CPIT-251 Syllabus

Catalog Description

CPIT-251 Software Engineering (I)
Credit: 3 (Theory: 3, Lab: 0, Practical: 1)
Prerequisite: CPIT-250
Classification: Department Required

The objective of this course is to study software engineering principles and techniques used in the specification, design, and testing of software systems. Major software development methodologies are reviewed including requirements, analysis and specification, design, testing, and documentation.

Class Schedule

Lab/Tutorial 90 minutes 1 times/week

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

Textbook

Roger Pressman, Bruce Maxim, , "Software Engineering: A Practitioner's Approach", McGraw-Hill Education; 8 edition (2014-01-23)

ISBN-13 9780078022128 **ISBN-10** 0078022126

Grade Distribution

Week	Assessment	Grade %
6	Exam 1	15
12	Exam 2	15
13	Group Project	20
13	Graded Lab Work	10
14	Homework Assignments	10
16	Exam	30

Topics Coverage Durations

Topics	Weeks
Introduction to software engineering	2
Requirement Engineering	2
Software modelling and Architectural design	2
Distributed Software Engineering	3
Software Reuse	1
Interface Design	1
Software Testing	1
Quality Management	1
Configuration Management	1

Last Articulated

December 18, 2017

Relationship to Student Outcomes

a	b	c	d	e	f	g	h	i	j	k	l	m	n
	x		x				x	x	x	x	x		

Course Learning Outcomes (CLO)

By completion of the course the students should be able to

1. Describe the need for and importance of software engineering and its objectives. (j)
2. Differentiate between the various activities involved in software engineering process (j)
3. Discover how to develop software as part of a software team (d)
4. Differentiate between different stages of requirements engineering process (k)
5. Analyze the requirements in the form of functional and non-functional requirements to be placed in the requirements specification document (b)
6. Discuss the role played by documentation and formal specifications in developing a system, and the properties they must have. Associate with IEEE standard (b)
7. Design software applications and formulate model representation utilizing the Unified Modeling Language (UML), based on given requirements (h)
8. **Create use case diagram to model the functional requirement (k)**
9. **Design class diagram from a given scenario (k)**
10. **Formulate a scenario for a use case and then create the interaction diagrams for this use case (k)**
11. **Decide which architecture is suitable for a given non-functional requirements (b)**
12. Distinguish between different distributed application architecture models (i)
13. Contrast between two tier and three tier client server architecture (i)
14. List the different characteristics of software as a service (i)
15. Illustrate the reuse-driven software engineering (k)
16. Discover the important rules to create user interface forms then evaluate some cases (i)
17. **List the basic concepts involved in conducting software tests (l)**

Coordinator(s)

Mr. Majed Alshishtawi, Lecturer