3 Quality plan → Quality plans are important for ensuring that the final product is of high Quality. The project Quality plan Identifies all the V & V activities that have to be performed at different stages in the development and how they are to be performed. Much of the Quality plan revolves around testing and reviews. effectiveness of reviews depends on how they are conducted. One particular process of conducting reviews called inspections.

4) Risk management → A software project is a complex undertaking. Unforseen events may have an adverse impact on a projects cost, schedule, or quality. Risk management is an attempt to minimize the chances of failure caused by unplanned events. The aim of Risk management is not to avoid getting into projects that have risks but to minimize the impact of risks in the projects that are Undertaken. Risk is a probabilistic event - it may or may not occur, but if they do occur, they have a negative impact on the project. To meet project goals even Under the presence of Risks requires proper risk management. Risk management requires that risks be identified, analyzed and prioritized. Then risks mitigation plans are made and performed to minimize the effect of highest priority Risks

5) project Monitoring plan →
for the plan to be successfully implemented it is essential that the project be monitored carefully. Activity level monitoring, status reports and milestone analysis are the mechanisms that are often used. for analysis and reports, the actual effort, schedule, defects and size should be measured. With these measurements

it is possible to monitor the performance of a project with respect to its plan. And based on this monitoring, actions can be taken to correct the course of execution, if the need arises. Overall project planning & monitoring lays out the path the project should follow in order to achieve the project objectives. with proper monitoring in place, these situations can be identified and plans changed accordingly. Basic project planning principles and techniques can be used for plan modification also

6) [Project Scheduling →

Once the effort is estimated, various schedules are possible, depending on number of resources put on the project

Schedules → functions of efforts, Project initiation regular activities, inception phase, elaboration phase with in different iterations, Construction phase with different iterations transition phase, back end work. involve all work breakdown structure.

Staffing → team structure, team organization, team development activities (training, knowledge sharing tasks etc), personnel planning, etc

Overall schedule is determined using a model, and then adjusted to meet the project needs and constraints. The detailed schedule is one in which the tasks are broken into smaller, schedulable tasks and then assigned to specific team members, while preserving the overall schedule and effort estimates. The detailed schedule is the most live document of project planning as it lists the tasks that have to be done: any changes in the project plan must be reflected

6 project Scheduling & Staffing →
Once effort is estimated, various schedules (or project duration) are possible, depending on the number of resources (people) put on the project. A schedule cannot be simply obtained from the overall effort estimate by deciding on average staff size and then determining the total time requirement by dividing the total effort by the average staff size. In project, the scheduling activity can be broken into two sub activities: determining the overall schedule (the project duration) with major milestones, and developing the detailed schedule of the various tasks.

7) Software Configuration management plan → The goal of Configuration management is to control the changes that take place during the project. The configuration management plan identifies the configuration items which will be controlled, and specifies the procedures to accomplish this and how access is to be controlled. The activities in this stage include the following
→ Identify configuration items, including customer-supplied and purchased items
→ Define a naming scheme for configuration items
→ Define the directory structure needed for CM
→ Define version management procedures and methods for tracking changes to configuration items
→ Define access restrictions
→ Define change control procedure
→ Identify and define the responsibility of CC.
→ Identify points at which baselines will be created.
→ Define back up procedure and a reconciliation procedure
→ Define a release procedure

✓ Risk management activities?

- Risk management
  - Risk assessment
    - Risk Identification
    - Risk analysis
    - Risk prioritization
  - Risk Control
    - Risk management planning
    - Risk resolution
    - Risk monitoring

Team or people management and personal planning

The people management maturity model defines the following key practice areas for software people, recruiting, Selection, performance management, training and compensation, career development, organization and work design and team/culture development.

Main players who can be categorized into one of five constituencies:

1) Senior Managers – who define the business issues
2) project managers (technical) – who must plan, motivate, organize and control the practitioners who do Software development work.
3) practitioners or programmer developers – who deliver the technical skills that are necessary to engineering a product or application
4) Customers → Which specify the requirements for the Software to be engineered and other stake holders who have a peripheral Interest in the out come.
5) end users – who interact with the Software once it is released for production use.

