UNIT I

Chapter 1 : Software Engineering Fundamentals **1-1 to 1-8**

Software Engineering Fundamentals : Introduction to software engineering, The Nature of Software, Defining Software, Software Engineering Practice.

1.1 Introduction **1-1**

1.2 Nature of Software **1-1**

1.2.1 Absence of Fundamental Theory 1-2

1.2.2 Ease of Change 1-2

1.2.3 Rapid Evolution of Technologies 1-2

1.2.4 Low Manufacturing Cost 1-2

1.3 Software Engineering : A Layered Technology **1-2**

1.3.1 Quality Focus 1-3

1.3.2 Process 1-3

1.3.3 Methods 1-3

1.3.4 Tools 1-4

1.3.5 The Characteristics of Software 1-4

1.3.6 Software Crisis 1-4

1.3.7 Legacy Software 1-5

1.4 Software Engineering Practice **1-5**

1.4.1 The Essence of Practice 1-5

1.4.2 Core Principles 1-6

Chapter 2 : Software Process **2-1 to 2-34**

Software Process : A Generic Process Model, defining a Framework Activity, Identifying a Task Set, Process Patterns, Process Assessment and Improvement, Prescriptive Process Models, The Waterfall Model, Incremental Process Models, Evolutionary Process Models, Concurrent Models, A Final Word on Evolutionary Processes. Unified Process, Agile software development : Agile methods, plan driven and agile development. Case studies - Agile Tools – JIRA.

2.1 A Generic Process Model **2-1**

2.1.1 Identifying a Task Set 2-2

2.2 Process Patterns **2-3**

2.3 Process Assessment and Improvement **2-4**

2.3.1 Approaches for Process Assessment 2-4

2.4 Prescriptive Process Models **2-5**

2.4.1 The Waterfall Model 2-5

2.4.2 Incremental Process Models 2-7

2.4.2(A) The Incremental Model 2-8

2.4.2(B) The RAD Model 2-9
2.4.3 Evolutionary Process Models 2-11
2.4.3(A) The Prototyping Paradigm 2-11
2.4.3(B) The Spiral Model 2-13
2.4.3(C) Concurrent Models 2-14
2.4.4 Differentiation between Prescriptive and Evolutionary Process Models 2-16
2.5 A Final Word on Evolutionary Processes 2-16
2.6 Unified Process 2-17
2.6.1 The Phases of Unified Process 2-17
2.6.2 The Iteration among Four Phases 2-19
2.7 Agile Software Development 2-19
2.7.1 Agile Process Model 2-20
2.7.1(A) Comparison between the Agile and Evolutionary Process Models 2-21
2.7.2 Agile Methods 2-21
2.7.3 Agile Manifesto 2-22
2.8 Plan-Driven and Agile Development 2-22
2.8.1 Comparison between Plan-driven and Agile Development 2-23
2.9 Agile Tools : JIRA 2-23
2.9.1 JIRA Platform 2-24
2.9.2 JIRA Architecture 2-24
2.9.3 JIRA Database Schema 2-25
2.9.4 JIRA Mobile Connect 2-25
2.9.5 JIRA Applications 2-25
2.9.6 JIRA APIs 2-25
2.10 Case Study : Agile Tools - JIRA 2-26
2.10.1 Introduction : Project Management Tools 2-26
2.10.2 Introduction : JIRA Software 2-26
2.10.3 JIRA Scheme 2-27

➢ **Model Question Paper (In Sem.) ..... M-1 to M-1**

UNIT II

Chapter 3 : Modeling 3-1 to 3-31

Modeling : Requirements Engineering, Establishing the Groundwork, Identifying Stakeholders, Recognizing Multiple Viewpoints, working toward Collaboration, Asking the First Questions, Eliciting Requirements, Collaborative Requirements Gathering, Usage Scenarios, Elicitation Work Products, Developing Use Cases, Building the Requirements Model, Elements of the Requirements Model, Negotiating Requirements, Validating Requirements. **Suggested Free Open Source Tools :** StarUML, Modelio, SmartDraw.

