

CSci 435/535: Software Engineering Syllabus

Where and when

- Class: 11:00-11:50 MWF, Jones Hall, Room 302
- Office hours: 1:00-2:30 MW, and by appointment. 140 McGloughlin-Street Hall

Instructor

- David Coppit (<http://www.cs.wm.edu/~coppit/>, coppit@cs.wm.edu)
- Research: software development methods, applied formal methods, component-based software development, software verification

Web page

- Course homepage: <http://www.cs.wm.edu/~coppit/csci435/>. Contains the definitive versions of assignments and such.

Description

- From the catalog: “The software life cycle. Software design methodologies. Testing and maintenance. Programming teams.”
- This course will provide an introduction to the discipline of software engineering. An emphasis will be placed on practical knowledge and experience, although some theoretical and historical material will be presented as well.
- It is assumed that students already have programming experience. This course will not devote much time to coding, debugging, or other basic software knowledge which the student should have acquired earlier in the curriculum. Instead, it will focus on the problems, design, techniques, and tools which are involved with the development of large software systems by groups of people.
- A significant portion of this course will be devoted to a project which you will complete outside of class. Note that you may need to devote significant time to the project during “crunch time” prior to a deadline.
- This course is cross-listed as CSci 535. If you are a graduate student with software experience, you are advised to take CSci 780 instead. Graduate students are the project managers, and are expected to perform more (and different) work than undergraduates.

Prerequisites

- CSci 312.
- Good individual programming skills.

- Initiative: I expect students of your caliber to not need hand-holding. “But you didn’t tell us to do that” only goes so far.

Topics

- Development Processes
- Project Management
- The Software Lifecycle
- Object-Oriented Analysis and Design
- Formal Methods

Books

Required

- *Software Engineering* (5th Edition) by Ian Sommerville (020139815X)
- *The Pragmatic Programmer: From Journeyman to Master* by Andrew Hunt and David Thomas (020161622X)

Suggested (Required for CSci 535)

- *The Mythical Man-Month: Essays on Software Engineering* (2nd Edition) by Frederick P. Brooks (0201835959)

Anyone who knows anything about software engineering has read this book. I strongly advise you to get it if you expect to write software for a living.

Course Structure

Classes

This course is about learning “book knowledge” along with “experiential knowledge”. Mondays and Wednesdays will be traditional lectures on SE topics. Friday class meetings will be related to the particular software project which we will be working on. (Details are in the document “Project Description” which will be given out Friday the 23rd.) Friday meetings may involve short informational presentations, inter-group communication and status reports, and elevation of issues.

Project

Unlike many group software classes in which groups of students all implement the same software, we will be working as one large team to develop a sizeable software system. The system will be developed incrementally as a series of milestones. Each student will receive a grade at the end of each milestone based on the points they have earned by resolving issues. (Details are in the document “Project Grading”.)

If the work for a milestone is not met, this will obviously affect student grades for that milestone. However, this should not necessarily penalize students in the next milestone.

To address this issue, managers will have to negotiate with the customer (i.e. the professor) to narrow the set of features or extend the milestone deadlines.

Management Organization

The class structure is as follows. The professor serves as the client, infrastructure manager, and division manager. The CSci 535 students serve as managers, reporting to the professor. The CSci 435 students are all developers. In addition to this overall structure, managers may group students working on related tasks, and select a member of the group to be a group leader. These groups are mainly meant to reduce communication and management overhead, and do not restrict developers from working with other people in other groups.

Project managers have several responsibilities. They will obviously manage the project: they will set milestones, plan development, giving technical assistance to developers, and resolve personnel issues. They will attend weekly manager meetings with the professor, and will run the Friday project meetings. They will submit weekly status reports.

Tracking Work

The work to be done will be managed using an issue tracking system. Both managers and developers can add issues to the tracking system, which can be anything: bugs, feature enhancements, development tasks, etc. After an issue is added, developers add themselves to the issue to indicate that they are working on it. Once an issue is resolved, any developer which “owns” it can set its status to “Waiting for Manager”. At this time the manager assigned to the task will review the work and certify it as complete by changing its status to “Closed”.

While the majority of issues will be resolved by one developer or a small group of developers, from time to time tasks will be given to everyone in the class as homework assignments. This gives everyone a chance to learn a particularly important topic. For example, every group will develop a requirements document as a homework assignment. The managers will select the best one being used for the project. In contrast, the implementation phase will have each group working on a different part of the overall system.

Generally speaking, managers will not complete any of the tasks for the project. Instead, they will create tasks to be completed by the undergraduate developers. They will also review and certify completed work. **Because points are assigned to tasks, the project managers will ultimately determine the project grade for each student in the class.** In addition to these responsibilities, graduate students will provide technical assistance, and will provide weekly updates to the professor. In the case when a manager must step in and complete the work for a task, the scoring methods will penalize the developers. (This is described in the document “Project Grading”.)

Managers will have their own task list which they (and the division manager) will use to track management related issues.

Grading

Point distribution (CSci 435)

- Homework: 15%. The homework assignments will be subject to a weighted average.
- Midterm exam: 15%
- Final Exam: 20%
- Project: 50%. There will be several project milestones, and your grades for the milestones may be subject to a weighted average. Each milestone grade is simply the “Performance Score” reported by issue tracker.

Point distribution (CSci 535)

- Midterm exam: 15%
- Final Exam: 20%
- Project: 25%. There will be several project milestones, and your grades for the milestones may be subject to a weighted average. Each milestone grade is the group grade reported by issue tracker.
- Management: 30%. Management grades will be given at each milestone based on weekly written reports, ability to manage developers, and ability to run the Friday meetings. These grades are subject to a weighted average.
- Developer Evaluation: 10%. At the end of the course, the developers will evaluate the managers.

The “No BS” rule

- The student gets 10% on short-answer questions in which the answer given is “No BS”. Note: not turning in a test or exam gets you a 0.

Letter grade

- $90\% \leq \text{grade} < +\infty \rightarrow A$
- $80\% \leq \text{grade} < 90\% \rightarrow B$
- $70\% \leq \text{grade} < 80\% \rightarrow C$
- etc.

... but the professor makes the final call on all grades.

Late work policy

- 20% of the points are lost per day (or portion thereof) that an assignment is late. Example: Due Monday at 5pm, and you turn it in 6pm Tuesday: -40%

Make-up policy

- Tests can be made up with approval of the instructor

- Your final exam schedule can only be altered by the Office of Student Affairs

Students who need accommodation

- Please see me after class or send email to set up a brief meeting.

Guidelines for class assignments

- The Honor Code is always in effect, unless the professor says otherwise
- External material: feel free to use the internet, books, etc. If you happen to find a solution to a problem, do yourself a favor and avoid it.
- Fellow students: Help each other with general questions, but don't give away the strategy or “secret” of a problem. Let the professor give hints if necessary.