COMPSCI 471: Software Engineering
3-credit course with 3 lecture hours per week
Course Coordinator: James Buffenbarger

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.

PREREQ: COMPSCI 342.

Required

Goals for the Course
Successful students will be expected to:

- understand software lifecycles and the need for a rigorous approach to software development
- understand software build-management and configuration-management techniques and tools
- know informal but systematic methods of documenting the requirements and design of a software system (e.g., UML)
- understand software-design methods, including procedure-oriented and object-oriented design
- know formal methods of specifying software
- have the experience of working as a member of a team during the early phases of a development project
- understand other contemporary topics

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
Software Management
Software Lifecycles
Configuration Management
Requirements Analysis
Software Design
Formal Specification
Automated Test-Driven Development
Aspect-Oriented Programming

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</td>
<td>20%</td>
</tr>
<tr>
<td>Exam</td>
<td>25%</td>
</tr>
<tr>
<td>Project</td>
<td>30%</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></td>
<td>3</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>