1. Introduction

1.1. What is pure::variants - Variability Exchange Language?

The pure::variants - Variability Exchange Language (VEL) feature enables the user to exchange variability information as description and as configuration with any engineering tool in a standardized way. For this a Variability Exchange Language document can be imported and generated by pure::variants. The VEL description document contains all relevant variation points of an asset of the engineering tool and the VEL configuration document contains the configuration of these variation points based on selected features from Feature Models in pure::variants.

For further information about the Variability Exchange Language please visit http://www.variability-exchange-language.org/.

Figure 1, “Overview of family-based software development with pure::variants” shows the four cornerstone activities of software product line development and the models used in pure::variants as the basis for these activities.

When building the infrastructure for your Product Line, the problem domain is represented using Feature Models. The solution domain, i.e. the concrete design and implementation of the software family, is implemented as Family Models.

The two model types used for Application Engineering, i.e. the creation of product variants / configurations, are complementary to the models described above. The Variant Description Model (VDM), containing the selected feature set and associated values, represents a single problem from the problem domain. The Variant Result Model describes a single concrete solution drawn from the solution family.
pure::variants manages the knowledge captured in these models and provides tool support for co-operation between
the different roles within a family-based software development process:

- The domain analyst uses the pure::variants Feature Model editor to build and maintain the problem domain
  model containing the commonalities and variabilities in the given domain.

- The domain designer uses assets of other engineering tools to describe the variable family and to connect it via
  appropriate rules to the Feature Models.

- The application analyst uses a Variant Description Model to explore the problem domain and to express the
  problems to be solved in terms of selected features and additional configuration information. This information
  is used to derive a Variant Result Model from the assets of the other engineering tool.

- The application developer generates a member of the solution (feature selections and variant specific assets)
  from the Variant Result Model by using the transformation engine.

1.2. About this manual

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

1.3. Installation

Please consult section “Additional pure::variants Plug-ins” in the pure::variants User's Guide for detailed infor-
mation on how to install the connector (menu Help -> Help Contents and then pure::variants User's Guide ->
Additional pure::variants Plug-ins).

2. Using the Variability Exchange Language Feature

2.1. Starting pure::variants

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

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

2.2. Importing a Variability Exchange Language Description

The import of a VEL description is started by selecting the Import... action either in the context menu of the
pure::variants Project view or with Import... menu in the File menu. Select Variant Models or Projects in the
opened wizard and press Next. On the following page select Import from VEL document. After pressing Next
a page opens, where the source VEL document and the target folder of the family model to be created is selected.
After pressing **Finish** the import is performed and one family model per variability exchange model is generated. The resulting models will be created in the selected target folder.

The Family Model editor for this type of models contains a special viewer. The **VEL Description** viewer shows in a compact way the variation points, the variations and their conditions by hiding all unnecessary information from the user. The full model can be seen in the **Tree** tab.
The **VEL Description** viewer supports the user by creating, changing, or deleting conditions and expressions at a variation.

### 2.3. Exporting a Variability Exchange Language Description

An imported VEL description family model can be changed within pure::variants. In order to be able to share a changed VEL description pure::variants supports the export of VEL description models to an VEL description XML file.

The export of a VEL description is started by selecting the Export... action either in the context menu of the pure::variants Project view or with Export... menu in the File menu. Select Variant Resources in the opened wizard and press Next. On the next page select the model, which shall be exported, and switch to the next page. Select Export to VEL document and press Next.

![Figure 4. Export VEL Description](image)

The last page of the wizard allows the user to select a target folder, in which the generated VEL description document will be stored.

### 2.4. Importing a Variability Exchange Language Configuration

The import of a VEL configuration is started by selecting the Import... action either in the context menu of the pure::variants Project view or with Import... menu in the File menu. Select Variant Models or Projects in the opened wizard and press Next. On the following page select Import from VEL document. After pressing Next a page opens, where the source VEL document and the target folder of the variant description model to be created is selected. The target folder has to be a configuration space folder.

**Note**

Before a VEL configuration can be imported the corresponding VEL description has to be imported to pure::variants. All resulting family models need to be added as input models to the target configuration space.
After pressing **Finish** the import is performed and one variant description model is generated within the target configuration space. Each variation, which is selected in the imported VEL configuration will be user selected in the resulting variant description model.

The Varint Model editor for variant description models containing at least one VEL description family model contains a special viewer. The **VEL Configuration** viewer shows in a compact way the variation points, the variations and their conditions by hiding all unnecessary information from the user. The full model can be seen in the **Family Models** tab. Besides the selections the tab shows the resulting value of the variation points and give indications on which condition is evaluated to true by a small green icon. Conditions with an red icon evaluate to false.
The VEL Configuration viewer supports the user by selecting variation(s) for a variation point.

### 2.5. Creating a Variability Exchange Language Configuration

To create a VEL configuration document a transformation module is used. First a Transformation Configuration has to be created. To create a Transformation Configuration click on Transformation button in the tool bar (see Figure 7, “Transform model”) and choose Open Transformation Config Dialog...

![Figure 7. Transform model](image)

The configuration space property dialog opens and the Transformation Configuration tab is shown. Next step is to add a new Module Configuration, by clicking the marked tool bar item (see Figure 8, “Transformation Configuration”). Now add a new Module to the Module Configuration, using the Add button.

![Figure 8. Transformation Configuration](image)

A new dialog comes up, choose VEL Configuration Writer module and enter a name. This transformation module provides one parameter. On the next page the output path for the generated VEL configuration document can be given. It is allowed to use pure::variants path variables here.

After finishing the dialogs the transformation can simply be used by clicking on the Transformation button (see Figure 7, “Transform model”) in the tool bar and choosing the transformation in the pull down menu.