CBCS SCHEME

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

BCS501

Fifth Semester B.E./B.Tech. Degree Examination, Dec.2024/Jan.2025
Software Engineering and Project Management

Time: 3 hrs. Max. Marks: 100

Note: 1. Answer any FIVE full questions, choosing ONE full question from each module.
2. M : Marks , L: Bloom's level , C: Course outcomes.

			M	L	C
		Module – 1			
Q.1	a.	Explain software process and software engineering practices.	10	L2	CO1
	b.	Explain the waterfall model and incremental model, with diagram.	10	L2	CO1
		OR			
Q.2	a.	Explain Boehm Spiral process model with a neat diagram. Mention its advantages and disadvantages.	10	L2	CO1
	b.	Explain the five activities of a generic process framework for software engineering.	10	L2	CO1
		Module – 2			
Q.3	a.	Explain the distinct tasks of requirement engineering.	10	L2	CO2
	b.	Illustrate the UML use case diagram for safe home system.	10	L2	CO2
		OR			
Q.4	a.	Explain Class-Responsibility-Collaborator(CRC) modeling and data modeling with an example.	10	L2	CO2
	b.	Explain the elements of analysis model in requirement modeling.	10	L2	CO2
		Module – 3			
Q.5	a.	Explain the principles of agile process development.	10	L2	CO3
	b.	Explain the following : i) Adaptive software development ii) SCRUM	10	L2	CO3
		OR			
Q.6	a.	Explain the concepts of extremes programming with a neat diagram.	10	L2	CO3
	b.	Explain design modeling principles that guide the respective framework activity.	10	L2	CO3
		Module – 4			
Q.7	a.	Illustrate the project management life cycle with a neat diagram.	10	L2	CO4
	b.	Explain : i) Different ways of categorizing software projects ii) Smart objectives	10	L2	CO4
		OR			
Q.8	a.	Explain the difference between traditional versus modern project management practices along with the role of management.	10	L3	CO4
	b.	Explain software development life cycle (ISO 12207) with a neat diagram.	10	L2	CO4
		Module – 5			
Q.9	a.	Explain Quality Management System with principles of BS EN ISO-9001-2000.	10	L2	CO5
	b.	Explain the following : i) McCall model ii) Garvin's Quality Dimensions.	10	L2	CO5
		OR			
Q.10	a.	Describe six generic functions allowed in automated estimation techniques of software projects.	10	L3	CO5
	b.	Explain COCOMO II model.	10	L2	CO5

* * * * *

Fifth Semester B.E. Degree Examination
Dec 2024 | Jan 2025

Software Engineering and Project Management: BCS501
Max Marks: 100

Staff: Sunita. Kallu

Solutions

Module - 1

Q1 a. Explain Software process and software engineering principles.

Ans :- A process is a collection of activities, actions, and tasks that are performed when some work product is to be created.

A process framework establishes the foundation for complete software engineering process by identifying a small number of framework activities. A generic framework for software engineering encompasses five activities:

1) Communication :- It is important to communicate and collaborate with the customer.

2) Planning :- Software project is a complicated journey and the planning activity creates a map. The map

3) Modelling :- A Software engineer does the thing by creating models to better understand Software requirements and design that will achieve those requirements.

4) Construction :- This activity combines the code generation and the testing that is required to uncover errors in the code.

5) Deployment :- The software is delivered to the customer who evaluates the delivered product and provides feedback based on the evaluation.

The Software Engineering Practices are

1) The Essence of Practice
   1) Understand the problem (communication and analysis).
   2) Plan a Solution (modelling and software design).
   3) Carry out the plan (Code generation).
   4) Examine the result for accuracy (testing and quality assurance).

2) Understand the problem :
   1) who has a stake in the solution to the problem
   2) what are unknowns? what data, function and

functions, and features are required to solve the problem.

3) Can the problem be represented graphically? Can an analysis model be created?

3) Plan the Solution

- Have you seen similar problems before?
- Has a similar problem been solved?
- Can subproblems be defined?
- Can a design model be created?

4) Carry out the Plan

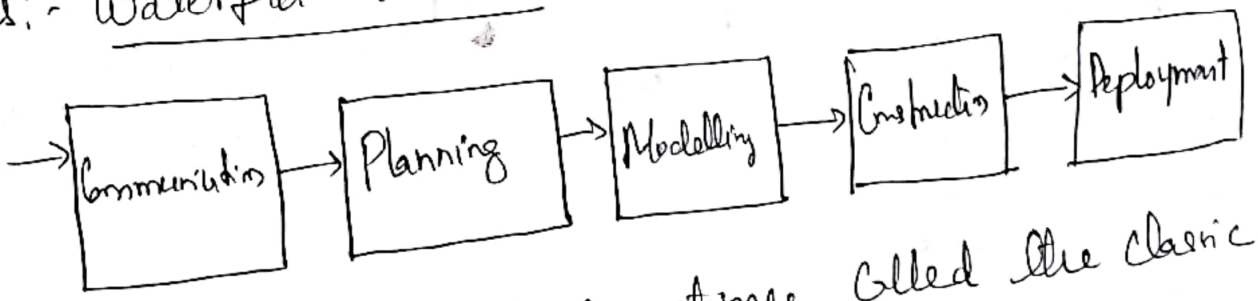
- Does the solution conform to the plan?
- Is each component part of the solution provably correct.

5) Examine the result

- Is it possible to test each component part of the solution
- Does the solution produce results that conform the data, functions and features that are required?

