pure::variants Connector for Simulink Procedures

Parametric Technology GmbH

Version 7.0.0.685 for pure::variants 7.0

2025

Table of Contents

1. Introduction

pure::variants for Simulink is a tool for modeling, presenting and configuring variability in variant-rich Simulink models. It is a member of the pure::variants product family and allows MATLAB/Simulink users to

- Model functional variability from an application perspective with the help of features
- Depict variability modeled in MATLAB/Simulink with the help of the Variant Block Set
- Define rules for combining features and variability in Simulink models

pure::variants for Simulink is referred to below simply as pure::variants.

Variability in Simulink models is described by a value-based variation point (in short variation point) that reflects a property that varies between the different instances of a product, such as the selection of an implementiation of a specific sensor type out of three possibilities to be connected as the input to the Simulink model. The variation point can be used for various value-based variability mechanisms in the Simulink model at which the behavior should change through the selection of a specific variation.

A variability mechanism refers to the technical implementation of a variation in the Simulink model. The following value-based variability mechanisms are worth mentioning, each of which depicts variability in its own specific way and which are offered by the variants block set as a library of blocks (*Effect* block):

- Conditionally executable subsystems such as an *Enabled* subsystem or a *Function Call* subsystem
- Subsystems such as the *If* block or *Switch Case* block
- Blocks for forwarding signals, such as the *Switch* block or *Multiport Switch* block
- Logical gates such as the *AND* block or *OR* block

Most of these blocks require an input signal in order to control the execution of the block. *Control* blocks of the Variant Block Set are used for this in order to execute the corresponding variation depending on the specific values of the Control blocks. These specific values are stored in parameters (of the MATLAB/Simulink workspace).

Two methods exist for defining and managing value-based variation points, and pure::variants offers synchronization between them. The Variation Point Explorer in MATLAB/Simulink can be used to create, change and remove variation points that are persistently stored in a TargetLink Data Dictionary. The variability model allows the management of variation points stored in this model directly within pure::variants. pure::variants offers synchronization of the variation points between MATLAB/Simulink and pure::variants.

Configurations of the variant-rich Simulink models that are derived from a feature selection as well as the defined rules are created and managed in pure::variants as variant models. A configuration of variation points created in this way can be exported to MATLAB/Simulink by assigning the selected, specific value of the variation as the parameters of the configuration.

This document describes the processes and activities required for this.

1.1. Term Definitions

Assignment	An assignment links a variation to features. The features are referenced in a condition, which can be true or false depending on a selection of features. If the condition becomes true, the variation to which the assignment applies is automatically selected by the auto resolver. The assignment corresponds to the expression <CONDITION> REQUIRES SELF.
Auto Resolver	The auto resolver attempts to automatically solve problems in the selection, such as selecting features or elements that are absolutely required in order to obtain a valid configuration. This behavior is utilized in the automatic selection of a variation based on the condition of its assignment.
Condition	A condition is a logical expression that is used to evaluate a selection of features. This requires that features be linked together by means of logical operators.

Configuration Space	The configuration space combines feature and family models to allow for the configuration of the product family. A variant model must be created in the configuration space for every configuration. A configuration space is therefore also required for a variation point configuration, based on which an instance of a variant-rich Simulink model can be created.
Data Dictionary	All information relating to a variation point such as all available variations and the currently selected variation can be stored in a TargetLink Data Dictionary.
Default Assignment	A variation designated as the default assignment for a variation point is defined as the default variation. Each variation point may have no more than one default variation. If no other variation is selected by the user or by an assignment, this default variation is always automatically selected.
Family Model	The family model of pure::variants describes the product family as a whole, with all products. Individual products can be created by selecting features. The variability model therefore also describes the totality of all Simulink model instances from which individual instances can be derived.
Feature	A feature is a property of the product in reference to commonalities and differences. Features are structured hierarchically in a feature model and classified as required, optional, alternative or in "or" groups.
Feature Model	A feature model in pure::variants describes commonalities and differences within the product family. A feature model contains features and their dependencies or relationships that guarantee a valid selection.
Feature Selection	See selection.
Configuration	A configuration describes a selection of elements of a family model as well as a corresponding, valid selection of features. A configuration is described by a variant model and is also referred to as a variant configuration. The configuration of the variation points, each with one selected variation, is referred to as a variation point configuration.
Parameter	A parameter stores the value of the currently configured variation of a variation point. The parameter is stored as a variable in the MATLAB/Simulink workspace.
Selection	A selection describes the selected set of features and/or family model elements. A selection that contains only features is referred to as a feature selection.
Simulink Model	A Simulink model is imported from MATLAB/Simulink and represented in pure::variants by a family model.
Transformation	A transformation is the creation of a product in the product family based on the configuration, such as the instance of a Simulink model. For example, the instance of the Simulink model can be generated by an M script, which sets the values of the variation points. The values of the variation points

	are stored in the parameters of the MATLAB/Simulink workspace. If a variation point is connected to a TargetLink variant block, the value is also written to them variable located in Data Dictionary.
Variability Model	A variability model in pure::variants is a family model that describes all variation points, their variations and all parameters of the MATLAB/Simulink workspace. It therefore depicts the variability for Simulink models. Variation points can be imported from MATLAB/Simulink or created directly in the model.
Variant Configuration	See configuration.
Variant Block	A Simulink or TargetLink variant block is configured via the current variation of a variation point. They can be either Control blocks or Effect blocks. Control blocks directly reference a parameter and control the variability in the Simulink model; a typical example is a Constant block. Effect blocks (variability mechanisms) are the implementation of the variability in the Simulink model.
Variant Block Set	The Variant Block Set offers a library of different variant blocks.
Variant Model	A variant model in pure::variants contains the feature selection and selection of elements of the family model with which a product of the product family is created. It therefore describes an instance of the Simulink model and/or its configuration.
Variation	A variation is a characteristic of the variation point, e.g. a specific sensor type.
Variation Point	A variation point is used to depict variability in Simulink models. A variation point consists of a list of alternatives and describes a property that varies between the various instances (variants). The selection of a specific alternative changes the behavior in the Simulink model, such as the selection of a specific sensor type out of three possibilities.
Variation Point (value-based)	A value-based variation point is used to depict value-dependend variability in Simulink models. Its alternatives are described by different variations.
Variation Point Configuration	See configuration.

1.2. Conventions

The conventions described below should be followed when modeling the individual elements to express variability.

Name

OCL strings are used for naming the elements. This means that the string may consist only of upper and lower case letters, numbers or underscores and that the string may not begin with a number.

Feature

A feature must have a unique name. The feature is referenced in conditions using this name. In addition, a feature can also have a visible name consisting of an arbitrary string that can be more descriptive. This visible name is used in pure::variants only for display in the user interface.

Variation Point

A variation point must have a unique name. This name must begin with the prefix defined in MATLAB/Simulink, which is by default defined as “VAR_”. In pure::variants, this prefix is automatically added upon creation of a variation point if it was not already entered by the user. In MATLAB/Simulink, on the other hand, the user is responsible for ensuring that the prefix is used.

Variation

A variation must have a unique label and a unique value within its variation point. The value must be numerical, and both integers and decimal numbers are permitted, such as -1, 2, - 1.4 or 4.9 (the US English style is used here).

Parameter

A parameter must have a unique name. This name must begin with the prefix defined in MATLAB/Simulink, which is by default defined as “VAR_”. In pure::variants, this prefix is automatically added if it was not already entered by the user upon creation of a variation point. In MATLAB/Simulink, on the other hand, the user is responsible for ensuring that the prefix is used.

Condition

A condition must comply with the constraints of the pvSCL expression language. It is important to note that only features may be linked together with the help of logical expressions such as AND, OR and NOT. In order to guarantee automatic selection of the variation based on the assignment, the methods for handling attribute values that are also offered by the pvSCL expression language may not be used. Information about the pvSCL expression language can be found in the pure::variants user documentation in section “9.8 Expression Language pvSCL”.

1.3. Software Requirements

The following software has to be present on the user's machine in order to support the pure::variants Connector for Simulink:

Operating System:
- Windows XP, Windows 7, Windows 2003 Server

Java: Java Virtual Machine (JVM) Version 5 is at least required. We recommend using a Sun JDK 5 compatible JVM. See http://www.java.com/ for a suitable JVM.

Matlab/Simulink: Matlab/Simulink 2006b and newer is required.

If the following programs are installed on the user's PC, alternative concepts can be used with pure::variants:

TargetLink: TargetLink of versions 2.x and newer or a corresponding TargetLink block set 2.x and newer. The TargetLink block set is provided for free by dSpace GmbH.

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

2. Configuring pure::variants

In order to use pure::variants together with MATLAB/Simulink, it must first be prepared and configured. This includes activating the correct perspective for the work and creating a project structure. The installation and configuration of pure::variants are described below.

2.1. Installation

pure::variants is either delivered with a CD or can be downloaded from the pure::variants update site as an archive. Information on downloading such as the URL and login details are provided in the registration e-mail. The down-

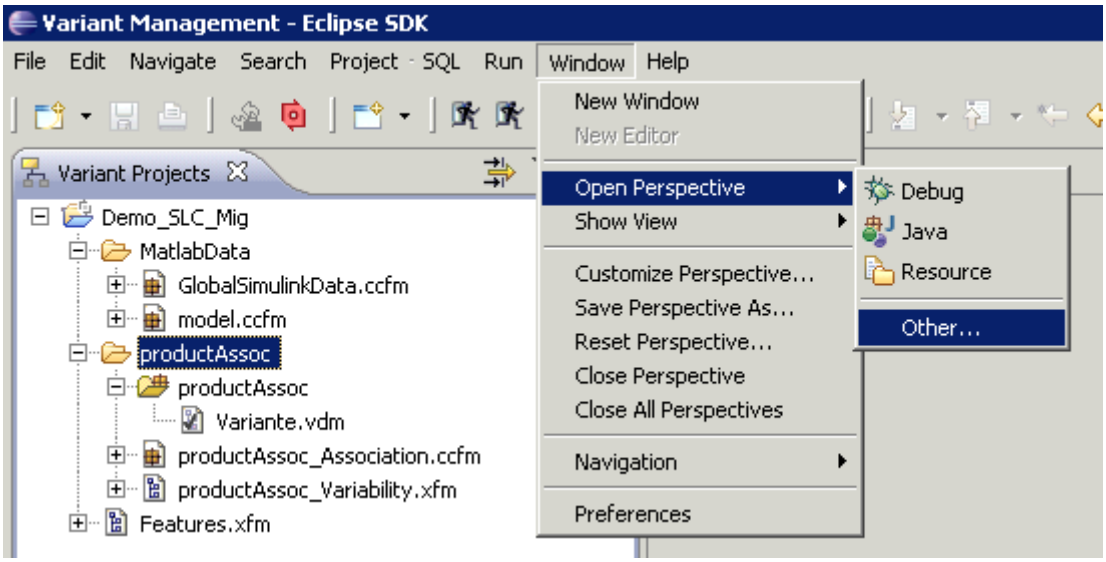
loaded archive must be decompressed before the installation can begin. The installation of pure::variants can be started from any location in the file system by running Setup*.exe, which carries out the installation process.

The result is an independent installation of pure::variants that can be easily uninstalled again without affecting other Eclipse or pure::variants installations. More information regarding the pure::variants installation process can be found in the pure::variants user documentation in section “2. Software and License Installation”.

2.2. MATLAB/Simulink Variant Management Perspective

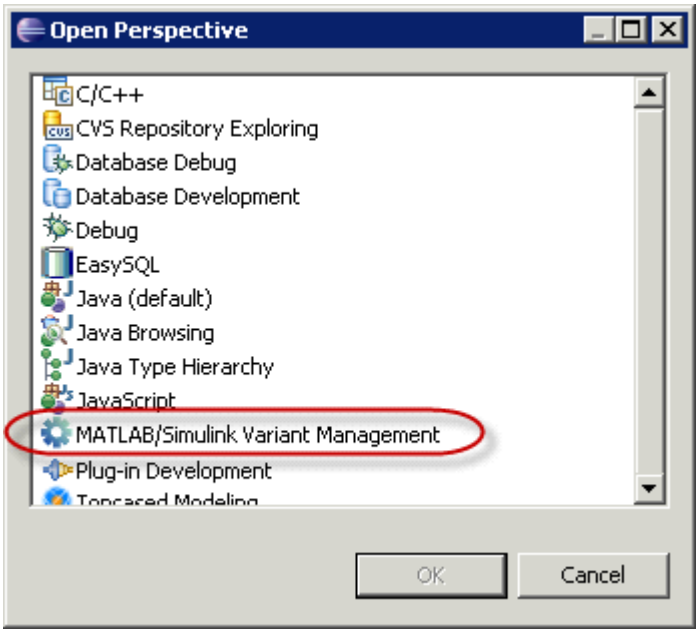
The MATLAB/Simulink Variant Management Perspective is available for working with pure::variants.

Figure 1. Opening the MATLAB/Simulink Variant Management Perspective



The perspective is activated via the "Window" menu with the menu item “Open Perspective -> Other…”.

Figure 2. Selecting MATLAB/Simulink Variant Management Perspective

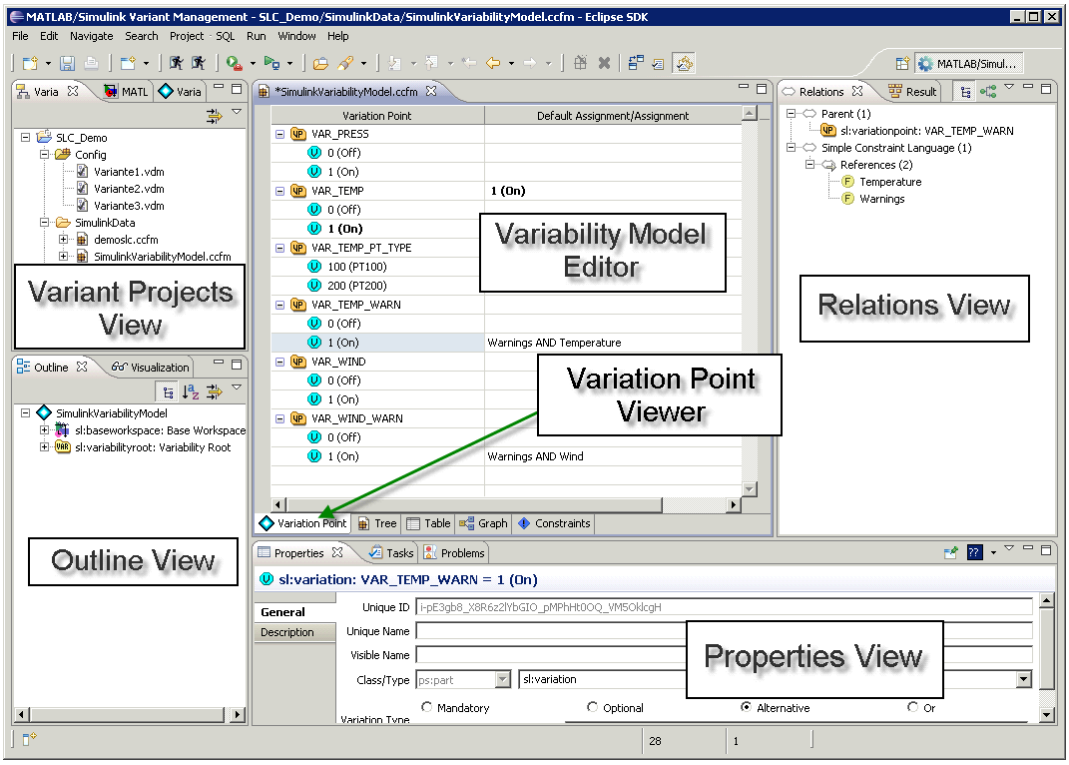


Next, the perspective “MATLAB/Simulink Variant Management” must be selected in the dialog that appears.

The activated perspective consists of a variability model opened in the editor, surrounded by a number of automatic views:

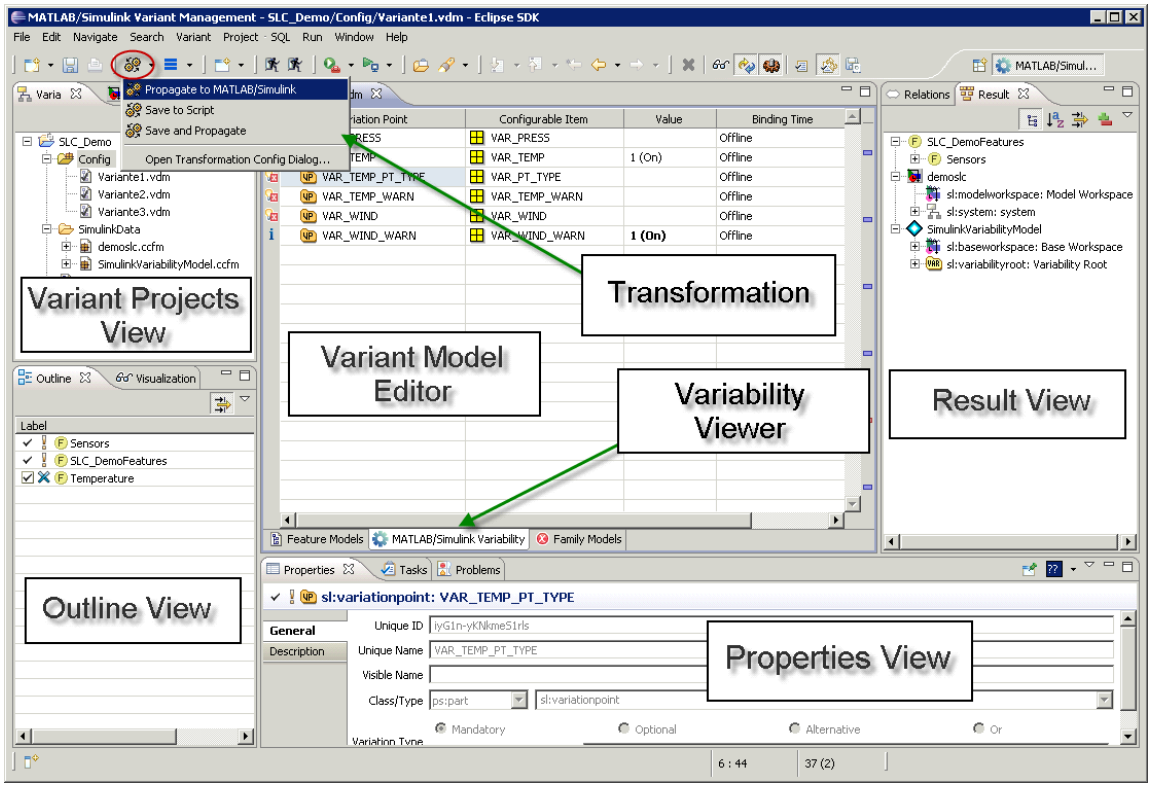
- Variability Model Editor for display and editing of the model
- Variation Point Viewer as a working view for the variability model in pure::variants
- Variant Projects View with all projects of the pure::variants workspace and their models
- Outline View with an overview of the model opened in the editor
- Relations View for displaying all relationships from and to the selected element in the editor
- Properties View for changing properties of the element selected in the editor

Figure 3. MATLAB/Simulink Variant Management Perspective (Modeling)



In addition to the Relations View, the Result View is also available for a variant model opened in the Variant Model Editor and provides an overview of the current configuration of the variability model. The editor also offers the Variation Point Viewer as a working view for configuration of a variability model. Among other actions, this configuration can be propagated to MATLAB/Simulink by means of a transformation.

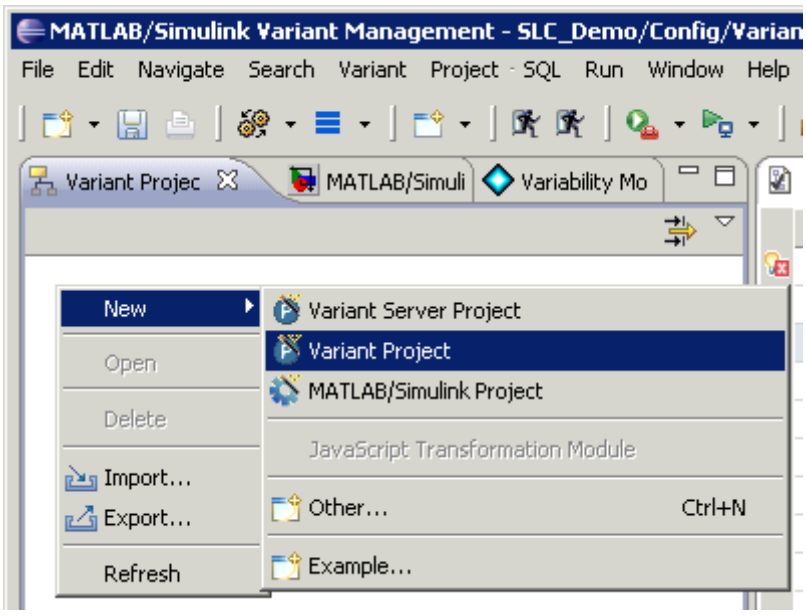
Figure 4. MATLAB/Simulink Variant Management Perspective (Configuration)



2.3. Creating a pure::variants Project

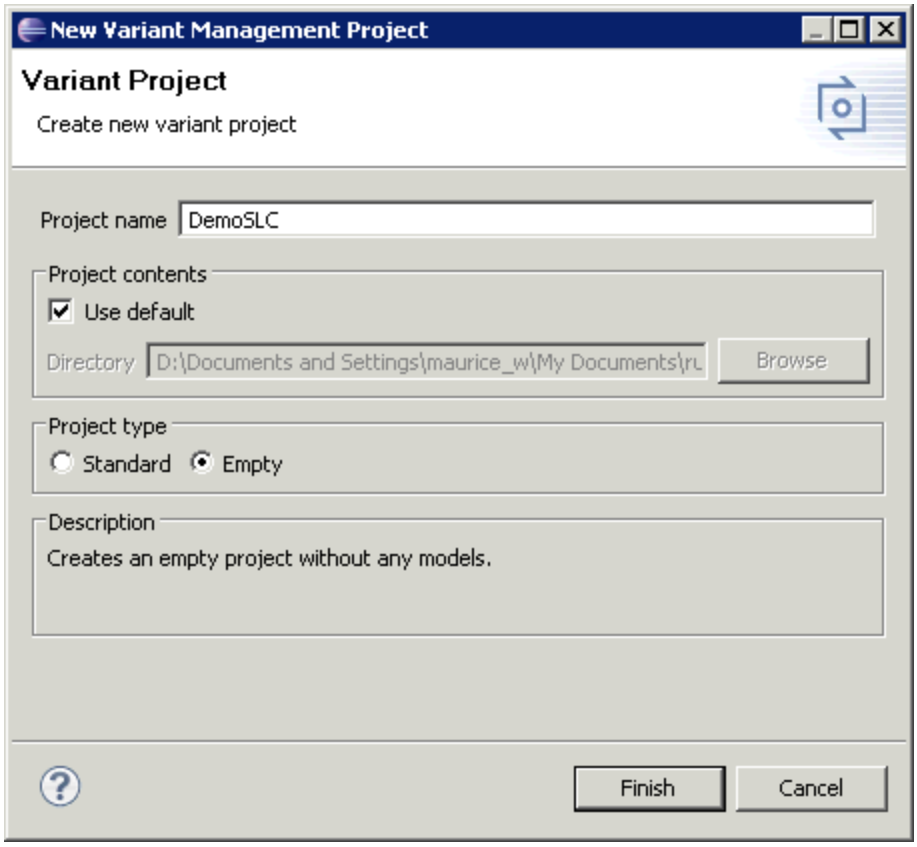
All models used for managing the variability can only be created within a pure::variants project or imported into such a project. The workspace of pure::variants is initially empty, and a new project must be created.

Figure 5. Creating a new pure::variants Project



Such a project is created with the context menu item “New -> Variant Project” of the “Variant Projects” view.

Figure 6. Settings for a pure::variants Project



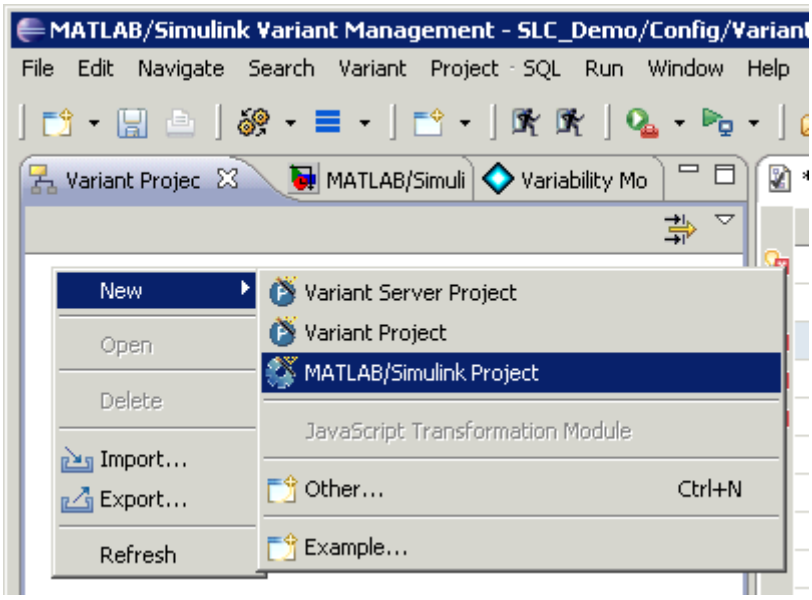
On the first page of the wizard that appears, the name of the new project is entered first. The configured location where the project should be created is the directory of the workspace in the file system. “Empty” is selected as the project type, which in contrast to a standard project has no pre-initialized structure with feature, family and variant models since special models such as the variability model are used for working with MATLAB/Simulink.

With the entered data, a pure::variants project can now be created and the wizard can be finished.

2.4. Creating a MATLAB/Simulink Project

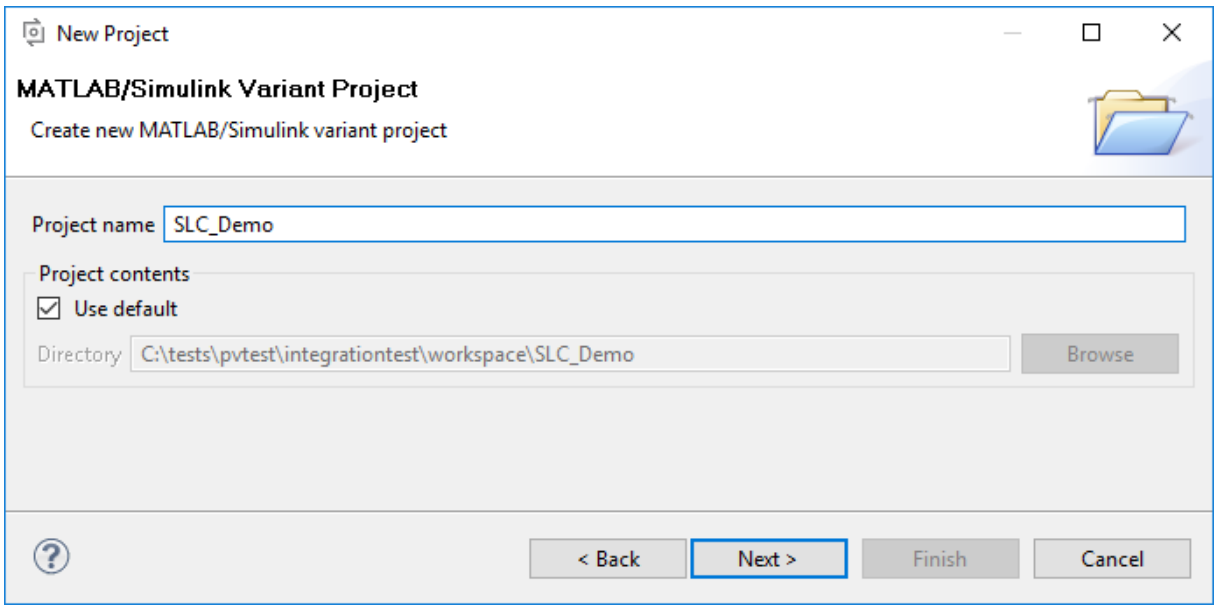
If MATLAB/Simulink is already configured (see Section 3, “Configuring MATLAB/Simulink”) and variation points have been modeled there, these can be imported directly upon creation of a new project. This is done by creating a pure::variants MATLAB/Simulink project that imports all variation points from MATLAB/Simulink upon creation and uses these to create a new variability model. In addition, a feature model and a variant model are created with the currently available variation point configuration in MATLAB/Simulink. Transformation configurations are also created that allow the variation point configuration to be written.

Figure 7. Creating a new pure::variants MATLAB/Simulink Project



A new pure::variants MATLAB/Simulink project is created via the context menu item “New -> MATLAB/Simulink Project” in the “Variant Projects” view.

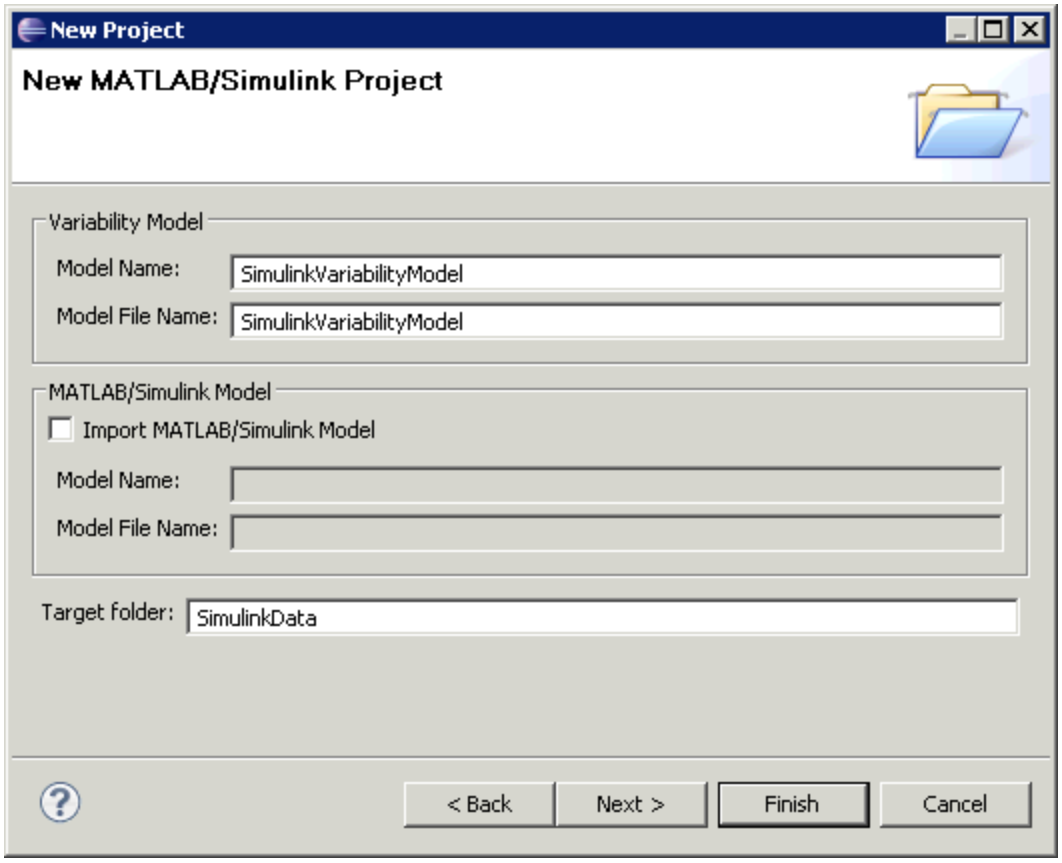
Figure 8. Settings for a pure::variants MATLAB/Simulink Project



On the first page of the wizard that appears, first enter the name of the new project. The configured location where the project should be created is the directory of the workspace in the file system.

On the second page of the wizard, all the settings for the data to be imported from MATLAB/Simulink are configured.

Figure 9. Simulink Models to be imported



All information about the imported variability model is entered in the group “Variability Model”. The name of the model and a deviating file name can be changed. By default the file name will be the same as the model name.

All information about the Simulink model in MATLAB/Simulink is entered in the group “MATLAB/Simulink Model”. By default, no Simulink model is imported to pure::variants. If you wish to import such a model, select the option “Import MATLAB/Simulink Model” in the group “MATLAB/Simulink Model” and enter the name of the Simulink model to import in the corresponding text field. A deviating file name can also be provided here. By default the file name will be the same as the model name.

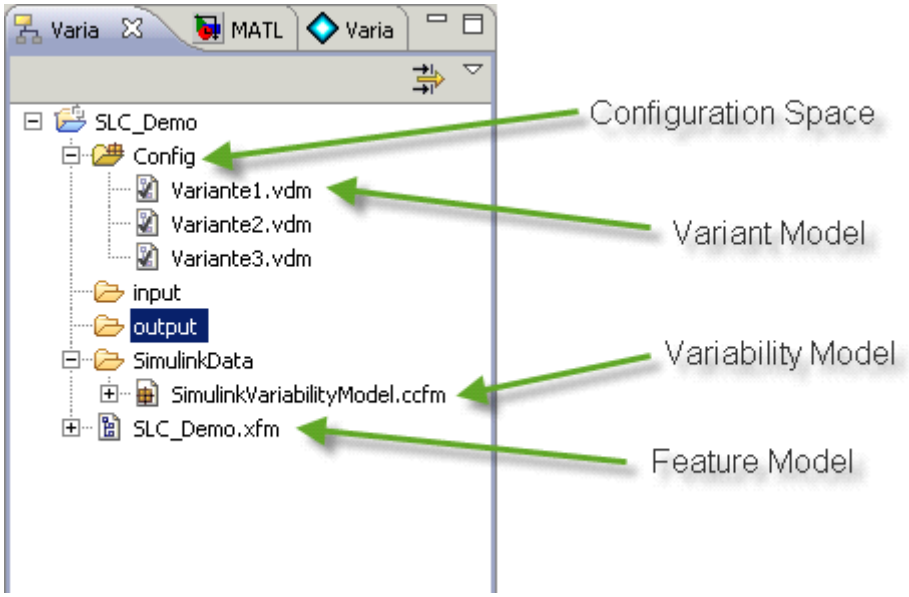
Note: The model name must match an existing model in the current MATLAB/Simulink directory. This is checked by the wizard, and an error message is displayed if no match is found.

The name of the directory in which the models imported by MATLAB/Simulink are created can be changed in the text field “Target folder”. If no directory is entered, the imported models are saved directly under the project. Creating such a directory allows models imported by MATLAB/Simulink to be kept separate from others.

The other pages are already preconfigured with default values, meaning that the wizard can be finished at this point. After the wizard is finished, the new project is created and the specified models are imported.

The new project now contains a variability model, a feature model, a Simulink model (if selected) and a configuration space with a variant model. Three transformations are configured for the configuration space these allow propagating and saving of a variation point configuration (see Section 6.8.3, “Activity: Propagating a Variation Point Configuration” and Section 6.8.4, “Activity: Saving a Variation Point Configuration”). The created variant model also contains the variation point configuration currently existing in MATLAB/Simulink.

Figure 10. MATLAB/Simulink Project Structure



In addition, the directories “input” and “output” are created; these are used by default as input and output directories by transformations.

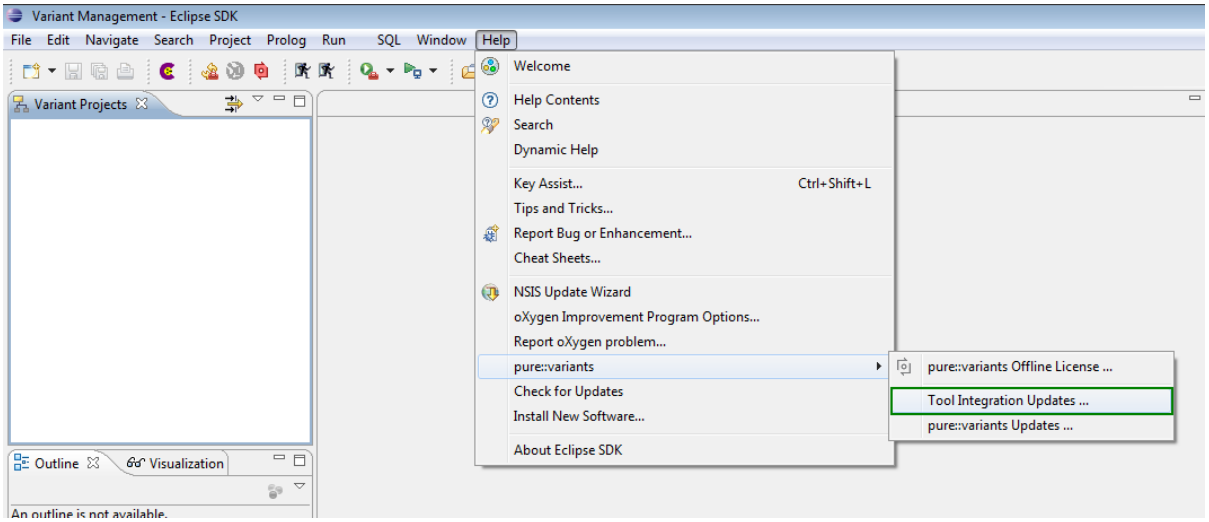
3. Configuring MATLAB/Simulink

In order to use pure::variants with MATLAB/Simulink, a special block set is required and a server used for communication must be started. The installation and configuration of this Variant Block Set and starting of the server are described below, as is opening of a Data Dictionary, which can be used as an alternative to the Simulink model for managing variation points and in which all variability information is persistently saved for that use case.

3.1. Installation of the Variant Block Set

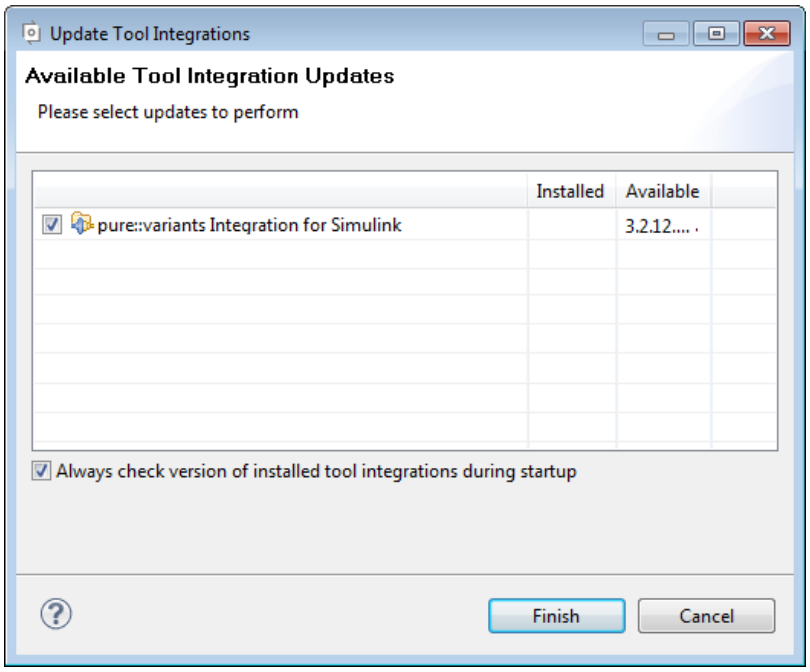
During installation of pure::variants, the Variant Block Set is automatically installed at the location specified by the user.

Figure 11. Installation of Tool Integrations



However, the Variant Block Set can also be installed or updated with the help of pure::variants via the menu item “Help -> pure::variants -> Tool Integration Updates...”.

Figure 12. Installation of the Variant Block Set



In the dialog that appears, select "pure::variants Integration for Simulink" and finish it. When the dialog is closed, the installer will be started automatically and installs the Variant Block Set at the location specified by the user. The "blockset" directory located below installation directory can be copied into MATLAB working directory.

3.2. Initializing the Variant Block Set and Starting the Server

If TargetLink is installed, TargetLink must be executed in simulation mode (not (**!**) stand-alone mode), in the current implementation of the Variant Block Set. This mode is set just once via the command *tl_switch_blockset* in the MATLAB console, and TargetLink then remains in simulation mode.

Then the Variant Block Set can be initialized with the command *init_variant_blockset* in the MATLAB console. The expected output in the MATLAB console upon successful initialization with *init_variant_blockset* is shown below.

Figure 13. Starting of the Variant Block Set

```
Server object created.
Creating registry on port 1099
Binding registry - remote reference alias is 'MatlabServer'
Server object registered ...
Matlab Server now available on port 1099 under alias 'MatlabServer'
```

```
Variant blockset/configurator environment was initialized successfully ...
Working path: C:\Programme\MATLAB\R2006b\work\variant_blockset
```

The initialiszation of the Variant Block Set can to automated so it is performed by MATLAB at MATLAB startup. To achive this put a startup.m file with the following content into the MATLAB startup folder.

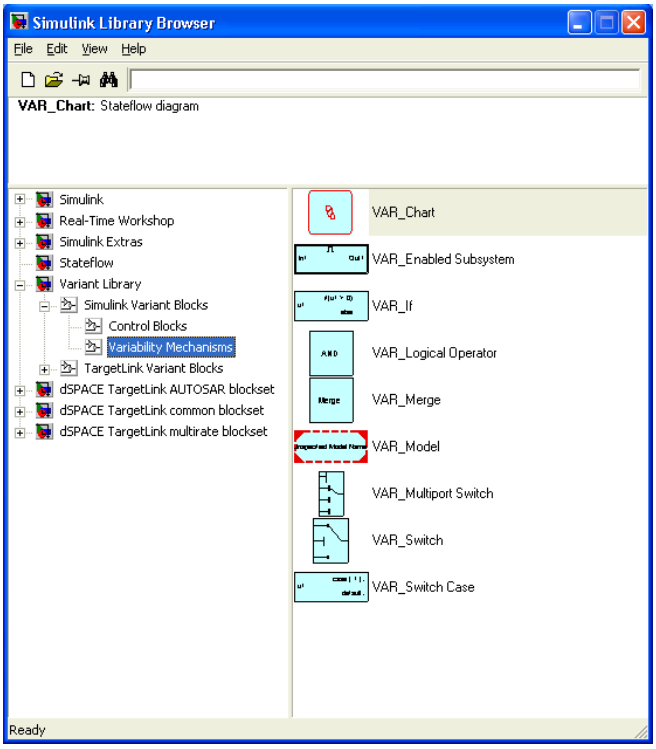
```
% Change directory to the Variant Block Set location
cd <Location of the Variant Block Set>

% perform initialization
init_variant_blockset
```

After performing the initialization, the Variant Block Set is ready for use. As can be seen in the output, the MATLAB server for communication was also started. If the server is not running, it can be started with the command *sl_server*. The initialization script must be executed after every restart of MATLAB, unless it is called automatically.

The variant block library installed with the Variant Block Set can now be used for modeling.

Figure 14. Variant Block Library



This library provides various variant blocks that can be dragged and dropped into the Simulink model.

3.3. Opening a Data Dictionary

Using a Data Dictionary to manage variability information in MATLAB is one possiblity. A Data Dictionary must be open in order for this information to be accessible, to create new information or edit existing information.

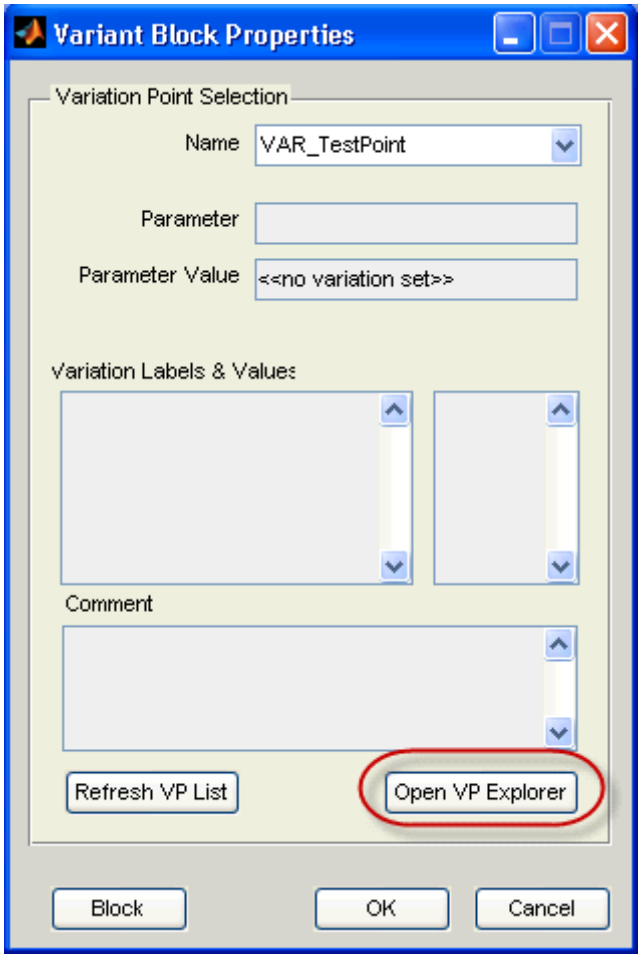
A Data Dictionary is opened by switching in MATLAB/Simulink to the directory containing the Data Dictionary and then executing the command *opendd* in the MATLAB console. By default, however, the Data Dictionary associated with a Simulink model is automatically opened upon opening of the model.

If no Data Dictionary exists yet when a Simulink model is opened, variability information can nevertheless be created. Upon the first use of the Variant Block Set, a Data Dictionary with the name “untitled.dd” is automatically created. All changes are initially stored in this Data Dictionary and held in memory and subsequently saved persistently with the option of renaming the dictionary.

3.4. Variation Point Explorer

The Variation Point Explorer is used in MATLAB/Simulink to edit the list of variation points. A variation point can be created, edited or deleted. Editing variation points includes configuration with a variation as well as the creation, changing or deleting of variations.

Figure 15. Variant Block Properties Dialog

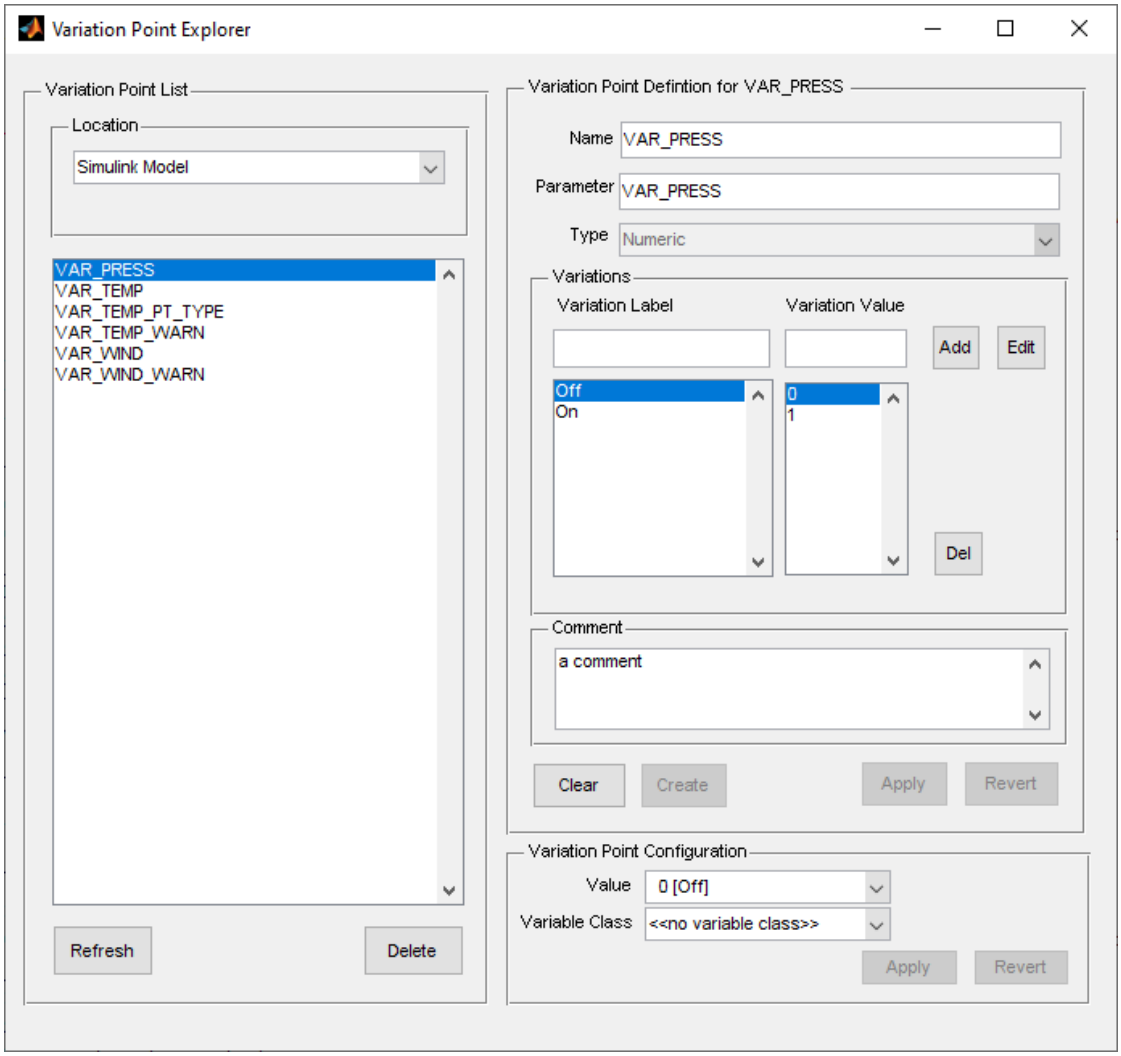


The Variation Point Explorer can be opened in two ways. In the first method, double-clicking on a block of the Simulink model that belongs to the Variant Block Set opens the dialog “Variant Block Properties”. The Variation Point Explorer can be opened here with “Open VP Explorer”.

Clicking on “Block” opens the original dialog for displaying and editing the block properties that is overridden by the “Variant Block Properties” dialog.

In the second method, the MATLAB command *vp_explorer* can be used in the MATLAB console. Upon opening the Variation Point Explorer via the block, the assigned variation point is selected; otherwise the first block in the list is selected, if one exists. The input form pre-initialized in this way must be cleared with “Clear” in order to avoid accidentally modifying the selected variation point when creating a new one.

Figure 16. Variation Point Explorer



With the help of the “Location” combo box it can be determined, if the information regarding the variation point shall be stored in the TargetLink Data Dcitionary or or in the Simulink model. “Create” creates a variation point, and “Apply” confirms changes and writes them to the Data Dictionary. All as yet unconfirmed changes for a variation point can be reset to the original values configured upon opening of the Variation Point Explorer by clicking “Revert”.

4. Procedure Overview

4.1. Roles in pure::variants

pure::variants differentiates between four different roles in working with variant-rich Simulink models. One user can take on all four roles:

- MATLAB/Simulink Variability Modeler
- pure::variants Variability Modeler
- Features Modeler
- Variant Creator

4.1.1. MATLAB/Simulink Variability Modeler

The MATLAB/Simulink variability modeler captures and models such variability as arises in Simulink models due to alternative or optional blocks or signal flows. For this purpose, the modeler expands the Simulink model with the various variability mechanisms of the Variant Block Set, which are available via variant blocks of the variant block library. In addition, he makes use of the Variation Point Explorer to model variation points consisting of variations (name-value pairs) that depict the variability in the respective variant blocks. Finally, he also assigns the corresponding variation point to a variant block via the block's properties dialog.

4.1.2. pure::variants Variability Modeler

The pure::variants variability modeler captures and models variability that arises in Simulink models due to new requirements and the resulting features. For this purpose, he expands the variability model in pure::variants with variation points consisting of variations that can subsequently depict the variability in the Simulink model. In addition, he places these new features into relation with the variations by inserting conditions into the variability model that express a conflict or a required aspect of a feature. Because the modeled variation points are required for generation of a variant of this Simulink model in MATLAB/Simulink, these must also be available and kept synchronized there.

4.1.3. Features Modeler

The features modeler captures and models shared and different features of all variants of the Simulink model as features in pure::variants using a feature model. In addition, he creates possible dependencies between features with the help of relations or restrictions. He also combines these features with variations of a variation point by inserting conditions into the variability model. However, if these variation points were captured exclusively using the MATLAB/Simulink variability modeler, they must first be imported into pure::variants in a variability model.

