

Software Engineering

Software development life cycle (SDLC)

A. Software Process Model / Lifecycle model

PHASES / STEPS

→ MODELS / Software development paradigm.

A strategy that defines SDLC

PHASE / STEPS

Communication

Requirement gathering & analysis

Feasibility analysis/study

→ Requirement analysis & specification

Design analysis

Software design

Coding & unit testing

Testing (Integration)

Maintenance

Disposition

MODELS

Waterfall

IV-Model

Prototype model

Evolutionary model

Iterative model

Spiral model

Agile model

RAD model

Tutorialspoint, n^(SDLC),
wikipedia, javatpoint

Series of steps to be followed
to develop software product

B.

Software project management (SPM)

- Project is collection of well-defined tasks
- Has a distinct goal.

Project management is ~~to~~
undertaking the responsibility
of executing software project.

Eg. Monitoring progress &
performance, Risk analysis etc



C.

Software quality management (SQM)

Undertaking the responsibility that
software product meets the quality
expected by the customers.

Software quality assurance

CMM, ISO, Six Sigma

Phases/Steps of software life cycle (SDLC)

1. Communication by user

User initiates request for desired software product.

Submits request to software developer/service provider.

2. Requirement gathering & analysis

UNDERSTANDABLE, CONSISTENT, UNAMBIGUOUS
COMPLETE

Software development team makes discussion with ~~customer~~ customers and tries to bring out information on their requirements.

Inconsistencies and incompleteness in these requirements are removed.

(a part contradicts with other part).

some part of the requirement is missing

When requirements are properly understood, requirement specification activity starts — i.e. organize requirements into SRS document (Software Requirement Specification)

Improperly documented requirement increase numbers of iterative changes + push-up development cost + set up ground for better customer developer dispute and legal battle. — (Need for SRS)

The engineers who gather and analyze customer requirement and write SRS are known as system analyst.

Once SRS is ready it is reviewed internally by the project team & then given to customer for review. When customer approves it, future development activities may start.

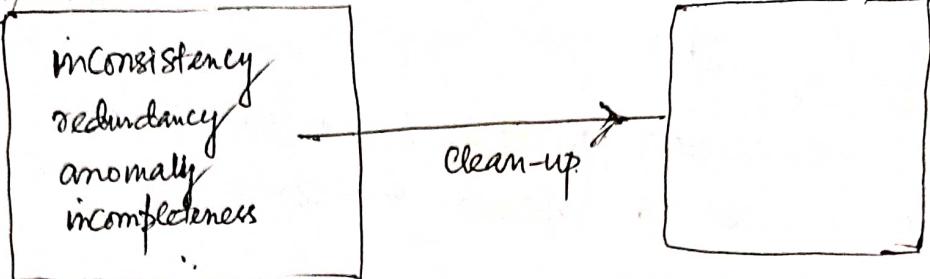
SRS document contains Functional requirement, Non-functional requirement and goals of implementation.

• Functional requirement: Set of functions performed by the system.

• Non-functional requirement: Characteristics of system cannot be expressed in terms of function e.g. maintainability, portability, usability etc.

• Goal of implementation: General suggestions regarding development.

Requirement



Messy

② through interview.

Collect
gather
examine

Clean-up.

Activities involved:

Problem recognition

objective of requirement

Software
requirement
analysis

2. Feasibility analysis / Feasibility study

Determine if it is financially or technically feasible to develop the product.

- Formulation of Abstract problem definition + important requirements are taken & rest of the requirements are ignored.
- Formulation of different solution strategies.
- Compare benefits & shortcomings in terms of resource required, cost of development, time of development etc. based on high level design of the problem.

If none of the solutions are feasible then the project is abandoned.

Otherwise, → Best solution is picked for development.

Feasibility study gives conclusion whether to go ahead or stop.
This also helps to identify risk factors involved in.
Enhances success rate.

Need for Feasibility

Types of Feasibility Study

1. Technical feasibility:

technical skills, capability of technical team,
technology used,

TOS LE

2. Operational feasibility

How ^{much} easy will be the product development

3. Economic feasibility

What will be the cost of development.

