Models

Waterfall model — classical
                 — Iterative

Prototyping

Spiral model

Classical waterfall model

- sequential

Feasibility study → Req analysis & spec → Design → coding & unit testing → Integration → maintenance
(technically & finance)

maintence - ~~error~~ corrective (errors)
- adaptive (changes)
- perfective (↑ performance)

feasibility study

- development of overall understanding of prblm
- formulation of various possible strategies for solving the prblm.
- evaluate each strategy and finalize one strategy

Req- Analysis & spec

- requirement gathering and analysis
- ^ req. specification (documents)
output - (SRS) SRS - software Requirement Specification
agreement or contract b/w customer & Dev team

Design (solution starts)

- High level design
  - architecture, entire view
  - software architecture
- Low level design
  - detailed design (Algorithms + data structure)

output of design is design document

Coding & unit design

- team can decide - procedural lang
- OOP

Individual modules are testing in unit testing

- white box
- Black box

After testing if failure occure we can't backtrack or go back to correct previous steps.

Integration & Systema testing

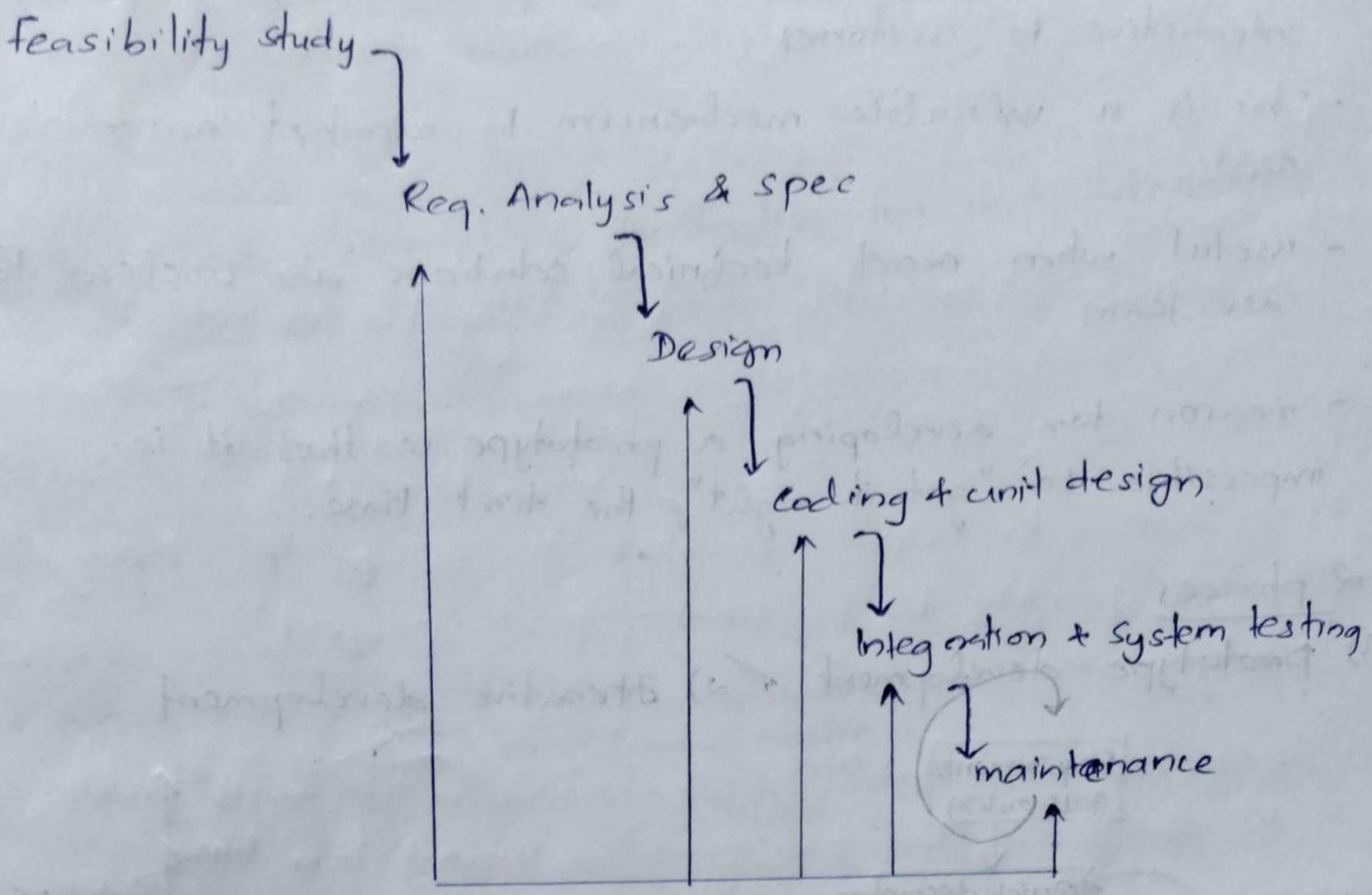
Integration modules
α, β testing

Problems in Classical Waterfall modal

- There is no feedback part
- Difficult to accomodate change request
- Inefficient error correction (at system testing difficult to correcting)
- there is no overlapping of phases

To rectify it iterative waterfall model is used.

Iterative Water fall modal.



- Simple & indutive

7/4/22

prototyping model

- extension of waterfall model
- building od prototype to gathering requirements - throw/remove
- rapid prototypang is used when software tools are used for prototype construction.
- limited func capability, low reliability
- we go for prototyping model for large user interface

- GUI is an application (Graphical User Interface)

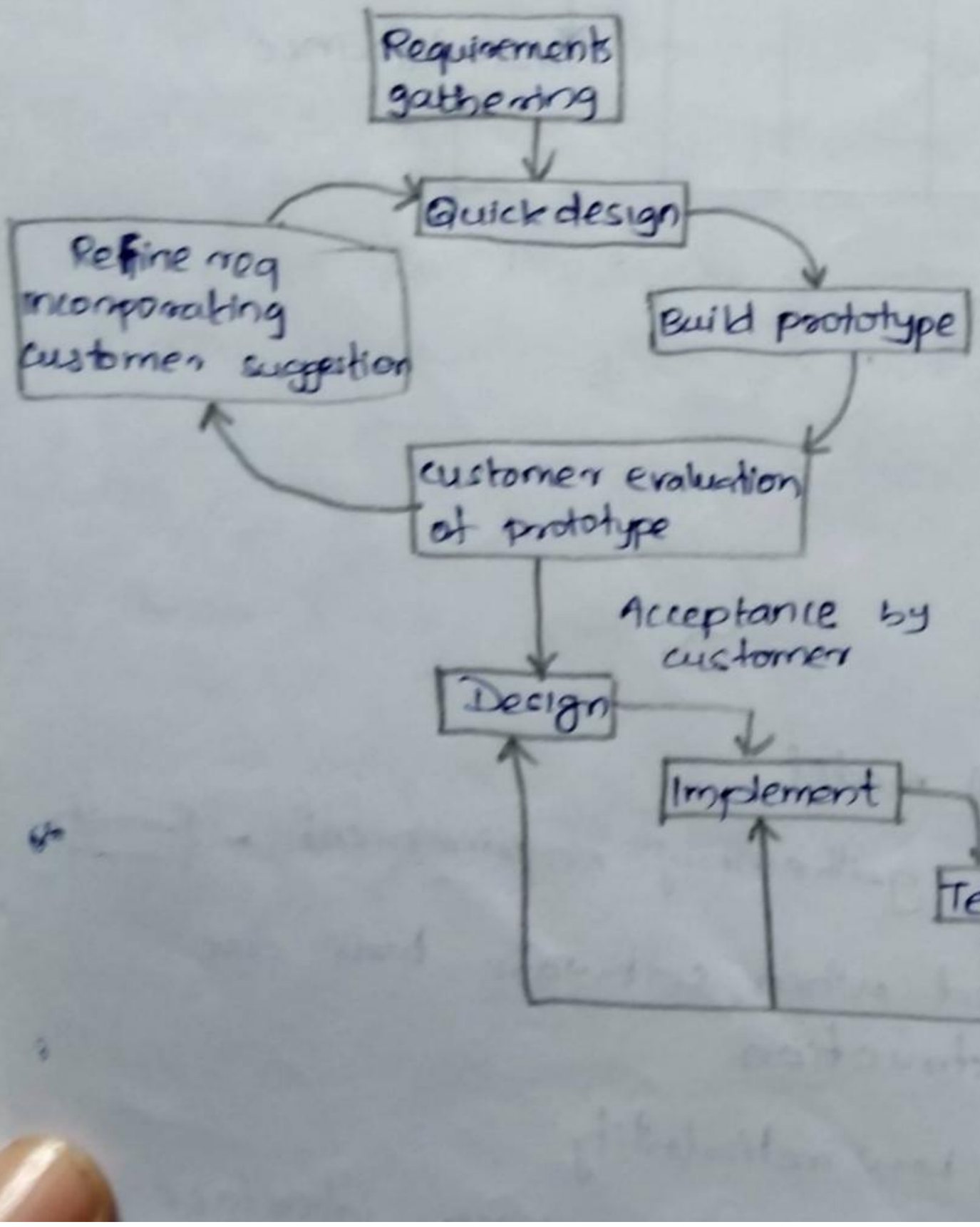
Adv
- Easier to illustrate input data formats, messages, report, interactive to customer
- This is a valuable mechanism to understand customers need

