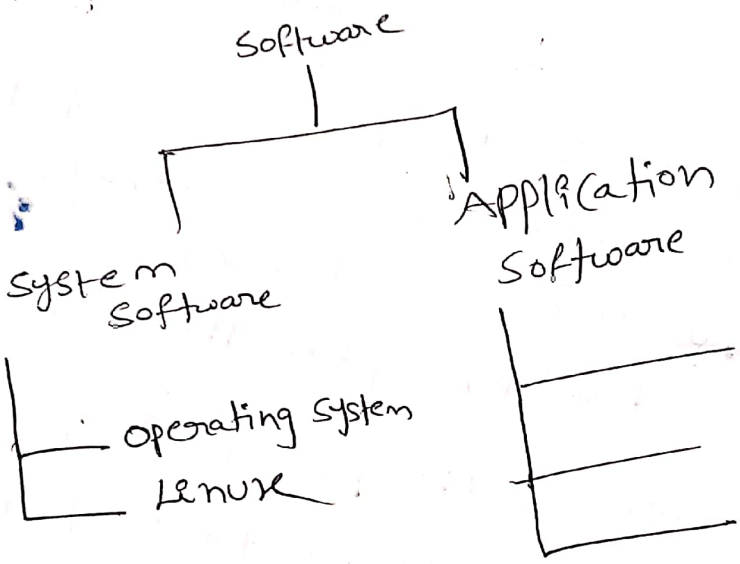
UNIT-1

Introduction to Software Engineering

Software engineering is the product of two words

Software and Engineering

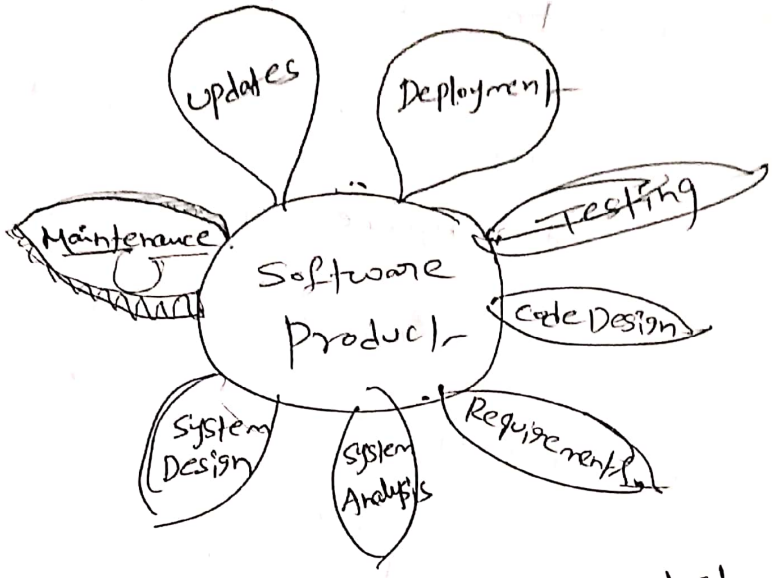
Software: is a collection of programs or set of instructions to perform a particular task



Application Software: It can perform specific task for end users

System Software: System software are mainly designed for system resources

Engineering: is the application of scientific and practical knowledge to invent, design, build, maintain and improve frameworks, processes, etc.



Updates: It is also known as patch
A set of changes to a computer program
patches usually called as bugfixes
Software developer update the software updates

Maintenance: is the process of changing, modifying and updating software to keep up with customer needs.

System Design: is the process of defining elements of a system like modules, architecture, components and their interfaces.

System Analysis: Identify the problems

Requirements: Convey the expectations of users from the software product

Code Design : To translate the design of a system into a computer language format (2)

Testing : is the process of executing the a program to find errors.

Deployment : is the process of delivering completed Software to the client who order it

Evolving role of Software

The process of developing a software product using software engineering principles and methods is referred to as software evolution

Software's Dual role

→ Software as a product
Transforms information - produces, manages, modify displays or transmits information

→ Software as a vehicle for delivering a product
- Controls other programs (operating system)
- Effects Communications (networking software)
- Helps build other software (Software tools & environments)

Software : is a collection of programs and set of instructions

Engineering : Application of science, tools & Methods to find cost effective solution to problems.

Software Engineering : It is defined as systematic disciplined & quantifiable approach for the development, operation & maintainance of software

Characteristics of Software:

- Software is developed or engineered it is not manufactured
- Software does not ware out
- Software is custom built

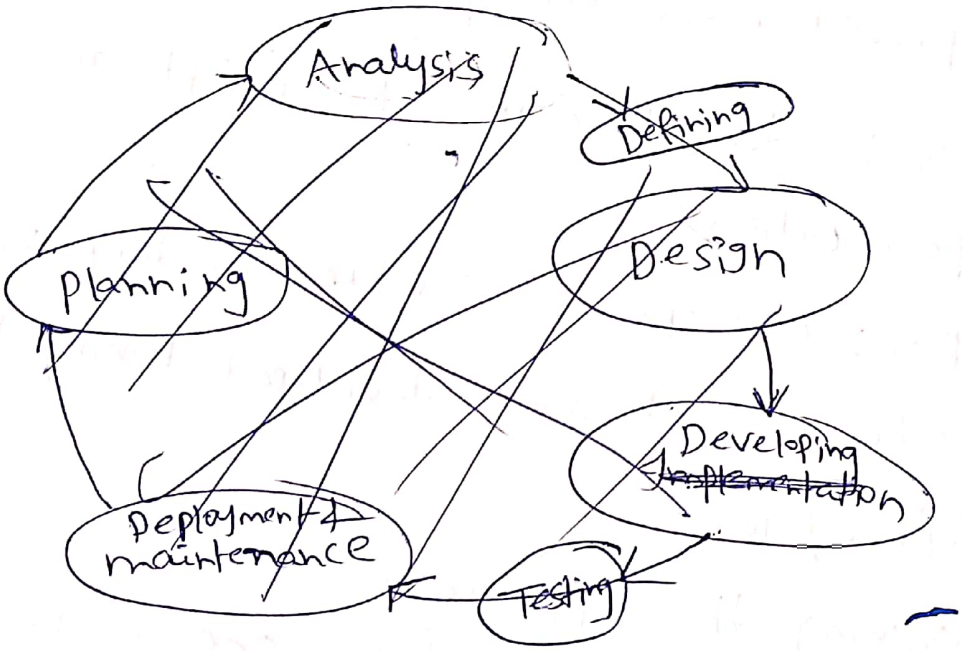
SDLC (Software Development Life Cycle) (4)

SDLC is a process used by software industry to design, develop and test software

→ SDLC produce High-quality software that meets customer expectation, reaches completion within times and cost estimates

→ SDLC is the key concept of the Software Engineering

→ It generally consists of series of stages



① Requirement Analysis : is the most important and fundamental stage in SDLC.

It is performed by the senior members of the team with inputs from all the stakeholders and Domain experts.

* Business requirements, Stakeholder requirements, solution requirements, Functional and Non functional requirements

Planning : planning for the quality assurance requirements and identification of the risks associated with the project is also done at this stage

2. Defining Requirements

once the requirement Analysis done ~~of~~ the next step is clearly define document the software requirements and get them approved from the project stakeholders

This is done through SRS (Software requirement specification)

SRS : consists of all the product requirements to be designed and developed during the project life cycle.

