SENG202 – Software Engineering Project Workshop

2020

Outline

1. Introduction

SENG202 is a project-based software engineering course for the first professional year of the BE (HONS) Software Engineering programme. It builds on, applies and extends material introduced in SENG201 (software engineering processes, analysis, design, testing, object-oriented programming in Java) in a professional environment. The course mirrors work habits from industry, including teamwork, basic project management, automated testing, revision control, and incremental / iterative development. Furthermore, the course allows students to reflect on their own practices as well as on the practices of their peers. The focus is on acquiring and developing individual professional skills and demonstrating them in both individual and group context. **Students will work in randomly assigned teams of students.**

By exposing students to state-of-the-art tools and approaches to developing comprehensive software systems, SENG202 bridges the gap between individual or pairwise programming assignments (e.g., in COSC121, COSC122, SENG201) and major group projects (e.g., in SENG302). The course is practice-based and will be the first opportunity for students to undertake a sizeable piece of practical work that spans sufficient time to expose some of the complexities of modern software development in a controlled fashion.

2. Course format

The course consists of lectures and lab sessions.

- One one-hour lecture per week: In addition to introducing material on relevant tools and techniques, lectures will also be used to introduce project tasks, manage groups, give general feedback and steer the project tasks in whole-class discussions. Also, lecture slots may be used for project work. **Students are expected to attend all lectures.**
- Two two-hour laboratories / workshops per week: Labs and workshops include unstructured activities (students will work on their project and be able to ask questions to the staff), as well as structured lab sessions and tutorials to teach essential skills, such as source code control and development environments. The labs are also an opportunity for the groups to present their on-going work to the class, including presentations of deliverables. **There are no lab streams but students are expected to attend all labs.** Labs will also be used for occasional quizzes. Note that quizzes, participation and presentations contribute to overall course assessment.

Important note: Students will be engaged in a medium-complexity software engineering project. Therefore, students are expected to work additional hours in their own time, either having group

meetings or working on their software development project. All students are expected to contribute to the project and contributions to the project contribute to overall course assessment.

3. Course goal and content

This course aims at developing basic individual professional and technical skills required to perform the software engineer role in the development of large and complex software-based systems, which are built in teams and which evolve over their lifetime. In addition to general professional skills, students will learn and practice skills and techniques peculiar to software engineering and software development. The course aims at simulating a real-world project, exposing students to problems faced when developing non-trivial software systems, mirroring work habits from industry, and giving students an opportunity to reflect on their own practices. The course will cover the following topics and related learning outcomes:

Requirements

- Understand and explain the typical difficulties of technical communication
- Analyse product requirements and project scope
- Explain typical requirements-related problems
- Recognize good and bad requirements

Design

- Design a software product
- Explain and apply good design principles
- Recognize and describe the importance of good design
- Understand different design alternatives and their impact on software quality

Implementation

- Implement high-quality code from a design using tools, environments and frameworks
- Identify, review and apply existing code, libraries and packages
- Improve code quality and productivity by using software tools
- Understand and apply revision control
- Break up implementation work into units for parallel implementation
- Coordinate implementation efforts in a team using a code repository

Testing

- Test code with unit tests, system tests, and user tests
- Understand and apply unit testing frameworks
- Understand and apply continuous testing

Other process-related aspects

- Work in a team setting

- Apply methods for identifying and mitigating software project risks
- Critically reflect on own practices and performance and practice and performance of others
- Understand, plan and document all phases of a software development project
- Present (orally and verbally) work to peers and non-technical audiences
- Analyse and solve technology based problems in a technology based environment

The key personal attributes that will be developed include problem-solving ability, practical research skills, ability to work effectively on an unexplored topic of relevance for professional software engineers, and written and oral communication skills. Students will also develop their ability to apply creative and critical thinking to the solving of professional challenges (e.g., adapting and tailoring standard methodologies for use in specific contexts). Furthermore, after completing this course, students will be able to better cope with tasks related to unfamiliar topics, and feel more comfortable speaking to a public audience. The points outlined above allow students to practice:

- Intellectual independence, critical thinking and analytic rigour
- Self-directed learning
- Acquire, understand, assess information from a range of sources
- Communication and collaboration

4. Assessment

This course will not include written exams. Performance of students will be internally assessed.

