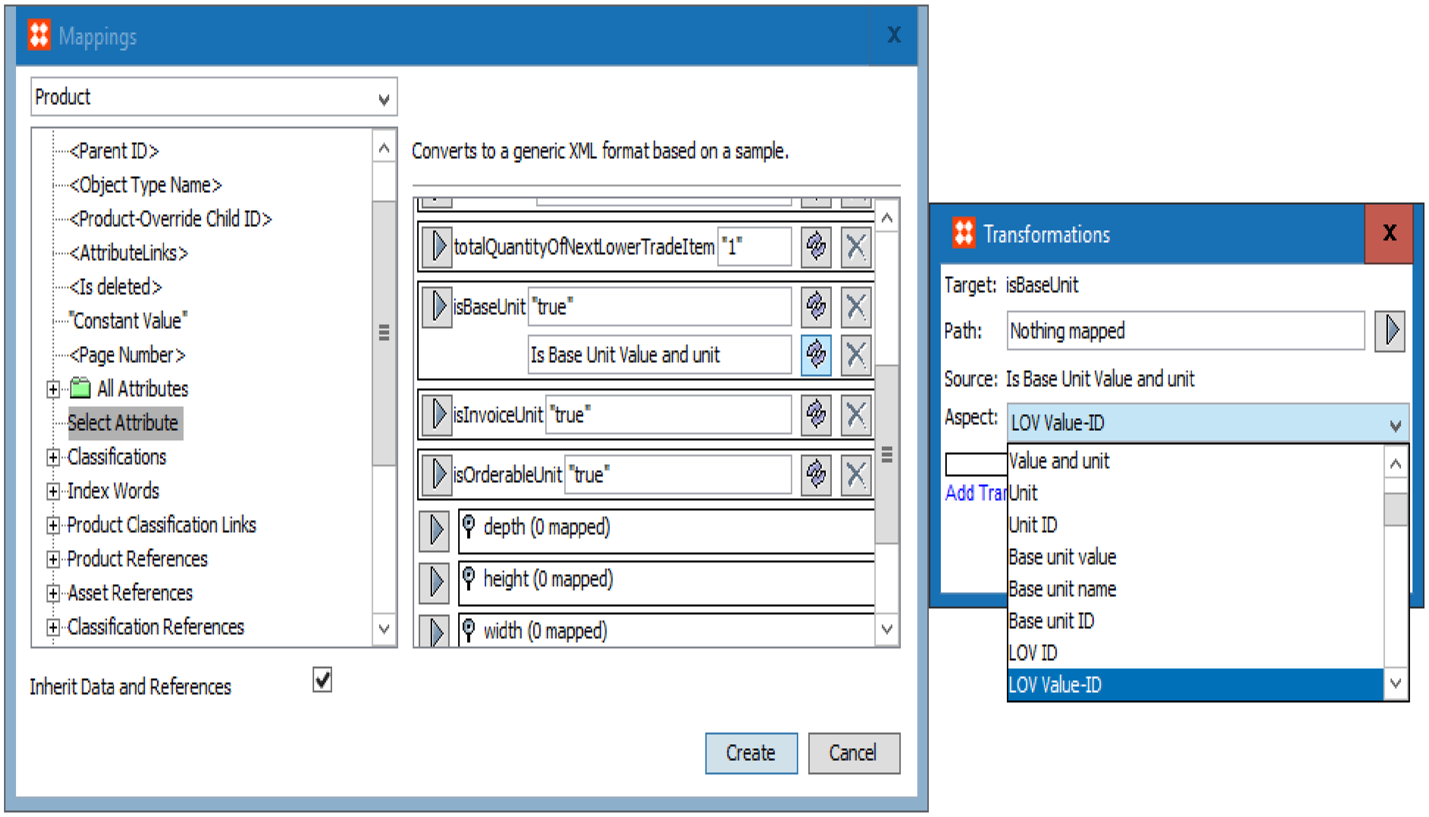
3. Choose the appropriate attribute.
4. Press **Select**.





LOVs With ID's

If you are mapping a value that uses an LOV and the LOV ID holds the GDSN value, change the mapping from the default aspect, **Value and Unit** to **LOV Value ID**.

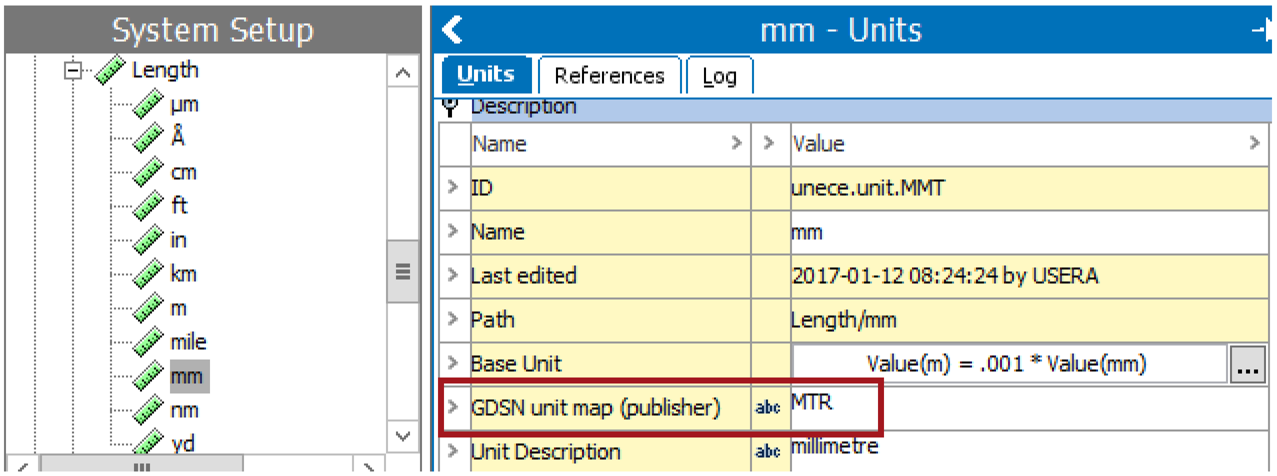


Units of Measure

If an attribute has a unit of measure qualifier, you will need to map that separately. GDSN uses specific UOM qualifiers that are not used for most publishing requirements in an MDM solution. For example, “70.0 mm” is expressed as “MTR” as shown here:

<attrQual name="depth" qual="**MTR**">70.0</attrQual>

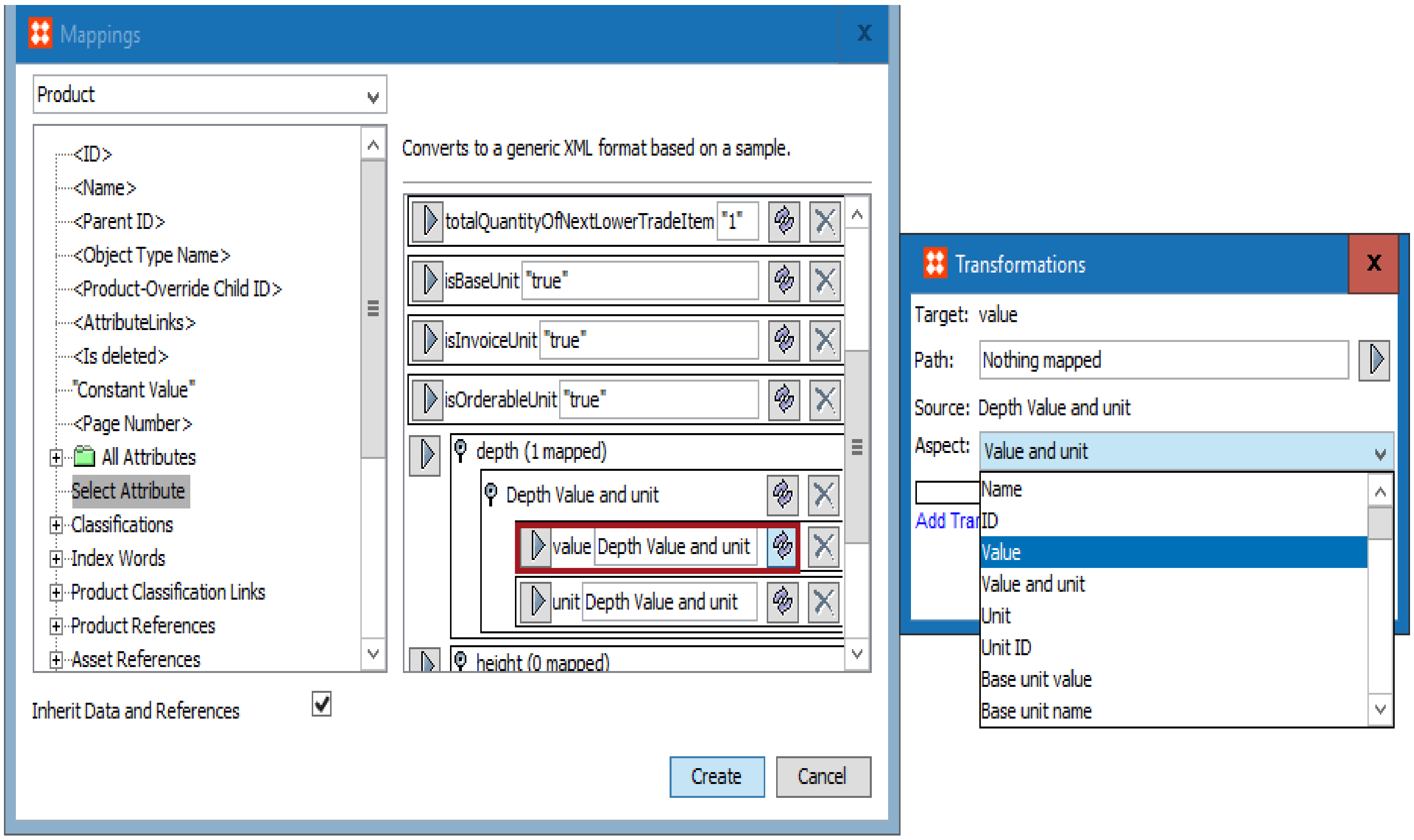
There is metadata on the STEP UOM called **GDSN unit map (publisher)**. This is used for the GDSN qualifiers in outbound messages. For the units of measures that you will be using to send to the GDSN, select the appropriate units under System Setup and enter the qualifier for each unit in the **GDSN unit map (publisher)** attribute.