The best team structure depends on the management style of your organization, the numbers of

people who will populate the team and their skill levels and the overall problem difficulty.
Mantei suggests three generic team size organizations →
1) Democratic decentralized (DD) → This Software engineering team has no permanent leader. Rather "task coordinators are appointed for short durations and then replaced by others who may coordinate different tasks". Decisions on problems and approach are made by group consensus. Communication among team members is horizontal.
2) Controlled decentralized (CD) → This Software Engineering team has a defined leader who coordinates specific tasks and secondary leaders that have responsibility for subtasks. problem solving remains a group activity, but implementation of solutions is partitioned among subgroups by the team leader. Communication among subgroups and individuals is Horizontal. Vertical communication along the control hierarchy also occurs.
3) Controlled Centralized (CC) → Top level problem solving and internal team coordination are managed by a team leader. Communication between the leader and team members is vertical.

Centralized structure completes tasks faster, it is the most adept at handling simple problems, Decentralized teams generate more and better solutions than individuals. Therefore such teams have a greater probability of success when working on difficult problems. Since the CD team is centralized for problem solving, either a CD or CC team structure can be successfully applied to Simple problems. A DD structure is best for difficult problems.

Another structure of team

For a small project, a one level hierarchy suffices for, larger projects this organization can be extended early by partitioning the project into modules and having module leaders who are responsible for all tasks related to their module and have a team with them for performing these tasks.

A different team organization is egoless team: Egoless teams consist of ten or fewer programmer. The goal of group are set by consensus and input from every member is taken for major decisions. Group leader ship rotates among the group members. Due to their nature, egoless teams are sometimes called democratic teams. This structure allows input from all members; which can lead to better decisions for difficult problems.

For very large product developments another structure has emerged team. This structure recognizes that there are three main tasks categories in Software development management related, development related and testing related. This type of team structure used by Microsoft.

personal planning involve the tasks which can be performed or planned by top management activities.

1) Recruitment & Selection plans
2) project allotment to Skill personnels
3) Training & workshops for Skills enchancement
4) Motivation / ~~Develop~~ Leadership Quality among Manager or top level and team. Communication at ~~lower level~~.

PROCESS MODELS OR SOFTWARE ENGINEERING PARADIGM

Software process is a collection of Software Engineering work tasks, project Milestones, work products and Quality assurance points - enable the frameworks activity to be adapted to characteristics of Software project and the requirements of the project team. The Software engineering Institute (SEI) has developed a comprehensive model predicated on a set of Software engineering capabilities that should be present as organizations reach different levels of process maturity. The grading scheme determines compliance with a capability maturity model (CMM) that defines key activities required at different levels of process maturity.

Level 1 : Initial → adhoc and occasionaly even chaotic Success depends on individual effort.

Level 2 : Repeatable → Basic project management procasses Repeat earlier successes on projects with similar applications + level 1

Level 3 : Defined :→ Software process include engineering activity documented, standardized and integrated + level 2

Level 4 : Managed → Detailed measure of Software process and product Quality are collected. + level 3

Level 5 : optimizing :→ Continuous process improvement is enabled by quantitative feedback from process + level ~~4~~.

a) Different process models →

o) Build and fix model

1) Linear Sequential model or waterfall model or classic life cycle model

2) prototyping model

3) Iterative Model or Incremental Model or RAD model

4) Spiral Model

5) V - life cycle model

o) In this model first version of product is built and it is continuously modified till the client is satisfied. This is known as adhoc model. Model mainly use for very small projects. product always in operational mode. It is a single person task and product is constructed without specification. No documentation require.

$$\begin{tikzcd} \text{Build Product} \arrow[r, bend left] & \text{Fix problems} \arrow[l, bend left] \end{tikzcd}$$

