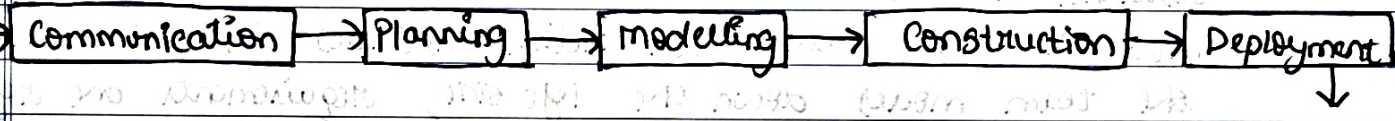
NAME - MD. ARHAAN ALI
Batch - T11
Roll no - 02

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

SEPM

Waterfall model :

- The waterfall model, sometimes called the classic life cycle, suggests a systematic, sequential approach to software development that begins with customer specification of requirements and progresses through planning, modelling, construction and deployment, culminating in ongoing support of the completed software.
- A variation in representation of the waterfall model is called the V-model.



The waterfall model

⇒ Advantages :
- Simple and easy to understand.
- Easy to manage.
- Best for smaller projects.
- Individual processing.

⇒ Disadvantages :
- Inflexible
- Late testing
- Not suitable for evolving projects
- Lengthy development cycle.

⇒ Example - In a library management system, phases include requirement analysis, system design, implementation testing, deployment and maintenance. Once a phase is finished, it does not return to previous stages.

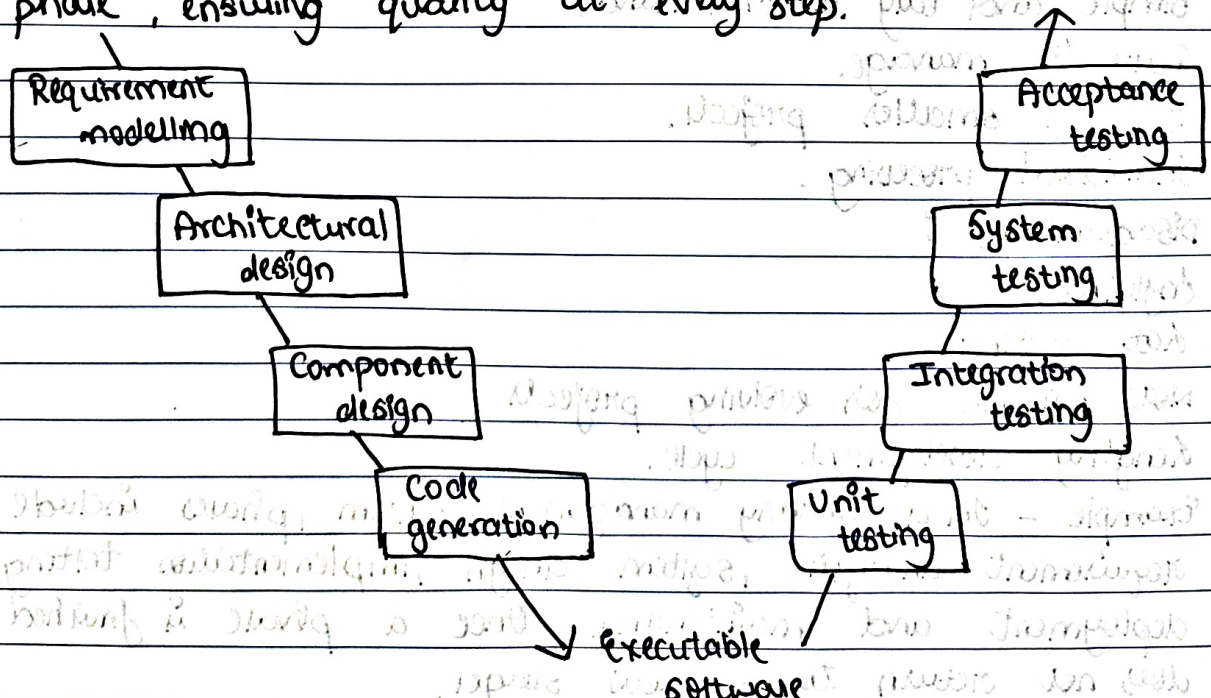
THADOMAL SHAH.
TSEC
ENGINEERING COLLEGE

When to use waterfall model?

- well understood requirements.
- Very little changes expected.
- Small to medium size projects
- Client prefers a linear and sequential approach.
- Limited resources.

V model :

- A variation in the representation of the waterfall model is called the V-model. It is also referred to as the verification and validation model. It depicts the relationship of quality assurance actions to the actions associated with communication, modelling and early construction activities. On the V-model, as the team moves down the left side, requirements are refined into detailed solutions, once coding is done, they move up the right side, performing tests to validate each development phase, ensuring quality at every step.

