Solution with scheme-Model Answer

Prof.Lynsha Helena Pratheeba HP and Prof.Rajeshwari R										
Internal Assessment Test 2 – December 2025										
Sub	**Software Engineering & Project Management**					Sub code	**BCS501**	Branch	**CSE**	
Date	**29.12.2025**	Duration	**90 mins**	Max Marks	**50**	Sem /Sec	**V / A, B, C**		**OBE**	

	Answer any FIVE FULL Questions	MARKS	CO	RBT
1. a)	How do the key design model principles provide guidance to the activities performed in the software framework?	[10]	CO3	L4
2. a)	Explain Safe Home Security functions for Flow Oriented Modeling and Behavioral Modeling.	[10]	CO3	L2
3. a)	With example explain i. different categories of Software Projects. ii. Smart Objectives.	[10]	CO3	L2
4. a) b)	Elucidate the concepts in activity planning in software project management. Differentiate between Software Projects V/S Other types of projects.	[6] [4]	CO4 CO4	L2 L2
5. a) b)	Explain the following: 1) McCall Model 2) Gravin's Quality Dimension Explain Dromey's Quality model.	[7] [3]	CO3 CO3	L2 L2
6. a)	Explain Quality Management Systems with Principles of BSENISO 9001:2000	[10]	CO4	L2

1. a) **How do the key design model principles provide guidance to the activities performed in the software framework?** [10]

Introduction [each 2 M]

Key design model principles guide software framework activities by embedding quality attributes and measurement into every planning and execution step, using structured steps (like Step Wise) to link objectives, infrastructure, risks, resources, and reviews to explicit quality targets and metrics. deliver high-quality software by aligning activities with explicit quality characteristics and measurable targets throughout the framework. State basis: models like ISO 9126 and the Step Wise planning framework provide the structure for specifying, measuring, and assuring quality in all phases.

Key design model principles [4M]

- Specify quality characteristics: identify and prioritize characteristics (functionality, reliability, usability, efficiency, maintainability, portability) based on stakeholder needs and context.
- Define metrics and measurements: select objective measures (e.g., MTBF for reliability, response time for efficiency) to operationalize quality goals.
- Plan quality assurance activities: integrate testing, verification, and validation aligned to the selected metrics and targets.
- Step Wise quality embedding: incorporate quality at scope/objectives, infrastructure standards, special characteristics analysis (e.g., safety-critical), and maintain quality focus through activity identification, effort estimation, risk planning, resource allocation, and plan review

Guidance across framework activities: [3M + 1M]

- Estimation and scheduling: use decomposition (LOC/FP/object points), effort and duration estimates, and timelines that include quality tasks (reviews, tests, refactoring buffers
- Resource and risk planning: allocate skilled human resources, reusable components, and tools appropriate to quality targets; identify and mitigate risks impacting quality early.
- Continuous quality tracking: maintain a quality plan, review overall quality aspects in plan reviews, and assess intermediate artifacts against defined metrics, not just final outputs

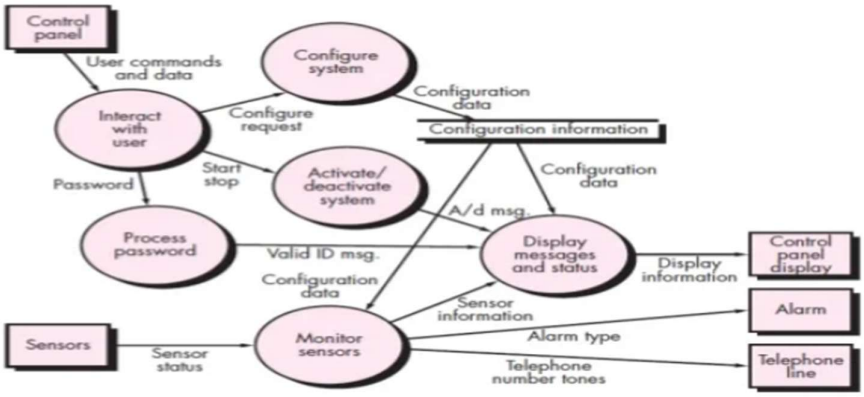
By tying explicit quality attributes and measurements to each Step Wise activity, teams ensure every action in the software framework advances verified quality outcomes and stakeholder satisfaction. This structure maps quality models and metrics to concrete project activities so that planning, execution, and reviews consistently drive toward measurable quality goals. **[1M]**

2. a) **Explain Safe Home Security functions for Flow Oriented Modeling and Behavioral Modeling.** [10]

Flow Oriented Modeling (Data Flow Diagram - DFD) [explanation 8M+ diagram 2M]

Flow Oriented Modeling focuses on how data moves through the system and how it transforms. It shows processes, data stores, data flow, and external entities.

