The University of Melbourne

SWEN90004: Modelling Complex Software Systems

Assignment 2

First Semester, 2017

Proposal Due Date: 5:00pm, Friday 5 May, 2017

Final Report Due Date: 11:00am, Friday 26 May, 2017

1 Introduction

This handout specifies Assignment 2, which is worth 25% of your final mark.

This project provides an opportunity for you to develop your skills in developing and conducting experiments with a computational model of a complex system, reporting on both the design of the model and the results of these experiments, and working in a group context. Your task is to replicate and extend an existing grid-based NetLogo model in Java, perform experiments to (a) verify that its behaviour matches that of the original model, (b) pose and answer a research question, and prepare a report on your findings.

It is expected that you will work on this Assignment in groups of three. You should establish groups as soon as possible (in class or using the LMS Discussion Board).

2 Motivation

The idea that a scientific experiment should be reproducible in order to be credible has a long history in science, dating back to early Greek philosophers. Because computational models of complex systems are used as the basis of scientific claims about the behaviour of those systems, it is essential that they are subject to the same level of rigorous evaluation.

Replication of a computational model demonstrates that the results of the original model were not an exceptional occurrence, and can help to increase our confidence in the validity of its behaviour. Replicating a model in a different computer language to the original model can help ensure that model behaviour is independent of any implementation details specific to a particular programming language.

3 Process

1. Select one of the following NetLogo models (available from the **NetLogo Model Library** in the **File** menu of NetLogo):
   - Ethnocentrism (Social Science)
   - Rebellion (Social Science)
   - Wealth Distribution (Social Science)
   - Daisy World (Biology)
2. Explore the behaviour of your chosen NetLogo model. How does it work? What behaviours that are exhibited by the model? Which outputs are measured? What assumptions does the model make?

3. Use your software engineering skills to design and implement an equivalent model in Java.
4. Experiment with your new model. Can you recapture the same behaviours as the original NetLogo model? Why/why not?
5. Extend your new model by implementing one of the suggestions in the model documentation (or a novel extension of your own). Design and run one or more experiments that illustrate the difference in model behaviour that results from this extension.
6. Write a report on your project, as described below.

You should start by implementing the simplest possible prototype (and simulation experiments) and ensure that it works well before proceeding with more complex designs. You must investigate the effects of model parameters on model behaviour. Appropriate analysis of the output of both models is expected; eg, reporting some output measure across a multiple model runs and parameter values; this includes choosing sensible approaches to measuring the behaviour of the models.

Please note marks will **not** be allocated for the development of new libraries or GUI interfaces – your model only needs to generate numerical output (eg, as a CSV file).

By the proposal deadline, you should have chosen a NetLogo model, done some background reading on the real world system represented by the model, explored the behaviour of the NetLogo model, started thinking about the design of your Java model, and thought about the breakdown of tasks and how you will allocate these amongst the members of your team.

4 Submission

Note that only one student from each group needs to make the submissions. **However, ensure that all names and student numbers are clearly visible on the first page.**

4.1 Proposal

The proposal is to be submitted via LMS by the proposal deadline (above). The proposal (named `A_B_C_proposal.pdf`, where `A`–`C` are the last names of each group member) is expected to be 1-3 pages (11pt font, reasonable margins) and contain:

- a descriptive overview of model you are replicating (eg, purpose, users);
- the design of the existing model (eg, states, update rules);
- the design of your model (eg, classes, attributes, methods);
- the experiments you intend to run (optionally, some results from the NetLogo model or early results from your model).
- an plan of how you intend to break down your project into tasks and assign them to group members, and a timeline for completion of these tasks. **NB: the contribution of each group member should be evenly spread across the duration of the project.**

The proposal is worth 0 marks, however failure to submit by the deadline will incur a 1 mark deduction in your final mark. The proposal will constitue a 'first draft' of your report for final assessment, and the interim submission is a valuable opportunity for you to get early feedback.

4.2 Final project

We will use the LMS for the final project submission. There are two separate submissions to be made:

1. a PDF copy of your report (named `A_B_C_report.pdf`, where `A`–`C` are the last names of each group member): Your report should describe the background for the model, the design of your implementation, results of your experiments, and a discussion of your findings. The criteria below (Appendix A) provides an indication of the content expected in your report, and could be used to structure the sections of your report.

   Your report should also include an appendix (maximum one page) outlining how your group worked together to achieve the project; eg, successes and challenges confronted, modifications to the plan outlined in your initial proposal.

