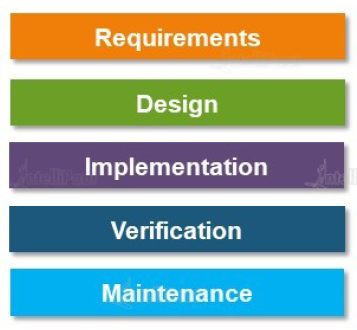
DevOps

What is Software Development?

It is a process of transforming customer requirements into a complete software project.

[What is SDLC-Software Development Life Cycle?](file:///D:\Intellipaat\DevOps\Class-1_14Aug22-Intro%20to%20DevOps%20by%20Trainer-Deepak\Class-1_14Aug22-Intro%20to%20DevOps%20by%20Trainer-Deepak\Software%20Development%20Lifecycle.mp4)

Involves following phases:



Managing application server

Artifact generated by your code, verified/validate/test

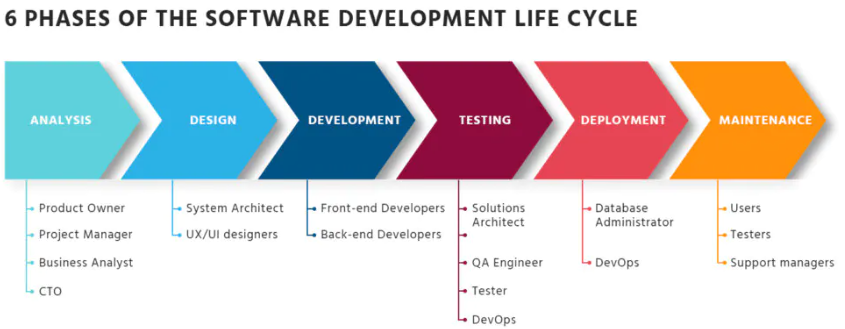
Start coding application

Blueprint

You get requirement from customer/client

Elaboration of above phases:

1. Requirements: This is where we will get requirements from customers and the requirement are then analyzed to ensure their feasibility.
2. Design: Once the requirements are received, the architect/developer transforms these requirements into technical specifications and plan the software components which have to be designed.
3. Implementation: The specifications are then passed on to the developers which create the application based on these specifications.
4. Verification: Once the development work has done on the application. It is verified by a group of testers to map the application’s functionalities with the specification given by the customers.
5. Maintenance: Once the code is verified, it is published to production. Post this, the application is updated with any future enhancements or optimizations, if and when required.



1. Analysis:

**Goal:**To gather requirements and define the direction of the software engineering process.

**Outcome:**Setting requirements and writing SRS

1. Design:

**Goal:** To convert requirements into detailed software architecture.

**Outcome:** Software design description.

1. Development:

**Goal:** Translate the design of the system into code.

**Outcome:** testable, functional software, and a Source Code Document.

1. Testing:

**Goal:**Code verification and bugs detection.

**Outcome:** Software is completely free of bugs and compliant.

1. Deployment:

**Goal:** Software delivery to a target device.

**Outcome:** Fully operational software in a live environment.

1. Maintenance:

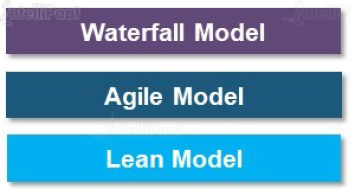
**Goal:** Ongoing security monitoring and update.

**Outcome:**Utter user experience and productivity.

Conclusion:

These steps are roughly the same from one software development life cycle model to another.

Software development models



[Waterfall Model](file:///D:\Intellipaat\DevOps\Class-1_14Aug22-Intro%20to%20DevOps%20by%20Trainer-Deepak\Class-1_14Aug22-Intro%20to%20DevOps%20by%20Trainer-Deepak\Waterfall%20Model_SDLC.mp4):



The [Waterfall model](file:///D:\Intellipaat\DevOps\Class-1_14Aug22-Intro%20to%20DevOps%20by%20Trainer-Deepak\Class-1_14Aug22-Intro%20to%20DevOps%20by%20Trainer-Deepak\Waterfall%20Model.mp4) is the earliest SDLC approach that was used for software development. The waterfall Model illustrates the software development process in a linear sequential flow. This means that any phase in the development process begins only if the previous phase is complete.

