Faculty Signature CCI Signature HOD Signature

USN | | | | | | | | | | |

Internal Assessment Test 1 – September 2025

Sub:	Software Engineering and Project Management					Sub Code:	BCS501	Branch:	AIML
Date:		Duration:	90 mins	Max Marks:	50	Sem/Sec:	V	OBE	

	Answer any FIVE FULL Questions	**MARKS**	**CO**	**RBT**
1	Explain Boehm Spiral process model with a neat diagram. Mention its advantages and disadvantages.	10M	CO1	L1
2 a)	Explain principles of agile process development.	6M	CO1	L2
2 b)	Explain different types of Software Application Domain.	4M	CO1	L1
3	Explain CRC Modelling and Data Modelling with an example.	10M	CO1	L2
4	Apply the elements of requirement modeling to design a railway reservation system.	10M	CO2	L3
5 a)	Draw the Activity diagram and Swimlane diagram for Safe home System?	5M	CO2	L3
5 b)	How would you apply Industrial Extreme Programming in a large enterprise project, and how would this differ from applying traditional Extreme Programming?	5M	CO3	L1
6	Explain: i. Dynamic System development Method ii. Feature Driven Development	10M	CO3	L2

Faculty Signature CCI Signature HOD Signature

USN | | | | | | | | | | |

Internal Assessment Test 1 – September 2025

Sub:	Software Engineering and Project Management					Sub Code:	BCS501	Branch:	AIML
Date:		Duration:	90 mins	Max Marks:	50	Sem/Sec:	V	OBE	

	Answer any FIVE FULL Questions	**MARKS**	**CO**	**RBT**
1	Explain Boehm Spiral process model with a neat diagram. Mention its advantages and disadvantages.	10M	CO1	L1
2 a)	Explain principles of agile process development.	6M	CO1	L2
2 b)	Explain different types of Software Application Domain.	4M	CO1	L1
3	Explain CRC Modelling and Data Modelling with an example.	10M	CO1	L2
4	Apply the elements of requirement modeling to design a railway reservation system.	10M	CO2	L3
5 a)	Draw the Activity diagram and Swimlane diagram for Safe home System?	5M	CO2	L3
5 b)	How would you apply Industrial Extreme Programming in a large enterprise project, and how would this differ from applying traditional Extreme Programming?	5M	CO3	L1
6	Explain: i. Dynamic System development Method ii. Feature Driven Development	10M	CO3	L2

USN | | | | | | | | | | |

Internal Assessment Test 1 – September 2025

Sub:	Software Engineering and Project Management					Sub Code:	BCS501	Branch:	AIML
Date:		Duration:	90 mins	Max Marks:	50	Sem/Sec:	V	OBE	

	Answer any FIVE FULL Questions	MARKS	CO	RBT
1	Explain Boehm Spiral process model with a neat diagram. Mention its advantages and disadvantages.	10M	CO1	L1

Explanation (4M)

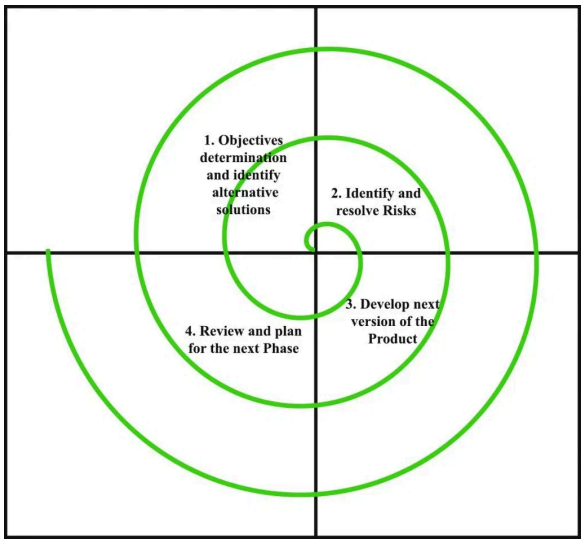
The Spiral model is an evolutionary software process model originally proposed by Barry Boehm. It is notable for coupling the iterative nature of prototyping with the controlled and systematic aspects of the waterfall model. The model is defined as a risk-driven process model generator used to guide multi-stakeholder concurrent engineering of software-intensive systems.

