
Print Screen
Home
Page Up
Insert
End
Page Down
Delete
Enter
Shift

DevOps


DevOps
PLAN
CODE
BUILD
TEST
RELEASE
DEPLOY
OPERATE
MONITOR
Enter
Shift
DevOps

Advantages of DevOps

Let us now discuss some advantages of DevOps :

Why DevOps is Very Popular

- Faster Product Release
- Higher Productivity
- Process Efficiency
- Shortened Prod. Cycle
- Better Operational support
- Engaged & Motivated Staff
- Customer Management
- Clear Product Vision
- Increased Deployment
- Increase product quality
- Better team efficiency
- Reduced Failure Chance

What is DevOps

- The word DevOps is a combination of two words that is development and operations.

- DevOps is a set of practices, principles, and cultural philosophies aimed at improving collaboration between development (Dev) and operations (Ops) teams to deliver software and IT services more efficiently and reliably.

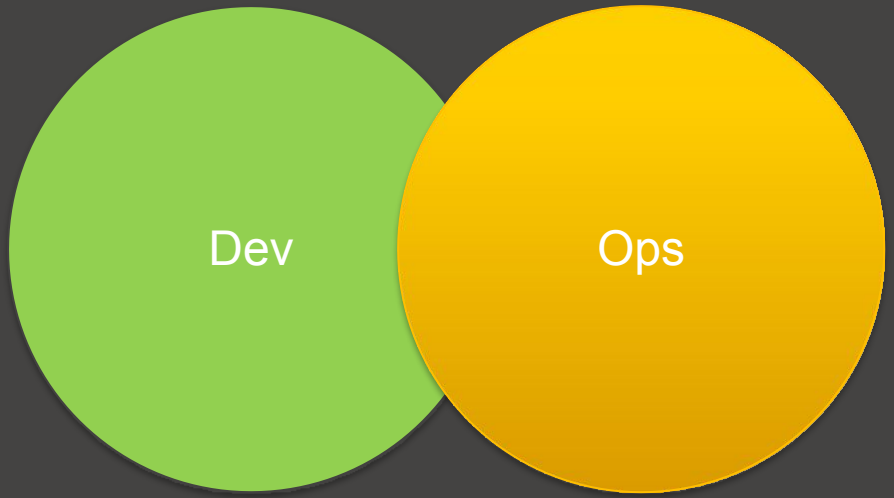
- It focuses on automating processes, integrating continuous testing and deployment, and fostering a culture of shared responsibility and continuous improvement.

DevOps is not ……

Separate Silos

DevOps is……

Smooth Collaboration



Why- DevOps

- Before DevOps, the **development and operation team** worked in **complete isolation.**

Break down the wall between development and operations

Why DevOps is needed

DevOps offers several key benefits that address common challenges in software development and operations:

- **Faster Delivery**: By streamlining processes and automating repetitive tasks, DevOps enables faster development, testing, and deployment cycles.
- **Improved Quality:** Continuous integration and automated testing help catch bugs and issues early in the development process.
- **Increased Efficiency:** Automation of tasks such as deployment and infrastructure management reduces manual effort and human error.
- **Enhanced Collaboration:** DevOps fosters better collaboration and communication between development, operations, and other teams.

Why DevOps is needed

- **Greater Flexibility:** DevOps practices enable teams to adapt quickly to changes, such as shifting requirements or emerging technologies.
- **Reduced Downtime:** Continuous monitoring and automated deployment processes help detect and address issues more rapidly, leading to fewer disruptions and reduced downtime for users.
- **Scalability:** Infrastructure as Code (IaC) and automated scaling allow organizations to manage and scale their infrastructure more easily, accommodating growth and changing workloads with minimal manual intervention.
- **Enhanced Security:** Automated security checks and continuous monitoring help identify and address vulnerabilities more quickly, leading to better overall security posture.

What Is the Goal of DevOps?

