Model Engineering Lab 188.923 IT/ME VU, WS 2016/17	**Assignment 3**
Deadline: Upload (ZIP) in TUWEL until Sunday, December 18th, 2016, 23:55 Assignment Review: Wednesday, January 18th, 2016	25 points

Model-to-Model Transformation

The goal of this assignment is to develop a model-to-model transformation using ATL, which transforms state system models (StatesML) into Petri net models (PNML).

Assignment Resources

```
ME_WS16_Lab3_Resources.zip
    at.ac.tuwien.big.statesml.transformations (ATL Project)
    lab3.pdf (this document)
```

Important Artifacts

The following artifacts provided in the assignment resources are important for your transformation:

- **metamodels/statesml.ecore**: Metamodel of the States Modeling Language (StatesML)
- **models/LineFollowingStateSystem.statesml**: Example state system (source)
- **models/LineFollowingStateSystem.pnml**: Petri net produced by your transformation from the example state system (target)
- **models/LineFollowingStateSystem_expected.pnml**: Petri net expected to be produced by your transformation from the example state system.
- **transformations/SML2PN.atl**: Predefined ATL file that has to be modified for the transformation of StatesML models into PNML models.

Setup

Before starting this assignment, make sure that you have all necessary components installed in your Eclipse. A detailed installation guide can be found in the TUWEL course[1].

In addition, you need to install a plug-in which provides the PNML metamodel for Petri nets:

- ePNK[2]

Therefore, start Eclipse and by using Help -> Install New Software insert the following Update Site URL into the field 'Work with:'.

```
http://www2.compute.dtu.dk/~ekki/projects/ePNK/1.1/update/
```

Now click 'Add...'. Enter an arbitrary name (e.g., "ePNK") in the popup window (Add Repository). After the confirmation of the name some plugins should be listed. Select all items and confirm your selection with 'Next'. Confirm the installation details on the next page by clicking 'Next'. Read and accept the terms of the license agreements and click 'Finish'. The installation should now pass without errors. If necessary, confirm the installation of 'unsigned content'. Finally, restart Eclipse.

[1] Eclipse Setup Guide: https://tuwel.tuwien.ac.at/mod/page/view.php?id=289051

[2] ePNK: http://www2.imm.dtu.dk/~ekki/projects/ePNK/index.shtml

Task Description

In this assignment you have to write an ATL model transformation to translate state systems (source models) into Petri nets (target models). The transformation has to be tested with the provided example models. The transformation is considered to be correct, if it generates the same Petri net (target) as the given expected example Petri net out of the example state system (source). Of course, the transformation should be defined in a general way. This means, that the transformation should work properly for all possible state systems. Please note that it is not required to have the same ordering of the elements as in the given expected example Petri net.

Make use of the following predefined strategies for the mapping from elements of state systems (source models) to elements of Petri nets (target models):

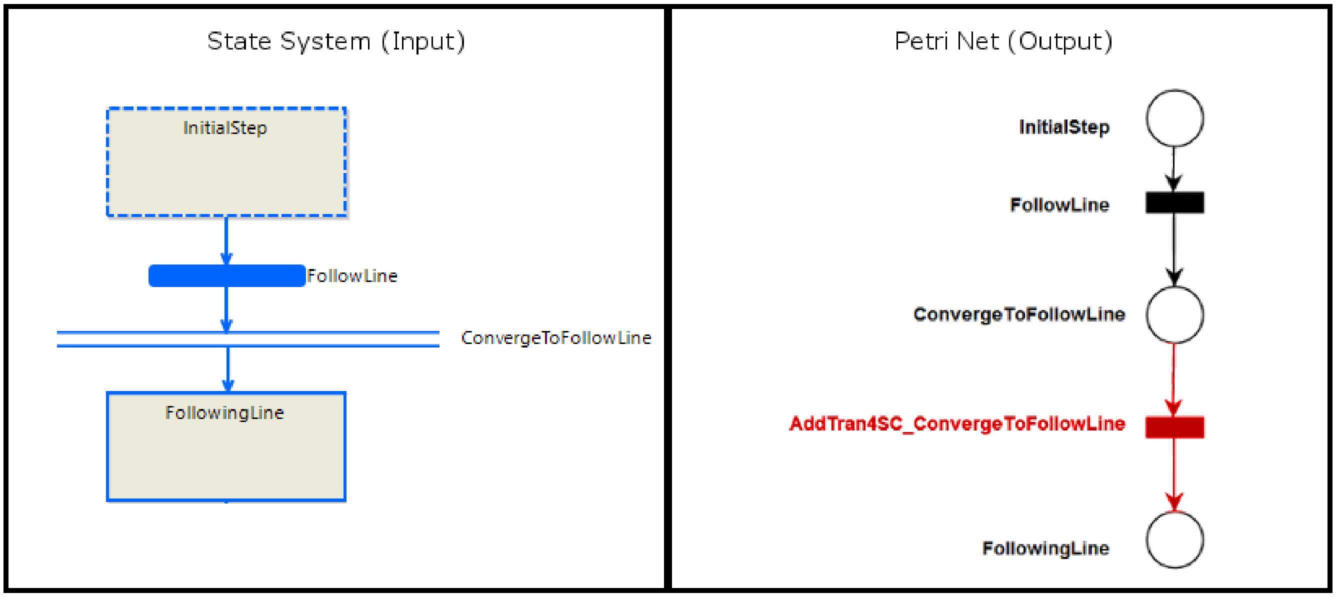
StatesML concept	PNML concept
StateSystem	PetriNet with one Page
State	Place
Transition	Transition
SelectionDivergence	Place with additional Transition for incoming edge
SelectionConvergence	Place with additional Transition for outgoing edge
Edge	Arc

