pure::variants - Connector for Codebeamer Manual

Parametric Technology GmbH

Version 7.0.0.685 for pure::variants 7.0

2025

Table of Contents

1. Introduction

The pure::variants connector for Codebeamer enables Codebeamer users to manage requirements as well as test artefacts variability using pure::variants. By coupling pure::variants' and Codebeamer's, knowledge about variability and variants can be formalized, shared, and automatically evaluated. This enables getting answers to questions about valid combinations of requirements and test artefacts in product variants quickly; permits easy monitoring of planned and released product variants at the requirements and test artefact level, and also permits very efficient

production of variant-specific requirements documents, test sets and test cases out of the requirements and test artefact repository respectively.

1.1. About this manual

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

1.2. Software Requirements

The following software is required by the pure::variants Connector for Codebeamer:

- PTC Codebeamer 22.10-LTS - 22.10-SP9, 2.0.0.0 - 2.0.0.7, 2.1.0.0 - 2.1.0.5, 2.2.0.0 - 2.2.0.2, 2.2.1.0 or 3.0.0.0 (when Codebeamer is deployed as single-node installation), or PTC Codebeamer 2.1.0.2 - 2.1.0.5, 2.2.0.0 - 2.2.0.2, 2.2.1.0 or 3.0.0.0 (when Codebeamer is deployed as cluster-architecture installation). Compatibility with other Codebeamer releases is not guaranteed.
- pure::variants server component for Codebeamer in the same version as the connector. The connector requires the deployment of pure::variants-specific components on the Codebeamer server.
- pure::variants Desktop Hub or Web Hub also in the same version as the connector itself:

  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, while the installation of the Web Hub is described in the pure::variants Setup Guide.

The pure::variants Connector for Codebeamer 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 Codebeamer connector, deployment of the components, as well as description of how to configure single sign-on are described in the **pure::variants Setup Guide**

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 Codebeamer Project

In order to get the variability information from Codebeamer items as well as to assign items to variants in Codebeamer, the Codebeamer trackers need to be prepared initially. To make the variability information available to pure::variants, an attribute has to be set for every Codebeamer tracker that shall be processed with respect to its variability.

To set this attribute for each selected tracker, go to Codebeamer and use the Configure option of the tracker. Here, select the Fields page and add a new custom field named 'pvRestriction' of the type 'Text'. Furthermore, to prepare the Enum transformation to store variability information in Codebeamer, a further custom field is needed, named 'pvVariants' of the type 'Text'.

For Test Steps within Test Cases, in the Table Definition, the custom fields 'pvRestrictionTestSteps' and 'pvVariantsTestSteps' of the type 'Text' need to be added, respectively.

2.3. Authentication

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

There are two authentication mechanisms supported:

1. Basic authentication using Codebeamer credentials

2. OpenID Connect (for Single-Sign-On)

During the use of the connector, for both mechanisms, the user will be prompted with a login dialog, which expects the user's credentials. In the case of single-sign-on a browser-based login dialog will be shown that is provided by the configured authorization server.

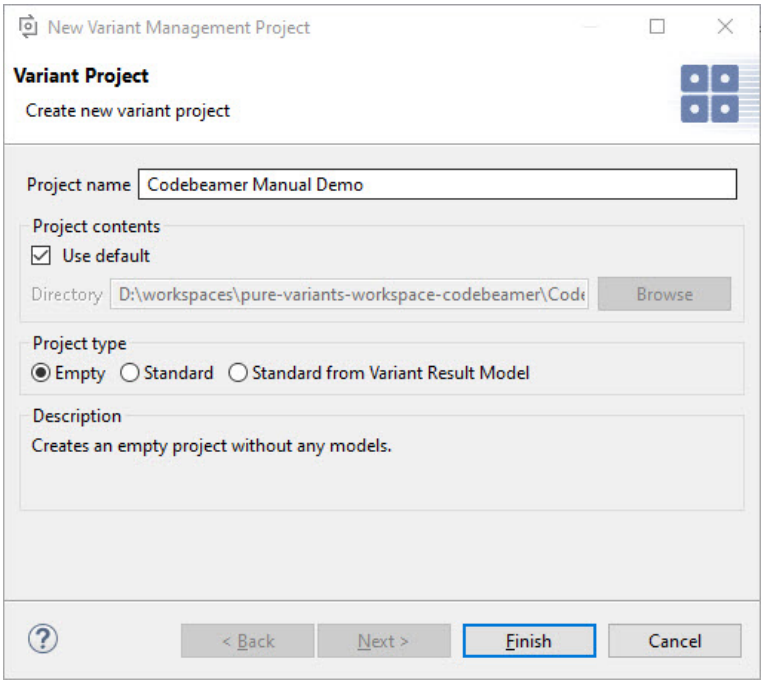
2.4. Creating the Initial Model(s)

The first step is always to create the corresponding family model for each relevant working set containing selected Codebeamer trackers.These initial family models serve as starting points for using existing variability information. The import procedure has to be executed only once for each Codebeamer working set but can be updated afterwards. Each tracker is represented by one model node element in the pure::variants family model that is 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, where the imported models will be stored. 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 the project type (see Figure 1, “Creating an empty Variant Management project for Codebeamer tracker import”)

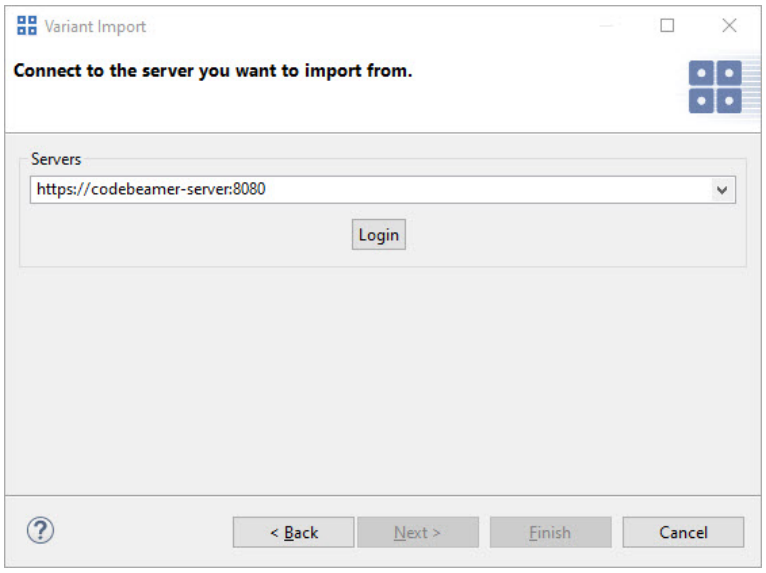
Figure 1. Creating an empty Variant Management project for Codebeamer tracker import



Import 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 Codebeamer trackers or working-sets*'.

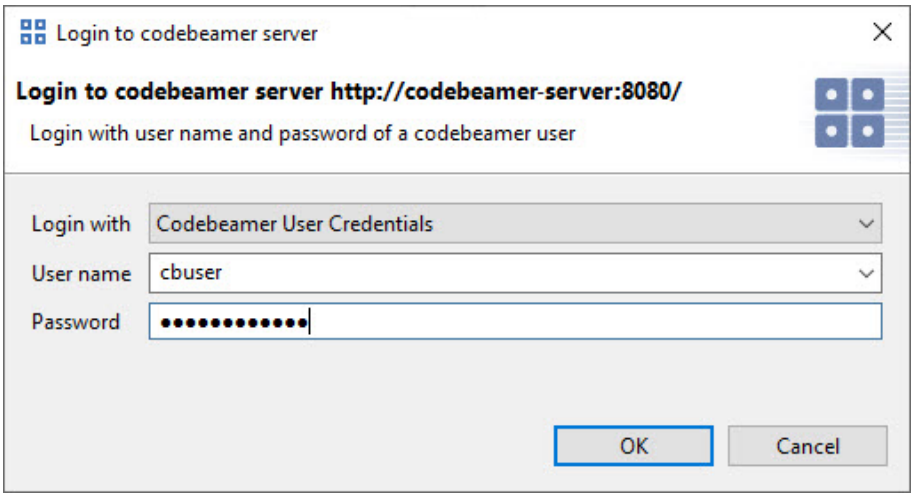
The import wizard appears. In the first page, you have to define or select the Codebeamer server address you want to import the trackers from.

Figure 2. The server selection page in the Codebeamer import wizard



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

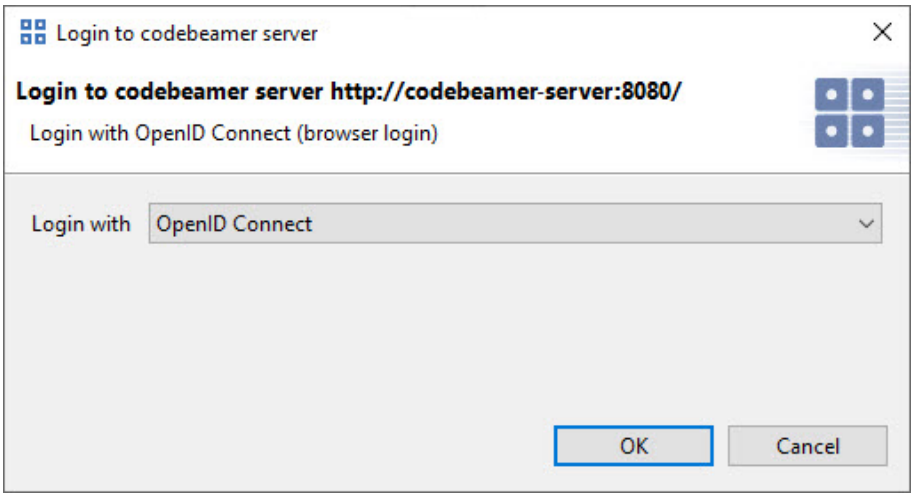
Figure 3. Connecting to the Codebeamer server using user credentials



1. The Codebeamer user name and password can be provided with the *'Codebeamer User Credentials'* option.

2. OpenID Connect authentication can be used by selecting the *'OpenID Connect'* option.

Figure 4. Connecting to the Codebeamer server using OpenID Connect

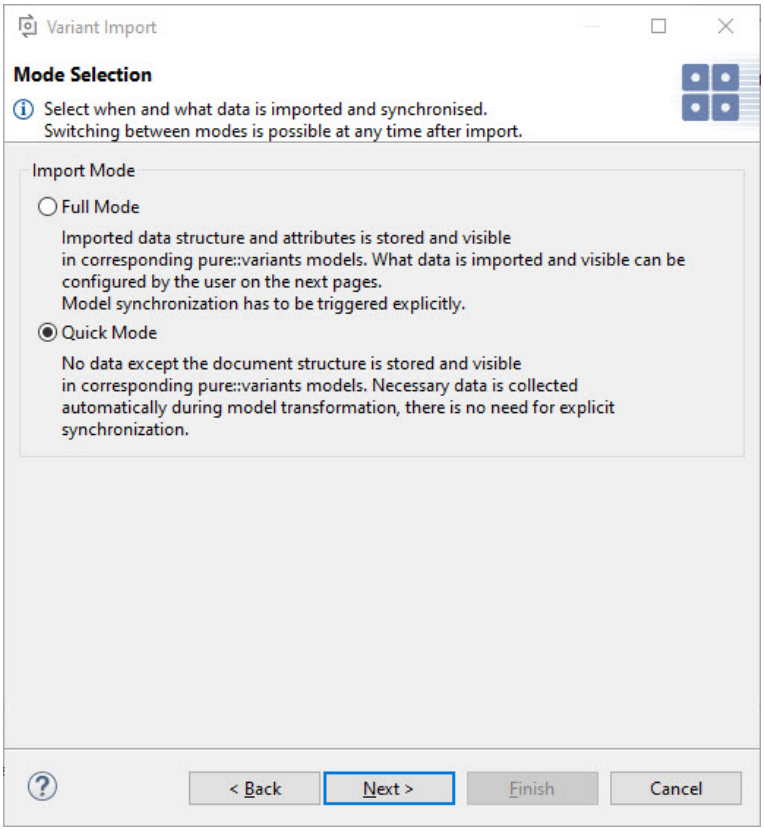


Which login method is working for you depends on the configuration of the Codebeamer server.

In the next page, you can decide whether you want to perform a full import of your Codebeamer tracker's variability information (Full Mode) or if you just want to import the module header (Quick Mode). In the latter case, the data is automatically synchronized before a transformation, whereas in full mode, the user is responsible for keeping the data synchronized, as the existing data is used to transform the variants.