When to use SDLC Waterfall Model?

Waterfall Methodology can be used when:

* Requirements are not changing frequently
* Application is not complicated and big
* Project is short
* Requirement is clear
* Environment is stable
* Technology and tools used are not dynamic and is stable
* Resources are available and trained

Advantages/Disadvantages of Waterfall model:

|  |  |
| --- | --- |
| Advantages | Disadvantages |
| Before the next phase of development, each phase must be completed | Error can be fixed only during the phase |
| Suited for smaller projects where requirements are well defined | It is not desirable for complex project where requirement changes frequently |
| Should perform quality assurance test (Verification and Validation) before completing each stage | Testing period comes quite late in the developmental process |
| Elaborate documentation is done at every phase of the software’s development cycle | Documentation occupies a lot of time of developers and testers |
| Project is completely dependent on project team with minimum client intervention | Clients valuable feedback cannot be included with ongoing development phase |
| Any changes in software are made during the process of the development | Small changes or errors that arise in the completed software may cause a lot of problems |

[Agile Model](file:///D:\Intellipaat\DevOps\Class-1_14Aug22-Intro%20to%20DevOps%20by%20Trainer-Deepak\Class-1_14Aug22-Intro%20to%20DevOps%20by%20Trainer-Deepak\Agile%20Model.mp4):

[What is Agile?](file:///D:\Intellipaat\DevOps\Class-1_14Aug22-Intro%20to%20DevOps%20by%20Trainer-Deepak\Agile%20Model_SDLC.mp4)

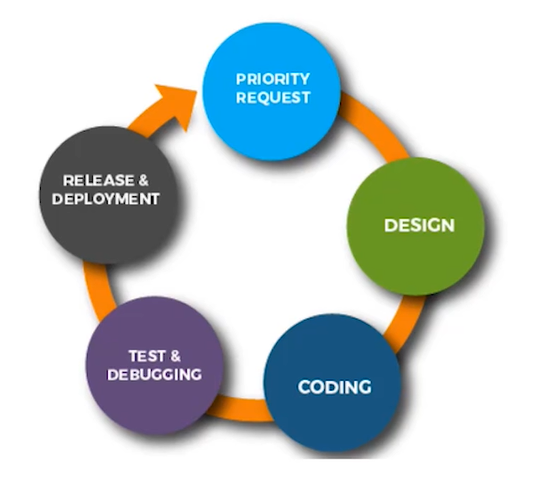
* The meaning of Agile is swift or versatile (Move Quickly).
* Agile methods break tasks into smaller chunks called iterations, or parts do not directly involve long term planning.
* Each iteration is considered as a short time "frame" in the Agile process model, which typically lasts from one to four weeks.
* Each iteration involves a team working through a full software development life cycle including planning, requirements analysis, design, coding, and testing before a working product is demonstrated to the client.
* Worked on all components parallelly.

Diagram is basically a release cycle of Agile Methodology.

1st stage is priority request

What happened in priority request is, prioritized on the feature of customer want. Say, the customer gave us the request have 15 features so you would pickup 3 or 4 features initially (what the feature is most prioritized by the customer) and you start working on them.

Basically, you won’t be working on all features simultaneously. First you pick some of them as per priority and deliver them to customer shortly.

2nd stage is Design

Once the requirements are received and prioritized, the architect/developer transforms these requirements into technical specifications and plan the software components which have to be designed initially.

3rd stage is Coding

The specifications are then passed on to the developers which create the application based on these specifications. As not all the features are to be developed initially, only the prioritized features are build to code and represent in front of customer to received its feedback.