4. Legal feasibility

Existence of legal barrier

5. Schedule feasibility

How much time is required to complete the project

3. Software design

Transform requirements into suitable forms which helps programmer to implement it. In software design we consider the system to be set of components or modules with clearly defined behaviours and boundary.

logical designing of the system is converted into physical designing

Purpose of design

Correctness	flexibility
Completeness	Consistency
Efficiency	Maintainability (by other designer)
Understandability	

Design approaches

Function oriented (viewed as collection of functions)
Object oriented (viewed as collection of objects)

4. Coding & unit testing

Translate software design into source code. End product of this phase is set of program modules. To enable the engineers to write good program, every software development organization normally formulates own coding standards. Coding standard involves code template, commenting guideline, variable & function naming etc.

After coding phase each module is unit tested to ensure correct working of all individual modules.

After a module is successfully compiled (i.e. elimination of syntax errors), Code review is carried out. Purpose of code review is to find algorithmic & logical errors.

Code review

1. Code walk through: Some members of development team are given the code for reading & understanding.

6. Maintenance

Software is updated according to changes taking place in user environment or technology.

Types of maintenance

Corrective: To rectify bugs observed while system is in use.

Adaptive: Customer used product on new platforms ie new operating system or ~~with~~ interfaced with new hardware.

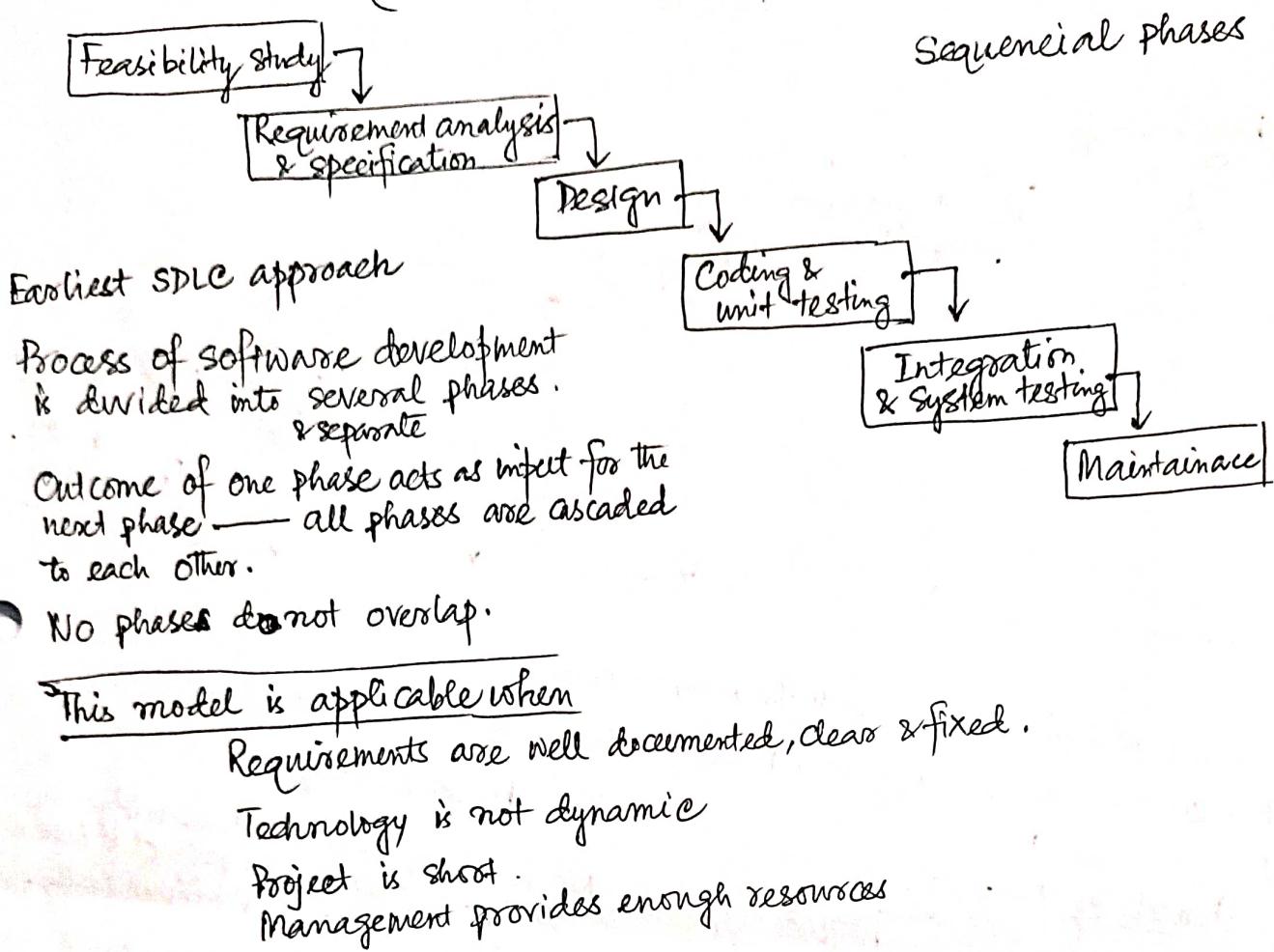
Perfective: Change functionality according to customer's demands.