Improve collaboration between all stakeholders from planning through delivery and automation of the delivery process in order to:

- Improve deployment frequency
- Achieve faster time to market
- Lower failure rate of new releases
- Shorten lead time between fixes
- Improve mean time to recovery

Principle of DevOps

Here, are six principles which are essential when adopting DevOps:

- Customer-Centric Action: DevOps team must take customer-centric action for that they should constantly invest in products and services.
- End-To-End Responsibility: The DevOps team need to provide performance support until they become end-of-life. This enhances the level of responsibility and the quality of the products engineered.
- Continuous Improvement: DevOps culture focuses on continuous improvement to minimize waste. It continuously speeds up the improvement of product or services offered.

Principle of DevOps

- Automate everything: Automation is a vital principle of DevOps process. This is not only for the software development but also for the entire infrastructure landscape.

- Work as one team: In the DevOps culture role of the designer, developer, and tester are already defined. All they needed to do is work as one team with complete collaboration.

- Monitor and test everything: It is very important for DevOps team to have a robust monit oring and testing procedures.

How is DevOps different from traditional IT

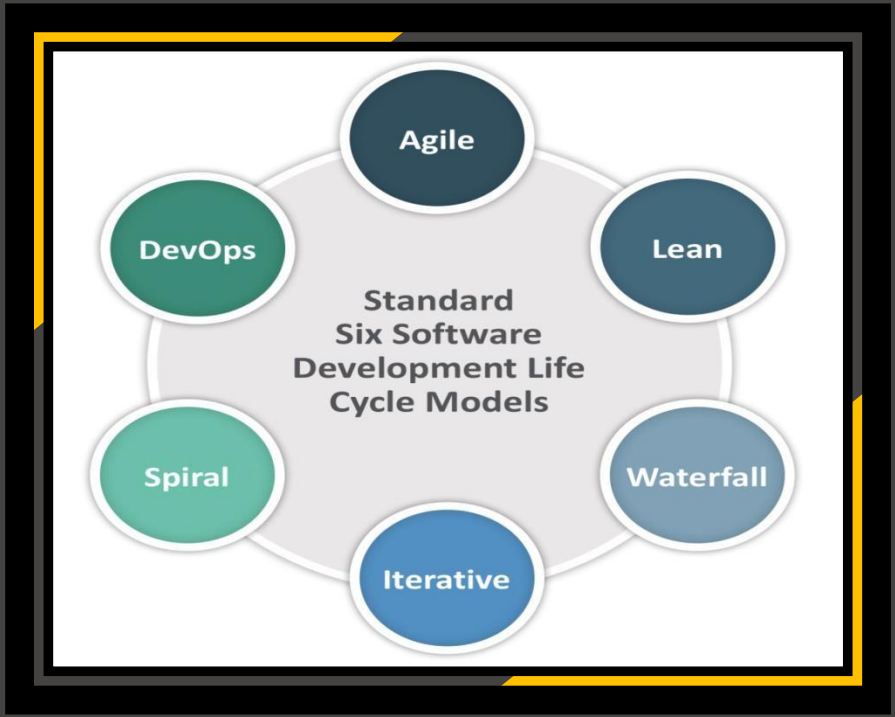
- Let's compare traditional software waterfall model with DevOps to understand the changes DevOps bring.

- We assume the application is scheduled to go live in 2 weeks and coding is 80% done.
  We assume the application is a fresh launch and the process of buying servers to ship the code has just begun

SDLC Models

- **SDLC** is a framework that outlines the stages involved in developing software applications. It includes various models and methodologies for managing and guiding the software development process.

- There are different software development life cycle models specify and design, which are followed during the software development phase. These models are also called **"Software Development Life Cycle"** Models.

