Date

SOFTWARE DEVELOPMENT LIFECYCLE

- Software development life cycle (SDLC) refers to a methodology with clearly defined processes for creating high quality Software. SDLC methodology focuses on the following phases
  1. Planning 2. Requirement Analysis 3. Designing
  4. Implementation / Coding 5. Testing 6. Deployement / Maintance.

→ **Advantages**

(1) efficient with regard to costs
(2) efficacies in terms of time
(3) enhances teamwork and coordination, defines suitable roles for employees and increases workplace transperancy
(4) Minimal danger when project is implemented.

→ **Disadvantages**

(1) project may take longer and cost more if the planning is not done properly
(2) Correcting problems in code can occasionally take a long time and cause deadlines to be missed if there are many of them.

→ SDLC Models

(1) **Classic Water fall**

Feasibility Study (Planning) → Requirement Analysis → design → Coding → System testing → Maintanence

Date

SOFTWARE DEVELOPMENT LIFECYCLE

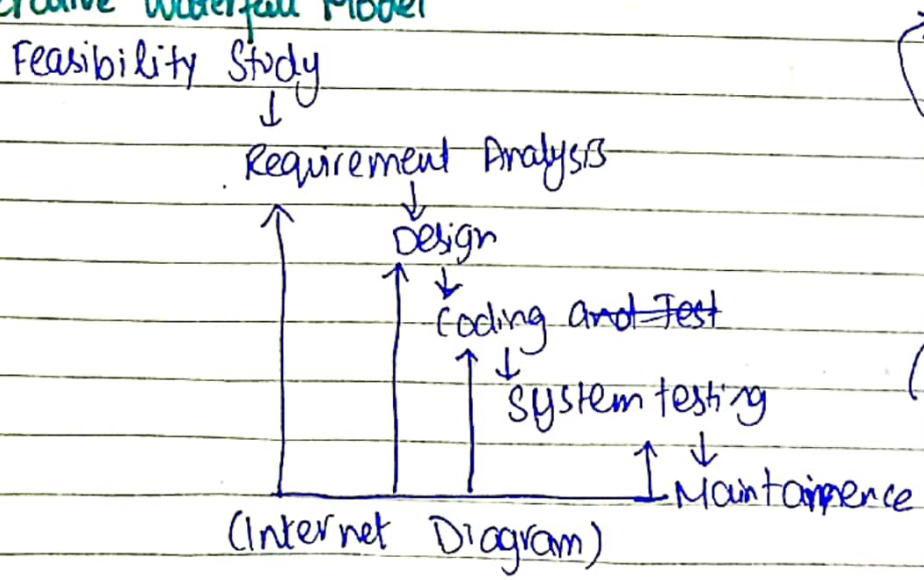
→ **Advantages**

↳ Simple and ~~eso~~ easy to understand
↳ Individual Processing (one at a time)
↳ Properly defined
↳ Clear Mile Stones
↳ Properly documented
↳ Reinforces good habits
↳ good for small projects

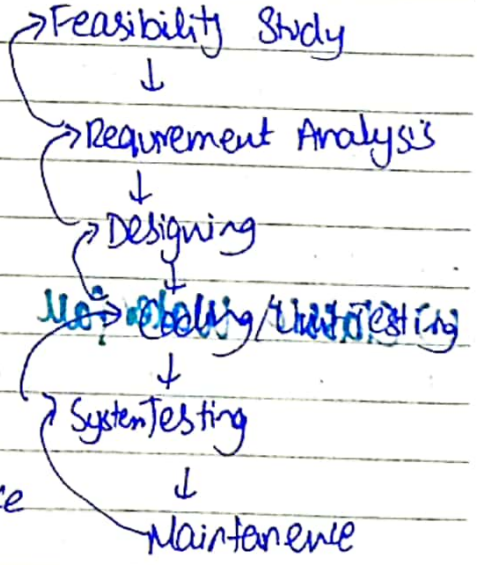
→ **Disadvantages**

↳ no feedback path
↳ difficult to accomodate changes
↳ no overlapping phase (new phase can started only previous is completed)
↳ limited ~~fel~~ flexibility
↳ late defect detection
↳ lengthy development cycle
↳ not suitable for complex projects

(2) Iterative Waterfall Model



(Internet Diagram)



