Software :- Software is a collection of programs, data, and documentation that tells a computer what to do and how to do.

(Hardware is the body, software is the soul/brain).

System software :- Operating system.

Application software :- MS word, MATLAB, whatsapp.

Embedded software :- Software inside washing machines.

Web and application :- websites.

Software engineering is the systematic, disciplied, and measurable approach to the development, ~~measurement~~, operation and maintainance of software.

Develop a software for adding two numbers

```
a = int(input("Enter first no"))
                              ))
b = int(
print("Sum =", a+b)
```

↑

this can be a software but engineering aspect has not been incorporated yet.

→ No design, no testing plan, no consideration for error

```
def add_num(a,b):
    return a+b
try:
    x = float(input("Enter
                ... :"))
    y = float(input("...
    result = add_num(x,y)
    print("Sum=", result)
except ValueError:
    print("Invalid Input.
          Enter numbers")
```

∴ Software engineering is not about how big or small the software is, but how systematically it is developed.

Software Development Life Cycle (SDLC).

Need | Divid

Software development Life Cycle is a structured framework used by software organizations to design, develop, and test high quality software. It defines the ~~software~~ entire process of software production, from the initial idea to the final development and maintainance.

→ Without structured SDLC, software development becomes chaotic.

Planning

- The software should take 2 nos as input
- → Software should display their sum
- ↳ Handle invalid

Defining

Plan how the software should work
Plan architecture, Logic

Building
~~Designing~~
Implementation/ Coding

Designing Architecture

↳ Technical blueprint
↳ HLD : Defines architecture, database design, and relationship between modules
↳ Low Level Design : Defines logic of individual component

Testing

Unit testing : testing individual sections
Integration testing : ensure module is working together
System testing – entire appln flow
User Acceptance Test
Verify it meets business needs

Deployment :- Released to end users.

need for SDLC.

→ Divide-and-conqur streategy.
→ With Software Development Life Cycle, the process of software design is divided into small parts, which makes the problem more understandable and easive to solve.

Waterfall model

→ Follows a traditional, linear, phase-by-phase approach; where each phase must be completed before moving to the next phase.
→ Doesnot allows backtracking and permits only minimal changes once a phase is completed.

Requirements → clearly understanding customer requirements and documenting it.
Software Requirement specification one formal agreement betn customer & development team.

↓

Design → Requirements from SRC are converted into system design.

↓

Development → Converting the design into actual working code

↓

Testing → Alpha testing → developers team
Beta testing → selected users
Acceptance testing → performed by Customers

↓

Deploying

↓

Maintainance.

Predictive: enhances features based on user neede

Corrective Maintainance: fixes errors found after release.
Adaptive Maintance: Updates software to work in environment

Advantages

→ Simple and easy
→ easy to implement
→ Good for small projects.

Disadvantages

Not flexible to change

Testing happens Late

The waterfall model assumes that requirements won't change — which is rarely ~~to~~ true in real life.

