pure::variants - Connector for Polarion Manual

pure-systems GmbH

Version 6.0.5.685 for pure::variants 6.0

Copyright © 2003-2024 pure-systems GmbH

2024

Table of Contents

1. Introduction 1
    1.1. About this manual 1
    1.2. Software Requirements 1
    1.3. Installation 2
2. Using the Connector 2
    2.1. Starting pure::variants 2
    2.2. Preparing the Polarion project 2
    2.3. Authentication 3
    2.4. Creating the initial model(s) 4
    2.5. Updating Family Models from Polarion 8
    2.6. Defining a Variant 9
    2.7. Transforming a Variant 10
3. Using the Integration 13
    3.1. Adding Variability Information Using the Desktop Hub 14
    3.2. Adding Variability Information Using the pure::variants Widget 14
    3.3. Introduction to Integration GUI 16
    3.4. Working with Restriction Editor 17
    3.5. Working with Calculations Editor 18
    3.6. Importing and Updating Models using the Web Client 19
    3.7. Variability in HTML Tables 20
    3.8. Link propagation of Polarion link types 21
4. Permission management in Polarion 21
    4.1. Configuring permissions in order to restrict access 22
5. Migration from Polarion Variants 23
6. Known limitations of the Polarion Connector 25
7. Troubleshoot 25
    7.1. Connection Issues - Timeouts, Interrupts, etc. 25

1. Introduction

pure::variants Connector for Polarion enables Polarion users to manage work items variability using pure::variants. By coupling pure::variants and Polarion, knowledge about variability and variants can be formalized, shared and automatically evaluated. This enables getting answers for questions about valid combinations of requirements and test artefacts in product variants quickly, easy monitoring of planned and released product variants at the requirements and test artefact level, as well as very efficient production of variant-specific documents.

1.1. About this manual

The reader is expected to have basic knowledge about and experiences with both tools, Polarion and pure::variants. The pure::variants manual is available as online help here as well as in printable PDF format.

1.2. Software Requirements

The following software is supported by the pure::variants Connector for Polarion:

Tool:

- At minimum Polarion with version 21R2 is required.
- The pure::variants connector for Polarion is tested for the Polarion versions 21R2 and 22R2.
- The Polarion connector for pure::variants requires a deployment of a pure::variants specific plugin on the Polarion server.
- In addition to the pure::variants license a license for Polarion VARIANTS provided by Siemens is needed for the user to use the connector and the integration.
- pure::variants Desktop Hub or Web Hub with version greater than 5.0.10. The pure::variants Desktop Hub is delivered with the pure::variants Enterprise windows installer package and can be installed by selecting the Integration Components in the installer wizard.

The pure::variants Connector for Polarion is an extension for pure::variants and is available on all supported platforms.

1.3. Installation

Please consult section **pure::variants Connectors** in the **pure::variants Setup Guide** for detailed information on how to install the connector (menu **Help** -> **Help Contents** and then **pure::variants Setup Guide** -> **pure::variants Connectors**).

Installation steps specific to the Polarion connector and the deployment of the components are described in the **pure::variants Setup Guide** as well.

2. Using the Connector

2.1. Starting pure::variants

Depending on the installation method used, either start the pure::variants-enabled Eclipse or under Windows select the **pure::variants** item from the **program** menu.

If the **Variant Management** perspective is not already activated, do so by selecting it from **Open Perspective** -> **Other...** in the **Window** menu.

2.2. Preparing the Polarion project

In order to manage the variability information in Polarion a few modifications to the project are necessary.

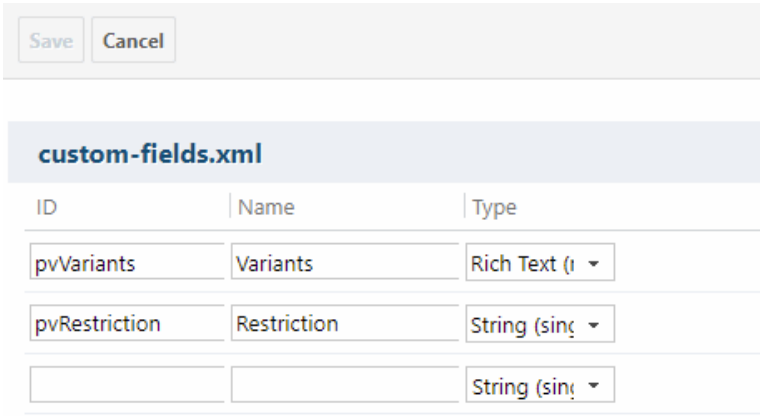
Preparing the custom fields for the work items

To make the variability information available to pure::variants, a field has to be set for every Polarion work item type that shall be processed with respect to its variability.

To set this attribute for each work item type, go to the Polarion settings, expand the '**Work Items**' menu and navigate to the '**Custom Fields**'. In this menu, select the work item types and add a new custom field with an ID in which you want to store the variability information (e.g. '**pvRestriction**'). The type of the custom field must be '**String (single line plain text)**'.

Further, to prepare the Enumeration Transformation to store variability information in the Polarion work items, another custom field is needed with an ID in which you want to store the variants (e.g. '**pvVariants**'). The type of this field must be '**Rich Text (multi line)**'.

Figure 1. Example for work item custom fields needed for the Polarion Connector

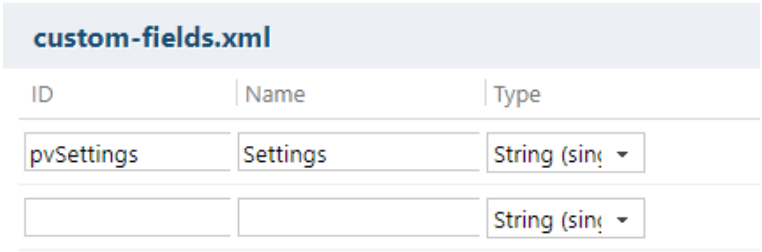


Preparing the custom fields for the document

To store the settings like substitution markers, restriction field ID, … which are set by the pure::variants widget in Polarion, a custom field for the document needs to be created.

To create this custom field go to the Polarion settings, expand the **Documents & Pages** menu and navigate to the **Document Custom Fields**. In this menu, select the document types and add a new custom field with the ID **pvSettings**. The type of the custom field must be **String (single line plain text)**.

Figure 2. Example for document custom fields needed for the Polarion Connector



Note

To configure the settings for the Polarion project or even globally for the Polarion server please have a look at the **pure::variants Setup Guide** at **Section 6.7.4**

2.3. Authentication

To use the connector it is always required to be authenticated by the Polarion application.

There are three authentication mechanisms supported, which have to be configured in Polarion:

1. Using Polarion credentials
2. Oauth 2.0 (for Single-Sign-On)
3. User Account Vault

During the use of the connector, a login dialog will be prompted for all mechanisms expecting the required user credentials. In case of Single-Sign-On a browser-based login dialog will show the Polarion landing page, where the Oauth 2.0 login method can be selected.

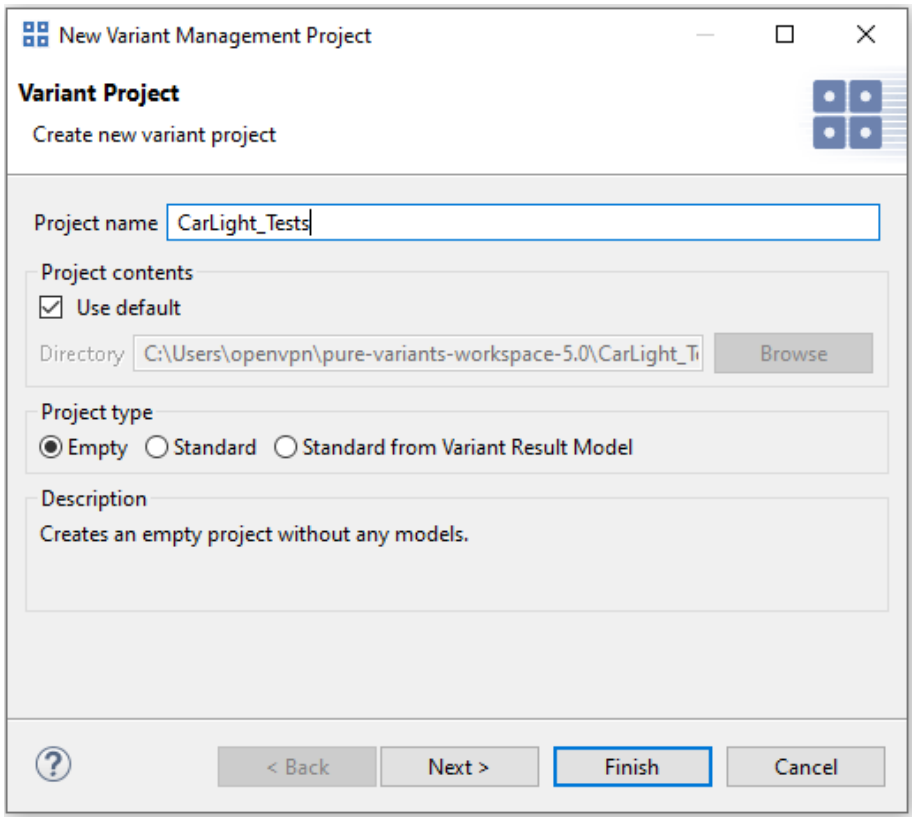
2.4. Creating the initial model(s)

The first step is always to create the corresponding family model for each relevant project containing selected Polarion documents. These initial family models serve as starting points for using existing variability information. The import procedure has to be executed only once for each Polarion document but can be updated afterwards. Each document selected in the import wizard will be represented by one family model in pure::variants, which are created during the import.

Both import and update are supported also using the Web Client, for more information see the section Importing and Updating Models using the Web Client in this document.