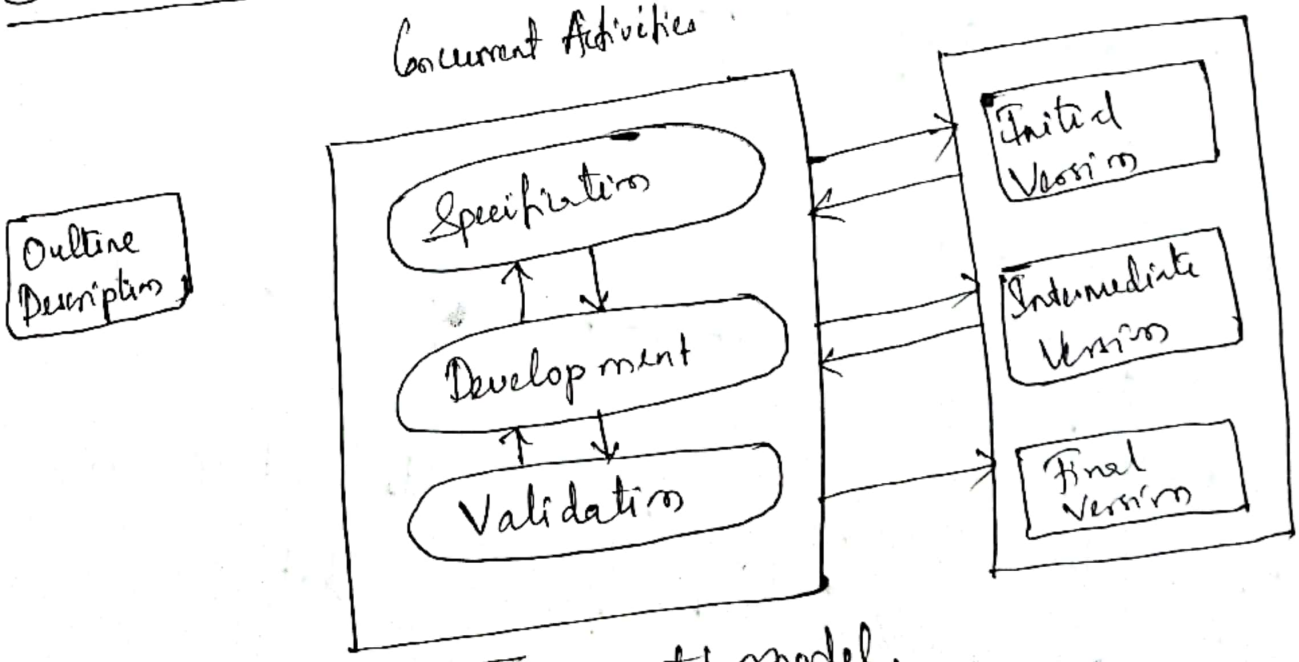
. b. Explain the waterfall model and incremental model, with diagram.

Ans:- Waterfall model



The waterfall model sometimes called the classic life cycle, suggests a systematic approach to software development that begins with customer specification of requirements and progresses through planning, modelling, construction and deployment

Incremental Model

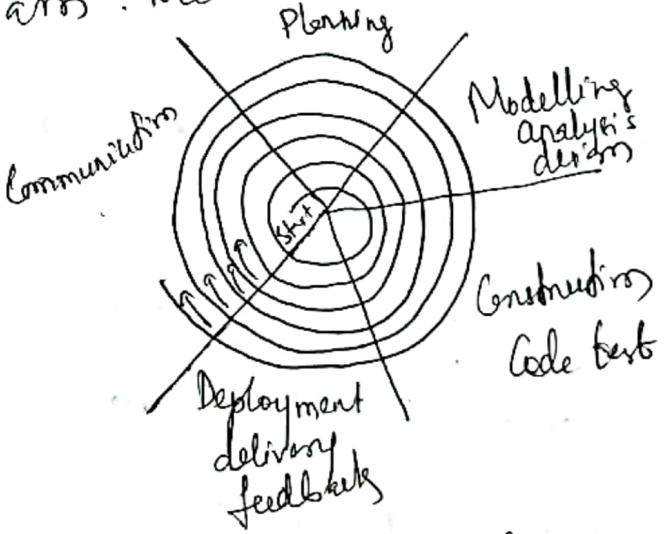


Incremental model.

The incremental model combines elements of linear and parallel process flows.
When an incremental model is used, the first increment is often a core product, the basic requirements are addressed. The core product is used by the customer. The incremental process model focuses on the delivery of an operational product with each increment. Early increments are stripped down the versions of the final product.

Q2.a. Explain Boehm Spiral process model with a neat diagram. Mention its advantages and disadvantages.

Ans:-



A Spiral model is divided into a set of framework activities defined by the software engineering team.
the spiral is an evolutionary software process model that couples the iterative nature of prototyping with the controlled and systematic aspects of the waterfall model.

Advantages and Disadvantages of Spiral Model

Advantages :-

1. Software is produced early in the software life cycle.
2. Risk handling is one of important advantages of Spiral model.
3. It is good for large and complex projects
4. It is good for customer satisfaction.
5. Strong approval and documentation control.
6. It is suitable for high risk projects.

Disadvantages :-

1. It is not suitable for small projects as it is expensive.
2. It is much more complex than other SDLC models
3. Too much dependable on risk analysis.
4. Difficulty in time management.
5. Spiral may go indefinitely.
6. End of the project may not be known early.

2b. Explain the five activities of a generic process framework for software engineering.

Ans:- The five activities of a generic process framework for Software engineering are Communication, planning, modelling, Construction and deployment.

1) Communication :- Communication is the crucial first step that involves gathering requirements from stakeholders understanding their needs, and defining the project objectives.

2) Planning :- Planning involves defining the project Scope, identifying risks, allocating resources and establishing a schedule. Effective planning ensures that the development team has a clear road map and can anticipate the potential challenges

3) Modelling :- This is activity that focuses on creating representations of the software system before it's actually built. This involves activities like requirements analysis, design and creating models that help understand the problem and solution

4) Construction :-

It is the phase where the software is actually built. This involves coding based on the design, followed by testing to ensure functionality and quality. It's the phase where the abstract design is translated into tangible working software.

5) Deployment :-

Deployment is the process of making a software application or update available for users to utilize. A generic deployment process typically involves planning, building, testing, staging, deploying, and monitoring the software. It is a crucial part of the software development life cycle, following development and testing and leading into ongoing operations.

MODULE- 2

Q3 a. Explain the distinct tasks of requirement engineering?