3.1 Introduction 3-1
3.2 Requirement Engineering 3-1
3.2.1 Inception 3-2
3.2.2 Elicitation 3-2
3.2.3 Elaboration 3-3
3.2.4 Negotiation 3-3
3.2.5 Specification 3-3
3.2.6 Validation 3-4
3.2.7 Requirement Management 3-4
3.2.8 Initiating the Requirement Engineering Process 3-4
3.3 Eliciting Requirements 3-6
3.3.1 Collaborative Requirements Gathering 3-6
3.3.2 Quality Function Deployment 3-7
3.3.3 Usage Scenarios 3-7
3.3.4 Elicitation Work Product 3-8
3.3.5 Elicitation Techniques 3-8
3.4 Developing Use Cases 3-8
3.4.1 Solved Examples on Use Case Diagram 3-9
3.5 Building the Requirements Model 3-16
3.5.1 Analysis Rules of Thumb 3-17
3.5.2 Domain Analysis 3-17
3.5.3 Requirements Modeling Approaches 3-18
3.6 Negotiating Requirements 3-18
3.7 Validating Requirements 3-18
3.8 Case Study - Open Source Tools : StarUML 3-19
3.8.1 Introduction : StarUML 3-19
3.8.2 Basic Concepts 3-20
3.8.3 Managing Project 3-21
3.8.4 Editing Elements 3-22
3.8.5 Extending Elements 3-23
3.8.6 Finding Model Elements 3-24
3.8.7 Formatting Diagram 3-24
3.8.8 Annotation Elements 3-29
3.8.9 Managing Extensions 3-30

UNIT III

Chapter 4 : Estimation For Software Projects **4-1 to 4-17**

Estimation for Software Projects : The Project Planning Process, Defining Software Scope and Checking Feasibility, Resources management, Reusable Software Resources, Environmental Resources, Software Project Estimation, Decomposition Techniques, Software Sizing, Problem-Based Estimation, LOC-Based Estimation, FP-Based Estimation, Object Point (OP) -based estimation, Process-Based Estimation, Process-Based Estimation, Estimation with Use Cases, Use-Case-Based Estimation, Reconciling Estimates, Empirical Estimation Models, The Structure of Estimation Models, The COCOMO II Mode, Preparing Requirement Traceability Matrix.

4.1 The Project Planning Process 4-1

4.2 Defining Software Scope and Checking Feasibility 4-1

4.2.1 Obtaining Information Necessary for Scope 4-2

4.2.2 Feasibility 4-2

4.3 Estimating Resources : Resource Management 4-3

4.3.1 Reusable Software Resources 4-3

4.3.2 Environmental Resources 4-4

4.4 Software Project Estimation 4-4

4.5 Decomposition Techniques 4-4

4.5.1 Problem Decomposition 4-5

4.5.2 Process Decomposition 4-5

4.6 Observations on Estimation 4-7

4.6.1 Software Sizing 4-7

4.6.2 Problem-based Estimation 4-8

4.6.3 LOC-based Estimation 4-8

4.6.4 FP-based Estimation 4-9

4.6.5 Process-based Estimation 4-10

4.6.6 An Example of Process-based Estimation 4-11

4.6.7 Estimation with Use-Cases 4-11

4.6.8 An Example of Use-Case based Estimation 4-12

4.6.9 Reconciling Estimates 4-13

4.7 Object Point (OP)-based Estimation (Estimation of Object-Oriented Projects) 4-13

4.8 Empirical Estimation Models 4-13

4.8.1 The Structure of Estimation Models 4-14

4.8.2 The COCOMO II Model 4-14

4.8.3 The Software Equation 4-16

4.9 Requirements Management : Preparing Requirement Traceability Matrix 4-16

Chapter 5 : Project Scheduling **5-1 to 5-10**

Project Scheduling : Project Scheduling, Defining a Task for the Software Project, Scheduling. **Suggested Free Open Source Tools :** Grantt Project, Agantty, Project Libre.

