COMET

A Tool-Chain for **CO**-designing using **M**odel-Driven **E**ngineering for Da**T**aflow Applications

Prerequisites

SDF3: Install from the following link.

http://www.es.ele.tue.nl/sdf3/download/

Eclipse: Download from the following link.

https://eclipse.org/downloads/

Epsilon and necessary plug-ins: Use the following link for guidance.

http://www.eclipse.org/epsilon/

UPPAAL Cora: Download from the following link.

http://people.cs.aau.dk/~adavid/cora/download.html

COMET Manual

It is suggested to create a project which keeps the necessary folder structure to keep your metamodels, models, transformation files, etc. in different folders in a modular way. An example initial folder hierarchy is given below:

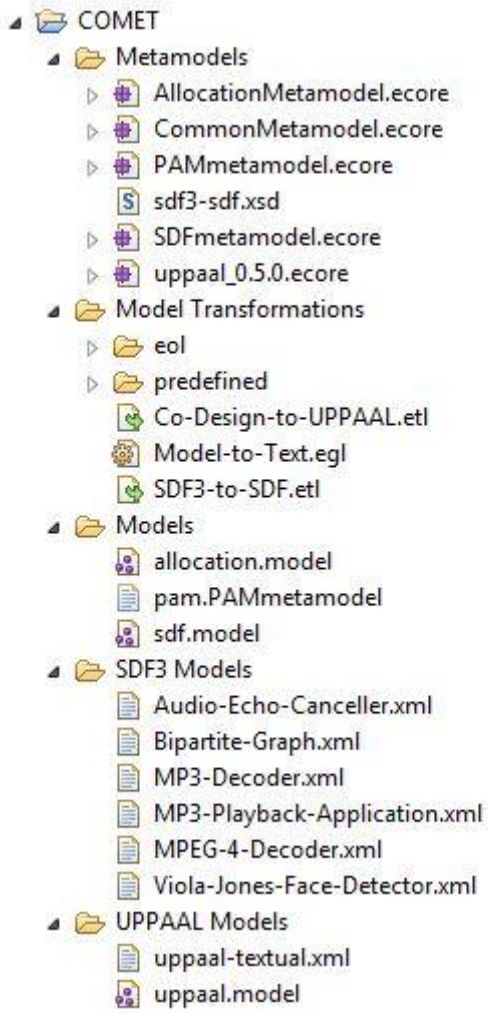

Figure 1: An example folder hierarchy

The following steps are suggested procedure for creating a priced timed-automata model for analysis. The COMET framework is represented in our FMCAD 2016 paper.

Step 1: SDF3 Model Creation

- Create an SDF model using SDF3, and name it, say ***SDF3file.xml***.

- Check the syntactic well-formedness of an SDF graph in the SDF3 format, using the following commands:

Deadlock freeness: sdf3analysis-sdf -- SDF3file.xml --algo deadlock

Consistency: sdf3analysis-sdf -- SDF3file.xml --algo consistency

Step 2: SDF3-to-SDF Transformation

This transformation transforms an SDF3 XML model to SDF models. To run this transformation, please do following:

- Define a transformation configuration instance with the source ***SDF3-to-SDF.etl*** in ***Model Transformations*** folder through “Configuration” screen of Eclipse.

- Select ***SDF3file.xml*** created in step 1 as the SDF3 model and name it “In” for the transformation through “Configuration” screen. The metamodel for SDF3 models is located at ***Metamodels*** folder with the name ***sdf3-sdf.xsd***. (The SDF3 models of the case-studies considered in our FMCAD 2016 paper can be found in the ***SDF3 Models*** folder.)

- Create a new file with the name ***sdf.model***. Name this file “Out” for the transformation through “Configuration” screen. The metamodel of SDF models is located at ***Metamodels*** folder with the name ***SDFmetamodel.ecore***.

- When you run the transformation, ***sdf.model*** will be modified according to the transformation definition specified by ***SDF3-to-SDF.etl.*** This corresponds to the creation of a SDF model from a SDF3 model.

Step 3: Platform Application Model (PAM) Creation

One can create a visual editor from an annotated metamodel using Eugenia plug-in of Eclipse, which can be found in http://www.eclipse.org/epsilon/doc/eugenia/. Here, we provide a short manual about how to achieve this. For details, please refer to Eugenia’s website.

- Create an EMF Model with tags. The PAM metamodel’s annotated EMF form is ***PAMmetamodel.ecore*** and it can be found in ***Metamodels*** folder.

- Create the ECore model by right-clicking on EMF model and selecting *"Generate ECore Model”,* say ***pam.ecore***.

- Derive gen model ***pam.genmodel*** by right-clicking on ECore file and selecting *“Eugenia->Create EMF Genmodel”.* Follow “*EMF Model->ECore Model (CDO Native)*”,.*"Load", "Next"* and *"Finish"* in the following screens.

- Right-click on the root of the genmodel and choose *“Generate all”* to create projects .edit and .editor.

- Right-click on ECore model and choose “*Eugenia->Generate GMF tool, graph and map models*".

- Right-click on genmodel and choose *“Reload”*.