1) Classic lifecycle model →

Some time called waterfall model, This model states that the phases are organized in a linear order, though variations of the model have evolved depending on the nature of activities and the flow of control between them. This model is also known by other names such as original or conventional or traditional waterfall software life cycle model. This model is named "waterfall model" because its diagrammatic representation resembles a cascade of waterfalls. The different phases of this model are feasibility study, requirement analysis and specifications design, coding and unit testing, integration testing and system testing and maintenance. During each phase of life cycle, a set of well defined activities are carried out. each phase typically requires relatively different amount of efforts.

$$\begin{tikzcd}[column sep=small] \text{Feasibility study} \arrow[dr] & & & & \\ & \text{Requirement analysis \& specification} \arrow[d] & & & \\ & \text{Design} \arrow[dr] & & & \\ & & \text{Coding and Unit testing} \arrow[dr] & & \\ & & & \text{Integration and System testing} \arrow[d] & \\ & & & \text{Maintenance} & \end{tikzcd}$$

Classical waterfall model

1) Feasibility study — The aim of feasibility study is to determine whether developing the product is financially and technically feasible or not. The feasibility study involves analysis of the problem and collection of data which would be input to the system, the processing required to be carried out on these data.

2) Requirement analysis and specification
The aim of this phase is to understand the exact requirements of customer and to document them properly. requirement analysis is to collect and analysis all related data and information with a view of understanding the customer requirements clearly and weeding out inconsistencies and incompleteness. During the requirement specification user requirements are properly organized and documented in SRS document. SRS documents address the functional requirements, the nonfunctional requirements and special requirements on the maintenance and development of software product. SRS document serves as a contract between the development team and the customer.

Design → The design phase translates the SRS into the design document which depicts the overall modular structure of the program and the interaction between these modules. Two distinct design approaches are being followed
Traditional design approach: This design approach requires two different activities to be performed
Structure analysis of requirements and structure analysis is transformed into software design also called software architecture.
→ object oriented design → In this technique

Various objects that occurs in ou problem domain and the Solution domain are first Identified and different kind of relationships that exist among there objects are Identified.

Coding and Unit testing → The purpose of this phase of software development is to translate the Software design into Source Code. During this phase, each Component of ~~the~~ design is implemented as a program module and each of these program module is Unit tested, debugged and documented.

purpose of Unit testing is determine the Correct working of the individual modules. Unit testing involves a precise definition of test cases, test Criteria and management of test cases.

Integration and System testing → During this phase the different modules are integrated in a planned manner. The different modules Making up a System are never integrated in a Single Shot. integration is normally Carried out through a number of steps. During each integration step, the partially integrated System is tested. The System testing usually Consists of three different kind of testing activities,

α testing

β. Testing

Acceptance testing

Testing is carried out according to a System test plan document. The system test plan Identifies all testing related activities. The system test plan prepared during the requirement Specification phase lists all the different test cases and ou expected outputs. The final output of the testing phase is the TEST REPORT.

maintance → maintenance involve performing any one or more of following kinds of activities.

1) corrective maintenance → correcting errors that were not discovered during the product developmed phase.

2) perfective maintenance → Improving the implementation of the System and enhancing the functionalities of the system according to Customer requirements.

3) adaptive maintenance → porting the Software to a new enviroment,

Advantages →

1) easy to Understand even by non technical person
2) each phase has well defined inputs and outputs
3) each stage has well defined deliverable or milestones
4) helps the project manager in proper planning of project.
5) It work well When Quality requirements dominate Cost and Schedule requirements.
6) It provides structure to a technically weak or inexperienced staff.
7) It tackles Complexity in an orderly way

