What is DevOps?

DevOps is a work culture primarily centered around collaboration, communication, and integration among the development teams. The term DevOps is a combination of two words namely Development and Operations. DevOps is a practice that allows a single team to manage the entire application development life cycle, that is, development, testing, deployment, Monitoring and operations.

The 6Cs of DevOps lifecycle

Continuous Development, Continuous Integration, Continuous Testing, Continuous Deployment, Continuous Monitoring and Continuous Operations constitute DevOps Life.

In this article, we will discuss the various phases involved and the some of the most commonly used tools.

Phase 1. Continuous Development

This is the phase that involves ‘planning‘ and ‘coding‘ of the software. The vision of the project is decided during the planning stage and the developers begin developing the code for the application. We will split this phase into subsection for better understanding.

1.1 Continuous Development Plan

Project requirements are gathered and discussed with stakeholders at this stage. Moreover, the product backlog is also maintained based on customer feedback which is broken down into smaller releases and milestones for continuous software development. Some of the most commonly used tools at this phase include Slack, JIRA, Blueprint, IBM etc

Many companies prefer agile practices for collaboration and use Scrum, Lean, and Kanban. Among all the tools, Jira is the most popular ones used for complex projects and the outstanding collaboration between teams while developing.

1.2 **Continuous Development Code**

Still under the development stage, the code are written in any language using an integrated development environment (IDE) like Visual Studio. These code are maintained by using Version Control tools. Maintaining the code is referred to as Source Code Management. The most popular tools used are Git, SVN, Mercurial, CVS. GitLab, TFS, BitBucket, Confluence and Subversion.

1.3 **Continuous Development Build**

Depending the team, some organization uses tools like Ant, Maven, Gradle and npm in this phase for building/ packaging the code into executable file that can be forwarded to any of the next phases.

Phase 2. Continuous integration (CI)

Continuous Integration is the most crucial phase in the entire DevOps lifecycle. In this phase, updated code or add-on functionalities and features are developed and integrated into existing code. Since there is continuous development of software, the updated code needs to be integrated continuously as well as smoothly with the systems to reflect changes to the end-users.

The developers commit changes to the source code. Every commit is then built and this allows early detection of problems if they are present. Building code not only involves compilation but it also includes code review, unit testing, integration testing, and packaging.

Jenkins is a very popular tool used in this phase. Whenever there is a change in the Git Repository, Jenkins fetches the updated code and it prepares a build of that code which is an executable file in the form of a war or a jar. This build is then forwarded to the test server or the production server.

Some of the tools commonly used to make the project workflow smooth and more productive include; Jenkin, Bamboo, GitLab CI, Buddy, TeamCity, Travis, and CircleCI

Phase 3. Continuous testing

Some teams carry out the continuous testing phase before the integration occurs, while others do it after the integration. In this article we will place this phase after the integration.

This is the phase where the developed software is continuously tested for bugs. For continuous testing, automation testing tools like Selenium, TestNG, JUnit, TestSigma etc are used. These tools allow QAs to test multiple code bases thoroughly in parallel to ensure that there are no flaws in the functionality using Docker containers. In case of a bug or an error, the code is sent back to the integration phase for modification.

Image source: Edureka

Selenium does the automation testing, and the reports are generated by TestNG. This entire testing phase can be automated with the help of a Continuous Integration tool called Jenkins. Suppose you have written a selenium code in Java to test your application. Now you can build this code using ant or maven. Once the code is built, it is tested for User Acceptance Testing (UAT). This entire process can be automated using Jenkins.

Phase 4. Continuous deployment (CD)

At this stage, the code is deployed to the production servers. Two set of tools are often used to archived CD; Configuration management and Containerization tools.

Configuration management is the act of releasing deployments to servers, scheduling updates on all servers and most importantly keeping the configurations consistent across all the servers. Since the new code is deployed on a continuous basis, configuration management tools play an important role in executing tasks quickly and frequently. Some popular tools that are used here are; Puppet, Chef, SaltStack, and Ansible.