7. Disposition

As time elapses software may decline on performance. It may go completely obsolete or may need intense upgradation. Developer decides to One down the system.

WATERFALL MODEL (classical)

Linear sequential life cycle model



Advantage

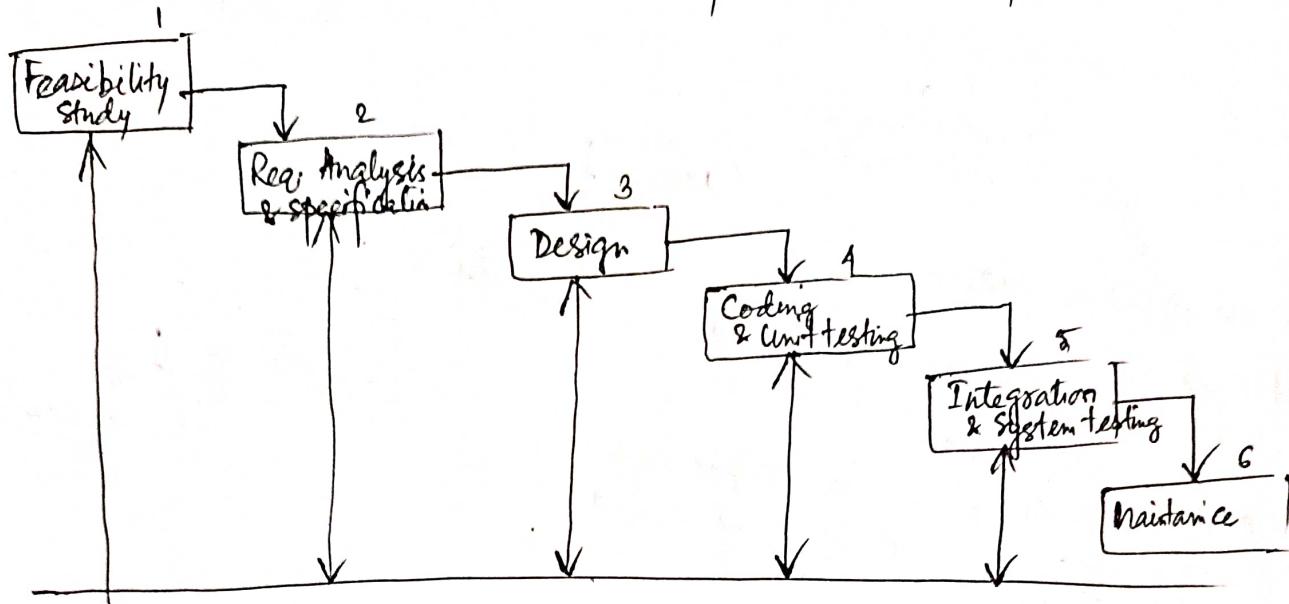
- Simple, Easy to understand
- Easy to use
- Each stage is clearly defined
- Process, Action & Results are well documented

Disadvantage

- High amount of risk & uncertainty
- Not good for complex project
- Very rigid: Cannot accommodate change
- No error correction mechanism.
- New phase starts only after previous stage ends — efficiency reduces.