Before the actual import can be started, a Variant Management project has to be created for storing the imported models. Select **Project** from **New** in the **File** menu. Choose **Variant Projects** below **Variant Management** in the first page of the **New project** wizard. Choose a name for the project and select **Empty** as project type (see Figure 3, “Creating an empty Variant Management project for Polarion document import”)

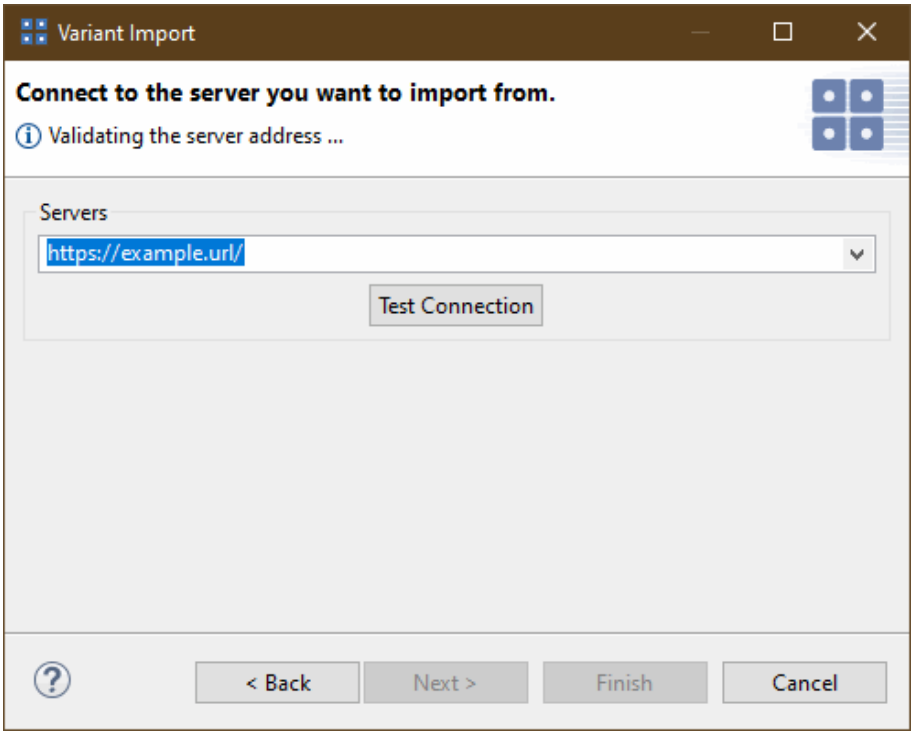
Figure 3. Creating an empty Variant Management project for Polarion document import



The import process is started by selecting the import action either in the context menu of the Project view or with **Import** menu in the **File** menu. Select **Variant Models or Projects** and press Next. On the following page select *'Import Polarion Documents'*.

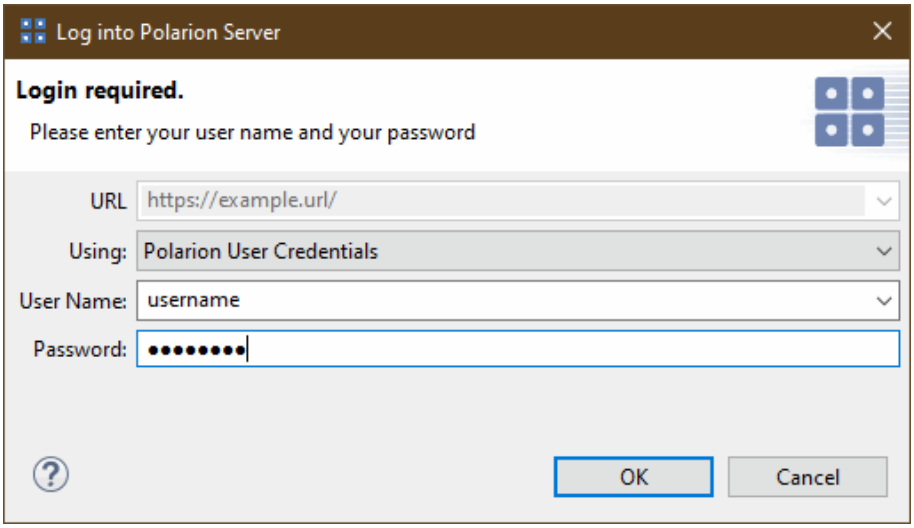
The import wizard appears. With the first page you have to define or select the Polarion server address you want to import the documents from.

Figure 4. The server selection page in the Polarion import wizard



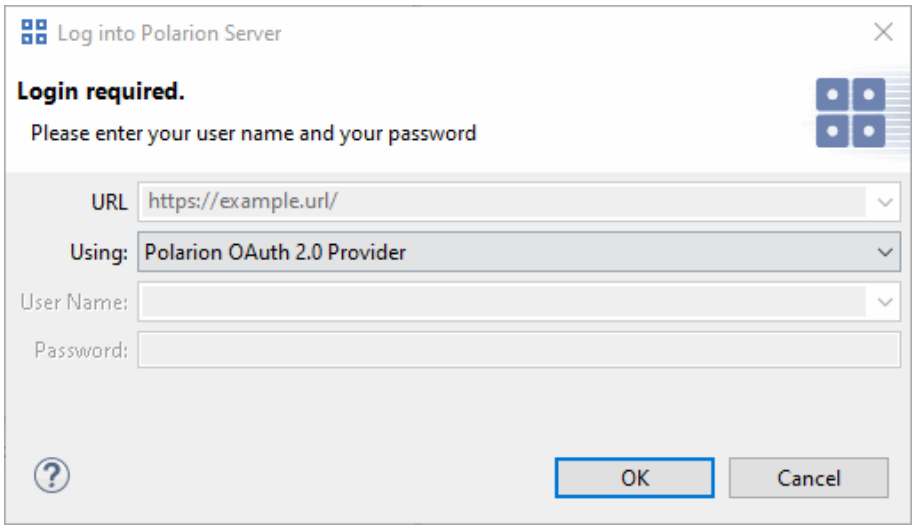
If you are not already authenticated, you can use **Test Connection**. This will open the login dialog that provides multiple options for authentication.

Figure 5. Connecting to the Polarion server using user credentials



1. The Polarion user name and password can be provided with the *Polarion User Credentials'* option.
2. The Polarion OAuth 2.0 Single-Sign-On mechanism is activated with the *'Polarion OAuth 2.0 Provider'* option.
3. The Token for the Polarion User Account Vault method can be provided with the *'Polarion User Account Vault'* option.

Figure 6. Connecting to the Polarion server using Authentication Server (OAuth2)

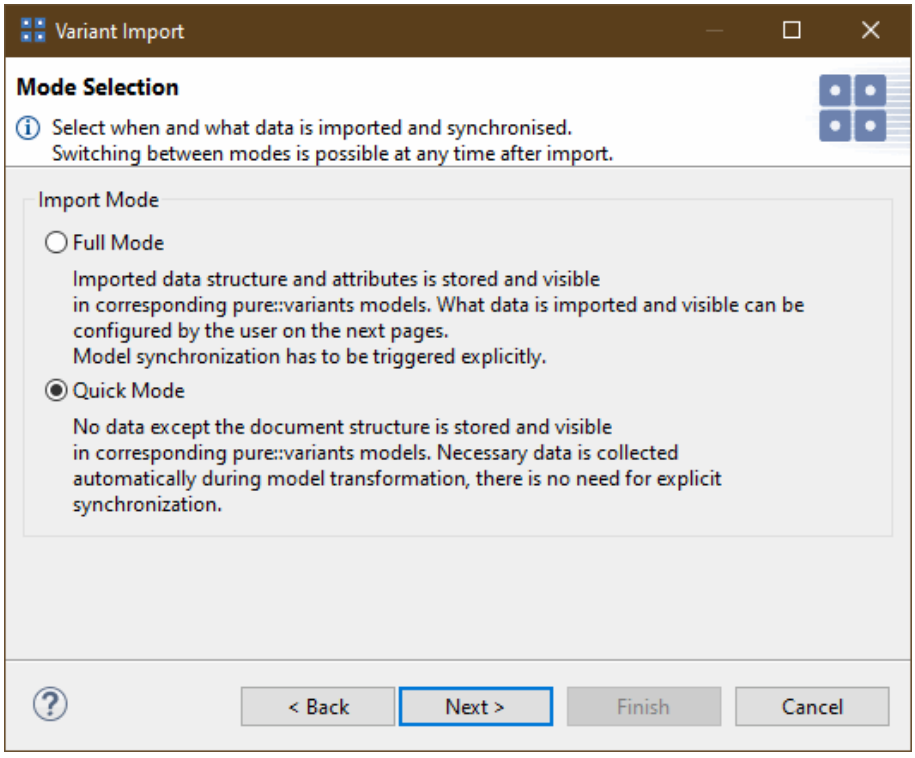


Which login method is working for you depends on the configuration of the Polarion server.

With the next page you can decide whether you want to perform a full import of your Polarion document's variability information (Full Mode), or if you just want to import the document header (Quick Mode). Choosing the latter mode lets the data be synchronized automatically before a transformation, whereas in full mode, the user is responsible to keep the data synchronized, as the existing data is used to transform the variants. Because of the synchronization we recommend to use the Quick Mode.

Using Full Mode, variation points found in documents are represented in the Family Model being created.

Figure 7. The mode selection page in the Polarion import wizard



The data to be imported can be selected by the user on the next page.

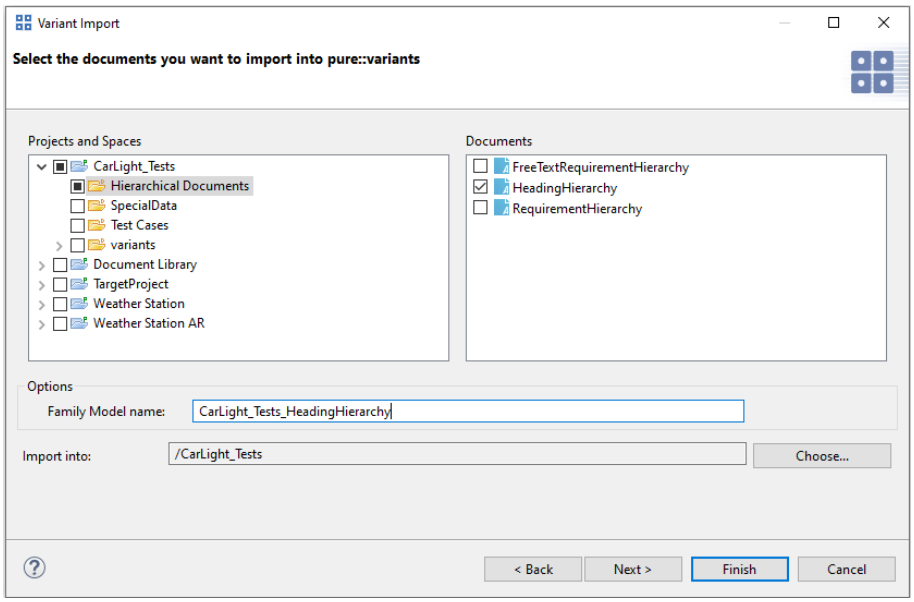
The complete list of projects of the Polarion repository is shown with their respective spaces listed below each project, if they are available. Navigate to the space containing the documents of interest and select the ones to

import using the check boxes on the left side. Multiple spaces from different projects can be selected at once for import. Selecting a check box on the left side of the space marks all documents for import. Selecting individual documents within one space is also possible using the right pane.