4.1.4. Variant Creator

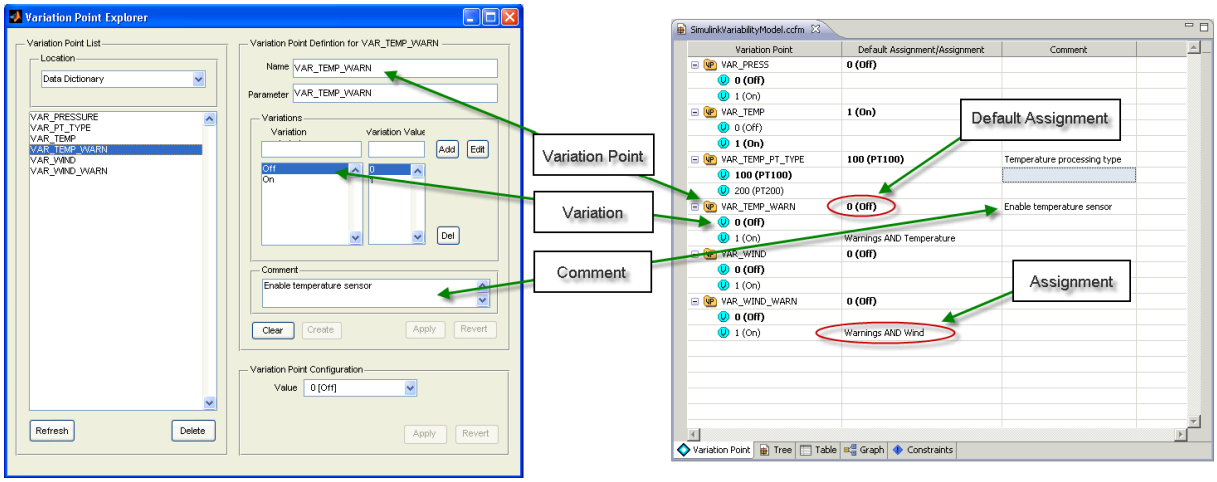
The variant creator configures the captured and modeled variability and generates variants of the Simulink model. He must first create a variant model in pure::variants that serves for configuration of the variation points. Then he selects variations and features in the created variant model for the automatic selection of variations, thereby generating a variation point configuration. In conclusion, he propagates this variation point configuration to MATLAB/Simulink, whereby the values of the selected variations are automatically written to the parameters of the corresponding variant blocks of the Simulink model.

4.2. MATLAB/Simulink vs. pure::variants

When working in pure::variants, the additional "Variation Point" viewer is available in the editor (of the family model) for editing the variability model and the additional "MATLAB/Simulink Variability" viewer is available in the editor (of the variant model) for configuration of the variation points. In addition, the Variation Point Explorer is available in MATLAB/Simulink for editing the variation points.

4.2.1. Editing Value-based Variation Points

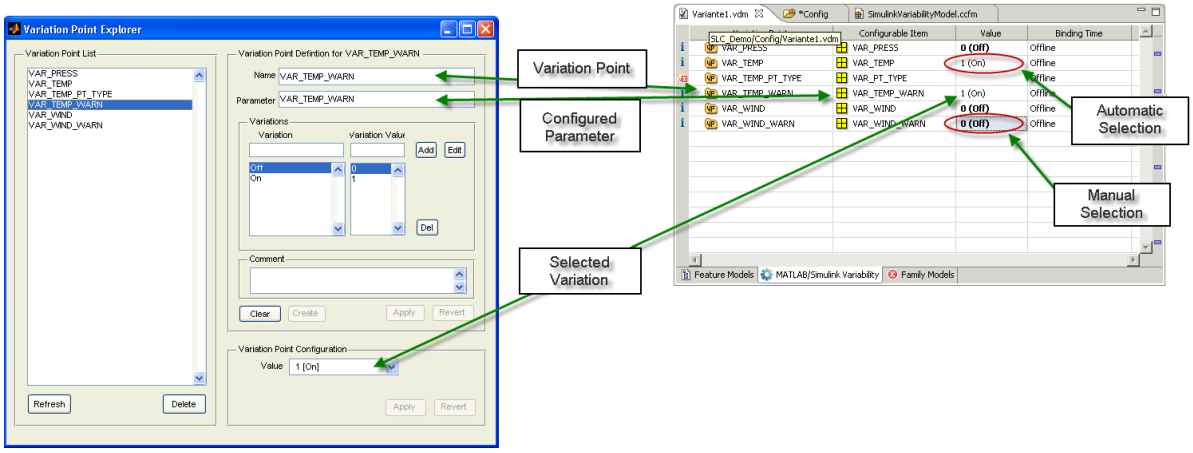
Figure 17. Editing value-based Variation Points



The diagram compares the options for editing the value-based variation points in MATLAB/Simulink on the left with those in pure::variants on the right. On the left, in the Variation Point Explorer, they are shown as a list and exclusively the information of the selected variation point is displayed, such as name, parameter (only on left) and variations. On the right, in the “Variation Point” viewer, they are shown with their variations in a table that offers an overview of the information of all variation points in the columns. It also shows the default assignment of a variation point only available in pure::variants as well as the conditions of the assignments of the individual variations.

4.2.2. Configuring Variation Points

Figure 18. Configuring value-based Variation Points



The diagram compares the options for configuring the value-based variation points in MATLAB/Simulink on the left with those in pure::variants on the right. On the left, in the Variation Point Explorer, the currently selected variation is shown for the selected variation point. On the right, in the “MATLAB/Simulink Variability” viewer, all variation points are listed in a table and all variations in the corresponding column (selectable) as well as the selected variation. In addition, manually selected variations as well as variations selected automatically on the basis of features can be differentiated on the right based on display differences.

4.3. Usage Scenarios

Based on three different scenarios, various activities within the procedures are employed below to demonstrate how to handle variability in variant-rich Simulink models.

The starting point is a weather station for which the modules for processing temperature, wind speed and pressure are already modeled in a Simulink model. In addition, a feature model already exists in pure::variants that is used for configuring variation points and creating a variant of the Simulink model (see also Section 6.5, “Creating a Feature and Variability Model”).

4.3.1. Adding Variability Starting with a Simulink Model

The Simulink model of the weather station contains a module for processing the wind speed. This is measured by a sensor and shown in a display. When the maximum permitted wind speed is exceeded, a warning is also always triggered. This signal for the warning should now be made switchable, which results in the use of variant blocks of the Variant Block Set.

MATLAB/Simulink	**pure::variants**
The Variant Block Set offers a Switch variant block that passes on one of the two input signals depending on the value at the control input. This value is made available by a Constant variant block. *This is done by dragging and dropping the Effect block “VAR_Switch” and the Control block “VAR_Constant” from the variant block library into the Simulink model (Section 5.5.1, “Activity: Adding a Variant Block”).*	
Setting or configuring the value of the Control block takes place by selecting a variation of a variation point. *It is therefore also necessary to create the variation point “VAR_Warn” (Section 5.1.1, “Activity: Creating a New Variation Point in Data Dictionary”) and the corresponding variations “Off(0)” and “On(1)” (Section 5.1.3, “Activity: Creating a New Variation”).* The value of the selected variation is used by the Control block as a constant, which is stored in a workspace parameter. *To do this, the Control block must be assigned the variation point “VAR_Warn” (Section 5.5.2, “Activity: Assigning a Variation Point to a Variant Block”).*	
	For the subsequent configuration in pure::variants, the variation point created in MATLAB/Simulink must be available. *This requires that the variation point has to be synchronized to pure::variants (Section 5.4.1, “Activity: Synchronizing to pure::variants”).* Passing of the warning should be automatically switchable via an already existing feature “Warnings”. *This requires linking the variation “On(1)” with the feature “Warnings” (Section 6.7.1, “Activity: Linking a Variation with Features”).* If the feature is not selected, the warning should not be passed.

	This requires that the variation "Off(0)" be set as the default assignment (Section 6.2.2, "Activity: Changing a Variation").
	By selecting the feature "Warnings", the variation point "VAR_Warn" is automatically configured with the variation "On(1)" and otherwise with the variation "Off(0)" (with warning / without warning). *This requires creation of a variation point configuration (Section 6.8.1, "Activity: Creating a Variation Point Configuration").* This variation point configuration is then used in MATLAB/Simulink by the variant blocks of the Simulink model. *It is therefore necessary to propagate this variation point configuration to MATLAB/Simulink (Section 6.8.3, "Activity: Propagating a Variation Point Configuration").*

After the propagation, the value of the variation is automatically available to the Control block for switching the signal flow of the Effect block and the warning in the module for processing the wind speed is switched on or off accordingly.

4.3.2. Adding Variability Starting with a Feature

The weather station contains a module for processing the temperature. This module should be designed to be switchable based on a requirement so that there can also be weather stations without temperature measurement.

MATLAB/Simulink	**pure::variants**
	The requirement is depicted by a feature that can be selected or deselected in a configuration. *The "Temperature" feature is modeled in the feature model of the weather station (Section 6.5.2, "Activity: Modeling Features").*
	Switching the module for processing the temperature on or off is depicted at a variation point with the corresponding variations. *It is therefore necessary to first create a variation point "VAR_Temp" (Section 6.1.2, "Activity: Creating a New Variation Point") and the corresponding variations "On(0)" (as default assignment) and "Off(1)" (Section 6.1.3, "Activity: Creating a New Variation").* The module must be switchable via the feature "Temperature". *This requires linking the variation "On(1)" with the feature "Temperature" (Section 6.7.1, "Activity: Linking a Variation with Features").*
	The variation point created in pure::variants must be available in MATLAB/Simulink.

	This requires that the variation point be synchronized to MATLAB/Simulink (Section 6.4.1, "Activity: Synchronizing to MATLAB/Simulink").
Matlab/Simulink offers an Enabled block that makes the module switchable based on the value of a Constant variant block from the Variant Block Set. *This requires that the Enabled block and the Control block be added to the Simulink model via drag and drop (see here Section 5.5.1, "Activity: Adding a Variant Block").* The value of the selected variation is used by the Control block as a constant, which is stored in a workspace parameter. *To do this, the Control block must also be assigned the variation point "VAR_Temp" (Section 5.5.2, "Activity: Assigning a Variation Point to a Variant Block").*	
	By selecting the feature "Temperature", the variation point "VAR_Temp" is automatically configured with the variation "On(1)" and otherwise with the variation "Off(0)" (with temperature / without temperature). *This requires creation of a variation point configuration (Section 6.8.1, "Activity: Creating a Variation Point Configuration").* This variation point configuration is then used in MATLAB/Simulink by the Control block of the Simulink model. *It is therefore necessary to propagate this variation point configuration to MATLAB/Simulink (Section 6.8.3, "Activity: Propagating a Variation Point Configuration").*

After the propagation, the value of the variation is automatically available to the Control block for switching the Enabled block and the module for processing the temperature is switched on or off accordingly.

4.3.3. Adding Variability Starting with a Variation Point

The Simulink model of the weather station contains a module for processing the temperature, which is measured with a sensor of type PT100 and adjusted in a calculation with the help of the constant 100. As a result of further development of the weather station, a second sensor of type PT200 is integrated. However, the additional constant 200 is required for adjusting the measured temperature.

MATLAB/Simulink	**pure::variants**
The two constants must be depicted at one variation point with the corresponding variations. *It is therefore necessary to create the variation point "VAR_PT" (Section 5.1.1, "Activity: Creating a New Variation Point in Data Dictionary") and the two variations "PT100(100)" and "PT200(200)" (Section 5.1.3, "Activity: Creating a New Variation").*	

The respective constants are made available with the help of a Constant variant block (Control block) from the Variant Block Set. *This requires that the Control block "VAR_Constant" be dragged and dropped from the variant block library into the Simulink model (Section 5.5.1, "Activity: Adding a Variant Block").* The value of the selected variation is stored in a workspace parameter and is provided by the Control block as a constant for adjusting the temperature. *To do this, the Control block must also be assigned the variation point "VAR_PT" (Section 5.5.2, "Activity: Assigning a Variation Point to a Variant Block").*	
	For the subsequent configuration in pure::variants, the variation point created in MATLAB/Simulink must be available. *This requires that the variation point be synchronized to pure::variants (Section 5.4.1, "Activity: Synchronizing to pure::variants").*
	Selecting the variation "PT100(100)" or "PT200(200)" causes the variation point "VAR_PT" to be configured accordingly (constant 100 / constant 200). *This requires creation of a variation point configuration (Section 6.8.1, "Activity: Creating a Variation Point Configuration").* This variation point configuration is then used in MATLAB/Simulink by the Constant variant block to provide the respective constant. *It is therefore necessary to propagate this variation point configuration to MATLAB/Simulink (Section 6.8.3, "Activity: Propagating a Variation Point Configuration").*

After the propagation, the value of the variation is available to the Control block and the module for processing the temperature can use the constant 100 for type PT100 or 200 for type PT200 for adjusting the sensor values.

5. MATLAB/Simulink Procedures for Value-based Variability

5.1. Creating Variability Information

5.1.1. Activity: Creating a New Variation Point in Data Dictionary

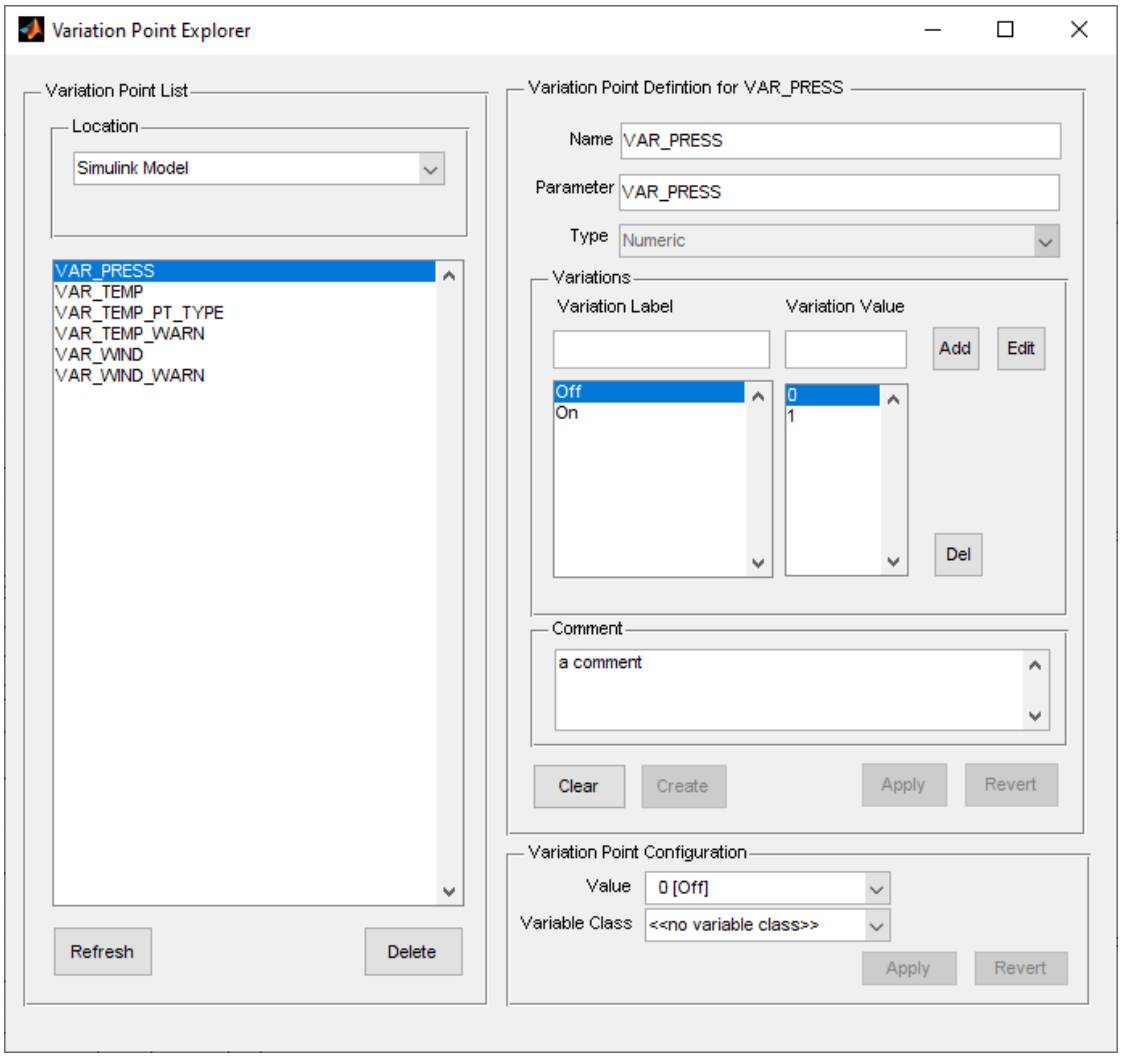
Prerequisites & Preliminary Work

- The Data Dictionary associated with the Simulink model must be open.
- A variation point with the new name must not yet exist in this Data Dictionary.

Process

The Variation Point Explorer is used for creating a new variation point in MATLAB/Simulink. Because a variation point (if present) is already selected upon opening the Variation Point Explorer, the input form on the right must be reset with “Clear”.

Figure 19. Creating a new Variation Point in Data Dictionary



A name that does not yet exist on the left side of the Variation Point Explorer must be entered in the text field as the name for the variation point. If the name does not satisfy the conventions for a variation point, the Explorer reports this problem in the status line on the bottom left. For more information on conventions, see Section 1.2, “Conventions”.

The name of the workspace parameter (Parameter) to be used can also be entered; this is automatically created if it does not already exist. In addition, one or more variations can also be added for the new variation point (see Section 5.1.3, “Activity: Creating a New Variation”).

The combo "Type" remains disabled for the Data Dictionary. Only "Numeric" type is allowed here, which allows numbers as values for the variations only.

A comment can also be added for the variation point in the group “Comment”. Like the other information, this comment can also be used in pure::variants after synchronization.

Creation of the variation point is completed with “Create”. The new variation point has now been added to the Data Dictionary and appears on the left side of the variation point list.

The new variation point can then be configured in the group “Variation Point Configuration” if variations were created for it. To do this, select a variation in the combo box and confirm with “Apply”.

Note: A parameter must be entered for configuration of the variation point.

After creating the variation point, both the Variation Point Explorer and the “Variant Block Properties” dialog can be closed.

Other Actions

To be able to use the new variation point for the variant configuration in pure::variants, the corresponding variability model must be synchronized in pure::variants (see Section 5.4.1, “Activity: Synchronizing to pure::variants”).

See also:

- Section 5.5.2, “Activity: Assigning a Variation Point to a Variant Block”

5.1.2. Activity: Creating a New Variation Point in Simulink model

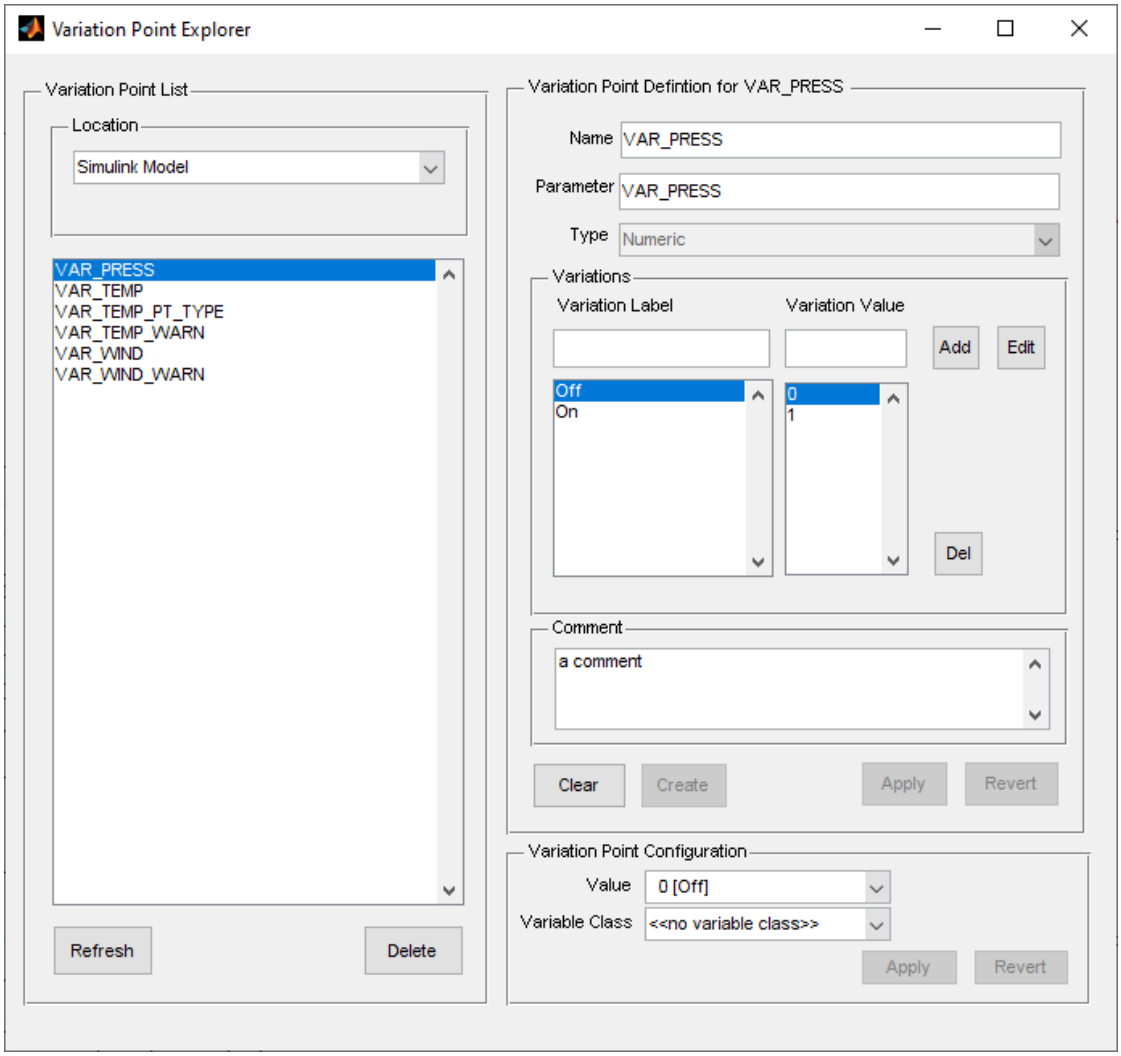
Prerequisites & Preliminary Work

- The Simulink model must be open.
- A variation point with the new name must not yet exist in this Simulink model.

Process

The Variation Point Explorer is used for creating a new variation point in MATLAB/Simulink. Because a variation point (if present) is already selected upon opening the Variation Point Explorer, the input form on the right must be reset with “Clear”.

Figure 20. Creating a new Variation Point in Simulink model



A name that does not yet exist on the left side of the Variation Point Explorer must be entered in the text field as the name for the variation point. If the name does not satisfy the conventions for a variation point, the Explorer reports this problem in the status line on the bottom left. For more information on conventions, see Section 1.2, “Conventions”.

The name of the workspace parameter (Parameter) to be used can also be entered; this is automatically created if it does not already exist. In addition, one or more variations can also be added for the new variation point (see Section 5.1.3, “Activity: Creating a New Variation”).

The type of the variation point can be defined with the combo "Type". Allowed types are "String" and "Numeric". The type can be defined only during variation point creation and is not changeable afterwards. Type "String" allows all string as Variation values, "Numeric" instead allows numbers only.

A comment can also be added for the variation point in the group “Comment”. Like the other information, this comment can also be used in pure::variants after synchronization.

Creation of the variation point is completed with “Create”. The new variation point has now been added to the Data Dictionary and appears on the left side of the variation point list.

The new variation point can then be configured in the group “Variation Point Configuration” if variations were created for it. To do this, select a variation in the combo box and confirm with “Apply”.

Note: A parameter must be entered for configuration of the variation point.

After creating the variation point, both the Variation Point Explorer and the “Variant Block Properties” dialog can be closed.

Other Actions

To be able to use the new variation point for the variant configuration in pure::variants, the corresponding variability model must be synchronized in pure::variants (see Section 5.4.1, “Activity: Synchronizing to pure::variants”).

See also:

- Section 5.5.2, “Activity: Assigning a Variation Point to a Variant Block”

5.1.3. Activity: Creating a New Variation

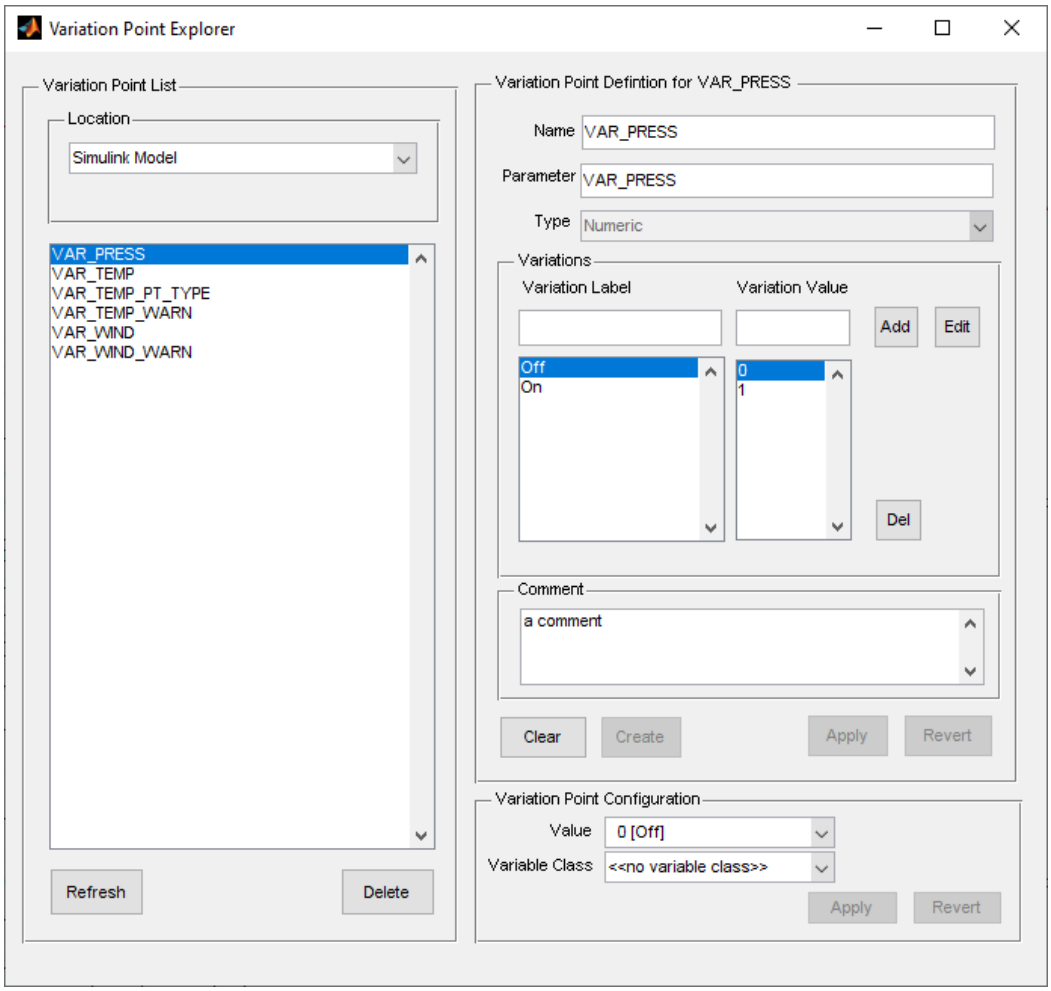
Prerequisites & Preliminary Work

- The Simulink model or the Data Dictionary associated with the Simulink model must be open.
- The variation point of the new variation must exist in the Data Dictionary or Simulink model.
- A variation with the new name or value must not yet exist in this variation point.

Process

The Variation Point Explorer is used for creating a new variation in MATLAB/Simulink. Select the variation point for which a variation should be created in the left side of the Explorer. The input form on the right side is automatically initialized with the current information of the variation point and all possible variations are listed in the group “Variations”.

Figure 21. Creating a new Variation



A name (label) and a value must be entered in the text fields of the group “Variations”. The name and value must be unique for the selected variation point. If the name or value does not satisfy the conventions for a variation, the Explorer reports this problem during the creation in the status line on the bottom left. For more information on conventions, see Section 1.2, “Conventions”.

The variation is added to the variation point with “Add”, whereby the changed variation point with the new variation is only added to the Data Dictionary upon clicking “Apply” in the group “Variation Point Definition”.

The changed variation point can then be configured with the new variation in the group “Variation Point Configuration”. To do this, select it in the combo box and confirm with “Apply” in this group.

After creating the variation and applying the changes to the variation point, both the Variation Point Explorer and the “Variant Block Properties” dialog can be closed.

Other Actions

To be able to use the new variation for the variant configuration in pure::variants, the corresponding variability model must be synchronized in pure::variants (see Section 5.4.1, “Activity: Synchronizing to pure::variants”).

5.2. Changing Variability Information

5.2.1. Activity: Changing a Variation Point

Prerequisites & Preliminary Work

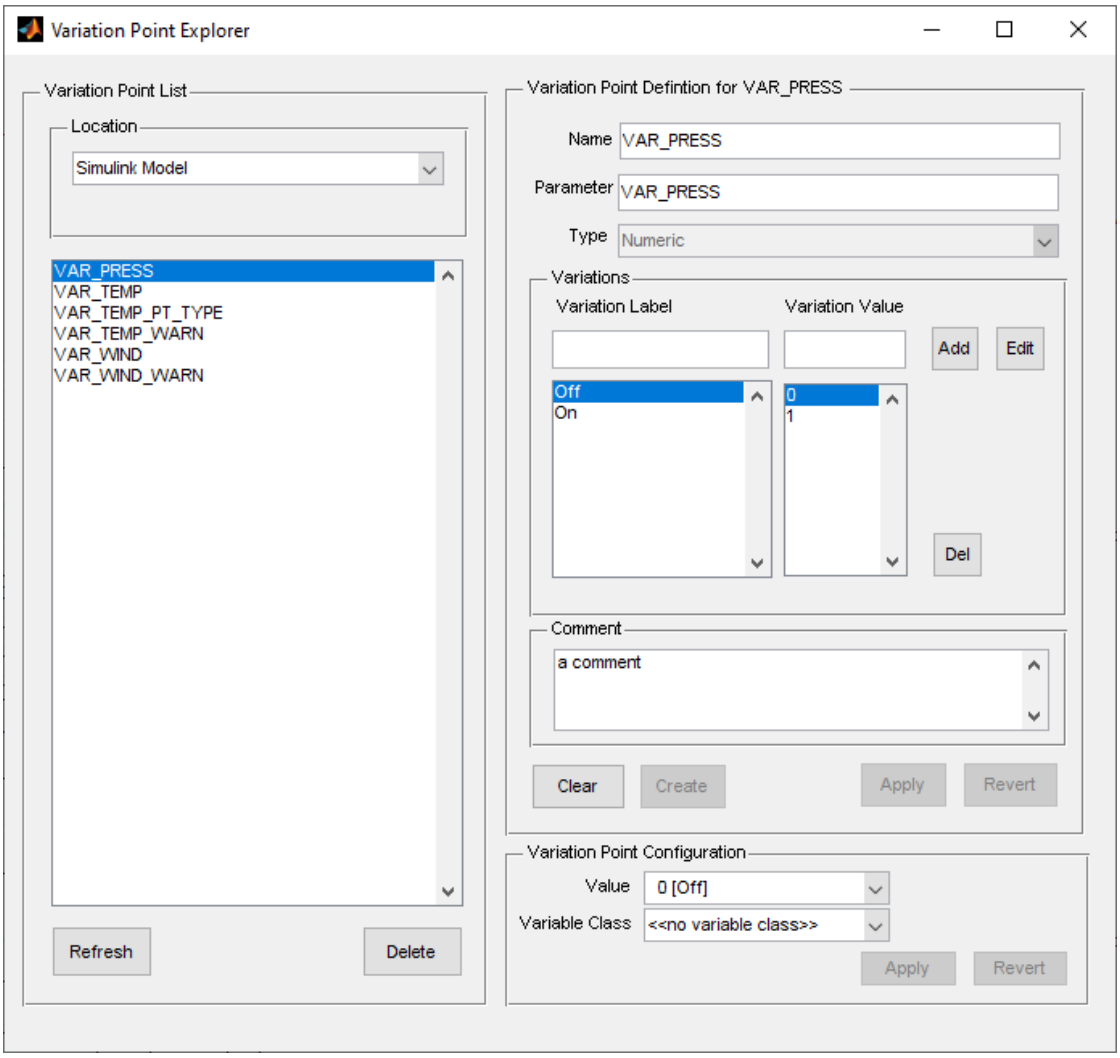
- The Simulink model or the Data Dictionary associated with the Simulink model must be open.

- The variation point to be changed must exist in this Data Dictionary or Simulink model.

Process

The Variation Point Explorer is used for changing a variation point in MATLAB/Simulink. Select this variation point on the left side of the Explorer. The input form on the right side is automatically initialized with the information of the variation point.

Figure 22. Changing a Variation Point



A changed name that does not yet exist must be entered in the text field as the name for the variation point. If the name does not satisfy the conventions for a variation point, the Explorer reports this problem in the status line on the bottom left. For more information on conventions, see Section 1.2, “Conventions”. To change a variation of a variation point, see also Section 5.2.2, “Activity: Changing a Variation”.

The type can not be changed. The type can only be defined during variation point creation.

The comment for the variation point can also be changed in the group “Comment”. Like the other changed information, this comment can also be used in pure::variants after synchronization.

If necessary, the name of the parameter can also be changed. It must be noted here that the previously used parameter is not automatically deleted, although the new one is automatically created.

The change to the variation point is completed with “Apply”. The changed variation point has now also been changed in the Data Dictionary or Simulink model.

Note: It is possible to change a variation point such that it no longer contains any variations. Such a variation point can no longer be configured.

The variation point can be reconfigured in the group “Variation Point Configuration” if variations exist for it. To do this, select a variation in the combo box and confirm with “Apply”.

Other Actions

To be able to use the changed variation point for the variant configuration in pure::variants, the corresponding variability model must be synchronized in pure::variants (see Section 5.4.1, “Activity: Synchronizing to pure::variants”).

5.2.2. Activity: Changing a Variation

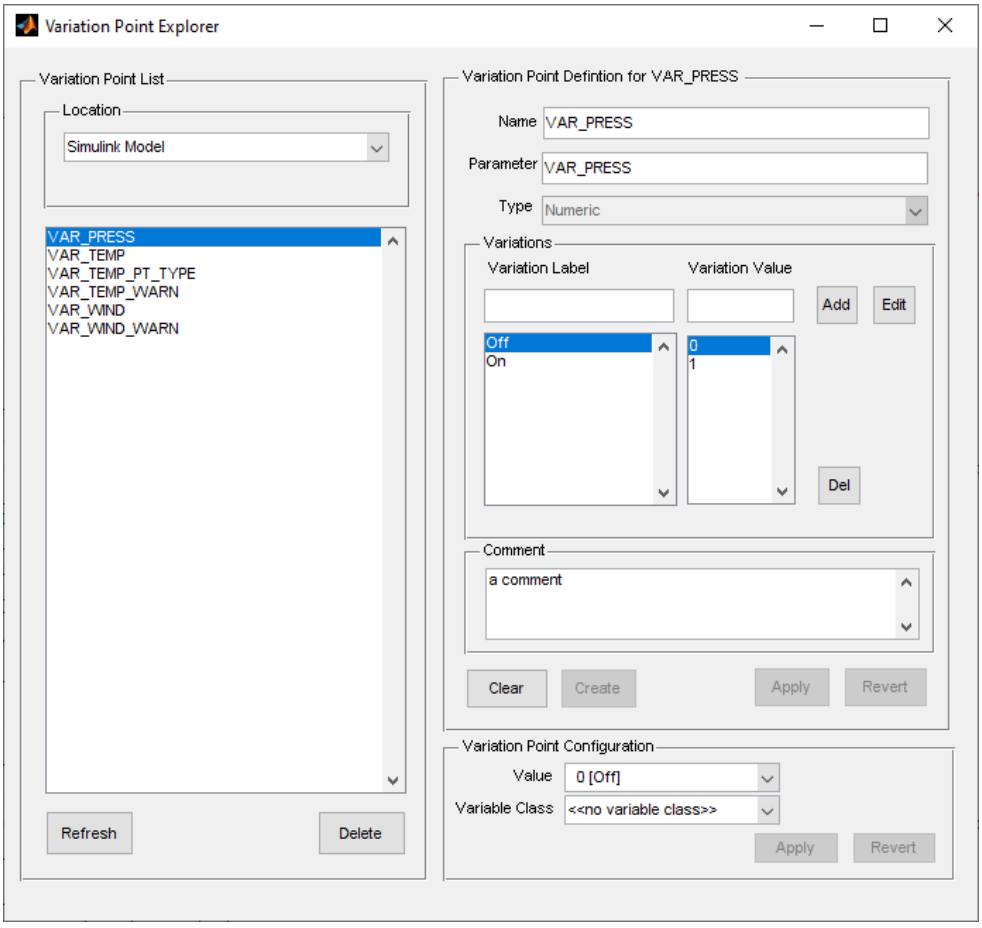
Prerequisites & Preliminary Work

- The Simulink model or the Data Dictionary associated with the Simulink model must be open.
- The variation to be changed must exist at a variation point in this Data Dictionary or Simulink model.

Process

The Variation Point Explorer is used for changing a variation in MATLAB/Simulink. Select the variation point that contains the corresponding variation in the left side of the Explorer. The input form on the right side is automatically initialized with the current information of the variation point and all defined variations of the variation point are listed in the group “Variations”. In the list, select the variation to be changed via its name (Variation Label) or value (Variation Value).

Figure 23. Changing a Variation



The changed name or value must be entered into the text fields. The changed name and value must be unique for the selected variation point. If the name or value does not satisfy the conventions for a variation, the Explorer reports this problem in the status line on the bottom left. For more information on conventions, see Section 1.2, “Conventions”.

The changing of the variation is completed with “Edit”. Clicking “Apply” in the group “Variation Point Definition” transfers the changed variation point with the changed variation to the Data Dictionary or Simulink model.

The variation point can then be configured with the changed variation in the group “Variation Point Configuration”. To do this, select the variation in the combo box and confirm with “Apply”.

After changing the variation and changing variation point, both the Variation Point Explorer and the “Variant Block Properties” dialog can be closed.

Other Actions

To be able to use the changed variation for the variant configuration in pure::variants, the corresponding variability model must be synchronized in pure::variants (see Section 5.4.1, “Activity: Synchronizing to pure::variants”).

5.3. Deleting Variability Information

5.3.1. Activity: Deleting a Variation Point

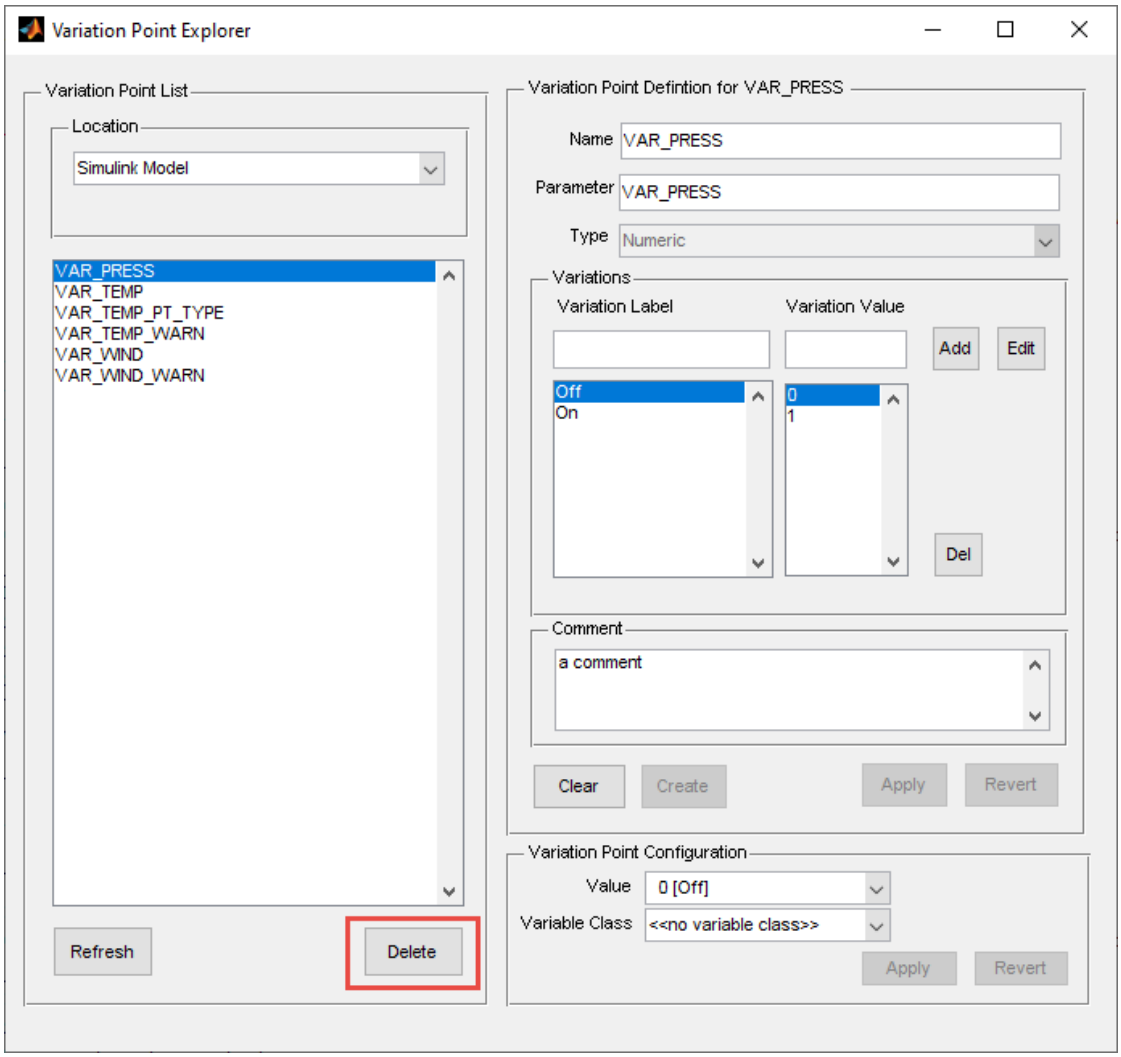
Prerequisites & Preliminary Work

- The Simulink model or the Data Dictionary associated with the Simulink model must be open.
- The variation point to be deleted must exist in this Data Dictionary or this Simulink model.

Process

The Variation Point Explorer is used for deleting a variation point in MATLAB/Simulink. The variation point can be selected in the left side of the Explorer and deleted as well as removed from the Data Dictionary or Simulink model with “Delete”.

Figure 24. Deleting a Variation Point



After deleting the variation point, both the Variation Point Explorer and the “Variant Block Properties” dialog can be closed. It must be noted here that the parameter previously used by the variation point is not automatically deleted.

Other Actions

The deleted variation point is no longer available for a configuration in MATLAB/Simulink. To remove the deleted variation point from pure::variants as well so that it is no longer available for variation point configuration, the corresponding variability model must be synchronized in pure::variants (see Section 5.4.1, “Activity: Synchronizing to pure::variants”).

5.3.2. Activity: Deleting a Variation

Prerequisites & Preliminary Work

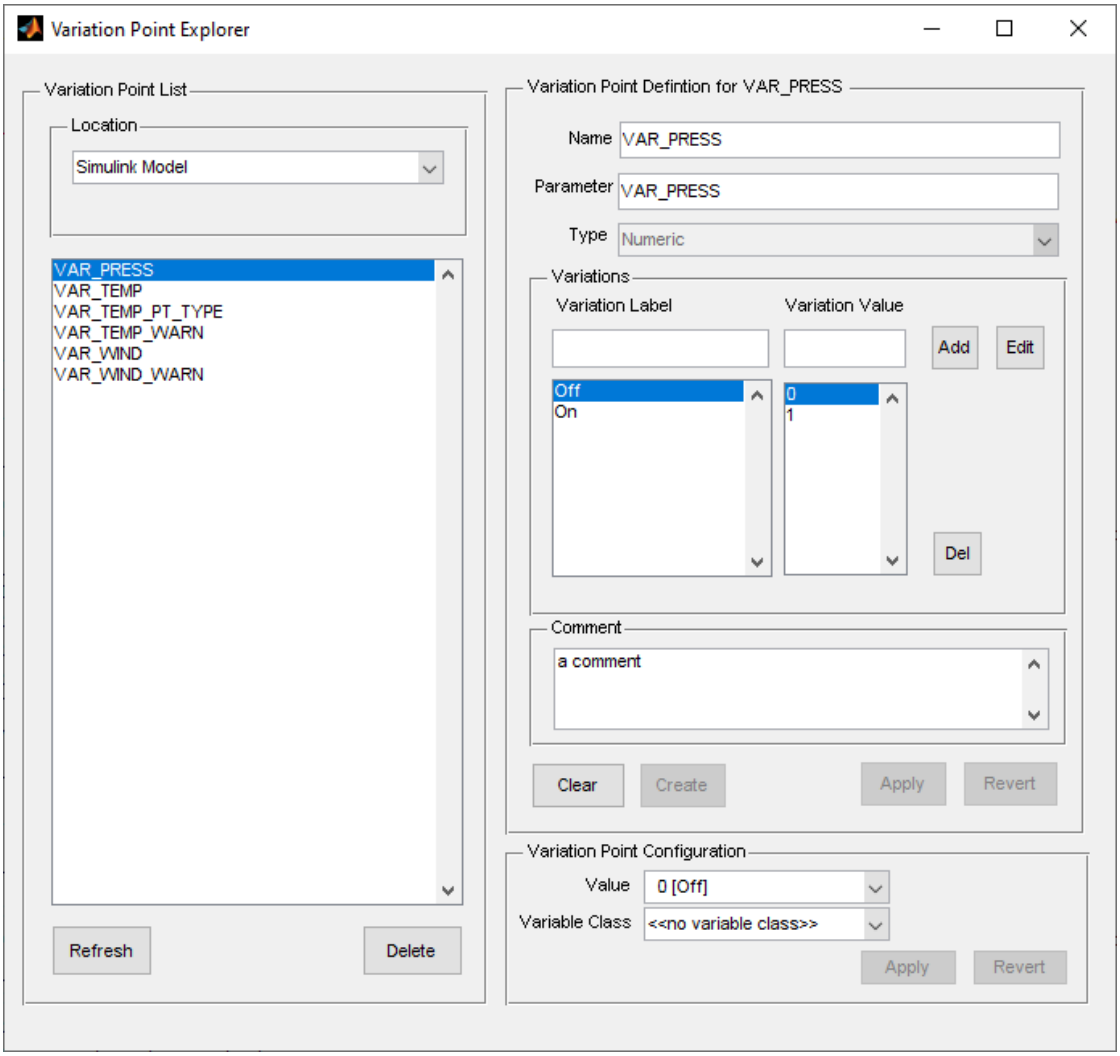
- The Simulink model or the Data Dictionary associated with the Simulink model must be open.
- The variation to be deleted must exist at a variation point in this Data Dictionary or this Simulink model.

Process

The Variation Point Explorer is used for deleting a variation in MATLAB/Simulink. Select the variation point that contains the corresponding variation in the left side of the Explorer. The input form on the right side is automatically initialized with the current information of the variation point and all possible variations are listed

in the group “Variations”. In the list, select the variation to be deleted via its name (Variation Label) or value (Variation Value).

Figure 25. Deleting a Variation



The selected variation is removed from the list of all variations with “Del”. Clicking “Apply” in the group “Variation Point Definition” transfers the changed variation point with the deleted variation to the Data Dictionary or Simulink model.

After deleting the variation and changing the variation point, both the Variation Point Explorer and the “Variant Block Properties” dialog can be closed.

Other Actions

The deleted variation is no longer available for a configuration in MATLAB/Simulink. To remove the deleted variation from pure::variants as well so that it is no longer available for variation point configuration, the corresponding variability model must be synchronized in pure::variants (see Section 5.4.1, “Activity: Synchronizing to pure::variants”).

5.4. Synchronizing Variability Information to pure::variants

5.4.1. Activity: Synchronizing to pure::variants

Prerequisites & Preliminary Work

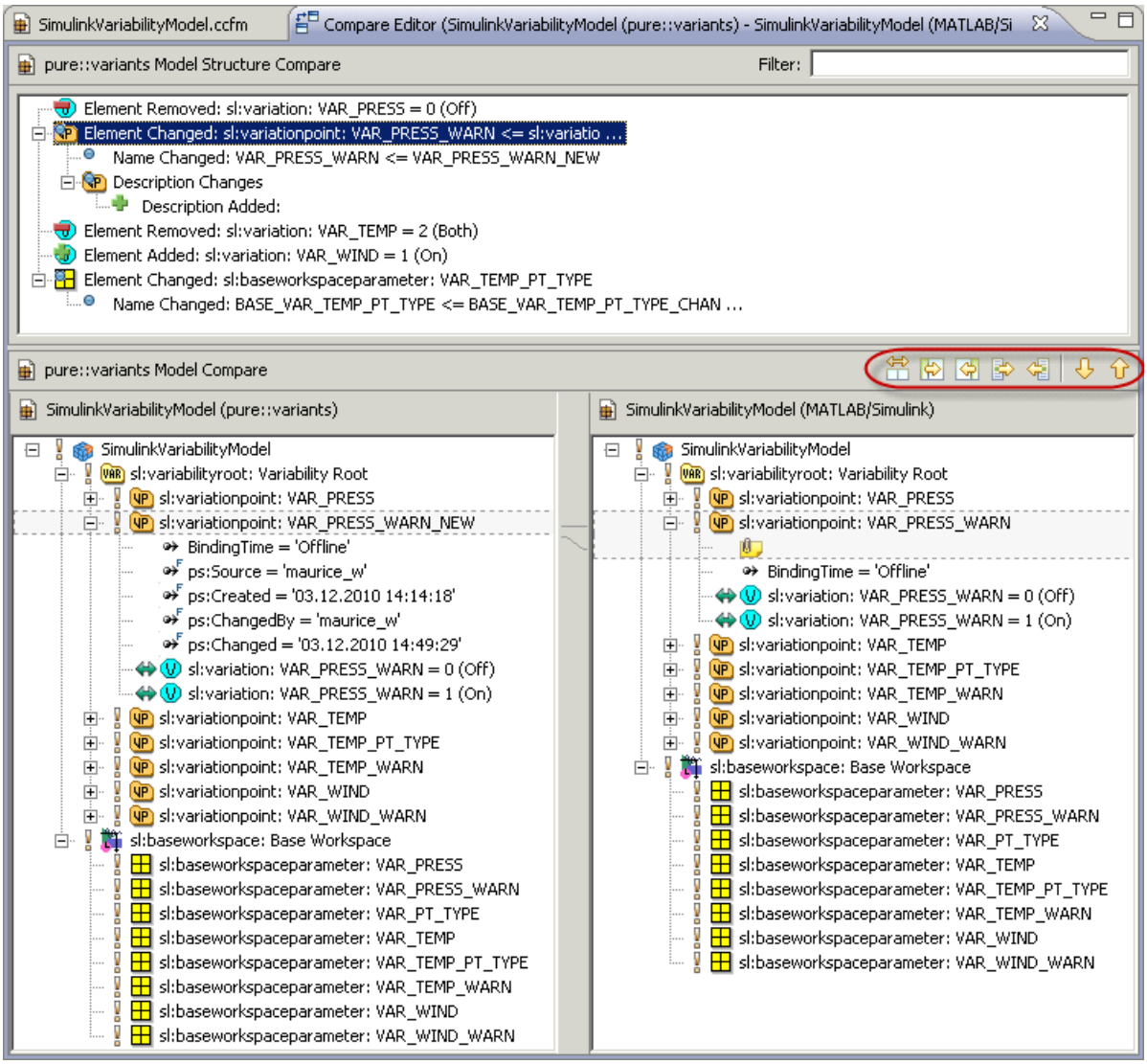
- The Simulink model or the Data Dictionary associated with the Simulink model must be open.

- The variability model associated with the Data Dictionary or the Simulink model must be open in pure::variants.

Process

The Compare Editor in pure::variants is used to synchronize variability information from MATLAB/Simulink to pure::variants. This compares the differences between the current variability information of MATLAB/Simulink, which was automatically imported, and the information of pure::variants to be changed. It is opened via the toolbar of pure::variants with the synchronization button ( ), which is only visible when a variability model is open.

