



Software Engineering

BCA/CSIT



Ramesh Singh Saud

Manoj Giri

BCA Syllabus

Course Title: Software Engineering (3 Cr.)

Course No: CACS253

Year/Semester: II/IV

Class Load: 4 Hrs. Week (Theory: 3 Hrs, Tutorial : 1)

Full Marks: 60 + 20 + 20

Pass Marks: 24 + 8 + 8

Course Description:

This course includes the topics that provide fundamental concept and standard of software engineering so that students will be able to develop software and/or handle software project using the global standard of software.

Course Objectives:

This course is designed to provide the students with the basic competencies required to identify requirements, documents the system design and maintain a developed system. It presumes a general understanding of computer and programming which are covered in the first and second semester of the degree.

Course Content

Unit 1 Introduction

4 Hrs

Definition of Software, Type of Software, Characteristics of Software, Attributes of Goods Software, Definition of Software Engineering, Software Engineering Costs, Key Challenges that Software Engineering Facing, System Engineering and Software Engineering, Professional Practices.

Unit 2 Software Development Process Model

8 Hrs

Software Development Process Model: The Waterfall Model, Evolutionary Development, Component-Based Software Engineering (CBSE); Process Iteration: Incremental Delivery, Spiral Development; Rapid Software Development: Agile Methods, Extreme Programming, Rapid Application Development, Software Prototyping; Rational Unified Rapid (RUP), Computer Aided Software Engineering (CASE); Overview of CASE Approach Classification of CASE tools.

Unit 3 Software Development Process Model

10 Hrs

(System and Software Requirements) Type of Software Requirements: Functional and Non-Functional Requirements, Domain Requirements, User Requirements; Elicitation and Analysis of Requirements: Overview of Techniques, View Points, Interviewing, Scenarios, Use-Case, Ethnography, Requirement Validation, Requirement Specification, Feasibility.

Unit 4 Software Design

10 Hrs

Design Concept: Abstraction, Architecture Patterns, Modularity: Cohesion, Coupling; Information Hiding, Functional Independence, Refinement; Architecture Design: Repository Model, Client Server Model, Layered Model, Modular Decomposition; Procedural Design Using Structure Methods, User Interface Design: Human Computer, Information Presentation, Interface Evaluation; Design Notation.

Unit 5 Coding

2 Hrs

Programming Language and Development Tools, Selecting Language and Tools Good Programming Practices

Unit 6 Software Maintenance

6 Hrs

Verification and Validation, Techniques of Testing: Black-box and White-box Testing, inspections: level of testing: Unit Testing, Integration Testing, Interface Testing, System Testing, Alpha and Beta Testing, Regression Testing: Design of Test Cases, Quality Management Activities, Product and Process Quality, Standard: ISO 9000, capability Maturity Model (CMM)

Unit 7 Software Maintenance

Evolving Nature of Software, Different Types of Maintenance: Fault Repair, Software Adaptation, Functionality Addition or Modification; maintenance prediction, Re-Engineering, Configuration Management (CM): Importance of CM, Configuration items, Versioning;

3 Hrs

Unit 8 Managing Software Projects

Needs for the Proper management of Software Projects, Management Activities: Project Planning, Estimating Costs, Project scheduling, Risk Management, managing People.

2 Hrs

Teaching Methods

The general teaching pedagogy includes class lectures, group discussion, case studies, guest lectures, research work, project work, assignments (theoretical and practical), and examination (write and verbal), depending upon the nature of the topics. The teaching faculty will determine the choice of teaching pedagogy as per the need of the topics.

Evaluation

| Examination Scheme | | | | |
|---------------------|-----------|---------------------|-----------|-------|
| Internal Assessment | | External Assessment | | Total |
| Theory | Practical | Theory | Practical | |
| 40 | - | 60 (3 Hrs.) | - | 100 |

Text Book

1. Roger S. Pressman, "Software Engineering: A Practitioner's Approach, 6th Edition, MCgraw Hill International edition, 2005

Reference Books

1. Ali Behforooz and Frederick J. Hudson, "Fundamentals of Software Engineering", OUP, 1996
2. Ian Sommerville, "Software Engineering", 9th Edition, Addison-Wesley, 2010, ISBN: 978-0137035151
3. Pankaj Jalote, "An Integrated Approach to Software Engineering", 2nd Edition, Springer, 1997

CSIT Syllabus

Course Title: Software Engineering

Full Marks: 60 + 20 + 20

Course No: CSC364

Pass Marks: 24 + 8 + 8

Nature of the Course: Theory + Lab Credit Hrs: 3

Semester: VI

Course Description:

This course familiarizes students with different concepts of software engineering mainly focusing on software process models, agile development, requirements engineering, models, design, implementation, testing, evolution, and software project management.