The Spiral model's two main distinguishing features are its cyclic approach for incrementally growing a system's degree of definition and implementation while simultaneously decreasing its degree of risk. Risk is explicitly considered as the team makes each revolution around the spiral.

Software is developed in a series of evolutionary releases. During early iterations, the release might be a model or a prototype. During later iterations, progressively more complete versions of the engineered system are produced.

The software team performs activities implied by a circuit around the spiral in a clockwise direction, beginning at the center. A set of anchor point milestones (a combination of work products and conditions attained along the path) are noted for each evolutionary pass to ensure stakeholder commitment to feasible solutions.

Diagram (2M)



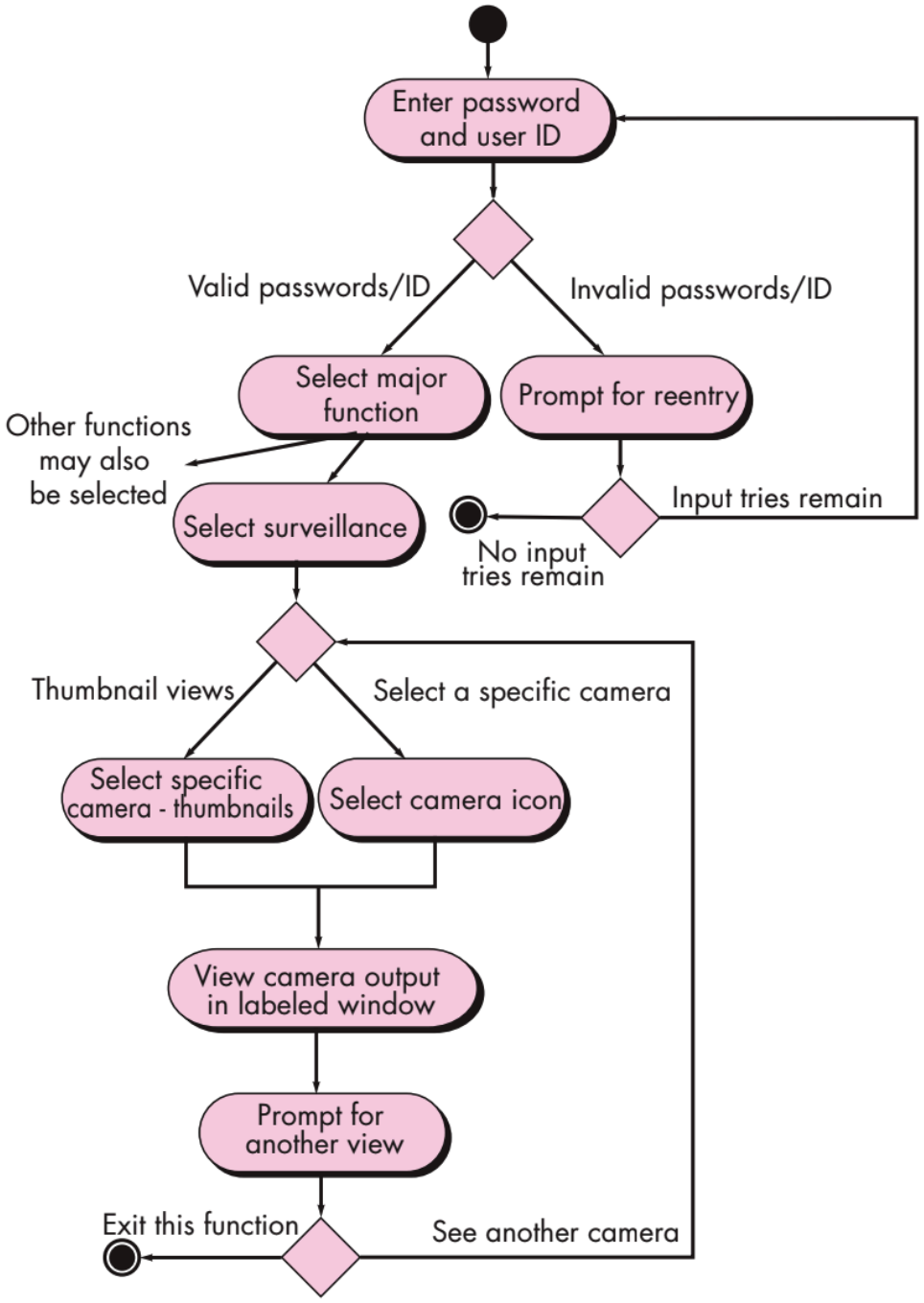
Communication: Includes activities like requirements gathering.

