What is Software Engg ?

- Software Engineering is an branch associated with development of software product using well-defined scientific principles, methods, procedures.

- The outcome of software engg is an efficient and reliable software product.

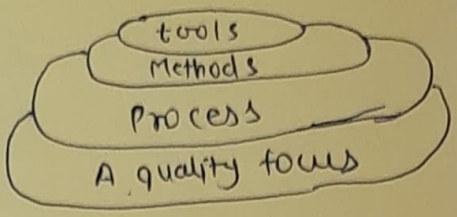
Software product :-

(1) Requirements
(2) System Analysis
(3) System Design
(4) Code Design
(5) Testing
(6) Deployment
(7) Maintenance
(8) Updates

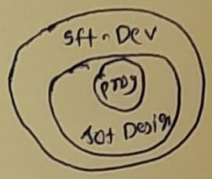
- It is systematic, displined, cost effective techniques for software development.

- Engg Approach to develop a software

Software Engineering is a layered technology



Software paradigms
steps taken while designing the software.



(1) Software Design para.
(2) programming paradigm

characteristics of good software :-

(1) Operational
(2) Transitional
(3) Maintenance

Need of Software Engg

- Large software
- Scalability
- Cost
- Dynamic Nature
- Quality Management

process frameworks

- It establishe the foundation for a complete software engg process by identifying a small number of framework activities that are applicable to all software projects

- Applied repeatedly through a numbea of project iterations.

Generic Process Framework

(1) Communication ✓
(2) Planning ✓
(3) Modeling ✓
(4) Construction ✓
(5) Deployment ✓

(1) communication
- Before any technical work
- communicate and collaborate
- customer, & talcholder

(2) Planning
- works as Map
- tasks
- risks
- work schedule

(3) Modeling
- Sketch
- prototyping
- wireframing

(4) construction
- code generation
- testing

(5) Deployment
- delivered to the customer
- tested

Umbrella Activities

(1) process framework activities ar complemented by a number of Umbrella activities

(2) Help a software team manage and control
- progress
- quality
- change
- risk

(i) project Tracking & control
- Monitoring Progress
- Maintaining schedule

(ii) Risk Management
- Identifying Risks
- Dealing with Risks

(iii) Quality Assurance
- Ensuring Quality

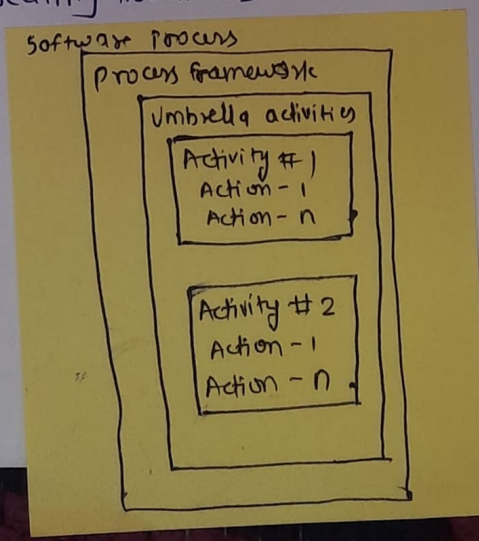
(iv) Technical Reviews
- finding Errors

(v) Measurement
- collecting Data

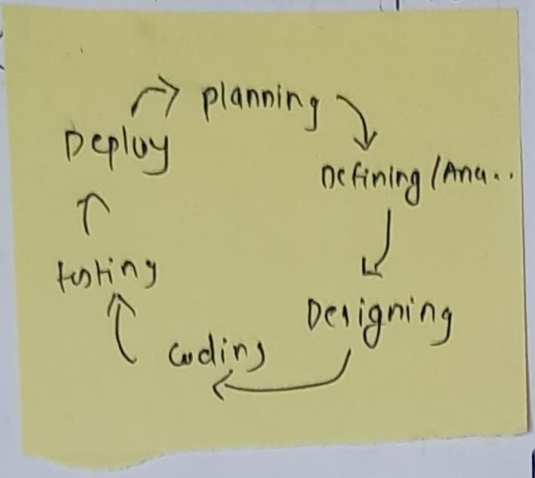
(vi) configuration Management
- Handling changes

(vii) Resusability Management
- Using Things Again

(ix) Work Product Preparation & Productio
- Creating Documents



Software Development Life Cycle (SDLC)