③ Designing the software :

It can be used more than one design approach for the product this is called DDS

Design Document specification

DPS is reviewed by all stakeholders, design modularity, budget and time

4. Developing the Software

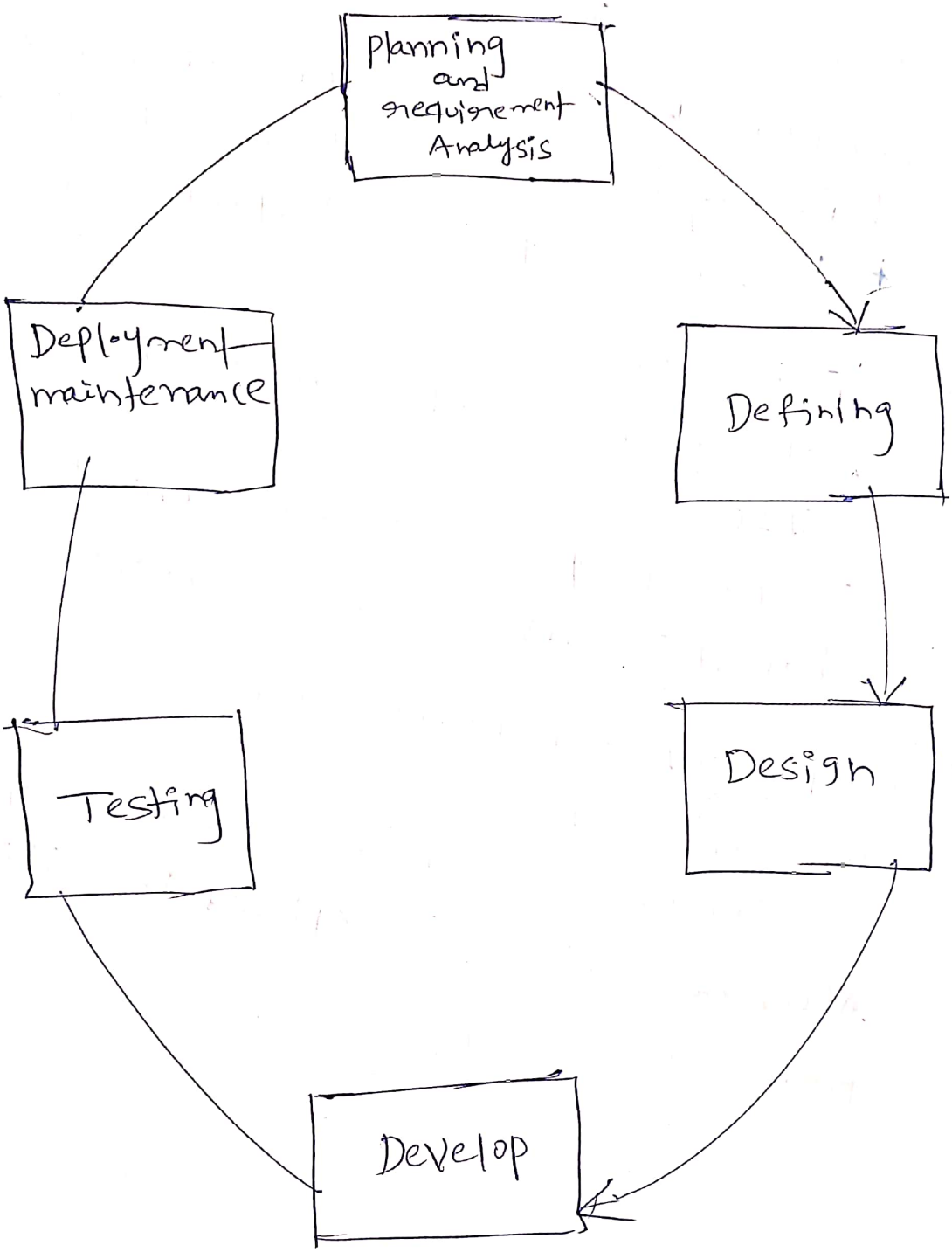
In this stage of SDLC the actual development starts and the product is built.
Programming Code is generated by DDS
Developers have to follow the Coding guidelines defined by their organization and programming tools like Compilers, interpretes, debuggers

5. Testing the Software :

This phase are used in all the phases of SDLC
Testing purpose is to detect the bugs bugs ; it is also called as errors

6. Deployment and Maintenance :

once the software is tested and no bugs or errors are reported then it is Deployed
After the software is Deployed then its maintenance starts

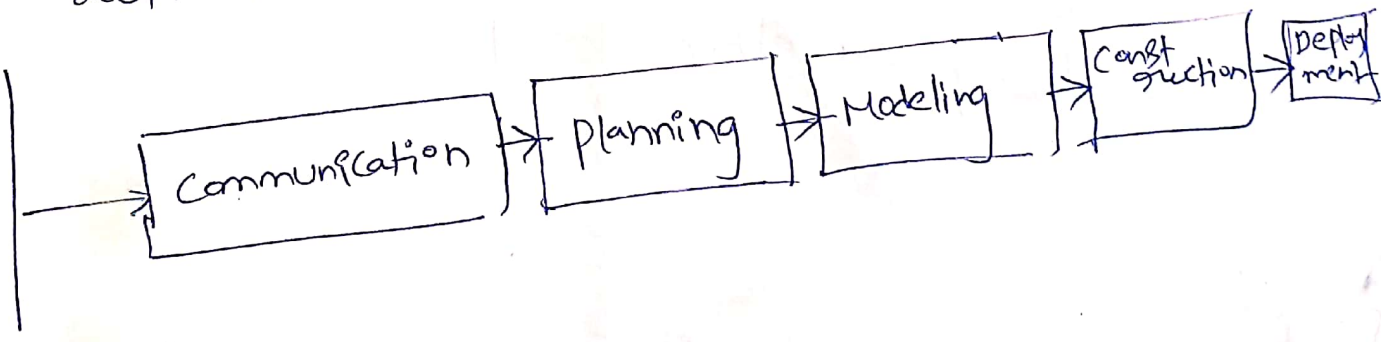


The diagram shows the layers of Software Development

Layered technology is divided into four parts

1. A quality focus: It defines the continuous process improvement principles of Software. It provides integrity that means providing security to the software so that data can be accessed by only an authorized person, no outsider can access the data. It also focuses on maintainability and usability

2. process: It is the foundation or base layers of Software engineering. Process is a key that binds all the layers together. The Software process covers all the activities actions and tasks



Communication: It is the first thing for the development Software. Communication is necessary to know the actual Demand of the client

planning: It basically means drawing a map for reduced the complication of development

Modeling: In this process, a model is created according to the client for better understanding

Construction: It includes the coding and testing of the problem

Deployment: It includes the delivery of software to the client for evaluation and feedback

(3) Method: It has the information of all tasks which include communication, requirement Analysis design modeling, program construction, testing and support

(4) Tools: Software engineering tools provide a self operating system for processes and methods. Tools are integrated which means Information created by one tool can be used by another tool

① A quality focus: It is organization standard ~~of~~ quality or commitment
The quality and commitment must be made by the Software
until and unless quality is focused ~~to~~ on process, Methods, tools
~~It~~ It include planning, Assurance control, Improvement
It is act as a Bedrock

② Process ~~&~~ is foundation layer.
It is act as Glue of Software Engineering
It provides structural support for software Engineering
It defines framework for effective and delivery
Glue between base layer and implementation

