How to create an EMF meta model?

François Tanguy, Didier Vojtisek, Zoé Drey, Marie Gouyette

Abstract

This tutorial show how to create an EMF model for the FSM example.
Preface. ................................................................................................................................. v

Chapter 1. Create a meta-model for Kermeta programs ................................. 1

  1.1. Editing a new meta-model with the EMF sample editor ......................... 1
  1.2. Resulting meta-model .................................................................................. 11
      1.2.1. Metamodel with EMF sample editor .................................................. 11
      1.2.2. Metamodel with Ecore Tools Diagram Editor ................................. 11
  1.3. EMF meta-model creation tips ................................................................. 12
  1.4. External documentation .......................................................................... 13
  1.5. Alternative ways to create a meta-model ............................................... 13

Chapter 2. Conclusion .................................................................................. 15
List of Figures

1.1. Selection of the model object (root element) .......................................................... 2
1.2. Choose a name for your new ecore model ............................................................. 3
1.3. Selection of the model object (root element) .......................................................... 4
1.4. New empty ecore model ......................................................................................... 5
1.5. How to show properties view ................................................................................. 6
1.6. ecore file into Eclipse ............................................................................................. 7
1.7. New child on EPackage root node .......................................................................... 8
1.8. New child on EPackage root node .......................................................................... 9
1.9. Setting the properties of an EReference ...............................................................10
1.10. A simple FSM meta-model .................................................................................... 11
1.11. Sample use of Ecore tool diagram editor on top of an ecore file ......................... 12
1.12. Generate Ecore meta-model from Kermeta ..........................................................14
Kermeta is a Domain Specific Language dedicated to metamodel engineering. It fills the gap let by MOF which defines only the structure of metamodels, by adding a way to specify static semantic (similar to OCL) and dynamic semantic (using operational semantic in the operation of the metamodel). Kermeta uses the object-oriented paradigm like Java or Eiffel. This document presents various aspects of the language, including the textual syntax, the metamodel (which can be viewed as the abstract syntax) and some more advanced features typically included in its framework.

**Important**

Kermeta is an evolving software and despite that we put a lot of attention to this document, it may contain errors (more likely in the code samples). If you find any error or have some information that improves this document, please send it to us using the bug tracker in the forge: http://gforge.inria.fr/tracker/?group_id=32 or using the developer mailing list (kermeta-developers@lists.gforge.inria.fr). Last check: v0.3.1

**Tip**

The most update version of this document is available on line from http://www.kermeta.org.
CHAPTER 1

Create a meta-model for Kermeta programs

For simplicity sake, the different steps (meta-model, model creation, as well as model handling through Kermeta) introduced in the scope of this tutorial are illustrated by means of the finite state machines classical example.

1.1. Editing a new meta-model with the EMF sample editor

1. Create a new General project if no project is ever created (for instance MyFirstEMFSamples).
2. Select in the tool bar menu (on top of Eclipse window) File > New > Other > Eclipse Modeling Framework folder > Ecore Model
Create a meta-model for Kermeta programs

3. Enter a name for the meta-model file (e.g. \textit{fsm.ecore}), then click on \textbf{Next} button.

\textit{Figure 1.1. Selection of the model object (root element)}
Figure 1.2. Choose a name for your new.ecore model

4. Choose **EPackage** as the Model Object (i.e. the root of the meta-model), and click on the **Finish** button.
5. If Omondo is installed, the meta-model can be edited by means of the Omondo graphical editor. However, this tutorial focuses on the creation of meta-models using the **Sample Ecore Editor**. In case the Omondo editor is set as the default editor, the sample Ecore editor can be opened (after having closed the Omondo editor) by right-clicking onto the Ecore file and selecting **Open with > Sample Ecore Model Editor**. (Another way is **Window > Show View > Other > General folder > Properties** like shown in the next figure). Then click on the created EPackage.
6. If the Properties view is not visible at the bottom (or left, or right) part of the Eclipse window, open it clicking into the file -> Show Property View;
7. Two properties of the created EPackage (which is initially displayed as null) have to be set through the Properties tab:

- its **Name**: fsm, for example;
- its **Ns URI**: the namespace URI of the Ecore model is mandatory to allow Kermeta being able to correctly load its potential instances, as well as for the Dynamic creation of instances tool. So put it http://www.kermeta.org/fsm which is the NSURI of the fsm metamodel.
8. At this stage, it is now possible to add children to the created root (which is `fsm` EPackage in the considered example). This could be achieved by right-clicking on the class and choosing the **New child** item. In the scope of the FSM example, three classes are added, one of them (the class `Fsm`) being considered as the “root class” (this should not be not mandatory, but allows a better EMF working):

   a. a `Fsm` class, for which the **Name** attribute (in the properties view) has to be set (other properties do not need to be considered at this stage);

   b. a `Transition` class (idem as for the `Fsm` class);

   c. a `State` class (idem as for previous classes).
Create a meta-model for Kermeta programs

Figure 1.7. New child on EPackage root node

9. To add a new DataType, right click on the fsm EPackage -> EData Type and set its name as String and its Instance Type Name as java.lang.String.
Adding operations, attributes, or references to the newly created classes is achieved in the same way classes have been added to the root EPackage, i.e. using New child on each created element.