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

Figure 5. The mode selection page in the Codebeamer import wizard



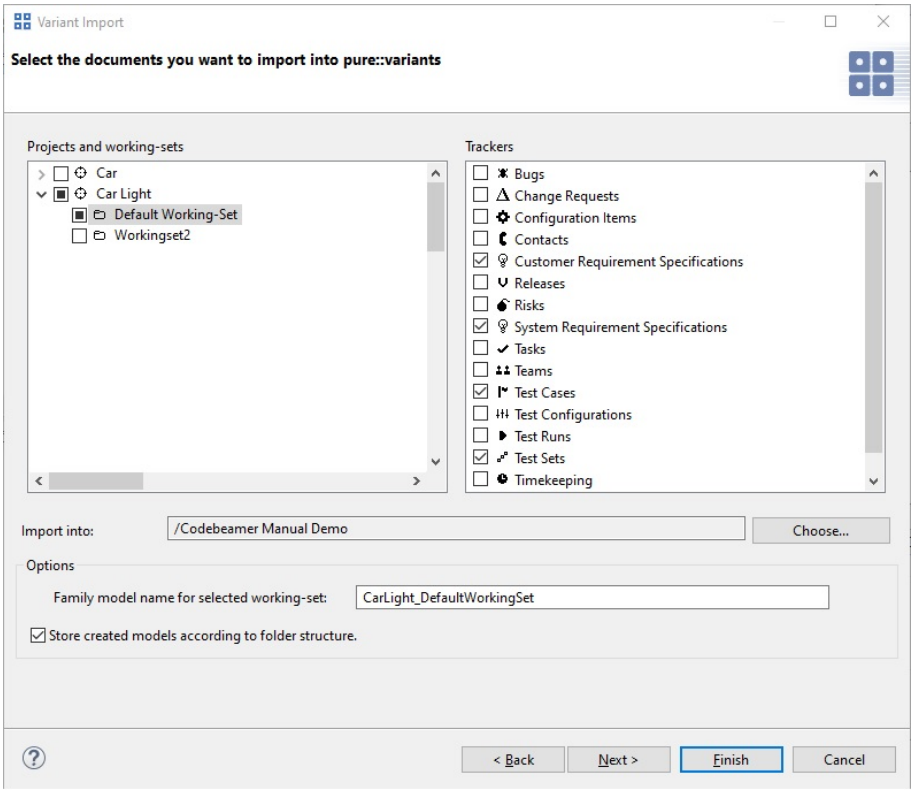
Which data will be imported can be configured by the user on the next page.

The complete list of projects of the Codebeamer repository is shown, as well as the working-sets are listed below each project that are available. Navigate to the Working-Set containing the trackers of interest and select the check

boxes on the left side. Multiple Working-Sets from different projects can be selected at once for import. Selecting a check box on the left side for the Working-Set marks all trackers for import. Selection for individual trackers within one Working-Set is also possible using the right pane.

Note: Please note, only information is presented to which the user has appropriate access rights.

Figure 6. The tracker selection page in the Codebeamer import wizard



Make sure that the import target location given next to "Import into" is correct. The location can be changed by using the Choose button. Selecting the option "Store created models according to folder structure," the import process creates folders for the project and the Working-sets respectively, in pure::variants for the family models.

The family models created are named by default according to the <Project>_<Working-Set> scheme, but this can be modified using the edit box.

Note: Although trackers of all types can be imported, only trackers of certain types will be considered during transformation (for a list of types, see chapter *Transforming a Variant* of this manual). Also, variation points are only considered during import in these trackers.

In the next page, baselines can be selected for each tracker that will be used as the source version for Working Set Transformation. In addition to selecting baselines, trackers can be made shared in the new working-sets created by activating shared option and Variability mode can be configured for each tracker.

- The selection can be performed on Working Set level for relevant baselines or separately for each of the trackers. On Working Set level those baselines are listed that are common for each of the trackers. The selection is assisted by a search function that filters the baselines to be selected.
- Alternatively, it can be defined for a tracker to be included as shared in the Working Set that is created by the Working Set Transformation.

  Note: The shared state of trackers can only be configured up until Codebeamer version 2.2.x.x In Codebeamer versions 3.0.0.0 and higher, it is not possible to make trackers shared in new working-sets created by the working-set transformation. Therefore, in the import dialog, only the shared state of trackers will be displayed and is read-only.

- Further, it is possible to ignore the inclusion of variability information in the imported family model (in Full Mode) or during transformation (in Quick Mode) for a specific tracker by selecting Variability mode 'Ignore'. When Variability mode is set to 'Process', variability information is processed during import but not applied in transformation. When Variability mode is set to 'Transform', variability information is processed during import and is applied in transformation. By default, Variability mode is set to 'Transform'. More details on how Variability mode affects the transformation can be found in section Transforming a Variant.

Note: The shared state of trackers that are by definition shared cannot be changed and is displayed as read-only in the dialog.

Figure 7. Select Tracker Baselines

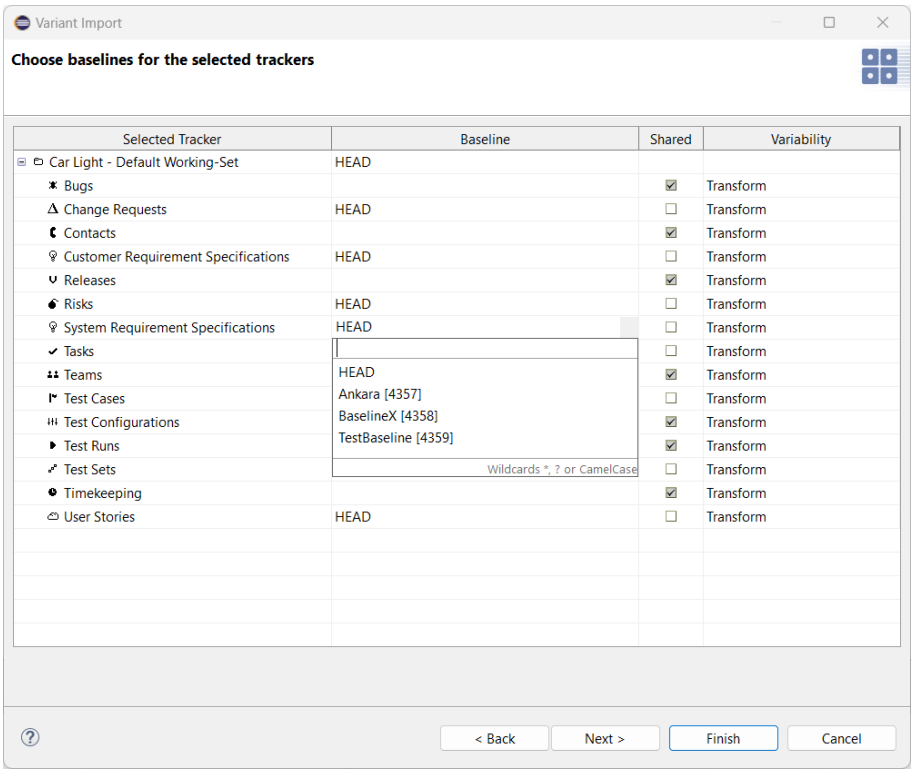
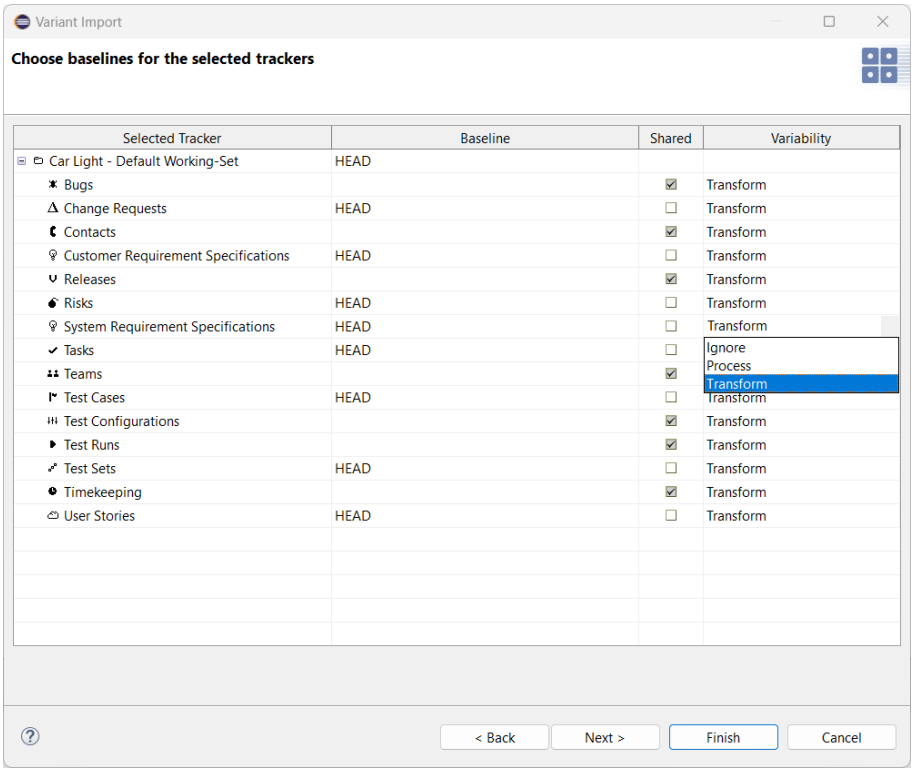


Figure 8. Select Variability Mode

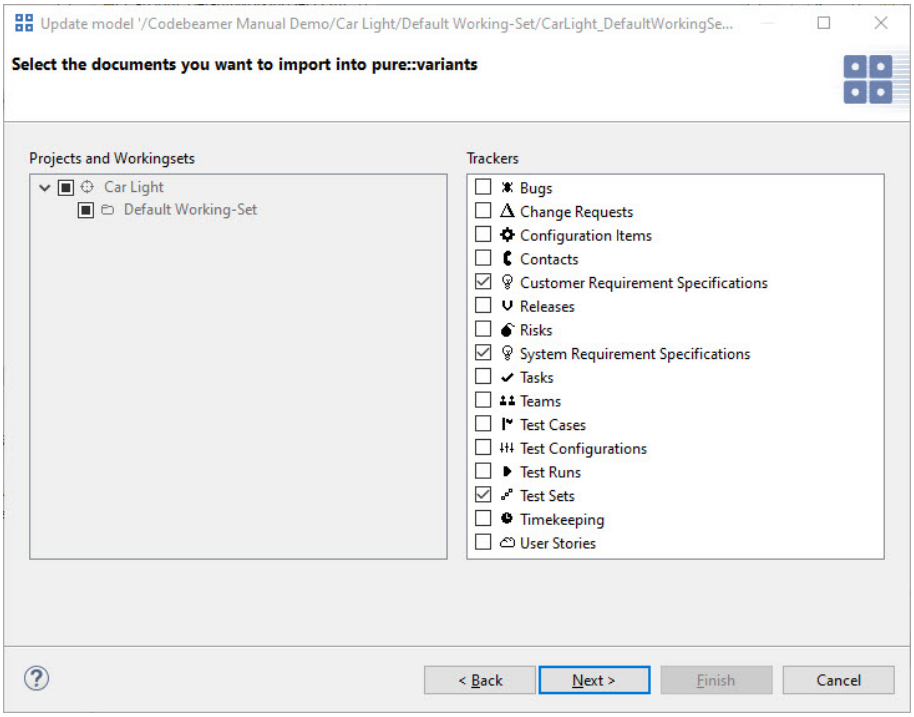


On the following page, the Import Rules are shown. On this page, you can select sets of Import Rules, that will be used to manipulate the resulting model after import. Import Rule Sets can be used to create specific pure::variants model elements like restrictions or constraints from Codebeamer artefacts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