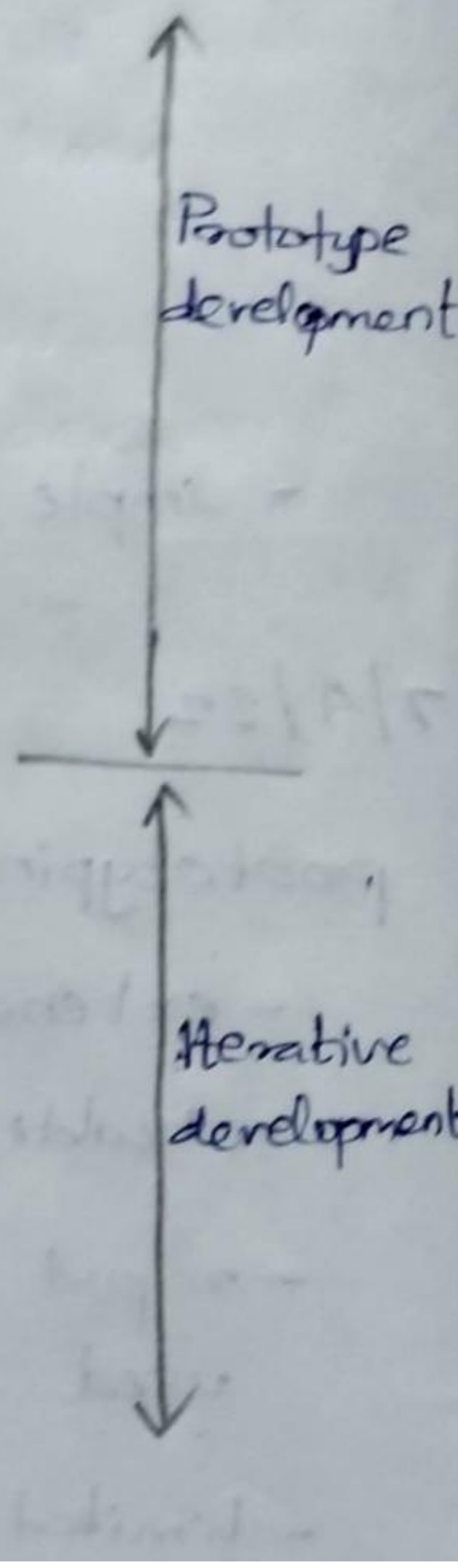
- useful when exact technical solutions are unclear to dev team

- reason for developing a prototype is that it is impossible to "get it right" for the first time.