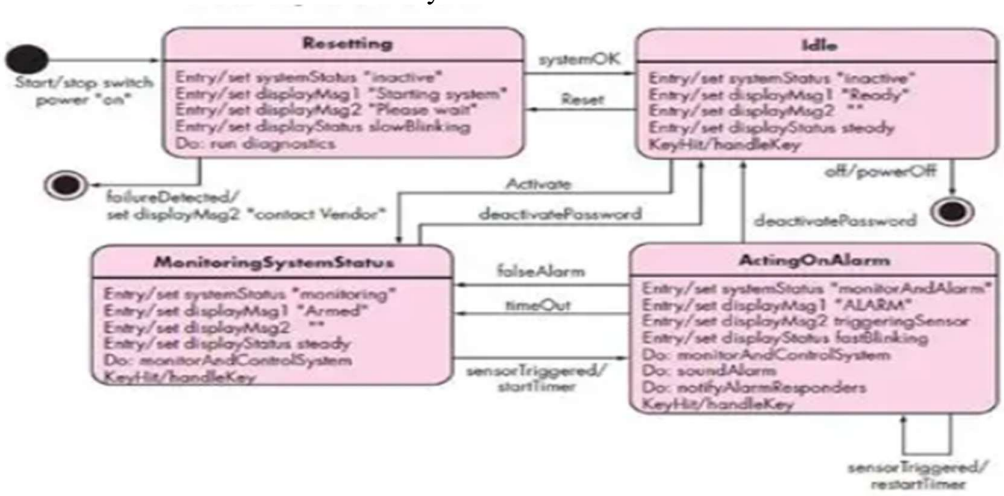
- The SafeHome system collects data from sensors (e.g., motion detectors, door sensors).
- Sensor data flows into the central control unit (processing).
- Control unit processes events such as intrusion detection.
- When an alarm condition occurs, the system sends alerts to the user and logs events in data storage.
- Users can send commands (e.g., arm/disarm) to the system



Behavioral Modeling (State Transition Diagram)

Behavioral Modeling describes how the system reacts to events over time, showing states and transitions between them based on triggers.

- The SafeHome system can be in different states such as Disarmed, Armed, Alarm Active.
- Events like "Arm System," "Cancel Alarm," "Detect Motion" cause transitions.
- For example, motion detected while armed switches system to Alarm Active.
- User code entered can transition system back to Disarmed state.



3. a) **With example explain [5M+5M]** [10M]

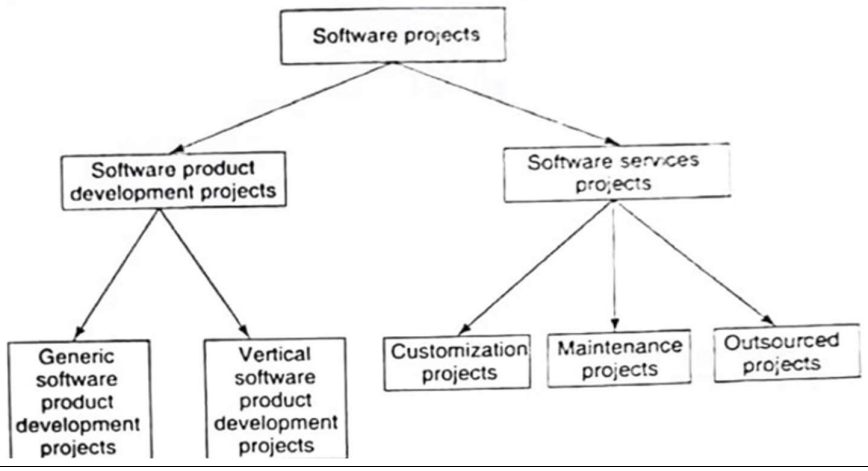
i. different categories of Software Projects.

ii. Smart Objectives

1) Software projects can be broadly categorized into two main types:

1. **Software Product Development Projects:** Projects focused on developing software intended to be sold off-the-shelf to a large number of customers.
    - **Generic Software Product Development Projects:** Products intended for the general consumer market (e.g., Microsoft's Windows Operating System, Oracle's Database Management Software).
    - **Vertical Software Product Development Projects:** Domain-specific software targeting specific segments of customers (e.g., BANCS from TCS, FINACLE from Infosys).
2. **Software Services Projects:** Projects covering a wide range of services like customization, maintenance, testing, outsourcing, and consultancy.
    - **Customization Projects:** Modifying existing software to meet specific client needs.
    - **Maintenance Projects:** Ongoing support and defect correction for existing software.

