Software Engineering and Project Management

① Define process and explain generic process framework for software engineering?

→ Process: A process is a collection of work, activities, actions and tasks.

The generic process framework for software engineering are:

1) Communication: Critical to understanding stakeholder objectives and gathering requirements of software features and function.

2) Planning: Creates a map or a software project plan that guides the team, It defines technical tasks, risks, resources, work products and schedules.

3) Modeling: Involves creating model to understand software requirement and design.

4) Construction: Combines code generated manually or automated and testing to uncover errors in the code.

5) Deployment: Delivers the software to the customer for evaluation and feedback.

These activities apply iteratively through a number of project iterations, each producing as software increment that adds to overall functionality and completeness.

Diagram at ans (12)

(2) Explain the different attributes are encountered in the vast majority of WebApps.

→ The following are the common attributes for Web apps:

1) Network Intensiveness: Resides on a network and serve a diverse community of clients.

2) Concurrency: Accessed by a large number of users simultaneously.

3) Unpredictable load: Users number can vary drastically day by day.

4) Performance: User retention depends on fast access, server side processing and client-side displaying.

5) Availability: Popular Webapps are expected to accessible 24/7/365

6) Data driven: Present hypermedia content like text, graphics, audio and video.

7) Content Sensitive: Quality and aesthetics of content are crucial for the overall quality of a webapp.

8) Continous Evaluation: Web app evolve continously, often with content updates on a minute-by-minute basis.

9) Immediacy: WebApps have fast time to market, often launched within days or weeks.

10) Security: strong security measures are needed to protect content and ensure secure data transmission

11) Aesthetics: The look and feel of a webapp play a significant role in its appeal.

(3) Discuss the David Hooker's seven principle of software engineering practice.

→ The core principles of software engineering practice are

1) First Principle: The Reason it all Exists.
All the decision should be made keeping in the mind that one reason for the existence of a software system it to provide a value to all its user.

2) Second Principle: Keep it simple, stupid:
To have more easily understandable and maintained system, the design should be simple as possible as one should kep in mind that software design is not a sloppy process.

3) Third principle: Maintain the Vision
For a software project to begin and end sucessfully, a clear vision is very neassary without which the software system weakens.

4) The fourth principle: what do you produce, other will consume.

Always specify and design and implement by keeping in mind, someone else will have to understand what you are doing. Make design keeping the implements in mind. Code with concern for those who will maintain and extend the system.

5) Fifth Principle: Be open to future:

A system with longlife time has more values. True industrial strength software system must last for long time. To do this sucessfully system must be ready to adapt changes.

6) Sixth Principle: Plan about for Reuse.

Reuse save time and effort, the reuse of code design has a major benefit of using object oriented technologies.

7) Seventh Principle: Think! Placing clear, complete thought before action almost produces better results.

A clear, complete thought before action almost always produces better results. Applying the first six principle requires intense thought, for which the potential rewards are enormous.

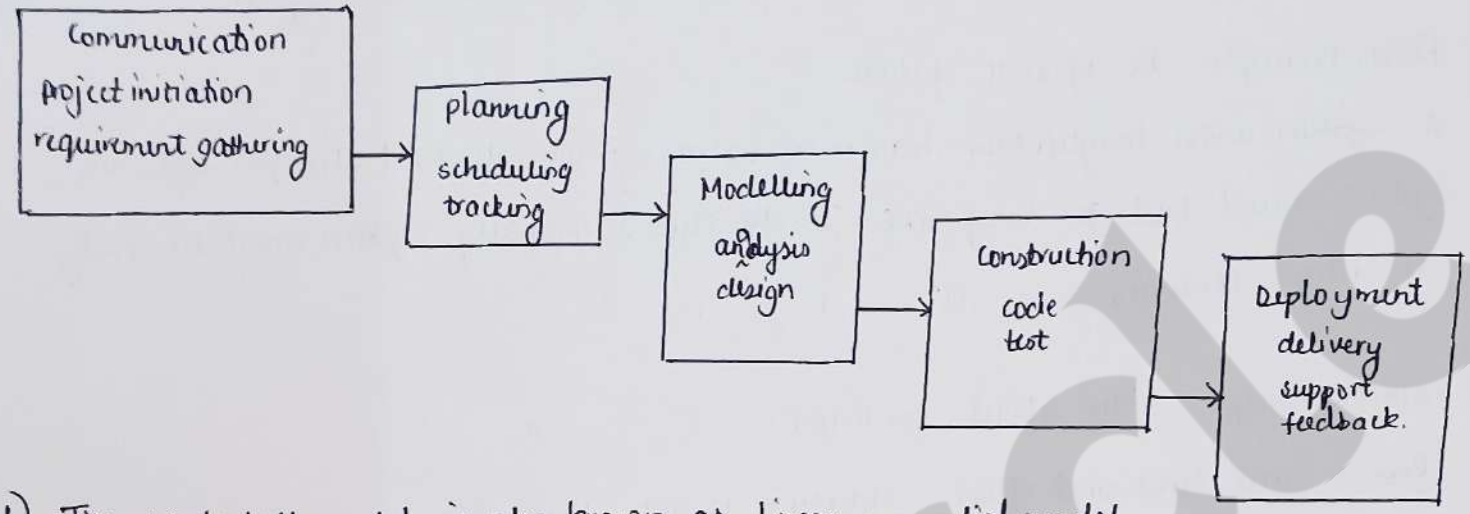
(4) Compare and contrast waterfall model and spiral Model.

Waterfall model	Spiral Model
1) Linear and sequential approach	Iterative and incremental approach
2) Low flexibility	High Flexibility
3) Risk management is limited focus.	Risk management is continous focus
4) Best for small to medium projects	Best for large complex projects.
5) Low customer requirements	High customer requirements.

6) Less expensive and less Time consuming	More expensive and more Time consuming

5) Explain Waterfall Model.

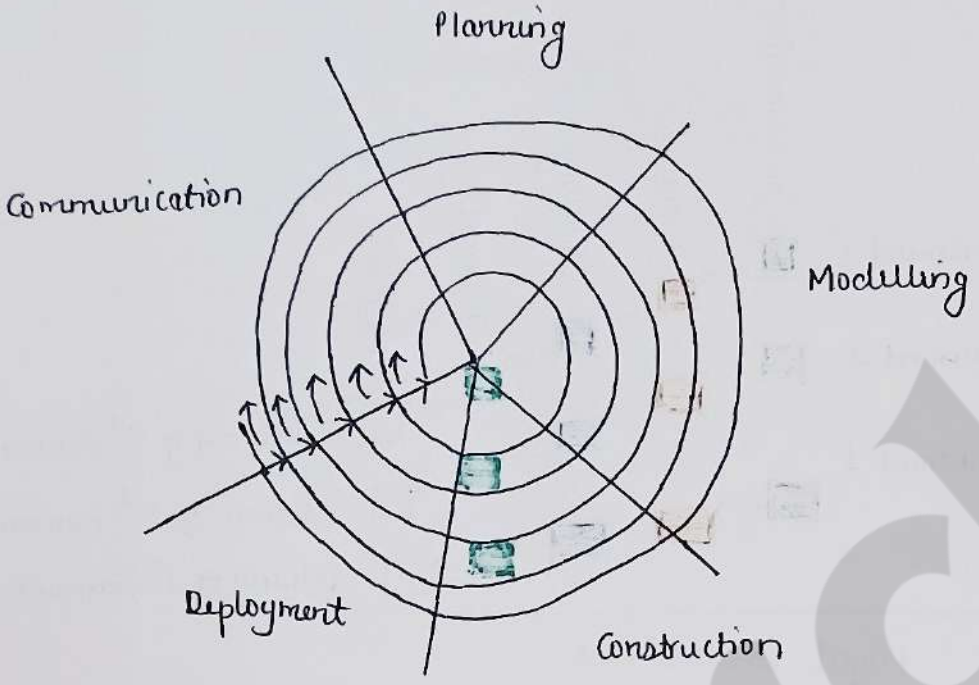
→



1) The Waterfall model is also known as Linear sequential model
2) The process starts with communication, where the requirements are gathered from the customer and recorded.
3) Then goes ~~with~~ to the planning stage where the cost and time constraints are estimated, a schedule is outlined and project tracking variable are defined.
4) Modelling is where a design based on the requirements ~~are~~ and keeping the project constraint in mind is created. After this code is generated and the actual building of product is stated in construction phase.
5) Testing is done after the code completion in this phase.
6) Deployment is the last stage where the product is delivered, customer feedback is received and support an maintainence for the product are provided.

⑥ Explain Spiral Model.

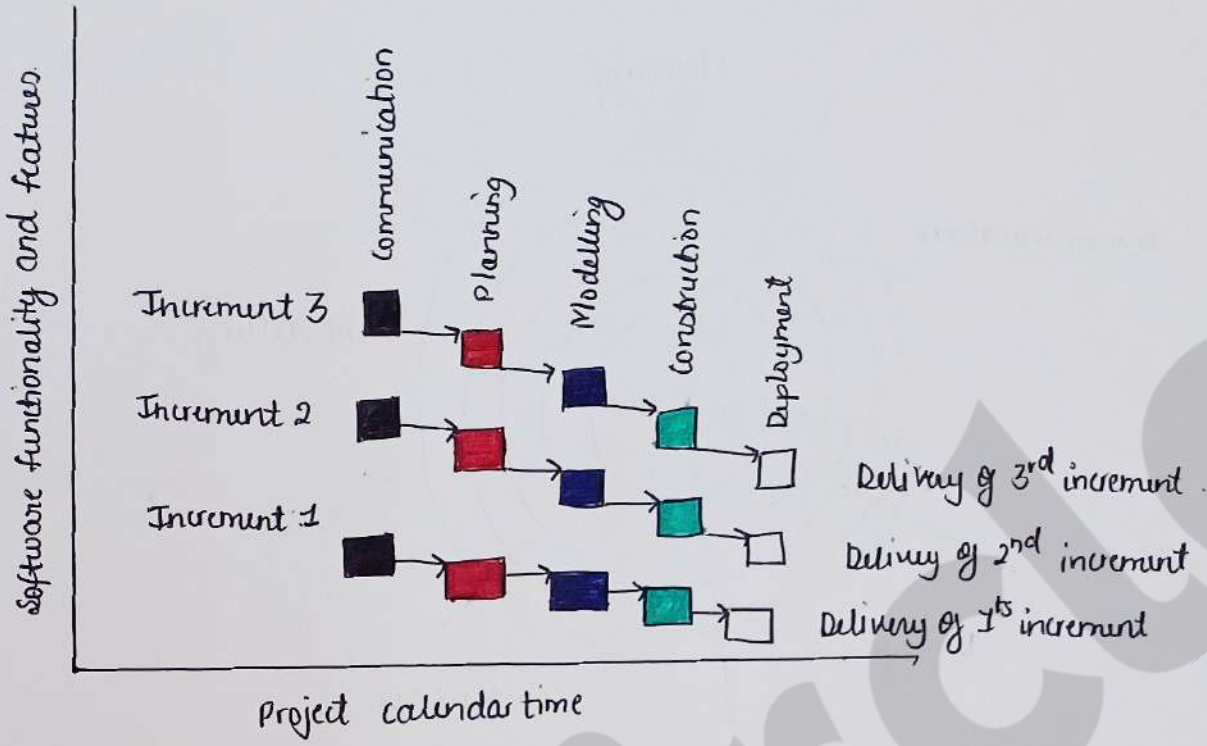
→



1) An evolutionary model which combines the best feature of the classical life cycle and iterative nature of prototype model.
2) Starts in the middle and continually visit the basic task of communication, Planning, Modelling, construction and deployment.
3) The spiral model is a risk driven process model generator that is used to guide multistakeholder concurrent engineering of software intensive system.
4) It has two main distinguishing feature:
   1) One is cyclic approach for incrementally growing a system's degree of definition and implementation while decreasing of risk.
   2) The other is a set of achor point milestones for ensuring stakeholder commitment to feasible and mutually satisfactory system solutions.
5) Realistic approach to the development of large scale system and software. Software ~~Processes~~ evolves as process progresses.
6) Better understanding between developer and customer.
7) The first circuit might result in the development of a product specification.
8) Subsequent circuit develops a prototype and ~~st~~ sophisticated version of software.

