COURSE DESCRIPTION

Dept., Number    CSC 190    Course Title    Senior Project: Part I
Semester hours   2        Course Coordinator    Robert Buckley
URL (if any):    http://gaia.ecs.csus.edu/~buckley/

Catalog Description

The first of a two-course sequence in which student teams undertake a project to develop and deliver a software product. Approved project sponsors must be from industry, government, a non-profit organization, or other area. Teams apply software engineering principles in the preparation of a software proposal, a project management plan and a software requirements specification. All technical work is published using guidelines modeled after IEEE documentation standards. Oral and written reports are required. Lecture one hour, laboratory three hours. Prerequisite: Senior status, passed the WPE, at least a C- grade in CSC 130, CSC 131, and four additional 3-unit upper-division CSC courses that fulfill the major requirements (excluding CSC 192-195, 198, and 199), and full CSC or MATH/CSC major status.

Textbook

No textbook required.

References

Jon Fairclough (Editor), Software Engineering Guides, Prentice-Hall, 1996.


**Course Goals**

Upon successful completion of this course, students and their teams should have demonstrated the following:

1. Team members worked effectively with one another, sharing the workload and responsibilities.
2. The project team was effectively managed by the team’s project manager.
3. The project team met at least once weekly to review progress and to update the schedule and work plan. Time on each work assignment was collected, new assignments were made and meeting minutes were kept.
4. The project team met weekly with their faculty adviser, provided accurate status on work in progress and the team’s schedule. Documents were reviewed and discussed, and all changes to the team’s work plan and schedule were discussed and approved by the project sponsor.
5. Development work followed the team’s management plan, including the team’s work plan and schedule.
6. The project team implemented an effective quality assurance process. For example, all work products were formally and thoroughly reviewed and approved by the group prior to submission to the faculty adviser.
7. Meetings with project sponsor/users were well-planned and effectively managed with results and follow-up well documented.
8. The project sponsor was informed about project status and progress throughout the development of the software product.
9. All documents were approved by the team’s faculty adviser prior to conveyance to the project sponsor.
10. All changes to baseline products were processed according to the team’s documented change control process.
11. The completed set of product documents provided traceable “threads” from each specified
requirement, to its design element, to its corresponding product component, to the system test case used to validate the implementation of the requirement.

12. All team members demonstrated an understanding of the software development process and the appropriate application of software engineering principles.

13. A formal technical review – with the project sponsor – was scheduled prior to the end of the semester. At that time the team reviewed their understanding of software requirements and described the work to be done in CSc 191. The project sponsor signed an acceptance approval form for the SRS document. In so doing, the sponsor believed that the team’s understanding was sufficient to “build” the software that they needed.

Prerequisites by Topic

Thorough understanding of:

- Software development process models.
- Software system engineering.
- Requirements engineering.
- Structured and object-oriented analysis.
- Use of semi-formal modeling languages (e.g., UML).
- Design (IEEE standards, design and design documentation).
- Structured and object-oriented design.
- Coding and integration.
- Verification, validation, and testing.
- Managing a software project.
- Planning the project.
- Controlling the project.

Basic understanding of:

- Software development process models.
- Software system Engineering.
- Requirements engineering.
- Structured and object-oriented analysis.
- Use of semi-formal modeling languages (e.g., UML).
- Design (IEEE standards, design and design documentation).
- Structured and object-oriented design.
- Coding and integration.
- Verification, validation, and testing.
- Managing a software project.
- Planning the project.
- Controlling the project.

Exposure to:

- Software development process models.
- Software system engineering.
- Requirements engineering.
- Structured and object-oriented analysis.
- Use of semi-formal modeling languages (e.g., UML).
- Design (IEEE standards, design and design documentation).
- Structured and object-oriented design.
- Coding and integration.
- Verification, validation, and testing.
- Managing a software project.
- Planning the project.
- Controlling the project.

**Major Topics Covered in the Course**

1. Software engineering principles and lifecycles: management and process.
2. Technical writing and technical presentations
3. Overview of management functions.
4. Introduction to project management.
5. Project initiation (including cost estimation).
6. Planning: tasks, costs, and schedules.
8. Organizing: line and staff organizations, project, functional, and matrix organizations, project team structure.
9. Staffing and training.
10. Directing, motivating, leading.
11. Controlling a project – including configuration management, quality assurance, and reviews.
12. Quality assurance using formal “walkthroughs” and “inspections.”

**Laboratory Projects**

1. Prepare a project proposal document (3 weeks).
2. Prepare a project management plan document (4 weeks).
3. Prepare a software requirements specification document (8-10 weeks).

**Estimated Curriculum Category Content (Semester hours)**

<table>
<thead>
<tr>
<th>Area</th>
<th>Core</th>
<th>Advanced</th>
<th>Area</th>
<th>Core</th>
<th>Advanced</th>
</tr>
</thead>
<tbody>
<tr>
<td>Algorithms</td>
<td></td>
<td></td>
<td>Data Structures</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software Design</td>
<td></td>
<td>0.25</td>
<td>Prog. Languages</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Comp. Arch.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Oral and Written Communications

Each student team is required to prepare a minimum of three (3) technical documents in which all team members are involved in the writing, reviewing and revising of each document. Documents must be approved by the team’s faculty adviser. Each student team is required to make two technical presentations in which all team members share equally in the presentations. The first presentation is made in class and typically is 15 to 20 minutes in length. The second presentation is made to the team’s sponsor.

Social and Ethical Issues

No significant component, although the social and ethical issues discussed use case studies that represent current examples of software development problems and the difficulties associated with various software products.

Theoretical Content

No significant component.

Problem Analysis

Students are involved in problem analysis throughout the class; including requirements’ analysis and project planning.

Solution Design

Student involvement in design typically does not occur until the second semester of Senior Project (CSC 191).