DevOps Certification Training

TECHNOLOGY

Introduction to DevOps

simplilearn

Learning Objectives

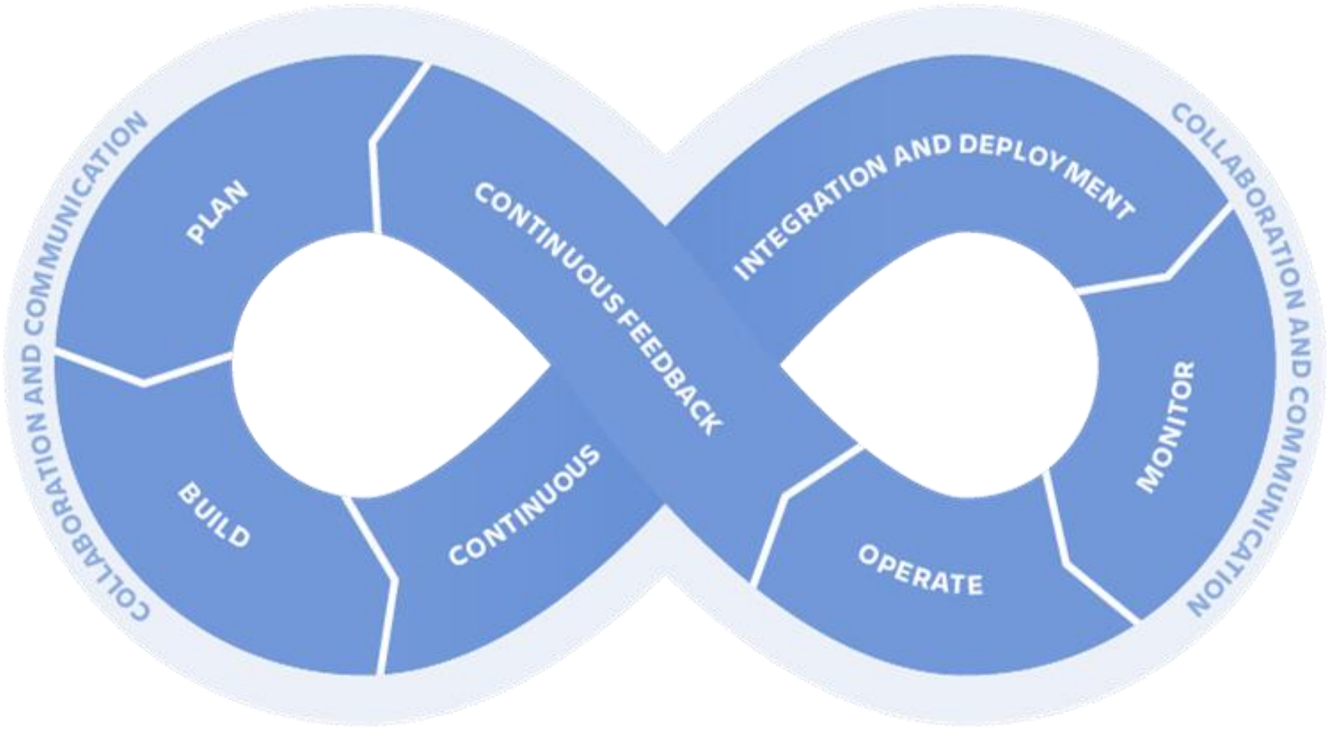
By the end of this lesson, you will be able to:

- Describe the DevOps life cycle
- Differentiate between DevOps and traditional Software Development Life Cycle (SDLC) approaches
- Explain the DevOps architecture and its tools
- Explain cloud services used in a DevOps life cycle

Getting Started

What Is DevOps?

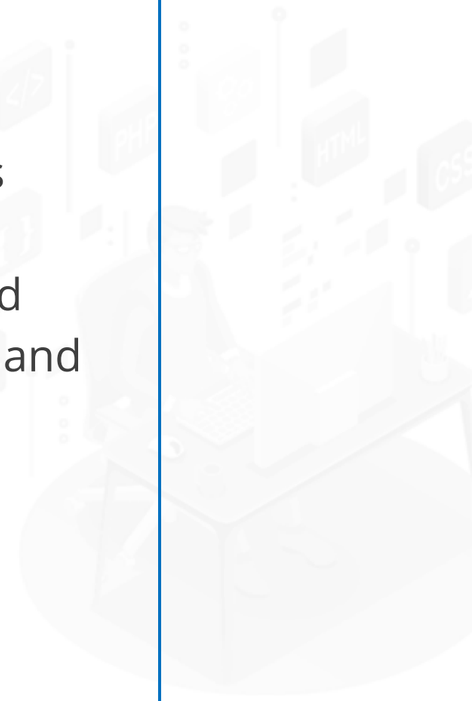
DevOps is a set of practices and tools designed to shorten the life cycle of a software development process.



Overview of DevOps

Overview

- DevOps is a combination of two words: “**Dev**elopment” and “**Op**erations”
- An agile relationship that bridges the gap between development and IT operations
- Used to automate and integrate the processes between software development and other operations so that the software can be built, tested, and released in a faster and reliable manner.
- Development includes planning, creating, verifying, and packaging.
- Operations include releasing, configuring, and monitoring.

