CSE 3231: Software Engineering Techniques

Course Description

Software engineering issues, techniques, methodologies and technologies; software lifecycle activities: requirements analysis, architecture, design, testing, deployment, maintenance; project management; enterprise software systems; frameworks.

Prior Course Number: CSE 757
Transcript Abbreviation: Software Eng
Grading Plan: Letter Grade
Course Deliveries: Classroom
Course Levels: Undergrad
Student Ranks: Junior
Course Offerings: Autumn, Spring
Flex Scheduled Course: Never
Course Frequency: Every Year
Course Length: 14 Week
Credits: 3.0
Repeatable: No
Time Distribution: 3.0 hr Lec
Expected out-of-class hours per week: 6.0
Graded Component: Lecture
Credit by Examination: No
Admission Condition: No
Off Campus: Never
Campus Locations: Columbus
Prerequisites and Co-requisites: Prereq: 3901 or 3902 or 3903.
Exclusions: Not open to students with credit for 5231 (757).
Cross-Listings:

Course Rationale: Existing course.

The course is required for this unit's degrees, majors, and/or minors: Yes
The course is a GEC: No
The course is an elective (for this or other units) or is a service course for other units: Yes

Subject/CIP Code: 14.0901
Subsidy Level: Baccalaureate Course

Programs

Abbreviation	Description
BS CSE	BS Computer Science and Engineering

Course Goals

Be competent with structured and agile software engineering frameworks; specifically structured and agile software engineering methodologies for requirements identification, analysis, architecture, design, testing, deployment and project management
Be familiar with the characterization of enterprise software systems
Be familiar with frameworks for analyzing the business context of enterprise IT systems, the concept of Business-IT alignment and related issues, and Enterprise Architecture
Be exposed to the trends impacting enterprise systems

Be exposed to the need for frameworks for software engineering

Course Topics

Topic	Lec	Rec	Lab	Cli	IS	Sem	FE	Wor
Characteristics of enterprise softw. sys.: scale, heterogeneity, distribution, federation by nature, lack of complete knowledge; organizational challenges; external drivers	1.5							
Understanding the business and the relationship between the business and information technology - business strategy, business-IT alignment and enterprise architecture	3.0							
Software engineering process - broadly characterized as structured or agile processes. Scenario-driven, Incremental and iterative development. Introduction to work-products and work-product-oriented development. Agile principles	2.5							
Requirements gathering. Structured and agile requirements work-products	4.0							
Analysis - domain, problem and solution analysis. Exposure to UML. Structured and agile analysis work-products. CRC-card based analysis	4.0							
Architecting softw. intensive sys: Designing, evaluating architectures; non-functional requirements & quality attributes in arch. Quality-driven design. Structured & agile architecture work-products	6.0							
Software project management: Structured and Agile project planning and management, linear and parametric software estimation, Risk planning. Software configuration management. Agile boot camp ? LEGO-based workshop on Agile development	6.0							
Software design: Responsibility-driven design concepts, application of responsibility-driven design in design patterns and enterprise technology frameworks, designing applications using enterprise technology frameworks	6.0							
Testing: Testing methodologies for enterprise systems. Testing in agile methodologies	4.0							
Deployment, Maintenance and Management: IT Infrastructure Library (ITIL) practices for infrastructure management	2.0							
Case studies in software engineering	2.5							

Representative Assignments

A small-team project that begins with the analysis of a business to identify where a software system might be of value, followed by the requirements identification, analysis and design of this system on paper.