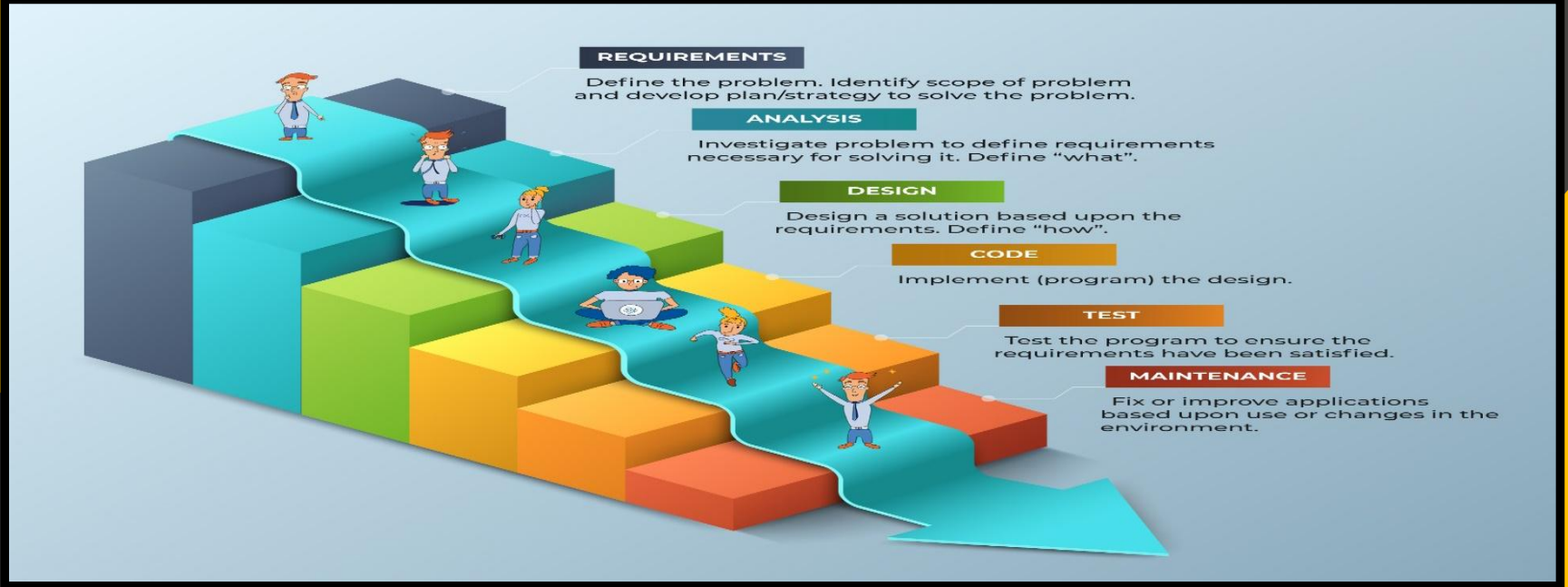


Common phases in the SDLC

- **Planning:** Defining the scope, objectives, and resources for the project.
- **Req. Analysis:** Gathering and analysing requirements from stakeholders.
- **Design:** Creating architectural and design specifications.
- **Coding:** Writing and coding the software.
- **Testing:** Verifying that the software meets requirements and is free of defects.
- **Deployment:** Releasing the software to users.
- **Maintenance:** Providing ongoing support and updates.

Understand Waterfall Methodology

The Waterfall model is one of the earliest SDLC methodologies and is characterized by a linear and sequential approach. Each phase must be completed before moving on to the next, with minimal overlap or iteration.



Understand Agile Methodology

Agile Methodology involves continuous iteration of development and testing in the SDLC process. This software development method emphasizes on iterative, incremental, and evolutionary development.
Agile development process breaks the product into smaller pieces and integrates them for final testing. It can be implemented in many ways, including scrum, kanban, scrum, XP, etc.
(SCRUM - At the end of each sprint if approved by the product owner. KANBAN - Continuous delivery or at the team's discretion. XP (Extreme Programming) - At the end of iteration.)

DevOps Vs Agile

DevOps

- Addresses the gap between development and Operation team
- Automation is the primary goal of DevOps.
- The DevOps divides and spreads the skill set between development and the operation team.
- It has a large team size as it involves all the stack holders.