2.5. Updating Models from Codebeamer

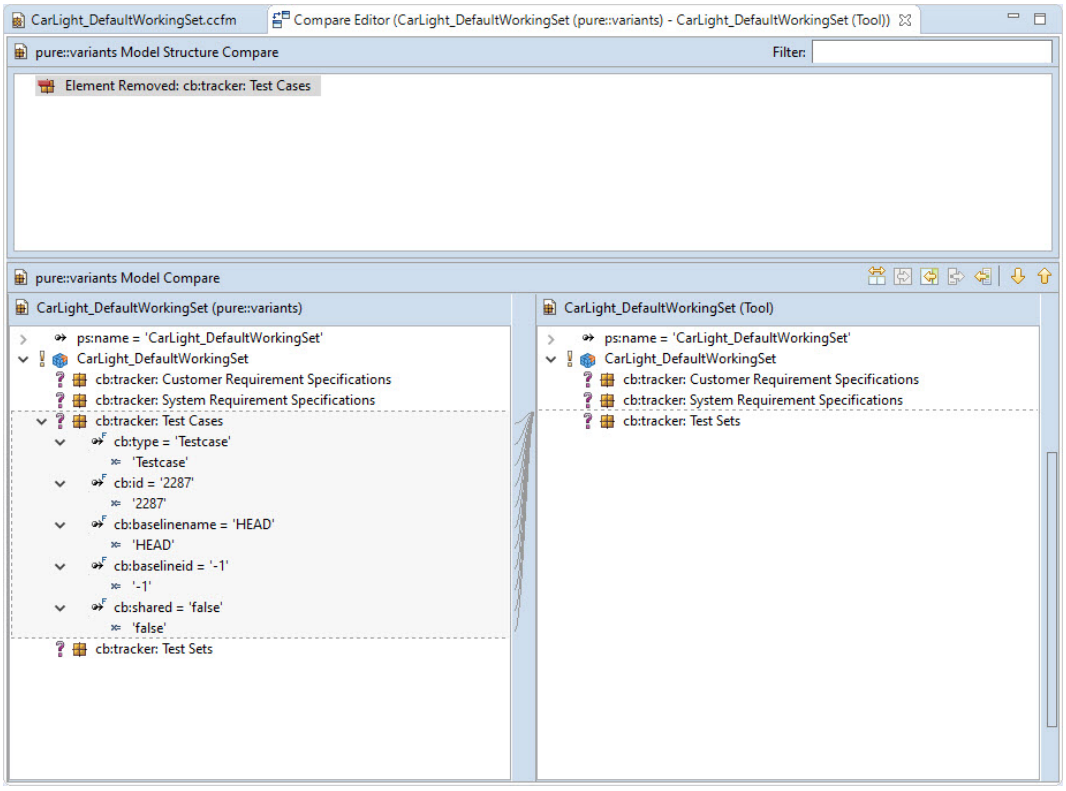
Using the Synchronize action, the set of trackers to be imported as part of a Working-Set can be modified. Additionally, when using Full Mode, it is necessary to update the pure::variants models with information from Codebeamer whenever relevant changes have been made. To start the update, open the model representing the Working-Set and press the Synchronize button in the tool bar.

Figure 9. Synchronize model



pure::variants will connect to Codebeamer to present the tracker selection page, the baseline selection page, and subsequently the Compare Editor for pure::variants models.

Figure 10. 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 Codebeamer 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 this example, the tracker 'Test Cases' was removed from the scope of the import.

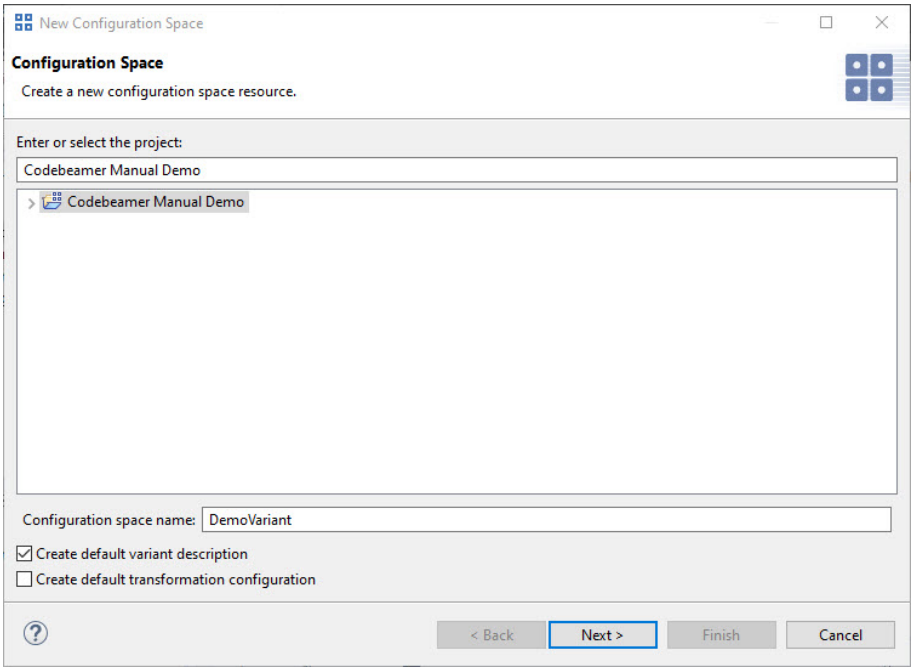
The Merge toolbar provides tools to copy single or even all (non-conflicting) changes as a whole from the current model to the out-dated model.

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 that are of interest to the users. Typically, this number is much smaller than the number of possible variants.

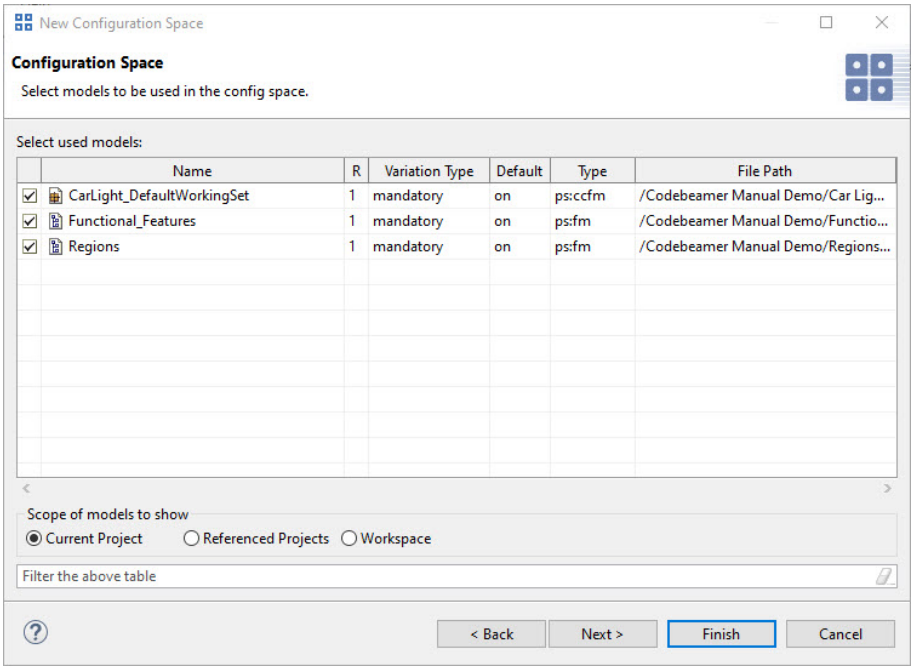
Variants are stored as separate entities called Variant Description Models (VDM). A VDM always belongs to a specific Configuration Space. Thus, before defining variants, a configuration space has to be created. Select the project containing the imported models in the Variant Projects view and open the context menu. Below the item **New** select **Configuration Space**. A wizard is opened. On the first page (Figure 11, “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 11. The Configuration Space Wizard



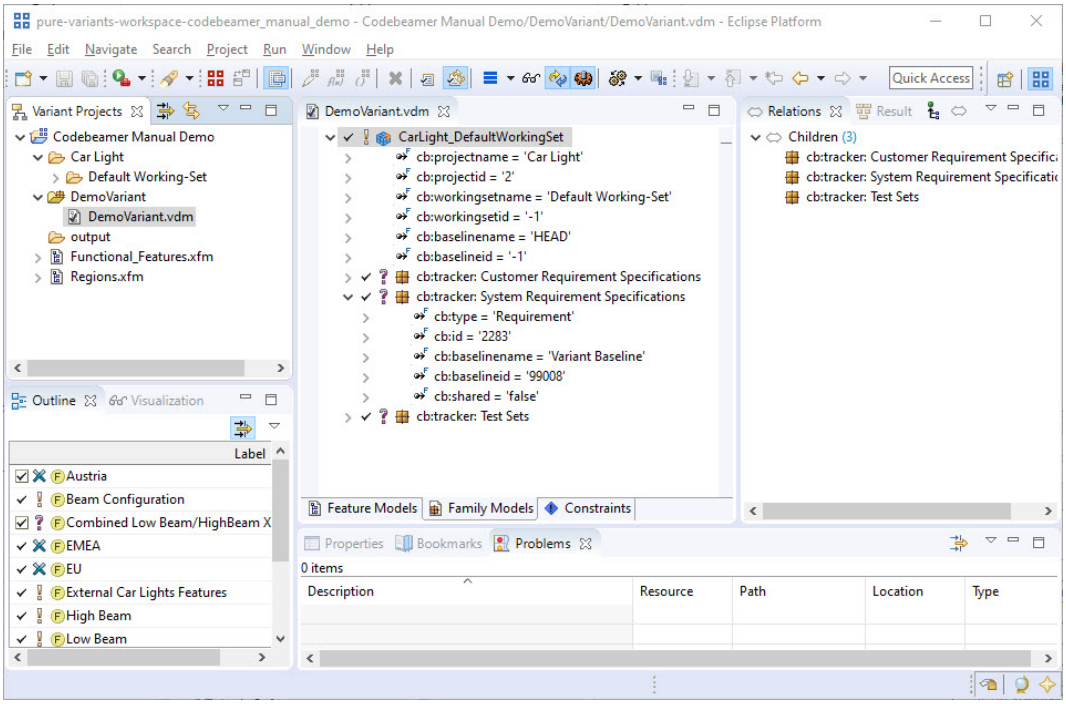
The next page is used to specify which models are to be included in this configuration space. Select here all models that represent the Working-Sets and so the trackers of interest. In the example below, just one Family Model is selected. Now press the **Finish** button.

Figure 12. The Configuration Space Wizard Model Selection Page



The resulting project structure is shown in (Figure 13, “Initial Configuration Space Structure”). The DemoVariants.vdm is created and immediately opened in case the **Create default variant description** was selected on the first page of the wizard.

Figure 13. Initial Configuration Space Structure



2.7. Transforming a Variant

Variants stored in a variant description model can be made available in Codebeamer. The Connector supports the following ways of representing variants: attribute-based.

Attribute-Based Variant Representation

In attribute-based representation, the variant field is filled up with the name of the variant (as a list, separated by a delimiter as defined in transformation configuration) if the tracker item is part of the variant.

This type of transformation is applicable to the following tracker types: Requirement, Test Case (incl. Test Steps), Test Set, Test Configuration, Configuration Items.

The result of this transformation also depends on the Variability mode set during import for each tracker as explained below:

- If Variability is set to Ignore or Process, no changes done to the tracker, i.e., variant names are not written into the defined variant field.
- If Variability is set to Transform, enumeration data is written to the tracker, i.e., variant names are written into the defined variant field.

Note: In case a tracker is excluded by the user for a variant in the Family Model, the variant name is going to be removed from this list of variants in Codebeamer.

Note: This transformation cannot be run on trackers with baselines other than HEAD and therefore pure::variants will report an error.

Working Set Transformation

Product Line assets (150%), i.e., trackers (requirements, test cases, etc.) are assigned to a dedicated Working Set or the Default Working Set in Codebeamer.

The Working Set Transformation can be used with or without update support:

- Without update support, the transformation creates a variant specific Working Set for each transformation run and for each variant (vdm) with trackers containing the variant specific subsets of tracker items (100%).

  By default, the naming convention for the variant specific Working Set created equals to the <VariantName>, e.g. ‚DemoVariant'. The default name can be changed by setting the transformation parameter 'WorkingSetName' (see chapter 'Preparing a Transformation '). In case a Working Set with the same name already exists, an error is reported and no transformation is performed.

  A practical way to create differently named working sets for each run is to append the timestamp of the transformation to the name. You can do it by setting the value of 'WorkingSetName' to '$(VARIANT)_$(QUALIFIER)'.

- With update support, the update of previously transformed variant specific trackers is supported. There are two modes available, the Manual Merge Mode, and the Full Overwrite Mode:

  Manual Merge Mode - in this mode, the variant is represented by two working sets, a reference- and a working-copy working set. The reference working set is created new in the first transformation run and is overwritten in each further transformation. The working-copy working set is also created by the first transformation and the content can be modified by the user. This working set is not updated automatically, but the changes made in the reference working set need to be merged manually into the working copy.

  Full Overwrite Mode: in this mode, the variant is represented by one working set that is created by the first transformation and is overwritten in each subsequent transformation.

  In needs to be decided upfront which update mode to use, switching between the modes after transformations were already performed is not possible.

  Note:

  - The user shall have the tracker level permission to replace the content of a branch (permission setting in Codebeamer: 'Branch - Replace content')

- In some tracker combinations, the working-set creation in Codebeamer changes the tracker configuration of the trackers in the new working set. However updating of trackers with a changed tracker configuration is not supported by the update process in Codebeamer. This restricts the usability of the update of certain tracker combinations, e.g. the trackers 'System Requirement Specification' and 'Customer Requirement Specification' always need to be included in the working set at the same time.
- See 'pure::variants User's Guide', Chapter 'Setting up a Transformation' for information how to enable update support.