5.1 Project Scheduling 5-1
5.1.1 Defining a Task Set for the Software Project 5-2
5.1.2 Scheduling 5-3
5.1.3 Tracking the Schedule 5-3
5.2 Earned Value Analysis 5-4
5.3 Schedule Tracking Tools 5-5
5.3.1 Microsoft Project 5-5
5.3.2 Daily Activity Reporting and Tracking (DART) 5-6
5.4 Case Study - Open Source Tool : Gantt Project 5-7
5.4.1 Introduction : Gantt Project 5-7
5.4.2 Cost Management 5-8
5.4.3 Manage Calendar Details 5-9

UNIT IV

Chapter 6 : Design Engineering **6-1 to 6-16**

Design Concepts : Design within the Context of Software Engineering, The Design Process, Software Quality Guidelines and Attributes, Design Concepts - Abstraction, Architecture, design Patterns, Separation of Concerns, Modularity, Information Hiding, Functional Independence, Refinement, Aspects, Refactoring, Object-Oriented Design Concept, Design Classes, The Design Model , Data Design Elements, Architectural Design Elements, Interface Design Elements, Component-Level Design Elements, Component Level Design for Web Apps, Content Design at the Component Level, Functional Design at the Component Level, Deployment-Level Design Elements.

6.1 Introduction to Design Engineering 6-1
6.2 Design Process 6-1
6.3 Design Quality 6-2
6.3.1 Quality of Design Guidelines 6-2
6.3.2 The Quality Attributes 6-2
6.4 Design Concepts 6-3
6.4.1 Abstraction 6-3
6.4.2 Architecture 6-4
6.4.3 Patterns 6-4
6.4.4 Modularity 6-4
6.4.5 Information Hiding 6-6
6.4.6 Functional Independence 6-6

6.4.7 Refinement 6-8
6.4.8 Refactoring 6-9
6.4.8(A) Importance of Refactoring 6-9
6.4.9 Design Classes 6-9
6.4.10 Difference between Abstraction and Refinement 6-9
6.5 The Design Model 6-10
6.5.1 Data Design Elements 6-11
6.5.2 Architectural Design Elements 6-11
6.5.3 Interface Design Elements 6-11
6.5.4 Component-Level Design Elements 6-12
6.5.5 Deployment-Level Design Elements 6-12
6.5.6 Translating Requirements Model to Design Model 6-13
6.5.7 Guidelines for the Data Design 6-14
6.6 Component Level Design for Web Apps 6-14
6.6.1 Content Design at Component Level 6-14
6.6.2 Functional Design at Component Level 6-15
6.7 Pattern-based Software Design 6-15
6.7.1 Describing a Design Pattern 6-15
6.7.2 Using Patterns in Design 6-16
6.7.3 Frameworks 6-16

Chapter 7 : Architectural Design 7-1 to 7-14

Architectural Design : Software Architecture, What is Architecture, Why is Architecture Important, Architectural Styles, A brief Taxonomy of Architectural Styles. **Suggested Free Open Source Tool :** Smart Draw

7.1 Introduction to Architectural Design 7-1
7.2 Architectural Design Decisions 7-3
7.3 Architectural Views 7-4
7.4 Software Architecture 7-6
7.4.1 Why is Architecture Important ? 7-6
7.5 Software Styles 7-7
7.5.1 Classification of Styles 7-7
7.5.1(A) Data-centered Architectures 7-7
7.5.2(B) Data-flow Architectures 7-8
7.5.2(C) Call and Return Architectures 7-8
7.5.2(D) Object-oriented Architectures 7-9
7.5.2(E) Layered Architectures 7-9
7.6 Architectural Design 7-10

7.6.1 Representing the System in Context 7-10
7.6.2 Defining Archetypes 7-10
7.6.3 Refining the Architecture into Components 7-11
7.6.4 Describing Instantiations of the System 7-11
7.7 Case Study - Open Source Tool : SmartDraw 7-12
7.7.1 Introduction : SmartDraw 7-12
7.7.2 Easy and Powerful Diagramming 7-12

UNIT V

Chapter 8 : Project Risk Management 8-1 to 8-14

Risk Management : Software Risks, Risk Identification, Risk Projection, Risk Refinement, Risk Mitigation, Monitoring, and Management, The RMMM Plan.