Agile

- Addresses gap between customer requirements and development teams.
- Agile does not emphasize on the automation.
- The Agile development emphasizes training all team members to have a wide variety of similar and equal skills.
- It has a small team size.

DevOps Vs Agile

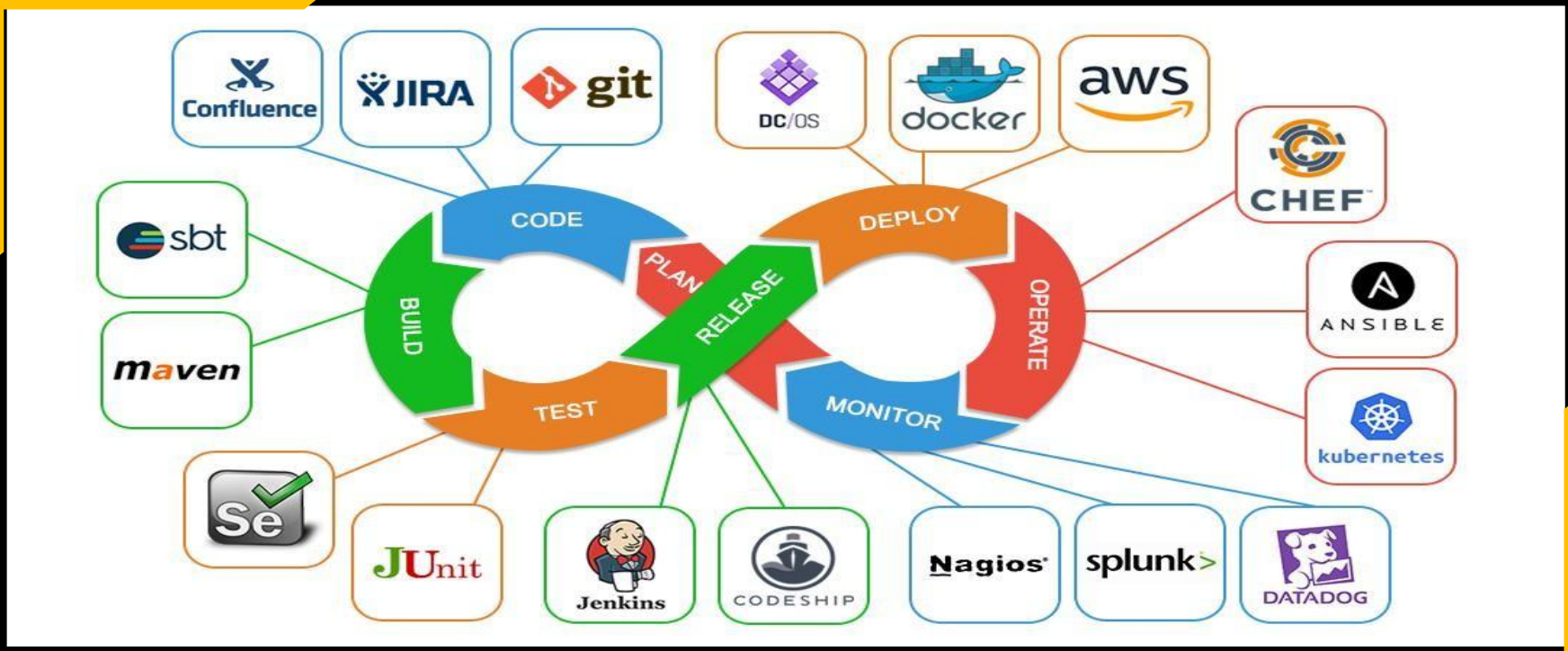
DevOps

- Git,Maven, AWS, Ansible, Terraform, Jenkins, Docker are popular DevOps tools
- It focuses on constant testing and delivery
- The ideal goal is to deliver the code to production daily or every few hours.
- Feedback comes from the internal team.

