Total No. of Questions: 6 *Total No. of Printed Pages:3*

Enrollment No......................................

Faculty of Engineering

End Sem (Even) Examination May-2022

CA5CO25 Software Engineering Principles

Programme: MCA Branch/Specialisation: Computer Application

Duration: 3 Hrs. **Maximum Marks: 60**

Note: All questions are compulsory. Internal choices, if any, are indicated. Answers of Q.1 (MCQs) should be written in full instead of only a, b, c or d.

Q.1 i. What is the core of software engineering? **1**

(a) Requirements Definition, Design Representation, Knowledge Capture and Quality Factors
(b) Managing Complexity, Managing Personnel Resources, Managing Time and Money and Producing Useful Products
(c) Maintaining Configurations, Organizing Teams, Channeling Creativity and Planning Resource Use
(d) Time/Space Tradeoffs, Optimizing Process, Minimizing Communication and Problem Decomposition

ii. Which of the following is a life-cycle concern? **1**

(a) Testing (b) Portability (c) Planning (d) Programming

iii. The SRS is said to be consistent if and only if **1**

(a) Its structure and style are such that any changes to the requirements can be made easily while retaining the style and structure.
(b) Every requirement stated therein is one that the software shall meet.
(c) Every requirement stated therein is verifiable.
(d) No subset of individual requirements described in it conflict with each other.

iv. Most software continues to be custom-built because **1**

(a) Component reuse is common in the software world.
(b) Reusable components are too expensive to use.
(c) Software is easier to build without using someone else's components.
(d) Off-the-shelf software components are unavailable in many application domains.

P.T.O.

v. Which of the following are not recognized process flow types? **1**
(a) Concurrent process flow (b) Iterative process flow
(c) Linear process flow (d) Both (b) and (c)

vi. Which type of DFD concentrates on the system process, and flow of data in the system? **1**
(a) Physical DFD (b) Logical DFD
(c) Flowchart DFD (d) System DFD

vii. The UML supports event-based modelling using **1**
(a) Deployment diagram (b) Collaboration diagram
(c) State chart diagram (d) All of these

viii. Which of the following is true? **1**
(a) Activity diagrams are used to model the processing of data.
(b) Model-driven engineering is just a theoretical concept.
(c) Both (a) and (b)
(d) None of these

ix. How many types of software testing exist? **1**
(a) 1 (b) 2 (c) 3 (d) 4

x. Which of the following testing is also known as white-box testing? **1**
(a) Structural testing (b) Error guessing technique
(c) Design based testing (d) None of these

Q.2 i. What is the need to learn Software Engineering Concepts? **2**
ii. Write the various phases of SDLC. **3**
iii. What is Incremental model? Write the advantages, disadvantages and when to use it. **5**
OR iv. Define “Software Engineering”. Illustrate various phases of life cycle of software development. **5**

Q.3 i. What is a software requirements specification? Write its component. **2**
ii. Define Agile and Scrum. Write the difference and similarity between Agile and Scrum. **8**
OR iii. Explain any four common metrics for agile. **8**

Q.4 i. What is mean by level-0 data flow diagram? **3**
ii. Discuss and compare function-oriented and object-oriented design. **7**
OR iii. What is cohesion and coupling? Write the difference between cohesion and coupling. **7**

Q.5 i. How do you ensure that your use case model is effective? **4**
ii. Explain state machine diagram with the help of an example. **6**
OR iii. Define event, state and transition. Discuss preparation of state chart diagram with example. **6**

Q.6 Attempt any two:
i. Explain bug life cycle or defect life cycle. **5**
ii. What is verification and validation in software testing? **5**
iii. Write the benefits of automation testing. **5**

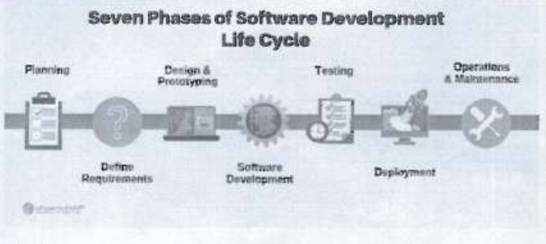
******

Scheme of Marking

	Faculty of Engineering End Sem (Even) Examination May-2022 CA5CO25 Software Engineering Principles
	Programme: MCA Branch/Specialisation:

Note: The Paper Setter should provide the answer wise splitting of the marks in the scheme below.

