DevOps Lifecycle Phases

Necessity is the mother of invention. This saying rightly fits in the fast-paced and ever-evolving software industry. The customer’s demand and requirements have changed drastically with time. To keep up with it, the technology revolution is the only way forward. One such step in this direction is the **DevOps**.

The process and planning are the two most crucial parts of any software development. The software development process has gone through phases of transition with time. A few years back, **Agile** took the industry by storm. However, with changing times the new buzz word is DevOps.

**How do DevOps come into the picture?**

They say that “History helps us develop a better understanding of the world”. To know how things work in the present, one needs to evaluate how it worked in the past. Let’s delve into the history of software development to get a better understanding of the DevOps.

The two most popular models used extensively were:

   Waterfall Model

   Agile Methodology

**Waterfall Model**

It is the most popular and extensively used model. Each process in the model is pre-defined and works in a sequence. The process can only move forward after the completion of the previous phases. Different phases of the Waterfall Model are:

         Requirement

         Design

         Implementation

         Verification

         Maintenance

The rigidity of the model was the main drawback as it does not define how to handle the changes in the requirement.

**Agile Methodology**

To accommodate the changes in the requirement quickly and to identify the gaps faster between the development and the testing teams. The Agile methodology came into the picture. Different phases of Agile Methodology are:

               Plan

               Design

               Develop

               Test

               Release

               Feedback

In this model, the development and testing teams were in complete sync. However, the main challenge faced in this model was collaborating and receiving feedback from the Operations team.

Sooner, the software industries realized the need to bring the different teams together especially the Operations teams with the development and testing team. So that the challenges faced while delivering the final product can be mitigated.  That is how the DevOps process was conceptualized.

**What is DevOps?**

Let’s take an example to understand the DevOps process. A group of developers in an organization has developed an android application for sales personnel for a telecom company. The team has spent months on the development and given whole and soul for the project. The end product is working flawlessly on the development server without any glitch.

Now, the development team hands over the application to the Operations team to be used in the real-time conditions. The developer team waits anxiously for the outcome; however, their hopes came crashing down when the application does not perform as per the expectations. The application which was running perfectly on the dev server is not getting the desired results when put outside the dev environment. The Ops team blames it on the dev team for any error that comes in the code. And dev team argues that the perfectly working code is not running due to the Ops team. Now, the blame game starts as both the teams try to be defensive and argue to be right.

This is a classic example of how a loosely connected team can potentially end up in providing a substandard end result. Had there been the involvement of both the teams from the inception of the project this situation would not have occurred.

**In a nutshell, the principle idea behind the DevOps is to bring the Development and the Operations team together which were traditionally isolated from each other. Integrate the teams’ to result in faster delivery, highly efficient products, and services.**

**DevOps Process Flow (Lifecycle)**

The various phases of the DevOps process are:

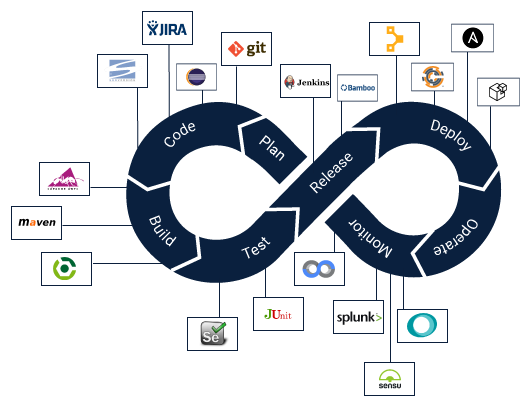
         Continuous Development

         Continuous Integration

         Continuous Testing

         Continuous Deployment

         Continuous Monitoring



**Image by [Harsh Binani** [**https://medium.com/cuelogic-technologies/what-is-devops-the-complete-guide-to-devops-with-examples-13db789dd1c**](https://medium.com/cuelogic-technologies/what-is-devops-the-complete-guide-to-devops-with-examples-13db789dd1c)**] via [Medium https://medium.com/cuelogic-technologies/what-is-devops-the-complete-guide-to-devops-with-examples-13db789dd1c]**

**Continuous Development (CD)**

Continuous Development is not just a single process; it is a collection of processes for developing the software. The processes include continuous integration, continuous testing, continuous delivery, and continuous deployment. By employing these processes throughout the lifecycle, helps in achieving better results and faster delivery.

**Continuous Integration (CI)**

The process implies a development practice where the developer’s team checks in the code regularly in the version control system. Using an integrated version control system helps in testing and analyzing the latest code in case of any conflict happens between the old and latest code. The commonly used and popular tools used are Git, JIRA, and SVN.

**Continuous Testing**

Continuous Testing is the process that allows the developers to regularly keep a check on the code. The process helps the developers to ensure that code is working as is. In this particular process, if any bug, error is found by the QA’s, the code then goes through the development stage again. Popular tools for testing the developed code are Selenium, JUnit, etc. are used.

The DevOps process employs automation testing tools such as selenium which generates the reports through TestNG (Testing framework). The testing phase is usually automated through the CI tool Jenkins.

**Continuous Deployment**

It is a fully automated process wherein the verified changes in the code are directly released to the users. The release of code in this manner helps in getting the feedback quickly to the developers.

**Continuous Monitoring**

This process ensures to keep a track of the real-time issues so that they are identified and can be solved. This helps in running a seamlessly perfect application.

The continuous collaboration among the teams will ensure a high-quality final product to the end-users at a great pace.