Ans;- 1) Inception :- How does a software project get started? Is there single event that becomes catalyst for new computer based system or product, or does the need evolve over time?

At project inception you establish a basic understanding of the problem, the people who want a solution the nature of the solution that is desired and the effectiveness of preliminary communication and collaboration between the other stakeholders and the software team.

Elicitation :- It certainly seems simple enough ask the customer, the users and others what the objectives for the system or product are what is to be accomplished how the system or product fits into the needs of the business and finally how the system or product is to be used on a day-to-day basis.

Elaboration :- The information obtained from the customer during inception and elicitation is expanded and refined during elaboration. Elaboration is driven by the creation and refinement of user scenarios that describe how the end user (and the other actors) will interact with the system. Each user scenario is parsed to extract analysis classes - business domain entities that are visible to end users

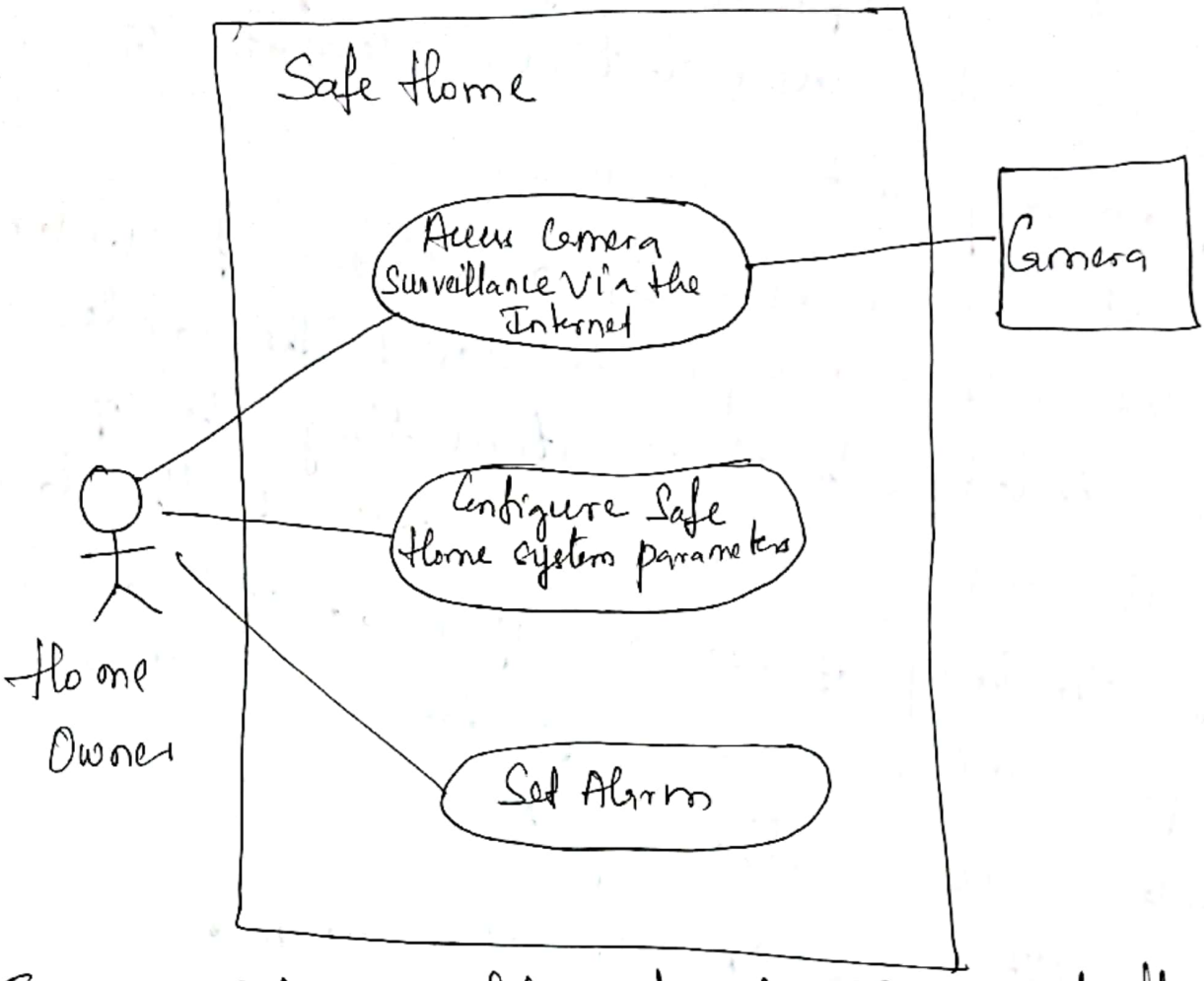
Negotiations:- Have to reconcile conflicts through a process of negotiations, customers, users, and other stakeholders are asked to rank requirements and then discuss conflicts in priority. Using an iterative approach that prioritizes requirements assesses their cost and risk, and addresses internal conflicts, requirements are eliminated combined or modified so that each party achieves some measure of satisfaction.

Specification:- Specification means different things to different people. A specification can be written document, a set of graphical models, a formal mathematical model, a collection of usage scenarios, a prototype or any combination of these.

Q 3 6. Illustrate the UML use case diagram for safe home system.

Ans:- Graphical representation of usage scenario the diagrammatic representation can facilitate understanding particularly when the scenario is

is complex.



Every modelling notation has limitation and the Use Case is no exception. A usecase focuses on functional and behavioral requirements and is generally inappropriate for non-functional requirements.

