Software Engineering (BHCS09) Discipline Specific Core Course - (DSC)

Credit: 06

Course Objective

The course introduces fundamental Software Engineering approaches and techniques for software development. The students also develop a case study using appropriate software model.

Course Learning Outcomes

On successful completion of the course, students will be able to:

1. Analyse and model customer's requirements and model its software design.
2. Use suitable software model for the problem at hand.
3. Estimate cost and efforts required in building software.
4. Analyse and compute impact of various risks involved in software development.
5. Design and build test cases, and to perform software testing.

Detailed Syllabus

Unit 1

Introduction: Software Engineering - A Layered Approach; Software Process – Process Framework, Umbrella Activities; Process Models – Waterfall Model, Incremental Model, and Evolutionary process Model (Prototyping, Spiral Model); Introduction to Agile – Agility Principles, Agile Model – Scrum.

Unit 2

Software Requirements Analysis and Specifications: Use Case Approach, Software Requirement Specification Document, Flow oriented Modeling, Data Flow Modeling, Sequence Diagrams

Unit 3

Design Modeling: Translating the Requirements model into the Design Model, The Design Process, Design Concepts - Abstraction, Modularity and Functional Independence; Architectural Mapping using Data Flow.

Unit 4

Software Metrics and Project Estimations: Function based Metrics, Software Measurement, Metrics for Software Quality; Software Project Estimation (FP based estimations, COCOMO II Model); Project Scheduling (Timeline charts, tracking the schedule).

Unit 5

Quality Control and Risk Management: Quality Control and Quality Assurance, Software Process Assessment and Improvement Capability Maturity Model Integration (CMMI); Software Risks, Risk Identification, Risk Projection and Risk Refinement, Risk Mitigation, Monitoring and Management.

Unit 6

Software Testing: Strategic Approach to Software Testing, Unit Testing, Integration Testing, Validation Testing, System Testing; Black-Box and White Box Testing, Basis Path Testing.

Practical

Practical problems related to

1. Requirement Analysis, Creating a Data Flow, Data Dictionary, Use Cases
3. Computing FP, Effort, Schedule, Risk Table, Timeline chart
4. Design Engineering, Architectural Design, Data Design, Component Level Design
5. Testing, Basis Path Testing

Sample Projects:

1. Criminal Record Management: Implement a criminal record management system for jailers, police officers and CBI officers
2. DTC Route Information: Online information about the bus routes and their frequency and fares
3. Car Pooling: To maintain a web based intranet application that enables the corporate employees within an organization to avail the facility of carpooling effectively.
4. Patient Appointment and Prescription Management System
5. Organized Retail Shopping Management Software
6. Online Hotel Reservation Service System
7. Examination and Result computation system
8. Automatic Internal Assessment System
9. Parking Allocation System
10. Wholesale Management System

References

1. Aggarwal, K. K., & Singh, Y. (2007). *Software Engineering*. 3rd edition. New Age International Publishers.

2. Pressman, R. S., & Maxim, B. R. (2015). *Software Engineering: A Practitioner's Approach*. 8th edition. McGraw-Hill.

Additional Resources

1. Jalote, P. (2005). *An Integrated Approach to Software Engineering*. 3rd edition. Narosa Publishing House.

2. Schwaber, K. & Sutherland, J. (2016). *The Definitive Guide to Scrum: The Rules of the Game*. [https://www.scrumguides.org/docs/scrumguide/v1/scrum-guide-us.pdf]

3. Sommerville. (2011). *Software Engineering*. 9th edition. Addison Wesley.

Course Teaching Learning Process

- Use of ICT tools in conjunction with traditional class-room teaching methods
- Interactive sessions
- Class discussions

Tentative weekly teaching plan is as follows:

Week	Content
1	Software - Nature of Software, Software Application Domains, Legacy Software; Software Engineering - A Layered Approach; Software Process – Process Framework, Framework and Umbrella Activities
2	Process Models – Waterfall Model, Incremental Model, and Evolutionary process Model (Prototyping, Spiral Model);
3	Introduction to Agile – Agility, Cost of Change, Agility Principles
4	Agile Model - Scrum; Software Process Assessment and Improvement - Capability Maturity Model Integration (CMMI).
5	Requirements Modeling - Requirements Modeling Approaches, Flow oriented

	Modeling, Data Flow Modeling,
6	Control Flow Model, Control Specification, Process Specification, Behavioral Model, State Diagram, Sequence Diagrams;
7	Design Modeling - Design Concepts, Translating requirements model into design model, Design Process, Abstraction, Architecture, Separation of concerns, Modularity, Information hiding, Functional Independence,
8	Refinement, Refactoring; Architectural Mapping using Data Flow.
9	Risk Management- Software Risks, Risk Identification, Risk Projection and Risk Refinement, Risk Mitigation, Monitoring and Management.
10	Function based Product Metrics, Software Quality Metrics;
11	Estimation for Software Project, Project Scheduling, Quality - Software Quality, McCall's Quality Factors, ISO 9126 Quality Factors, Achieving Software Quality;
12	Cost Impact of Software Defects, Defect Amplification and Removal, Formal Technical Reviews; Software Quality Assurance – SQA Tasks.
13-14	Software Testing - Strategic Approach to Software Testing, Unit Testing, Integration Testing, Validation Testing, System Testing;
15	Black-Box and White Box Testing, Basis Path Testing

Assessment Methods

Written tests, assignments, quizzes, presentations, projects as announced by the instructor in the class.

Keywords

Software models, requirement analysis, software design and testing, software risks and costs