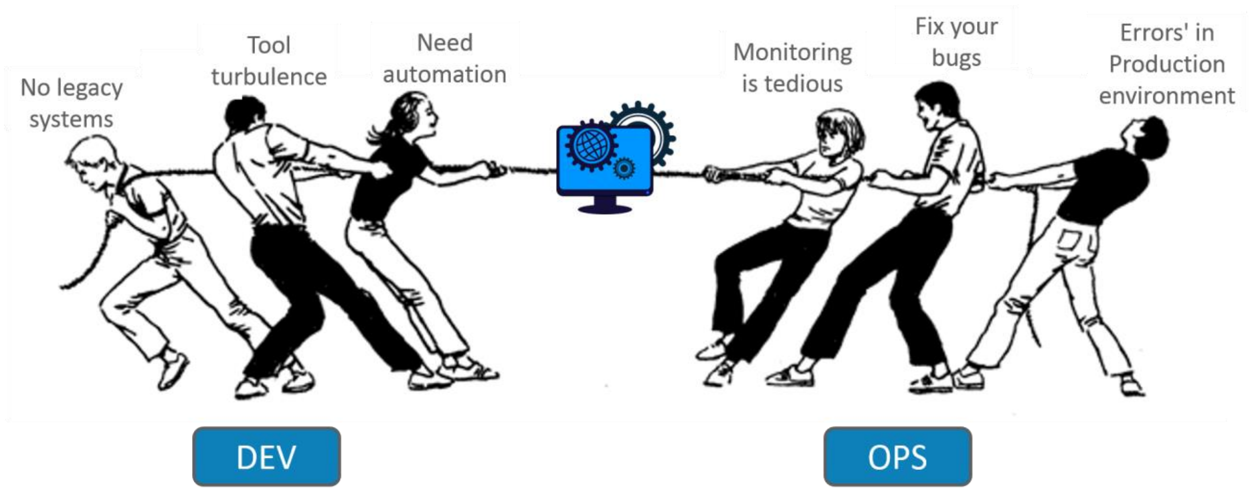
Intro to DevOps

The main intentions of DevOps is to create ease in work and provide collaboration between Development and Operations team. It is used for providing superior-quality software, more quickly and with more reliability while causing greater communication and collaboration between these teams.

In-fact, it is not the collaboration between these two teams which help deliver better software, but the oneness between 'Dev' & 'Ops' teams which results in improved software, delivered at a greater velocity. And let's not forget the role played by DevOps tools for achieving automation.

The Blame Game

When a client has complained about a software, the blame is internally thrown at each other. The 'Dev' team would point fingers at the 'QA' team. 'QA' team will then point fingers at the 'ITOps' team, who would redirect the blame to the 'Dev' team.



Irrespective of the problem residing in the code developed, or on the systems where the code is deployed, the problem remains in isolation, as nobody wants to take ownership for the screw-up.

Dev wants:

- Continuous Change
- Add new features

Ops want:

- Continuous Stability
- Create new services

Solution Automation:

Once you understand your culture, you can start with automation. Now, you can finalize various tools to achieve automation for DevOps.
Tools:

- release management
- provisioning
- configuration management
- systems integration
- monitoring and control
- orchestration become important pieces for DevOps.

Why Automation?

- Machines are good at doing the same task repeatedly
- Consistent and known state
- Fast and efficient
- Saves a lot of time

What can be automated?

- Builds
- Deployments
- Testing
- Monitoring
- Self-healing
- System rollouts
- System configuration

Technical benefits:

- Continuous software delivery
- Less complex problems to fix
- Faster resolution of problems

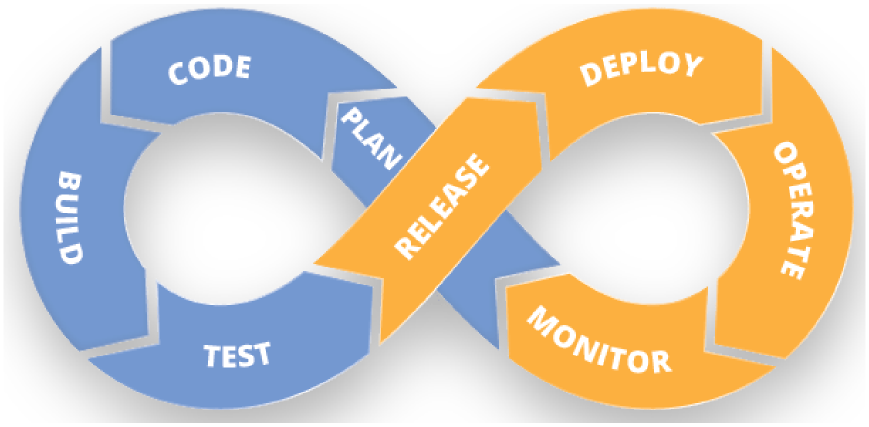
Business benefits:

- Faster delivery of features
- More stable operating environments
- More time available to add value (rather than fix/maintain)

What is DevOps?

