CARLETON UNIVERSITY
Department of Systems and Computer Engineering

SYSC 5104. METHODOLOGIES FOR DISCRETE EVENT MODELLING AND SIMULATION

Term project – Details

The goal of this document is to give you details about the development in the projects you have chosen and give you guidance about the steps to be followed in order to meet the requirements.

Deliverables (everybody)

The details of the deliverables will be discussed in detail for each of the projects. In every case, you should include a 15-20 pages Term Paper explaining the results of your project, showing any results obtained, using IEEE, ACM or SCS format). It should include a brief description of each of the models used (as a conceptual model definition), besides the formal specification. The Term Paper should be organized into:

1. Introduction (brief description of the project, ideas, background and results obtained; it should be organized as 1. Motivation (WHY are you doing this?); 2. Goals (WHAT are you doing it?); 3. Contributions (HOW did you do it?))
2. Background (any detailed information about background needed for the project)
3. Models defined (including an explanation of the models you built, any rules involved, and general description of the ideas implemented)
4. Simulation results (showing execution of the models in the simulation engine, and analyzing the results obtained)
5. Conclusions
6. References

Any source code should run cleanly in the simulation tools, using similar guidelines than the ones used in Assignment 1 and 2. In order to ensure proper execution, try to execute the same steps explained in the assignments by yourselves, before delivering the software.

DEADLINE: December 22, 2019 (earlier submissions are accepted).

Formatting instructions for the Term Papers:
http://www.acm.org/sigs/publications/proceedings-templates
http://www.scs.org/PDFs/formattingkit.pdf

(a good example on organization of the Term Paper can be found in:
http://www.sce.carleton.ca/faculty/wainer/papers/SIW-05s-Battle.pdf)

The authors of the best projects will be invited to be extended as Departmental Technical Reports, and might be submitted for publication, if the students are interested. The final projects will be published on-line in the course webpage for perusal of interested modelers.

The source code should run cleanly. Do not include the final executable file. Zip all the source code (header files, C++ code, .ma files, .ev files, .pal files). Include shell scripts to run the model, explaining how to run them, which input files are needed for each case. Projects not running in the first attempt **will not be graded, and the marks will be lost**. The zip file should include the document of the final report. Follow the guidelines for Assignment 1 and 2 for submitting. Use meaningful names everywhere ("FinalProject" or

“MyName” are examples of names you should not be using; instead, names like “WirelessNetworkSimulator.zip” are better).

The reports will be analyzed as follows:
. The files in the zip file submitted will be recompiled using the makefile provided by you (if any DEVS model was included). The 'make' should compile cleanly, generating the proper simulator in the end.
. The examples will be run using the scripts you provided.
. After these basic steps are carried out, different changes will be studied, in order to find possible errors.

In order to ensure proper execution, try to execute these same steps by yourselves, before delivering the software.

Include 'read.me' files explaining the detailed behavior for each one of the scripts provided. Include the electronic version of your final reports in the zip file.

ALL THESE MATERIALS MUST BE PROVIDED. In each of the projects, I include particular information to be included ('Detailed deliverables'), which DOES NOT exclude presenting the materials here discussed.

Extra Software for some of the project: The RISE middleware mentioned in some of the projects can be found here:

http://www.sce.carleton.ca/faculty/wainer/RISE_test.zip

There are examples, a user manual, and a guide on the syntax for the new interpreter.

Backup: if you fail running the RISE version of the software, you can download it to your computer here:

http://www.sce.carleton.ca/faculty/wainer/cd++.git.rar

It runs under Linux (you can create a Virtual Machine on your Windows computer; if you need a Linux computer, contact Prof. Wainer to get an account in a computer at VSim).

- The password is "hola"
- Ignore the VisualStudio folder.
- This file is a zipped git repository. If you use git you can see the incremental versions and branches. If you do not, simply use the simulator source files as is.

Details of the proposed projects

I. Building Advanced Cell-DEVS models

101141126
101134521

The CD++ toolkit was used to develop different complex systems, as seen in class. We have showed how to model different physical models, including 2D and 3D heat diffusion models, binary solidification, excitable media, surface tension, etc. The goal of this project is to define advanced examples of complex physical systems using Cell-DEVS. The initial version of the project should be an extended version of Assignment 2. The definitive version should be an improved version, including phenomena not originally included in the papers you used to define the models.

i) Read the documentation for the RISE middleware and the new CD++ interpreter.
ii) Run the examples, included in the tool
iii) Implement your own examples, which should include a version of your Assignment 2
iv) Enhance the model described in Assign 2 with new rules, extending the original ones

Software: MUST USE THE RISE MIDDLEWARE DESCRIBED ABOVE

Detailed deliverables:
Submission: similar to Assignment 2 (follow the To-do List)

- Definition of different examples with different input values.
- Execution results of these examples
- Final Report

II. Building an Advanced Cell-DEVS models of CO2 consumption in rooms

100892725

The CD++ toolkit was used to develop different complex systems, as seen in class. We have showed how to model different physical models, including 2D and 3D heat diffusion models, binary solidification, excitable media, surface tension, etc. The goal of this project is to define advanced examples of complex physical systems using Cell-DEVS. The objective is to study the flow of CO2 inside a closed room that has a CO2 sensor installed for the purpose of: (1) measuring the increase in CO2 level, as detected by the sensor, due to the presence of an occupant in the room and (2) measuring the latency between the arrival of an occupant and the detection of the increase in CO2.

