Software Engineering

Lecture Notes (Lecture No 1)

UNIT-1 (INTRODUCTION)

Content → Introduction to S/w Engineering.

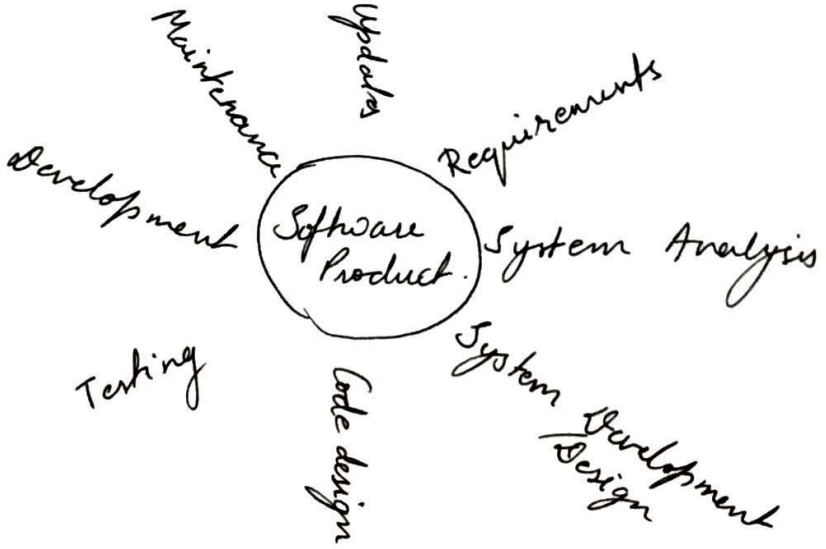
→ **Software -**

It is more than just a program. A program is an executable code, which serves some computational purpose.

While Software is considered to be collection of executable programming code, associated libraries and documentations. Software when made for a specific requirement is called software product.

→ **Engineering -**

It is the process of developing products, using well defined, scientific principles and methods.



→ **Software Engineering -**

It is an engineering branch associated with development of software product using well-defined scientific principles, methods and procedures. The outcome of Software Engineering is an efficient and reliable software product.

IEEE Defines Software Engineering as

1) The application of a systematic, disciplined, quantif--iable approach to the development, operation and maintenance of software.

2) It is the establishment and use of sound eng--ineering principles in order to obtain economically software that is reliable and work efficiently on real machines.

Scope & Necessity of Software Engineering -

→ It is required to achieve good quality software.
→ The budget can be planned in better way.
→ Maintenance cost gets reduced.
→ Large software can be be handled in better way.
→ It helps the developers to delivers the product according to the time plan.
→ The complexity can be resolved with the help of two techniques :-
  a) Abstraction.
  b) Decomposition.

Components of Software -

Software = Program + Documentation + Operating manual procedures

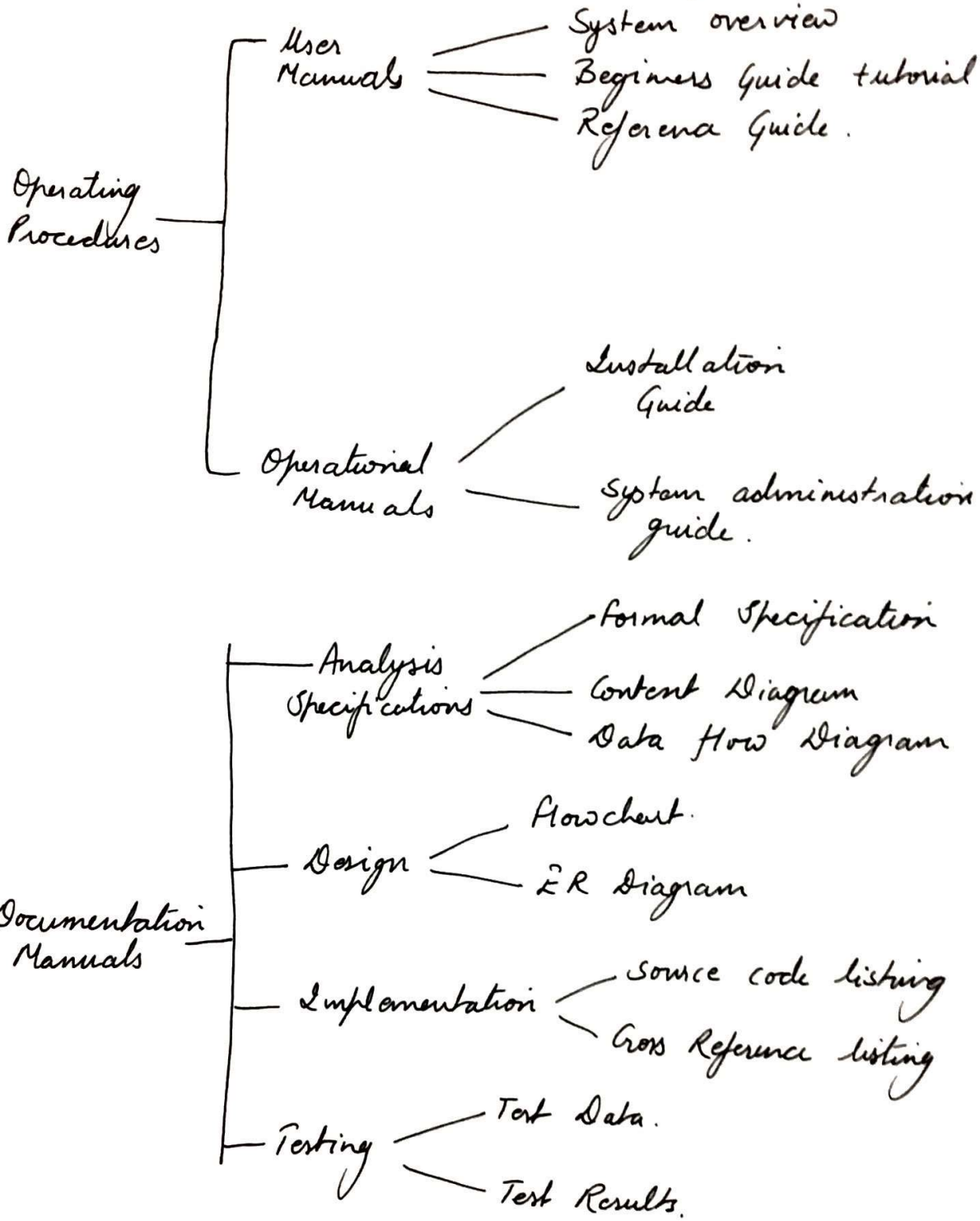
Operating Procedures -

It consists of instraction to set up and use software system and instruction on how to repeat to system failure.

It provide information about

1) What the software is

3) How to install
4) How to control all the activity of software.