Containerization tools plays an important role in the deployment phase. Docker and Vagrant are the popular tools used for this purpose. These tools help produce consistency across Development, Test, Staging and Production environments. Using these tools, there is no scope of errors/ failure in the production environment as they package and replicate the same dependencies and packages used in the

development/ testing/ staging environment. Besides this, they also help in scaling-up and scaling-down of instances swiftly.

Phase 5. Continuous monitoring

During this phase, the application's functionality and features are monitored continuously to detect system errors such as low memory, non-reachable server, etc. This process helps the IT team quickly identify issues related to app performance and the root cause behind it. If IT teams find any critical issue, the application goes through the entire DevOps cycle again to find the solution. However, the security issues can be detected and resolved automatically during this phase.

The popular tools used for this include Splunk, ELKStack, Nagios, NewRelic, Kibana, PagerDuty and Sensu. These tools help the team monitor the application's performance and the servers closely and also enable team to check the health of the system proactively.

Phase 6. Continuous operations

This phase is often merge with the Monitoring phase or not include in the DevOps life cycle by. The last in the DevOps lifecycle is crucial for reducing the planned downtime, such as scheduled maintenance. Generally, developers are required to take the server offline to make the updates, which increases the downtime and might even cost a significant loss to the company. Eventually, continuous operation automates the process of launching the app and its updates. It uses container management systems like Kubernetes and Docker to eliminate downtime. These container management tools help simplify the process of building, testing, and deploying the application on multiple environments. The key objective of this phase is to boost the application's uptime to ensure uninterrupted services. Kubernetes and Docker Swarm are the container orchestration tools used for the high availability of the application and to make the deployment faster.

Summary of DevOps lifecycle

DevOps lifecycle is a series of automated development processes or workflows within an iterative development lifecycle.

Brief overview of how the DevOps lifecycle works at every stage.

- **Plan:** this is the stage whereby the team identify the business requirement and collect end-user feedback. They create a project road map that will maximize the business value and deliver the desired product.
- **Code:** The code development takes place at this stage. The development teams use some tools and plugins like *Git* to streamline the development process, which helps them avoid security flaws and lousy coding practices.
- **Build:** In this stage, once developers finish their task, they commit the code to the shared code repository using build tools like Maven and Gradle.
- **Integrate:** The developers commit changes to the source code. Every commit is then built using tools like Jenkins and this allows early detection of problems if they are present.

Building code not only involves compilation but it also includes code review, unit testing, integration testing, and packaging.

- **Test:** Once the build is ready, it is deployed to the test environment first to perform several types of testing like user acceptance test, security test, integration testing, performance testing, etc., using tools like JUnit, Selenium, etc., to ensure software quality.
- **Release:** The build is ready to deploy on the production environment at this phase. Once the build passes all tests, the operations team schedules the releases or deploys multiple releases to production, depending on the organizational needs.
- **Deploy:** In this stage, Infrastructure-as-Code helps build the production environment and then releases the build with the help of different tools. The operations team at this stage takes care of server configuring and provisioning using tools like Chef.
- **Monitor:** In this stage, the DevOps pipeline is monitored based on data collected from customer behavior, application performance, etc. Monitoring the entire environment helps teams find the bottlenecks impacting the development and operations teams' productivity.