System Setup

mm - Units

Name		Value
ID		unece.unit.MMT
Name		mm
Last edited		2017-01-12 08:24:24 by USERA
Path		Length/mm
Base Unit		Value(m) = .001 * Value(mm)
GDSN unit map (publisher)	abc	MTR
Unit Description	abc	millimetre

First change the default mapping for the attribute from '**Value and unit**' to '**Value**'. The transformation button for this value can be found by pressing the flipper next to **Depth Value and unit**.


Converts to a generic XML format based on a sample.

Transformations

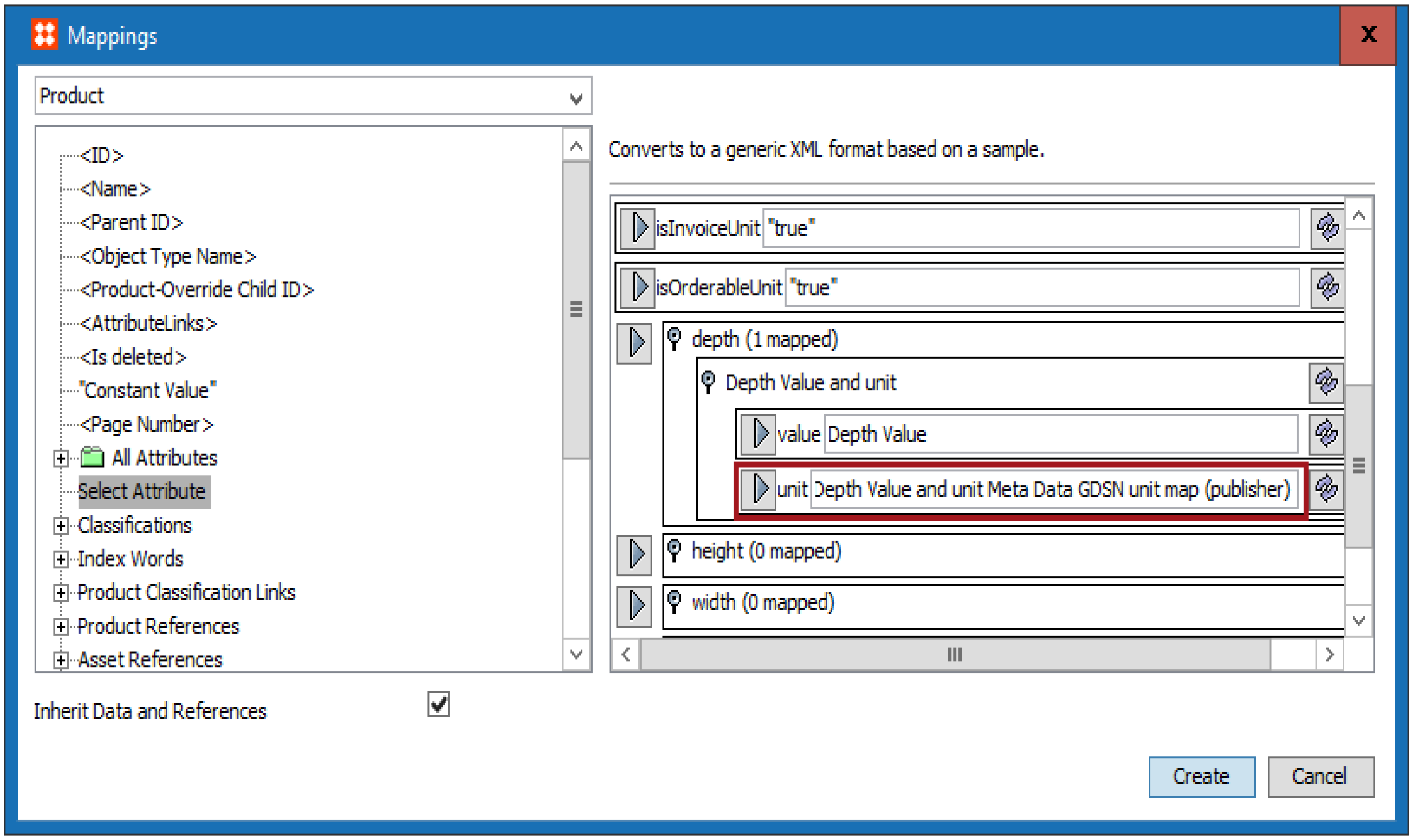
Target: value

Path: Nothing mapped

Source: Depth Value and unit

Aspect: Value and unit

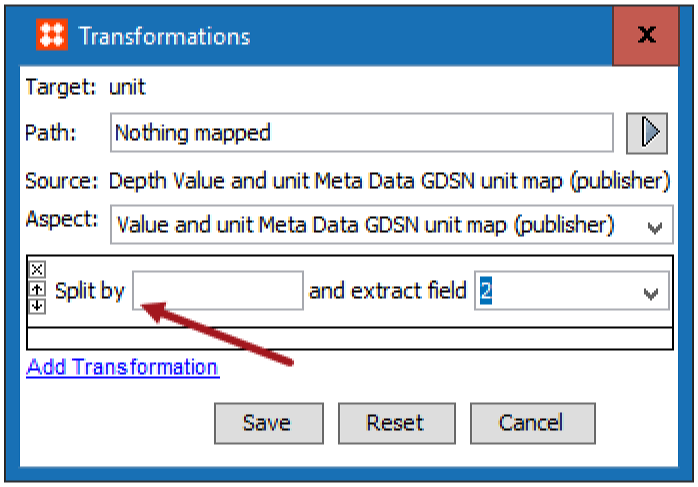
Next map the unit of measure as '**Value and unit meta Data GDSN unit map (publisher)**' as shown in the image below.



Converts to a generic XML format based on a sample.

isInvoiceUnit "true"
isOrderableUnit "true"
depth (1 mapped)
Depth Value and unit
value Depth Value
unit Depth Value and unit Meta Data GDSN unit map (publisher)
height (0 mapped)
width (0 mapped)

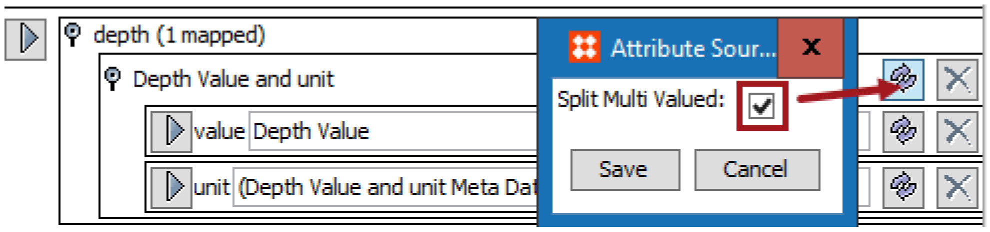
Since there is already a default value assigned to the unit of measure, (prior to the GDSN value), a transformation must be applied on the unit to remove the original value from the UOM. To do this, add the transformation **Split and extract**. Enter a space over the **x** in the **Split by** field, (where the arrow is in the image below), and change the **and extract field** to two. Save.



Target: unit
Path: Nothing mapped
Source: Depth Value and unit Meta Data GDSN unit map (publisher)
Aspect: Value and unit Meta Data GDSN unit map (publisher)
Split by and extract field 2

Multi-valued Attributes

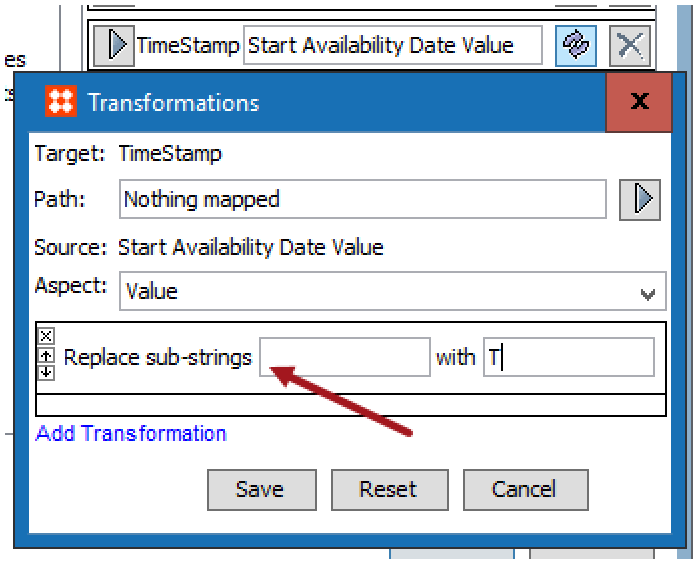
If an attribute is multi-valued, then the outbound message mapping should have the "Split multi-valued" box enabled.



