Course Outline

Contents

- Course Details
- Course Summary
- Assumed Knowledge
- Student Learning Outcomes
- Teaching Strategies
- Teaching Rationale
- Student Conduct
- Assessment
- Course Schedule
- Resources for Students
- Course Evaluation and Development

Course Details

Course Code	COMP1531
Course Title	Software Engineering Fundamentals
Convenor	Aarthi Natarajan (/users/z2136899)
Admin	Mei Cheng Whale (/users/z8470184)
Classes	**Lectures** : Wed 10:00 - 12:00 Law Theatre G04 (K-F8-G04) 1-9,10-12 Fri 14:00 - 16:00 Law Theatre G04 (K-F8-G04) 1-9,10-12 Timetable for all classes (/COMP1531/18s2/timetable)
Consultations	TBA
Units of Credit	6
Course Website	http://cse.unsw.edu.au/~cs1531/18s2/ (http://cse.unsw.edu.au/~cs1531/18s1/)
Handbook Entry	http://www.handbook.unsw.edu.au/undergraduate/courses/current/COMP1531.html (http://www.handbook.unsw.edu.au/undergraduate/courses/current/COMP1531.html)

Course Summary

This course provides an introduction to software engineering principles: software life-cycle concepts, software development methodologies, conceptual modeling and how these activities relate to software development. The students are exposed to agile software practices, team collaboration and effective communication through implementing a group project based on agile software methodologies that requires them to analyse, design, build and deploy a web-based application. This course is typically taken in the semester after completing COMP 1511, but could be delayed and taken later. It provides essential background for the teamwork and project management required in many later courses.

The goal of this course is to expose the students to:

- basic elements of software engineering - derived from the life-cycle of a software system, including requirements elicitation, analysis and specification; design; construction; verification and validation; deployment;
- software engineering methodologies, processes, tools and techniques
- agile software, collaboration and communication practices
- web-based system architecture and development practices on web platforms

Assumed Knowledge

We assume that students have taken a first programming course, which has included:

- ability to write simple programs in a programming language e.g. C and are familiar with loop structures, defining and invoking functions and returning results
- exposure to a moderate-sized, team-based project and some testing/debugging ideas

These are assumed to have been acquired in COMP 1511.

Student Learning Outcomes

After completing this course, students will be able to:

1. identify the complexities of software design and development
2. describe the phases of software development and life-cycle of software - and illustrate them from experience
3. analyse a problem domain and elicit user requirements
4. understand object oriented (OO) design principles and develop domain models applying OO based domain modelling techniques
5. understand and apply good software design principles and use a design paradigm to design a software system based on best practices
6. demonstrate effective usage of testing fundamentals (e.g., unit tests, integration tests, test plan/cases, test automation)
7. conceptual data modelling based on ER models, develop simple ER based data models
8. reflect on the choice of software engineering methodology (e.g, waterfall, RUP, agile) used in a project
9. understand agile software development practices and conducting analysis, design, implementation and testing in the context of an agile framework
10. describe common behavior that contribute to the effective functioning of a team and identify necessary roles in a software development team based on agile software practices
11. understand and apply GitHub as a source code management tool and as a medium for collaborating in the development of software applications
12. understand the significance of non functional requirements in software development

Teaching Strategies and Rationale

This course uses the standard set of practice-focused teaching strategies employed by most CSE foundational courses:

- Lectures
- Tutorials and Lab sessions
- Assignments .. will be in the form of both individual assignment, a team-based project and online quizzes

This course aims to provide the students with a strong foundation in the fundamental principles and practices of software engineering that will prepare them for the advanced software engineering workshops. As such, a broad range of key software engineering topics will be taught and reinforced through an individual assignment and a group project, that will enable students to apply the theoretical concepts acquired to solve a practical software engineering problem. An agile software delivery style has been chosen for the implementation of the group project, to make students familiar with modern agile development methodologies.

Lectures (3 hours per week, **#Hours over 12 weeks = 36 hours)**

Lectures will be used to present the theory and practice of the techniques in this course. Although the lectures will primarily focus on the key concepts of software engineering, some lectures will also include practical demonstrations of various key technologies required for the implementation of the group project. Lecture notes will be available on the course web page.

Tutorials and Lab Sessions

Each tutorial_lab_session will comprise a 1 hour tutorial followed by a 2 hour practical lab session. These sessions **commence** from **week 1** .

Tutorials will serve two purposes: (1) clarify ideas from lectures and work through exercises based on the lecture material (2) provide practical demonstrations in the tools and languages that students would need to learn to implement their group project. You should make sure that you use them effectively by examining in advance the material to be covered in each week's tutorial, by asking questions, by offering suggestions and by generally participating. The tutorial questions will be posted on the Web in the week before each tutorial.

