CS471: Software Engineering
3-credit course with 3 lecture hours per week
Course Coordinator: Jim Conrad

Textbook(s) and Supplemental Material


Catalog Description

A formal study of the software development process. Topics include: life cycle models, requirements definition, specification, design, implementation, validation, verification, maintenance, and reuse. Students work in small teams on significant projects. Creation of teams and specifications to be realized in CS481.

PREREQ: CS230 and CS321.

Required

Goals for the Course

Successful students will be expected to:

- Understand the need for a rigorous approach to software development
- Understand software build-management and source code management techniques and tools
- Know widely used methods of documenting the requirements and design of a software system
- Understand defect removal activities including pair programming, test-driven development, unit-level testing, code reviews, static analysis, integration and regression testing, acceptance testing, non-functional testing, and beta testing
- Choose appropriate quality goals, construct a quality plan, and instrument the development process to assess whether a project is meeting is meeting its goals
- Understand scrum roles and work as a member of an agile development team
- Demonstrate use of current software development tools individually and on a team

Outcomes Addressed

a. An ability to apply knowledge of computing and mathematics appropriate to the discipline
b. An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution
d. An ability to function effectively on teams to accomplish a common goal
f. An ability to communicate effectively with a range of audiences
i. An ability to use current techniques, skills, and tools necessary for computing practice
j. An ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the trade-offs involved in design choices
k. An ability to apply design and development principles in the construction of software systems of varying complexity

Outcomes Assessed: b, d, i, and k

Topics Covered

Introduction
The Elements of the Scrum Development Process
Waterfall Life Cycle
Software Quality Goals, Definition of Done and Defect Removal Models
Requirements Analysis with User Stories and Use Cases
Controlling Software Projects
Pre-Integration Software Engineering Practices: Pair Programming, Unit-Level Testing, Test-Driven Development, Static Analysis, Code Reviews
Quality Planning, Test Coverage, Equivalence Partitioning, Boundary Value Analysis
Non-Functional Testing
Principles of Object-Oriented Analysis and Design, UML Class, Statechart and Sequence Diagrams
Selected Topics from Software Engineering Research

Grading
At the end of the course, a letter grade is assigned to each student according to rank among classmates, which is determined from numerical scores assigned for performance of these activities:

<table>
<thead>
<tr>
<th>Activity</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>Homework/Quizzes</td>
<td>20%</td>
</tr>
<tr>
<td>Exam</td>
<td>25%</td>
</tr>
<tr>
<td>Project</td>
<td>20%</td>
</tr>
<tr>
<td>Project Peer Assessment</td>
<td>10%</td>
</tr>
<tr>
<td>Final</td>
<td>25%</td>
</tr>
</tbody>
</table>

Curriculum Category Content (Credits)

<table>
<thead>
<tr>
<th>Area</th>
<th>Core</th>
<th>Advanced</th>
</tr>
</thead>
<tbody>
<tr>
<td>Algorithms</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software Design</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Computer Architecture</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data Structures</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Programming Languages</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>