Grades

Aspect	Percent
Class participation	10%
Workshop participation	10%
Team Project	30%
In-class Quizzes	30%
Final examination	20%

Representative Textbooks and Other Course Materials

Title	Author
Software Engineering	Sommerville
Software Engineering, A Practitioners Approach	Pressman
Developing Object-Oriented Software, An Experience-Based Approach, 1997.	Object-Oriented Technology Center, IBM

ABET-EAC Criterion 3 Outcomes

Course Contribution		College Outcome
**	a	An ability to apply knowledge of mathematics, science, and engineering.
	b	An ability to design and conduct experiments, as well as to analyze and interpret data.
**	c	An ability to design a system, component, or process to meet desired needs.
*	d	An ability to function on multi-disciplinary teams.
**	e	An ability to identify, formulate, and solve engineering problems.
*	f	An understanding of professional and ethical responsibility.
*	g	An ability to communicate effectively.
*	h	The broad education necessary to understand the impact of engineering solutions in a global and societal context.
*	i	A recognition of the need for, and an ability to engage in life-long learning.
*	j	A knowledge of contemporary issues.
**	k	An ability to use the techniques, skills, and modern engineering tools necessary for engineering practice.

BS CSE Program Outcomes

Course Contribution		Program Outcome
*	a	an ability to apply knowledge of computing, mathematics including discrete mathematics as well as probability and statistics, science, and engineering;
	b	an ability to design and conduct experiments, as well as to analyze and interpret data;
**	c	an ability to design, implement, and evaluate a software or a software/hardware system, component, or process to meet desired needs within realistic constraints such as memory, runtime efficiency, as well as appropriate constraints related to economic, environmental, social, political, ethical, health and safety, manufacturability, and sustainability considerations;
*	d	an ability to function on multi-disciplinary teams;
**	e	an ability to identify, formulate, and solve engineering problems;
*	f	an understanding of professional, ethical, legal, security and social issues and responsibilities;
*	g	an ability to communicate effectively with a range of audiences;
*	h	an ability to analyze the local and global impact of computing on individuals, organizations, and society;
*	i	a recognition of the need for, and an ability to engage in life-long learning and continuing professional development;
*	j	a knowledge of contemporary issues;
**	k	an ability to use the techniques, skills, and modern engineering tools necessary for practice as a CSE professional;
**	l	an ability to analyze a problem, and identify and define the computing requirements appropriate to its solution;

Course Contribution		Program Outcome
**	m	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 tradeoffs involved in design choices;
**	n	an ability to apply design and development principles in the construction of software systems of varying complexity.

Additional Notes or Comments

Means for Achieving Learning Outcomes

Inverted Classroom: Lectures have been pre-recorded and accessible online. Students are expected to familiarize themselves with the lecture and prepare lecture notes before they come to class.

Active Learning: Software Engineering has traditionally been a difficult subject to teach in a purely lecture and examination based format. In this class we will attempt to teach SE through in-class activities and discussion, and a project.

Class discussions: You are expected to participate in discussions about the lecture material as well as the research presentations

Quizzes: There will be 15-minute quizzes at the beginning of each course segment. Students may use their prepared notes on the lecture to answer the quiz. Students may not refer to the actual lecture slides or the lecture itself for the quiz.

Workshop: In-class workshops will be used to provide hands-on experiences on certain software engineering topics such as Agile development.

Project: There will be an integrative paper project. Class sessions will be reserved to work on the project. However, students will need to reserve time outside of class to work on the project

Course Policies

The following course policy applies to the classroom and team interactions you are expected to show the same responsibility to your teammates as you do to me.

 Attendance: You are expected to attend all classroom sessions and team meetings, and do all the assigned work, self-study and readings.
 Class preparation: You are expected to be prepared for class, participate in the discussion, answer questions, etc., on the topic for the day.
 Missed classes, exams, presentations, and meetings: You are responsible for all class lectures, including handouts and notes. There will be no make up exams, presentations, lectures, etc.
 Assignments: Assignments (if any) are duein hardcopyat the beginning of class on the due date. Assignments must be typed and formatted appropriately.
 Workshops: We may reserve a weekly time outside of class for collaborative hands-on workshops.
 Contact and class information: The course web site (see above) and email communication will be extensively used and relied upon for this course. Please familiarize yourself with these resources, provide me with your email address in the questionnaire and check

your email at least once every day.
 Sharing and attribution of intellectual property and information: You are free to exchange and use any information from each others projects. You may also freely research and use information legally available from the Web or other sources. However, you must properly attribute each piece of borrowed intellectual property.

Prepared by: Bruce Weide