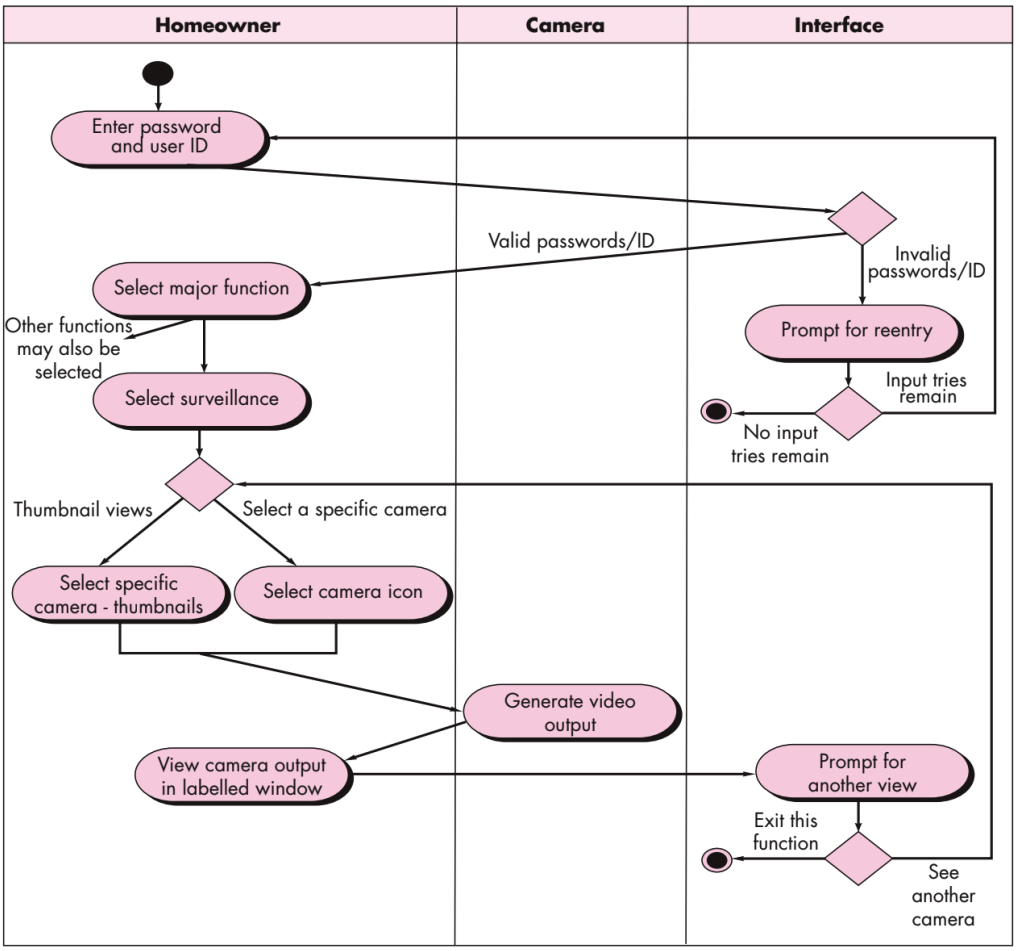
	Planning: Includes estimation, scheduling, and risk analysis. Each pass through this region results in adjustments to the project plan, and cost and schedule are adjusted based on customer feedback after delivery. Modeling: Includes analysis and design. Construction: Includes code and test. Deployment: Includes delivery and feedback. Arrows pointing inward along the axis separating the deployment region from the communication region indicate a potential for local iteration along the same spiral path. **Advantages (2M)** 1. Realistic for Large Systems: It is considered a realistic approach for the development of large-scale systems and software. 2. Explicit Risk Management: It demands a direct consideration of technical risks at all stages of the project. Because the software evolves incrementally, the developer and customer are able to better understand and react to risks at each evolutionary level. 3. Incorporates Prototyping: It utilizes prototyping as a risk reduction mechanism and allows the prototyping approach to be applied at any stage in the evolution of the product. 4. Iterative and Systematic: It maintains the systematic stepwise approach suggested by the classic life cycle but integrates it into an iterative framework that more realistically reflects the real world. **Disadvantages (2M)** 1. Controllability Concerns: It may be difficult to convince customers (especially in contract situations) that the evolutionary approach is controllable. 2. High Demand for Expertise: It demands considerable risk assessment expertise and relies heavily on this expertise for success. If a major risk is not uncovered and managed, problems will undoubtedly occur. 3. Conflict with Fixed Budgeting: If management demands fixed-budget development (which is generally noted as a bad idea), the model can be problematic because the project cost is revisited and revised after each circuit is completed.			
2 a)	**Explain principles of agile process development.** **Principles (6M - 0.5M each)** 1. The primary goal is to satisfy the customer through early and continuous delivery of valuable software. 2. Agile processes welcome changing requirements, even late in development, utilizing change for the customer's competitive advantage. 3. Working software is the primary measure of progress. 4. Working software must be delivered frequently, from a couple of weeks to a couple of months, with a preference for the shorter timescale. 5. Projects must be built around motivated individuals, who should be provided the necessary environment and support, and trusted to get the job done. 6. Business people and developers must work together daily throughout the project. 7. The most efficient and effective method of conveying information is face-to-face conversation.	6M	CO1	L1

