CARLETON UNIVERSITY
Department of Systems and Computer Engineering

SYSC 5104. METHODOLOGIES FOR DISCRETE EVENT MODELLING AND SIMULATION

Assignment 1

The goal of this assignment is to show your understanding on modeling discrete-event models using the DEVS formalism, and to use the CD++ simulation tool to execute discrete-event simulations. You must identify a real system that can be represented using DEVS, build a model of the system of choice, and run simulations in order to analyze different conditions on the system. The system of your choice can be any natural or artificial system (existing or not).

Part I:

The first stage is to identify a real system of choice, and to provide a one-page conceptual model description. This document should include:
- a description of the problem to be solved,
- a brief sketch of the model structure, and,
- for each component, a brief description of the behavior of the component.

The model should be decomposable in 2 to 4 levels (two levels means: the top model has a number of submodels, and, at least one of them is a structural model. The third level means that this model should also be decomposed into submodels). The model should have at least **three** atomic submodels (at least 2 of them must be different). The remaining submodels can be atomic or coupled. Students working in teams (**at most, three per team; see conditions in the Appendix**) must define more complex models. Teams should be identified in this first stage report.

Part II

The conceptual model will be used as a requirement document for the final work. You should:

. Organize your models as atomic/coupled, defining the structure and coupling scheme discussed in Part I. This could result in a redefinition of the structure proposed for the conceptual model document of Part I (which will be included in the final report).
. Write a DEVS formal specification for each of the coupled models.
. Write a DEVS formal specification for each of the atomic models, including a description for the transition functions (using pseudocode, state-based representations, DEVS Graphs or others).
. Propose a testing strategy for each one of the models. Document the tests to be executed. The test strategies should consider incremental expansion of the tests, starting at the level of an atomic model, and hierarchically expanding the test base in the model hierarchy. This will be the experimental framework for each of the models involved.

Part III

Use the specification you defined in Part II, and build each of the Atomic Models using CD++. Build individual tests for each of them following the test cases proposed in Part II, trying to ensure correctness of the simulation model. Once the models have been tested, build coupled models, repeating the testing strategies proposed in the specification document. Build the top model of your application, and do integration testing.

Run different simulation examples, changing the original experimental framework, and showing the reaction of the model to different inputs than those defined in the specifications. Use CD++Modeler to analyze the input/output trajectories of the model. Write a report showing these execution results and analyzing the behavior of the model according to the specifications. Document any changes done to the original specifications derived from model execution analysis. If you defined a coupled model using CD++Modeler, use it to study the behavior of the coupled model. Include printouts of the atomic model execution in your report (and analyze the results observed).

Bonus marks

If you have any atomic models that can be defined using DEVS Graphs, define them and implement them using the CD++ Modeler. You should write the DEVS graphs corresponding to the Atomic DEVS model, and provide the files generated as a result. Test the model. Assignments including these definitions will have a 25% bonus.

Important dates:

11/10 – Assignment 1 conceptual model document delivery, 6:00 pm. The document must be delivered as an attachment via email.
25/10 – **Assignment 1 Report delivery, 6:30 pm.** The document must be delivered as an attachment via email.

Marking Scheme

Part I: The goal of this activity is to evaluate the capacity in identifying real systems that can be specified as DEVS models. This first activity represents 20% of the final mark. Those students not able to provide a model by themselves will be given one with the proposed characteristics (but will lose these marks). The deliverable is the one-page report associated to this stage of the assignment.

Part II: This specification document is worth 25% of the final mark. It will include the model specification used in the third stage.

Part III: This part of the assignment is worth 55% of the final mark.

Deliverables

I) DEVSmodelsForm

You should fill out this form, and submit it with your assignment. This form must be filled up properly and submitted with the rest of the materials.

