# pure::variants - Connector for IBM Engineering Requirements Management - DOORS Next Manual

Parametric Technology GmbH

Version 7.0.0.685 for pure::variants 7.0



2025

## Table of Contents



## 1. Introduction

pure::variants - Connector for IBM Engineering Requirements Management - DOORS Next enables DOORS NG users to manage requirements variability using pure::variants. By coupling pure::variants and DOORS NG,

knowledge about variability and variants can be formalized, shared and automatically evaluated. This enables getting answers for questions about valid combinations of requirements in product variants quickly; permits easy monitoring of planned and released product variants at the requirements level and also permits very efficient production of variant-specific requirements documents out of the requirements repository.

The manual is available in online help inside the installed product as well as in printable PDF format. Get the PDF here.

## 1.1. About this manual

The reader is expected to have basic knowledge and experience with both tools, IBM Rational DOORS NG and pure::variants. Please consult their introductory material before reading this manual.

## 1.2. Software Requirements

### ...for pure::variants Connector for DOORS NG

The following software is supported by the pure::variants Connector for IBM Rational DOORS NG:

| | |
|---|---|
| IBM Rational DOORS NG: | IBM Rational DOORS NG 6.0.6.1 - 7.0.3 is required. Compatibility with other IBM Rational DOORS NG releases is not guaranteed. |

The pure::variants - Connector for IBM Engineering Requirements Management - DOORS Next is an extension for pure::variants and is available on all supported platforms.

IBM Rational DOORS NG and pure::variants communicate using the HTTP(S) protocol.

### ...for pure::variants Integration for DOORS NG

The following browser versions are at least supported by pure::variants Integration for DOORS NG:

- Firefox 54
- Internet Explorer 11
- Chrome 62
- Edge 15

The following software is required to use pure::variants Integration for DOORS NG:

| | |
|---|---|
| pure::variants Desktop Hub | 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 installer wizard. |

# 2. Installation

## 2.1. pure::variants (Administrator)

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**).

## 2.2. pure::variants Integration for DOORS NG (Administrator)

Please consult section **IBM Rational Doors NG Web Integration** 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** -> **IBM Rational doors NG Web Integration**).

## 2.3. Install pure::variants pvDesktopHub (Administrator)

pure::variants Integration for DOORS NG requires to communicate with pure::variants Desktop Hub. Please consult section **pure::variants Desktop Hub** 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 Desktop Hub**).

# 3. Using the Connector

## 3.1. 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.

## 3.2. Authentication

To use the the connector it is always required to be logged-in to the DOORS NG application. Currently there are two authentication mechanisms supported:

- Form-based
- OpenID Connect (for Single-Sign-On)

For both mechanisms the user will be prompted with a login dialog, which expects the user credentials. In case of Single-Sign-On a browser-based login dialog will be shown.

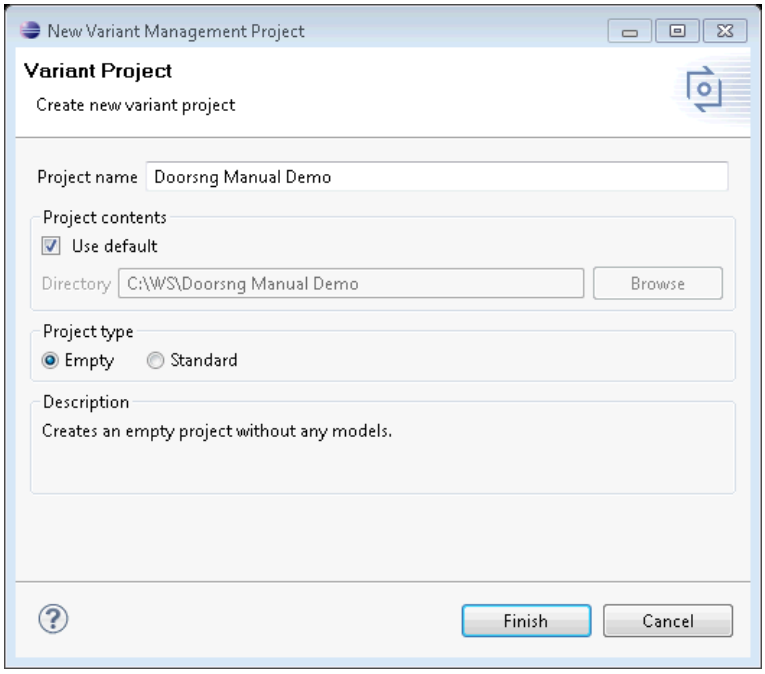
## 3.3. Creating the Initial Model(s)

The first step is always to create the corresponding family model for each relevant DOORS NG module. These initial family models serve as starting points for using existing variability information. The import procedure has to be executed **only once** for each DOORS NG module. Each module is represented by one pure::variants family model.

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 project type (see Figure 1, “Creating an empty Variant Management project for DOORS NG module import”)

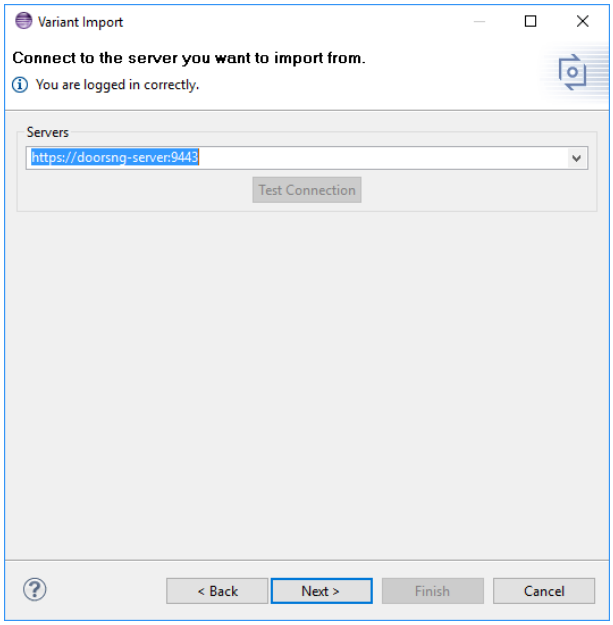
**Figure 1. Creating an empty Variant Management project for DOORS NG module import**




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 DOORS NG modules*.

The import wizard appears. With the first page you have to define or select the DOORS NG server address you want to import the modules from.

**Figure 2. The server selection page in the DOORS NG import wizard**

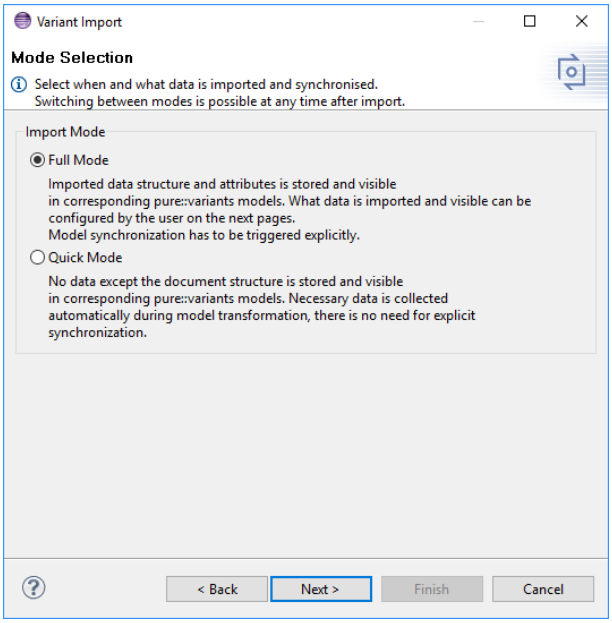



If you are not already authenticated, you can use **Test Connection**. This will open the login dialog and ask for your credentials. Alternatively you can pass the server selection page, so you will get ask when it is required.

**Figure 3. The login dialog in the DOORS NG import wizard**


With the second page you can decide whether you want to perform a full import of your DOORS NG Modules (**Full Mode**), or if you want just to import the module header (**Quick Mode**).

**Figure 4. The mode selection page in the DOORS NG import wizard**




By using **Full Mode** all data of the selected DOORS NG modules can be imported. Which data will be imported can be configured by the user on the next pages. The imported data is stored and visible in corresponding pure::variants models. The user can use the imported data to preview the variability of DOORS NG modules within pure::variants. The models need to be explicitly synchronized before transforming a variant.

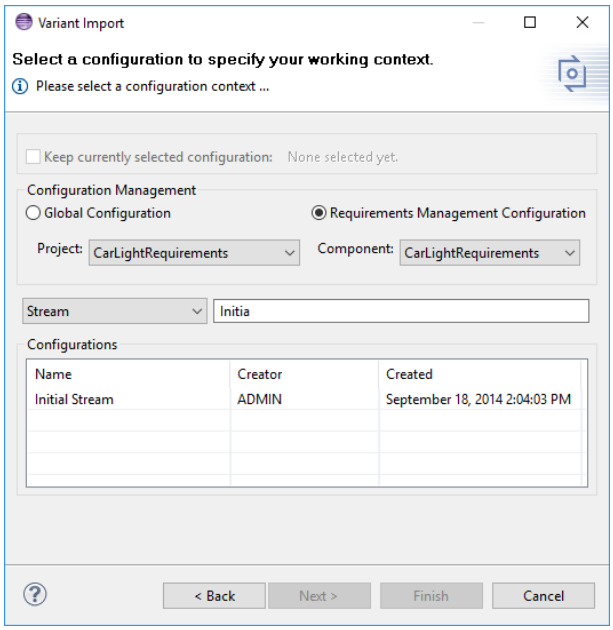
The Quick Mode is used to just import information pure::variants needs to access the DOORS NG modules during synchronization and transformation. The result are empty models in import. Each module is represented by a pure::variants model containing the link information to the related DOORS NG module. In **Quick Mode** there is no need to explicit synchronize the models imported from DOORS NG, this is done automatically during transformation.

Before selecting DOORS NG modules you have to define your configuration context. You can choose between a global (**Global Configuration**) and local configuration (**Requirement Management Configuration**), which can be of any type (Stream, Baseline, ...). Additionally, for a local configuration you have to select the project and its component where the configuration belongs to.

## Note

Choosing a component is only required since Jazz 6.0.3

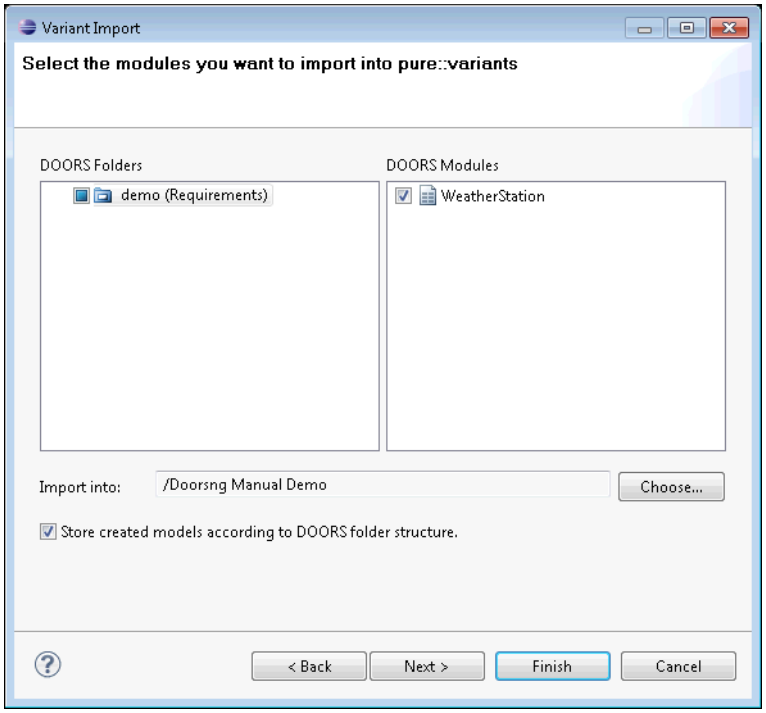
**Figure 5. The configuration selection page in the DOORS NG import wizard**




Going next to the module selection page, the complete project and folder structure of the DOORS NG repository is shown. Navigate to the folders containing the modules of interest and select the check boxes on the right side. Selecting a check box on the left side marks all modules inside this folder and its sub-folders for import. Make sure that the import target location given next to **Import into:** is correct. The location can be changed using the **Choose** button.

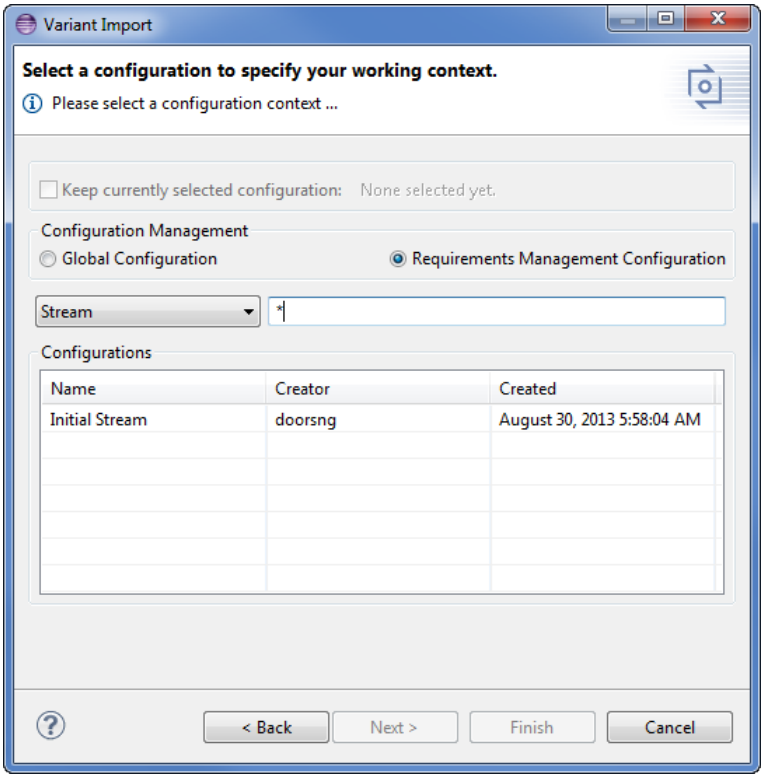
If **Store created models according to DOORS NG folder structure** is checked, the folder structure created in pure::variants will resemble the DOORS NG folder structure. Projects are treated as normal folders in this case. If unchecked, all modules are stored directly in the selected target location. *Use this only when all selected modules have unique names.*

**Figure 6. The module selection page in the DOORS NG import wizard**



The next page of the import dialog shows the selection page for the configuration, the module should be imported from. You can import the module from a Global Configuration or Requirements Management Configuration. Furthermore you can choose between stream, baseline and other configuration types.

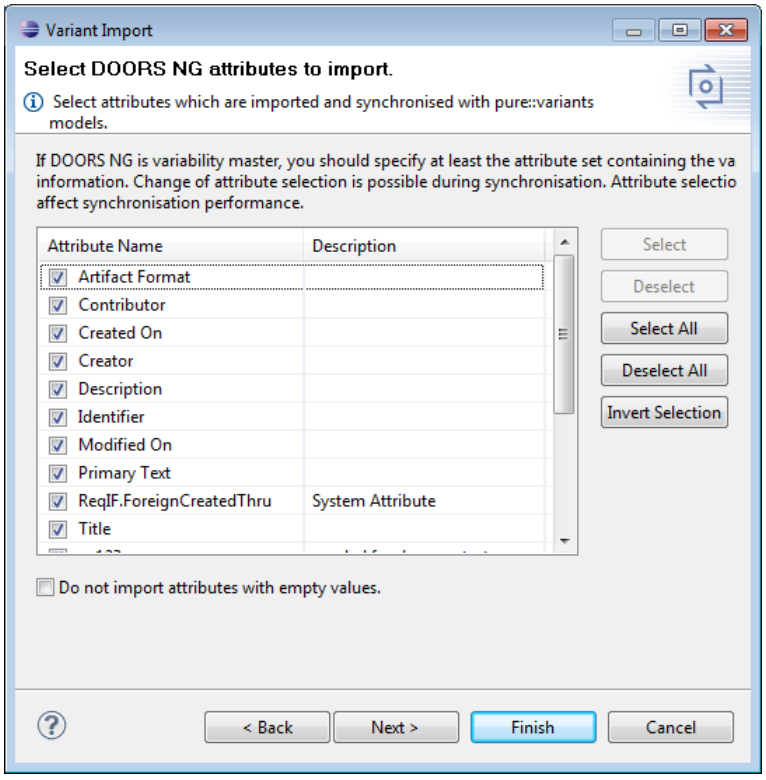
**Figure 7. DOORS NG Configuration to import from**




At the following page a list of DOORS NG attributes is shown. The attributes can easily be selected. Checked attributes will be imported, unchecked ignored.