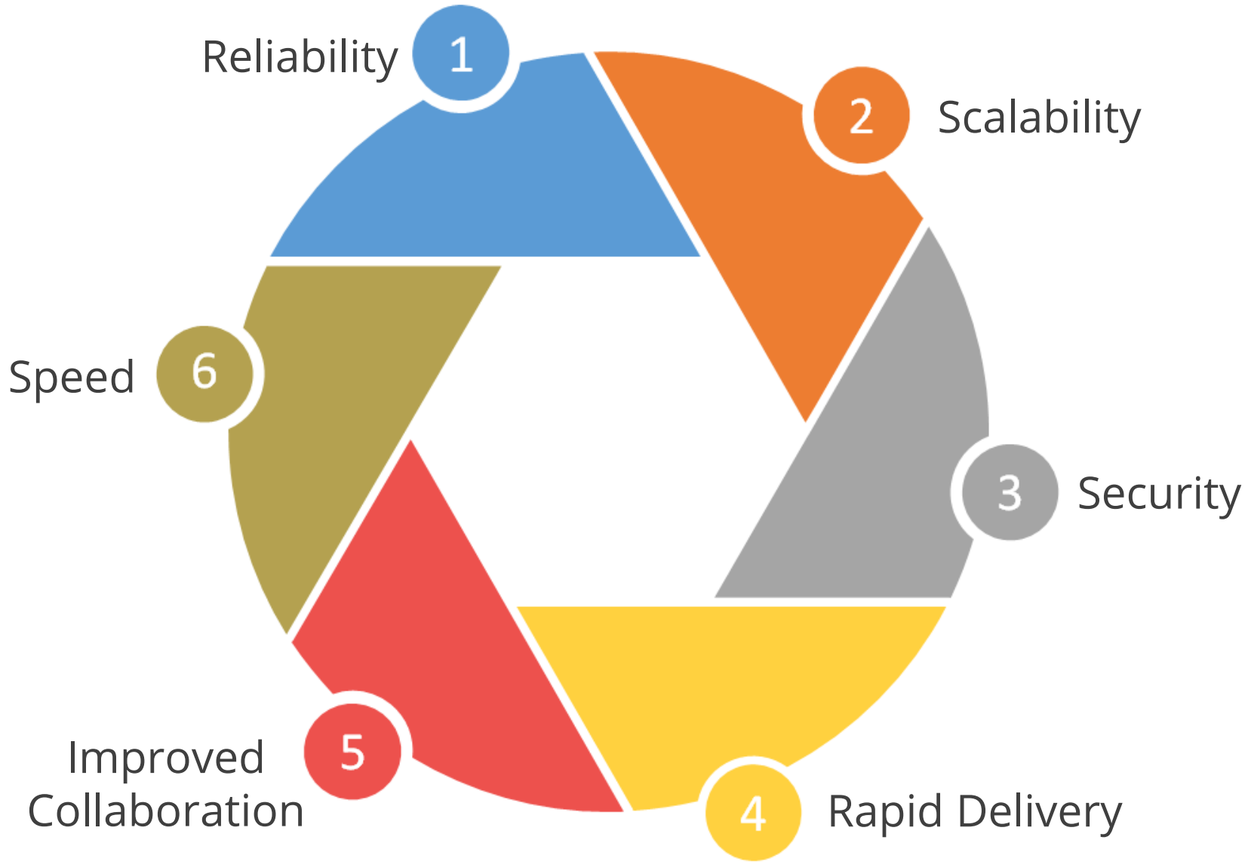


History of DevOps

History

- In 2008, the concept of DevOps emerged out as a result of a discussion between Andrew Clay and Patrick Debois, a Belgian consultant, project manager, and agile practitioner
- A presentation on “10+ Deploys per Day: Dev and Ops Cooperation at Flickr” helped in bring out the ideas for DevOps and resolve the conflict of It’s not my code, it’s your machines! ”
- DevOps blends lean thinking with agile philosophy
- It evolved from the need for adaptation and continuous improvement

Benefits of DevOps



Adopting DevOps Model

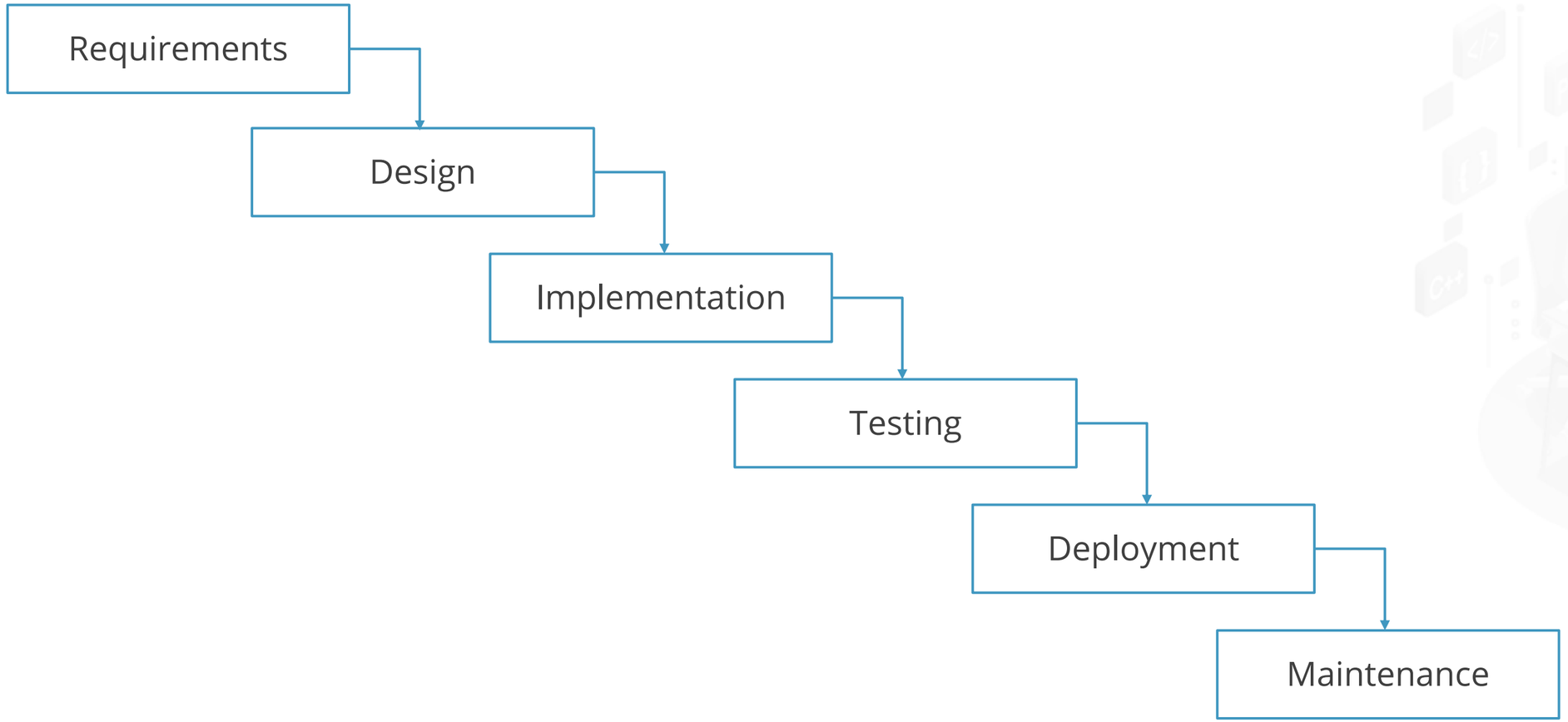
Best practices to adopt DevOps

- Continuous Integration
- Continuous Delivery
- Monitoring and Logging
- Infrastructure as Code
- Microservices
- Communication and Collaboration

DevOps and Other Frameworks

Traditional Approach

Traditional development approach has a sequence of activities for system designers and developers to plan, create, test, and deploy a software system.