   The report must be no longer than **10 pages** (including tables, figures and appendix; 11pt font, with reasonable margins). **Note: marks will be deducted for longer reports.**

2. a zip file (named `A_B_C_code.pdf`, where `A`–`C` are the last names of each group member) containing:

   - all source code
   - any scripts required to run the experiments documented in your report
   - clear instructions describing how to build and run your model

   Code will be tested in a Java SE 8 environment (as found on lab computers), and hence must be compliant with Java SE 8. **Note: marks will be deducted if it is not clear how to build and run your model, or your model does not build and run in a Java SE 8 environment without external dependencies! For example, running your code should not require the use of any third party libraries or an IDE. If you use an IDE to develop your code, you *must* check to ensure that it can be built and run independently of the IDE.**

Late submissions: Late submissions will attract a penalty of 1 mark for every day that they are late. If you have a reason to request an extension, email Nic *well before the due date* to discuss this.
Note that late or no submission of a proposal will also incur a 1 mark deduction as described previously.

Group contribution feedback: At the conclusion of the project, you are required to complete a short questionnaire on LMS rating your own contribution to the group's efforts and that of your fellow group members against the criteria listed in Appendix C. Where there is substantial disparity in contribution, this may be used as a basis for weighting marks assigned to individual group members.

A Criteria

There are two components: the final report (17%) and the quality of your model code (8%):

A.1 Report

Criterion	Description	Marks
Background	You have clearly stated the aims and objectives of your study and provided an appropriate review of background material on your chosen model, including describing why the system modelled is a “complex system” and why this domain is of interest.	2 marks
Model design	You have clearly described the design of your model in sufficient detail (eg, to enable someone else to replicate your work). This includes describing both the conceptual model of the system (how the model relates to the real world), and the design of your Java implementation.	2 marks
Replication	You have designed and executed appropriate experiments to explore and compare the behaviour of your Java model and the original NetLogo model. You have described a range of scenarios (parameter settings) used in your experiments.	3 marks
Extension	You have designed and implemented an appropriate extension to your Java model, specified a question that this extended model allows you to address, and designed and executed one or more experiments to address this question.	3 marks
Results	You have clearly presented the results of your investigations using appropriate tables and figures. All text in figures should be legible and all figures should be accompanied by appropriate captions	3 marks
Discussion	You have interpreted and discussed the results of your experiments, the outcome of the replication exercise, and the answer to the question addressed by your model extension.	2 marks
Writing	Your writing is well-expressed, clearly proof-read and demonstrates a coherent development of ideas. Your appendix outlines the successes and challenges involved in achieving your group’s plan.	2 marks
Total		17 marks

A.2 Code quality

Criterion	Description	Marks
Design	The design of the model is of high quality – clear and succinct – and is potentially extensible	4 marks
Code formatting	The implementation adheres to the code format rules (Appendix B).	2 marks
Executability	The submitted code builds and runs, and generates output consistent with the results provided in the report.	2 marks
Total		8 marks

B Code format rules

Your implementation must adhere to the following simple code format rules:

- Every Java class must contain a comment indicating its purpose.
- Every method must contain a comment at the beginning explaining its behaviour. In particular, any assumptions should be clearly stated.
- Constants, class, and instance variables must be documented.
- Variable names must meaningful.
- Significant blocks of code must be commented.

  However, not every statement in a program needs to be commented. Just as you can write too few comments, it is possible to write too many comments.
- Program blocks appearing in if-statements, while-statements, etc., must be indented consistently. They can be indented using tabs or spaces, and can be indented 2, 4, or 8 spaces, as long as it is done consistently.
- Each line should contain no more than 80–100 characters.

C Group Contribution Feedback Criteria

Feedback on group contributions will be conducted via LMS. You will be asked to assess your own contribution and that of the other members of your group against the following criteria:

1. Motivation, time management and responsibility: attends meetings on time, accepts fair share of work, and reliably completes work on time.
2. Creativity, originality: initiates new ideas, initiates group decisions.
3. Communication skills: good listener, effective contributor to group discussions.
4. General team skills: positive attitude, supports group decisions and helps to achieve consensus.
5. Technical skills: provides technical solutions to problems.

For each criterion, you will be asked to rate your/others contributions on the following scale:

- 4: better than most of the group;
- 3: about average for the group;
- 2: less good than most of the group;
- 1: no help at all to the group; or
- 0: a hindrance to the group.