For all modes of the Working Set Transformation, following needs to be noted:

- The user shall have permission to create working sets. (permission setting in Codebeamer: 'Working-Set - Admin').
- Included trackers (i.e. not shared trackers) are branched off from the trackers in the originating Working Set at the baseline defined by the user during import and are reduced to the variant specific subset. Shared trackers are only added to the Working Set but without any change as variability information is not considered there.
- The HEAD version of a variant specific tracker branch (if appropriate) can be only modified in Codebeamer to include variant exclusive content.

This type of transformation is applicable to following tracker types: Requirement, Test Case (incl. Test Steps), Test Set, Configuration Items.

Result of a working-set transformation also depends on the Variability mode set to the each tracker type during the import of a family model as explained below:

- If Variability is set to Ignore or Process, working-set(s) are created/updated and non-shared trackers are included, but tracker items are not processed (item removal and text substitution is not done).
- If Variability is set to Transform, working-set(s) are created/updated, trackers are included and tracker items are processed for non-shared trackers (item removal and text substitution is done).

Text Substitution

Following tracker types and fields are subject to text substitution:

- Requirement (Name, Description)
- Test Case (Name, Pre-Action, Post-Action, Test Parameters (both test parameter names and their values), Description)

  Test Steps (All fields of type Text and Wikitext)
- Test Set (Name, Test Parameters (both test parameter names and their values), Description)
- Configuration Items (Name, Description)

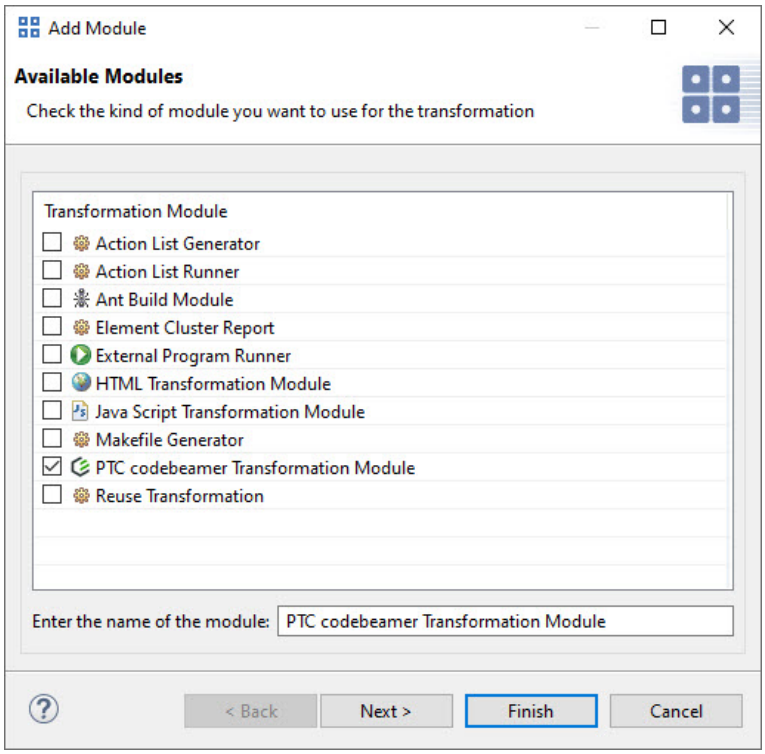
Note: Text substitution is performed only during 'Working Set Transformation' and only for trackers that are not included in a working set as shared.

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...*

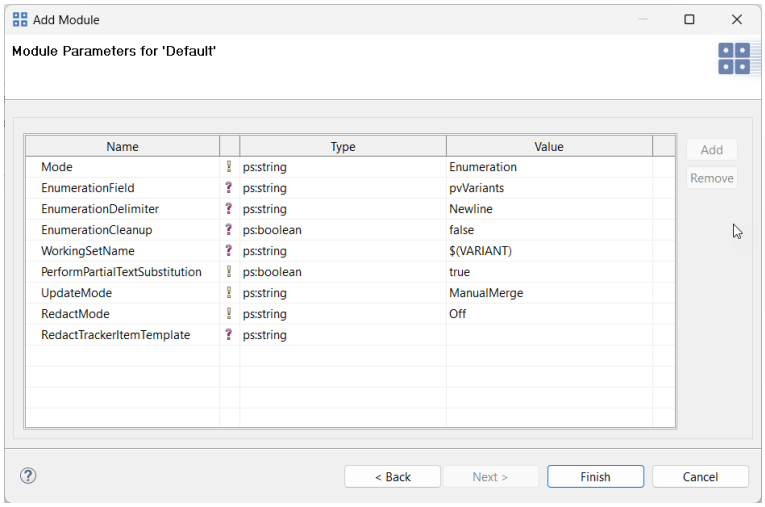
The configuration space property dialog opens, and the Transformation Configuration tab is shown. The next step is to add a new Module Configuration , by clicking the marked tool bar item. Now add a new Module to the Module Configuration, using the **Add** button.

Figure 14. Transformation Configuration



From the opened dialog, choose PTC Codebeamer 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: Define the transformation mode. The available modes are:

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

  Working-Set - this option stands for the Working Set Transformation.

- EnumerationField: Specifies the name of the tracker item field to be filled with variant names in Enumeration transformation mode. If not set, the default name ('pvVariants') is used.

- EnumerationDelimiter: Specifies the delimiter to be used to separate the variant names in the defined variant field in Enumeration transformation mode. Possible values are 'Newline', 'Space', 'Tab', ',', ';', '#' and '|'. If not set, the default delimiter 'Newline' will be used.

- EnumerationCleanup: If true is selected, the defined variant field is cleared in all the tracker's items before the variant names are written. If false, only the variant names of the transformed variant will be updated (either removed or added).

- WorkingSetName: Specifies the name of the Working Set created by the transformation.

- PerformPartialTextSubstitution: If true is selected, the partial text substitution is performed.

- UpdateMode: Defines the update mode for Working Set Transformation in case update support is enabled. The available update modes are:

  ManualMerge - this option stands for the Manual Merge Mode

  FullOverwrite - this option stands for the Full Overwrite Mode

- RedactMode: Enables redaction of tracker items during the working-set Transformation using the tracker item template defined in Codebeamer for each tracker, rather than removing the tracker items entirely. For more information see section Redact The available modes are:

  Off - Removes tracker items that are not included in the resulting working-set.

  HighestExcluded - Redacts the highest excluded tracker items that are not included in the resulting working-set and all child tracker items of these highest excluded tracker item will be removed.

  Note: By default, it is set to *Off*.

- RedactTrackerItemTemplate: Specifies the name of the tracker item template to be used by the working-set transformation for redacting tracker items that are not included in the resulting working set. By default, this value is set to empty

  Note: A tracker item template with the same name should be configured for each Codebeamer tracker that is included in working-set transformation.

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

Web Client Integration for transformation

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

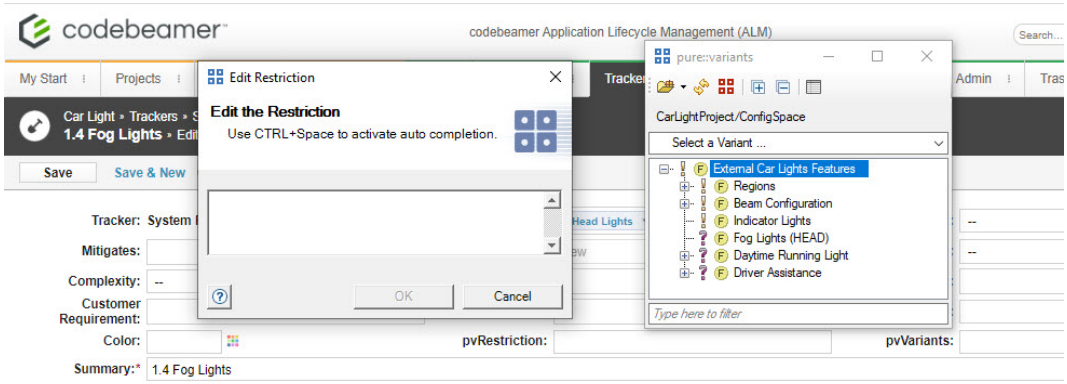
3. Using the Integration

In order to facilitate the pure::variants connector for Codebeamer variability, information needs to be added to the tracker items. This is performed by adding restrictions to tracker 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 Codebeamer.

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 it into active fields being edited by the user. In Codebeamer, the tracker item needs to be opened in edit mode first, then select the 'pvRestriction' field before activating Desktop Hub using the hotkey combination set.

Figure 16. Adding restrictions in Codebeamer 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 Codebeamer (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.

Prerequisites for the Desktop Hub Mode

In order to run the integration in Desktop Hub mode, a running instance of the Desktop Hub is required in the 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 the 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 Desktop Hub mode.

For loading 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 will be 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 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 the 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 to save the mode settings. Integration will then redirect to its main page and start running in Web Hub mode.

Permissions

The user requires to have the following permissions being set to edit settings in the widget (as described further sections). By the system administrator, the user has to be assigned to a *user group*, which has the *Rest / Remote API - Access* granted (via *System Admin->User Groups* menu). Furthermore, the user has to be assigned a project-specific *user role*, which has the *Trackers - Admin* permissions granted (via project's *Admin->Members* menu).

Define Transformation Related Settings

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

- Using the General tab, the restriction attribute and the attributes used for partial text substitutions can be defined.

  The default value for the restriction attribute is 'pvRestriction'.

  For the default attributes for partial text substitution for each tracker type see the section called “Text Substitution”

- For tracker items inside Table-Fields, the name of the field is appended without whitespace and special characters to the value defined above, e.g. for 'Test Steps' it results in 'pvRestrictionTestSteps'

- Using the Calculation tab, the text substitutions markers can be defined. The default values are:

  Opening character is [

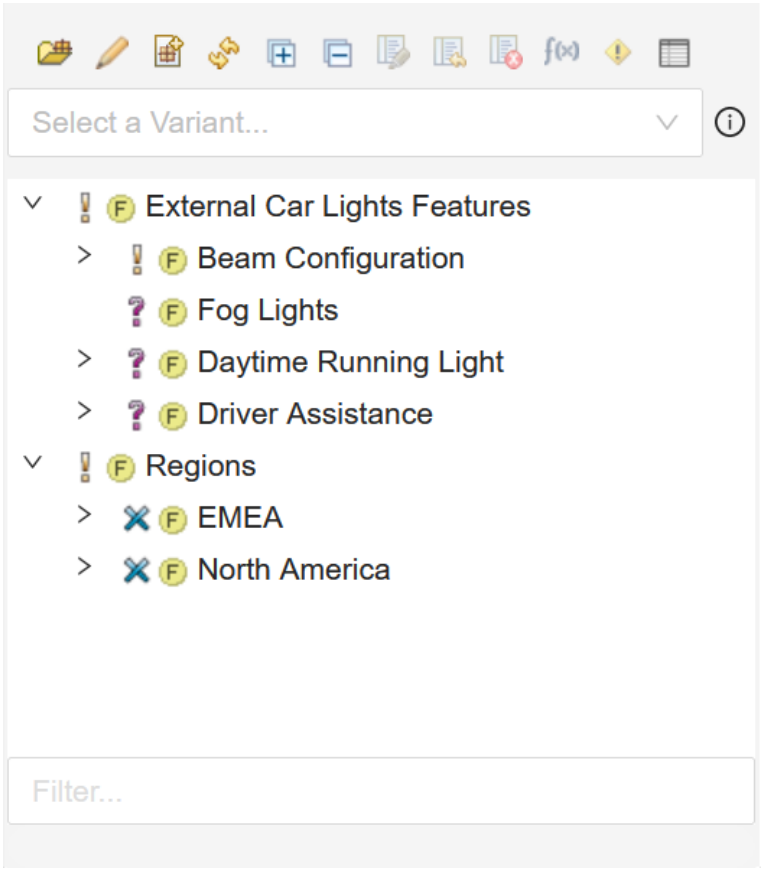
  Closing character is]

  Escape character is $

3.3. Introduction to *Integration* GUI

The *Main* page view of the Integration is shown in Figure 17, “Integration *Main* page view”

Figure 17. Integration *Main* page view



The integration is displayed in the side panel of the Document View and is available for supported tracker types where the DocumentView is available.

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

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

Note

While working on various browser tabs/windows (of same browser installation), the loaded configuration space in one tab is loaded in every other tabs as well.

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

10. **Calculations** button - click to open the *Calculations* page, so to edit Calculations present inside the fields of a tracker item.

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

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*. On selecting any of the variant model from the dropdown, the model will be rendered inside the *Tree-view*. *The* **Tree-view** lists the selected *Feature/Variant* model(s).