For situations in which the requirements model must have significant detail and precision

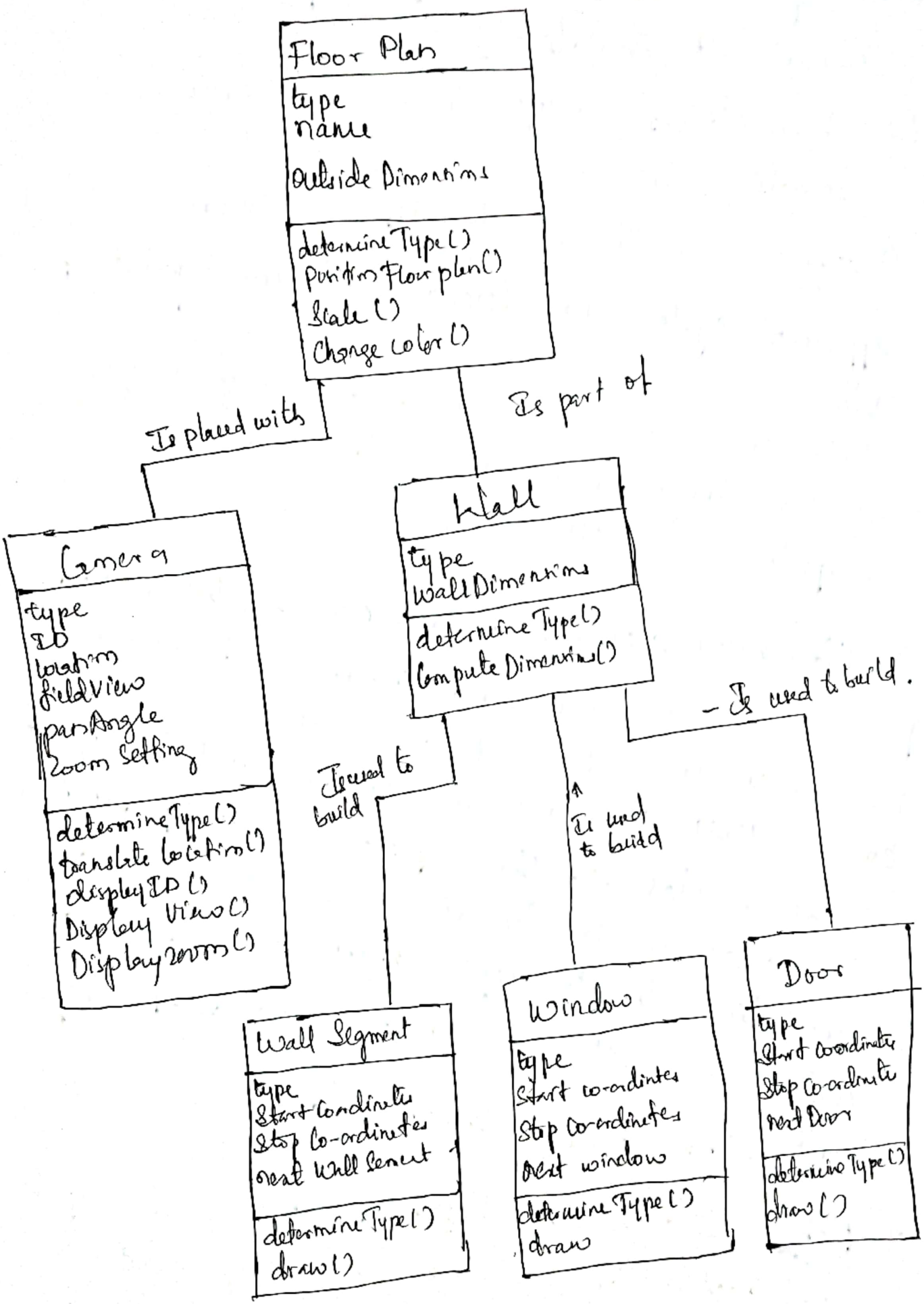
Q4 a. Explain Class-Responsibility (CRC) modelling and data modelling with an example.

Ans:- CRC is Class responsibility - Collaborator Modelling provides a simple means for identifying and organizing the classes that are relevant to system or product requirements.

A CRC model is really a collection of standard index cards that represent classes. The cards are divided into three sections. Along the top of the card you write the name of the class. In the body of the card you list the class responsibilities on the left hand and the collaborative on the right.

In reality, the CRC model may make use of actual or virtual index cards. The intent is to develop an organized representation of classes. Responsibilities are the attributes and operations that are relevant for the class. Collaborators are required to provide a class with the information needed to complete a responsibility.

Example:- A simple CRC index card for the FloorPlan class. The list of responsibilities shown on the CRC card is preliminary and subject to additions and modifications.



4.6 Explain the elements of analysis model in requirement modeling.

Ans:- Elements of analysis model in requirement modeling are:-

1) Scenario-based elements:- Using a scenario based approach, system is described from user's point of view
Ex:- basic use cases and their corresponding use-case diagrams evolve into more elaborate template based use cases.

2) Class based elements:- A collection of things that have similar attributes and common behaviours. i.e objects are categorized into classes.
Ex:- a UML case diagram can be used to depict a sensor class for the SafeHome security function. In addition to class diagrams, other analysis modelling elements depict manner in which classes collaborate with one another and relationships and interactions between classes

3) Behavioural Elements:- Effect of behaviour of computer based systems can be seen as design that is chosen and implementation approach that is

applied. Modeling elements that depict behaviour must be provided by requirements model.

4) Flow-oriented elements :-

As it flows through a computer-based system information is transformed. System accepts input applies functions to transform it and produces output in various forms. Input may be a control signal transmitted by a transducer, a series of numbers typed by human operator a packet of information transmitted on a network link, or a voluminous data file retrieved from secondary storage.

MODULE - 3

Q 5 a. Explain the principles of agile process development.

Ans:- The principles of agile process development are :-

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change

Change for the customer's competitive advantage.

3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.

4. Business people and developers must work together daily throughout the project.

5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.

6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

7. Working software is the primary measure of progress.

8. Agile processes promote sustainable development. The sponsors, developers and users should be able to maintain a constant pace indefinitely.