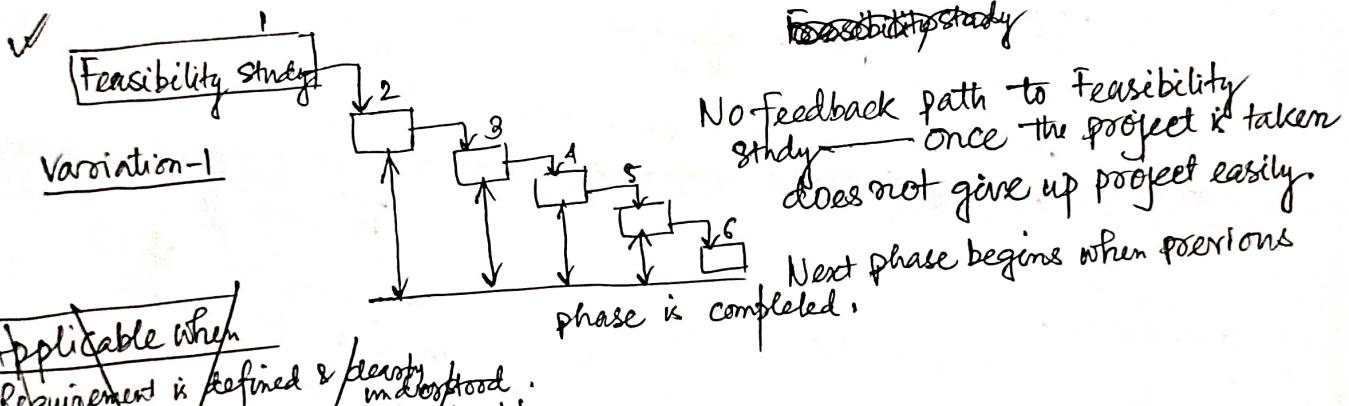
Iterative waterfall model

Extension of classical waterfall model.



Feedback path from every phase to its previous phase

When errors are detected at some phase, feedback path allows correcting errors.



Applicable when

Requirement is defined & ^{understood} by development team.
New technology is learnt by development team.

More costly

Advantage

- Feedback path allows correcting errors
- Simple to understand & use
- Customer involvement is not required during software development
- Suitable for comparatively large & complex project

Disadvantage

- To scope for any incremental delivery
- No overlapping of phases
- Has no risk handling mechanism

Limited customer interaction/review request

Customers interaction occurs at the start of the project & end of the project. Fewer customer interaction may lead to many problems.