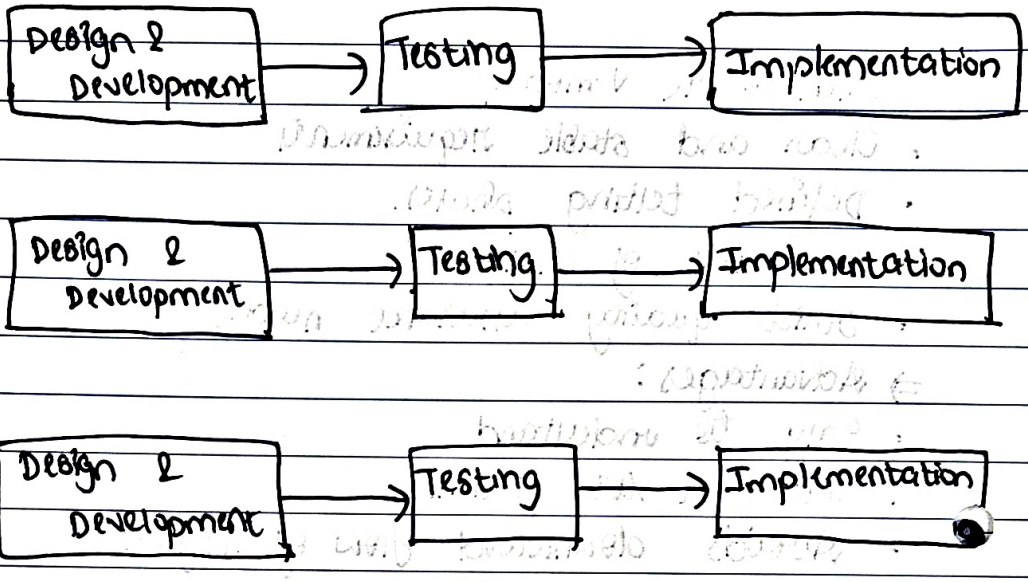


When to use V-model?
- Clear and stable requirements
- Defined testing phases.
- Low risk of changes
- Strict quality assurance needs.

⇒ Advantages:
- Easy to understand
- Saves a lot of time.
- Avoids downward flow of defects.

⇒ Disadvantages:
- Rigid & least flexible.
- Not good for complex projects.
- No early prototypes of the software are produced.

Incremental Process model:
- The incremental model combines elements of linear and parallel process flows. It applies linear sequences in a staggered fashion as ~~calendar~~ time progresses. When an incremental model is used, the first increment are often a core product i.e. basic requirements are addressed but many supplementary features remain undelivered. The core product is used by the customer (or undergoes detailed evaluation). As a result, a plan is developed for the next increment. The plan addresses the modification of the core product to better meet the needs of the customers and the delivery of additional features and functionality. This process is repeated following the delivery of each increment, until the complete product is produced.



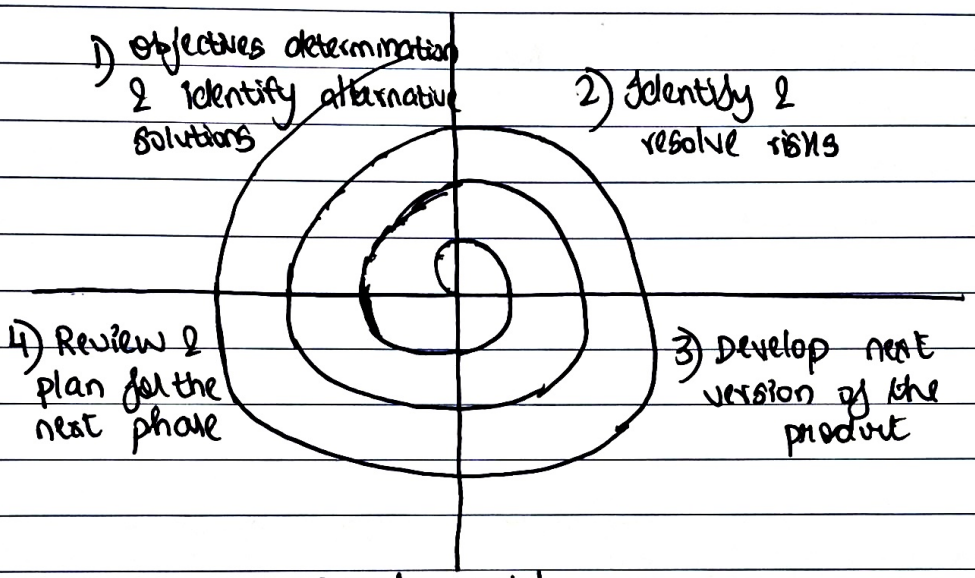
⇒ Advantages :
- Error are easy to be recognized.
- More flexible
- Easier to test & debug.