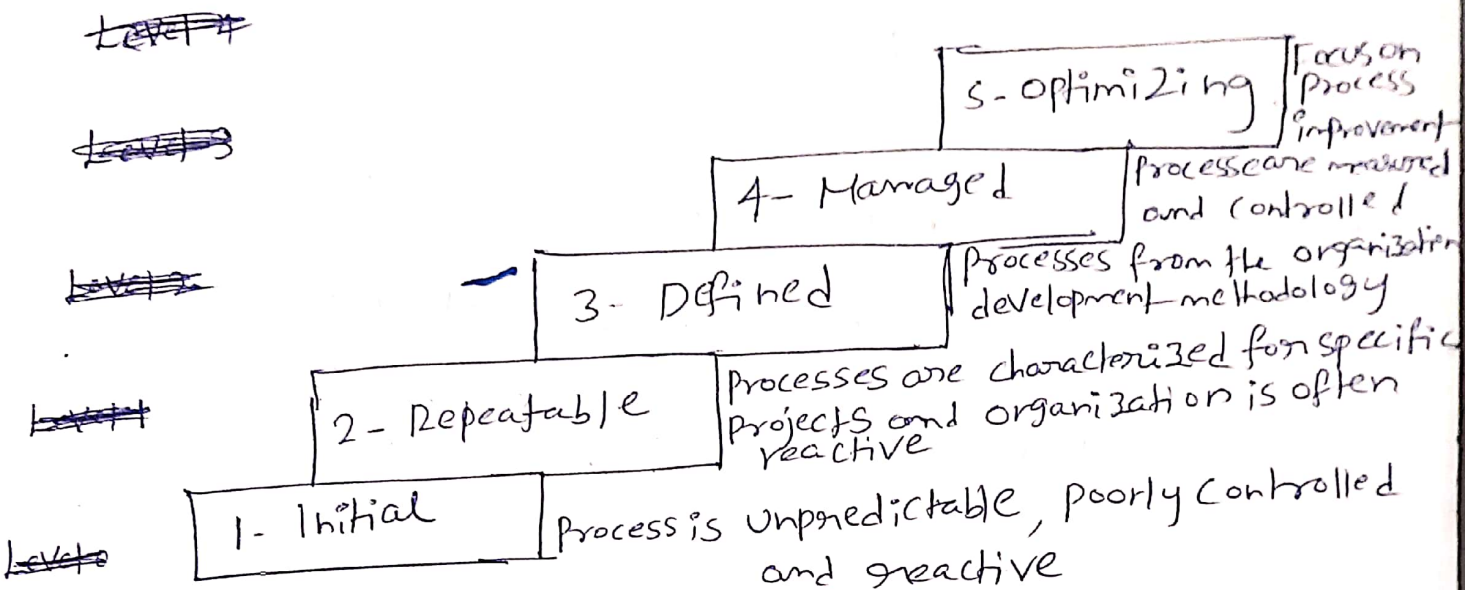
③ Methods: Methods of development of Software
It include requirement analysis, design modeling
Program construction, testing and supporting
It provide Technical questions" for building Software

④ Tools: Semi-automated, automated tools are used.

CMMI LEVELS : Capability Maturity Model Integration

(8)

- Level by Level
- To improve the quality of Software they developed



Level 1: Initial

A software development organization at this level is characterized by adhoc activities

Level 2: Repeatable

At this level, the basic project management practice such as tracking cost & schedule are established

Level 3: Defined

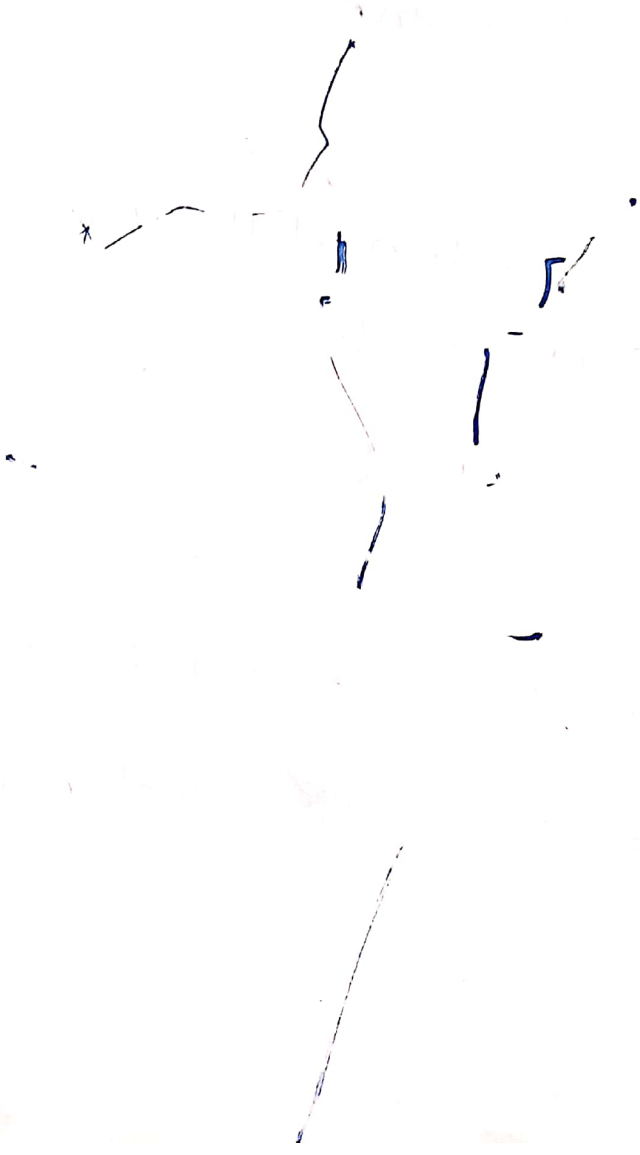
At this level, Process for both management & development activities are defined & Documented

