Software Engineering Laboratory

Bulletin Description
Organization and scheduling of software engineering projects, structured programming, and design. Each team designs, codes, and debugs program components and synthesizes them into a tested, documented program product.

General Course Info
Term: FALL 2017
Department: COMP
Course Number: 523
Section Number: 001
Time: MWF, 1:25-2:40
In addition, there will be weekly team meetings with the professor and the client
Location: SN011
Website: http://www.cs.unc.edu/Courses/comp523-f16

Instructor Info
Name: Diane Pozefsky
Office: FB 146
Email: pozefsky@cs.unc.edu
Phone: 919 590-6117
Web: http://www.cs.unc.edu/~pozefsky
Office Hours: Open Door Policy

Textbooks and Resources
There are no required or recommended textbooks. Sakai will be used primarily for the returning of comments and grades and for submitting individual essays. Weekly comments will be on a google document shared with the team. All team deliverables will be submitted through the team’s website. All recommended or required readings will be available or referenced on the class website. All required external resources will be available without cost.

Course Description
The goal of this course is to teach the skills necessary for building a software product as a team. The lecture portion of the class will cover the broader picture of software engineering that includes a wide range of software development projects in terms of size, complexity, and criticality. The course carries EE (experiential education) and CI (communications
intensive) tags and is an APPLES course. It also meets the requirements for the graduate programming product requirement.

**Target Audience**
This course is intended for upper class majors with an interest in building software for practical use. Students are expected to have enough experience to be able to learn new software systems on their own and to design a system using techniques and principles learned in other courses. This is an ideal course for those interested in getting real world experience in building software and communicating with others.

**Prerequisites**
COMP 410 and 411 plus two additional programming COMP courses numbered 426 or higher. The most useful courses to already have are COMP 426 (web applications are the most common distributed system today) and COMP 521 (everyone needs a database).

**Goals and Key Learning Objectives**
At the end of the course, each student will have experienced all aspects of a software development project, including
- working with a client to define goals and priorities
- designing a system
- scheduling and planning a multi-person project
- effective communications
- running meetings
- writing technical documentation
- preparing web content
- writing and testing code
- deploying the system
- public presentations

**Disclaimer**
The professor reserves to right to make changes to the syllabus, including assignment and project due dates. These changes will be announced as early as possible and will be reflected on the course website. If there are discrepancies between this syllabus and the website, the website is considered the definitive information.

**Course Requirements**
The essence of the course is the faculty-coached team project. Teams of 2-4 students spend the semester negotiating, estimating, scheduling, specifying, coding, debugging, integrating, documenting and testing a substantial programming product. Each project has a real client that is expecting a completed project. Each document will be submitted to the professor at each sprint and will be revised based on comments and to reflect changes in the product that is being produced.

There will be a final exam this semester; it will be held according to the university schedule, at noon on Monday December 11.
There will be individual assignments given to cover the key concepts of the course that are not well reinforced through the project and to expose you to the literature in the field. Specific plan is to assign weekly readings and ask you to respond to a question about the readings on Sakai.

There are a lot of new things happening in the field of software engineering that you are not exposed to within the department's curriculum. Each team will present a technology that they are using that has not been taught in other classes. If there is no appropriate technology the team is using, the team and instructor will select an appropriate technology. Teams will give a 30-35 minute presentation to the class. Preparation for the presentation includes two required meetings with the instructor: a discussion about what is going to be covered two weeks before the presentation and a walkthrough at least 48 hours before the presentation. Topics will be selected significantly before the presentations in order to schedule an appropriate order.

The class will not meet on Friday. This time is available for team or client meetings as all students in the class have it blocked for the class.

Project grades are based on code, documentation, ambition, effort, teamwork, and accomplishment. There will be 7 grades given for the project; they are equally weighted.

Final presentations of the end product will be held on the second reading day. The class has gotten too large to have them all run during the final exam period and I have been unable to get approval to move the final exam. The final presentations will be 3-7 on Wednesday, December 13 in SN 011.