- Right-click on gmfmap model and choose *“Create generator model”* to generate gmfgen model.

- Right-click on gmfgen model and choose “*Eugenia->Synchronize GMF genmodel”.*

- Right-click on gmfgen model and *choose “Generate diagram code”.*

Now, you have the necessary code for the visual editor as a plug-in. You need two more steps for customization of the generated code.

- Replace the pictures and icons in the generated projects with the pictures and icons that you want to display in the editor.

- If you have custom defined icons for your nodes, please follow the guidelines in http://www.eclipse.org/epsilon/doc/articles/eugenia-nodes-with-images/.

Finally,

- Go to Run Configuration and Run from *"Eclipse Applications*".

- In the newly running Eclipse application, if you display the window for “New”, then you see your diagram editor as an item under examples. When you click on it, you will see your editor. We provide a screenshot of an example PAM in the PAM Visual Editor here:

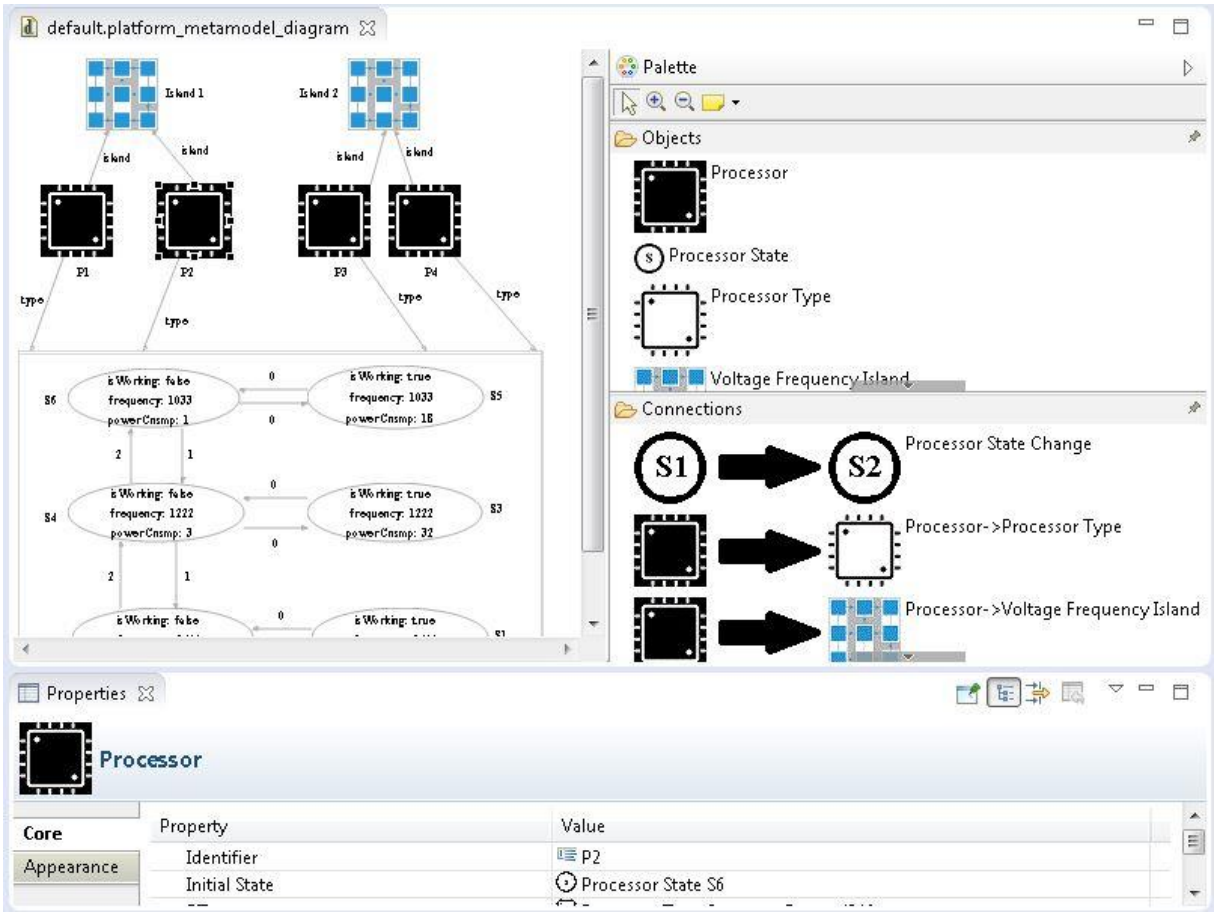

Figure 2: An example PAM in PAM Visual Editor

Here, we already provide PAM Visual Editor available. To run it,

- Open folder ***PAM Visual Editor*** and run *“Eclipse”* from the configuration.

- Then click *“Run”* and *“Run”*.

- In the newly running Eclipse application, create an empty project and create a new diagram by choosing *“Platform_Metamodel_diagram*” in the “*New*” window under “Examples”. You will get PAM Visual Editor.

- To use the model, please copy and paste the model file, say ***PAM.PAMmetamodel*** back to the original Eclipse workspace.