2 phases

1) Prototype development 2) Iterative development





12/4/22

Strength of prototyping model

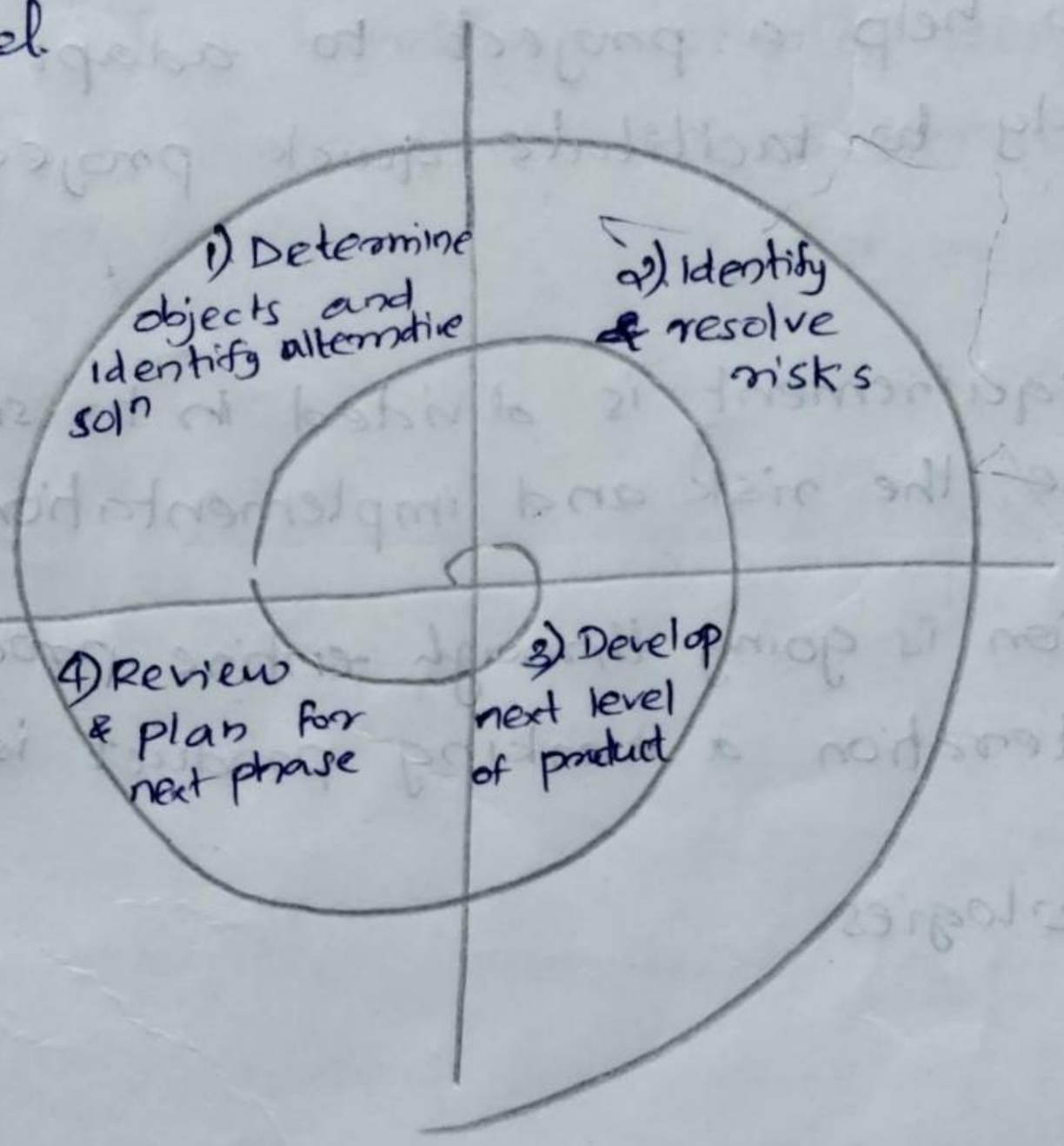
- most suitable for projects have some technical issues and requirement risks.

defect

- prototyping model is ineffective for risks identified later during the development cycle.