Level 4: Managed

At this level, the focus is on software metrices

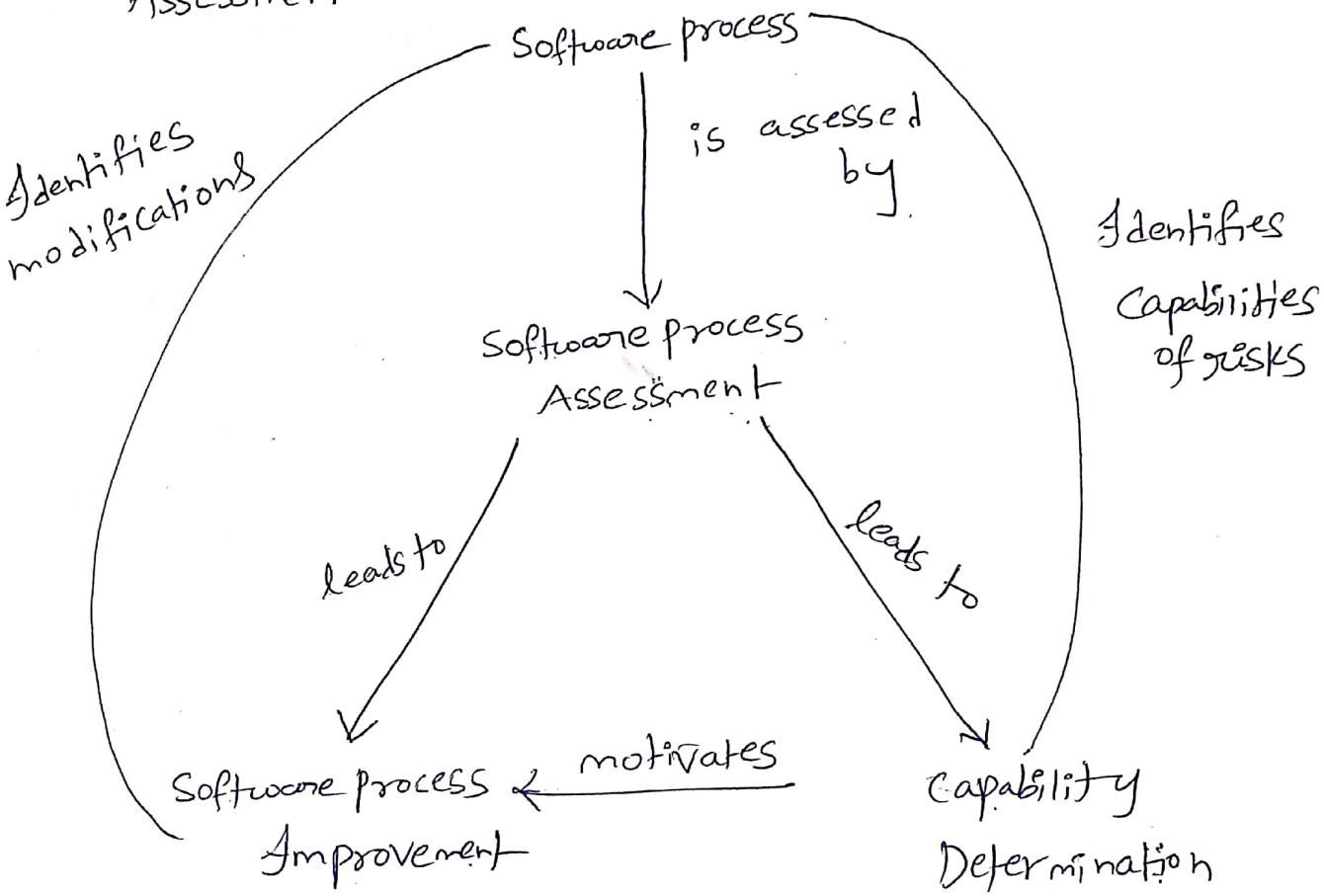
Level 5: Optimizing

At this Stage, Processes & product metrics are collected



Process Assessment

Software patterns, real time practices this entire process should be continuously accessed this process is called process Assessment.



Types of Software Assessment

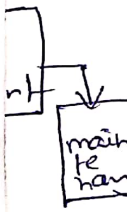
(1) Self Assessment: This is conducted internally by the people of their own organization

(2) Second party Assessment: This is Conducted by an external team

(3) Third party assessment: Software process Assessment should be performed in a transparent open, collaborative environment

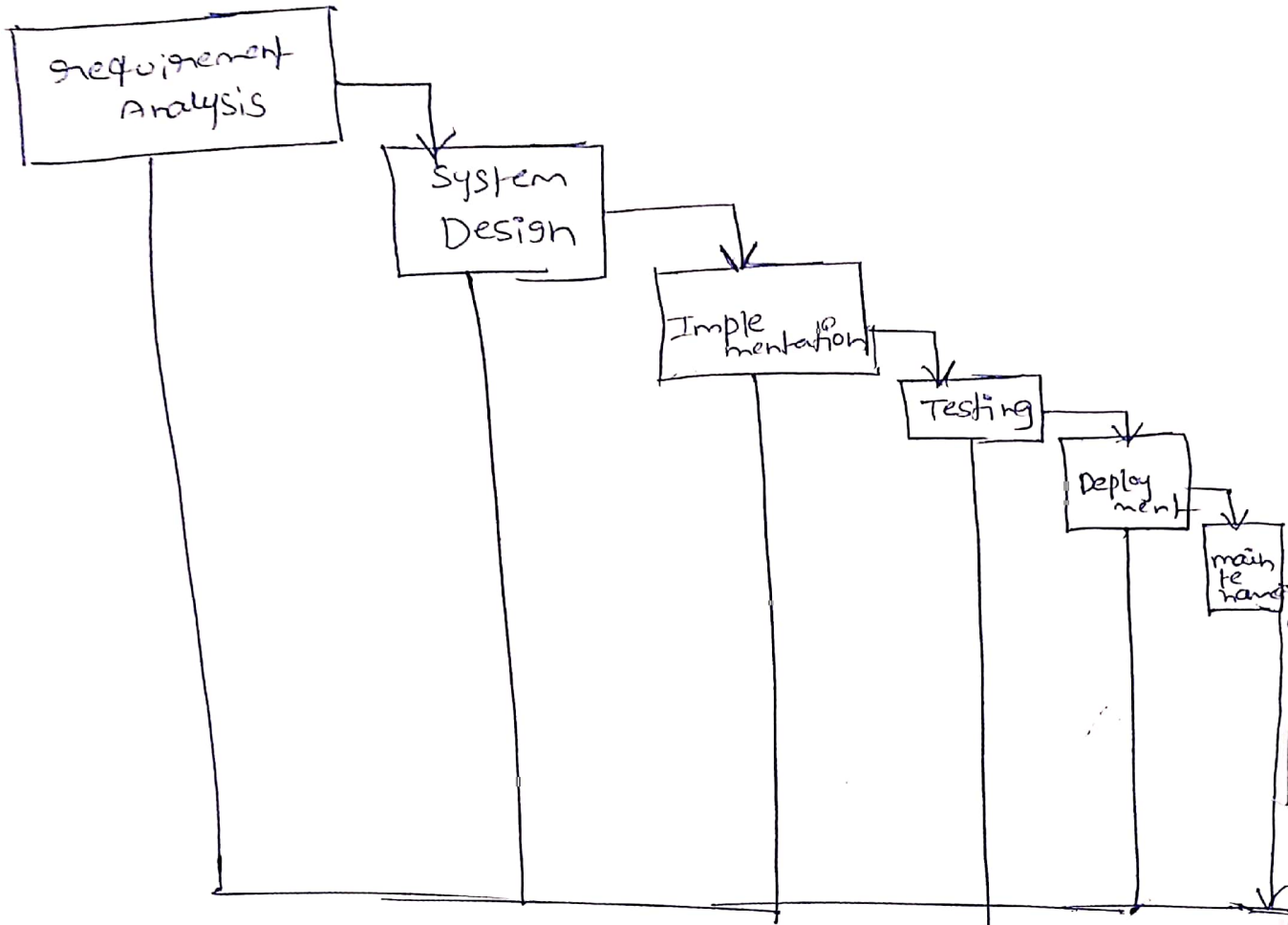
sequential

This is very important for the improvement of the Software and development of the product. The result of the Software process Assessment are confidential and are only accessible to the company The assessment team must contain at least one person from the organization that is being assessed.



le

Waterfall model : It is also called as linear sequential model or classical life cycle model (10)



It demands a systematic, sequential Approach

Requirement analysis the 1st phase of the waterfall mode It collect the exact information from the customers this information will be converted into document format

System Design : from SRS Document This phase makes a design

Implementation: is nothing but code
It divide into the small programs or small units

Testing: Integration testing
All the units are combined.