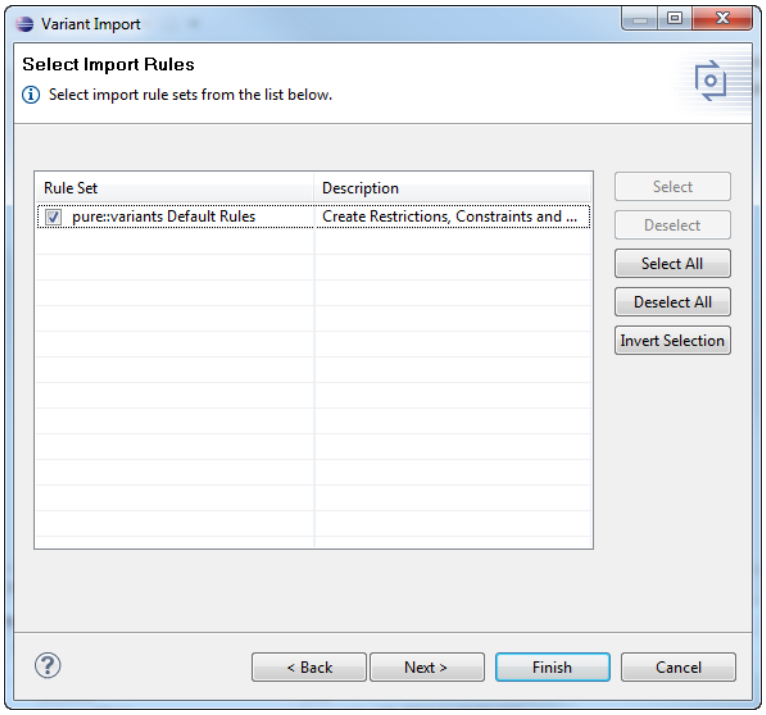
**Do not import attributes with empty values** ignores attributes with empty values during import, if checked.

**Figure 8. Set of DOORS NG attributes to import**




Pressing **Next** button brings the Import Rules page up. On this page you can select sets of Import Rules, which 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 DOORS NG module information.

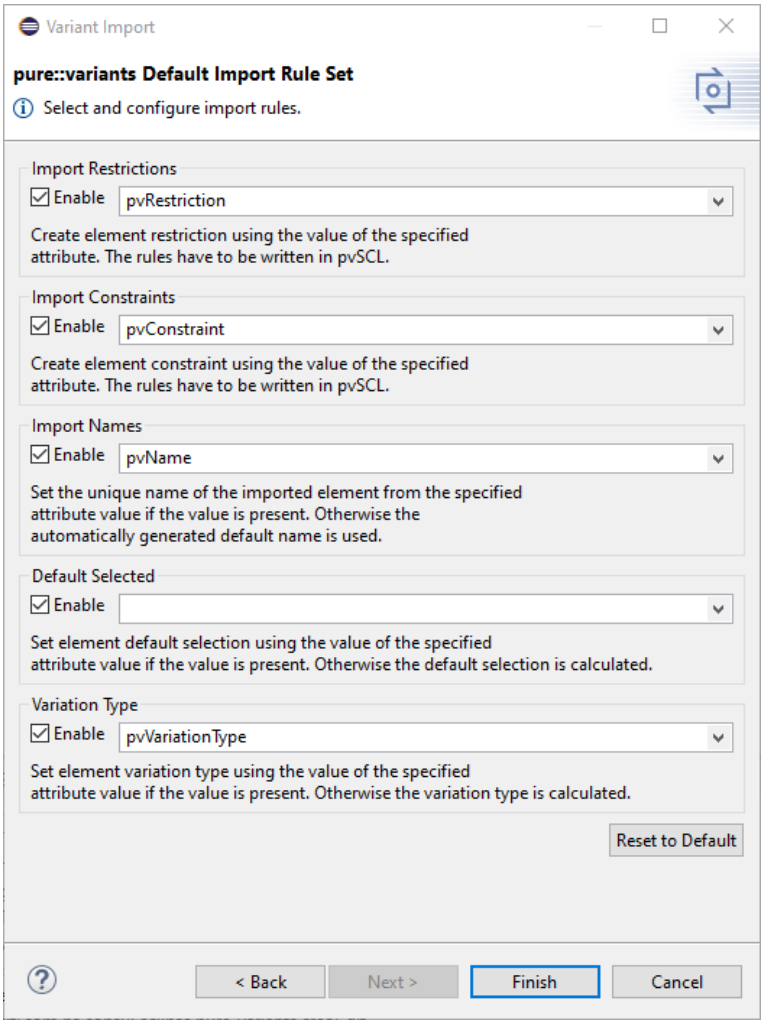
**Figure 9. Select Import Rules to use during import**




The last pages are showing the settings of the selected Import Rule Sets. For the **pure::variants Default Rules** you can choose which attribute value will be used for creating restrictions and constraints and setting the default

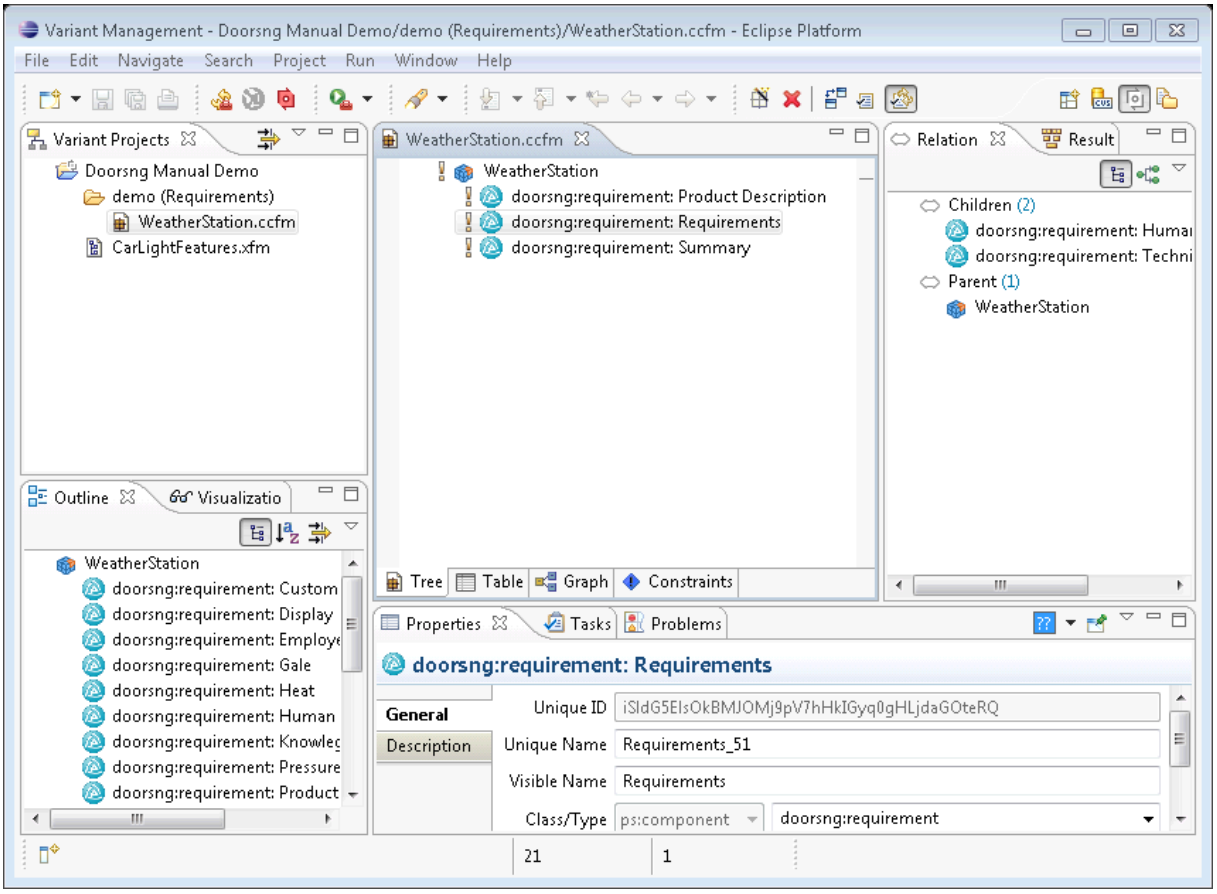
selection on elements and which attribute value will be used as unique name for elements. Also you can disable the creation of restrictions, constraints and using specific attribute for unique name, default selection and variation type.

**Figure 10. Settings for the pure::variants Default Import Rule Set**




The import result will be visible in the Variant Project view. If nothing shows up, use the item **Refresh** in the project's context menu (right mouse click) or press **F5** after selecting the project in the view. Each module is now represented by one pure::variants model. Models can be opened by double-clicking on them or selecting **Open** in the context menu. Figure 11, “Result of initial import of a DOORS NG module” shows the typical layout of a DOORS NG module after import.

**Figure 11. Result of initial import of a DOORS NG module**




The DOORS NG module is imported as family model and their requirements are represented as components. The elements follow the hierarchical structure as found in the original DOORS NG module.

If element attributes are not visible in your model view, you should enable attribute display via the context menu ( **Tree Layout** and select **Attributes**).

All requirement elements are imported as *mandatory* or default selected option (if a restriction was defined in DOORS NG) unless variability information was provided in the DOORS NG modules. This is explained in more detail in Section 5.1, “Adding Variability Information in DOORS NG”.

**Table 1. Overview of representation of DOORS NG entities in pure::variants**

| DOORS NG Entity | pure::variants Representation |
|---|---|
| Project | folder in project |
| Folder | folder in project |
| Module (Requirement Collection) | family model |
| Requirement | component in family model |
| Requirement title, Requirement short title | elements visible name (first non-empty value is used, long names are shortened) |
| Requirement attribute `pvRestriction` | element restriction in pvSCL language |
| Requirement attribute `pvConstraint` | element constraint in pvSCL language |
| Requirement attribute `pvVariationType` | element variation type. Valid input values are either `mandatory`, `or`, `optional`, `alterna\tive`, `ps:mandatory`, `ps:or`, `ps:optional`, or `ps:alternative`. |

| | |
|---|---|
| Requirement attribute `pvName` | element unique name. pure::variants will use the defined name as unique name. The name should be a valid unique name (see pure::variants User Guide for more information). If the name is not valid, pure::variants will generate a valid name from it. Uniqueness is not enforced during import, but later shown as model problem in the model editors. |
| Requirement attribute `pvDefaultSelected` | element default selection state. Valid input values are either `off` or `on`. Case is irrelevant, so `OFF` or `Off` are also valid input values. |
| Requirement id | element attribute `identifier` |
| Requirement attribute | element attributes with type ps:integer or ps:string |

## 3.4. Using Variability Information from DOORS NG Modules

During the synchronization, pure::variants can optionally use some information present in the DOORS NG modules to create the pure::variants model representation. During synchronization all information including element hierarchy, restrictions and constraints are compared and if different from the information stored in the pure::variants model, shown as mergeable difference.

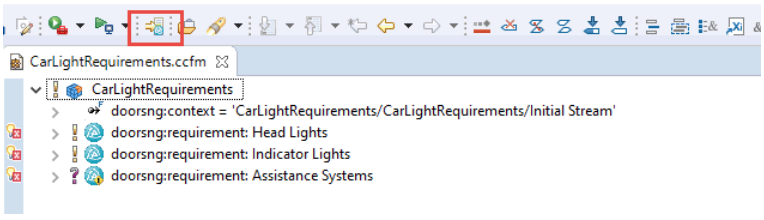
Which and how variability information can be represented in DOORS NG modules is explained in more detail in Section 5.1, “Adding Variability Information in DOORS NG”.

## 3.5. Synchronize all Restrictions to DOORS NG

If restrictions are changed in pure::variants, they can be written back to the DOORS NG module in order to keep the DOORS NG module and the imported family model in sync. For this, a synchronization has to be triggered that writes all changed restrictions available in pure::variants back to DOORS NG. During this synchronization, the restrictions are written to the attribute responsible for restrictions as defined in the **pure::variants Default Rule Set**. Thereby following changes are applied to the DOORS NG module:

- For an existing restriction at a requirement in the DOORS NG module
  i. the restriction remains unaffected if the restriction is equal to the restriction in pure::variants
  ii. the restriction is changed if the restriction is different from the restriction in DOORS NG
  iii. the restriction is removed if a restriction is missing

**Figure 12. Synchronize all restrictions button**



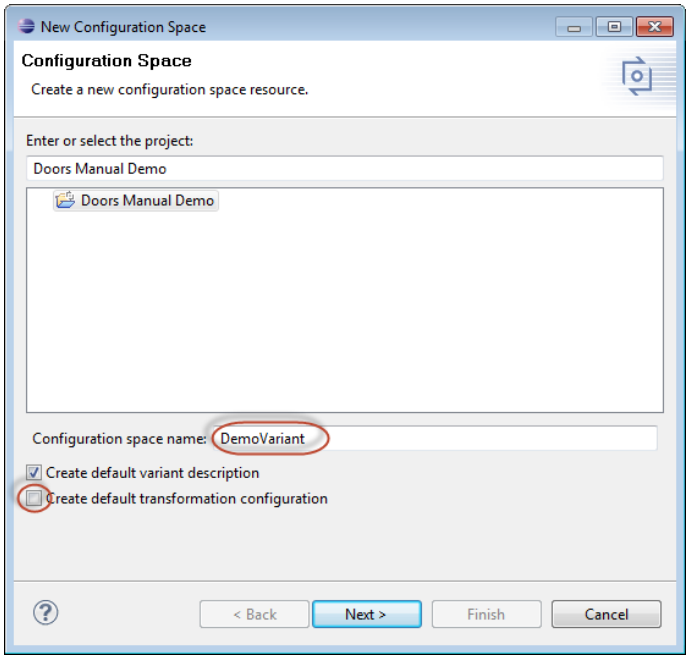

Before writing all feature mappings back the DOORS NG module, the user will be asked for confirmation if all existing restrictions in the DOORS NG module shall be updated.

