Inf2C – Software Engineering Coursework

Nigel Goddard
from original by Perdita Stevens

September 25, 2013

This document describes both of the two assignments for Inf2C – Software Engineering. Due dates are 16:00 (4 p.m.) on Friday of Week 4 (11th October 2013) and 16:00 (4 p.m.) on Friday of Week 8 (8th November 2013), respectively.

Reminder: **Late coursework will not be accepted** unless you have a Good Reason for being late, in which case you need permission to submit late from the course organiser (ask for this via the ITO's contact form at http://www.inf.ed.ac.uk/cgi-bin/iss/contact.cgi). Do not ask me for extensions, I'm not allowed to grant them! Deadlines will be arbitrarily strictly applied: do not be even a few minutes late.

Feedback

I aim to get your work marked and give you individual feedback by 2 weeks after the deadline; but please be aware that there are about 100 of you this year so this is ambitious and may slip slightly. Feedback will be in the form of (usually brief) comments and marks for each question, sent to you by email. If you particularly want feedback on any particular aspect of your work, feel free to mention this in your submission.

Introduction

The time allocated to Inf2-SE assignments in the planning of the course is 25 hours (25% of an estimated total workload of 100 hours for the course). (This assumes that you have attended lectures and tutorials and used "private study" time appropriately to keep up with all the course material before you start: time you spend understanding lecture notes, for example, does not count towards the assignment time.) The total credit available for coursework is 25% of the Inf2-SE mark. Each of the two assignments will be marked out of 10. Therefore, one mark on one assignment is worth one-twentieth of 25%, or 1.25% of your total Inf2-SE mark.

I point this out so as to emphasise that the main purpose of doing this assignment is not to get marks, but to take the opportunity to learn things you wouldn't learn by going to lectures, reading, and revising for exams. Remember that because of the "coursework hurdle" you need to get at least 25% of the available coursework marks to pass the course; but beyond that, your main aim should be to learn what you want to learn. This assignment gives you considerable freedom to go in the direction you want to.

Fundamentally this coursework is about becoming more comfortable working with complicated software. Technically, the main Intended Learning Outcome from the course descriptor supported by these assignments is:

- Use a modern IDE to build a large Java system, making appropriate use of configuration management, testing and other appropriate tools.

Depending on your choices, it may also support other listed learning outcomes, or others of your choice.

In later Informatics courses you will be expected, as a matter of course, to be able to cope with large software systems: building them, understanding them, modifying them. Many of you will end up in jobs doing this kind of thing. This coursework aims to let you introduce yourselves to working with real, large systems at your own pace and get credit for it.

Collaboration and plagiarism

What you may do You may discuss your work with others, indeed, I encourage you to do so. For example, you might find it useful to discuss issues that arise in open-source projects, each using your own chosen project as an example. If you need specific help with a problem on your own project, you may seek it (e.g., by posting to the bboard) and use it, **PROVIDED** that you explain in your own submission who gave you what help.

What you must not do You must not submit any of someone else's work, e.g. words, diagrams or code, as part of your submission without clearly acknowledging their source. If you made use of someone else's ideas, you must also say so.

Marking of work involving acknowledged collaboration In software engineering, you need to be able to solve your own problems, but you also need to be able to collaborate. So I *want* you to give and receive help, but I also want you to work on your own. Here's what I'll do. On each of the two assignments, I will use discretion to award up to 2 bonus marks out of 10 to any Inf2-SE students who are named by other Inf2-SE students as having been particularly helpful to others (with a brief description of how they were helpful). I will not penalise students for using help from others: that is, it will be possible to get full marks even if you do get a lot of help from others. However, I will also expect to see evidence that each student has worked hard and can solve problems on their own: if *all* your problems were solved with help from others, you will not score as well as if you had successfully tackled some on your own.

I am aware that this is an unfamiliar way to run coursework. If you are in doubt as to what it means in a particular practical case, contact me.

PART 1

Preparation. 4.5 hours: no credit

Visit open source websites e.g., `http://sourceforge.net`, which is the leading support site for open source software, and browse. Aim to "learn your way around": understand the structure of the site, what you expect to be able to find for a given project, etc.