Challenges in the Traditional Approach

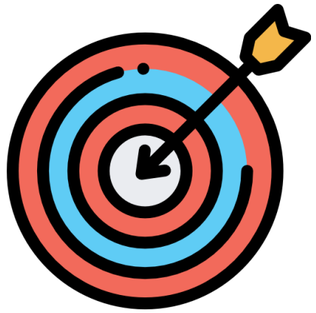
Waterfall model

Most development teams use waterfall method, which is time-consuming because of the larger size of the developer team, testers, and the code involved.

Productivity

Code that is huge is size is bundled into the release will result in jammed production and lower the productivity.

Challenges in the Traditional Approach



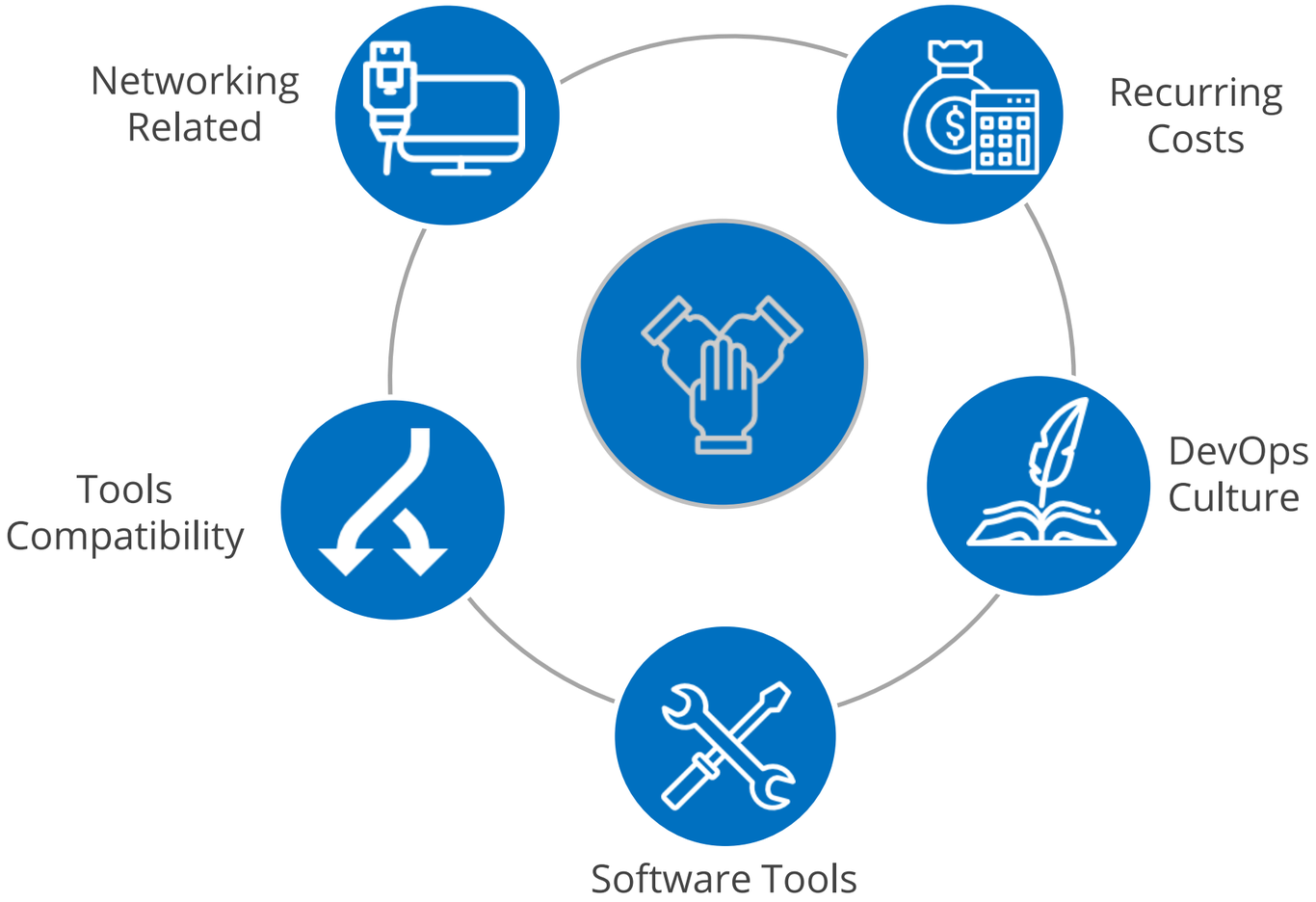
Difficult to Achieve Goal

Less investment on resources and constant work make it difficult for the developers to achieve goal or an outcome.

Investment in Planning Systems

More money is invested in schedule planning systems which are sensitive and inaccurate. As a result, it consumes more time to manage the systems.