Split Multi Valued: ✓

GDSN Dates

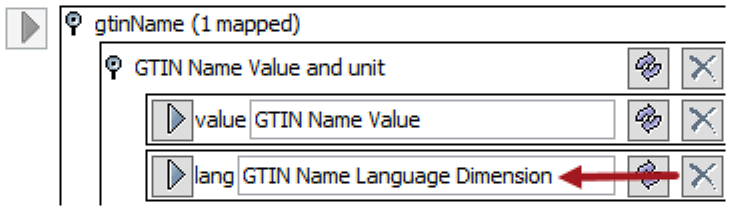
GDSN uses ULC format instead of ISOdatetime. A '**Replace sub-strings**' transformation must be added and the 'x' should be replaced with a space and the 'y; should be replaced with the letter 'T'.



Target: TimeStamp
Path: Nothing mapped
Source: Start Availability Date Value
Aspect: Value
Replace sub-strings with T

Language Qualifiers

Attributes that have a language qualifier are mapped for both the value and the language dimension.



GDSN Message Exception Handling

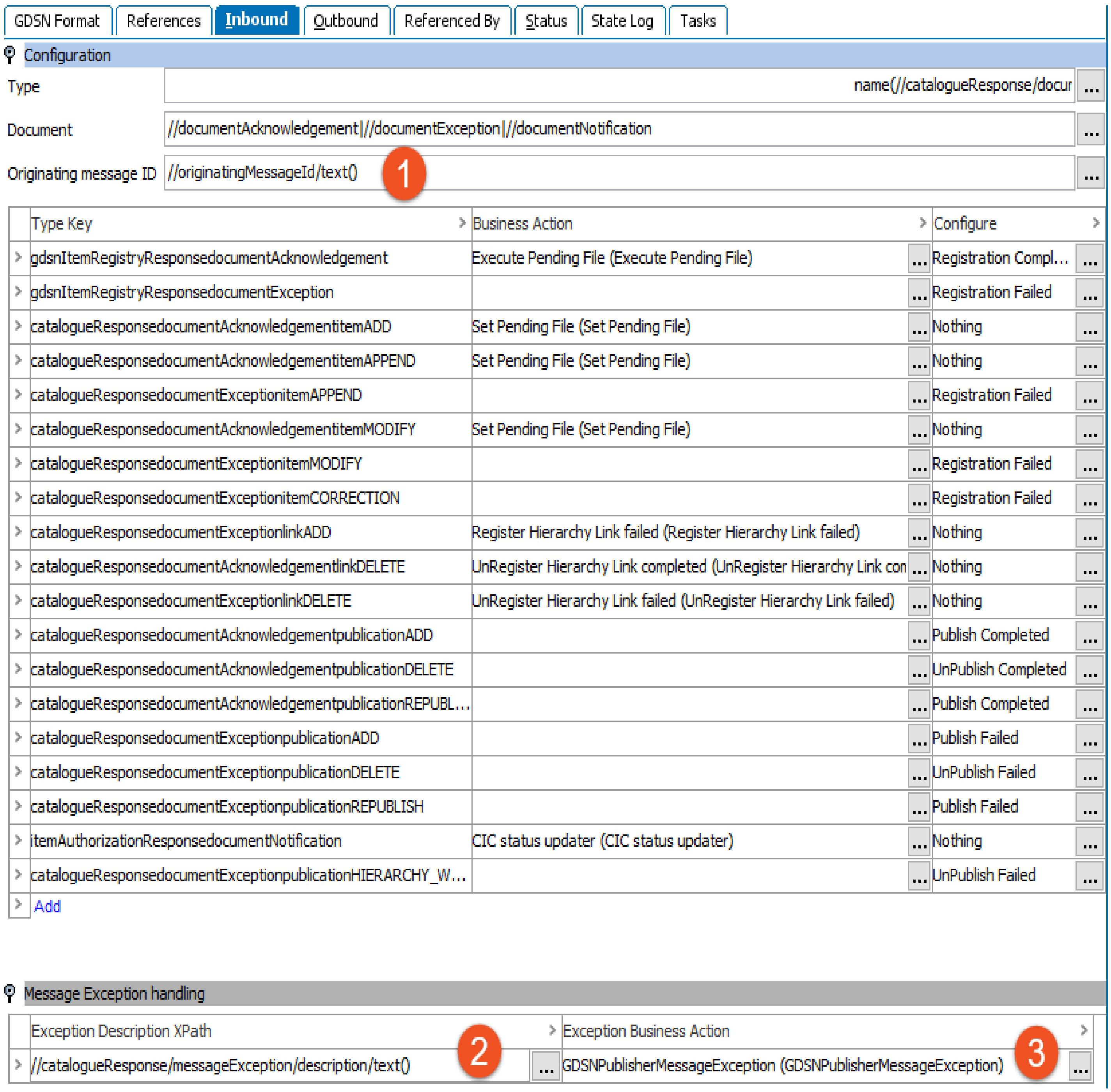
This topic covers general information regarding how message exception handling works in the GDSN Provider and Receiver solution. Ultimately it is up to each implementation to determine how to handle message exceptions when the response comes back from the data pool but an example is provided here as a guide to how the process works.

Process Overview

When a message is submitted to the data pool, a response message is sent back from the data pool acknowledging receipt of the original message. These can come in the form of a documentAcknowledgment, a documentException or, a messageException. Sometimes Message Exceptions are sent back from the data pool indicating an error with the original message. This process describes the Message Exception process where the only identifying data back to the original message is the message ID. STEP generates a unique message ID each time a message is sent to the data pool. When a Message Exception is sent back from the data pool, it contains the originating message ID along with any errors that may have been included in the data of the original message. STEP provides a way for the message exception from the data pool to be tracked so that corrective actions can be taken.

On the Inbound tab of the GDSN Format object type or the GDSN Receiver Format object type, there are three configurations that support the handling of GDSN message exceptions. All of these are provided by default and are identified in the images below.

1. **Originating message ID**: This is the XPath that captures the originating message ID.
2. **Exception Description XPath**: This is the XPath that captures the message exception.
3. **Exception Business Action**: This is for the business action to be configured for the way you want to handle message exceptions. The 'GDSNPublisherMessageException' business action for the GDSN Provider and a 'GDSNReceiverMessageException' for the GDSN Receiver are created when the **Easy setup of the GDSN data pool** runs. However, these business rules are not pre-configured as it is at the discretion of the GDSN admin user to determine how to handle message exceptions.



Type	name(//catalogueResponse/docur
Document	//documentAcknowledgement|//documentException|//documentNotification
Originating message ID	//originatingMessageId/text()

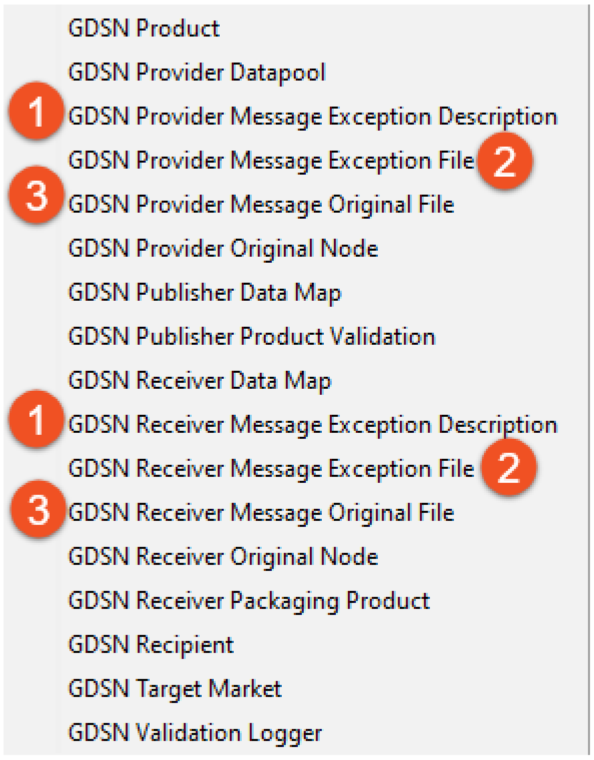
Type Key	Business Action	Configure
gdsnItemRegistryResponsedocumentAcknowledgement	Execute Pending File (Execute Pending File)	Registration Compl...
gdsnItemRegistryResponsedocumentException		Registration Failed
catalogueResponsedocumentAcknowledgementitemADD	Set Pending File (Set Pending File)	Nothing
catalogueResponsedocumentAcknowledgementitemAPPEND	Set Pending File (Set Pending File)	Nothing
catalogueResponsedocumentExceptionitemAPPEND		Registration Failed
catalogueResponsedocumentAcknowledgementitemMODIFY	Set Pending File (Set Pending File)	Nothing
catalogueResponsedocumentExceptionitemMODIFY		Registration Failed
catalogueResponsedocumentExceptionitemCORRECTION		Registration Failed
catalogueResponsedocumentExceptionlinkADD	Register Hierarchy Link failed (Register Hierarchy Link failed)	Nothing
catalogueResponsedocumentAcknowledgementlinkDELETE	UnRegister Hierarchy Link completed (UnRegister Hierarchy Link con	Nothing
catalogueResponsedocumentExceptionlinkDELETE	UnRegister Hierarchy Link failed (UnRegister Hierarchy Link failed)	Nothing
catalogueResponsedocumentAcknowledgementpublicationADD		Publish Completed
catalogueResponsedocumentAcknowledgementpublicationDELETE		UnPublish Completed
catalogueResponsedocumentAcknowledgementpublicationREPUBL...		Publish Completed
catalogueResponsedocumentExceptionpublicationADD		Publish Failed
catalogueResponsedocumentExceptionpublicationDELETE		UnPublish Failed
catalogueResponsedocumentExceptionpublicationREPUBLISH		Publish Failed
itemAuthorizationResponsedocumentNotification	CIC status updater (CIC status updater)	Nothing
catalogueResponsedocumentExceptionpublicationHIERARCHY_W...		UnPublish Failed

Exception Description XPath	Exception Business Action
//catalogueResponse/messageException/description/text()	GDSNPublisherMessageException (GDSNPublisherMessageException)

GDSN Message Exception Binds

While this topic does not describe business rules or JavaScript Binds, there are three binds specific to the message exception handling process that can be useful when creating JavaScript business rules for the business action.