→ Difference between Software Engineering & System Engineering -

* System engineering is concerned concerned with all aspects of development and evolution of complex systems where as software plays a major role in

in the software engineering concepts.

* System Engineering is therefore concerned with hardware development, policy and process design and system development as well as S/W System Engineering.

* System Engineering is an older engineering than Software Engineering but the percentage of software in system has increased ie software engineering technique such as usecase modelling and configuration management are being used in the system engineering process.

Difference between Engineering & Manufacturing

Engineering refers to planning & designing whereas Manufacturing refers to using of machines and raw materials to physically make the thing.

Lecture - 2

Contents → Software Characteristics & Software Crisis.

Characteristics of a good Software

3 main characteristics of good software

Operational Characteristics (Functionality - based factors Related to exterior quality)
- → Correctness
- → Usability
- → Integrity
- → Reliability
- → Efficiency
- → Security
- → Safety

Transition Characteristics (Engineering based factors. Related to interior quality)
- → Maintainability
- → Flexibility
- → Extensibility
- → Scalability
- → Testability
- → Modularity

Revision Characteristics
- → Interoperability
- → Reusability
- → Portability.

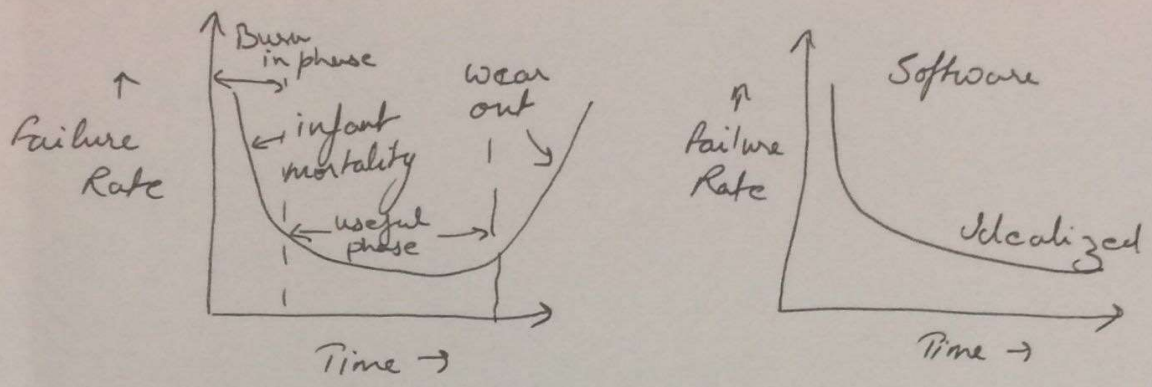
Software Engineering is a branch of computer science, which uses well-defined engineering concepts required to produce efficient, durable, scalable, in-budget, and on-time software products.

→ Characteristics of Software -

Software has characteristics that are considerably different from those of hardware.

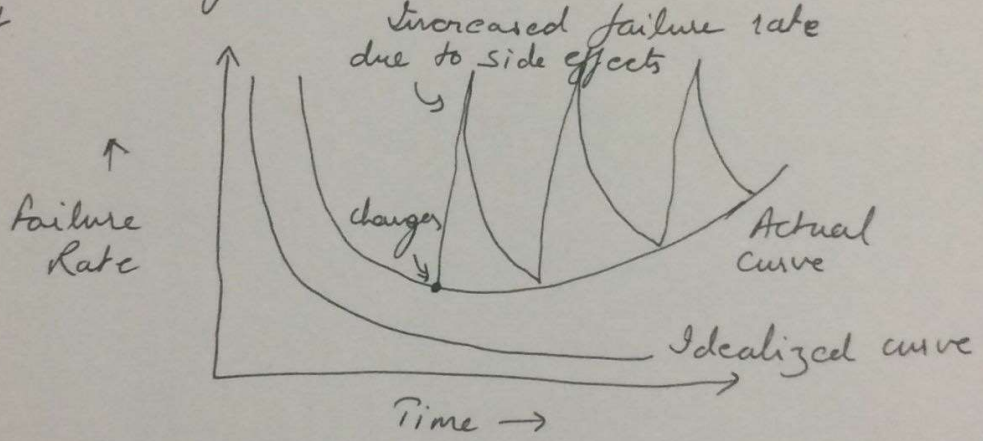
* Software is logical rather than physical.
* Software is developed or engineered, it is not manufactured in the classical sense.
*

* Software does not wear out.



Failure curve of hardware (Bath-tub curve)

* Although the industry is working towards component based assembly, most software continues to be custom built



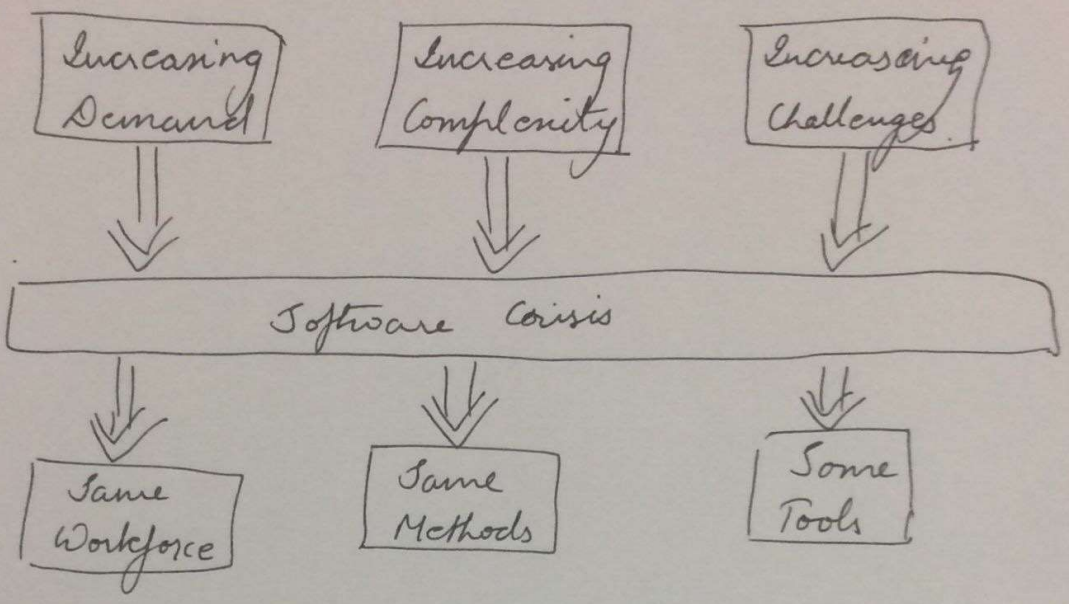
* Reusability of components
* Software is flexible.
* Software is intangible.

→ ⓐ Software Crisis -

* A number of large size projects fails and cause disaster that is called S/w run away and results into so called S/w Crisis.

