

# Tutorial: Using EMF to create the GCP domain model

Contributed by gdavenport  
Sunday, 17 February 2008  
Last Updated Thursday, 15 July 2010

Tutorial: Using EMF to create the GCP domain model

How to correct or update this document

The Eclipse Modeling Framework (EMF) is a modeling framework and code generation facility for building tools and other applications based on a structured data model.

This tutorial is about it: what is needed to do in order to use EMF to make changes in the GCP Domain Model

## Table of Contents

Preliminary Reading  
Quick summary  
What you need to do

Preliminary Reading  
Before you dive into technical chapters please make sure you understand about EMF first.  
We recommend you read the following tutorials and perhaps try out the examples

- Using EMF
- Model with the Eclipse Modeling Framework, Part 1:  
Create UML models and generate code
- Model with the Eclipse Modeling Framework, Part 2:  
Generate code with Eclipse's Java Emitter Templates
- Model with the Eclipse Modeling Framework, Part 2:  
Customize generated models and editors with Eclipse's JMerge
- We also suggest  
EMF articles in the Eclipse Resources

Set up (one needed to be done once)

- Check out from CropForge in the Pantheon Project (<https://svn.cropforge.org/svn/pantheon/>) the following modules

- /Demeter/projects/Demeter
- /Demeter/projects/DemeterOntology
- /Demeter/projects/Generator
- /Ceres/projects/Ceres

After checking the projects out, make sure the modules from the /projects folder compile cleanly. If there are errors due to maven dependencies, change pom files accordingly but do not commit. If there are still errors you may have to manually install jar files in your local maven repo.

- For eclipse users only:

Make you have Eclipse version 3.4. You will also need maven plugin for eclipse, see Tutorial: Using Maven from Ant for details

- Install the Eclipse plugins for Eclipse Modelling Framework Tools (EMFT) using the Eclipse 3.4's Install manager

Help > Software Updates... > Available Software > Add Site...  
Location: <http://download.eclipse.org/modeling/emft/updates/>  
choose Ecore tools All-in-one Feature

Updating the model using Eclipse and Ant (Semi-automated)

WARNING the ants scripts described below only work with ant using within eclipse  
Updating Demeter and Ceres

- Update the following projects to ensure you have the latest versions

- /Demeter/projects/Demeter
- /Ceres/projects/Ceres

- Make required changes to Java interfaces in Demeter making sure you update any @model and other EMF tags, and that you

- Update the Java documentation.

- Increase the version number of Demeter if necessary in the xmls/project.pom file. Please follow the version nomenclature defined in the best practices.

- Update the version numbers in Ceres so that it matches Demeter
- Update the Ceres version in the Ceres dependency of Demeter

- Run the update-all ant task in the build.xml in the Demeter project. This will run the following subtasks

- update-demeter to update Demeter to remove unrequired EMF classes

- update-ceres to update Ceres to move implementation from Demeter to Ceres
- Refresh the following projects (right click on the project name in the package explorer view and select 'refresh')
- /Demeter/projects/Demeter
- /Ceres/projects/Ceres
- Make sure these projects compile cleanly either using eclipse build or by running the compile ant task. You may need to compile and install the Demeter module in your local repo first.
- Commit the Demeter, DemeterOntology, Ceres and CeresEMF code to SVN
- Release the code to the GCP maven repository.  
See Tutorial: Using Maven from Ant for details on this
- Update the corresponding eclipse Demeter and Ceres plugins.  
See Tutorial: Using Pantheon with Equinox for details on this
- Release the Demeter and Ceres plugins to the GCP plugin update site
- Update the Java docs for these projects on the Pantheon website
- Update DemeterOntology (see next section)

#### Updating DemeterOntology

If Demeter is updated you MUST update DemeterOntology. However, if possible you may wish to make changes in DemeterOntology but not Demeter

- Update the following projects to ensure you have the latest versions
- /Demeter/projects/Generator
- /Demeter/projects/DemeterOntology
- Make changes to DemeterOntology only in the src/etc/config/extended\_ontology.xml file. For guidelines on how to edit this file see the section on Defining Extended Demeter Ontology terms . The version number of DemeterOntology does NOT need to match that of Demeter and Ceres, however if you update Demeter you need to advance the version number of DemeterOntology as well.
- If Demeter has been changed you will need to change the dependency version of Generator on Demeter to the current version. Generator requires access (via maven) to the latest version of Demeter to generate its classes.
- Run the following task from the build.xml in the Generator project.
- update-demeter-ontology to update DemeterOntology
- Refresh the following project (right click on the project name in the package explorer view and select 'refresh')
- /Demeter/projects/DemeterOntology

- Make sure the project compile cleanly either using eclipse build or by running the compile ant task
- Commit the DemeterOntology code to SVN
- Release the code to the GCP maven repository.  
See Tutorial: Using Maven from Ant for details on this
- Update the corresponding eclipse DemeterOntology plugin  
See Tutorial: Using Pantheon with Equinox for details on this
- Release the DemeterOntology plugin to the GCP plugin update site
- Update the Java docs for this project on the Pantheon website

Updating the model using Ant only (Fully-automated)

Not yet available...

#### Defining Extended Demeter Ontology Terms

Extended Demeter Ontology Terms are defined in the `/src/etc/extended_ontology.xml` file in the DemeterOntology project. It defined five different types of terms.

- Data type identifiers, suffixed with the text `_DATATYPE_ID`
- Data type names, suffixed with the text `_DATATYPE_NAME`
- Data type attribute identifiers, suffixed with the text `_DATATYPE_ATTRIBUTE_ID`
- Data type attribute names, suffixed with the text `_DATATYPE_ATTRIBUTE_NAME`
- General constants defined as a value, key pair

The first four are defined for a specific domain model interface and are automatically generated for all sub-interfaces of this interface.

Guy Davenport

Kevin Manansala

Jeffrey Morales

\$Id: UsingEMF.html 9941 2008-01-24 08:28:04Z klmanansala \$