MCA – 203 **Software Engineering** **L T**
3 1

Course Objectives:

Introduce students to software development life cycle and models for developing and effective and efficient software. Identify software requirements for manual or automated real-world systems. Compare and contrast software process models and software development methodologies. Provide the student with the opportunity to practice software development skills. Provide students with opportunities to develop basic computing skills with respect to preparation of documents and also to be able to check the correctness of a software design.

UNIT – I

Evolving Role of Software, Software Engineering, Changing nature of Software, Software Myths, Terminologies, Role of management in software development

Software Process Models: Software Process, Generic Process Model –Framework Activity, Task Set and Process Patterns; Process Lifecycle, Prescriptive Process Models, Project Management, Component Based Development, Aspect-Oriented Software Development, Formal Methods, Agile Process Models –Extreme Programming (XP), Adaptive Software Development, Scrum, Dynamic System Development Model, Feature Driven Development, Crystal, Web Engineering

Software Life Cycle Models: Build & Fix Model, Water Fall Model, Incremental Process Model, Evolutionary Process Models, Unified Process, Comparison of Models, Other Software Processes, Selection of a Model

Software Requirements Analysis & Specifications: Requirements Engineering, Types of Requirements, Feasibility Studies, Requirements Elicitation, Developing Use Cases, Requirements - Analysis Documentation, Software Requirement and Specification (SRS) Document. Validation and Management .

UNIT – II

Software Architecture: Its Role, Views, Component & Connector View and its architecture style, Architecture Vs Design, Deployment View & Performance Analysis, Software Quality: McCall's Quality Factors, ISO 9126 Quality Factors, Quality Control, Quality Assurance, Risk Management, Risk Mitigation, Monitoring and (RMMM); Software Reliability.

Software Project Planning: Relationship to lifecycle, project planning, project control, project organization, configuration management, version control, quality assurance, metrics Size estimation, Cost Estimation, COCOMO, COCOMO – II, Software Risk Management Project Scheduling and Staffing, Time-line Charts.

UNIT – III

Function Oriented Design: Design principles, Module level Concepts, Notation & Specification, Structured Design Methodology and Verification.

Object-Oriented Design: OO Analysis & Design, OO Concepts, Design Concepts, UML – Class Diagram, Sequence & Collaboration Diagram, Other diagrams & Capabilities,

Design Methodology – Dynamic and Functional Modeling, Internal Classes & Operations
Detailed Design: PDL, Logic/Algorithm Design, State Modeling of Classes, Verification – Design Walkthroughs, Critical Design Review, Consistency Checkers.

UNIT – IV

Software Testing: Verification and Validation; Error, Fault, Bug and Failure; Unit and Integration Testing; White-box and Black-box Testing; Basis Path Testing, Control Structure Testing, Deriving Test Cases, Alpha and Beta Testing; Regression Testing, Performance Testing, Stress Testing.
Software Configuration Management: Change Control and Version Control; Software Reuse, Software Re-engineering, Reverse Engineering.

Text Book:

1. R.S. Pressman, **Software Engineering: A Practitioner's Approach** (6th ed.), McGraw-Hill, 2006

Reference Books-

1. P. Jalote, **An Integrated Approach to Software Engineering** (3rd ed.), Narosa Publishing House, 2005
2. K.K. Aggarwal and Y. Singh, **Software Engineering** (revised 2nd ed.), New Age International Publishers, 2006
3. Sommerville, **Software Engineering** (6th ed.), Pearson Education, 2004 Douglas Bell,
4. **Software Engineering for Students** (4th ed.), Addison-Wesley, 2005.

Course Outcomes:

By the end of the course, Student will able to :
CO 1: Describe the software development life cycle as well as describing the various software development model.
CO 2: Illustrate the software requirement specification, and system design.
CO 3: Understand the advantages and disadvantages of each model.
CO 4: Learn project planning and can apply in course projects.
CO 5: Understand type of testing and enhance skills in the field of software testing.
CO 6: Design high quality software products.
CO 7: Learn skill of software requirement specification and software quality assurance techniques.

Note: In each theory paper, nine questions are to be set. Two questions areto set from each Unit and candidate is required to attempt one question from each unit. Question number nine will be compulsory, which will be of short answer type with 5-10 parts, out of the entire syllabus. In all, five questions are to be attempted.