Deployment: Functional and non functional requirements supported
All modules, All units are combined

maintenance; running in the market
If any issues.

Incremented process model

Requirements are divided into multiple modules

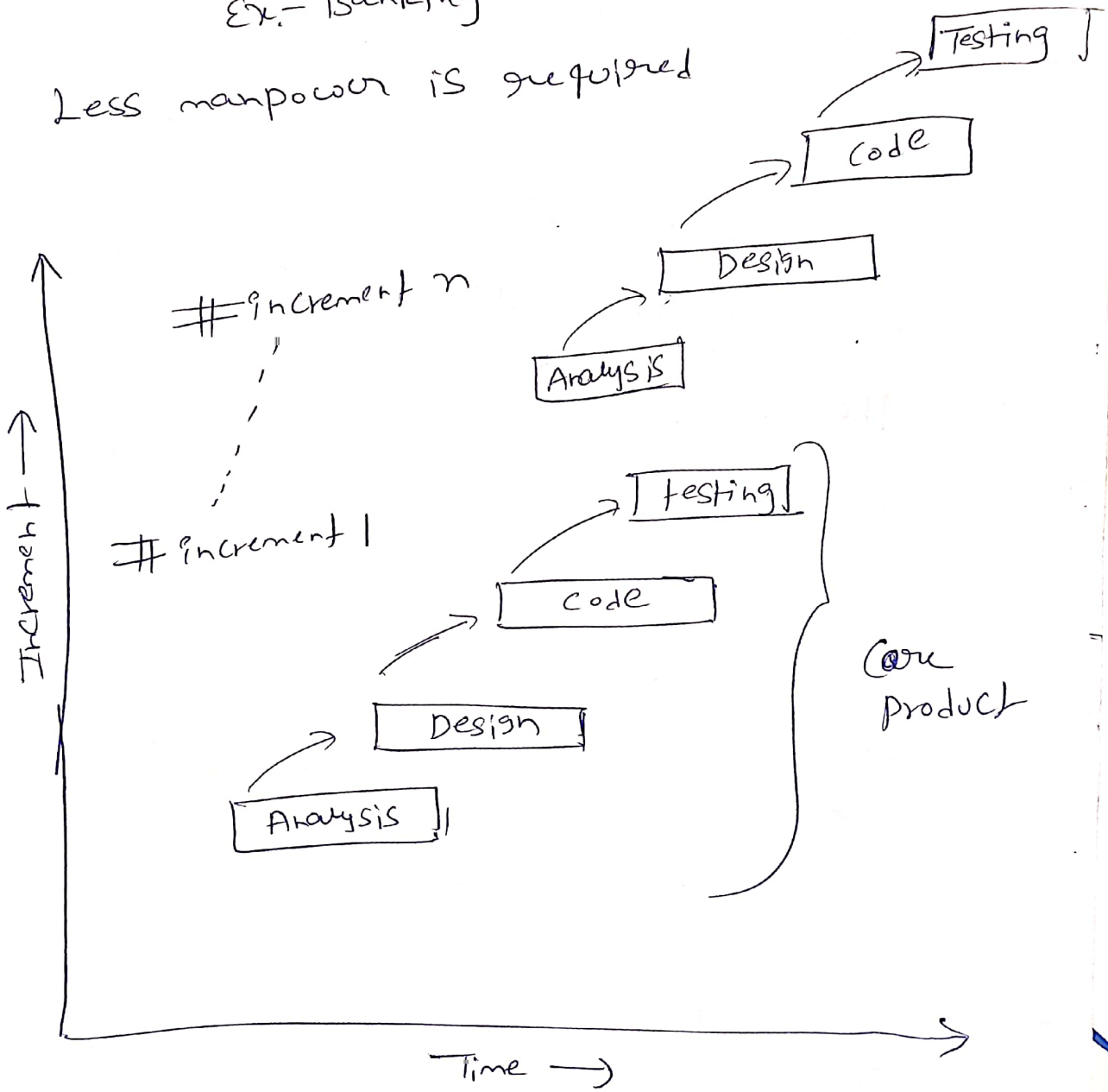
Each module → Analysis, design, code & testing

Used for software with less features

Used in day to day applications

Ex:- Banking

Less manpower is required



Core product is the Basic software

Requirement Analysis : Requirement and specification of the softwares are collected

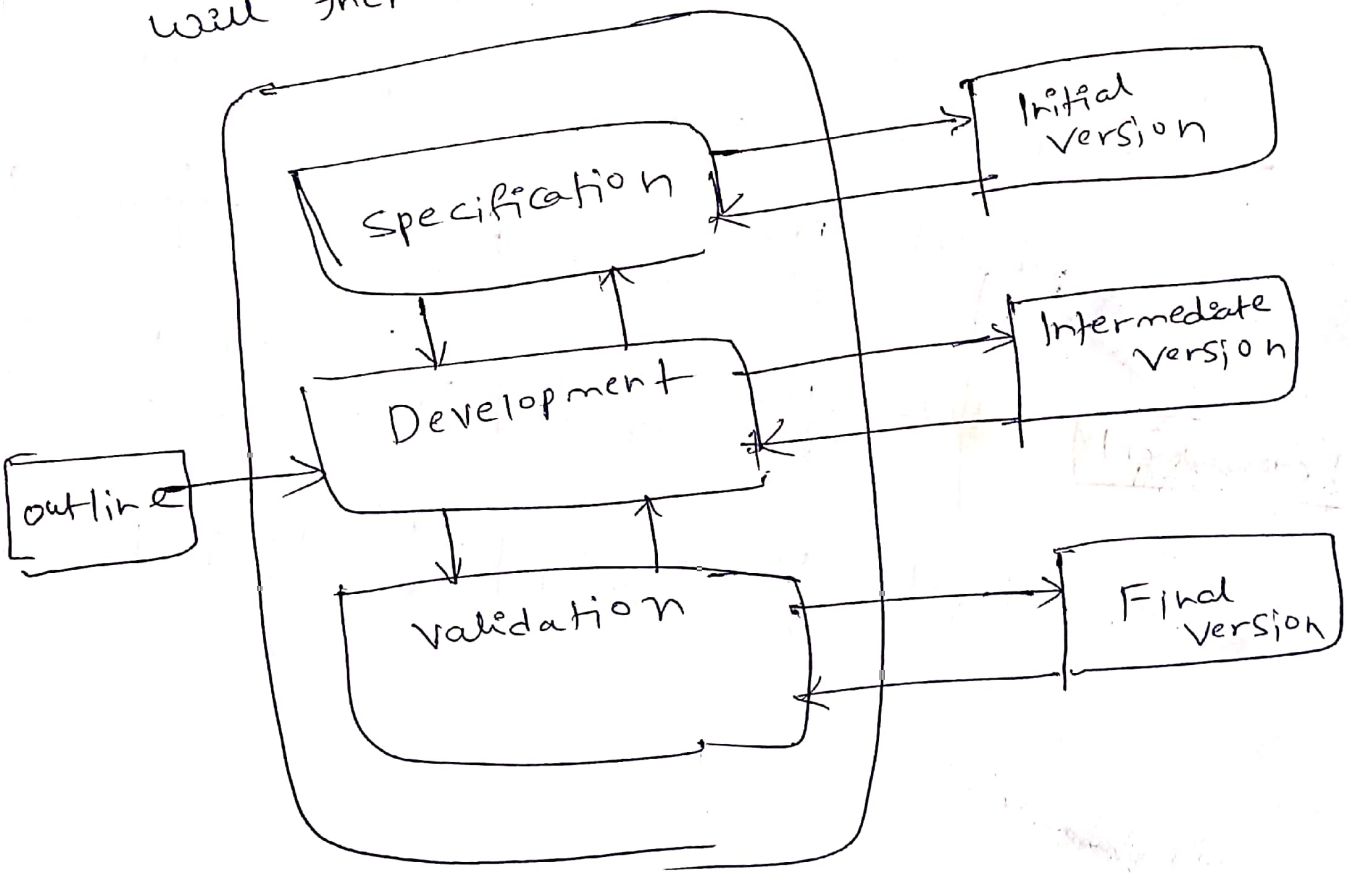
Design : Some high-end functions are designed during this stage