Step 4: Allocation Model Creation

The creation of Allocation models is supported out-of-the-box via the default tree-based model editor provided by EMF. To create a Allocation model, please do following steps.

- Create a new file with the name ***allocation.model***. The metamodel of Allocation models is located at ***Metamodels*** folder with the name ***Allocation.ecore***.
- Right click on the root of ***allocation.model*****,** and click ***Load Resources.*** Now select ***sdf.model,*** and ***pam.PAMmetamodel*** created in step 2 and 3 respectively.
- Now, right click on ***Allocation Root***, and click ***New Child***, and ***Capabilities Capability***.
- For newly created ***Capability***, select ***Actor*** from the drop-down menu, mention ***Exec Time***, and select ***Processor Type*** from the drop-down menu. Repeat this step for each mapping of an actor to the processor types.

An example Allocation model is as follows.

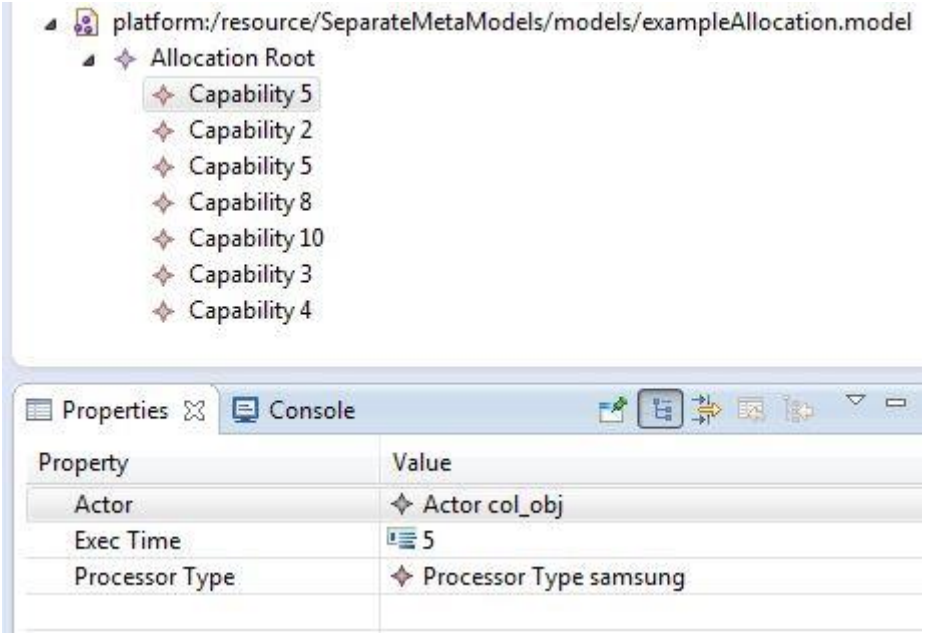

Figure 3: An example Allocation model

Step 5 and 6: Co-Design-to-UPPAAL Transformation and Model-to-Text Transformation

This transformation transforms the SDF, PAM, and Allocation models derived earlier, to UPPAAL Cora models. Co-Design-to-UPPAAL Transformation is achieved in the following way:

- Define a transformation configuration instance with the source ***Co-Design-to-UPPAAL.etl*** in ***Model Transformations*** folder through “Configuration” screen of Eclipse.

- Select ***sdf.model*** created in step 2 as the SDF model and name it “InSDF” for the transformation through “Configuration” screen. The metamodel of SDF models is located at ***Metamodels*** folder with the name ***SDFmetamodel.ecore***.

- Select ***pam.PAMmetamodel*** created in step 3 as the PAM and name it “InPlatform” for the transformation through “Configuration” screen. The metamodel of PAM models is located at ***metamodels*** folder with the name ***PAMmetamodel.ecore***.

- Select ***allocation.model*** created in step 4 as the allocation model and name it “InAllocation” for the transformation through “Configuration” screen. The metamodel of the allocation models is located at ***Metamodels*** folder with the name ***AllocationMetamodel.ecore***.

- Create a new file with name ***uppaal.model*** in ***UPPAAL Models*** folder. Name this file “Out” for the transformation through “Configuration” screen. The metamodel of UPPAAL Cora models is located at ***Metamodels*** folder with the name ***uppaal_0.5.0.ecore***.

- When you run the transformation, ***uppaal.model*** will be modified according to the transformation definition specified by ***Co-Design-to-UPPAAL.etl.***

Now, we need to transform ***uppaal.model*** to the XML format compatible with model checker UPPAAL Cora. To do this, we run a model-to-text transformation in the following way:

- Define a transformation configuration instance with the source ***Model-to-Text.etl*** in ***Model Transformations*** folder through “Configuration” screen of Eclipse.

- Select ***uppaal.model*** created in step 5 and name it “Uppaal” for the transformation through “Configuration” screen. The metamodel of UPPAAL Cora models is located at ***Metamodels*** folder with the name ***uppaal_0.5.0.ecore***.

- Create a new file with the name ***uppaal_textual.xml***. Annotate ***uppaal_textual.xml*** as a file to which the text generated from ***Model-to-Text.etl*** should be printed, through “Configuration” screen.