Following the tutorial class each week, there will be a **two-hour laboratory class** . The nature of the activities carried out in the lab sessions will vary each week. Lab sessions will comprise design and a variety of small practical exercises involving the tools and frameworks introduced during the lectures. These exercises will be released in the week preceding the lab class. Completed solutions (source-code, design artifacts etc) to each week's lab exercise **MUST** be uploaded to GitHub by the expected due-date. Your tutor will not award any marks for the lab exercise if the completed solution is not uploaded to GitHub. In addition, some lab exercises may require submission through the GIVE submission system. Details of due-date for the lab exercise and submission details will be outlined in each week's lab specification.

Most weeks, the lab exercises will involve tasks that are to be completed individually. Some lab sessions may be done in pairs, where you and partner will be "pair programming". Pair programming is an activity in agile software development activity, where one person is coding and the other person reviews if the software delivers the required goals. A pair-programming session will typically involve multiple exercises to enable you and your partner to swap the roles of the "coder" and "reviewer". You and you partner should discuss the exercises before going to the lab, to maximise the usefulness of the class and decide on the role you will be taking.

Summary: to obtain any lab marks for the Week X lab, you must do 2 things:

- *upload your completed solution to GitHub*
- *demonstrate your work to your tutor in the lab class in the week after it is due OR submit online through GIVE by the due-date if specified . For pair-programming sessions, you should demonstrate as a pair to your tutor during the lab . Pair-programming exercises cannot be demonstrated by only one student, each pair member must take turns .*

You **cannot** obtain marks by e-mailing lab work to tutors. Your tutor will provide feedback during a lab session on your approach to the problem and on the style of your solution. All labs must **normally** be demonstrated OR submitted by the due-date. Your tutor may grant you an extension on the lab exercise if you have been unable to complete the lab exercise due to illness or unavoidable circumstances (e.g, jury duty)

Periodically, the lab sessions will also be used to schedule demonstrations of your ongoing group project. As the implementation of the group project will be based on an **agile software development methodology** , you will required to demonstrate (as a team) progress of your project in iterations. No excuses will be accepted (unless unavoidable circumstances) for some members of the team not being present for the iteration demo.

As this course is has a significant practical component, laboratory classes are important to help you acquire the necessary skills in the relevant tools and frameworks that will be used in the implementation of the group project and also prepare you for the final exam. If you do not put a good amount of effort into the lab classes you risk failing the group project and the final exam.

The practical lab sessions will contribute to **12%** of your overall course mark. The mark for iteration demonstration of the group project will go towards the mark allocated for your group project.

Use of GitHub will be strongly enforced in this course, as using GitHub as the medium of team collaboration and source control is a key learning outcome that will be applied by students in several following courses through their degree.

Assignment, Group Project and Quiz

There will be one assignment (a design-based) to be completed individually which will be released in week 3 and will be due in week 4. This assignment will constitute 5% of the overall course mark.

There will one main group project which will run through the rest of the teaching period and contributes to **30%** of the overall course mark. The specification of the group project will be released in week 5. You are required to form groups of 3 (no more than 3) for the group project.

The group project will be implemented using an agile software delivery mode. As such, your team will be required to build and deliver the project in iterations. Each iteration will deliver a part of the requirements of the project and will encompass all the SDLC activities, namely analysis, design, coding and testing. At the end of the iteration, you (as a team) will demonstrate to your lab class the functionality implemented during that iteration cycle. Changes to project requirements are a natural and unavoidable part of any software project life-cycle. Hence, students will need to bear in mind that project requirements may be subject to change and enhancements to functionalities may be made at the end of each iteration. You will need to carefully design the solution for your current iteration, such that the solution is extensible to accommodate these changes. The lectures will guide you through good design principles to enable you to build modular, extensible solutions.

Each iteration must be demonstrated on the scheduled date. Late demonstrations will not be accepted. A mark will be awarded at the end of each iteration, which will go towards the overall mark for the group project. The final demo of the group project will be conducted in week 13.

Tutors will continually monitor the GitHub repositories to see the team's progress and individual member's contribution to the group project.

Periodically, online quizzes will be released and students will have the opportunity to complete the quiz at their own pace by the specified due-date. The quiz questions will be similar in style to the multiple-choice questions that students will be asked in the final exam and will test the content covered in the lectures in the previous weeks. Quizzes will contribute to **3** % of the overall course mark.

Final Exam

There will be a centrally timetabled final exam which will in your UNSW exam timetable. The exam will contain a mixture of multiple choice questions, "theory" and design questions and programming exercises. More specific details of the exam will be provided through the course.

