COMP3111/9008 Software Engineering Outline

Lectures	Course Information	Assessment
Project	Communication	Plagiarism Policy
Schedule	Resources	Further Information

Course Staff

Staff	Room	Phone	Email	Role	Consultation
Arcot Sowmya	K17-412E	56933		Lecturer-in-charge	Mondays 4-6 PM, by appointment
Nadine Marcus	K17-401B	55173		Lecturer	Tuesday 11am-12noon, by appointment
Mamatha Rudrapatna	K17-412-06	56906		Administrator	Mondays 4-6 PM, by appointment
Cat Kutay	K17-306	56182		Tool Support Mentor	Mondays & Wednesdays, 10 - 11 AM

Lectures

Day	Time	Location
Monday	6-9 PM	MAT B

Course Information

Units of credit	6	
Parallel teaching	yes	Both COMP3111 and COMP 9008 students attend these classes. The students include postgraduates and undergraduates.
Related courses and programs	*Assumed knowledge:* COMP2011 Data Organisation *or* COMP2711 Higher Data Organisation *or* COMP9024 Data Structures and Algorithms *or* enrolment in MIT program 8684 *or* GradCert program 7344	*Programming assumed knowledge:* We also assume that you have completed basic university computing courses, such as the UNSW courses COMP1021 or COMP1091 or COMP9021, and are familiar with programming in an object-oriented language such as Java or C++.
Course philosophy and teaching strategies	The learning foci in this course are lectures, weekly mentoring sessions, and a large project. The project is broken into concrete tasks with deadlines, which have marks assigned, which will help you to structure your learning. The course philosophy is learning by doing. You will learn the fundamentals of software engineering by actually applying them to a large group project. The lectures are structured to provide you a roadmap, and you are expected to supplement it by extra reading, for which pointers will be provided. Weekly mentoring sessions are planned to provide scaffolding and feedback for your learning, via the project.	
Course aims	Software Engineering is concerned with the processes, methods and tools for the development of high-quality software systems. This involves the study of the software process, the adoption of a life-cycle model and a systematic approach to specification, design, implementation, testing and documentation of software. This course is project-driven, which means students have an opportunity to perform hands-on software-requirements analysis, design, implementation and testing. They will be required to analyse a given problem and suggest a ``good'' solution to the problem in terms of cost, time and performance. ``Problems'' range from simple software to mission critical real-time applications. The role of people (i.e., managers, users, designers, programmers and analysts) and technology are critical to system development and an appreciation of their importance will also be emphasised.	
Learning outcomes	When students have successfully completed this course, they are expected to have: • An appreciation of the role of Software Engineering in software systems. • Experience with the development of project plans, design documents, status reports and results. • Knowledge and ability to use UML Use Cases, ERD and Object modeling methods to model small to medium size systems. • A recognition that production of quality software is a task demanding a disciplined approach to all stages of its development. • An understanding of the concepts and relevance of each stage of the software development life-cycle. • An appreciation of the many and varied issues involved when undertaking small size software projects. In particular, students will also learn and apply generic principles including project management skills, the review process, team skills, group work, brainstorming and managing the varying levels of uncertainty involved in these processes. A primary goal of this course is to teach students the importance of a team effort. Overall, students will learn that both behavioural (human based) and technical views (computer based) of design and implementation are equally important issues. In order to develop long-lasting and efficient software systems---in essence a quality product---Software Engineers need to be aware of and address both these factors in the development process.	
Course content	The course will cover the following topics: Introduction; Software Process; Software Life-cycle Models; Classical and Object-oriented Analysis; Informal Specifications; Design and UML; Visual Design Principles; Implementation; Project Management; Reviews and Testing.	

Back to Table of Contents

Course and Peer Assessment

Your performance in the course will be assessed by the group project. There is no formal final examination. Your project will be assessed against project management criteria discussed in lectures. These criteria involve dividing the project into a number of **deliverables**, which are required to be submitted periodically. Tentative deadlines for each deliverable are shown below; any changes will be publicised through lectures and the class

website.

Task	Mark	Due Date
Project Management	10%	week 4
Requirements & Specification	25%	week 7
Design Document	25%	week 10
Testing Document	10%	week 12
Final Presentation & Report	30%	week 14

In order to pass this course, a satisfactory performance is required for each deliverable of the course.

You will need to carry out a peer assessment for each deliverable except the project management (which are task schedules). In a peer assessment, you will grade the amount of work that each of your fellow group members contributed to the particular deliverable. Note that the peer assessment can lead to your project mark being raised or lowered. More details will be made available via the course website.

Late submission: Unless you have received special dispensation from your mentor, work that is submitted up to one day late will incur a penalty of 10% of your mark. Two days late will incur a penalty of 15%. Any later and your submission may not be accepted at all.