Note

Only those informations are presented to which the user has appropriate access rights.

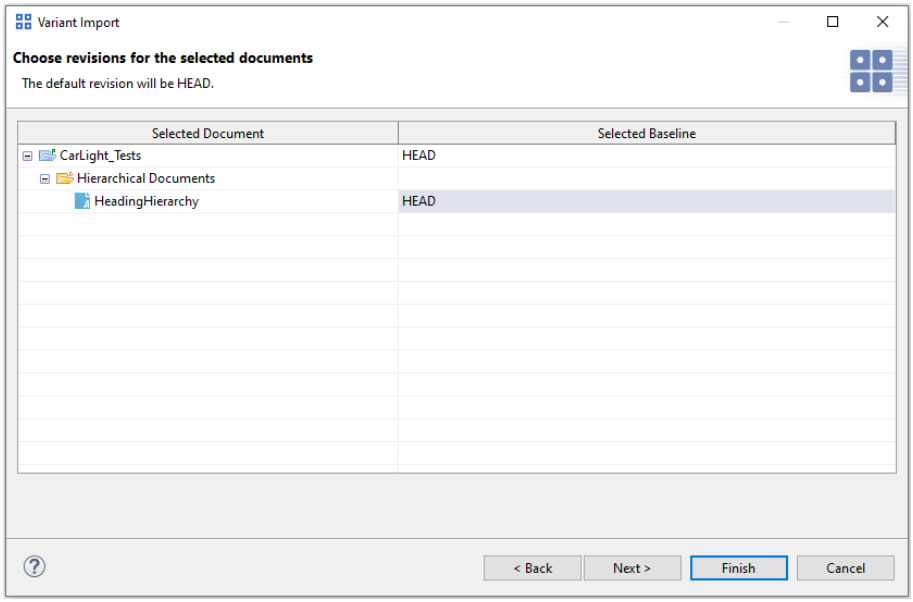
Figure 8. The document selection page in the Polarion import wizard



The imported family model will be stored in the folder defined within the selection menu **'Import Into'** and will be named as defined in the options field **'Family Model name'**.

Using the next page you can select the baseline for each selected document to be used as the source version for the copy- or link transformation. The selection can be performed on project level for relevant baselines, or separately for each of the documents. On project level only baselines are listed which have been created for the whole project. The selection is assisted by a search function that filters baselines to be selected.

Figure 9. Selection of baselines for documents

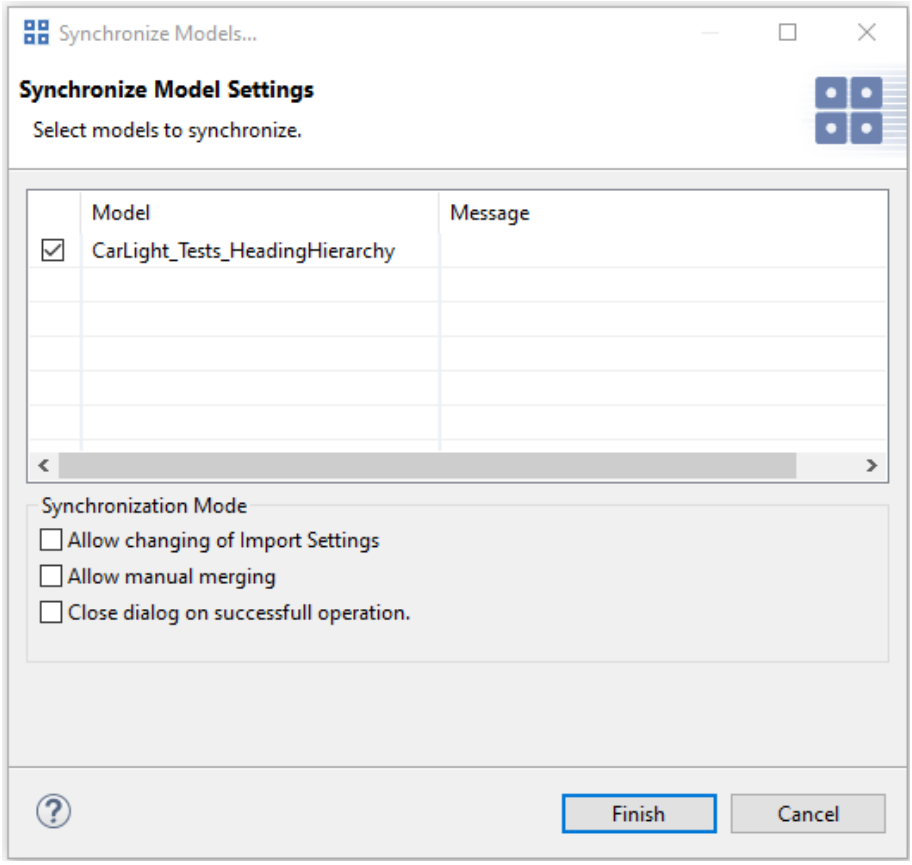


On the following page the Import Rules are shown. On this page you can select sets of Import Rules, which will be used to manipulate the resulting models after import. Import Rule Sets can be used to create specific pure::variants model elements like restrictions or constraints from Polarion artefacts information.

2.5. Updating Family Models from Polarion

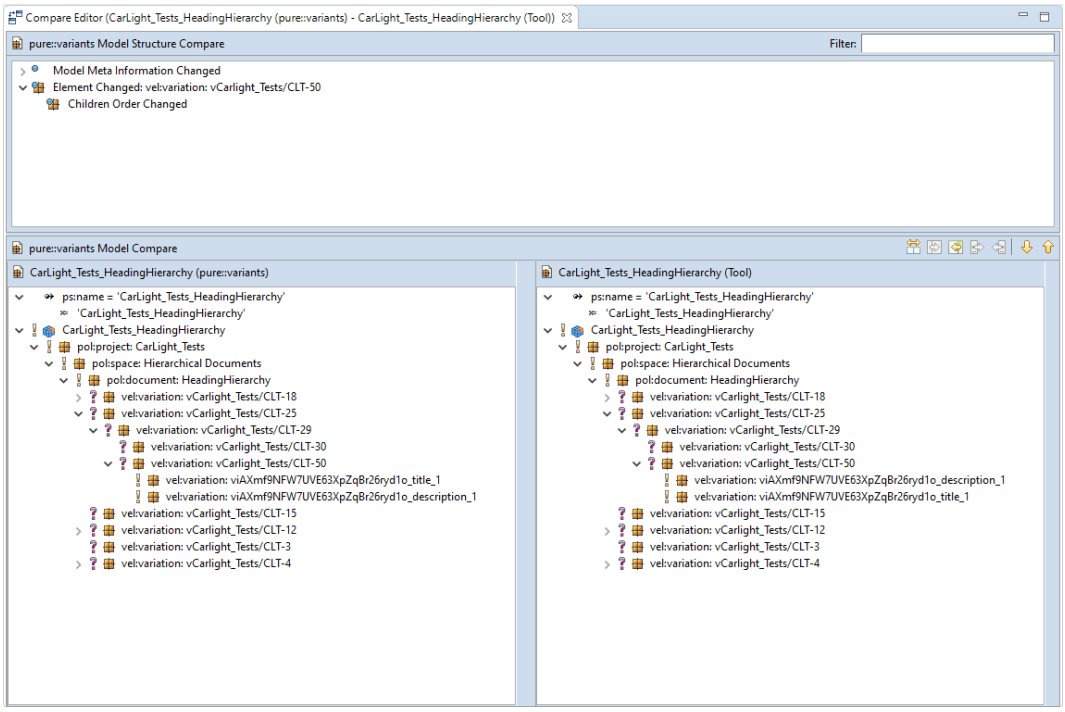
Using the 'Synchronize Models...' action, the mode of the model can be toggled between full- and quick mode, documents can be added or removed from the family model and the baselines of the imported documents can be changed. Additionally, when using Full Mode, it is necessary to update the pure::variants models with information from Polarion whenever relevant changes have been made. To start the update, open the model representing the updated document and press the Synchronize button in the tool bar.

Figure 10. Synchronize model



Pure::variants will connect to Polarion, check if all necessary data is still existent on the server and update the family model with respect to the changes made in the synchronization wizard. After the synchronization the compare editor for pure::variants models is opened.

Figure 11. Synchronize model compare



The compare editor is used throughout pure::variants to compare model versions, but in this case it is used to compare the Polarion data (displayed in the lower right side) with the current pure::variants model (lower left side). All changes are listed as separate items in the upper part of the editor, ordered by the affected elements. Selecting an item in this list highlights the respective change in both models. In the example, the changed restriction values are marked with boxes and connected with their respective counterparts in the other model.

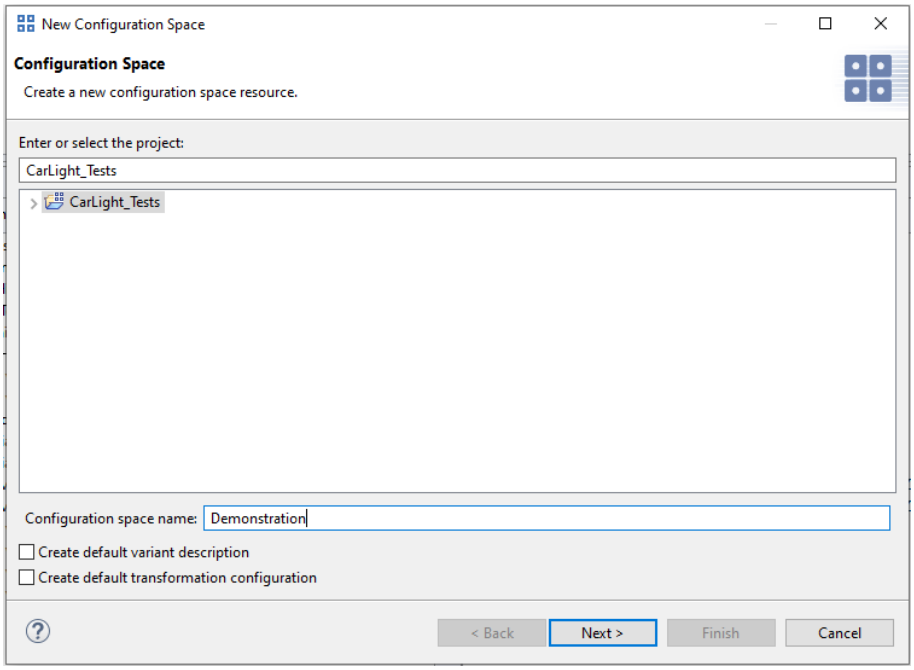
The merge toolbar provides tools to copy single or even all (non-conflicting) changes as a whole from the current model to the out-dated model.