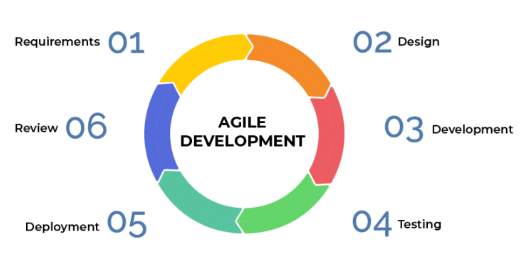
4th stage is Testing/Debugging

Once the development work has done on the application. It is verified by a group of testers to map the application’s functionalities with the specification given by the customers.

5th stage is Release and Deployment

Once the code is verified, it is published to production. Post this, the application is updated with any future enhancements or optimizations, if and when required.

Phases of Agile Model:



1. Requirements gathering: In this phase, you must define the requirements. You should explain business opportunities and plan the time and effort needed to build the project. Based on this information, you can evaluate technical and economic feasibility.

2. Design the requirements: When you have identified the project, work with stakeholders to define requirements. You can use the user flow diagram or the high-level UML diagram to show the work of new features and show how it will apply to your existing system.

3. Construction/ iteration: When the team defines the requirements, the work begins. Designers and developers start working on their project, which aims to deploy a working product. The product will undergo various stages of improvement, so it includes simple, minimal functionality.

4. Testing: In this phase, the Quality Assurance team examines the product's performance and looks for the bug.

5. Deployment: In this phase, the team issues a product for the user's work environment.

6. Feedback: After releasing the product, the last step is feedback. In this, the team receives feedback about the product and works through the feedback.

Advantages of Agile Model:

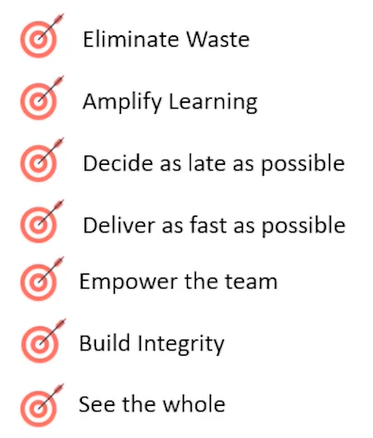
* Customer satisfaction is high,
* Less planning is required,
* Requirements can be dynamic in nature,
* Functionality can be created and tested quicky.

Disadvantages of Agile Model:

* Not suitable for handling complex dependencies in projects,
* Knowledge transfer to colleagues can be difficult since there is little documentation,
* Success of the project depends heavily on customer interaction.

Lean Methodology:

There are 7 principals of Lean methodology:



Customer and the employee will be happy overall.

Work culture should be integral, employee should feel yes, they are working in a great company.

Team becomes more expert in the kind of work in their domains.

Focus on work and deliver as fast as possible

Do not waste time in planning, focus on the work as much as possible.

If there is a new bee in the team, all what senior management have to do just assigned him a domain and let the team has to groom him in new domain.

Do not waste time in knowledge transfer/big meetings, working with people who are much experienced in their technology domain.

* Lean methodology carrying the same advantages as of Agile Methodology.
* Creates a positive work environment
* Customer feedback is given the utmost importance
* Limiting wastes saves time and money.

Disadvantages of Lean Methodology:

* Largely dependent on the skill set of the team, therefore requires a strong team
* No room for errors
* Success of the project depends heavily on customer interaction

Waterfall Vs Agile Vs Lean



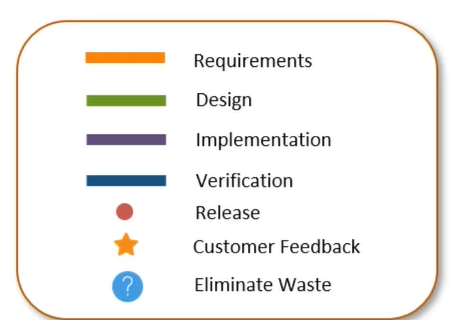
Let us assumed that the development takes a year, after a year Release will happen and finally at last customer feedback received.