Code : Coding of Software is done during this stage

Test : Once the system is deployed, it goes through the testing phase

Evolutionary Process models: is a combination of Iterative and incremental model of SDLC

This model has a number of advantages such as customer involvement, taking feedback from the customer during development, and building the exact product that the user wants, Because of the multiple iterations, the chances of errors get reduced and the reliability and efficiency will increase



Evolutionary model

(2)

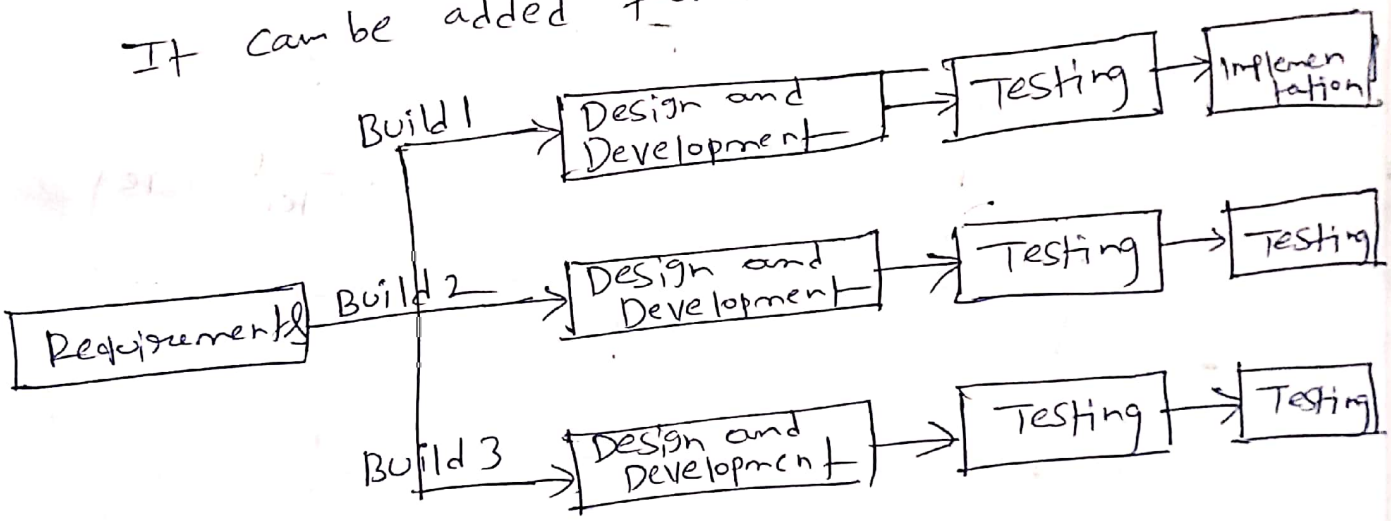
Types of Evolutionary Process models

1. Iterative model
2. Incremental model
3. Spiral model

① **Iterative model** : In the iterative model first, we take the initial requirements then we enhance the product over multiple iterations until the final product gets ready.

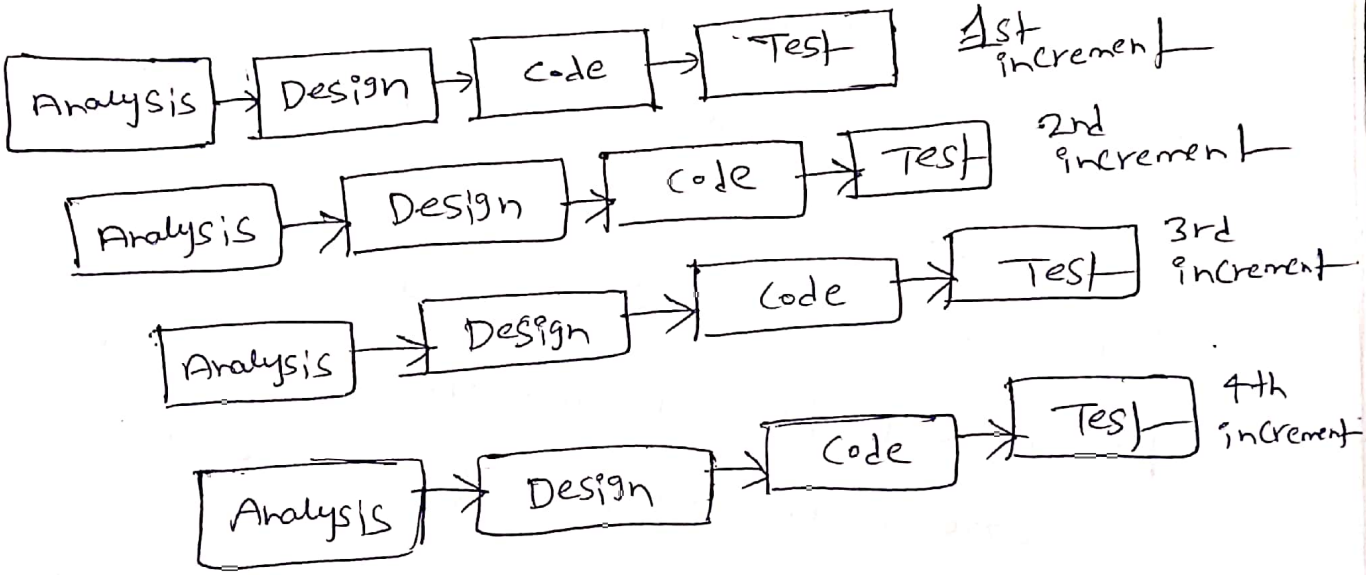
In every Iteration, Some design modifications and some changes are possible

It can be added functional requirements



(2) Incremental model :- We first build the project with basic features and then evolve the project in every iteration, It is mainly used for large projects (B)

The first step is to gather the requirements and then perform analysis, design, code and test and this process goes the same over and over again until our final project is ready



③. Spiral Model: The spiral model is a combination of waterfall and iterative model.