COMP9008 Software Engineering: Assessment in COMP9008 will differ from that in COMP3111 as follows:

The project groups will be homogeneous, ie COMP 9008 students will form their own groups, as will students of COMP 3111. As both courses are introductory and play a similar role in the different programs, no other differences are present.

Back to Table of Contents

Project Groups and Mentoring Sessions

There are NO formal tutorials for this course. Instead, you are required to form a group and attend a group mentoring session with an allocated mentor once a week. This meeting will last for at most 30 minutes, and attendance will be recorded. The meeting will be held at a time suitable to your group and your mentor. At each week's meeting, you will discuss progress on your group project, and receive mentoring advice to help you achieve project milestones.

Your mentor will be your first point of contact if you have queries. If your mentor cannot help you, then you will be asked to contact the course administrator. Please keep your mentor ``in the loop'' with any issue: he/she is there to support you and make sure any problems are resolved. Although consultation times have been posted for the lecturers/admin staff for the course, please email in advance if you wish to see one of us.

You should form your project group of size 4 and register your group once via the class website by end of week 1. The mentoring sessions will start in week 2. Please ensure that all your group members are enrolled in the same course: students enrolled in COMP3111 should team up with other students in COMP3111, and similarly for COMP9008.

The project requirements, and what we expect from you will be announced soon via the course website and lectures.

Support for Visual Paradigm Tool

The course will involve the use of Visual Paradigm which is a UML development tool. This tool will be supported by Cat Kutay (email:ckutay@cse.unsw.edu.au). Please contact her for any problems in setting up or using VP, including details of:

1. System you are using (Linux/WindowsXP etc)
2. What last two steps you were able to perform in your operations
3. Any feedback or error messages from the system during your operation

Ms Kutay will provide regular tutorial exercises during the session which give examples of the diagrams required in your analysis and development for the software project. These examples will be from the domain of the project to make it easier to relate to your work. The exercises will provide examples of operations you can do to become familiar with the working of the software, and answer common issues which arise. There will also be an FAQ page set up on the home page to answer common queries.

Back to Table of Contents

Communication with students

In addition to the UNSW email address, the School of CSE provides students with an email and login account. You are advised that it is the policy of the School that all contact and announcements will be made via your official CSE email address only.

The course website will be the other main source of announcements and information. You should check your school e-mail frequently in case of announcements relating to this course. We assume that you read e-mail sent to your CSE account by the next working day during teaching sessions.

How to forward your CSE E-mail to another e-mail address
CSE E-mail guidelines

Back to Table of Contents

Academic honesty and plagiarism

Copying assignments is unacceptable. Assignments will be checked. The penalties for copying range from receiving no marks for the assignment, through receiving a mark of 00 FL for the course, to expulsion from UNSW (for repeat offenders). Allowing someone to copy your work counts as plagiarism, even if you can *prove* that it is your work.

Further details of the School plagiarism policy can be found here. (You acknowledged receipt of these rules when you obtained your CSE computer account, and the link above is for your convenience so that you can review the rules now.)

We are aware that a lot of learning takes place in student conversations, and don't wish to discourage those. However, it is important, for both those helping others and those being helped, not to provide/accept any programming language code in writing, as this is apt to be used exactly as is, and lead to plagiarism penalties for both the supplier and the copier of the codes. Write something on a piece of paper, by all means, but tear it up/take it away when the discussion is over.

If you are new to studying in Australia, be aware that attitudes to plagiarism at UNSW may be different from those in your home country. Make sure you are clear about the rules here at UNSW. In brief, and for the purposes of COMP9444, plagiarism includes copying or obtaining all, or a substantial part, of the material for your assignment, whether programming language code, or written or graphical report material, without written acknowledgement in your assignment from:

1. a location on the internet;
2. a book, article or other written document (whether published or unpublished) whether electronic or on paper or other medium;
3. another student, whether in your class or another class;
4. a non-student (e.g. from someone who writes assignments for money)

Note that if you copy code or other material from another student or non-student with acknowledgement, you will not be penalised for plagiarism, but you are unlikely to get any marks for the copied material. If you use code found in a publication (on the internet or otherwise) then the marks you get for this will be at the marker's discretion, and will reflect the marker's perception of the amount of work you put into finding and/or adapting the code, and the degree to which you understand the code.

Note also that there is a big difference between being able to understand someone else's code, and writing that code yourself from scratch. A computer programmer has to be able to write code from scratch. The assignments provide opportunities for you to develop the skills necessary to write your own neural-network-related code. Use these opportunities!