1. **GDSN Message Exception Description**: This bind will provide the GDSN message exception description which is stored on the background process.
2. **GDSN Provider / Receiver Message Exception File**: This bind will provide the GDSN message exception file.
3. **GDSN Provider / Receiver Original File**: This bind will provide the original outbound message file.

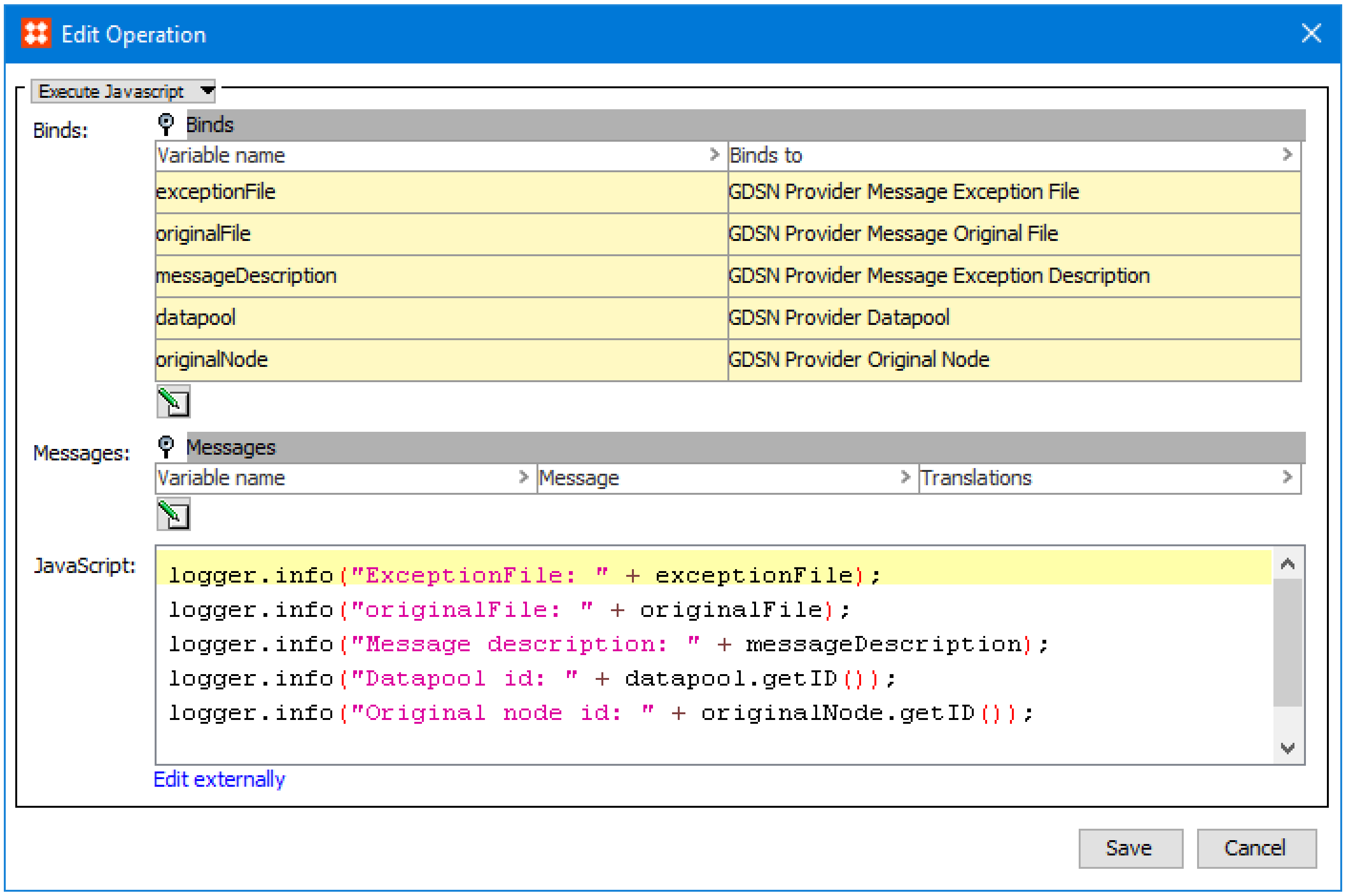

GDSN Product
GDSN Provider Datapool
1
GDSN Provider Message Exception Description
GDSN Provider Message Exception File
2
3
GDSN Provider Message Original File
GDSN Provider Original Node
GDSN Publisher Data Map
GDSN Publisher Product Validation
GDSN Receiver Data Map
1
GDSN Receiver Message Exception Description
GDSN Receiver Message Exception File
2
3
GDSN Receiver Message Original File
GDSN Receiver Original Node
GDSN Receiver Packaging Product
GDSN Recipient
GDSN Target Market
GDSN Validation Logger

For more information about Business Rules, please see the **Business Rules** section.

For more information about JavaScript Binds, see the **JavaScript Binds** section.

A Message Exception Handling Process Example

To understand the flow of how the message exception process works in STEP, an example is being provided. In this example, we will register a product to the data pool from the GDSN Provider solution using the following business action.



Binds:

Variable name	Binds to
exceptionFile	GDSN Provider Message Exception File
originalFile	GDSN Provider Message Original File
messageDescription	GDSN Provider Message Exception Description
datapool	GDSN Provider Datapool
originalNode	GDSN Provider Original Node

Messages:

Variable name	Message	Translations

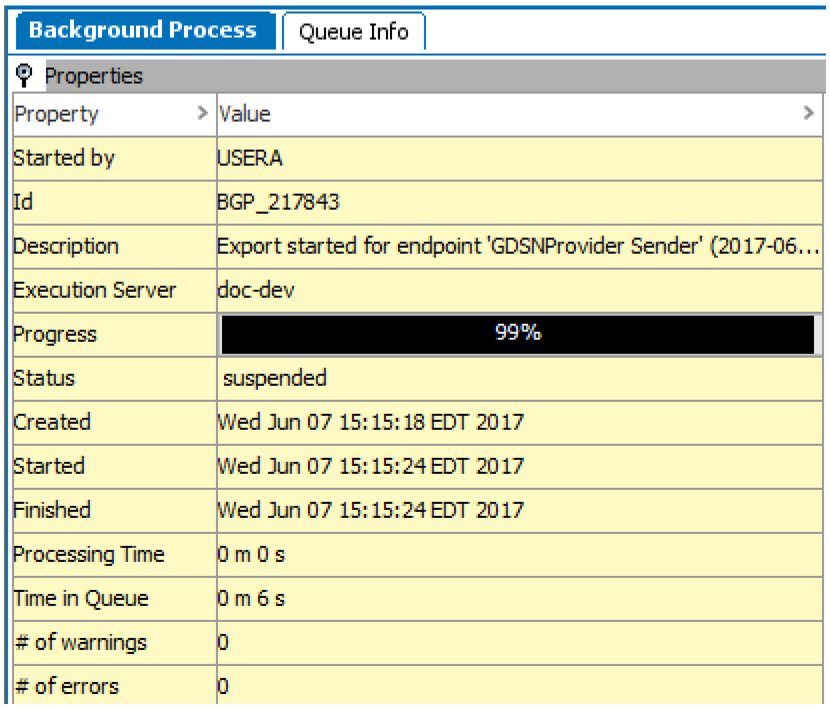
JavaScript:

```
logger.info("ExceptionFile: " + exceptionFile);
logger.info("originalFile: " + originalFile);
logger.info("Message description: " + messageDescription);
logger.info("Datapool id: " + datapool.getID());
logger.info("Original node id: " + originalNode.getID());
```

From the Web UI, a case of product has been registered to the GDSN. This generates the outbound message to the data pool containing the unique message ID as shown below.

```
<messageId>register_5867413948702#BGP_217843</messageId>
```

If generated successfully, the outbound background process will set to 99% progress. This is due to the fact that a response is expected from the data pool.



