Assignment 5
Code Generation in AToMPM

Joeri Exelmans
joeri.exelmans@uantwerpen.be

1 Practical Information

The goal of this assignment is to generate PythonPDDEVS-based code from a Waterway Network, in the visual modelling tool **AToMPM**.

This assignment does NOT build on the previous MDE assignments. You will be given the "WN2" (Waterway Network 2) formalism, which is based on this year's DEVS assignment. However, this assignment depends on a solution to (a part of) the MoSIS DEVS assignment.

The different parts of this assignment:

1. Create a simple Waterway Network model in AToMPM (or use the included example model).
2. Export it to MetaDepth.
3. Then, in an iterative manner:
   (a) Create an initial version of your EGL transformation, or extend/fix the previous version.
   (b) Run your EGL transformation on the exported MetaDepth model.
   (c) Observe the output, see if things need to be corrected, etc.
4. In the end, given that you have solved the DEVS assignment, you should be able to run the generated Python code with PythonPDEVS.
5. Write a report that includes a clear explanation of your complete solution and the modelling/analysis choices you made, as well as an explanation of your testing process. Also mention possible difficulties you encountered during the assignment, and how you solved them.

This assignment should be completed in groups of two if possible, otherwise individually is permissible.

Submit your assignment as a zip file (report in pdf + model files) on Blackboard before **3 January 2023, 23:59h**[1]. If you work in a group, only *one* person needs to submit the zip file, while all others *only* submit the report. Contact Joeri Exelmans if you experience any issues.

[1] Beware that BlackBoard's clock may differ slightly from yours.

2 Getting Started

2.1 WN2 formalism

Download the WN2 formalism : http://msdl.uantwerpen.be/people/hv/teaching/MSBDesign/WN2.zip, and place it under `AToMPM/Formalisms/WN2`.

2.2 MetaDepth Setup

As a starting point, download the file from: http://msdl.uantwerpen.be/people/hv/teaching/MSBDesign/exported_to_md.zip and place the contents in your `AToMPM/exported_to_md` folder. It contains the following files:

WN2.mdepth The AS meta-model of the WN2 formalism, exported to MetaDepth, and slightly patched (so don't overwrite this file by re-exporting the meta-model!)

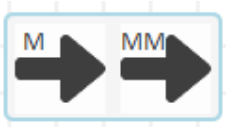
run.mcd The MetaDepth command file containing the necessary commands for running the model-to-text transformation.

main.egl A wrapper around `gen.egl`. Called by `run.mcd`.

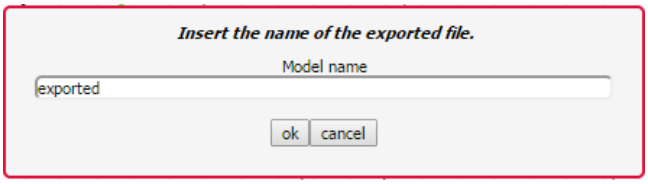
gen.egl This is the file that contains the model-to-text transformation (in the EGL language). **This is the file that you are supposed to edit.**

2.3 Exporting a WN2 model to MetaDepth

1. Open the included example model `/Formalisms/WN2/example.model`. You can also do this later with your own WN2 model.
2. Load the metaDepth toolbar (inside of the `/Toolbars/MetaDepth/` folder, it is called `Export.buttons.model`). This toolbar has two buttons: one for exporting models (M), and one for exporting metamodels (MM).



2. Export your model using the (M)-button. The following dialog will show up:



3. Fill in "exported" (without quotes). This will create a file `AToMPM/exported_to_md/exported.mdepth`.

2.4 Running the model-to-text transformation

1. Run metaDepth.jar in the `AToMPM/exported_to_md/exported.mdepth` directory.
2. In MetaDepth, type the command "`run run`" (without quotes) to run the transformation. This will output a `exported.py` file, which is the result of your model-to-text transformation.

3 Requirements

You must implement a model-to-text transformation in `gen.egl` such that it produces a Python module that contains one CoupledDEVS component that represents the topology of any WN2 model that was used as input. Your generated CoupledDEVS component will create instances of the following CoupledDEVS or AtomicDEVS components, and connect them correctly:

- ControlTower
- Generator
- Dock
- Anchorpoint
- Lock
- Canal
- Waterway
- Confluence
- Sea

The implementation of these components is part of the DEVS assignment. You must therefore complete (that part of) the DEVS assignment in order to complete this assignment. It is required that your generated CoupledDEVS is fully executable.

The precise implementation of all components in the DEVS assignment is somewhat open: you have to choose the constructor parameters of every component, and you have to choose which input/output ports every component has. This means your model-to-text transformation will be specific to *your* solution to the DEVS assignment.

Please include everything (i.e., all Python code) needed to run your generated CoupledDEVS when submitting your solution.

4 AToMPM workaround

As you should know by now, AToMPM cannot distinguish between different incoming or outgoing links of a node in a model: all incoming/outgoing links are an unordered set. However, in your DEVS components, you may want to assign port numbers to specific connections. Therefore, the included `gen.egl` file contains some initialization code that assigns port numbers to every Confluence's in/out connection. Furthermore, for every Confluence, routing information is also generated for finding the shortest path to every Dock (and this routing information is stored in the form of a mapping from Docks to the Confluence's port numbers). You may also want to use this information in your generated CoupledDEVS.

5 Report

There are a number of requirements for the report. Above all, I must be able to read the report and have a clear understanding of all aspects of the assignment, without having to investigate the model or transformation files. I.e., your model files will only be used as a support for your report, not the other way around!

Specifically, the report must contain:

- The complete code of your model transformation.
- Two WN2 model files that you created (i.e., not the included example). For each model, show:
  - A screenshot of the model in AToMPM
  - Your generated CoupledDEVS (also as code in a separate file)
- Your generated CoupledDEVS for the included example model of the Port of Antwerp.
- A discussion of any interesting decisions made and possible improvements to any model or language.

6 Useful Links and Tips

- AToMPM main page: https://atompm.github.io/
- Download and code: https://github.com/AToMPM/atompm
- Documentation: https://atompm.readthedocs.io/en/latest/

Acknowledgements

Based on earlier assignments by Randy Paredis and Bentley Oakes.