22CA2E1: SOFTWARE ENGINEERING

Course Name	Software Engineering		L	T	P	C	CIA	SEE	TM
Course Code	22CA2E1		4	0	0	4	30	70	100
Year of Introduction: 1991	Year of Offering: 2022	Year of Revision: 2022	Percentage of Revision: 10						
L-Lecture, T-Tutorial, **P**-Practical, **C**-Credits, **CIA**-Internal Marks, **SEE**-External Marks, **TM**-Total Marks									

Course Description and Purpose:
Software Engineering (22CA2E1) is a course that illustrates *Process Models*, *Agile Development*, *Core Principles*, *Requirements Modeling*, *Data Modeling*, *Software Quality Assurance*, *Software Testing Strategies*, *Testing Conventional Applications*, *Project Management Concepts*, *Process and Project Metrics*, *Formal Modeling and Verification* and *Estimation for Software Project.*

Course Objectives: The course will help the students to understand, learn and build *Process Models*, *Agile Models*, *Core Principles*, *Requirement Models*, *Data Models*, *Software Quality Assurance Procedures*, *Software Testing Strategies*, *Strategies to Test Conventional Applications*, *Project Management Concepts*, *Process and Project Metrics*, *Formal Modeling and Verification* and *Models to estimate Software Projects.*

Specific objectives include:

- To understand various *Software Engineering Methods*, *Practices*, *Process Models* and *Agile Development Strategies*.
- To understand and apply *Core Principles*, *Requirements & Modeling* Concepts.
- To understand and apply different *Software Testing Approaches* and various aspects of *Software Quality Assurance*.
- To understand and apply *Process & Project Management* Concepts.
- To understand and apply *Software Estimates for Projects* & apply *Formal Methods Modeling*.

Course Learning Outcomes:
Upon successful completion of the course, the student will be able to:
CO1: Understand various *Software Engineering Methods*, *Practices*, *Process Models* and *Agile Development Strategies.*
CO2: Understand and apply *Core Principles*, *Requirements & Modeling* Concepts.
CO3: Understand and apply different *Software Testing Approaches* and various aspects of *Software Quality Assurance*.
CO4: Understand and apply *Process & Project Management* Concepts.
CO5: Understand and apply *Software Estimates for Projects* & apply *Formal Methods Modeling*.

UNIT I (12 Hours)

Software and Software Engineering: The Nature of Software: Defining Software, Software Application Domains, Legacy Software, The Unique Nature of Web Apps, Software Engineering, The Software Process, Software Engineering Practices: The Essence of Practice, General Principles, Software Myths.

Process Models: A Generic Process Model: Defining a Framework Activity, Identifying a Task

Set, Process Patterns, Process Assessment and Improvement, Prescriptive Process Models: The Waterfall Model, Incremental Process Models, Evolutionary Process Models, Concurrent Models, A Final Word on Evolutionary Processes, Specialized Process Models: Component-Based Development, The Formal Methods Model, Aspect-Oriented Software Development, The Unified Process: A Brief History, Phases of the Unified Process, Personal and Team Process Models: Personal Software Process (PSP), Team Software Process (TSP).

Agile Development: What Is Agility, Agility and the Cost of Change, What Is an Agile Process: Agility Principles, The Politics of Agile Development, Human Factors, Extreme Programming (XP): XP Values, The XP Process, Industrial XP, The XP Debate, Other Agile Process Models: Adaptive Software Development (ASD), Scrum, Dynamic Systems Development Method (DSDM), Crystal,Feature Driven Development (FDD), Lean Software Development (LSD), Agile Modeling (AM), AgileUnified Process (AUP).

UNIT II (12 Hours)

Principles that Guide Practice: Core Principles: Principles That Guide Process, Principles That Guide Practice**,** Principles That Guide Each Framework Activity: Communication Principles, Planning Principles, Modeling Principles, Construction Principles, Deployment Principles.

Requirements Modeling: Scenarios, Information, and Analysis Classes: Requirements Analysis: Overall Objectives and Philosophy, Analysis Rules of Thumb, Domain Analysis, Requirements Modeling Approaches, Scenario-Based Modeling: Creating a Preliminary Use Case, Refining a Preliminary Use Case, Writing a Formal Use Case, UML Models That Supplement the Use Case: Developing an Activity Diagram, Swim lane Diagrams.

Data Modeling Concepts: Data Objects, Data Attributes, Relationships, Class-Based Modeling: Identifying Analysis Classes, Specifying Attributes, Defining Operations, Class-Responsibility-Collaborator (CRC) Modeling, Associations and Dependencies, Analysis Packages.