Background Process | Queue Info

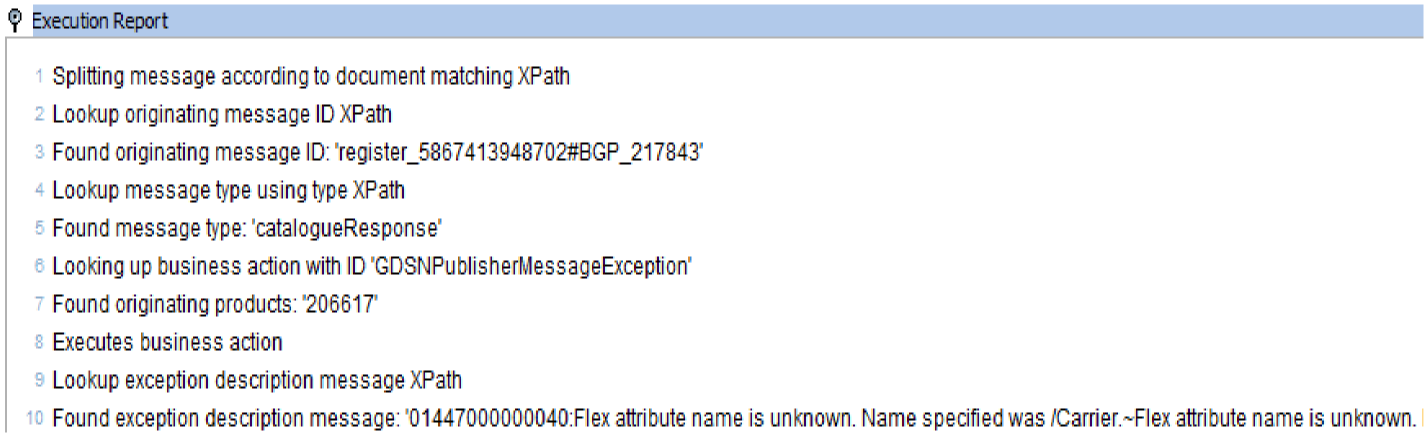
Properties

Property	Value
Started by	USERA
Id	BGP_217843
Description	Export started for endpoint 'GDSNProvider Sender' (2017-06...
Execution Server	doc-dev
Progress	99%
Status	suspended
Created	Wed Jun 07 15:15:18 EDT 2017
Started	Wed Jun 07 15:15:24 EDT 2017
Finished	Wed Jun 07 15:15:24 EDT 2017
Processing Time	0 m 0 s
Time in Queue	0 m 6 s
# of warnings	0
# of errors	0

A response message is sent back from the data pool with the originating message ID. This message ID is used to bind the inbound message back to the original message.

```
<originatingMessageId>register_5867413948702#BGP_217843</originatingMessageId>
```

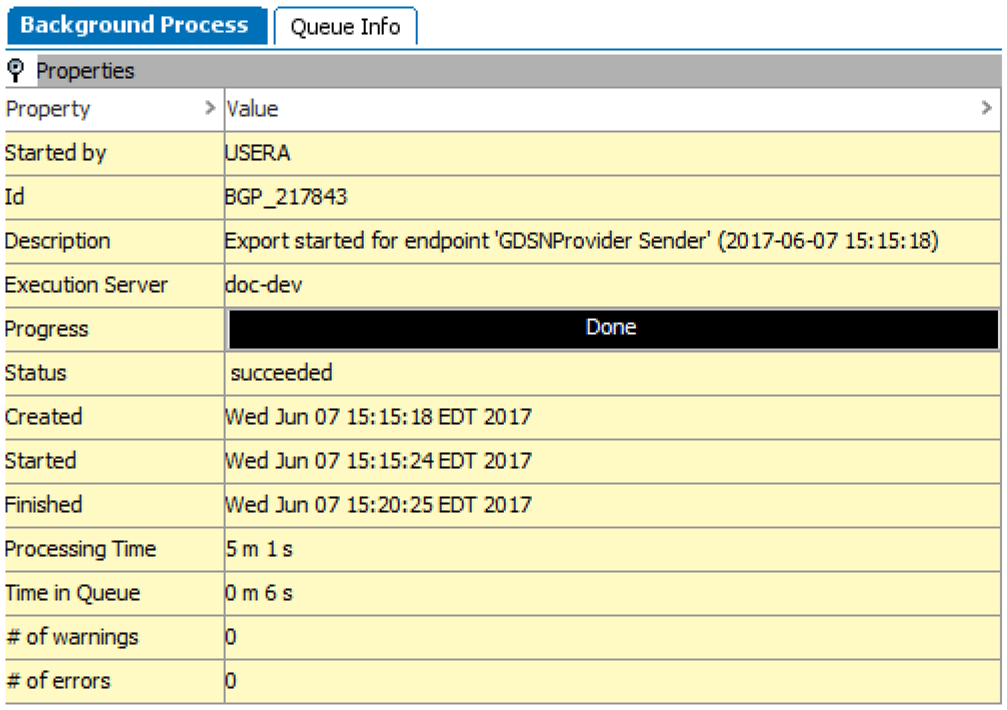
An error report is generated in the background process that can be used to send the product to a workflow.



Execution Report

```
Splitting message according to document matching XPath
Lookup originating message ID XPath
Found originating message ID: 'register_5867413948702#BGP_217843'
Lookup message type using type XPath
Found message type: 'catalogueResponse'
Looking up business action with ID 'GDSNPublisherMessageException'
Found originating products: '206617'
Executes business action
Lookup exception description message XPath
Found exception description message: '01447000000040:Flex attribute name is unknown. Name specified was /Carrier.~Flex attribute name is unknown.
```

The original outbound file will move to 'Done' in the Background Process Progress


Background Process | Queue Info

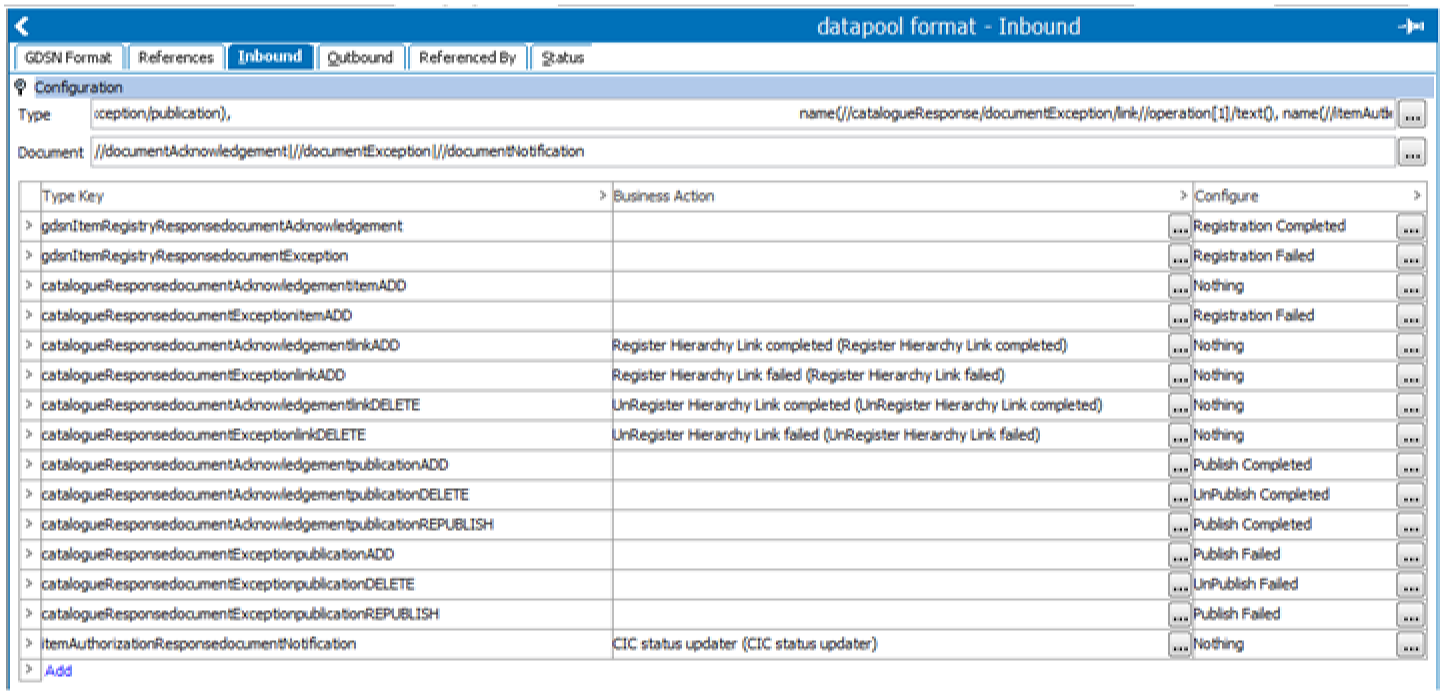
Properties

Property	Value
Started by	USERA
Id	BGP_217843
Description	Export started for endpoint 'GDSNProvider Sender' (2017-06-07 15:15:18)
Execution Server	doc-dev
Progress	Done
Status	succeeded
Created	Wed Jun 07 15:15:18 EDT 2017
Started	Wed Jun 07 15:15:24 EDT 2017
Finished	Wed Jun 07 15:20:25 EDT 2017
Processing Time	5 m 1 s
Time in Queue	0 m 6 s
# of warnings	0
# of errors	0

Configuring the Inbound Message Format

The inbound configuration defines the way the data pool receives messages from the GDSN. The inbound tab has 3 areas:

- **Type** field: XPath to find the type of the incoming message
- **Document** field: XPath to split messages into documents
- **Configuration** table: table to configure message types



Type: xception/publication), name(//catalogueResponse/documentException/link//operation[1]/text(), name(//itemAuth

Document: //documentAcknowledgement|//documentException|//documentNotification

Type Key	Business Action	Configure
gdsnItemRegistryResponsedocumentAcknowledgement		Registration Completed
gdsnItemRegistryResponsedocumentException		Registration Failed
catalogueResponsedocumentAcknowledgementitemADD		Nothing
catalogueResponsedocumentExceptionitemADD		Registration Failed
catalogueResponsedocumentAcknowledgementlinkADD	Register Hierarchy Link completed (Register Hierarchy Link completed)	Nothing
catalogueResponsedocumentExceptionlinkADD	Register Hierarchy Link failed (Register Hierarchy Link failed)	Nothing
catalogueResponsedocumentAcknowledgementlinkDELETE	UnRegister Hierarchy Link completed (UnRegister Hierarchy Link completed)	Nothing
catalogueResponsedocumentExceptionlinkDELETE	UnRegister Hierarchy Link failed (UnRegister Hierarchy Link failed)	Nothing
catalogueResponsedocumentAcknowledgementpublicationADD		Publish Completed
catalogueResponsedocumentAcknowledgementpublicationDELETE		UnPublish Completed
catalogueResponsedocumentAcknowledgementpublicationREPUBLISH		Publish Completed
catalogueResponsedocumentExceptionpublicationADD		Publish Failed
catalogueResponsedocumentExceptionpublicationDELETE		UnPublish Failed
catalogueResponsedocumentExceptionpublicationREPUBLISH		Publish Failed
itemAuthorizationResponsedocumentNotification	CIC status updater (CIC status updater)	Nothing

Type field

The type field contains an XPath to determine the type of the incoming message. It evaluates the XPath on the message and uses the result as a key to finding the message type.

