CPIS-250 Syllabus

Catalog Description

CPIS-250 Software Engineering
Credit: 3 (Theory: 3, Lab: 1, Practical: 1)
Prerequisite: CPCS-204
Classification: Department Required

The objective of this course is to introduce the basic concepts and required skills for software engineering. It covers the basic concepts and skills required for developing large scale applications that take long periods of time. The course presents the latest methods and techniques used in software engineering obtained from the actual practice in the field as well as latest advances accomplished by specialist research centers. Also, it particularly emphasizes on the role of team work on developing software and the skills required to work as part of a team. Topics include foundation for systems development, methodologies of IS development, software process models, process iteration, process activities, project management, project scope, project management life cycle, managing IS project, planning IS project, feasibility study of IS project, models of software development, determining system requirements (functional and non-functional requirements), data modeling and E-R-D model, object-oriented analysis and modeling, systems structure modeling, object, class, attributes, methods, classes relationships, generalization, specialization, association, class diagrams, object-oriented analysis and modeling.

Class Schedule

Lab/Tutorial 90 minutes 1 times/week

Meet 50 minutes 3 times/week or 80 minutes 2 times/week

Textbook

Ian Sommerville, , "Software Engineering", Addison-Wesley; 10 edition (2015-03-24)

ISBN-13 9780133943030 **ISBN-10** 0133943038

Grade Distribution

Week	Assessment	Grade %
4	Homework Assignments 1	5
5	Graded Lab Work 1	2.5
6	Graded Lab Work 2	2.5
7	Exam 1	15
8	Graded Lab Work 3	2.5
12	Graded Lab Work 4	2.5
12	Exam 2	15
13	Homework Assignments 2	5
14	Group Project	20
16	Comprehensive Final Exam	30

Last Articulated

April 4, 2018

Relationship to Student Outcomes

a	b	c	d	e	f	g	h	i	j
	x	x			x			x	

Course Learning Outcomes (CLO)

By completion of the course the students should be able to

1. Identify the phases of the System Development Life Cycle (SDLC) and their core activities correctly. (i)
2. Explain the different software development models (traditional, iterative, incremental and agile) and their processes correctly. (i)
3. **Use the appropriate software development model for a given software project and enumerating their arguments. (i)**
4. Develop a clear problem definition and complete domain analysis for a small to medium sized software system (b)
5. **Develop both functional and nonfunctional requirements for a given future software system by utilizing the knowledge of the requirements gathering methods/techniques, while respecting the completeness and the consistency goals. (i)**
6. Differentiate between analysis and design models in the context of software engineering and the uses of these models. (i)
7. **Design class diagrams that model the domain and the design for small to medium sized software systems, by articulating the knowledge of UML (Unified Modeling Language). (c)**
8. **Construct UML diagrams that illustrates the dynamic behavior of a system (sequence diagrams, activity diagrams and state diagrams). (i)**
9. **Identify some popular design patterns and explain their uses to enhance class diagram in the design model. (i)**
10. Use appropriate design patterns to enhance class diagram modeling for a given small to medium sized system. (i)
11. Identify popular software architectures and the influence of an architecture on the design of a software system. (i)
12. Explain the phases of maintenance and software evolution and factors affecting their processes (i)
13. Communicate orally and in writing the results of applying the acquired knowledge of software engineering phases of analysis and design and their relevant techniques to a small sized group project. (f)

Coordinator(s)

Dr. Alaa Khadidos, Associate Professor

CPIS-250 Syllabus

Topics Coverage Durations

Topics	Weeks
Software Development Models	3
Software Requirements Engineering	2
Interaction, Structural, Behavioral models	6
Design patterns and Architectural Styles	2
Software Evolution and maintenance	2