

Module - I

Syllabus : Nature of Software, Software Engineering, Software Process, Capability Maturity Model (CMM)s. Generic Process Model, Prescriptive Process Models, The Waterfall Model V-model, Incremental Process Models, Evolutionary Process Models, Concurrent Models, Agile process Agility Principles, Extreme Programming (XP), Scrum Kanban Model.

Chapter 1 : Introduction to Software Engineering **1-1 to 1-13**

1.1	Nature of Software.....	1-1
1.1.1	Absence of Fundamental Theory.....	1-1
1.1.2	Ease of Change.....	1-1
1.1.3	Rapid Evolution of Technologies.....	1-2
1.1.4	Low Manufacturing Cost.....	1-2
1.2	Software Definition.....	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
1.4	The Characteristics of Software.....	1-4
1.5	Software Crisis.....	1-4
1.6	Legacy Software.....	1-5
1.7	Software Application Domains.....	1-5
1.7.1	System Software.....	1-5
1.7.2	Application Software.....	1-5
1.7.3	Engineering/ Scientific Software.....	1-5
1.7.4	Embedded Software.....	1-6
1.7.5	Personal Computer Software.....	1-6
1.7.6	Business Software.....	1-6
1.7.7	Product-line Software.....	1-6
1.7.8	Real-time Software.....	1-7
1.7.9	Artificial Intelligence Software.....	1-7
1.7.10	Ubiquitous Computing.....	1-7
1.7.11	Net sourcing.....	1-7
1.7.12	Open Source Software.....	1-8
1.8	The Software Process.....	1-8
1.8.1	Umbrella Activities.....	1-9
1.9	CMM Models.....	1-10

Chapter 2 : Process Models 2-1 to 2-30

2.1 A Generic Process Model (or Generic Process Framework).....2-1

 2.1.1 Communication2-1

 2.1.2 Planning.....2-1

 2.1.3 Modelling.....2-1

 2.1.4 Construction2-2

 2.1.5 Deployment.....2-2

2.2 Prescriptive Process Models.....2-3

 2.2.1 The Waterfall Model.....2-3

 2.2.1(A) V-Model (Software Development)2-5

 2.2.2 Incremental Process Models.....2-5

 2.2.2(A) The Incremental Model.....2-6

 2.2.2(B) The RAD Model.....2-7

 2.2.3 Evolutionary Process Models.....2-9

 2.2.3(A) The Prototyping Paradigm2-9

 2.2.3(B) The Spiral Model2-10

 2.2.3(C) The Concurrent Development Model.....2-12

 2.2.3(D) Differentiation between Prescriptive and Evolutionary Process Models2-13

2.3 Agile Process Model2-14

 2.3.1 Comparison between the Agile and Evolutionary Process Models2-14

2.4 Agile Software Development2-15

 2.4.1 Agile methods.....2-15

 2.4.2 Agile Manifesto2-16

 2.4.3 Agility Principles.....2-16

2.5 Extreme Programming Practices.....2-17

 2.5.1 XP Values.....2-17

 2.5.2 The XP Process.....2-18

 2.5.3 Scrum.....2-18

 2.5.3(A) Process Flow.....2-19

 2.5.4 Scrum Roles2-20

 2.5.5 Scrum Cycle Description.....2-21

 2.5.6 Product Backlog2-22

 2.5.7 Sprint Planning Meeting2-22

 2.5.8 Sprint Backlog.....2-23

 2.5.9 Sprint Execution.....2-24

 2.5.10 Daily Scrum Meeting2-24

 2.5.11 Maintaining Sprint Backlog and Burn-Down Chart.....2-25

 2.5.12 Sprint Review and Retrospective2-25

2.6 Introduction to Agile Tools : Kanban.....2-26

 2.6.1 Kanban Boards.....2-26

 2.6.2 Kanban Cards2-27

 2.6.3 The Benefits of Kanban2-27

 2.6.4 Comparison between Kanban and Scrum.....2-29

Module - II

Syllabus : Requirement Elicitation , Software requirement specification (SRS), Developing Use Cases (UML), Scenario-based model, Class-based model, Behavioural model.