**Key Dates**

There will only be one in-class demo day: October 18 (the day before fall break). Client demos are required as part of sprints 2, 4 and 6.

Sprint completions for all but the last are at the team meeting (recitation) of the following weeks:

- Sprint 1 (requirements), September 11
- Sprint 2, September 25
- Sprint 3, October 9
- Sprint 4, October 23
- Sprint 5, November 6
- Sprint 6, November 27
- Completion, December 13

Required documentation for each phase is identified on the class website. Note that final code and documentation is due BEFORE the final presentations.

I will be scheduling a time between December 14 and 15 when I am doing the final testing of each project. At least one team member will be present when I do the testing. This
avoids the problem where a small bug prevents me from testing the bulk of the function or a poor user interface leaves me befuddled.

**Grading Criteria**

*Overall breakdown*

<table>
<thead>
<tr>
<th>Component</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project</td>
<td>75%</td>
</tr>
<tr>
<td>Technical Talk</td>
<td>10%</td>
</tr>
<tr>
<td>Readings</td>
<td>5%</td>
</tr>
<tr>
<td>Final Exam</td>
<td>5%</td>
</tr>
</tbody>
</table>

**Project**

I compute a single grade for the project for each sprint, based on the following percentages:

<table>
<thead>
<tr>
<th>Component</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process</td>
<td>30%</td>
</tr>
<tr>
<td>Code</td>
<td>40%</td>
</tr>
<tr>
<td>Documentation</td>
<td>30%</td>
</tr>
</tbody>
</table>

I will give you your grade and your feedback at the meeting.

I then apply an individual contribution multiplier for each person. This value is based on my observations as well as the evaluations by your client, any consultants, and your peers. The multiplier ranges from 0 to 1. Basically, I do not believe that you should be able to get a better grade than the product you produced and taking on more than your share is not always a sign of a good team member.

In order to get a 1, you must have

- met all role responsibilities (.1)
- attended and participated in all meetings (.1)
- committed to a reasonable amount of work (.2)
- delivered on your commitments (.2)
- helped team members when appropriate (.2)
- did not pre-empt work from others or denigrate their contribution (.2)

A few more details:

- Process includes whether you are interacting appropriately as a team, with me, and with your client. Are you addressing issues as they arise? It includes professionalism in your dealings with your client and your professor and whether you are usually on time with deliverables or habitually late. It includes your web site and the materials that you produce as steps to produce the other artifacts. I will be giving you process grades every week. Key considerations for the grade are whether you meet your milestones, you are adapting your process as you learn, and whether you are working well as a team.
- The code grade covers function, correctness and readability. The three components are equally important. Have you met the primary requirements? How
many bugs was I able to find? I will do a random review of the code that you produce. I expect to be able to understand it. This includes web pages as well as other code that you write.

- The documentation covers the formal deliverables: the functional spec, the design document, the user manuals, and the test plan. They are weighted equally. Remember that spelling, grammar, and readability are important; unreadable good content is not sufficient. Interim deliverables are part of the process grade.
- For the final presentation, your grade will include both content and style. The final presentation must include a video demonstration of your product. This assures that I can easily show your work to other people.

**Individual Reading Assignments**
There will be an assigned paper each week. The paper and the prompt will be posted on Sakai and the response will be posted inline on Sakai. The total points for the papers will be 50 points. You will be given the opportunity to earn 60 points. Answers are intended to be under 6 sentences. The intent is to introduce you to the classic literature of the field. Grading will be on a 4-point scale:

0 = not submitted
1 = read the title
2 = read the abstract
3 = skimmed the paper
4 = read and thought about the paper

No late responses will be accepted. Grades will be posted early the following week.

**Technology Talk**
Your grade will include both content and style. I will be looking to see if you understand what you are presenting and whether you are communicating well with your classmates. Topics for tech talks will be agreed upon by the team and the instructor after platforms have been selected. I prefer to let you cover a technology that you are losing, but I have no interest in 14 frameworks being presented. If you do not have a topic that is natural based on your project, we will negotiate a topic.