Break the task in four months duration, that means Release will be happens after every four months and simultaneously received customer feedback.



Just similar as Agile Model, only it eliminates the waste after every four months of project

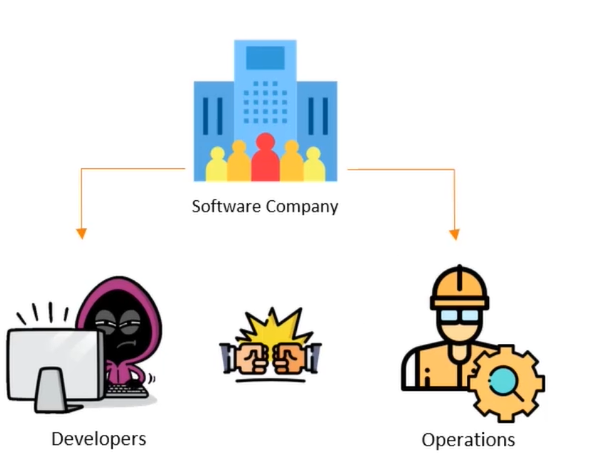


Summarizing:

Problem with Waterfall model was, the development lifecycle took a lot of time to complete. Therefore, by the time finished product was delivered, the customer requirements were no longer the same.

The problem was fixed by Lean and Agile Methodologies. These methodologies strictly focused on customer feedback and improving the software quality that too in a shorter development lifecycle.

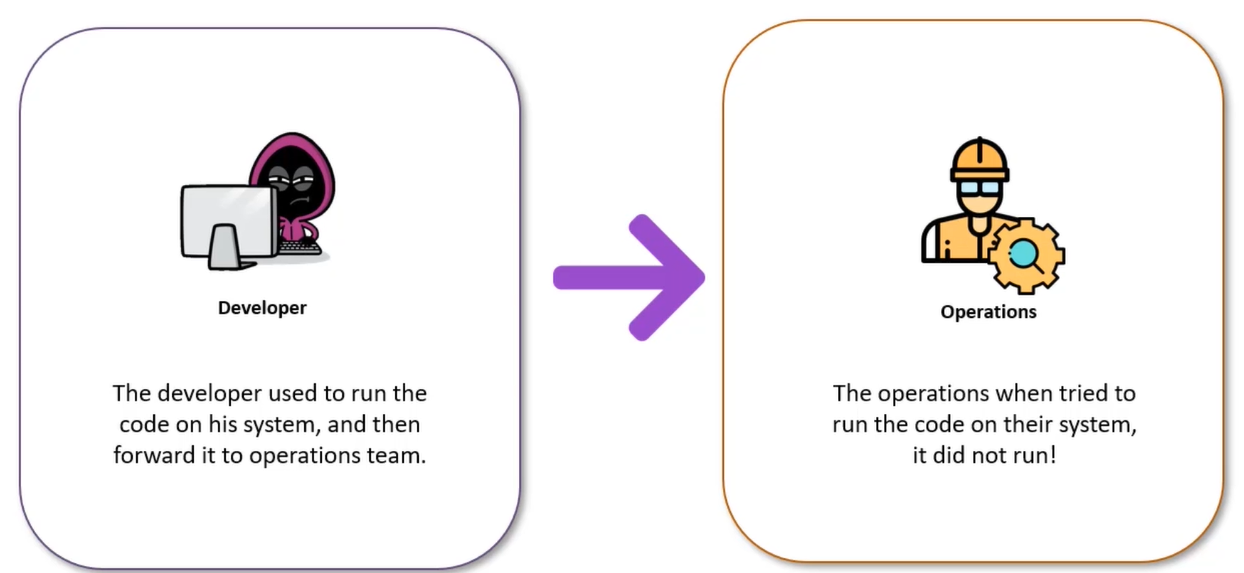
Why DevOps?