The following shows an XML sample:

```
<note>
<to>Bob</to>
<from>Frank</from>
<heading>Reminder</heading>
<body>Updates due on Tuesday.</body>
</note>
```

To find out whether this is a note or a letter message type, apply the following XPath:

```
concat(name(//note),name(//letter))
```

In this example, the note tag is returned because note is present in the XML.

Document

Sometimes there are multiple actions in a GDSN message, for example, when more products are registered at the same time. In this case several documents are returned for parsing. The document XPath is used to specify which documents are to be handled separately. In the XML example above, the following XPath returns 2 parts of the XML.

```
//body|//heading
```

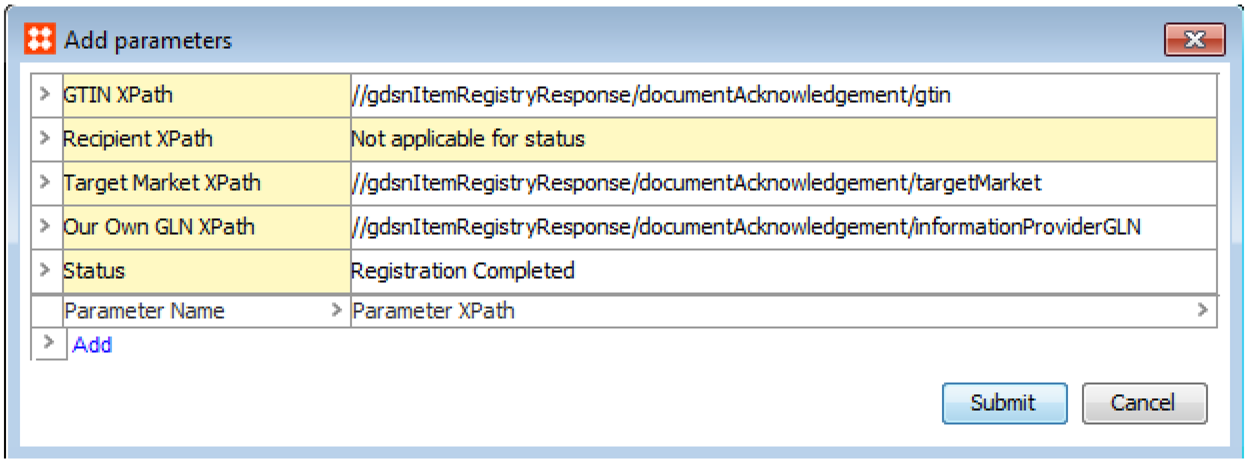
These XML parts are removed from the XML and inserted one at a time. The result messages are evaluated to determine the message type via the type XPath.

Configuration table of message types

The key that is found using the type from the previous example is used to determine what to do with the incoming message.

For each type key in the **Type Key** column, the related business rule is displayed when applicable in the **Business Action** column, and the related configuration is displayed in the **Configure** column.

1. To setup the configuration of a specific document type, click the ellipsis button **(...)** in the **Configure** column. The **Add Parameters** dialog is displayed.



GTIN XPath	//gdsnItemRegistryResponse/documentAcknowledgement/gtin
Recipient XPath	Not applicable for status
Target Market XPath	//gdsnItemRegistryResponse/documentAcknowledgement/targetMarket
Our Own GLN XPath	//gdsnItemRegistryResponse/documentAcknowledgement/informationProviderGLN
Status	Registration Completed
Parameter Name	Parameter XPath

2. Specify the XPaths that are applicable for the document type to identify important attributes in the document. In some cases the Recipient XPath is not applicable.
3. Specify the status that you want this action to set on the targeted item. If no status is applicable, choose None. If you chose none, verify that the status of your item is correct after this is run. You can for example use business action to verify this.

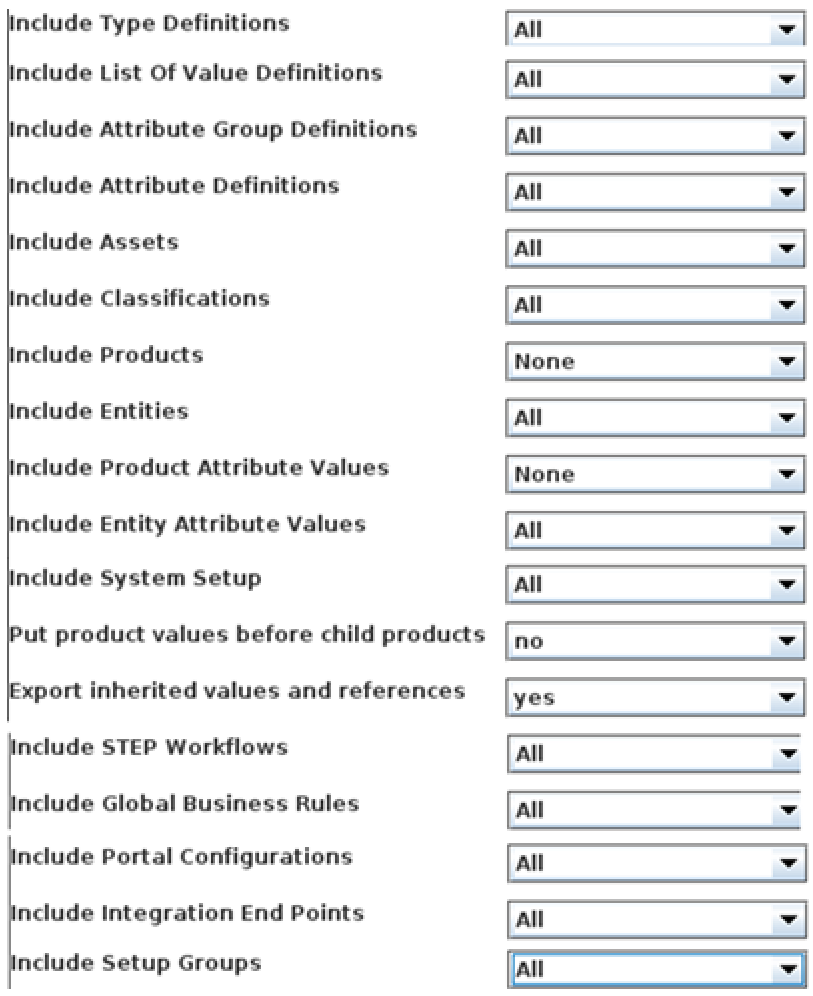
4. Optionally, add a parameter to the action. This requires a name and an XPath. The XPath is used to find the information needed in the document, and the name is used to identify this information later. The mapped information is accessed is by using the business rule that is associated with the current action. In the business action, use the JavaScript Plug-in and bind in the GDSN Data Map. The data can now be found using the parameter name.

Moving a GDSN Setup to a New System

For both **GDSN Receiver** and **GDSN Provider**, the following describes how to move a system that has been set up with the easy setup wizard. Although this is the DTAP-recommended approach (Development-Test-Acceptance-Production), the scenario should be modified to meet your company requirements.