You can find an excerpt of the used Petri net metamodel in the end of this specification.

Additional elements in output model

Places in Petri nets cannot be connected directly, i.e., only arcs between places and transitions are allowed. Therefore, when transforming SelectionConvergence or SelectionDivergence nodes, an additional transition has to be created (cf. Figure 1).

Figure 1: Transformation of a selection convergence in graphical concrete syntax



Additional Requirements

- Create at least one helper function
- Avoid imperative code (*do* section) as much as possible
- Use at least one rule inheritance construct
- Output model elements created for SelectionConvergence nodes and SelectionDivergence nodes should conform to the following naming scheme:
    - SelectionDivergence
        - AddTran4SD_<name> *(Transition)*
        - AddArc4SD_<name> *(Arc)*
    - SelectionConvergence
        - AddTran4SC_<name> (*Transition)*
        - AddArc4SC_<name> (Arc)

Usage of the generated Petri net in TINA

The generated Petri net can be analyzed in TINA (freely available at: http://projects.laas.fr/tina), which is a dedicated Petri net tool to execute and analyze different types of Petri nets.

Import the generated Petri net in TINA and run the TINA analysis tool. Document in a text file whether the Petri net is bounded or not. For this analysis, you have to add a token (marking = 1) to the place that corresponds to the initial state of the StatesML model. Furthermore, discuss in the text file also whether the opposite case (bounded/unbounded depending on your analysis result) is possible in general for StatesML models or not.

Additional Information

Developing the ATL Transformation

- Import the project *at.ac.tuwien.big.statesml.transformations* from the archive file *ME_WS16_Lab3_Resources.zip*. The folder *transformation* contains a predefined ATL file *SML2PN.atl*. This is the only file you need to change in this assignment. Modify the *SML2PN.atl* file in order to implement the transformation of state systems into Petri nets according to the principles described above. The provided example state system model *LineFollowingStateSystem.statesml* should be transformed into the provided Petri net model *LineFollowingStateSystem_expected.pnml*.
- To run the ATL transformation the launch configuration *SML2PN.launch* is already provided by the project in the folder *runner*. This launch configuration will run the transformation for the provided example StatesML model *LineFollowingStateSystem.xmi* and produce the target Petri net model *LineFollowingStateSystem.pnml*. For running the transformation, right-click on *SML2PN.launch* and select *Run As → SML2PN*.
- StatesML models (*.statesml) can be viewed without the generation of EMF editor code. For this, you have to register the corresponding Ecore metamodel. In order to do so, right-click on the metamodel *metamodels/statesml.ecore* and select *EPackages registration → Register EPackages into repository*. Thereafter, you can open a *.statemls model by right-clicking on it and selecting *Open With → Other... → Sample Reflective Ecore Model Editor*.
- PNML models (*.pnml) can also be visualized in a graphical GMF editor. In order to do so, open the *.pnml file in the default (tree) editor (for this double click on the *.pnml file), right-click on the *Page* element and select *ePNK → Start GMF Editor on Page.* Accept the confirmation dialog with *OK* and the GMF editor for the Petri net will open. You can auto arrange the diagram elements if you right-click in the editor and select *Arrange all*.

- Literature to solve this assignment is provided in the TUWEL course. Furthermore, you can find additional information, examples, and use cases on the ATL project homepage http://www.eclipse.org/atl/. All relevant ATL concepts for this assignment can be found in the ATL user manual.

Analyzing the Petri net model with TINA

- Import the Petri net model *LineFollowingStateSystem*.pnml produced by your transformation in TINA. For this, open the TINA tool (e.g., on Windows open "nd.exe") and select *File* → *Import* → *net from pnml*. Select the produced Petri net model. The Petri net model will then be opened and visualized graphically.

  Please make sure that you have graphically visualized your Petri net model in the GMF editor in Eclipse before in order to have a graphical concrete syntax which can be exchanged with TINA. As already described above, you can visualize your Petri net model in the following way: Open the Petri net model in the tree-based editor (double-click on the *.pnml file) and initialize its graphical representation (right-click on the *Page* element and select *ePNK* → *Start GMF Editor on Page*). Switch back to the tree editor and save the model (Crtl+S).