II) Final Report

Your should prepare only one final report, which will be organized as a mix of the documents you created in Parts I-III, all combined in a single document. This document will include:

. Part I
. Part II
. Part III, which should explain the execution results of your model, any variations you have done to the original specification, and the new results obtained by varying the original experimental framework.
. Bonus: if you build the Bonus parts, you should include a description of the DEVS Graphs implemented, and report the simulation results of the atomic models built using this technique.

Remember that he documents for the 3 parts should be combined in a single document to be delivered, included in the same zip file with the simulation software and documentation.

III) Simulation software

Besides the report, you should include the software developed. You should include:

- Source code (**.h/.cpp files**) for each of the atomic models
- Coupled model definitions (**.ma files**)
- **Scripts** to execute each of the models using different experimental frameworks
- '**read.me**' files explaining the detailed behavior for each one of the scripts provided.
- **.ev** files containing input values, if needed
- **.log** files showing the execution results
- Your final **report** document (.doc/.rtf files)
- The **DEVSmodelsForm**, briefly describing the model and giving author information
- If you've done the **Bonus parts**, include the corresponding files (.madesigner, .ggaddesigner, .gcm, .gam, .cdd).

Use meaningful and self-descriptive names <u>for every file included</u>. Consider using names that represent the models you built (DO NOT use "Assign1.doc", or "assign1.h"; nor "MyNameAssign1.cpp", etc… Use a name that represents the content of the model).

Zip <u>all the required files</u> (header files, C++ code, .ma files, .ev files, scripts, report document, etc.), and submit the zip file via email.

<u>You SHOULD NOT include:</u>

- .o files
- .exe files
- .out files

Failure in following these instructions will result in returning and resubmission of the assignment. Check the "alternatebitprotocol" model in the Samples webpage to see what is expected.

The files submitted should run without any problems using the simulation tool. The zip file will be installed in a "clean" project folder. Your assignment will be analyzed as follows (recommendation: **do the same yourselves**):

. The zip file will be installed in an empty project folder
. It will be recompiled. The files should compile cleanly, generating the proper simulator in the end.
. The examples will be run using the scripts you provided.
. The instructor will check the test cases provided.
. After these basic steps are carried out, the instructor will implement different changes to your files, in order to find possible errors. Therefore, be thorough when preparing test cases.

<u>In order to ensure proper execution, try to execute these same steps by yourselves, before delivering the software. DELIVER ONLY THE FILES REQUESTED. If any of the steps cannot be carried out, but can be fixed by the instructor, you might lose marks.</u>

APPENDIX

Students working in teams (**at most, three per team**) must define more complex models using the following constraints:

- Two-member groups: the model should have three to five levels, and at least five atomic models (at least three of them different) OR the complete model should include at least one atomic component completely specified using DEVS Graphs.

- Three-member groups: the model should have three to six levels, and at least five atomic models (at least three of them different) AND at least two of them must be specified completely using DEVS Graphs.

CHECKLIST

Read carefully the requirements specified before, and to everything there explained; if something is missing, the assignment will be returned unread.

Tick each one of these items, and verify you comply with them.

___ You have combined the 3 parts of the report in ONE report. Each section has a meaningful name (not "Part I,II,III", but a descriptive title explaining the content of the section). The report has a meaningful name.

___ You have included the report document in your zip file

___ You have only included the needed files:
- ____ Source code (*.h/*.cpp) for each of the atomic models
- ____ Coupled model definitions (.ma files)
- ____ Scripts to execute each of the models using different experimental frameworks
- ____ 'read.me' files explaining the detailed behavior for each one of the scripts provided.
- ____ .ev files containing input values, if needed
- ____ .log and .out files showing the execution results
- ____ Your final report document (.doc/.rtf files)
- ____ Bonus files (.GAM, .GCM, .CDD, etc.)
- ____ You have NOT included any other files (.project, .o, .exe, etc.).

___ You have put meaningful and self-descriptive names to each of the files

___ You have created a new project, and tried to recompile and run each of the tests, and you were successful in each of them.

___ You have checked the "alternatebitprotocol" zip file to see an example of what is required.