- High cost

Spiral Model



- for R&D projects
- In each loop, can develop a prototype and evaluate (risk analalysis in all phases)
- handle risks ~~aft~~ after the project started.

Risk

- any circumstance that might hamper the successful completion of a software project.
- This risk can be resolved by building a prototype

defect

Not suitable for outsource projects

Agile models

take one user story/case and implement it.

1) Scrum
2) Extreme progsamming

- Customer interaction is higher
- Can change requirement at any point of time
- Change request quickly.
- rapid shift from development of software products to development of customised software.
- Increased emphasis & scope for reuse

Principles

1. Working software over documentation
2. frequent ~~documentation~~ delivery

Here listnening & feedback is there in Agile model.

19/4/22

Requirement Analysis & Specification

1. Req. Gathering & Analysis
   - Gathering
   - Analysis (inconsistent req, incompleteness, anomaly, Ambiguity)

   Req elicitation

2. Req. specification → (SRS doc o/p)

Req
- functional (manual)
- non fun (quality attributes) scalability, user interface, portability

- understand exact req of customer and document it. without a clear understanding of problm and proper documentation of same, it is impossible to develop a software
- req and spec phase ends when req Spe doc has been developed & reviewed.
- system analysts - customise the req

Chara of SRC doc
- should be understandable, consistent, unambiguous and complete
- Once doc is over it is serve as contract

Availability - of working model is help to ~~gath~~ req gathering.

analyst . study i/p o/p forms and understand o/p forms

Gathering req through

- Studying existing doc → avail doc req system to be de
  ↳ customer provide SOP (statement of purpose)
- Interview - identify users & determine req - Delphi te
- Surveys
- task analysis → A service supported by a software is a ta
  task → steps → analysis
- scenario analysis
- form analysis

20/4/22

users of SRS doc - soft devps, Test engg, proje mngr

Why SRS is imp?

using good SRS doc decrease ~~de~~ reduce reworks.

→ provide a basis estimating cost

→ provide a baseline for validation (product) & verification (process)

→ facilitates future extension.

Chara of good SRS

- Concise, unambiguous, consistent
- Implementation - independent (no need to mention code or lang)
- specify as black box (only need external behaviour of system)

SRS also known as black box specification of software

- Traceable - from req we have to trace
  SRS ↔ code ↔ design vicevesa

- Modifiable (change requirements)
- Identification of response to undesired events
- verifiable (using functionality, it must be possible to design test cases)

Bad src

- over specification
- wishful thinking
- Noise

functional req	non-fn req
- i/o o/p fn A. high-level fn is one using which the user can get some useful piece of work done.	- concerning external interfaces - user interface - maintability - portability (diff os should be work same)

classification of req - easy to manage

Identify functional req

1) identify users (end users)
2) identify services expected from sftwr

document fn req (R - req)

i/p → process → o/p

R.1 → main fn

Description : ~~xxxxxx~~ about fn

R.1.1 → sub activity

Input : —

output : —

R.1.2 →

"

"

26/4/22

Module - 2 - Software Design (S/W archi)

1) High level design → can know diff modules, archi of mods, (Tree str) → 2) control flows/ communication

2) Detailed design → 3) data b/w 2 mods

{SRS → design process → Designed document} * Each mod have meaningful name

↳ 1) Data str ~~algo~~ } for each mod

2) Algorithms (efficiency of in time & space)

- Design also an iterative process.