Chapter 3 : Requirement Analysis		3-1 to 3-28
3.1	Requirements Analysis	3-1
3.1.1	Analysis Rules of Thumb	3-2
3.1.2	Domain Analysis	3-2
3.1.3	Requirements Modelling Approaches	3-3
3.2	Requirements Elicitation : Process.....	3-3
3.2.1	Collaborative Requirements Gathering	3-3
3.2.2	Quality Function Deployment	3-3
3.2.3	Usage Scenarios.....	3-4
3.2.4	Elicitation Work Product.....	3-4
3.2.5	Elicitation Techniques	3-5
3.3	Software Requirements Specification (SRS).....	3-5
3.3.1	Writing Software Requirements Specifications	3-6
3.3.2	What is a Software Requirements Specification?	3-6
3.3.3	What Kind of Information Should an SRS Include?.....	3-7
3.3.4	SRS Template.....	3-7
3.3.5	Characteristics of an SRS.....	3-7
3.3.6	Structured Specifications for an Insulin Pump Case Study	3-8
3.3.7	Tabular Specifications for an Insulin Pump Case Study	3-9
3.4	Developing Use Cases	3-9
3.5	Requirement Model.....	3-10
3.6	Scenario Based Modelling : UML Models	3-11
3.6.1	Diagramming in UML.....	3-11
3.6.2	Developing Use Cases Diagram	3-13
3.6.3	Developing Activity Diagram.....	3-14
3.6.4	Swim Lane Diagram.....	3-15
3.6.5	Class Diagram.....	3-16
3.7	Class-Based Model	3-19
3.7.1	Developing the Class Diagram	3-20
3.7.2	Collaboration Diagrams	3-23
3.7.3	CRC Models.....	3-23
3.8	Behavioural Modelling using State Diagrams	3-24
3.8.1	Identifying the Events with Use-Cases	3-24
3.8.2	Create the Sequence for Use-Case	3-24
3.8.3	State Machine Diagram with Orthogonal States	3-25
3.8.3(A)	Orthogonal States	3-26
3.9	Case Study : Hospital Management System	3-27

Module - III

Syllabus : Management Spectrum, 3Ps (people, product and process), Process and Project metrics, Software Project Estimation: LOC, FP, Empirical Estimation Models - COCOMO II Model, Specialized Estimation Techniques, Project scheduling : Defining a Task Set for the Software Project, Timeline charts, Tracking the Schedule, Earned Value Analysis

Chapter 4 : Project Scheduling and Tracking	4-1 to 4-21
4.1 The Management Spectrum	4-1
4.1.1 The People	4-1
4.1.1(A) Stake Holders	4-2
4.1.1(B) Team Leaders	4-2
4.1.1(C) Software Team.....	4-3
4.1.1(D) Agile Teams.....	4-3
4.1.1(E) Co-ordination and Communication Issues	4-4
4.1.2 The Product.....	4-4
4.1.3 The Process	4-4
4.1.4 The Project.....	4-5
4.2 Metrics in the Process and Project Domains	4-5
4.2.1 Process Metrics.....	4-5
4.2.2 Project Metrics.....	4-6
4.3 Software Project Estimation.....	4-6
4.4 Observations on Estimation.....	4-7
4.4.1 Software Sizing	4-7
4.4.2 Problem-Based Estimation.....	4-8
4.4.3 An Example of LOC-Based Estimation.....	4-8
4.4.4 An Example of FP-Based Estimation.....	4-9
4.4.5 Process-Based Estimation	4-10
4.4.6 An Example of Process-Based Estimation	4-11
4.4.7 Estimation with Use-Cases	4-11
4.4.8 An Example of Use-Case Based Estimation.....	4-11
4.4.9 Reconciling Estimates.....	4-12
4.4.10 Software Scope and Feasibility.....	4-12
4.4.10(A) Obtaining Information Necessary for Scope.....	4-13
4.4.10(B) Feasibility.....	4-13
4.4.10(C) A Scoping Example.....	4-13
4.5 Empirical Estimation Models.....	4-14
4.5.1 The Structure of Estimation Models.....	4-14
4.5.2 The COCOMO II Model.....	4-14
4.5.3 The Software Equation.....	4-16
4.6 Specialized Estimation Techniques.....	4-16
4.6.1 Estimation for Agile Development.....	4-17
4.6.2 Estimation for Web Engineering Projects.....	4-17

