Model Engineering Lab 188.923 IT/ME VU, WS 2011/12	**Assignment 4**
Deadline: Upload (ZIP) in TUWEL until Monday, January 9, 2012, 23:55 Assignment Review: Wednesday, January 11, 2012	25 Points

Overview

In the first three assignments of the Model Engineering Lab, you have developed a family of object-oriented modeling languages including their concrete syntax and transformation. According to the model-driven engineering paradigm, the first modeling language, called SOOML, was designed to cover the problem domain without targeting any specific platform or programming language. First steps towards implementation were realized with the transformation of SOOML to SOOPL, where some modeling concepts are explicated using object-oriented programming concepts. In this lab, you will develop a model-to-code transformation to generate executable Java code from SOOPL models.

Code Generation

The goal of this assignment is to develop a model-to-code transformation using *Xpand/ Xtend* [3], such that a SOOPL conform input model is translated into executable Java code.

The reference implementation for the expected Java code for the input model *mowersystem-soopl.xmi* is given in the assignment resources. For your convenience, the file *mowersystem-java.pdf* visualizes the reference implementation as UML Class Diagram. Your task is to implement one or more Xpand Templates which transform arbitrary SOOPL models into the respective Java code.

Install Xpand via the *Install Modeling Components* menu and import the given Xpand project *ME_WS11_Lab4*. The file *Template.xpt* has already been prepared for you as a starting point (located in *src/template/Template.xpt*)

As the SOOPL model covers only state transitions and actions but no "business logic" for implementing the actual methods, you have to generate a *protected region* with a dummy code snippet for halting the thread for a second.

The generated code may be simulated using the Java file *MowerSystemRunner.java*. The simulation is based on the Observer pattern. Thus, the global variable *IsSimulation* set in the generator file should be used to determine, whether the generated code is used for simulation or not. If the code should be simulated, each stateful class has to extend *java.util.Observable*. Further, the methods *setChanged()* and *notifyObservers()* have to be called. If the generated code is not simulated, *java.util.Observable* should not be extended and the respective methods should not be generated.

Keep in mind that the code generator also works with SOOPL models different to mowersystem-soopl.xmi!

Assignment Resources

This project is already configured for code generation and contains the following artifacts:

- ME_WS11_Lab4
  - src
    - metamodel
      - soopl.ecore (from lab 3)
    - template
      - GeneratorExtensions.ext (you may store utility methods in this extension file)
      - Template.xpt (the main code generator template; start here!)
    - Test
      - MowerSystemRunner.java (A basic simulation engine for visualizing states of the Mower System)
    - Workflow
      - generator.mwe (Xpand's Main file; *Run as MWE Workflow* to start code generation)
    - mowersystem-soopl.xmi (from lab 3)
    - mowersystem-java.pdf (UML Class Diagram for the reference implementation)
  - src-gen (target folder for code generation)
  - src-ref (reference implementation of the Mower System)

Testing

Code Generation

1. Right-click on the workflow file *generator.mwe* located in *ME_WS11_Lab4/src/workflow* and select *Run as…* and *MWE Workflow*.
2. The generation should run through without any errors and create the required artifacts in the folder *src-gen* of the same project.

Generated Code

1. Your generated code for the *MowerSystem* defined in the SOOPL input model may be simulated by executing the class MowerSystemRunner.java given in *ME_WS11_Lab4/src/test*.

Submission & Assignment Review

Upload the following components in TUWEL:

One archive file, which contains the Xpand project:

- Select the Xpand project, right-click them, and select *Export → Archive File*. Upload this archive file.

All group members have to be present at the assignment review. Registration for the assignment review can be done in TUWEL. The assignment review consists of two parts:

- Submission and **group evaluation:** 20 out of 25 points can be reached.
- **Individual evaluation:** every group member is interviewed and evaluated separately. Remaining 5 points can be reached [2].

Xpand Setup

Select *Help → Install Modeling Components → Xpand* and follow the setup steps.

Notes

[1] **Assignment Resources:** We provide a preconfigured Eclipse project for this assignment. The project contains the Xpand based code generator and a reference implementation of the Java code to generate. The archive file can be downloaded from TUWEL.

[2] **Evaluation:** The evaluation of your submission includes the **joint** development of this assignment. If a group member did not participate, (s)he can't reach any points.

[3] **Literature** about Xpand can be found at http://help.eclipse.org/indigo/topic/org.eclipse.xpand.doc/help/core_reference.html