ModelicaML – Tutorial

Getting Started

Wladimir Schamai
EADS Innovation Works
Systems Engineering
ModelicaML: Technology

1. System Modeling with ModelicaML

2. Modelica Code Generation

3. System Simulation with Modelica Tools
ModelicaML: Technology

Papyrus UML
ModelicaML Profile (Eclipse Plug-In)

Any Modelica Simulation Tool

Model to Text Transformation

Acceleo
ModelicaML Code Generator (Eclipse Plug-In)

Generated Modelica Code (.mo files)
ModelicaML

General Description of the UML-Based Graphical Notation
ModelicaML: Graphical Notation

Structure

Behavior

Requirements

- Requirements
  - id: 001
  - text: The level of liquid in a tank shall never exceed 80% of the tank height.
  - specifiesType: [Tank]

- Requirements
  - id: 002
  - text: The volume of the tank shall be 0.8m³.
  - specifiesObject: [TanksConnectedPI, tank1]

- Requirements
  - id: 003
  - text: Max level of liquid in a tank
  - specifiesType: [Tank]
  - specifiesObject: [TanksConnectedPI, tank1]
ModelicaML: Class Diagram

Class (model, block, record, connector)

variables (of primitive type) or components (of composite type)

extends relation (with type modifications)

(Two-Tanks System Example)
ModelicaML: Connection Diagram

Class (model, block)

Component (of composite type)

Connection

Component (port) of connector type

(Two-Tanks System Example)
ModelicaML: State Machine Diagram

(Two-Tanks System Example)
ModelicaML: Conditional Eq./Alg. Diagram

Set of equations or assignments

Start

Explicit state dependency

ModelicaML: Conditional Eq./Alg. Diagram

Calculate battery status

battery is used (discharging)

battery is not used (charging)

battery status doesn’t change

Conditional flow (if/when) with guards

Explicit state dependency

StateDependency

isInState = [unplugged, on]

logicalOperator = and

[ batteryLevel >= 0 ]

[ else ]

[ batteryLevel <= 1 ]

(Player Example)
ModelicaML

Hands-on Modeling Tutorial
Example: Two Tanks System

• From “source” liquid flows into the “tank1”
• Controller “piContinuous1” controls the level of liquid in “tank1”, based on a predefined reference value, by opening and closing the tank outflow valve.
• Liquid flows from “tank1” into “tank2”
• Controller “piContinuous2” controls the level of liquid in “tank2”

ModelicaML Papyrus MDT GUI Overview
ModelicaML Papyrus MDT GUI Overview

1. Model Browser: Shows model elements
2. Properties View: Shows the properties of selected element
3. Diagram Editors (different UML-based diagrams)
4. Palette (different for each diagram)
5. ModelicaML code generation and validation buttons
6. Component tree: Shows the components hierarchy of the selected class
ModelicaML Project Setup
Create ModelicaML Project

• Open Eclipse
• Change the Perspective to Papyrus Perspective
Create ModelicaML Project

Go to File -> Create …
Papyrus Project Files

Diagram file
(can only be edited using Papyrus MDT)

Model file
(can be edited using any UML2 tool)
Configure Model Explorer

1. Open the Model Explorer.
2. Load customizations: Double-click on "ModelicaMLCustomization".
3. Select "Load referenced facets" and click "OK".
Change and Customize the Perspective

Select and customize the ModelicaML perspective – disable all features that are not used

Typical ModelicaML customized perspective:
Model Setup
Create Model Structure

Create Packages and Classes using ModelicaML menus

ModelicaML Actions
Hint: Setting type of components
Hint: Setting type of components

ModelicaML Primitive Types are:
- ModelicaReal
- ModelicaInteger
- ModelicaString
- ModelicaBoolean
Hint: Setting of the component properties (Declaration, Causality, Variability, etc.)
Create Class Components

- Set variability to “parameter”
- Set causality to “input”
- Set declaration
Create Function Arguments

1 => Set causality to "input"

0 => Set causality to "output"
Inheritance/Extension Modeling
Create Class Diagram
General: Working with diagrams

Creating edges:

1. Select the palette tool

2. Click on the edge source element and hold the mouse button

3. Move the mouse to the target element and release the mouse button
General: Working with diagrams

Deletion of elements:

- Right click on a diagram element

- (preferred) Select the option “Delete Selected Element” in order to delete it from the model. This is recommended in order to keep the model and the diagram consistent.