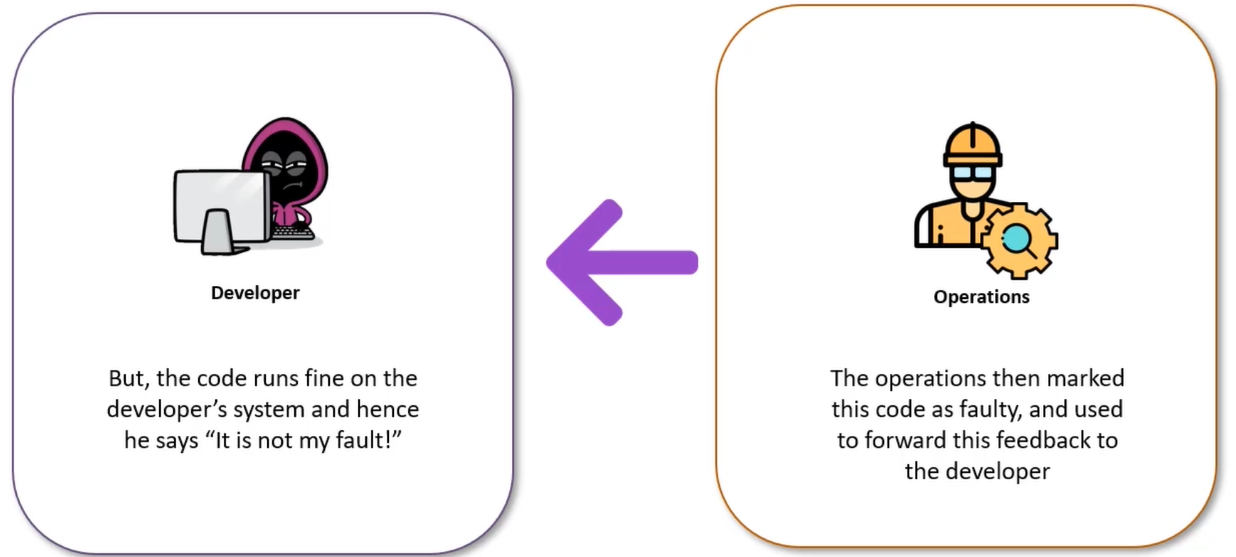


A typical software development consists of Developers and Operations employees.

Let us their job roles.

Where is the problem now?

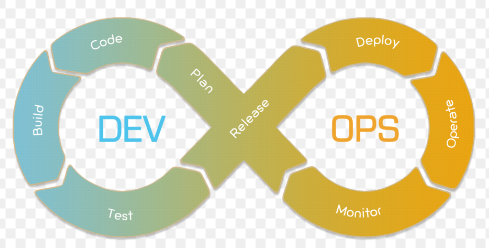


Then…the operation’s guy returns the code to developer with remark that the code is not run on his system there might be an error in code.

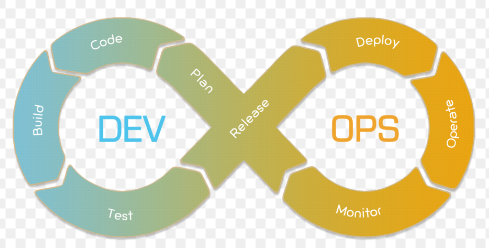
At here what is happening, the developer guy’s job role is not to fix the system error of tester/operation guy. The code is working fine into his system. But, on the other hand the operation’s guy is also not intended to make changes in his system every time to check the code from other developers, If the code is not running on his system.

This led to a lot of back and forth between the developer and operations team, hence impacted efficiency.

The problem was solved using DevOps!

