14/11/2025

SEPM

Sadhak Phase T21 77

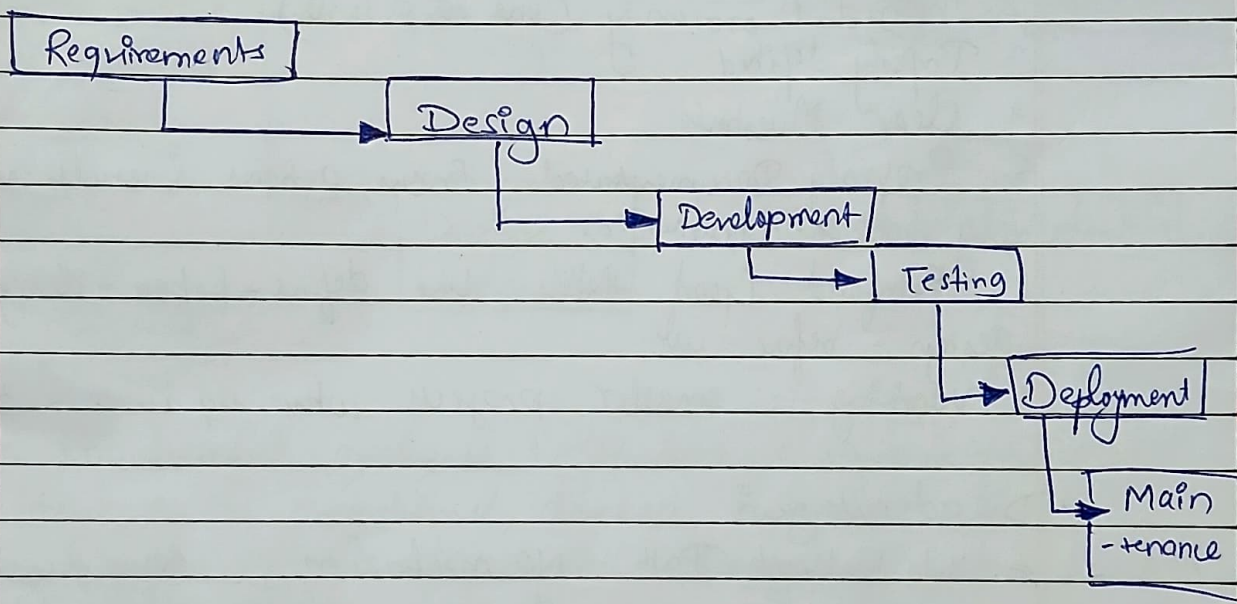
- Water fall Model

→ It is a classical software development methodology introduced in 1970 by Winston W. Royce.

→ It is a linear and sequential approach to software development that consists of several phases. Thus, must be completed in a specific order. Single & idealistic. Once very popular.

→ Phases include requirements gathering, design, development, testing, deployment and maintainence.

- Model:



- Phases:

1. Requirements: Gathering requirements from stakeholders & analyzing them to understand the scope & objectives of project
2. Design: Creating a detailed design document that outlines software architecture, user interface, and system components.
3. Development: Implementation, involves coding the software based on design specifications. Also includes unit testing to ensure that each component of the software is working as expected.

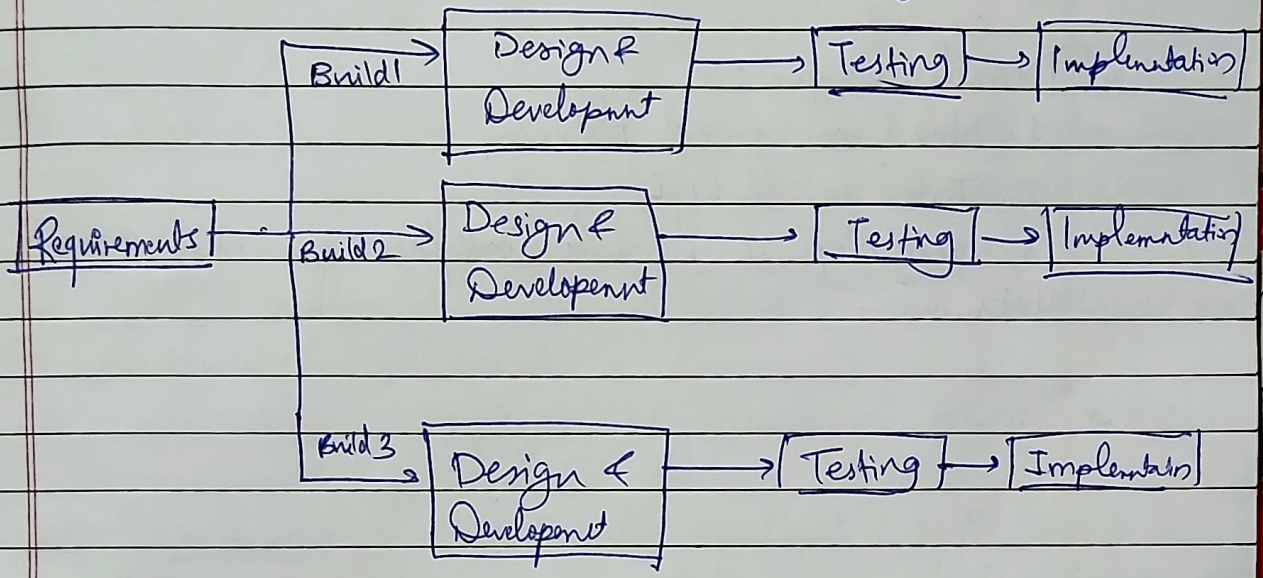
4. Testing: Software is tested as a whole to ensure that it meets its requirements. & is free from defects
5. Deployment: Once the software is tested as a whole to ensure that it meets the requirements & is free from defects.
6. Maintenance: Involves fixing any issues that arise after the software has been deployed & ensuring that it continues to meet the requirements over time