    Outsourced Projects: Developing parts of a project or the entire project by contracting it out to external companies, often utilized when the contracting company lacks sufficient expertise or determines parts can be developed cost-effectively.



- **Voluntary Systems vs. Compulsory Systems:** Systems where users choose to use them (like computer games) versus systems that staff are required to use (like an order processing system in an organization).
- **Information Systems vs. Embedded Systems:** Information systems enable staff to carry out office processes, while embedded systems control machinery (like stock control systems or air conditioning equipment).
- **Object-Driven Development:** Projects distinguished by whether their aim is to produce a product or meet a certain objective

II) ***SMART Objectives:***

Acronym	Definition	Explanation and Example
S - Specific	Concrete and well-defined.	Objectives must clearly state what needs to be achieved, avoiding vague terms like 'improve customer relations'.
M - Measurable	Must include measures of effectiveness.	Allows for the objective progress to be tracked successfully. **Example:** Instead of "reduce customer complaints" (vague), define it as **"The project will be a success if customer complaints reduce by 40%."**.
A - Achievable	Must be within the power of the individual or group to accomplish.	Ensures that the goals are realistic given the team's capabilities and resources.
R - Relevant	The objective must be relevant to the true purpose of the project.	Ensures the objectives align with the overall project and business goals.
T - Time constrained	There must be a defined point in time for achievement.	Objectives must be scheduled to prevent them from dragging on indefinitely. **Example:** "Can we install the new software by 1 November 2011?".

4. **Elucidate the concepts in activity planning in software project management.** [10]

Explanation [5 M] + Fig [1M]

Project planning is a core management activity involving deciding what is to be done. The goal of planning is to establish a pragmatic strategy for controlling, tracking, and monitoring a complex technical project to ensure the end result is done on time and with quality. Initial project planning involves estimating several characteristics of a project, based on which all subsequent project activities are planned.

During the project planning phase, the project manager performs well-defined activities and creates crucial documents, including:

1. Estimation

This involves determining quantitative attributes of the project:

- **Cost:** How much it is going to cost to complete the project?
- **Duration:** How long is it going to take to complete the project?
- **Effort:** How much effort would be necessary for completing the project (e.g., person-months)?

Estimation techniques often involve **Decomposition**, where the complex problem is broken down into smaller, manageable pieces (functions or processes), and then estimation variables like Lines of

Code (LOC) or Function Points (FP) are applied.

2. Scheduling

Based on the estimations of effort and duration, schedules for manpower and other resources are developed. The schedule needs to establish a meaningful task set, define a task network, and use tools to develop a timeline chart and schedule tracking mechanisms.

3. Defining Required Resources

This task requires determining the human, reusable software, and environmental resources needed to complete the project.

- **Human Resources:** Planners evaluate project scope and select the skills required, including organizational position (e.g., manager, engineer) and specialty (e.g., telecommunications).
- **Reusable Software Resources:** This includes Commercial Off-The-Shelf (COTS) components, full-experience components (existing components similar to the required software), partial-experience components (components requiring substantial modification), and new components.
- **Environmental Resources:** The Software Engineering Environment (SEE) incorporates hardware and software tools to support the project

4. Risk Management

Activity planning includes developing a **Risk Plan** that lists the identification of potential risks, their prioritization, and a plan for actions to contain them. Effective risk management helps proactively mitigate unforeseen issues and reduce uncertainty