9. Continuous attention of technical excellence and good design enhances agility.

10. Simplicity – the art of maximizing the amount of work not done

5b. Explain the following:

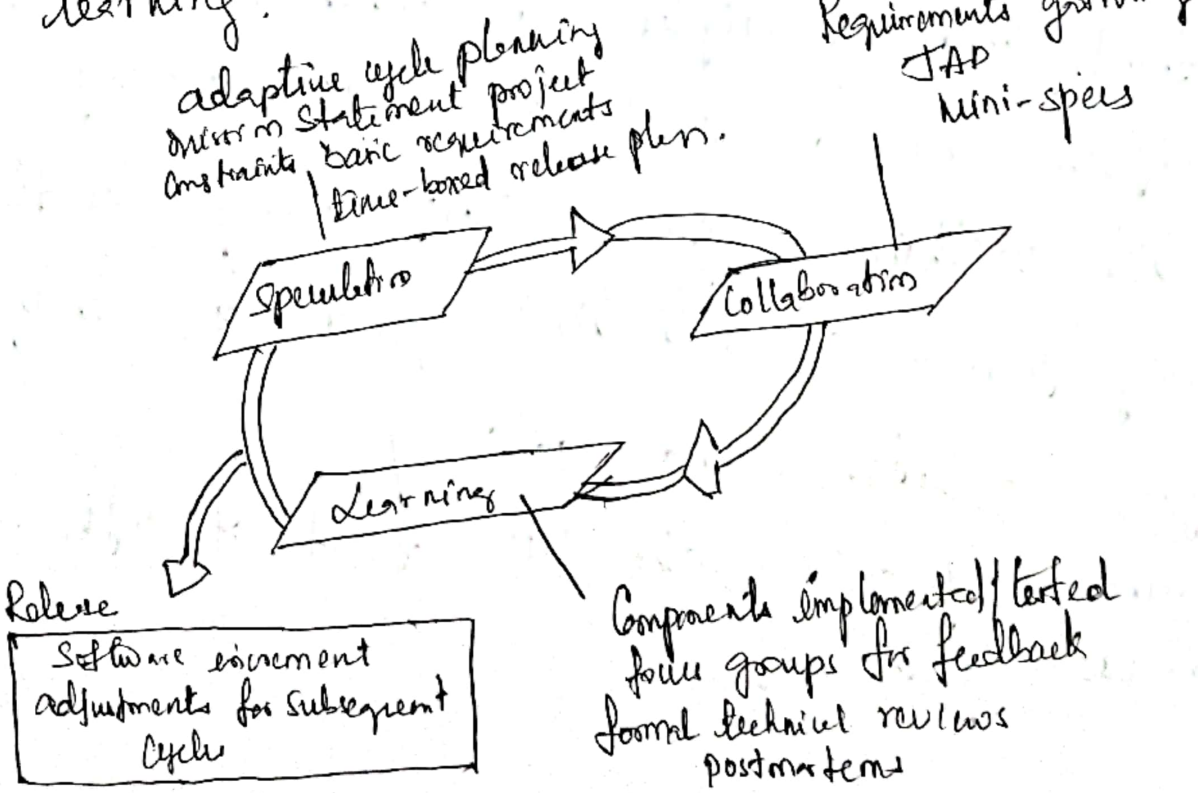
1) Adaptive Software development

2) SCRUM.

Ans:- 1) Adaptive Software development

Adaptive Software development (ASD) has been a technique for building complex software and systems.

Adaptive development approach based on collaboration is a much a source of order on our complex interactions as discipline and engineering. it define an ASD "life cycle" that incorporates three phases, speculation, collaboration, and learning.



During speculation the project is initiated and adaptive cycle planning is conducted.

Adaptive cycle planning uses project initiation information — the customer's mission statement project constraints and basic requirements — to define the set of release cycles that will be required for the project.

Collaboration is used by the motivated people in a way that multiplies their talent. This approach is a recurring theme in all agile methods. But collaboration is not easy. It encompasses communication and teamwork.

2) Scrum:- Scrum is an agile software development method. Scrum principles are consistent with agile manifesto and are used to guide development activities within a process that incorporates the following framework activities: requirements, analysis, design, evolution and delivery.

Scrum emphasizes the use of set of software process patterns.

1) Backlog:- a prioritized list of project requirements or features that provide business value for the customer. Items can be added to the backlog at any time. The product manager assesses the backlog and updates priorities as required.

2) Sprints:- Consist of work units that are required to achieve a requirement defined in the backlog that must be fit into predefined time box

Q6 a. Explain the concepts of extreme programming with a neat digram.

Ans:- Extreme Programming (XP) the most widely used approach to agile Software development.

The extreme programming uses an object-oriented approach as its preferred development paradigm and encompasses a set of rules and practices that occur within the context of four framework activities: planning, design, coding and testing.