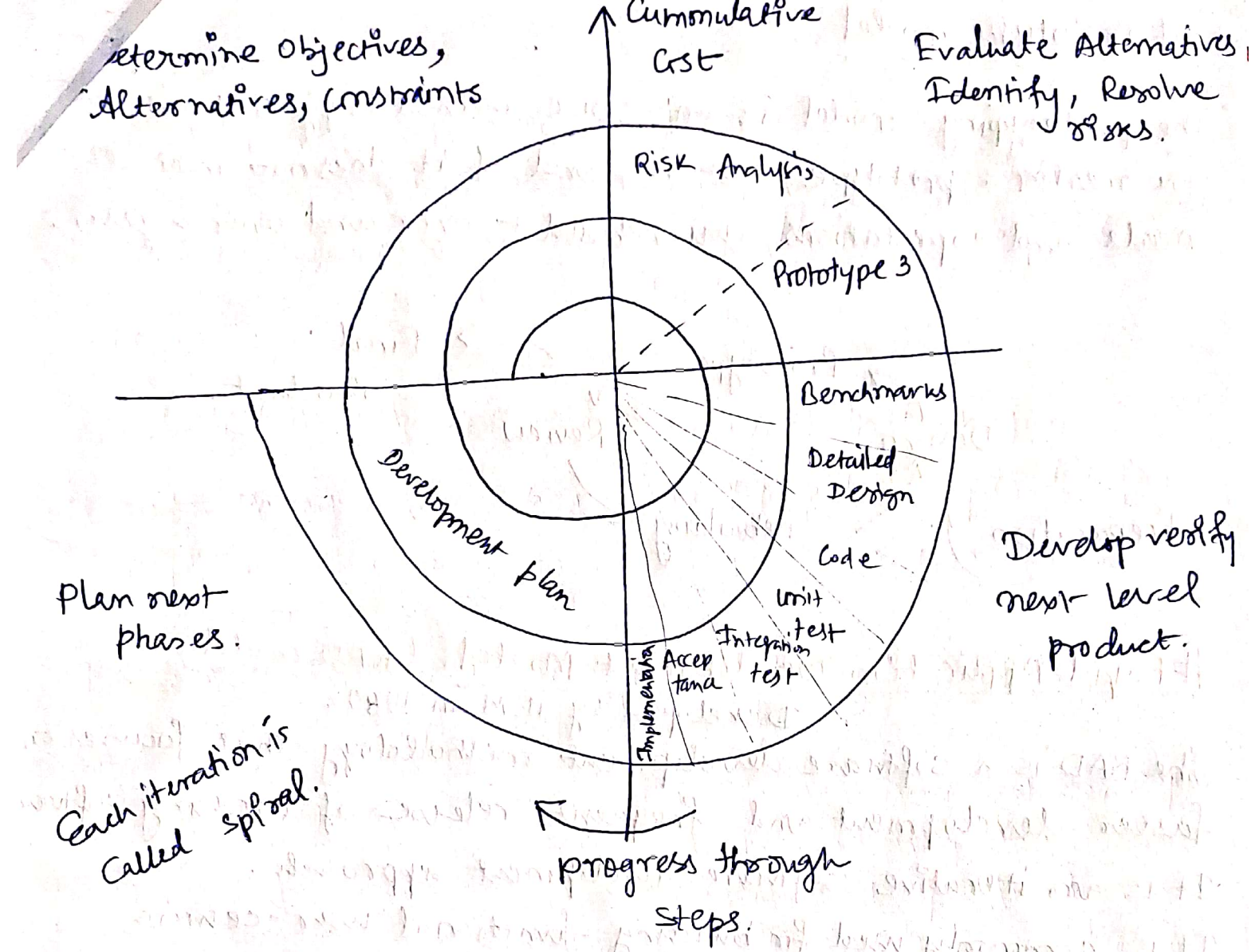
Iterative Waterfall model

It combines the structured, linear phases of the traditional waterfall model with cyclic, iterative, and feedback loops.

Strength :- Sequential structure of waterfall model
Flexibility of iterative and feedback.

eg. Payroll management system
Salary calculations are generally fixed
Iterations due to tax rule update
~~Allowab~~ Allowance modif^{n}.

Iterative waterfall is what organisation wants when they want structured flow with scope of improvements.



Stage 1 :- Team members try to gather the project objectives, requirements, alternative designs etc.

Stage 2 :- Risks are possible conditions or events that prevent the development team from its goals. Wide range of risks are there which includes trivial and fatal. Prioritize risks according to importance.

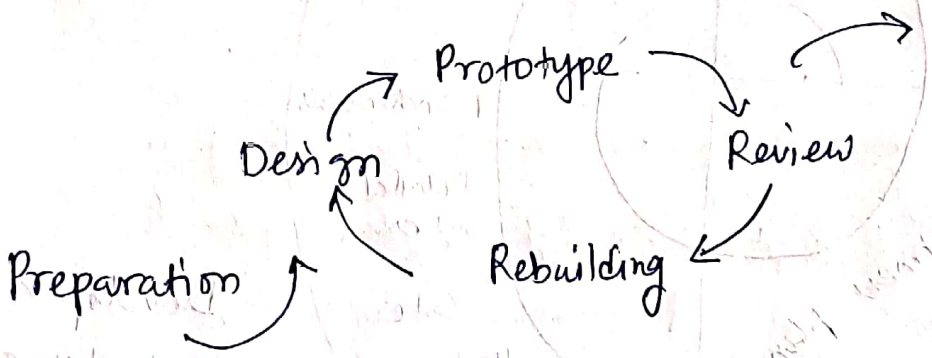
Step 3 :- During first spiral, Proof of concept is develpm
Later on an working version of the product is build.