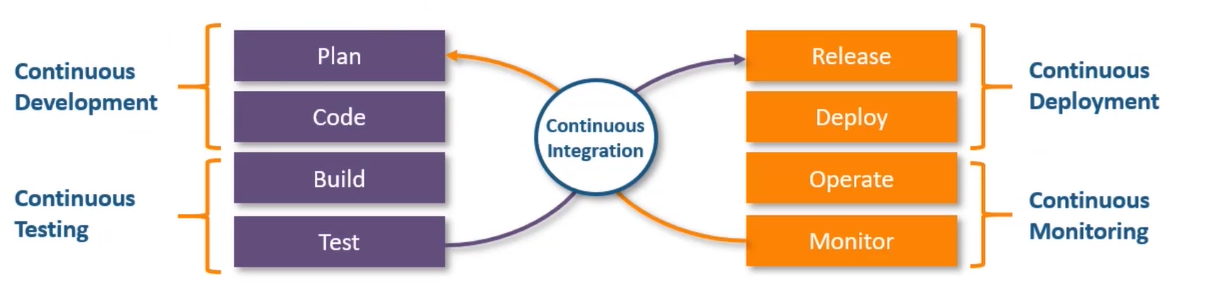
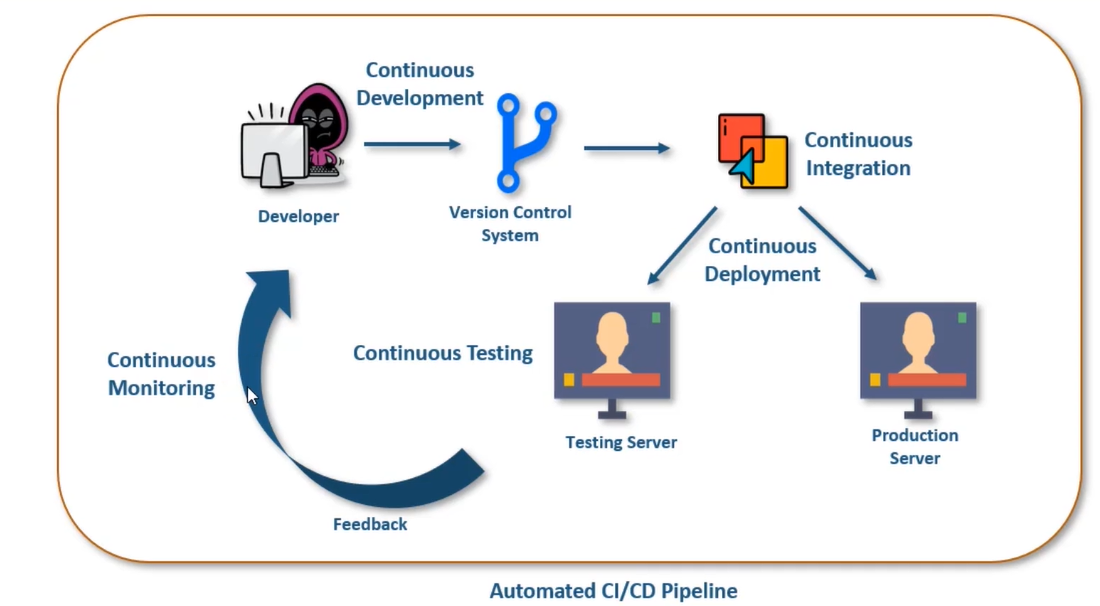


What is DevOps?

DevOps is a methodology of solving the problem between a developer and operation team using various automation tools.

These automation tools are implemented using various stages which are the part of the DevOps Lifecycle.

How DevOps works?

The DevOps Lifecycle divides the SDLC lifecycle into the following stages:

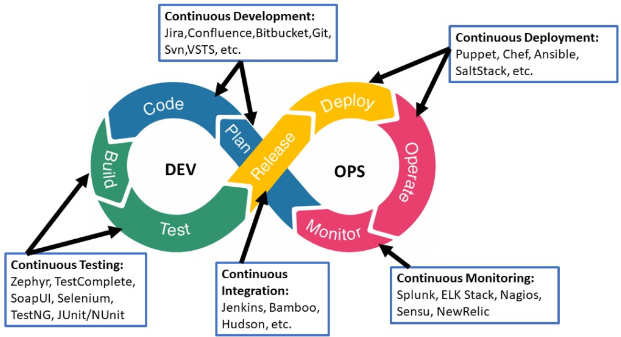
Developer starts coding, the versions of the code are stored in a version control system. The continuous integration tool than test the code if found bug-proof than it directs towards production server, else it collects feedback and through continuous monitoring it sends back the code to developer.

