COURSE CODE**: CSC2036**	L-T-P**: 4-1-1**
COURSE NAME**: SOFTWARE ENGINEERING**	CONTACT HOURS/WEEK**: 7**
COURSE TYPE**: CORE**	TOTAL MARKS**: 100 (**INTERNAL**: 60,** EXTERNAL**: 40)**
NUMBER OF CREDITS**: 6**	NATURE**: GRADED**

COURSE OBJECTIVES:

1. To provide students the knowledge of SE challenges, Software process, S/W development process model and problem analysis.
2. The give students the concepts of role of software architecture, architecture views, software cost estimation model, quality plan, and risk management.
3. To familiarize students with concepts of module level concept, OO Analysis and OO Design, UML, Coding process, refactoring, verification, testing fundamentals.

COURSE PREREQUISITE:

- Basic knowledge of database management system.

COURSE OUTCOMES:

At the end of the course, students will be able to:

- Compare different software development processes and their challenges.
- Create software require specification and translate it into an implementable design, following a structured and organize process.
- Implement different software estimation metrics such as cost, effort size, staffing etc.
- Make effective use of UML, along with design strategies such as defining software architecture, separation of concerns and design patterns.

COURSE CONTENT:

Unit No & Name	Components of the Unit	No of contact hours	Marks
UNIT-I: **Software challenges and Software process**	• Problem domain, SE challenges • SE approach. Software process, Characteristics of SW process, SW development process model.	10	10
UNIT-II: **Software requirement and specification**	• SW requirement, problem analysis, requirement specification. • Functional specification, validation, matrices.	10	15

UNITIII:**Software architecture views and cost estimation model.**	• Role of SW architecture, architecture view, component and connector view, style for C&C view. • Process planning, Effort estimation, Software Cost Estimation based on COCOMO II cost model. • Scheduling and staffing. • SW configuration management plan, quality plan, risk management, project monitoring plan.	30	35
UNIT-IV: **Design principles and Methodology**	• Design principle • Module level concept • Design notation and specification, • Structured design methodology verification. • OO Analysis and OO Design. OO Design concept, UML. OO Design methodology.	25	25
UNIT-V: **Detail Design and Testing fundamentals**	• Detail design and PDL • Verification, Metrices, Programming principles and guidelines, coding process, refactoring, verification. Testing fundamentals.	15	15
	Total	90	100

TEXTBOOKS/ RECOMMENDED READINGS:

- Jalote P. ; *An integrated Approach to Software Engineering*; Narosa Publishing House
- Patton R.; *Software Engineering;* Pearson Education.
- Agarwal K. K., Singh Y,; *Software Engineering*; New Age International Publisher.
- Sommerville I.; *Software Engineering*; Pearson Education (Addison Wesley)
- Pressman R.S.; *Software Engineering: A practitioner's Approach*; McGraw Hill.

COURSE ASSESSMENT DETAILS:

Internal assessment: Class tests, Assignments, Laboratory tests
External assessment: End Semester Examination

DEPARTMENT OF COMPUTER SCIENCE

Gopinath Bordoloi Nagar, Gauhati University

Guwahati-781014, Assam, India

LESSON PLAN

Subject Name : **Software Engineering**

Paper Code : **CSC2036/INF2036** **Session:** **2019-2020**

Program Name : **M.Sc. (CS/IT)** **Semester:** **Second**

Faculty Name : **Dwipen Laskar**

Date : **Jan, 2020 to June, 2020**

Detailed Lesson Plan

UNIT-I (Software challenges and Software process)

Lecture No	Topics to be Covered
1	Problem domain: Software vs Program, Process vs Product, Exploratory Style of Programming
2	SE Challenges, Software Crisis, Software Process Model, SE approaches
3	Characteristics of SW process, Emergence of Software Engineering, Notable Changes in Software Development Practices
4	SW development process model: basic concepts, Build and Fix model, Waterfall model, Classical Waterfall Model
5	Iterative Waterfall Model, V-Model, Prototyping Model
6	Incremental Developmental Model, Evolutionary Model
7	Spiral Model, Agile Software Development, Comparison of Different Life Cycle Models