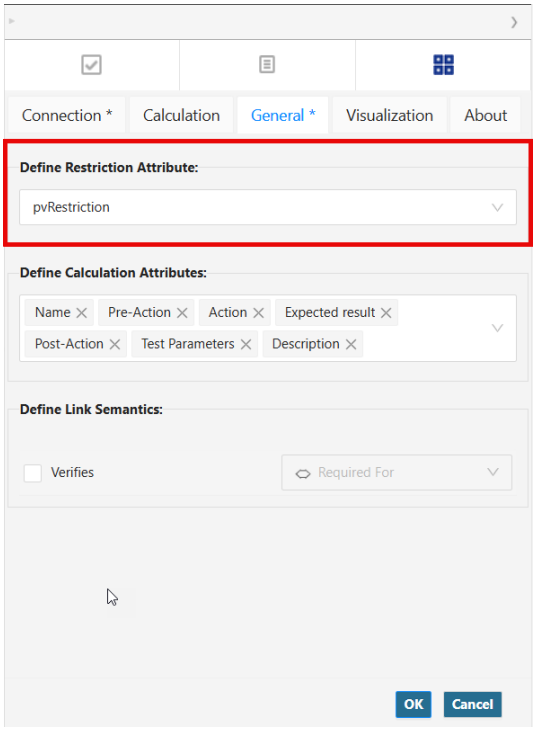
Note:

- The button for Error Check is disabled in case the incorrect DesktopHub or WebHub version is used (see Chapter 'Software Requirements').
- During preview vertical scrolling of the main document section is supported to enable review of the document.

3.4. Define Restriction attribute

In order to associate the restrictions with the tracker items, a custom field needs to be created in each of the trackers, for example a *Text* field with label *pvRestriction*. This can then be selected as a restriction attribute in the pure::variants Integration for Codebeamer by opening Settings () and navigating to the **General** tab as shown in Figure 18, “General Settings”, selecting the name of the target attribute, and then pressing **OK**.

Figure 18. General Settings

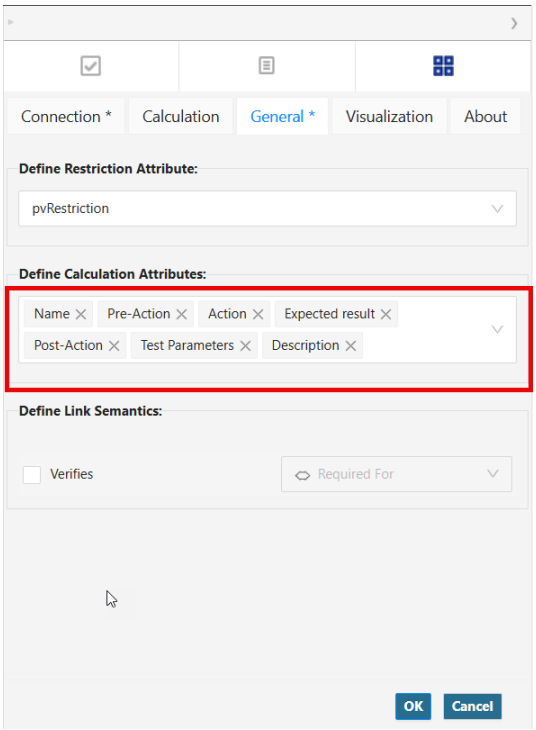


Note: By default, no attribute is selected as a restriction attribute.

3.5. Define Calculation attributes

In order to enable substitution of various tracker items' attributes, the user can configure the selection of attributes in the **Define Calculation Attributes** dropdown menu on the **General** settings page, see ???. This attribute selection will then be considered during preview and transformation.

Figure 19. General Settings



By default, only the Calculations present in following fields are replaced with values depending on the tracker type.

1. **Requirement** - *Summary* and *Description*

2. **Configuration Items** - *Summary* and *Description*

3. **Test Cases** - *Name*, *Description*, *pre-action*, *post-action*, *Test Parameters*, *Action* and *Expected result*

4. **Test Sets** - *Summary*, *Description* and *Test Parameters*

The selected attributes by default can be freely deselected, and other attributes selected, which are selectable in the dropdown menu.

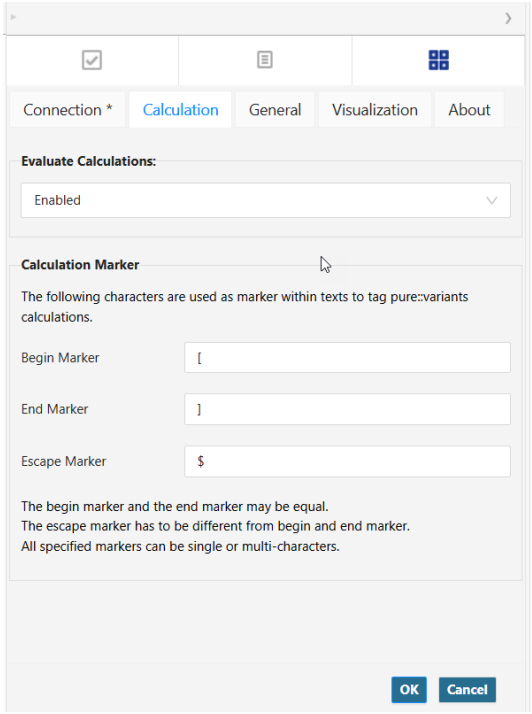
3.6. Define custom substitution markers

(Optional) Adding calculations to tracker item fields is working out of the box using configurable *Calculation Markers*. By default, the markers are set as follows:

- *Opening character is* ***[***
- *Closing character is* ***]***
- *Escape character is* ***$***

If this does not fit the requirement, defining custom markers is also possible via the pure::variants Integration for Codebeamer. The editing of custom markers can be done by opening Settings () and navigating to the **Calculation** tab as shown in figure Figure 20, “Calculations Settings”.

Figure 20. Calculations Settings

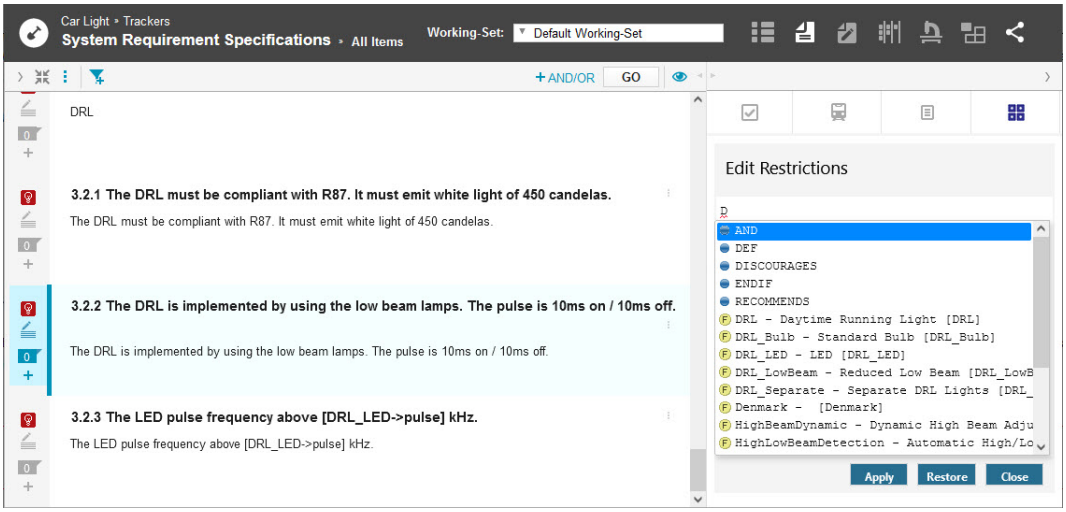


Calculations get substituted by their calculated values only then, when the **Evaluate Calculations** value is set to **Enabled**.

3.7. Working with the 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 tracker. The Restriction Editor provides the ability of auto-completion proposals and syntax highlighting while editing restrictions for tracker items.

Figure 21. Working with the Restriction Editor

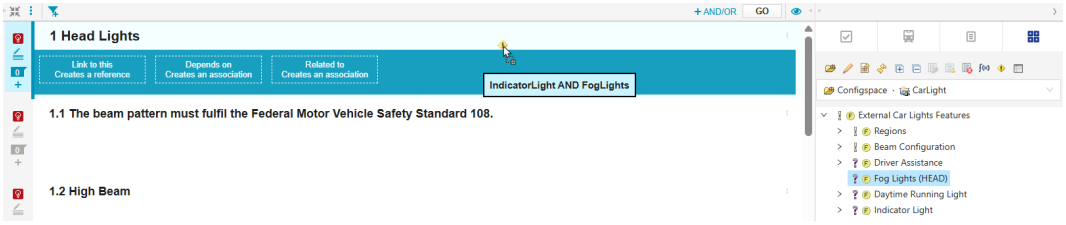


For a more intuitive and simpler way of adding variability to a tracker item, drag'n'drop functionality can be used. Drop of single or multiple features as restriction is allowed.

- Select a tracker item for which restriction should be added.
- In the integration's model tree, select one or more features (hold CTRL for multi-selection).
- Drag and drop the selected feature(s) onto the tracker item.

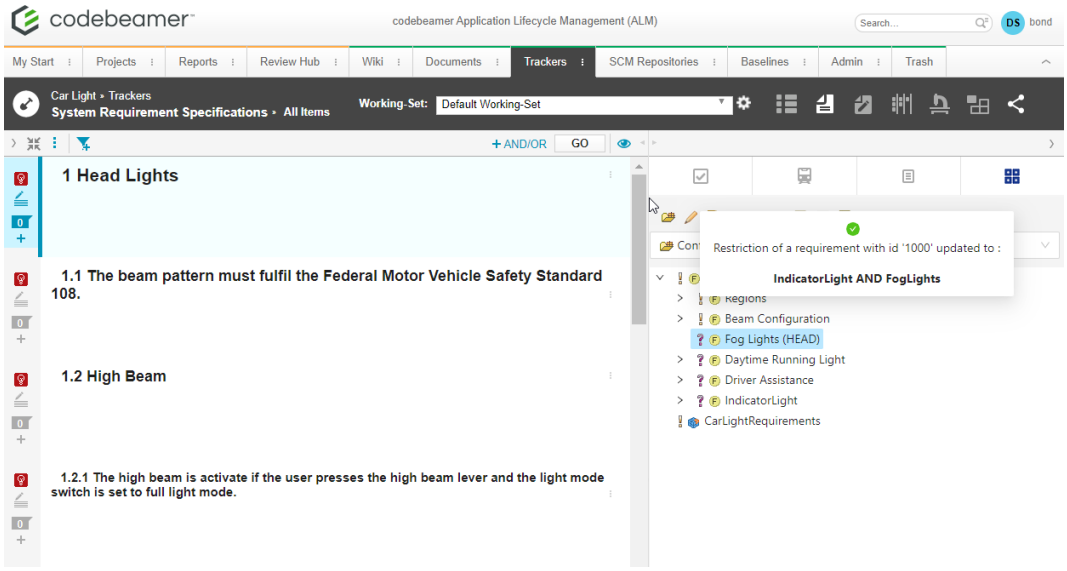
By default, the dropped restriction concatenates with the existing pvSCL expression using "AND". If the CTRL key is held during the drop action, the existing pvSCL expression will be concatenated using 'OR' instead. To provide clarity on the resulting restriction after the drop, a preview is displayed.

Figure 22. Drag a single feature to add a restriction



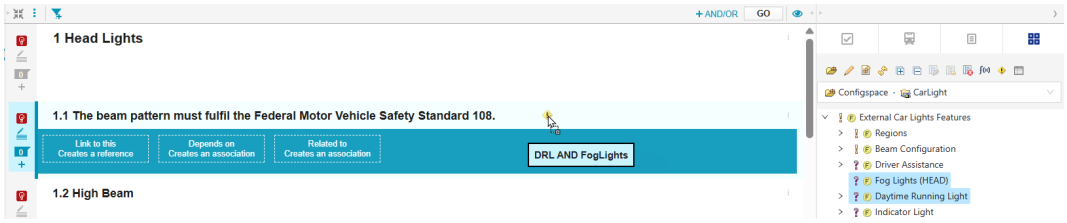
Restriction is added to the tracker item.

Figure 23. Drop a restriction on tracker item



When multiple features are dropped, they are concatenated with "AND" by default; holding CTRL changes it to "OR".