4.7	Project Scheduling	4-17
4.7.1	Defining a Task Set for the Software Project	4-18
4.7.2	Scheduling	4-19
4.7.2(A)	Time-line Charts.....	4-19
4.7.3	Tracking the Schedule	4-20
4.8	Earned Value Analysis.....	4-20

Module - IV

Syllabus : Design Principles, Design Concepts, Effective Modular Design, Cohesion and Coupling. Architectural Design, Component-level design and User Interface Design.

Chapter 5 : Software Design **5-1 to 5-7**

5.1	Design Principles	5-1
5.2	Concept of Design.....	5-2
5.2.1	Abstraction	5-2
5.2.2	Architecture	5-2
5.2.3	Patterns.....	5-2
5.2.4	Modularity	5-2
5.2.5	Information Hiding	5-3
5.3	Effective Modular Design	5-3
5.4	Cohesion and Coupling.....	5-4

Chapter 6 : Architectural, Component-level and User Interface Design **6-1 to 6-22**

6.1	Introduction to Architectural Design.....	6-1
6.2	Architectural Design Decisions	6-3
6.3	Architectural Views	6-4
6.4	Architectural Patterns.....	6-5
6.4.1	Software Architecture	6-6
6.5	Application Architectures.....	6-6
6.5.1	Transaction Processing Systems	6-7
6.5.2	Language Processing Systems	6-8
6.6	Modelling Component Level Design.....	6-10
6.7	User Interface Design.....	6-11
6.7.1	Type of User Interface	6-12
6.7.2	Characteristics of Good User Interface.....	6-14
6.7.3	Benefits of Good Interface Design	6-14
6.8	The Golden Rules.....	6-14
6.8.1	Place the user in Control	6-14
6.8.2	Reduce the User’s Memory Load	6-15
6.8.3	Make the Interface Consistent	6-16
6.8.4	Necessity of a Good User Interface	6-16

6.9	Shneiderman’s 8 Golden Rules for UI Analysis.....	6-17
6.10	Interface Analysis and Design Models.....	6-18
6.10.1	Introduction to Interface Analysis and Design Models.....	6-18
6.10.2	User Interface Design Process.....	6-18
6.11	Interface Design Steps and Analysis	6-19
6.11.1	Applying Interface Design Steps	6-20
6.11.2	User Interface Design Patterns.....	6-20
6.11.3	Interface Design Issues	6-20
6.11.4	Interface Design Evaluation.....	6-21

Module - V

Syllabus : Risk Identification, Risk Assessment, Risk Projection, RMMM. Software Configuration management, SCM repositories, SCM process, Software Quality Assurance :Task and Plan, Metrics, Software Reliability Formal Technical Review (FTR), Walkthrough

Chapter 7 : Risk Management **7-1 to 7-11**

7.1	Introduction to Risk Management.....	7-1
7.1.1	Reactive Versus Proactive Risk Strategies	7-2
7.1.2	Various steps in Risk Management.....	7-3
7.2	Risk Identification.....	7-3
7.2.1	Risk Components and Drivers	7-4
7.3	Risk Assessment	7-5
7.4	Risk Projection.....	7-6
7.4.1	Developing a Risk Table	7-6
7.4.2	Assessing Risk.....	7-7
7.5	RMMM (Risk Mitigation, Monitoring and Planning).....	7-8
7.5.1	The RMMM Plan.....	7-8

Chapter 8 : Software Configuration Management and Quality Assurance **8-1 to 8-19**

8.1	Software Configuration Management (SCM).....	8-1
8.1.1	SCM Basics (Configuration Management System Elements).....	8-2
8.1.2	Baselines.....	8-2
8.1.3	Software Configuration Items	8-2
8.2	The SCM Repository	8-4
8.2.1	The Role of the Repository	8-4
8.2.2	General Features and Content.....	8-4
8.2.3	SCM Features.....	8-5
8.3	The SCM Process.....	8-6
8.3.1	Identification of Objects in the Software Configuration	8-7
8.3.2	Version Control.....	8-7
8.3.3	Change Control	8-8