(1) Planning
- communication
- Draft

(2) Defining
- Estimation
- Analysis
- cost time
- charges
- SRS
- formal written document

(3) Designing
- Modules
- coupling
- DFDs

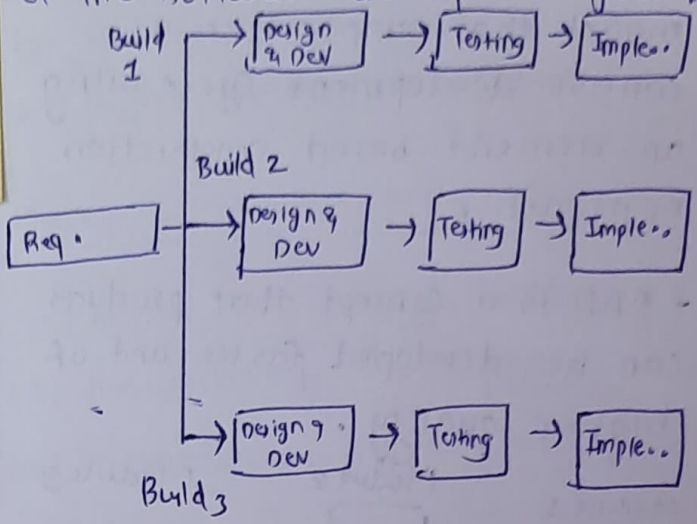
(4) Coding / Implementation

(5) Testing

(6) Deploy

Incremental Model

Incremental Model is a process of software development where requirements divided into multiple standalone modules of the software development cycle.



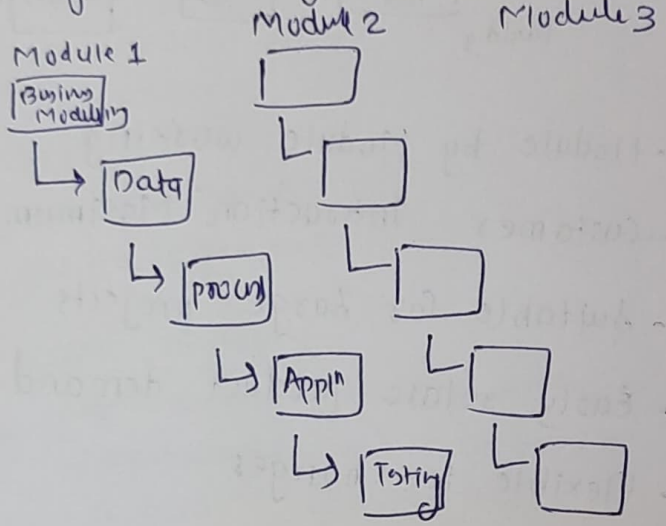
- Module by Module working
- Customer Interaction Maximum
- Suitable for large projects
- Early relase product demand
- Flexible to changes

Advantages:-
- Errors are easy to be recognized
- Easier to test & debug
- More flexibile
- simple to manage risk because it handled during its iteration

Disadvantages:-
- Need for good planning
- Total cost is high
- well defined module interfaces are needed

RAD Model (Rapid Application Development Model)

- RAD is a linear sequential software development process model that emphasizes a concise development cycle using an element based construction approach.
- RAD is a concept that products can be developed faster and of higher quality.



(1) Business Modelling
(2) Data Modelling
(3) Process Modelling
(4) Application Generation
(5) Testing & Turnover

Advantage :-

- This model is flexible for change.
- In this model, changes are adoptable
- It reduced development time
- It increases the reusability of features