- Advantages:
  1. Easy to understand
  2. Individual processing (one at a time)
  3. Properly defined
  4. Clear Milestones
  5. Properly Documentated: Process, actions & results are very well documented.
  6. Reinforced Good Habits: like define-before-design & design-before-code.
  7. Working: smaller projects where req are well understood

- Disadvantages:
  1. No Feedback Path: NO mechanism for error correction Assumes that no error is ever committed by develop.
  2. Difficult to accomodate Change Requests: Assumes that all customer requirements can be correctly defined at beg.
  3. No overlapping of phases.
  4. Limited Flexibility
  5. Limited Stakeholder Involvement.
  6. Late Defect Detection
  7. Lengthy Development cycle.

Incremental Model

→ Process of software development where requirements divided into multiple standalone module of software development cycle.
Each module goes through the requirements, design, implementation & testing phases. Process continue until complete system achieved.



→ Phases:
1. Requirement analysis : Product analysis expertise identifies the requirement, develop, software.
2. Design & Development : Design of system functionality & development method are finished with success. When software develops new practicality, model uses style & development phase.
3. Testing : Checks performance of each existing function as well as additional functionality, thus various methods used
4. Implementation : Enable coding phase of development system. Involves final coding. After this phase, no of products working is enhanced & upgraded to final system product.

- Advantages:
  1. Errors are easy to be recognized
  2. Easier to test & debug
  3. More flexible
  4. Simple to manage risk because it handled during its iteration.
  5. Client gets important functionality early.

- Disadvantages:
  1. Need for good planning
  2. Total cost is high
  3 Well defined module interfaces are needed; exact no unknown.

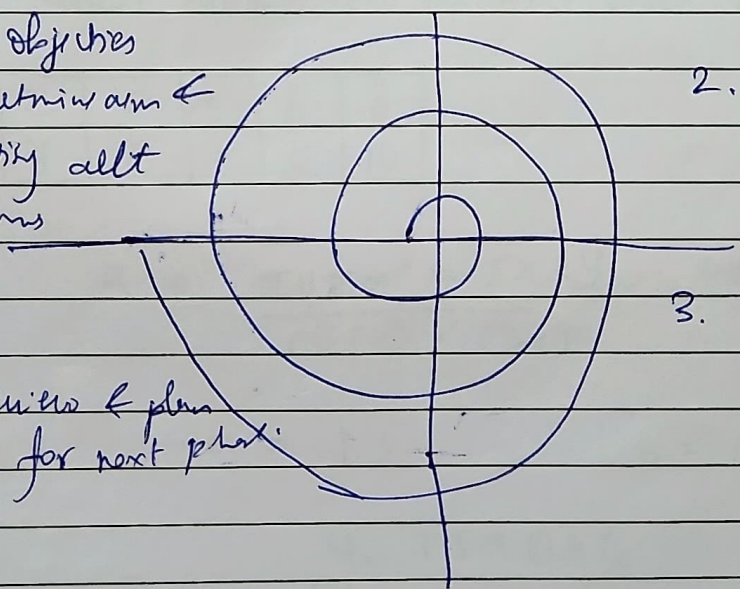
classmate
Date
Page

Spiral Model.

→ combination of waterfall model & Iterative model.
→ Provids support for Risk handling.
→ Proposed by Barry Boehm.
→ Diagrammatic representation looks like spiral with many loops.

Phases:

1. Objectives Defined : First phase, we clarify the project aims to achieved, including functional & non functional requirements.
2. Risk Analysis : Risks associated with the project are identified & evaluated.
3. Engineering : software is developed based on the requirements gathered in previous iterations.
4. Evaluation: Software is evaluated if it meets the customer's requirements & if it is of high quality.
5. Planning : Next iteration of spiral begin with new planning phase, based on results of evaluation.



4 Quadrants:

1. Objectives determination & Identify alternate solutions: Req gathered from customs & obj are identified elaborated & analyzed at the start of every phase.
2. Identify & resolve Risk: All possible soln are evaluated to select best possible soln.
3. Develop next version of Product: Identified feats are developed & verified through testing.
4. Review & plan for next phase: Customs evaluate & the so far developed version of software

Sadhak Phanse
T21.
77

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

SEPM Assignment 2

Aim:- To understand DevOps : Principles, Practises, and DevOps Engineer Role and Responsibility.