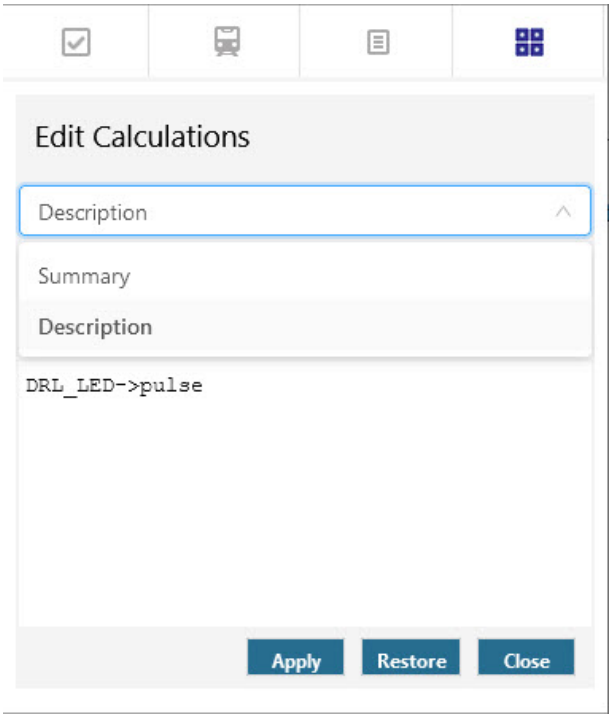
Figure 24. Drag multiple feature to add a restriction



3.8. Working with the Calculations Editor

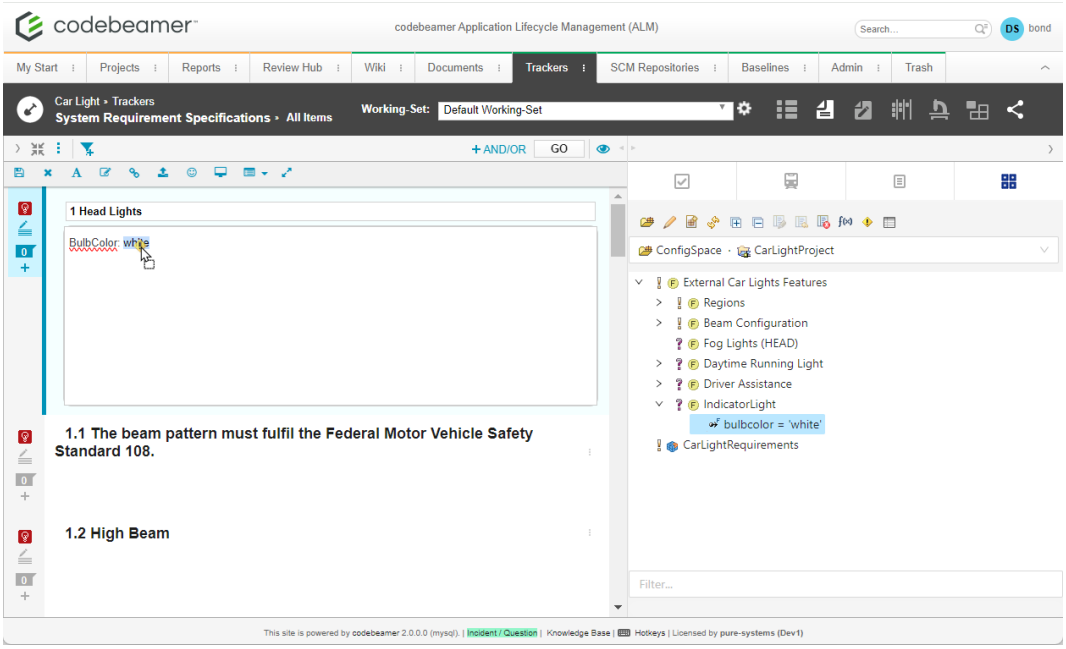
The Calculations Editor can be used to edit the Calculations present in the fields of a tracker item. You can open it by clicking the **Calculations** icon. Calculations can be edited by selecting an item in a tracker and then, in the Calculations Editor, select the field of an item that contains the Calculation 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 supports auto-completion of proposals and syntax highlighting while editing Calculations.

Figure 25. Calculations Editor of the Integration



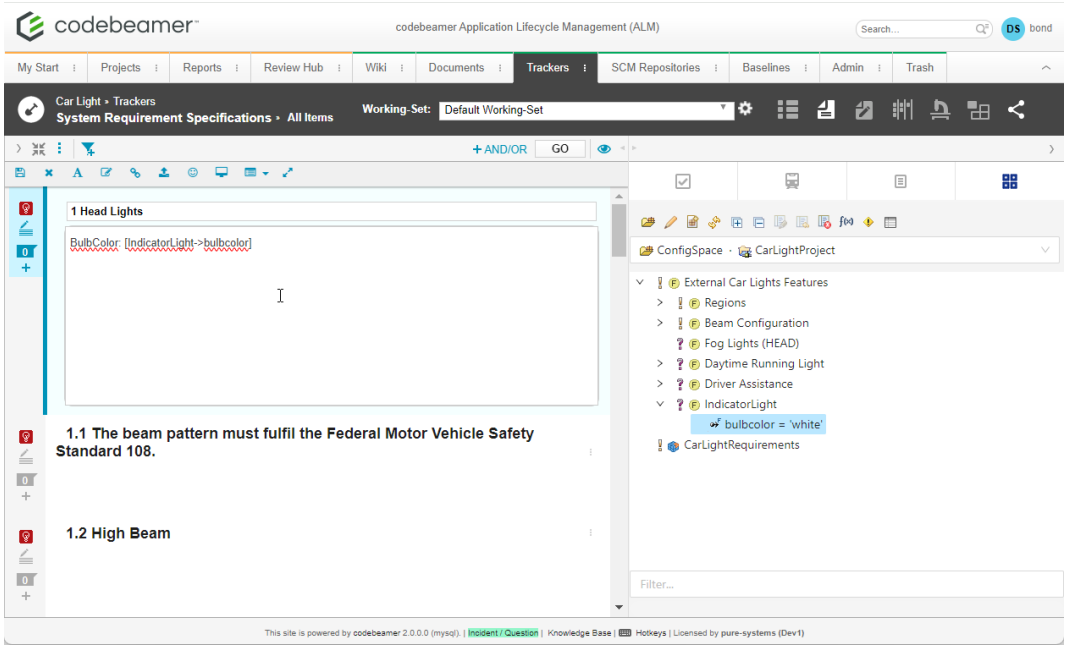
Calculations to a tracker items can also be easily added by using drag'n'drop functionality. Enable tracker item's edit mode and select an attribute from the integration's model tree, drag it, and drop it onto the edited field of the tracker item.

Figure 26. Drag an attribute to add a Calculation



Calculation will be added at the respective position.

Figure 27. Drop the Calculation on editable tracker item field



Note

1. Drag'n'Drop of Calculation is only supported for tracker item's Description and TestStep table fields.

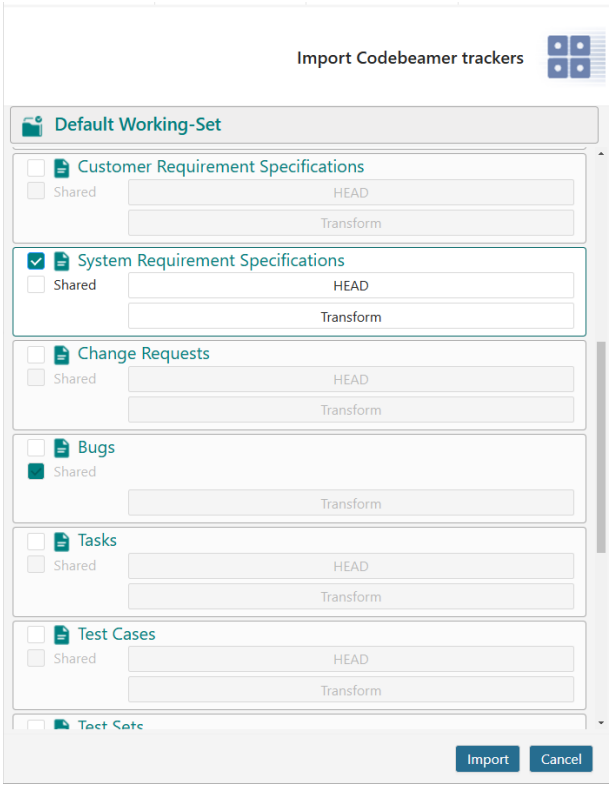
2. Drag'n'Drop of property as a Calculation on tracker item's Description and TestStep table fields is not supported in Firefox. This functionality is only available in Chrome and Microsoft Edge.

3.9. Importing and Updating Models using the Web Client

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

On press of **Import** button, all trackers of the actual working set are listed in the dialog (Figure 28, “Import Dialog of Integration”), with the opened tracker preselected. Here, select the trackers to be included in the import along with the baseline information or alternatively specify if a tracker shall be included as shared in the family model. Additionally, Variability mode can be set to each of the selected tracker. By default, it is set to 'Transform'. More details on Variability modes can be found in section Creating the Initial Model(s). 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 28. 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.

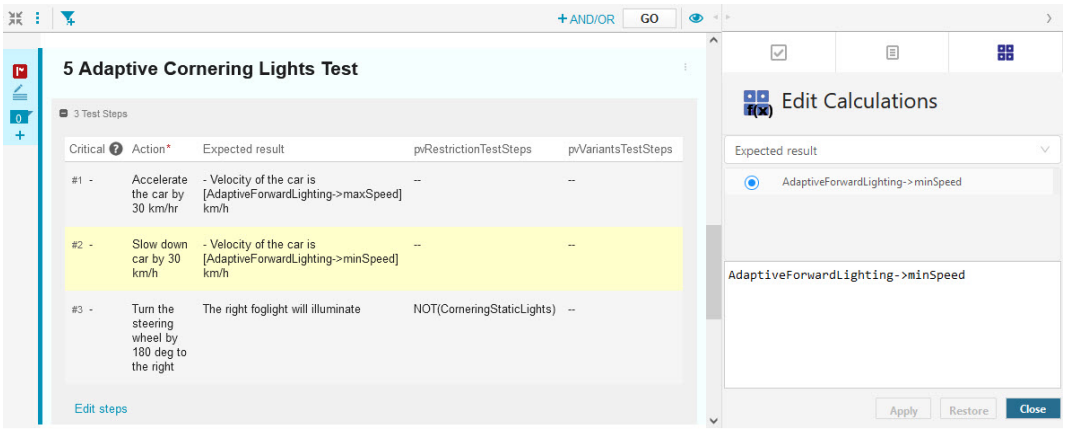
On press of **Update** button, all trackers of the actual working set are listed in a dialog, where the trackers that were previously imported are preselected. After confirming the selection, the Web Client opens where a wizard guides through the rest of the update process.

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.

3.10. Working with Test Steps

Both the Restriction and the Calculation Editor support Test Steps as listed inside Test Cases in the Document View of Codebeamer. Here, a concrete row representing a Test Step needs to be selected in the expanded Tests Steps table, and so the data of this Test Step will appear in the editors. Restrictions will be added to the 'pvRestrictionsTestSteps' field, while Calculations will be added at the locations indicated by substitution markers inside supported fields, as shown in the example below (Figure 29, “Editing Calculations Inside Test Steps”).

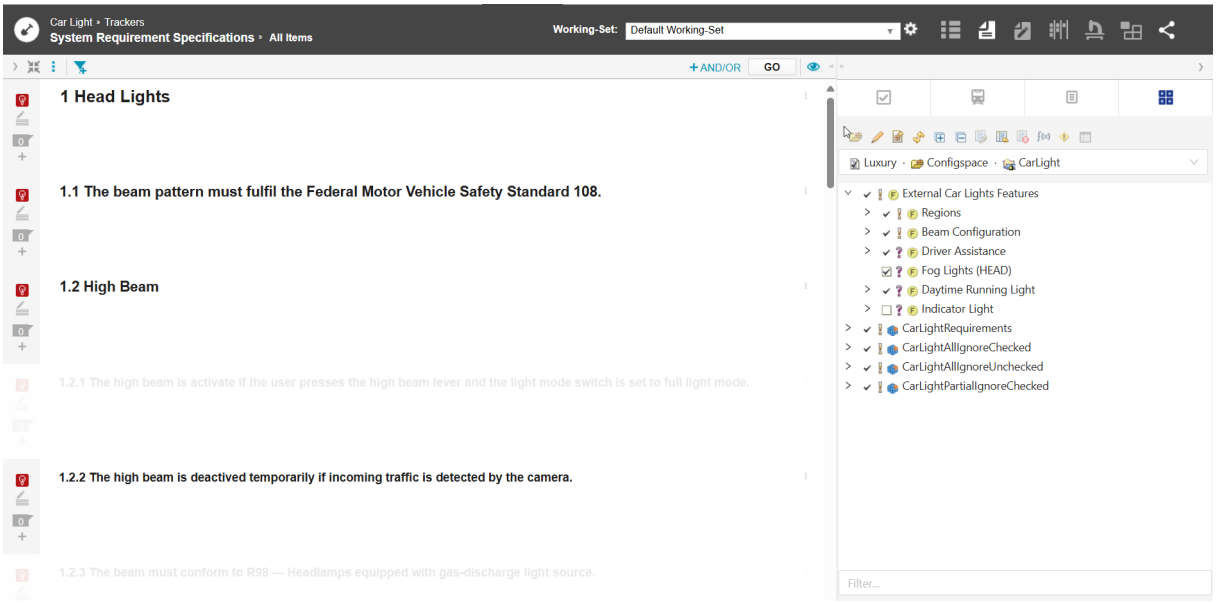
Figure 29. Editing Calculations Inside Test Steps



3.11. Visualizing Variability Information (Preview)

The Integration is capable of visualizing the variability information (Preview) of a particular variant model. In order to see the *Preview*, select a variant model from the **VDM selector** dropdown list. On selection, the feature model inside the **Tree-view** will be replaced by the variant model. Then press the **Show Preview** button and a variability preview of the module will be shown in context to the selected variant model. In *Preview* mode, selected requirements retain the normal view, while the unselected requirements are greyed out, as shown in Figure 30, “Variability Preview”, and all substitutable attributes will be substituted with the variant specific values.