Agile

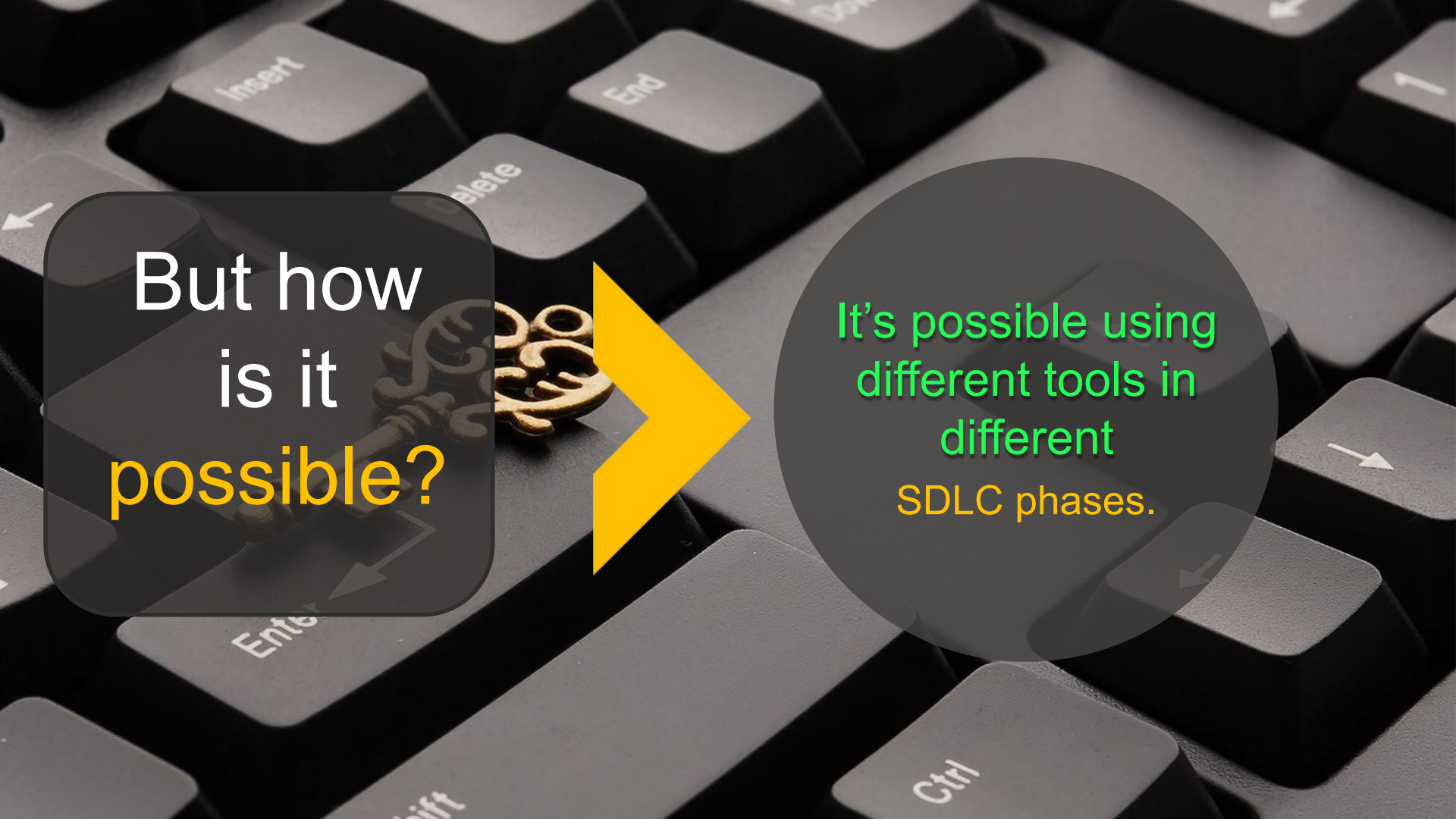
- Bugzilla, Kanboard, JIRA are some popular Agile tools.
- It focuses on constant changes.
- Agile development is managed in units of sprints. So this time is much less than a month for each sprint.
- feedback is coming from the customer.