* A S/w Crisis is a mismatch b/w what software can deliver and the capabilities of the computer system as well as expectations of their users.

* It is a set of difficulties or problem encountered while developing a software.



S/w Crisis.

→ Reasons or Causes of S/w Crisis -

It is characterized by the inability to develop S/w on time with the required requirements and the main reason are -

* Lack of communication between the S/w developer & user
* Increase in size of S/w.
* Increase in cost of developing S/w.
* Project Management Problem.
* High optimistic estimates regarding S/w development
* Shortage of skills.

→ Results or After Effect of S/w Crisis. -

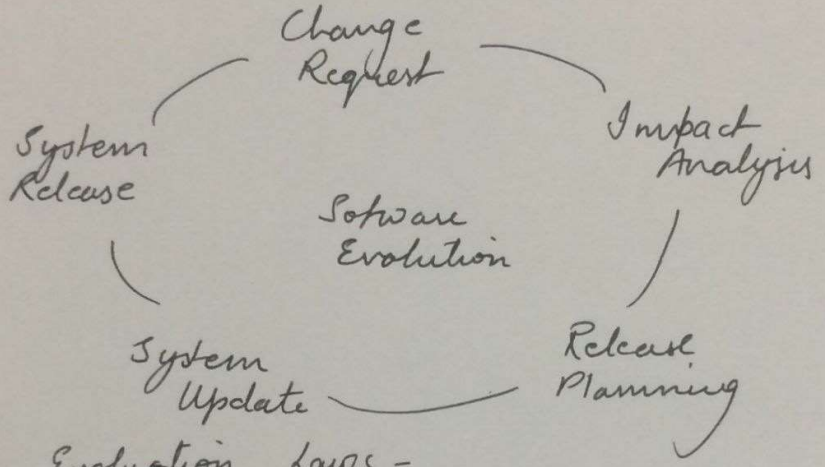
* Project running over budget.
* Projects running over time.
* Inefficient S/w.
* Low quality S/w
* Code difficult to maintain

Lecture No-3

Content - Software Evolution, Software Paradigm

→ **Software Evolution -**

The process of developing a s/w product using s/w engineering principles and methods are called as software evolution.



→ **Sopware Evolution Laws -**

Lehman has given laws for s/w evolution.
Divided into 3 categories.

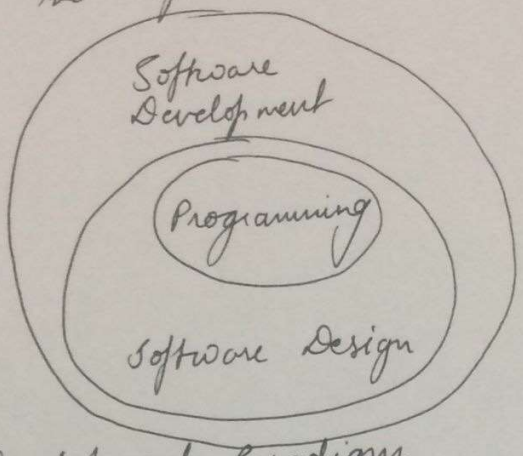
* **S-type** (Static type) -
This is a s/w which strictly works according to defined specifications & solutions.
It is least subjected to changes.

* **P-type** (Practical-type)
This is a s/w with a collection of procedures.
Specification can be described but the solution is not instant.

* **E-type** (Embedded-type)
This s/w works closely as real environment.
High degree of evolution as various changes are done.

Software Paradigms-

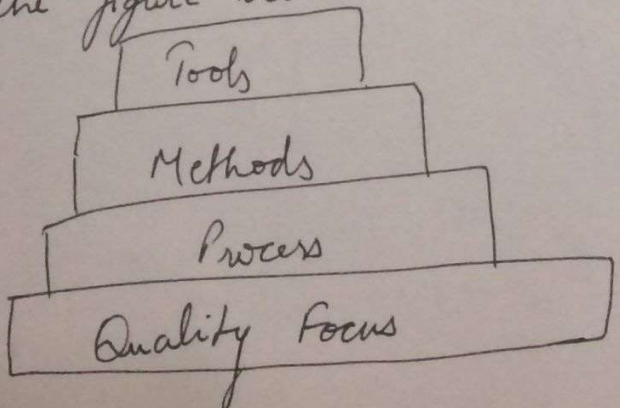
It refers to the methods and steps which are taken while designing the software



- Software Development Paradigm
  It consists of
  - → Requirement Gathering
  - → Software Design.
  - → Programming.
- Software Design Paradigm -
  - → Design
  - → Maintenance
  - → Programming
- Programming Paradigm -
  - → Coding
  - → Testing.
  - → Integration

Software Engineering (A Layered Technology)

S/w Engineering is called as layered technology based on the figure below.



* The bedrock that supports S/w engg is a quality focus factors included are
  - → Reliability
  - → Portability
  - → Maintainability
  - → Platform Independence.
  - → Traceability.
* Foundation is Process Layer. Process defines a framework for a set of key process area that must be established for effective delivery of S/w engg product.
* S/w Engg Methods provide the technical how-to for building S/w. This method encompasses a broad array of task that include.

  i) Req" Analysis ii) Design iii) Coding iv) Testing v) Implementation & maintenance.
* Tools provide automated or semi-automated support for Methods & Processes used.

Lecture No - 4

CONTENTS - Software Engineering Processes, Software Applications

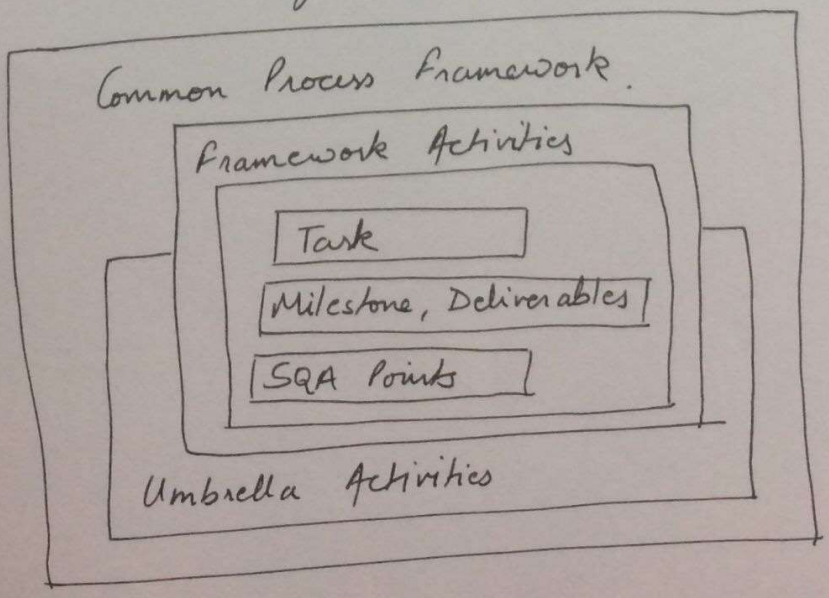
→ Software Engineering Process -

It is a set of activities whose goal is the development or evolution of software.