UNIT III (12 Hours)

Software Quality Assurance: Background Issues, Elements of Software Quality Assurance, SQA Tasks, Goals, and Metrics: SQA Tasks, Goals, Attributes, and Metrics, Formal Approaches to SQA, Statistical Software Quality Assurance: A Generic Example, Six Sigma for Software Engineering, Software Reliability : Measures of Reliability and Availability, Software Safety, The ISO 9000 Quality Standards, The SQA Plan.

Software Testing Strategies: A Strategic Approach to Software Testing : Verification and Validation, Organizing for Software Testing, Software Testing Strategy-The Big Picture, Criteria for Completion of Testing, Strategic Issues, Test Strategies for Conventional Software: Unit Testing, Integration Testing, Test Strategies for Object-Oriented Software: Unit Testing in the OO Context, Integration Testing in theOO Context, Test Strategies for Web Apps, Validation Testing: Validation-Test Criteria, Configuration Review, Alpha and Beta Testing, System Testing: Recovery Testing, Security Testing, Stress Testing, Performance Testing, Deployment Testing, The Art of Debugging: The Debugging Process, Psychological Considerations, Debugging Strategies, Correcting the Error

Testing Conventional Applications: Software Testing Fundamentals, Internal and External Views of Testing, White-Box Testing, Basis Path Testing: Flow Graph Notation, Independent

Program Paths, Deriving Test Cases, Graph Matrices, Control Structure Testing: Condition Testing, Data Flow Testing, Loop Testing, Black-Box Testing: Graph-Based Testing Methods, Equivalence Partitioning, Boundary Value Analysis, Orthogonal Array Testing.

UNIT IV (12 Hours)

Project Management Concepts: The Management Spectrum: The People, The Product, The Process, The Project, People: The Stakeholders, Team Leaders, The Software Team, Agile Teams, Coordination and Communication Issues, The Product: Software Scope, Problem Decomposition, The Process: Melding the Product and the Process, Process Decomposition, The Project, The W5HH Principles.

Process and Project Metrics: Metrics in the Process and Project Domains: Process Metrics and Software Process Improvement, Project Metrics, Software Measurement: Size-Oriented Metrics, Function-Oriented Metrics, Reconciling LOC and FP Metrics, Object-Oriented Metrics, Use-Case– Oriented Metrics, Web App Project Metrics, Metrics for Software Quality: Measuring Quality, Defect Removal Efficiency.

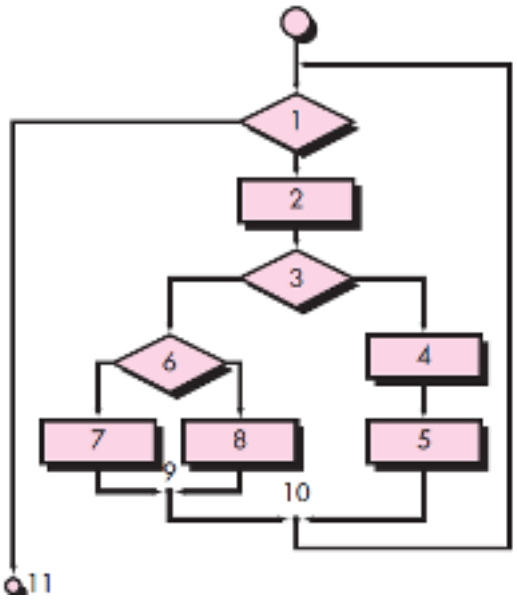
UNIT V (12 Hours)

Formal Modeling And Verification: The Cleanroom Strategy, Functional Specification: Black-Box Specification, State-Box Specification, Clear-Box Specification, Cleanroom Design: Design Refinement, Design Verification, Cleanroom Testing: Statistical Use Testing, Certification, Formal Methods Concepts, Applying Mathematical Notation for Formal Specification, Formal Specification Languages: Object Constraint Language (OCL), The Z Specification Language.

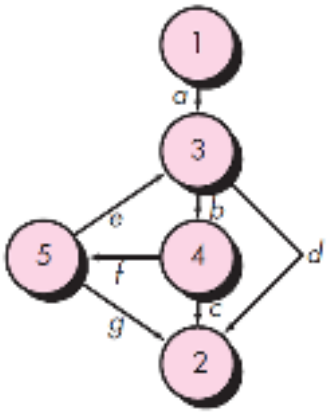
Estimation for Software Projects: Resources: Human Resources, Reusable Software Resources, Environmental Resources, Software Project Estimation, Decomposition Techniques: Software Sizing, Problem-Based Estimation, An Example of LOC-Based Estimation, An Example of FP-Based Estimation, Empirical Estimation Models: The Structure of Estimation Models, The COCOMO II Model, The Software Equation, Estimation for Object-Oriented Projects.

