SE 3540 : INTRODUCTION TO SOFTWARE ENGINEERING

Semester Hours: 3.0  
Contact Hours: 3

Coordinator: Robert Green
Text: Head First Software Development
Author(s): DAN PILONE AND RUSS MILES
Year: 2007

SPECIFIC COURSE INFORMATION

Catalog Description:
Overview of software engineering as a discipline. Software life-cycle models and phases of the software development process. Introduction to Human Computer Interaction (HCI), user-centered development, teams and project management. Prerequisite: Grade of C or better in CS 2020.

Course type: REQUIRED

SPECIFIC COURSE GOALS

• I can understand key terms used when analyzing human interaction with software.
• I can analyze, specify and document software requirements for a software system.
• I can understand user interface design standards.
• I can express and understand the importance of professional ethics, etiquette, and communication in a software development environment.
• I can develop alternative design solutions to a given problem and recommend the best one within limitations of cost, time, knowledge, existing systems, and organizations.
• I can apply the process of project initiation, planning, execution, and management.
• I can analyze and compare various software development lifecycle methods that include requirements analysis, design, implementation, testing and maintenance.

COMPUTER SCIENCE STUDENT OUTCOMES ADDRESSED BY THIS COURSE

• CS 1 Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions
• CS 4 Recognize professional responsibilities and make informed judgments in computing practice based on legal and ethical principles
• CS 5 Function effectively as a member or leader of a team engaged in activities appropriate to the program’s discipline
• CS 6. Apply computer science theory and software development fundamentals to produce computing-based solutions

SOFTWARE ENGINEERING STUDENT OUTCOMES ADDRESSED BY THIS COURSE

• SE 1 An ability to identify, formulate, and solve complex engineering problems by applying principles of engineering, science, and mathematics
• SE 2 An ability to apply engineering design to produce solutions that meet specified needs with consideration of public health, safety, and welfare, as well as global, cultural, social, environmental, and economic factors
• SE 4 An ability to recognize ethical and professional responsibilities in engineering situations and make informed judgments, which must consider the impact of engineering solutions in global, economic, environmental, and societal contexts
• SE 5 An ability to function effectively on a team whose members together provide leadership, create a collaborative and inclusive environment, establish goals, plan tasks, and meet objectives
• SE 6 An ability to develop and conduct appropriate experimentation, analyze and interpret data, and use engineering judgment to draw conclusions
• SE 7 An ability to acquire and apply new knowledge, as needed, using appropriate learning strategies
LIST OF TOPICS COVERED

- Software Processes and Models
  - Software engineering concepts
  - SDLC, Process model
  - Agile Software Development
- Planning and Requirements Analysis
  - User and/or system requirements
  - Effort estimation
- Design and Development Methodologies
  - Human and Computer Interaction and Design
  - Team design
  - Architecture and design patterns
  - Internal and external design factors
  - Coding methods and guidelines
- Documentation, Testing and Evaluation
  - Standards and best practices
  - Testing methods
  - Assurance and acceptance criteria
  - Reliability and performance
- Project Management
  - Resources and configuration control
  - Risk analysis
  - Product integration
  - Best practices
  - Release management and source control
<table>
<thead>
<tr>
<th>Information Management (related) Topic</th>
<th>Description</th>
<th>Textbook Reference</th>
<th>Class Hours Estimate</th>
</tr>
</thead>
<tbody>
<tr>
<td>*Version Control Systems</td>
<td>How to organize files/folders for software development, notion of local vs remote repositories, merging, branching, etc.</td>
<td>Ch 6(^1)</td>
<td>2</td>
</tr>
<tr>
<td>*UML</td>
<td>Show modeling object-oriented classes using UML class diagram notation, including modeling relationships between classes; representation of collaborating objects using UML sequence diagrams</td>
<td>Ch 3, 4, &amp; 5(^2)</td>
<td>2</td>
</tr>
<tr>
<td>Relational database tables and simple queries</td>
<td>When introducing class diagrams, can also show how the data could be represented in a table structure. Show parallels between classes relationships (aggregation/composition) and table relationships. Show very simple SELECT queries for accessing data from relational tables.</td>
<td>None</td>
<td>&lt;1</td>
</tr>
</tbody>
</table>

\(^1\)Textbook: *Head First Software Development, by Miles and Pilone*  