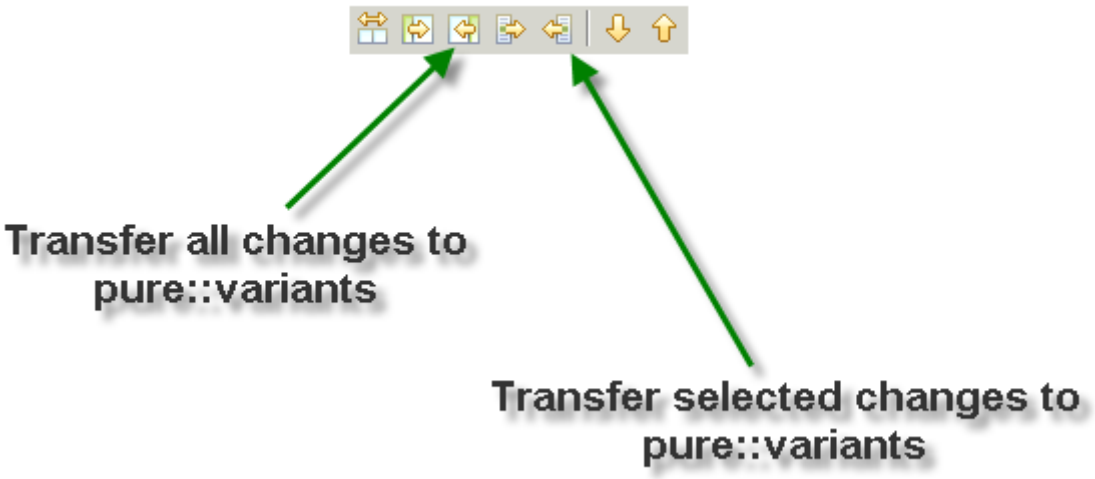
Figure 26. Synchronizing to pure::variants



The lower left side shows the variability information of the variability model of pure::variants and the lower right side shows the current variability information of MATLAB/Simulink as a newly imported variability model. The top part shows all changes in the two models. Marking a change causes the associated elements in the lower trees to be highlighted and the differences are displayed there more precisely.

The Compare Editor makes it possible to transfer these changes from MATLAB/Simulink (right side) to pure::variants (left side). For this purpose, there is a toolbar immediately below the listed changes that offers various functions in the different directions (according to the arrows). In each case, a tool tip briefly describes the possible action.

Figure 27. Toolbar of Compare Editor



Changes can be accepted individually or all at once. It should be noted that changes accepted in the variability model in pure::variants are initially only available in the model currently open in memory. In order to permanently accept the changes in pure::variants, the variability model must either be saved via the Compare Editor or the actual model editor.

5.5. Adding Variability to Simulink Models

5.5.1. Activity: Adding a Variant Block

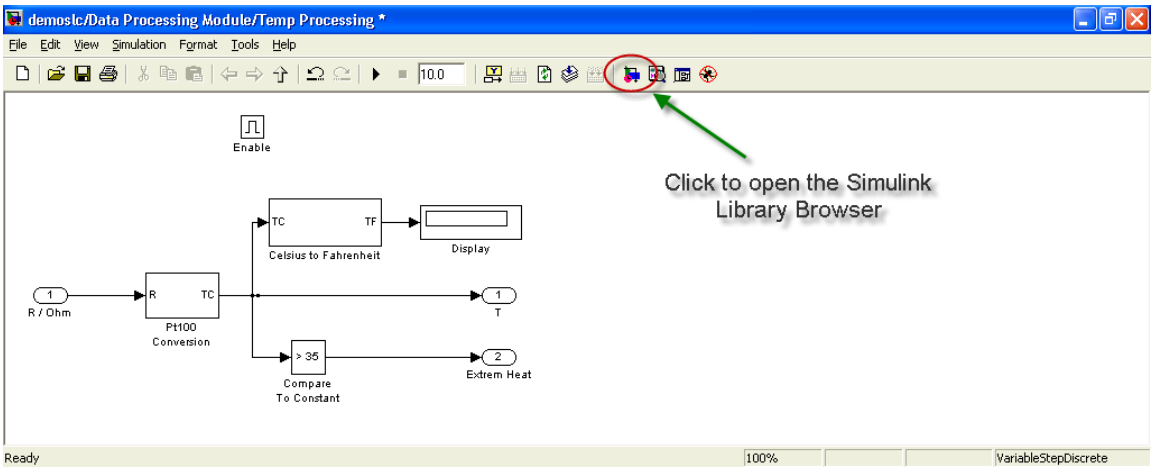
Prerequisites & Preliminary Work

- The Variant Block Set must be installed and initialized (see Section 3.1, “Installation of the Variant Block Set” and Section 3.2, “Initializing the Variant Block Set and Starting the Server”).
- The Simulink model must be open.

Process

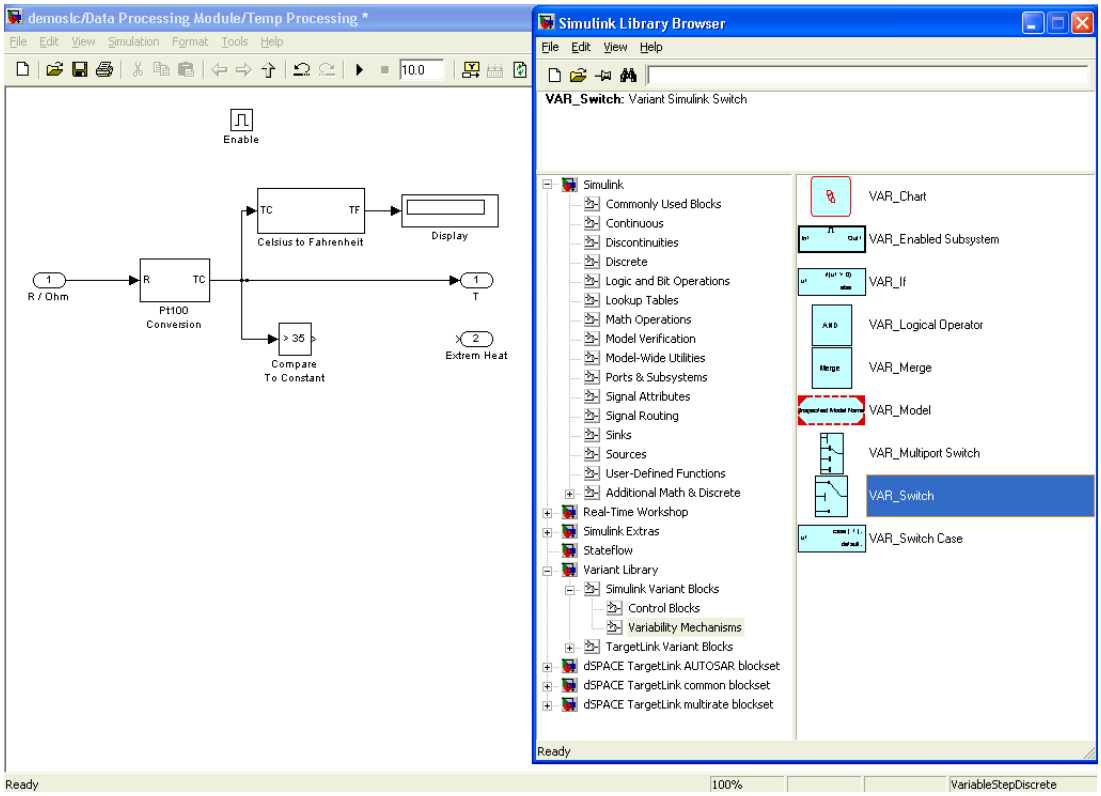
The Simulink Library Browser is used to add a variant block to a Simulink model.

Figure 28. Opening the Variant Block Library



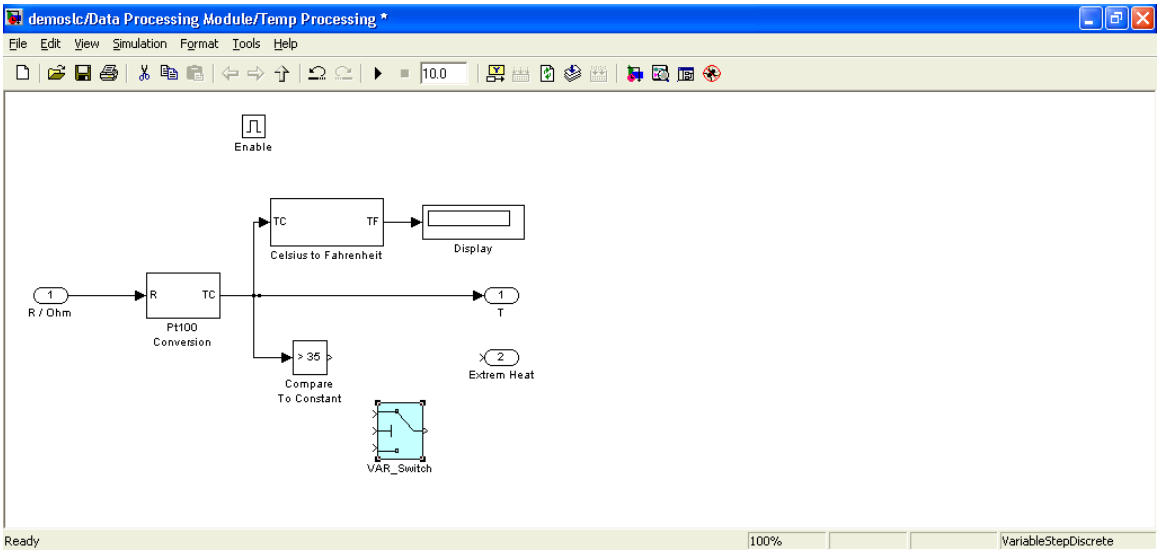
The Simulink Library Browser lists all available Simulink blocks and TargetLink blocks (TargetLink has to be installed.) as well as all variant blocks that are located in the variant block library.

Figure 29. Selecting a Variant Block



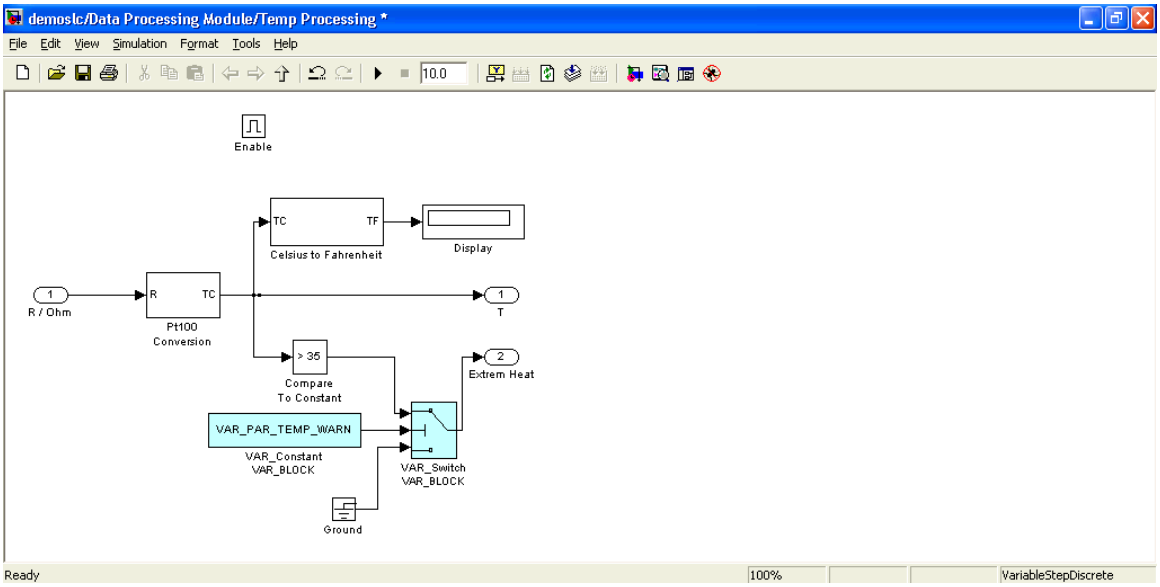
For example, a Switch variant block should be inserted into the open Simulink model that either passes on a signal or ignores it by using the signal of the Ground block.

Figure 30. Inserting a Variant Block



The control input of the variant block “VAR_Switch” is then used to configure which signal is passed on. The selection of the corresponding signal is controlled by the variant block “VAR_Constant” by assigning a corresponding value to the constant. This is also added to the Simulink model by dragging and dropping from the Simulink Library Browser.

Figure 31. Connecting a Variant Block



Afterward, all other outputs and inputs of the blocks must be connected accordingly.

Other Actions

In order that the control block can configure the signal flow of the switch, it must be assigned to a variation point (see Section 5.5.2, “Activity: Assigning a Variation Point to a Variant Block”). With the help of the variation point, the control input of the switch is set such that it either passes on the signal of the Ground block or the other signal.

5.5.2. Activity: Assigning a Variation Point to a Variant Block

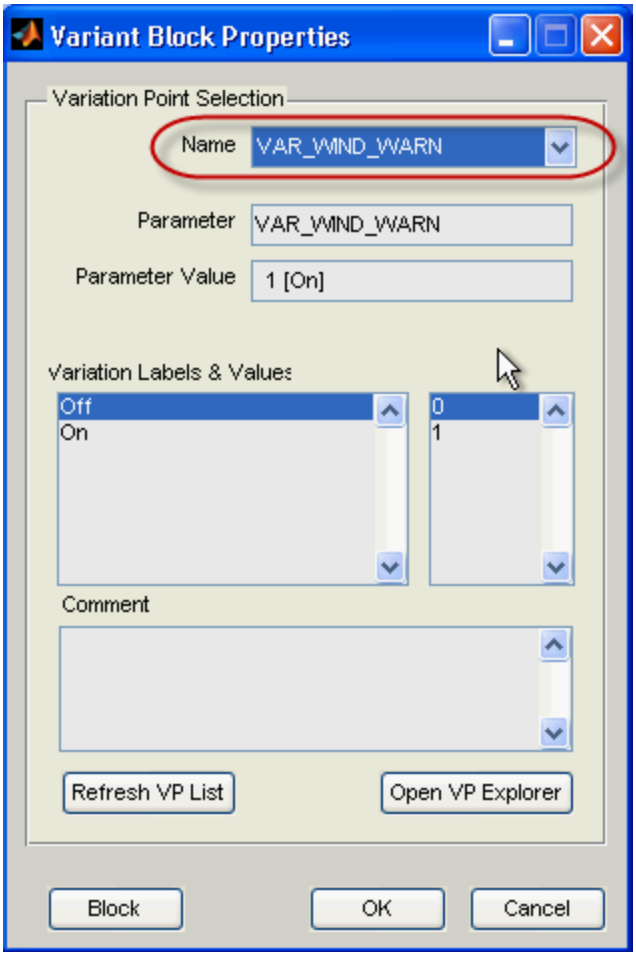
Prerequisites & Preliminary Work

- The Simulink model must be open and must contain at least one variant block.
- If the variation point is located in a Data Dictionary the Data Dictionary must be open.

Process

To assign a variation point to a variant block, its “Variant Block Properties” dialog is used. This can be opened either by double-clicking on the variant block or via its context menu entry “Open Block”.

Figure 32. Opening Variant Block Properties Dialog



In the “Variant Block Properties”, the combo box with the name of a variation point must be selected. After this has been selected, the fields of the dialog are filled out automatically and display additional information about the selected variation point.

Note: Changes to the selected variation point are not possible in the “Variant Block Properties” dialog. The Variation Point Explorer, where variation points can be edited, can be opened via the “VP Explorer” (see Section 5.2.1, “Activity: Changing a Variation Point”).

After selecting a variation point, the dialog “Variant Block Properties” can be closed.

Other Actions

With the help of the specified variation point, the variant block can now be configured independently of a variation.

6. pure::variants Procedures

6.1. Creating Variability Information

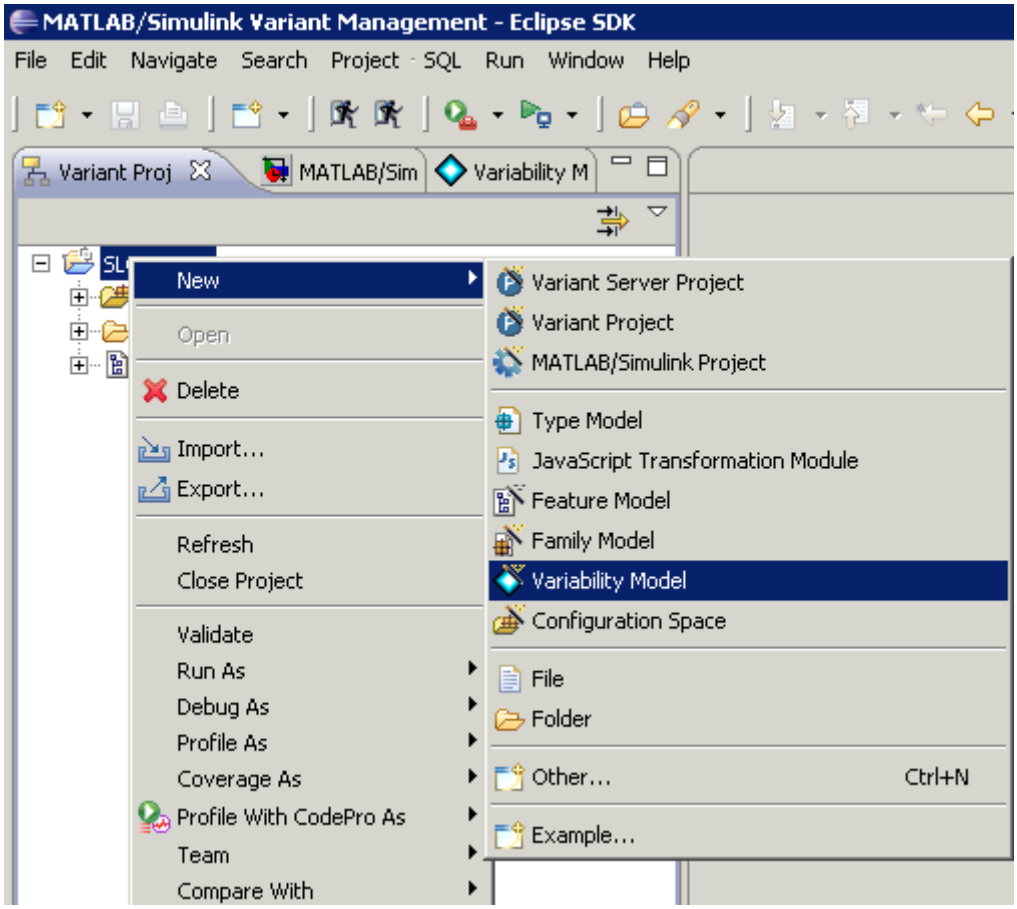
6.1.1. Activity: Creating a Variability Model

Process

A variability model is required for modeling variability points in pure::variants. Depending on whether the modeling is begun in pure::variants or variation points have already been modeled in MATLAB/Simulink, a variability model is created or generated from the variation points simulated from MATLAB/Simulink.

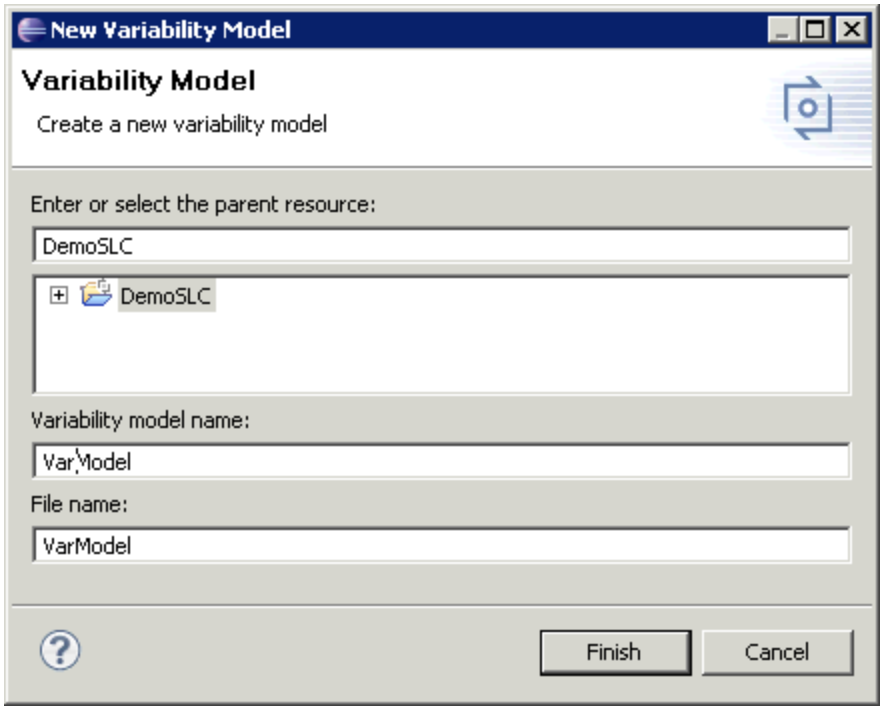
To create a new variability model in pure::variants, select the project that should contain the new model in the view “Variant Projects” (see also Section 2.3, “Creating a pure::variants Project”). It can then be created via the context menu item “New -> Variability Model”.

Figure 33. Creating a new Variability Model



On the first page of the wizard that appears, the chosen project is already selected as target directory. The name of the model to be created must be entered into the text field “Variability model name”. Optionally, a file name that differs from the model name may be entered into the text field “File name”.

Figure 34. Settings for a Variability Model



When the wizard is closed, the new model is created within the project. To be able to model new variation points in pure::variants, the new model must be opened by double-clicking in the “Variant Projects” view.

6.1.2. Activity: Creating a New Variation Point

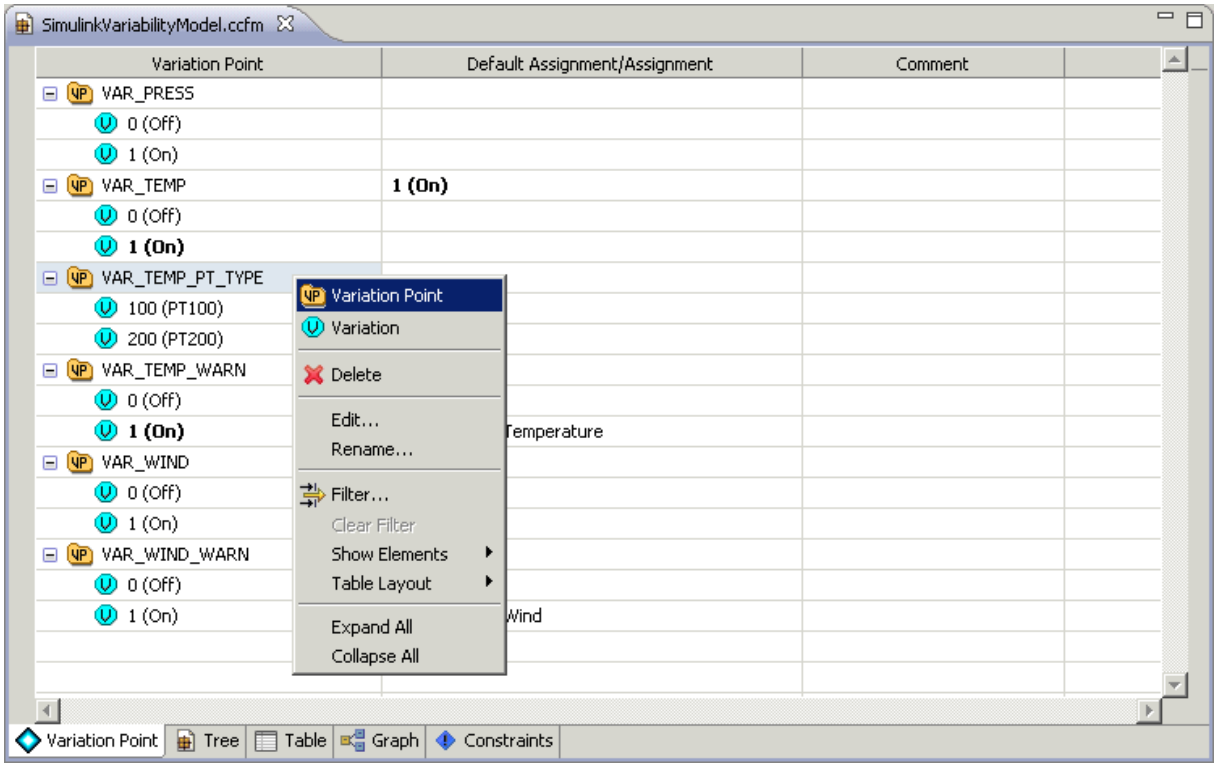
Prerequisites & Preliminary Work

- The variability model must be open.
- A variation point with the new name must not yet exist in this variability model.

Process

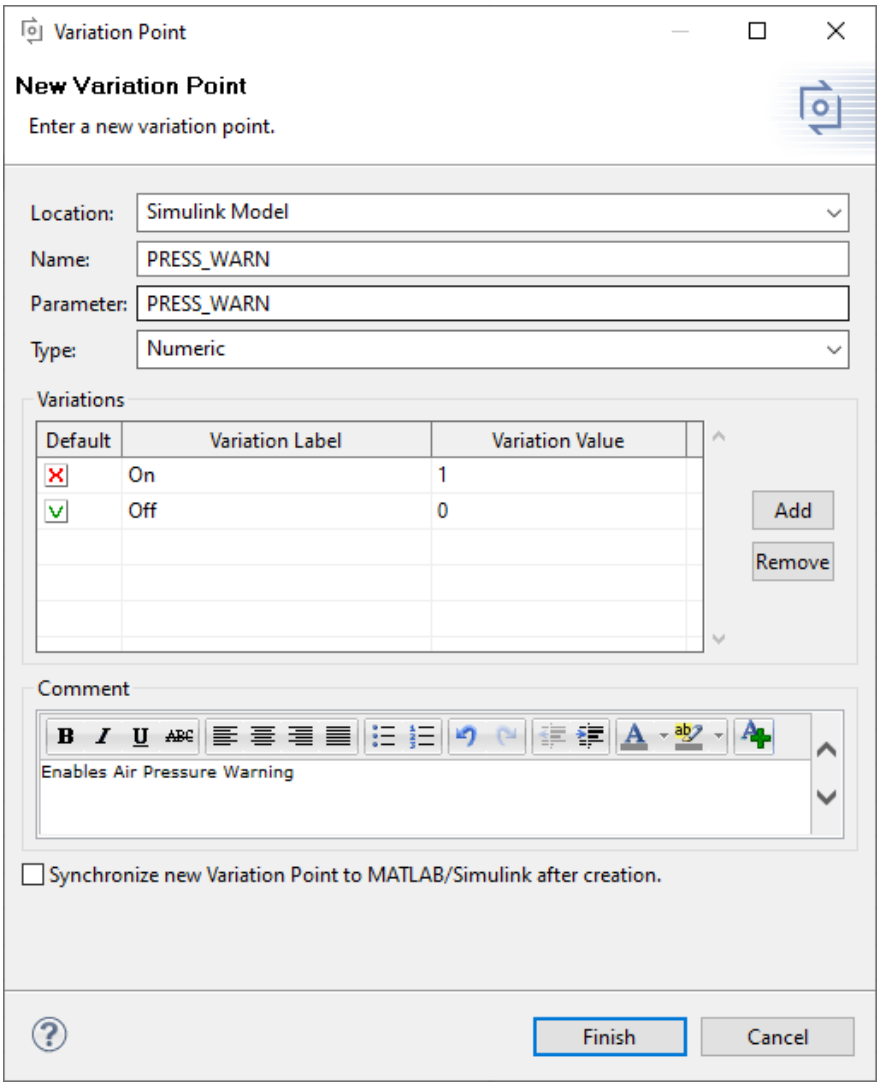
The “Variation Point” viewer of the Variability Model Editor is used for creating a new variation point in pure::variants. The context menu item “Variation Point” opens a wizard for creating a new variation point.

Figure 35. Creating a new Variation Point



A name that does not yet exist in the variability model must be entered in the text field as the name for the variation point. If the name does not satisfy the conventions for a variation point, the wizard reports this problem. For more information on conventions, see Section 1.2, “Conventions”. The name of the workspace parameter (Parameter) to be used must also be entered; this is automatically created if it does not already exist.

Figure 36. Variation Point Dialog



In addition, one or more variations can also be added for the new variation point. “Add” adds a new variation, and “Remove” removes it again. The first column of the table (“Default”) defines whether a variation is used as “Default Assignment” () or not (). Only one variation can be selected as “Default Assignment”. The two other columns correspond to the name (Label) and the value of the variation (Value) and can be edited directly in the cell (see also Section 6.1.3, “Activity: Creating a New Variation”).

The Location defines where the new variation point is created. It can be created either in a "Simulink Model" or in the "Data Dictionary".

If the variation point is create in a "Simulink Model" then the variation point type can be changed. Possible types are "String" and "Numeric". The type can be defined only during variation point creation and is not changeable afterwards. Type "String" allows all string as Variation values, "Numeric" instead allows numbers only.

A comment can also be added for the variation point in the bottom text field. Like the other information, this comment can also be used in MATLAB/Simulink.

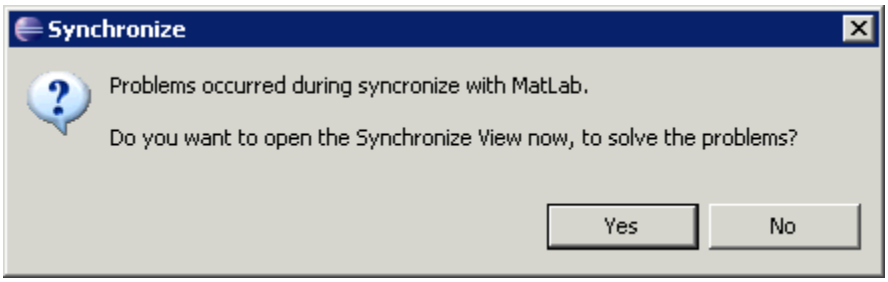
After the wizard is finished, the new variation point is created in the variability model.

The new variation point can then be automatically written to MATLAB/Simulink by selecting the option “Synchronize new Variation Point to MATLAB/Simulink after creation.” This requires that the Data Dictionary associated with the Simulink model be open in MATLAB/Simulink.

Possible Error Causes / Known Restrictions

During automatic writing to MATLAB/Simulink, a message may appear to report problems.

Figure 37. Error while synchronizing Variation Point



These could be conflicts in the synchronization of the variation point with existing variation points or within its variations. To solve these problems, it is possible to manually synchronize to MATLAB/Simulink with the Compare Editor (see Section 6.4.1, “Activity: Synchronizing to MATLAB/Simulink”).

Other Actions

The created variation point can now be used directly for modeling dependencies and for variant configuration in pure::variants. It should be noted here that before writing a variation point configuration to MATLAB/Simulink, the new variation point must first be synchronized to MATLAB/Simulink (see Section 6.4.1, “Activity: Synchronizing to MATLAB/Simulink”).

6.1.3. Activity: Creating a New Variation

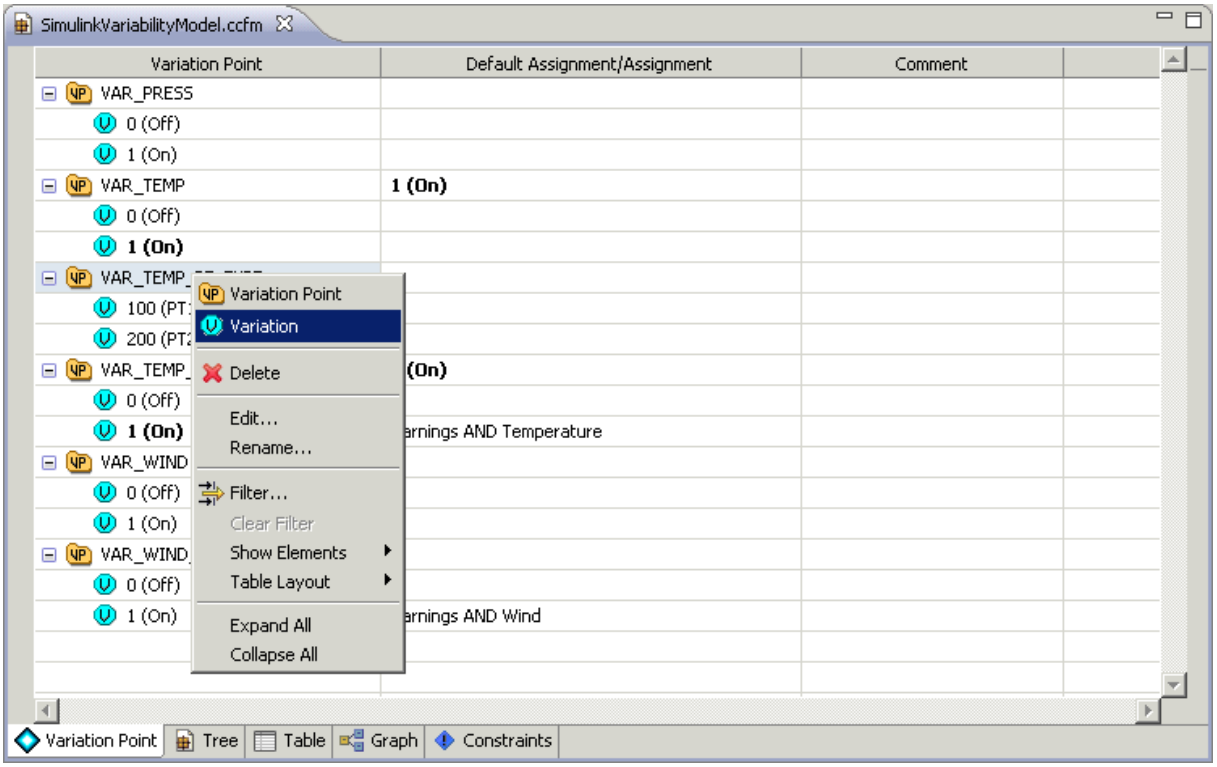
Prerequisites & Preliminary Work

- The variability model must be open.
- The variation point of the new variation must exist in the variability model.
- A variation with the new name or value must not yet exist in this variation point.

Process

The “Variation Point” viewer of the Variability Model Editor is used for creating a new variation in pure::variants. In the viewer, select the variation point for which a new variation should be created. The context menu item “Variation” of the selected variation point opens a wizard for creating a new variation.

Figure 38. Creating a new Variation

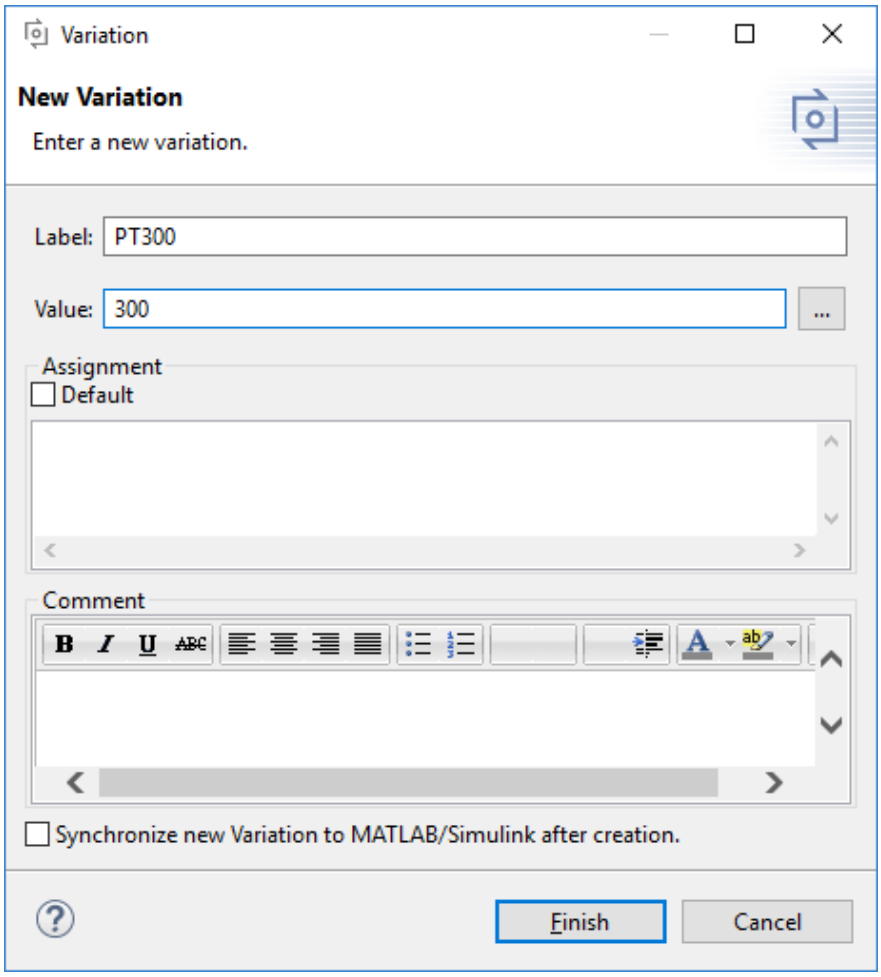


The name (Label) and the value (Value) of the new variation must be entered into the text fields. The name and value must be unique for the selected variation point. If the name or value does not satisfy the conventions for a variation, the wizard reports this problem. For more information on conventions, see Section 1.2, “Conventions”.

All other information is optional. The new variation can be defined as the ”Default Assignment” of the variation point. Alternatively, the variation can also be linked with features via a condition (see Section 6.7.1, “Activity: Linking a Variation with Features”). This condition is entered in the text field “Assignment”.

A comment can also be added for the variation in the bottom text field. This comment is used only in pure::variants and is not available in MATLAB/Simulink.

Figure 39. Variation Dialog



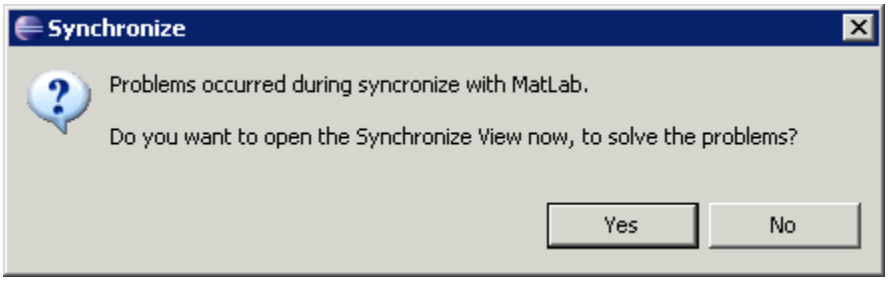
After the wizard is finished, the new variation is created at the selected variation point in the variability model.

The new variation can then be automatically written to MATLAB/Simulink by selecting the option “Synchronize new Variation to MATLAB/Simulink after creation.” However, in order for the new variation to be written to MATLAB/Simulink, the Data Dictionary associated with the Simulink model must be open in MATLAB/Simulink.

Possible Error Causes / Known Restrictions

During automatic writing to MATLAB/Simulink, a message may appear to report problems.

Figure 40. Error while synchronizing Variation



These could be conflicts in the synchronization of the changed variation point. To solve these problems, it is possible to manually synchronize to MATLAB/Simulink with the Compare Editor (see Section 6.4.1, “Activity: Synchronizing to MATLAB/Simulink”).

If a variation is selected as “Default Assignment”, it is not possible to formulate a condition with regard to a feature selection. However, such a condition can be formulated later and added (see Section 6.7.1, “Activity: Linking a Variation with Features”).

Other Actions

The created variation can now be used directly for modeling dependencies and for variant configuration in pure::variants. It should be noted here that before writing a variation point configuration to MATLAB/Simulink, the new variation must first be synchronized to MATLAB/Simulink (see Section 6.4.1, “Activity: Synchronizing to MATLAB/Simulink”).

6.1.4. Activity: Creating a New Calculated Variation

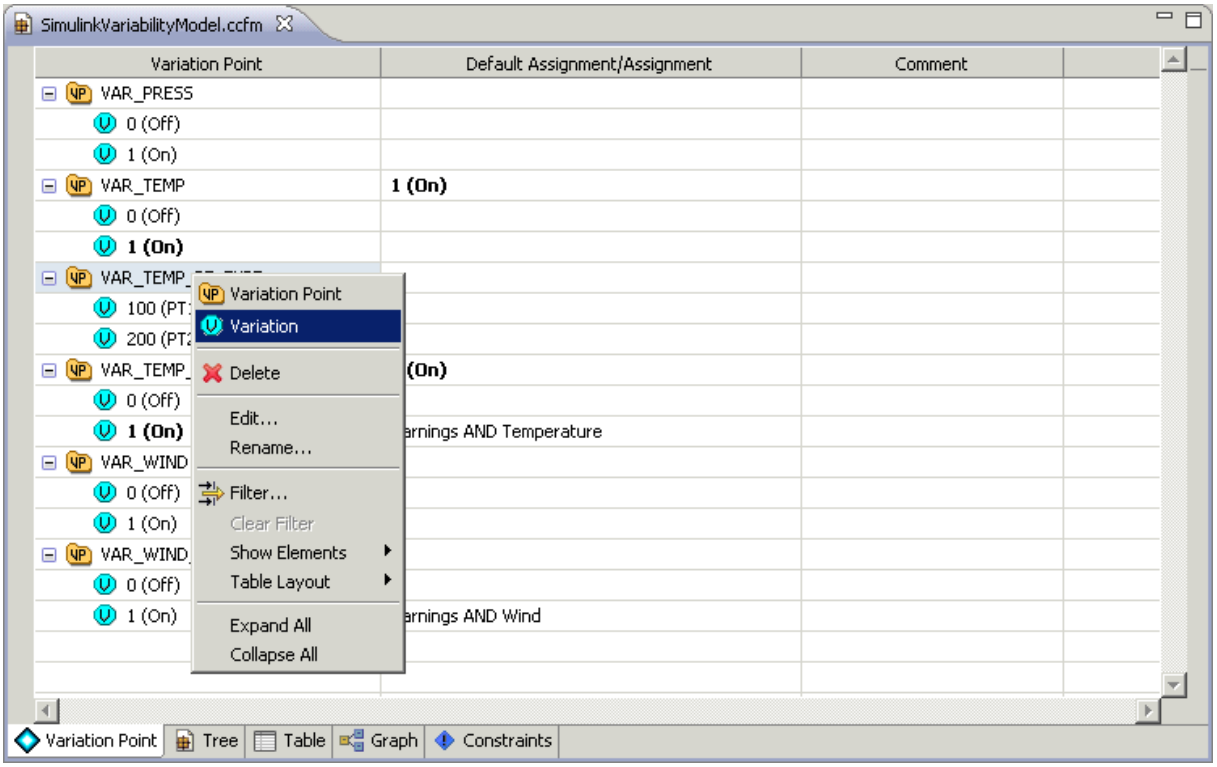
Prerequisites & Preliminary Work

- The variability model must be open.
- The variation point of the new variation must exist in the variability model.
- A variation with the new name or value must not yet exist in this variation point.

Process

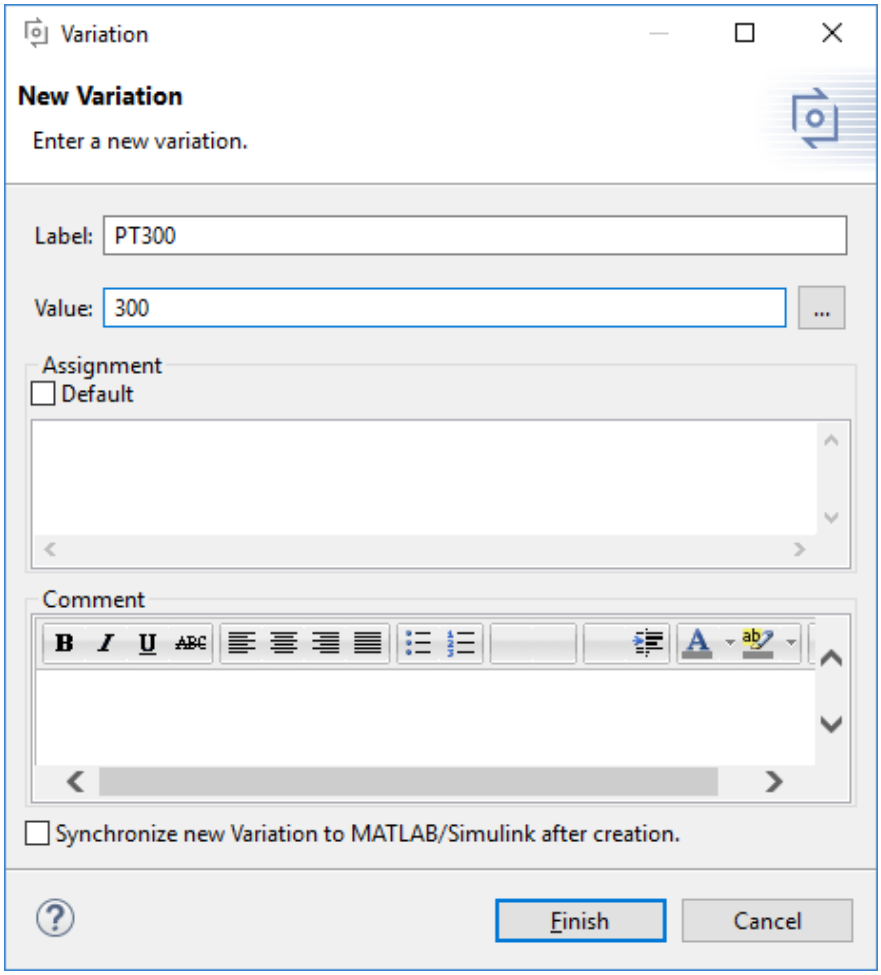
The “Variation Point” viewer of the Variability Model Editor is used for creating a new variation in pure::variants. In the viewer, select the variation point for which a new variation should be created. The context menu item “Variation” of the selected variation point opens a wizard for creating a new variation.

Figure 41. Creating a new Variation



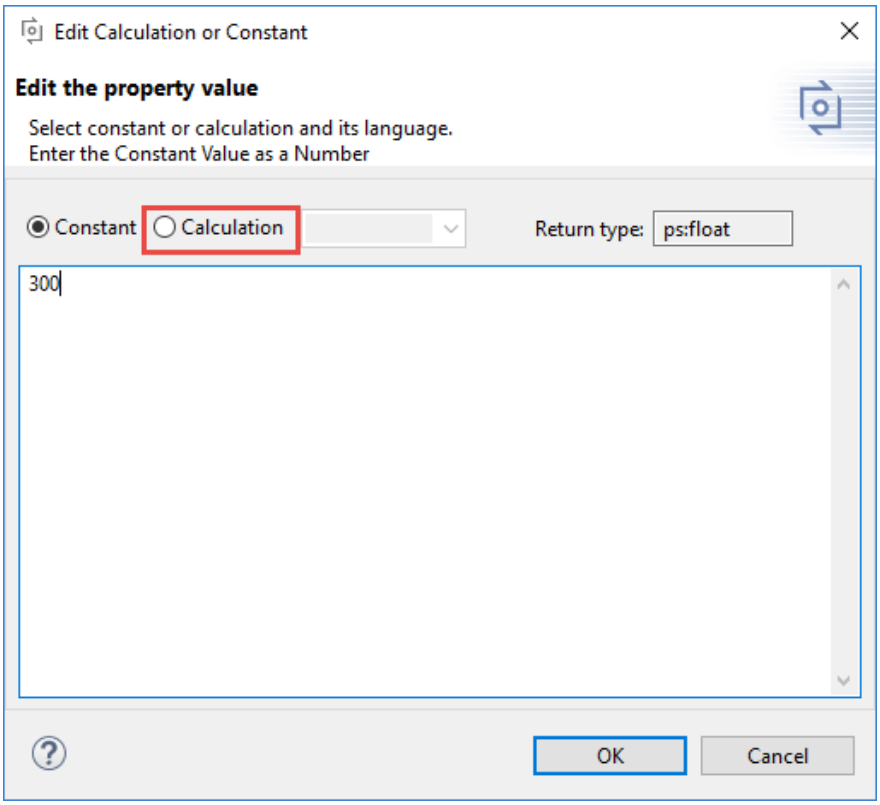
The name (Label) of the new variation must be entered into the text field. The name and value must be unique for the selected variation point. If the name does not satisfy the conventions for a variation, the wizard reports this problem. For more information on conventions, see Section 1.2, “Conventions”.

Figure 42. Variation Dialog



To add the calculation the button behind the value text filed is used. In the Code Editor use the *Calculation* button to switch to creating a calculation. The code editor aids the user during calculation creation.

Figure 43. Code Editor Dialog



All other information is optional. The new variation can be defined as the ”Default Assignment” of the variation point. Alternatively, the variation can also be linked with features via a condition (see Section 6.7.1, “Activity: Linking a Variation with Features”). This condition is entered in the text field “Assignment”.

A comment can also be added for the variation in the bottom text field. This comment is used only in pure::variants and is not available in MATLAB/Simulink.

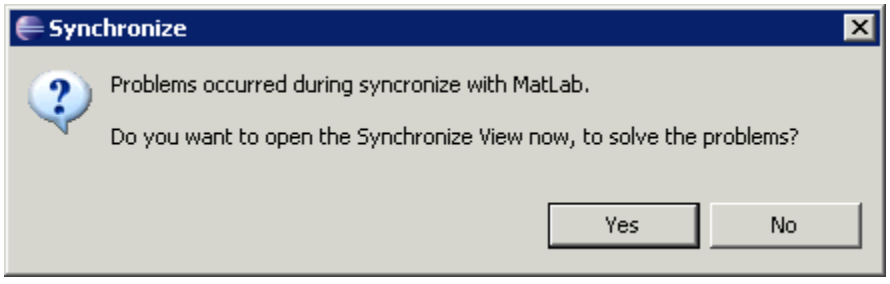
After the wizard is finished, the new variation is created at the selected variation point in the variability model.

The new variation can then be automatically written to MATLAB/Simulink by selecting the option “Synchronize new Variation to MATLAB/Simulink after creation.” However, in order for the new variation to be written to MATLAB/Simulink, the Data Dictionary associated with the Simulink model must be open in MATLAB/Simulink.

Possible Error Causes / Known Restrictions

During automatic writing to MATLAB/Simulink, a message may appear to report problems.

Figure 44. Error while synchronizing Variation



These could be conflicts in the synchronization of the changed variation point. To solve these problems, it is possible to manually synchronize to MATLAB/Simulink with the Compare Editor (see Section 6.4.1, “Activity: Synchronizing to MATLAB/Simulink”).

If a variation is selected as “Default Assignment”, it is not possible to formulate a condition with regard to a feature selection. However, such a condition can be formulated later and added (see Section 6.7.1, “Activity: Linking a Variation with Features”).

Other Actions

The created variation can now be used directly for modeling dependencies and for variant configuration in pure::variants. It should be noted here that before writing a variation point configuration to MATLAB/Simulink, the new variation must first be synchronized to MATLAB/Simulink (see Section 6.4.1, “Activity: Synchronizing to MATLAB/Simulink”).

6.2. Changing Variability Information

6.2.1. Activity: Changing a Variation Point

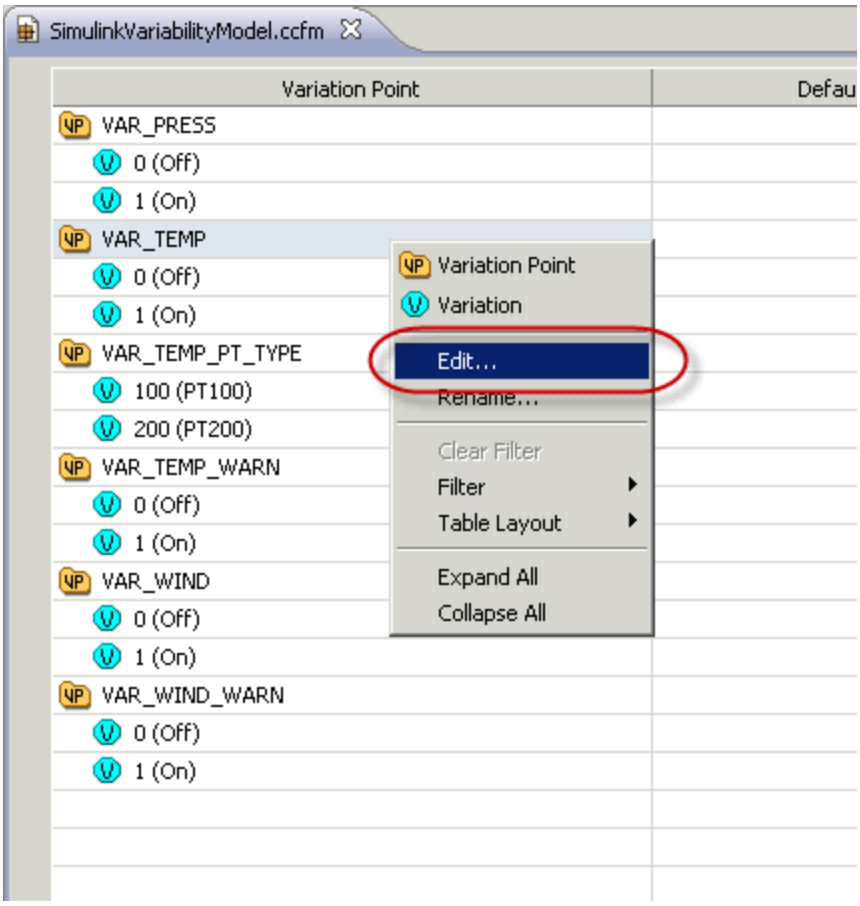
Prerequisites & Preliminary Work

- The variability model must be open.
- The variation point to be changed must exist in this variability model.