## 3.6. 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.

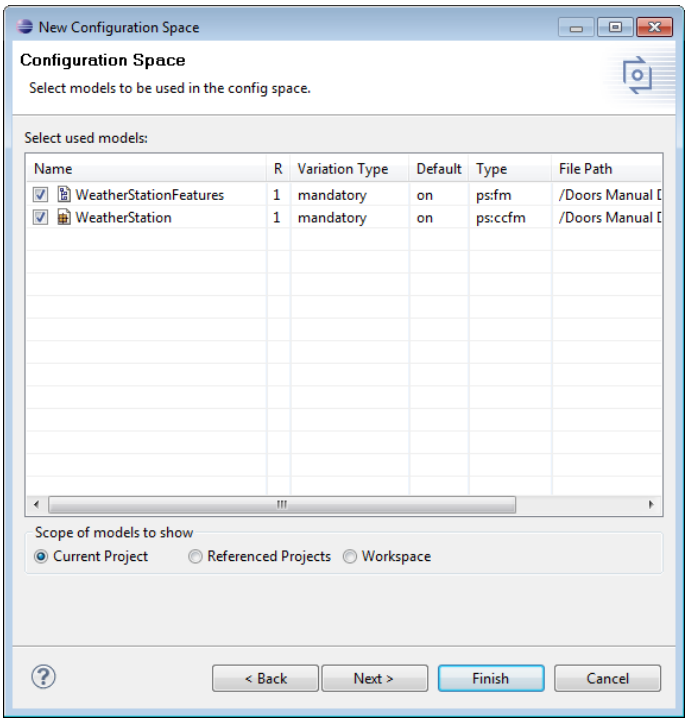
Variants are stored as separate entities called *Variant Description Models* (VDM). A VDM always belongs to a specific *Configuration Space*. Thus before defining variants, a configuration space has to be created. Select the project containing the imported models in the Variant Projects view and open the context menu. Below the item **New** select **Configuration Space**. A wizard is opened. On the first page (Figure 13, “The Configuration Space Wizard, page 1”), 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 13. The Configuration Space Wizard, page 1**



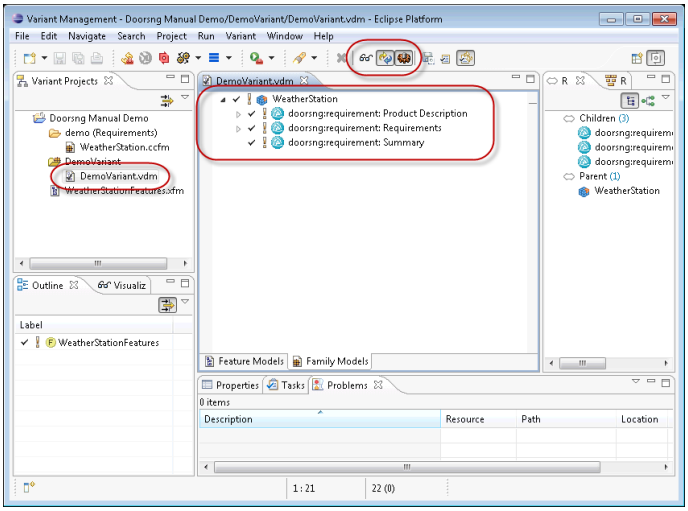

The next page is used to specify which models are to be included in this configuration space. Select here all models that represent the DOORS NG modules of interest. In the example below just one model is selected. Now press the **Finish** button.

**Figure 14. The Configuration Space Wizard Model Selection Page**




The resulting project structure is shown in Figure 15, “Initial Configuration Space Structure”. The `DemoVariants.vdm` is created and immediately opened. It resembles the structure of the previously defined model(s), but has a check-box in front of each element to permit the user to select elements for this variant by clicking on it. The buttons marked in the tool-bar control the evaluation of configurations. The left-most button ( ) initiates a manual check of the variant configuration. The middle button ( ) toggles between manually checking and automatic checking after changes to the VDM. Finally the right-most button ( ) toggles the auto-resolver on or off. The auto-resolver provides automatic resolution of configuration problems where possible.

**Figure 15. Initial Configuration Space Structure**




**Tip:** For small to medium sized models (up to several thousand elements, depending on the speed of the processor), it is convenient to turn on both auto-checking and auto-resolving by clicking on the respective tool-bar buttons.

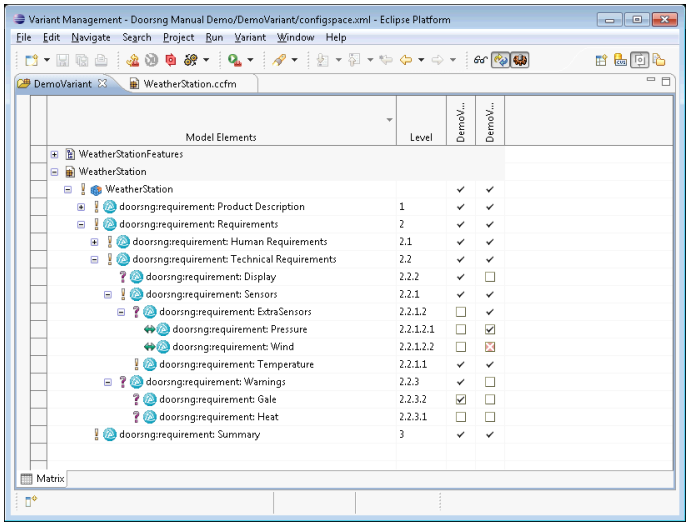
Problems may be indicated by pure::variants during the selection of requirement elements. There are several places where problems are shown. Firstly, the Problems view (usually located in the lower right part of the pure::variants

perspective) lists all problems such as incompatible elements or a missing selection from *alternative* elements. In addition, problems are shown in the model editor directly in front of the element causing the problem. A tooltip (which can be seen by moving the mouse over the icon) explains the problem, and the context menu for the problem (right mouse button) provides possible fixes for the problem. E.g. for conflicting elements the fix is to deselect either one or the other element.

Each variant can be represented in its own VDM. To create a new VDM, either select **Clone** from the context menu (in Variant Project view) of an existing variant or use **New->Variant Model** in the context menu of the configuration space. When a valid variant is configured, it can be stored and exported to DOORS NG. The next section explains this in detail.

Variants may be compared at the element level by using the matrix editor (see Figure 16, “Matrix Editor for comparing variants” ). This editor is activated by double-clicking on the enclosing configuration space icon. The **Table Layout** item in the context menu can be used to customize the list of variants to compare and the **Show Elements...** and **Filter** items in the same menu can be used to select elements of interest.

**Figure 16. Matrix Editor for comparing variants**




## 3.7. Transforming a Variant

Variants stored in a variant description model can be made available in DOORS NG. The Connector supports multiple ways of representing variants: *attribute-based*,*module-based*,*linked-based* and *stream-based*.

### Attribute-Based Variant Representation

In the attribute-based representation we define an attribute for DOORS NG requirements which are part of the module. This transformation modus adds the name of the variant if the requirement is part of the variant. The name of this attribute can be user defined. Default is *pvVariants*.

### Module-Based Variant Representation

The module-based representation creates variant-specific copies of each module in a designated DOORS NG folder, using the folder structure of the original DOORS NG modules. Only those modules that are included in the configuration space as pure::variants models are considered.

pure::variants recreates the folder structure from the import project in the output folder per default. The module paths are recreated relatively to the project the modules are located in.

In case this is not sufficient customized output paths can be used. To specify the path, where the module will be copied relative to the variant root folder an attribute is added to the root element of the corresponding family model. The attributes name is *doorsng:outputPath*.

For defining an output path which specifies a custom module name as well, please create the attribute *doorsng:outputModulePath* on root element of the appropriate family model instead. The last path segment is intepreted as the resulting DOORS NG module's name.

The *doorsng:outputModulePath* takes precedence over the *doorsng:outputPath* attribute.

## Link-Based Variant Representation

In the link-based representation each variant is represented by a single DOORS NG module with links to all requirements included in the variant. The requirements to include are defined by the list of requirement element selected in the respective variant description model.

## Stream-Based Variant Representation

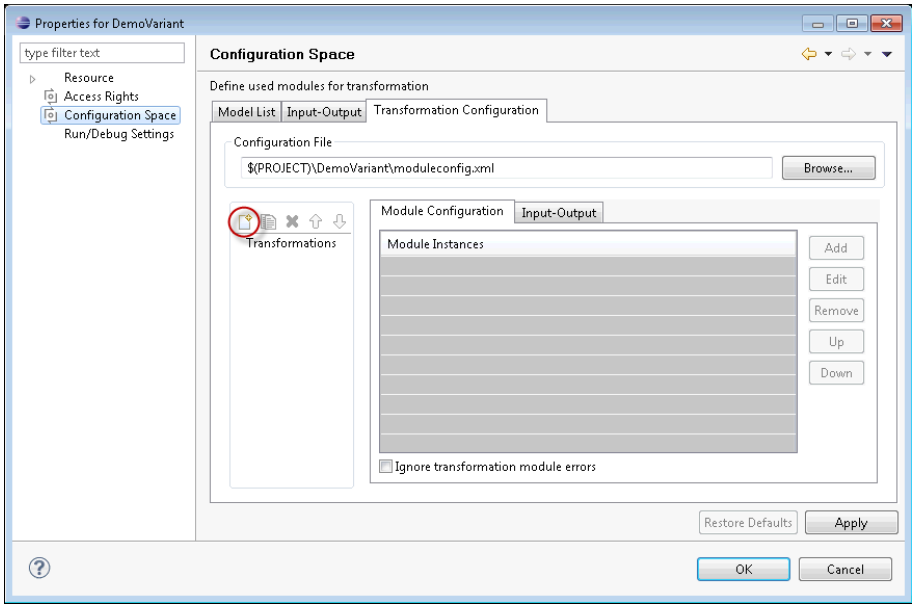
In the stream-based representation each variant is represented by a new DOORS NG stream. The new variant stream is derived from the stream/baseline the DOORS NG module was imported from. The DOORS NG modules of the ConfigSpace are being considered while transformation. While transformation, these DOORS NG module being part of the variant, are variant-specifically tailored on this variant stream. Other DOORS NG modules, being not part of the variant, are deleted on this variant stream.

## Preparing a Transformation

To transform a variant, first a *Transformation Configuration* has to be created. To create a Transformation Configuration open pull down menu of **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 (see Figure 17, “Transformation Configuration”). Now add a new Module to the Module Configuration, using the **Add** button.

**Figure 17. Transformation Configuration**




From the opened dialog, choose **IBM Rational DOORS NG Module** and enter a name. The next page shows all parameters. The *Modus* parameter specifies one of the variant result representations, as described above. One of the following modes can be selected:

- *Variant Enumeration* (see the section called “Attribute-Based Variant Representation”) adds the current variant for selected requirements to the enumeration attribute.

- *Copy with Duplicates* (see the section called “Module-Based Variant Representation”) instantiates a new module including all variant specific requirements.

- *Copy with Links* (see the section called “Link-Based Variant Representation”) instantiates a new module including all variant specific requirements as links.

- *Create Streams* (see the section called “Stream-Based Variant Representation”) derives a new stream - the variant stream - from the initial stream. The initial stream represents your product line and is chosen in Import resp. Synchronization wizard (see Figure 19, “Synchronize model”). In context of the variant stream, the DOORS NG modules involved in the transform process are tailored accordingly to the selection configuration. For the support of Global Configurations, the Global Configuration Transformation module must be configured additionally, see *pure::variants OSLC Base Components Manual* of the *pure::variants - OSLC Base Feature* feature extension.

**Table 2. *Modus*-related transformation module parameters:**

| Parameter | Description | Variant Enumeration | Copy With Duplicates | Copy With Links | Create Streams |
|---|---|---|---|---|---|
| Username | The user name for connecting to DOORS NG server. | Optional | Optional | Optional | Optional |
| Password | The password for connecting to DOORS NG server. | Optional | Optional | Optional | Optional |
| VariantRoot | The folder or project path the variant should be exported to. | Unsupported | Required | Required | Unsupported |
| PerformPartial-TextSubstitution | If **true** is selected, the partial text substitution is performed. | Unsupported | Optional | Unsupported | Optional |
| Name | Specifies the name for the enumeration attribute. If not set, the standard name (*pvVariants*) is used. Please note that the attribute with the defined name and type **String** needs to be existing in your DOORS NG requirement type before the transformation is started. | Optional | Unsupported | Unsupported | Unsupported |
| Cleanup | If **true** is selected, all existing variant attributes are removed be- | Optional | Unsupported | Unsupported | Unsupported |

| Parameter | Description | Variant Enumeration | Copy With Duplicates | Copy With Links | Create Streams |
|---|---|---|---|---|---|
| | fore exporting the current variant. | | | | |
| StreamName | Specifies the name for the derived variant stream. The default stream name is a concatenation of the variant name and the date and time the transformation was started. | Unsupported | Unsupported | Unsupported | Required |
| Name of Changeset | The name of the changeset. If the value is empty the default name "pure::variants Tailoring of $(VARIANT)_$(QUALIFIER)" is used. | Optional | Optional | Optional | Optional |
| Deliver Changeset | If set to "false" the changeset is not delivered to the stream. | Optional | Optional | Optional | Optional |
| Align Linkage | If set to "true" the links between artifacts of DOORS NG modules are aligned between variant modules.<br><br>This parameter does not permit the usage of doorsng:outputModulePath property in family models.<br><br>The output folder for DOORS NG modules must be same, if the linkage between its requirements should be aligned. (If used in conjunction with doorsng:outputPath attribute) | Unsupported | Optional | Unsupported | Unsupported |
| WorkItem | If set to valid work-item URI, | Optional | Optional | Optional | Optional |

| Parameter | Description | Variant Enumeration | Copy With Duplicates | Copy With Links | Create Streams |
|---|---|---|---|---|---|
| | the work-item is linked to the changeset, created by the transformation module. | | | | |
| Soft Delete | If set to "true", the requirements will be removed from the requirements module but still kept in the project's variant stream.<br><br>By default requirements inside of the project's variant stream are permanently deleted if they are not used in any other module. | Unsupported | Optional | Unsupported | Optional |
| RedactMode | If set to "RedactHighestExcluded", the top-level requirements (usually the headline requirements) that are exluded in variant are redacted by replacing title/name with placeholder text (see "RedactText" value). | Unsupported | Unsupported | Unsupported | Optional |
| RedactText | Define the placeholder text for the redacted requirement, using "RedactHighestExcluded" mode as redact mode. | Unsupported | Unsupported | Unsupported | Optional |

For automatic generation of the stream name all standard variant path variables can be used. There are three additional variables

- $(BASELINE)
  - Adds the name of the Baseline, which was used during module import or the name of the baseline, which is created as source for the new stream during stream creation.
- $(COMPONENT)

  - Adds the name of the Component the imported module is located in.

- $(STREAM)
  - Adds the name of the Stream, which was used during module import.

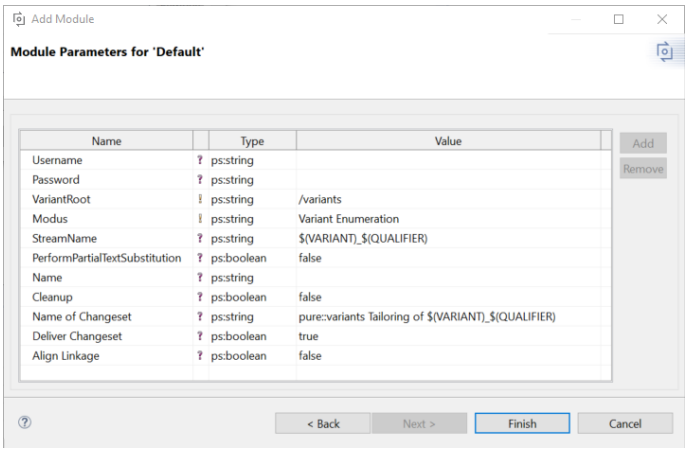
### Note

You can use *User Parameters* to refer to a user defined parameter value.

E.g. You can define a user parameter, like *WORKITEM*, in scope of your transformation module configuration, and refer to its value by writing $(PARAM:WORKITEM).

The value for *WORKITEM*parameter is queried by transformation framework in the beginning, once.

**Figure 18. Module Parameter Page**




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

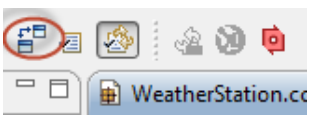
## Web Client Integration for transformation

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.

# 3.8. Updating Models from DOORS NG

Since there is no live connection between the DOORS NG database and pure::variants, it is necessary to update the pure::variants models with information from DOORS NG whenever relevant changes have been made. To facilitate the synchronization, pure::variants provides a **Synchronize** action. To start the update, open the model representing the DOORS NG module and press the **Synchronize** button in the tool bar (see Figure 19, “Synchronize model”). pure::variants will connect to DOORS NG and present the so called Compare Editor for pure::variants models (see Figure 20, “Model update from DOORS NG in Compare Editor”).