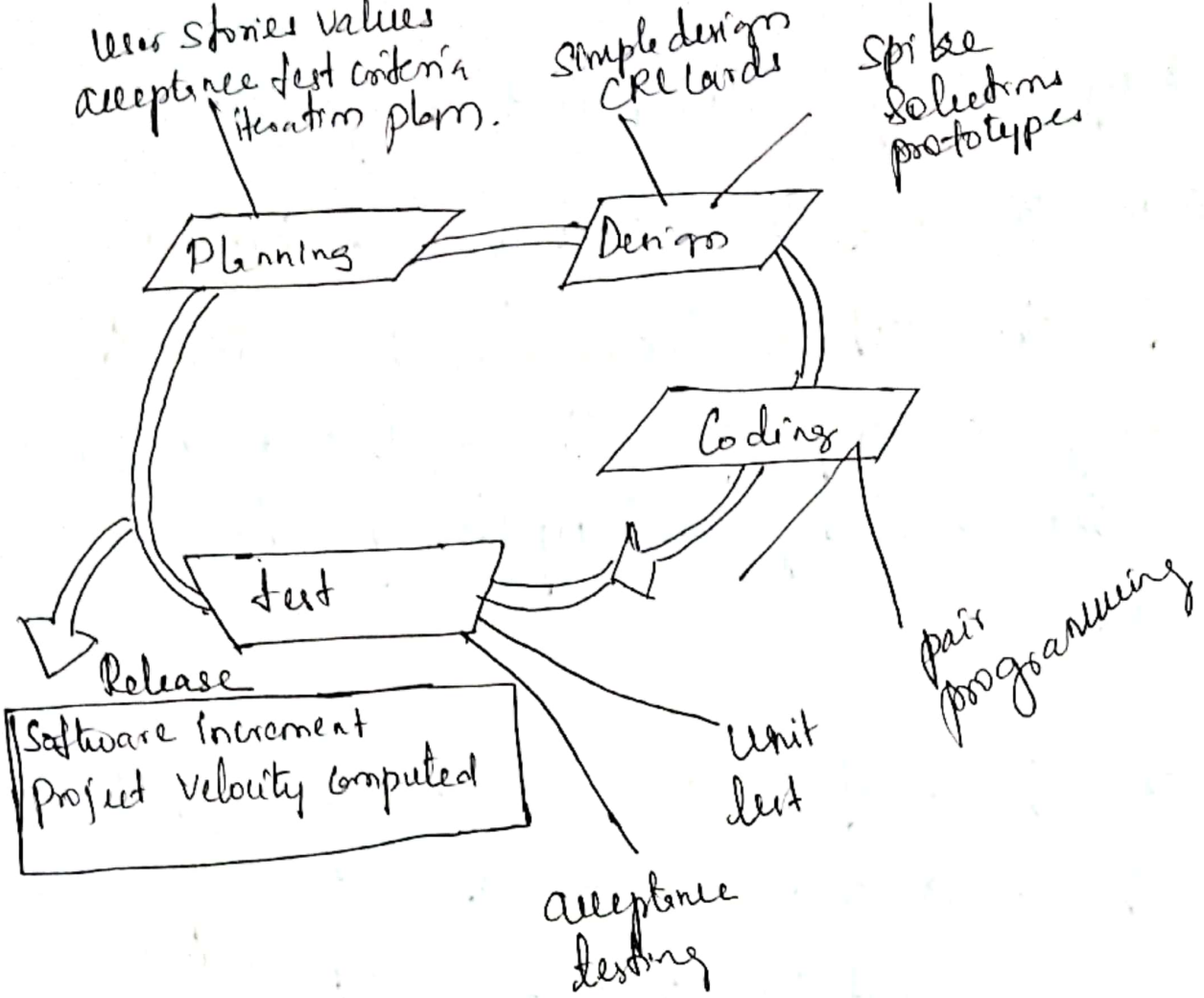


Fig:- The Extreme Programming Process.

Planning:- The planning activity begins with listening a requirements gathering activity that enables the technical members of the XP team to understand the business context.

Design:- A simple design is always preferred over a more complex representation.

Coding:- Be the preliminary design work is done the team does not move to code, but

rather develops a series of unit tests will exercise Once the test has be created, the developer better focuses on what must be implemented to pass the test. Once the code is complete, it can be unit-tested immediately.

Testing :- The unit tests that are created should be implement using a framwork. This encourages a regression testing strategy.

6. b. Explain design modelling principles that guide the respective framework activity.

Ans :- Principles that guide the respective framework activity are :-

1) Communication Principles

Before customer requirements can be analyzed, modeled, or specified they must gather through the communication activity. Effective communication with the customer and Stakeholders and with project managers is the most Challenging activities.

Principle 1 :- Listen

Principle 2 :- Prepare before you communicate.

Principle 3: Someone should facilitate the activity.

Principle 4: Face-to-face communication is best.

Principle 5: Take notes and document decisions

Principle 6: Strive for collaboration.

Principle 7: Stay focussed, modularize your discussion

Principle 8: If something is unclear, draw a picture.

Planning :-

Principle 1 :- Understand the Scope of the project

Principle 2: Involve stakeholders in the planning activity.

Principle 3: Recognize that planning is iterative.

Principle 4: Estimate based on what you know.

Principle 5: Consider risk as you define the plan.

Principle 6: Be realistic

Principle 7: Adjust granularity as you define the plan.

Principle 8: Define how you intend to ensure quality.

Principle 9: Describe how you intend to accomodate Change.

Principle 10: Track the plan frequently and make adjustments as required.

Modelling Principles :- We create models for better understanding of the actual entity to built.

Principle 1: The primary goal of the software team is to build software, not create models.

Principle 2: Travel light - don't create more models than you need.

Principle 3: Strive to produce the simplest model that will describe the problem or the software.

Principle 4: Build models in a way that makes them amenable to change.

Principle 5: Be able to state an explicit purpose for each model that is created.

Principal 6: Adapt the models you develop to the system at hand.

Construction Principles

The construction activity encompasses a set of coding and testing tasks that lead to operational software

Coding principles

Before you write one line of code be sure you:

1) Understand of the problem you're trying to solve
2) Understand basic design principles and concepts.
3) Pick a programming language that meets the needs of the software to be built and the environment in which it will operate.
4) Select a programming environment that provides tools that will make your work easier.

Testing Principles

Testing

Principle 1: All tests should be traceable to customer requirements.

Principle 2: Tests should be planned long before testing begins.

Principle 3: The Pareto principle applies to software testing.

Principle 4: Testing should begin in the small and progress toward in the large.

Principle 5: Exhaustive testing is not possible.