8.3.4	Configuration Audit	8-9
8.3.5	Status Reporting.....	8-9
8.4	Software Quality Assurance (SQA)	8-10
8.4.1	Software Quality Assurance Activities	8-11
8.4.2	SQA Relationships to Other Assurance Activities.....	8-11
8.5	Quality Metrics	8-14
8.5.1	Measuring Quality	8-15
8.5.2	Defect Removal Efficiency	8-15
8.6	Software Reliability	8-15
8.6.1	Measures of Reliability and Availability.....	8-15
8.6.2	Software Safety.....	8-16
8.7	Formal Technical Reviews (FTR)	8-16
8.7.1	Review Meetings	8-17
8.7.2	Review Guidelines.....	8-17
8.8	Walkthrough.....	8-17
8.8.1	Differentiation Between FTR and Walkthrough	8-18

Module - VI

Syllabus : Strategic Approach to Software Testing, Unit testing, Integration testing, Verification and Validation Testing, System Testing, Software Testing Fundamentals, White-Box Testing, Basis Path Testing, Control Structure Testing, Black-Box Testing, Software, maintenance and its types, Software Re-engineering, Reverse Engineering.

Chapter 9 : Software Testing and Maintenance **9-1 to 9-26**

9.1	Strategic Approach to Software Testing.....	9-1
9.1.1	Verification and Validation.....	9-2
9.1.1(A)	Difference between Verification and Validation	9-3
9.1.2	Organizing for Software Testing	9-3
9.2	Unit testing.....	9-4
9.3	Integration Testing.....	9-5
9.4	Verification and Validation Testing.....	9-7
9.4.1	Validation Test Criteria.....	9-7
9.4.2	Configuration Review	9-7
9.4.3	Acceptance Testing.....	9-7
9.4.4	Alpha and Beta Testing	9-8
9.4.4(A)	Difference between Alpha Testing and Beta Testing	9-8
9.5	System Testing.....	9-9
9.5.1	Recovery Testing	9-9
9.5.2	Security Testing.....	9-9
9.5.3	Stress Testing	9-9
9.5.4	Performance Testing.....	9-10
9.5.5	Verification and Validation.....	9-10

9.6	Software Testing Fundamentals	9-10
9.6.1	Test Characteristics (Attributes of good test).....	9-12
9.7	Principles of Testing.....	9-12
9.8	White-Box Testing.....	9-12
9.8.1	Basis Path Testing	9-13
9.8.1(A)	Flow Graph Notation.....	9-13
9.8.1(B)	Independent Program Paths.....	9-14
9.8.1(C)	Deriving Test Cases.....	9-15
9.8.1(D)	Graph Matrices	9-15
9.8.2	Control Structure Testing	9-16
9.8.2(A)	Condition Testing	9-16
9.8.2(B)	Data Flow Testing.....	9-17
9.8.2(C)	Loop Testing	9-17
9.9	Black-Box Testing.....	9-18
9.9.1	Graph-Based Testing Method.....	9-19
9.9.2	Equivalence Partitioning.....	9-19
9.9.3	Boundary Value Analysis	9-20
9.9.4	Orthogonal Array Testing	9-20
9.9.5	Differentiation between White-box and Black-box Testing.....	9-21
9.10	Software Maintenance and its Types.....	9-21
9.10.1	Modifiability.....	9-22
9.10.2	Types of Maintenance.....	9-22
9.10.2(A)	Corrective maintenance.....	9-23
9.10.2(B)	Adaptive maintenance.....	9-23
9.10.2(C)	Perfective maintenance	9-23
9.10.2(D)	Preventive maintenance.....	9-23
9.10.3	Need of Maintenance.....	9-23
9.10.4	Steps for Creating a Maintenance Log	9-24
9.11	Software Re-engineering.....	9-24
9.12	Reverse Engineering	9-24
9.12.1	Abstraction Level.....	9-25
9.12.2	Completeness	9-25
9.12.3	Directionality.....	9-25