Step 4 :- Evaluation of the output of the product

Spiral model is called meta model because it uses both waterfall and prototype models. But it is important to note that

Prototype model

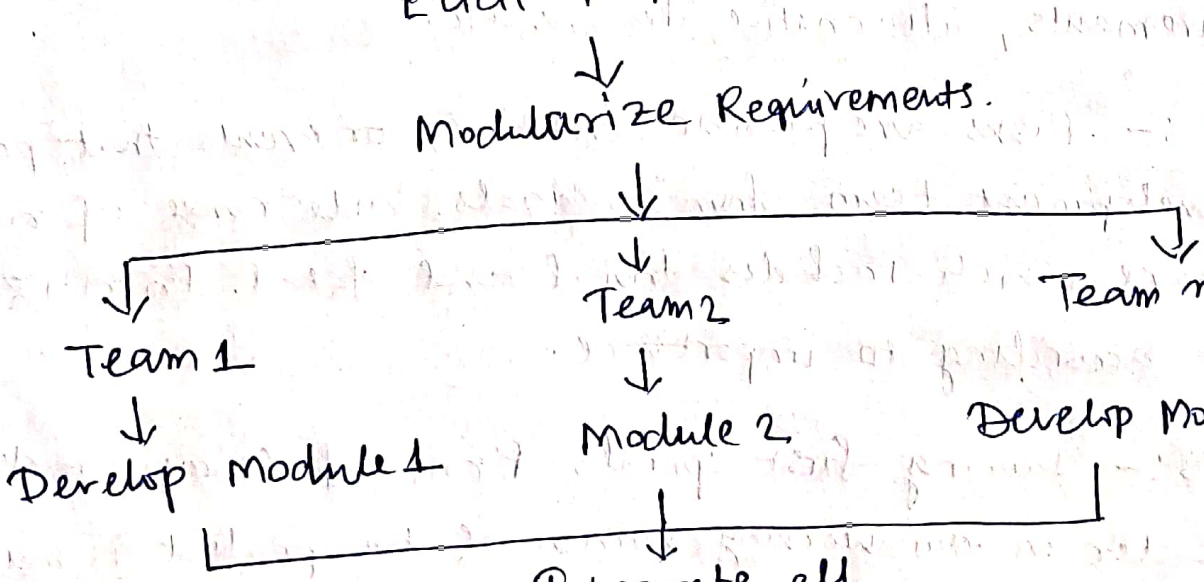
The prototyping model is an SDLC methodology that implies you master a prototype, test it, and if it doesnot meet the needs and expectations, you rebuild it over and over again.



Rapid Application Development Model. (RAD model)

Developed by IBM in 1980s

The RAD is a software development methodology that focuses on faster development and frequent releases of working software
It is an iterative software development approach.
It is commonly used for building functional user-centric applications like HR-system, mobile apps, and e-commerce.



1. Analyze
2. Design
3. Code
4. Test

comprises of four different phases:-

→ Requirement Planning - In this phase, developers and users meet to discuss what software should do. All requirements are gathered, analyzed, and a basic plan for the project is created.

→ User Design:- Developers creates simple prototype or models of the software. User try them, and gives feedback.

→ Construction:- The actual software is developed in this phase.

→ Cutover:- The complete software is tested in this phase. Different modules are integrated.

When to use RAD models?

- Well understood Requirements
- Time-sensitive projects.
- Small to medium sized projects.
- High - user involvement.
- Prototyping.