Challenges in the Traditional Approach

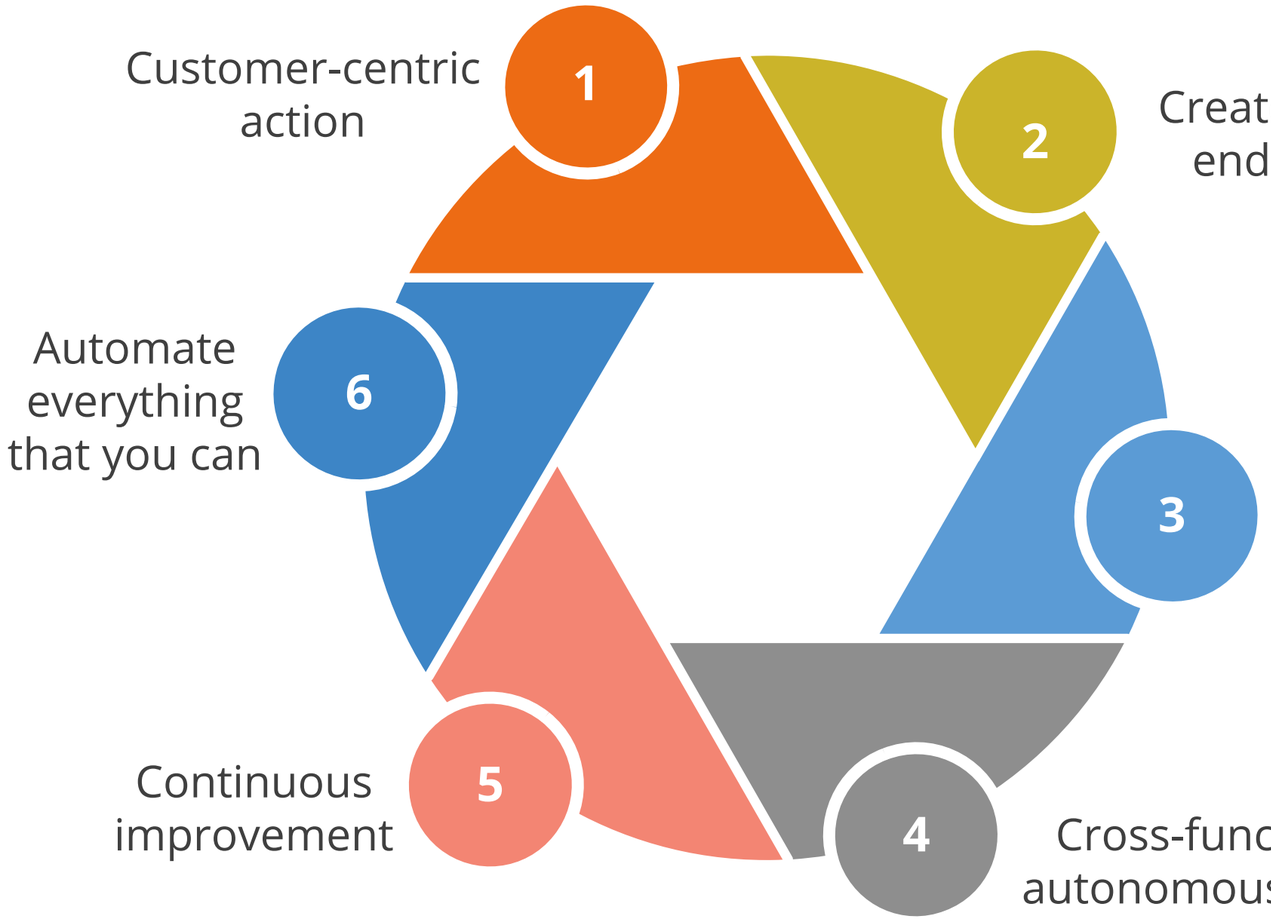


Challenges in the Traditional Approach

Constantly Changing Challenges:

DevOps Culture	Adapts to continuous changes: People's resistance, organization, culture, and pair-programming
Software Tools	Preference on tools across teams, standardization of tools across the organization in terms of licensing, version incompatibilities, maintenance and support
Tools Compatibility	Existence of legacy systems. Hybrid environments and tools must be cloud ready
Network Related Issues	Internet connectivity, Data Center connectivity, Virtual LANs, Cloud connectivity
Cost Related Issues	New hardware requirements, software licensing, training, and reduced efficiency while learning

DevOps Principles



DevOps Vs. Traditional Approach

DevOps

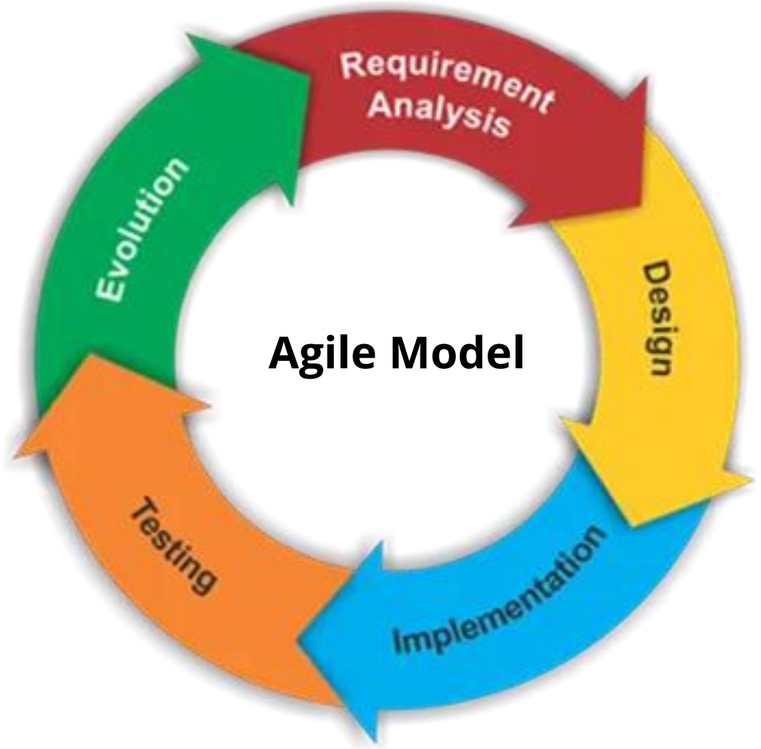
- Teams focus more on improving infrastructure
- Teams spend less time on fixing issues and recovers from failures faster
- Teams release applications at twice the speed of traditional IT teams
- Continuous integration and deployment
- All teams are equally responsible, which leads to better team engagement and productivity