(7) Explain Incremental process model. (6)

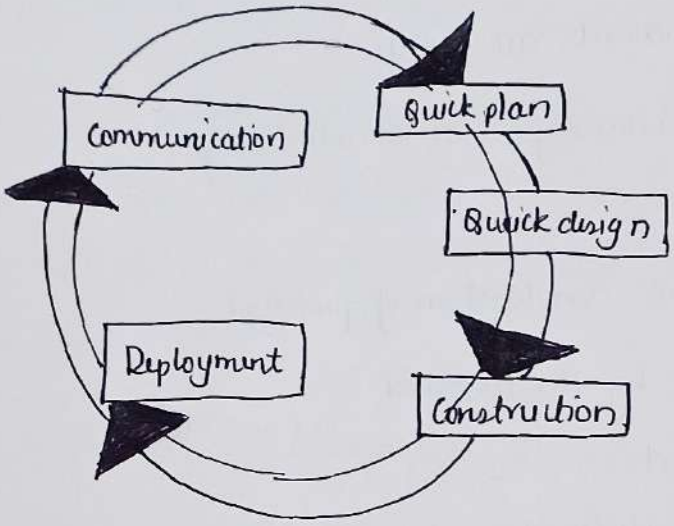
→



1) Incremental model is suited for the project which are iterative in nature.
2) Software is developed in increments.
3) The Incremental model delivers a series of releases, called increments, that provide progressively more functionality for a customer as each increment is delivered.
4) The incremental model combines the elements of waterfall model and they are applied in an iterative fashion.
5) The first increment in this model is generally a core product.
6) Each increments builds the product and submits it to customer for any suggested modifications.
7) The next increment implements on the customer's suggestion and add additional requirement in previous increment
8) The process is repeated until the product is finished.
9) Ex: Word-processing software is developed using incremental model.

(8) Explain Evolutionary Process Model

→



1) Evolutionary process model are opted when the requirements may tend to change and also when complete sophisticated product delivery cannot be done before a given deadline, but the delivery of a limited version of it is possible.

2) In case when the requirements are unclear and are likely to change or when the developer is doubtful about working of an algorithm a solution is to build a prototype and find out what actually needed.

3) Here in this model, one or more prototypes are made with unrefined currently known requirement before actual product is made.

4) A quick design is what occurs in a prototype model. The client evaluates the prototype and gives feedback and other requirement which are incorporated in next prototype.

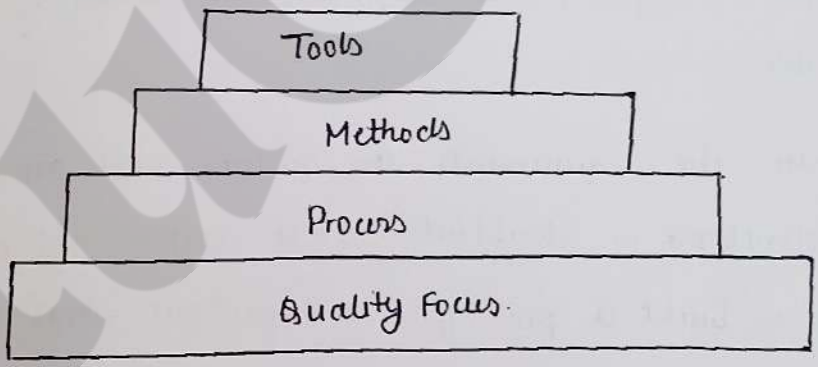
5) This is repeated until the prototype becomes a complete product that is acceptable to the client.

* Steps in prototyping:

1) Begins with requirement gathering
2) Identify whatever requirements are known
3) Outline areas where futher definition is mandatory.
4) A quick design occur.
5) Quick design leads to the construction of prototype.
6) Prototype is evaluated by the customer
7) Requirements are refined.
8) Prototype is turned to satisfy the nuds of customer.

(9) What is software Engineering? and Explain layeud technology of software engineering.

→



Software engineering is a process, a collection of methods, and an array of tools used by professional to build high quality computer software.

1) Quality focus: A quality focus is a first layer of software engineering Any engineering discipline, including software engineering must set its focus on quality of the products throughout the discipline. whenever method, tool or process you use in software engineering discipline focuses must be on quality products.

2) Process: Process is a second layer of software engineering. The process layer act as a co-ordinator to combine other layers of technology. Measurement of work, speed and efficiency is possible because of their layer.

3) Methods: Methods is a third layer of software engineering, which provides "how to's" to accomplish task like analysis, design, coding, debugging. Because of these ~~models~~ methods it is possible to produce quality products within a required time limits.

4) Tools: Tools is a fourth layer of software engineering. This layer provides automated or semi automated support for "process", "Methods", layers to produce quality product.

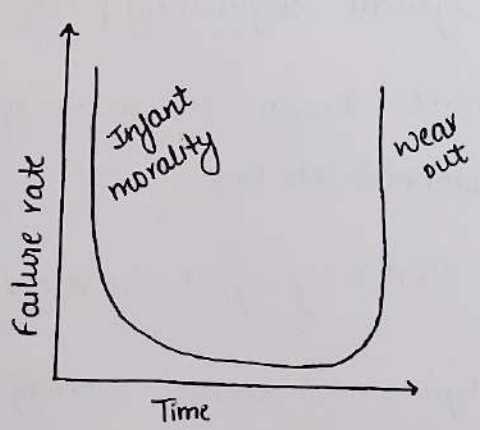
(10) what is software? what are characteristic of good software.

→ A software is a instruction that when executed provide desired function, feature and performance; data structures that enables the program to adequality manipulate information and; descriptive information in both hard copy and virtual forms that describe the operation and use of program.

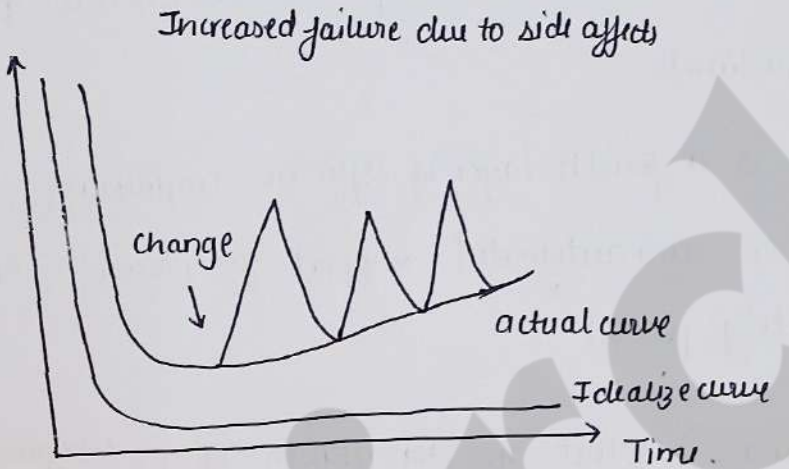
Characteristics:

1) Software is developed or engineered, but is not manufactured in the classical sense.

2) Software does not wear out, but it deteriorates due to change



The figure of failure curve is called as bathtub curve indicates that hardware exibits relatively high failure rates early in life defects are corrected and the failures rate drop to a steady state for some period of time. Failure raises again as hardware components suffer from cumulative affects of environmental maladies.



Software is not susceptible to the environmental maladies. Undiscovered defects will cause high failure rates in the beginning of the program, those failures are corrected and then curve flattens. Spikes in the curve mainly due to the some new defect.

3) Software is custom built rather than assembling existing components.

(11) Identify the umbrella activities in software engineering process.

→ The umbrella activities in software engineering process are

1) Software project Tracking and Control: Compare the progress of project with the plan and take steps to maintain a planned schedule

2) Risk Management: assesses risk that may affect the outcome and quality.

3) Software quality assurance: defines and conduct activities to ensure quality.

4) Technical Reviews: Assessment of errors and correction done at each stage of activity.

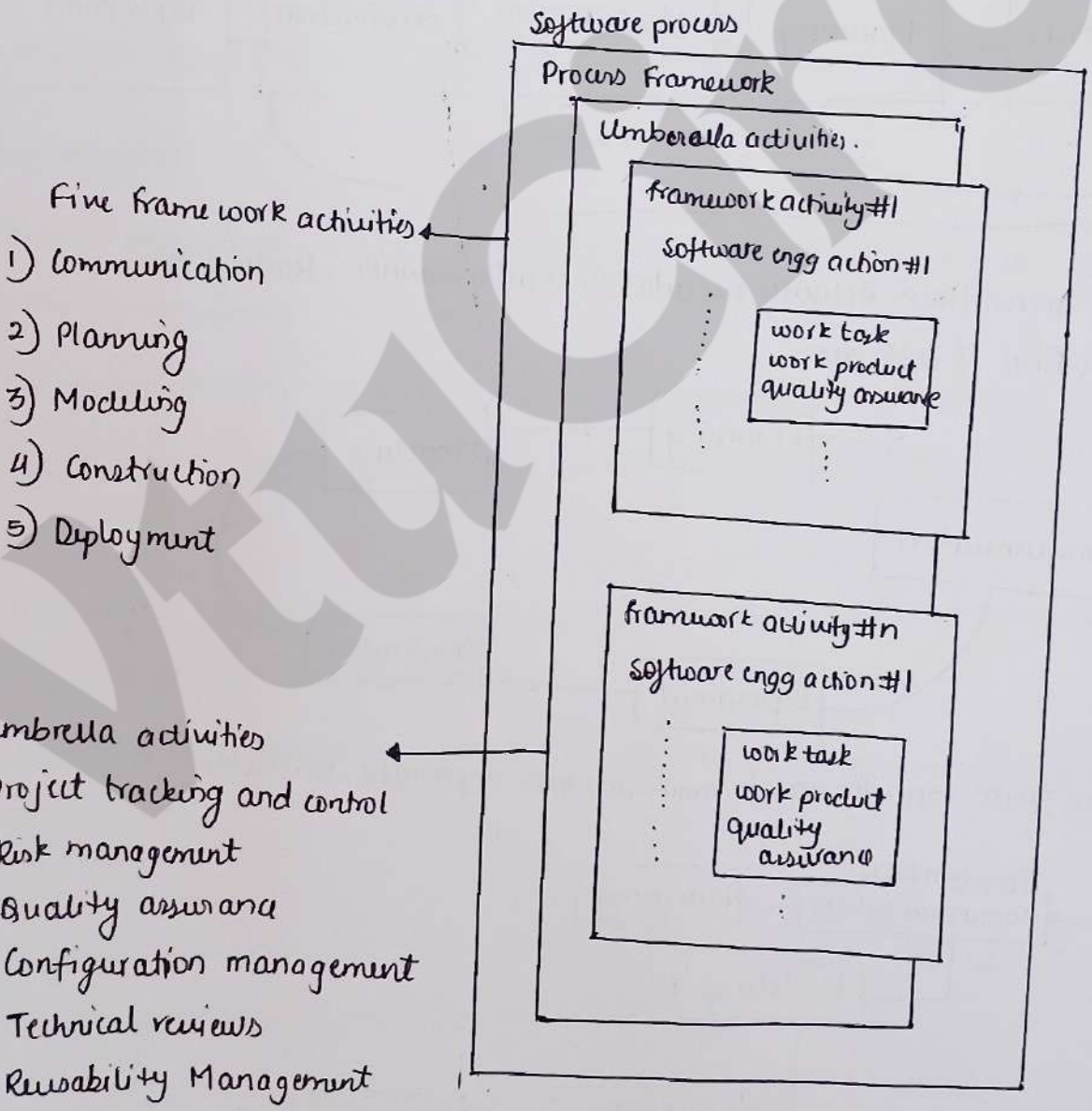
5) Software Configuration Management: Managing of configuration process when any change in software occurs.

6) Reusability Management: Reusable work items should be backed up and reusable software components should be achieved.

7) Work product Preparation and Production: The activities to create models, document, log, forms and list are carried out.

(12) Explain design of software design process model.

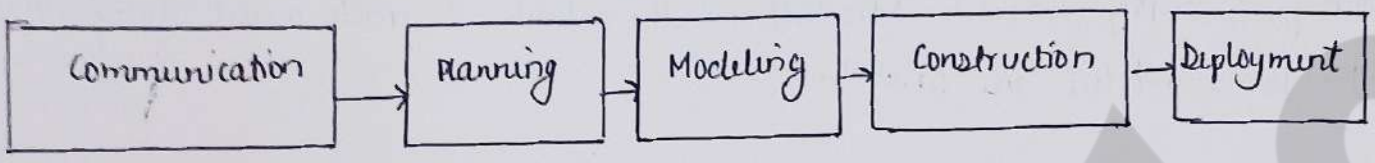
→



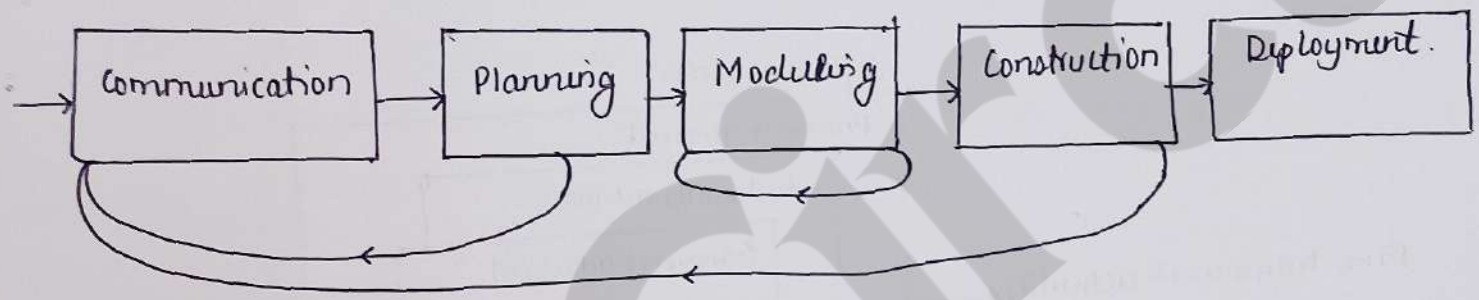
Process flow : Describes organization of framework activities, actions, tasks with respect to sequence and time.

Types of process flow:

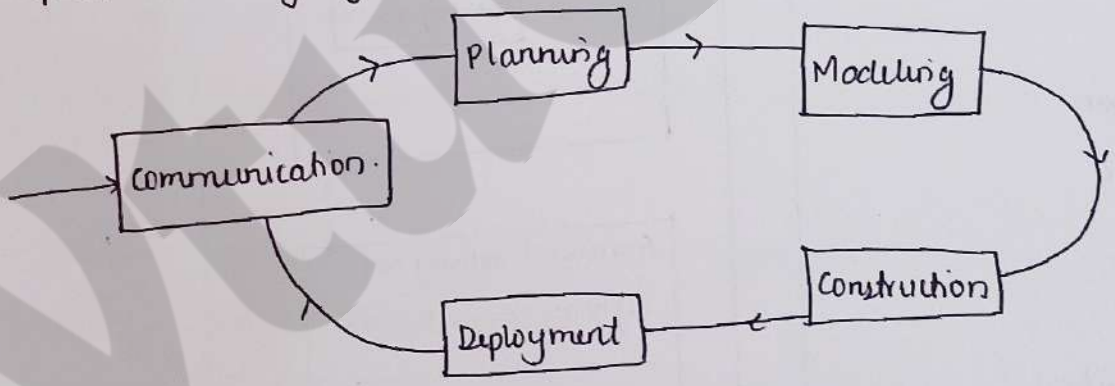
1) Linear process flow: Executes each framework activity in sequence.



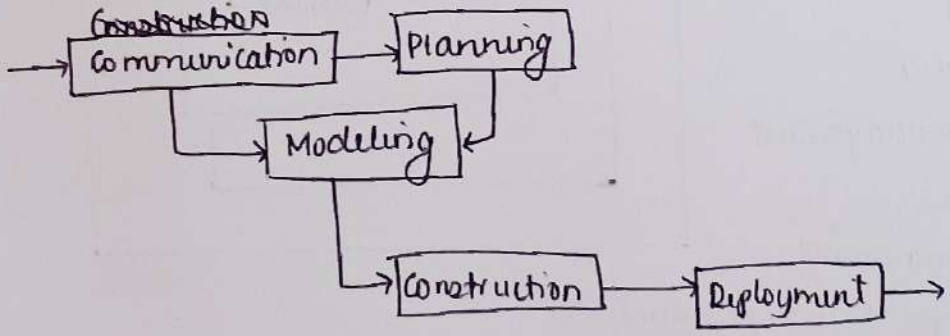
2) Iterative process flow: Repeats one or more activities before proceeding.



3) Evolutionary process flow: Activites executed in circular manner, leading to o more complete version of software.



4) Parallel process flow: Executes one or more activites in parallel with others.



Software Engineering and Project Management

① With an example, describe the class Responsibility - Collaborator (CRC) modeling.

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

* Classes:

1) Entity classes: Entity classes are directly extracted from the statement of the problem.
2) Boundary classes: Boundary classes are used to create the interface that user sees and interact with as the software is used.
3) Controller classes: controller classes manage a unit of work from start to finish.

* Responsibilities:

1) System Intelligence should be distributed across classes to best address the need of problem.
2) Each responsibility should be stated as generally as possible.
3) Information and the behavioural related to it should reside within the same class.
4) Information about one thing should be localized with a single class, not distributed across multiple classes.
5) Responsibilities should be shared among related classes, when appropriate.

* Collaborations:

1) A class can use its own operations to manipulate its own attributes, thereby fulfilling a particular responsibility
2) A class can collaborate with other classes.

Example.

Class : Floor Plan	
Description:	
Responsibility	Collaborator
Define floor plan name/type	
Manages floorplan positioning	
Scales floorplan for display	
Scales floor plan for display	
Incorporates walls, doors, window	wall
shows position of video camera	Camera

(2) Explain three types of QFD with examples.

→ Quality function Deployment is a technique to convert the business needs of the customer into a more technical requirement for software.

The three types of QFD are:

1) Normal Requirement: They reflect objectives and goals of the product. If these requirement are present in final product then customer is satisfied.

Ex: System function, Graphical display, Certain queries.

2) Expected Requirement: Customer does not explicitly state them, but they are implied. These have to be understood by SE team, extracted from the customer in the process of discussion.

Ex: User friendly, UI, Reliability, Operational ease.

3) Exciting Requirement: These are features that go beyond the customer expectation and customer get elated to get them.

Ex: In a graphical display provide choice of graph, help messages.

(5) Explain scenario based model with example.

→ Scenario based model decript how the user interact with the system and specific sequence of activities that occurs as the software is used.

* Creating a preliminary use case:

A use case describes a specific usage scenario in straight forward language from the point of view of a defined actor.

There are question that must be answered if use case are provide value as requirement modeling tool.

1) What to write about
2) How much to write about
3) How detailed to make about your description
4) How to organize the description.

* Refining the Preliminary use case:

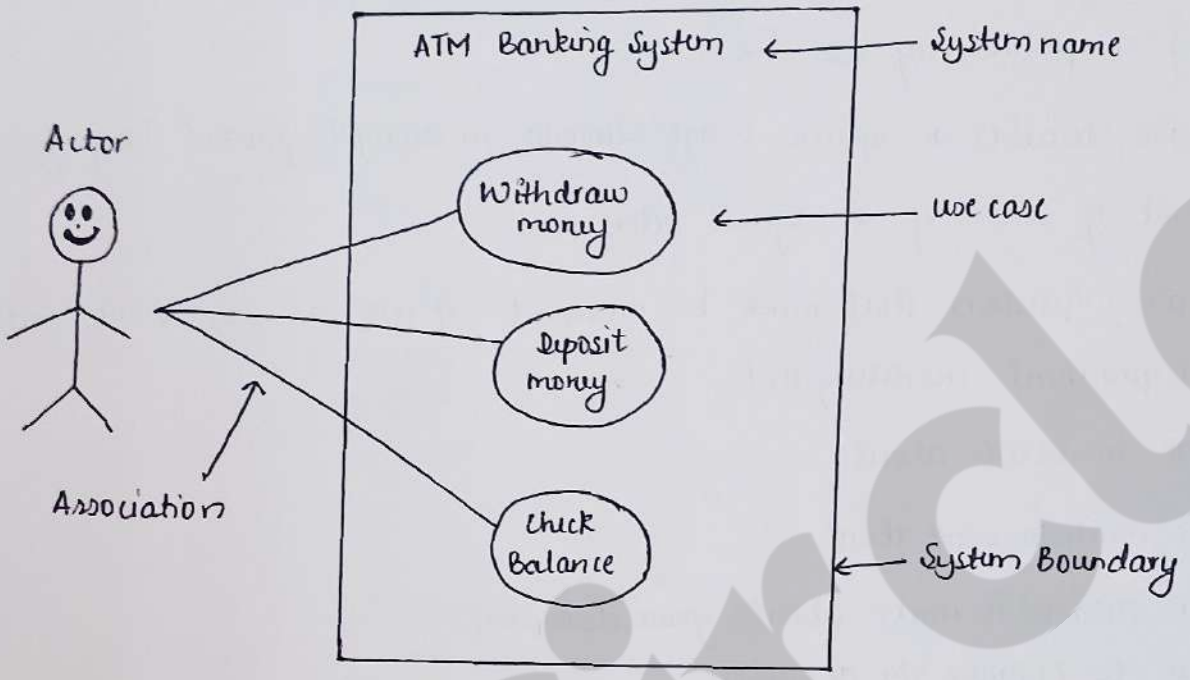
Each step in the primary scenario is evaluated by asking the following questions.

1) Can the actor can take some other action at this point.
2) Is it possible the actor will encounter some error condition at this point? If so, what might it be?
3) Is it possible the actor will encounter some other behaviour at this point? If so, what might it be?

* Writing a formal use case:

1) The goal in the context identifies the overall scope of the use case.
2) The pre condition describes what is known to be true before the use case is initated.
3) The trigger identifies the event or condition that gets the usecase started.
4) The scenario list the specific action that are required by the actor and appropriate system response.

6) The Exception identify the situation uncovered as the preliminary use case is refined.

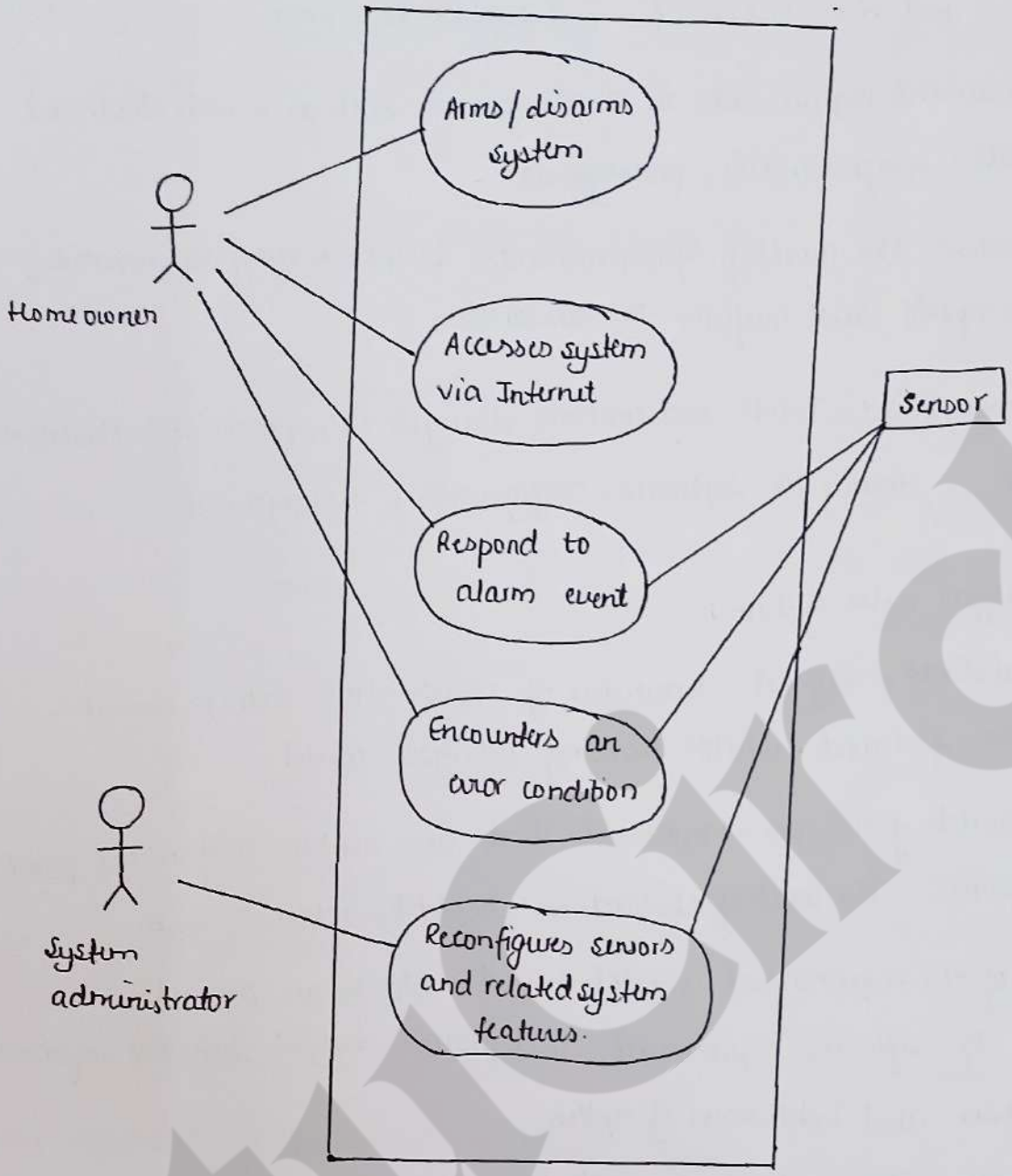


(4) How can you develop an effective use case? Develop a UML use case diagram for home security function

→ To develop an effective use case

1) Define the system scope: Identify the boundaries of the system.
2) Identify the Actors: Determine the entities interacting with the system, like users, or external systems
3) Define the use case: Specify the main function or goal the actor want to achieve
4) Describe each use case: Write a brief narrative or steps for how the actor interacts with the system to accomplish a goal
5) Prioitrize Use Cases: Focus on the most critical or frequently used functionalities first
6) Validate the use cases: Ensure that they are clear, feasible and meet system goals

UML use case diagram for home security function.



(5) What are the different ~~process~~ tasks of ~~understanding software~~ requirements engineering?

→ The different ~~process~~ task of understanding the software requirements engineering are:

1) Inception: Establishes a basic understanding of the problem, stakeholders, desired solution and preliminary communication and collaboration.

2) Elicitation: Involves gathering requirements from customers, often encountering problem of scope, understanding and volatility.

3) Elaboration: Refines and expands the information obtained during inception and elicitation, developing a detailed requirement model.

4) Negotiation: Reconciles conflicts between different stakeholders, requirement through prioritization, cost and risk assessment and iterative discussion.

5) Specification: Document requirements in various forms, such as written document graphical models, usage scenario, prototype etc.

6) Validation: Assesses the quality of requirements to ensure they are unambiguous consistent, complete and confirm to standards.

7) Requirement Management: Tracks and controls changes to requirement throughout the project life, similar to software configuration management.

⑥ Explain the analysis rules of thumb.

→ Arlow and Neustadt suggest a number of worth while rules of thumb that should be followed while creating analysis model.

1) The model should focus on requirement that are visible within the problem or business domain. The level of abstraction should be relatively high.

2) Each element of the requirement model should add to an overall understanding of software requirement and provide insight into the information domain, function and behaviour of system.

3) Delay consideration of infrastructure and other non functional model until design: That is a database may be required, but the classes necessary to implement it, the function required to acess it and the behaviours that will be exhibited at it is used should be considered only after domain analysis have been completed.

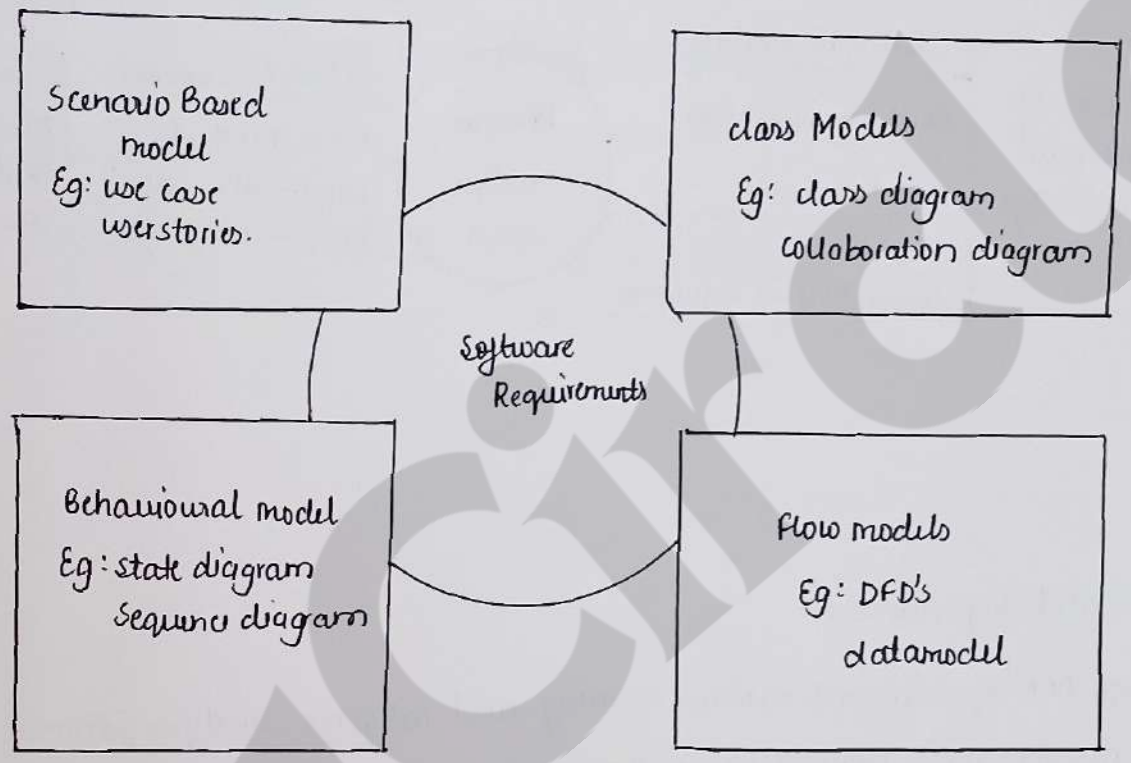
4) Minimize coupling throughout the system: It is important to represent relationship between classes and function. However the level of interconnectedness is extremely high, effort should be made to reduce it.

5) Be certain that requirement model provides value to all the stakeholders: Each constituency has its own use for the model.

6) Keep the model as simple as it can be: Don't create additional diagram when they add no new information, Dont use complex notational forms when a simple list will do.

7) Explain the Requirement modelling approaches with a neat diagram.

→



1) One approach of requirement modeling, called structured analysis considers data and the processes that transform the data as seprate entities.

2) Second approach to analysis modeling, called object oriented analysis, focuses on the defination of classes and the manner in which they collaborate with one another to effect customer requirement.

3) Scenario Based model decript how the user interacts with the system and the specific sequence of activities that occur as the software is used.

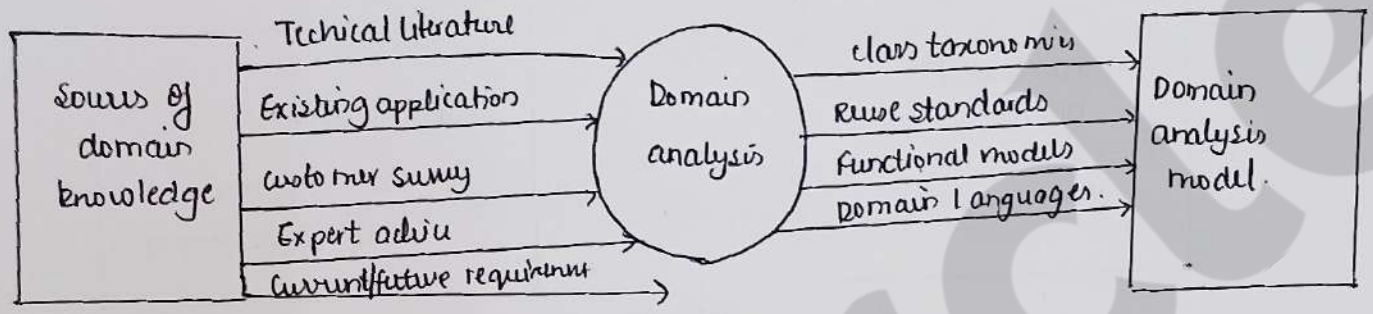
4) class Baosed element model: the object that the system will manipulate, the operations that will be applied to objects to effect the manipulation, relationship between objects and collaboration that occur between the classes are defined

5) Behavioural elements decipt how external event change the state of the system or the classes that reside within it.

6) Flow oriented elements represents the system as an information transform, decipting how data object are transformed as they flow through various system function.

(5) Explain domain analysis with neat diagram.

→



1) Purpose and Importance:

a) Recognition of Recurrent Patterns: Identify and categorize analysis patterns that often reoccur across application within a specific business domain.

b) Expedited Model creation: Recognizing and applying these patterns expedites the creation of the analysis model.

c) Improved Time to Market and Reduced costs: Increases the likelihood of using design patterns and executable software components, leading to faster development and lower costs.

2) Domain analysis defination:

a) Software Domain Analysis: Identification, analysis and specification of common requirements within a specific application domain for reuse across multiple projects.

b) Object oriented Domain Analysis: Focuses on identifying, analyzing and specifying common, reusable capabilities in terms of common objects, classes, and frameworks.

3) Application domain: The goal is to find or create broadly applicable analysis classes and patterns for reuse.

Eg: banking, multimedia video games.

4) Ongoing activity: Domain analysis is a continuous software engineering activity not tied to a specific project.

5) Input and output domain Analysis:

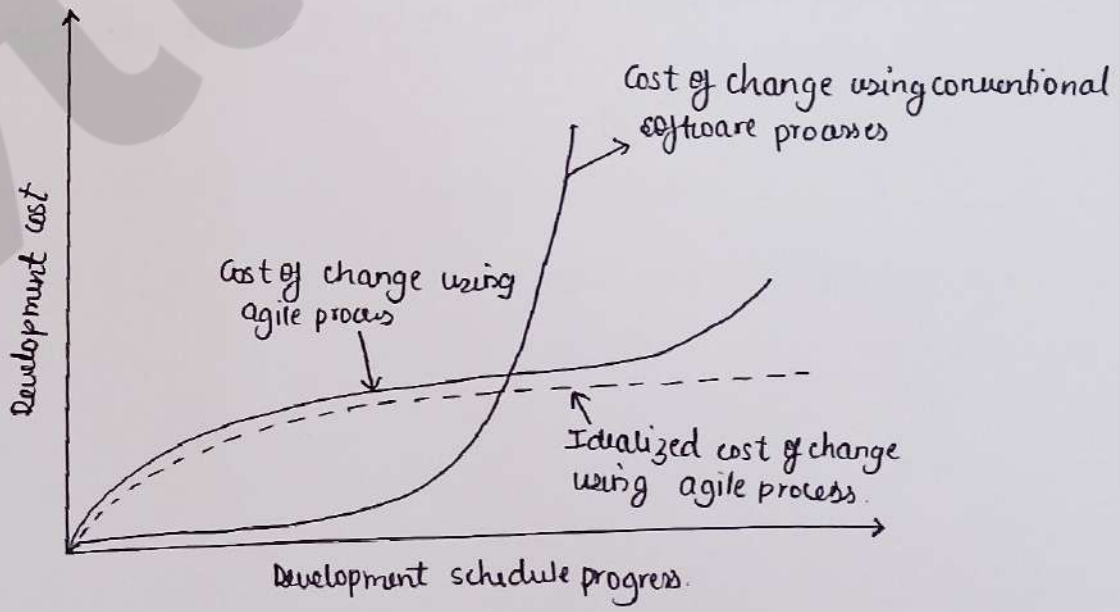
a) Sources of domain knowledge: Surveyed to identify object that can be reused across the domain.

b) output: Analysis pattern, Analysis classes, and related reusable information.

① What is Agility? Explain agility with the cost of change with diagram. Explain the principles of Agile software development.

→ Agility: Agile is a software development methodology to build software incrementally using short iterations of 1 to 4 weeks so that the development process is aligned with the changing business needs.

1) In conventional software development the cost of change increases non linearly as project progresses.
2) An ~~agil~~ agile process reduces the cost of change because software is released in increments and change can be better controlled within an increment.
3) Agility argue that a well designed agile process "flattens" the cost of change curve. allowing a software team to accomodate changes late in a software project without dramatic cost and time impact.
4) When incremental delivery is coupled with other agile process practices such as continous unit testing, and pair programming, the cost of making change is reduced.

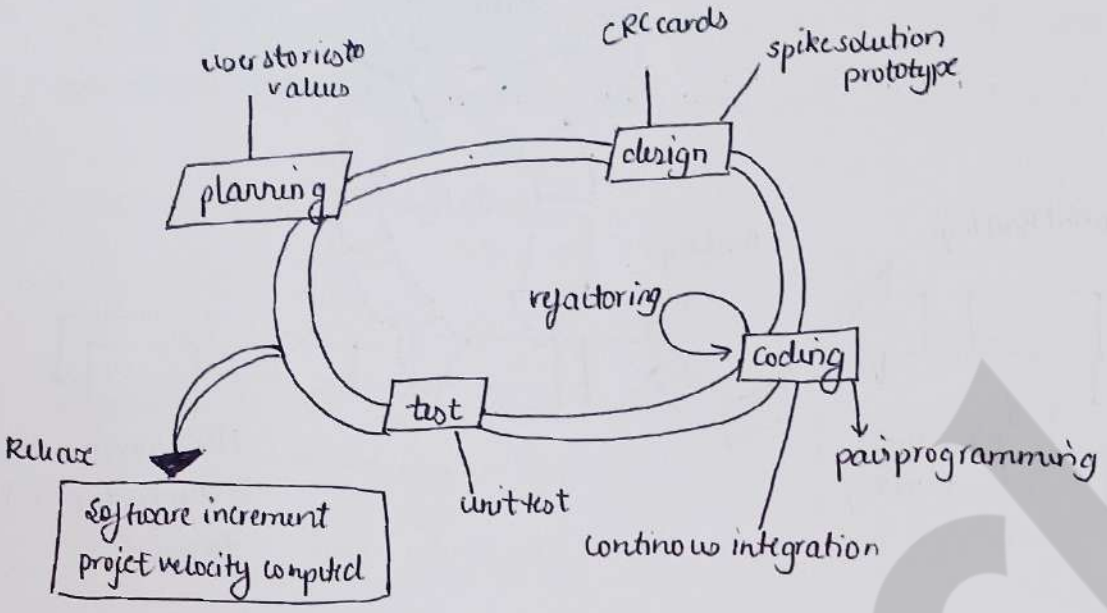


Principle of agile software development

1) Our highest priority is to satisfy customer through early and continous delivery of valueable software.
2) Welcome changing requirements, even late in development for the customers competitive damage
3) Deliver working software frequently, with preference of shorter timescales
4) Busines people and developers must work together daily throughout the project
5) Build projects around motivated individuals, providing them with the environment and support they need.
6) The most efficient method of conveying information within a development team is face to face communication.
7) Working software is the primary measure of progress.
8) Promote sustainable development, maintaing a constant pace indefinitely.
9) Continous attention to technical excellence and good design enhances agility
10) Maximize the amount of work not done
11) The best architecture, requirement and design emerge from self organization teams
12) At regular intervals, the team reflects on how to become more ~~efficient~~ effective & adjust to behaviour accordingly.

② Elucidate the concepts of extreme programming (XP) with its functional diagram.

→



1) Planning: Planning begin with requirement gathering and creating user stories are priotrized, estimated and grouped into releases. Project velocity is used to estimate delivery dates and schedule future releases.

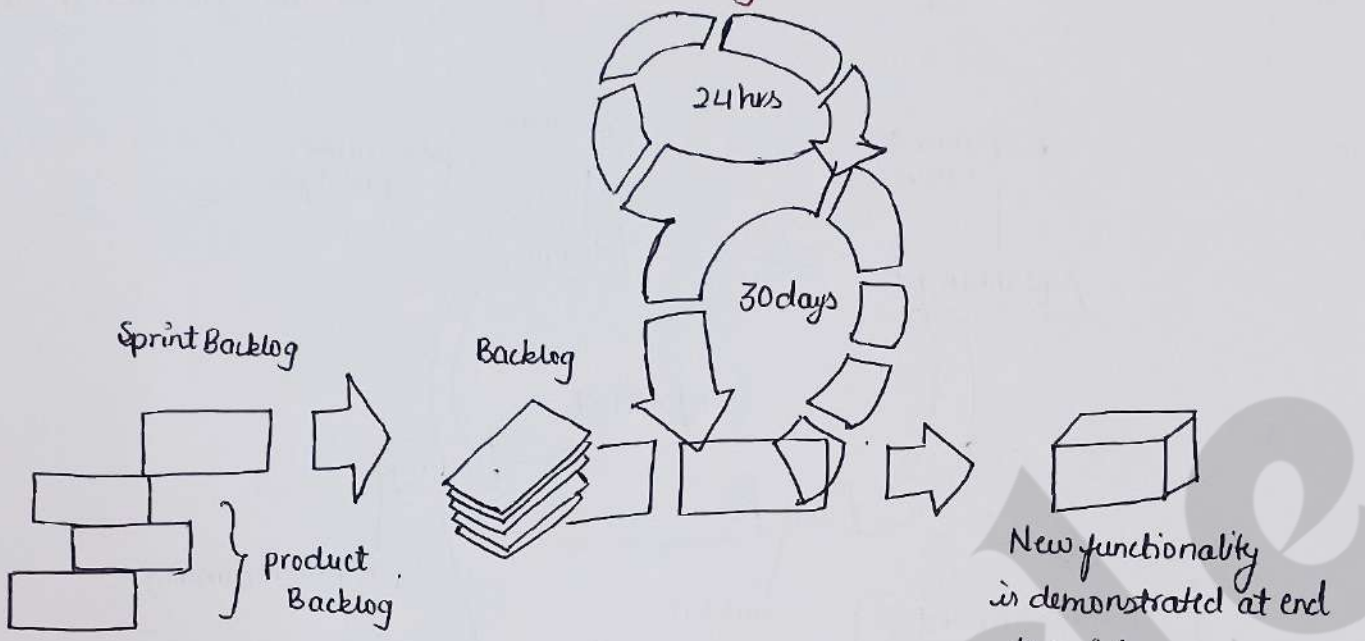
2) Design: Follows keep it simple principle. Design work is minimal and often uses CRC cards. Refactoring is encouraged to improve design continously.

3) Coding: Unit test are written before coding begins. Pair programming is a key practice, ensuring real time problem solving and quality assurance

4) Testing: Unit test should be automated to enable frequent regression testing. Acceptance test focus on overall system features and functionality as specified by customers.

(3) Elucidate SCRUM process with a neat diagram.

→



1) Scrum is an agile software development method which is used to guide development activities within a process that incorporates the following framework activities: requirement, analysis, design, evolution, and delivery.

2) Within each framework activity work ~~task~~ task occur within a process called sprint.

3) The work conducted within a sprint is adapted to the problem at hand and is defined and often modified in real time by Scrum team.

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

5) Sprints : consist of work units that are required to achieve a requirement defined in the backlog that must be fit into a predefined time box. Changes are not introduced during sprint. Hence the sprint allow team members to work in a short-term but stable environment.

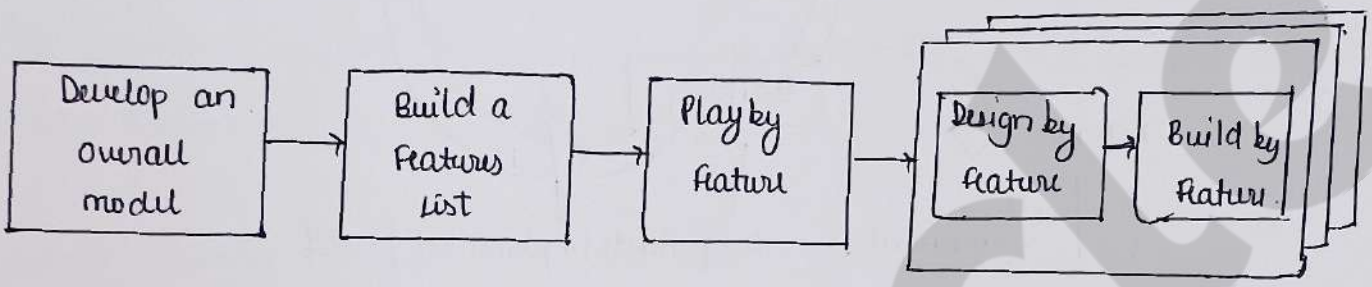
6) Scrum meetings : are short typically 15mins held daily by scrum team, Three key question are asked
   1) what did you do since last meeting
   2) what obstacles are you encountering
   3) what do you plan to accomplish by the next ~~ton~~ team meeting?

7) A team leader, called scrum master, leads the meeting and assesses the response from each person.

8) Demos - deliver the software increment to the customer so that functionality that has been implemented can be demonstrated and evaluated by the customer.

(14) Explain feature driven development.

→



1) Like other agile approaches, FDD adopts a philosophy that

a) emphasizes collaboration among people on an FDD team

b) Manages problem and project complexity using feature based decomposition followed by integration of software increments.

c) communication of technical detail using verbal, graphical, text based means.

2) In the context of FDD, a feature, "is a client-valued function that can be implemented in two weeks or less. The emphasis on the definition of feature provide the following benefits.

a) Because features are small blocks of deliverable functionality, users can describe them more easily. understand how they relate to one another more readily

b) Features can be organised into a hierarchical business related grouping

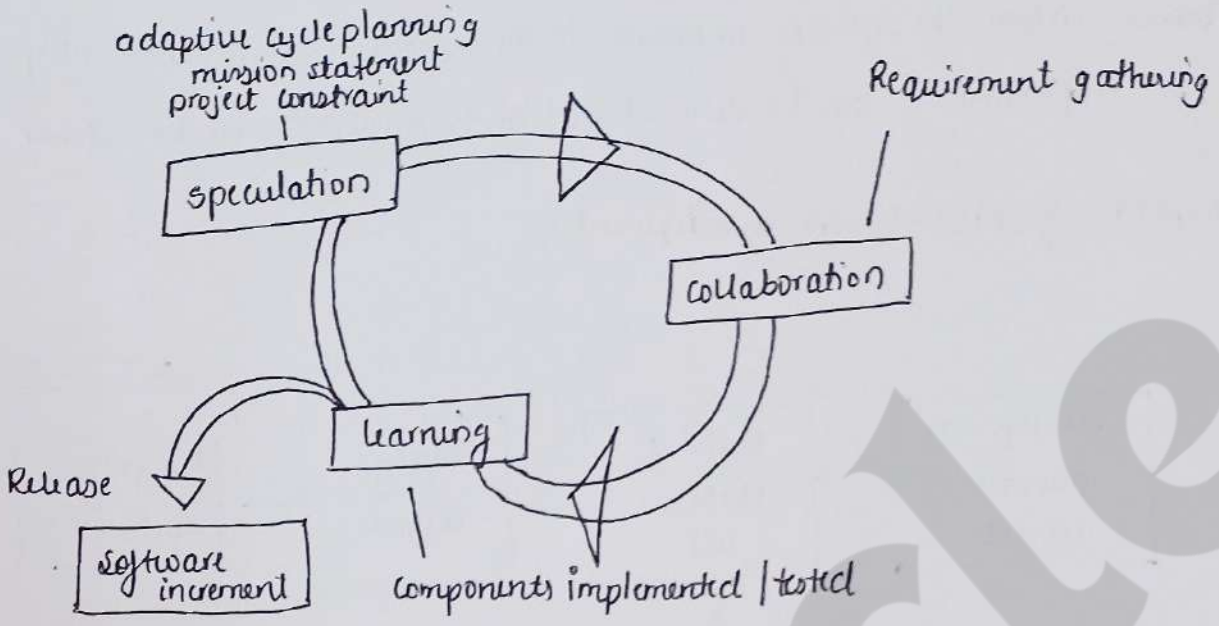
c) Since a feature is the FDD deliverable software increment, the team develops operational feature every two weeks.

d) Because features are small, their design and code representation are easier to inspect effectively

e) Project planning, scheduling and taking are driven by the feature hierarchy rather than an arbitrarily adopted software engineering task set.

⑤ Explain Adaptive software development.

→



1) Speculation: The project is initated 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 a set of release cycles that will be required for the project.

2) Collaboration: This approach is a recurring theme in all agile methods.

It encompasses communication and teamwork, but it also emphasizes individualism, because individual creativity plays an important role in collaborative thinking. Its all about the matter of trust.

People working together must trust eachother

1) critize without disliking
2) assist without feeling of anger
3) work as hard as or harder that they do
4) have the skill set to contribute to the work at hand
5) communicat proass or concerns in a way that leads to effective action

3. Learning: Learning will help them to improve their level of real understanding

ASD teams learn in 3 ways: focus group, technical review, and project postmortems.

⑥ State and explain design modeling principles.

→ Principle 1: Design should be traceable to the requirement model

The requirement model describe the information domain of problem, user visible function, system behaviour and a set of requirement classes that package business objects with the methods that service them.

Principle 2: Always consider the architecture of the system to be built.

Software architecture is the skeleton of the system to be built. It affects interfaces, data structures, program control flow and behaviour in which testing can be conducted.

Principle 3: Design of data is important as design of processing functions.

Data design is an essential element of architectural design, a well structured data design helps to simplify program flow, makes implementation of software component easier and makes more efficient.

Principle 4: Interfaces must be designed with care.

A well-designed interface make integration easier and assist the tester in validating component function.

Principle 5: User interface design should be tuned to the needs of end user.

The user interface is the visible manifestation of the software. No matter how sophisticated its internal function, no matter how comprehensive its data structure, no matter how well designed its architecture, a poor interface design often leads to the perception that software is bad.

Principle 6: Component level design should be functionally independent.

The functionality that is delivered by a component should be cohesive. that is it should focus on one and only one function or subfunction.

Principle 7: Component should be loosely coupled to one another and to the external environment.

Component coupling should be kept as low as it reasonable.

Principle 8: Design representation should be easily understandable

The purpose of design is to communicate information to practioners who will generate code, to those who will test software, and to others who may maintain the software in the future. If the design is difficult to understand it will not serve as an effective communication medium.

Principle 9: The design should be developed iteratively.

The first iteration work to refine the design and correct errors, but later iterations should strive to make the design as simple as possible.

(7) State Requirement modeling principles.

→ Principle 1: The information domain of a problem must be represented and understood.

Principle 2: The function that the software performs must be defined.

Principle 3: The behaviour of the software must be represented.

Principle 4: The model that depict information function and behaviour must be partitioned in a manner that uncovers detail in a layered fashion.

Principle 5: The analysis task should move from essential information toward implementation detail.

Module 4

Software Engg and Project Management.

① Explain the procedure of setting objectives for sucessfull completion of software project.

→ The procedure of setting objectives for sucessful completion of software project are

1) Project Owners: Stakeholder who own the project, control its financing and set the objectives.

2) Object Dyfinition: Objectives dyfine what the project team must achieve for project sucess and identify shared intentions among stakeholders.

3) Outcome Focused: Objectives focus on desired outcomes rather than specific tasks

4) Multiple Routes to Sucess: There are often several ways to meet an objective, which is advantageous.

5) Project Authority: When multiple stakeholders claim project ownership, a project steering committee with overall authority is needed to set, monitor, and modify objectives.

6) Project Managers Role: The project manager runs the project daily and reports regularly to the steering committee.

Subobjectives and Goals:

1) Effective Objectives: Must be within the control of the individual or team responsible

2) Steps to Achieving Objectives: Sub objectives or goals are steps toward achieving the main objective.

SMART objectives:

1) Specific: Objectives should be concrete and well defined.

2) Measurable: There should be measures of effectiveness to determine succes.

3) Achievable: The objective must be within the power of the individual or group to achieve

4) Relevant: The objective should be relevant to the true purpose of the object.

5) Time Constrained: There should be a defined point in time by which the objective should be achieved.

② With example explain different categories of software Project.

→ Categories

① Compulsory vs Voluntary Users:

a) Compulsory users

→ users must use the system to perform task

→ Easier to elicit precise requirements from users.

→ Ex: recording a scale.

b) Voluntary users:

→ users choose to use the system

→ Difficult to elicit precise requirement

→ Ex: computer games.

② Information System vs. Embedded System

a) Information System

→ Enables staff to carry out office processes

→ Ex: stock control systems.

b) Embedded System:

→ Control machines

→ Ex: Air conditioning control in a building.

c) Hybrid Systems:

→ Combine elements of both information and embedded system.

→ Ex: Stock control system that also controls an automated warehouse.

(3) Outsourced Projects:

1) Commercial Sense of outsourcing: Companies may outsource parts of a project if they lack expertise or find its cost effective.

2) Project characteristics: Outsourced projects are typically small and need to be completed within a few months.

3) Management challenges: Managing outsourced project ~~exists~~ entails special challenges due to their size and time constraints

4) Indian Software Companies: Indian companies are renowned for executing outsourced software project and are beginning to focus on product development.

5) Revenue Impact: Generic software products provide long term revenue, while outsourced project offer one time revenue.

(4) Objective Diven Development:

1) Project Distinctions: Projects can aim to produce or meet certain objectives.

2) Client Responsibility: In product creation, the client justifies the product.

3) objective-driven Projects: These projects identify solutions to problem, leading project recommendations.

4) ~~Two~~ Two stage Projects: Initial objective driven stage leads to recommendation followed by creation if needed.

5) Technical work by External Group: Useful when user needs are unclear, allowing for a preliminary design and subsequent implementation based on agreed requirement.

(3) Define project with its characteristics.

→ Project: Project is a planned activity and temporary activity under taken to create a unique product or service.

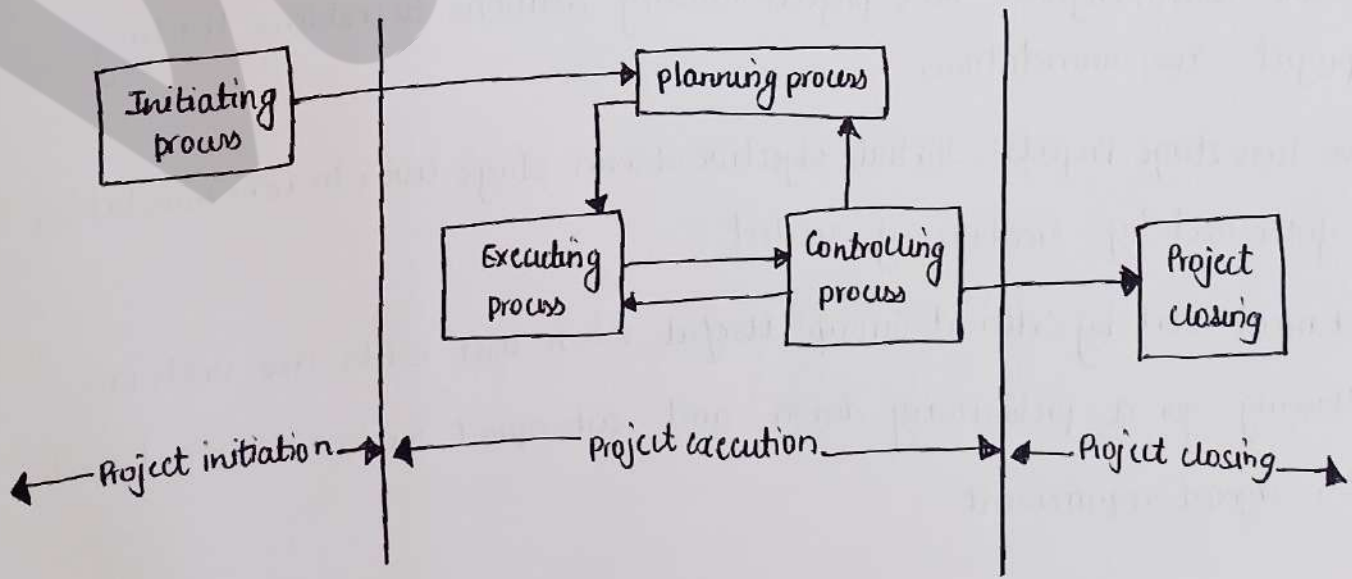
Characteristic of a project are:

1) Non routine task are involved
2) Planning is required
3) Specific objectives are to be met or a specified product is to be created.
4) The project has pre determined time span
5) Work is carried out of for someone other than yourself
6) Works involves several specialism's
7) Work is carried out in several phases.
8) The project is large or complex
9) The resources that are available for use on the project are constrained.

(4) What is management and explain the principal of project management process.

→ Management: Management is the process of planning, organizing, leading and controlling resources to achieve specific goal efficiently and effectively.