Disadvantages →

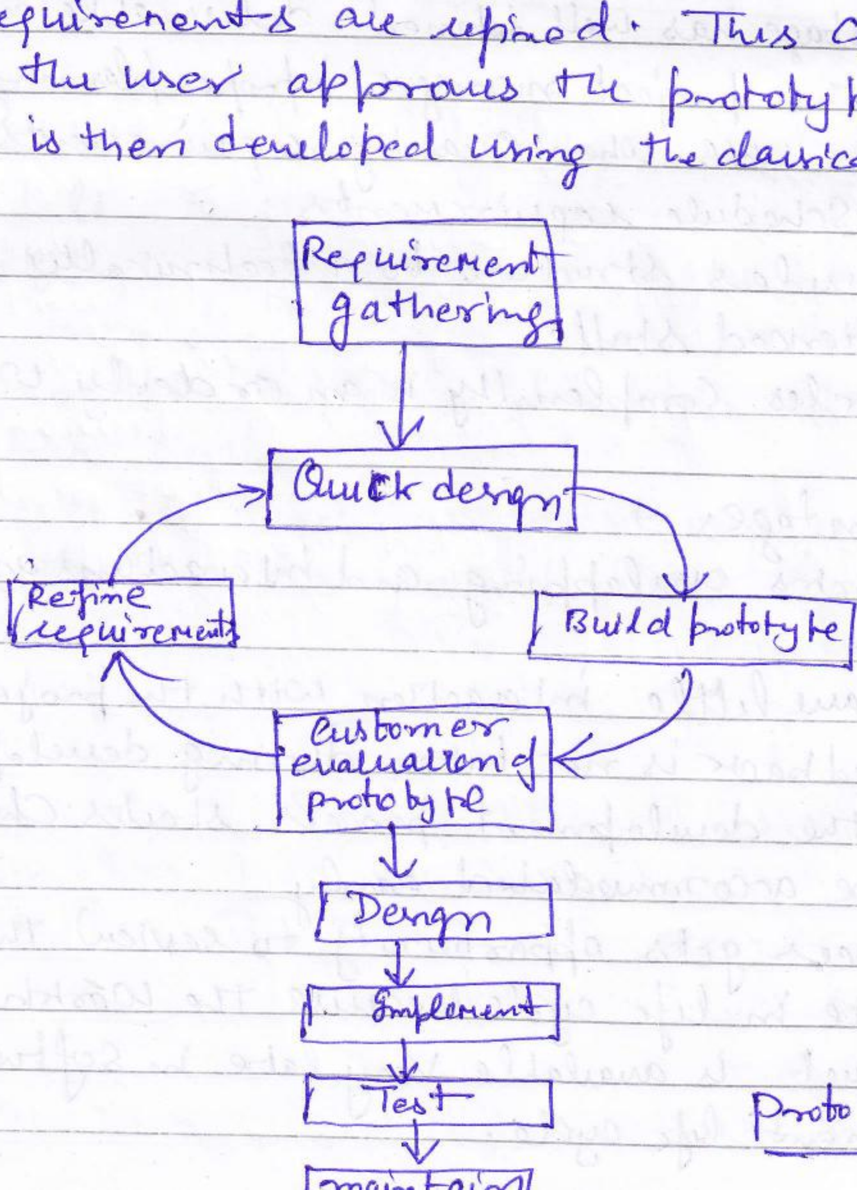
1) It lacks overlapping and interactions among phases.
2) users have little interaction with the project team their feedback is not taken during development.
3) After the development process starts Changes Cannot be accommodated easily
4) Customers gets opportunity to review the product Very late in life cycle because the working version of product is available Very late in Software development life cycle.

2) prototype life cycle model →

The prototype model suggest that before developing the actual software, a working prototype of the system should be built. A prototype is partially developed product. It has limited functional capabilities, low reliability and inefficient performance.