	8. The best architectures, requirements, and designs emerge from self–organizing teams. 9. Agile processes promote sustainable development, meaning sponsors, developers, and users should be able to maintain a constant pace indefinitely. 10. Continuous attention to technical excellence and good design enhances agility. 11. Simplicity - the art of maximizing the amount of work not done—is essential. 12. The team reflects at regular intervals on how to become more effective, then tunes and adjusts its behaviour accordingly.			
2 b)	**Explain different types of Software Application Domain.** 1. System Software: This category consists of a collection of programs written specifically to service other programs. This software area is characterized by heavy interaction with computer hardware, usage by multiple users, concurrent operation (requiring scheduling and resource sharing), complex data structures, and multiple external interfaces. Examples include compilers, editors, file management utilities, operating system components, and drivers. 2. Application Software: These are stand-alone programs that solve a specific business need. Applications process either business or technical data in a manner that facilitates business operations or aids in management/technical decision making. This also includes applications used to control business functions in real-time, such as point-of-sale transaction processing. 3. Engineering/Scientific Software: Historically characterized by "number crunching" algorithms. Applications range widely, covering fields from astronomy to automated manufacturing, and from automotive stress analysis to space shuttle orbital dynamics. Modern applications in this area are moving beyond conventional numerical algorithms (e.g., computer-aided design and system simulation). 4. Embedded Software: This software resides within a product or system and is used to perform a limited and specialized set of functions (e.g., controlling the functions of a digital keypad, or embedded within a dashboard instrument system). 5. Product-line Software: This software is designed to provide a specific capability for use by many different customers. It may target niche esoteric marketplaces (like inventory control products) or large mass consumer markets (such as spreadsheets, word processing, database management, and entertainment software). 6. Web Applications (WebApps): This is a network-centric software category that spans a wide range of applications. They have evolved from simple forms to sophisticated computing tools that provide stand-alone function and integrate with corporate databases and business applications. 7. Artificial Intelligence (AI) Software: This type of software makes use of non-numerical algorithms to address complex problems that are not susceptible to straightforward analysis. Applications include expert systems, robotics, pattern recognition (image and voice), theorem proving, artificial neural networks, and game playing.	4M	CO1	L2
3	Explain CRC Modelling and Data Modelling with an example. **CRC (Class-Responsibility-Collaborator) Modelling (5M)**	10M	CO1	L2