- Add in TINA one token to the place that corresponds to the initial state of the StatesML model ("InitialStep"). For this, enable in the tool bar of TINA the tool "Attributes" (9th icon from the left ✎) and click on the place. A dialog "attributes" will show up. Enter the value "1" for the field "marking" and press *OK*. As a result, one token will be shown in the diagram.

- Analyze the Petri net model in TINA by selecting *Tools* → *reachability analysis*. You don't have to change the analysis options. Press *OK* and the analysis result will be shown.

- Document in a text file whether the example Petri net model is bounded ("Y" in the field "bounded" means that it is bounded and "N" means that it is unbounded) and whether the opposite case is in general possible for StatesML models[3].

Submission & Assignment Review

Upload the following components in TUWEL:

You have to upload a ZIP file named according to the following pattern *WS16_Lab3_<Groupnumber>.zip* containing:

1) The adapted ATL file named according to the following pattern:
   - *WS16_Lab3_Transformation_<Groupnumber>.atl*
2) The boundedness property of the example model as well as the argumentation about this property in general for StatesML models:
   - *WS16_Lab3_Boundedness_<Groupnumber>.txt*

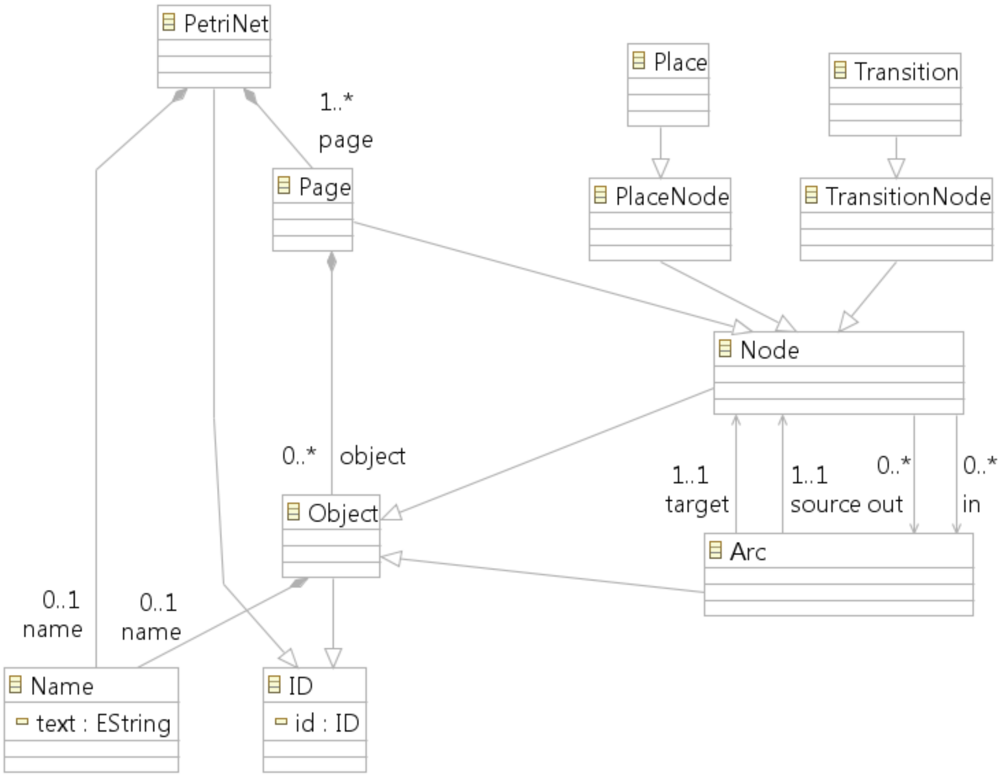
At the assignment review, you will have to present your solution for the ATL transformation and the argumentation about boundedness of StatesML models. You also have to show that you understand the theoretical concepts underlying the assignment.

[3] A Petri net is bounded if the set of all its reachable markings (i.e., all token distributions reachable from the initial token distribution) is finite. More information on Petri nets can be found at http://www2.informatik.hu-berlin.de/~popova/tutorial.html

All group members have to be present at the assignment review. The 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. The remaining 5 points can be reached. If a group member does not succeed in the individual evaluation, the points reached in the group evaluation are also revoked for this student, resulting in a negative grade for the entire course.

PNML Metamodel (Necessary Excerpt for Lab 3)


PetriNet
Place
Transition
1..*
page
Page
PlaceNode
TransitionNode
Node
0..*
object
1..1
target
1..1
source
0..*
out
0..*
in
Object
Arc
0..1
name
0..1
name
Name
text : EString
ID
id : ID