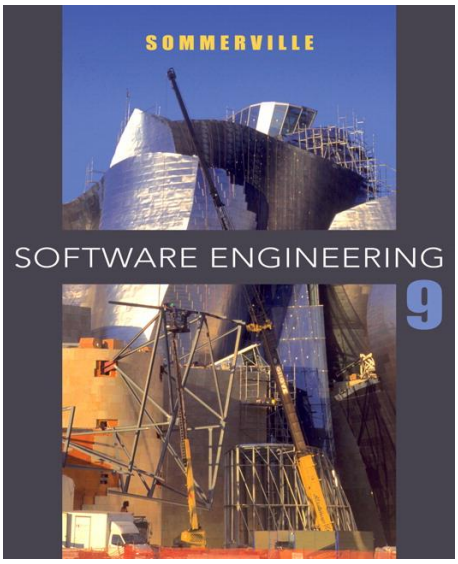
Sudan University of Science & Technology
College of Business Studies
Management Information System (MIS) Department

Introductory Course in Software Engineering



Course Syllabus

13 LECTURES @3 hours

Course Details

Course Description:	Software engineering is an engineered discipline in which the aim is the production of software products, delivered on time and within a set budget, that satisfies the client's needs. It covers all aspects of software production ranging from the early stage of product concept to design and implementation to post-delivery maintenance. This course introduces the major concepts and techniques of software engineering so that students can prepare for their future careers as software engineers. Moreover, through group projects, students can obtain hands-on experiences on entire phases and workflow of the software process.
Course Objectives:	▪ understand what software engineering is and why it is important; ▪ understand that the development of different types of software systems may require different software engineering techniques; ▪ understand some ethical and professional issues that are important for software engineers; ▪ understand the concepts of software processes and software process models; ▪ understand the rationale for agile software development methods, the agile manifesto, and the differences between agile and plandriven development; ▪ know the key practices in extreme programming and how these relate to the general principles of agile methods; ▪ understand the concepts of user and system requirements and why these requirements should be written in different ways;

	▪ understand the differences between functional and nonfunctional software requirements; ▪ understand how graphical models can be used to represent software systems; ▪ understand why different types of model are required and the fundamental system modeling perspectives of context, interaction, structure, and behavior; ▪ understand why the architectural design of software is important; ▪ understand the decisions that have to be made about the system architecture during the architectural design process; ▪ understand the most important activities in a general, object-oriented design process; ▪ understand some of the different models that may be used to document an object-oriented design; ▪ understand the stages of testing from testing, during development to acceptance testing by system customers; ▪ understand that change is inevitable if software systems are to remain useful and that software development and evolution may be integrated in a spiral model; ▪ understand software evolution processes and influences on these processes;
Course Grade Components:	• Final Exam 100 • Midterm Exam 20 • Test 10 • Seminar or Project 10 • Attendance 10 • Lab Work 50
Course Material:	Ian Sommerville, *Software Engineering, 9th Edition*, Addison Wesley, ISBN: 978-0-13-703515-1

Contact Information:	Students may contact me at any time, during office hours. You can also meet with me when I am at the University. If I am free, I will meet with you without an appointment. Otherwise, you should request an appointment. Office: Laser Building 2nd floor – Computer Center. Phone: +249 15566 1670 Email: abdelmonim@sustech.edu
Teaching Languages	Arabic – English.

Course Syllabus and Schedule:

Chapter 1 Introduction
1.1 Professional software development
1.2 Software engineering ethics
1.3 Case studies

Chapter 2 Software processes
2.1 Software process models
2.2 Process activities
2.3 Coping with change
2.4 The rational unified process

Chapter 3 Agile software development
3.1 Agile methods
3.2 Plan-driven and agile development
3.3 Extreme programming
3.4 Agile project management
3.5 Scaling agile methods

Chapter 4 Requirements engineering
4.1 Functional and non-functional requirements
4.2 The software requirements document
4.3 Requirements specification
4.4 Requirements engineering processes
4.5 Requirements elicitation and analysis
4.6 Requirements validation
4.7 Requirements management

Chapter 5 System modeling
5.1 Context models
5.2 Interaction models
5.3 Structural models
5.4 Behavioral models
5.5 Model-driven engineering

Chapter 6 Architectural design
6.1 Architectural design decisions
6.2 Architectural views
6.3 Architectural patterns
6.4 Application architectures

Chapter 7 Design and implementation

7.1 Object-oriented design using the UML
7.2 Design patterns
7.3 Implementation issues
7.4 Open source development

Chapter 8 Software testing

8.1 Development testing
8.2 Test-driven development
8.3 Release testing
8.4 User testing

Chapter 9 Software evolution

9.1 Evolution processes
9.2 Program evolution dynamics
9.3 Software maintenance
9.4 Legacy system management