- Select the option “Hide Selected Element” in order to remove the element from the diagram. The element will still exist in the model and can be shown on the diagram by drag&drop.
Configure Diagram Palette

right click here…
Create Extends Relation

• Drag & drop BaseController and PIcontinuousController onto diagram

• Use the palette tool “Extends Relation”
Hint: Element Appearance: Show stereotype name

1. Select an element

2. Appearance

3. Apply stereotype:
   - Model (from ModelicaML:ModelicaClassConstants)

4. Click stereotype name to show
Hint: Element Appearance: Compartments

1. Right-click the element you want to modify.
2. Select "Show/Hide Compartments".
3. Choose the compartments to show or hide.
Behavior Modeling
Hint: Editing Modelica Code

• Syntax highlighting and code completion is supported in code editors
• Hit Ctrl + Space for code completion when editing Modelica code
Create Behavior

```plaintext
error = ref - cIn.val;
cOut.act = outCtr;
pLim := if p>pMax then pMax
   else if p<pMin then pMin
   else p;
```
Create Behavior

Components
- LiquidSource
  - components (2)
  - behavior (1)
    - eq: set outgoing flow level

PIcontinuousController
- components (1)
- behavior (1)
  - eq: calculate

Equations
- Out.flow = if time > 150 then 3*flowLevel else flowLevel;
  - ModelicaML
  - Profile
  - Advanced

Equations
- $\text{der}(x) = \text{error}/T$;
  - $\text{outCtrl} = K*(\text{error} + x)$;
Create Behavior

\[
\text{der(liquidLevel)} = (q\text{Inflow} - q\text{Outflow})/\text{area}; \quad \text{// Mass balance equation}
\]

\[
q\text{Outflow} = \text{model}\_\text{Design}\_\text{Library}\_\text{limitValue}(\text{minV}, \text{maxV}, -\text{flowGain}\_\text{Actuator}\_\text{act})
\]

\[
t\text{Sensor.val} = \text{liquidLevel};
\]
Create State Machine
Configure Diagram Palette

right click here…
General: Working with diagrams

State Transitions:

1. Select the palette tool

2. Click on the transition source state (click on label) and hold the mouse button

3. Move the mouse to the target state (to its label) and release the mouse button
Create State Machine
Architecture Modeling
Create Connection Diagram

• Create a ModelicaML Connection Diagram under the TanksConnectedPI class
• Use components tool from the palette to create components inside the class on the diagram
• Define the types of components
• Use Model Explorer to find the ports
• Drag&Prop ports into respective components
• Arrange the components
• Use the “Connection” tool from the palette for connecting ports
Model System Architecture
Create Connection Diagram

1. Create components using the palette tools
2. Set component type
3. Drag & drop the port into component
4. Connect components
Hint: Hide the name of all connectors

- Click on the compartment of the class
- Right-click -> “Select” -> “All Connectors”
- Right-click on one of the selected connectors -> “Filters” -> “Show/Hide connector Labels” -> “No connector Labels”
Hint: Element Appearance: Hide the name of the connection stereotype

Select an element
Component Modifications
Define Component Modifications

1. Select the controller1 component.
2. Open the Modifications tab in the properties editor.
3. Click on the modification editor to add a new modification.
4. Set the modification value to 0.25.

Modification: ref = 0.25
Define Component Modifications

- **Source**: LiquidSource
  - UML: flowLevel = 0.02

- **Tank1**: Tank
  - UML: area = 1

- **Tank2**: Tank
  - UML: area = 1.3

- **Controller2**: PIcontinuousController
  - UML: ref = 0.4
Model Validation
Validate Model
Modelica Code Generation
Launch Modelica Code Generation
Generated Modelica Code