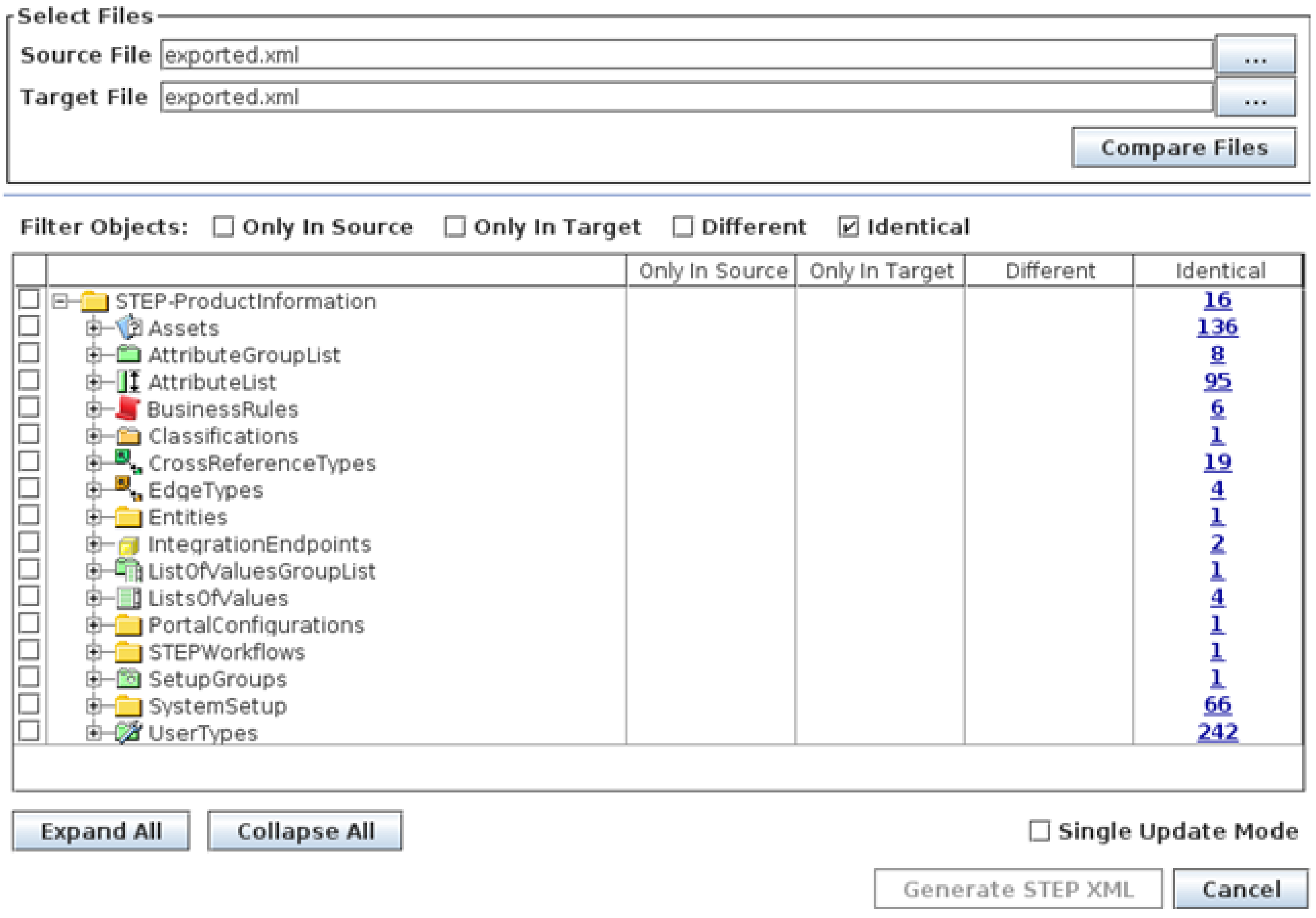
To move a GDSN setup from one system to another, use the Export Manager's STEPXML format and the Export Comparison tool. For information about exporting data and assets, see Creating a Data Export or Exporting Assets.

1. In the **Tree**, locate the XSD asset that has been uploaded for GDSN.
2. Right-click the XSD asset and click **Export Images & Documents**.
3. In the Tree, right-click the relevant data pool and click **Export data below** to export the remaining system setup. The export must not include the component model. This image shows the relevant STEPXML options.



Include Type Definitions	All
Include List Of Value Definitions	All
Include Attribute Group Definitions	All
Include Attribute Definitions	All
Include Assets	All
Include Classifications	All
Include Products	None
Include Entities	All
Include Product Attribute Values	None
Include Entity Attribute Values	All
Include System Setup	All
Put product values before child products	no
Export inherited values and references	yes
Include STEP Workflows	All
Include Global Business Rules	All
Include Portal Configurations	All
Include Integration End Points	All
Include Setup Groups	All

4. From the **File** menu, point to **Export** and click **Compare System Setup Exports**. Remove any exported elements that should not be imported into the new system.



Select Files

Source File: exported.xml

Target File: exported.xml

Filter Objects: ☐ Only In Source ☐ Only In Target ☐ Different ☑ Identical

		Only In Source	Only In Target	Different	Identical
☐	STEP-ProductInformation				16
☐	Assets				136
☐	AttributeGroupList				8
☐	AttributeList				95
☐	BusinessRules				6
☐	Classifications				1
☐	CrossReferenceTypes				19
☐	EdgeTypes				4
☐	Entities				1
☐	IntegrationEndpoints				2
☐	ListOfValuesGroupList				1
☐	ListsOfValues				4
☐	PortalConfigurations				1
☐	STEPWorkflows				1
☐	SetupGroups				1
☐	SystemSetup				66
☐	UserTypes				242

☐ Single Update Mode

5. For **Source File** and **Target File**, locate the XML file exported and check the Filter Objects **Identical** option.
6. Select the elements needed for the import and click **Generate STEP XML**. A new export file is created that can be imported.
7. Import the export file into the new system.

Note: For **GDSN Receiver** solutions only, when the file is imported, three errors are generated in the BGP Execution Report since references point to objects that do not yet exist. The Data Pool Entity references the product folder 'GDSN IMPORT ROOT' and the assets 'GDSN Receiver DatapoolCINSample' and 'GDSN Receiver DatapoolXSD'. This is expected and does not affect the system import.

8. Import the XSD asset on top of the one that is already in the system to get the content. If the object type of the XSD asset is Zip file, change it to the correct file type as specified in the component model.
9. Manually set the references on the **Data Pool Format Entity** references tab.
10. Create a Import Root product folder on the target system.
11. Set the reference to the Import Root product folder on the **Data Pool Entity** references tab.

12. Locate the component model, and specify the correct object types, attributes and references.

GDSN Conversion

If you are using the GDSN solution available prior to STEP Trailblazer 7.1 and want to upgrade to the current STEP GDSN solution, you can use the GDSN conversion script. Before you run the script, do the following:

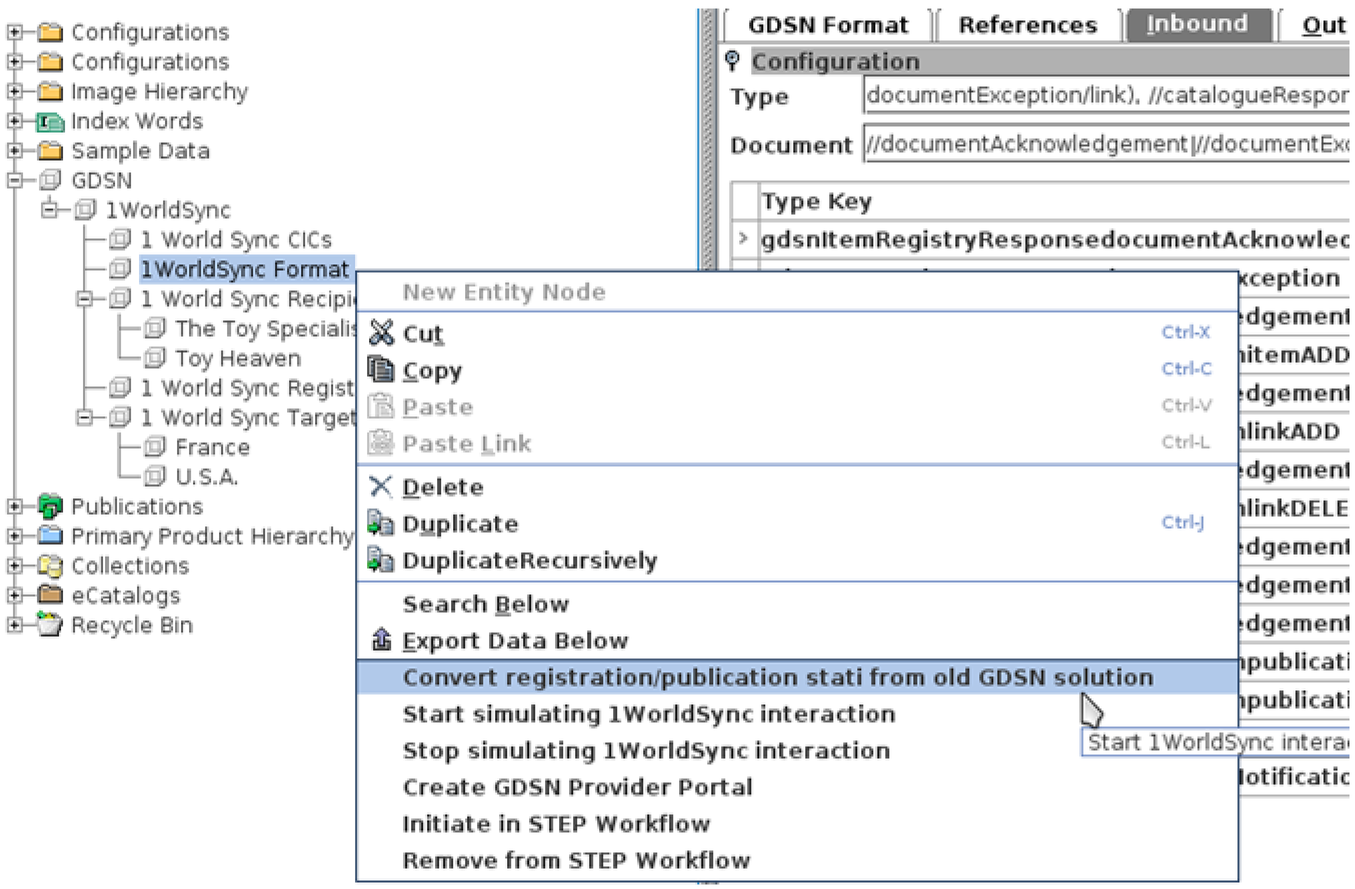
1. Set up the new GDSN Component model.
2. Set up provider data pools that match the provider data pools in the old GDSN solution.
3. For the new data pools, create recipients that match the old GDSN recipients.
4. Create Target Markets for the provider data pools.