The model starts with an initial requirements gathering phase. A quick design is carried out and the prototype model is build using several shortcuts. The shortcuts might involve using inefficient, inaccurate or dummy functions

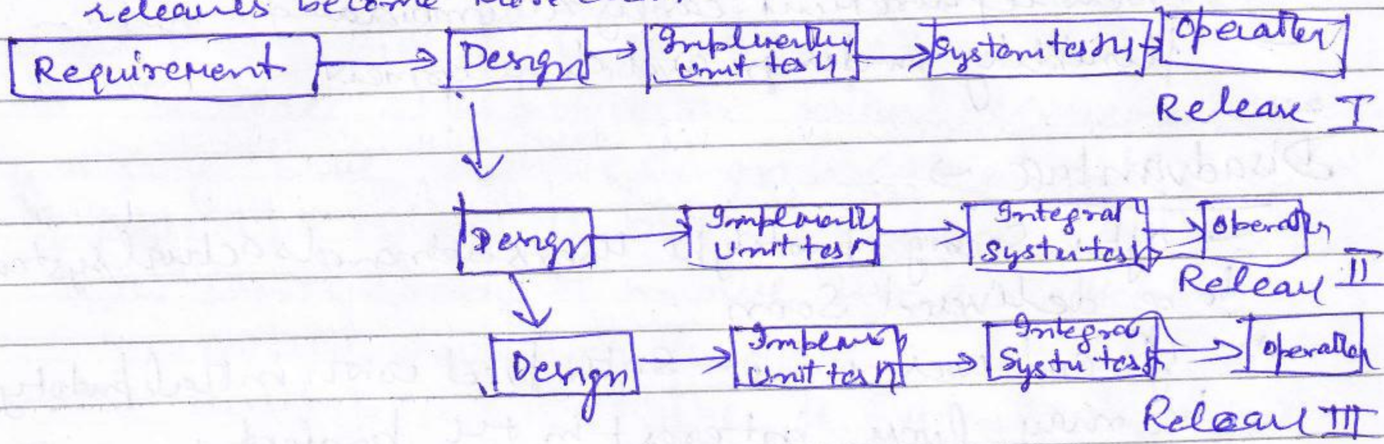
The developed prototype is submitted to the customer for his evaluation. Based on the user feedback the requirements are refined. This cycle continues until the user approves the prototype. The actual system is then developed using the classical waterfall model.



Prototype model

because works is to be delivered in parts
→ product is to be delivered in parts, total cost of the project is distributed.

→ End user's feedback requirements for successful releases become more clear



Iterative enhancement Model

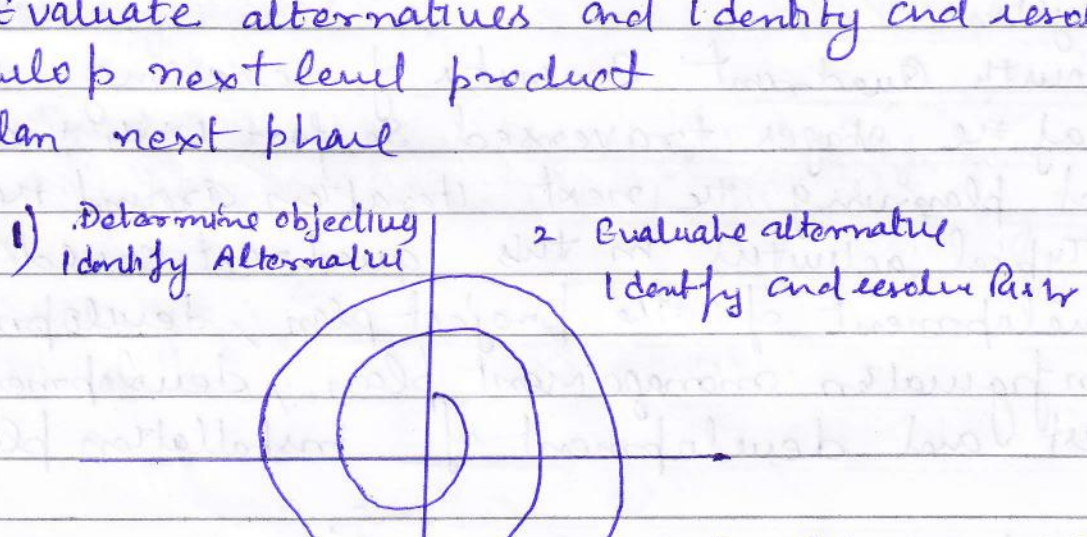
Disadvantages →
Model requires well defined project planning schedule to distribute the work properly.
→ Testing of modules results into overhead and increased cost.
→ product is delivered in parts, total development cost is higher
→ Well defined interfaces are required to connect modules developed with each phase