Q.1	i)	**(B) Managing Complexity, Managing Personnel Resources, Managing Time and Money and Producing Useful Products**	1
	ii)	**(C) Planning**	1
	iii)	(D) No subset of individual requirements described in it conflict with each other	1
	iv)	(D)Off-the-shelf software components are unavailable in many application domains.	1
	v)	(A) Concurrent process flow	1
	vi)	(B) Logical DFD	1
	vii)	(C)State chart diagram	1
	viii	(A) Activity diagrams are used to model the processing of data.	1
	ix)	(B) 2	1
	x)	(A) Structural Testing	1
Q.2	i.	Learning only Computer Languages will not help you to make/develop applications or Software. Software engineering would include the concepts, models one can follow to make software development easier.	2
	ii.	the most common phases of SDLC. • Requirement Analysis • Design • Coding • Testing • Maintenance	3
	iii.	In incremental model the whole requirement is divided into various builds. Multiple development cycles take place here, making the life cycle a "multi-waterfall" cycle. Cycles are divided up into smaller, more easily managed modules. Incremental model is a type of software development model like V-model, Agile model etc. In this model, each module passes through the requirements, design, implementation and testing phases. A working version of software is produced during the first module, so you have working software early on during the software life cycle. Each subsequent release of the module adds function to the previous release. The process continues till the complete system is achieved. Advantages of Incremental model: • Generates working software quickly and early during the software life cycle. • This model is more flexible – less costly to change scope and requirements. • It is easier to test and debug during a smaller iteration. • In this model customer can respond to each built. • Lowers initial delivery cost. • Easier to manage risk because risky pieces are identified and handled during it'd iteration. Disadvantages of Incremental model: • Needs good planning and design. • Needs a clear and complete definition of the whole system before it can be broken down and built incrementally. • Total cost is higher than waterfall.	2 1.5 1.5
OR	iv.	Software engineering is defined as a process of analyzing user requirements and then designing, building, and testing software application which will satisfy those requirements. Software Development Life Cycle is the application of standard business practices to building software applications. It's typically divided into six to eight steps: Planning, Requirements, Design, Build, Document, Test, Deploy, Maintain. Some project managers will combine, split, or omit steps, depending on the project's scope. These are the core components recommended for all software development projects. SDLC is a way to measure and improve the development process. It allows a fine-grain analysis of each step of the process. This, in turn, helps companies maximize efficiency at each stage.	1 4



Planning:
In the Planning phase, project leaders evaluate the terms of the project. This includes calculating labor and material costs, creating a timetable with target goals, and creating the project's teams and leadership structure.
Define Requirements:
Defining requirements is considered part of planning to determine what the application is supposed to do and its requirements. For example, a social media application would require the ability to connect with a friend. An inventory program might require a search feature.
Design and Prototyping:
The Design phase models the way a software application will work. Some aspects of the design include:

- **Architecture**
- **User Interface**
- **Platforms**
- **Programming**
- **Communications**
- **Security**

Q.3 i. A software requirements specification (SRS) is a description of a software system to be developed. It lays out functional and non-functional requirements, and may include a set of use cases that describe user interactions that the software must provide. Components of SRS include: 2

- Create an Outline
- Define the Purpose
- Give an Overview

ii. Agile software development refers to a group of software development methodologies based on iterative development, where requirements and solutions evolve through collaboration between self-organizing cross-functional teams. Agile methods or Agile processes generally promote a disciplined project management process that encourages frequent inspection and adaptation, a leadership philosophy that encourages teamwork, self-organization and accountability, a set of engineering best practices intended to allow for rapid delivery of high-quality software, and a business approach that aligns development with customer needs and company goals. 2

Scrum is a subset of Agile. It is a lightweight process framework for agile development, and the most widely-used one. 2

- A "process framework" is a particular set of practices that must be followed in order for a process to be consistent with the framework. (For example, the Scrum process framework requires the use of development cycles called Sprints, the XP framework requires pair programming, and so forth.)
- "Lightweight" means that the overhead of the process is kept as small as possible, to maximize the amount of productive time available for getting useful work done.

Difference between Agile and Scrum – Agile is a broad spectrum, it is a methodology used for project management while Scrum is just a form of the Agile that describes the process and its steps more concisely. Agile is a practice whereas scrum is a procedure to pursue this practice. 4

The similarity between Agile and Scrum – The Agile involves completing projects in steps or incrementally. The Agile methodology is considered to be iterative in nature. Being a form of Agile, Scrum is same as that of the Agile. It is also incremental and iterative.