Back to Table of Contents

Course schedule

Lectures: Monday 6-9PM; Room: MAT BSchedule

Lecture notes for each week will become available through the appropriate link in the afternoon of the lecture day (Monday).

Schedule: (the order and content shown below may vary - this is an indication only).

Week	Date	Lecturer	Topic	Project Task Specs(Planned)
1	27 Feb	Sowmya	Outline and Introduction	
2	6 Mar	Nadine	Project Management	Project Specs
3	13 Mar	Sowmya	Requirements	
4	20 Mar	Nadine	Software Lifecycle and Software Process	
5	27 Mar	Sowmya	Specification and ERDs	Requirements and Specs Document
6	3 Apr	Sowmya	OO analysis, UML	
7	10 Apr	Sowmya	Design, UML 1	
	17 Apr		Mid-session Break	
8	24 Apr	Sowmya	Design, UML 2	Design Document
9	1 May	Nadine	Visual Design	
10	8 May	Sowmya	Implementation	
11	15 May	Nadine	Testing	Testing Document
12	22 May	Sowmya and Nadine	Project Consultations	
13	29 May	Sowmya and Nadine	-	Demo and Report Specs
14i	5 June	Sowmya and Nadine	Project Demos	

Back to Table of Contents

Resources for students

Resources for students include the course lecture notes, which will appear on the class web page linked at this location as the course runs. In some cases, these notes provide pointers to further resources, such as internet sites and publications. Also the required software which is a CASE tool called Visual Paradigm is available for download or use in the labs (see below).

Text Books

Curtis HK Tsang, Clarence SW Lau and Ying K Leung, *Object-oriented Technology from Diagram to Code with Visual Paradigm for UML*, McGraw-Hill Education, 2005.
Stephen R. Schach, *Object-Oriented and Classical Software Engineering*, McGraw-Hill International, 6th edition, 2005.

Recommended Reference Books

F.P. Brooks, *The Mythical Man-Month. Essays on Software Engineering*, Addison-Wesley, 1975.
M. Page-Jones, *The Practical Guide To Structured Systems Design*, Yourdon Press/Prentice-Hall, 2nd edition, 1988.

P.Sallis, G.Tate, S.MacDonell, *Software Engineering: Practice, Management, Improvement*, Addison-Wesley, 1995.
E. Yourdon, *Modern Structured Analysis*, Prentice-Hall, 1989.
J. A. Hoffer, J. F. George, J. S. Valacich, *Modern Systems Analysis and Design*, Third Edition, Pearson Education, NJ, 2002.
B. Bruegge A. H. Dutoit, *Object-oriented Software Engineering using UML, Patterns and Java*, Pearson Education International, NJ, 2004.

Software Availability

Through the University we have an academic license available for you to use VP in the lab or download onto your home computer. VP-UML can be run in the cse labs with the command /home/cs3111/vpapp/launcher/run_vpuml in a terminal window. Please read the license conditions at http://www.cse.unsw.edu.au/~cs3111/vpapp/. We also have copies of the software available on a local mirror at ftp://ftp.cse.unsw.edu.au/pub/users/cs3111/download/. The license file which you will need to install this at home is available through cse (lab machines remote login using 'scp' (Linux) or 'pscp.exe' (Windows)) at ~cs3111/licence/*.zvpl

Back to Table of Contents

Continual course improvement

Student feedback on this course, and on the lecturing in this course, will be gathered via questionnaires held at or after the end of the course. Student feedback is taken seriously, and continual improvements are made to the course based in part on this feedback. The course questionnaire results go to the Head of the School of Computer Science and Engineering, who reads the results and follows up in cases where action is clearly needed.

Further information

- Students enrolled in COMP3111 and COMP 9008 are expected to attend all classes
- The use of School of Computer Science and Engineering computing laboratories is subject to rules described in the Yellow Form, which you acknowledge (electronic) receipt of when you receive your computing account.
- The Yellow Form also states the supplementary assessment policy and outlines what to do in case illness or misadventure that affects your assessment, and supplementary examinations procedures within the School of Computer Science and Engineering. While there is no final examination for this course, the policy will be enforced for all assessments during the session, with suitable modifications.
- Information on UNSW Occupational Health and Safety policies and expectations
- Equity and Diversity issues
- Students are reminded that the university regards academic misconduct as a very serious matter. Students found guilty of academic misconduct are usually excluded from the university for 2 years. Because of the circumstances in individual cases, the period of exclusion can range from one session to permanent exclusion from the university.

Back to Table of Contents

Document created by:
Arcot Sowmya and Nadine Marcus
Phone: 9385 6933 and 9385 5173
E-mail: {sowmya,nadinem}@cse.unsw.edu.au

CRICOS Provider Number: 00098G
Last updated: