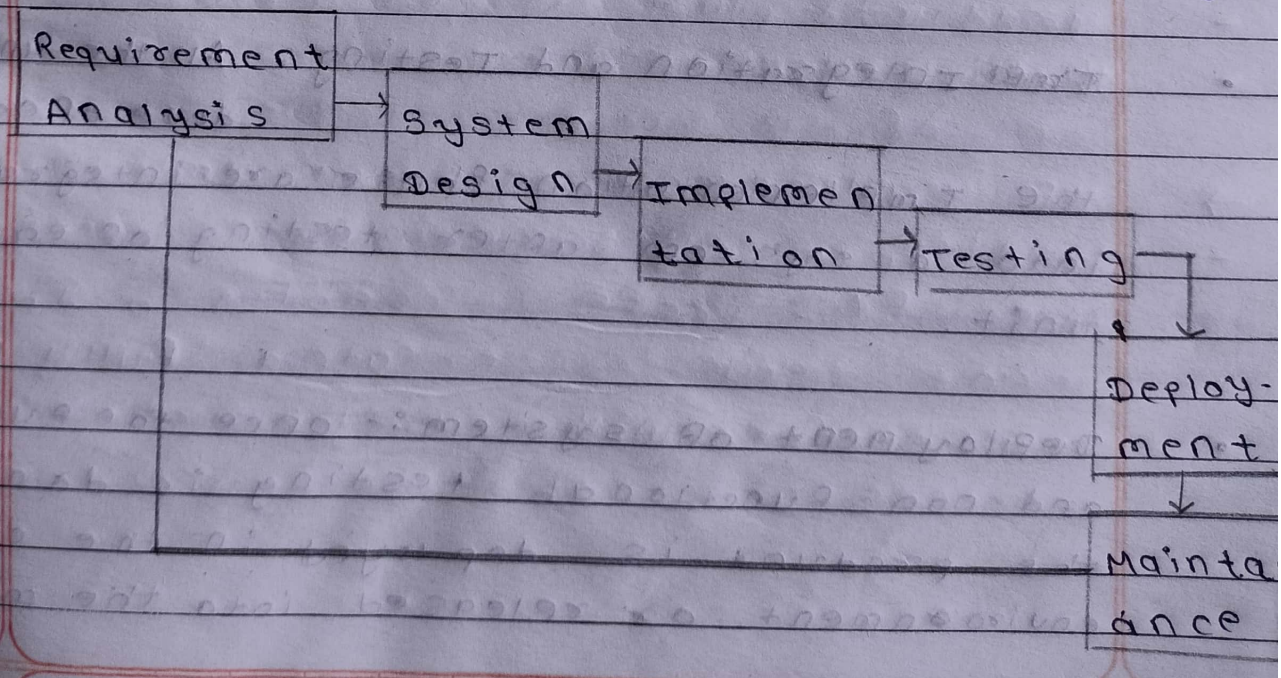
# Assigment - 1

Q-1 Explain the following in detail:-

a) Waterfall model

→ The waterfall model was the first process model to be introduced. It also referred as a linear-sequential life cycle model. It is very simple to understand and use. In waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phase.

→ In "Waterfall" approach the whole process of software development is divided into separate phase. In this waterfall model typically the outcome of one phase acts as the input for the next phase sequentially.



→ The sequential phases in Waterfall model are -

- Requirement Gathering and analysis :- All possible requirements of the system to be developed are captured in this phase and documented in a requirement specific document.
- System Design :- The requirement specifications from first phase are studied in this phase and the system design is prepared.
- Implementation :- With inputs from the system design, the system is first developed in small programs called units which are integrated in the next phase.
- Integration and Testing :- All the units developed in the Implementation phase are integrated into a system after testing of each unit.
- Deployment of system :- once the function and non-functional testing is done, the product is deployed in the customer environment or released into the market

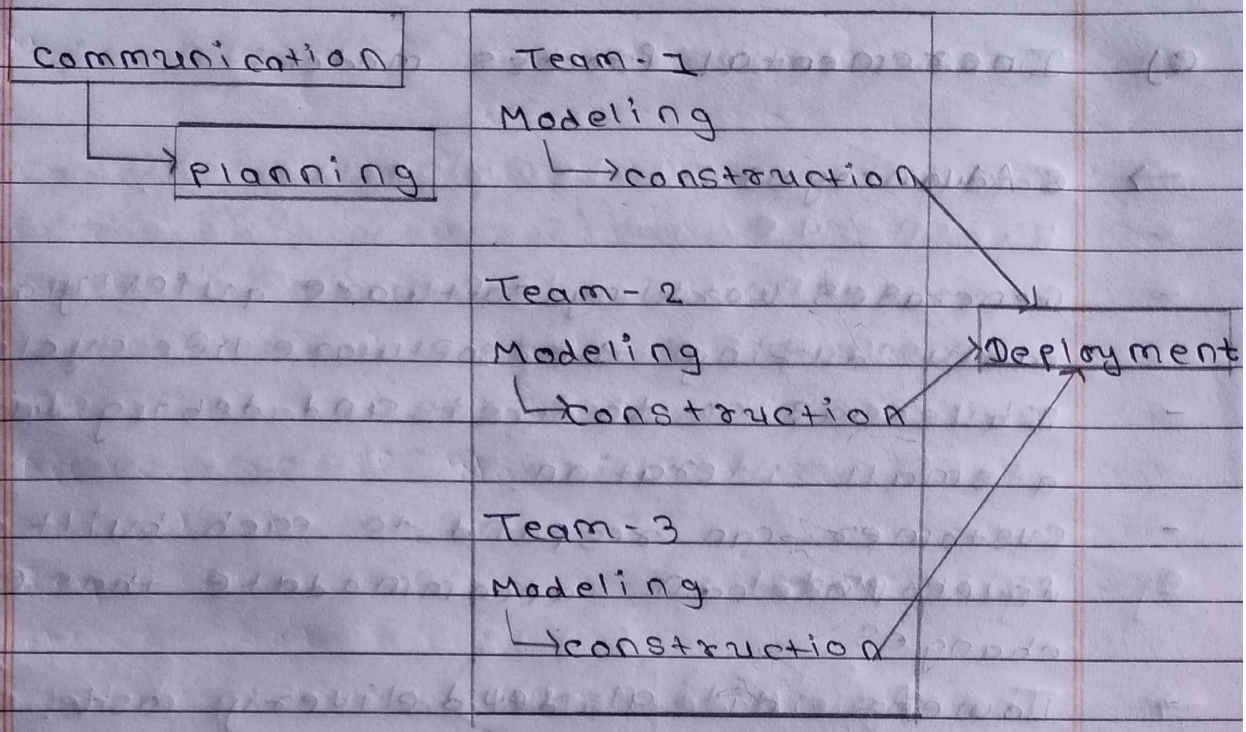
c) Incremental Process model :-

→ Advantages :-

- Generates working software quickly and early during the software life cycle.
- It is easier to test and debug during a small iteration.
- Customer can respond to each built and developer can accommodate those changes.
- Lowers initial dev delivery cost.
- Easier to manage risk, because risky pieces are identified and handled during iterations.
- Works with small sized team.

→ Disadvantages :-

- Needs good planning and design.
- Needs a clear and complete defining definition of the whole system before it can be broken down and built incrementally.
- Problems might cause due to system architecture as such not all requirements collected up front for the entire software life cycle.




The RAD Model is a "high-speed" adaptation of the linear sequential model in which rapid development is achieved by using component-based construction.

→ Requirement Planning :- It involves the use of various techniques used in requirements elicitation.

→ User Description :- This phase consists of taking user feedback and building the prototype using developer tools.

→ Disadvantages of Waterfall model :-

- High amounts of risk and uncertainty
- Poor model for long and ongoing projects.
- It is difficult to measure progress within stages.
- Adjusting scope during the lifecycle can end a project.

b) RAD Model

→ It is based on prototyping and iterative development with minimal planning involved.

→ The functional modules are developed in parallel as prototypes and are integrated to make the complete product for faster product delivery.

→ If requirements are well understood and project scope is constrained the RAD process enables a development team to create a "fully functional system" within very short time periods (e.g., 60 to 90 days)

- Maintenance :- There are some issues which come up in the client environment, to fix those issue, patches are released, maintenance is done to deliver these changes in the customer environment.

→ Application of Waterfall model :-

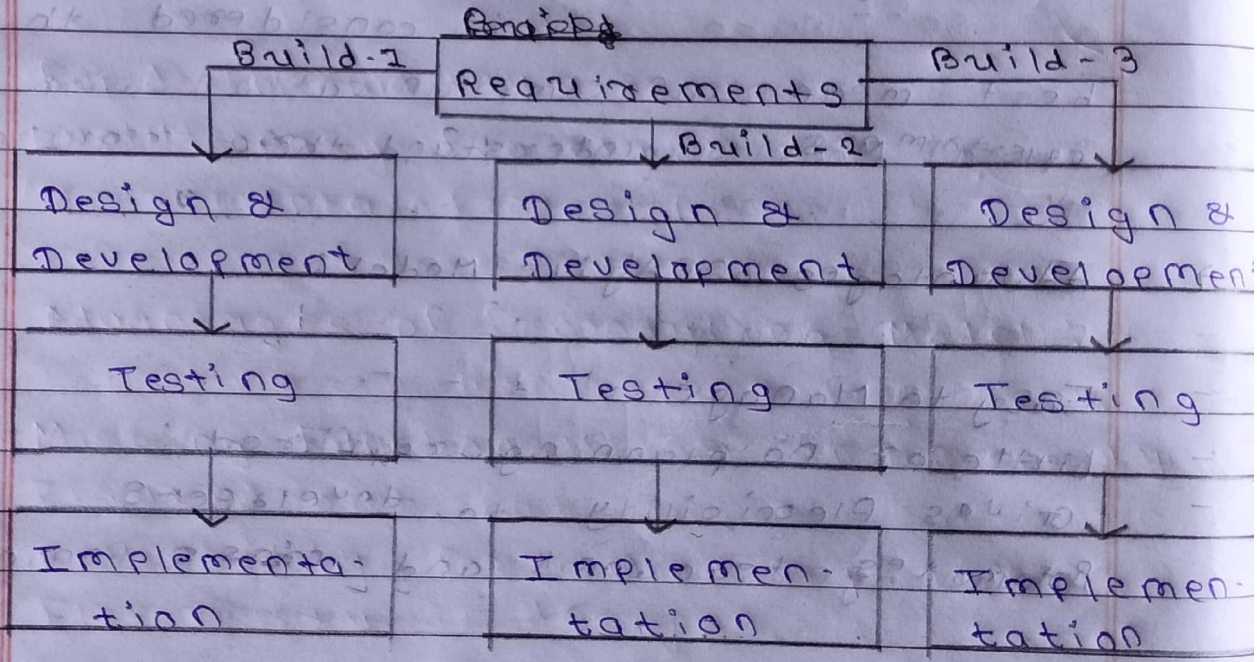
- Requirements are very well documented, clear and fixed.
- Product defination is stable.
- Technology is understood and is not dynamic
- There are no ambiguous requirements.
- Ample resources with required expertise are available to support the product.
- The project is short.

→ Advantages of waterfall model :-

- Easy to arrange tasks
- Process and results are well documented
- Well understood milestones
- Clearly defined stages
- Phases are processed and completed one at a time.

f) Iterative Model

→ In the Iterative model iterative process starts with a simple implementation of a small set of the software requirements and iteratively enhances the evolving versions until the complete system is implemented and ready to be deployed.




→ Advantages of the Iterative Model.

- Results are obtained early and periodically
- Parallel development can be planned
- Progress can be measured.
- Less costly to change the scope requirements
- Testing and debugging during smaller iteration is easy.

→ Following are the Agile Manifesto principles -

- Individuals and Interactions - In Agile development, self-organization and motivation are important, as are interactions like co-location and pair programming

- Working Software - Demo working software is considered the best means of communication with the customers to understand their requirements

→ Advantages of Agile Model :-

- Easy to Manage
- Little or no planning required
- Gives flexibility to developers
- Promotes teamwork and cross training

→ disadvantages of Agile Model :-

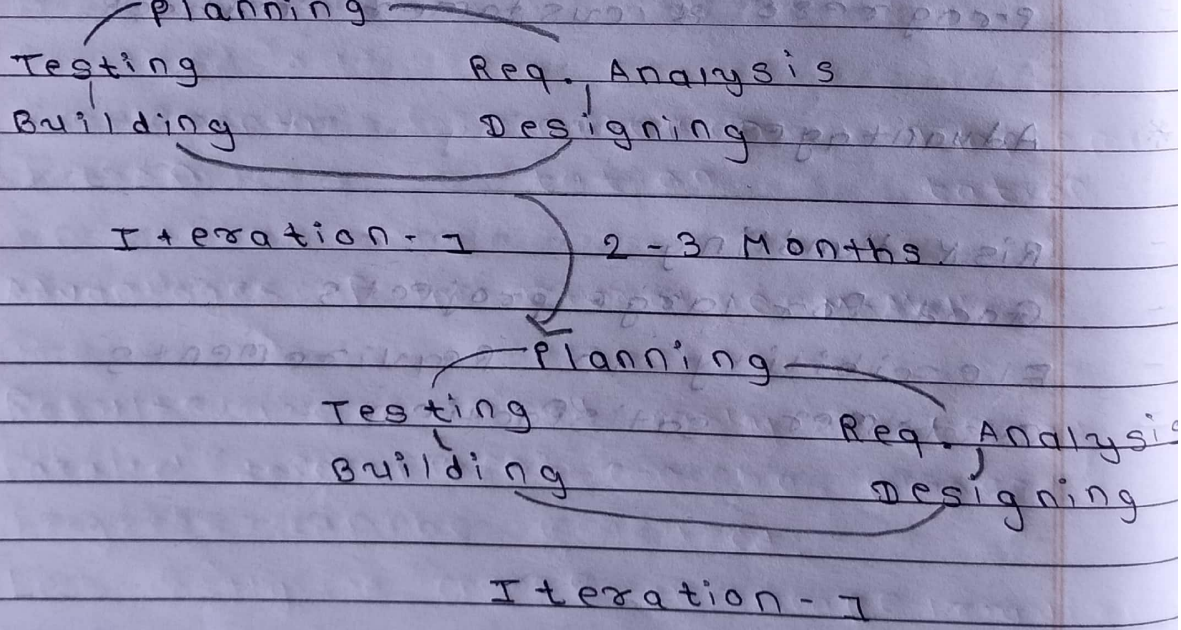
- Not suitable for handling complex dependencies.
- More Risk of sustainability, maintainability and extensibility.

* Disadvantages :-

- Complex as it is risk driven.
- Expensive
- Too much dependable on Risk analysis
- Difficulty in time management.
- Requires knowledgeable and experienced staff.

ej Agile Model :-

→ The Agile Model was primarily developed that every project needs to be handled differently and the existing methods needs to be tailored to best suit the project requirements.



d) Spiral Model

→ Couples iterative nature of prototyping with the controlled and systematic aspects.
→ Using spiral, software developed in a series of evolutionary release.
→ Early iteration, release might be on paper or prototype.
→ Later iteration, more complete version of software.

* Spiral Model Phases

- Identification / Planning
- Risk Analysis
- Engineering (Build + Testing)
- Progress & customer feedback

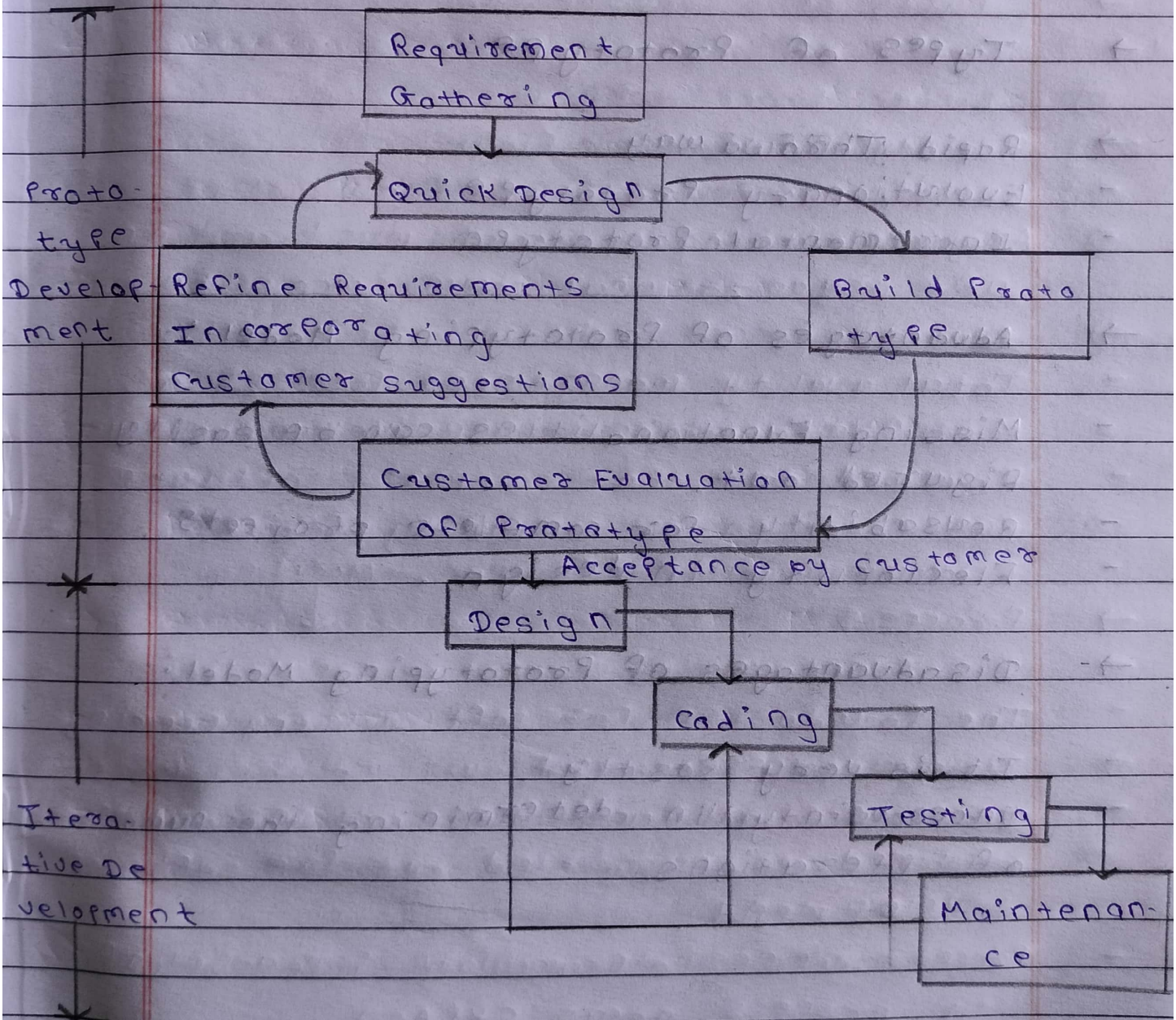
* Advantages

- Risk Handling
- Good for large projects
- Flexibility in Requirements
- Customer satisfaction

→ Disadvantages of the Iterative model :-

- More resources may be required
- More management attention is required
- Not suitable for smaller projects
- Management complexity is more

g) Prototype Model :-



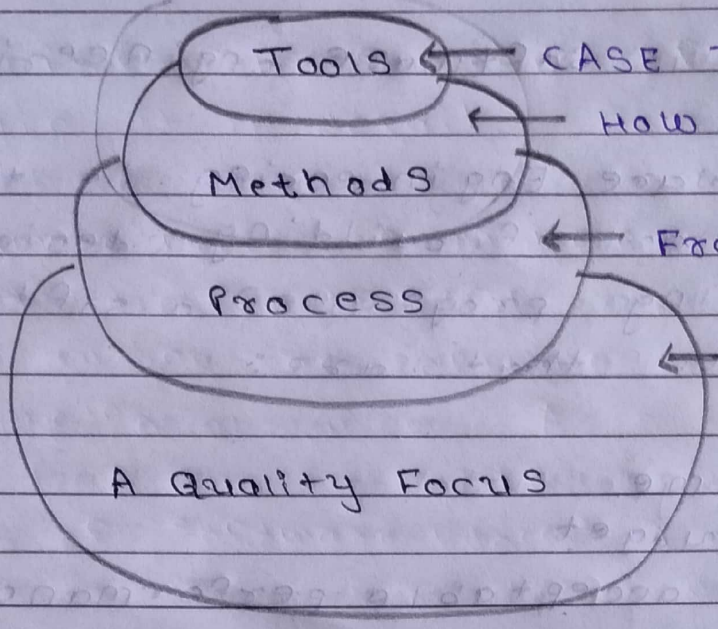
- KPA establish the context in which technical methods are applied, work products (models, documents, data, reports, forms, etc) are produced, milestones are established, quality is ensured and change is properly managed.

→ Methods :

- Software engineering methods provide the technical how-to's for building software.
- It provides the technical way to implement the software.
- Methods encompass a broad array of tasks that include requirements analysis, design, program construction, testing, and support.

→ Tools :

- Software engineering tools provide automated or semi automated support for the process and the methods.
- The tools are integrated i.e the information created by one tool can be used by the other tool.
- CASE (computer-aided software engineering) tools automate many of the activities involved in various life cycle phase.




→ Quality Focus :-

- Any engineering approach must rest on an organizational commitment to quality.
- Total quality management fosters a continuous process improvement culture.
- The bedrock that supports software engineering is a quality focus.

→ Process

- The Foundation for software engineering is the process layer.
- It covers all activities, actions and tasks required to be carried out for software development.
- Process defines a framework for a set of key process areas (KPAs)

2) What is Software Engineering?

→ Software Engineering is the science and art of building (designing and writing programs) a software system that are:-

i) on time.
ii) on budget
iii) with acceptable performance
iv) with correct operation.

3) Explain Software characteristics?

→ There are several characteristics of Software are as follows :-

- Software is developed or engineered, it is not manufactured.
- Software doesn't "wear out"
- The industry is moving towards component-based assembly, most Software continues to be custom built.

4) Explain software as a layered technology.

→

→ The Prototyping Model

- Requirement Gathering
- Quick Design
- Build a Prototype
- Initial User Evaluation
- Refining Prototype
- Implement Product and Maintain

→ Types of Prototype

- Rapid Throwaway
- Evolutionary Prototype
- Incremental Prototype

→ Advantages of Prototyping Model :-

- Missing Functionalities can be easily figured out.
- Reusability for complex projects.
- Flexibility in design.

→ Disadvantages of Prototyping Model :-

- It is very costly
- Uncertainty in determining the number of iterations

- Reusability Management: It defined criteria for work product reuse (including software components) and establishes mechanisms to achieve reusable components.

- Measurement :- Defines and collects process, project and product measures that assist the team in delivering software that meets stakeholders' needs.

- Risk Management :- Evaluates risk that many affect the outcome of the project or the quality of the product.

7) What is Process ? Differentiate Process and Product

→ A process refers to a set of defined activities, methods, and practices followed to achieve a particular objective

| Process | Product |
|---|---|
| - A series of steps followed to achieve a goal. | The outcome or result of a completed process |
| Emphasizes how something is done | Emphasizes what is delivered to the customer |

- Software Project tracking and control : It allows the software team to assess progress against the project plan and take any necessary action to maintain the schedule.

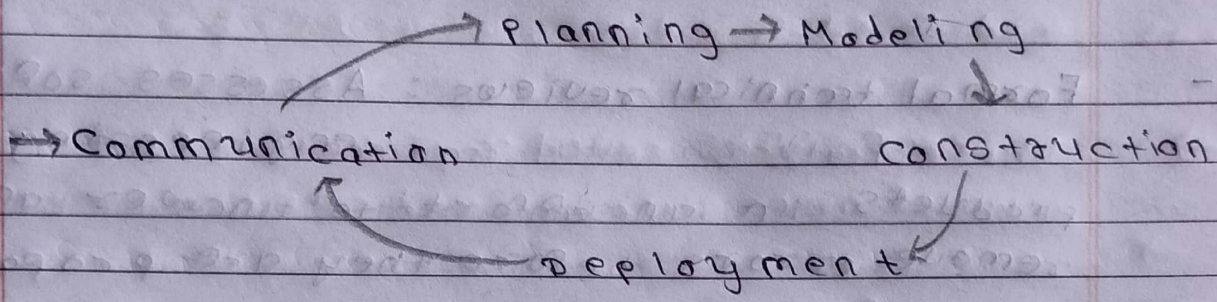
- Formal technical reviews : Assesses software engineering work products in an effort to uncover and remove errors before they are propagated to the next activity.

- Software quality assurance : Defines and conducts the activities required to ensure software quality.

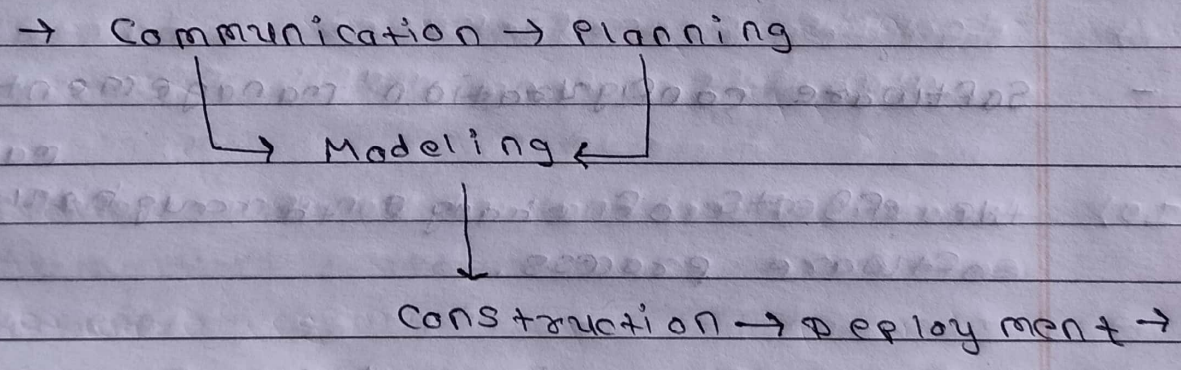
- Software configuration management : It manages the effects of change throughout the software process.

- Document Preparation and Production : It encompasses (includes) the activities required to create work products such as models, documents logs, forms and lists.

5) → Evolutionary Process Flow :- In this model, the software is developed in small increments with each increment adding new functionality.




→ Parallel Process Flow :- In a Parallel Process flow, multiple activities are carried out simultaneously to speed up development.




6) Explain Generic and Umbrella Activities

→ Umbrella activities applied throughout the software project & help a software team to manage and control progress, quality, change & risks

5) What is Software Process Flow? Explain it in detail.

→ A Software Process Flow defines the sequence of activities, tasks and steps that take place during the software development life cycle.

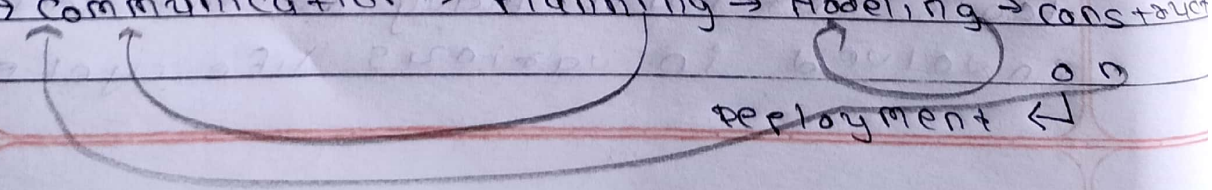
→ Types of Software Process Flows

1) Linear Process Flow
2) Iterative Process Flow
3) Evolutionary Process Flow
4) Parallel Process Flow

→ Linear Process Flow :- In this Flow, each phase is completed sequentially before moving to the next phase.

→ Communication → Planning → Modeling
↓
← Deployment ← Construction

→ Iterative Process Flow :- In this model the development proces is divided into smaller cycles.



8) What is Requirement Engineering?

→ Requirements analysis, also called requirements engineering is the process of determing user expectations for a new or modified product.

→ These features, called requirements, must be quantifiable, relevant and detailed.

→ In software engineering, such requirements are often called functional specifications.

9) List out various non-functional requirements.

→ Non-functional requirements of a system include performance, maintainability scalability among many others.

10) What is Functional and Non-Functional Requirements?

→ A Functional requirement specifies something that the application or system should do.

→ often this is defined as a behaviour of the system that takes input and provides output.

→ Non-functional requirements sometimes also called quality requirements describe how the system should be as opposed to what it should do.

11) Explain SRS in detail.

→ The Requirements are specified in specific format known as SRS.
→ This document is created before starting the development work.
→ The software requirement specification is an official document.

* Goals of SRS :-

- Facilitating Reviews
- Describing the scope of work.
- Providing a reference to software designers
- Including features to customer requirements.

12) Explain requirement elicitation with its different method.

→ Requirement elicitation is the practice of collecting the requirements of a system from users, customers and other stacke holders. this practice is also sometimes referred to as

## Requirement Gathering.

→ Requirements elicitation is a part of the requirements engineering process, usually followed by analysis and specification of the requirements.

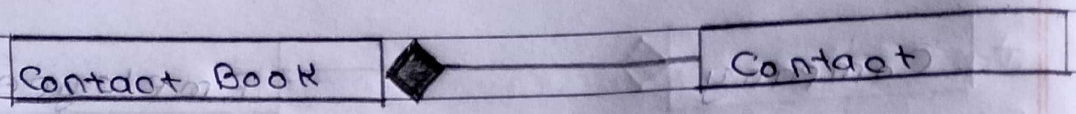
→ Requirements elicitation practices include interviews, questionnaires, user observation workshops, brainstorming, use cases, role playing and prototyping.

→ Before requirements can be analyzed, modeled or specified they must be gathered through an elicitation process.

→ Requirements elicitation is perhaps the most difficult, most error-prone and most communication intensive software development.

→ It can be successful only through an effective customer-developer partnership. It is needed to know what the users really need.

→ There are a number of requirements elicitation methods. Few of them are listed below-




15) Explain class diagram with example.

→ To describe a complete aspect of the system it is suggested to give a meaningful name to the class diagram.

→ The objects and their relationships should be acknowledged in advance.

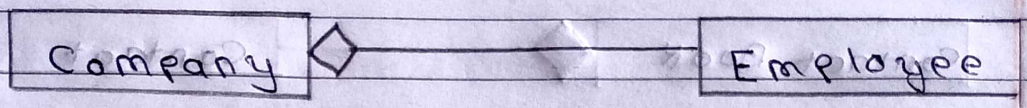
→ The attributes and methods (responsibilities) of each class must be known.

→ A minimum number of desired properties should be specified as more number of the unwanted property will lead to a complex diagram.

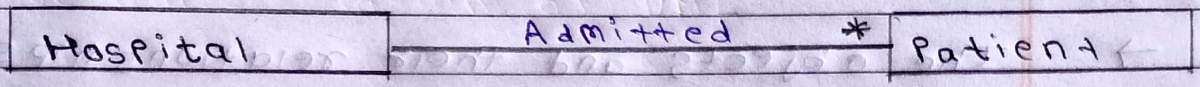
→ Notes can be used as and when required by the developer to describe the aspects of a diagram.

→ The diagram should be redrawn and reworked as many times to make it correct before producing its final version.

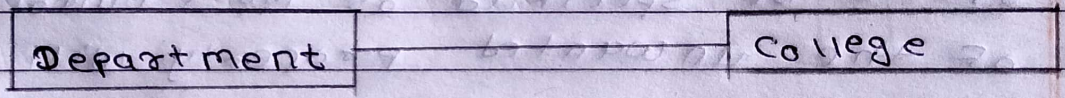
→ A class diagram describing the sales order system is given below.



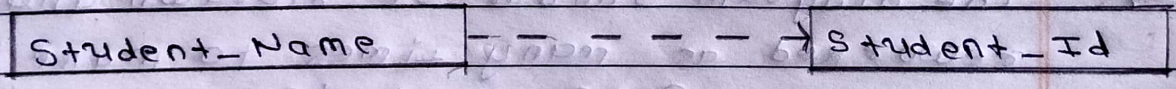

c) Multiplicity :- It defines a specific range of allowable instances of attributes. In case if a range is not specified, one is considered as a default multiplicity.




d) Association :- It describes a static or physical connection between two or more objects. It depicts how many objects are there in the relationship.




e) Dependency :- A dependency is a semantic relationship between two or more classes where a change in one class cause changes in another class.



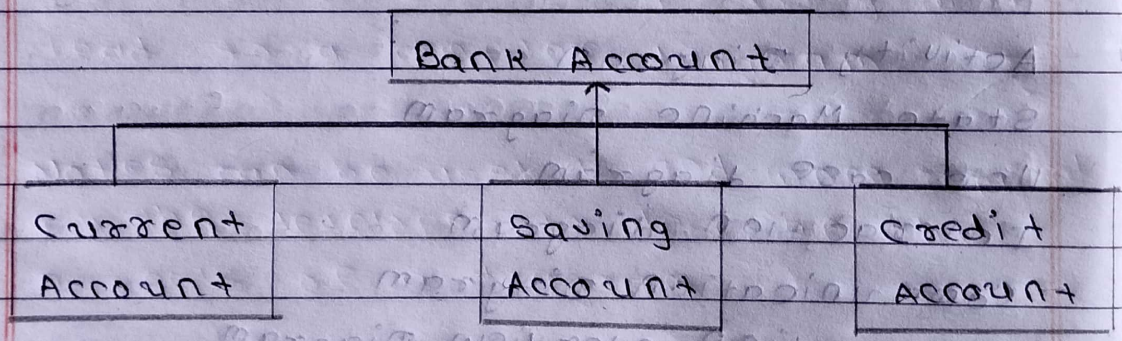

f) Composition :- The composition is a subset of aggregation. It portrays the dependency between the parent and its child which means if one part is deleted then the other page also gets discarded.

→ Object Diagram
→ Composite structure diagram
→ Deployment diagram
→ Package Diagram

14) Explain following relationship:

a) Generalization :- A Generalization is a relationship between a parent class (superclass) and a child class (subclass). In this, the child class is inherited from the parent class.

→ For example, The Current Account, Saving Account, and Credit Account are the generalized form of bank account.




b) Aggregation :- An aggregation is a subset of association, which represents has a relation. It is more specific then association. It defines a part-whole or part-of relationship

1. Interviews
2. Brainstorming sessions
3. Facilitated Application specification Technique (FAST)
4. Quality Function Deployment (QFD)
5. Use case Approach.

13) Give classification of UML

→ The UML diagrams are categorized into structural diagrams, behavioral diagrams, and also interaction overview diagrams. The diagrams are hierarchically classified in the following figure:

1) Behaviour Diagram

→ Activity Diagram
→ State Machine Diagram
→ Use case diagram
→ Interaction diagram
- Communication Diagram
- Interaction overview Diagram
- sequence Diagram
- Timing Diagram

2) structure Diagram

→ class Diagram
→ component Diagram