Spiral life cycle model

The Spiral model encompasses the strengths of the waterfall model while including risk analysis, risk management, and support and management processes. It also allows for the development of the product to be performed using a prototyping technique or rapid application development through the use of fourth generation languages and development tools.

Spiral model software development is carried out in four main phases

1) Determine objective, alternatives and Constraints
2) Evaluate alternatives and identify and resolve Risks
3) Develop next level product
4) plan next phase



1) Determine objective
Identify Alternative
2 Evaluate alternative
Identify and resolve Risk
3. Develop the next level of the product
4) Customer evaluation of the prototype

First quadrant identifies the objectives of the product and alternative solutions possible. objectives such as performance, functionality, ability to accommodate change, hardware/software interface etc

In Second Quadrant the alternative solutions are evaluated and the potential ~~product~~ project risks are

Identified and dealt with by developing an appropriate prototype. The Identification and resolution of risks like risk management, cost effective strategy for resolving sources, evaluation of remaining risks where money could be lost by continuing development occurs.

Third Quadrant consist of developing and verifying the next level of the project. Typical activities in this quadrant could be creation off design, review of a design, development of code, inspection of code, testing, packaging of the product. The degree of change from one build to the next diminishes with each build, eventually resulting in a operational system.

Fourth Quadrant Consists of reviewing the result of the stages traversed so far with the customer and planning the next iteration around the spiral. Typical activities in this quadrant could be development of the project plan, development of configuration management plan, development of the test and development of installation plan.

Advantages → → A wide range of options are available to accommodate the good features of other life cycle models

→ Model incorporates software quality objectives into software development.

→ Risk analysis and validation steps eliminate errors in early phases of development.

→ It provides early and frequent feedback from users to developers, ensuring a correct product with high quality.

→ It provide productivity improvement through

→ It allows users to be closely tied to all planning, risk analysis, development and evaluation activities.

Disadvantages →

→ This Model is not Suitable for small project as cost of risk analysis may exceed the actual cost of the project.

→ Different persons involved in the project may find it complex to use.

→ Model requires expertise in Risk management and excellent Management Skills.

→ large nos of documentation process at every state.

→ use of model may be expensive and even unaffordable.

→ It is hard to define objective, verifiable milestones that indicate readiness to proceed through the next iteration.

V-LIFE CYCLE MODEL

This model was developed to relate the analysis and design activities with the testing activities and thus focuses on Verification and validation activities of the product. As this model relates the analysis and design phase to testing phase, testing activities are done in parallel.