→ Generic Activities in all S/w Processes are:

* Specification -
  What the system should do and its development constraints
* Development -
  Developing the s/w system to meet the specification.
* Validation -
  Checking that the s/w is what the customer wants
* Evolution -
  Changing the s/w in response to changing demand.

→ Software Process Diagram -



S/w Process.

→ Why it is difficult to improve S/W processes :-

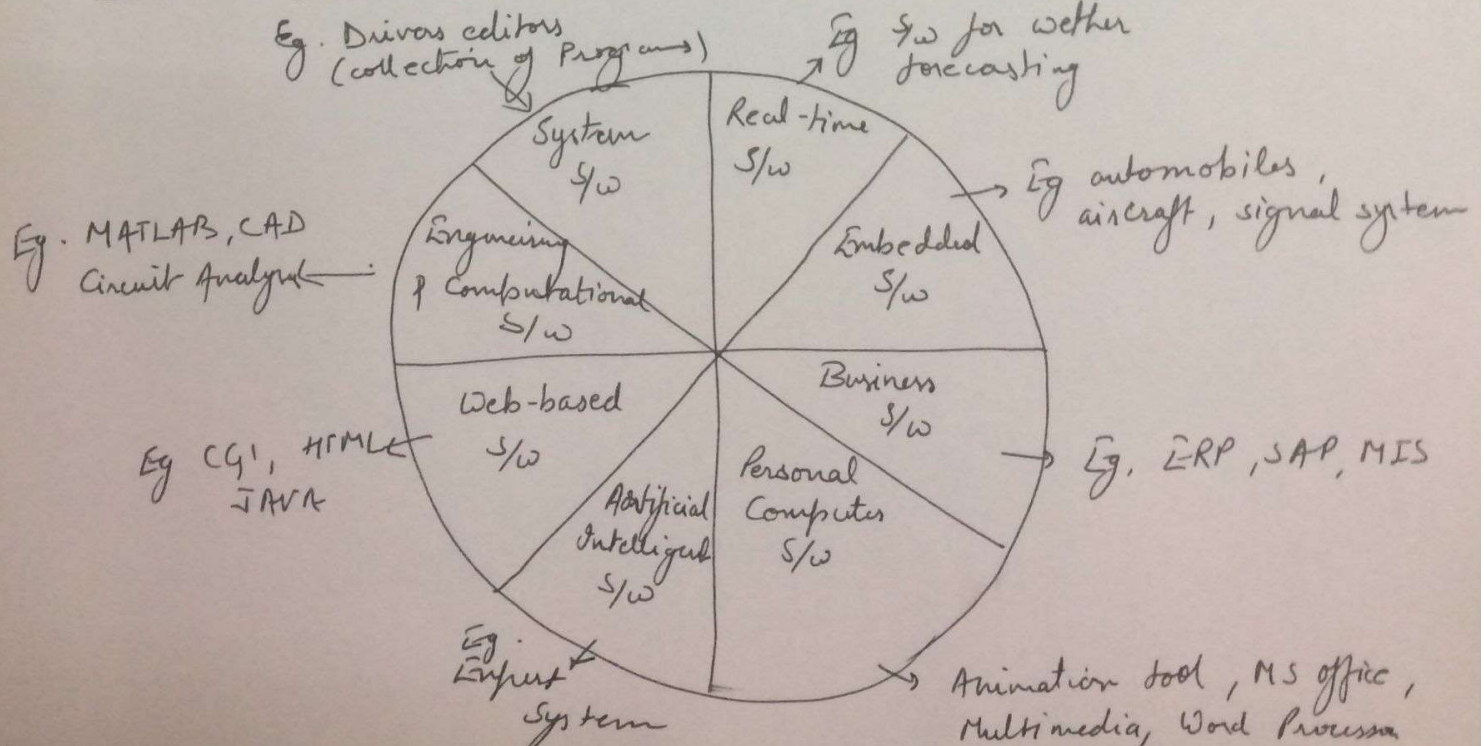
* Not enough time
* Lack of knowledge
* Wrong motivations
* Insufficient commitment

△ Common Process Framework :- It is established by defining a small no of framework activities that are applicable to all S/W projects., regardless of their size & complexity.

No of Task Sets- Each is a collection of S/W engineering work tasks, milestones, work products and quality assurance points. It enable the framework activities to be adapted to the characteristics of S/W project and the requirement of the project team.

Umbrella Activities- Such as Quality Assurance, SCM, Software Configuration Management & measurement of the process model. These are independent of any framework activity and occur throughout the process.

→ S/W Applications -

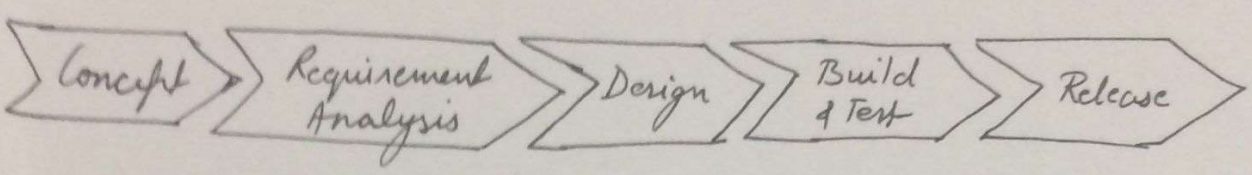


Lecture No - 5

Content - Software Development Life Cycle.

SDLC -

Software Development Life Cycle -



SDLC Phases.

* It is the life cycle of the software starting from the point where the concept of S/W was originated till the S/W is developed and delivered.
* This life cycle has different phases which altogether is called as software development life cycle.
* The final outcome of SDLC is the S/W product.
* Each phase has its entry and exit point.
* The activities included in each phase are as under.

Concept -

→ Vision & Strategy.
→ Defines Roles & Responsibilities
→ Define change Management Process.
→ Communication Plan.
→ Concept level schedule and Budget estimation.