	Class-Responsibility-Collaborator (CRC) modeling provides a simple means for identifying and organizing the classes that are relevant to system or product requirements. It is an effective mechanism for thinking about the software in an object-oriented context. A CRC model is typically a collection of standard index cards (or virtual cards) divided into three key sections: 1. Class: Written along the top of the card, this is the name of the object-oriented class being defined. 2. Responsibility (R): Listed on the left side, responsibilities are the attributes and operations relevant for the class. Simply put, a responsibility is "anything the class knows or does". 3. Collaborator (C): Listed on the right side, collaborators are those classes that are required to provide the current class with the information needed to complete a responsibility. A collaboration implies either a request for information or a request for some action. The CRC model is integral to class-based modeling. It helps ensure stakeholder commitment to feasible solutions by facilitating effective architectural design. Teams often role-play through use-case scenarios using these index cards to review the design and determine if classes, responsibilities, and collaborations are complete and consistent. For a hypothetical security function, a CRC index card for the FloorPlan class might include responsibilities such as Manages floor plan positioning and collaborators such as Wall and Camera, as the FloorPlan needs information from those objects to fulfill its responsibilities. **Data Modelling Concepts (5M)** Definition and Purpose: Data modeling is performed when software requirements include the need to create, extend, or interface with a database or when complex data structures must be constructed and manipulated. It focuses on defining the information space. The output, often a pictorial representation like the Entity-Relationship Diagram (ERD), addresses all data objects processed by the application, their relationships, and pertinent attributes. Key Elements: 1. Data Objects: A representation of composite information that must be understood by the software. A data object encapsulates data only; it contains no reference to operations that act on the data (distinguishing it from an object-oriented class). Examples include a person, a place (e.g., a warehouse), or a structure. 2. Data Attributes: Attributes name a data object, describe its characteristics, and sometimes reference another object. One or more attributes must be designated as an identifier. 3. Relationships: These indicate the manner in which data objects are connected to one another. Relationships are defined by establishing object/relationship pairs. The properties of these relationships define their cardinality (the number of occurrences of one object related to another) and modality (whether the relationship is mandatory or optional). Example: Consider two data objects, person and car. Relevant relationships between them might be defined by the pairs: • A person owns a car.			