OR iii. the detailed description of some common metrics for Agile is as follows: 8
Velocity – Velocity is the average number of points from last 3-4 sprints. It is measured by the summation of the all approved estimates of the stories. It gives an idea of the capacity, progress etc.
Cumulative Flow Diagram – With the help of a cumulative flow diagram, an inspection is done over the uniform workflow. In this diagram/graph, the x-axis represents time whereas the y-axis represents the number of efforts.
Work Category Allocation – Work category allocation is an important factor that gives a quick information of the time investment i.e. where the time is being invested and which task should be given priority as a factor of time.
Time Coverage – It is the time that is given to a code during testing. It is calculated in percentage as a factor of the number of lines of code called by the test suite and the total number of relative lines of code.

Business Value Delivered – It is a term which denotes the working efficiency of the team. The business objectives are assigned numerical values 1,2,3.. and so on, as per the level of priority, complexity, and ROI.
Defect Removal Awareness – It is the factor that helps the team to deliver a quality product. The identification of an active number of defects, their awareness, and removal plays an important role in delivering a high-quality product.
Defect Resolution Time – It is a procedure through which the team members detect the defects (bugs) and set a priority for the defect resolution. The procedure of fixing errors/bugs or defect resolution comprises of multiple processes such as clearing the picture of defect, schedule defect fixation, completing defect fixation, generation, and handling of resolution report.
Sprint Burn Down Matric – The sprint burndown chart is a graph to represent the number of non-implemented or implemented sprints during as Scrum cycle. This matric helps to track the work completed with the sprint.

Q.4 i.

- Highest abstraction level is called Level 0 of DFD.
- It is also called context level DFD.
- It portrays the entire information system as one diagram.

3

ii.

Function Oriented Design :
Function oriented design is the result of focusing attention to the function of the program. This is based on the stepwise refinement. Stepwise refinement is based on the iterative procedural decomposition. Stepwise refinement is a top-down strategy where a program is refined as a hierarchy of increasing levels of details.
Object Oriented Design :
Object oriented design is the result of focusing attention not on the function performed by the program, but instead on the data that are to be manipulated by the program. Thus, it is orthogonal to function -oriented design. Object-oriented design begins with an examination of the real world "things". These things are characteristics individually in terms of their attributes and behaviour.

3

COMPARISON FACTORS	FUNCTION ORIENTED DESIGN	OBJECT ORIENTED DESIGN
Abstraction	The basic abstractions, which are given to the user, are real world functions.	The basic abstractions are not the real world functions but are the data abstraction where the real world entities are represented.
Function	Functions are grouped together by which a higher level function is obtained.	Function are grouped together on the basis of the data they operate since the classes are associated with their methods.
State information	In this approach the state information is often represented in a centralized shared memory.	In this approach the state information is not represented is not represented in a centralized memory but is implemented or distributed among the objects of the system.
Approach	It is a top down approach.	It is a bottom up approach.
Begins basis	Begins by considering the use case diagrams and the scenarios.	Begins by identifying objects and classes

4

OR iii.

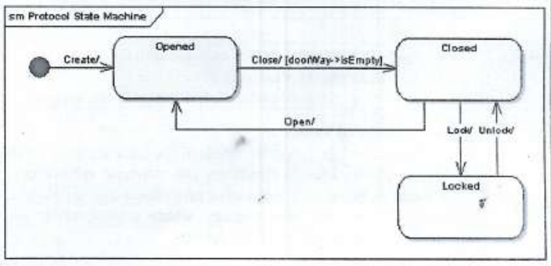
Coupling: The coupling term is used to describe the degree of interdependence between the modules. Lower the coupling between the modules, the better the software is.
Cohesion: The cohesion term is used to describe the degree to which the elements of the module are related. Cohesion is a glue that connects all the modules together. All the elements of the module work together to perform a single task.