Process

The “Variation Point” viewer of the Variability Model Editor is used for changing a variation point in pure::variants. Select this variation point in the viewer. The context menu item “Edit” opens a dialog for editing the variation point. This dialog is initialized with the current information of the variation point.

Figure 45. Changing a Variation Point



Note: The dialog can also be opened with the context menu item “Rename…”, which allows quick editing of the name since the corresponding text field is automatically selected.

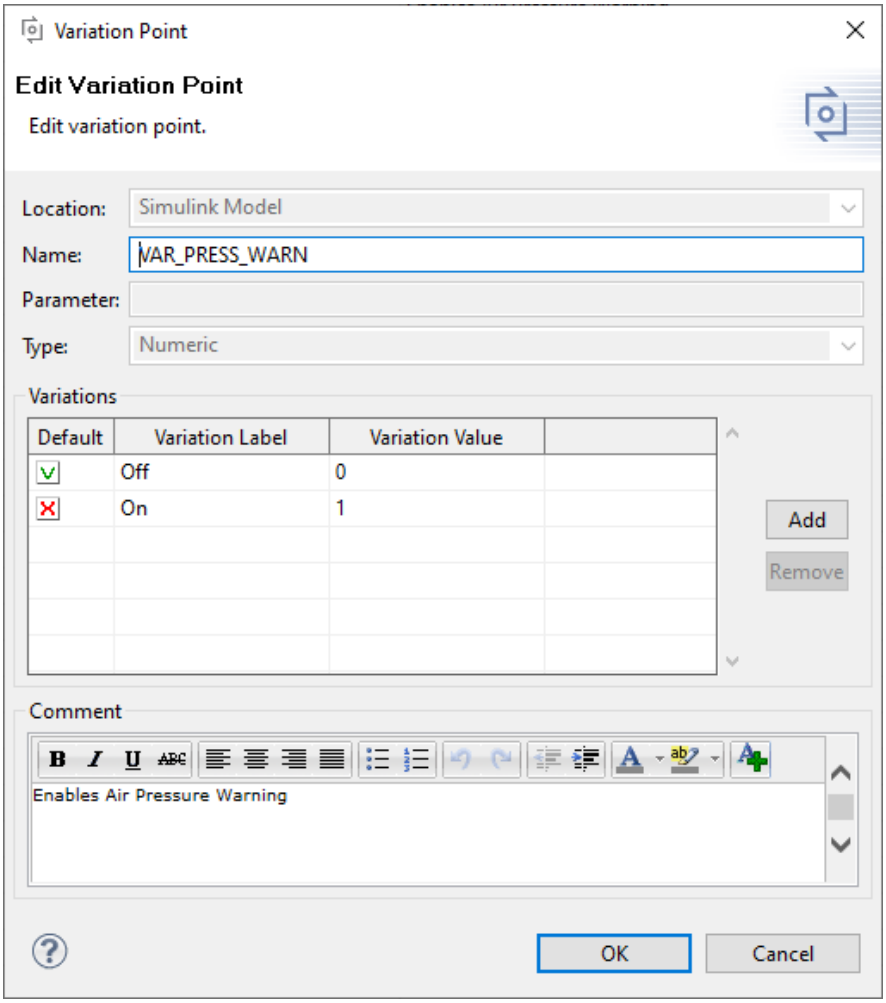
In contrast to MATLAB/Simulink, the parameter associated with the variation point can no longer be changed. This was specified upon creation of the variation point and remains associated with this until it is deleted.

A changed name that does not yet exist in the variability model must be entered in the text field as the name for the variation point. If the name does not satisfy the conventions for a variation point, the dialog reports this problem. For more information on conventions, see Section 1.2, “Conventions”.

In addition, one or more variations can also be added for the variation point. “Add” adds a new variation, and “Remove” removes an existing one. The first column of the table (“Default”) defines whether a variation is used as “Default Assignment” ( ) or not ( ). Only one variation can be selected as “Default Assignment”. The two other columns correspond to the name (Label) and the value of the variation (Value) and can be edited directly in the cell (see also Section 6.2.2, “Activity: Changing a Variation”).

A comment can also be added for the variation point in the bottom text field. Like the other information, this comment can also be used in MATLAB/Simulink.

Figure 46. Variation Point Dialog



After the dialog is closed, the changes are applied and the variability model is saved.

Other Actions

To be able to use the changed variation point for the variation point configuration in MATLAB/Simulink, the variability model must be synchronized to MATLAB/Simulink (see Section 6.4.1, “Activity: Synchronizing to MATLAB/Simulink”).

6.2.2. Activity: Changing a Variation

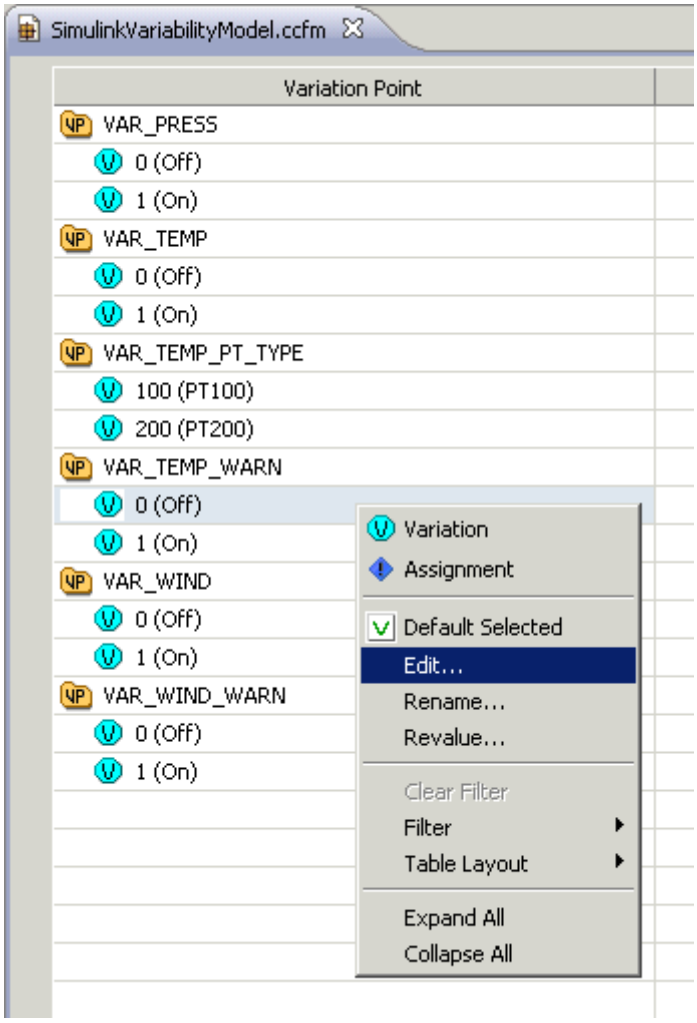
Prerequisites & Preliminary Work

- The variability model must be open.
- The variation to be changed must exist at a variation point in this variability model.

Process

The “Variation Point” viewer of the Variability Model Editor is used for changing a variation in pure::variants. Select this variation in the viewer. The context menu item “Edit” of the selected variation opens a dialog for editing the variation. This dialog is initialized with the current information of the variation.

Figure 47. Changing a Variation



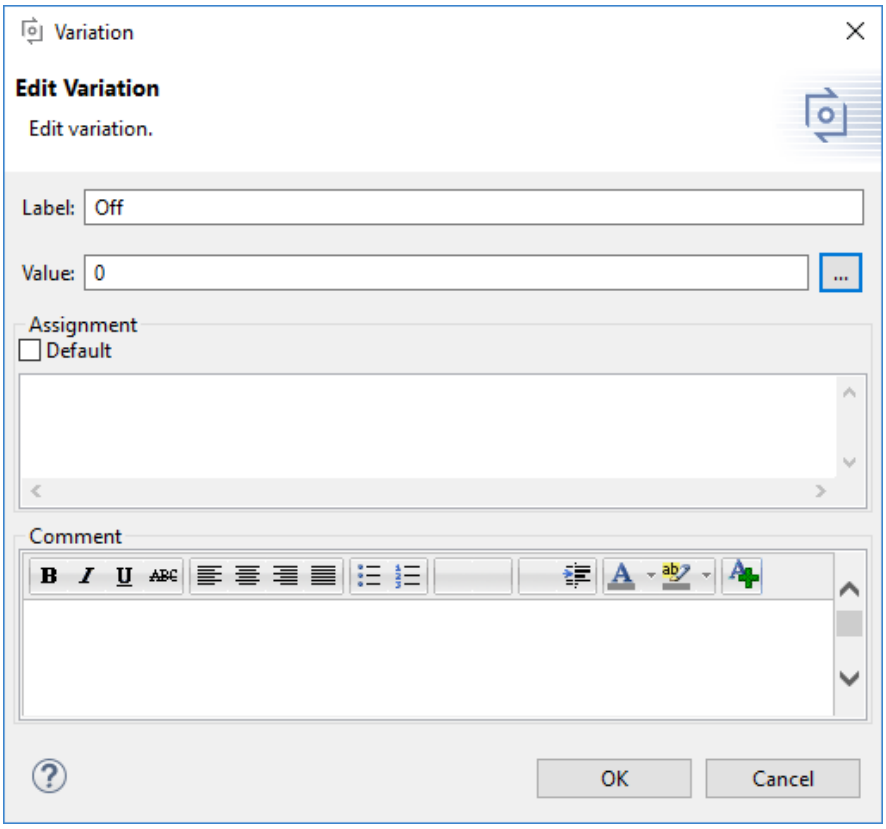
Note: The dialog can also be opened with the context menu item “Rename…” or “Revalue…”, which allows quick editing of the name or value since the corresponding text field is automatically selected.

The changed name or value must be entered into the text fields. The changed name and value must be unique for the variation point. If the name or value does not satisfy the conventions for a variation, the dialog reports this problem. For more information on conventions, see Section 1.2, “Conventions”.

All other information is optional. The variation to be changed can be defined as the ”Default Assignment” of the variation point. Alternatively, the variation can also be linked with features via a condition (see Section 6.7.1, “Activity: Linking a Variation with Features”). The condition is entered in the text field “Assignment”.

A comment can also be added for the variation in the bottom text field. This comment is used only in pure::variants and is not available in MATLAB/Simulink.

Figure 48. Variation Dialog



After the dialog is closed, the changes are applied and the variability model is saved.

Other Actions

If a variation is selected as “Default Assignment”, it is not initially possible to formulate a condition with regard to a feature selection. However, such a condition can be formulated later and changed (see Section 6.7.2, “Activity: Changing a Condition”).

To be able to use the changed variation point for the variation point configuration in MATLAB/Simulink, the variability model must be synchronized to MATLAB/Simulink (see Section 6.4.1, “Activity: Synchronizing to MATLAB/Simulink”).

6.2.3. Activity: Changing a Calculated Variation

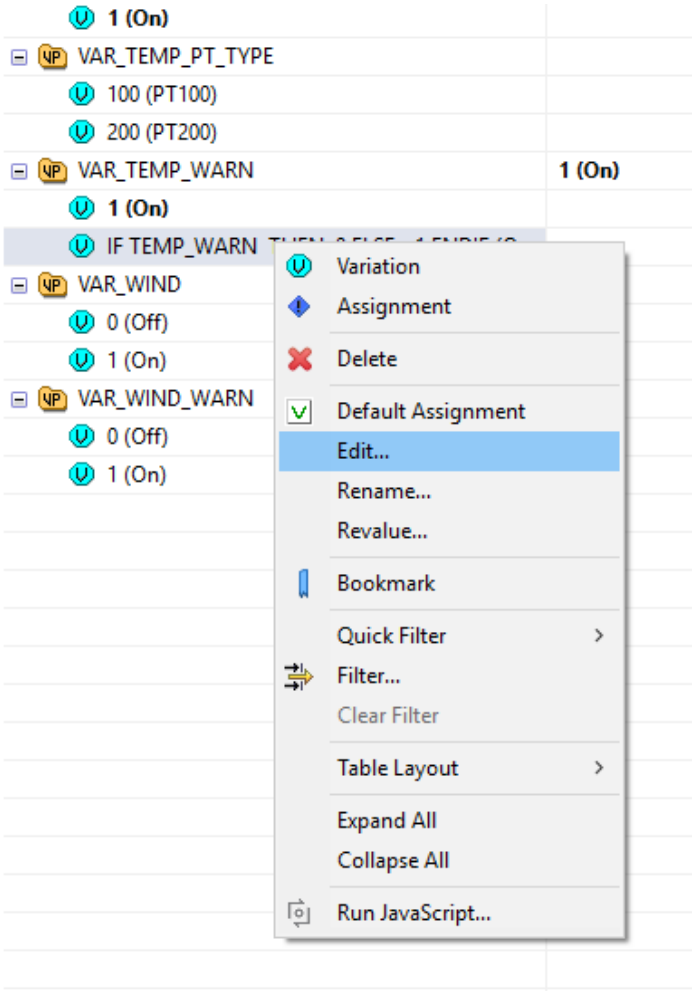
Prerequisites & Preliminary Work

- The variability model must be open.
- The variation to be changed must exist at a variation point in this variability model.

Process

The “Variation Point” viewer of the Variability Model Editor is used for changing a variation in pure::variants. Select this calculated variation in the viewer. The context menu item “Edit” of the selected variation opens a dialog for editing the variation. This dialog is initialized with the current information of the variation.

Figure 49. Changing a Calculated Variation



Note: The dialog can also be opened with the context menu item “Rename…” or “Revalue…”, which allows quick editing of the name or value since the corresponding text field is automatically selected.

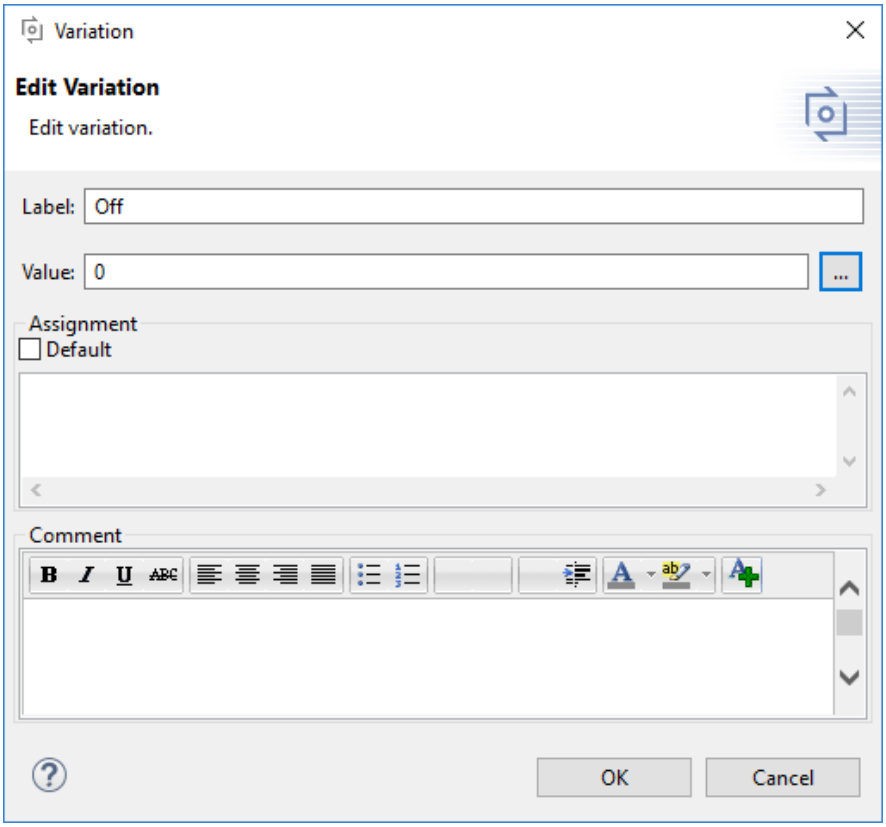
The changed name or value must be entered into the text fields. The changed name must be unique for the variation point. If the name does not satisfy the conventions for a variation, the dialog reports this problem. For more information on conventions, see Section 1.2, “Conventions”.

To edit the calculation the button behind the value text filed is used. The Code Editor opens and aids the user during calculation creation.

All other information is optional. The variation to be changed can be defined as the ”Default Assignment” of the variation point. Alternatively, the variation can also be linked with features via a condition (see Section 6.7.1, “Activity: Linking a Variation with Features”). The condition is entered in the text field “Assignment”.

A comment can also be added for the variation in the bottom text field. This comment is used only in pure::variants and is not available in MATLAB/Simulink.

Figure 50. Variation Dialog



After the dialog is closed, the changes are applied and the variability model is saved.

Other Actions

If a variation is selected as “Default Assignment”, it is not initially possible to formulate a condition with regard to a feature selection. However, such a condition can be formulated later and changed (see Section 6.7.2, “Activity: Changing a Condition”).

To be able to use the changed variation point for the variation point configuration in MATLAB/Simulink, the variability model must be synchronized to MATLAB/Simulink (see Section 6.4.1, “Activity: Synchronizing to MATLAB/Simulink”).

6.3. Deleting Variability Information

6.3.1. Activity: Deleting a Variation Point

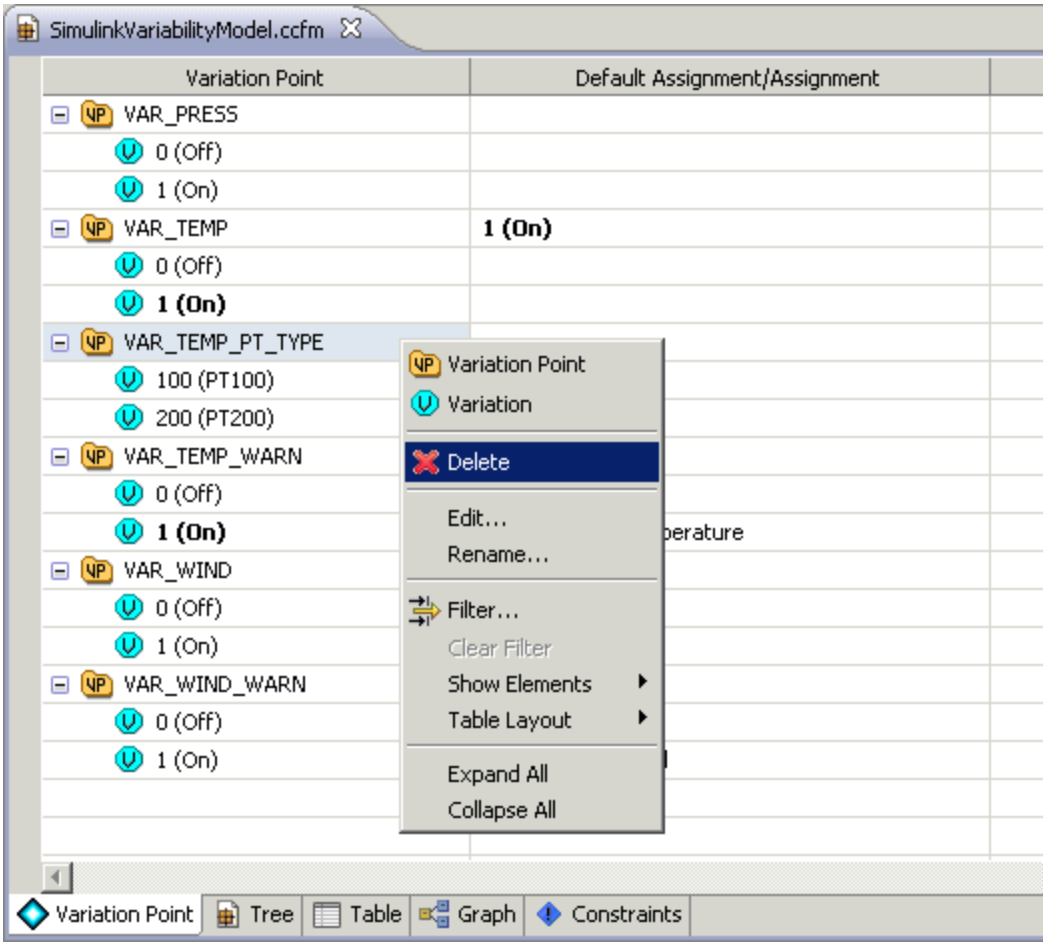
Prerequisites & Preliminary Work

- The variability model must be open.
- The variation point to be deleted must exist in this variability model.

Process

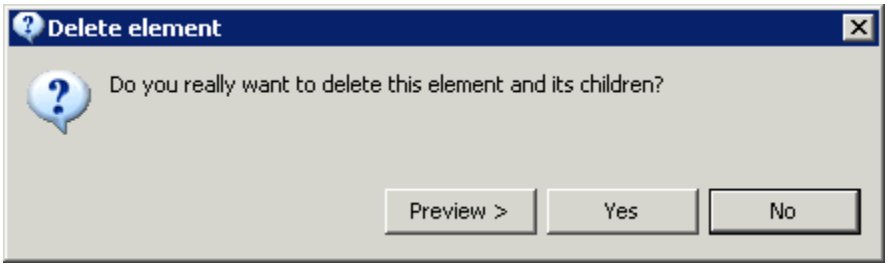
The “Variation Point” viewer of the Variability Model Editor is used for deleting a variation point in pure::variants. Select the variation point in the viewer. The context menu item “Delete” opens a dialog for deleting this variation point as well as all its references.

Figure 51. Deleting a Variation Point



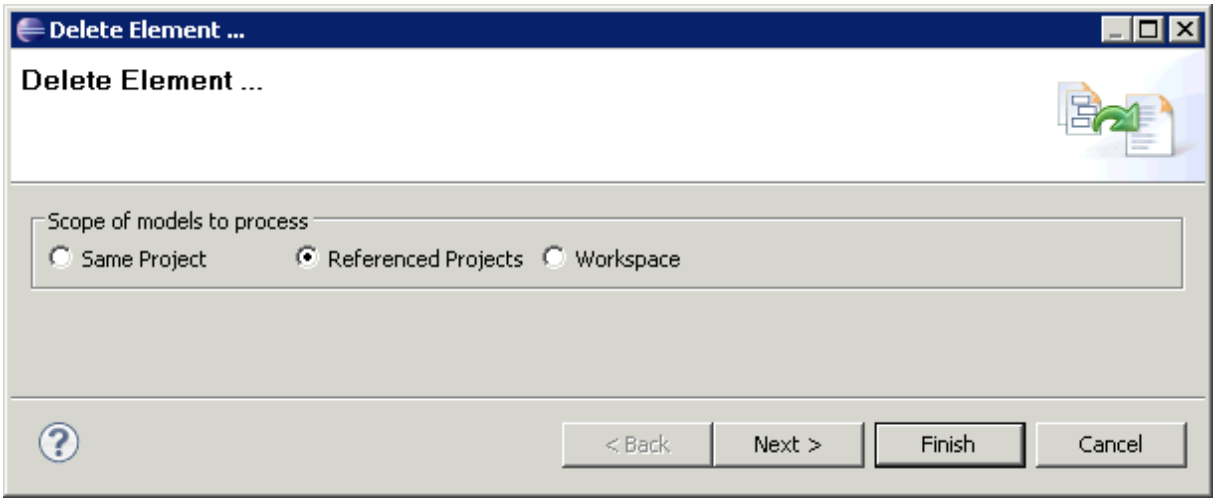
The dialog that opens allows immediate deletion of the variation point with “Yes”. It also offers a preview of the changes with “Preview >”.

Figure 52. Confirmation of the Deletion of a Variation Point



If a preview of the changes is selected, a Refactoring Wizard appears. On the first page of the wizard, it is possible to change the preconfigured search scope, which permits a search for references to this variation point in the current project and its referenced projects. This can be restricted to the current project or expanded to the entire workspace of pure::variants.

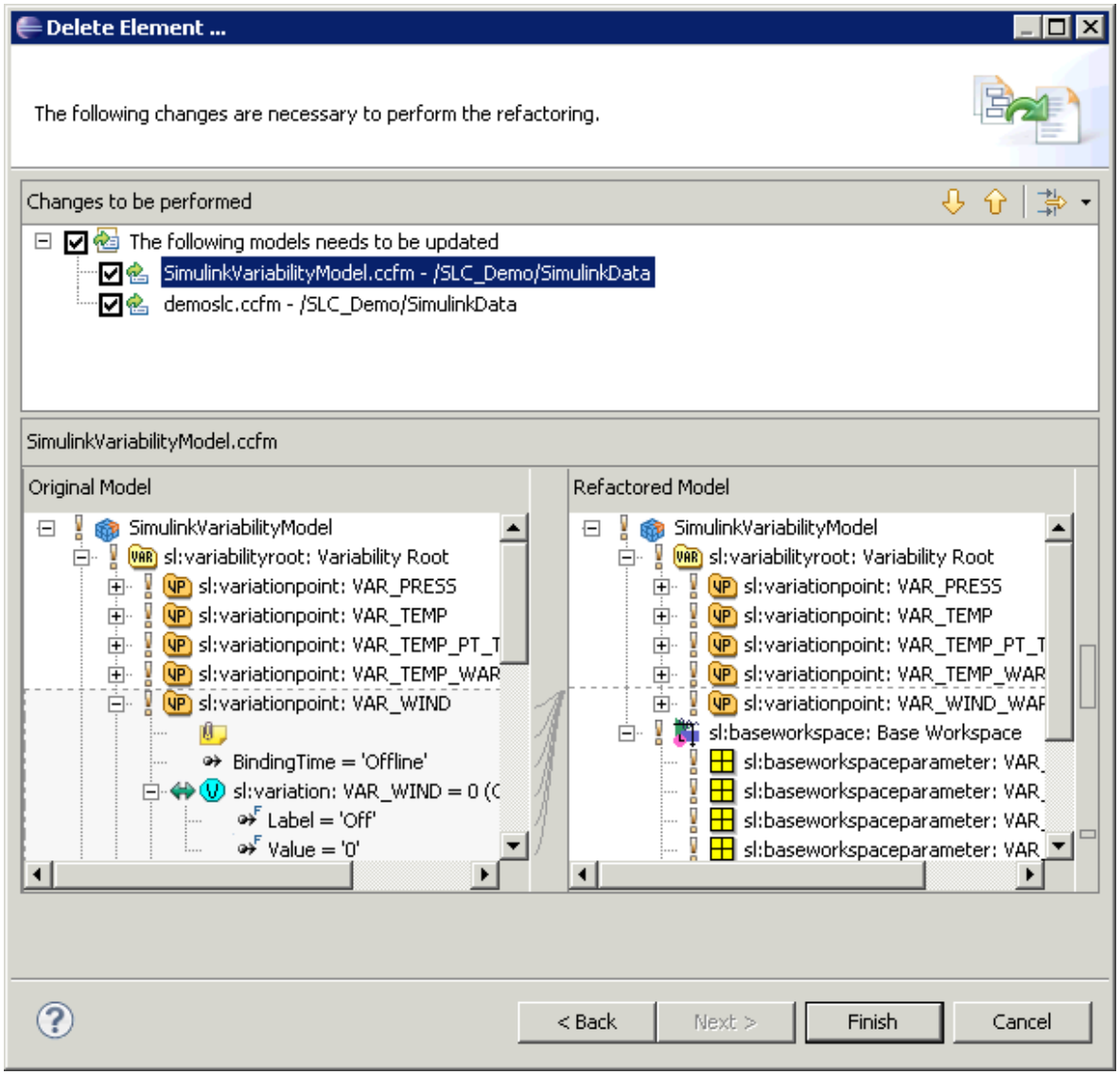
Figure 53. Search Scope for References of a Variation Point to be deleted



The second page shows all models affected by the change and offers a preview of all necessary changes in these models. The models to be adjusted are shown in the top part, and the trees below compare the corresponding changes in detail. The left tree contains the original state and the right tree the changed state.

Changes can be deselected on a per-model basis in the top part by leaving models unselected.

Figure 54. Preview of resulting Changes



When the wizard is finished, the displayed changes are executed on a per-model basis and the variation point is removed from the variability model.

Note: Before deleting, all open models must be saved, which is also automatically supported by a corresponding confirmation dialog. After deleting the variation point, all changed models are automatically saved.

Other Actions

The deleted variation point is no longer available in pure::variants. To remove the deleted variation point from MATLAB/Simulink as well so that it is no longer available for variation point configuration, a synchronization to MATLAB/Simulink must be performed (see Section 6.4.1, “Activity: Synchronizing to MATLAB/Simulink”).

6.3.2. Activity: Deleting a Variation

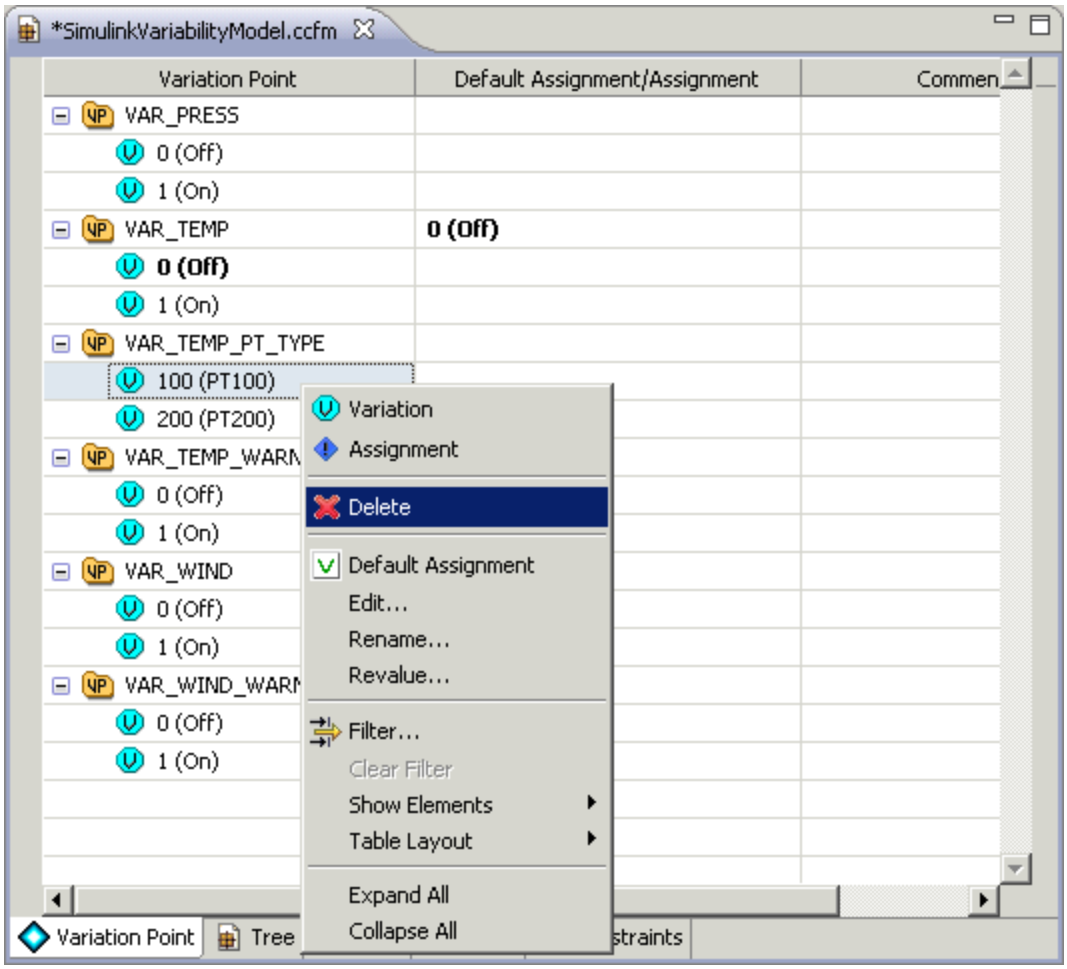
Prerequisites & Preliminary Work

- The variability model must be open.
- The variation to be deleted must exist in this variability model.

Process

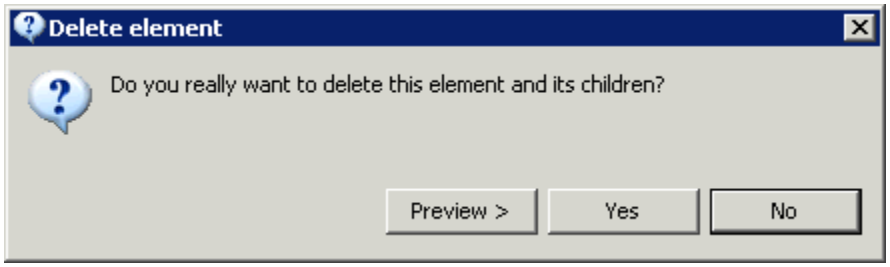
The “Variation Point” viewer of the Variability Model Editor is used for deleting a variation in pure::variants. Select the variation in the viewer. The context menu item “Delete” opens a dialog for deleting this variation as well as all its references.

Figure 55. Deletion of a Variation



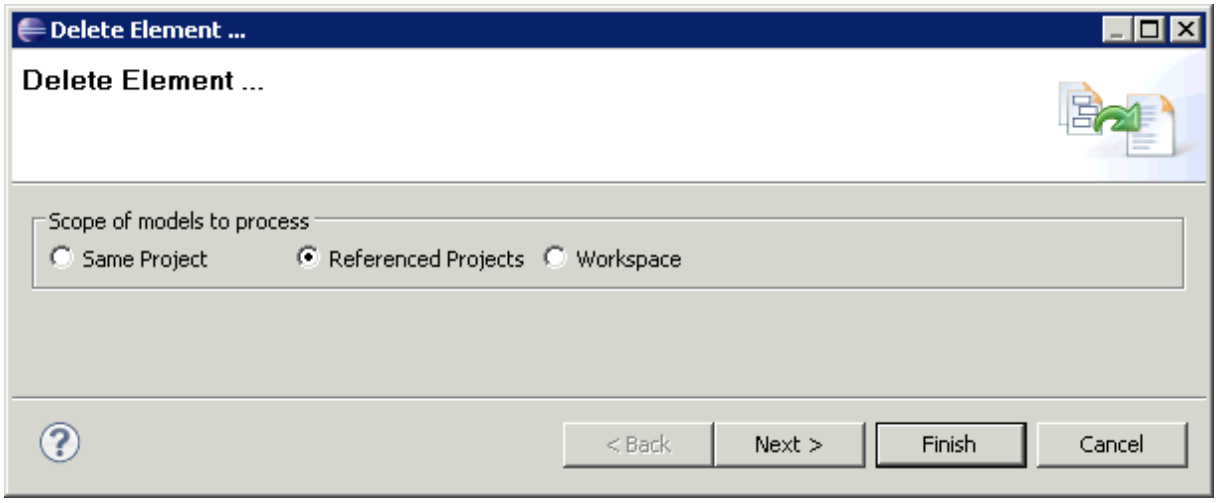
The dialog that opens allows immediate deletion of the variation with “Yes”. It also offers a preview of the changes with “Preview >”.

Figure 56. Confirmation of the Deletion of a Variation



If a preview of the changes is selected, a Refactoring Wizard appears. On the first page of the wizard, it is possible to change the preconfigured search scope, which permits a search for references to this variation point in the current project and its referenced projects. This can be restricted to the current project or expanded to the entire workspace of pure::variants.

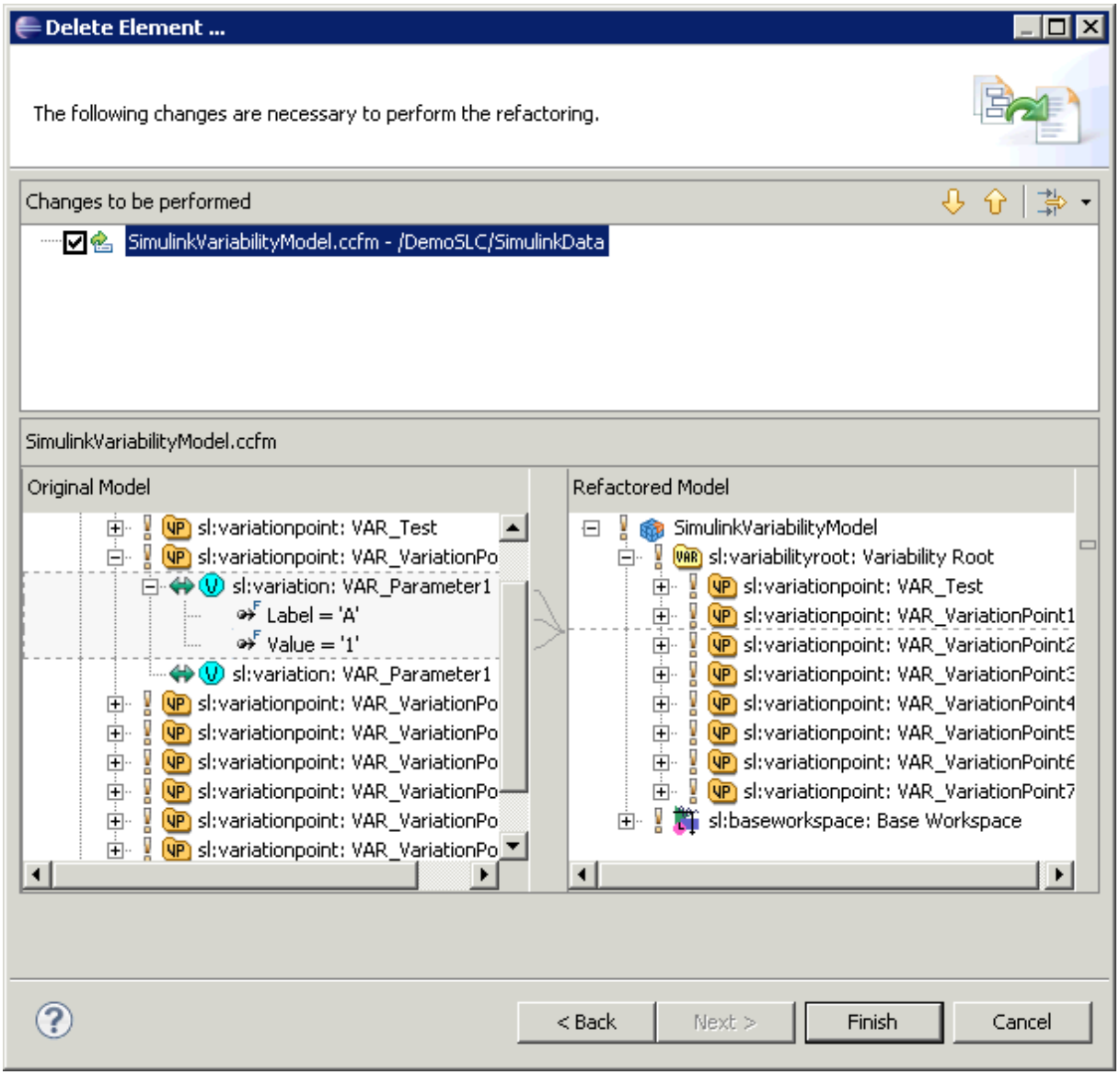
Figure 57. Search Scope for References of a Variation to be deleted



The second page shows all models affected by the change and offers a preview of all necessary changes in these models. The models to be adjusted are shown in the top part, and the trees below compare the corresponding changes in detail. The left tree contains the original state and the right tree the changed state.

Changes can be deselected on a per-model basis in the top part by leaving models unselected.

Figure 58. Preview of resulting Changes



When the wizard is finished, the displayed changes are executed on a per-model basis and the variation is removed from the variability model.

Note: Before deleting, all open models must be saved, which is also automatically supported by a corresponding confirmation dialog. After deleting the variation, all changed models are automatically saved.

Other Actions

The deleted variation is no longer available in pure::variants. To remove the deleted variation from MATLAB/Simulink as well so that it is no longer available for variation point configuration, a synchronization to MATLAB/Simulink must be performed (see Section 6.4.1, “Activity: Synchronizing to MATLAB/Simulink”).

6.4. Synchronizing Variability Information to MATLAB/Simulink

6.4.1. Activity: Synchronizing to MATLAB/Simulink

Prerequisites & Preliminary Work

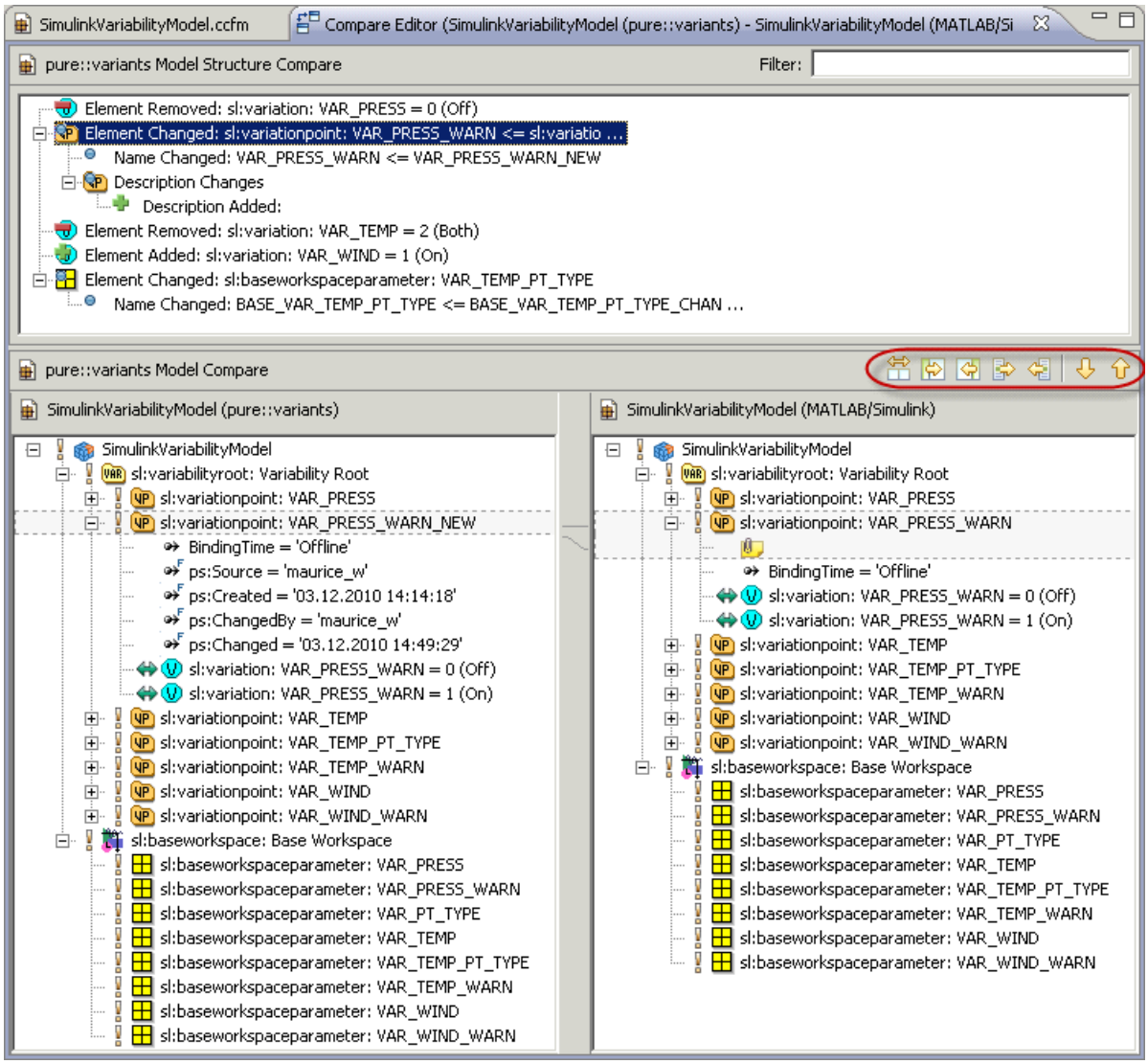
- The Simulink model or the Data Dictionary associated with the Simulink model must be open.

- The variability model associated with the Data Dictionary or Simulink model must be open in pure::variants.

Process

The Compare Editor in pure::variants is used to synchronize variability information from pure::variants to MATLAB/Simulink. This compares the differences between the current variability information of pure::variants and the information to be changed in MATLAB/Simulink, which is automatically imported. It is opened via the toolbar of pure::variants with the synchronization button ( ), which is only visible when a variability model is open.

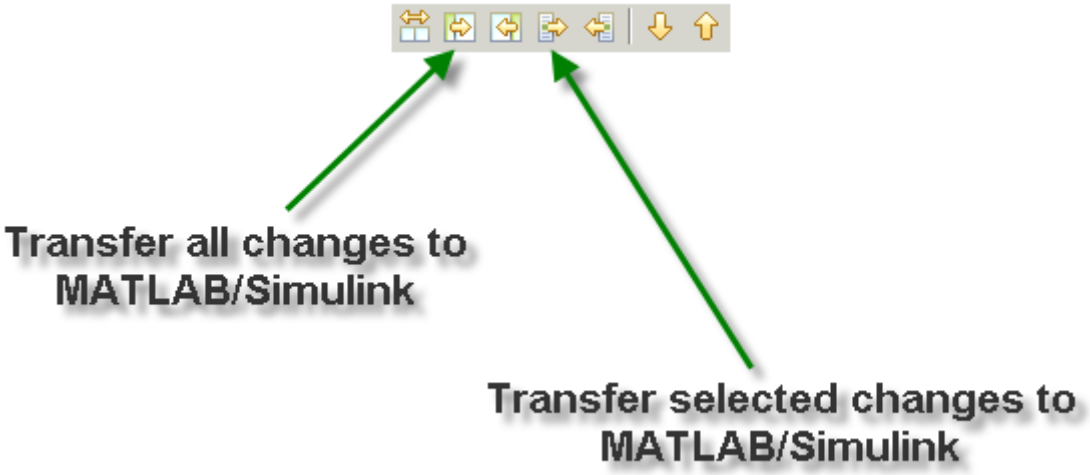
Figure 59. Synchronization to MATLAB/Simulink



The lower left side shows the current variability information of the variability model of pure::variants, and the lower right side shows the variability information of MATLAB/Simulink as a newly imported variability model. The top part shows all changes in the two models. Marking a change causes the associated elements in the lower trees to be highlighted and the differences are displayed there more precisely.

The Compare Editor makes it possible to transfer these changes from pure::variants (left side) to MATLAB/Simulink (right side). For this purpose, there is a toolbar immediately below the listed changes that offers various functions in the different directions (according to the arrows). In each case, a tool tip briefly describes the possible action.

Figure 60. Toolbar of Compare Editor



Changes can be accepted individually or all at once. It should be noted that applied changes are initially only visible on the right in the displayed variability model. The Compare Editor must be saved for a final application of the changes in MATLAB/Simulink.

6.5. Creating a Feature and Variability Model

6.5.1. Activity: Importing Variation Points

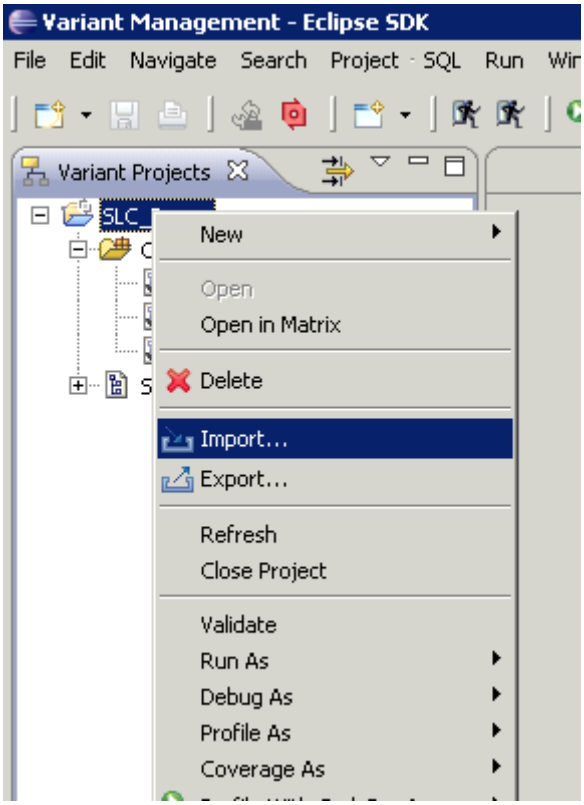
Prerequisites & Preliminary Work

- The Simulink model or the Data Dictionary associated with the Simulink model must be open.

Process

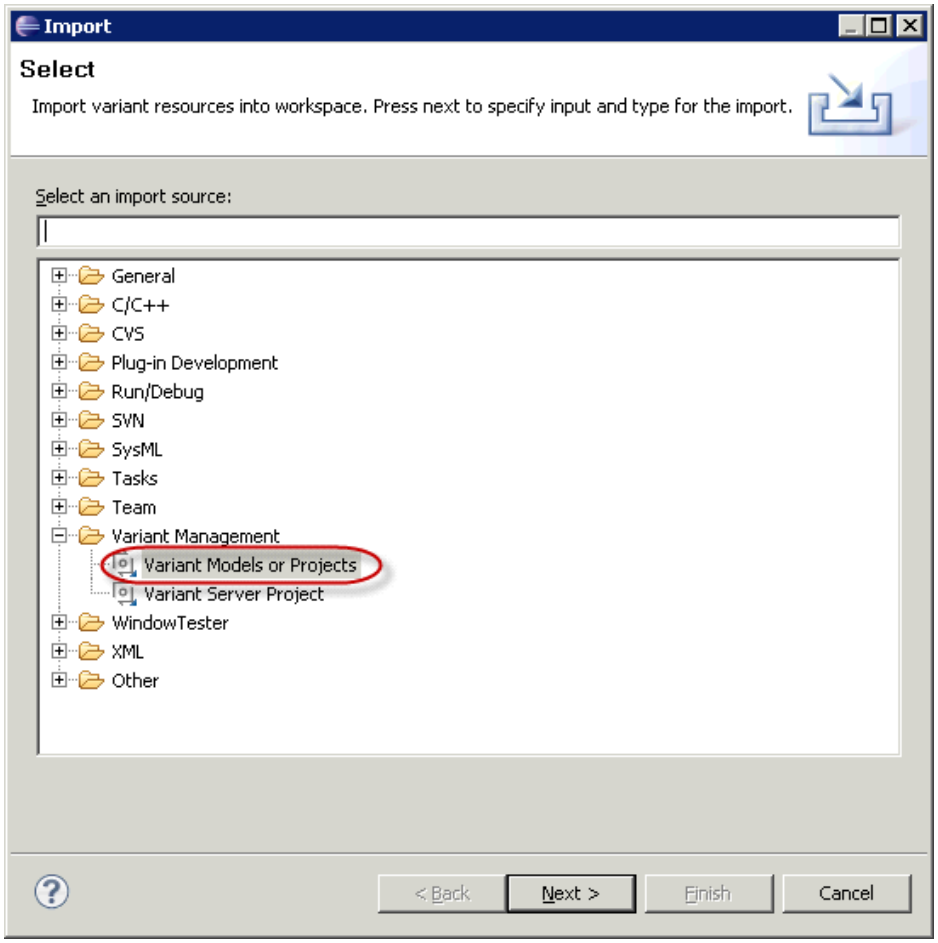
If the variation points were modeled in MATLAB/Simulink, these must be imported into a variability model in pure::variants with the help of the Import Wizard. To do this, select a pure::variants project in the view “Variant Projects”. The wizard can be opened via the context menu item “Import…”.