Course Objectives:

The main objective of this course is to provide knowledge of different concepts of software engineering so that students will be able to develop high quality software using different software management skills.

Course Contents:

Unit 1: Introduction

(2 Hrs.)

Software and its Types; Attributes of Good Software; Software Engineering and its Importance; Fundamental Software Engineering Activities; Difference between Software Engineering and Computer Science; Difference between Software Engineering and System Engineering; Challenges and Cost of Software Engineering; Professional Software Development; Software Engineering Diversity; Internet Software Engineering; Software Engineering Ethics

Unit 2: Software Processes (5 Hrs.)

Software Process; Software Process Models (Waterfall Model; Incremental Development; Software Process; Software Process Models (Waterfall Model; Incremental Development; Integration and Configuration); Software Process Activities (Software Specification, Software Design and Implementation; Software Validation; Software Evolution); Coping with Change (Prototyping, Incremental Delivery); Process Improvement

Unit 3: Agile Software Development (3 Hrs.)

Agile Development; Plan-Driven vs. Agile Development; Agile Methods; Agile Development Techniques; Introduction to Agile Project Management

Unit 4: Requirements Engineering (3 Hrs.)

Concept of User and System Requirements; Functional and Non-Functional Requirements; Requirements Engineering Process; Requirements Elicitation; Requirements Specification; Requirements Validation; Requirements Change

(6 Hrs.)

Unit 5: System Modeling

Introduction to System Modeling; Context Models; Interaction Models; Structural Models; Behavioral Models; Model-Driven Architecture

Unit 6: Architectural Design

Introduction; Architectural Design Decisions; Architectural Views; Architectural Patterns; Application Architectures

(6 Hrs.)

Unit 7: Design and Implementation

Introduction; Object-Oriented Design using UML; Design Patterns; Implementation Issues; Open-Source Development

(5 Hrs.)

Unit 8: Software Testing

Introduction; Validation and Verification Testing; Software Inspection; Software Testing Process; Development Testing; Test-Driven Development; Release Testing; User Testing

(5 Hrs.)

Unit 9: Software Evolution

Evolution Process; Legacy Systems; Software Maintenance

(3 Hrs.)

Unit 10: Software Management (7 Hrs.)

Software Project Management; Project Management Activities (Project Planning, Risk Management, People Management, Reporting and Proposal Writing); Project Planning (Software Pricing, Plan-Driven Development, Project Scheduling, Estimation Techniques, COCOMO Cost Modeling); Introduction to Quality Management and Configuration Management

Laboratory / Project Work:

Students should prepare a project report along with software product using different concepts of software engineering. The project can be done in groups with at most four members in each group using any suitable database, programming, interfacing technologies, and project management concepts.

Text Book:

1. Software Engineering, 10th Edition, Ian Sommerville, Pearson Education 2016

References Books:

1. Software Engineering: A Practitioner's Approach, 8th Edition, Roger S. Pressman and Bruce R. Maxim, McGraw-Hill Education 2015
2. Beginning Software Engineering, Rod Stephens, John Wiley & Sons Inc 2015

Table of Contents

1

Chapter

INTRODUCTION

| | |
|---|----|
| Software..... | 2 |
| Types of Software | 2 |
| Application Software..... | 4 |
| Web based and Mobile Application Software | 5 |
| Mobile Application | 6 |
| Software Characteristics | 8 |
| Attributes of Good Software | 10 |
| Software Engineering..... | 11 |
| Need of Software Engineering..... | 11 |
| Key Challenges Facing Software Engineering..... | 12 |
| Professional and Ethical Responsibility | 13 |
| Software Engineering Diversity | 15 |
| Internet and Software Engineering | 15 |
| □ Exercise | 16 |