2.6. Defining a Variant

The next step is the definition of the actual variants of interest. Since the variability model usually permits the definition of a very large number of variants, pure::variants keeps track only of those variants which are of interest for the users. Typically this number is much smaller than the number of possible variants.

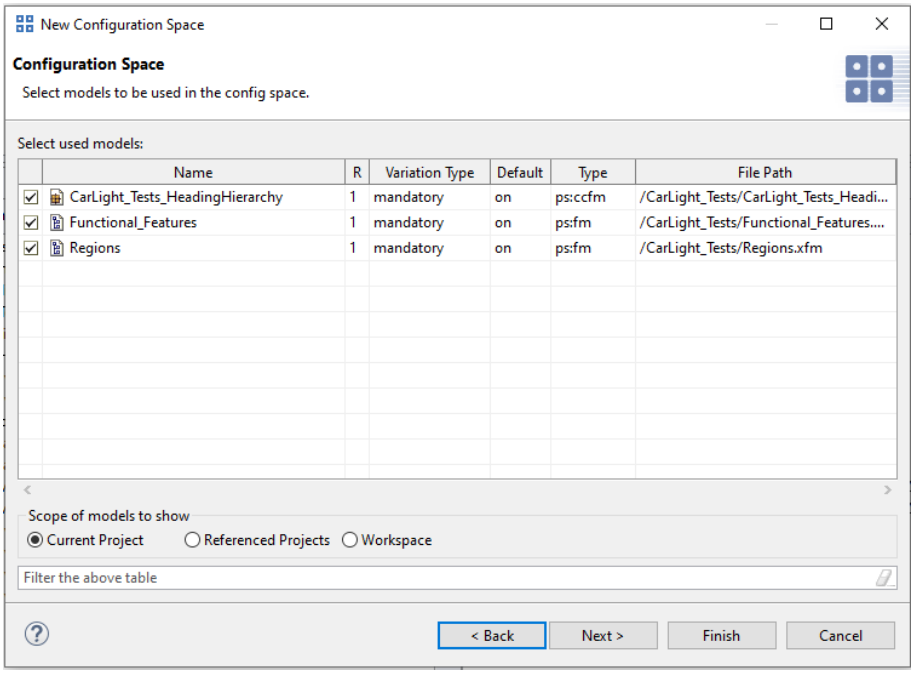
Variants are stored as separate entities called Variant Description Models (VDM). A VDM always belongs to a specific Configuration Space. Thus before defining variants, a Configuration Space has to be created. Select the project containing the imported models in the Variant Projects view and open the context menu. Below the item **New** select **Configuration Space**. A wizard is opened. On the first page (Figure 12, “The Configuration Space Wizard”), enter a name for the Configuration Space. The name has to follow strict rules (no spaces, no special characters). Uncheck the box before **Create standard transformation**, since for pure requirements models the standard transformation does not provide any relevant functionality (see the **pure::variants User Manual** for more information on transformations).

Figure 12. The Configuration Space Wizard



The next page is used to specify which models are to be included in this Configuration Space. Select here all models that represent the spaces and so the documents of interest. In the example below just one Family Model is selected. Now press the **Finish** button.

Figure 13. The Configuration Space wizard model selection page



2.7. Transforming a Variant

Variants stored in a Variant Description Model can be made available in Polarion. The Connector supports following ways of representing variants:

Attribute-Based Variant Representation (Enumeration)

In the attribute-based representation we define a custom field for each selected Polarion document to be added to each work item. This transformation modus adds the name of the variants (as a list, separated by new lines) if the work item is part of the variant. The name of this attribute can be user defined for a transformation, default is 'pvVariants' (see EnumerationField [13]).

Note

This transformation can not be run on documents with baselines other than HEAD. If the baseline deviates from HEAD pure::variants will report an error.

Copy Transformation

Product Line assets (150%) i.e. documents (requirements, test cases, etc) are assigned to a dedicated space or the default space in Polarion.

The Copy Transformation creates a new or uses a currently existing structure of spaces (see). In this target directory the selected documents are copied and modified regarding their variant result. If the given path is already existing it will be emptied before starting the transformation process.

The naming convention for the variant specific space created is by default <VariantName>, e.g. ‚DemoVariant' for 'DemoVariant'. This can also be modified in the transformation configuration (see SpaceName [13]).

Included documents are copied off from the documents with the baseline selected by the user during the import and are reduced to the variant specific subset of work items. All copied work items will receive a new ID.

Link Transformation

As the Copy Transformation the Link Transformation is also creating a new document for each transformed asset. Whereas the Copy Transformation is creating a duplicated document with new IDs for all work items, the Linked Transformation is not duplicating the work items (Requirements, Test Cases, …) but referencing them to the original item.

The naming convention of the variant specific space created is by default also <VariantName> and can also be modified in the transformation configuration.

The naming convention for the variant specific space created is by default <VariantName>, e.g. ‚DemoVariant' for 'DemoVariant'. This can also be modified in the transformation configuration (see SpaceName [13]).

Included documents are branched off from the documents also with the baseline selected by the user.

Text Substitution

Following work item fields are subject to text substitution:

- Title
- Description

Note

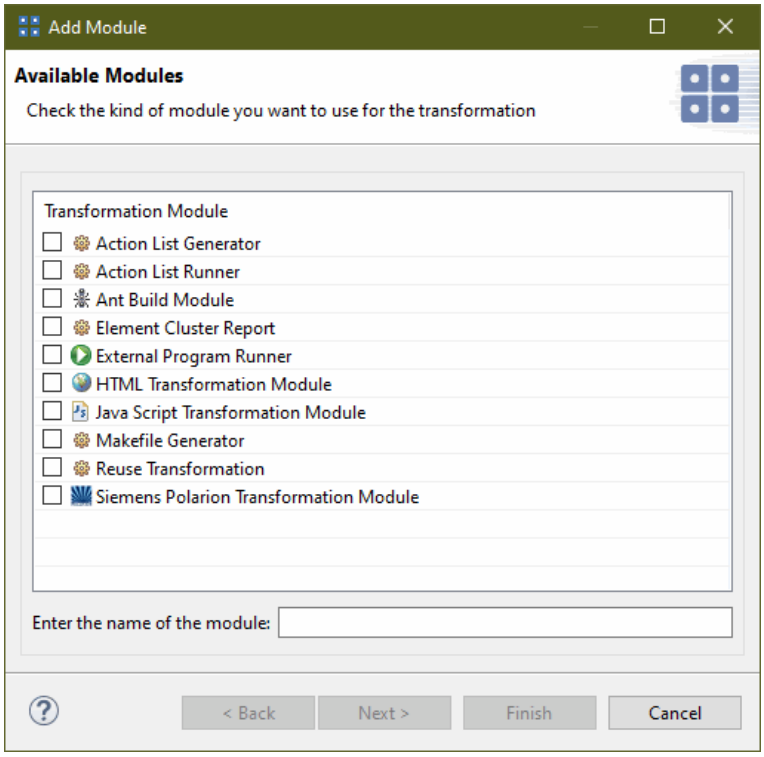
Text substitution is performed only during 'Copy Transformation' (see PerformPartialTextSubstitution [13]).

Preparing a Transformation

To transform a variant, first a Transformation Configuration has to be created. To create a Transformation Configuration click on the arrow next to the **Transformation** button in the tool bar and choose *Open Transformation Config Dialog...*

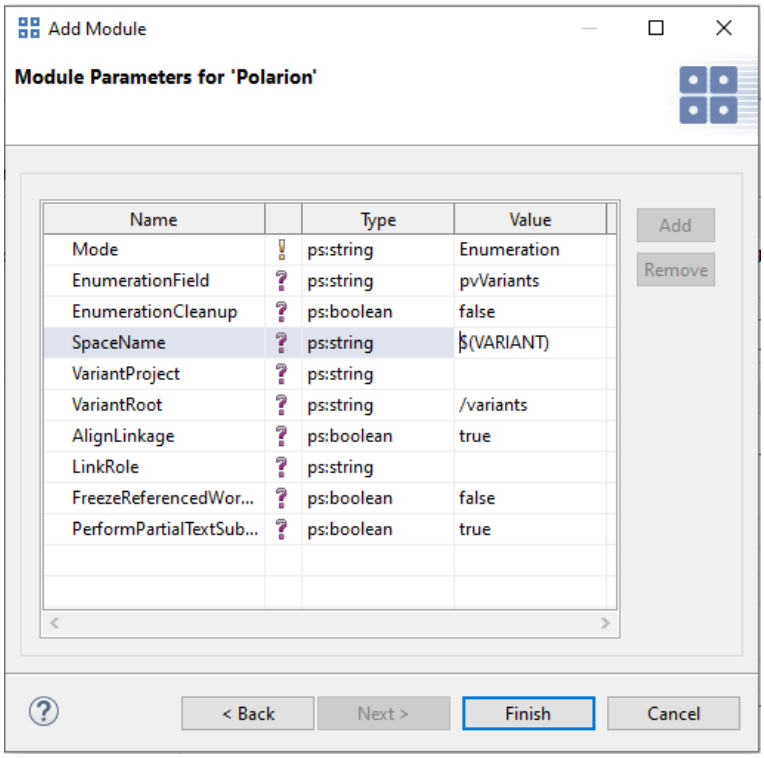
The configuration space property dialog opens and the Transformation Configuration tab is shown. Next step is to add a new Module Configuration , by clicking the marked tool bar item. Now add a new Module to the Module Configuration, using the **Add** button.

Figure 14. Transformation Configuration



From the opened dialog, choose 'Siemens Polarion Transformation Module' and enter a name. The next page shows all parameters. The Modus parameter specifies one of the variant result representations, as described above.

Figure 15. Module Parameter Page



Following parameters need to be defined:

- **Mode**: Defines the transformation mode. The available modes are:

  Enumeration - this option stands for the Attribute-Based Variant Representation

  Copy - this option stands for the Copy Transformation

  Link - this option stands for the Link Transformation

Enumeration Transformation:

- **EnumerationField**: Specifies the name of the work item field to be filled with variant names in enumeration transformation mode. If not changed the standard name ('pvVariants') is used.
- **EnumerationCleanup**: If true is selected, all existing variants in the attributes are removed before exporting the current variant. If false, only the names of the transformed variant will be updated (either removed or added).

