Knight Foundation School of Computing and Information Sciences

Course Title: Software Engineering II **Date:** 05/27/2020

Course Number: CEN 4021

Number of Credits: 3

Subject Area: Software Engineering	**Subject Area Coordinator:** Monique Ross **email:** moross@fiu.edu
Catalog Description: Issues underlying the successful development of large scale software projects: Software Architectures; Software Planning and Management; Team Structures; Cost Estimation	
Textbook: • *Agile Project Management*, Second Edition, by Jim Highsmith. • *The Software Project Manager's Handbook*, Second Edition, by Dwayne Phillips	
References: The Mythical Man-Month: Essays on Software Engineering, Addison-Wesley Pub. The CHAOS Report: http://www.projectsmart.co.uk/docs/chaos-report.pdf COCOMO II Definition Manual: http://sunset.usc.edu/research/COCOMOII/Docs/modelman.pdf	
Prerequisites Courses: CEN 4010	
Corequisites Courses: None	

Type: Elective for Computer Science Track (Applications group);
Required for Software Design and Development Track

Prerequisites Topics:

- Software Development Life Cycle
- Requirements specifications
- Software Design and Implementation

Course Outcomes:

1. Master techniques of planning and monitoring the progress of a software project
2. Master software project cost estimation techniques
3. Be familiar with software architectures
4. Be familiar with software development team structures

Relationship between Course Outcomes and Program Outcomes

BS in CS: Program Outcomes	Course Outcomes
a) Demonstrate proficiency in the foundation areas of Computer Science including mathematics, discrete structures, logic and the theory of algorithms	
b) Demonstrate proficiency in various areas of Computer Science including data structures and algorithms, concepts of programming languages and computer systems.	
c) Demonstrate proficiency in problem solving and application of software engineering techniques	1, 2, 3, 4,
d) Demonstrate mastery of at least one modern programming language and proficiency in at least one other.	
e) Demonstrate understanding of the social and ethical concerns of the practicing computer scientist.	
f) Demonstrate the ability to work cooperatively in teams.	1, 4
g) Demonstrate effective communication skills.	1, 4

Assessment Plan for the Course & how Data in the Course are used to assess Program Outcomes

Student and Instructor Course Outcome Surveys are administered at the conclusion of each offering, and are evaluated as described in the School's Assessment Plan: http://www.cis.fiu.edu/programs/undergrad/cs/assessment/

Knight Foundation School of Computing and Information Sciences
CEN 4021
Software Engineering II

Outline

Topics	Number of Lecture Hours	Outcome
• Software Processes o Software lifecycle, Waterfall, Agile	3	1
• Software Project Management o Team participation ➢ Team processes including responsibilities for tasks, meeting structure, and work schedule ➢ Roles and responsibilities in a software team ➢ Team conflict resolution ➢ Risks associated with virtual teams (communication, perception, structure) o Effort Estimation (at the personal level) o Risk (cross reference IAS/Secure Software Engineering) ➢ The role of risk in the lifecycle ➢ Risk categories including security, safety, market, financial, technology, people, quality, structure and process o Team management ➢ Team organization and decision-making ➢ Role identification and assignment ➢ Individual and team performance assessment o Project management ➢ Scheduling and tracking ➢ Project management tools ➢ Cost/benefit analysis o Software measurement and estimation techniques o Software quality assurance and the role of measurements o Risk ➢ Risk identification and management ➢ Risk analysis and evaluation ➢ Risk tolerance (e.g., risk-adverse, risk-neutral, risk-seeking) ➢ Risk planning o System-wide approach to risk including hazards associated with tools	21	1,2,4
• Tools and Environments o Software configuration management and version control o Release management o Programming environments that automate parts of program construction processes (e.g., automated builds) ➢ Continuous integration o Tool integration concepts and mechanisms	6	1
• Software Design o Software architecture concepts and standard architectures (e.g. client-server, n-layer, transform centered, pipes-and-filters) o The use of components in design: component selection, design, adaptation and assembly of components, components and patterns, components and objects (for example, building a GUI using a standard widget set)	8	3
• Software Construction o Integration strategies: top-down, bottom-up, sandwich	2	1

Knight Foundation School of Computing and Information Sciences
CEN 4021
Software Engineering II

Course Outcomes Emphasized in Laboratory Projects / Assignments

Outcome	Number of Weeks
1. Software Project Charter Outcomes: 1	3
2. Software Project Cost Estimate Outcomes: 2	3
3. Schedule Tracking: Outcomes: 2	3

Oral and Written Communication:

Written Reports		Oral Presentations	
Number Required	Approx. Number of pages for each	Number Required	Approx. Time for each
3	20	2	20-25 minutes per group (5 minutes per student)

Social and Ethical Implications of Computing Topics:

No significant coverage

Topic	Class time	Student Performance Measures

Knight Foundation School of Computing and Information Sciences
CEN 4021
Software Engineering II

Approximate number of credit hours devoted to fundamental CS topics

Topic	Core Hours	Advanced Hours
Algorithms:		
Software Design:		0.5
Computer Organization and Architecture:		0.5
Data Structures:		
Concepts of Programming Languages:		

Theoretical Contents

Topic	Class time

Problem Analysis Experiences

Software Project Charter

Solution Design Experiences

Knight Foundation School of Computing and Information Sciences
CEN 4021
Software Engineering II

The Coverage of Knowledge Units within Computer Science Body of Knowledge[1]

KA	Knowledge Unit	Topic	Lecture Hours
SE	Software Processes	• Software lifecycle, Waterfall, Agile Outcome: Core-Tier1 #3	3
SE	Software Project Management	• Team participation ○ Team processes including responsibilities for tasks, meeting structure, and work schedule ○ Roles and responsibilities in a software team ○ Team conflict resolution ○ Risks associated with virtual teams (communication, perception, structure) • Effort Estimation (at the personal level) • Risk (cross reference IAS/Secure Software Engineering) ○ The role of risk in the lifecycle ○ Risk categories including security, safety, market, financial, technology, people, quality, structure and process • Team management ○ Team organization and decision-making ○ Role identification and assignment ○ Individual and team performance assessment • Project management ○ Scheduling and tracking ○ Project management tools ○ Cost/benefit analysis • Software measurement and estimation techniques • Software quality assurance and the role of measurements • Risk ○ Risk identification and management ○ Risk analysis and evaluation ○ Risk tolerance (e.g., risk-adverse, risk-neutral, risk-seeking) ○ Risk planning • System-wide approach to risk including hazards associated with tools Outcome: Core-Tier2 #1, #2, #3, #4, #5, #6, #7, #8, #9 Outcome: Elective #10, #11, #12, #13, #14, #15, #16, #17, #18, #19, #20, #21, #22,#23, #24, #25	21
SE	Tools and Environments	• Software configuration management and version control • Release management	6

[1] See https://www.acm.org/binaries/content/assets/education/cs2013_web_final.pdf for a description of Computer Science Knowledge units

		• Programming environments that automate parts of program construction processes (e.g., automated builds) o Continuous integration • Tool integration concepts and mechanisms Outcome: Core-Tier2 #1, #2, #3, #6	
SE	Software Design	• Software architecture concepts and standard architectures (e.g. client-server, n-layer, transform centered, pipes-and-filters) • The use of components in design: component selection, design, adaptation and assembly of components, components and patterns, components and objects (for example, building a GUI using a standard widget set) Outcome: Core-Tier2 #8, #9, #10, #13, #14	8
SE	Software Construction	• Integration strategies: top-down, bottom-up, sandwich Outcome: Core-Tier2 #5	2