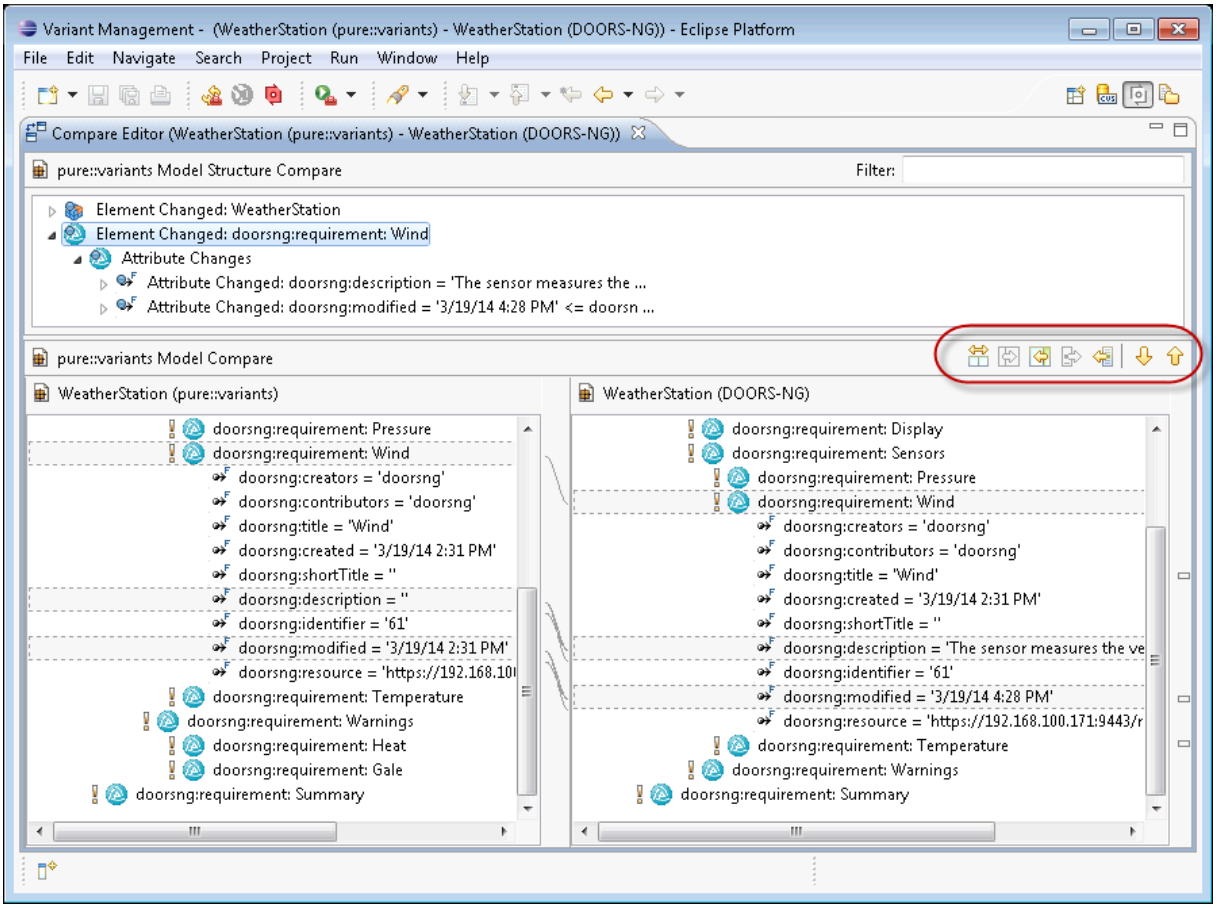
**Figure 19. Synchronize model**




The compare editor is used throughout pure::variants to compare model versions but in this case is used to compare the DOORS NG 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 attribute values are marked with boxes and connected with their respective counterparts in the other model.

**Figure 20. Model update from DOORS NG in Compare Editor**




The Merge tool-bar (see marked area between upper and lower editor windows at right) provides tools to copy single or even all (non-conflicting) changes as a whole from the DOORS NG model to the family model.

# 4. Using Integration

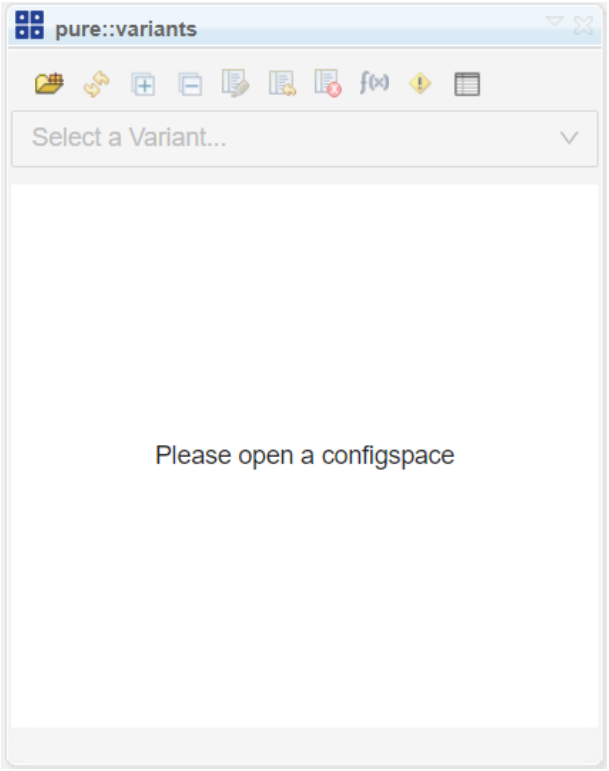
In order to facilitate the pure::variants - Connector for IBM Engineering Requirements Management - DOORS Next to add variability information to requirements of DOORS NG modules, an in-tool integration for DOORS NG is provided called *pure::variants Integration for DOORS NG*. For brevity, moving on we will refer to pure::variants Integration for DOORS NG as *Integration*.

Once the Integration has been added to the DOOR NG (see: section **Add pure::variants widget to Doors NG** 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. Once the desired mode has been selected, every other time the Integration is re-added (after removing) or reloaded/refreshed the *Main* page of the Integration will always be loaded, as shown in Figure 21, “Integration *Main* page view with no config-space selected”.

## Note

*The mode settings are browser dependent settings. If the browser changed or reset, the settings will be required again.*

**Figure 21. Integration *Main* page view with no config-space selected**



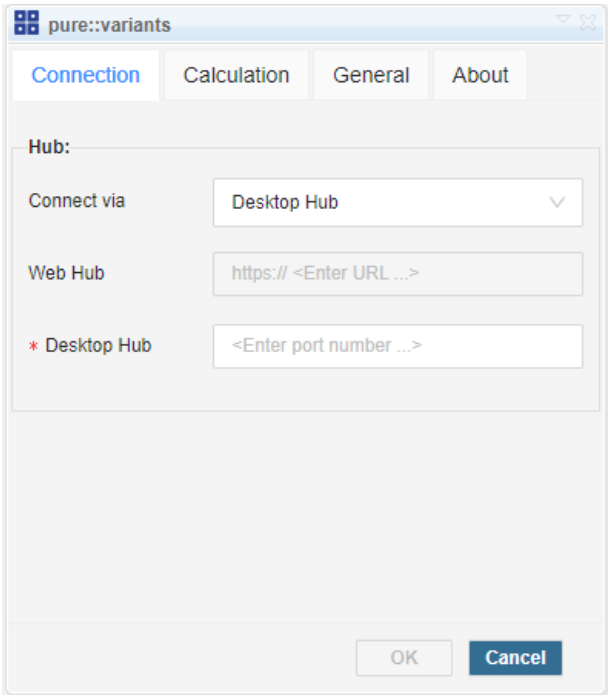

## 4.1. Prerequisites

### Desktop Hub

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 (see: Figure 22, "*Desktop Hub* configuration view"). Integration will then redirect to its *Main* page and start running in the Desktop Hub mode.

**Loading Configuration Space In Desktop Hub Mode:** In order to select a *Config-space* please press the **Open Config Space** button from the Integration's menu bar. The Desktop Hub's file selection dialog shows to select the desired *Config-space*. Once the *Config-space* is selected, the Integration will immediately show the selected *Config-space*.

**Figure 22. *Desktop Hub* configuration view**




## Web Hub

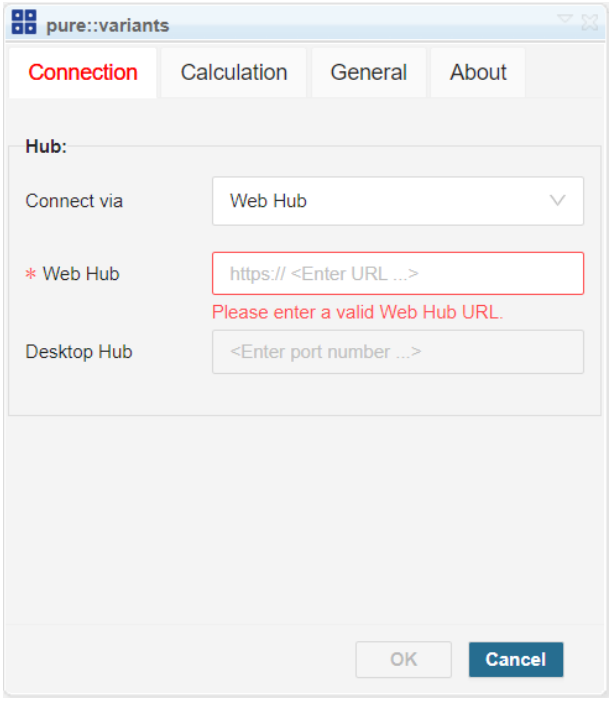
As discussed above, Integration can only run in one of the two possible modes. Hence, In order to run the Integration in the *Web Hub* mode, a running instance of the *pure::variants Web Components* is required (see: *pure::variants Web Components Manual* pdf file). While the *pure::variants Web Components* 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. Afterward, press the **OK** button so to save the *mode* settings (see: Figure 23, "*Web Hub* configuration view"). Integration will then redirect to its *Main* page and start running in the Web Hub mode.

**Authentication:** Since the *pure::variants Web Components* is only accessible via authenticated user, the *login* page might show up first. However, once successfully logged-in, the Integration redirects back to its *Main* page.

### Note

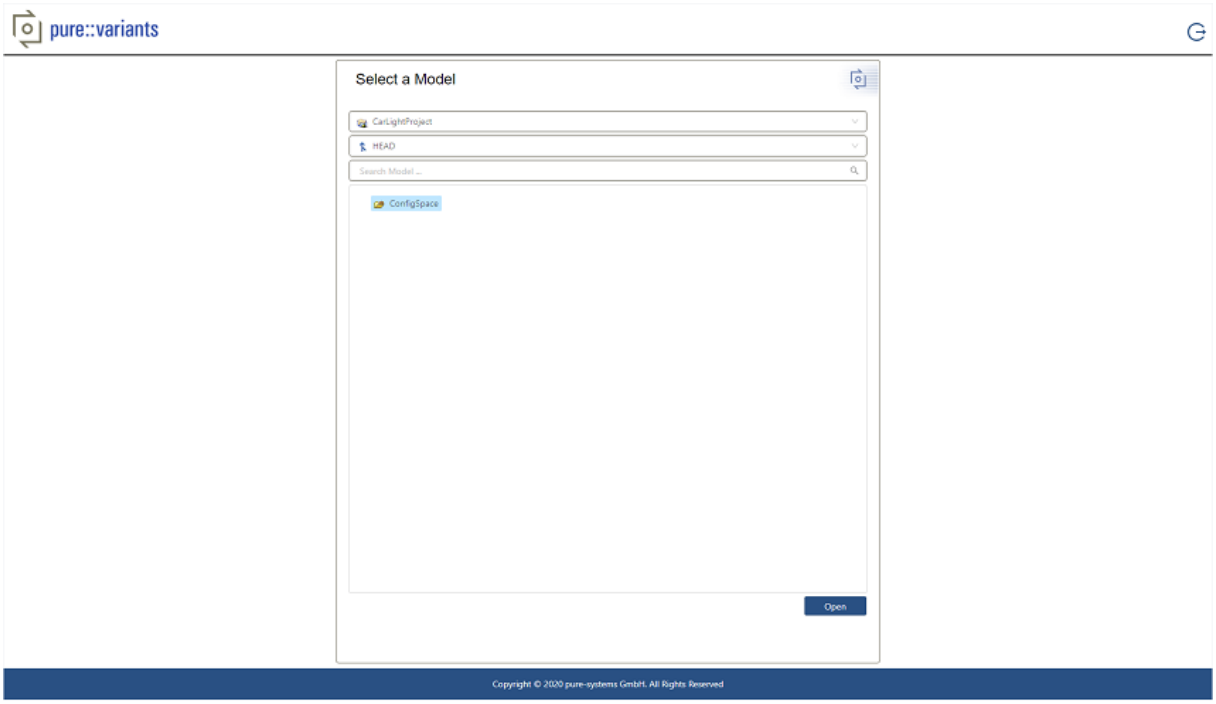
*If redirection to Main page fails, please ensure that your browser settings to access the third-party cookies are enabled.*

**Figure 23. *Web Hub* configuration view**




**Loading Configuration Space In Web Hub Mode:** To select a *Config Space* click the **Open Config Space** button from the Integration's menu bar. A new window called *pure::variants Model Picker* will be opened. Please select a *Project* first, choose a *Revision*, and then select your desire *Config Space* folder. Finally, press the **Open** button in order to open the chosen *Config Space* inside Integration (see: Figure 24, “pure::variants Model Picker”). The *pure::variants Model Picker* window will automatically close and the chosen *Config Space* will be shown inside the Integration.

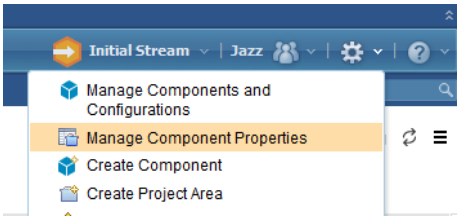
**Figure 24. pure::variants Model Picker**



## (Required) Define PVSubstitutionMarkers attribute

It is required to create a attribute called **PVSubstitutionMarkers** in the DOORS NG requirement module type. Therefore, navigate to **Administration**-> **Manage Component properties** (or **Manage Project Properties**) as shown in Figure 25, “Navigate to project/component properties”. In the tab **Artifact Attributes**, define the **PVSubstitutionMarkers** attribute of data type **String**.

**Figure 25. Navigate to project/component properties**

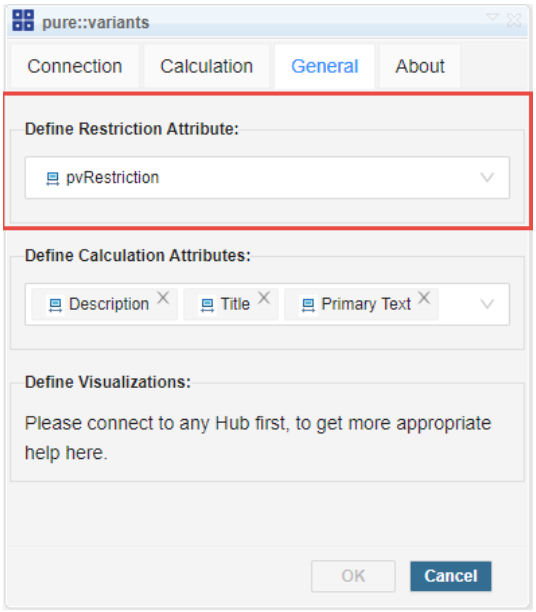



When defining a URI for the newly created attribute, the last path segment must be equal to *PVSubstitutionMarkers* (e.g. "http://company.xyz/types/PVSubstitutionMarkers"). After defining the **PVSubstitutionMarkers** attribute, switch to the **Artifact Types** tab and add the **PVSubstitutionMarkers** attribute to your requirement module type(s).