DevOps is a software development approach which involves:
1.Continuous Development
2.Continuous Testing
3.Continuous Integration
4.Continuous Deployment
5.Continuous Monitoring

These activities are possible only in DevOps, not Agile or waterfall, and therefore Facebook and other top companies have chosen DevOps as the way forward for their business goals. DevOps is the preferred approach to develop high quality software in shorter development cycles which results in greater customer satisfaction. Check out the below video on What is DevOps before you go ahead.



Continuous Development:

Continuous deployment is a strategy for software releases wherein any code commit that passes the automated testing phase is automatically released into the production environment, making changes that are visible to the software's users. This stage involves the Coding and Building phases and makes use of tools such as Git for maintaining the different versions of the code, and tools like Ant, Maven, Gradle for building/ packaging the code into an executable file that can be forwarded to the QAs for testing.

Continuous Testing:

This is the stage where the developed software is continuously tested for bugs. For Continuous testing, automation testing tools like Selenium, TestNG, JUnit, etc. are used. These tools allow the QAs to test multiple code-bases thoroughly in parallel to ensure that there are no flaws in the functionality. In this phase, use of Docker containers for simulating ‘test environment’ on the fly, is also a preferred choice. Once the code is tested, it is continuously integrated with the existing code.

Continuous Integration:

This is the stage where the code supporting new functionality is integrated with the 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 changed code, should also ensure that there are no errors in the runtime environment, allowing us to test the changes and check how it reacts with other changes.
Jenkins is a very popular tool used for Continuous Integration. Using Jenkins, one can pull the latest code revision from GIT repository and produce a build which can finally be deployed to test or production server. It can be set to trigger a new build automatically as soon as there is a change in the GIT repository or can be triggered manually on click of a button.

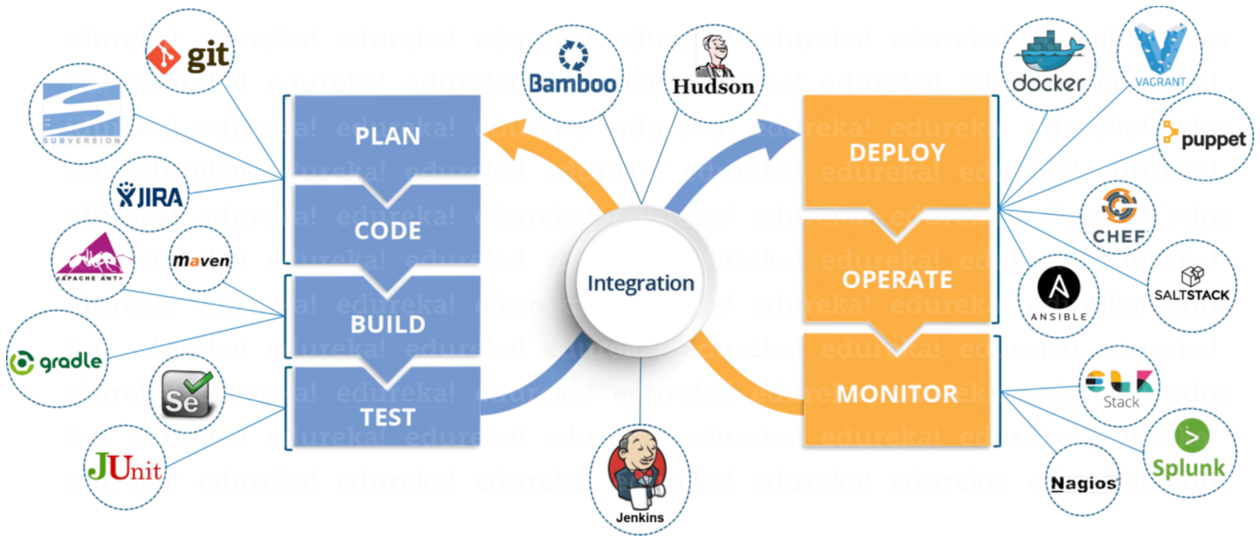
Continuous Deployment:

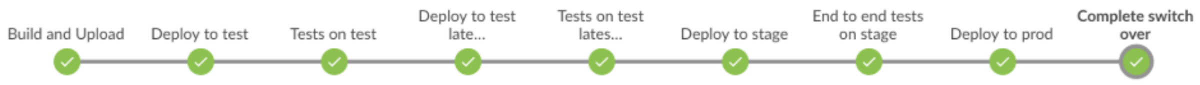
It is the stage where the code is deployed to the production environment. Here we ensure that the code is correctly deployed on all the servers. If there is any addition of functionality or a new feature is introduced, then one should be ready to welcome greater website traffic. So, it is also the responsibility of the Sysadmin to scale up the servers to host more users.
Since the new code is deployed on a continuous basis, configuration management tools play an important role for executing tasks quickly and frequently. Puppet and Ansible are some popular tools that are used in this stage.
Containerization tools also play an important role in the deployment stage. Docker and Vagrant are the popular tools which help produce consistency across Development, Test, Staging and Production environments. Besides this, they also help in scaling-up and scaling-down of instances easily.