UNIT-II (Software requirement and specification)

13	SW requirement: Requirement Gathering and Analysis, Software Requirement Specification(SRS):-users of SRS, Characteristic of Good and bad SRS
14	Functional Specification: how to identify and Document, Functional Requirements, Traceability
15	Representation of complex logic in SRS, Decision Tree and Decision Table
16	Formal Technique
17	Axiomatic Specification, validation, matrices

UNIT-III (Software architecture views and cost estimation model)

18	Role of SW architecture, architecture view, component and connector view, style for C&C view.

19	Software Project Management Complexities, Responsibilities of SW Project Manager
20	Process planning, Software Planning, Sliding Window Planning
21	Software Size Estimation: LOC, Function Point Metric, Feature Point Metric
22	Empirical Estimation Techniques, Expert Judgment, Delphi Cost Estimations
23	Effort estimation,
24	Cost Estimation based on COCOMO: Basic, Intermediate and Complete COCOMO
25	Staffing Level Estimation: Norden's Work, Putnam's Work, Jensen's Model
26	Software Scheduling and staffing: Work Breakdown Structure, CPM,
27	Activity Network, PERT Charts, Gantt Chart
28	Organization Structure: Chief Programmer team, Democratic Team and Mix Team
29	Team Structure: Project Format, Functional Format, Matrix Format
30	Risk Management, Risk Identification, Types of Risks, Risk Abatement Policies
31	SW configuration management plan, Source Code Control System, RCS, Project monitoring plan.
32	Software Reliability, Hardware vs Software Reliability, Reliability Metrics
33	Reliability Growth Model, Statistical Testing, Quality plan: McCall's Quality factors, ISO9000
34	SEI Capability Maturity Model, Comparison of SEI/CMM and ISO 9000

UNIT-IV (Design principles and Methodology)

35	Design principle, Outcome of design process, classification of design activities and design methodologies
36	Characteristics of good design, Module level concept, Design notation and specification
37	Layered architecture of modules, Cohesion and Coupling
38	Overview of SA/SD, Structured Analysis (DFD)
39	Structured Analysis (DFD): Context Level Diagram, Types of DFDs, Notations of DFD, Data dictionary
40	Transformation of DFD into Structure Chart, Detailed Design
41	OO Analysis and OO Design: OO Design concept, Origin of UML, Evolution of UML, UML Diagrams
42	UML Diagrams: Use Case Diagram, Representation, Factoring, Use Case Packaging
43	UML Diagrams: Class Diagram, Object Diagram
44	UML Diagrams: Interactive Diagrams, State Chart Diagram, Deployment and Component Diagram

45	Programming principles and guidelines, coding process, Code Walkthrough, Code Inspection and Clean Room Testing, Software Documentation
46	Software Testing: basic concepts, Unit Testing, Integration and System Testing, Acceptance Testing
47	Black Box Testing-Equivalence Partitioning, Boundary Value Analysis
48	White Box Testing-Statement Coverage, Branch Coverage
49	Condition Coverage, Path Coverage, McCabe's Cyclomatic Complexity, Data Flow based Testing, Mutation Testing
50	Debugging Approaches, Guidelines, Program Analysis Tools: Static and Dynamic

UNIT-V (Detail Design and Testing fundamentals)

52	Detail design and Process Design Language (PDL)
53	Software Verification and Validation,
54	Code Refactoring concepts and Techniques
55	Integration Testing and System Testing: Smoke Testing, Performance Testing, Error Seeding
56	Software Maintenance: Reverse Engineering, Estimation of Maintenance Cost, CASE tools

(**Dwipen Laskar**)
(Assistant Professor, Dept. of Computer Sc., GU)