8.1 Risk Analysis and Management 8-1
8.1.1 Software Risks 8-1
8.1.2 Reactive versus Proactive Risk Strategies 8-2
8.2 Risk Identification 8-3
8.2.1 Assessing Overall Project Risk 8-4
8.2.2 Risk Components and Drivers 8-4
8.3 Risk Projection 8-5
8.3.1 Developing a Risk Table 8-6
8.3.2 Assessing Risk 8-7
8.3.3 Project Plan 8-8
8.4 Risk Refinement 8-8
8.5 Risk Mitigation, Risk Monitoring and Risk Management (RMMM) 8-9
8.5.1 The RMMM Plan 8-9
8.6 The RMMM Plan for Case Study Project 8-12
8.6.1 The General Overview of RMMM Plan for WMITS 8-12
8.6.2 The Description of Risk for WMITS 8-13
8.6.3 Risk Mitigation, Monitoring and Management for WMITS 8-14

Chapter 9 : Software Configuration Management 9-1 to 9-20

Software configuration management : Software Configuration Management, The SCM Repository, The SCM Process, Configuration Management for any suitable software system. **Suggested Free Open Source Tools :** CF Engine configuration Tool, Puppet configuration Tool.

9.1 Software Configuration Management 9-1
9.1.1 SCM Basics 9-2

9.1.2 Baselines 9-2
9.1.3 Software Configuration Items (SCI) 9-2
9.2 SCM Repository 9-4
9.2.1 The Role of the Repository 9-4
9.2.2 General Features and Content 9-5
9.2.3 SCM Features 9-6
9.3 SCM Process 9-7
9.3.1 Identification of Objects in the Software Configuration 9-8
9.3.2 Version Control 9-8
9.3.3 Change Control 9-8
9.3.4 Configuration Audit 9-10
9.3.5 Status Reporting 9-10
9.4 SCM Tools such as GitHub 9-10
9.5 Computer-Aided Software Engineering (CASE) 9-11
9.6 Emerging Software Engineering Trends 9-15
9.6.1 Model-driven Development 9-15
9.6.2 Test-driven Development 9-16
9.7 Case Study - CFEngine Configuration Tool 9-18
9.7.1 Introduction : CFEngine 9-18
9.7.2 Policy Language and Compliance 9-18
9.7.3 CFEngine Policy Servers and Hosts 9-19
9.7.4 CFEngine Component Applications and Daemons 9-19
9.7.5 CFEngine Features 9-19
9.7.6 Choose a CFEngine Version 9-19
9.7.7 Installation 9-20

UNIT VI

Chapter 10 : Software Testing **6-1 to 6-18**

Strategic Approach to Software Testing, Verification and Validation, Organizing for Software Testing, Software Testing Strategy - The Big Picture, Criteria for Completion of Testing, Strategic Issues, Test Strategies for Conventional Software, Unit Testing, Integration Testing, Test Strategies for Object-Oriented Software, Unit Testing in the OO Context, Integration Testing in the OO Context, Test Strategies for WebApps, Validation Testing, Validation-Test Criteria, Configuration Review.
Suggested Free Open Source Tools : Selenium, JUnit.

10.1 A Strategic Approach to Software Testing 10-1
10.1.1 Verification and Validation 10-1
10.1.2 Organizing for Software Testing 10-2

10.1.3 Software Testing Strategy 10-2

10.2 The Big Picture 10-4

10.3 Criteria for Completion of Testing 10-4

10.4 Strategic Issues 10-4

10.5 Test Strategies for Conventional Software 10-5

10.5.1 Unit Testing 10-6

10.5.2 Integration Testing 10-7

10.6 Test Strategies for Object-Oriented Software 10-9

10.6.1 Unit Testing in OO Context 10-9

10.6.2 Integration Testing in OO Context 10-9

10.7 Test Strategies for WebApps 10-10

10.8 Validation Testing 10-10

10.8.1 Validation-Test Criteria 10-10

10.8.2 Configuration Review 10-11

10.8.3 Alpha and Beta Testing 10-11

10.9 Case Study - Open Source Tool : Selenium 10-11

10.9.1 Introduction : Selenium 10-11

10.9.2 Architecture of the Selenium 10-12

10.9.3 Selenium Integrated Development Environment (IDE) 10-12

10.9.4 Testing Using Selenium 10-14

10.9.5 Examination of Result 10-17

10.9.6 Exporting the Test Results 10-18

➢ **Model Question Paper (End Sem.) ..... M-1 to M-2**

❑❑❑