Disadvantage :-

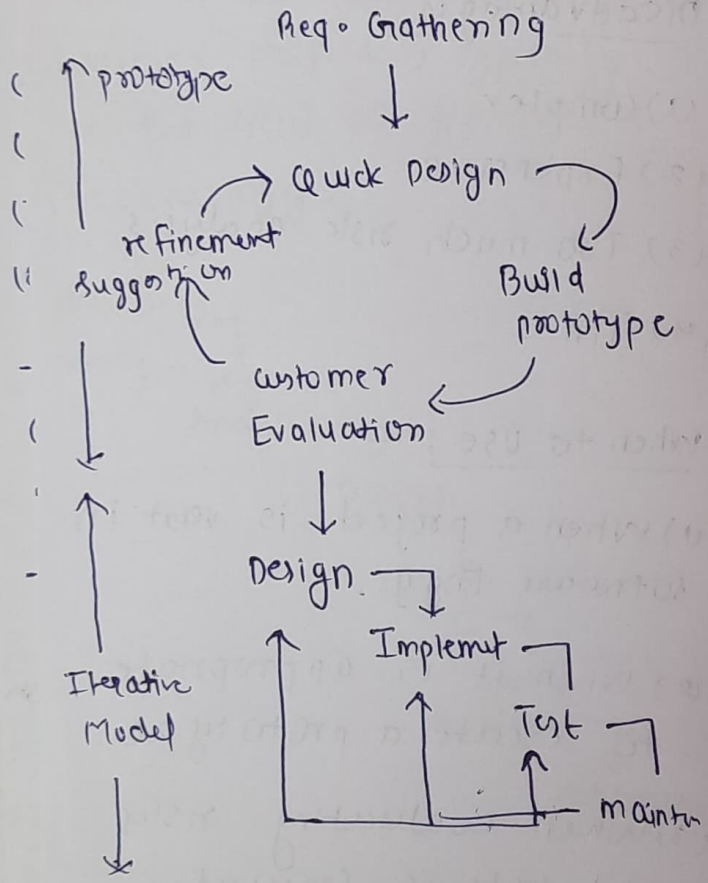
- It required highly skilled designers
- Required user involvement
- on the high technical risk, it is not suitable

Prototype Model

- Prototyping is defined as the process of developing a working replication of a product or system that has to be engineered.

- Kind of dummy Model / Creating a structure

- This model is used when the customer is not clear with idea.



Steps :-

(1) Requirement Gathering & Analysis
(2) Quick Design
(3) Build a Prototype
(4) Initial User Evaluation
(5) Refining Prototype
(6) Implement Product & Maintain

Types of prototyping Models

(1) Rapid Throwaway Prototyping
(2) Evolutionary Prototying
(3) Incremental Prototyping
(4) Extreme Prototyping

Advantages

(1) Missing functionalities can be easily figure out
(2) Flexibility in design
(3) The Customers get to see the partial product early in the life cycle.
(4) Increase the customer Staisfaction

Disadvantages

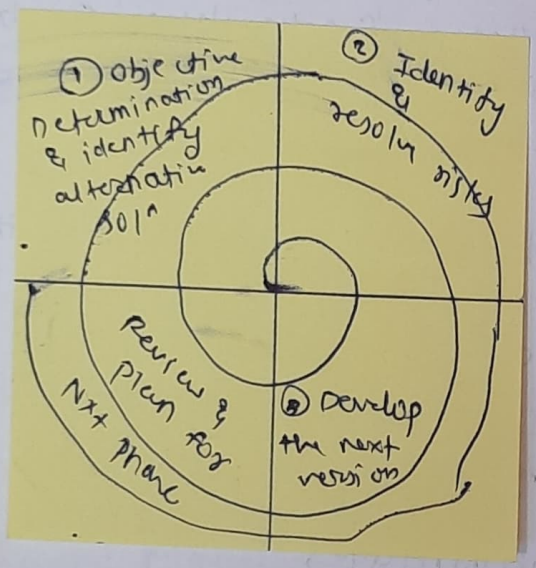
(1) Costly with respect to time as well as money
(2) Additional Training
(3) May not be scalable to meet the future needs of the Customer.

Spiral Model

The Spiral Model is one of the important Software Development life cycle models, which provides support for Risk Handling.

- In its diagrammatic representation it looks like a spiral with many loops. The exact number of loops of the spiral is unknown and can vary from project to project.

- Each loop is called a phase of the software development process.



Phases :-

① objectives determination and identify alternative solutions

② Identify and resolve Risks

③ Develop next version of the product

④ Review and plan for the next phase

* Risks Handling
* Radius of spiral = Cost
* Angular Dimension = progress
* It is also known as meta model

Advantages

(1) Risk Handling
(2) Large projects
(3) flexible
(4) Customer satisfaction

Disadvantages

(1) Complex
(2) Expensive
(3) Too much risk analysis
(4) Time

When to use :-

(1) When a project is vast in software Engg.

