CSC 444: Software Engineering I
Syllabus 2017

Software engineering covers engineering disciplines that are applied to computer software production and maintenance.

Traditional software engineering courses usually focus on subdisciplines related to the different stages of the "*waterfall model*" used by many, especially larger companies for developing software, including IBM, BCE, Rogers, Telus, CGI, and any of the big banks or insurance companies. The disciplines often covered in these traditional courses include requirements analysis, system modeling, architectural design, software testing, software maintenance, software reuse, project planning, project management, and quality management.

This course will be different. It will primarily focus on Agile software development and management, as used by modern companies, such as Facebook, Netflix, and Etsy. As opposed to teaching in the abstract, all of the material taught will be practiced in the context of a larger programming project that students will work on in groups. Most of the time spent working on this course will be spent working on the group project. Some of the languages that will have to be used in conjunction with the project are Ruby, Rails, HTML, DOM, CSS, JavaScript, JSON, JSAPI, and JQuery.

Teaching Staff

Michael Stumm	Instructor	stumm at eecg.toronto.edu
Jonathan-F Baril	TA	jonathanf.baril at mail.utoronto.ca
Hao Wang	TA	haoece.wang at mail.utoronto.ca

Course Structure

Lectures	3 hours / week
Tutorials	1 hour / week
Labs	3 hours / 2 weeks

Lectures

Friday 3-6pm BA 1210 Starts September 8

Lectures are used to teach fundamental concepts of agile software engineering and much of the material needed for successfully completing the group project. Please see full schedule further below.

(Note that I did not chose these lecture times...)

Tutorials

TUT 01 Monday 10-11am BA2165
TUT 02 Wednesday 12noon-1pm AP120

Tutorials are used to go over the material taught in the lectures. At the beginning of the term, tutorials may be used to teach new material related to one of the programming languages so as to allow faster start on group project work.

The precise schedule of these tutorials will be announced in class.

Labs

Tuesday 9-12noon GB251 Starts September 12

Although scheduled on a two-week schedule, students may attend any and all of these lab sessions. TAs will be available throughout to answer questions and assist in making the group project a success.

Mark Composition

Project Mark: 25%

Midterm: 35%

Final Exam: 40%

Note that the 25% of the mark being assigned to the project does not correspond to the workload associated with the project. Much more time and effort will have to be spent on the project than on anything else. Therefore, a significant part of the midterm and final exams will be dedicated to examining students' proficiency of material learned as part of the group project.

Textbook

There is no required textbook for this course. All of the required reading is available online, as are all of the required manuals. Excellent tutorials relevant to this course are also available online. Nevertheless, although not necessary or required, students may be interested in purchasing one of the excellent books available on Ruby and Rails.

Lecture Schedule

Please note that this schedule will be modified and updated during the term.

Week 1: September 8	Introduction to the course Software Engineering the course project Ruby basics
Week 2: September 15	Regular expressions More on Ruby Introduction to Model View Controller Frameworks Introduction to Rails
Week 3: September 22	More on Rails REST Version control and git Continuous release and continuous deployment
Week 4: September 29	Frontend languages I (HTML, DOM, CSS, Javascript) Responsive web design
Week 5: October 5	Frontend Languages II (JSON, JSAPI, JQuery) Sessions AJAX
Week 6: October 13	Programming in the large Behavior-driven design Test-driven design
Week 7: October 20	SOLID Midterm
Week 8: October 27	Software testing

Week 9: November 3	No Lecture Work on project
Week 10: November 10	Software Quality Software Metrics Code reviews Code smells UML
Week 11: November 17	No Lecture Work on project
Week 12: November 24	Traditional software engineering practice using the waterfall model Software maintenance
Week 13: December 1	6 hour session for project presentations