Figure 61. Opening Import Wizard



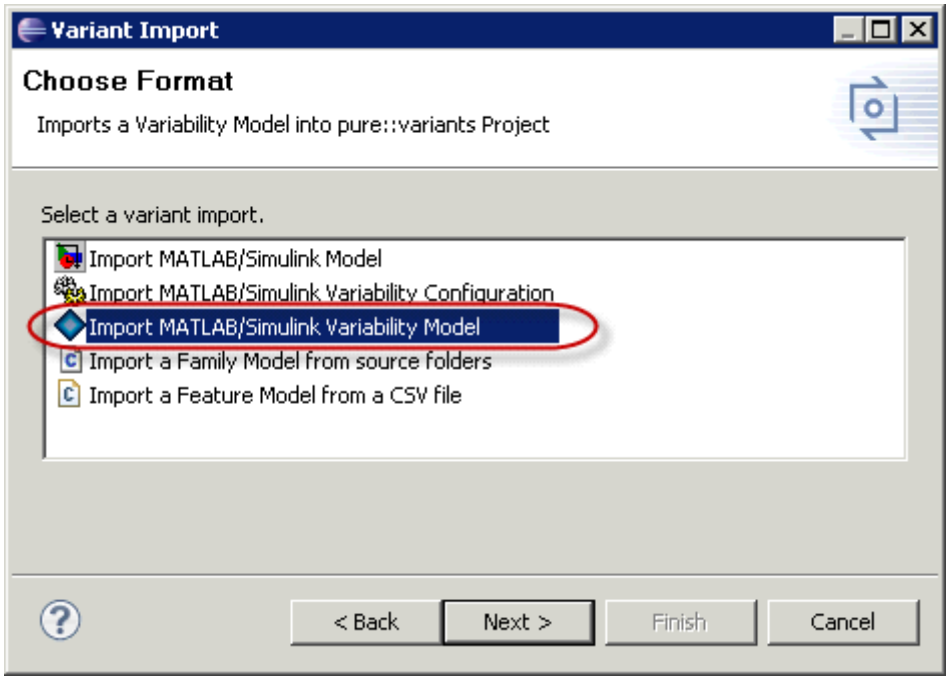
On the first page of the wizard that appears, select the entry “Variant Management Variant Models or Projects”.

Figure 62. Selecting Variant Model Import



The second page shows all currently possible imports to pure::variants. To import the variation points to pure::variants, the entry “Import MATLAB/Simulink Variability Model” must be selected.

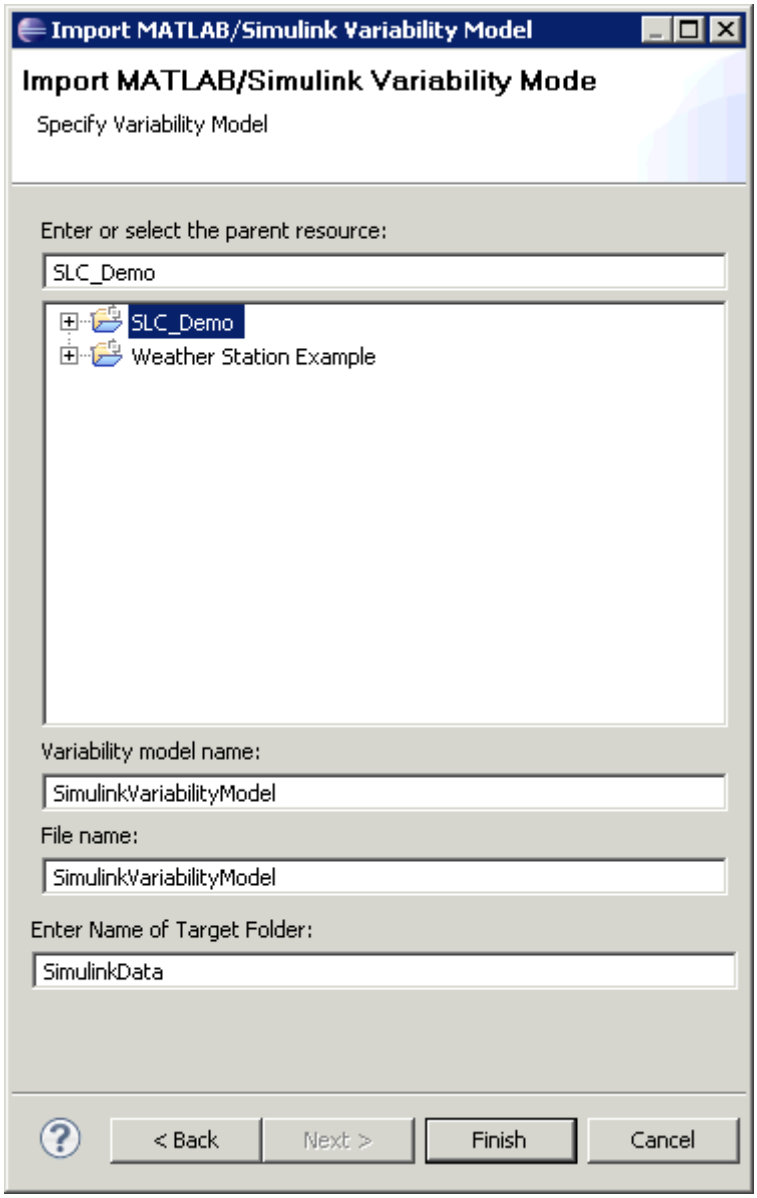
Figure 63. Selecting MATLAB/Simulink Variability Model Importer



On the next page of the wizard, all the settings for the variability model to be created from MATLAB/Simulink are configured. In the top part of the page the parent resource for the import can be chosen.

In the lower part of the page the model name and a deviating file name for the imported variability model can be changed. By default the file name will be the same as the model name. Additionally a target folder can be entered which will be created in the parent resource selected above. The imported variability model will be placed there. Creating such a directory allows the imported model to be kept separate from others. If no folder is specified, the model will be placed directly under the selected parent resource.

Figure 64. Settings for a Variability Model



After finishing the wizard, all variation points are imported and the variability model is created in the specified target directory.

6.5.2. Activity: Modeling Features

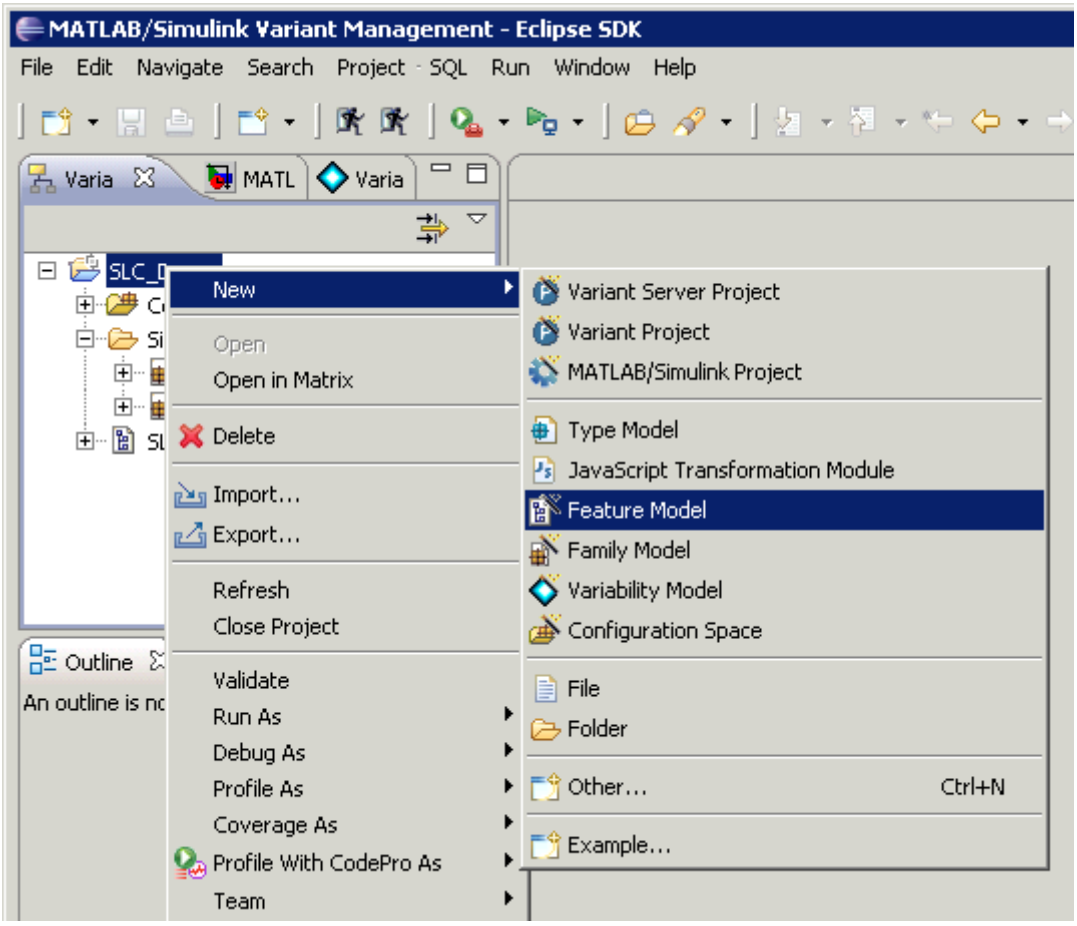
Process

If you wish to configure the variation points using features, a feature model is required in addition to a variability model imported from MATLAB/Simulink or created in pure::variants. An already existing feature model can be

used for this, or a new one can be created. Please refer to sections “5.3 Feature Models” and “7.3 Editors” of the pure::variants user documentation for more information on modeling features and their structures.

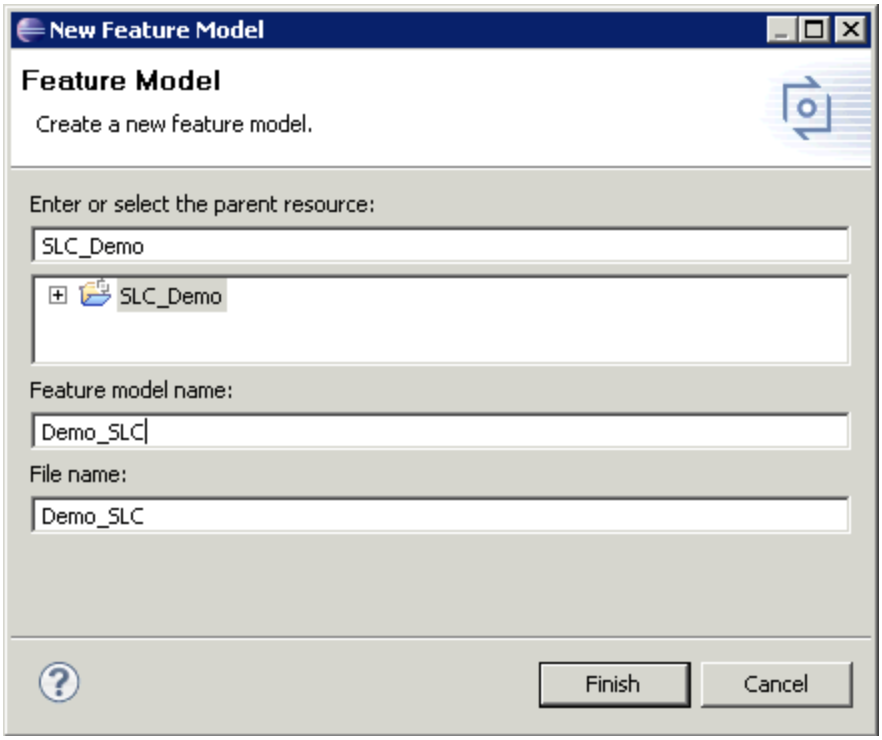
A new feature model can be created via the context menu item “New -> Feature Model” of the view “Variant Projects”.

Figure 65. Creating a new Feature Model



On the first page of the wizard that appears, enter the name for the new feature model and optionally a deviating file name. By default, the same name as the model is used here. The project is already selected as the target directory. If the new feature model should be created in another project or in a subdirectory, this must be selected in the tree.

Figure 66. Settings for a Feature Model



Upon finishing the wizard, the feature model is created and can now be used.

6.6. Creating a Simulink Model

6.6.1. Activity: Importing a Simulink Model

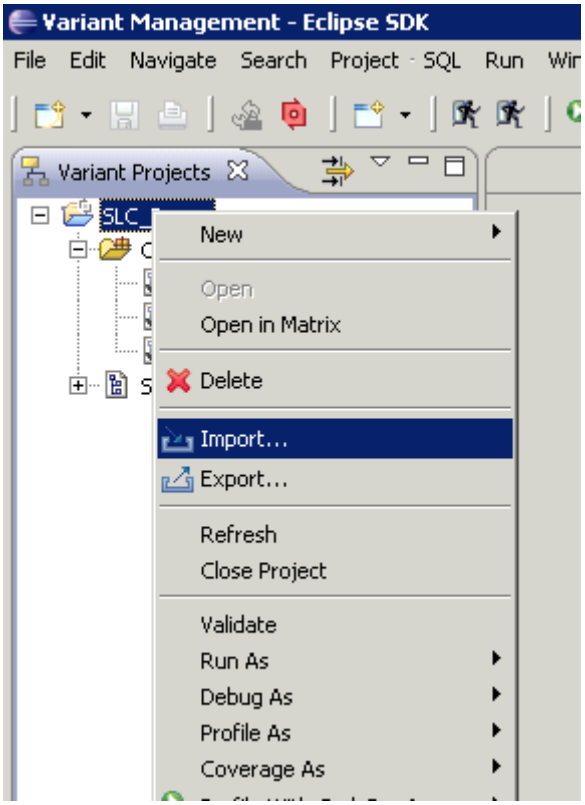
Prerequisites & Preliminary Work

- The Simulink model must exist in the current MATLAB/Simulink directory.
- If the variability information is stored in a Data Dictionary, this Data Dictionary must be open.

Process

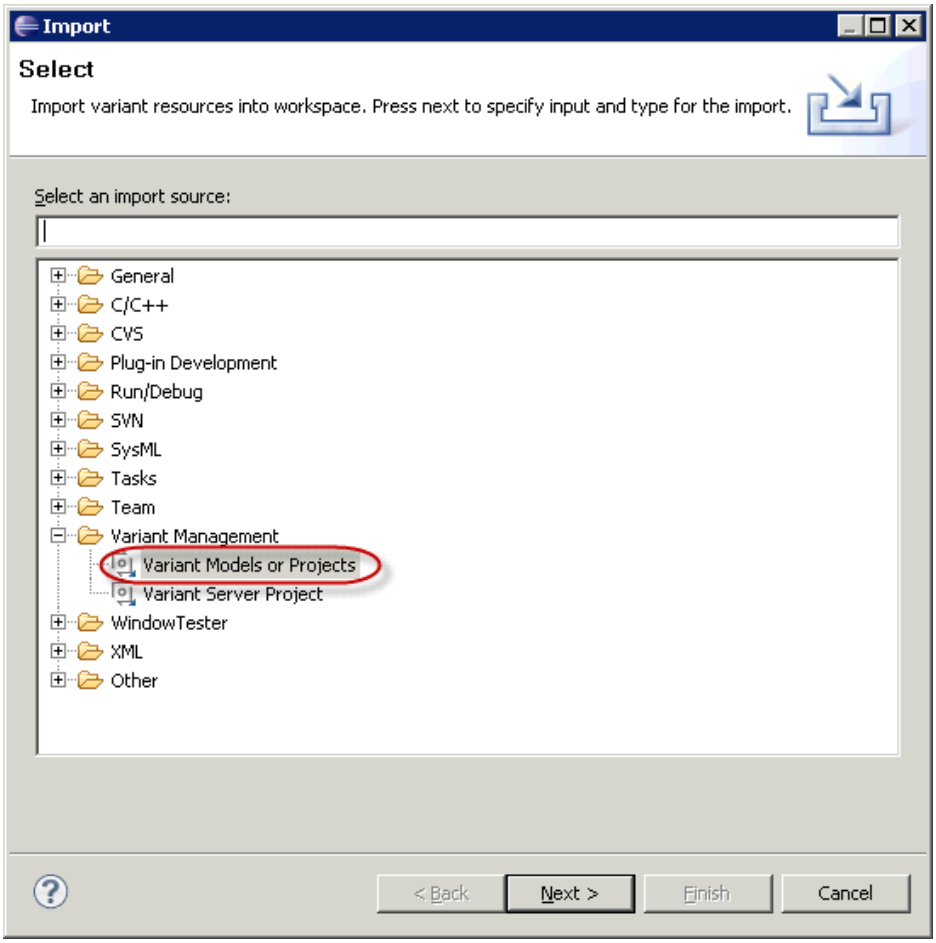
In pure::variants, the Import Wizard is used to import a Simulink model. To do this, select a pure::variants project in the “Variant Projects” view. The wizard can be opened via the context menu item “Import…”.

Figure 67. Opening Import Wizard



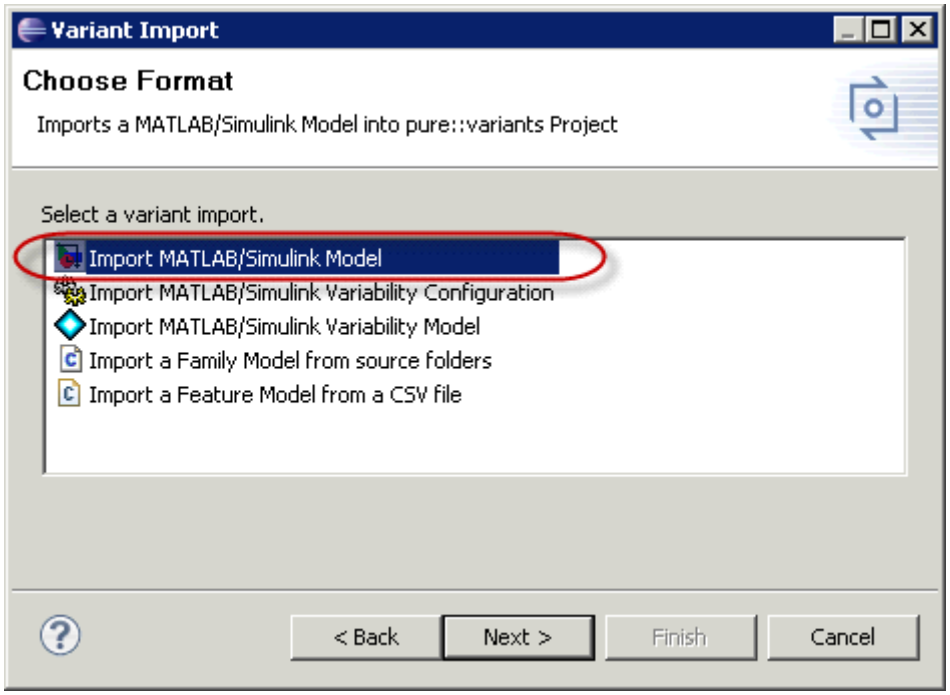
On the first page of the wizard that appears, select the entry “Variant Management Variant Models or Projects”.

Figure 68. Selecting Variant Model Import



The second page shows all currently possible imports to pure::variants. To import the Simulink model to pure::variants, the entry “Import MATLAB/Simulink Model” must be selected.

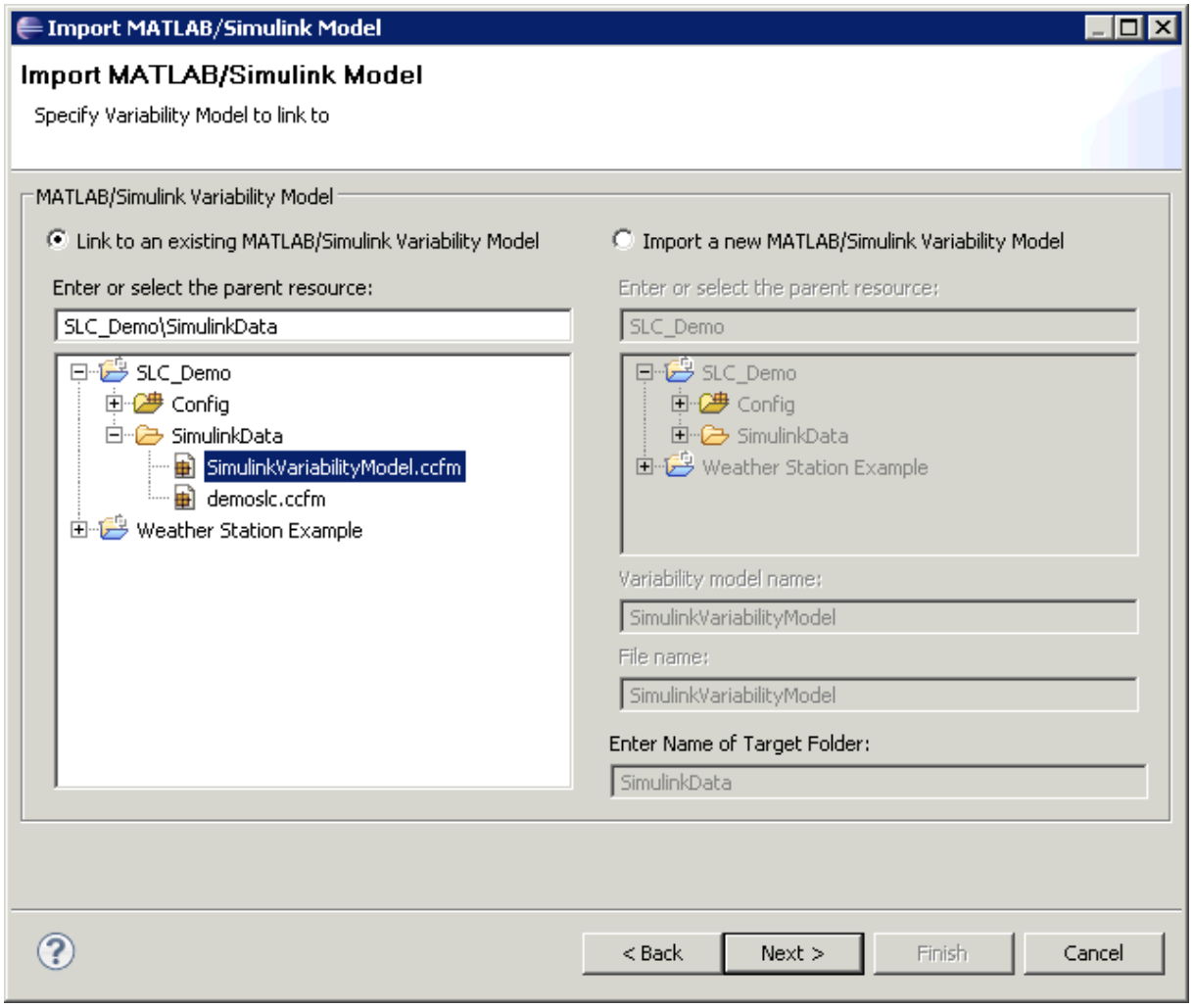
Figure 69. Selecting MATLAB/Simulink Model Importer



On the third page of the wizard, the settings for linking a variability model to the imported Simulink model are configured. The association of variant blocks in the Simulink model to variation points is also depicted in pure::variants. For this reason, the variability model must be known at the time of the Simulink model import.

One differentiates here between the creation of a new variability model into which all variation points are imported from MATLAB/Simulink first and linking to an already imported variability model. If an already imported model should be used the left part of the pane must be activated by selecting the item “Link to an existing MATLAB/Simulink Variability Model”. The variability model to link can then be selected. For importing a new variability model the right part must be activated by selecting “Import a new MATLAB/Simulink Variability Model”.

Figure 70. Settings for a Variability Model

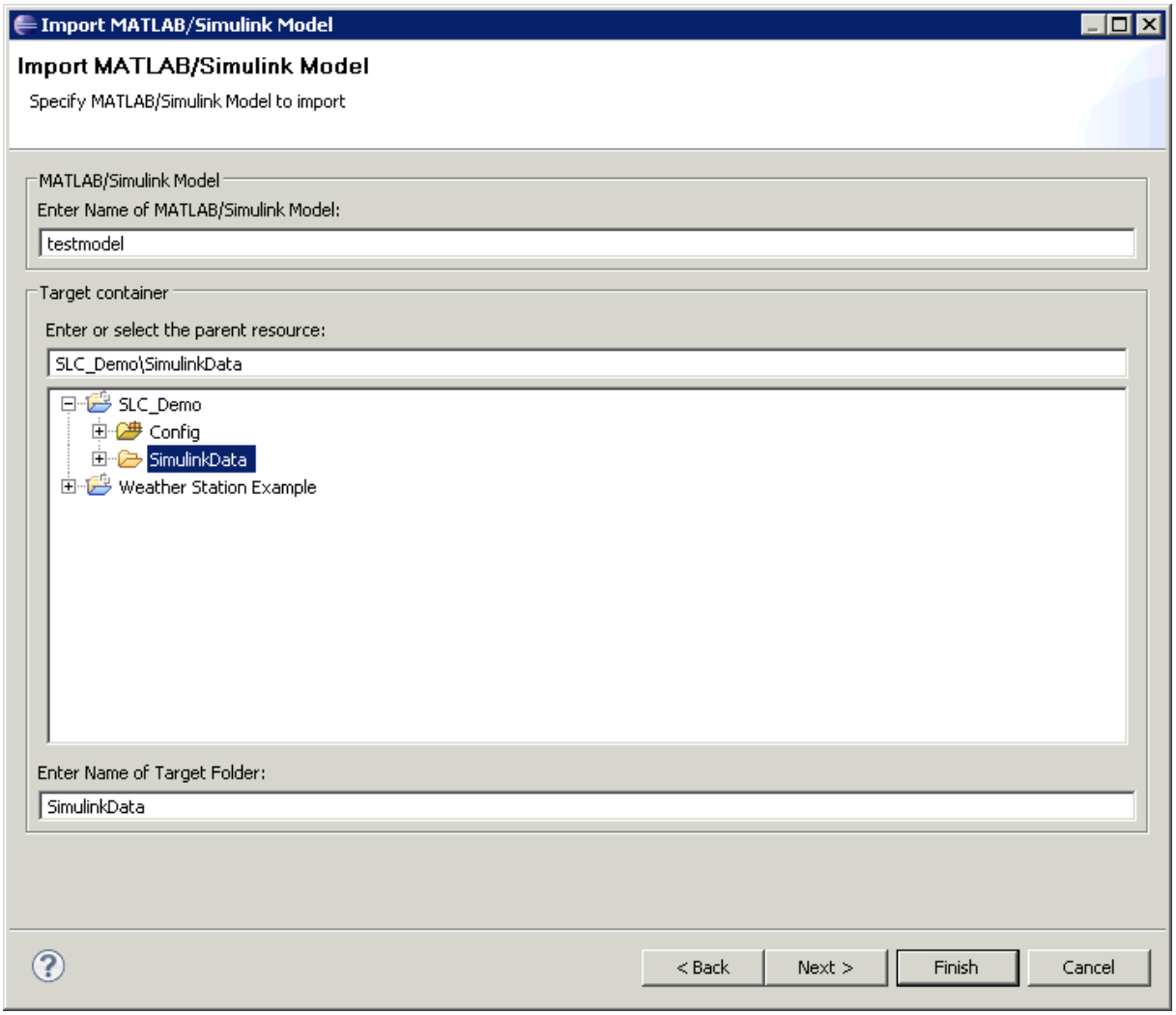


On the fourth wizard page the settings for importing the Simulink model are configured. In the top group “MATLAB/Simulink Model” the name of the Simulink model to import is specified. The name of the imported model and its corresponding file will be the same.

Note: The model name must match an existing model in the current MATLAB/Simulink directory. This is checked by the wizard, and an error message is displayed if no match is found.

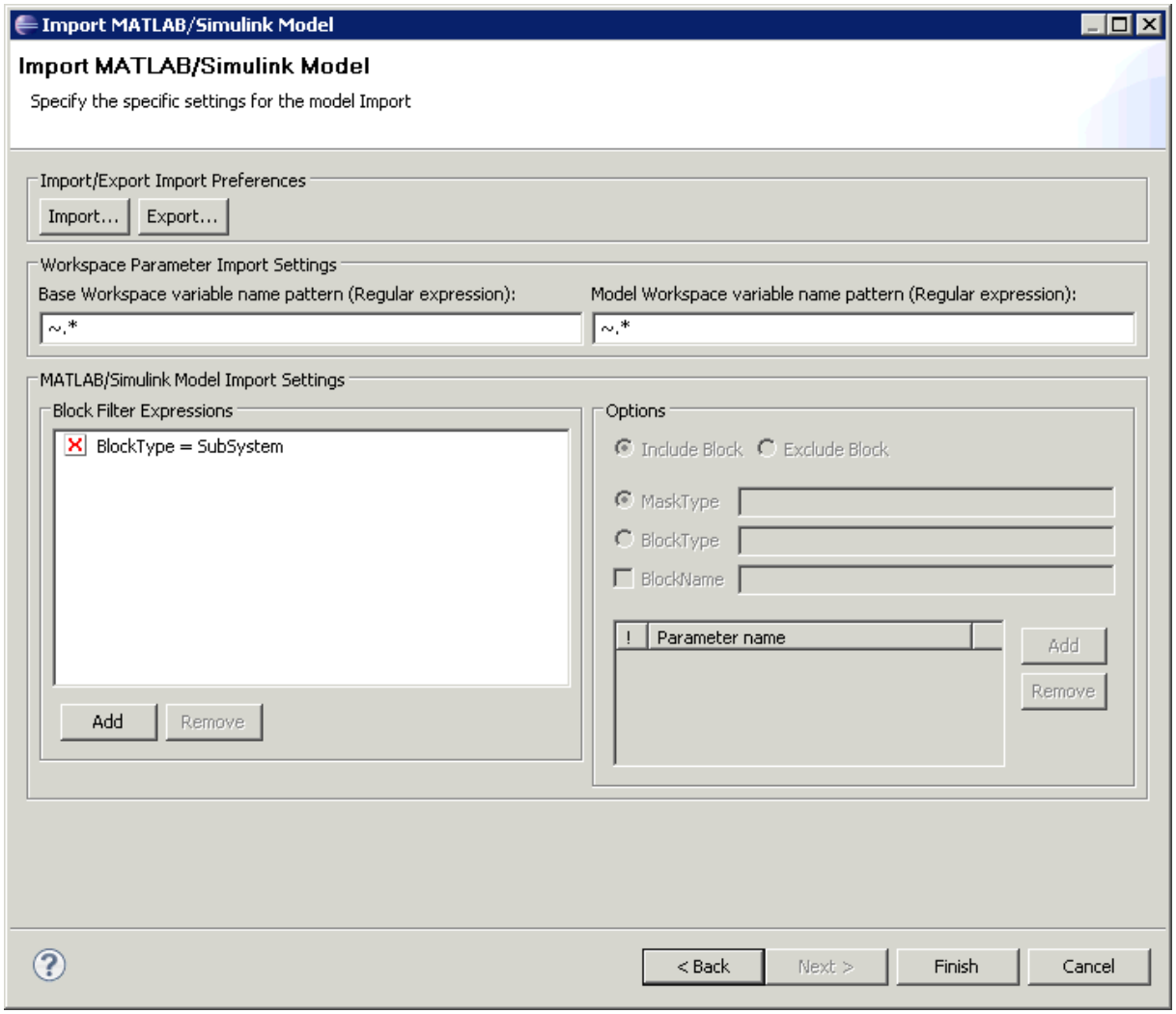
The parent resource and an optional target folder are selected in the middle and bottom group. If no target folder is entered, the imported model is saved directly under parent resource. Creating such a directory allows models imported by MATLAB/Simulink to be kept separate from others.

Figure 71. Settings for a Simulink Model



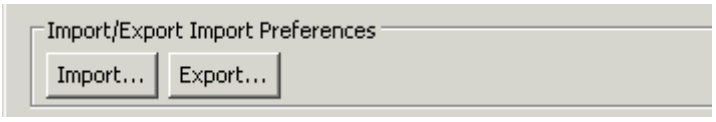
On the next page, settings for importing the Simulink model can be configured. By default, all information relevant to the variability is imported from the Simulink model. To import additional blocks or parameters, the settings can be adjusted accordingly.

Figure 72. Setting for Importer



The settings can be written to a file for subsequent reuse with “Export…”. This file can be reloaded with “Import…”, which fills in the page values with the corresponding information.

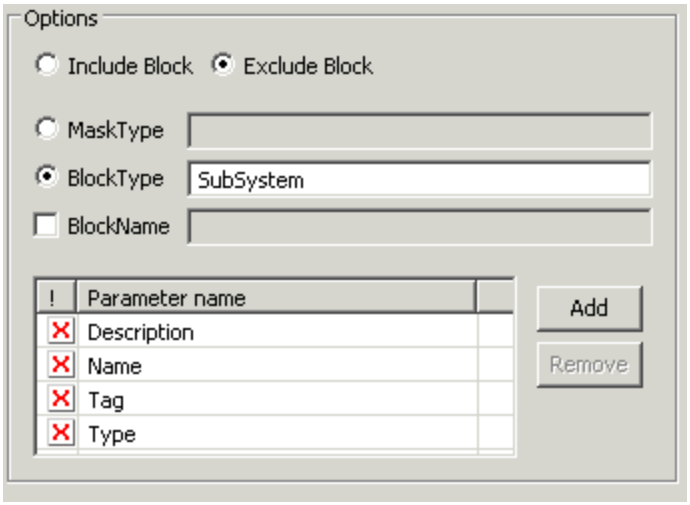
Figure 73. Import/Export of Settings



The settings for importing MATLAB/Simulink workspace and model workspace parameters can be configured in the group “Workspace Parameter Import Settings”. This requires that a regular expression be entered in the corresponding text fields.

Filter expressions for the blocks of the Simulink model can be created in the group “MATLAB/Simulink Model Import Settings” with “Add” and removed with “Remove”. All existing filter expressions are shown on the left side of the table. The symbol at the start of the line indicates whether blocks matching the defined filter are read ( ) or explicitly ignored ( ). If one of the expressions is selected, the right side shows all the configured information.

Figure 74. Setting of Blocks to be imported



“Include Block” and “Exclude Block” specify whether the selected block should be read or ignored. The entered expressions can filter by mask type (with “MaskType”) or by block type (with “BlockType”). It is also possible to enter a regular expression for the name of the block after activating the corresponding text field with “BlockName”.

If parameters should be read or explicitly ignored, these can be added with “Add”. The name of the parameter can be changed directly in the cell. The first column of the table specifies whether the parameter should be read ( ) or ignored ( ). “Remove” takes the selected parameter out of the list.

After finishing the wizard, the Simulink model is imported and the corresponding model is created in the specified target directory.

6.7. Modeling Dependencies for Variations

6.7.1. Activity: Linking a Variation with Features

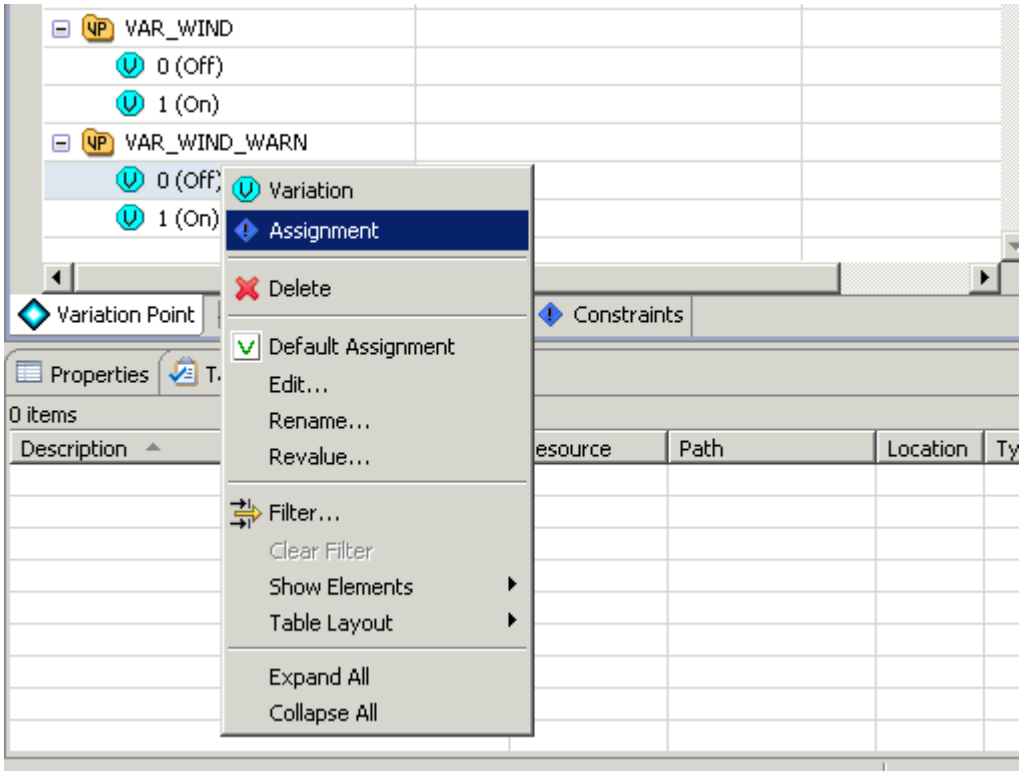
Prerequisites & Preliminary Work

- The variability model must be open and must contain at least one variation point with at least one variation.
- A feature model must exist with at least the feature to be linked.

Process

The “Variation Point” viewer of the Variability Model Editor is used for linking a variation with one or more features in pure::variants. In the viewer, select the variation for which an assignment should be created. The context menu item “Assignment” of the selected variation opens a wizard for creating an assignment.

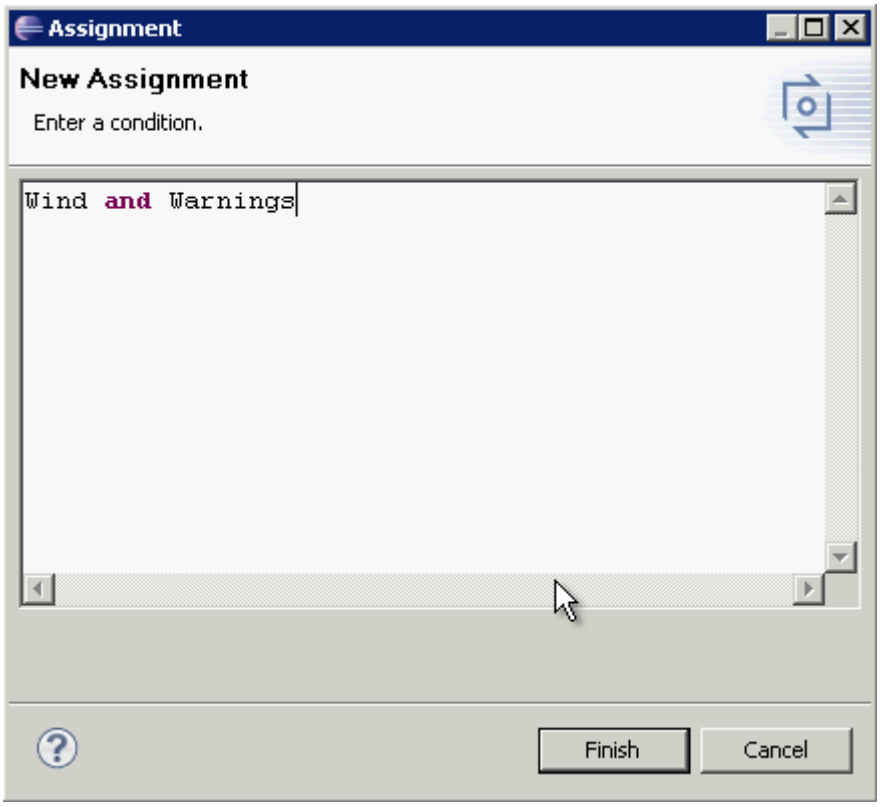
Figure 75. Creating a new Assignment



The condition of the variation must be entered in the text field. The condition must comply with the constraints of the pvSCL expression language and specific conventions. For more information on conventions, see Section 1.2, “Conventions”. Information about the pvSCL expression language can be found in the pure::variants user documentation in section “9.8 Expression Language pvSCL”.

The wizard offers an auto-completion feature that can be activated with the key combination “CTRL + Spacebar”. This helps in selecting features and logical relationships.

Figure 76. Assignment Dialog



After the wizard is finished, the condition is created as an assignment in the variability model and is visible in the column “Default Assignment/Assignment” in the viewer.

Possible Error Causes / Known Restrictions

It is possible to enter conditions for different variations of the same variation point such that more than one variation can be automatically selected during a variant point configuration. In this case, the auto resolver will arbitrarily select one of these variations. In the editor of the variant model, a marker will be attached to the variations that were not selected to indicate a failed condition.

The power of the pvSCL expression language opens up the risk of defining conditions that cannot be true. For example, “AND” relationships can be assigned to alternatives. Such rules are indicated in the variant model with a marker. However, it is also possible to not only reference features but also define comparisons of attribute values of features. For information about the conventions that apply to conditions, see Section 1.2, “Conventions”.

Other Actions

With the help of the created condition, automatic selection of a variation by the auto resolver is possible in pure::variants, which can simplify the process of configuring a variation point. If all required conditions have been added to a variability model, a complete variant configuration can be created (see Section 6.9.1, “Activity: Comparing Variation Point Configurations”).

6.7.2. Activity: Changing a Condition

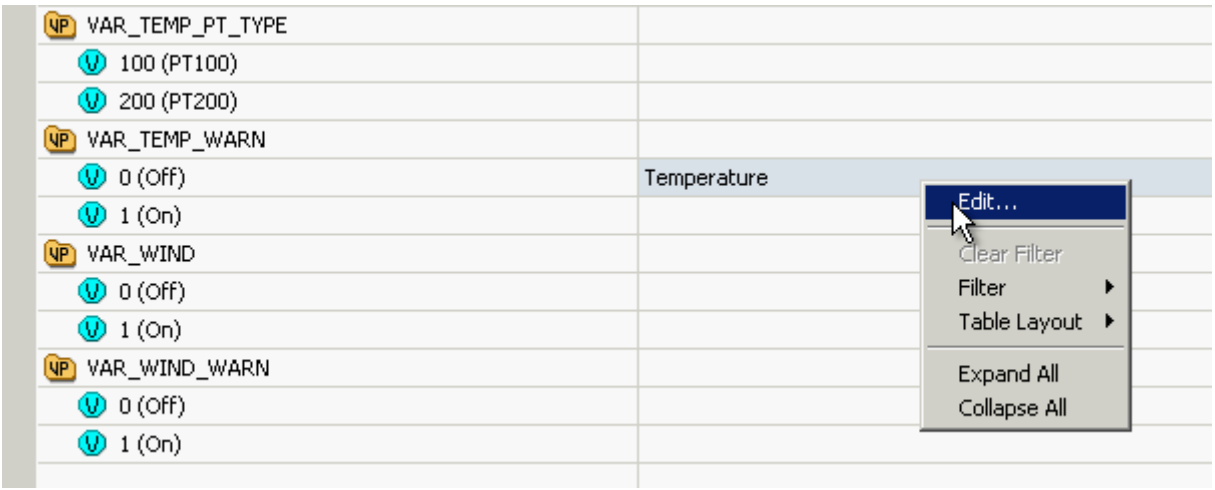
Prerequisites & Preliminary Work

- The variability model must be open and must contain at least one variation point with at least one variation.
- A condition exists for this variation.
- A feature model exists with at least the feature to be linked.

Process

The “Variation Point” viewer of the Variability Model Editor is used for changing a condition in pure::variants. Select this condition in the viewer. The context menu item “Edit” of the selected condition or a double-click on it opens a dialog for editing the condition. This dialog is initialized with the current information of the condition.

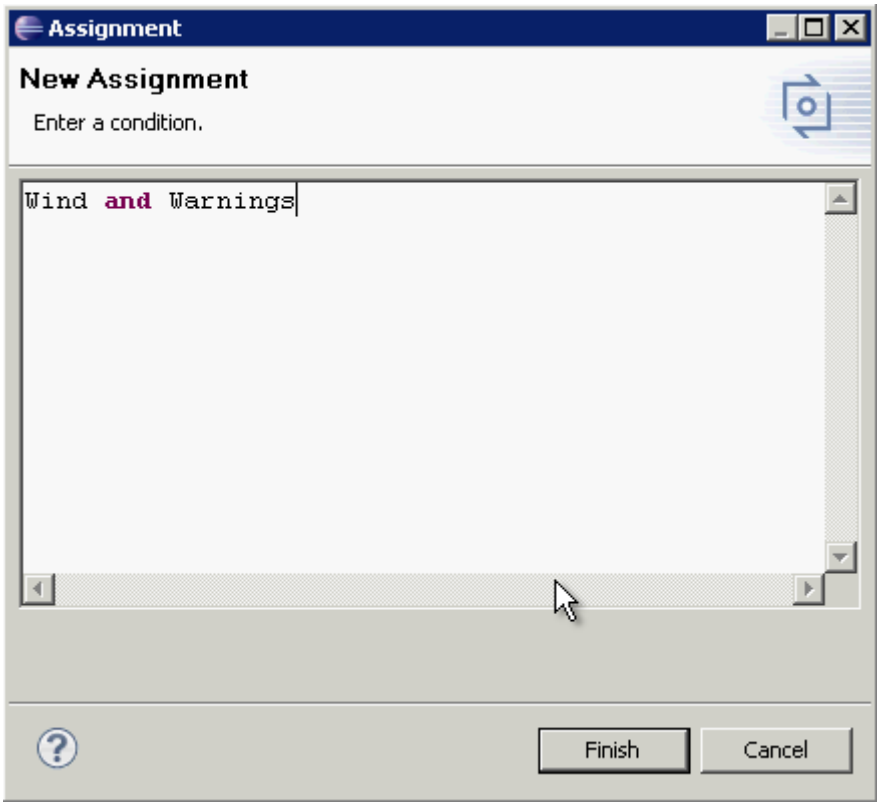
Figure 77. Changing an Assignment



The condition of the variation must be changed in the text field. The condition must comply with the constraints of the pvSCL expression language and specific conventions. For more information on conventions, see Section 1.2, “Conventions”.

The dialog offers an auto-completion feature that can be activated with the key combination “CTRL + Spacebar”. This helps in selecting other features or logical relationships.

Figure 78. Assignment Dialog



After the dialog is closed, the corresponding assignment in the variability model with the condition is changed and the changed condition is visible in the column “Default Assignment/Assignment” of the viewer.

Possible Error Causes / Known Restrictions

It is possible to enter conditions for different variations of the same variation point such that more than one variation can be automatically selected during a variant point configuration. In this case, exactly one of these variations is selected, without loss of generality. A marker is attached to the variations that were not selected to indicate a failed condition.

The power of the pvSCL expression language opens up the risk of defining conditions that cannot be true. For example, “AND” relationships can be assigned to alternatives. Such rules are indicated in the variant model with a marker. However, it is also possible to not only reference features but also define comparisons of attribute values. For information about the conventions that apply to conditions, see Section 1.2, “Conventions”

Other Actions

With the help of the changed condition, automatic selection of a variation by the auto resolver is possible in pure::variants, which can simplify the process of configuring a variation point. If all required conditions are available in a variability model, a complete variant configuration can be created (see Section 6.8.1, “Activity: Creating a Variation Point Configuration”).

6.7.3. Activity: Separating a Variation from Features

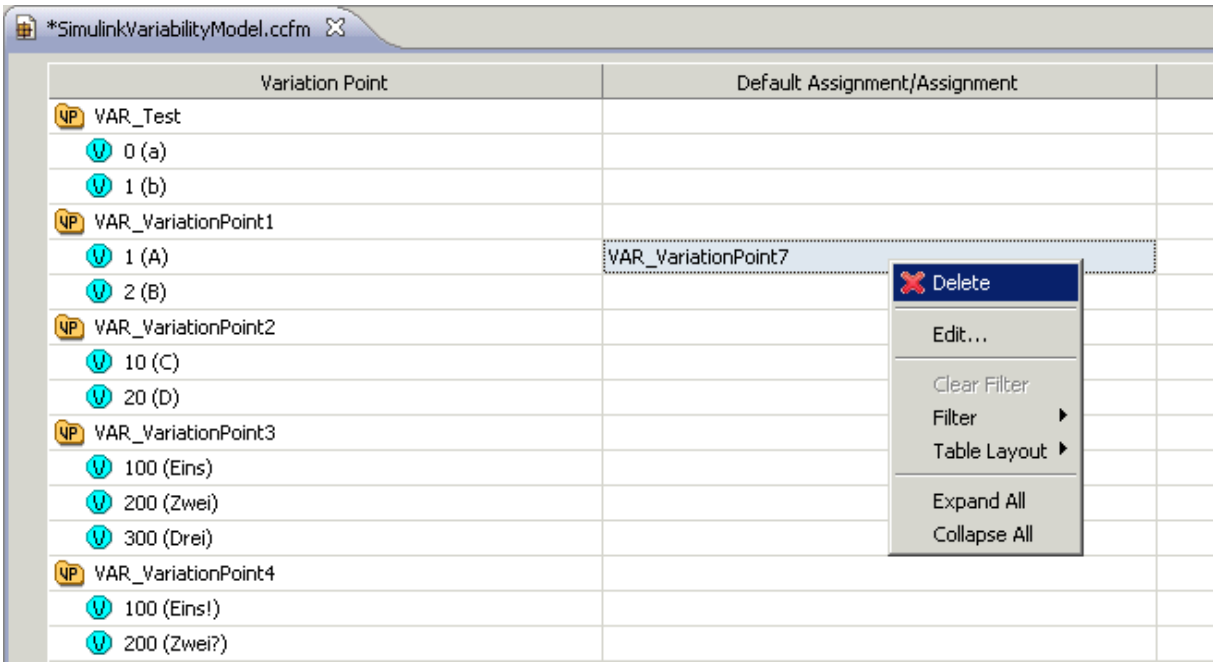
Prerequisites & Preliminary Work

- The variability model must be open and must contain at least one variation point with at least one variation.
- A condition must exist for this variation.

Process

The “Variation Point” viewer of the Variability Model Editor is used for deleting a condition in pure::variants. Select this condition in the viewer. The context menu item “Delete” of the selected condition opens a dialog for confirmation.

Figure 79. Deleting an Assignment



Upon confirmation of the deletion in the dialog that appears, the condition as well as the link between the variation and the features are eliminated and the assignment is removed from the variability model.

6.8. Creating Configurations

6.8.1. Activity: Creating a Variation Point Configuration

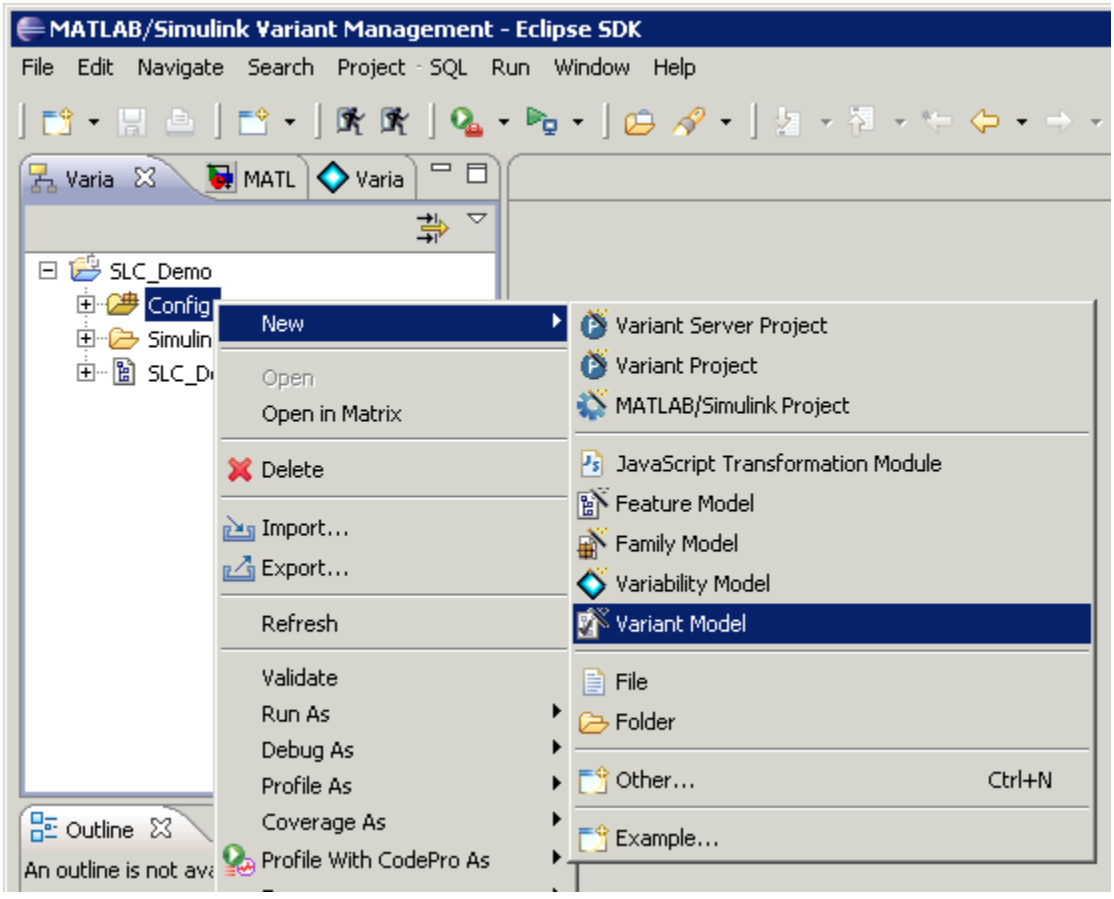
Prerequisites & Preliminary Work

- The variability information associated with the variability model must be synchronized (see also Section 6.4.1, “Activity: Synchronizing to MATLAB/Simulink”).
- A configuration space must exist that contains this variability model as well as at least one feature model.