Item	Description	When	Weight
Quizzes	Online in labs	Throughout semester	20%
Deliverable 1	See instructions	August 3, 2020*	25%
Deliverable 2	See instructions	September 21, 2020*	25%
Deliverable 3	See instructions	October 5, 2020*	30%
Demo	Demo to staff	Lecture/labs week 11	25% of Deliverable 3

***Deliverables must be submitted by 5:00pm on the submission date. There is no drop-dead date.**

Important notes:

- Students must record project activities. If a student does not record activities for a phase, this phase will be considered as "failed" for that student.
- Students will be penalized for unprofessional practice (e.g., missing weekly presentations, missing labs, not following submission guidelines, late or missing reflections, etc.). Professional behaviour will be assessed for each project deliverable.
- All deliverables include a presentation. If a student does not present a deliverable, that student will receive a penalty of 25% for that deliverable.
- Project deliverables will be tested and must run on the platform and environment provided in the computer labs. This means, deliverables must be compile-able and runnable on the environment provided by the department on the lab machines (including the operating system, version of Java, IDEs, etc.).

- The final product will be demoed to teaching staff (lecturers, tutors). Each team will have 20 minutes to demo their product and to answer questions about their project and product.
- Students who do well in SENG202 show balanced code/non-code contributions to the project, demonstrate strong teamwork and leadership skills, and are confident and engaging in presentations.

Your performance is monitored throughout the course and the assessment team will consider your portfolio of achievements when determining your final grade. You will work as part of a team throughout the semester and assessment will be an on-going process that gauges the quality of your individual contributions. You will be assessed individually, but the success of your team will significantly influence your personal results.

You final grade will be determined not just by the quality of the code you write but also by factors such as the level of professionalism, the quality of reports and presentations, the quality of software development artefacts (test cases, diagrams, etc.) and the way you contribute to your group. **Please note that late work cannot be accepted for any of the SENG202 items (no drop-dead date).**

In order to pass the course you must meet the absolute passing mark for the class. In other words, the percentage on all your scored items must meet the absolute passing mark set for the class. Marks are sometimes scaled to achieve consistency between courses from year to year, thus, specific passing marks are identified on a class by class basis. For example, a total mark of 50% would typically be required to achieve a C pass, but this will vary. A C pass indicates that we believe you have (just) mastered the material in the course. Your ranking in the class may be a more useful indicator of your progress; students ranked above you will receive higher grades, and vice versa.

There are several important documents available online about departmental regulations, policies and guidelines. We expect all students to be familiar with these.

You are encouraged to discuss the project with others. However, anything you submit for credit must be entirely your own work and not copied, with our without modification, from any other person unless appropriate acknowledgments and citations are provided. If you need help with specific details relating to your work, or are not sure what you are allowed to do, then contact teaching staff for advice.

Students may apply for special consideration if their performance in an assessment is affected by extenuating circumstances beyond their control. Applications for special consideration should be submitted via the Examinations Office website http://www.canterbury.ac.nz/exams within five days of the assessment. Where an extension may be granted for an assessment, this will be decided by direct application to the course coordinator and an application to the Examinations Office may not be required. Special consideration is not available for items worth less than 10% of the course and/or the following items of assessment: n/a. Students prevented by extenuating circumstances from completing the course after the final date for withdrawing, may apply for special consideration for late discontinuation of the course. Applications must be submitted to the Examinations Office within five days of the end of the main examination period for the semester.

5. Prerequisites and recommended preparation

Prerequisites are SENG201 and subject to approval by Dean of Engineering and Forestry. Students should have a basic understanding of software development and software engineering. SENG202 is restricted to and mandatory for BE (HONS) Software Engineering students.

6. Teaching staff

- Matthias Galster (lecturer and course supervisor), matthias.galster@canterbury.ac.nz
- Patricia Inez de Andrade (senior tutor), patriciainez.deandrade@canterbury.ac.nz
- Sam Shankland (tutor and tech support), sjs227@uclive.ac.nz
- Luke Walsh (tutor and tech support), lwa383@uclive.ac.nz

Office ours: by appointment.

7. Textbooks and recommended reading

Given the nature of the course, and the rapid change in the software industry, there is no single generic text book. Advice will be available from the course coordinator for appropriate textbooks and other resources. Some example resources are listed below.

General software engineering

- I. Sommerville. Software Engineering
- R.S. Pressman. Software Engineering: A Practitioner's Approach

Java

- C. Horstmann. Big Java
- C. Horstmann. Object-oriented Design and Patterns
- B. McLaughlin et al. Object-Oriented Analysis & Design
- E. Freemann et al. Head First Design Patterns

Websites

- Stack Overflow (stackoverflow.com)
- Blogs, forums, Google, etc.

Others as required

- API's, documentation, notes on Learn, etc.