Date

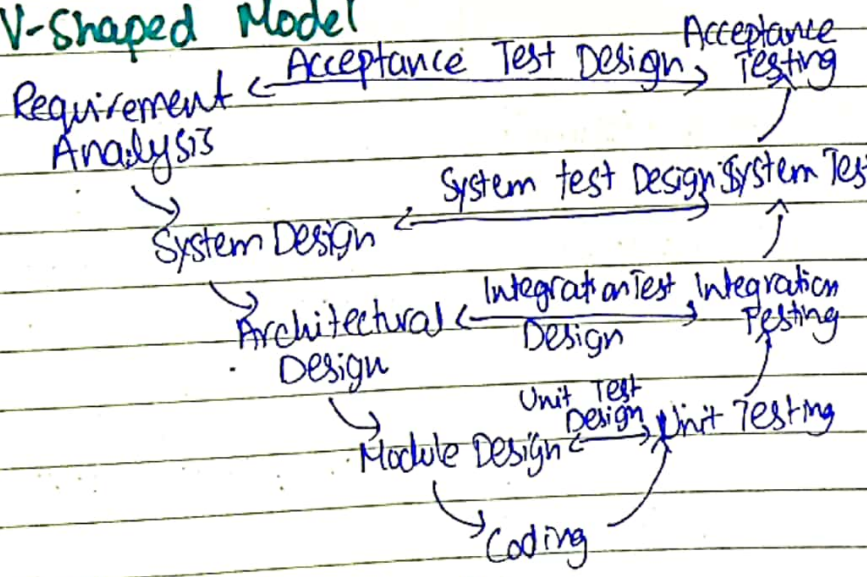
→ **Advantages**

↳ feedback path (allows error correction)
↳ Simple
↳ Well organised
↳ Cost effective
↳ Quality assurance
↳ Risk Reduction
↳ improved customer satisfaction
↳ Predictable Outcomes
↳ faster time to market
↳ Easy to Manage.

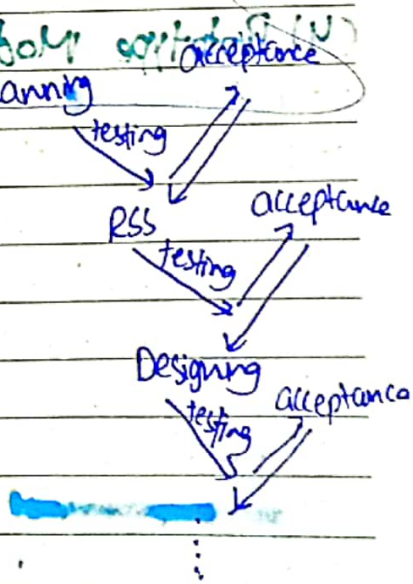
→ **Drawbacks**

↳ Difficult to incoperate changes (customer might make changes after some time but it is not possible to incorporate changes after development starts)
↳ ~~Later~~ Incremental delivery not supported (customers have to wait longer)
↳ Over lapping phases not supported
↳ Risk handling not supported
↳ Limited Customer Interaction

(3) V-Shaped Model



(Internet Diagram)



and so on

(Sir's Diagram)

Date

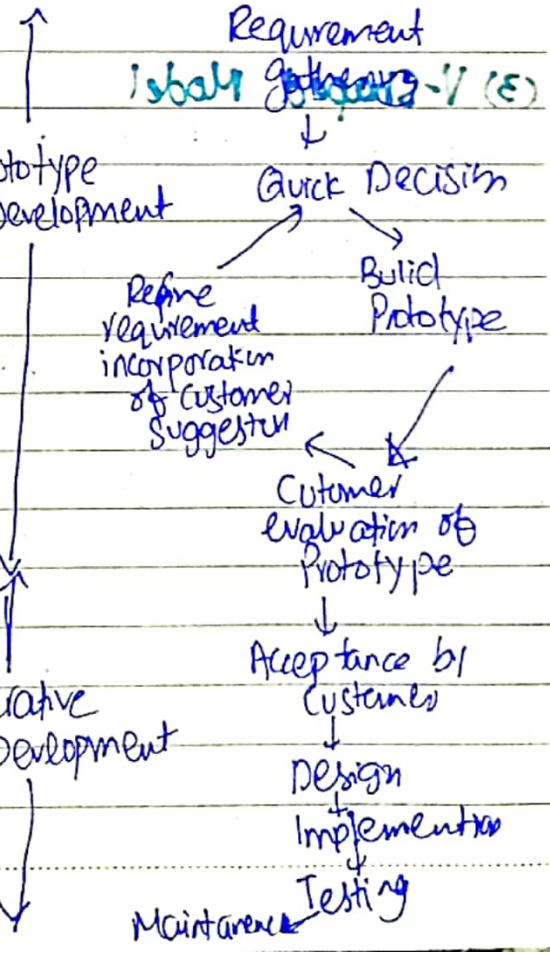
→ **Advantages**
- ↳ easy to understand
- ↳ testing methods (eg planning, test designing) happens well before coding
- ↳ Saves time
- ↳ avoids downward flow of errors
- ↳ works well for small plans
- ↳ each phase tracked side by side