⇒ Disadvantages :
- Cost is high
- Need for good planning.
- Well defined module interfaces are needed.

Spiral model :
- It couples the iterative nature of prototyping with controlled and systematic aspects of the waterfall model.
- The spiral development model is a risk driven model generator that is used to guide multi-stakeholder concurrent engineering of software intensive systems. It has 2 main distinguishing features. One is a cyclic approach for incrementally growing a system's degree of definition & implementation while decreasing its degree of risk. The other is a set of anchor point milestones for ensuring stakeholder commitment to feasible and mutually satisfactory system solutions.

THADOMAL SHAHANI
TSEC
ENGINEERING COLLEGE

- A spiral model is divided into a set of framework activities defined by the software engineering team.



Spiral model

⇒ Advantages:
- Risk handling
- Good for large projects
- Cutomer satisfaction
- Improved quality

⇒ Disadvantages:
- Complex
- Expensive
- Difficulty in time management

- Spiral model delivers high quality software by promoting risk identification, iterative development and continuous client feedback when a project is vast in software engineering, a spiral model is utilized.

NAME - MD. ARHAAN ALI
Batch - T11
Roll no - 02

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

SEPM

To understand DevOps : Principles, Practices & DevOps Engineer Role & Responsibilities.

What is DevOps?

- DevOps is a collaborative approach where teams work together to build and deliver secure software efficiently. It combines software development (dev) and operation (ops) to accelerate delivery through automation, collaboration, fast feedback & iterative improvement. Built on agile methodology, DevOps creates a culture of accountability, collaboration and shared responsibility for business outcomes.

Core Principles of DevOps :

- Develop & test in production-like environments.
- Develop builds frequently.
- Continuously validate operational quality.

Key Practices of DevOps :

1) Continuous Deployment -
- Continuous delivery and deployment originate from continuous integration, a method to rapidly develop, build and test new code with automation so that only code that is known to be good becomes part of a software product.

2) Continuous Development -
- This is the phase that involves planning and coding, versioning and managing builds of the software application functionality.
- Ex - Git, Github, Maven

THADOMAL TSEC ENGINEERING COLLEGE

3) Continuous Testing-
- Continuous testing is, executing automated tests, continuously and repeatedly against the code base and the various deployment environments.
- It is a software testing methodology which focuses on achieving continuous quality and improvement.
- Ex- Appium, Bamboo.

4) Continuous Integration-
- Continuous Integration refers to the build and unit testing stages of the software release process.
- Every revision that is committed triggers an automated build and test.
- Ex- Jenkins, Travis CI

5) Infrastructure management-
- Without automation, building and maintaining large-scale modern IT systems can be a resource intensive undertaking and can lead to increased risk due to manual error.
- Configuration and resource management is an automated method for maintaining computer systems and software in a known, consistent attack.

6) Configuration management-
- Infrastructure as code is the practice of describing all software runtime environment and networking settings and parameters in simple textual format, that can be stored in your version control system (VCS) and versioned on request.
- These text files are called manifests and are used by DevOps

tools to automatically provision and configure build servers, testing, staging and production environments.
- Ex - Chef, saltstack.

Devops Engineer Role :
- A Devops engineer manages the IT infrastructure, bridging development and operation.
- The primary goal is to automate processes and improve efficiency through the software development lifecycle

Key Roles :
1) Facilitator of Collaboration -
- Bridging the gap between development, operation and QA teams to streamline communication.
2) Automation Specialist -
- Automate repetitive tasks like testing, deployment and monitoring.
3) Continuous Integration & Continuous Delivery (CI/CD) -
- Design, implement and maintain CI/CD pipelines to enable faster, reliable and repeatable software releases.
4) Infrastructure as code -
- Use tools like Terraform, Ansible or cloud formation to define and provision infrastructure through code.
5) Monitoring and Incident Management -
- Set up monitoring systems to track application performance and troubleshoot issues in real time.
- It also ensures that systems are resilient and downtime is minimized.

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

6) Cloud and Infrastructure Management-
- Deploy, manage and optimize applications on cloud platform like AWS, Azure or Google Cloud.
- It also handles containes orchestration.

Key Responsibilities :
1) Collaboration and Planning-
- Work with development and operations teams to plan and design scalable solutions.
2) Configuration Management-
- Use tools like Puppet, chef or Ansible to manage servers configuration and ensure consitency.
3) Pipeline Management-
- Maintain CI/CD pipelines to ensure seamless build, test and deployment workflows.
4) Monitoring and Logging-
- Implement monitoring tools like Prometheus, Curafana or Splunk to track system health and measurement performance.
5) Support and Troubleshooting-
- Respond to incidents and resolve production issues promptly and identify root causes of failure and implement fixes.
6) Documentation and Reporting-
- Document system configurations, deployment processes and troubleshooting guides.