Continuous Monitoring:

This is a very crucial stage in the DevOps life cycle which is aimed at improving the quality of the software by monitoring its performance. This practice involves the participation of the Operations team who will monitor the user activity for bugs / any improper behavior of the system. This can also be achieved by making use of dedicated monitoring tools which will continuously monitor the application performance and highlight issues.
Some popular tools used are Splunk, ELK Stack and Nagios. These tools help you monitor the application and the servers closely to check the health of the system proactively. They can also improve productivity and increase the reliability of the systems, reducing IT support costs. Any major issues found could be reported to the development team so that it can be fixed in the continuous development phase.

These DevOps stages are carried out on loop continuously until the desired product quality is achieved. The diagram given below will show you which tools can be used in which stage of the DevOps life cycle.





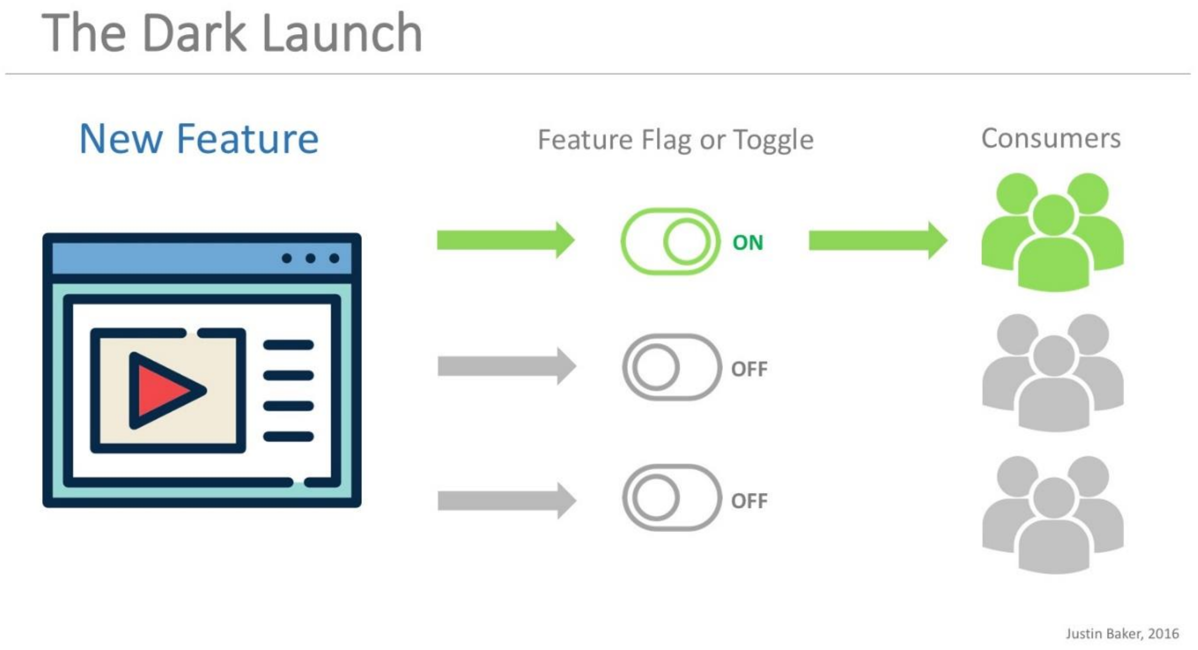
Example Use-Case:

Facebook Dark Launching Technique

In 2011, Facebook rolled out a slew of new features – timeline, ticker and music functionalities – to its 500 million users spread across the globe. The huge traffic that was generated on Facebook following the release led to a server meltdown. The features that were rolled out garnered mixed response from users which led to inconclusive results of the effectiveness of the new features, leaving them with no actionable insights.

Dark launching is the process of gradually rolling out production-ready features to a select set of users before a full release. This allows development teams to get user feedback early on, test bugs, and stress test infrastructure performance. A direct result of continuous delivery, this method of release helps in faster, more iterative releases that ensure that application performance does not get affected and that the release is well received by customers.

In the Dark Launching technique, features are released to a small user base through a dedicated deployment pipeline. In the below given diagram of Facebook Dark Launch, you can see that that only one deployment pipeline is turned on to deploy the new features to a select set of users. The remaining hundreds of pipelines are all turned off at this point. The specific user base on which the features have been deployed are continuously monitored to collect feedback and identify bugs. These bugs and feedback will be incorporated in development, tested and deployed on the same user base until the features becomes stable. Once stability is achieved, the features will be gradually deployed on other user bases by turning on other deployment pipelines.



Tools we use:

- Git
- Jenkins
- Docker
- Puppet
- Ansible
- Selenium
- ELK stack and many more…