Requirement Analysis -

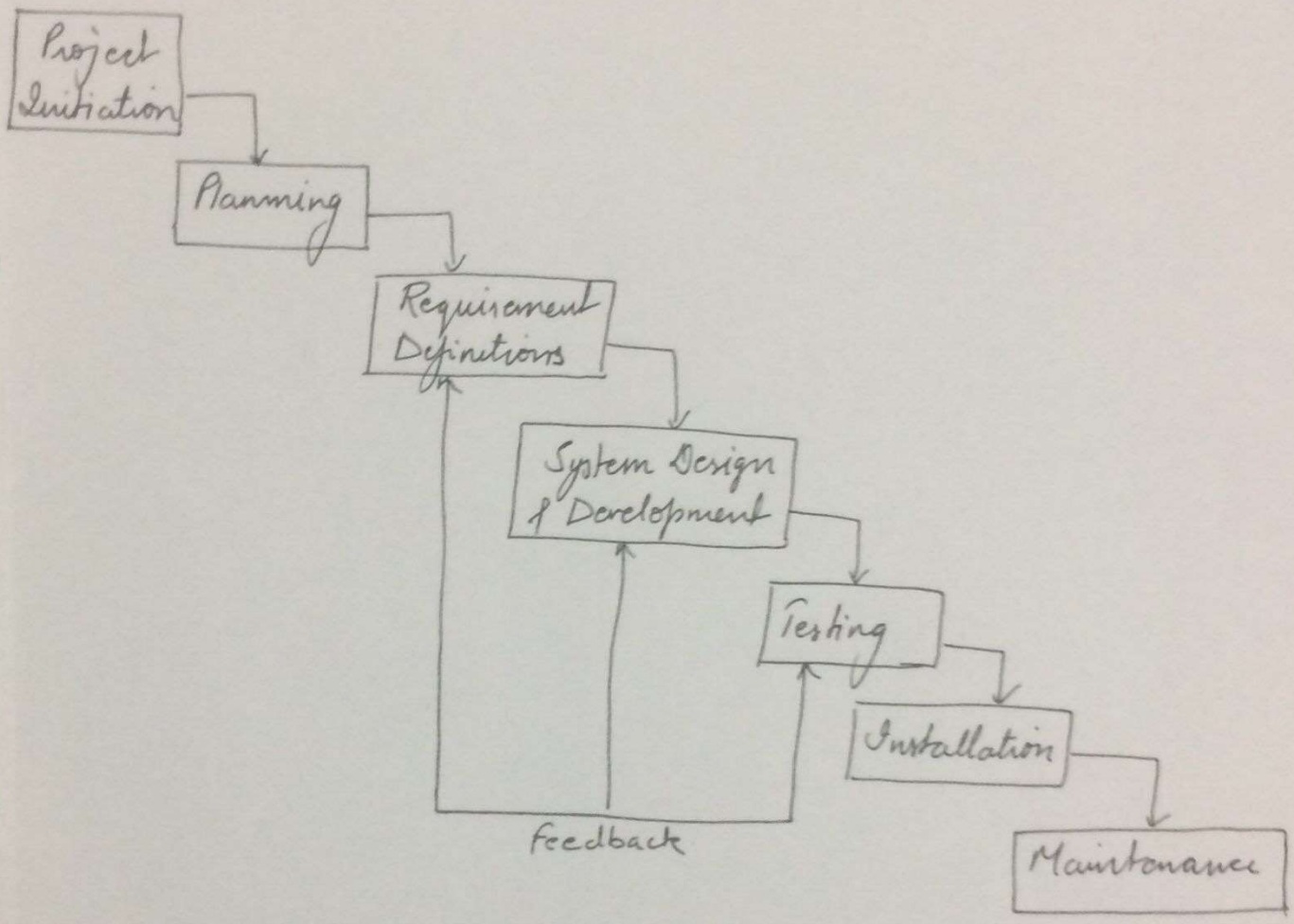
- → As-Is Environment Analysis.
- → Biz rules & Tech Dependencies
- → Competitor Analysis
- → Product Requirements
- → Process Model to be used
- → Use Cases Activity Diagrams.
- → Document To-Be Product.

- Design -
  - → Technical Feasibility
  - → UI (User Interface Design, Visual Creative)
  - → Proof of concept, Prototyping
  - → Full Architecture, Product & Systems
  - → SEO
  - → Re-estimate schedule and Budget.

- Build & Test - (Development & Testing)
  - → Web Coding (Creative + code + SEO)
  - → Unit testing.
  - → Integration Testing
  - → Code Reviews.
  - → Test Cases Execution
  - → Bug fix cycles.
  - → Beta Readiness
  - → UAT
  - → Support Plan & Release Checklist.

Release -

→ Web Release
→ Live Test
→ Transition to Support & Maintenance
→ Lessons Learned.



Detailed Software Development Life Cycle

Lecture No. - 6

Contents - Process Models - Build n' fix, Waterfall Model

→ **Process Model or Life Cycle Model.**

* It describes different activities that need to be carried out to develop a s/w product and sequencing of these activities.
* Also referred to as S/w life cycle model.
* It is a descriptive & diagrametic representation of the S/w life cycle.
* It is a series of identifiable stages that a software product undergo during its life time.

→ **Objective of Process Model -**

* Improvement and guarantee on the quality.
* Cost for the whole life cycle can be checked.
* Communication between different parties is improved

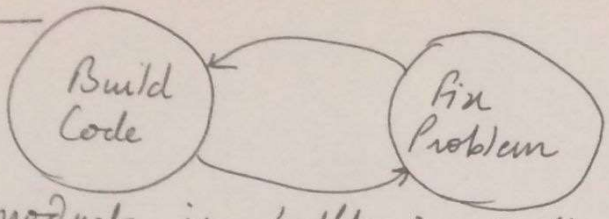
→ **Need of Process Model -**

* Without it the development of Sw product would not be in a systematic and disciplined manner.
* It gives clear understanding to all the team members and when and what to do?
* Monitoring the progress of project becomes easy with process model.
* It defines the entry & exit criteria for every phase.

→ **Different Sw Life Cycle Model -**

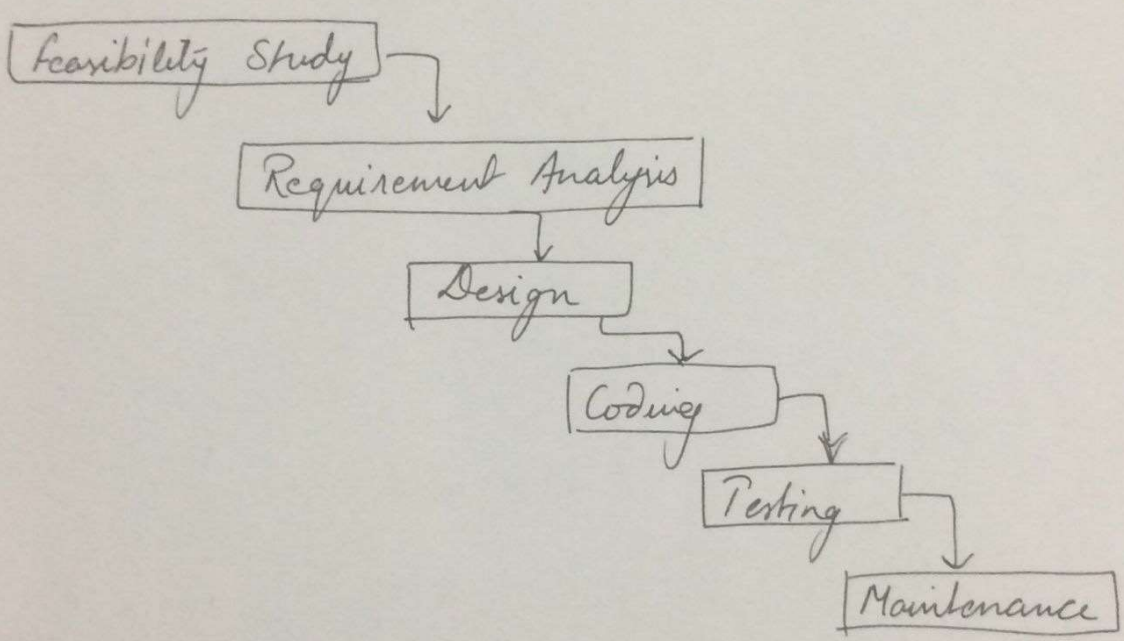
i) Build & fix Model.
ii) Classical Waterfall Model.
iii) Iterative Model
iv) Prototype Model
v) Evolutionary Model.
vi) Spiral Model.