DevOps Tools

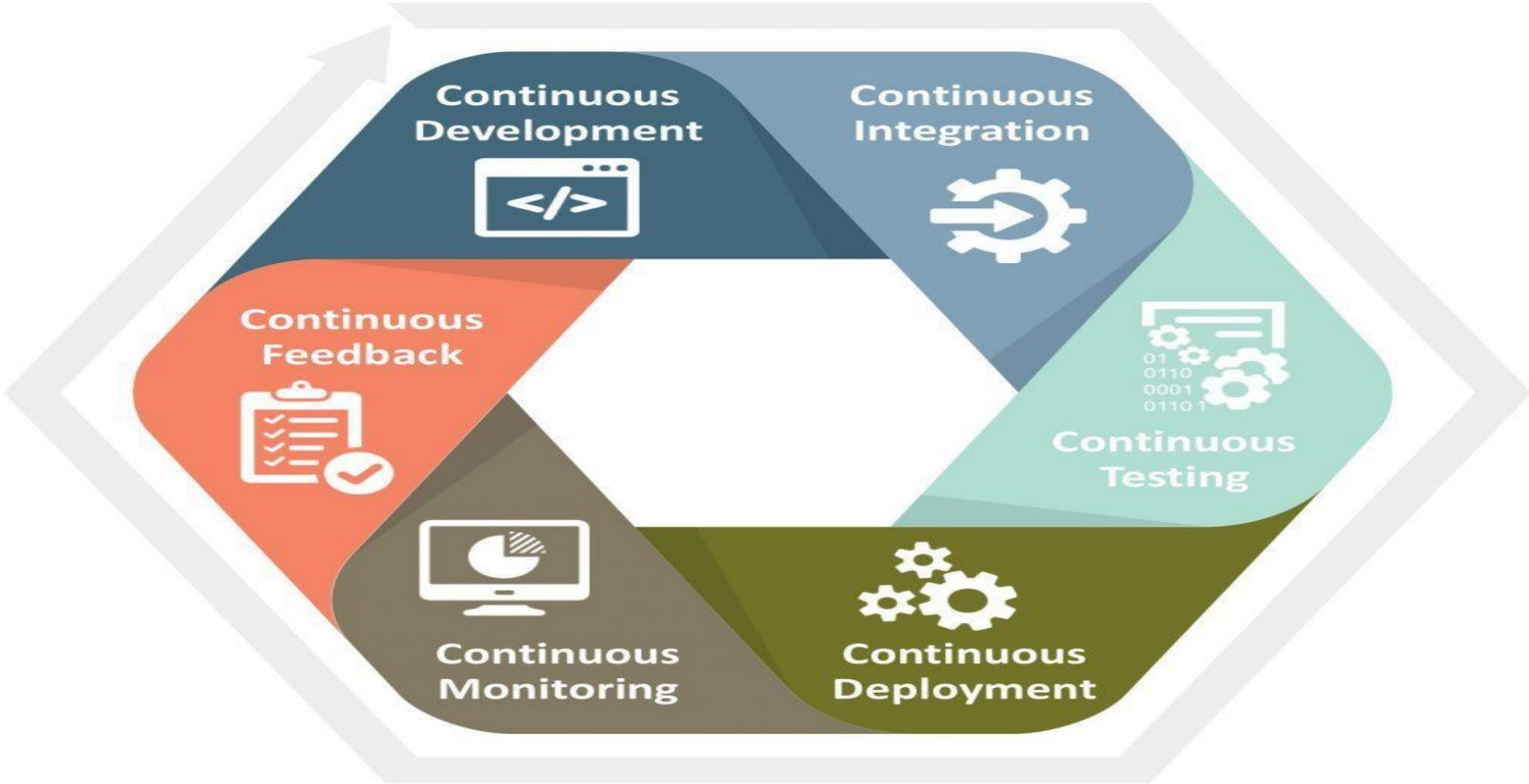


Achieving DevOps

It is a collaborative and unified effort between the development and the operations. Gone are the days when a developer used to write a code for days and then have to wait for hours to get it deployed. DevOps has been the game-changer in optimizing the entire effort of building and deploying the code achieving DevOps Automation.