Pick a package, which you will work on for BOTH parts of the SE coursework, which:

- you find interesting
- is reasonably active: check for evidence that *several different people* have been actively working on the project recently (i.e., within the last six months).
- has source code you're prepared to read. I recommend Java projects, because you'll be using Java in the rest of your degree. But it's ok if you pick a project in another language, e.g. one which you want to learn, provided you're prepared to spend the extra time it will take. Also be aware that neither I nor the lab demonstrators will be able to help you with languages other than Java - it is your own risk. Please don't avoid Java because you find Java hard — if that's the case, it would be a good idea to take this opportunity to improve your Java, because you really will need it in the rest of the degree.
- you expect to present you with some challenge.

You may find `http://sourceforge.net/top/` useful.

Once you have decided on your package, register it against your name using the link from the Inf2-SE page.

Sourceforge is the primary source of projects for this coursework, and there is no need to look at other sites for projects. However, if you *wish* to, you may alternatively choose an open-source project from Google Code (`http://code.google.com/hosting/`), Eclipse Labs (`http://code.google.com/a/eclipselabs.org/hosting/`), GitHub (https://github.com/) or Ohloh (http://www.ohloh.net/). If you particularly want to choose an open-source project hosted elsewhere, email me: I'll probably say yes. The requirement that the project be open-source with its code, etc., readily available on the web is almost non-negotiable, however.

Every student must use a different package. (There are over 100,000 registered projects on Sourceforge, so this should not be too restrictive: but you may want to investigate several packages in parallel, in case your first choice is taken by the time you come to sign up for it.)

Browse its SourceForge (or other repository) page. Download its *source* code and build the package. Check that it runs, and play with it. (You may work on any machine, any platform. Do remember, though, that if you work on a non-DICE machine, it is your responsibility to back up your work.)

Question 1: 4 marks, 2 hours

Explain

- why you picked this package
- briefly, what the package is for
- *exactly* what platform you built it on, whether you encountered any problems, and how you solved them if so.
- some kind of evidence that you actually ran the package, such as a screen shot with your name showing in a field of the GUI.

Question 2: 3 marks, 2 hours

Browse the source code of your project, and see how much you can understand of the project's architecture: that is, of how it is organised into parts and how the parts relate. How are the files organised into directories? You might find it helpful to look at any build and installation scripts. What language(s) are used in the project? Which classes (files, directories) contain what you see as the key functionality of the system? Which control the user interface? Are there any files whose purpose you don't understand? You might need to google for relevant manuals, tutorials etc: save any useful links.

Explain what you have learned. You should aim to write something that you will find a useful reminder when you come to do Part 2 of this assignment.

You may use diagrams (not included in the word count) if you find them helpful. Later in the course, we will meet standard ways to record this kind of information in the Unified Modeling Language.

Question 3: 3 marks, 4 hours

Based on working with the package, reading the project page, bug list, source code, mailing lists, and anything else relevant: discuss what kinds of work contributions might be useful to the project, and why, if it is to continue successfully. (See below for ideas of the kinds of contributions you might consider, though you may well be able to think of others.) What would be *most* useful, and why? Explore what would be involved in each contribution, to the point where you can discuss how hard you think it would be for you to make each kind of contribution that you consider.

Length and style When you create your answers, remember that you will get *m*ore marks for clarity and brevity than you will for quantity. Keep it concise and to the point! Do not write more than 500 words for any of the three questions. For most projects, 250 *well-chosen* words per question is ample for full marks. You can include as many images and diagrams as you want.

Submission Your answer to this assignment should be an HTML web page `index.html` (along with other files in the same directory if appropriate). Very basic HTML is fine, but use links into the SourceForge site as appropriate: I want to be able to mark this by starting by reading your `index.html` in a browser (Firefox, but try not to depend on that!) and looking at the links you give me from it. I don't want to have to go searching for the things you discuss: if you want me to see something, link to it. Put all the files you wish to submit, including `index.html`, in a directory called `inf2c-se1`. Make sure you are in the parent directory of `inf2c-se1`, and submit using the command:

```
submit inf2c-se se1 inf2c-se1
```

PART 2

Develop a contribution, of your choice, to the project.