## (Required) Define pvRestriction attribute

In order to associate the restrictions with the requirements, the DOORS NG type model of the requirements needs to be extended. A custom *pvRestriction* attribute needs to be created and associated with the requirement type. Therefore, navigate to **Administration**->**Manage Component properties** (or **Manage Project Properties**) as shown in Figure 25, “Navigate to project/component properties”. In the tab **Artifact Attributes**, define any custom named attribute e.g. *SystemFeature* of data type **String**. After defining the attribute, switch to the **Artifact Types** tab and add the newly custom created attribute to the requirement type(s).

**Figure 26. General Settings**

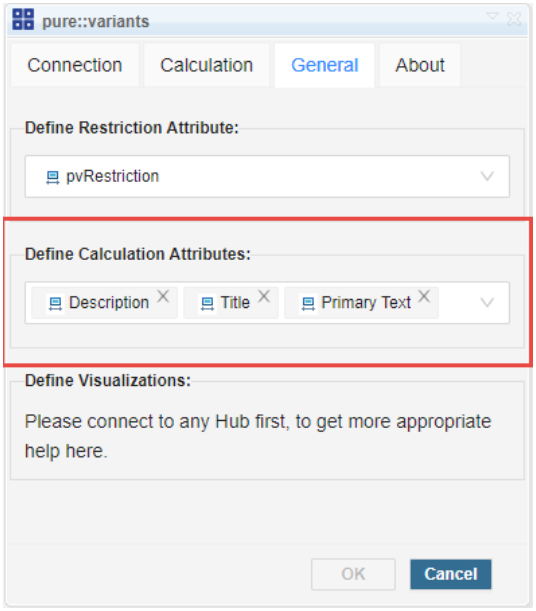



Finally, refer to this newly created attribute from pure::variants Integration for DOORS NG by pressing the Settings ( ) menu button and navigate to the **General** tab as shown in the figure Figure 26, “General Settings”, enter the name of the newly created attribute (if no attribute name is defined **pvRestriction** is consider as default) and then press **OK**, this stores the name to the **PVSubstitutionMarkers** attribute. Therefore, **PVSubstitutionMarkers** needs to be created upfront (see: the section called “(Required) Define PVSubstitutionMarkers attribute”).

## (Optional) Define Calculation attributes

In order to substitute various attributes of requirements, the user can configure the selection of attributes on the General settings page. This attribute selection will be considered while preview and transformation.

**Figure 27. General Settings**




By default, only the **Calculations** present in **Contents**, **Primary Text** and **Description** columns are replaced with the calculated values for a DOORS NG Requirement. You can select any other attribute, which is provided by the dropdown menu.

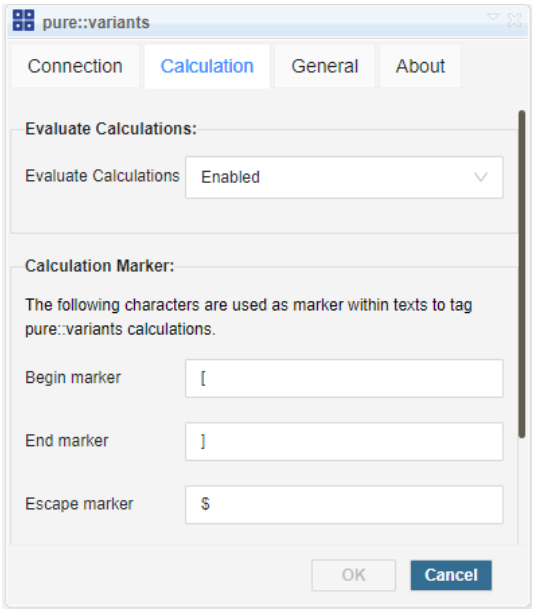
## (Optional) Define custom substitution markers

The writing of calculations to requirements is working out of the box. The standard markers are as follows:

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

If this does not fit the requirements text, defining custom markers is also possible via pure::variants Integration for DOORS NG. This requires a custom attribute in your *DOORS NG requirement module* type called **PVSubstitutionMarkers** (see: the section called “(Required) Define PVSubstitutionMarkers attribute”). The editing of custom markers can be done in the pure::variants Integration for DOORS NG by opening Settings ( ) and navigating to **Calculation** tab as shown in figure Figure 28, “Calculations Settings”.

**Figure 28. Calculations Settings**

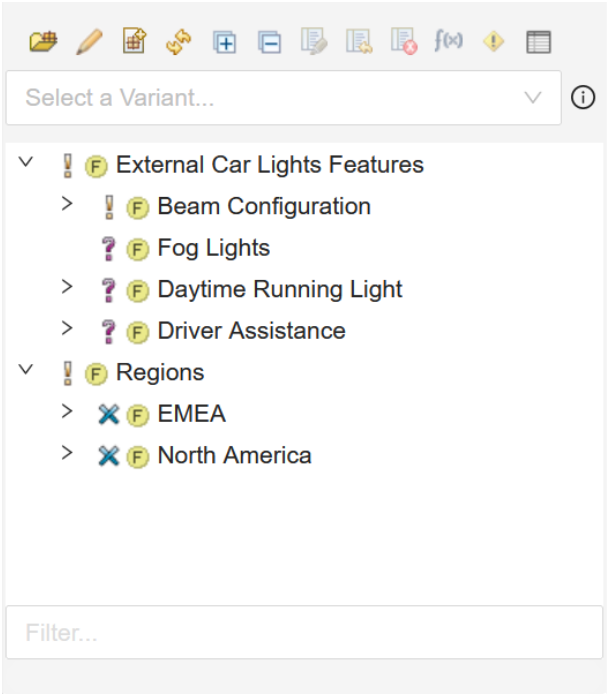



On the **Calculation** tab, if the **Evaluate Calculations** is set to **Enabled** then the calculations in the preview mode will be substituted by its calculated values.

## 4.2. Introduction to *Integration* GUI

The *Main* page view of the Integration is shown in Figure 29, “Integration *Main* page view”

**Figure 29. Integration *Main* page 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 (see: the section called “Desktop Hub” and the section called “Web Hub”).

### 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 app. (Only visible in the Web Hub mode)

***3.*** **Import/Synchronize** button ( or ) - 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 (see: Section 4.6, “Visualizing Variability Information (Preview)”)

***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 attributes of *DOORS NG Requirement* (see: Section 4.4, “Working with Calculations Editor”).

***11.*** **Restriction** button ( ) - click to open the *Restriction* page, so to edit the restriction inside the *pvRestriction* attribute column of the selected *DOORS NG Requirement* (see: Section 4.3, “Working with Restriction Editor”).

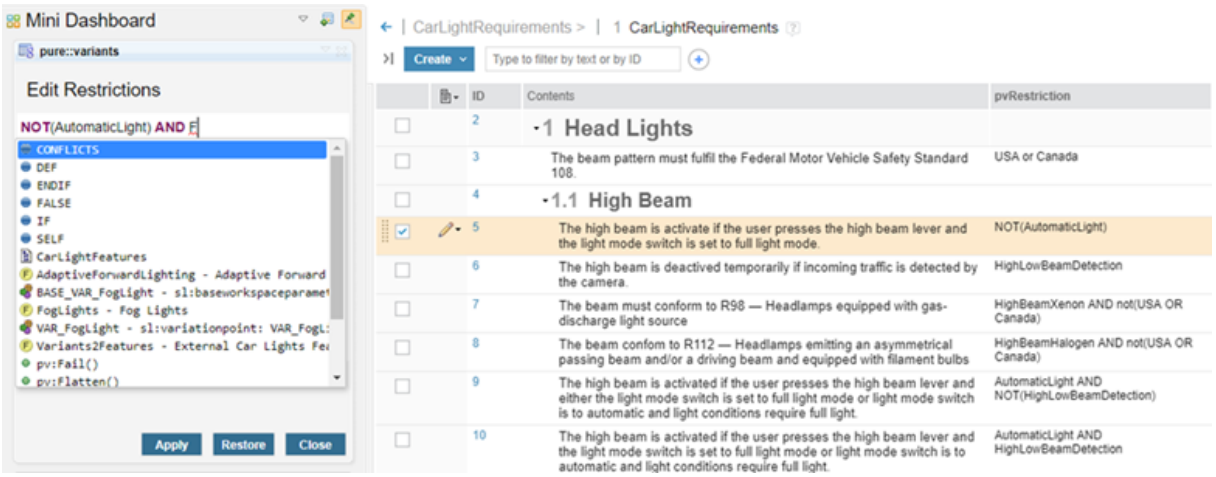
***12.*** **Settings** button ( ) - click to navigate to the *Settings* page so to configure the *General* settings, *Calculations* specific settings, and, also to see the Integration specific information.

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).

## 4.3. Working with Restriction Editor

The *Restriction Editor* can be opened by clicking the icon. Edit a restriction in *Restriction Editor* either by selecting a requirement from a list of requirements or by opening a requirement. The *Restriction Editor* provides the ability of auto-completion proposals and systax highlighting while editing restrictions for DOORS NG requirements.

**Figure 30. Restriction Editor of pure::variants Integration for DOORS NG**

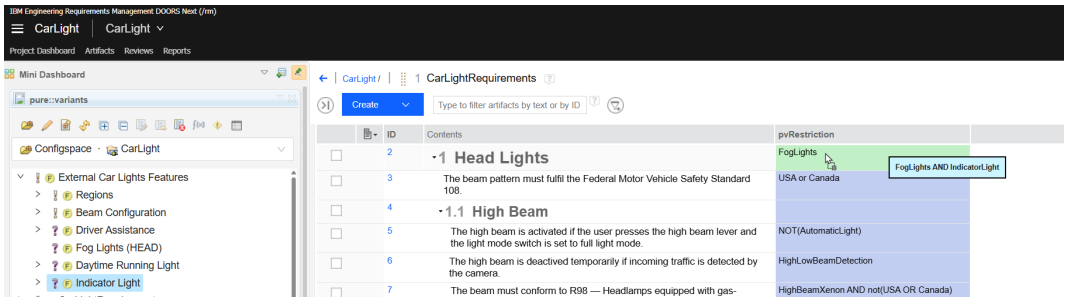



For a more intuitive and simpler way of adding variability to a requirement, drag'n'drop functionality can be used. Drop of single or multiple features as restriction is allowed.

- 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 requirement.

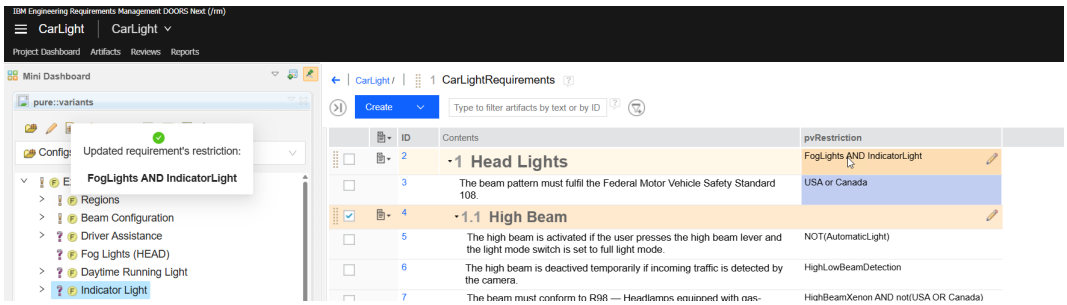
By default, the dropped restriction concatenates with the existing pvSCL expression using "AND". If the CTRL key is held during the drop action, the existing pvSCL expression will be concatenated using 'OR' instead. To provide clarity on the resulting restriction after the drop, a preview is displayed.

**Figure 31. Drag a single feature to add a restriction**



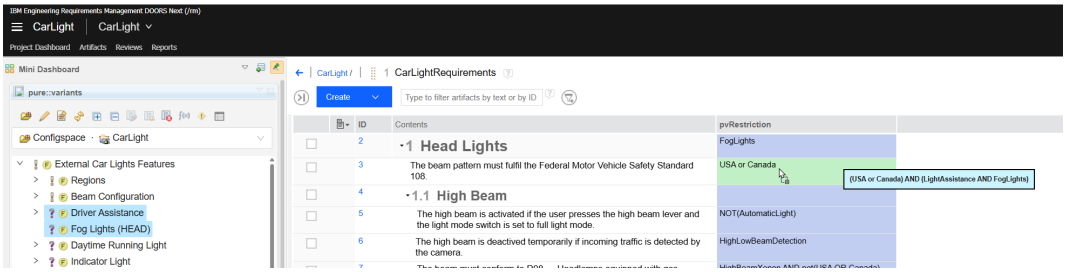

Restriction is added to the tracker item.

**Figure 32. Drop a restriction on tracker item**




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

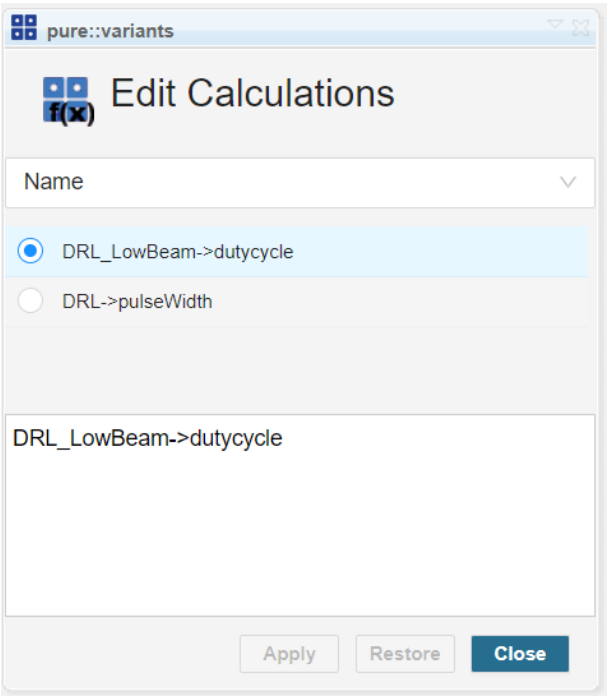
**Figure 33. Drag multiple feature to add a restriction**




## 4.4. Working with Calculations Editor

*Calculations Editor* can be used to edit the calculations present in the configured substitutable attributes which are available for a *DOORS NG Requirement*. You can open it by clicking the f(x) icon. Calculations can be edited by selecting a requirement in a module or opening a requirement from a module. In *Calculations Editor* select the attribute of a requirement that contains the calculations markers. After selecting an attribute, all the calculations in that attribute 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.

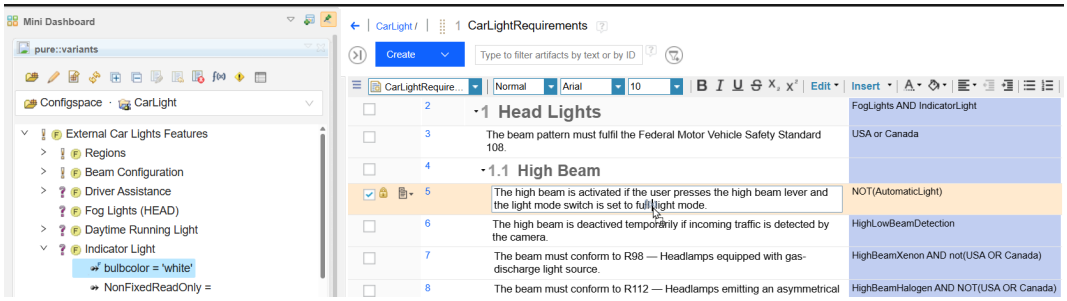
**Figure 34. Calculations Editor of Integration**




The Calculation editor will show only the substitutable attributes, which were configured on General settings page. Please note, to add the open and end marker into the specific attribute, to be editable in Calculation editor.

