Versioning How To

Alexander Weitzmann
Tanya Harizanova (update)

March 22, 2012

Purpose [1]

The vespucci project is evolving quickly and its save format changes from time to time. In order to support older files, we[2] have written a version update framework. It can support new versions with a few simple steps, which are described in this how-to.

Overview [3]

Creating a new version consists of the following steps:

1. Change 'Ns URI'
2. Specify the transformation for:
    - the semantic model
    - the diagram notation
3. Write a new version class.
4. Update the field for the newest version in the version template class.

Note that you update from the last version to your new version, as the updating works incremental. Old versions must not be updated. Changing previous versions will most likely break the version chain.
After following these steps you're done and older diagram files should be updated to your new version.

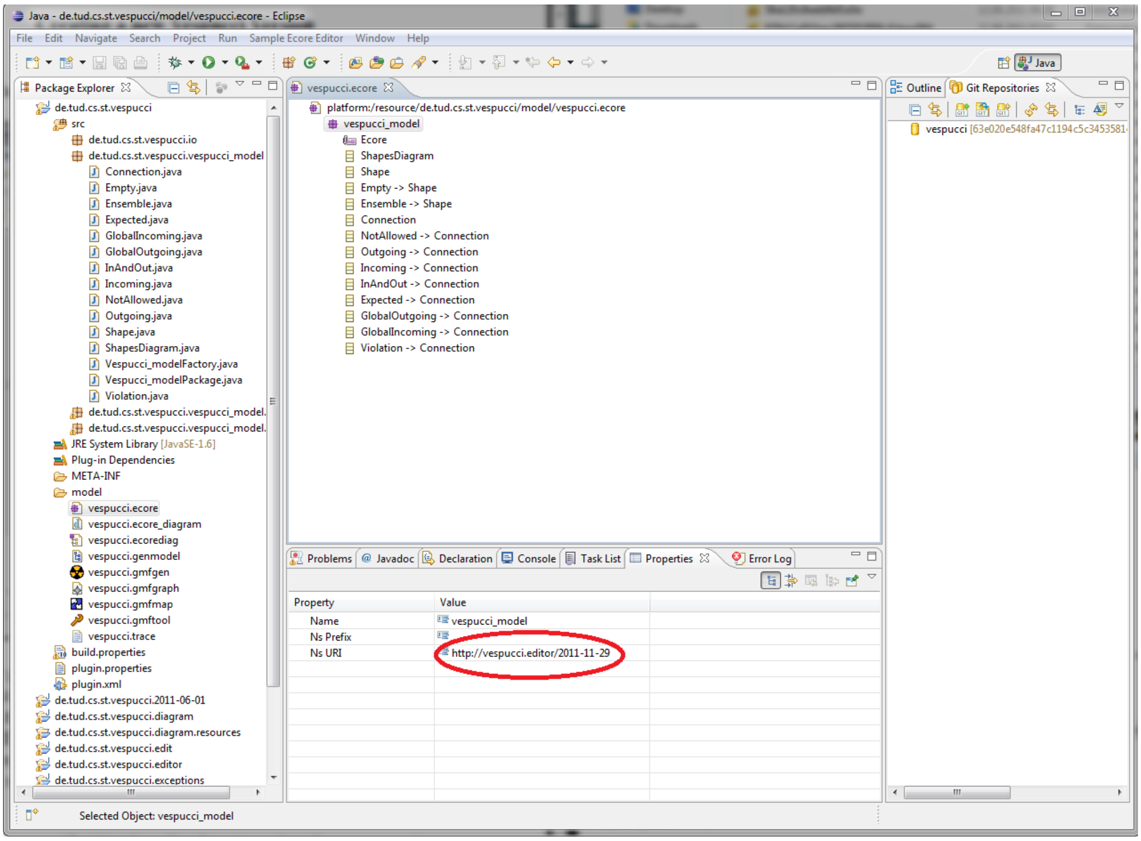
[1] written by Alexander Weitzmann

[2] In fact it was written by Dominic Scheurer and refined by the lab team of SS11.

[3] written by Alexander Weitzmann

Changing the 'Ns URI' [4]

At first you have to change the project version 'Ns URI' as pictured.



Property	Value
Name	vespucci_model
Ns Prefix	
Ns URI	http://vespucci.editor/2011-11-29

After you have done that, regenerate the code and all autogenerated classes are updated. You only have to change 'Ns URI' in the file 'vespucci.trace' manually, which can be found in the same folder as 'vespucci.model'. If you modified plenty of the model, for example 'refactoring', you may additionally need the predecessor version of the project in your workspace. Then you can use it in the next step by Metamodel Mapping.