Possible contributions (or parts of contributions: you don't have to choose just one of these) include:

- Fixing one or more bugs from the bug tracking system (including testing!)
- Reporting bugs into the bug tracking system (be careful that they really are bugs, that you submit useful bug reports, and that you check that the bugs are not already known).
- Writing or improving user documentation for the project (translating existing documention is not appropriate). Note this can sound easier than it is!
- Developing documentation of the design of the project, e.g. a UML class diagram of the overall structure of the project (be sure it's correct UML, correctly corresponds to the project, and is helpfully laid out)
- Improving the design of the system by applying *relevant* refactorings
- Answering questions on the project's mailing lists, e.g., questions from new users of the project
- Adding tests, perhaps for a particularly critical part of the system, or a part known to be buggy.
- Adding Javadoc (including descriptions of what the class is for and what it offers, or how the method works).

You do not necessarily have to choose one of the contributions you identified as "most useful" in PART 1: you may take into account your own skills and indeed your own learning objectives. You should, however, be doing something which is genuinely useful to the project.

Notice that, while in PART 1 you might have built the package from the "latest stable version", some contributions, notably bug fixes, will need you to download the latest sources (typically from the CVS, SVN or GIT archive).

Consider actually adding your contribution to the live project, and do so if appropriate: different projects will have different mechanisms for doing so. (However, only do this once you are confident that the quality of your contribution is high enough to be a genuine benefit to the project.)

Submission As for PART 1, write an HTML web page `index.html`, with links as appropriate, which explains what you did. If you successfully added your contribution to the project, so that you can point to it on SourceForge, do so. Otherwise, submit it to me. Submit any files that you added or that you changed (and if changed, then submit the original beofore you made any changes), together with your explanation. Put all the files you wish to submit, including `index.html`, in a directory called inf2c-se2. Make sure you are in the parent directory. Submit using the command:

```
submit inf2c-se se2 inf2c-se2
```

Timings The estimated time for this assignment is 12.5 hours. You should expect to spend 2 hours on writing up your contribution (though I suggest that you do this as you go along, rather than leaving it until the end). Within the 10.5 hours for making the contribution, the split will of course depend on what you choose to do. I suggest that you identify at least two subtasks. For example, perhaps start with a task you think relatively easy which you expect to take no more than 2 hours, followed by a related but harder task taking the remainder of the time available. I advise you to mention in your submission how long it took you to do each part of your contribution.

Marking guidelines A passing mark (4/10 or above) means that your submission showed that you have spent the recommended time on the assignment, and that in the process you demonstrated competence in the relevant skills from the course. (E.g., UML if your contribution involved UML, JUnit testing if you wrote tests, refactoring and understanding of good design if you refactored, etc.)

A first-class mark (7/10 or above) means that your submission also showed that you had made a substantial, genuinely useful contribution to the project.

Recall that up to 2 bonus marks out of 10 are available for helping others, at my discretion. (E.g., you could obtain 8/10 even if your own submission wasn't quite first-class in the sense above, if in my judgement you had been exceptionally helpful to others.)

Bear in mind:

- The correctness of your contribution is more important than its size. If there's one thing worse than a bugfix that introduces another bug, it's documentation that misleads the reader. Consider carefully what evidence you can provide that your contribution is correct.
- I can only give you credit for something if I understand what you did, so be careful with explanations.
- Usefulness is in the eye of the user, so if you have (verifiable!) evidence that other people appreciate your contribution (e.g., positive mail on the project's mailing list), reference it in your submission.
- You will not be penalised for seeking help from others, provided that you also solve some problems yourself, so if you get stuck and think others might be able to help you get unstuck, ask (e.g., in the lab or on the bulletin board – I will read the bulletin board, so you need only mail me if you want to ask something in private). Remember to mention who helped you in your submission.
- If you find it useful to spend more than the recommended time on the assignment, that's fine. However, if you have kept up with the course material and are competent to pass overall, you should be able to obtain at least a pass mark within the recommended time (and I expect to be able to give full marks to some exceptional students who have spent the recommended time).

If any of this is unclear, ask on the bulletin board

Have fun!