The conversion script only needs to read the GLN of the provider data pool to find the corresponding data pool in the old solution. The data pool does not need to be fully configured before you run the conversion script. The GDSN Format Inbound and Outbound mappings, for example, do not need to be configured in order for the conversion script to work.

However, you have to create the target markets before you run the conversion script because the new registrations must link to these registrations. You also have to create the new recipients before you run the conversion script because the publication in the new model needs to create a link to the recipients.

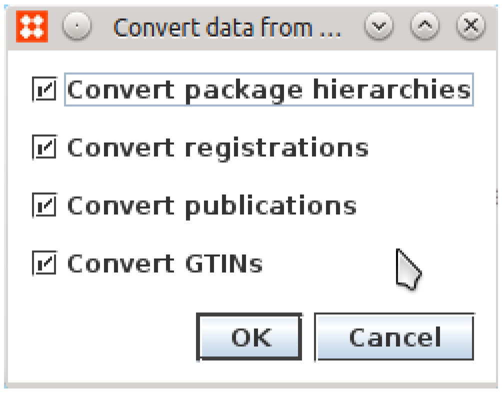
To start the GDSN conversion script

1. In the **Tree**, expand **GDSN**, right-click the relevant GDSN data pool and select **Convert registration/publication stati from old GDSN solution**.



2. In the **Convert data from** dialog, specify which data you want to convert from the old GDSN solution. You have the following options:

- **Convert package hierarchies:** Convert the old GDSN hierarchies into package hierarchies using the new GDSN model.
- **Convert registrations** Convert the status of old GDSN registrations to match the new GDSN solution.
- **Convert publications:** Convert the status of old GDSN publications to match the new GDSN solution.
- **Convert GTINs:** Convert the GTIN of the old GDSN products to the GTIN attribute of the new GDSN Component model.

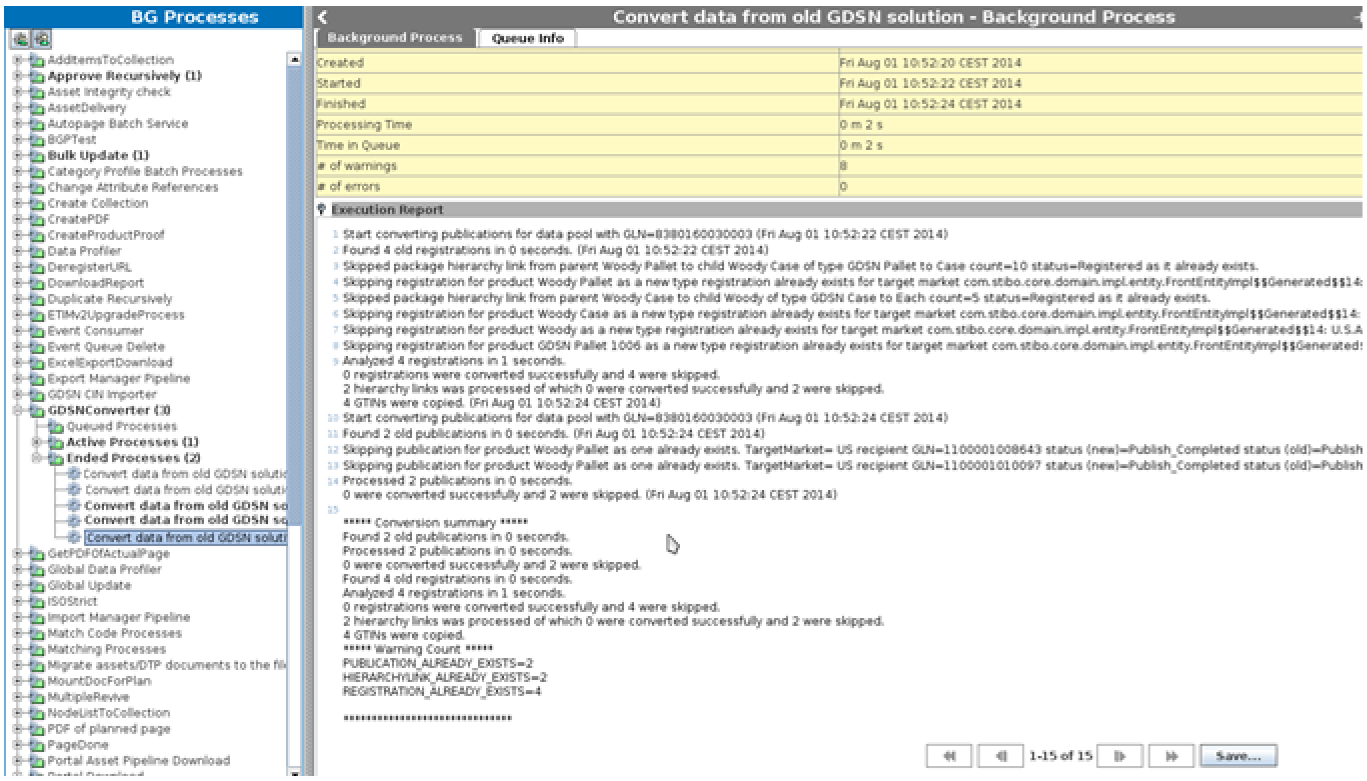


The GDSN conversion script uses the selected data pool to find the corresponding data pool in the old GDSN model. The script finds all old registrations, package hierarchies, and publications for the selected data pool and starts converting them into the new GDSN model.

In the old GDSN solution, the GTIN that a product is registered with is written in a hidden attribute. When you select **Convert GTINs**, this value is copied to the GTIN attribute you have selected for the new GDSN component model. This option is best used when the GTIN attribute used by the new GDSN Component model is not the same as the one that was used when the products were registered. You can also select this option to ensure that the GTIN that is used is the GTIN the product was registered with in case it has been changed by accident.

The conversion is runs as a background process because the conversion takes time if there are many GDSN products for the selected data pool. The conversion script continues to run even if it encounters errors. Information about errors is shown in the execution report of the background process. The execution report also provides information about the progress of the conversion script.

The conversion script is able to handle that data has already been converted. So, if you receive errors, for example, because the GDSN object type hierarchy is not correctly configured, you can restart the conversion script, and then only check the option to convert package hierarchies. When the conversion script encounters data that has already been converted, it skips the conversion of that particular registration, publication, or hierarchy and describes in the execution report that it was skipped.



Created	Fri Aug 01 10:52:20 CEST 2014
Started	Fri Aug 01 10:52:22 CEST 2014
Finished	Fri Aug 01 10:52:24 CEST 2014
Processing Time	0 m 2 s
Time in Queue	0 m 2 s
# of warnings	8
# of errors	0

Execution Report

```
Start converting publications for data pool with GLN=8380160030003 (Fri Aug 01 10:52:22 CEST 2014)
Found 4 old registrations in 0 seconds. (Fri Aug 01 10:52:22 CEST 2014)
Skipped package hierarchy link from parent Woody Pallet to child Woody Case of type GDSN Pallet to Case count=10 status=Registered as it already exists.
Skipping registration for product Woody Pallet as a new type registration already exists for target market com.stibo.core.domain.impl.entity.FrontEntityImpl$$Generated$$14:
Skipped package hierarchy link from parent Woody Case to child Woody of type GDSN Case to Each count=5 status=Registered as it already exists.
Skipping registration for product Woody Case as a new type registration already exists for target market com.stibo.core.domain.impl.entity.FrontEntityImpl$$Generated$$14:
Skipping registration for product Woody as a new type registration already exists for target market com.stibo.core.domain.impl.entity.FrontEntityImpl$$Generated$$14: U.S.A
Skipping registration for product GDSN Pallet 1006 as a new type registration already exists for target market com.stibo.core.domain.impl.entity.FrontEntityImpl$$Generated$
Analyzed 4 registrations in 1 seconds.
0 registrations were converted successfully and 4 were skipped.
2 hierarchy links was processed of which 0 were converted successfully and 2 were skipped.
4 GTINs were copied. (Fri Aug 01 10:52:24 CEST 2014)
Start converting publications for data pool with GLN=8380160030003 (Fri Aug 01 10:52:24 CEST 2014)
Found 2 old publications in 0 seconds. (Fri Aug 01 10:52:24 CEST 2014)
Skipping publication for product Woody Pallet as one already exists. TargetMarket= US recipient GLN=1100001008643 status (new)=Publish_Completed status (old)=Publish
Skipping publication for product Woody Pallet as one already exists. TargetMarket= US recipient GLN=1100001010097 status (new)=Publish_Completed status (old)=Publish
Processed 2 publications in 0 seconds.
0 were converted successfully and 2 were skipped. (Fri Aug 01 10:52:24 CEST 2014)

***** Conversion summary *****
Found 2 old publications in 0 seconds.
Processed 2 publications in 0 seconds.
0 were converted successfully and 2 were skipped.
Found 4 old registrations in 0 seconds.
Analyzed 4 registrations in 1 seconds.
0 registrations were converted successfully and 4 were skipped.
2 hierarchy links was processed of which 0 were converted successfully and 2 were skipped.
4 GTINs were copied.
***** Warning Count *****
PUBLICATION_ALREADY_EXISTS=2
HIERARCHYLINK_ALREADY_EXISTS=2
REGISTRATION_ALREADY_EXISTS=4

*****************************
```

The same type of error or warning is only output a maximum of 100 times in the running feedback. Apart from the running feedback, there is a Conversion summary at the end of the execution report. The summary details how many registrations and publication that were converted and how long it took. It also provides an overview of the number of errors and warnings that were encountered during the conversion process.