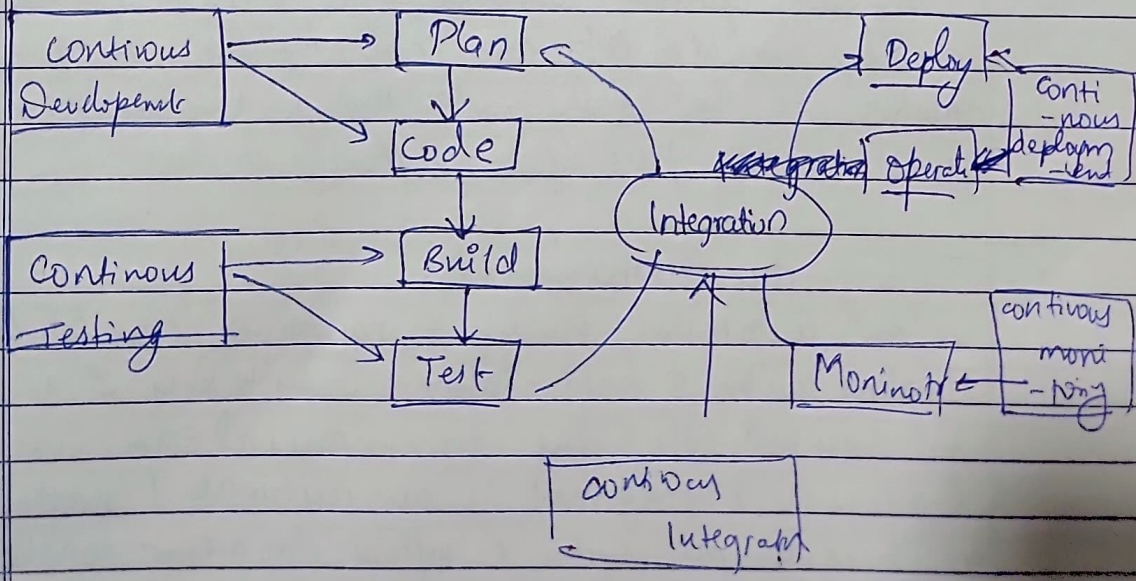
What is DevOps?

DevOps is a collaborative approach where teams work together in to build and deliever secure software efficiently. It combines Software development (dev) and operations (ops) to decide how to acctuate delivery through automation collaboration, fast feedback, and iterative improvement. Built on Agile methodology; DevOps creates a culture of accountability, collaboration and more responsibilities for busines outcomes.

Core Principles:

1. Develop and test in productive like environment.
2. Deploy builds frquently.
3. Continuously validate operational quality.

Practices:



- **Continous Development :**
This is the phase that involved planning & writing version & managing build of the software applications functionality.
Eg:- git, github, maven, etc.

- **Continous Testing :**
It is executing automated tests, continously and repeadly against the code base and various deployment environments. It is a software testing methodology which focuses on achieving continous quality & improvement.
ex:- Bamboo, appium.

- **Continous Integration :**
Refers to the build and unit testing stages of the software testing release process. Every revision that is committed triggers an automated build & test.
ex:- Jenkins, Travis CI, circle CI.

- **Continous Dliny & Deplogment.**
Originate from continous integration, a method to develop, build & test new code rapidly with auto-matin so that only code that is known to be good become part of a software product.

- **Infrastructure management.**
Without automation, building & maintaining large scale modern IT systems can be a resource-intensive undertaking & user lead to increased risk due to manual error. Configuration & resources management is an automated method for maintaing computer systems & software in a known consistent state.

- **Configuration management:**
Infrastructure as code is the practice of describing our software runtime environment and networking setting and parameter in simple textual format; that can be stored in your version control system (VCS) and versioned or request. These text files are called manifest & are used by DevOps tools to automatically provision & configure build servers, testing & production environment.

- **Micro service Architecture:**
Docker is a tool designed to make it easier to create, deploy and run applications by using containers. They allow a developer to package up an application with all of the parts it needs. such as libraries & other dependencies & deploy it as one package. By doing so, thanks to the container the developer can rest assured that the application will run on any other linux machine regardless of any customized settings that machine might have.
ex:- Nagios, Splunk, etc.

- **Cloud Based DevOps:**
Devops automation is becoming cloud centric. Most public & private cloud computing provide support. To DevOps systematically on their platform including continous integration & continous development tools.
ex:- Amazon web services, Amazon lambda, Google cloud etc.

DevOps Engineer Role:

A DevOps engineer manages a company's IT infrastructure, bridging development & operation. Key responsibilities include :-

→ Technical Responsibilities :

1. Implement development, testing & automation tools.
2. Set up infrastructure & tools
3. Code review & responsibilities.
4. Bug facing & trouble shooting.
5. Build & Maintain ci/cd ~~mode~~ piplines
6. Security Implementation & monitoring

→ Management Responsibilities :

1. Understand customer requirements & needs.
2. Plan & train structure and activities
3. Manage stakeholders
4. Define development & operational processes
5. Coordinate team communication.
6. Monitor customer experiens.
7. Provide periodic progress reports.
8. Mentor team members.