DEPARTMENT OF COMPUTER SCIENCE AND SOFTWARE ENGINEERING

COMP 5541 Tools and Techniques for Software Engineering

Winter 2017

Course Info

Course Instructor:	Ali Jannatpour			
Email:	alij@encs			
Office:	EV-3.301, by appointments			
Schedule:	-T-----	17:45-20:15	H-429 SGW	
Labs:	-T-----	20:30-22:10	H-849 SGW (DDDI)	Nader Kesserwan
Labs:	-T-----	20:30-22:10	H-847 SGW (DDDJ)	Nidhi Arora

Course Web Page

This information sheet gives important technical data about the course, which may be subject to change during the semester. Information about the course, assignments, important deadlines, and updates are kept on the course web-page for the winter 2016 term:

http://users.encs.concordia.ca/~alij/comp5541

Course Objective

Through these three types of instruction you are taught the different perspectives of Software Engineering discipline: basic principles, formalisms, tools and teamwork. You will be learning a disciplined process of developing software and practicing it in a small project.

You should expect to average a total of 10-12 hours per week on this course. For individual weeks, it will be much higher depending on your role in the project and the phase of the project. So, plan your time wisely.

Recommended Text-Books

- Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and the Unified Process, by Craig Larman, Prentice-Hall.
- Roger Pressman, Software Engineering: A Practitioner's Approach, McGraw-Hill Education;

Course Materials

- Some lecture notes will be posted
- Taking notes during the lecture
- Additional online books and papers will be made available to supplement lecture notes throughout the semester

Attributes

As part of both the Computer Science and Software Engineering program curriculum, the content of this course includes material and exercises related to the teaching and evaluation of graduate attributes. Graduate attributes are skills that have been identified by the Canadian Engineering Accreditation Board (CEAB) and the Canadian Information Processing Society (CIPS) as being central to the formation of Engineers, computer scientists and information technology professionals. As such, the accreditation criteria for the Software Engineering and Computer Science programs dictate that graduate attributes are taught and evaluated as part of the courses. This particular course aims at teaching and evaluating 3 graduate attributes. The following is a description of these attributes, along with a description of how these attributes are incorporated in the course.

A Knowledge Base for Engineering: Demonstrated competence in university level mathematics, natural sciences, engineering fundamentals, and specialized engineering knowledge appropriate to the program. The course will expand students ability to integrate and use specialized engineering knowledge to the program in the form of programming knowledge, standard mathematical skills for cost and time estimation, testing knowledge and project management knowledge.

Problem analysis: An ability to use appropriate knowledge and skills to identify, analyze, and solve complex engineering problems in order to reach substantiated conclusions. The project in this course is defined in such a way that requires the students to analyze the problem at hand before and determine for themselves exactly what needs to be done, and then determine how and with the help of what tools and software libraries it can be achieved.

Design is the ability to design solutions for complex, open-ended engineering problems and to design systems components or processes that meet specified needs with appropriate attention to health and safety risks, applicable standards, and economic, environmental, cultural and societal considerations. The project in this course is presented in an open-ended fashion, and its size and complexity is such that it needs to be tackled in teams of 4-5. The individual assignments provide a platform for designing at a smaller level, and provide the additional difficulty of having to be integrated in the larger design of the project.

Use of Engineering tools is the ability to create, select, apply, adapt, and extend appropriate techniques, resources, and modern engineering tools to a range of engineering activities, from simple to complex, with an understanding of the associated limitations. The course teaches the use of the Java language, and leaves the students free to select what programming environment and libraries that they will use in the assignments and project. Selection and use of the right tools and libraries is a crucial aspect of accomplishing the practical work. Also the use of Computer Aided Software Engineering tools will be an integrated part of the course content.

Individual and Team Work: An ability to work independently and as a member and leader in diverse teams and in multi-disciplinary settings. The course project is such that it needs to be tackled in teams of 4-5. Students will cover a complete project development lifecycle, with project leader responsibilities rotating throughout the project. Project tasks will be assigned throughout the project to individual team members, which will require periodical integration of individual tasks into the common project context.

Communication Skills: An ability to communicate complex engineering concepts within the profession and with society at large. Such abilities include reading, writing, speaking and listening, and the ability to

comprehend and write effective reports and design documentation, and to give and effectively respond to clear instructions. Students will have to document their project progress throughout the project lifecycle by creating different types of software documents, such as requirements, design and testing documentation.

Economics and Project Management: An ability to appropriately incorporate economics and business practices including project, risk and change management into the practice of engineering, and to understand their limitations. Students will learn key aspects of risk management, cost estimation, and general project management skills.

Evaluation Scheme

Tests	45%
Project	45% (3 iterations, each 15%)
Individual Performance	10%

The course grade is based on a clear pass (50%) in each of (i) the tests, and (ii) project work. The project work is evaluated in all three phases — based on the deliverables (meeting deadlines and quality levels appropriate to a student project), reviews (class presentations and discussions), and the final demonstration of the product (completeness with respect to the requirements, error free code, etc;). The entire team is awarded points in the project. Your grade will depend on both your performance in the tests and your performance in the project. A poor performance in any single component may bring down your grade. There is no simple direct correlation between your total mark and the grade.

The Individual Performance consist of evaluating the individual performance of each team-member with regards to his/her duties in the team, as well as the lab assignments.

Project

The project objective is to provide you with an opportunity to complete a software development process to experience the various SE activities first hand. Be realistic in terms of the scope of the project that you tackle, and the amount of time you devote to the project. It is better to work carefully and methodically throughout the term, than to attempt a last minute all-nighter. Details will be announced.

The project detail will be given out during week two, after which the teams are formed and finalized. Each team consists of various roles, which will be assigned on a rotation basis.

The Project will be completed in an iterative approach. There will be three iterations. The details of the artifacts / deliverables will be announced during the assignments. The goal of the project is to practice the SDLC that is taught throughout the class, using the UP in practice.

There will be dedicated presentation sessions, during which, each team will present their work to the class.

The dates will be announced via the course web site.

Schedule

W1	Course overview, Introduction to Software Engineering
W2	SDLC, Process Models
W3	Requirements Analysis, Use Cases
W4	Domain Model, Specification
W5	Sequence Diagram
W6	Software Design
W7	MIDTERM
W8	Design Patterns
W9	GRASP (General Responsibility Assignment Software Patterns / Principles)
W10	Architectural Views
W11	“
W12	Software Testing
W13	Final Presentations + Exam Review

The order of the lectures may change due to the presentations and the lab assignment.