Process

To create a variation point configuration in pure::variants, one uses a variant model that will also contain the corresponding feature selection. This variant model must first be created in the “Variant Projects” view. The context menu item “New -> Variant Model” of the selected configuration space opens a wizard for creating a new variant model.

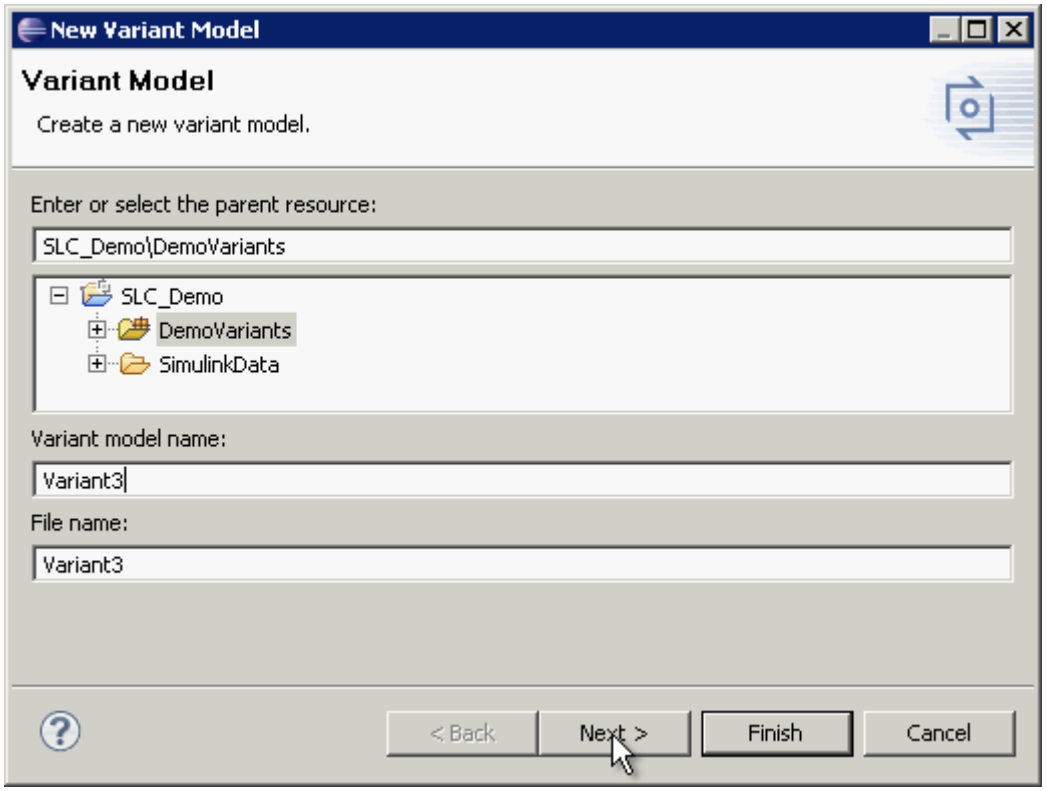
Figure 80. Creating a new Variation Point Configuration



On the first page of the wizard that appears, enter the name for the new variant model and optionally a deviating file name. By default, the same name as the model is used here. The configuration space is already selected as the target directory. If the new variant model should be created in another configuration space, this must be selected in the tree.

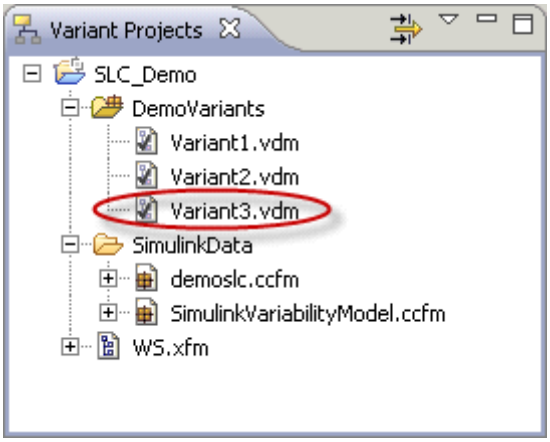
Note: A variant model must always be created within a configuration space.

Figure 81. Settings for a Variation Point Configuration



After the wizard is finished, the new variant model is created within the configuration space. This contains neither a valid variant point configuration nor a valid feature selection. However, the auto resolver may have selected a variation modeled as “Default Assignment” for each of the different variation points.

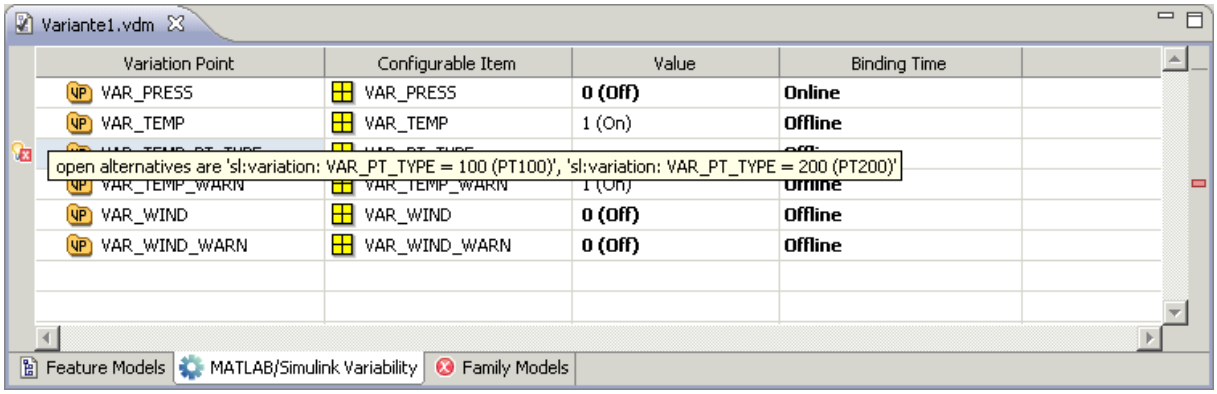
Figure 82. Variation Point Configuration in a Project



Other Actions

Due to the lack of selected features and the incomplete selection of a variation for a variation point, the variant model is considered by pure::variants as invalid. This is indicated with markers at the places in the editor where errors exist. These markers describe the applicable problem more precisely and offer possible solutions.

Figure 83. Configuring value-based Variation Points



The desired configuration can now be created in the newly created and opened variant model. The corresponding features can be selected in the “Feature Models” viewer. Manual configuration of the variation points through a variation can be done in the “MATLAB/Simulink Variability” viewer.

6.8.2. Activity: Importing a Configuration of Variation Points

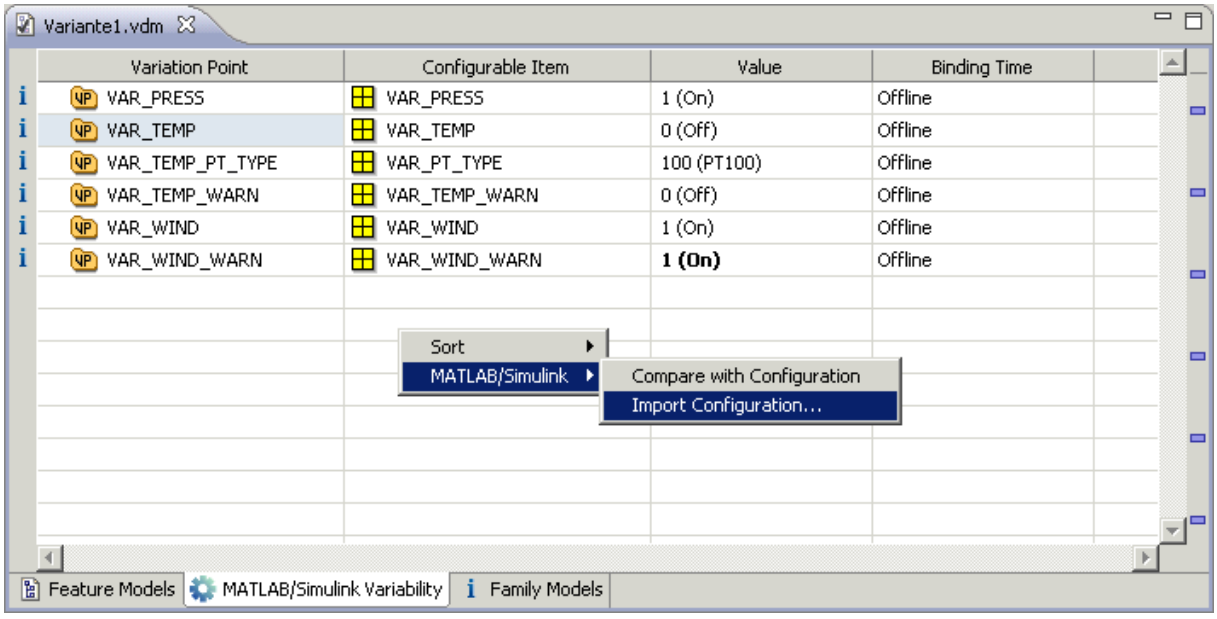
Prerequisites & Preliminary Work

- The Simulink model or the Data Dictionary associated with the Simulink model must be open.
- The variability information associated with the variability model must be synchronized (see also Section 6.4.1, “Activity: Synchronizing to MATLAB/Simulink”).
- A variant model with a variation point configuration must be open.

Process

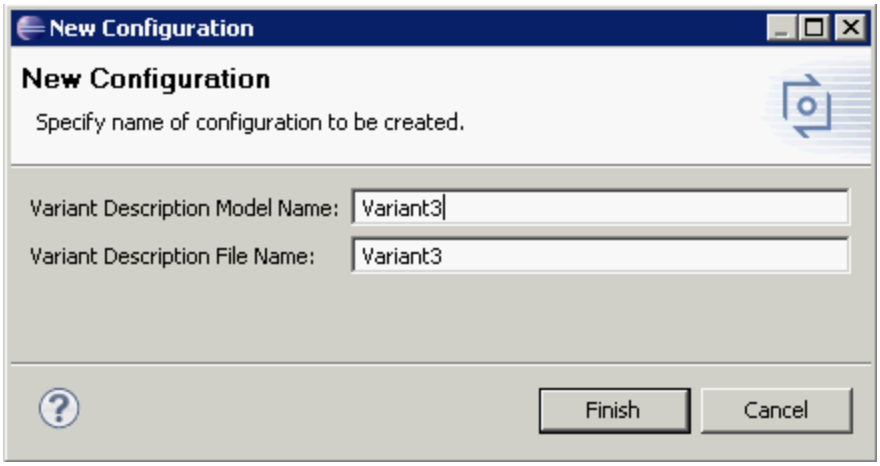
In pure::variants, the “MATLAB/Simulink Variability” viewer is used to import a configuration of variation points from MATLAB/Simulink to an existing feature selection. This requires that a variant model with a corresponding feature selection be open. The context menu item “MATLAB/Simulink -> Import Configuration…” opens a wizard for importing the configuration of variation points into a new variant model, in which the feature selection of the open one is performed automatically.

Figure 84. Importing a Variation Point Configuration



On the first page of the wizard that appears, enter the name for the new variant model and optionally a deviating file name. By default, the same name as the model is used here. The new variant model is created in the same configuration space as the open model.

Figure 85. Settings for a Variation Point Configuration



After the wizard is finished, the new variant model is created and contains the variation point configuration that exists in MATLAB/Simulink and the feature selection of the variant model used for the import.

If all variations of a variation point in this variant model are marked as "Excluded" (), these are either not configured in MATLAB/Simulink and/or configured with a variation that does not exist in pure::variants or the entire variation point does not exist in MATLAB/Simulink.

Other Actions

The newly created variant model with the variation point configuration from MATLAB/Simulink can be used for comparison with a variation point configuration that already exists in pure::variants (see Section 6.9.2, "Activity: Comparing Variant Models"). If undesired differences are discovered, the desired result may be achieved by adjusting the feature selection in the existing variant model and/or adjusting the conditions for the automatic configuration of the variation points.

6.8.3. Activity: Propagating a Variation Point Configuration

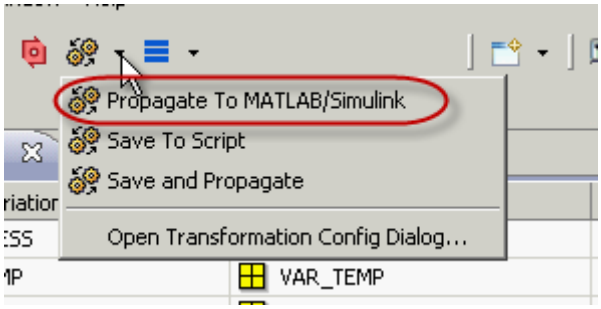
Prerequisites & Preliminary Work

- The Simulink model or the Data Dictionary associated with the Simulink model must be open.
- The variability information associated with the variability model must be synchronized (see also Section 6.4.1, "Activity: Synchronizing to MATLAB/Simulink").
- The variant model with the variation point configuration must be open.
- The configuration space must have a transformation configuration with the module "Simulink Configuration Propagator" (see here Section 7.4.6, "Activity: Creating Transformation Configurations")

Process

In pure::variants, a transformation is used to propagate a variation point configuration to MATLAB/Simulink. This requires that the variant model with the variation point configuration be open. The transformation button in the toolbar of pure::variants, which is only visible when a variant model is open, starts the corresponding transformation.

Figure 86. Propagating a Variation Point Configuration



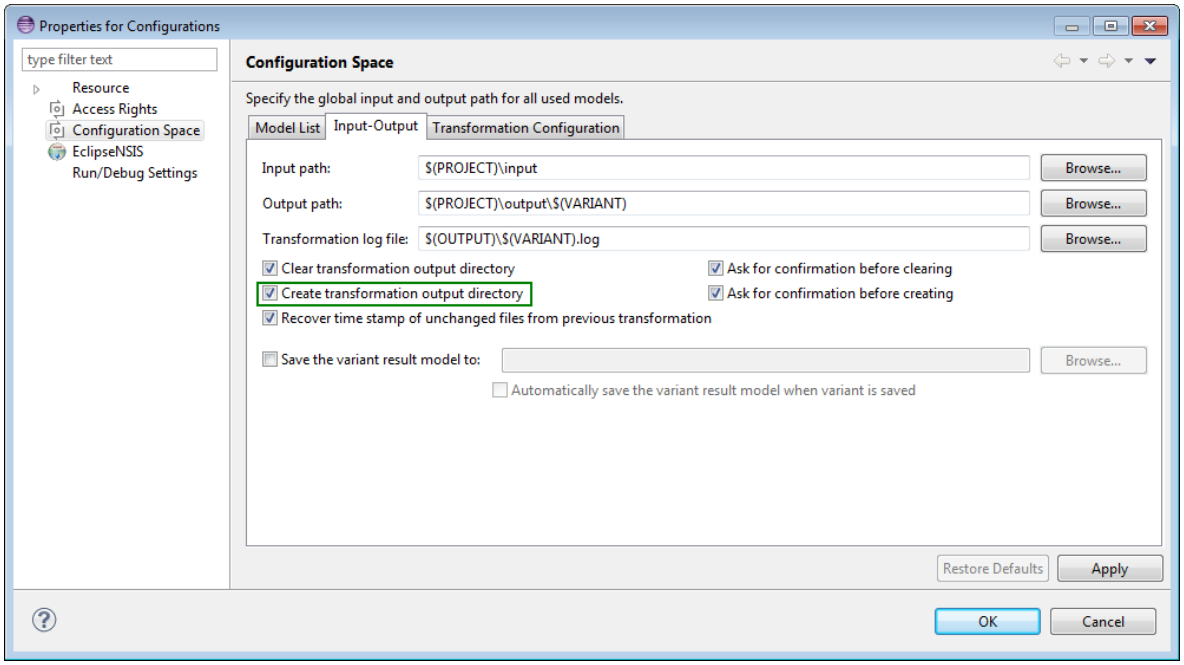
It should be noted that the name displayed in the menu matches the name specified in the transformation configuration. After the transformation has been executed, the variation point configuration has been propagated to MATLAB/Simulink and all variation points in MATLAB/Simulink have been configured with the respective variation.

It is not necessary for the variant model to contain a valid feature selection or variant configuration. Before the propagation, a confirmation dialog will appear to inform you of this situation. However, the user is permitted to propagate the variation point configuration to MATLAB/Simulink anyway.

Possible Error Causes / Known Restrictions

If the transformation is ended with the error message that the output directory was not found, this directory must either be created manually or the the option “Create transformation output directory” must be set in the transformation configuration (see Section 7.4.4, “Activity: Creating a Configuration Space”).

Figure 87. Configuring Transformation Pathes



6.8.4. Activity: Saving a Variation Point Configuration

Prerequisites & Preliminary Work

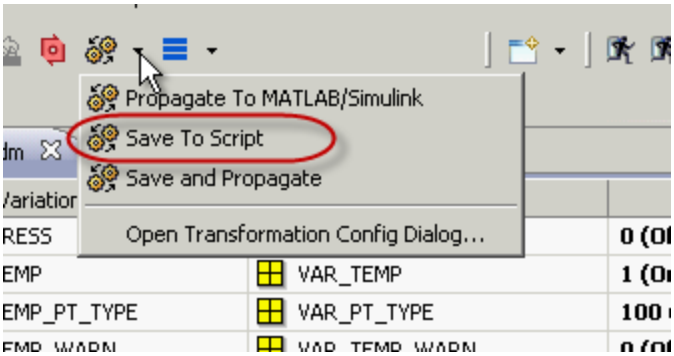
- The variability information associated with the variability model must be synchronized (see also Section 6.4.1, “Activity: Synchronizing to MATLAB/Simulink”).
- The variant model with the variation point configuration must be open.

- The configuration space must have a transformation configuration with the module “Simulink Configuration m-File Writer” (see here Section 7.4.6, “Activity: Creating Transformation Configurations”)

Process

In pure::variants, a transformation is used to save a variation point configuration as a MATLAB script. This requires that the variant model with the variation point configuration be open. The transformation button in the toolbar of pure::variants, which is only visible when a variant model is open, starts the corresponding transformation.

Figure 88. Saving a Variation Point Configuration



It should be noted that the name displayed in the menu matches the name specified in the transformation configuration. After the transformation has been executed, the variation point configuration is saved in a MATLAB script in the output directory. With the default settings of pure::variants, this directory is <PROJECT_NAME>/output/<VARIANT_MODEL_NAME>/script.m.

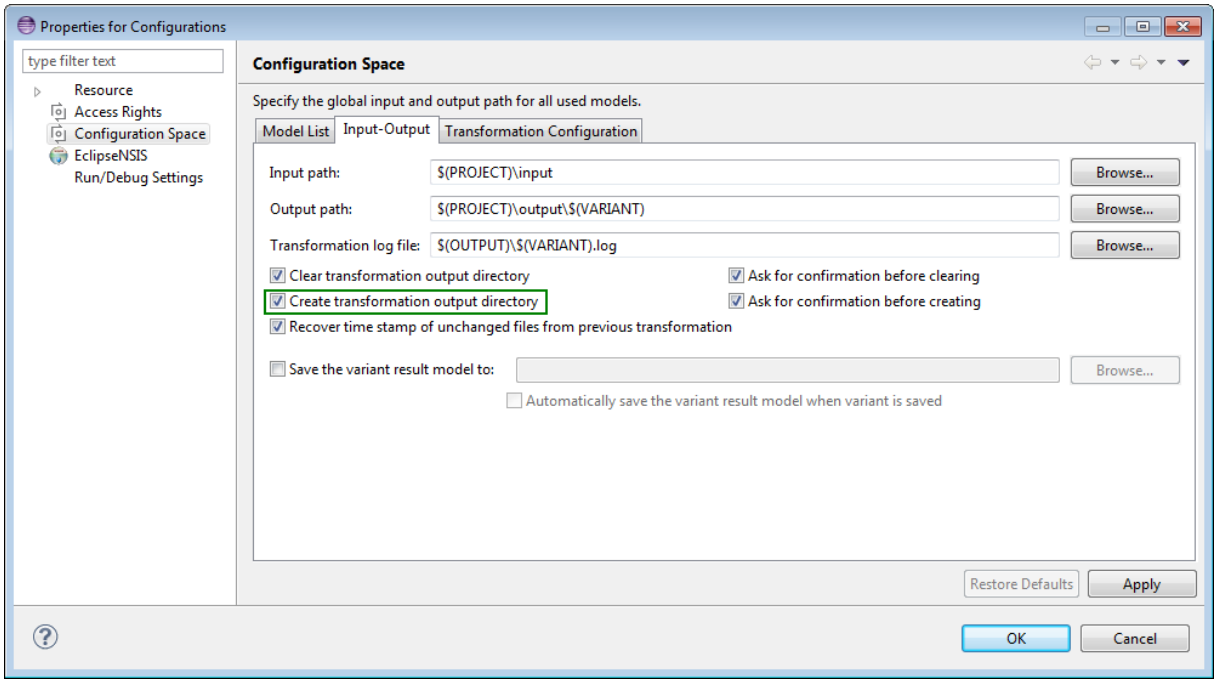
It is not necessary for the variant model to contain a valid feature selection or variant configuration. Before the propagation, a confirmation dialog will appear to inform you of this situation. However, the user is permitted to write the variation point configuration to a MATLAB script anyway.

Possible Error Causes / Known Restrictions

The transformation module “Simulink Configuration m-File Writer” allows the specification of an output file for the MATLAB script. If no file name is specified, a file selection dialog appears during the transformation. Otherwise, this file name is used for saving the MATLAB script. A path may also be placed before the name which is interpreted relative to the output directory.

If the transformation is ended with the error message that the output directory was not found, this directory must either be created manually or the option “Create transformation output directory” must be set in the transformation configuration (see Section 7.4.4, “Activity: Creating a Configuration Space”).

Figure 89. Configuring Transformation Pathes



Other Actions

The generated MATLAB script can be executed directly in MATLAB/Simulink. The variation point configuration saved in the script is then placed in the open Data Dictionary, and all variation points in MATLAB/Simulink are configured with the respective variation.

6.9. Comparing Configurations

6.9.1. Activity: Comparing Variation Point Configurations

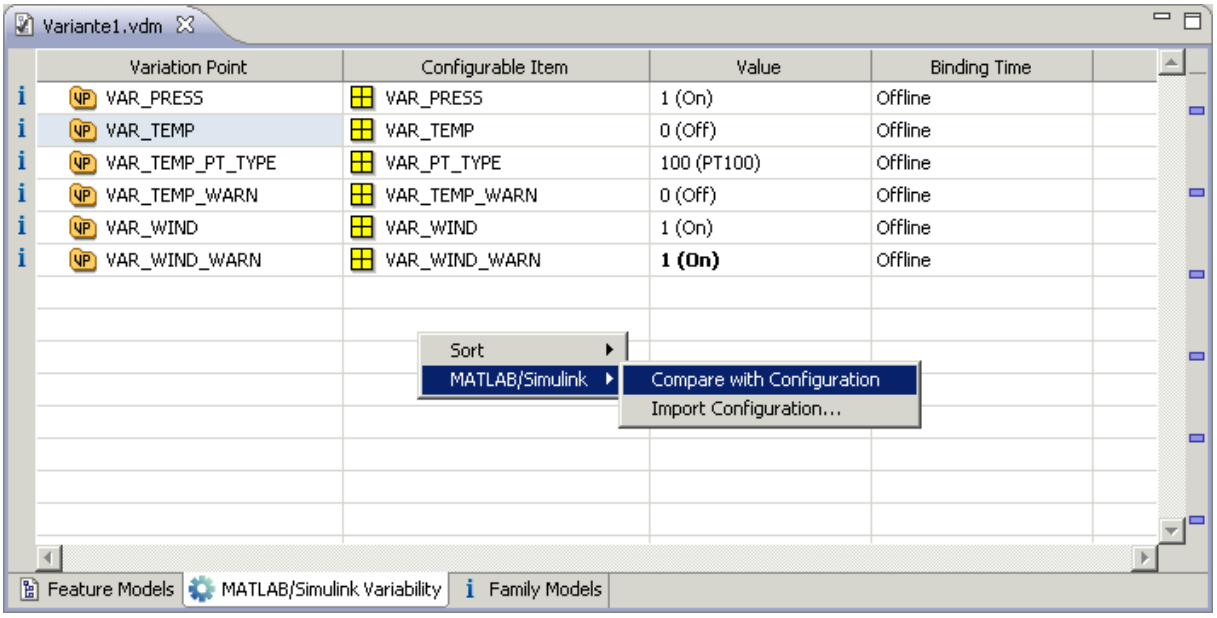
Prerequisites & Preliminary Work

- The Simulink model or the Data Dictionary associated with the Simulink model must be open.
- The variability information associated with the variability model must be synchronized (see also Section 6.4.1, “Activity: Synchronizing to MATLAB/Simulink”).
- The variant model with the variation point configuration must be open.

Process

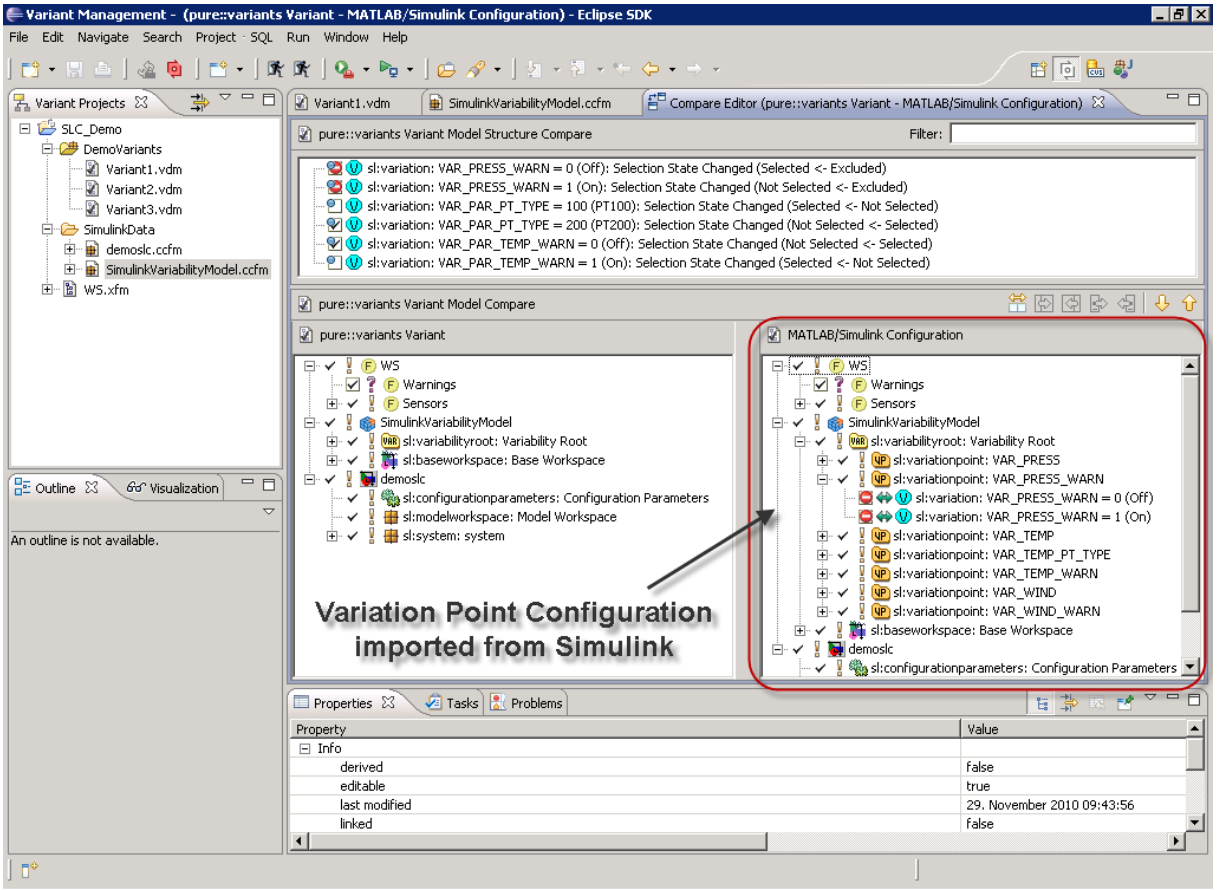
In pure::variants, the “MATLAB/Simulink Variability” viewer is used to compare a variation point configuration from MATLAB/Simulink with a variation point configuration of a variant model. This requires that a variant model be open. The context menu item “MATLAB/Simulink -> Compare with Configuration” opens the Compare Editor for comparing the variation point configurations.

Figure 90. Comparing Variation Point Configurations



The lower left side shows the current variability information of the variability model of pure::variants, and the lower right side shows the current variability information of MATLAB/Simulink as a newly imported variability model. The top part shows all changes in the two models. Marking a change causes the associated elements in the lower trees to be highlighted and the differences are displayed there more precisely.

Figure 91. Compare Editor



If all variations of a variation point in the lower right variant model are marked as “Excluded” ( ), these are either not configured in MATLAB/Simulink and/or configured with a variation that does not exist in pure::variants or the entire variation point does not exist in MATLAB/Simulink.

6.9.2. Activity: Comparing Variant Models

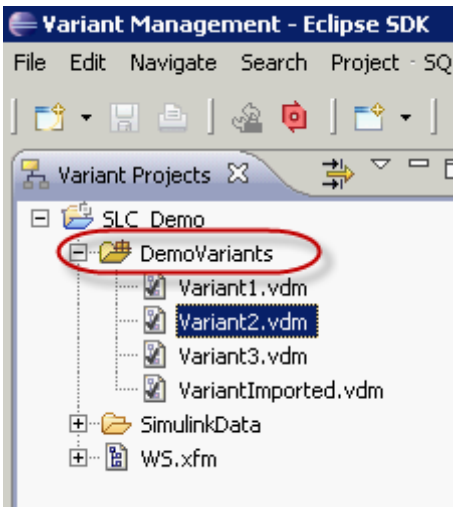
Prerequisites & Preliminary Work

- At least two variant models must exist in the same configuration space.

Process

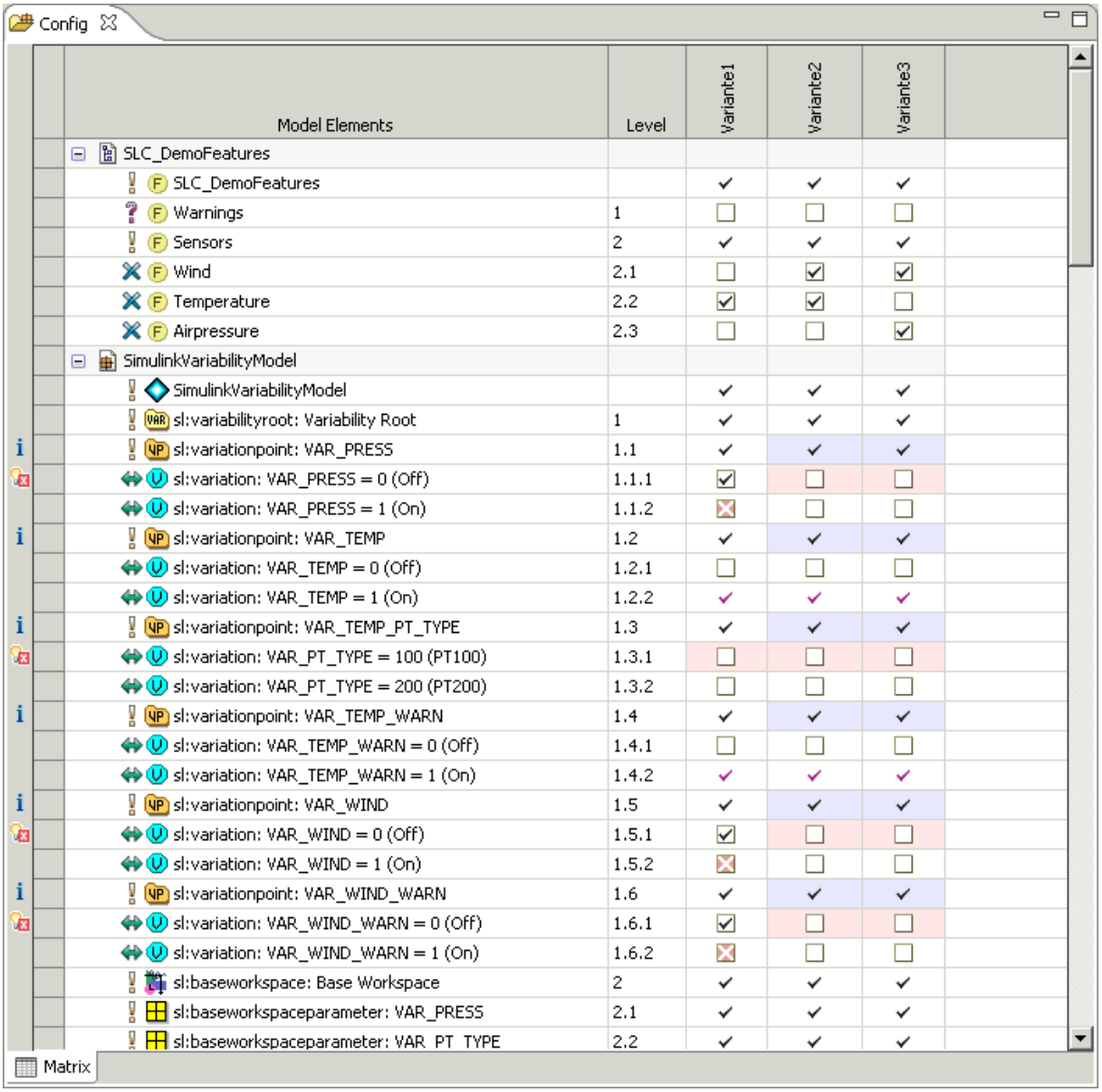
In pure::variants, the Configuration Space Editor is used to compare two or more variant models. To do this, select the variant model to be compared in the “Variant Projects” view and open it in the Configuration Space Editor via the context menu item “Open in Matrix”. If you wish to compare all variant models of the configuration space, you can open these in the Configuration Space Editor all at once by double-clicking on the configuration space.

Figure 92. Configuration Space in a Project



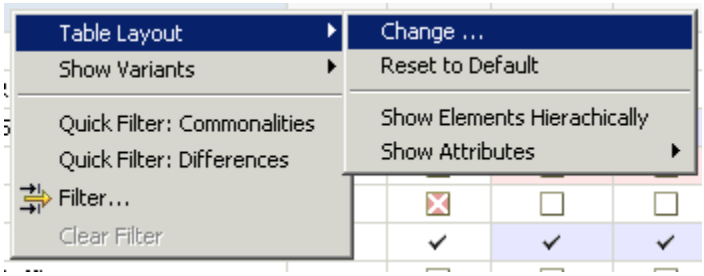
The Configuration Space Editor that appears compares all selected variant models in a table in columns. This allows a line-by-line comparison of the individual configurations, such as feature selections and variation point configurations. If you discover undesired differences, the variant models can be reconfigured in the Configuration Space Editor by selecting or deselecting features or a variation.

Figure 93. Matrix Editor



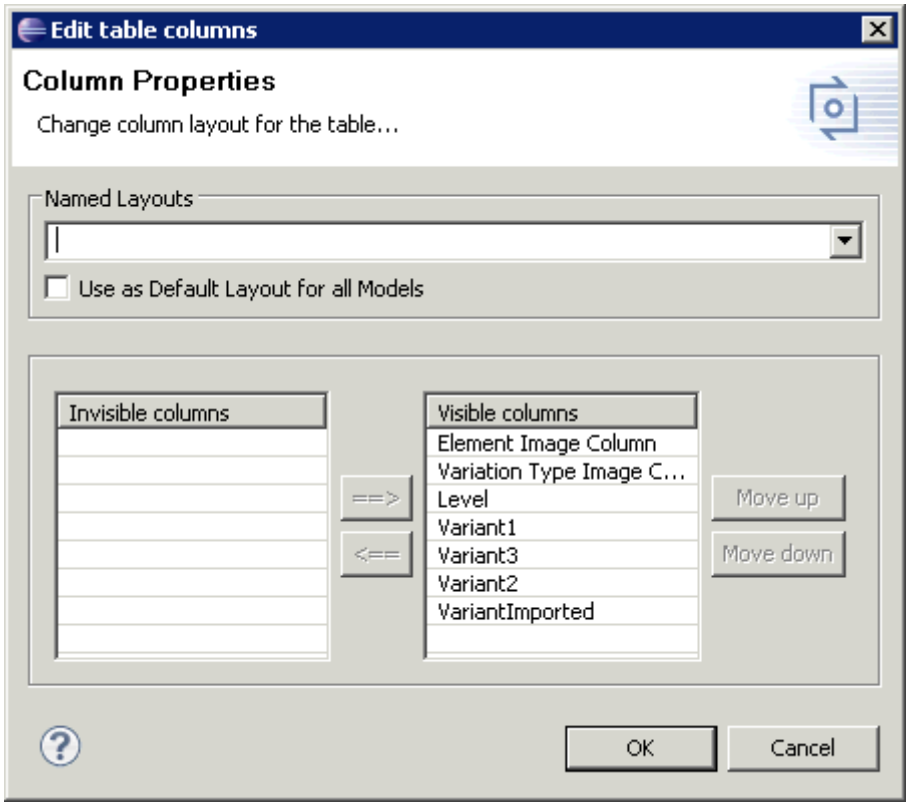
The columns in the table can be reordered, hidden or unhidden. For this purpose, the “Edit table columns” dialog can be opened via the context menu item “Table Layout -> Change…”.

Figure 94. Changing Table Layout



All columns listed under “Visible columns” are displayed in the Configuration Space Editor. To hide a column, select it and use “<==” to move it to “Invisible columns”.

Figure 95. Table Layout



The visible columns can be reordered with “Move Up” and “Move Down”.

Other Actions

If you only wish to compare two variant models at one time, the Compare Editor in pure::variants can also be used (see Section 7.4.3, “Activity: Comparing Two Variant Models”).

7. Additional pure::variants Procedures

7.1. Migration to the New Version

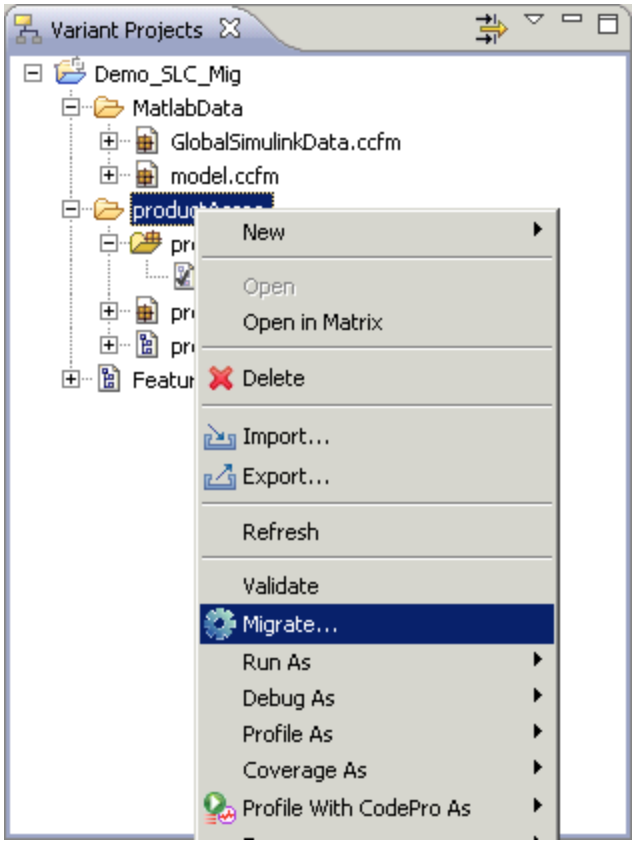
Migration to the new version is performed automatically by pure::variants and can only be executed once. It changes the structure and content of the models. The original models are backed up with the ending “migrated”. These are neither changed nor deleted, making them easy to locate.

7.1.1. Activity: Migration of the Models

Process

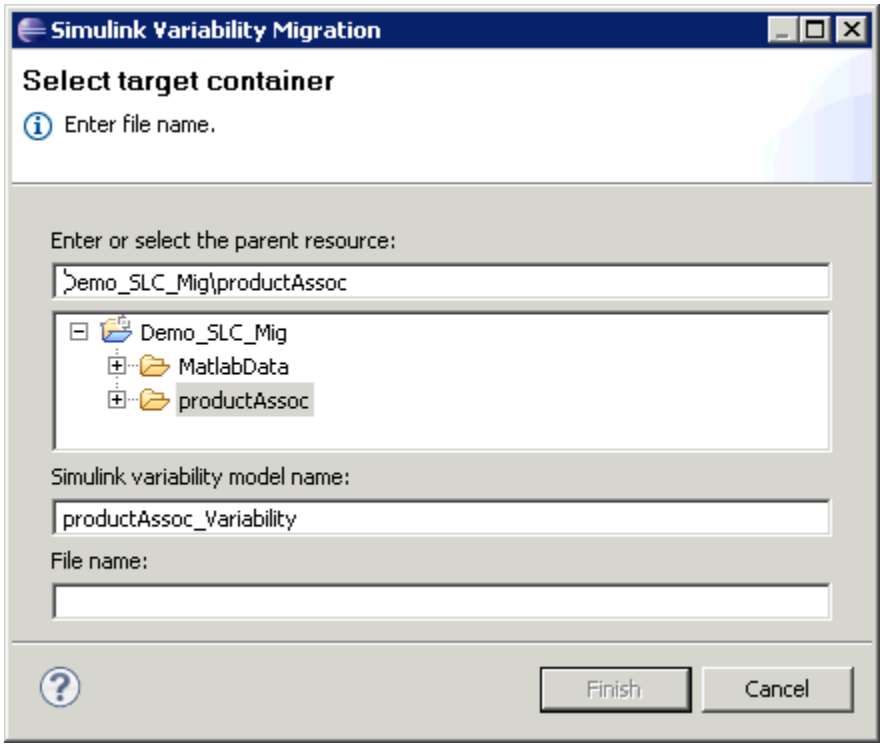
To perform the migration, the association directory, which contains the association model and the variability model, must be selected in the “Variant Projects” view. The migration dialog can be opened via the context menu item “Migrate…”.

Figure 96. Migration to the new Version



A file name for the new variability model must be entered into the corresponding text field in the dialog that appears. The name of the new variability model is initialized with the name of the old model. The association directory is already selected as the target directory, but this can be changed.

Figure 97. Settings for a Variability Model



When the dialog is closed, the automatic migration of the models begins.

1. The variability model (<Name>_Variability.xfm) and the association model (<Name>_Assiciation.ccfm) in the association directory are merged into the new variability model that was specified in the dialog.

2. If a configuration space exists in the association directory, the list of referenced models is adapted in that the old variability model and the old association model are removed and the new variability model added.

3. The existing variant models in the configuration space are adjusted in that they now link to elements of the new variability model.

All models that served exclusively as predecessor versions remain available and unchanged after the migration except that “migrated” has been appended to their names. The variant models contained in the configuration space are also backed up in this way. This means that no data loss can occur even if no version control system is used.

7.2. Importing from MATLAB/Simulink

7.2.1. Activity: Importing a Variation Point Configuration

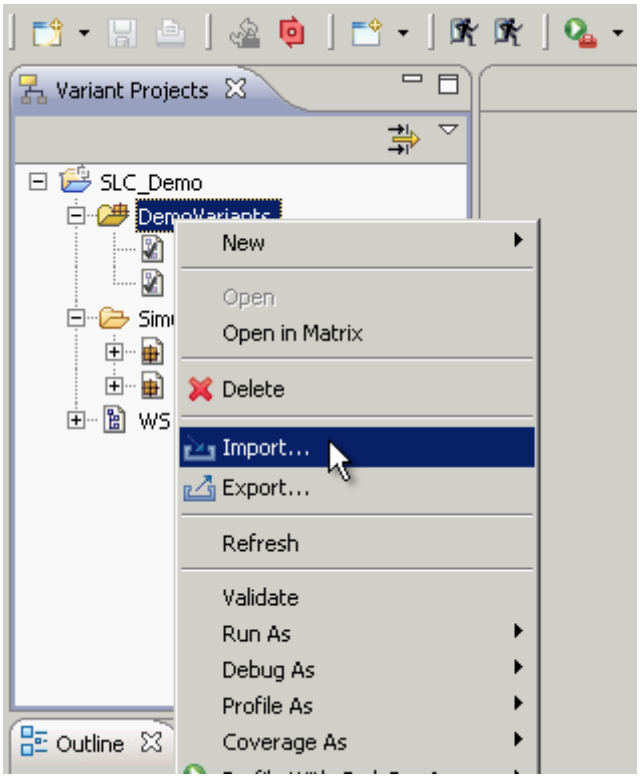
Prerequisites & Preliminary Work

- The Simulink model or the Data Dictionary associated with the Simulink model must be open.
- The variability information associated with the variability model must be synchronized (see also Section 6.4.1, “Activity: Synchronizing to MATLAB/Simulink”).
- A configuration space containing this variability model must exist.

Process

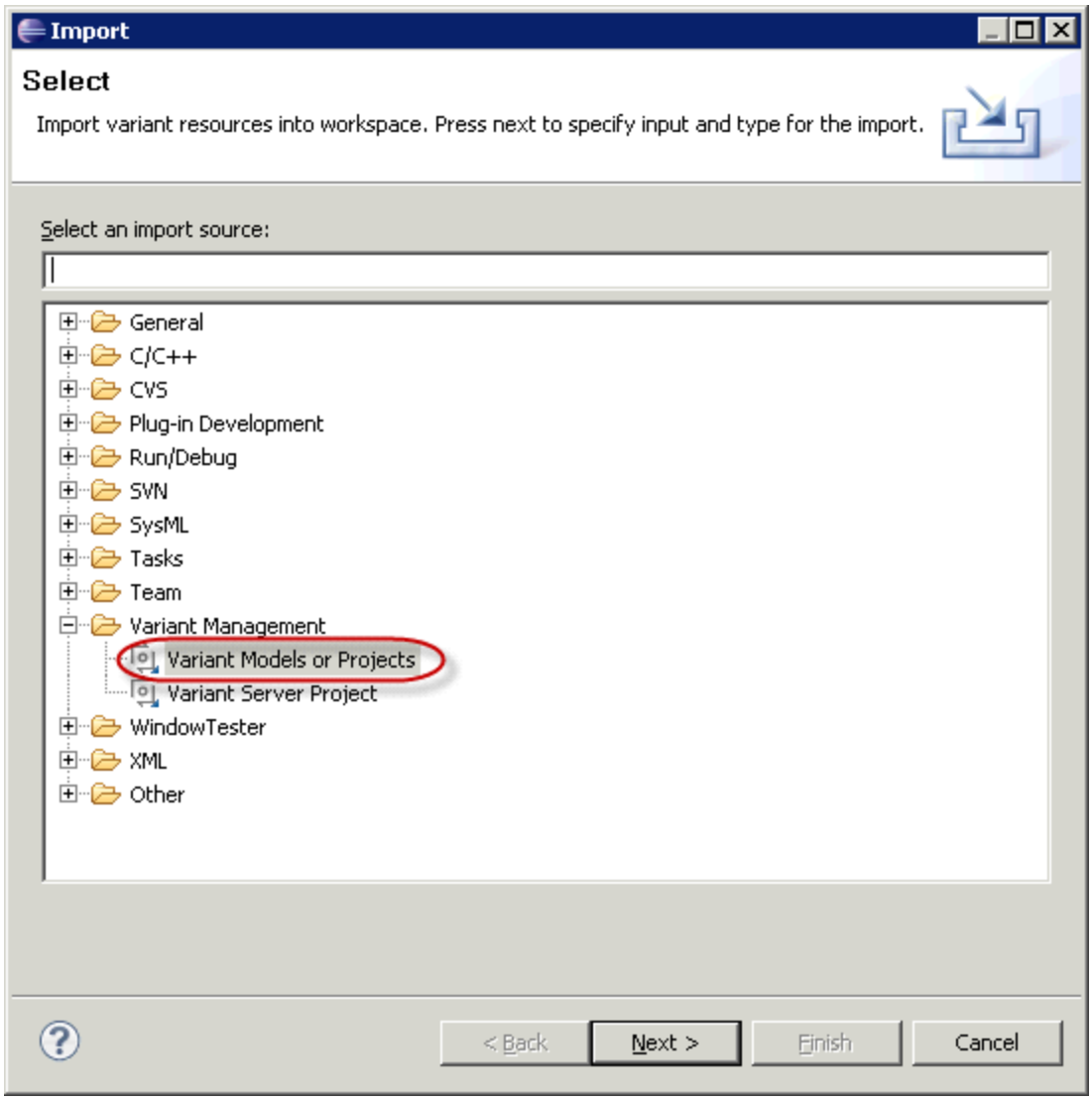
In pure::variants, the Import Wizard is used to import a variation point configuration into a variant model. To do this, select a configuration space in the “Variant Projects” view. The wizard can be opened via the context menu item “Import…”.

Figure 98. Importing a Variation Point Configuration



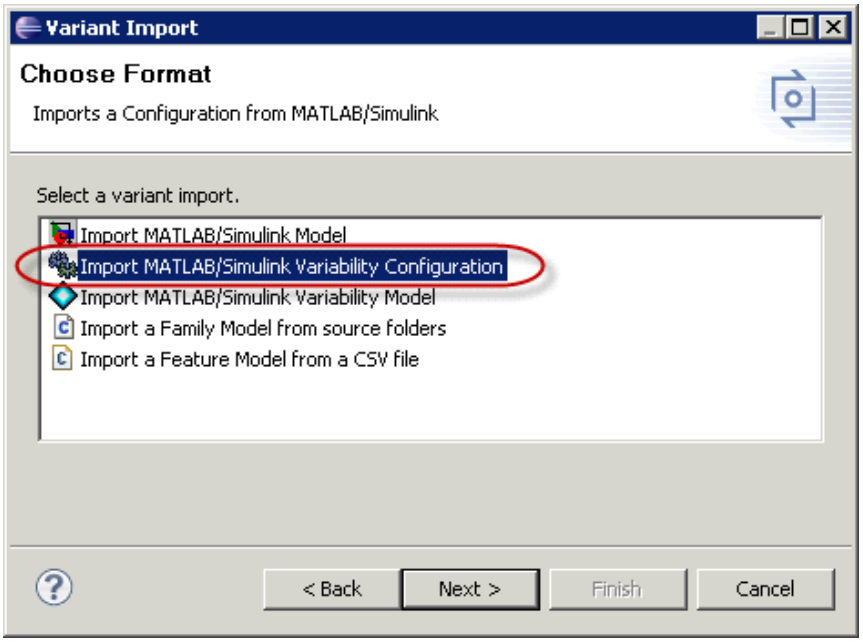
On the first page of the wizard that appears, select the entry “Variant Management Variant Models or Projects”.

Figure 99. Selecting Variant Model Import



The second page shows all currently possible imports to pure::variants. To import the variation point configuration to pure::variants, the entry “Import MATLAB/Simulink Variability Configuration” must be selected.

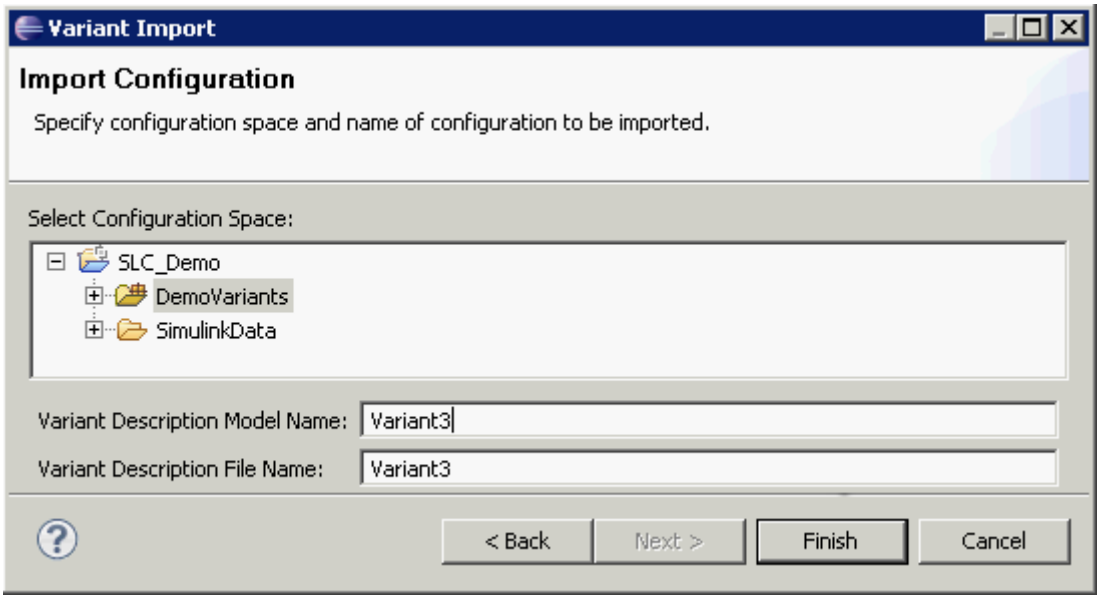
Figure 100. Selecting MATLAB/Simulink Variability Configuration Importer



On the next page, enter the name for the new variant model and optionally a deviating file name. By default, the same name as the model is used here. The configuration space is already selected as the target directory. If the new variant model should be created in another configuration space, this must be selected in the tree.

Note: A variant model must always be created within a configuration space.

Figure 101. Settings for a Configuration



After the wizard is finished, the new variant model is created and contains the variation point configuration that exists in MATLAB/Simulink. However, it does not contain any feature selection.

If all variations of a variation point in this variant model are marked as “Excluded” (), these are either not configured in MATLAB/Simulink and/or configured with a variation that does not exist in pure::variants or the entire variation point does not exist in MATLAB/Simulink.

Other Actions

The newly created variant model with the variation point configuration from MATLAB/Simulink can be used for comparison with a variation point configuration that already exists in pure::variants (see Section 6.9.2, “Activity: Comparing Variant Models”). If undesired differences are discovered, the desired result may be achieved by adjusting the feature selection in the existing variant model and/or adjusting the conditions for the automatic configuration of the variation points.

A valid feature selection can now be determined for the imported variation point configuration (see Section 7.4.1, “Activity: Determining a Valid Feature Selection”).

7.3. Refactoring Feature Models

In addition to creating or moving features, the modeling of feature models also includes deleting and changing them. Because features can be linked with variations via conditions, a refactoring is required for deleting and renaming (of the unique name). Otherwise such actions would result in invalid references in the conditions, which would have to be corrected manually.

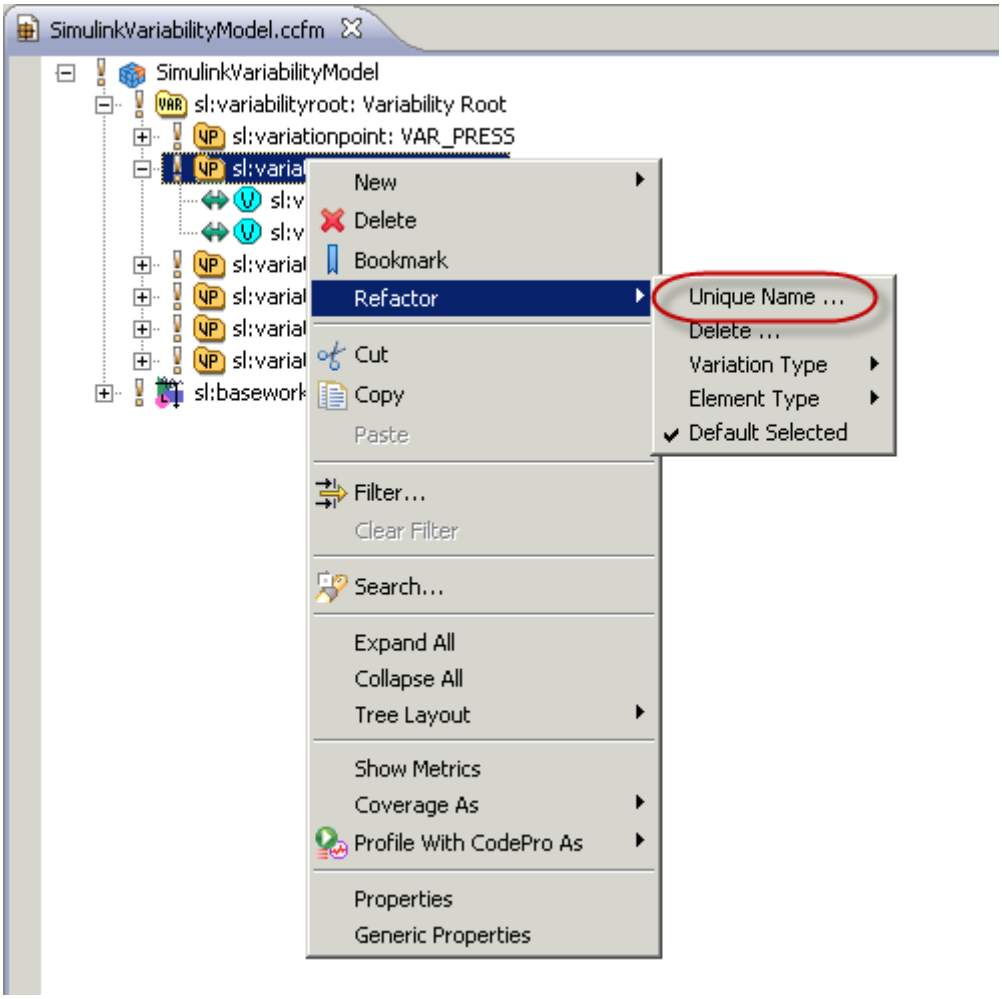
- Section 7.3.1, “Activity: Renaming Features”
- Section 7.3.2, “Activity: Deleting Features”

7.3.1. Activity: Renaming Features

Process

In pure::variants, the Refactoring Wizard is used to rename (the unique name of) a feature. To do this, select the feature to be renamed in the “Tree” viewer of the Feature Model Editor and open the wizard via the context menu item “Refactor -> Unique Name…”.

Figure 102. Renaming Features

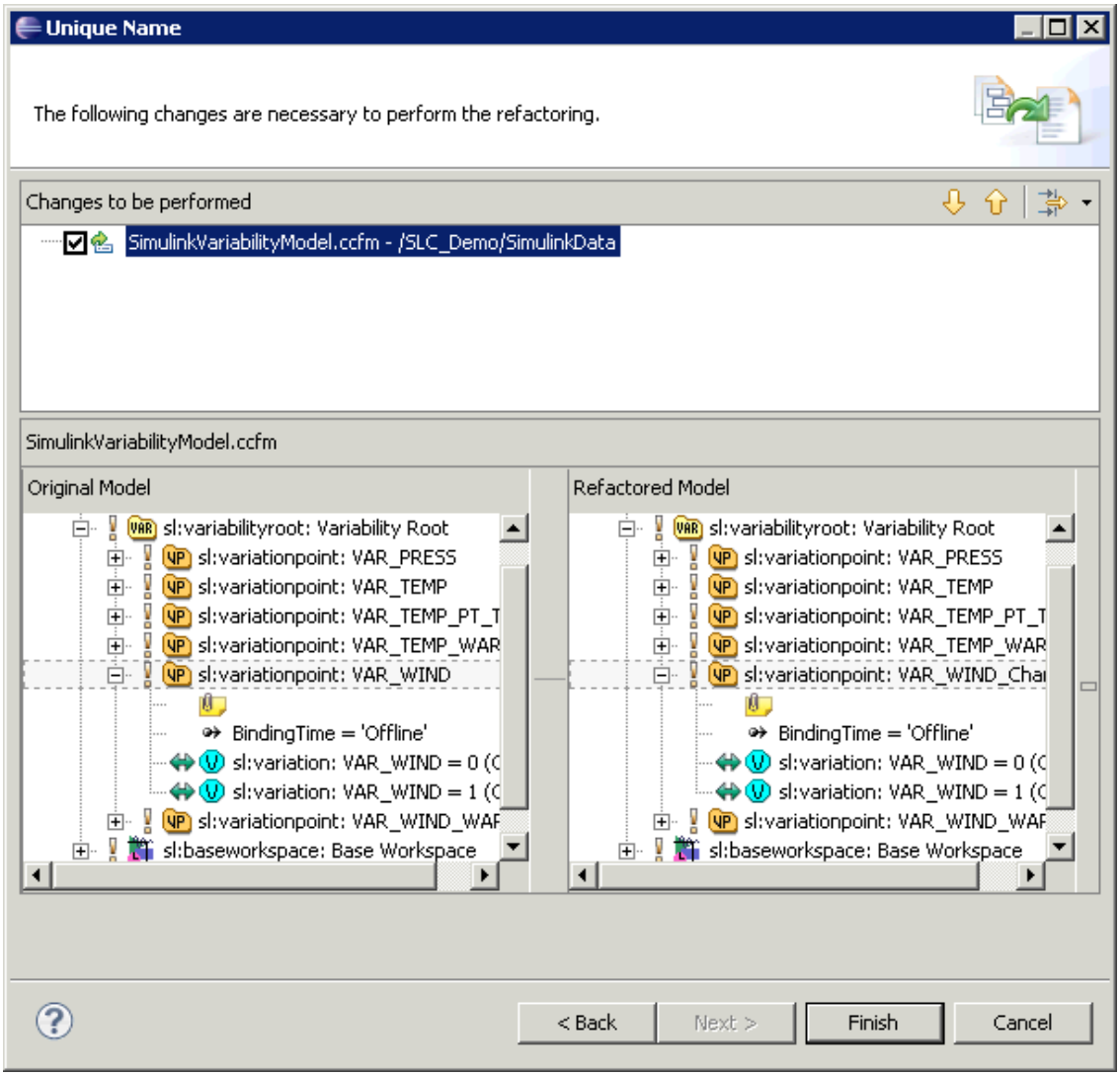


On the first page of the wizard, enter the new unique name in the corresponding text field. The preconfigured search scope, which permits a search for references to this feature in the current project and its referenced projects, can be changed. This can be restricted to the current project or expanded to the entire workspace of pure::variants.

The second page shows all models affected by the change and offers a preview of all necessary changes in these models. The models to be adjusted are shown in the top part, and the trees below compare the corresponding changes in detail. The left tree contains the original state and the right tree the changed state.

Changes can be deselected on a per-model basis by leaving models unselected.

Figure 103. Preview of resulting Changes



When the wizard is finished, the displayed changes are executed on a per-model basis and the feature is renamed in the feature model.

Note: Before renaming, all open models must be saved, which is also automatically supported by a corresponding confirmation dialog. After the renaming, all changed models are automatically saved.

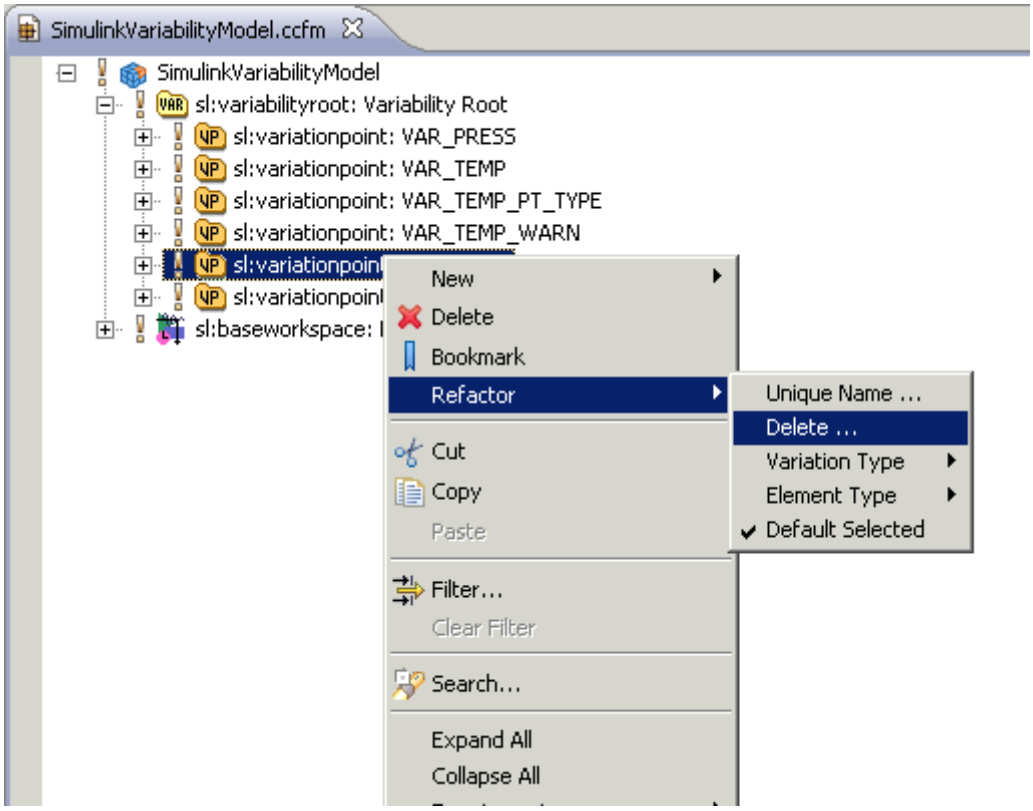
7.3.2. Activity: Deleting Features

Process

In pure::variants, the Refactoring Wizard is used to delete a feature. To do this, select the feature to be deleted in the “Tree” viewer of the Feature Model Editor and open the wizard via the context menu item “Refactor -> Delete…”.

Note: The context menu item “Delete” in the top level deletes the feature without a dependency analysis.

Figure 104. Deleting Features

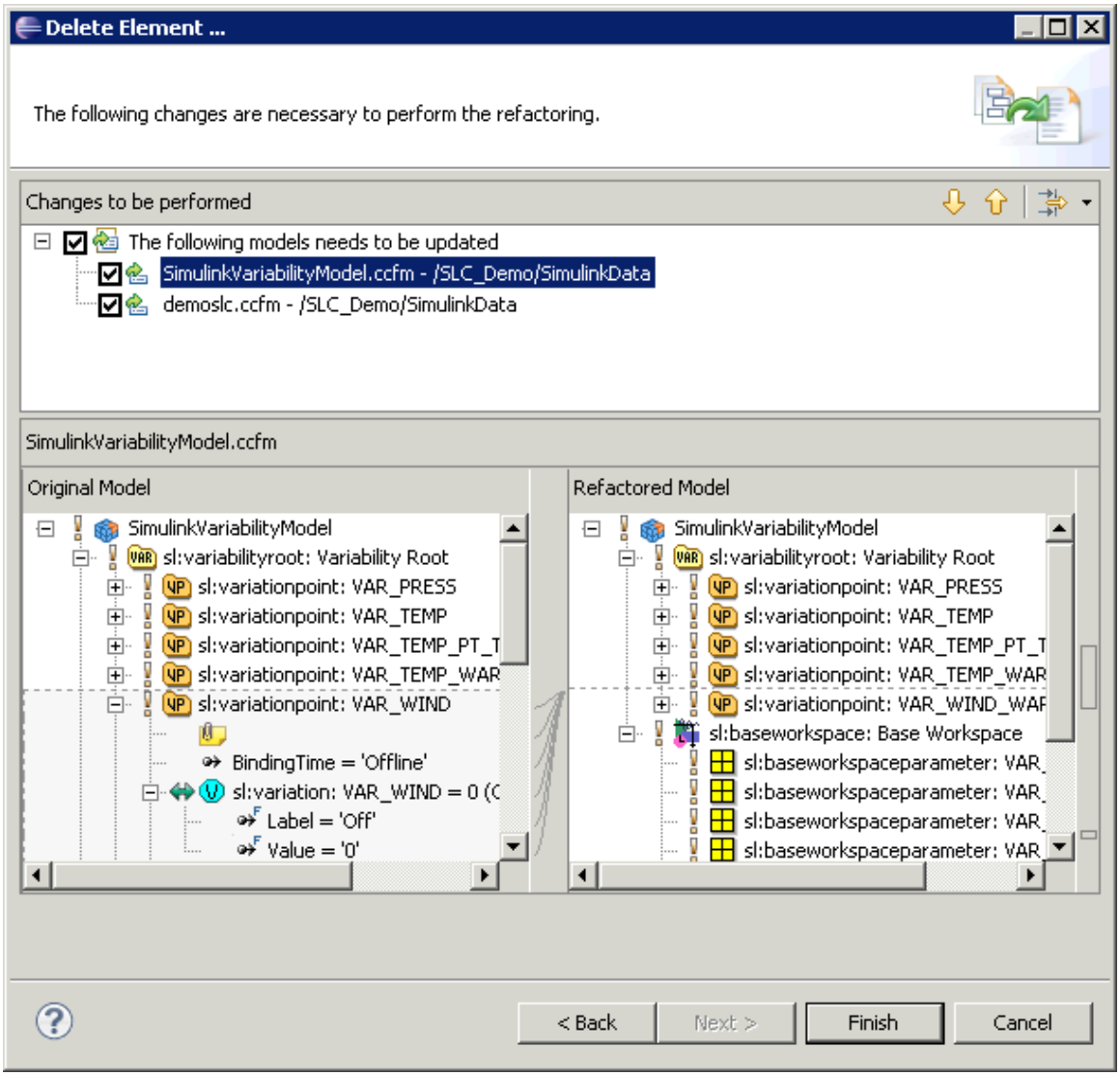


On the first page of the wizard, it is possible to change the preconfigured search scope, which permits a search for references to this feature in the current project and its referenced projects. This can be restricted to the current project or expanded to the entire workspace of pure::variants.

The second page shows all models affected by the change and offers a preview of all necessary changes in these models. The models to be adjusted are shown in the top part, and the trees below compare the corresponding changes in detail. The left tree contains the original state and the right tree the changed state.

Changes can be deselected on a per-model basis by leaving models unselected.

Figure 105. Preview of resulting Changes



When the wizard is finished, the displayed changes are executed on a per-model basis and the feature is removed from the feature model.

Note: Before deleting, all open models must be saved, which is also automatically supported by a corresponding confirmation dialog. After the deletion, all changed models are automatically saved.

Other Actions

During the deletion, all references to the deleted feature in the corresponding conditions are replaced by the expression “DELETED_BY_PV”. In the respective model editors, these conditions are highlighted with a marker because “DELETED_BY_PV” cannot be resolved. This makes it possible to quickly locate these conditions and adjust the semantics appropriately.

7.4. Modeling a Variant

7.4.1. Activity: Determining a Valid Feature Selection

Prerequisites & Preliminary Work

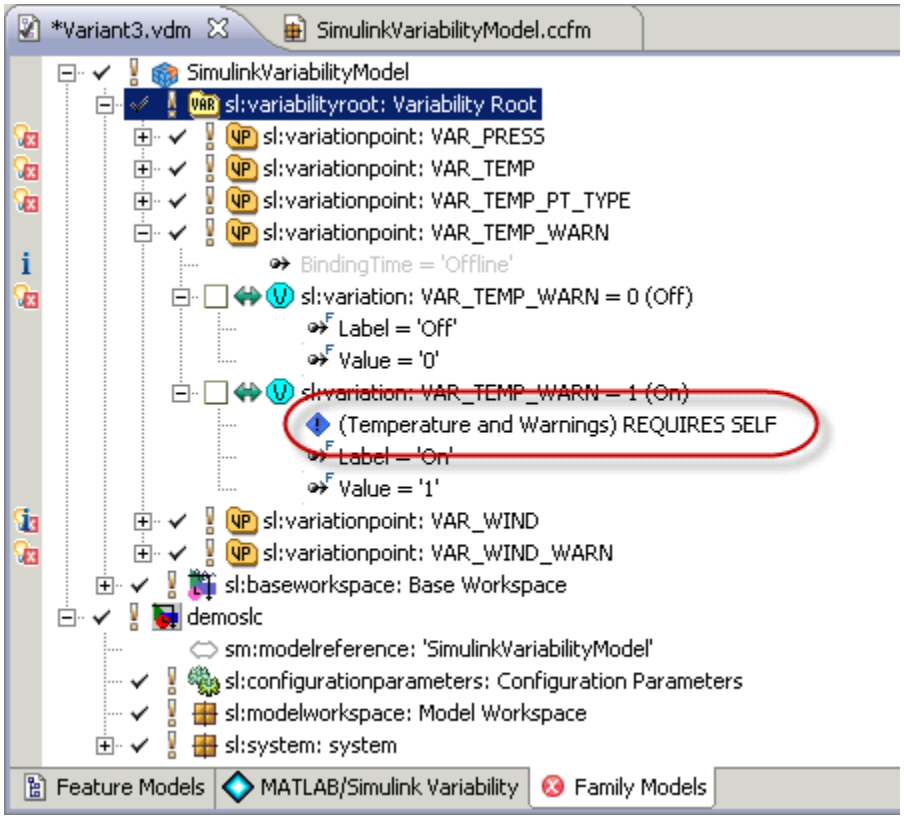
- The variability information associated with the variability model must be synchronized (see also Section 6.4.1, “Activity: Synchronizing to MATLAB/Simulink”).

- The variant model with the variation point configuration must be open.

Process

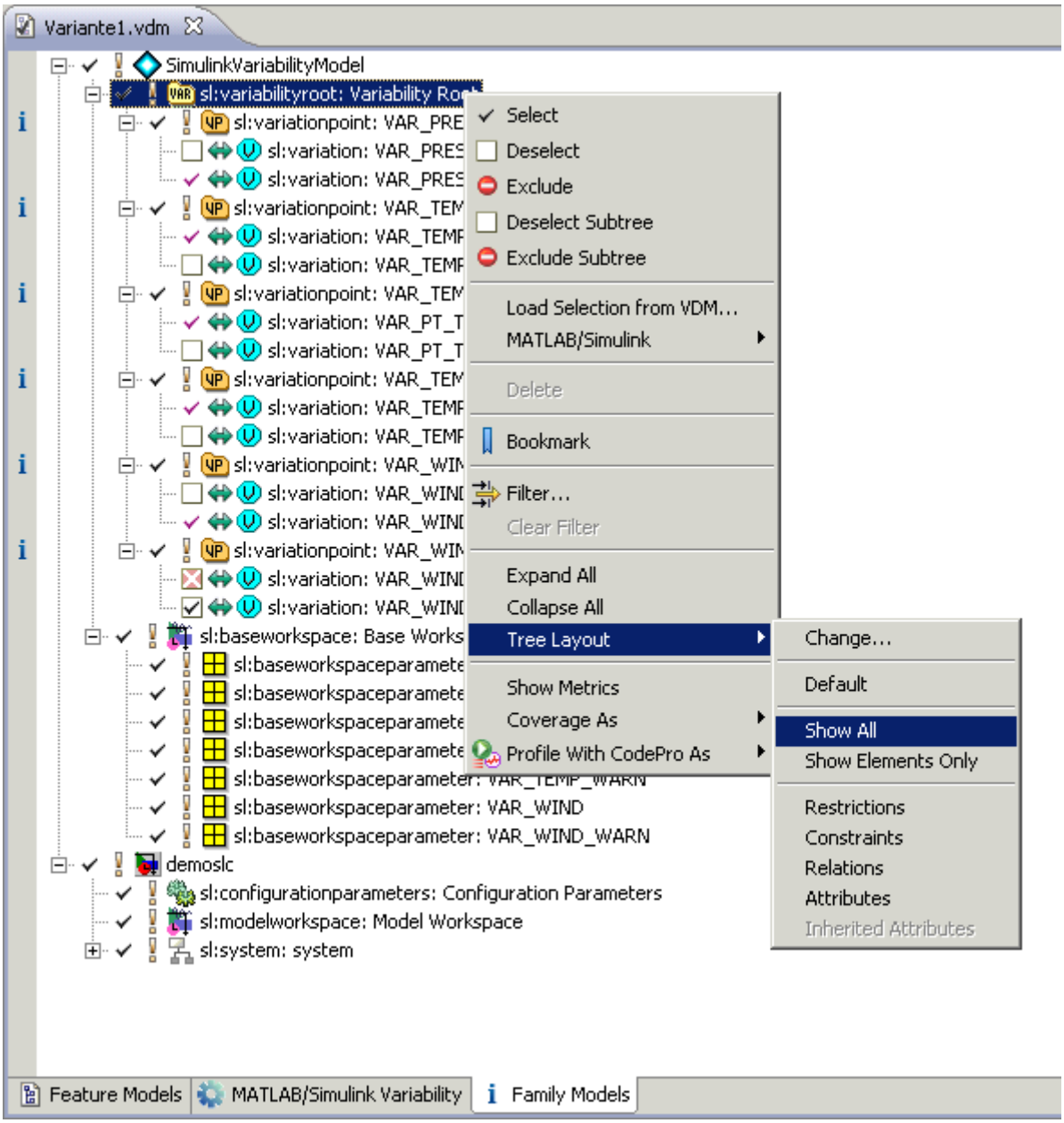
First, the “Family Models” viewer must be activated in the opened Variant Model Editor.

Figure 106. Variant Model Editor



A variation point must be selected here and the conditions of the variations analyzed step-by-step. The condition of a variation can be seen directly in the assignment under a variation in the tree. If it is not visible, the layout must be adjusted. This is done via the context menu item “Tree Layout -> Constraints”.

Figure 107. Tree Layout



Note: The following procedure only applies if at most one condition was defined per variation. This is supported only by the “Variation Point” viewer in the Family Model Editor.

a. Semi-automatic resolving

   i. The semi-automatic resolving requires that the “Extended Auto Resolver” be active (default). In some cases, this must be activated or reconfigured. More information on this can be found in the section “Evaluating Variant Descriptions Configuring the Auto Resolver”.

   ii. For every variation point, the condition of the variation selected during importing is also modeled as a constraint in the Feature Model Editor (directly at the root feature of the feature model).

   iii. The “Extended Auto Resolver” now attempts to resolve these constraints in the Variant Model Editor. If the resolving is successful, the calculated features can be marked as manually selected.

   Note: The “Extended Auto Resolver" cannot automatically resolve all constraints. In particular, this affects constraints that reference not only features but also comparisons of the attributes of an element, for example.

iv. Once no further resolving by the “Extended Auto Resolver” is possible and all calculated features have been marked as manually selected, the additional constraints should be removed in the Feature Model Editor.

b. Manual resolving

i. Select a variation with a condition that references only one feature or features linked with “AND”. In this case, all features contained here must already be or become selected. To do this, switch to the “Feature Models” viewer of the Variant Model Editor and manually select the corresponding features there. This must be repeated for all conditions with the specified structure.

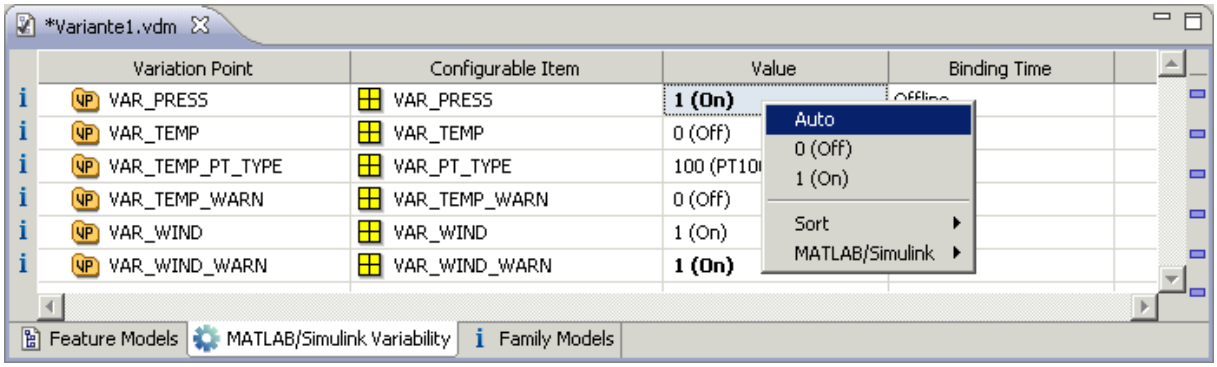
Note: If the features have already been added via automatic selection, they should still be manually selected via the context menu so that this automatic selection does not disappear due to the selection of additional features.

ii. Select a variation with a condition that references features linked with “OR”. Now just a single feature should be selected according to the procedure described above. This must be repeated for all conditions with the specified structure. If this is not possible without deselecting a previously selected feature, a different feature should be selected. If this cannot be accomplished for a condition, it may be necessary to select different features for previously processed conditions of other variations as well. If it is not possible to find a valid feature selection for a condition, this variation point configuration cannot be fulfilled without a rule change in the family model and/or feature model.

iii. For all other variations whose conditions do not correspond to the scheme described above, no general procedure for fulfilling the requirements can be given.

If a valid feature selection has been found, the “manual” selection of variations imported from MATLAB/Simulink cannot be cleared for all variation points. To do this, select all variation points in the “MATLAB/Simulink Variability” of the Variant Model Editor and set them to “Auto” via the context menu.

Figure 108. Enabling Feature-based Selection



7.4.2. Activity: Copying a Variant Model

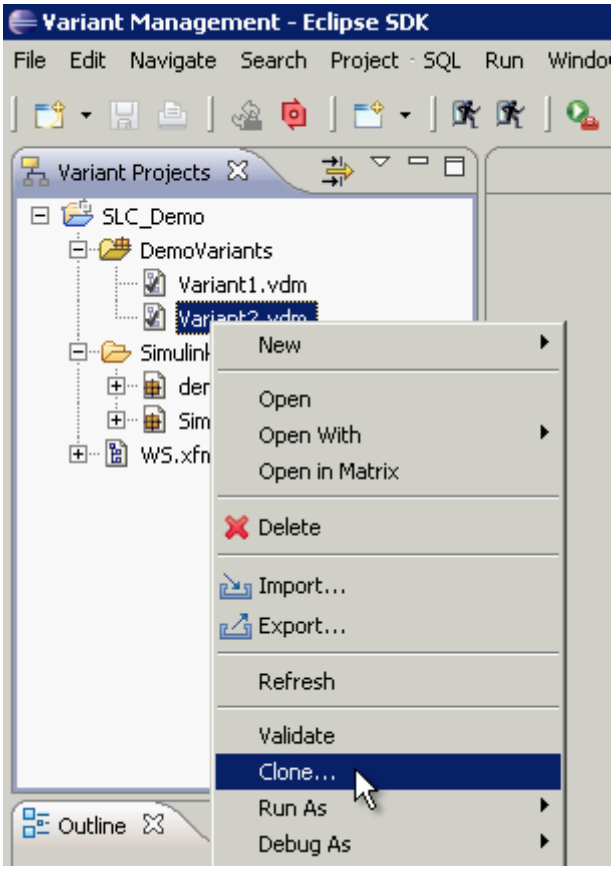
Prerequisites & Preliminary Work

- A variant model must exist in a configuration space.

Process

In pure::variants, model cloning is used to copy a variant model. Cloning means that all information except for IDs is copied. All IDs in the variant model are generated from scratch since they must be unique across model borders. The variant model to be copied must be selected in the “Variant Projects” view. The context menu item “Clone…” opens a wizard for copying the model.

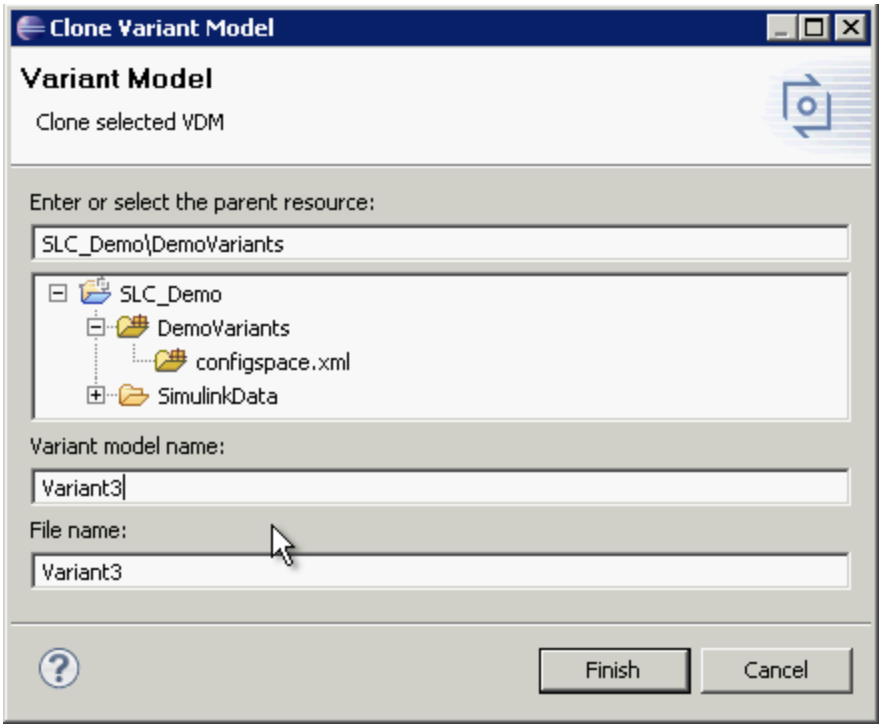
Figure 109. Copying a Variant Model



On the first page of the wizard that appears, enter the name for the copy of the original variant model and optionally a deviating file name. By default, the same name as the model is used here. The configuration space of the model to be copied is already selected as the target directory. If the new variant model should be created in another configuration space, this must be selected in the tree.

Note: A variant model must always be created within a configuration space.

Figure 110. Settings for a copied Variant Model



After the wizard is finished, the copy of the original variant model is created within the configuration space. The created variant model contains exactly the same feature selection and the same variation point configuration as the copied variant model.

7.4.3. Activity: Comparing Two Variant Models

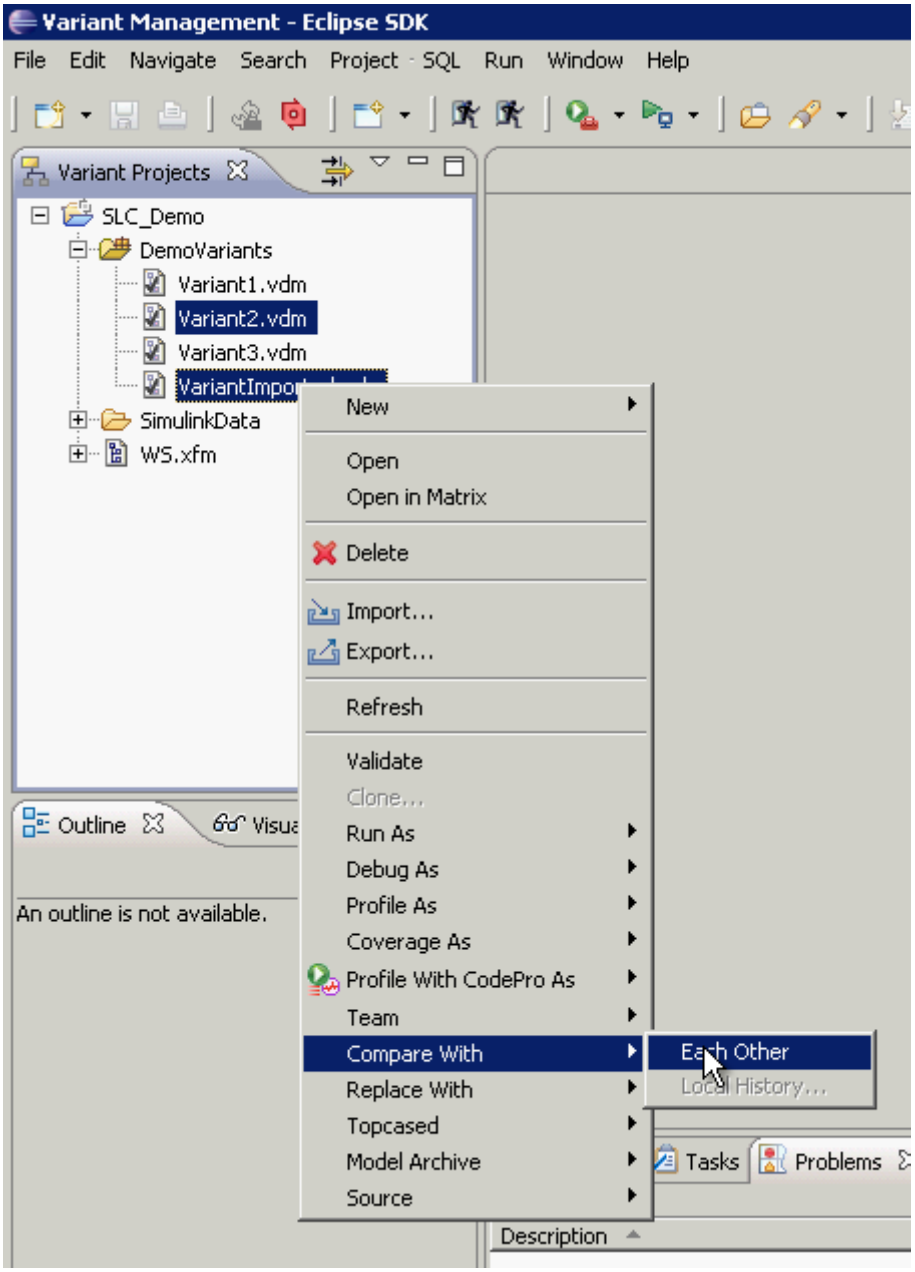
Prerequisites & Preliminary Work

- At least two variant models must exist in the same configuration space.

Process

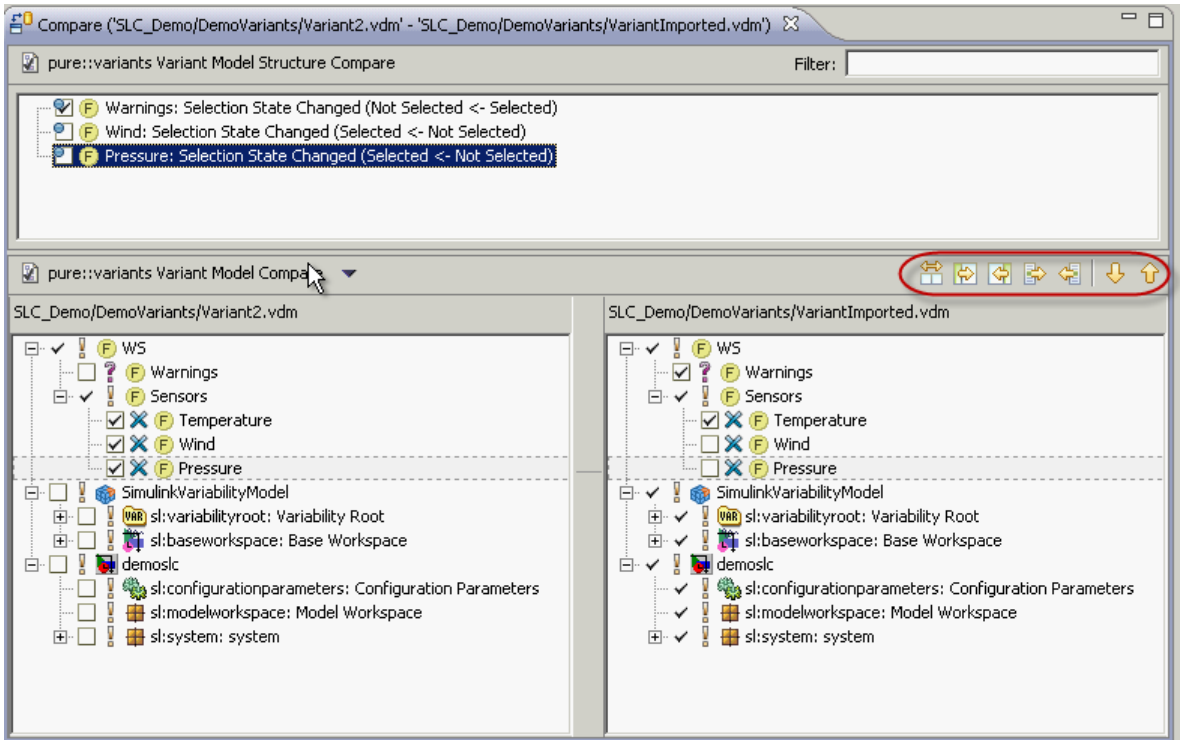
In pure::variants, the Compare Editor can be used to compare two variant models. To do this, select the two variant models to be compared in the “Variant Projects” view and open them in the Compare Editor via the context menu item “Compare With -> Each Other”.

Figure 111. Comparing two Variant Models



The top part shows all differences between the two variant models. Marking one of the differences causes the associated elements in the lower trees to be marked and the differences are displayed there more precisely.

Figure 112. Variant Model Compare Editor



Other Actions

The Compare Editor offers functions for not only listing the differences between two variant configurations but also for making changes. These can be transferred from one variant model to the other either individually or all at once. For this purpose, there is a toolbar immediately below the listed differences that offers various functions in the different directions (according to the arrows). In each case, a tool tip briefly describes the possible action.

Figure 113. Toolbar of Compare Editor

Note: The Compare Editor is also described in the pure::variants user documentation in section “6.6 Comparing Models”.

7.4.4. Activity: Creating a Configuration Space

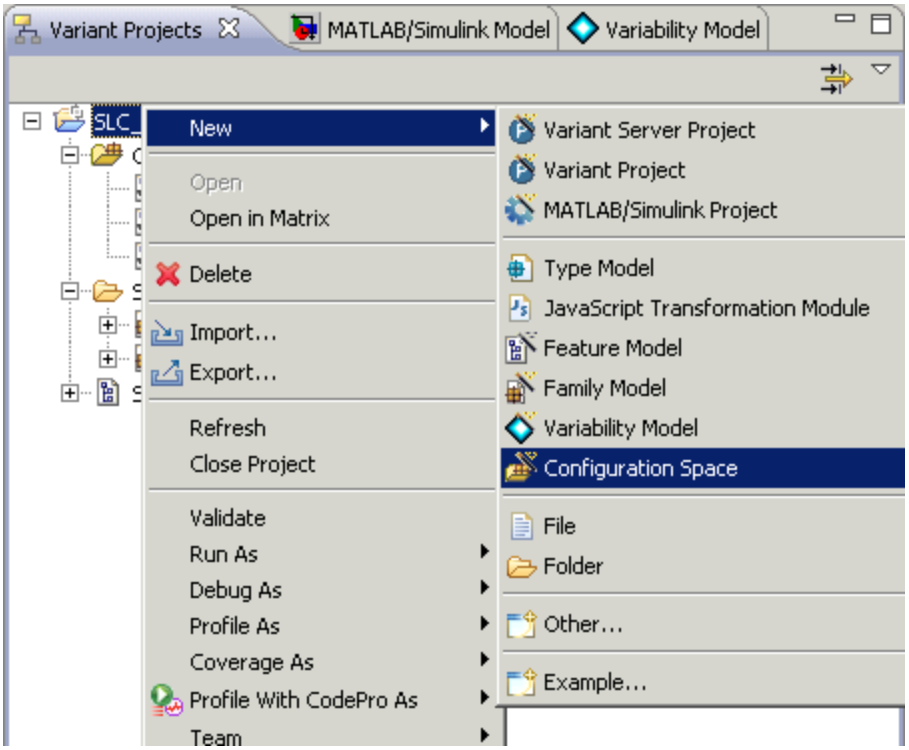
Prerequisites & Preliminary Work

- A pure::variants project must exist that contains at least one feature model and the variability model.

Process

Feature and family models that belong to a product and contribute to its variant configuration are collected together in pure::variants within a configuration space. This is done by selecting a project in the “Variant Projects” view and opening the wizard for creating a configuration space via the context menu item “New -> Configuration Space”.

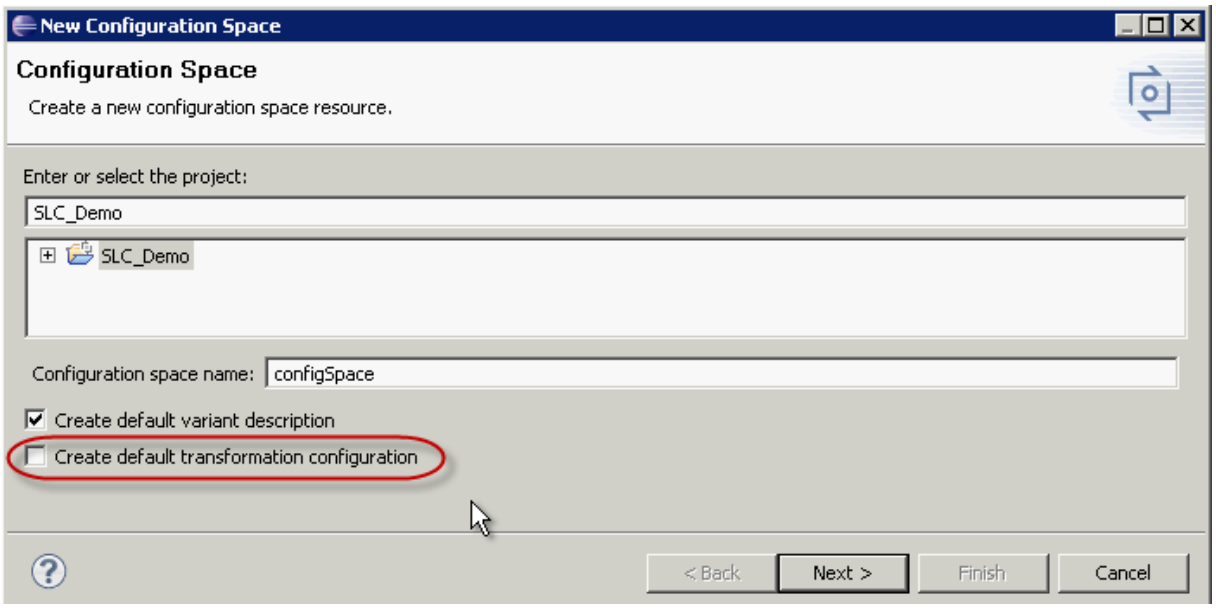
Figure 114. Creating a new Configuration Space



On the first page of the wizard that appears, enter the name for the new configuration space. The project is already selected as the target directory. If the new configuration space should be created in another project or subdirectory, this must be selected in the tree.

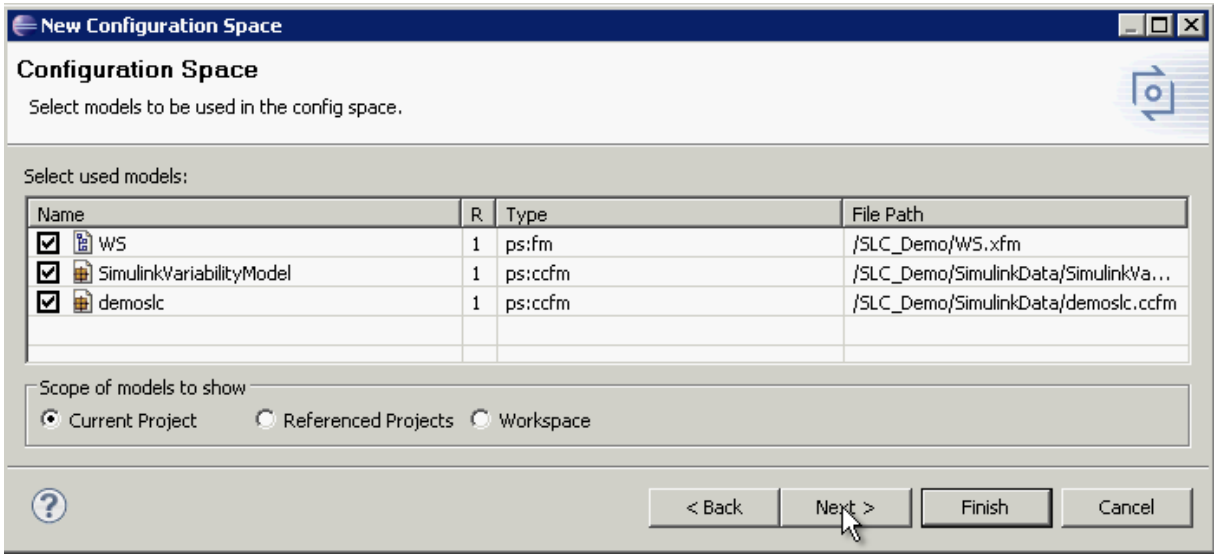
If the option “Create default variant description” remains selected, a variant model will automatically be created with the name entered for the configuration space. If you do not want a variant model to be created, deselect this option. The option “Create default transformation configuration” should be deselected because it creates a pure::variants default transformation configuration for the new configuration space that cannot be used for propagating or saving a variation point configuration.

Figure 115. Settings for a Configuration Space



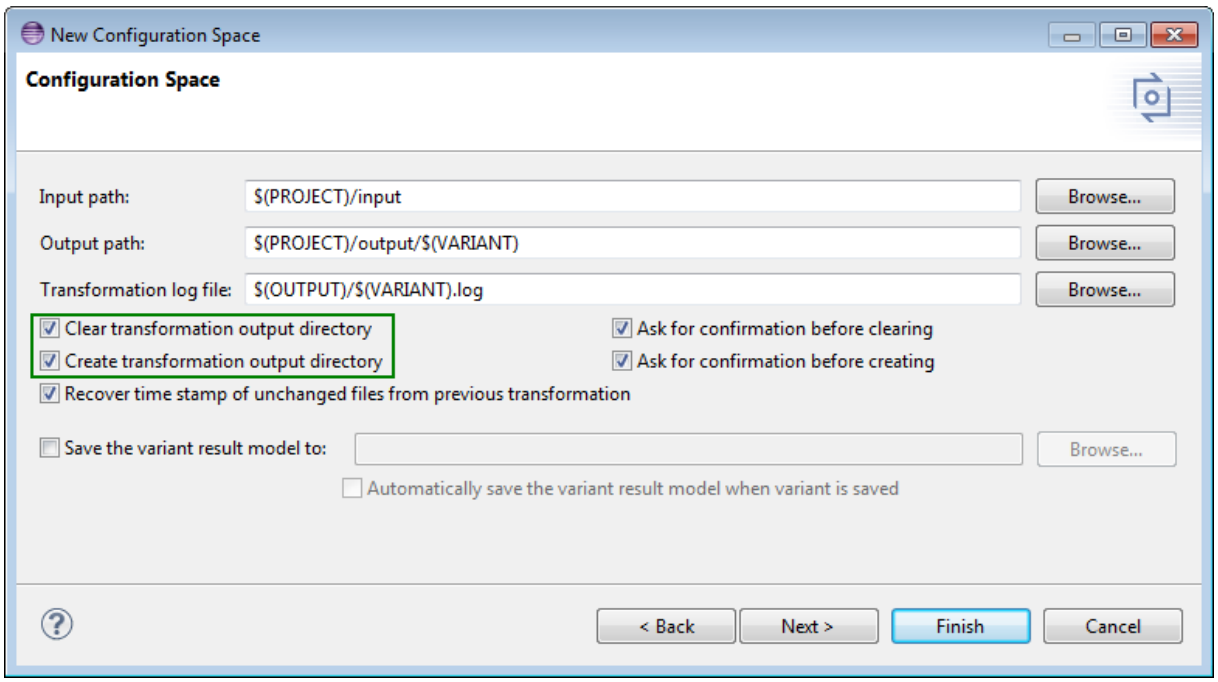
The second page allows the selection of models to be collected together in the new configuration space for variant configuration. For configuration of the variation points, it is necessary to select at least the variability model and the feature model whose features should be used to configure the variation points.

Figure 116. Input Models of a Configuration Space



The third page allows you to set the input and output directories of the transformations for the new configuration space. Some values are pre-initialized. However, it is recommended that you select the options “Clear transformation output directory” and “Create transformation output directory” to automate necessary steps in preparing the transformations.

Figure 117. Transformation Settings



The next page offers the opportunity to add transformation configurations (see Section 7.4.6, “Activity: Creating Transformation Configurations”) that should act on the configured models of the configuration space.

When the wizard is finished, the new configuration space is created and, if the corresponding option is selected on the first page, a new variant model is also created.

Note: The use of a configuration space is also described in the pure::variants user documentation in section “4.3 Using Configuration Spaces”.

Other Actions

Once the configuration space has been created, variant models can be created for configuration of the variation points (see Section 6.8.1, “Activity: Creating a Variation Point Configuration”).

7.4.5. Activity: Changing a Configuration Space

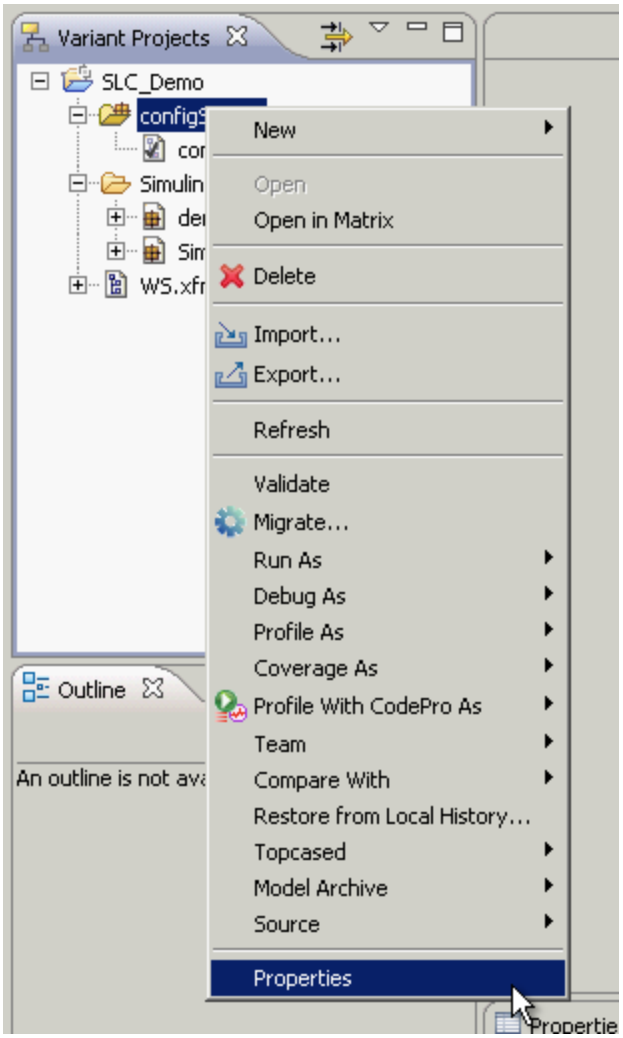
Prerequisites & Preliminary Work

- A pure::variants project must exist that contains at least one configuration space.

Process

In pure::variants, a configuration space is changed using its properties dialog. To do this, the configuration space must be selected in the “Variant Projects” view. The context menu item “Properties” opens the dialog for changing a configuration space.

Figure 118. Changing a Configuration Space



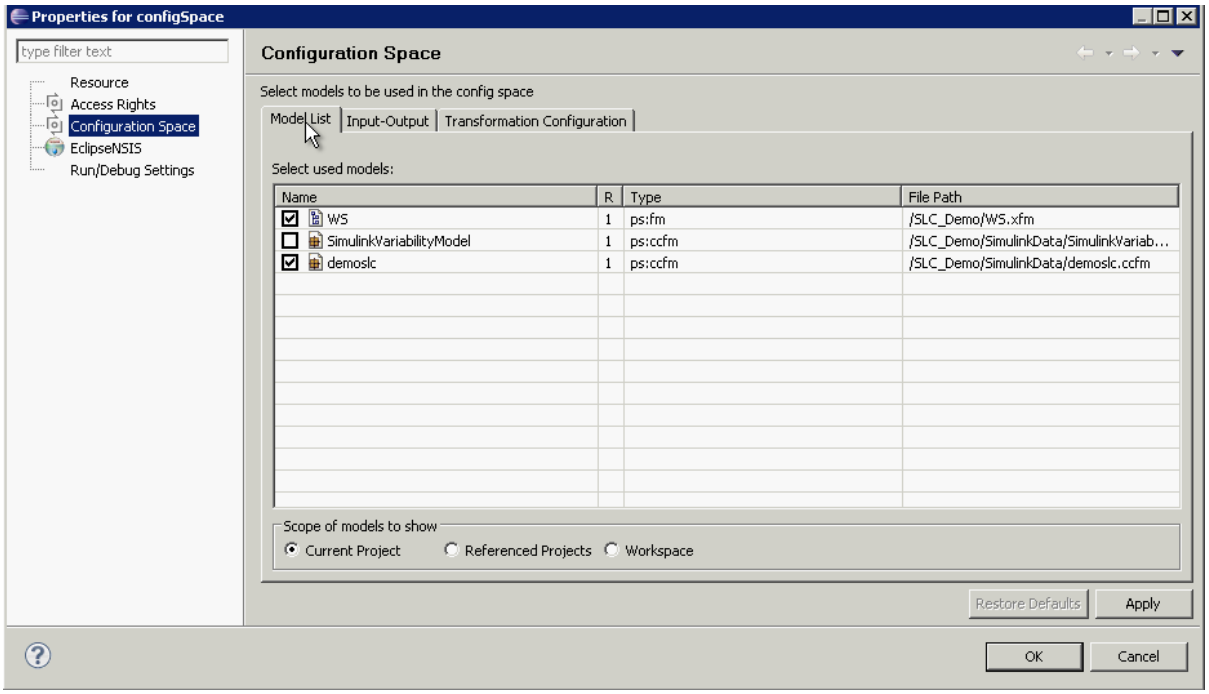
In the dialog that appears, “Configuration Space” must be selected on the left side and then “Model List” activated on the right side. The table now visible lists the models available for configuration. New models can be selected here and undesired models deselected. Unselected models will not be available later for configuration.

If you attempt to change the list of models while a variant model belonging to the configuration space is open, an error is displayed above the table.

Figure 119. Warning because of open Variant Models

The corresponding variant model can be closed with “Close”, and the list of models can then be edited.

Figure 120. Settings for a Configuration Space



After the dialog is closed, the list of models is updated and the configuration space is changed.

Other Actions

Adding or removing models that contribute or have contributed to a variant configuration can result in previously existing variant models in the configuration space becoming invalid. Adding a feature model can result in new dependencies that must be taken into consideration in the variant configuration. For this reason, all variant models must be checked by opening them and verifying that they are valid.

7.4.6. Activity: Creating Transformation Configurations

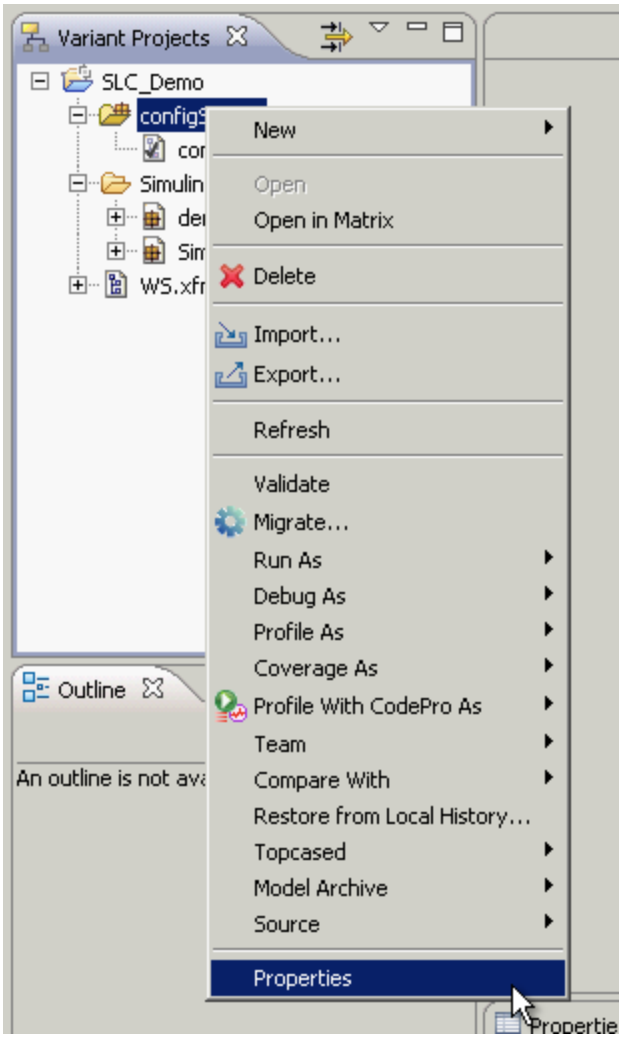
Prerequisites & Preliminary Work

- A pure::variants project must exist that contains at least one configuration space.

Process

In pure::variants, the properties dialog of a configuration space is used to create a transformation configuration for the configuration space. To do this, the configuration space must be selected in the “Variant Projects” view. The context menu item “Properties” opens the dialog for changing a configuration space.

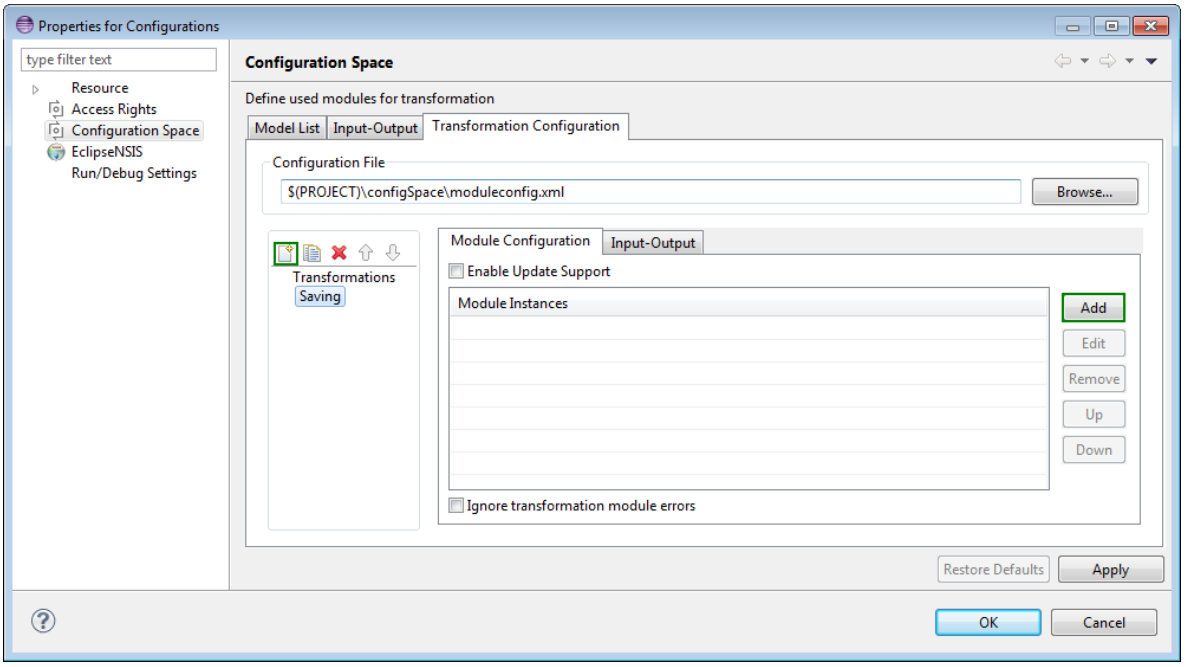
Figure 121. Opening the Properties of a Configuration Space



In the dialog that appears, “Configuration Space” must be selected on the left side and then “Transformation Configuration” activated on the right side.

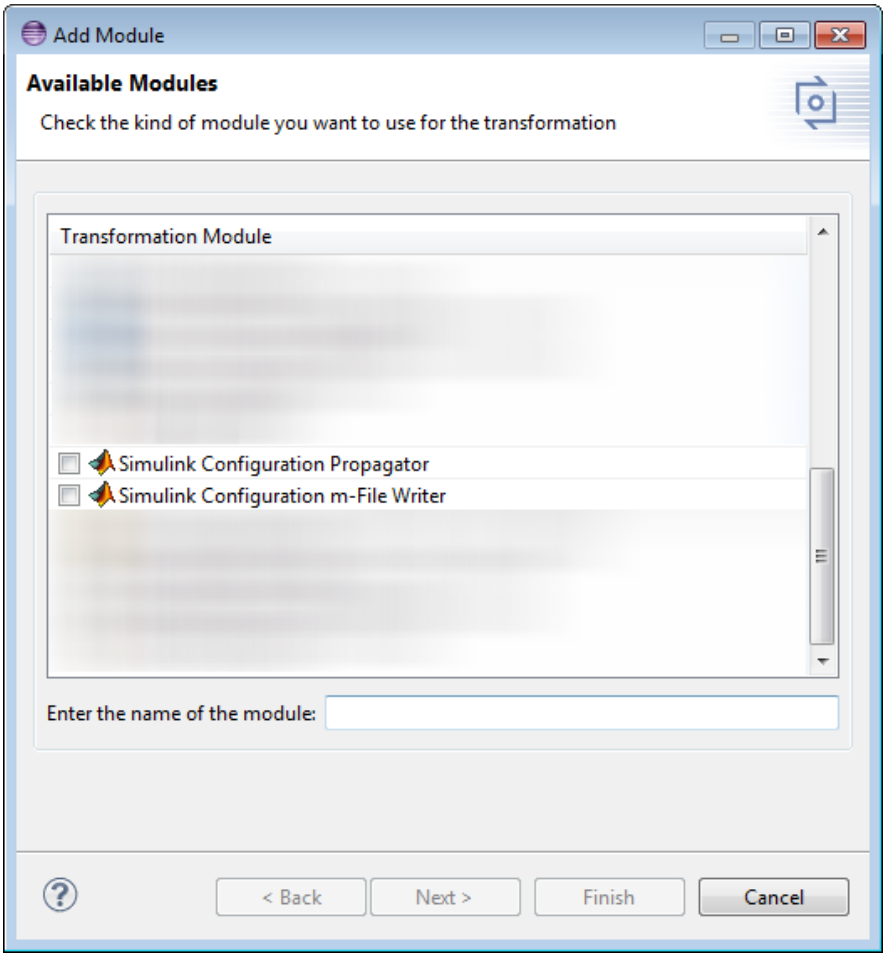
First, click on  to create a new transformation configuration, which automatically receives “Default” as a name. This can be changed by double-clicking on the name. Then add a new module that carries out the actual transformation using “Add”.

Figure 122. Creating a new Transformation Configuration



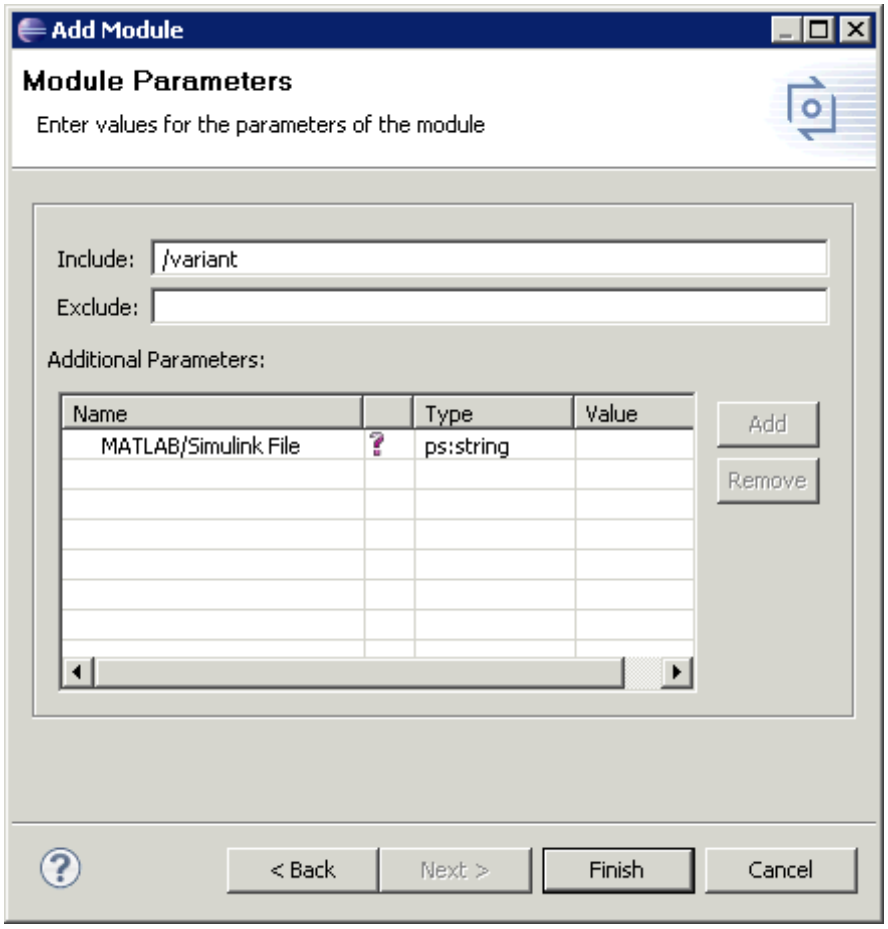
On the first page of the wizard that appears, all available modules are listed. Exactly one of these must be selected. To propagate a variation point configuration to MATLAB/Simulink, select the module “Simulink Configuration Propagator”, and to save a variation point configuration in a MATLAB script, select the module “Simulink Configuration m-File Writer”. A name must still be entered for the selected module.

Figure 123. Selecting a Transformation Module



The second page lists existing module-specific parameters. All mandatory parameters (!) must be assigned values. Values may be assigned to the optional parameters.

Figure 124. Settings for a Transformation Module



When the wizard is finished, the configured module is created and becomes visible in the module list of the transformation configuration. Subsequent changes to a module can be initiated either with a double-click on the name of the module or with “Edit”. The wizard opens again for editing the module.

After the dialog is closed, the list of models is updated and the configuration space is changed.

When the dialog is finished, the configuration space is created and the new transformation configuration is available for every variant model in this configuration space.

Note: The creation of a transformation configuration is also described in the pure::variants user documentation in section “6.3 Transforming Variants”.

Other Actions

The created transformation configuration can be used for propagating (see Section 6.8.3, “Activity: Propagating a Variation Point Configuration”) or saving (see Section 6.8.4, “Activity: Saving a Variation Point Configuration”) a variation point configuration.

7.5. Data Dictionary Variable Classes

7.5.1. Activity: Importing Variable Classes

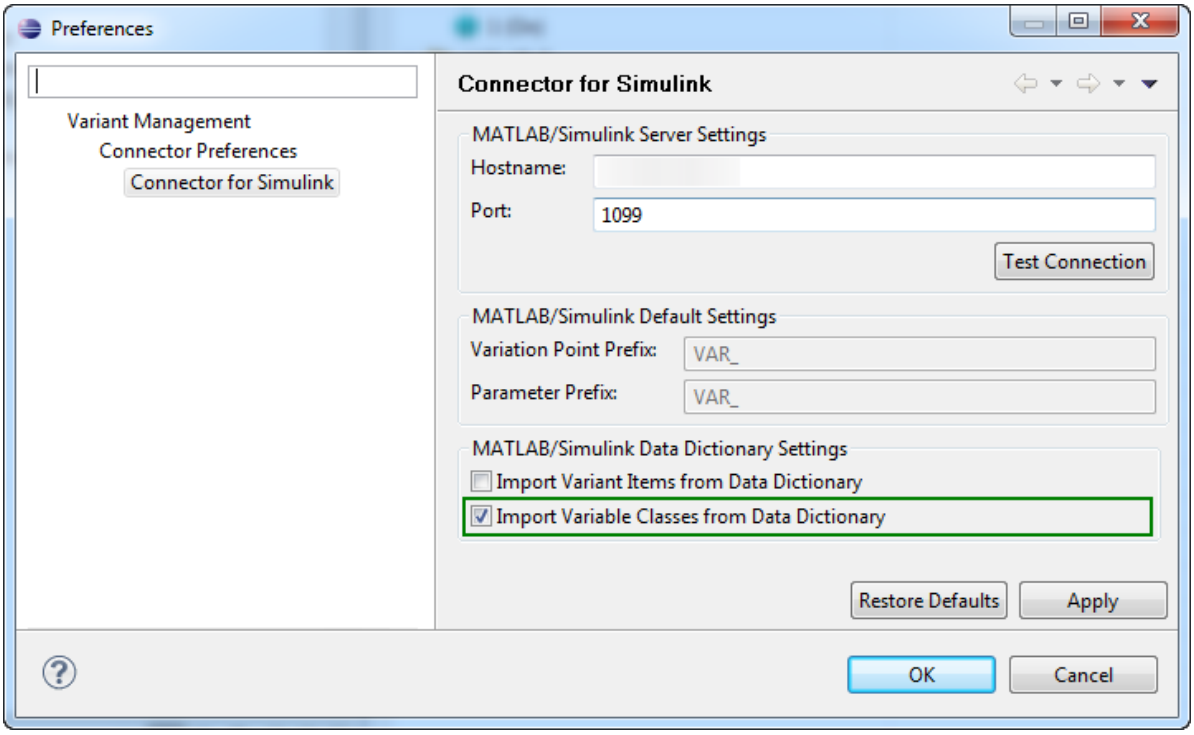
Prerequisites & Preliminary Work

- The Data Dictionary associated with the Simulink model must be open.

Process

Importing variable classes is part of importing variation points (see Section 6.5.1, “Activity: Importing Variation Points”). But importing variable classes has to be enabled explicitly in order to be handled while import. For this purpose the appropriate option has to be chossen in the preferneces.

Figure 125. Enable Import of Variable Classes



Other Actions

- Section 7.5.2, “Activity: Adding a Variable Class”

7.5.2. Activity: Adding a Variable Class

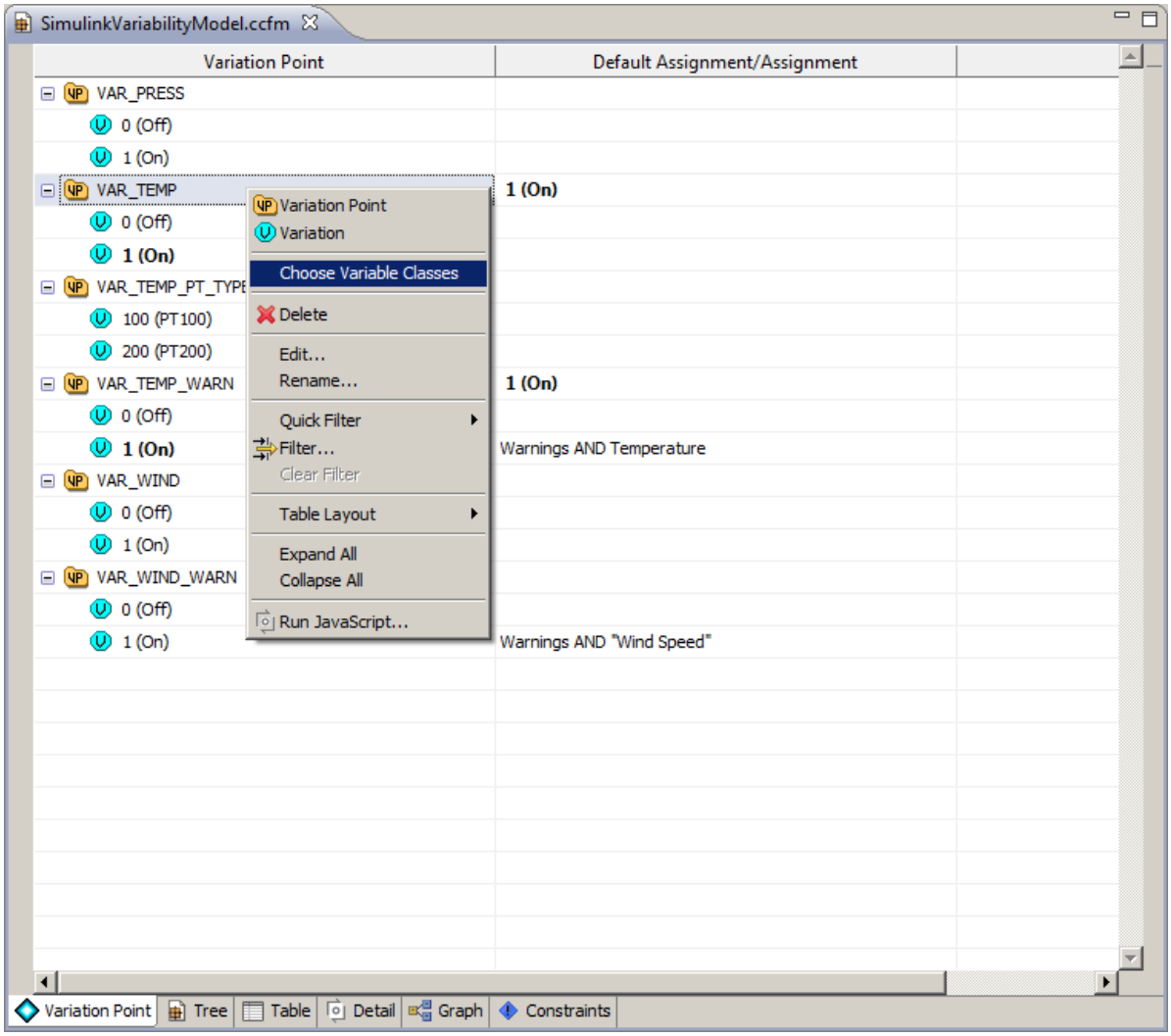
Prerequisites & Preliminary Work

- The variability model must be opened.
- The variation point for which variable classes should be added must exist in the variability model.
- Variable classes must not exist in this variation point.

Process

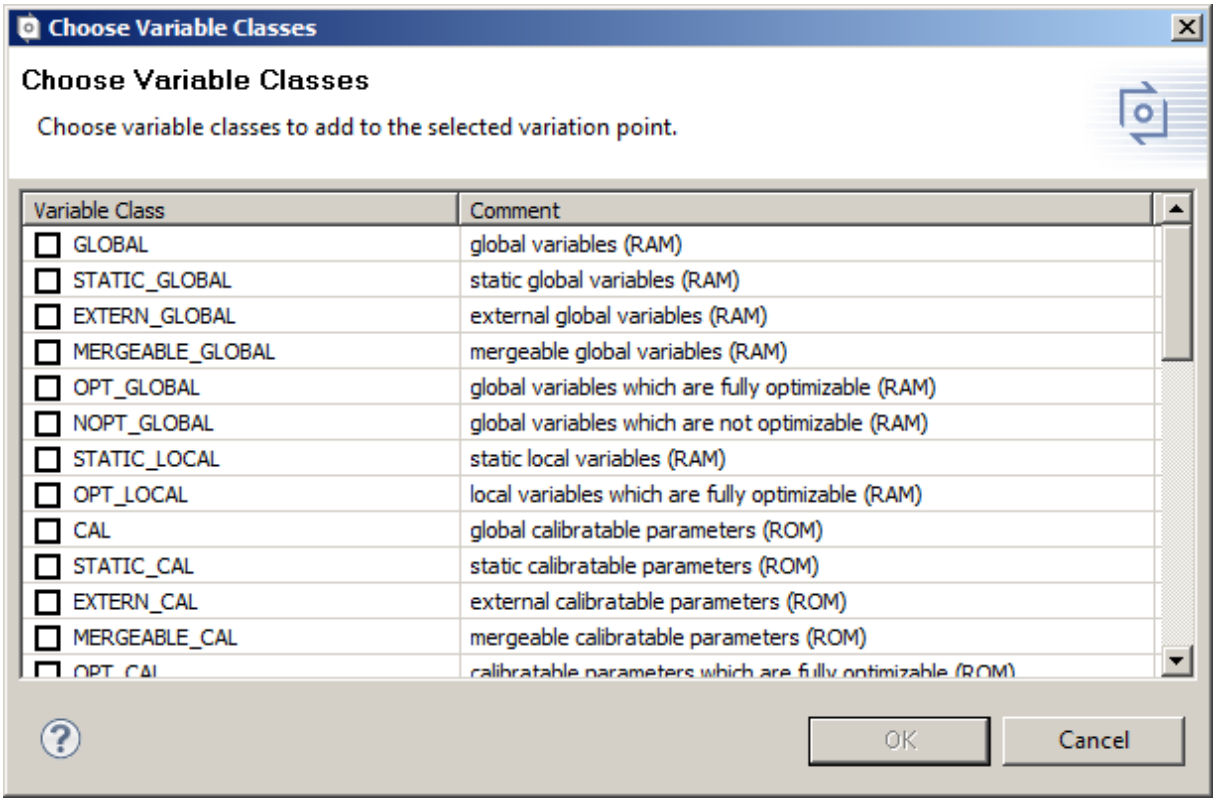
The “Variation Point” viewer of the Variability Model Editor is used for adding a variable class to variation points in pure::variants. In this viewer, select the variation point for which variable classes should e added. The context menu item “Choose Variable Classes” of the selcted variation point opens a dialog for chossing variable classes.

Figure 126. Choosing Variable Classes



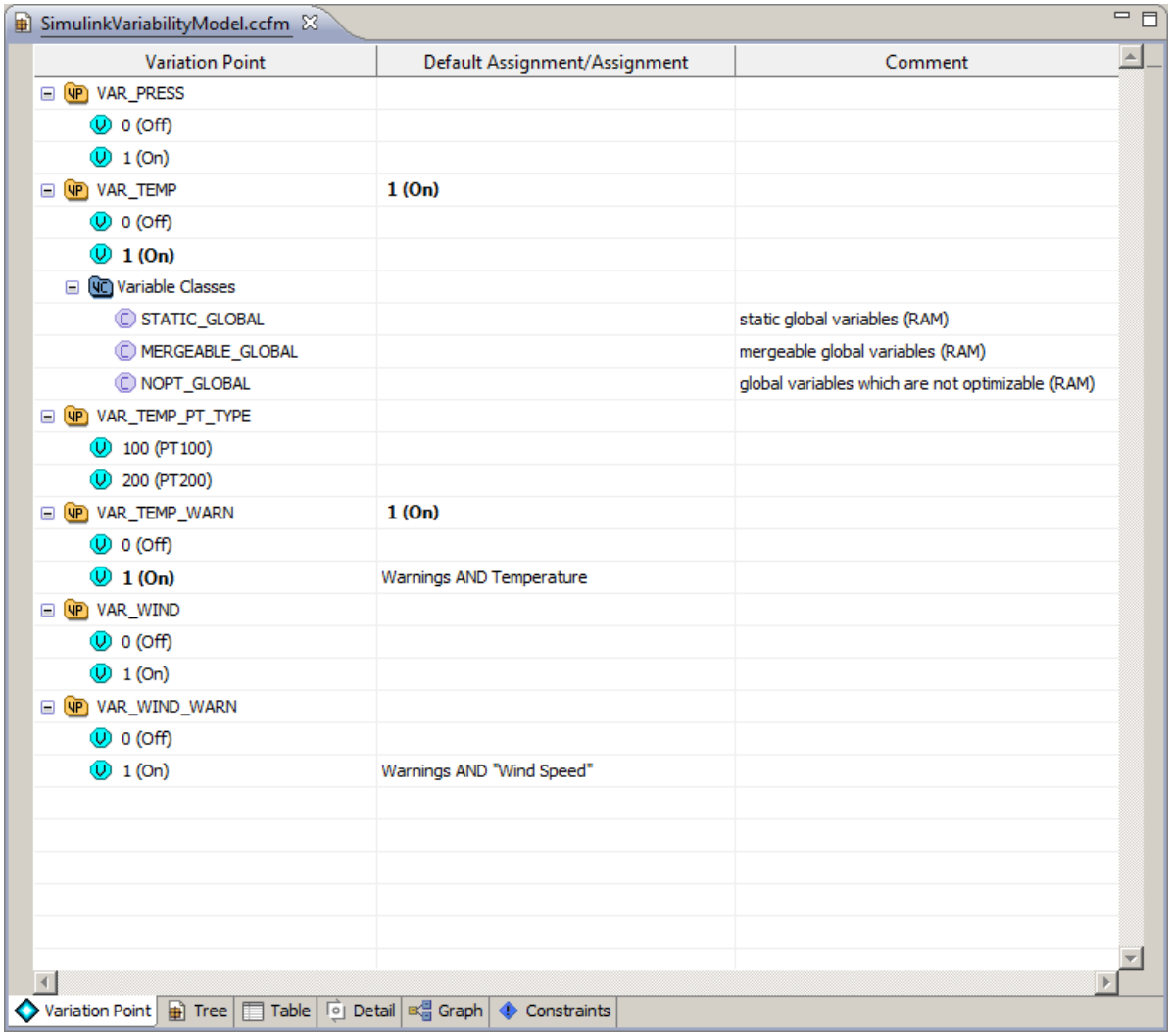
The opened dialog lists all variable classes. Variable class to be added to the variation point has to be marked with a tick in front. A multi-selection of several variable classes is possible.

Figure 127. Choose Variable Classes Dialog



After the dialog is finished, the marked variable classes are added to the selected variation point and shown in the “Variation Point” viewer of the Variability Model Editor.

Figure 128. Added Variable Classes



One of the added variable classes can be defined as „Default Assignment“, but variable classes can also be linked with features. The “Variation Point” viewer of the Variability Model Editor is used for linking a variable class with one or more features in pure::variants. In the viewer, select the variable class for which an assignment should be created. The context menu item “Assignment” of the selected variation opens a wizard for creating an assignment. (siehe Section 6.7, “Modeling Dependencies for Variations”).

Other Actions

The added variable classes can now be used directly for modeling dependencies and for variant configuration in pure::variants.

7.5.3. Activity: Changing Variable Classes

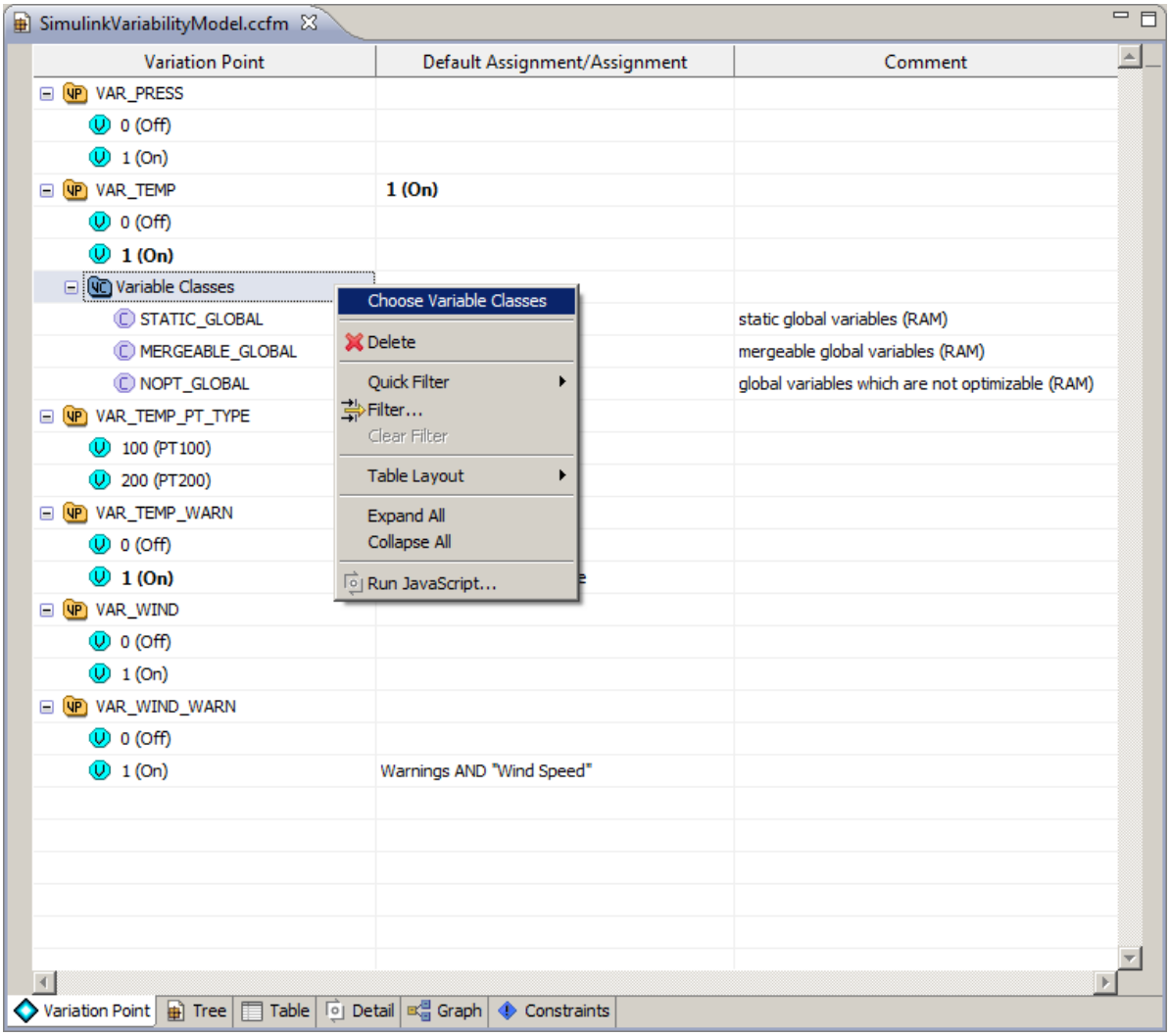
Prerequisites & Preliminary Work

- The variability model must be open.
- At least one variation point to be changed must exist in this variability model.
- Variable classes exists in this variation point.

Process

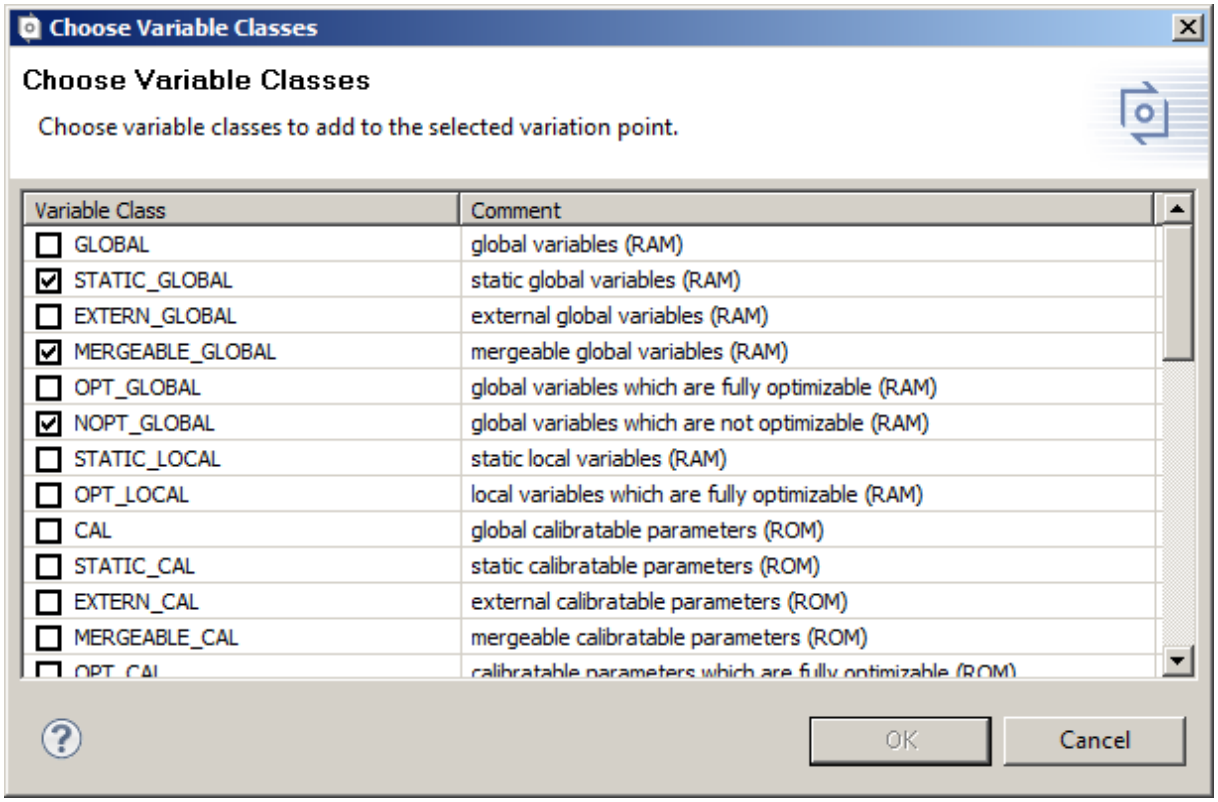
The “Variation Point” viewer of the Variability Model Editor is used for changing a variable classes of a variation point in pure::variants. Select this variation point or the “Variable Classes” object in the viewer. The context menu item “Choose Variable Classes” opens a dialog for changing variable classes. This dialog is initialized with the current variable classes of the variation point.

Figure 129. Choosing Variable Classes



The opened dialog lists all variable classes and marks also already added variable classes. Variable classes to be added to the variation point has to be marked with a tick in front. A multi-selection is possible.

Figure 130. Choose Variable Classes Dialog



After the dialog is finished, additionally marked variable classes are added to the selected variation point and shown in the “Variation Point” viewer of the Variability Model Editor(see Section 7.5.2, “Activity: Adding a Variable Class”).

Deselected existing variable classes are removed from the selected variation point. A Refactoring Wizard with preview appears.

Other Actions

The changed variable classes can now be used directly for modeling dependencies and for variant configuration in pure::variants.

7.5.4. Activity: Deleting a Variable Class

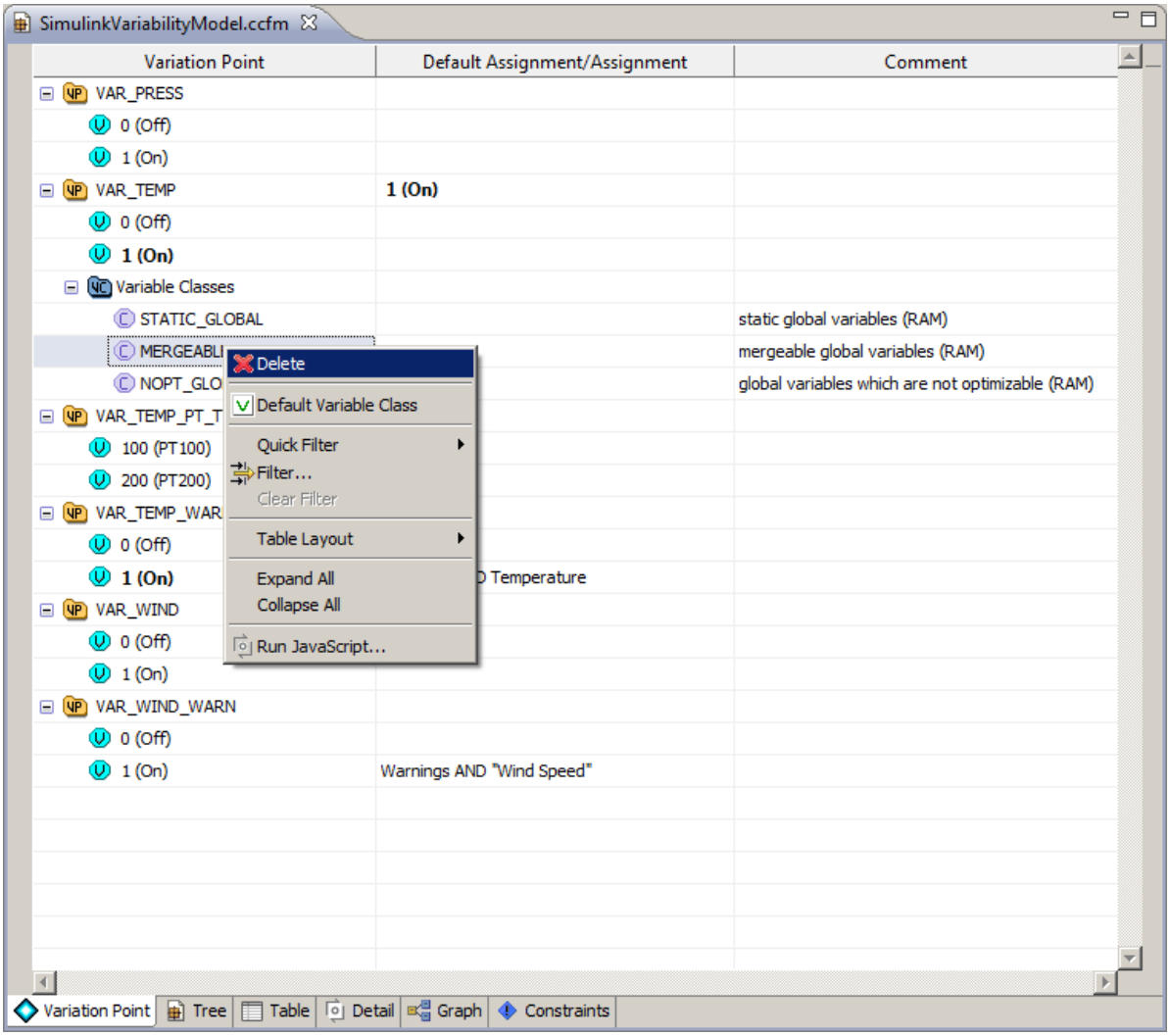
Prerequisites & Preliminary Work

- The variability model must be open.
- At least one variation point to be changed must exist in this variability model.
- The variabale class to be deleted must exist in this variation point.

Activity

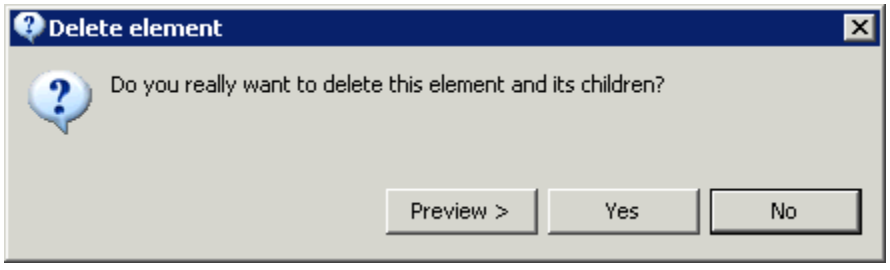
The “Variation Point” viewer of the Variability Model Editor is used for deleting a variable class in pure::variants. Select the variable class in the viewer. The context menu item “Delete” opens a dialog for deleting this variable class as well as all its references.

Figure 131. Deleting a Variable Class



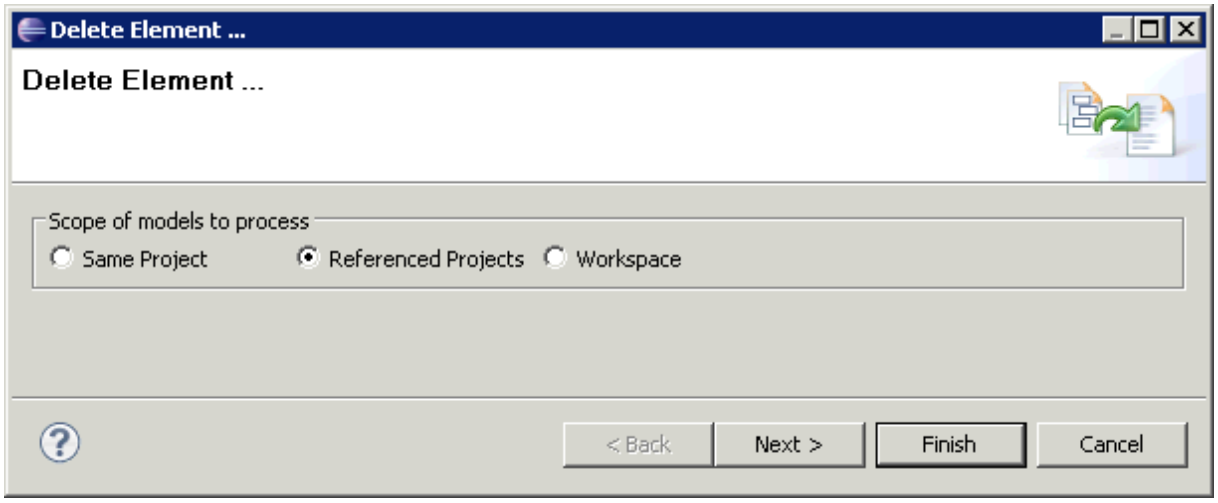
The dialog that opens allows immediate deletion of the variable class with “Yes”. It also offers a preview of the changes with “Preview >”.

Figure 132. Confirmation of the Deletion of a Variable Class



If a preview of the changes is selected, a Refactoring Wizard appears. On the first page of the wizard, it is possible to change the preconfigured search scope, which permits a search for references to this variabel class in the current project and its referenced projects. This can be restricted to the current project or expanded to the entire workspace of pure::variants.

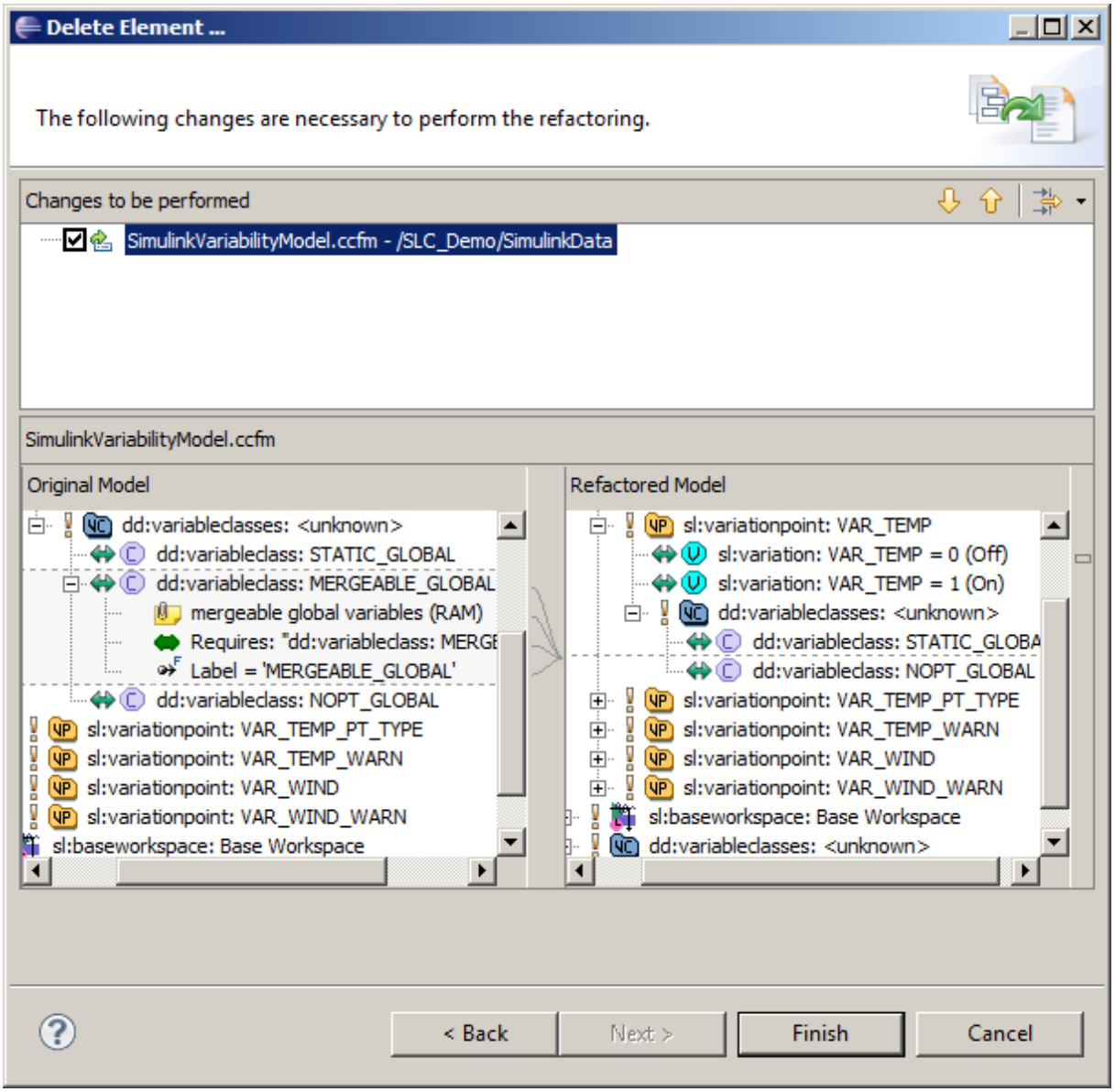
Figure 133. Search Scope for References of a Variable Class to be deleted



The second page shows all models affected by the change and offers a preview of all necessary changes in these models. The models to be adjusted are shown in the top part, and the trees below compare the corresponding changes in detail. The left tree contains the original state and the right tree the changed state.

Changes can be deselected on a per-model basis in the top part by leaving models unselected.

Figure 134. Preview of resulting Changes



When the wizard is finished, the displayed changes are executed on a per-model basis and the variable class is removed from the variability model.

Note: Before deleting, all open models must be saved, which is also automatically supported by a corresponding confirmation dialog. After deleting the variable class, all changed models are automatically saved.

Note: Deleting the last Variable Class result salso in deleting the “Variable Classes” object at the selected variation point.

7.5.5. Activity: Modeling Dependencies for Variable Classes

Modeling dependencies for variable classes is the same procedure as modeling dependencies for variations. The activity for variations in Section 6.7, “Modeling Dependencies for Variations” is applicable to variable classes.