Using Existing Modelica Models in Modeling with ModelicaML

Wladimir Schamai, EADS Innovation Works
February 6, 2012
Table of Contents

• Introduction
• Motivation
• Typical Usage Scenario
• Live Demo
• OMC API Enhancements
• Conclusion
Introduction / Motivation

- ModelicaML integrates a subset of the UML and the Modelica language
- vVDR (Virtual Verification of Designs against Requirements) is a method that enables a model-based design verification against requirements
- vVDR is supported in ModelicaML

- How to enable the usage of existing Modelica models in ModelicaML?
  - E.g. Libraries or models that are created using Modelica tools
Typical Usage Scenario

- Use a **Modelica** tool to:
  - Develop system design models
  - Simulate models

- Use a **ModelicaML** tool to:
  - Import Modelica models
  - Formalize/model requirements, model test / verification scenarios
  - Compose verification models, simulate verification models and generate reports
  - Visualize dependencies using UML graphical notation (e.g. inheritance)
Concept

- Modelica models can be stored in the “code-sync” folder in ModelicaML Eclipse projects.
- A dedicated viewer allows the browsing of the contained Modelica models.
- A dedicated helper translates Modelica models from the “code-sync” into ModelicaML and marks them as “proxies”.
  - Restriction: the top level Modelica models must be packages and not have any import or extends relations.
- The translated models can be synchronized with the ModelicaML proxies whenever the Modelica models have been modified.
- When synchronizing any identifiable element is updated, other are re-created (references will get lost).
- The created “proxies” can be used in ModelicaML models (i.e. referenced, instantiated).
- No code is generated from “proxies” classes.
- For the simulation the code from both folders must be loaded.
  - the generated ModelicaML model code from “code-gen” folder
  - and the code from the “code-sync” folder.
Implementation
Live Demo
OMC API Enhancements

- Queering of Modelica models using OMC CORBA API
  - \texttt{getImportCount(M1)}, \texttt{getNthImport(M1, 1)}
  - \texttt{getInitialAlgorithmCount(M1)}, \texttt{getNthInitialAlgorithm(M1, 1)}
  - \texttt{getAlgorithmCount(M1)}, \texttt{getNthAlgorithm(M1, 1)}
  - \texttt{getInitialEquationCount(M1)}, \texttt{getNthInitialEquation(M1, 1)}
  - \texttt{getEquationCount(M1)}, \texttt{getNthEquation(M1, 1)}
  - \texttt{getNthComponentCondition(M1, 1)}
  - \texttt{isEnumeration(M1)}
  - \texttt{getEnumerationLiterals(M1)}
  - \texttt{isReplaceable(M1, "C1")}
  - \texttt{getAnnotationCount(M1)}, \texttt{getNthAnnotationString(M1, 1)}

- In progress: \texttt{constrainedBy} and partial derivative function relations
Thank you for your attention!

Wladimir Schamai
EADS Innovation Works (Germany)

wladimir.schamai@eads.net