Case Studies:

- Draw example for Process Pattern when requirements are uncertain.
- Draw UML use case diagram for Safehome Security Function.
- Draw UML Activity Diagram for Access camera surveillance via the Internet - display camera views function.
- Draw UML Swimlane Diagram for Access camera surveillance via the Internet - display camera views function.
- Draw UML Class Diagram for Floor Plan.
- Draw UML Package for specifying Environment, Characters of the Game and Rules of the Game.
- Draw Level 1 DFD for Safehome Security Function
- Draw State diagram for Safehome Security Function
- Draw Sequence Diagram (partial) for the Safehome Security Function
- A UML Deployment Diagram for Safehome Security Function.
- Draw Flow Graph for Flow Chart and find the Cyclomatic Complexity.



- Draw the Graph Matrix for the Flow Graph



Flow graph

- Draw Generalization diagram by specifying Structural Constraint.
- Specify sample (a) Project Metrics (b) Product Metrics
- Specify (i) Decision Table (ii) Decision Tree in Block Box Testing
- Draw the Block Diagram for Block Handler and also specify the logic using Object Constraint Language (OCL)
  1. No block will be marked as both unused and used.
  2. All the sets of blocks held in the queue will be subsets of the collection of currently used blocks
  3. No elements of the queue will contain the same block numbers.
  4. The collection of used blocks and blocks that are unused will be the total collection of blocks that make up files.
  5. The collection of unused blocks will have no duplicate block numbers.
  6. The collection of used blocks will have no duplicate block numbers.
  7. Using Z Specification Language describes the state of the block handler and the data invariant:

Reference Text Books:

1. Roger S Pressman, Software Engineering - A Practitioner's Approach, Ninth Edition, McGraw - Hill, A Business Unit of The McGraw-Hill Companies, Inc., 2020.
2. Roger S Pressman, Software Engineering - A Practitioner's Approach, Seventh Edition, McGraw - Hill, A Business Unit of The McGraw-Hill Companies, Inc., 2010.
3. Sommerville, Software Engineering, 7th Edition, Pearson Education, 2004.
4. S.A.Kelkar, Software Engineering - AConcise Study, PHI, January 2007.
5. Waman, Software Engineering, TMH, June 2004.
6. AH Behforooz and Frederick J.Hudson, Software Engineering Fundamentals, Oxford, 2008.

PARVATHANENI BRAHMAYYA SIDDHARTHA COLLEGE OF ARTS & SCIENCE

(An Autonomous College in the jurisdiction of Krishna University)

M.C.A Second Semester

Course Name: Software Engineering

Course Code: 22CA2E1

(w.e.f admitted batch 2022-23)

Time: 3 Hours **Max Marks: 70**

SECTION-A

Answer ALL questions. All Questions Carry Equal Marks. (5×4 = 20 Marks)

1. (a) What are various aspects of is *PSP* and *TSP*? (CO1, L1)

(or)

(b) What is *SCRUM*? Explain it in detail. (CO2, L1)

2. (a) What are the phases of *Extreme Programming (XP)*? (CO2, L1)

(or)

(b) What is *Class-Based Modeling*? Explain it by writing Class Diagram (CO2,L1)

3. (a) What is *Software Reliability*? Explain in detail. (CO3,L1)

(or)

(b) What is *Alpha* and *Beta* Testing? Explain in detail. (CO3,L1)

4. (a) List W5HH Principles. (CO4,L1)

(or)

(b) What is *Use Case Diagram*? Demonstrate with example. (CO4,L2)

5. (a) State various *resources* of Information System. (CO5,L5)

(or)

(b) What is *Software Sizing*? Explain it (CO5,L5)

SECTION-B

Answer ALL questions. All Questions Carry Equal Marks. (5×10 = 50 Marks)

6. (a) Explain various types of *Software Myths*. (CO1, L2)

(or)

(b) Explain *Incremental Process Models*. (CO1,L2)

7. (a) List *(i) Planning Principles (ii) Modeling Principles*. (CO2, L4)

(or)

(b) Examine various aspects of *Scenario-Based Modeling.* (CO2, L4)

8. (a) Develop various test strategies to test *Conventional Software.* (CO3,L3)

(or)

(b) Develop various strategies for *White Box Testing.* (CO3,L3)

9. (a) Discuss the *Management Spectrum* in detail. (CO4, L6)

(or)

(b) Discuss *(i) Size-Oriented Metrics (ii) Function-Oriented Metrics* in detail. (CO4, L6)

10. (a) Explain *Functional Specification* of *Cleanroom Strategy*. (CO5,L5)

(or)

(b) Explain *(i) The COCOMO II Model (ii) The Software Equation* of EmpiricalEstimation Models. (CO5, L5)