5. Miscellaneous Plan

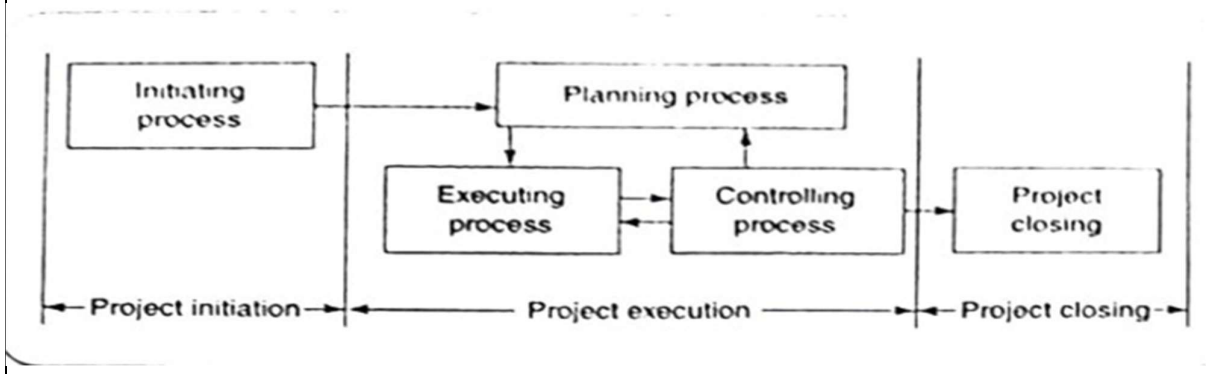
The planning phase also generates several other important documents and plans:

- **Project Plans:** Identifies project tasks, resources, and time frames.
- **Resource Plan:** Lists the manpower and equipment required.
- **Financial Plan:** Documents costs for manpower, equipment, and other items.

Quality Plan: Details quality targets and control plans.

[4]

[10]



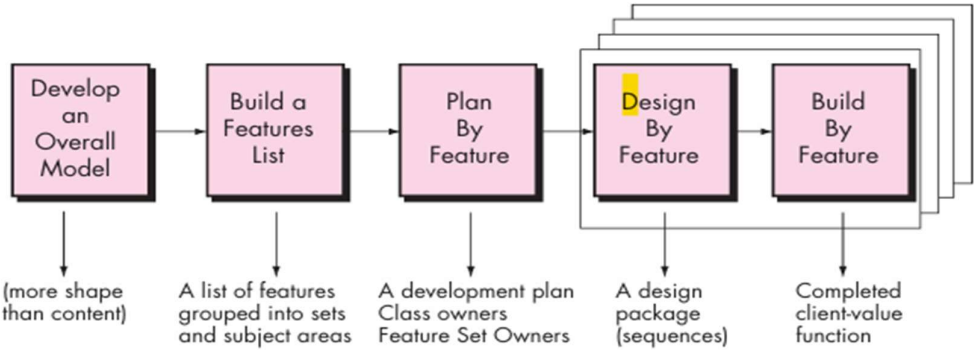
b) **Differentiate between Software Projects V/S Other types of the project.**

Characteristic	Software Projects (S/W)	Other Types of Projects (e.g., Civil)
Invisibility	**Software is not visible**; progress is difficult to track because it is intangible. You cannot physically see the progress being made.	Progress in physical artifacts (like a bridge or road) is visible and can be physically tracked.

[4]

Complexity	Software products often contain **more inherent complexity** than other engineered artifacts, often due to intricate systems and interdependencies.	Complexity generally relates to physical laws and materials (cement, steel).
Conformity	Developers must conform to the requirements of human clients, which may be **inconsistent** or constantly changing. Conformity relates to syntax and semantics.	Physical systems must conform to consistent physical laws and engineering constraints.
Flexibility	Software can be easily changed or modified (redevelop functionalities), which is generally a strength but leads to a **high degree of change and volatility** when integrating with physical systems.	Changes are expensive, difficult, and time-consuming once physical components are built.

FIGURE 3.5
Feature Driven Development [Coa99] (with permission)



5. a) Explain the following: [7]

i. McCall Model

The McCall Model is a hierarchical software quality model that defines the quality of a software product in terms of three broad parameters, based on eleven quality attributes.

The three broad parameters are:

1. **Product Operations:** Reflects the software's operational characteristics (how well it works).
2. **Product Revision:** Reflects how easy it is to fix defects and make changes.
3. **Product Transition:** Reflects how easy it is to port the software to different platforms.

These three parameters are decomposed into 11 quality attributes:

Parameter	Quality Attributes	Definition
Product Operations	Correctness	Extent to which a program satisfies its specification and fulfills the user objective.
	Reliability	Probability of the product working satisfactorily over a given duration.
	Efficiency	Amount of computing resources required to perform required functions.