It is focused on risk handling along with developing the project

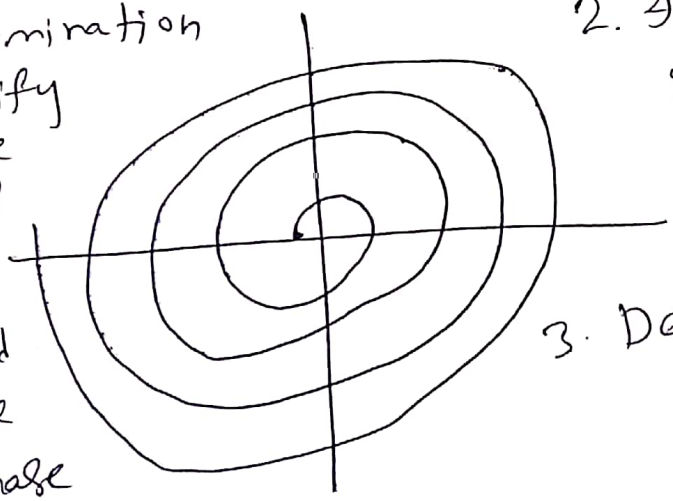
It producing the output quickly

It is good for big projects

The software is created through multiple iterations using a spiral approach

It can be used incremental and iterative Approach.

Later on, after successive development the final product will develop, and the customer iteration is there so the chances of error get reduced



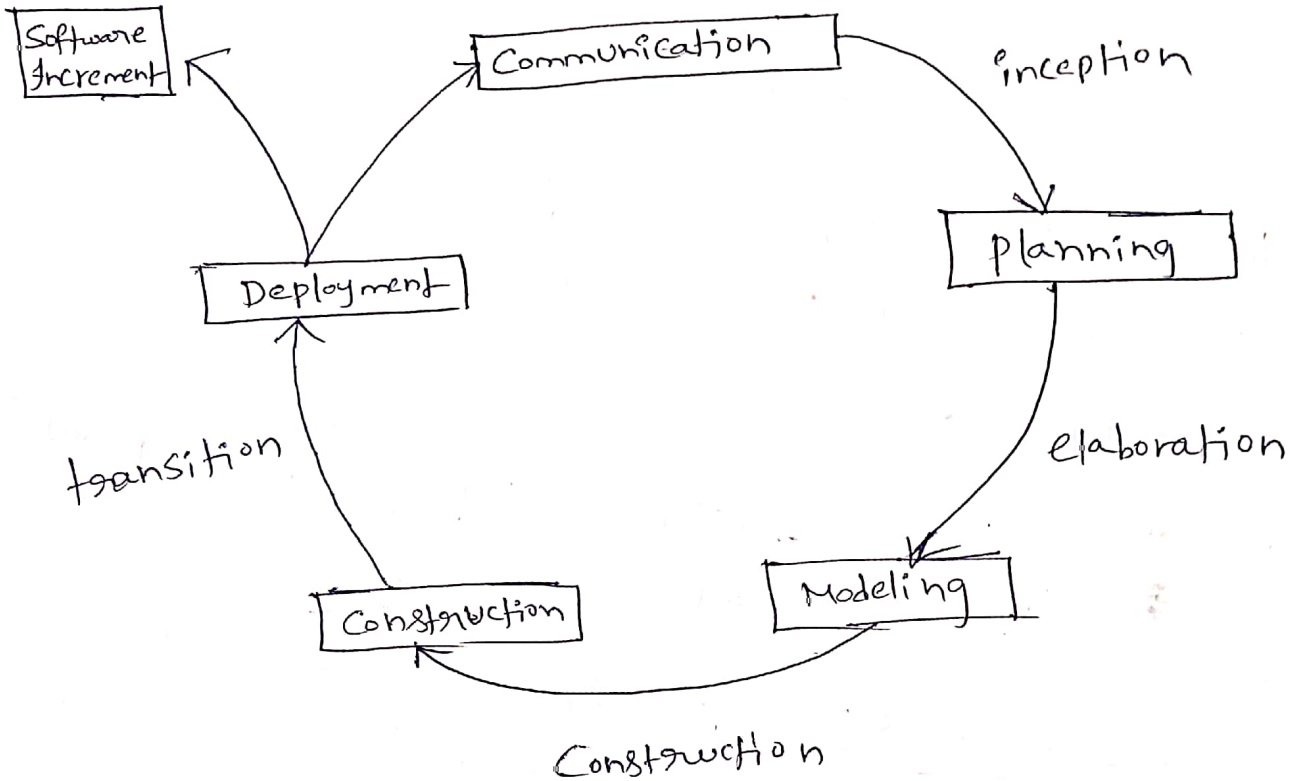
Advantages of the Evolutionary process model

1. During the development phase, the customer gives feedback regularly
2. After every iteration risk gets Analyzed
3. suitable for big complex projects

Disadvantages of the Evolutionary process model

1. It is not suitable for small projects
2. The complexity of the spiral model can be more than the other sequential models
3. The cost of developing a product through a spiral model is high

Unified process



Inception phase: Communication with the customer made and all the features are identified and plan for the development process

Elaboration phase: Identified features are modeled

Construction phase: Constructed and developed

Transition phase: The software is deployed on the Client side

It follows incremental and iterative processes
It is used for object oriented software Development
The incremental and iterative means requirements are identified upfront.
The software is deployed at the client side

The output of each phase:

Inception phase: The initial use case models includes the features, the business case, risk list, project plan prototype

Elaboration phase: Analysis model, preliminary model manual use case model

Construction phase: The software is developed, the test plan made, the user manual, the installation manual are written

Transition phase: Software increment, the beta test report, and the user feedback

Software requirement: is a condition needed by a user to solve a problem

Requirement should be clear, correct and well-defined

Types of Requirements

(1) Functional requirements

(2) Non Functional requirements

Difference between Functional & Non Functional requirements

	Functional	Non Functional
(1)	Helps to understand the functions of the system	Help to understand the system's performance
(2)	Mandatory requirements	Not mandatory requirements
(3)	They are easy to define	They are hard to define
(4)	It concentrates on the user's requirement	It concentrates on the expectation of the user
(5)	These requirements are specified by the user	These requirements are specified by the Software developers, architects and technical persons.
(6)	Describe what the product does	Describe how the product works

7. There is functional testing such as API testing, system, Integration etc	7. There is Non functional testing such as Usability performance, Stress security
8. First completion of Functional requirements allows the system to perform Non functional requirements.	8. While system not only work for Non functional requirements.

User requirements :

(1) Written for customers

(2) In Natural language

(3) Describe the services & features provide by system

(4) May include diagrams & tables

(5) Understandable by system users who don't have technical knowledge

(6) For client managers, System users, Contractors managers, System Architects

System Requirements:

