APT 2080: INTRODUCTION TO SOFTWARE ENGINEERING

Pre-requisites – APT 2010: System Analysis and Design

3 Credit Units

Rationale

The course introduces the fundamentals of analyzing a problem and then implementing a solution as a computer software system. Students are introduced to the software life cycle and an overview is given of the basic hardware and software components of a computer system. Students learn about problem solving strategies, top-down program development and programming style. The course provides a basic introduction to data abstraction and object-oriented analysis and design. Emphasis is placed on programming and testing. This course aims to give students both a theoretical and a practical foundation in software engineering. In the theoretical part, students will learn about the principles and methods of current software engineering practices focusing on use of the UML in developing software systems. In the practical part, students will become familiar with the development of software products from an industry perspective, including generation of appropriate documents under tight schedules and limited resources.

Course Description

Software Engineering covers technical and non-technical (management) methods, techniques, and practices used to develop software-dominated systems. The course will cover the software development process; a survey of techniques and practices used throughout the software development process to improve quality, increase productivity, and reduce risk; and quality assurance related to dependable systems.

Learning Outcomes

At the end of this course students should be able to:

1. Define software development projects, roles, and practices
2. Compare and contrast classical and agile development processes
3. Identify and describe role and purpose of software engineering practices within and across the software development lifecycle;
4. Describe quality assurance which is important for all software projects
5. Use knowledge about the software process and software engineering practices to select and justify approaches to use given a project, its teams, and its constraints.
6. Write and test small programs.
7. Identify and describe the fundamental data structures and techniques used in computer science.

8. Use good programming principles and show why these principles are important in constructing quality software.
9. Demonstrate competence in using arrays and structures.
10. Identify and explain the concepts as well as practices of software engineering;
11. Apply knowledge and skills of both a commercial software engineering modeling tool and a commercial software development environment; and
12. Demonstrate skills of analyzing, designing and implementing an application as part of a systems development team.

Course Content

This unit introduces students to the foundations of programming and testing. Software quality and how it can be achieved is an underpinning theme. Frameworks are presented for problem solving in terms of fundamental data structures and algorithms. Data structures include arrays and structures. Techniques include functions, recursion and use of libraries. Analysis of algorithms, including measures of complexity, will be introduced. Implementations of the basic algorithms in a programming language will be explored. Those parts of the software engineering process which are applicable to an introductory unit are covered.

Teaching Methodologies

Lectures, Presentations by members of the class, Case discussions, Tutorials, Assignments, Continuous assessment tests, Lab Practical, Library, appropriate software, manual/notes, simple projects.

Instructional Materials/Equipment

Course text, Handouts, White board, Presentation slides, Journals

Methods of Evaluation

Class assignments, tests, projects to demonstrate use of software tools.

Laboratory Work	20%
Project	20%
Assignments	10%
Mid-semester	20%
Final semester exams	30%
Total	**100%**

Course Text

Software Engineering: (8th Edition), Ian Sommerville, Addison Wesley; 8 edition, June 4, 2006

Recommended reading

Supplementary Uml-Based Software Engineering Text Books

Using UML: Software Engineering with Objects and Components, Second Edition, P. Stevens, Addison-Wesley, 2006.

The Unified Software Development Process, I. Jacobson, G. Booch, and J. Rumbaugh, Addison-Wesley, 1999.

Unified Modeling Language Reference Books

The Object Constraint Language: Getting Your Models Ready for MDA, 2/E, J. Warmer and A. Kleppe, Addison-Wesley, 2003.

UML Distilled: A Brief Guide to the Standard Object Modeling Language, 3/E, M. Fowler with K. Scott, Addison-Wesley, 2004.

The Unified Modeling Language Reference Manual, 2/E, J. Rumbaugh, I. Jacobson, and G. Booch, Addison-Wesley, 2005.

Unified Development Process Reference Books

Use Cases: Requirements in Context, 2/E, D. Kulak and E. Guiney, Addison-Wesley, 2004.

The Unified Process Inception Phase: Best Practices in Implementing the UP, S.W. Ambler and L.L. Constantine, Elsevier Ltd , 2000.

The Unified Process Elaboration Phase: Best Practices in Implementing the UP, S.W. Ambler and L.L. Constantine, Elsevier Ltd, 2000.

The Unified Process Construction Phase: Best Practices in Implementing the UP, S.W. Ambler and L.L. Constantine, Elsevier Ltd, 2000.

The Unified Process Transition and Production Phases: Best Practices in Implementing the UP, S.W. Ambler and L.L. Constantine, Elsevier Ltd, 2001.

Visual Modeling with Rational Rose 2002 and UML, 3/e, T. Quatrani, Addison-Wesley, 2003.

User Interface Development Reference Books

Designing Object-Oriented User Interfaces, D. Collins, The Benjamin/Cummings Publishing Company, Inc., 1995.

Principles and Guidelines in Software User Interface Design, D.J. Mayhew, Prentice-Hall, Inc., 1992.

Project Management Reference Books

Object-Oriented Project Management with UML, M.R. Cantor, John Wiley & Sons, Inc., 1998.

Managing a Programming Project: Processes and People, 3/E, P. Metzger and J. Boddie, Prentice-Hall, Inc., 1996.

Object-Oriented Software Engineering: A Use Case Driven Approach, I. Jacobson, M. Christerson, P. Jonsson, and G. Overgaard, Addison-Wesley, 1992.

Object-Oriented Software Engineering: Using UML, Patterns, and Java (3rd Edition) Bernd Bruegge , Allen H. Dutoit 2009

Object-Oriented Modeling and Design with UML, 2/E, J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, and W. Lorensen, Prentice-Hall, Inc., 2005.

Software Engineering: A Practitioners Approach, 7/E, R.S. Pressman, McGraw Hill, Inc., 2010.

Fundamentals of Software Engineering, 2/E, C. Ghezzi, M. Jazayeri, and D. Mandrioli, Prentice-Hall, Inc., 2003.

Software Engineering: Theory and Practice, 3/E, S.L. Pfleeger, Prentice-Hall, Inc., 2006.

Software Engineering, 8/E, I. Sommerville. Addison-Wesley, 2006.