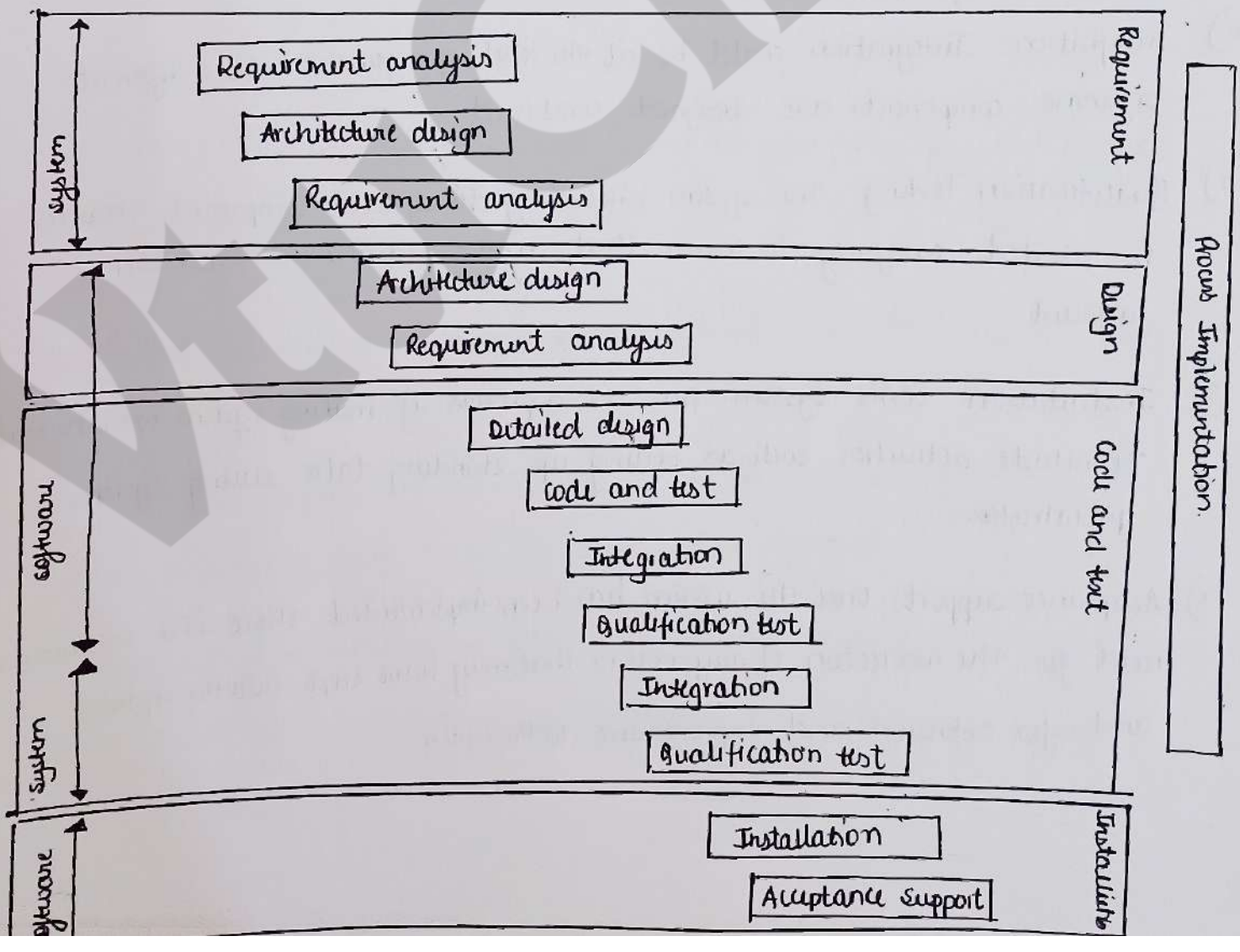
Principal of project management process:



1) In the project initiation stage, an initial plan is made.
2) As a project starts, the project is monitored and controlled to process as planned.
3) Initial plan is revised periodically to accommodate additional detail and constraints about the project as they become available.
4) Finally the project is closed.
5) Initial project is undertaken immediately after the feasibility study phase and before starting the requirement analysis and specification process.
6) Initial project planning involves estimating several characteristics of a project.
7) The Monitoring activity involves monitoring the progress of the project.
8) Control activities are initiated to minimize any significant variation in their plan.
9) Project Planning is an important responsibility of the project Manager, during project planning the project manager needs to perform well defined activities.

⑤ Describe ISO software development life cycle.

→



1) Requirement analysis: starts with requirement elicitation or requirement gathering which establishes what the potential users and their managers require of new system. It could be quality requirement how well the function must work.

2) Architecture design: This maps the requirement to the components of the system that is to be built. At the system level, decision will need to be made about which processes in the new system will be carried out by the user and which can be computerized. This design of the system architecture thus forms an input to the development of the software requirements.

3) Detailed designed: Each software component is made up of a number of software units that can be suprately coded and tested.

4) coding: This may refer to writing code in a procedural language or an object oriented language or could refer to the use of an application builder.

5) Testing: Careful testing will be needed to check that the proposed system meets the requirement.

6) Integration: Integration could be at the level of software where different software components are combined.

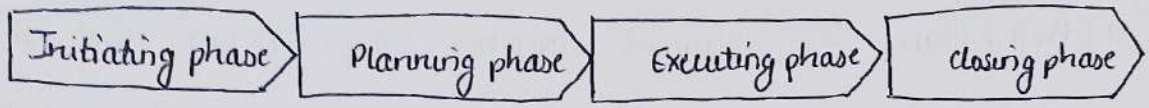
7) Qualification Testing: The system, including the software components, has to be tested carefully to ensure that all the requirement have been fulfilled.

8) Installation: This is a process of making system operational It include activities such as setting up standing data setting system parameters.

9) Acceptance support: Once the system has been implemented there is a continous need for the correction of any errors that may have crept into the system and for extension and improvement to the system.

⑥ Illustrate project management life cycle with its steps

→



1) Project initiation. The project initiation phase start with project concept development During concept development the different characteristics of the software to be developed are understood, which include, the scope of the project and project constraints. Based on this understanding a feasibility study is undertaken to determine the project would be financially and technically feasible.

Based on feasibility study, the business case is developed. Once the top management agress the Business case, the project manager is appointed, the project charter is written and finally project team is formed. This sets the ground for the manager to start the project planning phase.

2) Project Binding: Once the top management agrees the Business case, the project charter is developed.

The types of Binding techniques are

a) Request for quotation: An organization advertises an RFQ if it has good understanding of the project and possible solutions.

b) Request for proposal: The purpose of RFP is to get an understanding of the alternative solutions possible that can be deployed.

c) Request for Information: Based on vendor respose to the RFI, the organization can assess the competencies of the vendor and shortlist the vendor who can bid for work.

3) Project Planning : During the project planning the project manager carries out several process and create following documents.

a) Project Plan: This document identifies the project task and schedule for the project task that assigns project resources and time frames to the tasks.

b) Resource Plan: It list the manpower, resources and equipment that would be required to execute the project.

c) Functional Plan: It documents the plans for manpower, equipment, and other cost.

d) Quality Plan: Plan of quality targets and control plans are included in this world.

e) Risk plan: This document list the identification of potential risk, their prioritization and a plan for the actions that would be taken to contain the different risk.

4) Project Execution: In this phase the task are executed as per the project plan developed during the planning phase. Once all the ~~deliverables~~ deliverables are produced and accepted by the customer, the project execution phase completes and the project closure phase starts.

5) Project closure: Project closure involves completing the release of all required deliverables to the customer along with the necessary documentation.

Finally, a post implementation ~~review~~ review is undertaken to analyze the project performance and to list the lessons for use in future project.

⑦ Differentiate between traditional v.s modern project management practices.

→	Traditional Project Management	Modern project Management.
1)	Linear and sequential approach Eg: waterfall	Iterative and flexible Eg: Scrum
2)	Detailed upfront planning with fixed scope	Adaptive planning throughout the project life cycle.
3)	Follow a predefined plan with minimal changes	Frequent iteration with evolving requirements
4)	Centralized control by the project manager	Decentralized self organizing teams
5)	Delivered as a complete product at the end	Delivered incrementally in smaller iterations.
6)	Traditional Tools	Modern tools
7)	Succes measured by scope, time, and budget.	Succes measured by value delivered and customer feedback.

Software Engineering and project management

① Define Software quality and explain place of software quality in project management.

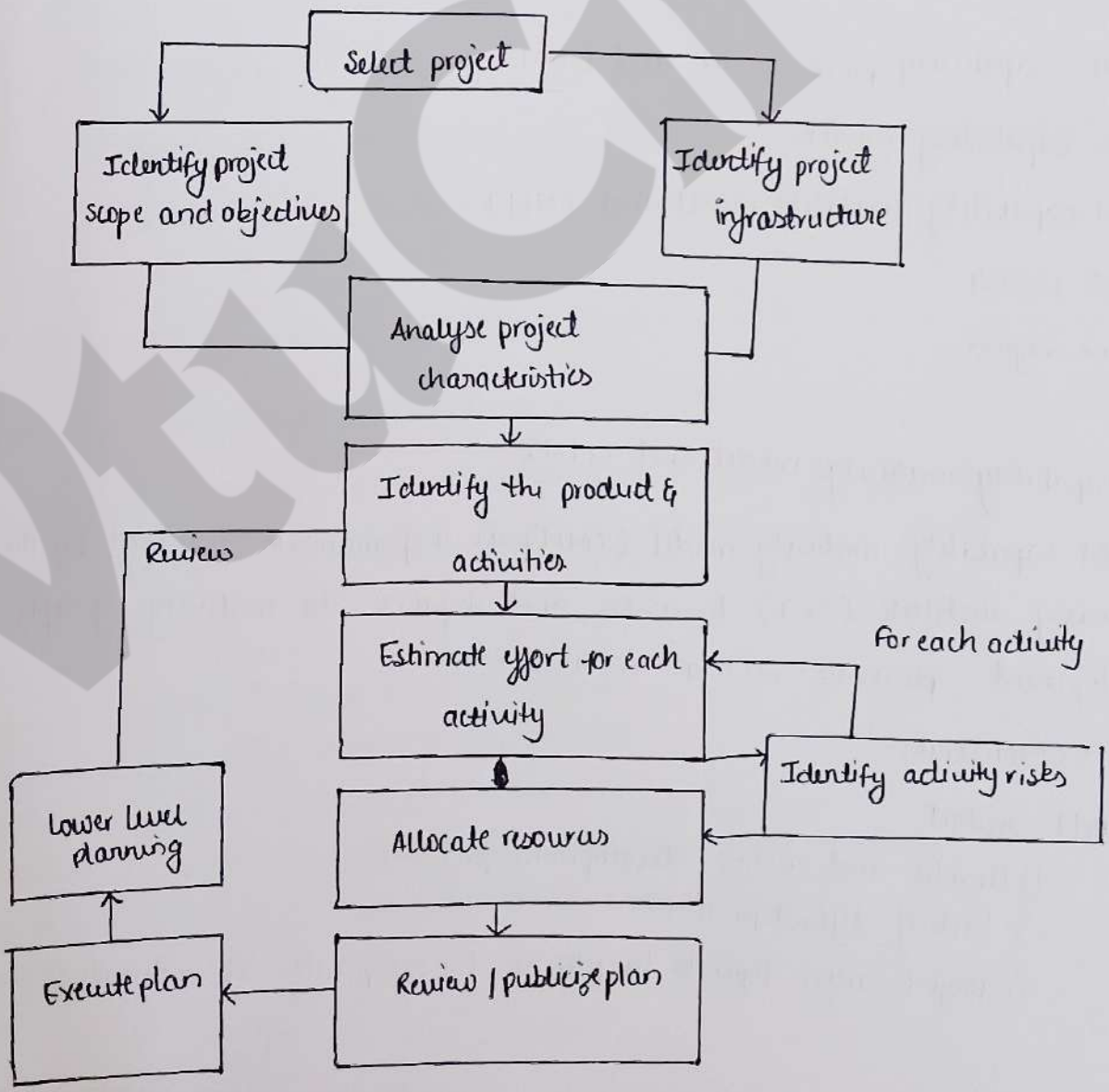
→ Software quality: For any system there should be three specifications

a) A functional specification describing what the system is to do

b) A quality specification concerned with how well the function operate

c) A resource specification concerned with how much is to be spent on the system.