There is a hurdle requirement on the final exam. If you do not score at least 45% on the exam, you cannot pass this course. If your overall course score exceeds 50%, despite scoring very poorly (<40%) on the exam, the hurdle will be enforced via a grade of UF. Of course, if your overall course score is less than 50%, then your grade will be FL.

If you cannot attend the final exam because of illness or misadventure, then you must submit a Special Consideration request, with documentation, through MyUNSW within 48 hours of the exam. If your request is reasonable, then you will be awarded a Supplementary Exam. If your overall course mark is between 47 and 50, then you will also be awarded a Supplementary Exam, however, the **maximum mark** awarded following a supplementary exam is **50** (applies only to students who have already failed the final exam and have been granted a supplementary exam)

Student Conduct

The **Student Code of Conduct** (Information (https://student.unsw.edu.au/conduct) , Policy (https://www.gs.unsw.edu.au/policy/documents/studentcodepolicy.pdf)) sets out what the University expects from students as members of the UNSW community. As well as the learning, teaching and research environment, the University aims to provide an environment that enables students to achieve their full potential and to provide an experience consistent with the University's values and guiding principles. A condition of enrolment is that students *inform themselves* of the University's rules and policies affecting them, and conduct themselves accordingly.

In particular, students have the responsibility to observe standards of equity and respect in dealing with every member of the University community. This applies to all activities on UNSW premises and all external activities related to study and research. This includes behaviour in person as well as behaviour on social media, for example Facebook groups set up for the purpose of discussing UNSW courses or course work. Behaviour that is considered in breach of the Student Code Policy as discriminatory, sexually inappropriate, bullying, harassing, invading another's privacy or causing any person to fear for their personal safety is serious misconduct and can lead to severe penalties, including suspension or exclusion from UNSW.

If you have any concerns, you may raise them with your lecturer, or approach the School Ethics Officer (mailto:ethics-officer@cse.unsw.edu.au) , Grievance Officer (mailto:grievance-officer@cse.unsw.edu.au) , or one of the student representatives.

Plagiarism is defined as (https://student.unsw.edu.au/plagiarism) using the words or ideas of others and presenting them as your own. UNSW and CSE treat plagiarism as academic misconduct, which means that it carries penalties as severe as being excluded from further study at UNSW. There are several on-line sources to help you understand what plagiarism is and how it is dealt with at UNSW:

- Plagiarism and Academic Integrity (https://student.unsw.edu.au/plagiarism)
- UNSW Plagiarism Procedure (https://www.gs.unsw.edu.au/policy/documents/plagiarismprocedure.pdf)

Make sure that you read and understand these. Ignorance is not accepted as an excuse for plagiarism. In particular, you are also responsible that your assignment files are not accessible by anyone but you by setting the correct permissions in your CSE directory and code repository, if using. Note also that plagiarism includes paying or asking another person to do a piece of work for you and then submitting it as your own work.

UNSW has an ongoing commitment to fostering a culture of learning informed by academic integrity. All UNSW staff and students have a responsibility to adhere to this principle of academic integrity. Plagiarism undermines academic integrity and is not tolerated at UNSW. Plagiarism at UNSW is defined as using the words or ideas of others and passing them off as your own.

If you haven't done so yet, please take the time to read the full text of

- UNSW's policy regarding academic honesty and plagiarism (https://student.unsw.edu.au/plagiarism)

The pages below describe the policies and procedures in more detail:

- Student Code Policy (https://www.gs.unsw.edu.au/policy/documents/studentcodepolicy.pdf)
- Student Misconduct Procedure (https://www.gs.unsw.edu.au/policy/documents/studentmisconductprocedures.pdf)
- Plagiarism Policy Statement (https://www.gs.unsw.edu.au/policy/documents/plagiarismpolicy.pdf)
- Plagiarism Procedure (https://www.gs.unsw.edu.au/policy/documents/plagiarismprocedure.pdf)

You should also read the following page which describes your rights and responsibilities in the CSE context:

- Essential Advice for CSE Students (https://www.engineering.unsw.edu.au/computer-science-engineering/about-us/organisational-structure/student-services/policies/essential-advice-for-cse-students)

Special Consideration

If your work in this course is affected by unforeseen adverse circumstances, you should apply for Special Consideration through MyUNSW, including documentation on how you have been affected. If your request is reasonable and your work has clearly been impacted, then

- for a practical lab session, you may be granted an extension
- for the final exam, you may be offered a Supplementary Exam

Note the use of the word "may". None of the above is guaranteed. It depends on you making a convincing case that the circumstances have clearly impacted your ability to work.