In order to do this, the following steps are needed:

i) Read the documentation for the RISE middleware and the new CD++ interpreter.
ii) Run the examples, included in the tool
iii) Meet with Prof. Wainer to discuss the model (as soon as you can)
iv) Implement the model
v) Enhance the model described in Assign 2 with new rules, extending the original ones

Software: MUST USE THE RISE MIDDLEWARE DESCRIBED ABOVE

Detailed deliverables:
Submission: similar to Assignment 2 (follow the To-do List)

- Definition of different examples with different input values.
- Execution results of these examples
- Final Report

III. <u>Definition of advanced Fire models using Cell-DEVS</u>

101166611
101133588
101138167

The CD++ toolkit was used to develop different fire models. The goal of this project is to define advanced models of fire spreading and compare the results of different techniques for this model. This involves running existing models using quantized versions of the models, and to do cost/benefit analysis between reduced traffic and increased error was discussed in recent works. The goal here is to show the applicability of the approach when used in timed Cell-DEVS. The students will execute diverse models, quantifying them using different techniques, and trying to obtain performance-related conclusions.

a) We have defined Cell-DEVS models of this application. You should:

i) Download the model from the Models webpage (FIRE – QUANTIZED (FireQuantized.zip))
ii) Execute it.
iii) Run them using Quantized DEVS (you need to modify the model accordingly). Run the models using different quantum sizes. According to the initial values of the models (read the initial values in the model files), use: 0.01% of average value; 1% of average value; 10% of average value. Compute the number of messages (count the number of lines in the text log files). Compute the average error for each case, using the Cell-DEVS perspective to analyze the execution results (use the tools available there)
iv) Analyze the results. Do modifications to the original equations as needed. THIS PART OF THE PROJECT WILL BE DISCUSSED IN PERSON WITH PROF. WAINER.
v) Propose your own modifications and implement them. THIS PART OF THE PROJECT WILL BE DISCUSSED IN PERSON WITH PROF. WAINER. The modifications will include fire suppression.

Software: MUST USE THE RISE MIDDLEWARE DESCRIBED ABOVE

Detailed deliverables:
Submission: similar to Assignment 2 (follow the To-do List)

- Definition of different examples with different input values.
- Execution results of these examples
- Final Report

IV. Real-Time DEVS models

101167141

CD++ has been modified to include a runtime system that allows defining real-time deadlines. The idea of this project is to create a sample application in which there will be Real-Time models built as DEVS components.

Details will be discussed with Prof. Wainer. The idea is to build a number of sample models like the ones defined in Assignment 1 but using E-Cadmium.

Detailed deliverables:
Submission: similar to Assignment 1 (follow the To-do List; you must include videos of any experiments done with the robots – a digital camera can be provided if needed)

- Definition of different examples with different input values.
- Execution results of these examples
- Final Report

V. Development of application model libraries for CD++

101138246
101152226

The CD++ and Cadmium tools allow to model and simulate complex systems using the DEVS formalism. It has been used to define many different formalisms. In this case, we want to build a library of Finite State Machines in Cadmium. In the course model samples pages you will find the models called FSM and FSM2. The idea is to extend these methods to be able to run in Cadmium. You must adapt the two models, build different examples (at least three examples of each type: Moore and Mealy machines), and conduct simulation analysis.

Submission: similar to Assignment 1 (follow the To-do List)

- Definition of different examples with different input values.
- Execution results of these examples
- Final Report

VI. Visualization of Cell-DEVS models of pedestrian behavior

101166325

The goal of this project is to define advanced visualization Cell-DEVS and advanced visualization engines. The project will have several steps:

i) Download and study the different pedestrian models
ii) Meet with Prof. Wainer to discuss the software tools to be used (as soon as possible)
iii) Implement your own examples

Detailed deliverables:
Submission: similar to Assignment 2 (follow the To-do List)

- Definition of different examples with different input values.
- Execution results of these examples
- Final Report

VII. Building Advanced visualization

101132722

The goal of this project is to define advanced visualization of DEVS and Cell-DEVS models.

The idea is to improve the web viewer by integrating sophisticated and user-friendly cell visualization options into the JavaScript Cell-DEVS Viewer. The topics will be discussed in detail with Prof. Wainer (as soon as possible):

Defining Coupled Models in SVG files
- Define and document proper syntax for SVG files used to define coupled models' graphical representations (this software exists; it needs to be properly documented)
- Provide a clear feedback to the user through interaction with the diagram
- Add visualization for input and output ports
- Loading an SVG file at the same time as the simulation files

WebViewer applications
- Determine a single file format to store all the information required for the WebViewer (log file optimized contents, current state of simulation, viewer layout, etc.).

UI improvements :
- Figure out a color scheme that follows best practices of accessibility and usability
- Design and develop a layout scheme based on HCI principles (consistency, guidance, feedback, error handling, simple displays, etc.)
- Create a user configurable viewer option
- Modify the widget configuration process.

Detailed deliverables:
- Software applications
- Definition of different examples with different input values.
- Videos with simulation results and user manuals
- Final Report

VI. From DEVS coupled model specification in XML to Cadmium code

300093261

Implementing models formally defined in DEVS may is sometimes complex to understand, in particular for domain experts that are interested in modeling but not in programming. The objective of this project is to automate the implementation of DEVS coupled models using a graphical interface. You will need to define a library that uses a graphical version of DEVS coupled models and generates an XML file containing models, ports and link definition. An extension (or a different project) would take as input that XML file containing the definition of a DEVS coupled model and would generate a template to

implement the coupled model in Cadmium.
Details will be discussed with Prof. Wainer (communicate asap).

Detailed deliverables:

- Software applications
- Execution results of different examples
- Final Report