	Integrity	Extent to which data remains valid, and access is controlled.
	Usability	Effort required to learn, operate, prepare input, and interpret output.
Product Revision	Maintainability	Ease with which it is possible to locate and fix bugs in the software.
	Flexibility	Effort required to adapt the software product to changing requirements.
	Testability	Effort required to test the product to ensure it performs its intended function.
Product Transition	Portability	Effort required to transfer the software product from one hardware or software system environment to another.
	Reusability	Extent to which a software can be reused in other applications.

ii. Gravin's Quality Dimensions

David Gravin, a professor at Harvard Business school, defined the quality of any product in terms of eight general attributes. These dimensions provide a framework for evaluating product quality from various perspectives:

1. **Performance:** How effectively the product performs its intended jobs or primary operating characteristics.
2. **Features:** How well the product supports the required features or secondary characteristics that supplement the basic functioning.
3. **Reliability:** The probability of the product working satisfactorily without failure within a specified period of time.
4. **Conformance:** The degree to which the product meets pre-established requirements or industry standards.
5. **Durability:** A measure of the product's expected useful life or longevity.
6. **Serviceability:** The speed and effectiveness with which maintenance, repair, or defect resolution can be accomplished.
7. **Aesthetics:** The subjective characteristics relating to the look, feel, taste, or sound of the product.
   Perceived quality: The user's opinion or reputation concerning the product quality, often inferred from external attributes

b) **Explain Dromey's Quality model.** [3]

Dromey's model, like McCall's, is a hierarchical quality model. Dromey proposed that software product quality depends on four major high-level properties of the software:

1. **Correctness:** This relates to how accurately the software fulfills its requirements.
2. **Internal Characteristics:** These properties depend on the internal structure of the software itself (e.g., coding practices, modules).
3. **Contextual Characteristics:** These properties relate to how the software interacts with its environment or external elements.
4. **Descriptive Properties:** These properties relate to the documentation and how the software is described.

These high-level properties, in turn, depend on several lower-level quality attributes. Dromey's model illustrates how different high-level properties map to various software attributes, some of which overlap:

High-Level Property	Dependent Software Attributes
Correctness	Functionality, Reliability
Internal	Maintainability, Efficiency, Reliability
Contextual	Maintainability, Reusability, Portability, Reliability
Descriptive	Maintainability, Reusability, Portability, Usability

This model emphasizes that software quality is not monolithic but rather a consequence of ensuring specific properties (like correctness or good internal structure) which are measured through observable attributes.

6. Explain Quality Management Systems with Principles of BSENISO 9001:2000. [10]

The sources provide detailed information on software quality standards, specifically focusing on **ISO 9126**, but they **do not contain information regarding the Quality Management Systems (QMS) or the specific Principles of BSENISO 9001:2000**.

However, given the context of software quality, the most relevant standard discussed is **ISO 9126** (introduced in 1991), which provides a significant standard for defining software quality attributes and offering a framework for assessing them.

ISO 9126: Key Quality Characteristics

ISO 9126 identifies six major external software quality characteristics:

6. **Functionality:** The functions provided by the software to satisfy user needs. Sub-characteristics include suitability, accuracy, interoperability, security, and compliance.
7. **Reliability:** The capability of the software to maintain its level of performance under stated conditions. Sub-characteristics include maturity (frequency of failures), fault tolerance, and recoverability.
8. **Usability:** The effort needed to use the software. Sub-characteristics include understandability, learnability, operability, and attractiveness.
9. **Efficiency:** The ability to use resources in relation to the amount of work done. Sub-characteristics include time behavior and resource utilization.
10. **Maintainability:** The effort needed to make changes to the software. Sub-characteristics include analyzability (ease of identifying causes of failure), changeability (ease of modification), stability (low risk of unexpected effects from modification), and testability.
11. **Portability:** The ability of the software to be transferred from one environment to another. Sub-characteristics include adaptability, installability, and co-existence.

ISO 9126 as a QMS Guideline

ISO 9126 provides guidance on how to use these quality characteristics in a systematic manner for software quality management:

1. **Prioritization:** Assign importance ratings to quality characteristics based on the application's relevance (e.g., reliability is critical for safety-critical systems).
2. **Measurement:** Select relevant external quality measurements (e.g., MTBF for Reliability, response time for Efficiency).
3. **Assessment:** Use measurements to map results onto user satisfaction ratings and calculate weighted scores for all characteristics to determine an overall product quality score.

While this is not BSENISO 9001:2000, ISO 9126 provides the structured quality framework utilized within software project management contexts presented in the sources.