1) Build & Fix Model



* In this product is built & continuously modified till the client is satisfied.
* Uses adhoc approach.
* Used for small projects only.
* Cost increases with the size.
* Maintenance is difficult
* Product is developed without customer specifications.

2) Classical Water Fall Model -

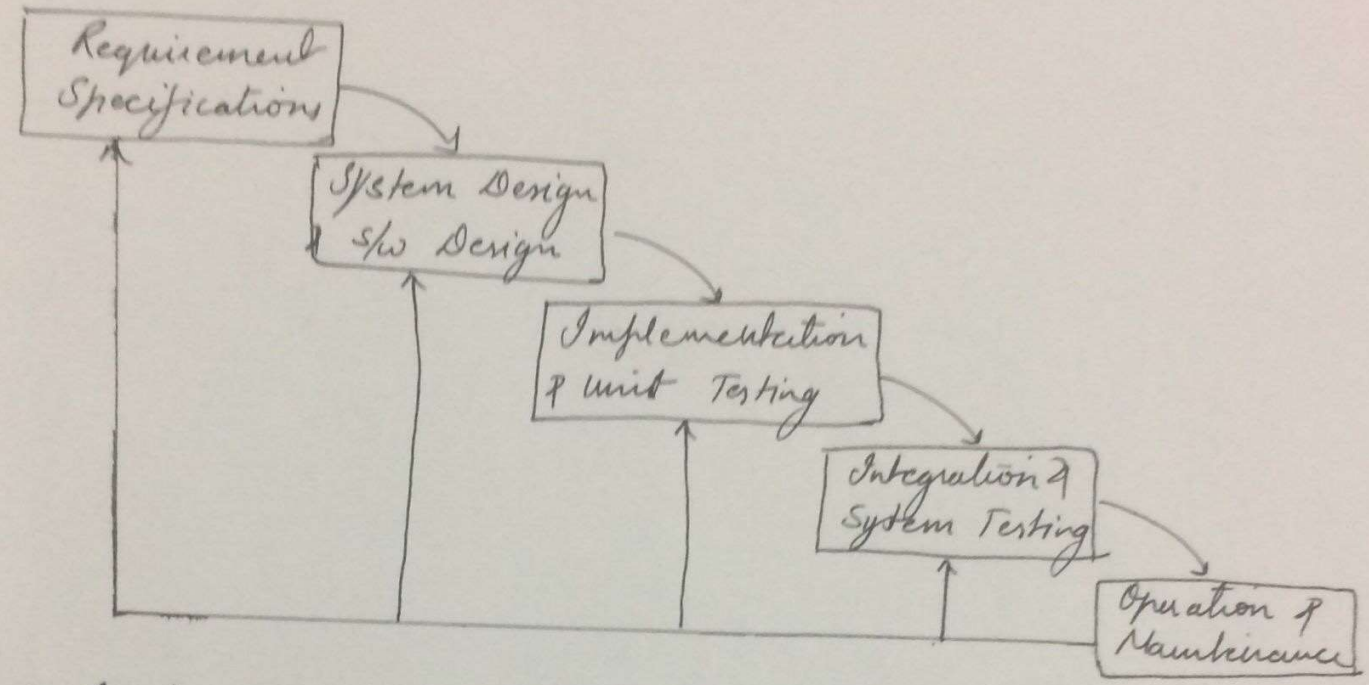


* Diagramatic Representation resembles cascade form of waterfall.
* During each phase a set of different activities are performed which requires relatively different amount of efforts.
* Each phase has well defined entry & exit criteria.

Lecture No - 7

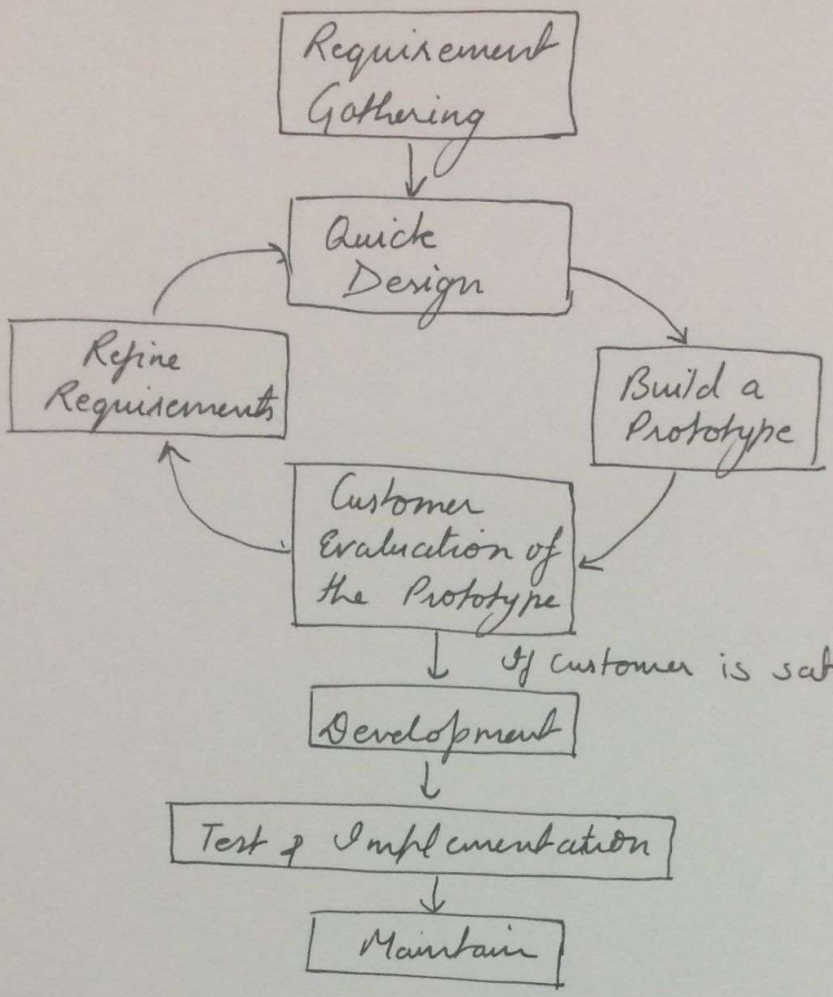
Contents - Iterative Model, Prototype Model