	• A person is insured to drive a car. These relationships would be illustrated graphically, often indicating multiplicity (cardinality) to show if one person can own zero, one, or many cars.			
4	Apply the elements of requirement modeling to design a railway reservation system. Requirement modeling defines what the system should do by analyzing and specifying user needs. The main elements of requirement modeling are: **Scenario-based elements (2M)** These describe how users interact with the system. Example: Use Case 1 – Book Ticket: Passenger selects source, destination, date, train, class, and confirms booking. Use Case 2 – Cancel Ticket: Passenger enters PNR number and requests cancellation; system updates seat availability and refunds fare. Actors: Passenger, Admin, System. **Class-based elements (2M)** Identify key objects and their relationships. Example classes: Passenger, Train, Ticket, Payment, Reservation. Relationships: A Passenger books one or more Tickets. A Train has multiple Reservations. **Behavioral elements (2M)** Show how the system behaves in response to events. Example: When the Book Ticket button is clicked → System checks seat availability → Generates ticket → Updates train record. Represented using state diagrams or activity diagrams. **Flow-oriented elements (2M)** Define how data moves through the system. Example Data Flow Diagram (DFD): Level 0: Passenger → Railway Reservation System → Train Database → Ticket Output. Level 1: Includes processes like Check Availability, Book Ticket, Generate PNR, Cancel Ticket, Refund Payment. **Functional elements (2M)** Specify what functions the system performs. Example Functions: Login/Registration Search Trains Check Seat Availability	10M	CO2	L3

	Book/Cancel Ticket Generate Reports (for admin)			
5 a)	Draw the Activity diagram and Swimlane diagram for Safe home System? **Activity Diagram (3M)**  **Swimlane Diagram (2M)**	5M	CO2	L3



5 b)	How would you apply Industrial Extreme Programming in a large enterprise project, and how would this differ from applying traditional Extreme Programming? **Application of IXP in a Large Enterprise Project (3M)** Scaling Agile Practices: Implement XP core practices (pair programming, test-driven development, continuous integration) across multiple teams using coordination tools and shared repositories. IXP adds project management and business alignment practices to handle large project scope. Enhanced Communication & Planning: Use structured communication channels and release planning meetings involving multiple stakeholders, departments, and customers. Process Integration: Integrate with enterprise systems like DevOps pipelines, CI/CD, and enterprise architecture frameworks to ensure scalability and maintainability. Governance and Risk Management: IXP introduces practices like Project Community and Readiness Assessment to manage risks and organizational readiness. **Differences from Traditional XP (2M)**	5M	CO3	L3

	Aspect	**Traditional XP**	**Industrial XP (IXP)**		
	Project Size	Suited for small teams (5–10 members)	Designed for large, distributed te		
	Formality	Informal and lightweight	Adds formal management reporting processes		
	Focus	Code quality and developer collaboration	Enterprise alignment, scalabilit risk control		
	Documentatio n	Minimal	Moderate – to support ent governance		
6	Explain: i. Dynamic System development Method ii. Feature Driven Development **Dynamic System Development Method (DSDM) (5M)** DSDM is an agile project delivery framework that emphasizes rapid development, user involvement, and incremental delivery of systems that meet business needs. **Key Principles (2M)** Focus on business need. Deliver on time. Collaborate effectively. Never compromise quality. Build incrementally from firm foundations. Develop iteratively. Communicate continuously and clearly. Demonstrate control. **Phases of DSDM Life Cycle (2M)** Pre-project: Feasibility and business case preparation. Feasibility Study: Determines whether DSDM is suitable. Business Study: Understands the business and sets priorities. Functional Model Iteration: Develops prototypes and gathers feedback. Design and Build Iteration: Refines and builds working components. Implementation: Deploys system into the business environment. Post-project: Evaluates performance and benefits. **Advantages (1M)** Delivers business value quickly. Encourages active user participation. Provides strong project control and predictability. **Feature Driven Development (FDD)** FDD is a model-driven, short-iteration process focused on designing and building features that deliver tangible results in each iteration.			10M	CO3	L2

	Main Process Steps (2M) Develop an overall model. Build a features list. Plan by feature. Design by feature. Build by feature. **Key Roles in FDD (1M)** Project Manager Chief Architect Chief Programmer Class Owner Domain Expert **Characteristics & Advantages (2M)** Emphasizes object-oriented modeling and small feature sets. Encourages frequent builds and early visible progress. Provides clear progress tracking through features completed. Well-suited for large teams and long-term projects.			

Faculty Signature CCI Signature HOD Signature