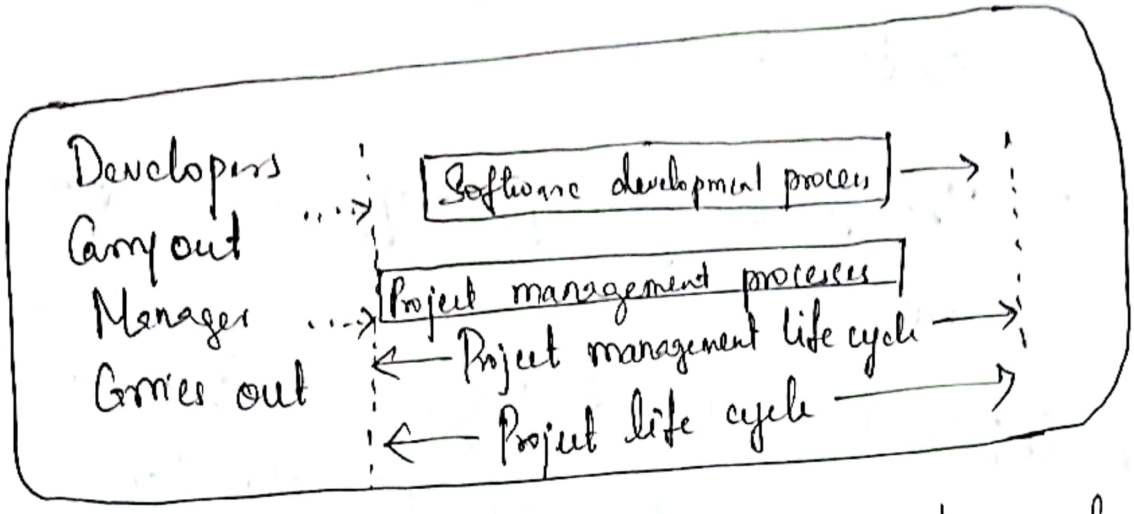
MODULE-4

Q7 a. Illustrate the project management life cycle with a neat diagram.

Ans:- Software development life cycle denotes (SDLC) the stages through which a software is developed. In contrast to SDLC, the project management life cycle typically starts well before the software

development activities start and continues for the entire duration of SDLC.

In Project Management process, the project manager carries out project initiation, planning, executing, monitoring, controlling and closing.



1) Project initiation :- The project initiation phase starts with project concept development. During concept development and the different characteristics of the software to be developed and throughly understood which includes the scope of the project, the project constraints, the cost that would be incurred and the benefits that would accrue. Based on this understanding, a feasibility study is undertaken to determine the project would be both financially and technically feasible.

Q6. Explain :-

1) Different ways of Categorizing Software projects.

2) Smart objectives.

1) Different ways of Categorizing Software projects.

1) Size:-

- Small scale - designed for a single user or small group.
- Enterprise level:- Designed for large organizations and complex needs.

2) Type

- New software development: Building a new applications
- Software updates/maintainence: enhancing or fixing existing software.
- Mobile applications: Specifically designed for mobile devices.

3) Application

- Information systems: Used for managing data and

and workflows within an organization.

- Embedded Systems :- Used to control hardware and devices

- By Project Management Methodology :-

- Agile :- Iterative and incremental approach (Eg :- Scrum)
- Waterfall :- Sequential, linear approach.
- Extreme Programming (XP) :- Emphasizes close collaboration and frequent testing.

d) Smart Objectives

Smart objectives in software projects are goals that are specific, measurable, achievable, relevant and time bound. These criteria help ensure that project goals are clear, realistic and trackable. Here are examples of SMART objectives tailored to software projects.

1) Specific

- Clearly define what needs to be done.

2) Measurable:-

- Include metrics to track progress or success

3) Achievable:-

- Ensure the goal is realistic given resources and constraints.

4) Relevant:-

- Align with broader project or business goals.

5) Time-bound

- Set a clear deadline.

Q 8.a Explain the difference between traditional versus modern project management practices along with the role of management.

Ans:- The difference between traditional versus modern project management are:-

- Planning Incremental Delivery:- Earlier projects were simpler and therfore more predictable than the present-day projects. In those days projects were planned with sufficient detail much before the actual project execution started. After the project initiation, monitoring and control activities were

Carried out to ensure that the project execution proceeded as per plan. Now the projects are required to completed over a much shorter duration and rapid application development and deployment are considered.

- Quality Management :- Customer awareness about product quality has increased significantly. The key responsibility of manager of project now includes assessment of project progress and tracking the quality of all intermediate artifacts.

- Change Management :- Earlier when the requirements were signed-off by the customer any changes to the requirements were rarely entertained. Customer suggestions are now actively solicited and incorporated throughout the development process.

- Requirement Management :- In older development methodologies, the requirements has to be identified upfront and these were signed off by the customer and frozen before the development could start. At present most projects the requirements change frequently during the development cycle.

• Release Management : Release management concern planning, prioritizing and controlling the different releases of a software. Modern development process such as Agile development process advocate frequent and regular releases of the software to be made to the customer during the software development.

• Risk Management :- In modern software development practices. Effective risk management is considered very important to the success of a project. A risk is any negative situation that may arise as the project progresses and may threaten the success of the project Risk Management

• Scope Management :- Modern software development encourages customer to come up with change requests. While accepting the requests, three critical projects parameters: Scope, schedule and project cost are interdependent and related.

Q8 6. Explain Software development life cycle (ISO 12207) with a neat diagram.

Ans:- Software development life cycle is a structured process that is used to design, develop and test high-quality software.

SDLC is a process followed for software building with software organization. SDLC consists of precise plan that describes how to develop, maintain, replace, and change specific software. The life cycle defines a method for improving the quality of software and all around development process. SDLC Specifies the tasks to be performed at various stages by the software engineer or developer. It ensures that the end product is able to meet the customer's expectations and fits within the overall budget. SDLC is a collection of these six stages and the stages ~~or~~ of SDLC are as follows:-

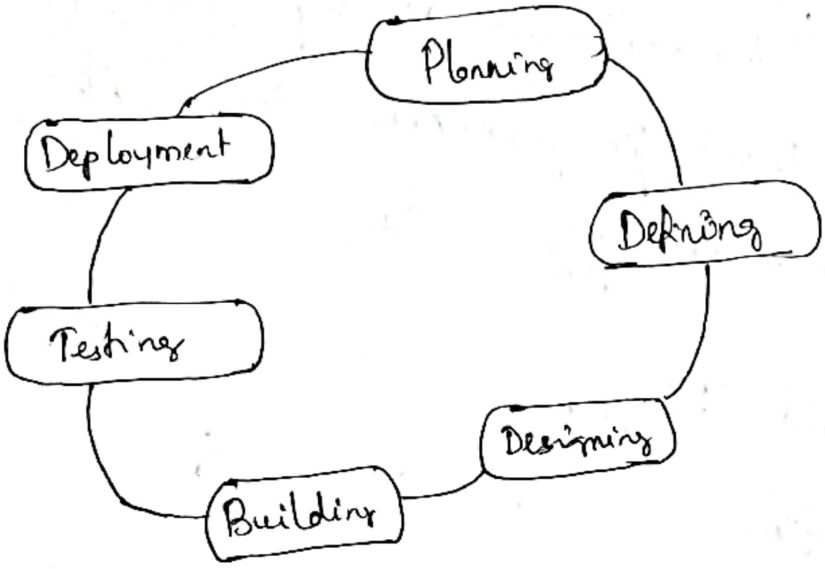


fig:- SDLC

Stage 1: Planning and Requirement Analysis
its is the crucial step in software development in this same stage; requirement analysis is performed by the developers of the organization.