2

Chapter

SOFTWARE PROCESS AND PROCESS MODEL

| | |
|--|----|
| Software Process | 16 |
| Software Specification..... | 16 |
| Software Design and Implementation..... | 17 |
| Software Validation..... | 17 |
| Software Evolution..... | 19 |
| Software Development Process Models..... | 19 |
| ① Waterfall Model..... | 20 |
| When to use the waterfall Model | 22 |
| ② Evolutionary Development Model/Incremental Development Model | 22 |
| Basic Principles of the Component-based Software Engineering | 24 |
| ③ Prototyping Model..... | 26 |
| Process Iteration | 28 |
| Incremental Delivery | 28 |
| ④ Spiral Model | 30 |
| ⑤ Rapid Software Development | 32 |
| Rational Unified Process (RUP)..... | 34 |
| Inception Phase | 34 |
| Elaboration Phase | 35 |

| | |
|---|----|
| Construction Phase | 35 |
| Transition Phase | 35 |
| Advantages of Rational Unified Process | 36 |
| Disadvantages of Rational Unified Process | 36 |
| Computer Aided Software Engineering (CASE)..... | 38 |
| Case Tools Types | 40 |
| <input type="checkbox"/> Exercise | 40 |

3

Chapter

AGILE SOFTWARE DEVELOPMENT

| | |
|---|----|
| Agile Methods..... | 42 |
| Agile Principles | 42 |
| Plan-driven and Agile Development | 45 |
| Extreme Programming..... | 46 |
| Communication..... | 47 |
| Simplicity | 47 |
| Feedback | 47 |
| Courage..... | 48 |
| Respect..... | 48 |
| Testing in XP..... | 50 |
| Pair Programming..... | 50 |
| Agile Project Management | 51 |
| Scrum..... | 52 |
| Scrum Roles..... | 52 |
| Scaling Agile Methods | 56 |
| <input type="checkbox"/> Exercise | 56 |

4

Chapter

SOFTWARE REQUIREMENT & REQUIREMENT ENGINEERING

| | |
|--|----|
| (Software Requirement)..... | 58 |
| Types of Requirements..... | 58 |
| User Requirement..... | 58 |
| System Requirement..... | 59 |
| Functional Requirement..... | 59 |
| Non-Functional Requirements | 60 |
| Examples of Non-functional requirements..... | 60 |
| Advantages of Non-Functional Requirement | 61 |
| Types of Non-Functional Requirements | 61 |
| Domain Requirement | 62 |
| Software Requirement Specification..... | 62 |
| Natural Language Specification..... | 63 |

(5)

| | |
|---|----|
| Requirement Engineering Process..... | 64 |
| Feasibility Study..... | 65 |
| Requirement Elicitation and Analysis..... | 67 |
| Ethnography..... | 70 |
| Viewpoints..... | 70 |
| Requirement Specification..... | 71 |
| Requirement Validation..... | 71 |
| Requirement Validation Techniques..... | 72 |
| Requirement Change Management..... | 72 |
| □ Exercise..... | 74 |

5

Chapter

SYSTEM MODELLING

| | |
|---------------------------------------|----|
| Introduction to System Modelling..... | 76 |
| Context Models..... | 77 |
| Interaction Models..... | 78 |
| Structural Models..... | 80 |
| Behavioural Models..... | 82 |
| Data-driven Modeling..... | 82 |
| Event-driven Modeling..... | 82 |
| Model-Driven Architecture..... | 84 |
| □ Exercise..... | 84 |

6

Chapter

SOFTWARE ARCHITECTURAL DESIGN

| | |
|---|----|
| Introduction..... | 85 |
| Design Concept..... | 85 |
| Architectural Design..... | 85 |
| Architectural Views..... | 85 |
| System Organization/System structuring..... | 85 |
| Application Architectures..... | 85 |
| Information Presentation..... | 85 |
| Introduction..... | 86 |
| Design Concept..... | 86 |
| Abstraction..... | 86 |
| Architecture..... | 87 |
| Pattern..... | 87 |
| Modularity..... | 88 |
| Information Hiding..... | 88 |

| | |
|--|-----|
| Functional Independence..... | 88 |
| Refinement..... | 89 |
| Refactoring..... | 89 |
| Cohesion and Coupling..... | 90 |
| Architectural Design..... | 90 |
| Architectural Design Decisions..... | 91 |
| Architectural Views..... | 92 |
| System Organization/System structuring..... | 92 |
| Repository Model..... | 93 |
| Client Server Model..... | 94 |
| Layered Architectural Style..... | 95 |
| MVC Architecture..... | 97 |
| Pipe and Filter Model..... | 98 |
| Application Architectures..... | 98 |
| Procedural Design..... | 99 |
| Using Structured Methods..... | 99 |
| User Interface Design..... | 100 |
| Human Computer Interaction (User Interaction)..... | 101 |
| Information Presentation..... | 102 |
| Interface Evaluation..... | 103 |
| Design Notation..... | 104 |
| <input type="checkbox"/> Exercise..... | 104 |