Figure 30. Variability Preview



While the *Preview* is enabled, selecting another variant model from the **VDM selector** dropdown is also possible. Changing the varaint model will update the preview again with respect to the newly selected variant model. To reset the *Preview* mode, press the **Reset Preview** button .

Note

It is not recommended to edit the tracker item fields in preview mode. Doing so could lead to discrepancies in the already applied preview result.

3.12. Error Check

Added variability information can be checked for errors using the Error Check function. Errors in pvSCL scripts are reported if the script's syntax is not pvSCL compliant, or if an element is unknown based on the loaded pure::variants models.

The problems are displayed in a list containing following information:

- Message: Description of the error.
- pvSCL: The pvSCL expression containing the error.
- Field: The name of the field that contains the error.
- Item Link: URI of the affected item.

Severity of the problem is indicated by an icon (error or warning).

Note: To check with hierarchical feature model structures containing variant instances, a variant model also needs to be selected for proper evaluation.

3.13. Variability in WIKI-Tables

To add variability to WIKI-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 they need to contain the specified keyword, which is also used to indicate a restriction, on e.g. a requirement. By default, this keyword is pvRestriction.

Figure 31. Example WIKI-Table

Color	Technology	Static Cornering Light	
White	LED	increased fog light brightness	LED
Yellow	LED	increased fog light brightness	NOT(EU) AND LED
White	Halogen	-	Halogen
Yellow	Halogen	-	NOT(EU) AND Halogen
		CorneringStaticLights	**pvRestriction**

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 values of its row and its column. In the example the whole column "Static Cornering Lights" will only be part of the variant, if the feature CorneringStaticLights was selected. The cell below the heading in that column will be included in a variant if CorneringStaticLights AND LED was selected.

The variability information cells (e.g. the marked, yellow pvRestriction cells in the example) are removed from the variant by default.

Calculations will also be computed if they are marked with the respective open and close characters, and nested tables, so tables with cells, that themselves again hold a table, are supported and comply to the same rules as described above.

4. Advanced Topics

4.1. Link Propagation

Link propagation allows to evaluate the dependency relations between tracker items of the same tracker or tracker items of different trackers with the usage of Pure Variants relation types. Therefore, for each link type defined

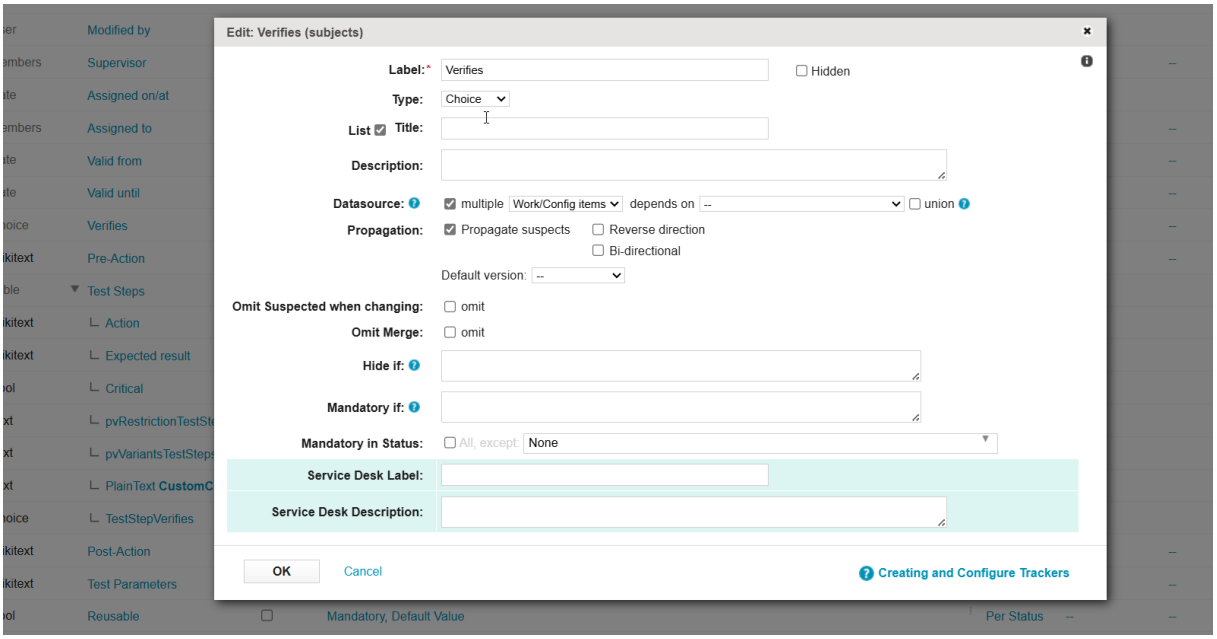
in the Codebeamer integration, an appropriate Pure Variants relation type is added in the imported family model. With the Codebeamer integration, the user can easily configure the appropriate mapping of link and relation type for each tracker as needed.

Link propagation is only supported within a set of trackers. Codebeamer references are supported for link propagation, if the reference source is an item or table row (in HEAD or in any revision) and the reference target is an item or item revision. References to non-items are not supported and shall be ignored. In the Codebeamer connector, only fields of the given set of trackers are used as link types where:

- Field type is Choice
- Choice Data Source is Work/Config items
- Item set limit is set to the currently imported project or to any project
- Item set limit is set at least to one of the currently imported trackers or any trackers

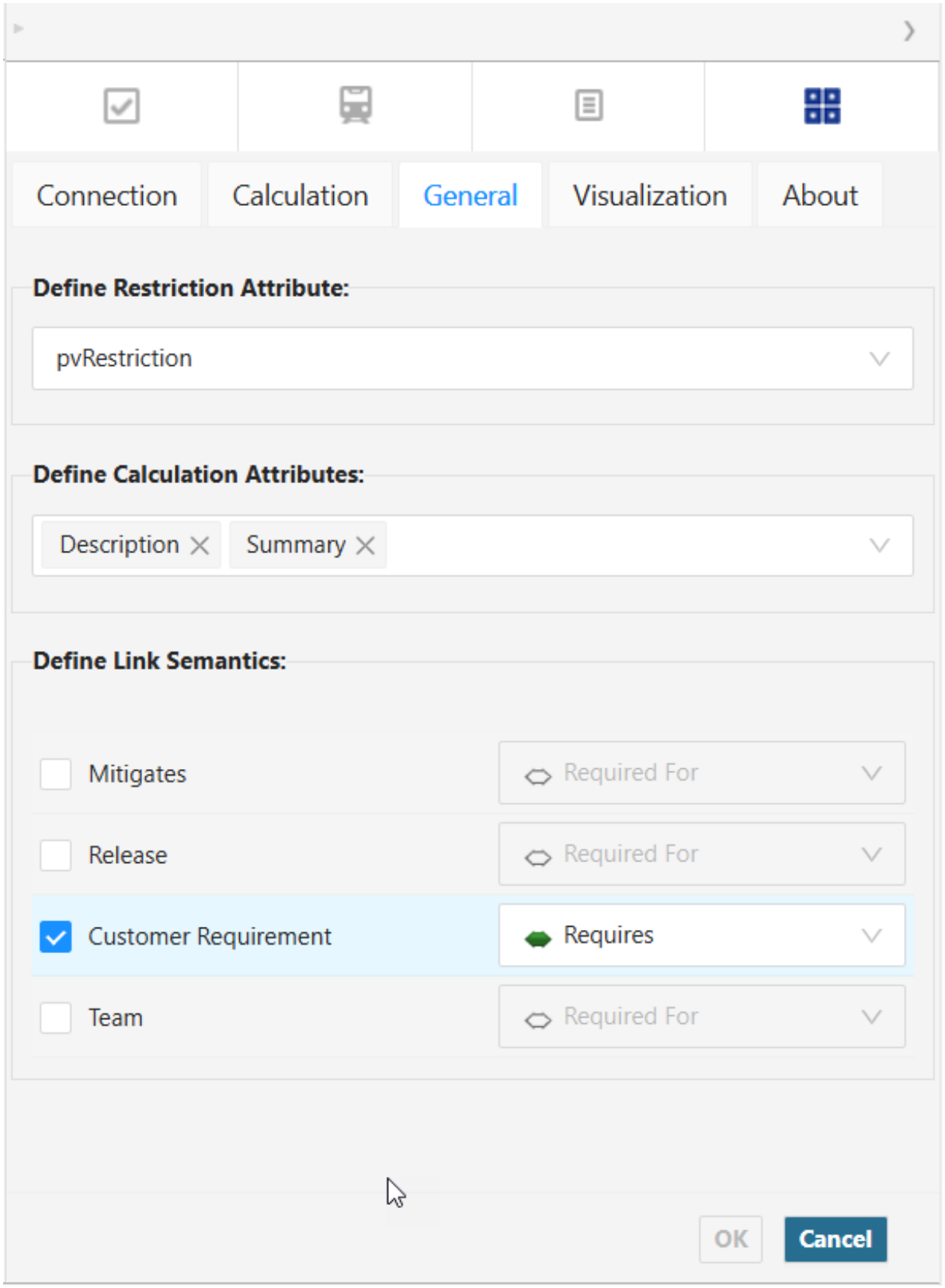
For example, in trackers of type Test Case, a predefined field "Verifies" exists as shown below, which fulfills the above criteria.

Figure 32. Define links via Pure Variants Integration



The link type set shall contain only those fields, which can potentially define references among the imported tracker set. Using that restricted set of fields/reference types used as link types, the user can create a mapping from those link types to Pure Variants relation types as shown below:

Figure 33. Define links via Pure Variants Integration



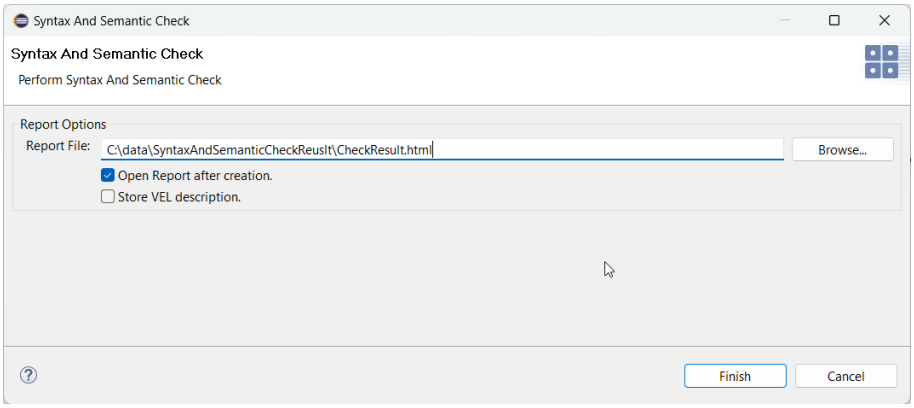
For each dependency, the corresponding Pure Variants relations are created between them (for each one-to-n dependency, a one-to-n relation is created).

4.2. Checking all Codebeamer tracker connected to one Configuration Space for semantic and syntactic problems

Before transforming a lot of modules it is possible to check all pvscl rules in these modules. This ensures the transformation will not fail due to problems with the pvscl rules or a misconfiguration. To use this functionality chose **Perform Syntax and Semnatic Check** from the context menu of the config space or the context menu of a selection of variant models.

A dialog pops up. In the dialog specify the options and an output path to the result report and click finish. The check now imports the variability information form Codebeamer to a VEL model and checks all the pvscl rules. Afterwards all selected variant models are evaluated to make sure there is no misconfiguration.

Figure 34. Syntactic and Semantic check dialog.