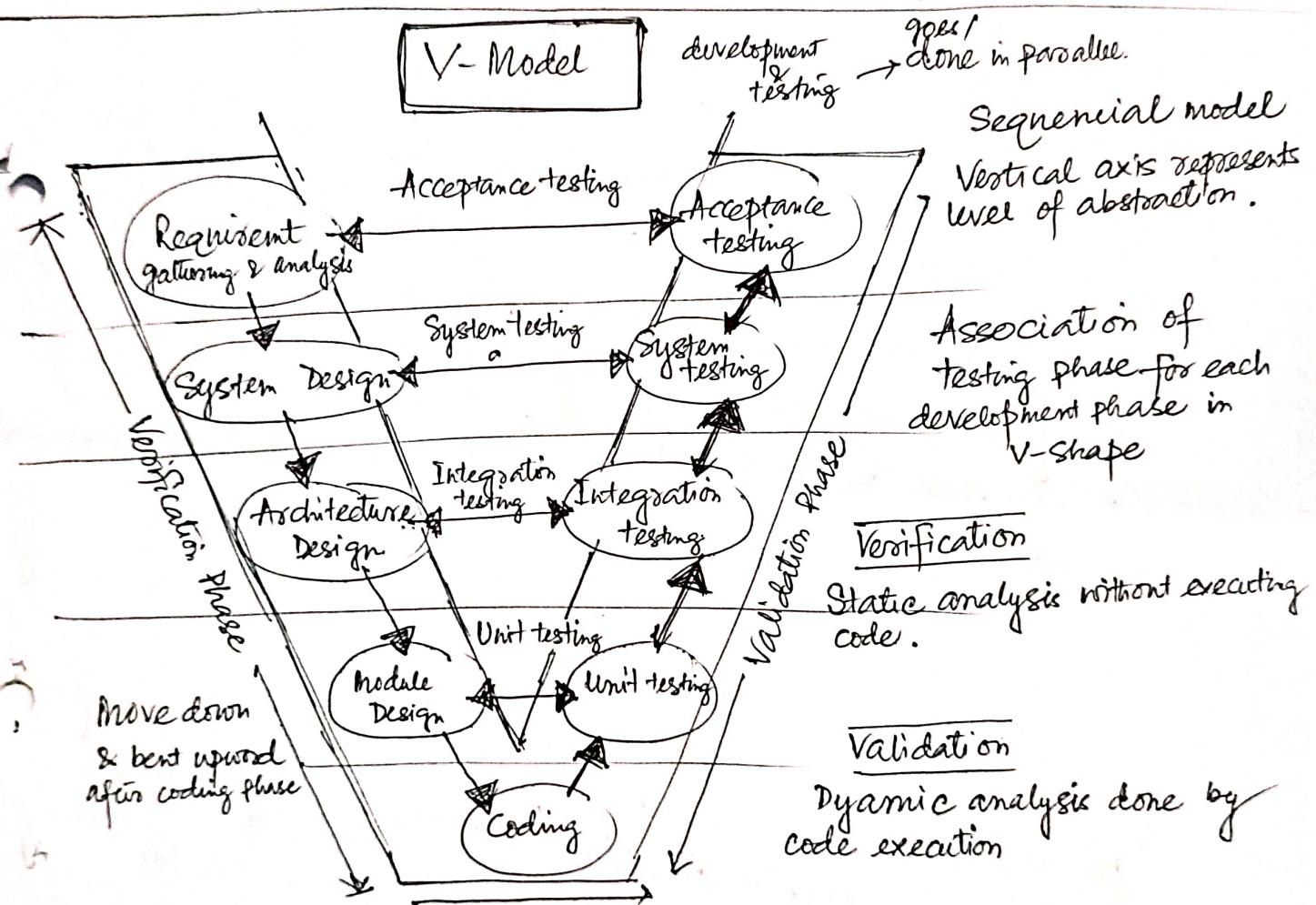
Preview

Applicable when

Major requirements are defined but the minor details may evolve when time goes.

New technologies are learned by development team.

Experienced development team.



System design Complete hardware & communication setup.

Architecture design

System design is broken down into modules with different functionalities
data transfer & communication between internal modules & outside the world.

Module design (Low level design (LLD))

Detail design of modules

Applicable when

Medium-sized ~~small~~ projects where requirements are clear
Most frequently in medical field

Advantage

Focuses on verification & validation in early life cycle,
— probability of building error free good quality product

Enables project management to track progress accurately.

Easy to manage due to rigidity of the model

Each phase is deliverable and can be reviewed.

Simple & understandable

Disadvantage

Does not support Iteration — difficult to go back & change functionality.

Does not handle concurrent phases / parallel execution

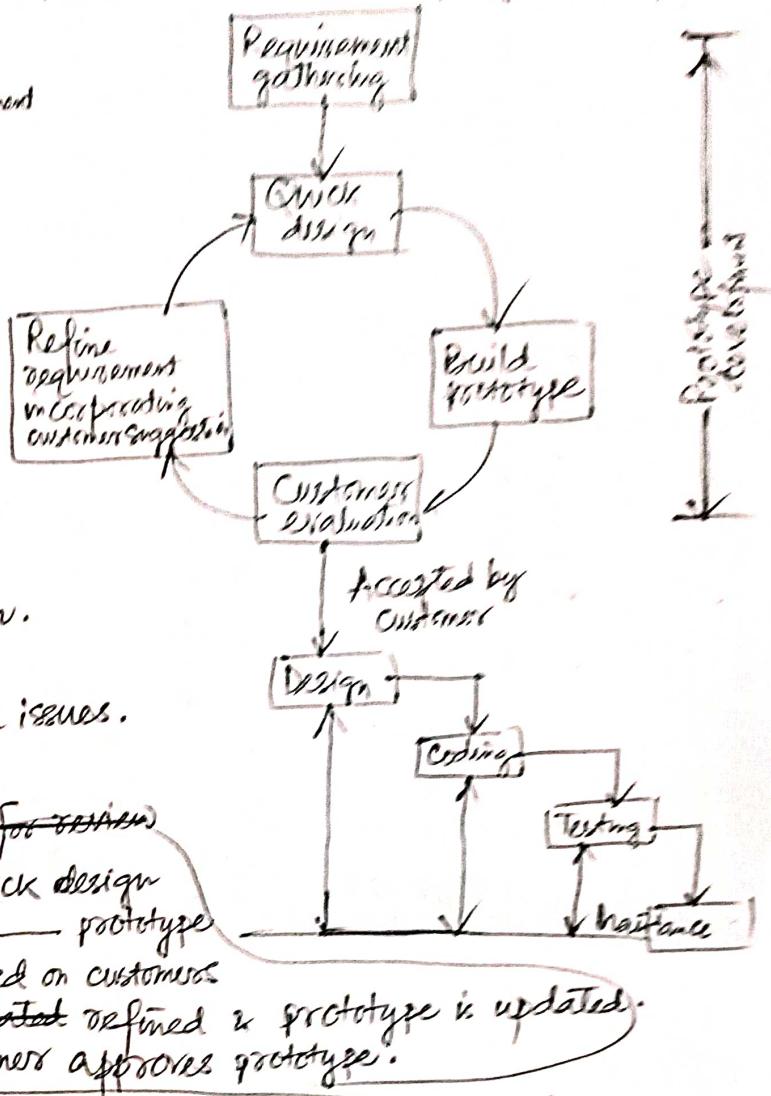
Not good for complex, long project.