Traditional Approach

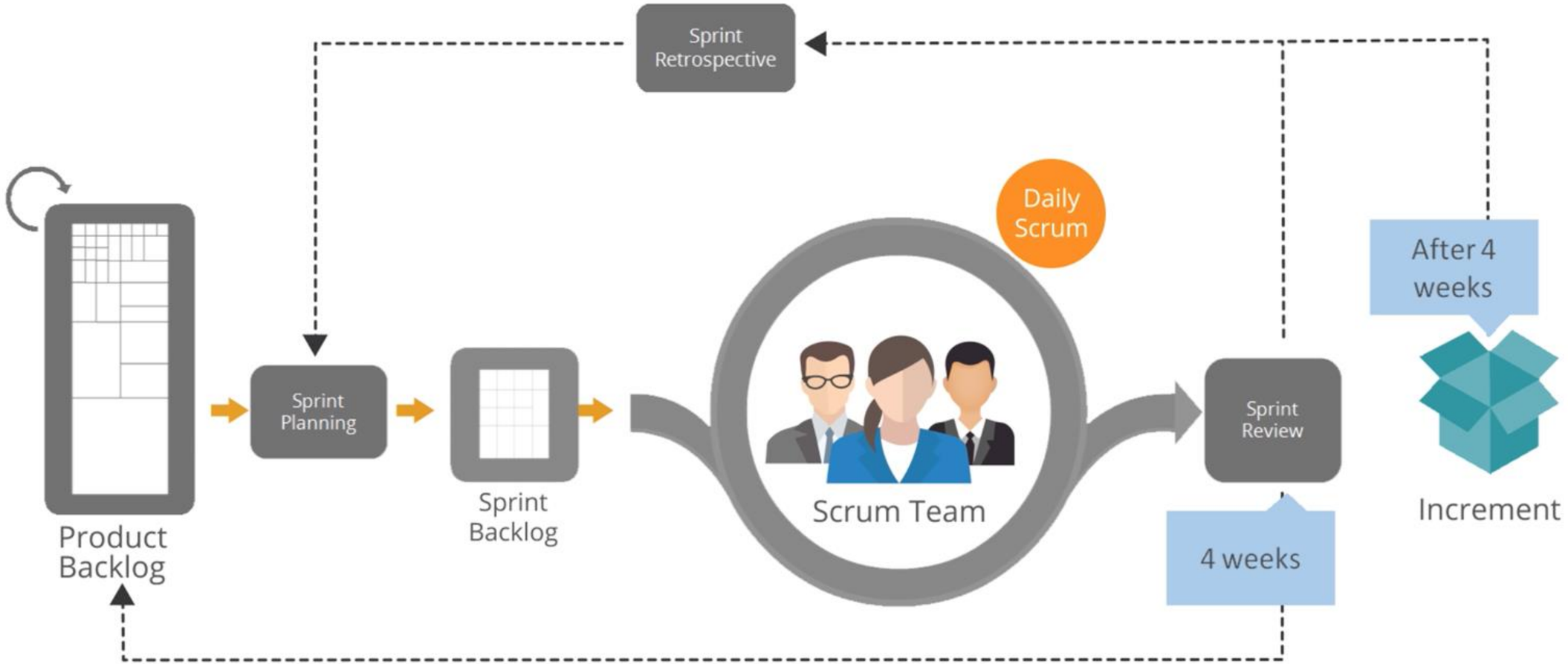
- Teams have limited focus on improving infrastructure
- Requires more time to recover from failures
- Teams requires more time to release applications
- Linear model of development, testing, and deployment
- Individual teams have have dedicated tasks, which leads to inefficient collaboration

What Is Agile?

Agile is an iterative development process that promotes continuous iteration of development and testing throughout the project life cycle.



What Is Agile?



Example of an Agile Model

Advantages of Agile

Advantages of Agile model

- Regular interactions with stakeholders and team members
- Fast and continuous delivery leads to customer satisfaction
- Continuous attention to technical excellence and better design
- Regular adaptation to changing requirements
- Continuous testing and deployment results in better quality software

DevOps with Agile

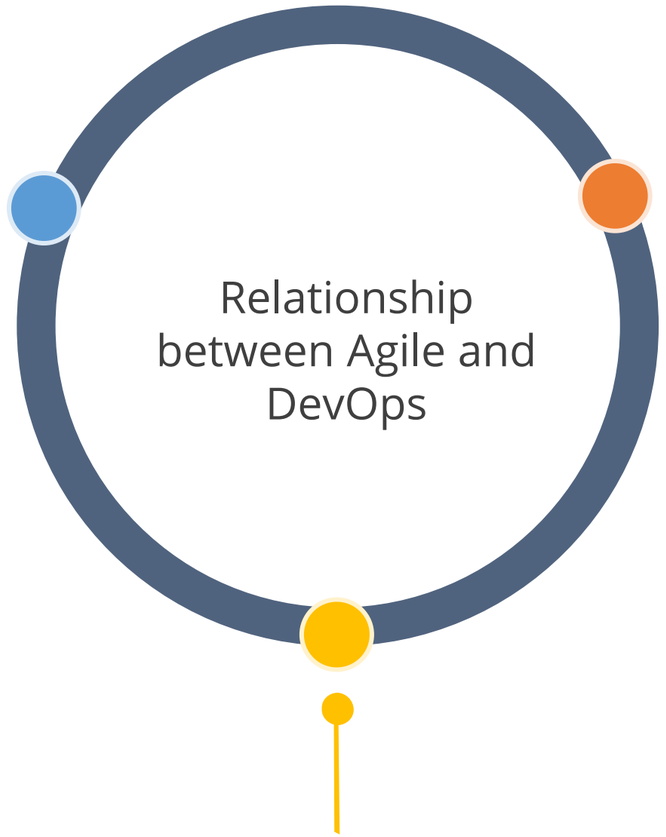
Satisfy customers through early and continuous delivery of valuable software