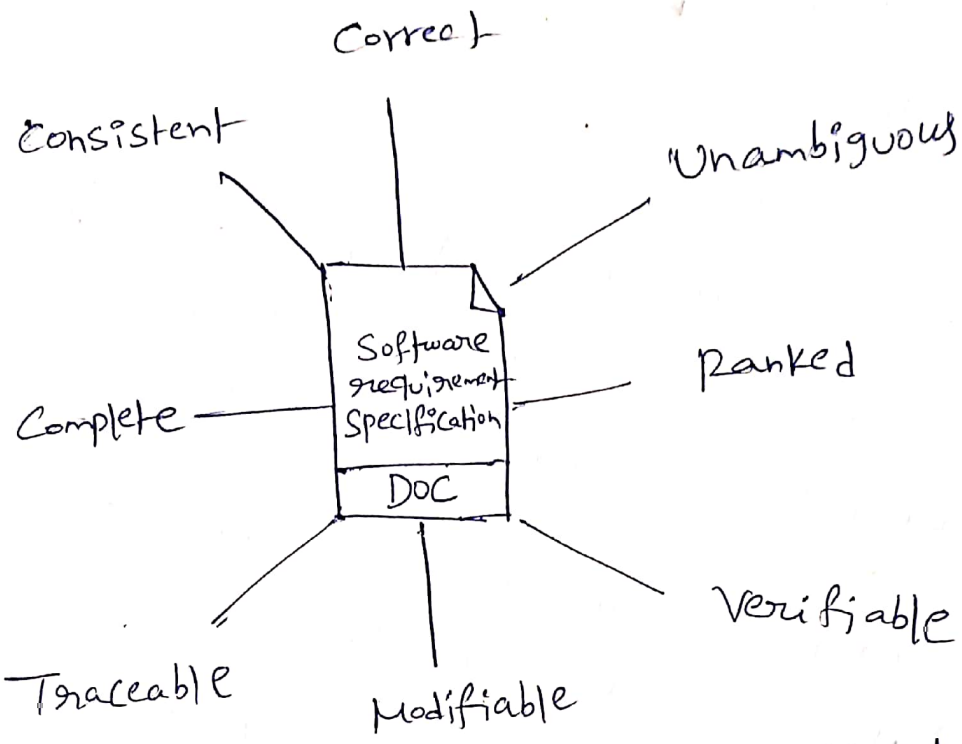
1. Written for Implementation team
2. In Technical Language
3. Describe the detail description of services features & complete operations of a system
4. May include system & models
5. Understandable by implementation team who have technical knowledge
6. For system Architects, Software Developers Client Engineers, system users

Software requirement Document:-

SRS is a Behaviour of system
SRS is a Functional & Non Functional requirements of system
SRS is a formal report

Users of SRS

Client
Development team
Maintenance team
Technical writers



(1) Correctness: Provide accurate functional & Non functional requirements

(2) Completeness: It include functionality, performance design, features

(3) Consistency: It supported by tabular format or diagram format

(4) Unambiguous: Everything mention uniquely & clearly

(5) Ranking: Importance and Stability

(6) Modifiability: Capable of quickly changes to the system without affecting other

(7) Verifiability: The requirements are verified with the help of reviewers & Stakeholders

(8) Traceability: Every requirement having unique number that is easy to use in future development

Requirement Engineering Process

The process to gather the software requirements and analyze the document is known as requirement Engineering

. Feasibility studies: find out the current user needs and budget

Types of Feasibility study

1. Technical Feasibility: Evaluate the current technologies, achieve customer requirements within the time and budget
2. operational Feasibility: Required software solve business problems and customer requirements
3. Economic Feasibility: It can provide financial profit for an organization

Requirement Elicitation & Analysis: is a process of gathering actual requirements about the needs and expectations of stakeholders for a software system

Techniques of requirement Elicitation

1. Interviews: One-on-one conversions with stakeholders
2. Surveys: It is also called as questionnaries
3. Focus Groups: It focus on small group of stakeholders

(4) Observation: Observing the stakeholders in their work environment to gather information

(5) Prototyping: Creating a working model of the Software system

Requirement Validation

After SRS developed to check the requirement documents Validated or Tested

Requirement Validation done through requirement reviews taken by customers

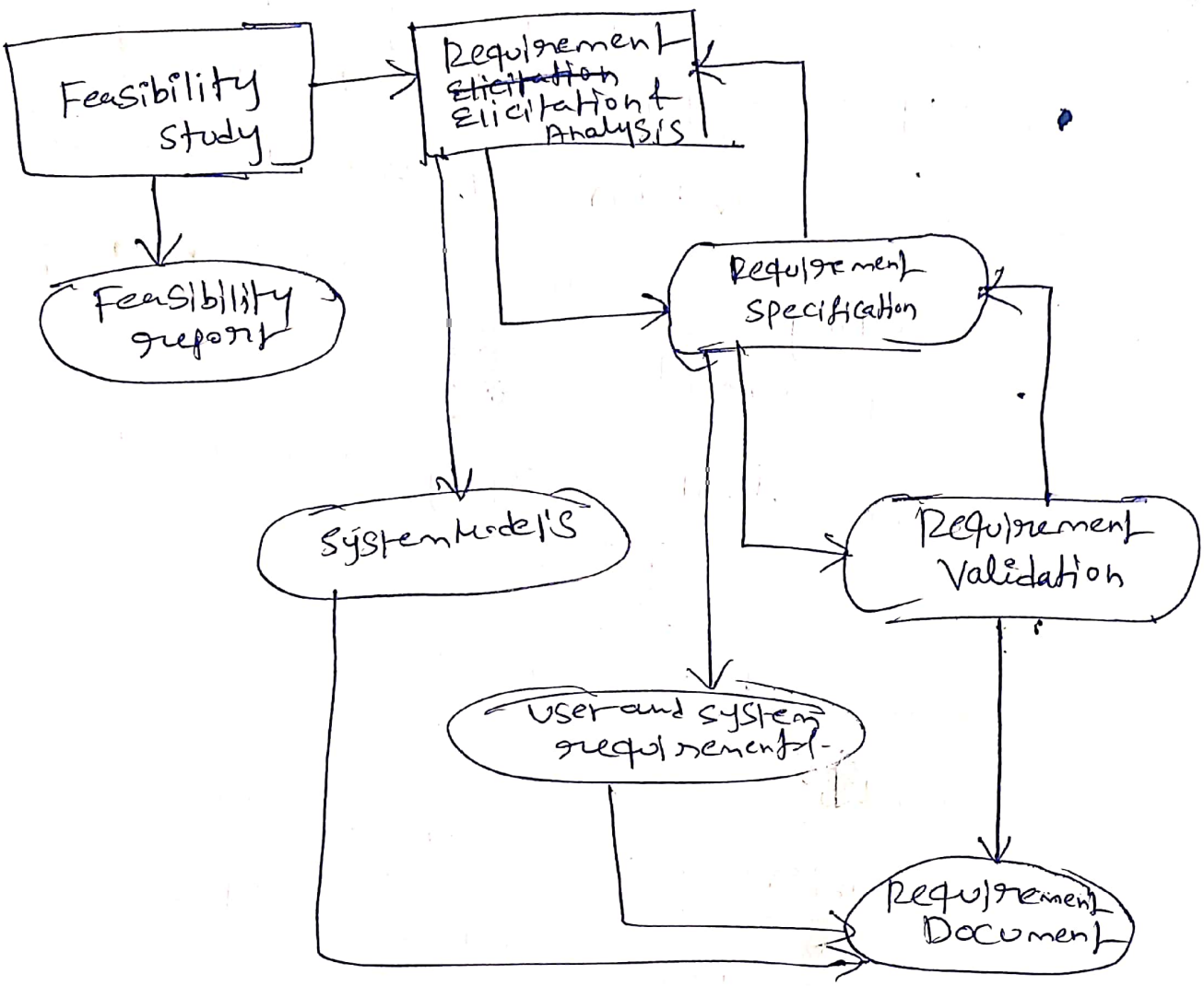
Requirements can be the check against the following conditions

(1) If they can practically implement
(2) If they are correct
(3) If they are any ambiguities
(4) If they can describe
(5) Any changes

Software requirement Management: is the process of managing changing requirements during the requirement Engineering process

Activities involved in Requirement management

1. Tracking & Controlling Changes: Changing requirements from the customer
2. Version Control: keeping track of different versions of system
3. Traceability: Trace to software design develop
4. Communication: If changing requirements has any issues contact with clients within time
5. Monitoring & Reporting: Monitoring development process & report the status



Requirement Engineering Process