DevOps tools Summary

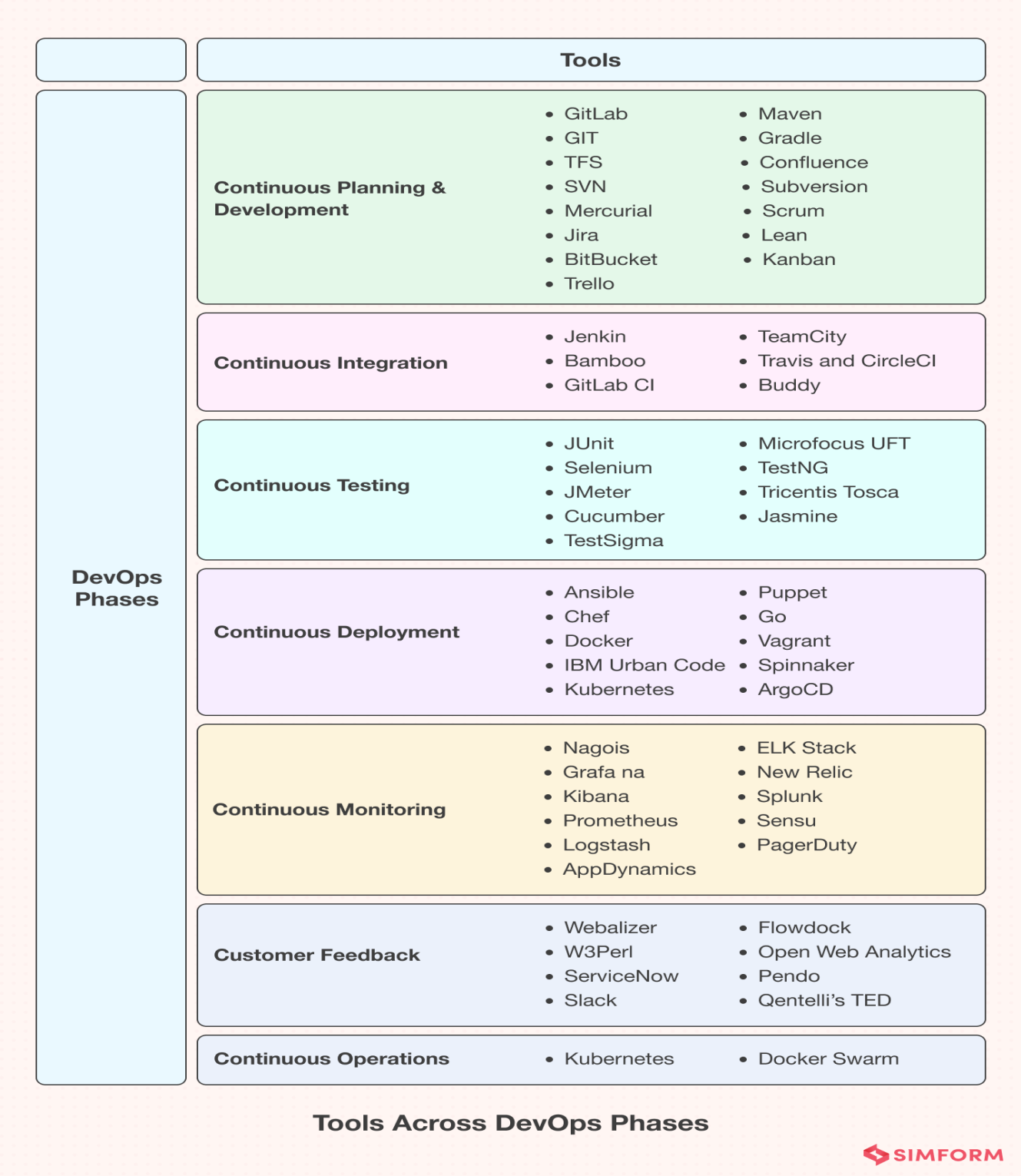

Image source: SIMFORM

Reference;

Arvind, May 24 2023: DevOps Life Cycle: 5 Different Phases of DevOps

https://www.edureka.co/blog/devops-lifecycle

Hiren Dhaduk, January; DevOps Lifecycle: 7 Phases Explained in Detail with Examples

https://www.simform.com/blog/devops-lifecycle

Damonlang Lamare. My 01 2023; Understanding DevOps Tools – Development, Testing & Deployment Technologies Involved In DevOps

https://www.edureka.co/blog/devops-tools