Deliver working software frequently with a preference for shorter timescale

Stakeholders and developers must work together throughout the project

DevOps with Agile

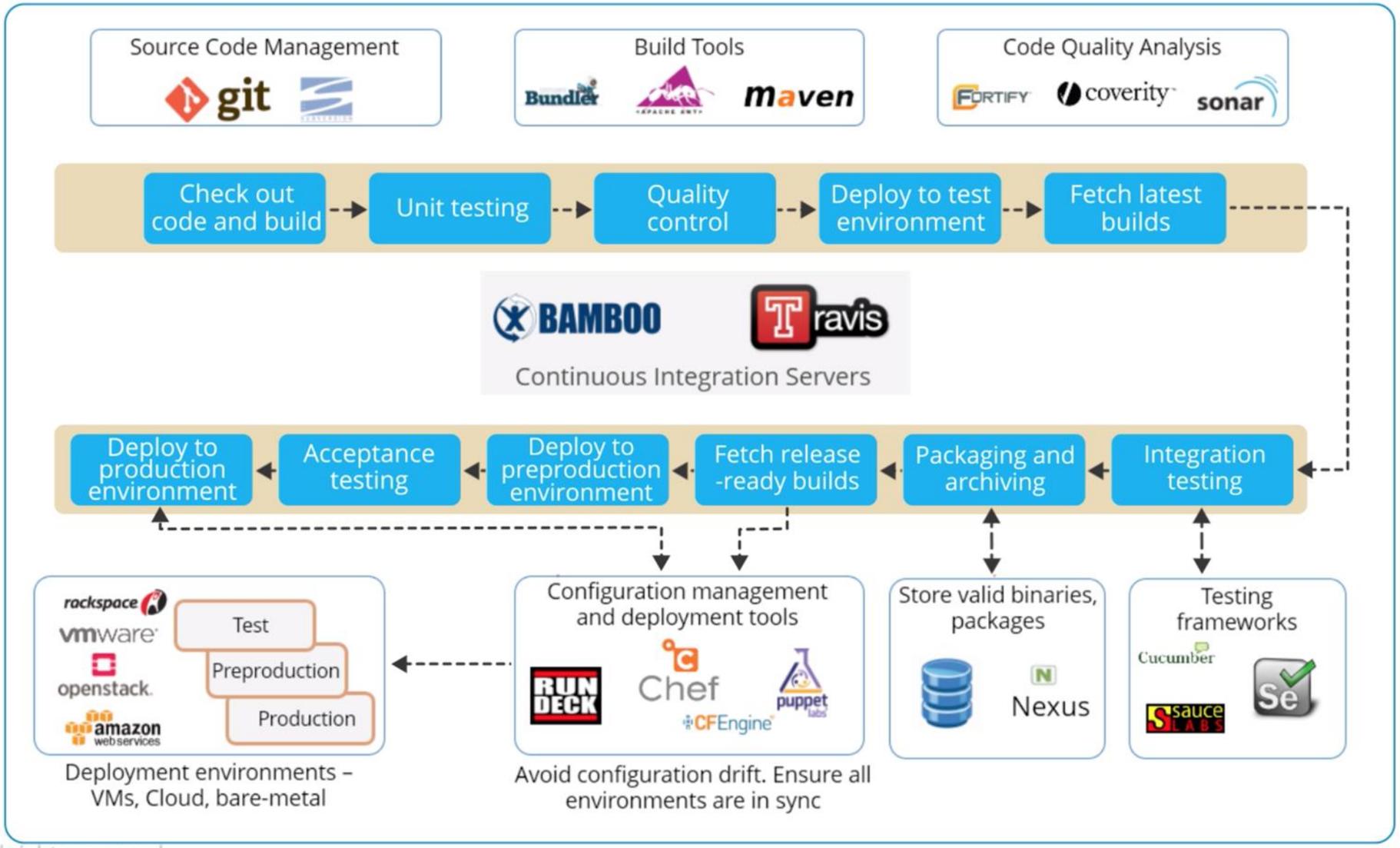
Replace non-human steps using automation tools