Again, the developer makes corrections and the whole process repeats again and again till we get desired artifacts.

Now let us understand each step one by one…

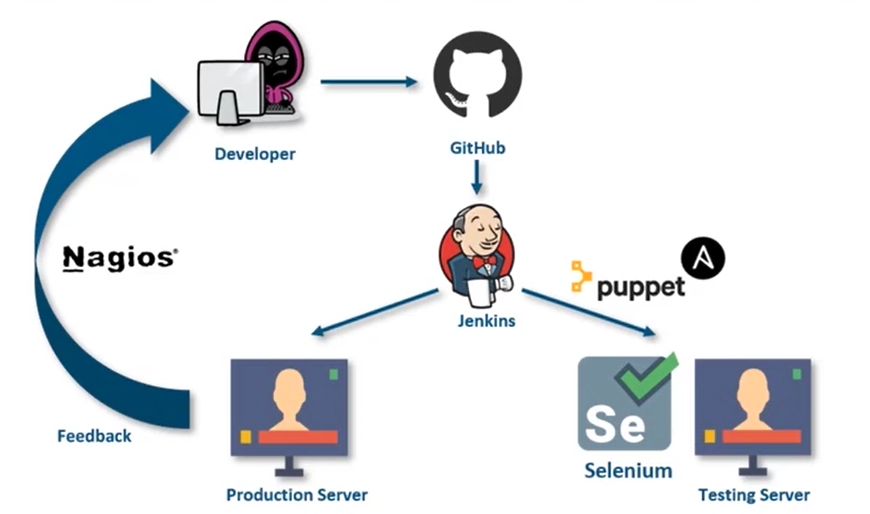
* 1. Continuous development: This stage involves committing code to version control system like Git or SVN for maintaining the different versions of code, and tools like Ant, Maven, Gradle for building/packaging the code into an executable file that can be forwarded to QAs for testing.
  2. Continuous Integration: It deals with integrating the different stages of the DevOps lifecycle, and is therefore the key in automating the whole process.
  3. Continuous Deployment: In this stage code is deployed to a particular server, it could be a testing server or production server. The key processes in this stage are Configuration Management, Virtualization, and Containerization.
  4. Continuous Testing: This stage deals with automated testing of application pushed by the developer. If the code has error, the message is sent back to the integration tool, this tool in turns notifies the developer of the error. If the code is error free, the message is sent to integration tool which pushes the code on the production sever.
  5. Continuous Monitoring: The stage is continuously monitors the deployed application for bugs or crashes. It can also be setup to collect user feedback. The collected data is then sent to developers to improve the application.

DevOps Tools



1. Continuous Development: Git is most popular distributed version control system for tracking changes in computer files and coordinating work on those files among multiple people.

It is primarily used for source-code management in software development.

1. Continuous Integration: Jenkins is an open-source automation server written in Java. It helps to automate the non-human part of software development process, with continuous integration and facilitating technical aspects of continuous delivery.
2. Continuous Deployment: Basically, it means two things. One is configuration management and another is virtualization/containerization. We use Docker for containerization, Puppet or Ansible for configuration management.
3. Continuous Testing: Selenium is a portable software testing framework used for web applications. It is an open-source tool used for automating the tests carried out on web browsers (Web applications are tested using any web browser).
4. Continuous Monitoring: Nagios is an open-source DevOps tool which is used for monitoring systems, networks and infrastructure. It also offers monitoring and alerting services for any configurable event.

**Nagios monitors the system, and the dashboard shared all detail helpful for developers.**

**If the code is bug free, pushes to production server**

**It pulls the code, and pushes to testing server**

**From GitHub, Jenkins Detects changes in code**

**Pushes code to**