

Software Engineering I

CS 341 - Fall 2013

Instructor: Justin Miller
Assistants: Kimberly Barth, Samuel Micka
Course Email: cs341@uwosh.edu

Justin Office Hours:	TR	7:00-8:00 AM	HS 214
Samuel Office Hours:	M	11AM - 12:30 PM	HS 213
	T	3-4:30 PM	HS 213
Kimberly Office Hours:	W	8:00-9:30 AM	HS 213
	R	11:15am – 12:45pm	HS 213

Labs:	T	8:00-9:30 AM	HS 101C
Lectures:	R	8:00-9:30 AM	HS 260

Prerequisites: CS262 with a grade of C or better *and* junior-level standing.

Textbook: There is no required textbook for this course, but here are some classical references:

1. *Object-Oriented Software Engineering*; Stephen R Schach, 1st Edition.
2. *Software Engineering: Modern Approaches*; Eric J. Braude, Michael E. Bernstein, 2nd Edition
3. *Software Engineering: A Practitioner's Approach*; Roger S. Pressman, 7th Edition
4. *Software Engineering*; Ian Sommerville, 9th Edition

Course Website: UWO D2L (<http://www.uwosh.edu/D2L>)

Note: If you have special needs, please come and talk to me at the end of the first class.

Course Overview

Software engineering involves a full range of activities from initial inception of a software idea or definition of a problem, to installation and use of computer software by an individual or organization. This course is the first of a two semester course sequence; this first semester provides a broad overview of software engineering methods and focuses on writing good, clean code, requirements analysis and definition, software architecture, modeling, and design skills. The active learning experiences will include a team contribution to a software development project. The second semester follows a more project based approach.

The specific learning objectives for Software Engineering are listed below.

1. Describe the concepts and principles that guide software engineering practice.
2. Define the requirements engineering process and associated activities for software systems.
3. Define the analysis modeling process and associated activities for software systems.
4. Define the design engineering process and associated activities for software systems.
5. Describe testing strategies and techniques for software systems.
6. Describe architectural design, styles, and patterns for software systems.
7. Define the guiding principles of good user interface design.
8. Describe the use of component-level design in software engineering.

9. Describe the project management concepts applied to software engineering projects.
10. Apply software engineering principles and methods to produce a solution to a significant problem.
11. Select and apply appropriate analysis modeling practices and tools to a significant problem.
12. Select and apply appropriate requirements capture techniques and tools to a significant problem.
13. Select and apply appropriate design engineering techniques to a significant problem.
14. Design and implement an effective test strategy and plan.
15. Apply good user interface design principles to the development of a solution to a significant problem.
16. Identify appropriate situations for use of different software architectures.
17. Identify appropriate situations for use of components-based design.
18. Establish effective communication plans with clients and co-developers.
19. Conduct effective presentations for clients.
20. Conduct effective design reviews with co-developers.
21. Develop and adhere to project plans and schedules.
22. Seek timely assistance from others when individual efforts to resolve problems have been unsuccessful.
23. Establish an atmosphere of trust and mutual commitment between team members.

Course Grading Policy: Your grade for this course will be based on four components, namely exams, lab exercises, readings, and a team-based software engineering project. Your overall numerical grade for the course will be computed as the weighted sum of the component grades using the following weights:

Component	Weight
Exams (2)	20%
Labs (12)	24%
Readings (10)	10%
Team Project (based on deliverables, demos, and peer evaluations)	46%

Each component will have specific grading criteria and policies that will be described later.

Your letter grade for the course will be computed as follows:

Numerical Score	Grade	Numerical Score	Grade
>=92	A	72-78	C
90-92	A-	70-72	C-
88-90	B+	68-70	D+
82-88	B	62-68	D
80-82	B-	60-62	D-
78-80	C+	<60	F

While this overall grading scheme is fixed, I will be happy to discuss any issue you may have with individual grades. If you notice a mistake or have a question regarding a specific grade, please come and talk to me *as soon as possible*. Do not wait until the end of the semester to bring up grading issues.

If you miss a scheduled exam, you **may** be able to take a make-up exam provided you give the instructor a valid justification (in the form of a written excuse from a medical doctor or the Dean of Students Office) ahead of time if possible.

Team Project: Software engineering is a complex process, and it is impossible to teach it well strictly in a classroom environment. Even the best textbooks, case studies, or simulations fall short of preparing a learner for participation in the actual process of software engineering. Thus, as mentioned previously, this course will require the successful completion of a significant software engineering project as one of the components of your grade. Students will be organized into teams, and each team will choose a project, subject to some restrictions. The project will comprise 46% of your overall course grade, distributed among the different phases of your project. Each phase will have deliverables and deadlines.

You must commit to meeting all due dates and deadlines as stated both by the instructor and per your own team's plan. If you fail to meet a stated due date or deadline, you must commit to a definite date/time by which the item will be completed in writing to the instructor prior to the missed date/deadline. If you fail to comply with this requirement, you will receive no points for the missed item.

Since many of the activities you will be completing are based on prior steps, falling behind has a major impact on your learning outcome in the course. Please plan to meet all deadlines and due dates as stated, or email me a date/time by which a late item will be submitted. If you miss a due date without prior discussion and approval of the instructor, you will receive no points for the assigned item.

Calendar: Please see the course site for detailed information about weekly course activities, labs, projects, and related due dates.

Academic Honesty: Unless otherwise stated, all work must be that of a single student (the one whose name appears on the submission, that is). You may not "borrow" any piece of code or design of any length from someone else, unless you can live with a zero and the other potential academic sanctions of cheating (see [UWO Student Discipline Code 2007](#), Chapter UWS 14).

Attendance and Participation: You are expected to not only attend **every** class meeting but also to come **prepared** for and **participate** actively in it. I **strongly encourage you to ask any question** or raise any issue you have with the course either during or at the end of class, or during my office hours. I will also gladly meet with you by appointment.