Improve the collaboration between the teams

Automate to create a potentially shippable increment

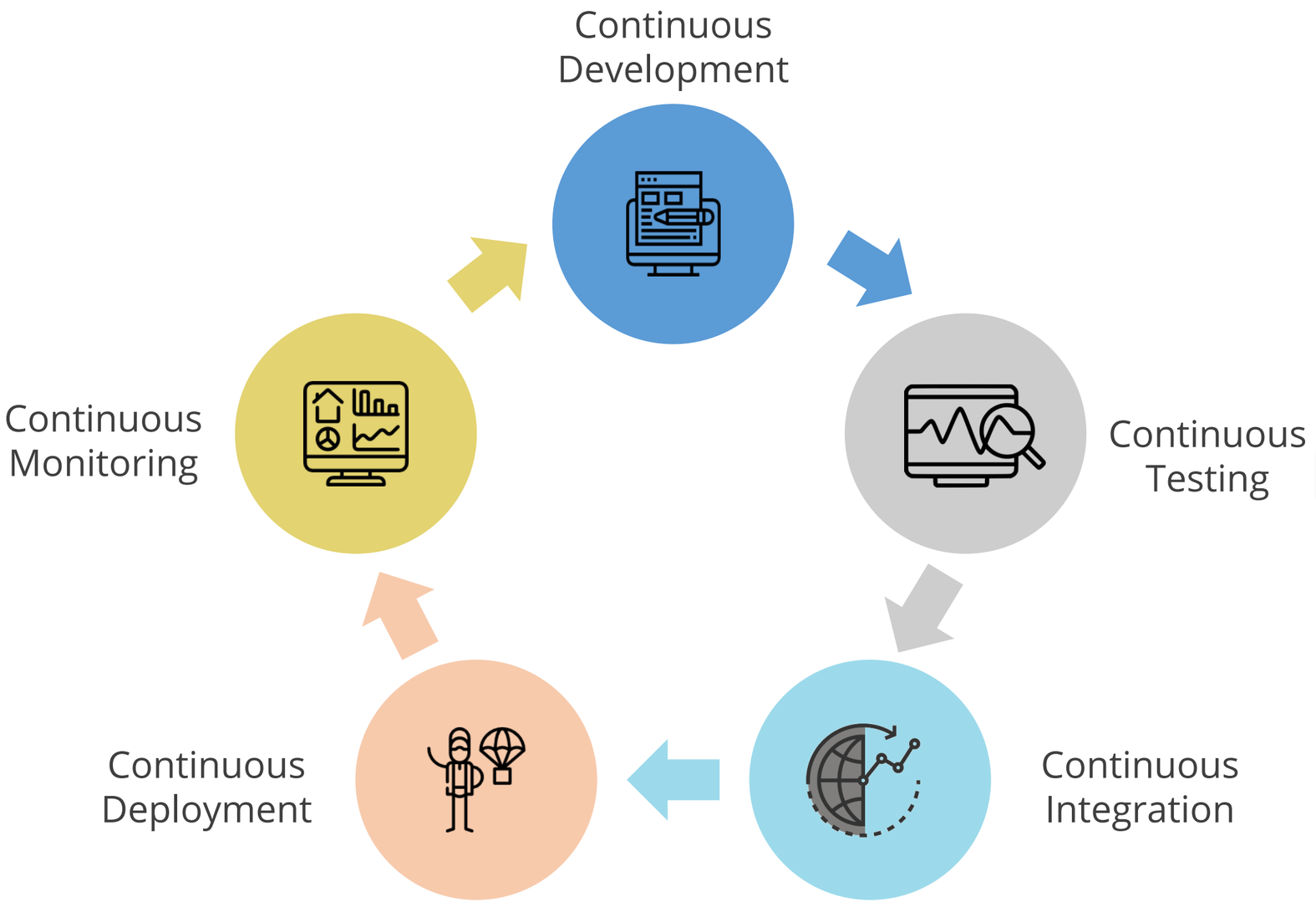
DevOps with Agile



Example of DevOps with Agile model

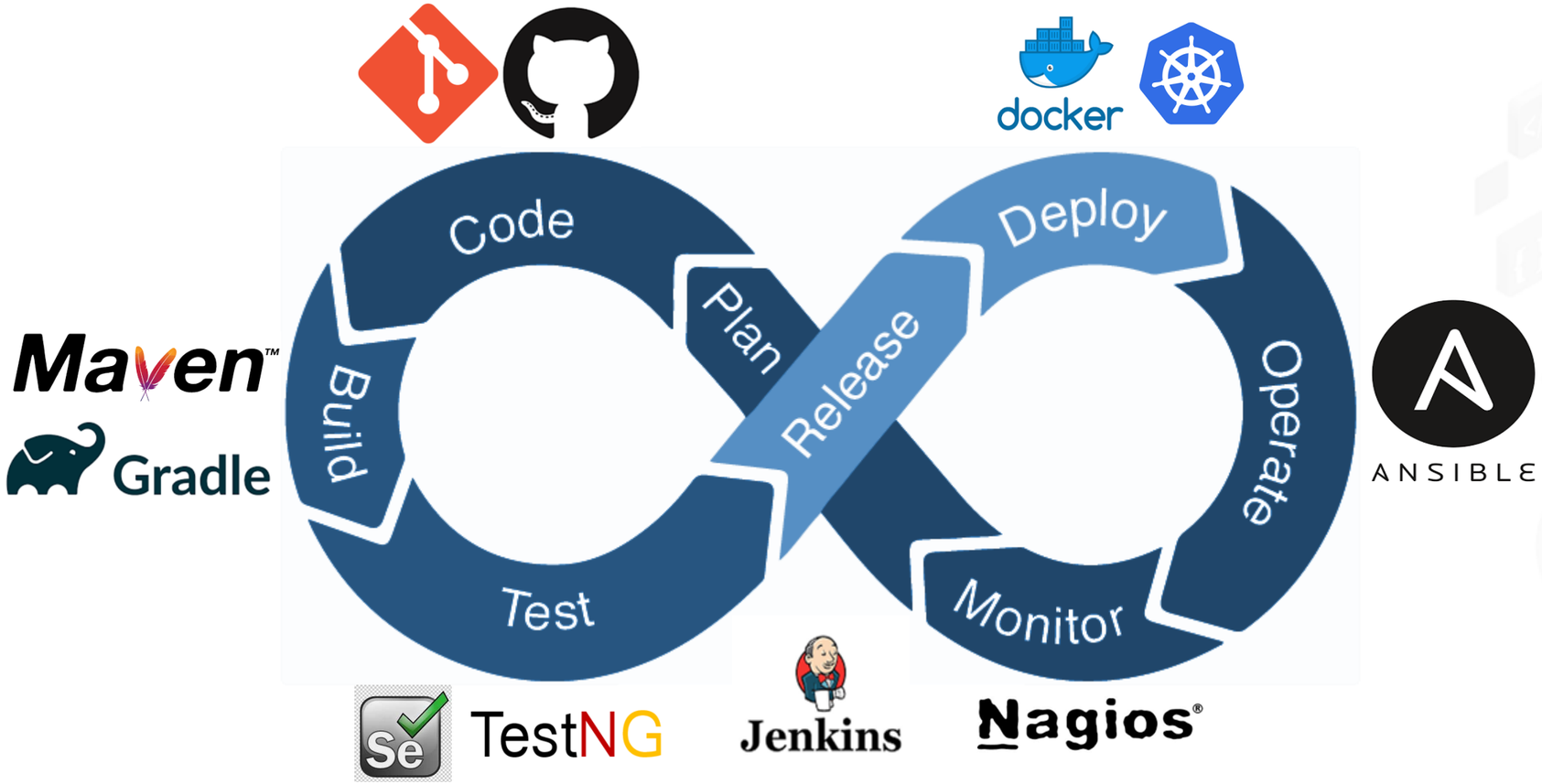
DevOps Tools

DevOps Architecture

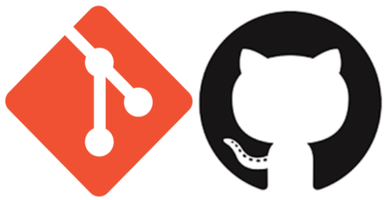


DevOps Tools

To implement DevOps and work within the DevOps life cycle, following tools required:



DevOps Tools



SCM tools

For Source-Code Management (SCM) ,version control tools such as Git, GitHub, Subversion, TFS, and Mercurial are used.

Software build tools

For automating the build process of an executable application from source code, software build tools such as Maven, Gradle, Ant, and Grunt are used.

DevOps Tools

TestNG

Testing tools

In continuous testing phase, the built software is continuously tested for bugs using testing tools such as Selenium, TestNG, and JUnit.

Jenkins

Integration tools

CI/CD pipelines are created for procuring updated source code and constructing the build into *.exe* format using tools such as Jenkins.

DevOps Tools

CMT and Deployment tools

For deployment and operations phase, CMT and automation tools such as Jenkins, AWS CodeDeploy, Chef, Puppet, Ansible, and Terraform are used.

Nagios®

Monitoring tools

For monitoring system performance and productivity, to reduce (or even eliminate) downtime, monitoring tools such as Nagios are used.

DevOps Tools

Containerization tools

For packaging an application with its required libraries, frameworks, and configuration files to efficiently run it in various computing environments, containerization tools such as Docker and Kubernetes are used.

