CS 580 Advanced Software Engineering
Course Syllabus – Fall 2018

Instructor	Dr. Yu Sun
Email	yusun@cpp.edu
URL	http://yusun.io
Phone	909-869-3449
Office	8-13
Office Hours	Tuesday and Thursday 8am - 10am; or email me for an appointment
Course Session	Tuesday and Thursday 2:30pm - 3:45pm, 8-348
Course URL	Blackboard

Course Focus

This course offers a survey and training of software engineering (SE) research and practice. This is a graduate course, and is designed to complement some of the things that you may have learned in an undergraduate SE course. The course will aim to strike a balance between topics that are of a research focus, and topics that are more practical to software development in the industry today. In this course, you will read some of the most classic papers of SE and also learn the trending SE approaches, tools, skills and practices that will aid you in the ability to construct better software.

Prerequisite

You should be comfortable writing code in Java. In addition, if you have not had the equivalent prerequisite of CS 4800, you are advised to take them first.

Textbooks

I will hand out some papers in class for you as reading assignments. No textbook is required, although I will list some recommended books that are helpful for the key topics related with this course.

All the key course content will be documented in slides, which will be available in the course website after each lecture.

Course Work

There will be NO midterm or final exams for this course.

Team Project. Students will be divided into teams (3-4 students per team) in the beginning of the course and work on a course project through the course. The project will be based on a web service application (the project idea will be determined by each team). The project will be demonstrated in class at the end of the quarter.

Assignments. In order to better evaluate the progress of the course project, 10 homework assignments will be given during this course as 10 project milestones (all the assignments contribute to your project, rather than being unrelated course work). The 10 assignments are designed to cover the key topics of this course, which allow you to practice the techniques demonstrated in the course and reflect on their effectiveness. By doing the assignments, you will experience and practice the important SE theories, principles, tools, and methods, which is very similar to the common work you are expected to accomplish as a software engineer in the software industry today.

The weights for grades are as follows:

- Assignments (50%)
- Project (50%)

Final grades will break at 90(A), 85 (B+), 80(B), 75 (C+), 70(C), 65, (D+), 60(D).

Honor Statement

Each student is to do his or her own work. This means that you are not to seek out the help of other students (or give help, if asked) in order to solve specific problems of your homework assignments. It also means that you should not sign up for mailing lists and ask for detailed help from others on the Internet. Of course, you may discuss generalities about an assignment with your fellow students. If you are unsure of what is permitted, in terms of discussing an assignment problem, please ask me for clarification.

Disabilities

If you have any disability that would put you at a disadvantage in performing an assignment, please meet with me privately to discuss ways in which I can assist you as you perform the required work in this course.

Tardiness

You are expected to arrive on time so that you do not cause a disruption in the middle of class. I would like to start the class at the scheduled time. If you cannot make it on time for some reason, please let me know. Persistent tardiness will be noted.

Tentative Schedule

This schedule and the order of the topics/assignments are subject to change.

Week	Day	Date	Topic
1	Thu.	8/23	Course Introduction
2	Tue.	8/28	Requirement Analysis
2	Thu.	8/30	No Silver Bullet
3	Tue.	9/4	Web Service Foundation
3	Thu.	9/6	Team Idea Pitch
4	Tue.	9/11	User Interface and User Experience
4	Thu.	9/13	Version Control - Introduction to Git
5	Tue.	9/18	Build Automation - Maven
5	Thu.	9/20	Agile Development: Introduction to Scrum
6	Tue.	9/25	Scrum: User Stories and Task Estimation
6	Thu.	9/27	Unit Test - jUnit
7	Tue.	10/2	Maven + jUnit + Test Coverage
7	Thu.	10/4	Code Review
8	Tue.	10/9	Software Deployment
8	Thu.	10/11	Continuous Integration
9	Tue.	10/16	Docker Container
9	Thu.	10/18	Software Design - Transparency
10	Tue.	10/23	Software Design - On the Criteria
10	Thu.	10/25	Clean Code I
11	Tue.	10/30	Clean Code II
11	Thu.	11/1	Software Maintenance I
12	Tue.	11/6	Software Maintenance II
12	Thu.	11/8	Software Scalability I

13	Tue.	11/13	Software Scalability II
13	Thu.	11/15	Test-Driven Development
14	Tue.	11/20	Software Publishing and Demo
14	Thu.	11/22	No Class - Thanksgiving Day
15	Tue.	11/27	Project Demo - I
15	Thu.	11/29	Project Demo - II
16	Tue.	12/4	Tech Interviews Explained
16	Thu.	12/6	Course Summary

Course Readings

The following papers will be handed out to you in class as reading assignments.

[Brooks86] Fred P. Brooks. (1986). "No Silver Bullet — Essence and Accident in Software Engineering". Proceedings of the IFIP Tenth World Computing Conference: 1069–1076.

[Gelperin88] David Gelperin and Bill Hetzel, "The Growth of Software Testing," Communications of the ACM, vol. 31, no. 6, June 1988, pp. 687-695.

[Mens04] Tom Mens and Tom Tourwé, "A Survey of Software Refactoring," IEEE Transactions on Software Engineering, vol. 30, no. 2, February 2004, pp. 126-139.

[Parnas72] David L. Parnas. (1972). "On the Criteria to be Used in Decomposing Systems into Modules". Commun. ACM 15, 12 (December 1972), 1053-1058.

[Parnas75] David L. Parnas and Daniel P. Siewiorek. (1975). "Use of the Concept of Transparency in the Design of Hierarchically Structured Systems". Commun. ACM 18, 7 (July 1975), 401-408.

[Zhu97] Hong Zhu, Patrick Hall, and John May, "Software Unit Test Coverage and Adequacy," ACM Computing Surveys, vol. 29, no. 4, December 1997, pp. 367-427.

The following books are recommended readings. They are NOT required textbooks.

[GoF95] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley, 1995.

[Martin08] Robert Martin, Clean Code: a Handbook of Agile Software Craftsmanship, Pearson Education, 2008.