Copy Transformation:

- **AlignLinkage**: If set, the links between work items of the original documents are restored to the newly copied ones.
- **LinkRole**: The link role ID which will be used for the creation of the new document. As default no link role is used.
- **PerformPartialTextSubstitution**: If true is selected, the partial text substitution is performed.

Link Transformation:

- **FreezeReferencedWorkItems**: If set the original work items will be freezed before the link transformation is started.

Copy- & Link Transformation:

- **SpaceName**: Specifies the name of the space created by the transformation for each variant. The default space name is a concatenation of the variant name and the date and time the transformation was started.
- **VariantProject**: If set with a proper project ID the transformation will place their outputs in the project defined.
- **VariantRoot**: Defines the root folder in which the variant space with the name defined via the parameter 'Space-Name' is created. By setting the variant root to '/', the variant space is directly created in the respective project.

After finishing the dialogs the transformation can simply be used by clicking on the **Transformation** button in the tool bar and choosing the transformation in the pull down menu.

Note

Updating the previously transformed variant specific documents is currently not supported.

Web Client Integration for transformation

Please consult section **Transformation Help Contents** in the **pure::variants Web Client Manual** for detailed information on how to perform Transformation using Web Client Integration.

3. Using the Integration

In order to facilitate the pure::variants Connector for Polarion variability information needs to be added to the work items. This is performed by adding restrictions to work items and is assisted by the Desktop Hub application

that is provided by the pure::variants client installation or the in-tool integration called pure::variants Integration for Polarion.

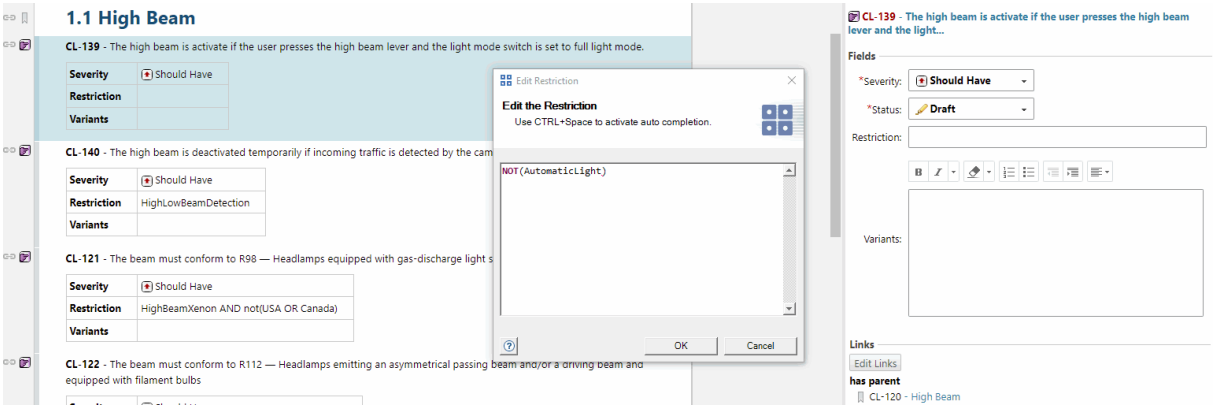
Note

The pure::variants view for documents in Polarion is a read only view. Significant changes like changing the text of a requirement or adding new calculations can only be performed in the editors which are available in polarion itself.

3.1. Adding Variability Information Using the Desktop Hub

The Desktop Hub uses the clipboard to insert information from pure::variants into other applications by pasting them into active fields being edited by the user. The default shortcut to open the pvSCL editor of the DesktopHub is **ctrl + alt + P** and can be configured within the preferences of the DesktopHub in the '**Services**' tab.

Figure 16. Adding restrictions in Polarion using the Desktop Hub



Note

More information on the Desktop Hub can be found in the dedicated *'pure::variants Desktop Hub Manual'*.

3.2. Adding Variability Information Using the pure::variants Widget

Once the Integration has been added to Polarion (see respective chapter *'pure::variants Connectors'* in the **pure::variants Setup Guide**) for the very first time, the General tab view under the Settings page will be shown which basically takes the input from the end-user to select between one of the two available modes, Integration should run into i.e. Desktop Hub mode or Web Hub mode. By default, Desktop Hub mode is being set as the default mode.

Prerequisites for the Desktop Hub Mode

In order to run the Integration in the Desktop Hub mode, a running instance of the Desktop Hub is required in background. While the Desktop Hub instance is running, inside the Integration, go to the General tab view under the Settings page. Notice, that the Desktop Hub is already selected in Connect via drop-down (that's because Desktop Hub is the default mode setting of the Integration) the only thing required is the port number on which the Desktop Hub instance is running, hence, enter the port number inside the given Desktop Hub input type. Afterward, press the OK button in order to save the mode settings. Integration will then redirect to its Main page and start running in the Desktop Hub mode.

For loading a Configuration Space in Desktop Hub Mode: In order to select a Configuration Space please press the Open Configuration Space button from the Integration's menu bar. The Desktop Hub's file selection dialog is

shown to select the desired Configuration Space. Once the Configuration Space is selected, the Integration will immediately show the selected Configuration Space.

Prerequisites for the Web Hub Mode

In order to run the Integration in the Web Hub mode, a running instance of the pure::variants Web Components is required (see chapter **pure::variants Web Components** in the **pure::variants Setup Guide**). While the pure::variants Web Components is running, inside the Integration on the General tab view under Settings page, select the Web Hub value from the Connect via drop-down and then enter the URI to the running instance of the pure::variants Web Components in the given Web Hub input type. Afterwards, press the OK button so to save the mode settings. Integration will then redirect to its Main page and start running in the Web Hub mode.

Define Transformation Related Settings

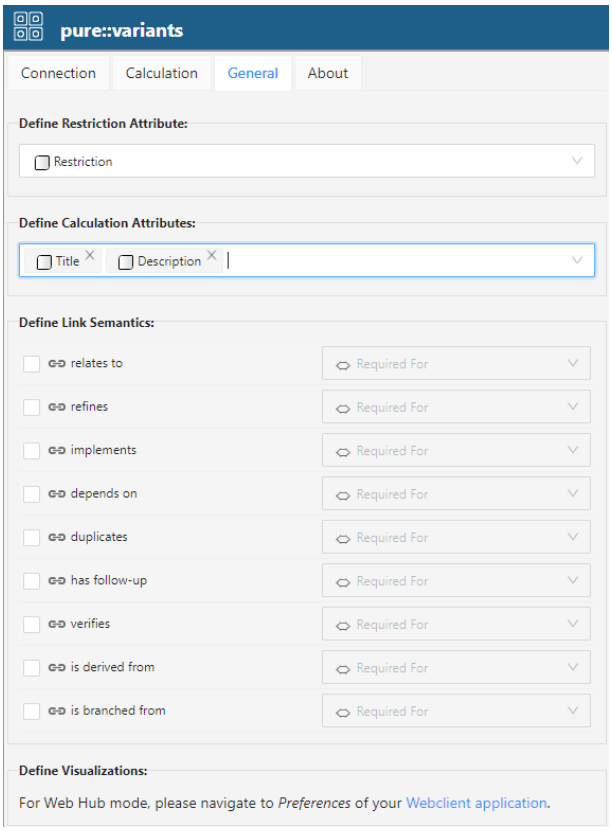
On the Settings page, further transformation relevant settings can be defined for the active document:

- Using the General tab, the restriction field and fields used for partial text substitutions and can be defined. Fields with icon ☐ are standard fields available for all work item types.

  The default value for restrictions is 'pvRestriction'.

  The default value for calculations is 'Title' and 'Description'.

Figure 17. General Tab in pure::variants Widget



- From Calculation Tab, Evaluate Calculations can be Enabled or Disabled from the drop down list of Evaluate Calculations.
- Using the Calculation tab, the text substitutions markers can be defined. The default values are:

  Opening character: '['

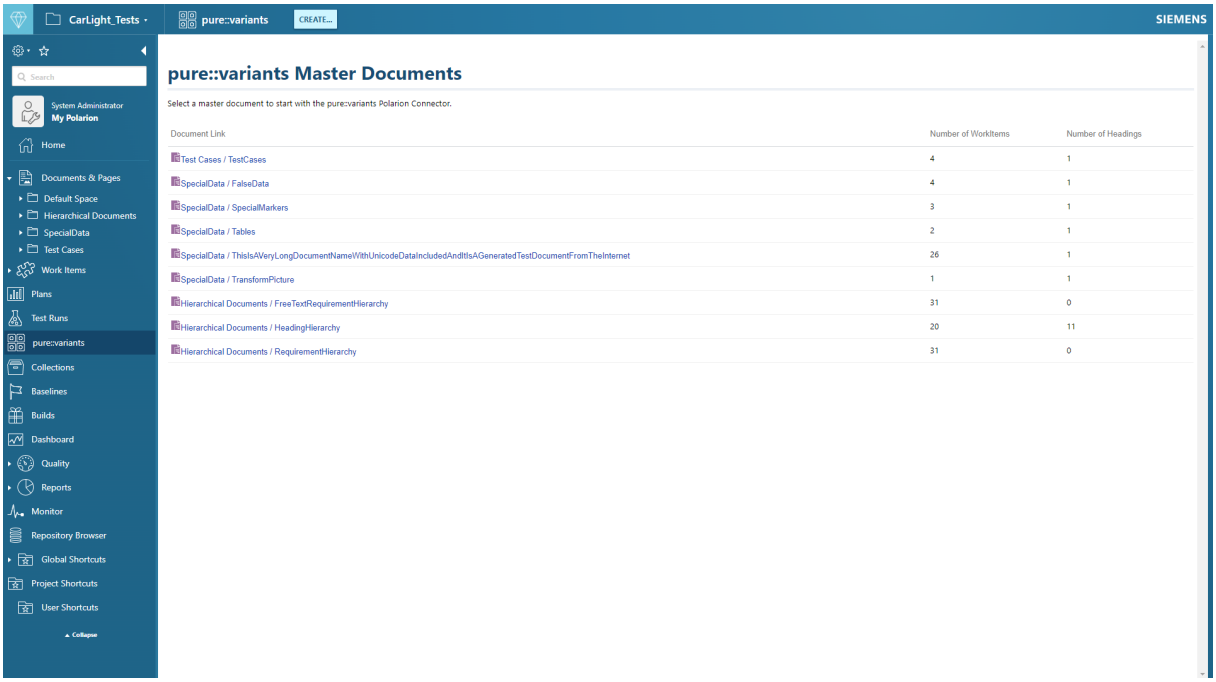
  Closing character: ']'

Escape character: '$'

3.3. Introduction to *Integration* GUI

The *Main* page view of the Integration is shown in Figure 18, “Integration Main page view”

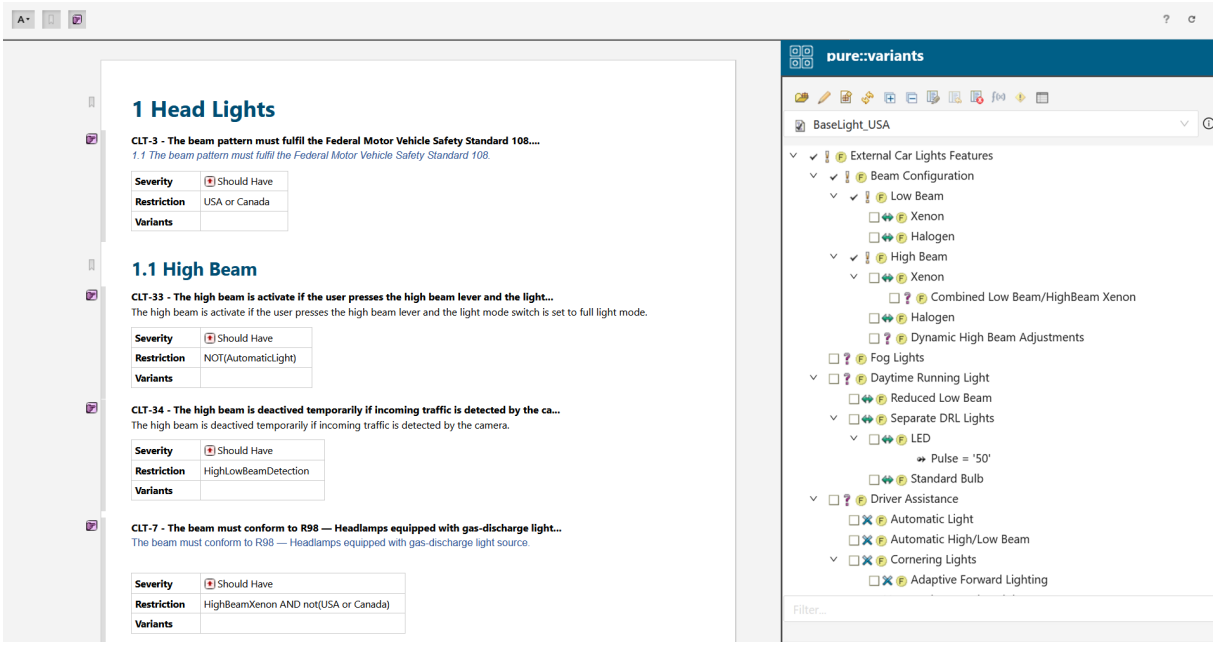
Figure 18. Integration *Main* page view



The landing page can be accessed by a click on the pure::variants topic which is available after following the steps of the setup guide. There you will find all documents available in your project.

By selecting a document the pure::variants view of the document is opened, alongside the integration is shown.

Figure 19. Integration *Document* view



Function of the buttons of the menu bar, from left to right:

1. **Open Config Space** button - click to select the *config space* as explained in the *Desktop Hub* and the *Web Hub* sections.

2. **Model Viewer** button - click to open currently selected Configspace/VDM in the *Model Viewer* web application. (Only visible in the Web Hub mode).

3. **Import/Synchronize** button - click to import trackers as new family models or to update existing ones.

4. **Refresh** button - click to refresh the *Feature/Variant* model tree inside the *Tree-view*.

5. **Expand** button - click to expand the entire tree inside the *Tree-view*.

6. **Collapse** button - click to collapse the tree rendered inside the *Tree-view*.

7. **Show Preview** button - click to enable the preview for visualizing *variability* Information; available in the Document View and supports Wiki format only for fields with the type WikiText.

8. **Reset Preview** button - click to disable the *Preview*.

9. **Calculations** button - click to open the *Calculations* page, so to edit calculations present inside the fields of a work item.

10. **Restriction** button - click to open the *Restriction* page, so to edit the restriction inside the *pvRestriction* fields of a work item.

11. **Error Check** button - click to open the *Error Check* view, to see the errors in pvSCL rules.

12. **Settings** button - click to navigate to the *Settings* page so to configure the *General* settings, *Calculations* specific settings, and, also to see the Integration specific information.

Below the menu bar, there is the **VDM Selector** dropdown that lists all the variant models attached to the selected *configuration space*. Once selecting any vairant model from the dropdown, the model will get render inside the *Tree-view*. The **Tree-view** lists the selected *Feature/Variant* model(s).

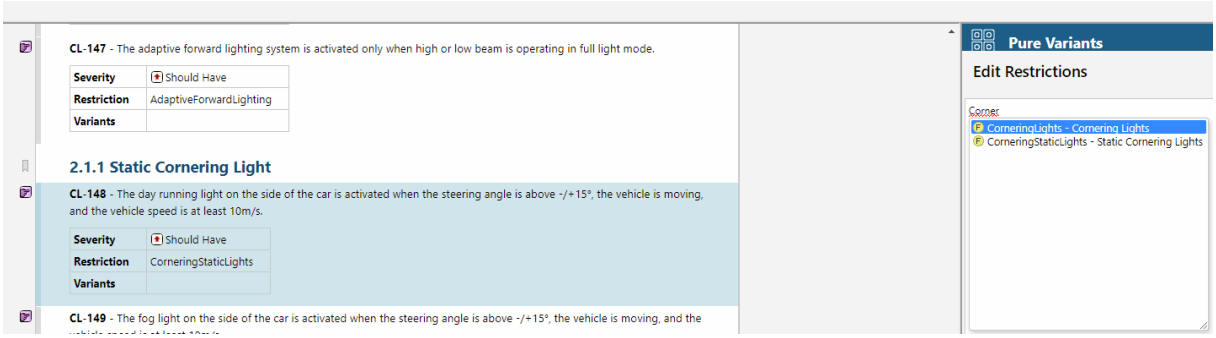
Note

The button for Error Check is disabled in case not the proper DesktopHub or WebHub versions are used (see Section 1.2, “Software Requirements”).

3.4. Working with Restriction Editor

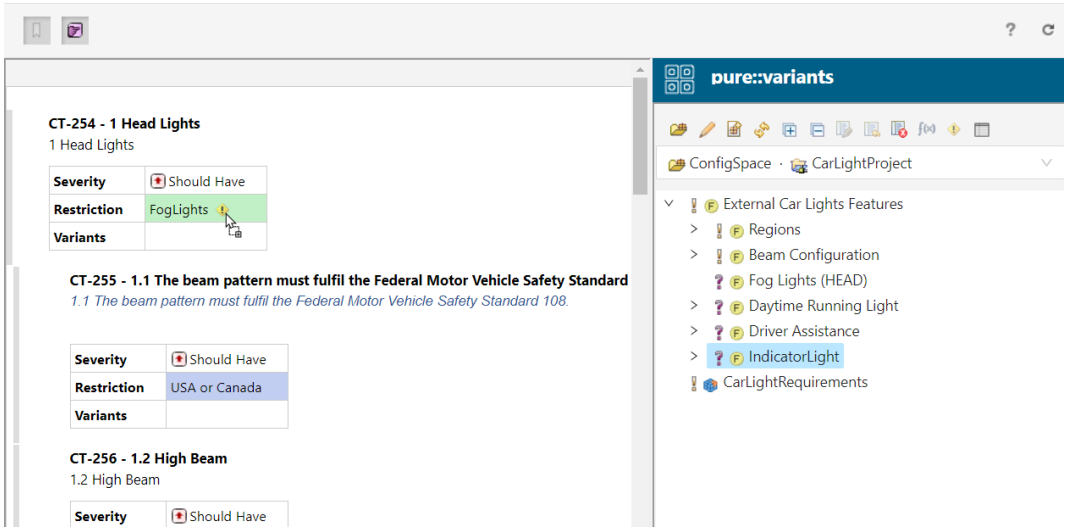
The Restriction Editor can be opened by clicking the **Restrictions** icon. Edit a restriction in Restriction Editor by selecting an item in a document. The Restriction Editor provides the ability of auto-completion proposals and syntax highlighting while editing restrictions for work items (ctrl + space).

Figure 20. Working with the Restriction Editor



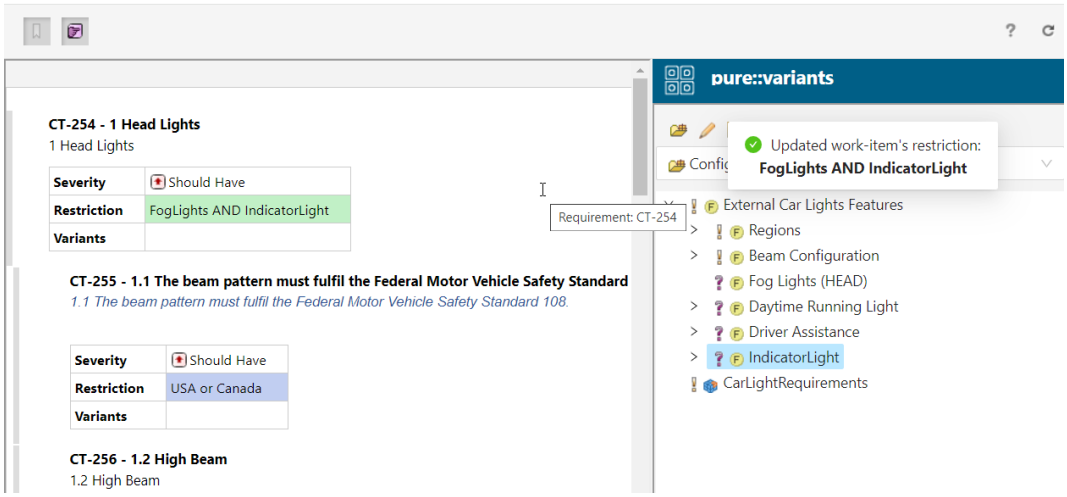
For a more intuitive and simpler way of adding variability to a document item, drag'n'drop capability can be used. Select a feature in the integration's model tree, drag the feature and drop it on the document item's restriction field. By default, the drop of restriction concatenates with the existing pvSCL expression using 'AND' keyword. If the CTRL key is pressed during the drop action, the existing pvSCL expression will be concatenated using 'OR' keyword.

Figure 21. Drag feature to add a restriction



Restriction is added to the document item.

Figure 22. Drop a restriction on tracker item



3.5. Working with Calculations Editor

Calculations Editor can be used to edit the calculations present in fields of a work item. You can open it by clicking the **Calculations** icon. Calculations can be edited by selecting an item in a document and then in the Calculations Editor select the field of an item that contains the calculations markers. After selecting a field, all the calculations in that field appear in the list below. Select a calculation from the list and edit it in the editor below. Calculations Editor provides the ability of auto-completion of proposals and syntax highlighting while editing the calculations.

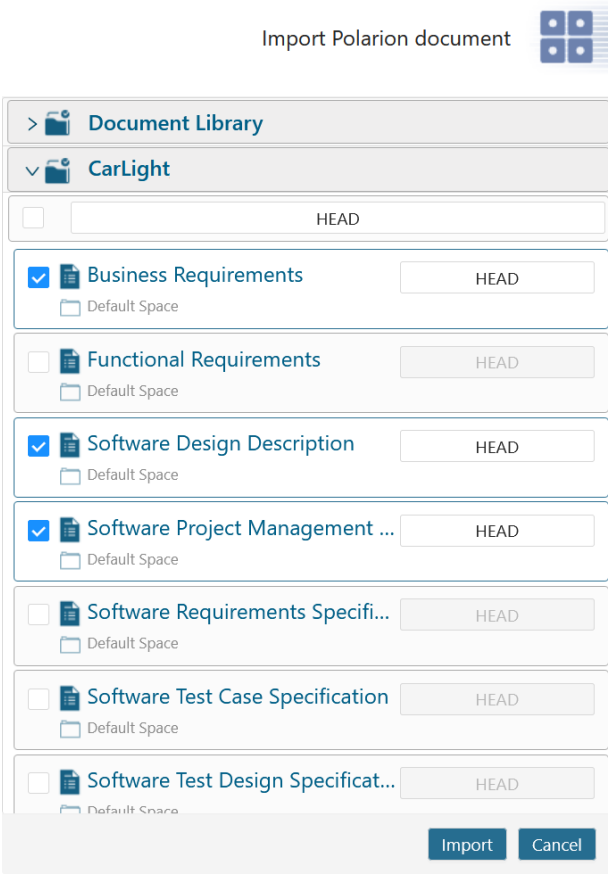
Figure 23. Calculations Editor of the Integration

3.6. Importing and Updating Models using the Web Client

To **import** a document, first open the document in Polarion. Then using the integration open a configuration space of the target pure::variants project where the corresponding family model shall be created after import.

When pressing the **Import** button, all documents and projects are listed in a dialog (Figure 24, “Import Dialog of Integration”), with the opened document preselected. Here, select the documents to be included in the import along with the baseline information. After confirming the selection, the Web Client opens where a wizard guides through the rest of the import process. The import automatically adds the created model to the opened configuration space.

Figure 24. Import Dialog of Integration



To **update** a Family Model, select the root element of the model to be updated. The icon for the button toggles to update mode.

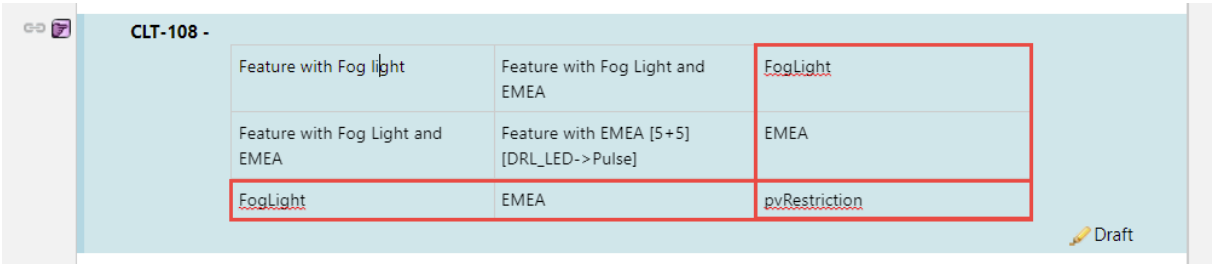
When pressing the **Update** button, all documents and projects are listed in the dialog, where the documents that were previously imported are preselected. After confirming the selection, the Web Client opens where a wizard guides through the rest of the update process.

Please consult section **Import Assets as Family Models Using the WebClient** in the **pure::variants Web Client Manual** for detailed information on how to perform the rest of the steps.

3.7. Variability in HTML Tables

To add variability to HTML tables there needs to be a explicit row and column to hold the variability information. This column and line can be added anywhere in the table, but needs to hold the specified keyword, that is also used to indicate a restriction on e.g. a requirement. By default, this keyword is pvRestriction.

Figure 25. Example HTML Table



As depicted in the example table, the highlighted pvRestriction cells describe the variability for their respective row and column. The variability information of a specific cell in the table is the AND product of the restriction value of its row and its column. In the example the whole column and row 'FogLight' will only be part of the variant, if the feature FogLight is selected.

The variability information cells (e.g. the marked, red pvRestriction cells in the example) are removed from the variant by default and are kept in case of a partial transformation.

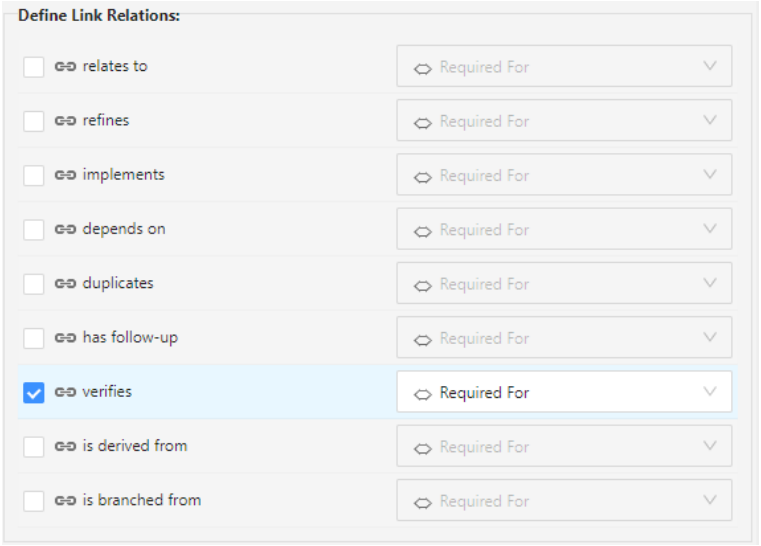
Calculations will also be computed, as they are for work items, if they are marked with the respective open and close characters.

3.8. Link propagation of Polarion link types

The pure::variants connector for Polarion is able to evaluate links between work items in Polarion if both the source- and target document are part of the family model import and the Polarion link type is mapped to a pure::variants relation. The mapping must be stored in the document where the links are outgoing (source document). If the previous conditions are fulfilled the link is automatically evaluated into a pure::variants relation stored in the imported family model. For instance, if there is a link of type `verifies` between a test and a requirement which is mapped to the pure::variants relation `ps:requiredFor` or `ps:requiredForAll`. The test case is automatically contained or removed according to the existence of the requirement in the transformed variant even if the test case does not have a restriction.

The mapping between Polarion link types and the desired pure::variants relation type is done directly in the pure::variants integration in Polarion. To open the mapping table open the `Preferences` and navigate to the `Gen\eral` tab. Enable the check box for the Polarion link types you want to map and select the pure::variants relation type in the drop down menu. When all link types are mapped, save the current settings by pressing the `Okay` button.

Figure 26. Link propagation table in pure::variants widget in Polarion



4. Permission management in Polarion

Work items which has become a final state should be protected against any modifications. These work items could be Requirements, System-Requirements, Testcases, etc. If they are reviewed, accepted, approved and reached their final status they should not be modified anymore. This is also true for the pure::system's Polarion connector when it comes to changing features, restrictions, calculations etc.

Since Polarion provides different protection mechanisms, as an administrator of Polarion you should keep in mind that protecting work items for being changed with the pure::system's Polarion connector is only possible by the Polarion permission management. Setting the work item attributes (either Polarion fields or custom fields) to read-only for specific states is not enough in terms of the connector. The connector is implemented with the provided APIs by Polarion. Unfortunately the APIs cannot ask for modify permissions of read-only protected

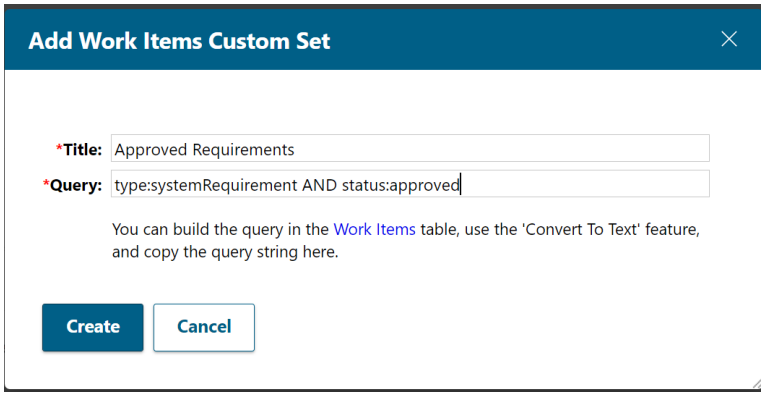
attributes. The only way to check protected fields is by having them configured with the Polarion permissions in the administration. If you want to protect work items against changes in a specific state one needs to configure a customset as in the following example.

4.1. Configuring permissions in order to restrict access

This example shall explain how to reject changes for the fields title, description and restriction if the work item is approved.