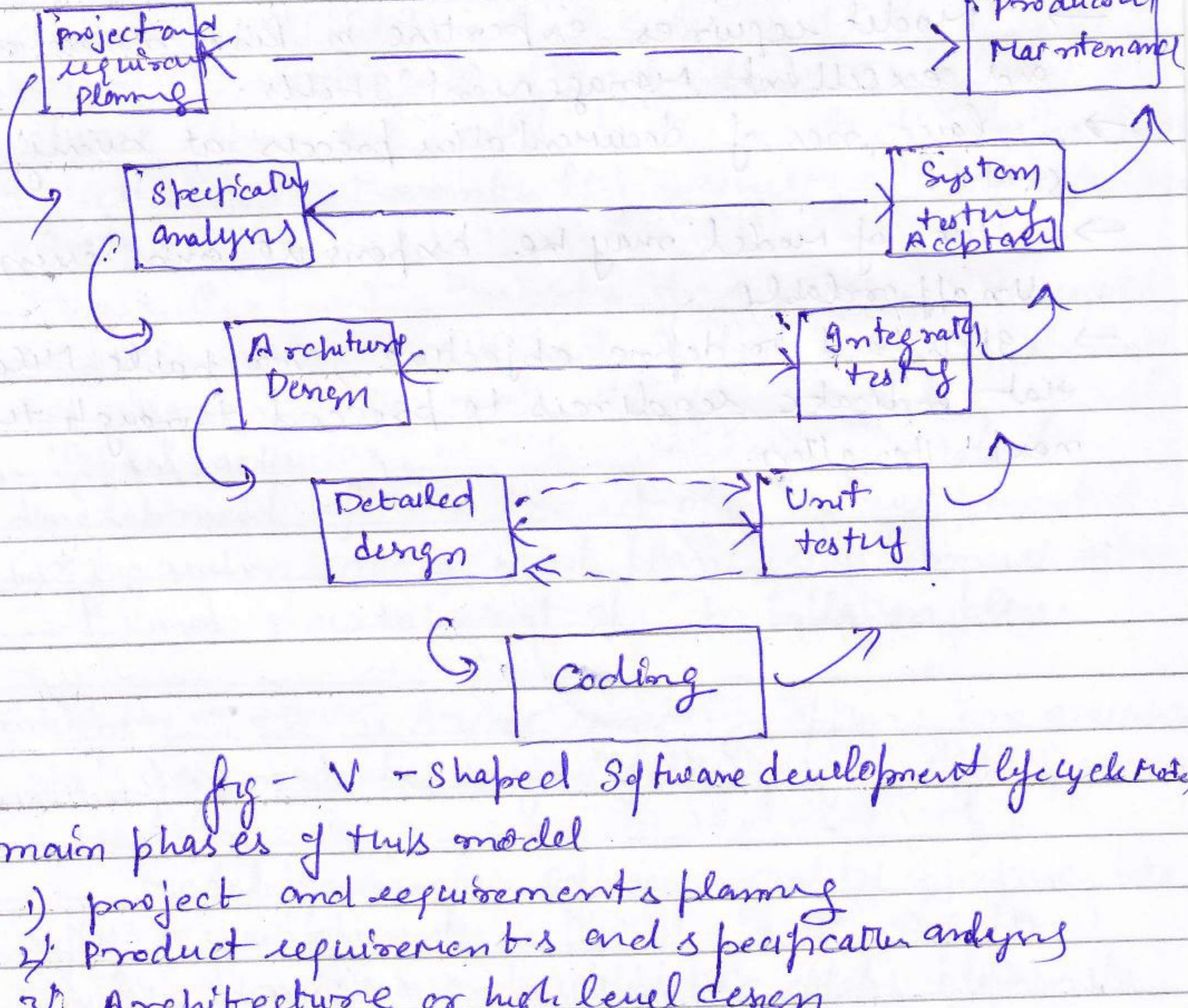


fig - V - shaped Software development lifecycle model

main phases of this model

1) project and requirements planning
2) product requirements and specification analysis
3) Architecture or high level design
4) Detailed design
5) coding
6) Unit testing
7) Integration and testing
8) ~~maintenance~~

Advantage of V-shaped Model

→ The Model is simple and easy to use
→ The Model focuses on Verification and validation of the intermediate products not only the final Software.
→ It defines the products that the development process should generate; each deliverable must be testable
→ It enables project management to track progress accurately; the progress of the project follows a time-line and completion of each phase is a Milestone.

Disadvantage →

→ The Model does not support iteration of phases and change in requirements throughout the life cycle
→ It does not take into account Risk analysis
→ It does not easily handle concurrent events.
→ The Model is not equipped to handle dynamic changes in requirements throughout the life cycle.

When to use the V-shaped Model →
The V-shaped Model is an excellent choice for systems that require high reliability such as hospital patient control applications and embedded software for air bag chip controllers in automobiles.