→ **Disadvantages**
- ↳ very rigid and least flexible
- ↳ not good for complex project
- ↳ Software is developed during implementation stage so no early prototypes of software are produced
- ↳ if any changes happens in midway, then test doucments along with required documents have to be updated
- ↳ no feedback ↳ no changes

(4) Prototype Model

→ **Advantages**
- ↳ Reduce risk of incorrect user requirement
- ↳ good where requirement are changing
- ↳ regular visible process aids management
- ↳ Support early product marketing
- ↳ Reduce maintenance cost
- ↳ errors can be ~~debu~~ detected eavlier

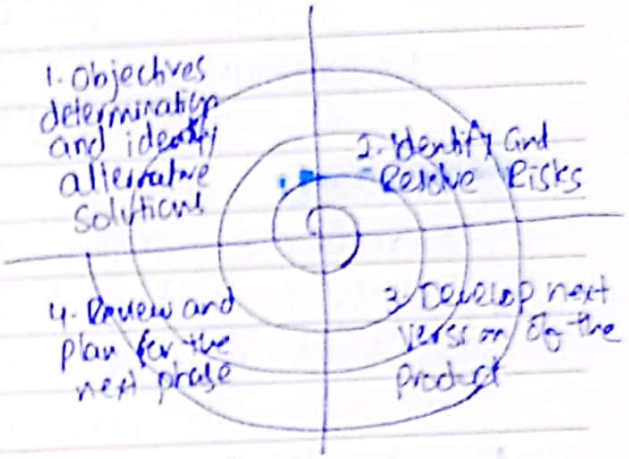
→ **Disadvantages**
- ↳ unstable / badly implement prototype often becomes the final product
- ↳ Requires extensive customer collaboration
- ↳ project might last long
- ↳ time consuming



Date

↳ Special prototype tools required which are highly user expensive.

(5) Spiral Model



→ **Advantages**

↳ Risk handeling
↳ good for large projects
↳ flexibility in requirements
↳ Customer Satisfaction
↳ Iterative and Incremental Approach
↳ Emphasis on risk management
↳ Improved Communication ↳ Improved quality

→ **Disadvantages**

↳ Complex ↳ expensive ↳ Complexity ↳ time Consuming
↳ Resource Intensive ↳ difficulty in time management
↳ too much dependability on Risk analysis.

(6) Scrum Model

- uses agile Methodology
- has three roles

① Scrum team (individual working to create project)
② Scrum master (make sure team works properly)
③ Product Owner (aka 'Customer')

→ **Advantages**

↳ gives freedom to developers
↳ high priority requirements are addressed first
↳ shortens times (product owner is part of team)
↳ reviewing each stage before moving to next - less errors

→ **Disadvantages**

↳ works well in small teams only
↳ losing team member can hurt the progress

Date

(7) Incremental Model

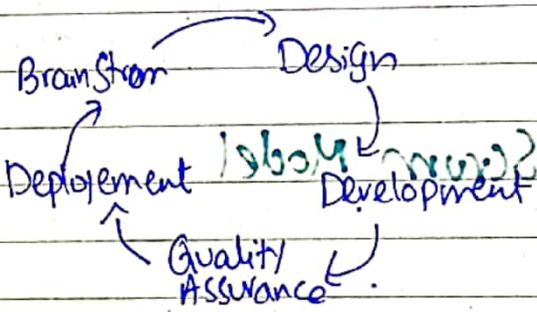
Requirements → Build 1 → Design and Development → Testing → Implementation
Requirements → Build 2 → Design & Development → Testing → Implementation
Requirements → Build 3 → Design & Development → Testing → Implementation

→ **Advantages**
- ↳ errors are easy to recognize
- ↳ easy to test and debug
- ↳ Client gets important functionality early
- ↳ Simple to manage risk
- ↳ more flexible

→ **Disadvantages**
- ↳ need for good planning
- ↳ total cost is high
- ↳ well defined module interfaces are needed.

(8) Agile Model

- breaks tasks into smaller iterations (does not involve long term planning)



→ **Advantages**
- ↳ Frequent Delivery
- ↳ face to face communication with clients
- ↳ efficient design fulfills the business requirement
- ↳ Any Anytime changes are acceptable
- ↳ reduces total development time

→ **Disadvantages**
- ↳ due to Shortage of formal documents, creates confusion
- ↳ " " " proper documentation, maintenance of finished project is difficult.