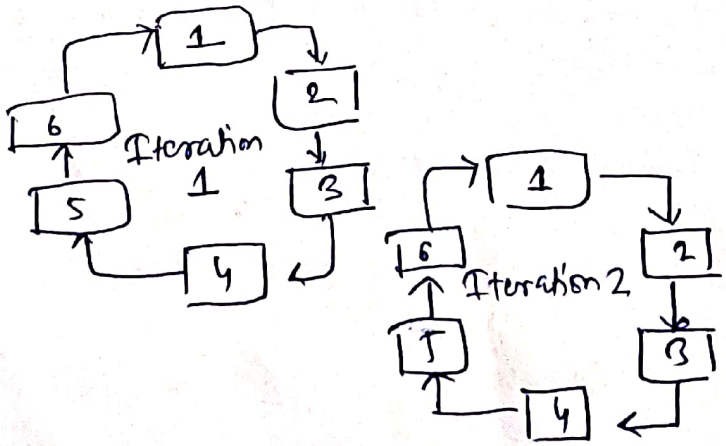
Agile Development Models

Agile SDLC model is a combination of iterative and increment processes models with focus on product adaptibility and consumer satisfaction by rapid delievery of working software product.

Agile methods breaks the product into small incremental builds these builds are provided in iterations.

Steps in Agile Models

1. Requirement Gathering – Development team must gather the requirements by interaction with the customers.
2. Design the requirements :- The development team will use user-flow diagram or high-level UML diagrams to show the working of new features and show how they will apply to the existing software.
3. Construction :- Development team starts working on the project.
4. Testing :- Involves unit test, Integration test, & system testing which help in agile development models.
5. Deployment :- Development team deploy the working project to end users.
6. Feedback – Team receives feedback from customers.



When to use the Agile Model :-

→ Agile believes that the need of end users are always changing.
→ In Agile, unlike waterfall very few planning is required.
→ New changes can be implemented at very low cost.

Understanding Agile Frameworks

Agile is not just a methodology, its a mindset. It emphasizes flexibility, collaboration, and delievering values to the customer. Breaking projects into bite-sized pieces, Agile methodologies empower teams.

Scrum: Structure and sprints. Scrum is a time-boxed iterative framework used to build product incrementally. Its a go-to Agile framework for teams seeking structure and regular delieverables. Its build upon short focussed work periods called sprints, typically lasting 2-4 weeks.

During each sprints, the team commits to completing a set of tasks pulled from a prioritized backlogs.

One of scrums ~~best~~ greatest strength is its clarity in roles.

Scrum master	Product Owner	Development Team
ensure scrum is followed	Definer what to build	builds the product.

Best for : → Projects with changing requirements
→ Teams that needs structure + agility.

Kanban

Kanban provides a highly visual approach to project management by mapping out work items on a kanban board. Tasks are represented as cards that move through different stages such as to-do, In progress, Done.

What sets Kanban apart from Scrum is its lack of time-box sprints. Instead it focuses on continuous delievery, allowing team to more tasks through the work flow as they are completed.

unlike scrum it doesn't requires predefined roles or daily meetings.

Lean

Originally developed in manufacturing, Lean focuses on maximizing value by eliminating waste.

Lean principles includes reducing lead times, improving the quality of outputs and continuously evaluating processes.

The idea is to deliever the highest value in shortest possible time by striping away anything unnecessary.

XP (Extreme Programming)

Tailor made for software development teams that prioritize high quality code and frequent releases.

XP encourages Test Driven Development (TDD) where tests are written before code to catch issues easly.

XP also emphasizes high level communication.

Framework	Core focus	Team Structure	Workflow	Best suited for	Challenges.
1. Scrum	Iterative develop^{t}, time-box sprints	Defined roles Scrum Master Product Owner Development team	Sprint-based (2-4 weeks)	Teams with clear goal and stable project scope.	Can feel regid in dynamic environment.
2. Kanban	Continuous delievery, visual management.	No predefined roles	Continuous flow	Teams needing flexibility, support teams	Risk of undefined deadlines without control.
3. ~~Lean~~ XP	~~Efficient waste elimination~~ Code quality, frequent releases	Technical Roles: Developers, Testers	Continuous releases, TDD, Pair programming	Software team prioritizing quality and frequent updates	overhead for non-technica environments
4. Lean	Efficient waste elimination	Varies (often flexible)	Continuous improvement	Large enterprises focussed on eliminating inefficiencies	Requires continuous evaluation of processes