**Team Meetings**
Everyone is expected at every team meeting. I expect an email for planned misses and you need to contact a team member for unexpected misses or lates.

The agenda for each meeting:
1. What were you supposed to deliver
2. What did you deliver (demo)
3. Review of deliverables (**sprint only**)
4. What went right and what went wrong?
5. Brainstorming of solutions
6. Review of process (sprint only)

7. What is your plan for next week.

Sprint 1 deliverable is a well-defined functional spec with prioritized requirements, platform selection, and working hello world.

All other sprints are a complete cycle:
- deployed and documented code
- updated documentation (functional spec, design document, deployment instructions)

Course Policies

Attendance: While attendance is not taken in class, I expect student attendance at all peer presentations. Specifically, I expect you to be at demo presentations. For other classes, I only point out that there is no textbook in this class because the content is not available in any simple form. If you are interested in the content, you need to listen to lectures. We also have a number of outstanding not-to-be-missed outside speakers during the class.

The course final is given in compliance with UNC final exam regulations and according to the UNC Final Exam calendar.

Honor Code

Collaboration and peer-learning are necessary for team projects. Only the individual assignments are not to be done collaboratively. These are open book, open notes, and open network. My goal is to give you essays that require individual thought and reflection and the work must be that of the student. Directly taking text from other sources is not acceptable. Short excerpts from other sources may be quoted and properly cited.
Course Schedule

The following is a draft of the class schedule. The up-to-date schedule is posted on the web site. This schedule only covers class lectures, not assignment deliverables, which are detailed on the web site.

<table>
<thead>
<tr>
<th>Date</th>
<th>Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aug 23</td>
<td>Introduction</td>
</tr>
<tr>
<td>28</td>
<td>Client Presentations</td>
</tr>
<tr>
<td>30</td>
<td>Working with Clients (Sept 1 Client meetings)</td>
</tr>
<tr>
<td>Sept 4</td>
<td>Labor Day</td>
</tr>
<tr>
<td>6</td>
<td>Requirements and functional specs</td>
</tr>
<tr>
<td>11</td>
<td>tools for collaboration</td>
</tr>
<tr>
<td>Sprint 1</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>CI/CD</td>
</tr>
<tr>
<td>20</td>
<td>Test-driven development</td>
</tr>
<tr>
<td>25</td>
<td>Teams</td>
</tr>
<tr>
<td>Sprint 2</td>
<td></td>
</tr>
<tr>
<td>Oct 2</td>
<td>User experience</td>
</tr>
<tr>
<td>4</td>
<td>Accessibility</td>
</tr>
<tr>
<td>9</td>
<td>Design</td>
</tr>
<tr>
<td>Sprint 3</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Models and modeling</td>
</tr>
<tr>
<td>18</td>
<td>DEMOS</td>
</tr>
<tr>
<td>23</td>
<td>Testing</td>
</tr>
<tr>
<td>Sprint 4</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>TECH TALKS</td>
</tr>
<tr>
<td>Nov 1</td>
<td>TECH TALKS</td>
</tr>
<tr>
<td>6</td>
<td>TECH TALKS</td>
</tr>
<tr>
<td>Sprint 5</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>TECH TALKS</td>
</tr>
<tr>
<td>20</td>
<td>Ethics</td>
</tr>
<tr>
<td>27</td>
<td>Security</td>
</tr>
<tr>
<td>Sprint 6</td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>Privacy</td>
</tr>
<tr>
<td>Dec 4</td>
<td>Deployment</td>
</tr>
<tr>
<td>6</td>
<td>Other development models</td>
</tr>
<tr>
<td>11</td>
<td>FINAL (noon)</td>
</tr>
<tr>
<td>13</td>
<td>PRESENTATIONS (3 pm)</td>
</tr>
<tr>
<td>Completion</td>
<td></td>
</tr>
</tbody>
</table>

Last updated 8/23/2017