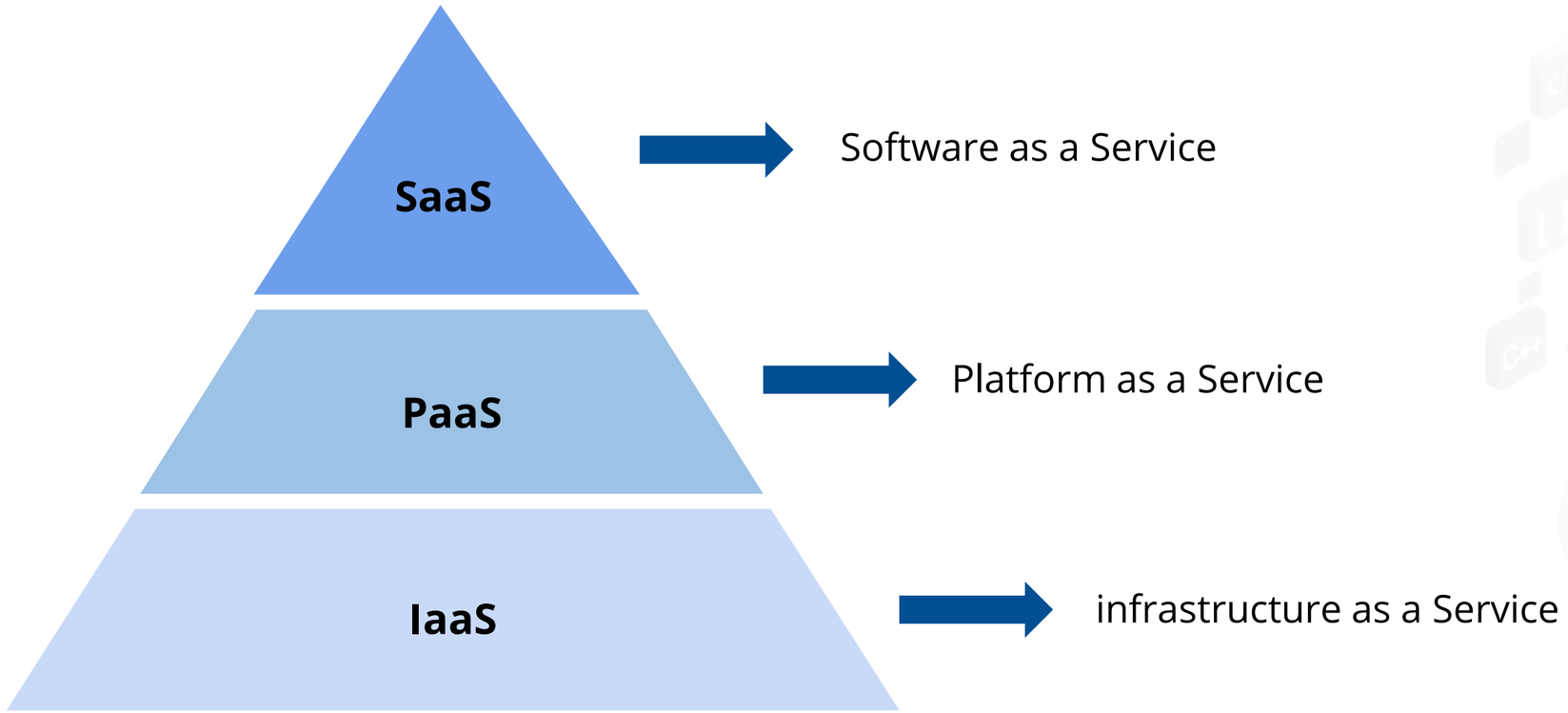
TECHNOLOGY

DevOps in Cloud Environment

simplilearn

Introduction to Cloud Service Model

Cloud computing services are categorized into three models:



Introduction to Cloud Service Model

Software as a Service

SaaS is a method of delivering applications as a service so that users are free from complex software and hardware management.

Platform as a Service

PaaS is a method where users are allowed to build, test, debug, deploy, host, and update their applications in the same environment.

Infrastructure as a Service

IaaS is an instant computing infrastructure service that provides virtualized computing resources over the internet.

Overview of Cloud Services from AWS

AWS IAM

Identity and Access Management (IAM) helps in creating and managing AWS users and groups, and use permissions to allow and deny their access to AWS resources.

AWS CodePipeline

A CI/CD service to build, test, and deploy code every time whenever is an update, based on the predefined release process models. This helps in delivering features and updates rapidly and reliably.

Overview of Cloud Services from AWS

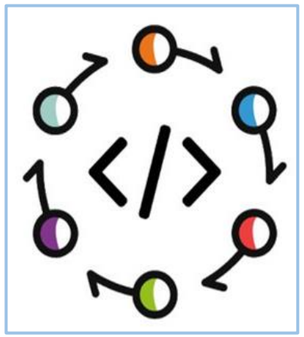
AWS CodeBuild

A fully managed build service to compile source code, run tests, and produce ready-to-deploy software packages. It scales continuously and processes multiple builds concurrently leaving no builds waiting in a queue.

AWS CodeDeploy

AWS CodeDeploy automates code deployments to any instance by avoiding downtime during application deployment and handling the complexity of updating applications.

Overview of Cloud Services from AWS



AWS CodeStar

A unified user interface to manage all software development activities in one place. An entire continuous delivery toolchain can be set up in minutes, allowing users to start releasing the code faster.

Amazon ECS

Elastic Container Service (ECS) is container management service to support Docker containers and allow users to run applications on a managed cluster of EC2 instances without any hassles.

Overview of Cloud Services from AWS

AWS Lambda

A serverless computing service that allows the users to run code without provisioning or managing servers. It takes care of everything required to run and scale code with high availability.

AWS CloudFormation

Developers and system administrators can easily create and manage a collection of AWS resources, provisioning, and updating them in an orderly and predictable fashion using AWS CloudFormation.

Overview of Cloud Services from AWS

AWS OpsWorks

A configuration management service that uses Chef to automate how servers are configured, deployed, and managed across EC2 instances or in on-premises compute environments.

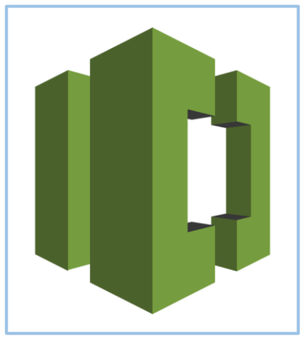
Amazon CloudWatch

A monitoring service for AWS cloud resources and applications running on AWS to collect and track metrics, collect and monitor log files, set alarms, and automatically react to changes in AWS resources.

Overview of Cloud Services from AWS

AWS Elastic Beanstalk

An easy-to-use service for automatic handling the deployment and scaling web applications and services developed with Java, .NET, Node.js, Python, Go, and Docker on familiar servers such as Apache, Nginx, Passenger, or IIS.



AWS CodeCommit

A fully-managed source control service for hosting secure and highly scalable private Git repositories to store anything from source code to binaries.

Key Takeaways

- DevOps is a set of practices and tools designed to shorten the life cycle of a software development process
- Networking and cost-related issues, tools and software compatibility, and DevOps culture are a few challenges in traditional SDLC approach
- Agile is an iterative development process that promotes continuous iteration of development and testing throughout the life cycle of a project
- Continuous development, continuous testing, continuous integration, continuous deployment, and continuous monitoring are the five phases of DevOps
- Saas, PaaS, and IaaS are the three models of cloud computing services