7

Chapter

DESIGN AND IMPLEMENTATION

| | |
|---|-----|
| Introduction..... | 106 |
| Object Oriented Design using the UML..... | 106 |
| • System Context and Interactions..... | 107 |
| • Architectural Design..... | 108 |
| • Object Class Identification..... | 109 |
| Design Models..... | 110 |
| Design Patterns..... | 116 |
| Implementation Issues..... | 116 |
| Open Source Development..... | 117 |
| <input type="checkbox"/> Exercise..... | 118 |

8

Chapter

CODING

| | |
|--|-----|
| Programming Languages..... | 120 |
| Imperative Programming Languages..... | 120 |
| Functional Programming Languages..... | 120 |
| Object-Oriented Programming Languages..... | 121 |

| | |
|-------------------------------------|-----|
| Logical Programming Languages | 121 |
| Program Development Tools | 121 |
| Selecting Language and Tools | 130 |
| Good Program Practice | 130 |
| ❑ Exercise | 132 |

9

Chapter

SOFTWARE TESTING

| | |
|--|-----|
| Quality Management | 133 |
| Software Verification and Validation | 134 |
| Verification | 134 |
| Validation | 135 |
| Software Inspection | 136 |
| Inspection key Points | 136 |
| Inspection Roles | 136 |
| Inspection Process | 137 |
| Software Testing | 138 |
| Software Testing Necessicity | 138 |
| Software Testing Objectives | 138 |
| Software Testing Process | 139 |
| Test Case and Test Data | 139 |
| Software Testing Levels | 142 |
| Testing Methods | 145 |
| Black Box Testing | 145 |
| White Box Testing | 146 |
| Functional Testing | 147 |
| Non-functional Testing | 148 |
| Path Testing | 148 |
| Quality Management | 148 |
| Quality Assurance | 149 |
| Quality Planning | 149 |
| Quality Control | 150 |
| Software Measurement and Metric | 150 |
| Software Measurement Process | 151 |
| Product Metrics | 151 |
| CMM (Capability Maturity Model) | 152 |
| ❑ Exercise | 154 |

10

Chapter

SOFTWARE EVOLUTION & MAINTENANCE

| | |
|----------------------------------|-----|
| Software Evolution | 156 |
| Program Evolution Dynamics | 157 |
| Software Maintenance | 158 |

| | |
|--|-----|
| | 159 |
| Conducting Maintenance | 159 |
| Types of Maintenance | 161 |
| The Cost of Maintenance | 162 |
| Factors Influencing Maintenance Cost | 163 |
| Maintenance Effort Distribution | 163 |
| Maintenance Cost | 164 |
| Maintenance Prediction | 166 |
| Legacy System | 166 |
| Legacy System Components | 167 |
| Software Re-engineering | 168 |
| Software Reengineering Process | 170 |
| Configuration Management | 172 |
| Change Management | 173 |
| Version Management | 175 |
| Release Management | 175 |
| Factors Influencing Release Strategy | 176 |
| System Building | 176 |
| <input type="checkbox"/> Exercise | 176 |

11

Chapter

MANAGING SOFTWARE PROJECT

| | |
|---|-----|
| Introduction | 178 |
| Software Project Manager | 178 |
| Management Activities | 179 |
| Project Planning | 180 |
| Types of Plan | 181 |
| Mile Stones and Deliverable | 182 |
| Milestones | 182 |
| Project Scheduling | 182 |
| Bar Charts and Activity Networks | 183 |
| Risk Management | 185 |
| Risk Management Process | 186 |
| Cost Estimation Techniques | 188 |
| Software Pricing Factors | 189 |
| Estimation Techniques | 189 |
| Top down Approach | 189 |
| Bottom up Approach | 190 |
| COCOMO Model | 191 |
| The Basic COCOMO | 193 |
| COCOMO 2 | 194 |
| <input type="checkbox"/> Exercise | 196 |
| <input type="checkbox"/> MCQ FOR SOFTWARE ENGINEERING | 197 |
| <input type="checkbox"/> Answers | 204 |
| <input type="checkbox"/> Bibliography | 205 |
| <input type="checkbox"/> TU Questions | 206 |