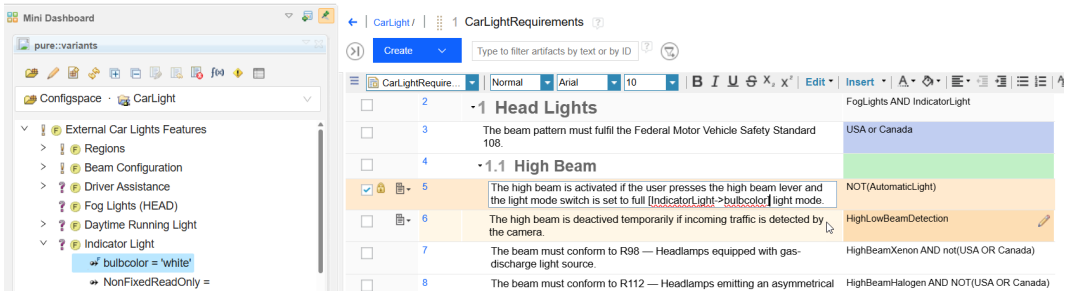
Calculations to a requirement can also be easily added by using drag'n'drop functionality. Therefore, the user must first activate the inline edit mode for the requirement. Alternatively, the requirement can be opened in edit mode.

**Figure 35. Drag an attribute to add a Calculation**




Calculation will be added at the respective position.

**Figure 36. Drop the Calculation on editable tracker item field**

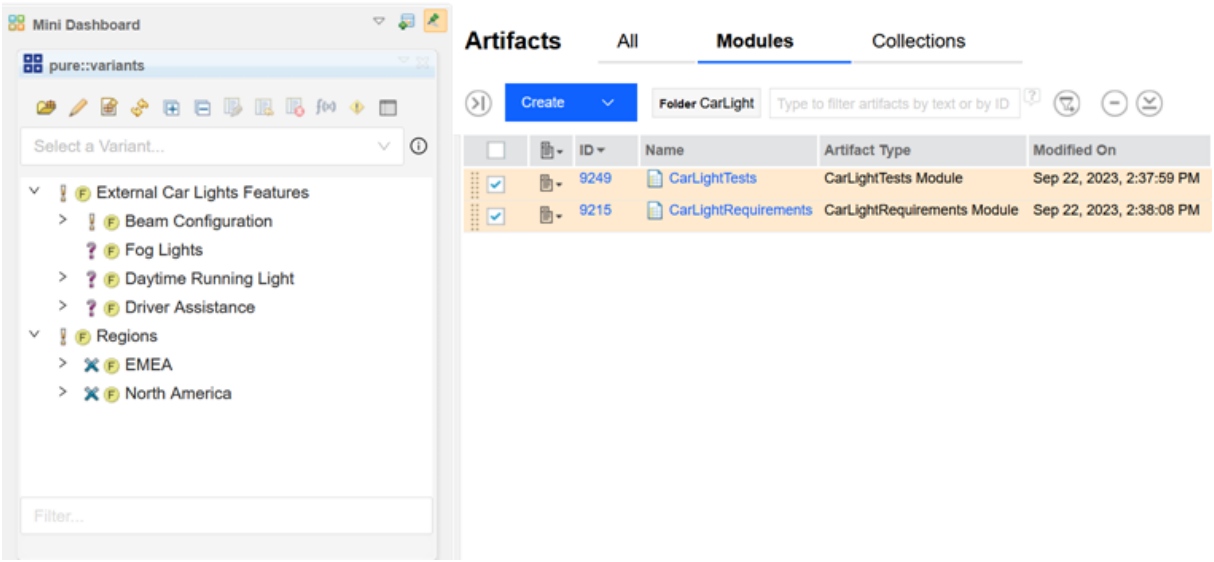



## 4.5. Importing and Updating Models using the Web Client

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

Alternatively, multiple modules can be selected for import in Artifacts view of DOORS NG (Figure 37, “Import Modules from Artifacts View”) by using the checkboxes in the Modules list.

**Figure 37. Import Modules from Artifacts View**

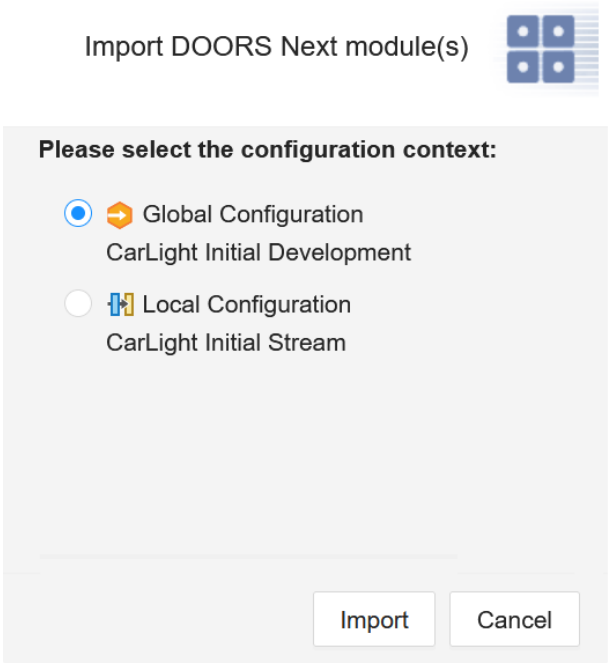



When pressing the **Import** button, in case the current context is within a Global Configuration, the context to be stored in resulting family models needs to be selected (Figure 38, “Import Dialog of Integration”). After confirming

the selection, the Web Client opens where a wizard guides through the rest of the import process. In case the pure::variants project was opened in a Global Configuration context using the integration, the Web Client will be opened in *Global Configuration* mode.

The import automatically adds the created models to the opened configuration space.

**Figure 38. Import Dialog of Integration**




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

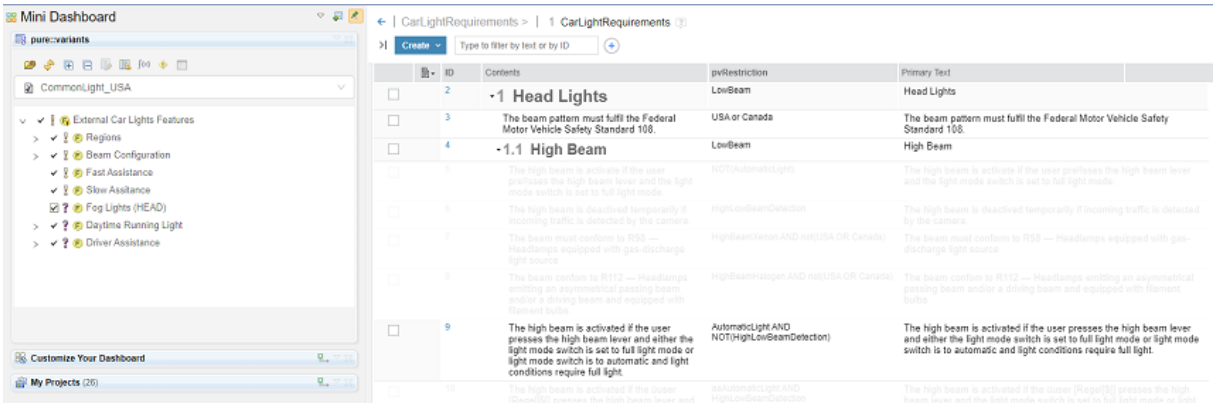
When pressing the **Update** button, in case the current context is within a Global Configuration, the context to be stored in resulting family models can be changed using the same dialog as for the import. After confirming the selection, the Web Client opens where a wizard guides through the rest of 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.

## 4.6. 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, press the **Show Preview** button , a variability preview of the module is shown with respect to the selected variant model. In *Preview* mode, selected requirements retains the normal view while the unselected requirements are greyed out as shown in the figure Figure 39, “Variability Preview”. Thereby, all substitutable attributes (which are selected for the DOORS NG requirements module) will be previewed, resp. substituted with the variant specific values. Please ensure to configure the substitutable attributes to be shown in your requirements module's view, using  of view's column head.

**Figure 39. Variability Preview**




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

### Note

*In Preview mode the requirement list becomes read-only, hence, restrain from adding/removing any column, because doing so, will make the entire page unresponsive, and then, the only way to get out of this situation will be to refresh the whole page.*

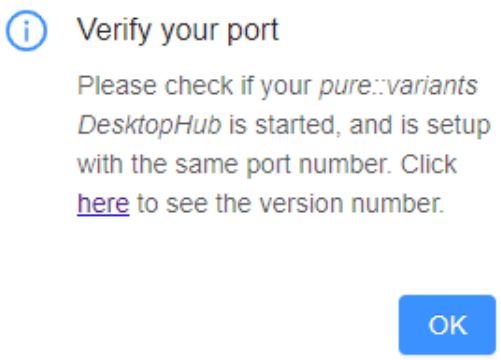
The previewed requirements module is only shown in time of preview, and **cannot** be exported with the capabilities of DOORS Next application, as PDF or Word or Excel document.

## 4.7. Troubleshooting

### Connecting with Desktop Hub

The following dialog is shown in the Integration when there is a problem to connect to the Desktop Hub.

**Figure 40. Unable to connect to Desktop Hub**




Please verify your settings according to the following actions:

- Check if the *port number* of the Integration and the Desktop Hub running instance does match. To check for Desktop Hub, open context-menu on Desktop Hub running instance taskbar icon and navigate to **Hub Configuration->Preferences->Services ->Model Access**. For Integration, navigate to **Settings** ()->**General** tab view.

- Check if a secured connection is enabled for the Desktop Hub running instance. Hence, open context-menu on DesktopHub toolbar icon and navigate to **Hub Configuration**->**Preferences**->**Services**->**Model Access** and ensure the **Use secure connection (HTTPS)** option is checked.

- Check if your web-browser accepts Desktop Hub connection, if secured connection is enabled. Therefore, navigate to the following address *https://localhost:[port]/pv/version* in your web-browser and when asked, accept the self-signed certificate once.

> ### Note
>
> *Please substitute **[port]** with your configured port number in Desktop Hub* (**Hub Configuration**->**Preferences**->**Services**->**Model Access**).

## Preview Problems

Various types of error messages can show up while *Preview* generation.

> ### Note
>
> *Please note that the settings configured in* **JTS**->**Advanced Properties** *can take up to ten minutes to take effect. If the **Preview** doesn't work even after waiting for ten minutes then there could be some misconfigured settings in **JTS**.*

#### RM Whitelist Configuration Problem

If the URL of the server hosting the Integration has not been correctly added in the **RM Whitelist**, an error like Figure 41, “RM Whitelist Problem” will show up.

**Figure 41. RM Whitelist Problem**

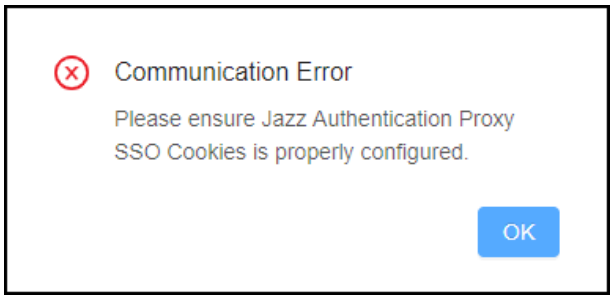

To fix this problem, go to the **RM**->**Whitelist (Outbound)** and enter the correct URL of the server hosting the Integration in the following format *https://[server]:[port]/*.

#### JTS Advanced Properties Problem

Couple of error messages can show up depending on different configuration problems in JTS Advanced Properties.

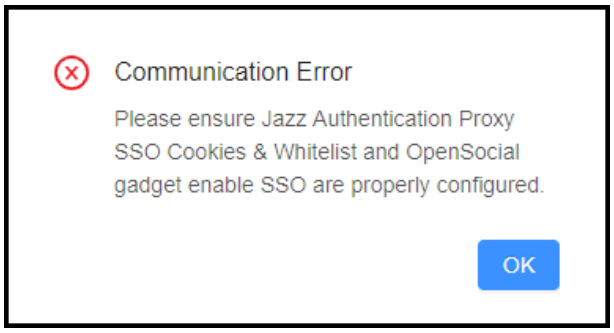
If the error message shown in Figure 42, “JTS Cookies Setting Problem” appears. It means that **Jazz Authentication Proxy SSO Cookies** is not properly configured. Therefore, navigate to **JTS**->**Advanced Settings** and go to **Jazz Authentication Proxy SSO Cookies**. By default, this field contains `LtpaToken, LtpaTo\ken2, JSESSIONIDSSO`, please change its value to following string `LtpaToken, LtpaToken2, JSESSIONIDSSO, JSA_SESSION_IDENTITY`.

**Figure 42. JTS Cookies Setting Problem**




If the error message shown in Figure 43, “JTS Settings Problems” appears. It could be because of various misconfigurations inside the **JTS** settings.

**Figure 43. JTS Settings Problems**



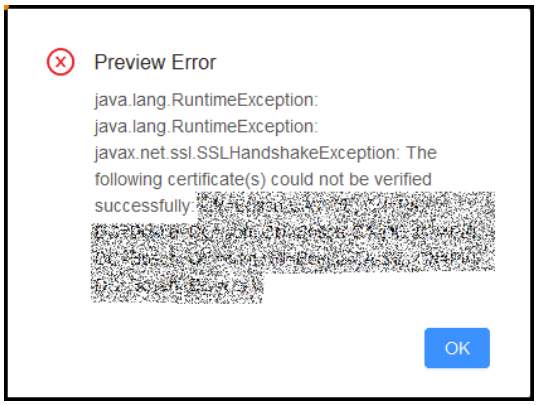

To fix this problem make sure to configure the following settings in **JTS**->**Advanced Settings**:

- Set the **OpenSocial gadget enable SSO** to **true**.
- Set the value of **Jazz Authentication Proxy SSO Cookies** to `LtpaToken, LtpaToken2, JSESSIONIDSSO, JSA_SESSION_IDENTITY`.
- Set the value of **Jazz Authentication Proxy SSO Whitelist** to a URL in the following format *https://[server]:[port]/pvwidget/vel* (presuming that the war file was renamed to *pvwidget.war* before deploying on the server). For proper working of *Preview*, the URL must end with */vel*.

**Certificate(s) could not be verified problem**

If pure::variants Integration and DOORS NG are deployed on different web application servers, please make sure that these web application servers trust each others SSL certificates.