PROTOTYPE Model

Prototyping model

Pilot production pathway
Serves as basis for system specification

- Before carrying out actual development prototype of a system is built.
- Prototype is like a toy that exhibits limited functionality compared to actual software.
- May contain shoddy implementation (e.g. table lookup instead of actual computation)
- Mechanism to understand what customer needs, rather than trying to imagine the working of system.
- Developed prototype can help engineer examine the technical issues.



~~Prototype is submitted to customer for review~~

Requirement is gathered — quick design is carried out to build prototype — prototype submitted to customers — based on customers suggestion ~~proto~~ requirement is updated defined & prototype is updated.

— cycle continues until customer approves prototype.

Experience gathered from prototype involve additional cost, reduces overall cost with unclear customers requirement, minimizes change request from customer (involving redesign cost)

Types of prototype:

(a) Rapid throwaway prototype

After customer feedback, the prototype is rejected & new prototype is built from scratch.

(b) Evolutionary prototype

Prototype is incrementally refined based on customers feedback

(c) Incremental prototype

Multiple ~~proto~~ prototypes are developed individually. These different prototypes are merged into single product.

Applicable when

Requirement is unclear

(d) Extreme prototype

Full user interface is developed (to draw attention) ~~very little~~, with limited functionality.

Advantage

- Errors detected at initial stage. Missing functionality
- Customer satisfaction exists as the customer feels the product at a very early stage.
- Encourages innovation & flexible design.
- Prototype may offer early training for future users of the software system → Working model of product
- Increase customer involvement in development.
- Reduces overall time & cost.

Confusion/
Errors/

Disadvantage

Too much ~~effort~~ effort.

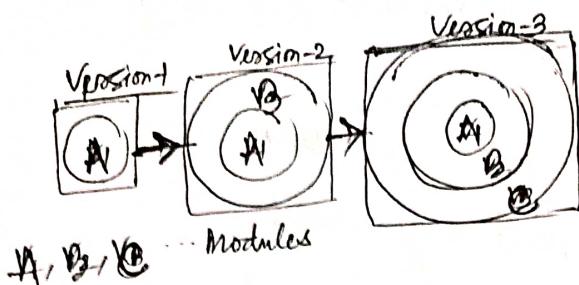
- Slow & time-taking process — Prototype building & approval
- Cost of developing prototype is total waste if it is thrown-off.
- Prototyping may encourage excessive change request.
- Poor documentation because requirements of customers are changing.

- [customer may get confused]*
- After seeing prototype customer may think that actual product will be delivered soon.
 - Customer may lose interest in final product if he is not happy with initial product.

- Developer who builds prototype quickly, may end up sub-standard development solution.

EVOLUTIONARY MODEL

SUCCESSIVE VERSION MODEL
INCREMENTAL MODEL



Software is broken down into several modules (Functional unit)

These modules are constructed incrementally

Initially core module of the system is developed.

Skeleton of the product

Each evolutionary version is developed using Iterative waterfall model

Each successive version of the product performs more useful task than the previous ones.

Useful task

User gets a chance to experiment with partially developed software much before the complete version is released.

Applicable when

Very large product where modules are implemented incrementally.

Customer prefers to receive product in increments

System is partitioned into stand alone units

Object oriented software design.

Rough requirement specification

Identify the core and other parts to be developed incrementally

Develop core part using iterative waterfall model

Collect customer feedback & modify requirements

Develop next identified features using Iterative waterfall model

Maintaince

Advantage

Customer's confidence increases as he gets ~~services~~ services/goods constantly from developers.

Reduces errors because core modules get tested thoroughly.

Chance to experiment with a partially developed software.

Does not need for large resources at a time.

Good when complete version is impossible due to tight market deadline.

Risk analysis is better.

Supports environmental change.

Disadvantage

Sometimes it is hard to divide problems into several versions that will be acceptable by the customers.

SPIRAL Model

Combination of Iterative development & linear development

Based on customer's evaluation software development enters next Iteration.

Single loop spiral model is waterfall model with high emphasis on risk analysis.

Each phase is represented by sector/quadrant

Number of loop depends on developer and project.

Radius of the spiral represents cost whereas angular dimension represents progress.

