

The University of Texas at Tyler
Master of Science in Computer Science

Course Syllabus

Course Number:	COSC 5370
Course Title:	Software Engineering
Course Description:	Software Engineering is the systematic development of software. Software is increasingly being used in a wide range of human endeavors but quality of software produced by the industry is questionable. This course teaches techniques and tools needed to engineer high quality software in a team environment. Students will learn object-orientation, project management, team formation, software processes, requirements elicitation, analysis, design, and issues in implementation and testing. Students will present papers on select topics in software engineering and student teams will work on term projects. Offered once every two years in the Fall of even years.
Pre-requisites:	COSC 2315, COSC 2336
Credits:	3 credit hours
Text(s):	<i>Object-Oriented Software Engineering</i> by Stephen Schach, 1 st Edition, McGraw Hill.
Languages Used: (if applicable)	Unified Modeling Language, Pseudo-Code, Java, C++
Topics:	<ol style="list-style-type: none"> 1. Scope of Object-Oriented Software Engineering 2. Software Life-Cycle Models 3. The Software Process 4. Teams 5. The Tools of the Trade 6. Testing 7. From Modules to Objects 8. Planning and Estimating 9. The Requirements Workflow 10. The Analysis Workflow 11. More on UML 12. The Design Workflow 13. The Implementation Workflow 14. Postdelivery Maintenance
Additional Materials:	

Evaluation Method: (only items in dark print apply)	
1. Examination/Quiz	2. Homework
3. Paper/Report	4. Computer Program
5. Project	6. Presentation
7. Class Participation	8. Peer Review

Course Objectives¹: By the end of this course students are expected to:
1. Identify important challenges in software engineering. [1, 3]
2. Select the most appropriate process for a software project. [1, 5]
3. Organize an appropriate team for completed a software project. [1, 5]
4. Develop software as part of a team. [4, 5]
5. Evaluate software artifacts using standard techniques. [4, 5, 6, 7]
6. Document software using industry standard approaches. [4, 5]
7. Present software artifacts to peers. [4, 5, 6, 7]
¹ Numbers in bracket refer to method(s) used to evaluate the course objective.

Relationship to Program Outcomes: (only items in dark print apply)² This course supports the following computer science graduate program outcomes, which state that our students at the time of graduation are expected to:
1. possess an enhanced breadth of knowledge in computer science, combined with a depth of knowledge in critical core areas of computing [1, 2, 4];
2. possess the skills and knowledge for lifelong learning in computer science [1, 4, 6, 7];
3. possess knowledge of the theoretical foundations of computing and have strong practical application experience [2, 3, 4, 5, 6, 7];
4. posses and demonstrate oral and written communication skills [6,7];
5. understand and respect the professional standards of ethics expected of a computer scientist and be knowledgeable concerning the history of computing field;
6. possess a knowledge of computer security and computer security management;
7. analyze and compare relative merits of alternative software design, algorithmic approaches, and computer system organization, with respect to a variety of criteria relevant to the task (e. g. efficiency, scalability, security); [2, 3, 4, 5]
8. Implement algorithms in multiple programming languages, on multiple hardware platforms, and multiple operating system environments.
² Numbers in brackets refer to course objective(s) that address the Program Outcome.

Prepared By: Nary Subramanian	Date:11/21/2008