**Figure 44. SSL certificte of Integration's web application server not trusted**



### Connection Issues - Timeouts, Interrupts, etc.

Since pure::variants Connector and Integration are heavily relying on integration to DOORS NG tool, there has to be lot of 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:

- Activate retry strategy: `PV_HTTP_CLIENT_REQUEST_RETRY=true`
- Increase retry count (default: 3): `PV_HTTP_CLIENT_REQUEST_RETRY_COUNT=4`
- 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_REQUEST_RETRY=true
-DPV_HTTP_CLIENT_REQUEST_RETRY_COUNT=4
-DPV_HTTP_CLIENT_CONNECTION_TIMEOUT=240000
-DPV_HTTP_CLIENT_READ_TIMEOUT=240000
...
```

Additionally, you may need to define these parameters for the web-based pure::variants Integration for DOORS NG, as well. In this case, you need to specify the parameters as environment variable or system property for Java runtime.

- **WebSphere Liberty**: Add the lines into the *jvm.options* file of your server instance's configuration directory (see Liberty: Directory locations and properties:

```
-DPV_HTTP_CLIENT_REQUEST_RETRY=true
-DPV_HTTP_CLIENT_REQUEST_RETRY_COUNT=4
-DPV_HTTP_CLIENT_CONNECTION_TIMEOUT=240000
-DPV_HTTP_CLIENT_READ_TIMEOUT=240000
```

- **Apache Tomcat**: Add the lines into the *server.env* file of your server configuration directory (*<tomcat-install-dir>/config*):