(2) When it is appropriate to create a prototype

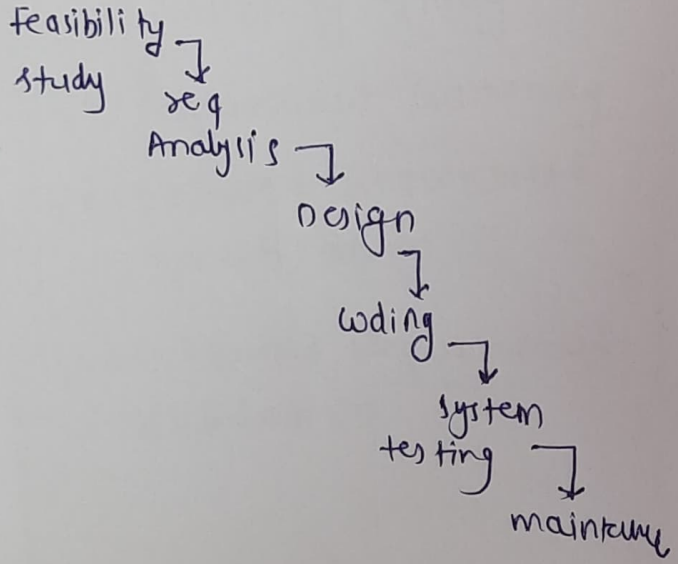
(3) When evaluating risks and costs is crucial.

Waterfall Model

(1) Classical Waterfall Model (1970s)

- It is the basic SDLC model. It is very simple but idealistic. Earlier this model was very popular but nowadays it is not used.

- The waterfall model is a software development model used in the context of large, complex projects, typically in the field of Information Technology.



phases :-

(1) Feasibility study
(2) Requirement Gathering and analysis
(3) Design
(4) Coding & Unit testing
(5) System testing and Integration
(6) Maintance

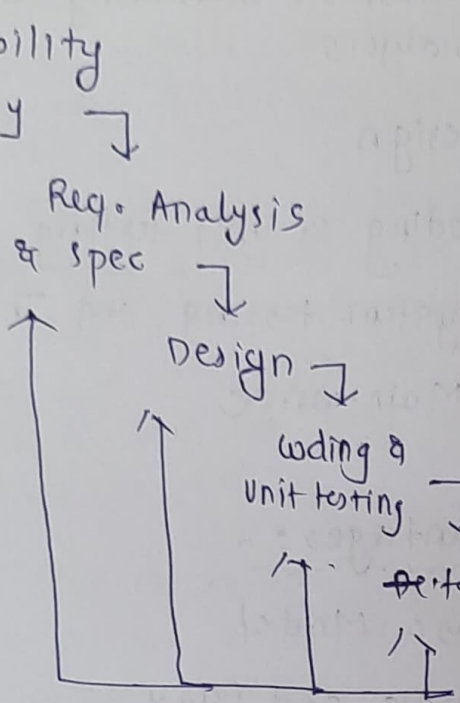
Advantages :-

* Base Model
* Simple and Easy
* Small projects

Disadvantage :-

- NO feedback
- NO Experiment
- NO parallerism
- High Risk
- 60% Efforts Maintenance

Iterative Waterfall Model



Phases

1) feasibility study
2) Requirement Analysis and specifications
3) Design
4) coding & Unit testing
5) testing
6) Maintenance

Advantages

(1) Base Model
(2) simple and Easy
(3) Small projects
(4) feedback

DisAdvantages

(1) NO phase overlapping
(2) NO intermediate Delivery
(3) Rigid (NO changes)
(4) Less customer interaction

Software Requirements

- It is the description of features and functionalities of the target system.
- It is the description of what the system should do.
- RE (Requirement Engineering) refers to the process of defining documenting, and maintaining requirements in the engg design process
- It is a four step process, which includes :—
  (1) feasibility Study
  (2) Requirement Gathering
  (3) Software Requirement Specification
  (4) Software Requirement Validation.

Tools Support for RE

- Observation reports
- Questionnaries
- Use cases
- Used stories
- Requirement workshops
- Mind Mapping
- Role-Playing
- Prototyping

Functional Vs Non-function Requirements

functional :-

Requirements, which are related to functional / working aspects software fall into this category

Non-functional :-

Requirements are expected characteristics of target software.