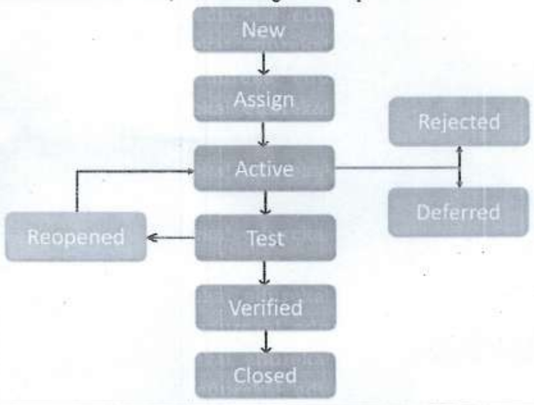
The major difference between cohesion and coupling is that cohesion deals with the interconnection between the elements of the same module. But, coupling deals with the interdependence between software modules.

3

Cohesion	Coupling
Cohesion is defined as the degree of relationship between elements of the same module.	Coupling is defined as the degree of interdependence between the modules.
It's an intra-module approach	It's an inter-module approach
High cohesion is preferred due to improved focus on a particular task.	Low Coupling is preferred as it results in less dependency between the modules.
Cohesion is used to indicate a module's relative functional strength.	Coupling is used to indicate the relative independence among the modules.
In cohesion, the module focuses on a particular task.	In coupling, a particular module is connected to other modules.
Cohesion is also known by the name 'Intra- module Binding'.	Coupling is also known by the name 'Inter-module binding'.

4

Q.5	i.	Use case models include actors and use cases. An effective model must identify both comprehensively. Yet, analysts usually fail to identify all of the relevant actors. Starting with a separate actor model places a focus on actor analysis. This is the process of identifying as many distinct roles as possible within the scope of the business area or system. Then, each of the actors can be organized by showing actor generalization/specialization (inheritance). Showing actor generalization/specialization in a separate diagram can help keep the use case diagram clean, since generalized actors may not need to be shown initiating use cases. However, it can still be helpful to show this hierarchical organization of actors separately as it aids the viewer of the diagram in understanding how each actor is similar or different. The actor diagram is just one piece of the overall actor analysis. Each actor should be accompanied by a description of the role which can be conveniently maintained in a spreadsheet. Understanding what the actor does at a high level (2-3 sentences) as well as what the actor does not do (differentiation of actors), helps the audience of the model gain a well rounded understanding of the actors involved.	4
	ii.	A state machine diagram models the behaviour of a single object, specifying the sequence of events that an object goes through during its lifetime in response to events.  The door can be in one of three states: "Opened", "Closed" or "Locked". It can respond to the events Open, Close, Lock and Unlock.	6
OR	iii.	Event An event is an occurrence of a stimulus that can trigger a state change and that is relevant to the object or to an application. State All objects will have a state in a system. The current state of an object is a result of the events that have occurred to the object, and is determined by the current value of the object's attributes and the links that it has with other objects. Some attributes and links of an object are significant for the determination of its state while others are not. Transition Movement from one state to another is called a transition, and is triggered by an event.	2

		Preparation of State chart diagram: The preparation of a state chart from a set of interaction diagrams using this behavioural approach has the following sequence of steps. 1. Examine all interaction diagrams that involve each class that has heavy messaging. 2. Identify the incoming messages on each interaction diagram that may correspond to events. Also identify the possible resulting states. 3. Document these events and states on a state chart. 4. Elaborate the state chart as necessary to cater for additional interactions as these become evident, and add any exceptions. 5. Develop any nested state charts if already identified. 6. Review the state chart to ensure consistency with use cases. In particular, check that any constraints that are implied by the state chart are appropriate. 7. Iterate steps 4, 5 and 6 until the state chart captures the necessary level of detail. 8. Check the consistency of the state chart with the class diagram, with interaction diagrams and with any other state charts	~~6~~ 4
Q.6			
	i.	A defect life cycle is a process in which a defect goes through various phases during its entire lifetime. It starts when a defect is found and ends when a defect is closed, after ensuring it's not reproduced.  New Assign Rejected Active Deferred Reopened Test Verified Closed	5

ii.	Verification: It is a static analysis technique. Here, testing is done without executing the code. Examples include – Reviews, Inspection, and walkthrough. Validation: It is a dynamic analysis technique where testing is done by executing the code. Examples include functional and non-functional testing techniques.	5
iii.	Benefits of Automation testing are: • Supports execution of repeated test cases • Aids in testing a large test matrix • Enables parallel execution • Encourages unattended execution • Improves accuracy thereby reducing human-generated errors • Saves time and money	5

******