The place of software quality in project management:



1) Step1: Identifying project scope and objectives: Objectives may include qualities of the application to be delivered.

2) Step2: Identify project infrastructure: Within this step activity 2.2 involves identifying installation standards and procedures.

3) Step3: Analyze project characteristics: In this activity the application to be implemented will be examined to see if it has any special quality requirement.

4) Step4: Identify the product and activities of the project: It is at that point the entry, exit and process requirement are identified for each activity.

5) Step8: Review and publicize plan: At this stage the overall quality aspects of the project plan are reviewed.

(2) Explain capability process model and CMM key areas.

→ Process capability model:

1) SEI capability maturity model and CMMI
2) ISO 15504
3) Six Sigma.

1) SEI capability maturity model and CMMI

The SEI capability maturity model (CMM) is a framework developed by the software engineering institute (SEI) to assess and improve the maturity of software development processes within organization.

SEI CMM Levels

1) Level1: Initial
   1) Chaotic and adhoc development processes
   2) Lack of defined processes
   3) Project sucess depends largely on the capabilities of individual team members.

2) Level 2: Repeatable

1) Basic project management practices like planning and tracking are inplace

2) Organization can repeat sucessful practies of similar projects

3) Level 3: Defined

1) Processes for both management and development activities are defined and document.

2) Roles and responsibilities are clear across the organization

3) Consistent and standardized processes across organization.

4) Level 4: Managed:

1) Quality goals are set and measured aganist project outcomes

2) Project metrics are used to improve project performance.

5) Level 5: Optimizing:

1) Poast metrics are analyzed to identify areas for improvement.

2) Continous innovation and improvement in processes.

* CMMI (capability, maturity model Integration)

The CMMI was produced to provide unified framework that could be applied across various software development, including system engineering, product development and services.

Levels:

Level 1: Initial (Similar to CMM level 1)
Level 2: Managed (Similar to CMMI level 2)
Level 3: Defined (similar to CMMI level 2)
Level 4: Quantitatively managed (an extension of CMMI level 4)
Level 5: Optimizing (an extension of CMM level 5)

Key process areas

Definition : Similar to CMM, each maturity level in CMMI is characterized by a set of key process areas. These KPA represent activities that when performed collectively achieve a set of goals for enhancing process capability.

2) ISO 15504 process assessment

ISO/IEC 15504, also known as SPICE is a standard for assessing & improving software development processes.

Process attributes

1) Process performance : Measures the achievement of process specific objectives
2) Performance management : evaluates how well the process is managed and controlled
3) Work product management : assesses requirement like design document etc
4) Process definition : Focuses on how well the process is defined and documented
5) Process deployment : Examines how process is deployed within organization
6) Process Measurement : Evaluates the use of measurement to manage & control
7) Process control : Monitoring and control mechanism
8) Process Innovation : Improvement in process
9) Process optimization : Improve efficiency and effectiveness.

(3) Explain Quality Management System with principles of BSEN ISO 9001 : 2000

→ The British standard Institution have engaged in the creation of standard for quality management system. The British standard is now called BSEN ISO 9001 : 2000

ISO 9001 describes how QMS can be applied to a creation of product and provision of services.

1) Fundamental Features: Describe the basic principle of QMS, including customer focus, leadership, involvement of people, process approach.

2) Applicability of software development: ISO 9001:2000 can be applied to software development by ensuring that the development processes are well defined, monitored and improved.

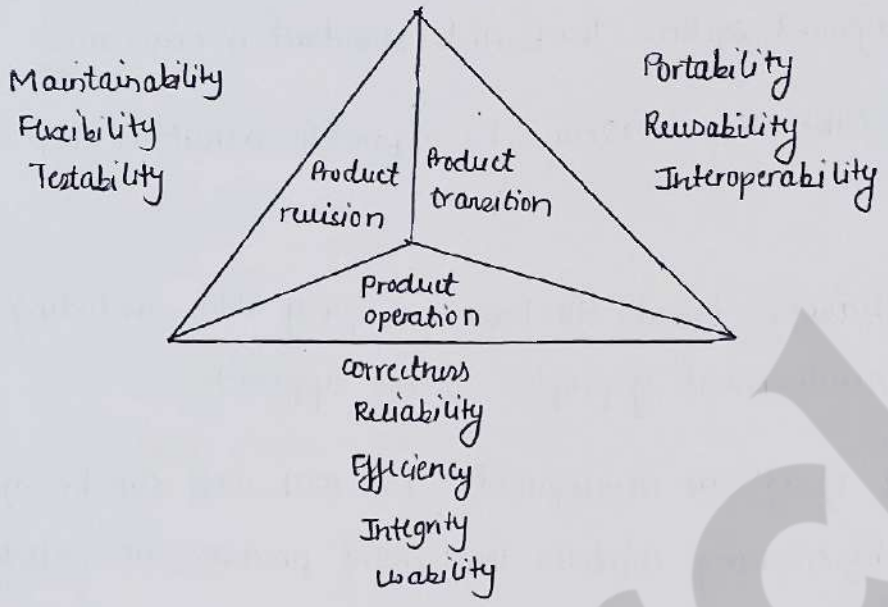
3) Certification process: Certification demonstrates that the organization meets the requirement of ISO 9001 : 2000 and has implemented an effective QMS.

4) Quality Management Principles:

1) Customer Focus: Meeting customer requirement and enhancing customer satisfaction
2) Leadership: establishing unity of purpose & direction.
3) Involvement of people: achieving quality objectives.
4) Process approach: Managing activities and resources to achieve desired outcomes.
5) Continous improvement: Continually improving QMS effectiveness.

(4) Explain McCall's Model.

→



1) Mc Calls software Quality Model was introduced in 1977. This model incorporates with many attributes, termed software factors, which influence software.

2) The model distinguishes between two levels of quality attributes
   1) Quality Factor
   2) Quality criteria

3) Quality Factor: The higher level quality attributes that can be accessed directly are called quality factor. These attributes are external.

Quality criteria: The lower level quality attributes that can be accessed directly either subjectively or objectively are called as quality criteria. These attributes are internal.

4) Product Operation: Product operation have 5 quality factors
   a) Correctness refers to whether the software component behaves as expected, fulfilling functional requirements.
   b) Efficiency: The number of hardware resources and code the software, need to perform a function.
   c) Integrity: The extent to which the software can control the unauthorized person from accessing the data or software.

d) Reliability: The extent to which software performs its intend function without failure

e) Usability: The extent of effort required to learn, operate and understand the function of software

5) Product Revision: Product Revision include 3 software quality factors.

a) Maintainability: The effort required to ~~correct~~ detect and correct an error during maintenance.

b) Flexibility: The effort needed to improve an operational software program.

c) Testability: The effort required to verify software to ensure it meets the specified requirement.

6) Product Transition: Product Transition incule 3 software quality factors.

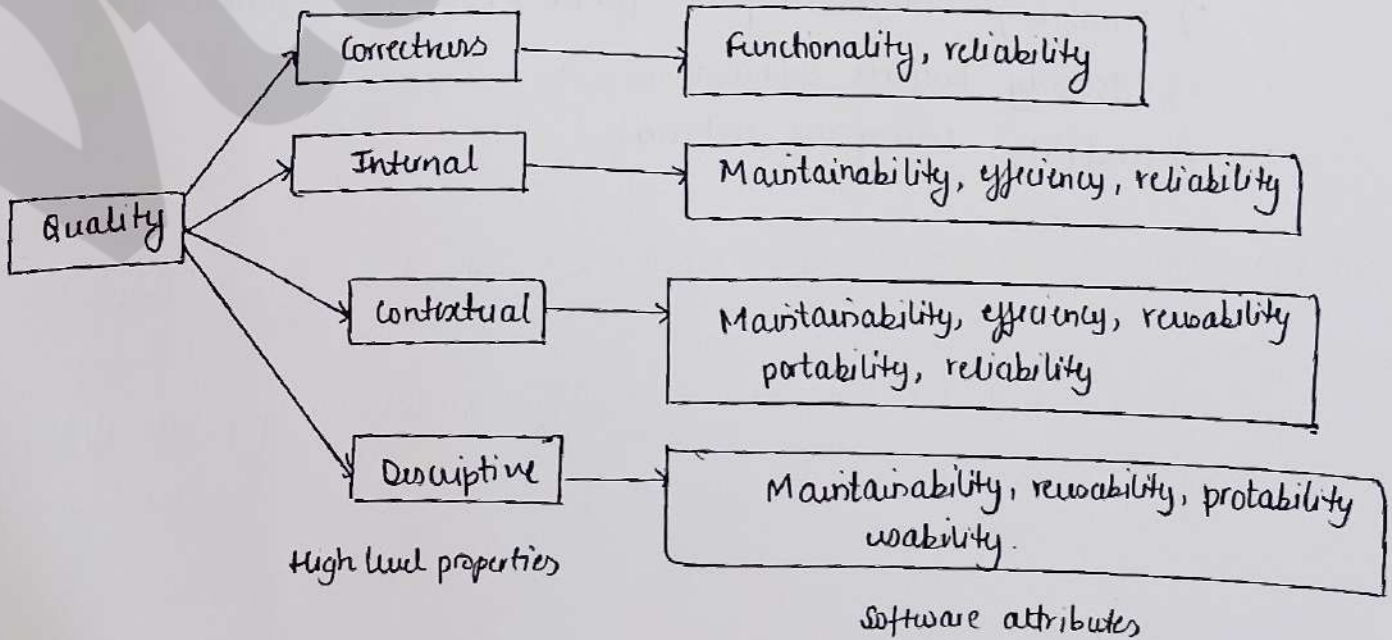
a) Portability: The effort required to transfer a program from one platform to another

b) Reusability: The extend to which the program code can be reused in other application

c) Interoperability: The effort required to integrate two system.

(5) Explain ~~~~ dromey's quality model.

→



1) Dromey proposed that software product quality depend on four major high-level properties of software

2) Correctness, internal characteristics, contractual characteristics, and descriptive properties.

3) Each of these high-level properties of a software product, is turn depend on several lower level quality attributes.

4) Correctness refers to whether the software component behaves as expected fulfilling functional requirements

5) Internal properties focuses on the internal structure of component such as consistency and maintainability of the code.

6) Contractual properties ensure that the component behave correctly in the system context, interacting properly with other component.

7) Descriptive properties: addresses the external representation of component such as usability and maintainability.

8) Software attributes:

1) Functionality: The ability to provide expected future
2) Reliability: Consistent and fault tolerant operation
3) Maintainability: Ease of making changes or updates
4) Reusability: Ability to reuse components in different system.
5) Portability: Adaptability to different platforms or environments
6) Efficiency: Resource optimization
7) Usability: Ease of use end users.

⑥ Why do we need software quality models? Explain Garvin's quality dimension.

→ The quality models give a characterization of software quality interms of a set of characteristics of the software. The bottom level of the hierarchial can be directly measured, thereby enabling a quantitative assessment of quality of software.

Garvin Quality dimensions: David Garvin, a professor of Harvard Business school, defined the quality of any product in terms of eight general attributes of product.

1) Performance: How well it performs the job
2) Features: How well it supports the required ~~function~~ features.
3) Reliability: Probability of a product working satisfactorily within a specified period of time.
4) Conformance: Degree to which product meets the requirement.
5) Durability: Measure of product life.
6) Serviceability: Speed and effectiveness maintainence.
7) Aesthetics: The look and feel of product
8) Perceived quality: User opinion about the product quality.