Why is spiral model called meta model?

It covers all other SDLC models. For example it incorporates stepwise approach like classical waterfall model. Existence of loop makes it iterative. waterfall model. At risk handling stage spiral model uses the approach of prototyping model. Iteration along the spiral can be considered as evolutionary model.

Advantage

Support for better risk handling — useful for mission-critical projects.

Good for large project

Flexibility in requirements/change of requirements is accommodated. Customer satisfaction.

User can see the system early

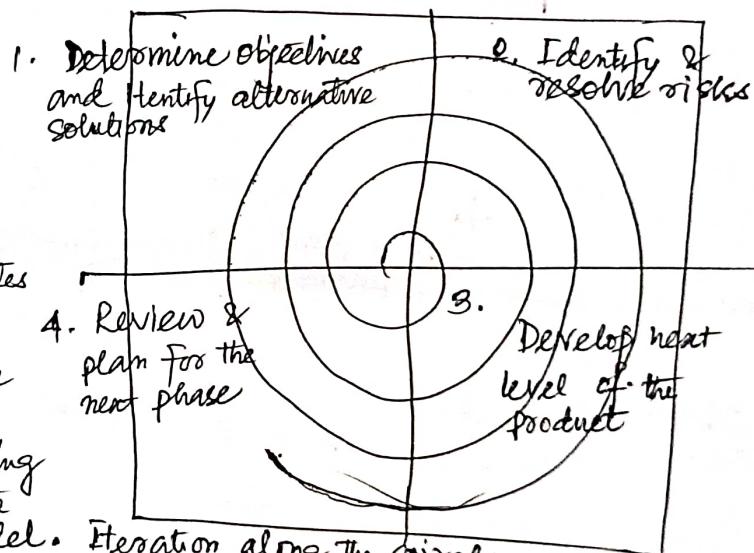
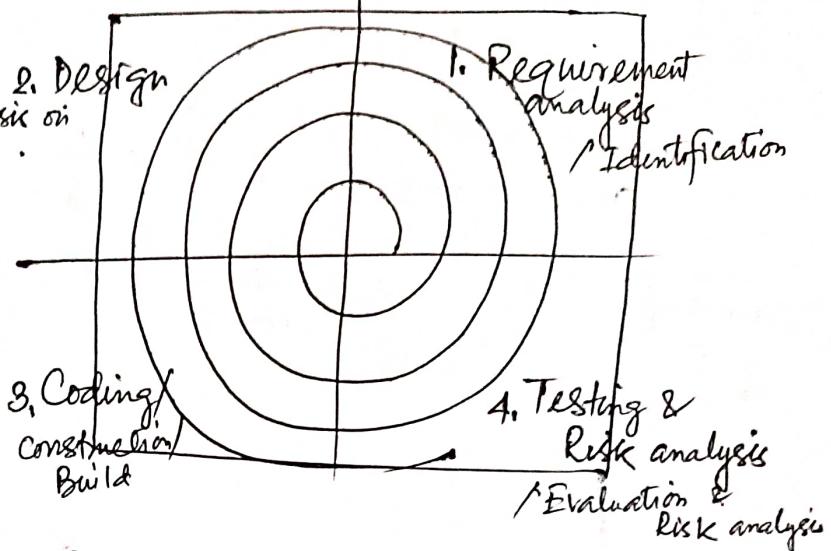
Extensive use of prototype

Incremental refinement & incremental release of product around the spiral

CYCLIC Model

META Model.

Learning with maturity



Disadvantage

Management
Most complex, among others SDLC model

Expensive — not suitable for small project.

Difficulty in time management as numbers of loops are not fixed always.

End of the project is not known early

Spiral may go indefinitely
Excessive documentation required.