Iterative Waterfall Model -



* It has same working as of waterfall but the only difference is it provide feedback paths for error correction as & when detected later in a phase.
* Advantage ① We get its working model at very early stage of development which makes it easier to find functional or design flaws.
② This in turn helps in correcting the bugs in limited budget.
* Disadvantage - It is applicable only to large & bulky projects. because it is difficult to break a small s/w system into further small serviceable increments / modules.
* Developers implement the specified requirements in one or more cycles of design, implementation, test based.
* The complete product is divided into releases and the Developer delivers the product released by Release.

Prototype Model -

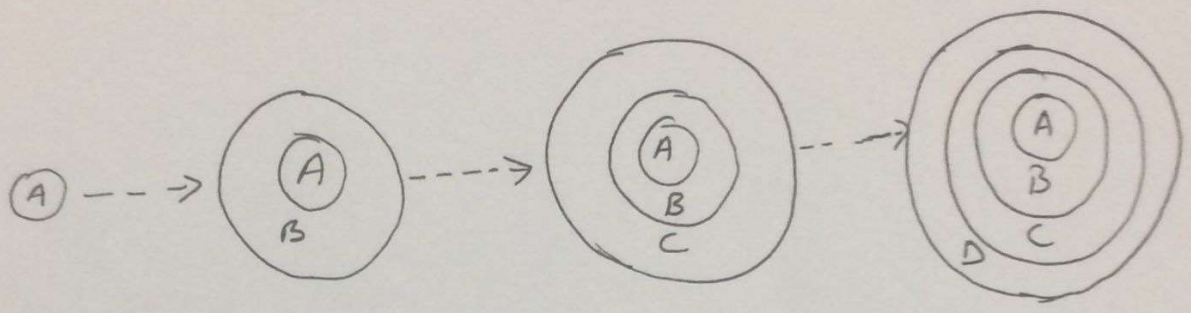


* Before developing the actual software a working prototype of the system should be built.
* Prototype is a dummy software with limited functionalities
* Types of Prototype model.
    - → Exploratory Development Prototype Model.
    - → Throw Away exploratory Prototype Model.
* If the customer is satisfied with the dummy s/w then Development takes place else it is updated according to the feedback of customer.
* Then the s/w is built or developed using the Waterfall Model.
* It includes less maintenance cost.

Lecture No - 8

Contents - Evolutionary Development Model, Spiral Model.

→ Evolutionary Model -



A, B, C, D are modules of a S/w product that are incrementally developed and delivered.

Fig - Evolutionary development of S/w product.

* Also called as incremental model or successive versions model.
* In this S/w requirements is first broken down in several modules / functional units that can be incrementally constructed.
* Firstly one module is developed.
* Then refined and incremented with addition of new functionalities and delivering new version.
* Each successive version of the product is fully functioning S/w, capable of performing more work than the previous versions.
* Suited for large projects and Object oriented concept.
* Do not require large resource at once.
* Experiments with partially developed systems.
* Reduces changing requirement after delivery.

→ Spiral Model -

* Diagramatic representation of this model appears like a spiral with many loops.
* Exact no. of spirals is not fixed & can vary from project to project.
* Each loop is called a phase of the S/w process. Each phase is split into 4 years sectors.
* This model is more flexible compared to other models.
* It was developed by Bohemn in 1988.

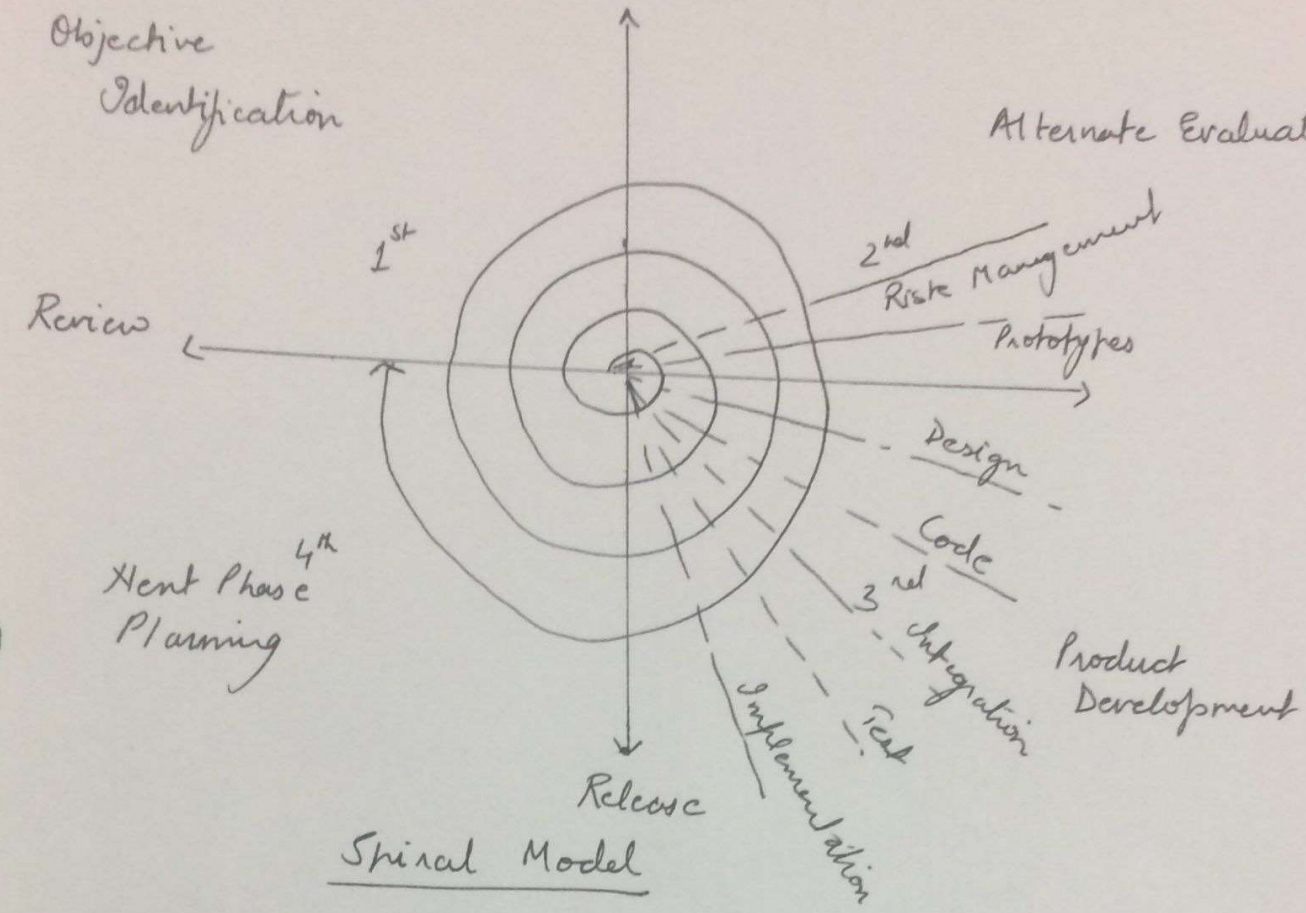
1st Quadrant - (Objective Setting)
- → Identify objectives
- → Examine Risk associated with these objectives.