If you are registered with Disability Services, please forward your documentation to Aarthi Natarajan within the first two weeks of semester.

Assessment

Item	**Marks**
Quizzes	3%
Labs	12%
Assignment 1 (Individual)	5%
Group Project	30%
Final Exam	50%

Lab marks vary each week. Your total Lab mark is computed by summing all of your lab marks and mapping into a mark out of 12.

Total marks for the group project will be the sum of the marks obtained at each iteration demo and the final demonstration.

Final Mark

Your final mark for this course will be computed using the above assessments as follows:

Course_Work_Mark	=	lab_mark + quiz_mark + assignment_1_ mark + group_project_mark	out of 50
Exam_Mark	=	Final Exam Mark	out of 50
Exam_OK	=	ExamMark ≥ 25/50	true/false
Final_Mark	=	Course_Work_Mark + Exam_Mark	out of 100
Final_Grade	=	UF, if !Exam_OK FL, if Final_Mark< 50/100 PS, if 50/100 ≤ Final_Mark < 65/100 CR, if 65/100 ≤ Final_Mark < 75/100	

DN, if 75/100 ≤ Final_Mark < 85/100
HD, if Final_Mark ≥ 85/100

Students are eligible for a Supplementary Exam if and only if:

- they cannot attend the final exam due to illness or misadventure
- their Final_Mark is in the range 47 ≤ FinalMark < 50 and Exam_Mark ≥ 24/50. (The maximum mark that a student who has failed the final exam and been granted a supplementary exam is 50.)

A Supplementary Exam will not be awarded for any other reason.

Course Schedule

The following is a **tentative** schedule of when course topics will be covered. The structure of the content and the order in which they are presented will differ from the previous semester and may be adjusted to align with the UNSW 3+ program next year.

Week	Lectures
1	Course Overview, Assessment Criteria Introduction to Software Engineering, Software Development Life Cycle (SDLC) Team Collaboration, GitHub Overview and Demo Introduction to Python programming
2	Requirements Engineering (Requirements Analysis/ Feasibility Study/Requirements Specification) Requirements Analysis using use-cases and user-stories Introduction to UML More Python Programming Release of Assignment 1
3	Object Oriented Design - Objects, Classes, Encapsulation, Inheritance, Relationships Object Oriented Programming with Python Domain modelling using class diagrams
4	Introduction to World Wide Web, HTTP Request/Response Front-end UI Design based on HTML/CSS Introduction to Flask
5	Software Development Methodologies (Waterfall, RUP, Agile methods - XP, Scrum) Software Architecture and Design (Overview, Architectural styles) Building a software application - understanding the choice of development methodology and architecture Release of Group Project
6	OO Design Principles - SOLID principles, Modularity, Cohesion, Coupling Agile Design Techniques
7	OO Design Principles continued.. Applying OO Design to a practical case-study
8	Software testing and validation using testing frameworks Test Driven Development Exception Handling Unit Tests, Integration Tests and User Acceptance Tests
9	Overview of databases Conceptual Design and data-modelling based on ER model
-	Mid semester break
10	Relational models Mapping ER Model to Relational Model Overview of relational databases, tables, primary keys and referential integrity
11	Understanding the importance of non-functional requirements Introducing software application security Guest Lecture
12	Course Review, Final Exam Guidelines

Resources for Students

There is no single text book that covers all of the material in this course at the right level of detail and using the same technology base as we are. The lecture notes should provide sufficient detail to introduce topics, and you will then study them in further depth in the tutorials, labs and group project.

There are also many online resources available, and we will provide links to the most useful ones. Some are listed below. If you find others, please post links in the Comments section on the Course Outline page.

Some suggestions for books that cover at least some of the topics in this course

- *Software Engineerings* , by Ivan Marsic, Rutgers, The State University of New Jersey
- *Agile Software Development: Principles, Patterns and Practice* , by Robert C Martin, Pearson

Course Evaluation and Development

This course is evaluated each session using the myExperience system. However, you are couraged to provide feedback during the semester so that we can address the problems ASAP. In the previous offering of this courses, students have noted that the labs being due in the same week were proving to be difficult particularly for the students having a tut/lab early through the week. This has been noted and we will ensure that students are given adequate time to complete their labs each week. Most lab exercises will now be due on the Sunday of the week they are released, but this may vary depending on the nature of the task.

Resource created 3 years ago (Wednesday 18 July 2018, 09:26:57 PM), last modified 3 years ago (Saturday 18 August 2018, 10:53:41 AM).

Comments

Error getting response from server. Try refreshing the page.

There are no comments yet.