Along with the result report, a log file is written. This contains detailed information if the process fails.

By enabling the option *Store VEL description* the imported VEL descriptions are stored into the same folder as the report is stored. The VEL XML files can be imported to pure::variants for further analysis of the problems.

4.3. Redact tracker items during transformation

The *RedactMode* parameter enables the selective redaction of hierarchical tracker items based on rules defined on the tracker items during the working-set transformation process. Performing a working-set transformation with redaction enabled, i.e. *RedactMode* set to *RedactHighestExcluded*, top-level tracker items, that are not part of the resulting working-set, are redacted. This means, that they are replaced with the configured tracker item template's fields' values. The template is configured in the transformation parameter *RedactItemTemplate*. The available *RedactMode* options are:

- *Off* - Peforms a working-set transformation with no redaction, i.e. removes tracker items, that are not included in the resulting working-set during the working-set transformation.
- *RedactHighestExcluded* - If set, the tracker items get redacted instead of removed according to their hierarchy, when they are not included in the variant result during the working-set transformation.

  i. The top-level excluded tracker item is kept, and not removed as usual.

  ii. The kept tracker item is being redacted with the defined tracker item template's values.

  iii. All its child tracker items are getting removed as usual.

  If the top-level tracker item in the hierarchy is included and few/ all of its children are excluded, then

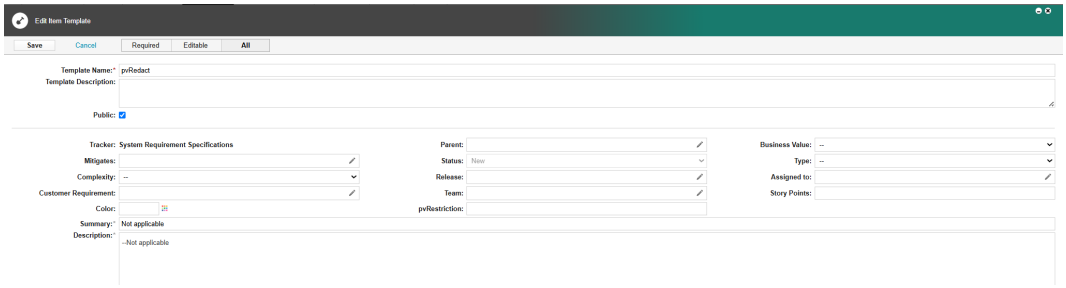
  i. The top-level tracker item in the hierarchy is kept as it is.

  ii. Few/all off its child items are redacted with the defined tracker item template's values.

To enable redact support during working-set transformation, the following steps needs to be performed:

1. Create a tracker item template for each of the Codebeamer trackers, that will be part of the working-set transformation. More details can be found in the Codebeamer documention (see Creating Tracker Item Templates).

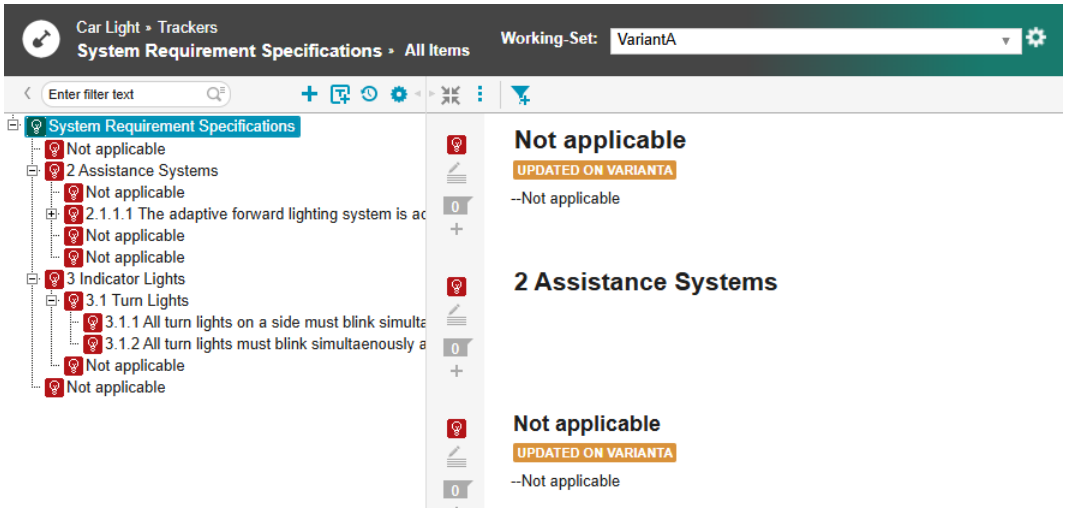
Figure 35. Tracker item template example usable for redaction



2. Configure the created tracker item template's name in the transformation configuration parameter *RedactItemTemplate*

   Note: when *RedactItemTemplate* is not configured, the default template name 'pvRedact' is used in a transformation.

Figure 36. Transformation result example with RedactMode: RedactHighestExcluded



Redact test cases and test steps during transformation

In addition to redacting tracker items, test steps within test case trackers can also be redacted. During a working-set transformation, test steps, that are not part of the variant result, are redacted, i.e. replaced with predefined test step field values from a tracker item template. Additionally to the behavior described in Section 4.3, “Redact tracker items during transformation”, the test steps get handled during the transformation as follows:

- *Off* - Peforms a working-set transformation with no redaction, i.e. removes test case tracker items and test steps, that are not included in the resulting working-set during the working-set transformation.
- *RedactHighestExcluded* - If the top-level test case tracker item in the hierarchy is not included in the variant result during the working-set transformation, then

  i. The top-level excluded test case tracker item is kept as usual.

  ii. The kept test case tracker item is being redacted with the defined tracker item template's values.

  iii. All of its test steps are removed.

  If the top-level test case tracker item in the hierarchy is included and few/all of its test steps are excluded, then

i. The top-level test cases tracker item in the hierarchy is kept as it is.

ii. Few/all of its test steps are redacted with test step values defined in the tracker item template.

Note:

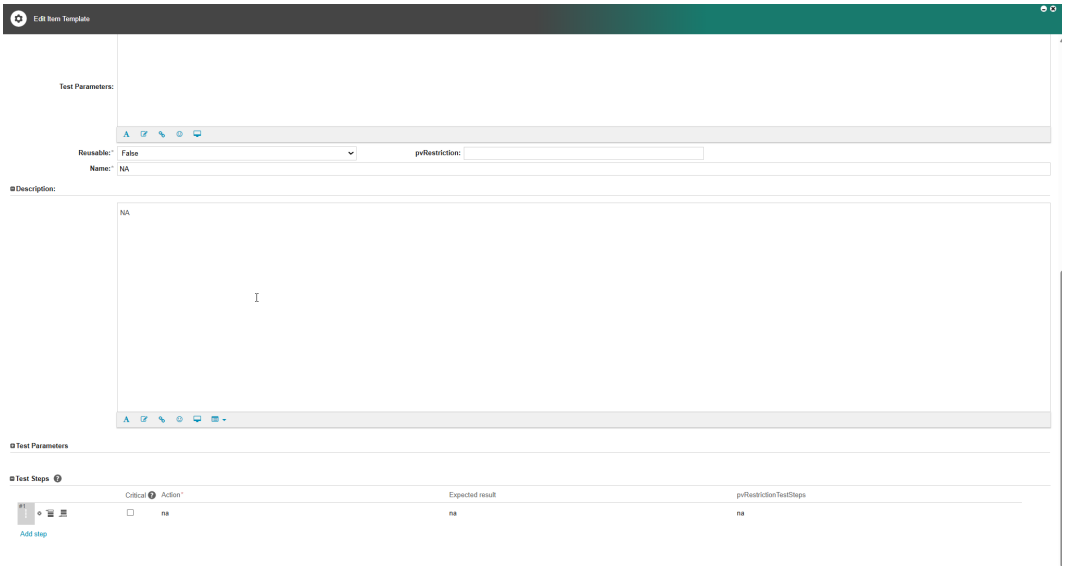
1. If test steps are not defined in the item template, the redaction of the test steps is skipped.

2. If multiple test steps are present in the item template, only the first test step's values are used for redaction during a transformation.

To enable redact support for test steps in test case tracker items during the working-set transformation, the following steps need to be performed:

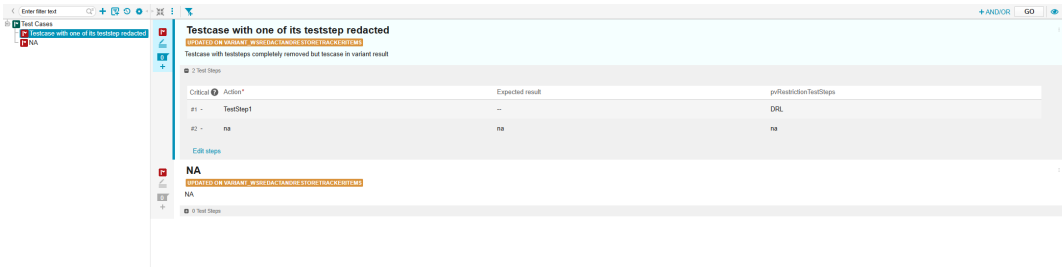
1. Create a tracker item template containing a test step with predefined values as show in the example in Figure 37, “Tracker item template example with a test step used for redaction”

Figure 37. Tracker item template example with a test step used for redaction



2. Configure the created tracker item template's name in the transformation configuraion parameter *RedactItemTemplate*

Figure 38. Transformation result example with test steps for Redact mode: RedactHighestExcluded



5. Restrictions

5.1. Limitations Relating Text Substitution in WikiText Fields

Since the set of pure::variants substitution marker characters can conflict with WikiText special characters (e.g. '[...]' defines a WikiText hyperlink), the pure::variants text substitution processing looks for the WikiText-escaped

form of those special characters (so e.g. '~[' and '~]' for the default markers '[' and ']'). This form is created in most cases by the Codebeamer WikiText Rich Text editor when adding these characters there. Also the integration widget functionality will use this escaped form.

The usage of the Codebeamer WikiText RichText editor and the WikiText syntax restrictions in general will result in limitations of the usage of text substitutions within WikiText content:

- Text substitution sections are only supported where formatted text can be used within WikiText content. So it is for example not supported to add substitutions in WikiText control sequences or in the target URL part of a WikiText hyperlink.
- Since the text sequence '${...}' has special meaning in Codebeamer, it is not recommended to use '{' and '}' as substitution open and close markers and '$' as substitution escape marker.
- It is not supported to use text formatting within or across text substitution section boundaries. This can result in invalid pvSCL expressions or in the creation of invalid WikiText content during transformation. The only exception is the usage of formatting within a pvSCL text literal.
- In Codebeamer, the item description field format can be switched from WikiText to plain text for each single item. The usage of text substitutions in such plain text descriptions is not supported.

5.2. Known Limitations of the Supported Codebeamer Versions

In this section known issues of Codebeamer are listed, which cause limitations of the functionality of the pure::variants Connector for Codebeamer:

- Associations between tracker items are added to a newly created variant working-set but are not updated during working-set update in any of the Codebeamer versions.
- Working-set update fails to reflect changes made to tracker items in the source working-set when using Codebeamer versions 2.0.0.2 to 2.0.0.4, 2.1.0.0 to 2.1.0.2 and 2.2.0.0. Codebeamer Issue#11504164/
- Working-set update fails when new test case items are added to the source working-set in Codebeamer versions 2.0.0.0 to 2.2.0.0. Codebeamer Issue#13054370
- Full-mode family model imports and transformations will only work in Codebeamer cluster-architecture deployments when Codebeamer version 2.1.0.2 or 2.2.0.0 is used.
- Updating a variant working-set based on newer baseline/ HEAD is not correctly done in Codebeamer versions 2.0.0.0 to 2.2.0.1. This results in a not updated variant working-set. Codebeamer Issue#11504164
- Merge operation fails to reflect changes in the order of tracker items in source or target branch in Codebeamer version 2.2.0.1 PTC Issue 17483492.
- Merge operation fails to reflect the addition of tracker items from the source branch to the target branch if those tracker items were previously removed in the target branch in Codebeamer version 2.2.0.1 PTC Issue 17483611.
- Merge operation fails to merge changes when items are deleted from a source branch in Codebeamer version 2.2.0.1 PTC Issue 17483350.
- Merge operation fails to maintain consistency in the tracker items order as per the source branch in Codebeamer version 2.2.0.1 PTC Issue 17483509.

6. Troubleshoot

6.1. Connection Issues - Timeouts, Interrupts, etc.

Since the pure::variants Connector for Codebeamer relies on the Codebeamer REST APIs, frequent network communication is required, and performance can vary based on the company's infrastructure. If there are issues such

as slow network connectivity or delayed server deployments, these challenges can be mitigated 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**.