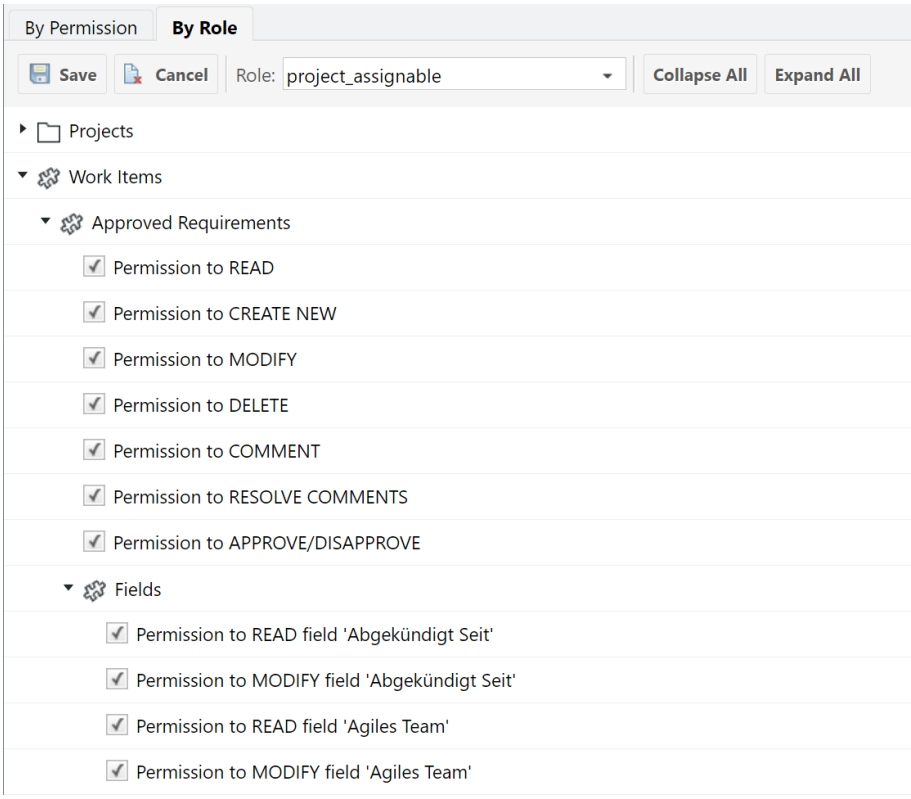
First go to the **Administration** and select the **User Management**. There you can select **Permissions** and add a new **customset**.

Figure 27. Add new customset for all approved requirements



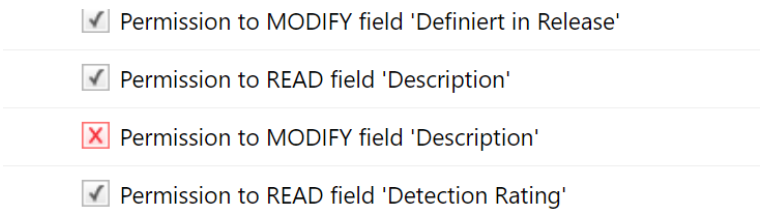
Then switch to the role settings tab and select the project role to configure the permissions

Figure 28. Polarion role settings configuration



Open the fields section and uncheck the modify flag for all fields which should be protected

Figure 29. Field permission configuration in Polarion



Then the connector checks if the field is protected when adding/changing restrictions and calculations to polarion and shows an adequate error message with the id of the work item in case of a permission violation.

5. Migration from Polarion Variants

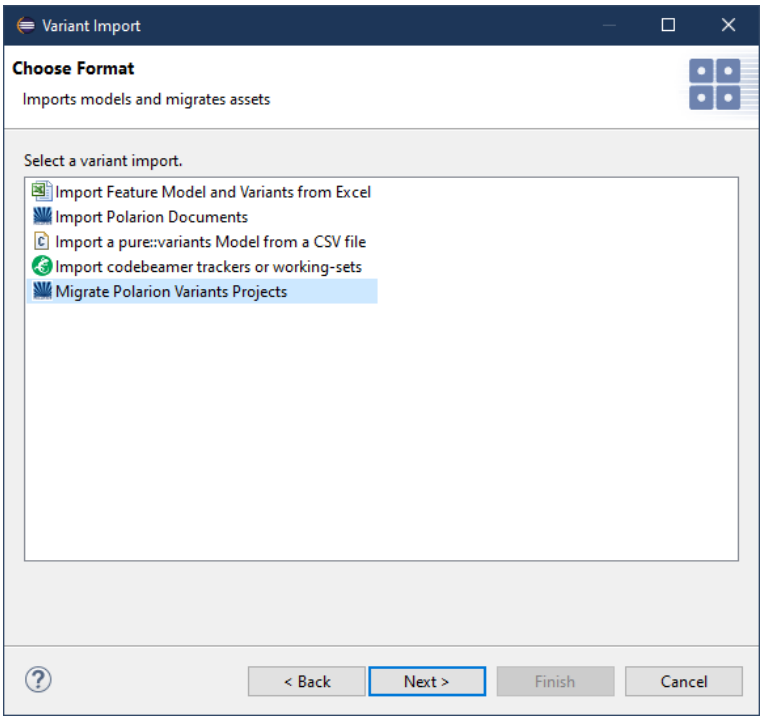
If a user wants to migrate the existing Polarion Variants project data into corresponding pure::variants project data formats, a process is offered by the pure::variants Connector for Polarion.

For migration, there are four actions which will be performed:

1. The feature model will be imported to pure::variants.
2. A new configuration space will be created and the imported feature model will be referenced. Additionally all variants from the Polarion server will be imported and placed in the created configuration space.
3. The Polarion Variants data will be merged in order that the pure::variants connector is able to interpret the existing data. The merge will change the feature IDs to pure::variants valid ones and the type of the custom fields which will store data of variability.
4. This step will remove the old Polarion Variants feature model from the Polarion project on the Polarion server.

To trigger the migration process please open the migration wizard with the name '**Migrate Polarion Variants Projects**' in the Variant Management Import section (see Figure 30, “Migrate Polarion Variants Project”).

Figure 30. Migrate Polarion Variants Project



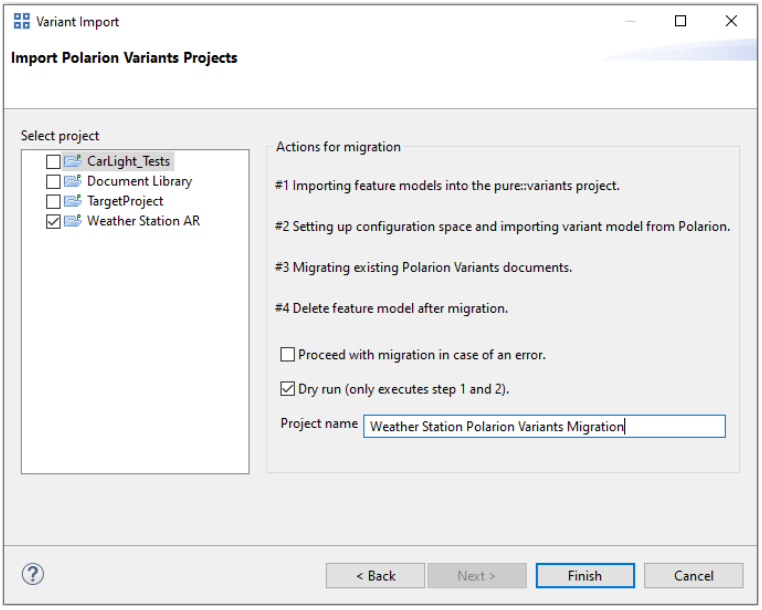
By pressing next you will get to the same Polarion server selection page introduced in Section 2.4, “Creating the initial model(s)” ,Figure 4, “The server selection page in the Polarion import wizard”.

Once you have authenticated with Polarion server, you are able to see all the Polarion projects available for the user on this server. You can manually select the Polarion Variants project you would like to migrate and get managed by pure::variants. The newly created project will be specified by the entered project name (see Figure 31, “Project Selection Page during Migration”).

The option '**Proceed with migration in case of an error**' is to force proceed the migration process even though an unexpected error has occurred.

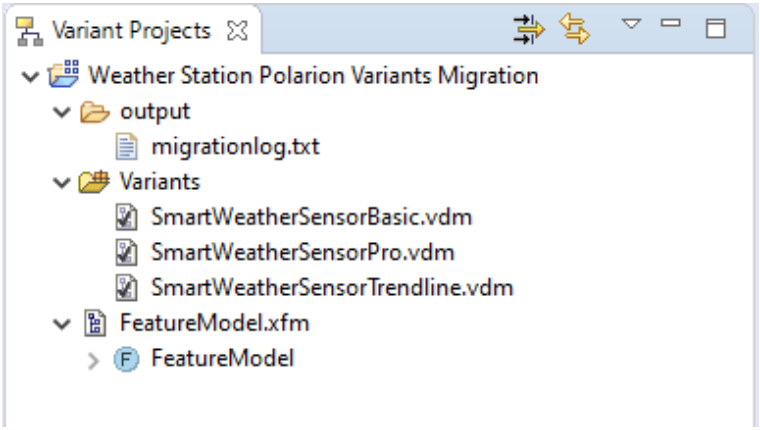
To test the importing part of the migration process there is a '**Dry run**' option which allows you only to execute the first and second action which are importing models from Polarion Variants to pure::variants meaning there will be no modification of existing Polarion data.

Figure 31. Project Selection Page during Migration



Once you press the finish button, you will be able to see your Polarion project in the '**Variant Projects**' area (see Figure 32, “Polarion project after migration”).

Figure 32. Polarion project after migration



Note

The created migration log will contain the log details for the steps #3 and #4 mentioned in Section 5, “Migration from Polarion Variants”

6. Known limitations of the Polarion Connector

In this section those issues are listed that are already known to cause a limitation of the functionality and need to be addressed in future releases of Polarion.

- Test steps and other table fields are currently not supported for work items
- The pure::variants view for documents in Polarion is a read only view. Significant changes like changing the text of a requirement or adding new calculations can only be performed in the editors which are available in polarion itself.

7. Troubleshoot

7.1. Connection Issues - Timeouts, Interrupts, etc.

Since pure::variants Connector for Polarion is heavily relying on the communication to the Polarion server via network communication, which may work out better or worse depending on the company's infrastructure setup. If there are infrastructural problems, like slow network connectivity or slow server deployments, you may overcome these issues, by defining the following parameters:

- Extend connection timeout (in milliseconds, default: 120000): *`PV_HTTP_CLIENT_CONNECTION_TIMEOUT=120000`* (equal to 2 minutes)
- Extend response timeout (in milliseconds, default: 120000): *`PV_HTTP_CLIENT_READ_TIMEOUT=120000`* (equal to 2 minutes)

These parameters can be defined in the *eclipse.ini* file of your pure::variants installation (directly after line **-vmargs**), as follows:

```
...
-vmargs
-DPV_HTTP_CLIENT_CONNECTION_TIMEOUT=240000
-DPV_HTTP_CLIENT_READ_TIMEOUT=240000
...
```

Note

Please ensure to prefix the parameter names with **-D**.