Applicable when

Large & high budget

Requirement is unclear & complex

Changes may require at a time

1 Objective identification
& identify alternate
solutions

Review

EVALUATION

Objective setting

Identification of purpose for that cycle

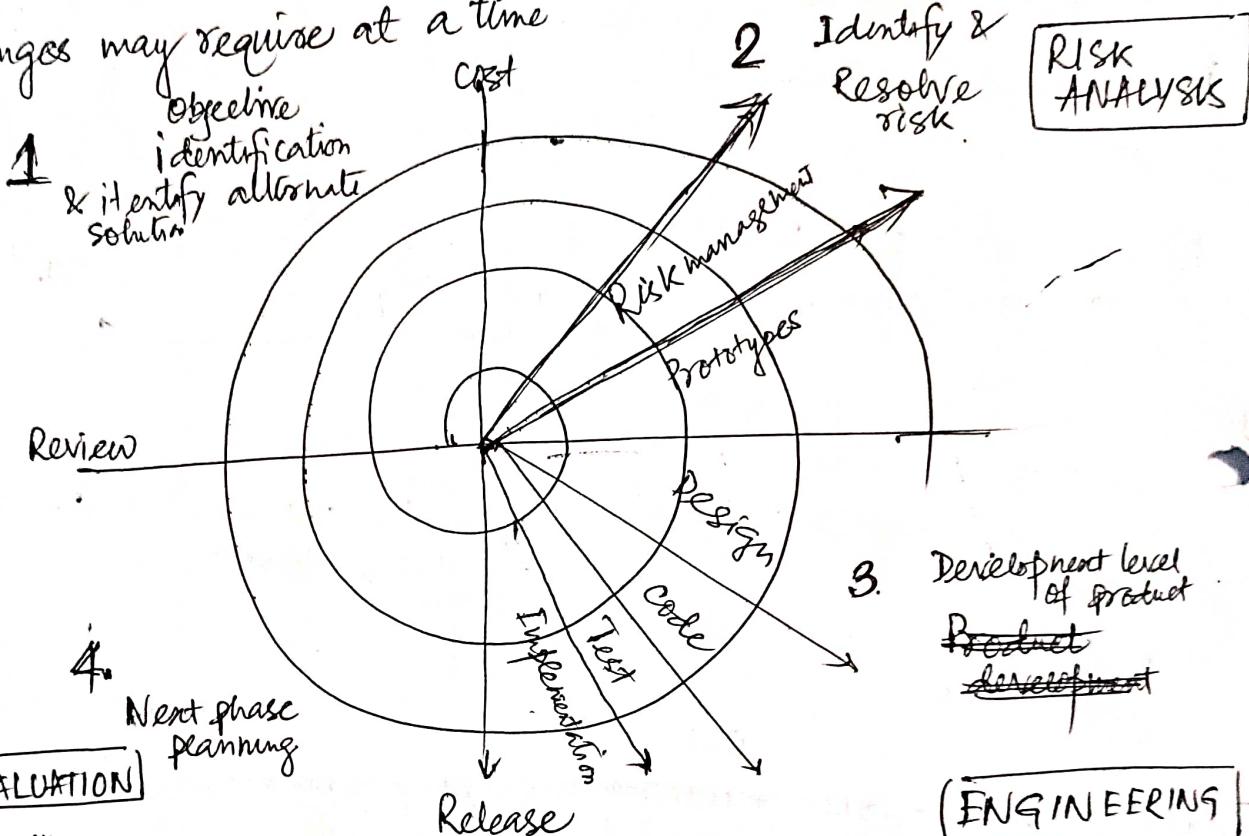
Risk assessment & reduction

Risk pattern

Development & validation

Planning

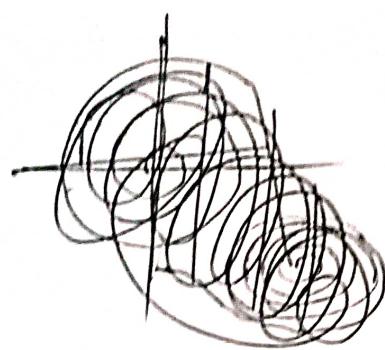
The project is reviewed — choice is made whether to continue spiral again.



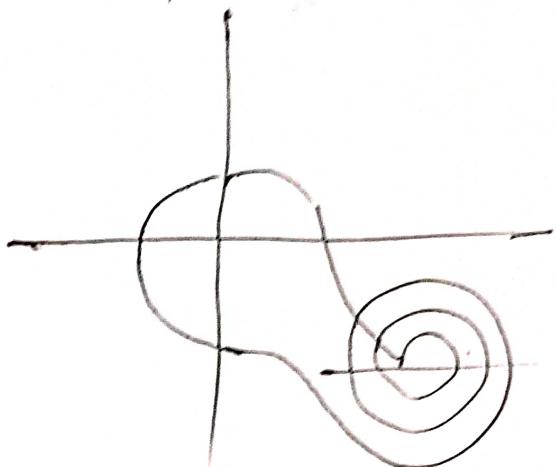
ENGINEERING

3. Development level
of product
~~Product
development~~

RISK
ANALYSIS



Spiral in one phase



f-319, Pohlmann Book

User interface design process