2nd Quadrant - (Risk Assesment & Reduction)
- → Detailed analysis of each risk.
- → Steps taken to reduce the risk.

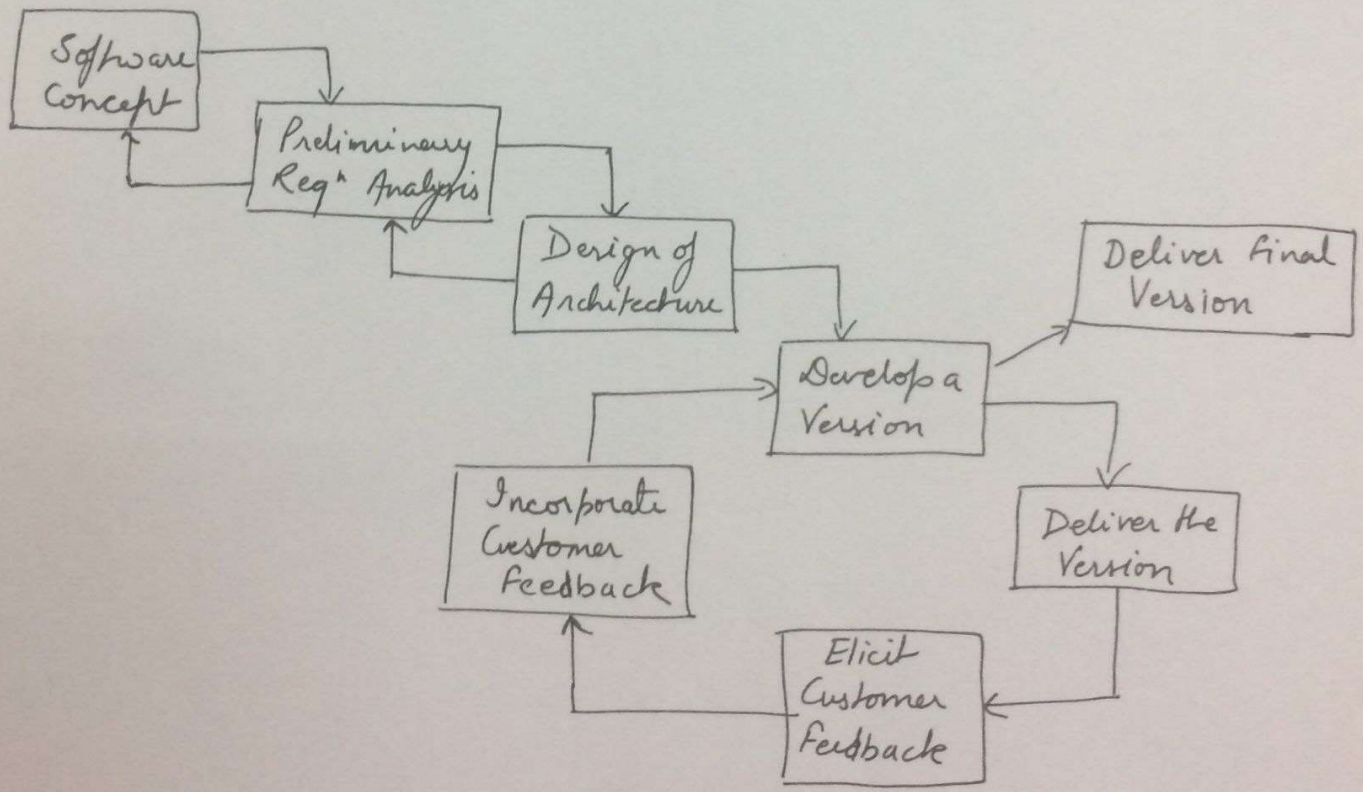
4th Quadrant - (Review & Planning)
- → Review the results achieved with customer.
- → Progressively more complete version of the S/w gets build with each iteration.

3rd Quadrant - Development and Validation
- → Develop and validate the next level of the product after rel resolving the identified risk.

Advantages -
- → Flexible
- → More Reliable
- → Reduced Risk
- → Can be used for complex projects.



Spiral Model



Evolutionary Model

Lecture No -9

Content - Comparison of all the process models.

Waterfall Model

Strength

- → Simple
- → Easy to execute
- → Initiative & Logical

Weakness

- → Requirement changes not possibe
- → All or nothing approach
- → Does not allow change & user feedback later
- → Encourages requirements bloating

Types of Projects

- → For well understood problems only
- → For short duration projects
- → best for automation of existing manual system

Prototype Model

Strength

- → Helps in Requirement Elicitation
- → Reduces Risk
- → leads to a better system

Weakness

- → Front Heavy process
- → Possibly higher Cost

Types of Projects

- → System with no voice users
- → Used when uncertainty is present.
- → Used when user interface is important

Iterative Enhancement Model

Strength

- → Regular & quick delivery.
- → Reduces Risk.
- → Accomodate Changes
- → Allows user feedback
- → Allows reasonable exit points and avoid requirement bloating

Weakness

- → Planning overhead (each iteration)
- → Cost may increase
- → System Architecture & Structure may suffer as frequent changes are made.

Types of Projects

- → When project is of less time.
- → Risk cannot be taken in the project.
- → Where requirement are not known at once they are known with time.

Spiral Model

Strength

→ It is a meta-model.
→ Can encompasses all other models.
→ Risk handling is inherently built

Weakness

→ Much more complex than other models.
→ Not used in small and ordinary projects.

Type of Project

→ For development of technically challenging s/w products that are prone to several kinds of risks.

Selection of Basis for appropriate life cycle model for a project. -

The parameters to be kept in mind while selecting a process model for the project which proves to be best are :-

→ Characteristics of the software to be developed.
→ Characteristics of the development team.
→ Characteristics of the customer.
→ Boundation of Time, Budget & Quality.