But how is it possible?
It’s possible using different tools in different SDLC phases.

Achieving DevOps



DevOps Automation

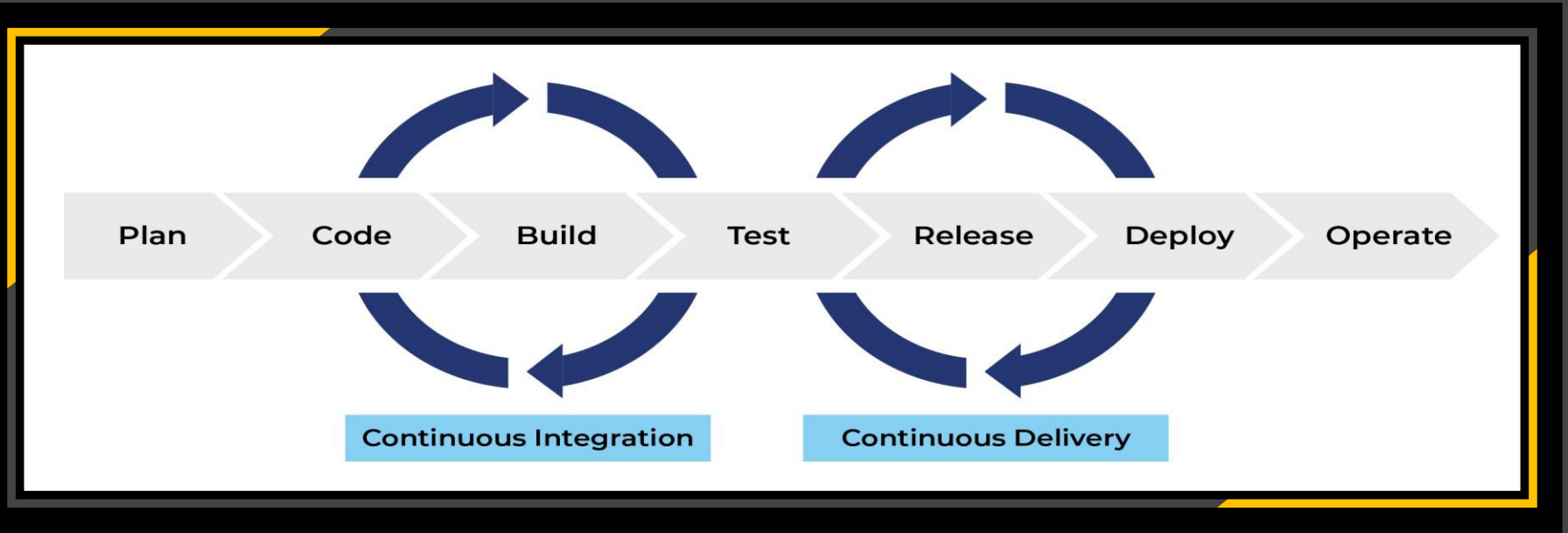
DevOps Automation

Prerequisites for DevOps

- Networking Concepts
- Knowledge of Virtualization
- Knowledge of Linux Commands
- Knowledge of Clod (AWS, Azure etc)
- Knowledge of Coding

What is CICD

CI and CD are two acronyms frequently used in modern development practices and DevOps.
CI stands for continuous integration, a fundamental DevOps best practice where developers frequently merge code changes into a central repository where automated builds and tests run. But CD can either mean continuous delivery or continuous deployment.



Thank you