Transformation specification [5]

The transformation specification describes how to update from one version to another. For every new version a new transformation must be specified;

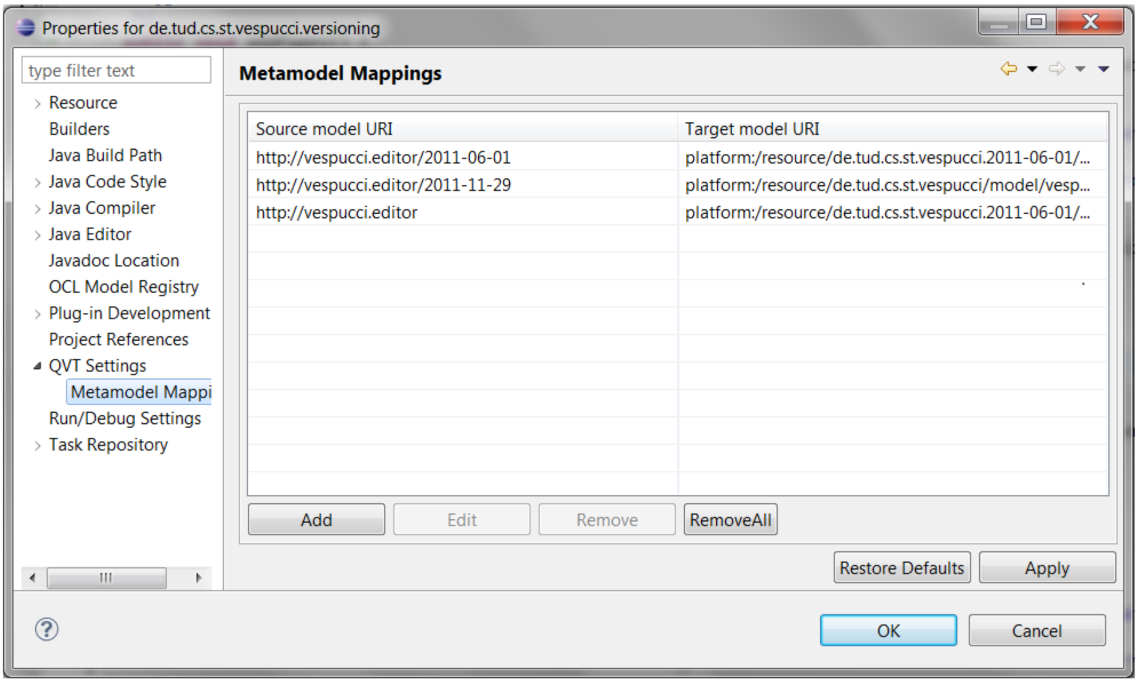
[4] written by Tanya Harizanova

[5] written by Alexander Weitzmann and updated by Tanya Harizanova

if that is not the case a new version is not needed at all. The transformation is realized with QVT (Query/View/Transformation). It consists of the following two parts:

- The **model transformation** updates the semantic model of the diagram, for example setting new relationships between semantic element or adding new ones.
- The **notation transformation** updates the view components of the diagram, for example introducing a new (visual) description field for ensembles or changing the shape of attachments.

Existing transformations can be found in the folder 'transformation' in the 'de.tud.cs.st.vespucci.versioning' project. This is also where you should put your new QVT-transformations. Ignore the error "Cannot find operation (remember())".



Source model URI	Target model URI
http://vespucci.editor/2011-06-01	platform:/resource/de.tud.cs.st.vespucci.2011-06-01/...
http://vespucci.editor/2011-11-29	platform:/resource/de.tud.cs.st.vespucci/model/vesp...
http://vespucci.editor	platform:/resource/de.tud.cs.st.vespucci.2011-06-01/...

Important: Right click on the project 'de.tud.cs.st.vespucci.versioning', then go to 'properties' → 'QVT Settings' → 'Metamodel Mapping' and add the 'Ns URI' of your Version with the path to the model of it. If you keep the old project edit its 'Ns URI' and the path to it with the right one.

Writing a new version class [6]

New version classes are to be placed in the 'de.tud.cs.st.vespucci.versioning.versions' package. All new versions must extend the class 'VespucciVersionTemplate', which is also found in said package. This template defines all methods, which must be implemented.

They essentially...

- point to the associated transformations
- point to the preceding version
- define the creation date, which is also the default identifier.
- the new namespace of the new vespucci version

The last step is to update the field 'NEWEST_VERSION' in the version template, that points to the newest version. You must set this to your newly created version.

Hint: It is very important to add 'plugin.xml' file to your new metamodel URI and the transformations URI.

Hint: Add in 'META-INF' → 'MANIFEST.MF' → 'Dependencies' → 'Imported packages' all of the project packages that are needed.

[6] written by Alexander Weitzmann and updated by Tanya Harizanova