Still through the Properties tab, the EContainment property will have to be set to true for each reference which intended to contain instances. In the scope of the automaton example, the state reference of the Fsm class are defined as containment reference, and also the outgoingTransition reference in the State metaclass. By this mean, it will be possible to create, in an EMF model of it, a collection of transitions and states.
Figure 1.9. Setting the properties of an EReference

Setting properties, such as the upper and lower bounds, the type (EType), of the attributes, operations, and references is achieved through the Properties view. The main properties to consider are: ESuperType, EType, Name, Upper Bound, Lower Bound, Containment (for the diamond-ed associations), Ordered, Unique, and EOpposite (opposite property). For simplicity purpose, other properties can be ignored in the scope of this tutorial.
Lower and upper bound properties: 0, 1, -1 (stands for *) are allowed.

Do not forget to save the created meta-model.

1.2. Resulting meta-model

1.2.1. Metamodel with EMF sample editor

At this stage of the tutorial, the designed meta-model should look like the following meta-model:

![Platform: resource/org.kermeta.fsm.emf/model/fsmStatic.ecore](platform:/resource/org.kermeta.fsm.emf/model/fsmStatic.ecore)

```
platform:/resource/org.kermeta.fsm.emf/model/fsmStatic.ecore

fsm
  FSM
    ownedState : State
    initialState : State
    finalState : State
  State
    owningFSM : FSM
    name : String
    outgoingTransition : Transition
    incomingTransition : Transition
  Transition
    source : State
    target : State
    input : String
    output : String
  String [java.lang.String]
```

*Figure 1.10. A simple FSM meta-model*

1.2.2. Metamodel with Ecore Tools Diagram Editor

You can also edit these ecore file using a diagram editor from Eclipse Ecore Tools Project. For example, the ecore tools project provides a class diagram editor that works on top of ecore files. The diagram informations (ie. shape and position of the classes on the sheet) are stored in a separated file : *.ecorediag. To create this .ecorediag file right click on the .ecore file -> Initialize ecore diagram file ... and name the new .ecorediag file. Another way to create this file is File-> New -> Others -> Ecore Tools -> Ecore Diagram. Then select the .ecore file needed or create a new metamodel.
1.3. EMF meta-model creation tips

Creating a good meta-model is sometime difficult due to limitations of the used tools. For a better experience in using EMF tools and Kermeta, it is advised to respect the following rule:

Create an element that will contain directly or indirectly all the other elements. The reflexive editor and the editor generated by EMF allow to create only one root element and then, from this element, create contained elements.

This problem occurs only for model element creation from the editors. The editors correctly display models from meta-models that do not follow this rule if you are able to create such models by another mean. Kermeta is not affected by this constraint.
1.4. External documentation

Additional documentation on EMF can be found at the following links:

1. EMF Documentation
2. Eclipse Corner Article : Using EMF

More generally, most of EMF documentation can be found onto the Eclipse website.

1.5. Alternative ways to create a meta-model

EMF tools are not the only way to create Ecore meta-models. Any tool that can manipulate Ecore can do the same. Here is a small list of tools that can be used to create your meta-model:

1. Omondo / Eclipse UML has a nice graphical editor for Ecore models. It even enables to directly generate the EMF editor from this tool.

2. Kermeta allows users to specify meta-models by means of the Kermeta syntax and then translate them into Ecore using the Kermeta2Ecore function.
The Ecore import/export function and the Omondo editor can be used in order to graphically display the Kermeta classes of a model. It will act as a basic manual roundtrip editor.
CHAPTER 2

Conclusion

This tutorial gives you the basis to create an EMF metamodel. It may be useful to use it with Kermeta for:

1. Creating model instances conform to an.ecore metamodel (cf the tutorial FSM example chapter Editor);
2. Thanks to the Kermeta key word require you can load the.ecore metaclasses and use it into a Kermeta program;
3. Kermeta use EMF to save and load programmaticaly a model (cf the tutorial FSM example chapter Model manipulation in Kermeta)