Stage 2: Defining Requirements
In this stage all the requirements for the target software are specified. These requirements get approval from customer, market analyst and Stake holders.

Stage 3:- SRS is a reference of software designers to come up with best architecture for the software.

Stage 4: Developing product

At this stage the fundamental development of product starts Conventional programming tools like Compilers, interpreters, debuggers etc.

Stage 5: Testing and Integration

Testing of the software is necessary to ensure its smooth execution.

Stage 6:- After detailed testing the conclusive product is released in phases as per the organization's strategy, then it is tested in real industry environment it is important to ensure its smooth performance.

MODULE - 5

Q 9. a Explain Quality Management System with principles of BS EN ISO - 9001 - 2000.

Ans:- A Quality Management System (QMS) is a structured system of procedures and processes covering all aspects of an organization's operations to consistently meet customer and regulatory requirements Principles of BS EN ISO - 9001 - 2000 are

1) Customer Focus :-

Organizations depend on their customers and should understand their needs, meet requirements and exceed expectations.

2) Leadership and Teamwork :-

Teamwork establish purpose and direction, creating an environment where people are fully involved in achieving quality objectives.

3) Involvement of People :-

People at all levels are the essence of the organization and their full involvement enables their abilities to be used for the organization's benefit.

4) Process Approach :-

A desired result is achieved more efficiently when activities and related resources are managed as a process.

5) System Approach to Management

Identifying, understanding and managing interrelated processes as a system contributes to the

organization's effective and efficiency.

6. Continuos improvement

Continual improvement of the organization's overall performance should be a permanent objective.

Q9. 6. Explain the following

1) McCall model 2) Garvin's Quality Dimensions.

Ans:- 1) McCall model: McCall defined the quality of a software in terms of three broad parameters: its operational characteristics, how easy it is to fix defects and how easy it is to different platforms These three high-level quality attributes are defined based on the following attributes of software.

1) Correctness :- the extent to which a software product satisfies its specifications.

2) Reliability :- the probability of software product working satisfactorily over a given duration.

3) Efficiency :- the amount of computing resources required to perform the required functions.

4) Integrity :- The extent to which the data of the software product remain valid.

5) Usability :- The effort required to operate the software product

6) Maintainability :- The ease with which it is possible to locate and fix bugs in the software product.

7) Flexibility :- the effort required to adapt the software product to ensure that it performs its intended function.

8) Portability :- the effort required to transfer the software product from one hardware or software system environment to another.

9) Reusability :- the extent to which a software can be reused in other applications.

10) Interoperability :- the effort required to integrate the software with other software.

2) Garvin Quality Dimensions :-

Defined the quality of any product in terms of following attributes of product.

- **Performance** :- How well it performs the job.
- **Features** :- How well it supports the required features.
- **Reliability** : Probability of a product working satisfactorily within specified period of time.
- **Conformance** :- Degree to which the product meets the requirements
- **Durability** :- Measure of the product life.
- **Serviceability** :- Speed and effectiveness maintainance
- **Aesthetics** :- The look and feel of the product.
- **Perceived quality** :- User's opinion about the product quality.

Q 10. a. Describe six generic functions allowed in an automated estimation techniques of software projects.

Ans:- Six generic functions are

1. Project Characterization
   - Collecting and defining the parameters and attributes of the software project.
   - Includes: size (LOC, function points), complexity, required reliability, team experience and technology used.
2. Data Collection and Calibration:-
   - Gathering historical project data to calibrate estimation models.
3. Model Selection and Parameter Tuning
   - Choosing an appropriate estimation model (eg:- COCOMO II)
4. Effort Estimation :-
   - Estimating person-hours, person-months or other units of effort required.
   - Output can be total effort, phased effort (eg:- design,

Coding, testing) or task specific.

5) Cost and Schedule Estimation

- Translating effort into overall and project schedule
- Often considers overheads, resource availability productivity factors and deadlines.

6) Result Analysis and Validation

- Reviewing and validating the estimation results
- May involve comparison with expert judgments or sensitivity analysis.

Q10. b. Explain COCOMO II model.

Ans:- COCOMO Constructive Cost Model. The original COCOMO model became one of the most widely used.

COCOMO II is actually a hierarchy of estimation models that address the following areas:-

- Application Composition model:-
Used during the early stages of software engineering when prototyping of user interfaces, consideration

of software and system interaction, assessment of performance, and evaluation of technology

- Early design stage model:- Used once requirements have been stabalized and basic software architecture has been established.

- Post-architecture-stage model

Used during the construction of the software like all estimation models for software, the COCOMO II models require sizing information. Three different sizing options are available as part of the model hierarchy: object points, function points and lines of source code.

The COCOMO II application composition model uses object points. Like function points, the object point is an indirect software measure that is computed using counts of the number of (1) screens (at the user interface)

2) reports, and (3) components likely to be required to build the first application.

Each object instance (eg: a screen or report) is classified into one of three complexity levels (i.e simple, medium or difficult)

In essence complexity is a function of the number and source of the client and server data tables that are required to generate the screen or report and the number of views or sections presented as part of the screen or report.

Object point count is adjusted:-

$$NOP = (\text{object points}) \times (100 - \%\text{reuse}) / 100)$$

where NOP is defined as new object points.

Sumit