```
-DPV_HTTP_CLIENT_REQUEST_RETRY=true
-DPV_HTTP_CLIENT_REQUEST_RETRY_COUNT=4
-DPV_HTTP_CLIENT_CONNECTION_TIMEOUT=240000
-DPV_HTTP_CLIENT_READ_TIMEOUT=240000
```

#### Note

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

Alternatively, if preferred, these parameters can be specified via JAVA_OPTS or any other mechanismen, which allows to set these Java sytem properties.

# 5. Advanced Topics

## 5.1. Adding Variability Information in DOORS NG

### Defining an attribute in DOORS NG (in General)

In the succeeding sections are several DOORS NG attributes explained, which can be used in conjunction with pure::variants. This section gives a brief introduction on how to add a custom attribute in DOORS NG application.

A custom attribute must be defined per project's component, resp. for its appropriate configuration. Please ensure to have opened the DOORS NG project's component and configuration.

Then, navigate to **Administration(⚙)** -> **Manage Component properties** (or **Manage Project Properties**). In the tab **Artifact Attributes**, please press **New Attribute...** to open the formular for a new attribute.

Please see the *Setup* sub-section of each succeeding section for the appropriate setup of attributes.

Once the specific attributes are created, these must be added to the specific artefact types, like a *Requirement* or *Heading*. (These artefact types might have any other name. It doesn't have to be named the same!)

Therefore, navigate to **Artifact Types** and select each type instance, and press **Add Attributes...** to add theses attributes for variability information.

### Note

Please note that names of attributes and attribute types can be named as you like. The here proposed names are taken as default names. In time of import into pure::variants, these can be mapped accordingly, if custom names are defined.

## Defining an Element Variability Type

A DOORS NG attribute named `pvVariationType` can be used to provide pure::variants with information about the intended variability type of an object. For each object with this attribute, the attributes value is matched against the four possible variability types for elements (use strings `mandatory`, `alternative`, `or`, `optional`, `ps:mandatory`, `ps:alternative`, `ps:or`, `ps:optional` ) during initial import or synchronization. If this attribute is not defined or contains any other value than listed above, the default value of `ps:mandatory` or `ps:optional` (if the element has a restriction defined in attribute `pvRestriction`) is used.

### Setup

This attribute requires a specific set of allowed values. Therefore press **New Data Types...** and enter the name *pvVariationTypesType*, select **Enumerated list of values**, and press **Add Multiple Values** to enter each variability type (e.g. mandatory, or ps:mandatory) on a new line and press **OK**.

## Defining an Element Name

Each imported DOORS NG object gets only a visible name assigned. But to use the DOORS NG requirement in restrictions, constraints and calculations of pure::variants, a unique name is required.

If the DOORS NG attribute `pvName` is defined and not empty, this value is used as unique name. To prevent later problems the name is checked during import and synchronization. In case of violation of the naming conventions pure::variants automatically converts the name to a compatible name. However, pure::variants does not prevent creation of non-unique names during import and synchronization. If duplicate names are existing, they will be marked in the model editor.

### Setup

This attribute requires to be of type *string* only.

## Defining Element Restrictions

A DOORS NG attribute named `pvRestriction` can be used to generate a pure::variants restriction for the related pure::variants element. The language used for restriction definition is pvSCL. The pvSCL language is described in detail in the pure::variants User Guide.

Restrictions (as usual in pure::variants) may refer to elements in the same model or in any other model used together with the defining model in a configuration space. For example, a requirement element should only be

selectable, if a feature with unique name "MyFeature" or the feature "MyOtherFeature" is selected, simply use "`MyFeature or MyOtherFeature`" as value for the restriction attribute `pvRestriction`.

A restriction on a parent element, which is evaluating to false does deselect the whole subtree below the restricted element. None of the children is selected anymore. The restrictions on the child elements do not have any influence on the selection of the children in that case.

### Setup

This attribute requires to be of type *string* only.

## Defining Element Constraints

A DOORS NG attribute named `pvConstraint` can be used to generate a pure::variants constraint for the related pure::variants element. The language used for restriction definition is pvSCL. The pvSCL language is described in detail in the pure::variants User Guide.

Constraints (as usual in pure::variants) may refer to elements in the same model or in any other model used together with the defining model in a configuration space. So if the selection of a requirement should imply the selection of two other elements with the names "MyRequirement" or the feature "MyOtherRequirement", simply use `SELF IMPLIES MyFeature or MyOtherFeature` as value for the restriction attribute `pvConstraint`.

### Setup

This attribute requires to be of type *string* only.

## Defining Element Default Selection

Each imported DOORS NG object gets an default selection state.

This is calculated using the following rules:

- If variation-type is mandatory or optional the element gets default selected.
- If the variation-type is undefined and a restriction or constraint is defined, the element gets optional and default selected.
- In all other cases the element is default selected off.

If this does not fit your needs you can specify the default selection for each element by using the DOORS NG attribute `pvDefaultSelected`.

Allowed values are (ignoring case) `on` and `off`.

### Setup

This attribute requires a specific set of allowed values. Therefore press **New Data Types...** and enter the name *pvDefaultSelectionType*, select **Enumerated list of values**, and press **Add Multiple Values** to enter *on* and *off* each on a new line and press **OK**.

# 5.2. Calculations within attribute texts

Texts in DOORS NG attributes may have variable parts, while the most part of the text will remain equal in all of your variants.

In this case you can add a pvSCL statement to be evaluated by pure::variants. The statements will be replaced with actual values of your variant by pure::variants, if you perform *partial text substitution* during the transformation of your variant.

A pvSCL statement starts with an opening marker '[' followed by an pure::variants pvSCL calculation rule and is ended with an closing marker ']'. To escape a statement the escape character is used '$'. This will prevent pure::variants from evaluating and replacing the escaped statement.

Example: *The maximum allowed speed is [Speed->Max] km/h.* in an DOORS NG attribute will be replaced with the value of attribute *"Max"* on Feature *"Speed"* in the exported variant. The result could be: *The maximum allowed speed is 100 km/h.*

Escaping the rule in the previous example, like *The maximum allowed speed is $[Speed->Max] km/h.*, forces pure::variants to ignore the rule. The result, would be *The maximum allowed speed is [Speed->Max] km/h.* in that case. The rule is not changed, but the escape character is removed.

### Note

This functionality is an optional feature and only available if transformation modus *Copy with Duplication* or *Create Streams* is set. It has to be explicitly selected during transformation, cause it may slow down your export process.

## 5.3. 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 45. Example HTML 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 value 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) can be included in a variant if keepConstraint is set to true (where supported), but must be included for partial transformations, since the information is needed to later create the 100% variants.

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. But a ** tag is not supported.

## 5.4. Link Propagation

The link propagation allows to evaluate the dependency relations betwenn DOORS Next requirements, with the usage of pure::variants relation types. Therefore, the user can specify for each link type in DOORS Next an appropriate pure::variants relation type, as equivalent. With the DOORS Next integration, the user can easily do the appropriate mapping of link and relation type for each requirements module, as needed. When importing a requirements module, the user should activate the link propagation.

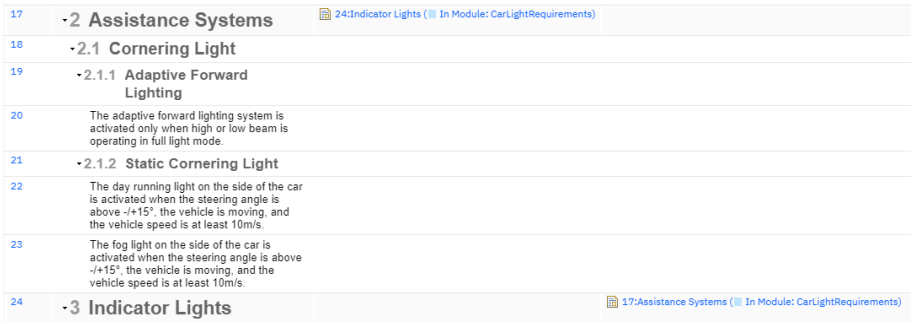
To make a requirement *R1* dependend on another requirement *R2*, as follows `R1 # R2`, the user has to create a link from requirement *R1* to requirement *R2*, using the DOORS Next's native links, e.g. *Link To*, or *References*.

Furthermore, for a link type, which is mapped to pure::variants ***Requires*** or ***Requires All***, the user must define the *pvVariationType* to *ps:optional* and the *pvDefaultSelected* to *off* for the target requirement *R2*. Thus, the pure::variants evaluation will only include the target requirement *R2*, if and only if the source requirement *R1* is included in the variant result, as well.

For a link type, which is mapped to pure::variants ***Required For***or ***Required For All***, the user must define the *pvVariationType* to *ps:optional* and the *pvDefaultSelected* to *off* for the source requirement *R1*. Thus, the pure::variants evaluation will only include the source requirement *R1*, if and only if the target requirement *R2* is included in the variant result, as well.

Please see the following simple example:

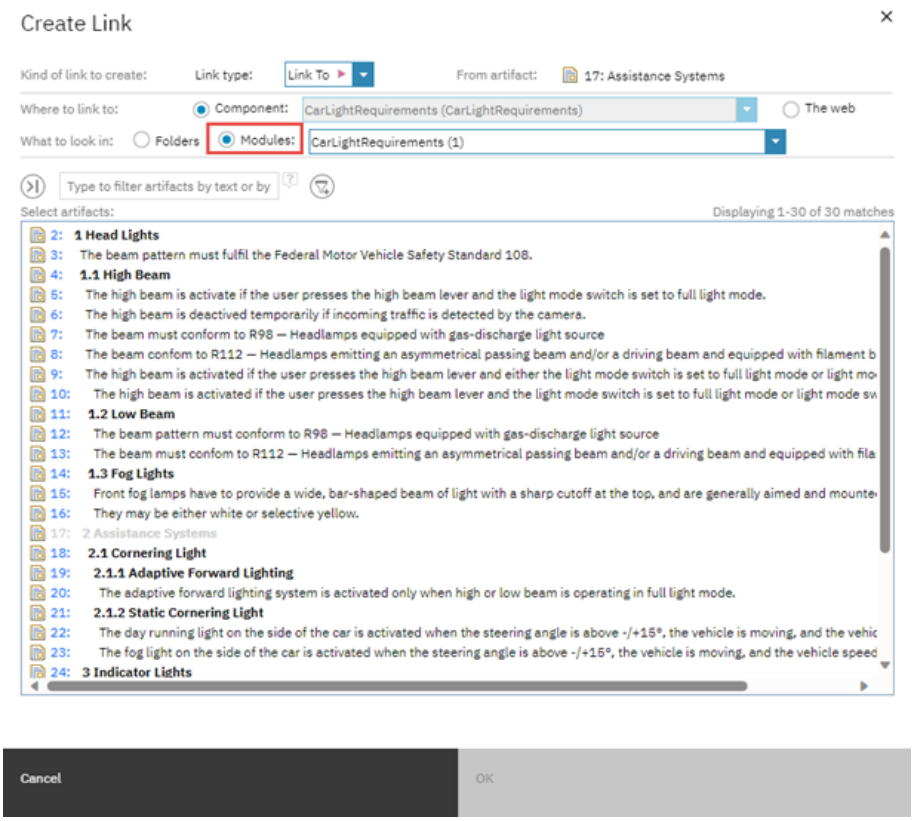
**Figure 46. Linkage of requirement *R1* to requirement *R2*.**




For demonstration purpose, these requirements exist in the same requirements module. However, if these requirements exist in different requirements modules, the link propagation is still applicable with pure::variants.

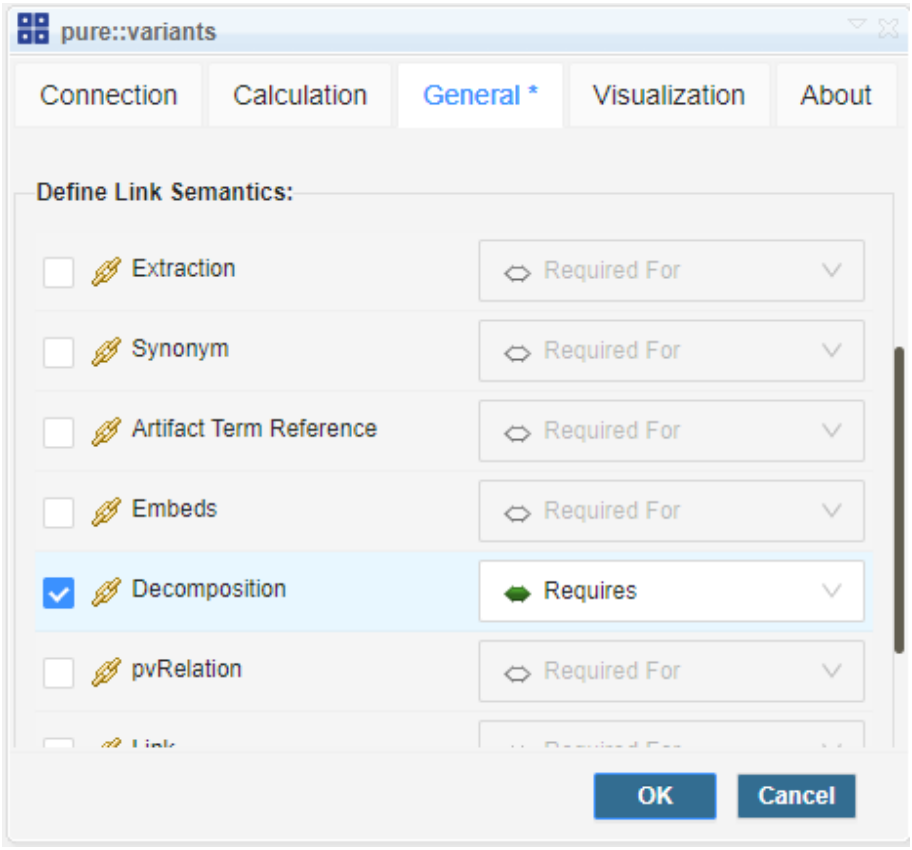
Please be aware, to only create links between requirements, using ***module-scoped links*** (red rectangle), as shown below:

**Figure 47. Create link in DOORS Next**



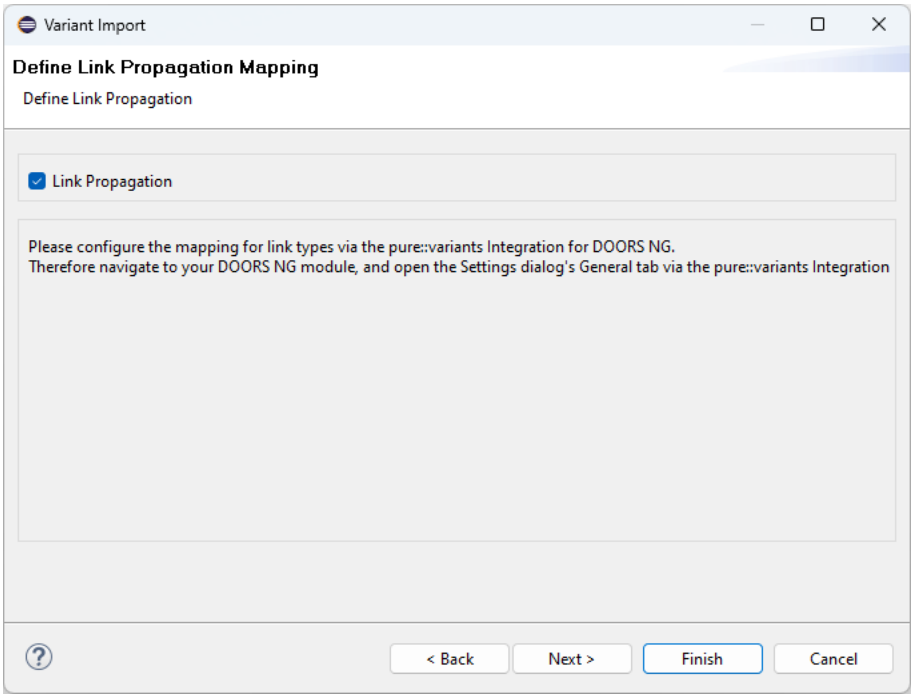
If user decides to apply the link propagation on *Decomposition* type, the user has to configure the link type mapping in the source requirements module, as shown below.

**Figure 48. Link type mapping in DOORS Next integration**



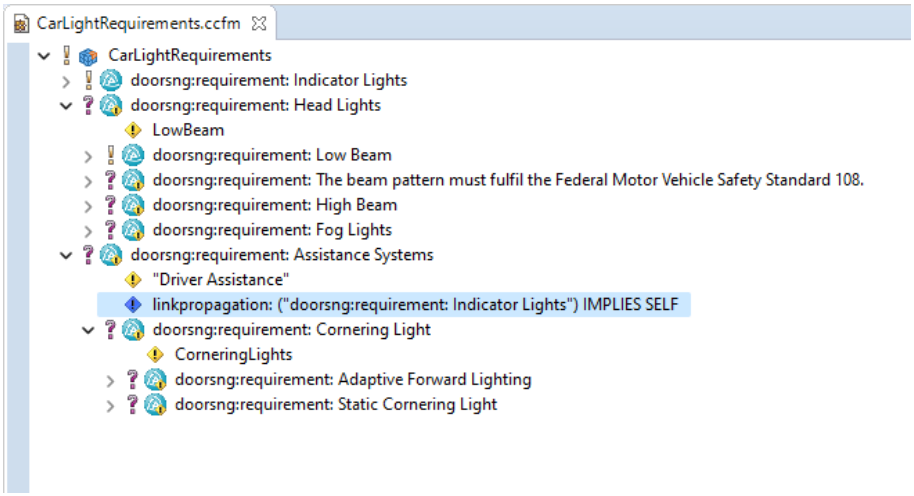

Once, the DOORS Next assets are prepared, as described above, the user has to enable the *Link Propagation* when doing the import of these DOORS Next requirement module(s) as family models, as seen below:

**Figure 49. Import Wizard - Link Propagation**



In the following screenshot, you can see the imported family model, which contains the adapted modelling for proper evaluation:

**Figure 50. Family Model - Adapted to link propagation support**




For applying the link propagation, the user must import all DOORS Next requirements modules, which contain linked requirements. The family models must be imported from the same ***Global Configuration***, and these must be either present in ***Full Mode*** or ***Quick Mode***, but *not mixed*, in the Configspace.

The Link Propagation is currently only applicable with the ***Stream-based transformation*** mode. The transformation modes are not supported.

### Link Propagation between DOORS Next and Test Management assets

For link propagation between DOORS Next's requirements and Test Management's test cases, please see the documentation for the *pure::variants - Connector for IBM Engineering Test Management*.

## 5.5. ANT transformation and synchronization

While the transformation and synchronization is supported for DOORS NG modules a valid authentication is required. There are two possibilities for ANT transformation. Either the user credentials (user name and password) are provided in the DOORS NG transformation module or defined as environment variables. Therefore define *PV_DOORSNG_USER* for user name and *PV_DOORSNG_PASSWORD* for password. For ANT synchronization only environment variables are suitable.

## 5.6. DOORS NG update capability

As already motivated in the pure::variants User's Guide; chapter 5.10, it is necessary to update variants while the product line changed over the time. In case of DOORS NG, the following transformation modes are update-aware:

**Table 3. Transformation Modes Update Capability**

| Transformation mode | Update Capable | Description |
|---|---|---|
| *Attribute-based Approach* | Yes | The attribute, named *pvVariants* by default, will have added the actual variant name. This comprises the removal of the variant name for requirements, no more part of the actual variant. |
| *Module-based Approach* | No | A variant DOORS NG module is always newly created and can't be updated. |

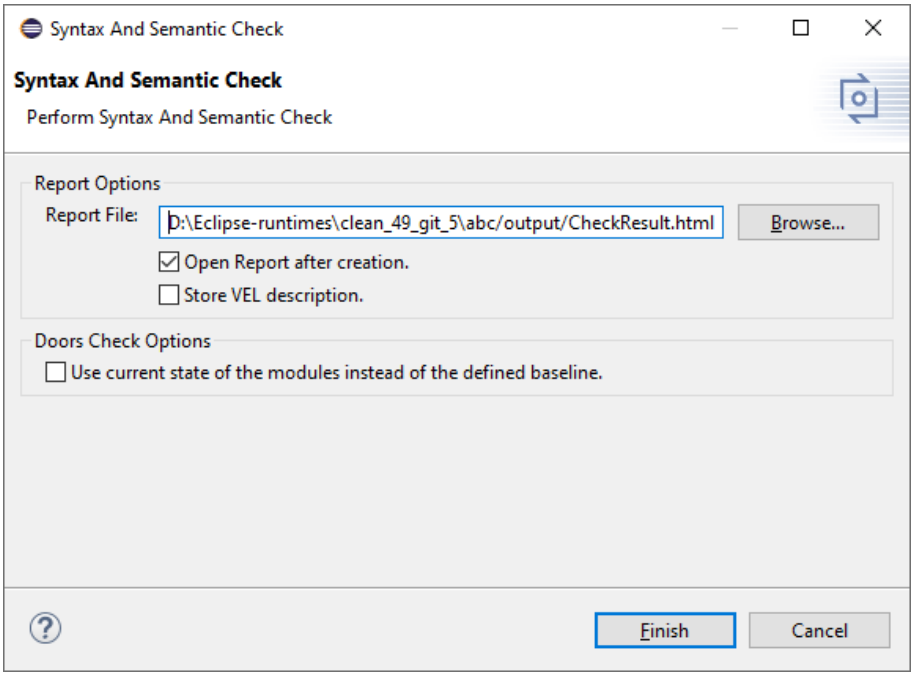
| Transformation mode | Update Capable | Description |
|---|---|---|
| *Link-based Approach* | No | A variant DOORS NG module is always newly created and can't be updated. |
| *Stream-based Approach* | Yes | A variant DOORS NG stream is always newly created. Then the user is responsible to transfer his changes from the previously derived variant stream to the new variant stream version. DOORS NG offers the functionality to merge changes between stream (with Delivering/Accept changesets). For details, see the documentation of DOORS NG. In case of transforming a DOORS NG module, referenced in a Global Stream, the stream is created as described previously. If having included transformation modules, which also support Global Streams, in the same transformation configuration, only one global variant stream is generated. This requires that all variability-aware models, participating in the transformation process, originate from the same global stream. |

## 5.7. Checking all DNG modules 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 DOORS NG 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 51. Syntactic and Semantic check dialog.**




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

With enabeling 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.

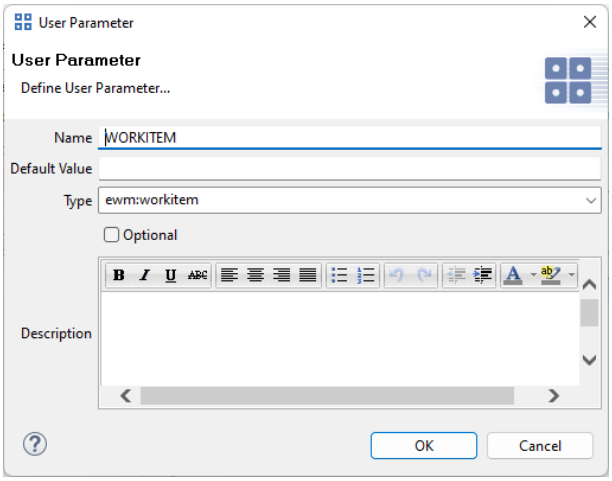
# 5.8. Linking Work-Item to a Change Set in Doors NG Transformation

A work-item can be linked to a change set during a Doors NG transformation by creating a user parameter, adding it to the transformation module and finally selecting or creating the work-item during the transformation.

## Creating a User Parameter

To link a work-item with a change set, create a new user parameter of type **ewm:workitem**. (See Figure 52, “Adding Work-Item as User Parameter”.) Please refer to pure::variants User's Guide for details on User Parameters.

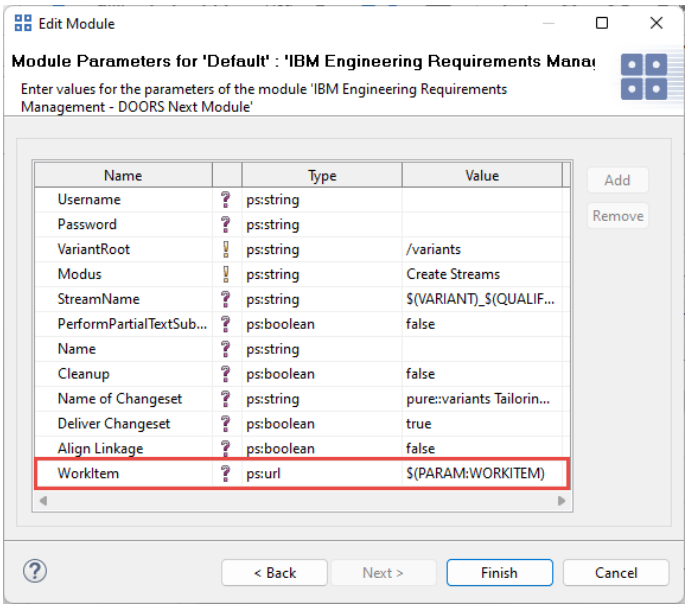
**Figure 52. Adding Work-Item as User Parameter**




## Adding a Transformation Module Parameter

The user paramter has to be linked to the transformation module configuration, by adding a transformation module parameter with name **WorkItem** and value **$(PARAM:WORKITEM)**. This parameter is queried during the transformation. (See Figure 53, “Linking Work-Item in Module Configuration”.)

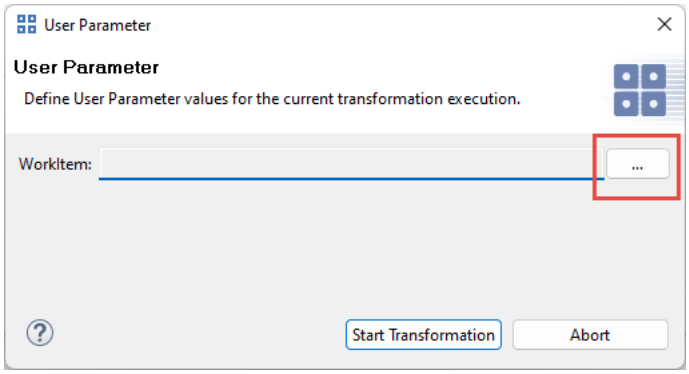
**Figure 53. Linking Work-Item in Module Configuration**



## Work-Item Selection Dialog

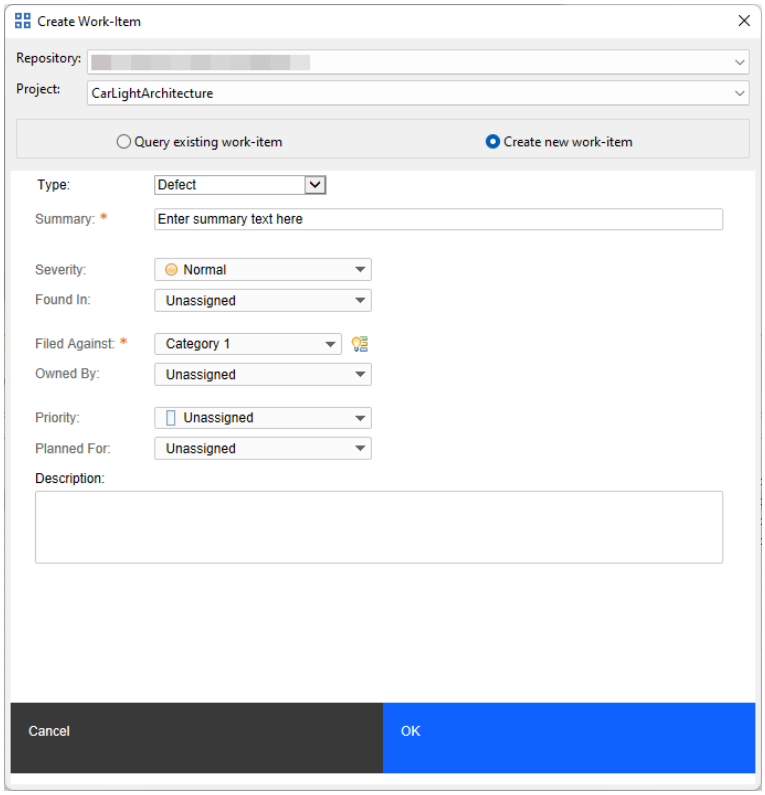
During transformation, you can select the work-item from the User Parameter dialog. (See Figure 54, “User Parameter Selection Dialog”.)

**Figure 54. User Parameter Selection Dialog**




In the Create Work-Item dialog (See Figure 55, “Create or Query Work-Item”), the servers listed under the **"Known Servers"** (Window->Preferences->Variant Management), in the **"Configuration & Change Management"** category are listed in the **'Repositories'** dropdown box. All the projects in the selected repository are listed in the **'Projects'** dropdown box. Based on the selected project, the user can choose to either create a new work-item or to query an existing work-item.

**Figure 55. Create or Query Work-Item**




Once the user presses **OK**, the transformation will proceed and the work-item is linked to the changeset.

## 5.9. Redact requirements while transformation

Redact enables the selective redaction of hierarchical requirements based on rules defined on the requirements, during Create-Stream transformation process. Performing Create-Stream transformation with redact enabled, i.e.,

redact mode set to *RedactHighestExcluded*, requirements which are not part of the resulting stream are redacted with the placeholder text configured in the transformation parameter *RedactText*, rather than removing the requirements entirely. Following are the available redact modes:

- Off - Peforms Create-Streams transformation with no effect of redact, i.e., removes requirements that are not included in the resulting stream during the transformation.
- RedactHighestExcluded - If redact mode is set to RedactHighestExcluded, and the top-level requirements in the hierarchy is not included in the variant result during the Create-Stream transformation, then

  i. The highest excluded requirement is kept, and not removed as usual.

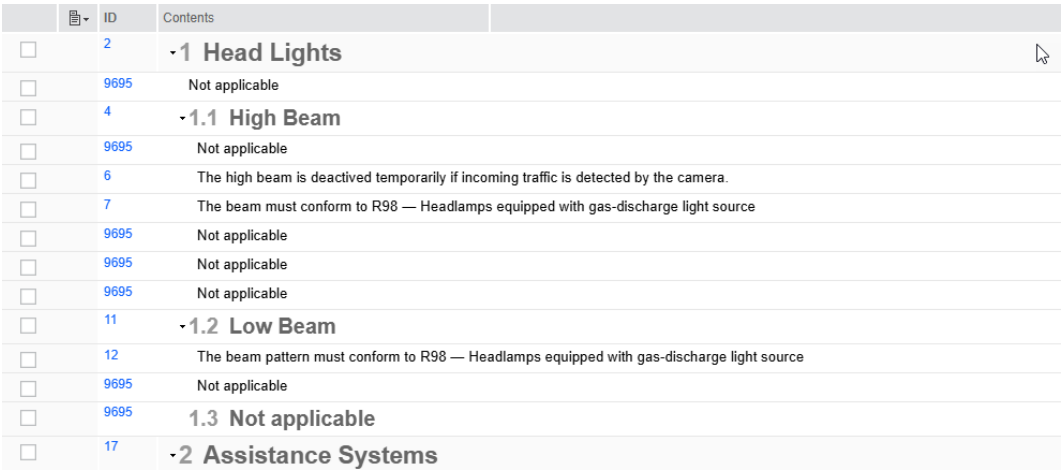
  ii. The kept requirement is being redacted with defined placeholder text, which are defined by the *RedactText* transformation module parameter.

  iii. All its children requirements are getting removed as usual.
  If the top-level requirements in the hierarchy is included and few/ all of its children are excluded, then

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

  ii. Few/ all off its child requirements are redacted.

**Figure 56. Transformation result with Redact mode: RedactHighestExcluded**




# 6. Known Issues

None