
aws
certified
DevOps
Engineer
PROFESSIONAL

❖ DEVOPS:

- Devops is a mashup of two words **'DEVELOPMENT'** and **'OPERATIONS'.**
- DevOps is the combination of cultural philosophies, practices, and tools that increases an organization's ability to deliver applications and services at high velocity: evolving and improving products at a faster pace than organizations using traditional software development and infrastructure management processes.
- This speed enables organizations to better serve their customers and compete more effectively in the market.

(or)

- DevOps combines **Development (Dev)** and **Operations (Ops)** to unite people, process, and technology in application planning, development, delivery, and operations.
- DevOps enables coordination and collaboration between formerly siloed roles like development, IT operations, quality engineering, and security.

➢ CORE DEVOPS PRINCIPLES:

AUTOMATION OF THE SOFTWARE DEVELOPMENT LIFECYCLE:

- This includes automating testing, builds, releases, the provisioning of development environments, and other manual tasks that can slow down or introduce human error into the software delivery process.

COLLABORATION AND COMMUNICATION:

- A good DevOps team has automation, but a great DevOps team also has effective collaboration and communication.

CONTINUOUS IMPROVEMENT AND MINIMIZATION OF WASTE:

- From automating repetitive tasks to watching performance metrics for ways to reduce release times or mean-time-to-recovery, high performing DevOps teams are regularly looking for areas that could be improved.

HYPERFOCUS ON USER NEEDS WITH SHORT FEEDBACK LOOPS:

- Through automation, improved communication and collaboration, and continuous improvement, DevOps teams can take a moment and focus on what real users really want, and how to give it to them.

➢ HOW DEVOPS WORKS:

- Under a DevOps model, development and operations teams are no longer "siloed." Sometimes, these two teams are merged into a single team where the engineers work across the entire application lifecycle, from development and test to deployment to operations, and develop a range of skills not limited to a single function.
- In some DevOps models, quality assurance and security teams may also become more tightly integrated with development and operations and throughout the application lifecycle.
- When security is the focus of everyone on a DevOps team, this is sometimes referred to as **DevSecOps.**

➢ DEVOPS GOALS AND BENEFITS:

ACCELERATE TIME TO MARKET:

- Through increased efficiencies, improved team collaboration, automation tools, and continuous deployment--teams are able to rapidly reduce the time from product inception to market launch.

ADAPT TO THE MARKET AND COMPETITION:

- A DevOps culture demands teams have a customer-first focus. By marrying agility, team collaboration, and focus on the customer experience, teams can continuously deliver value to their customers and increase their competitiveness in the marketplace.

MAINTAIN SYSTEM STABILITY AND RELIABILITY:

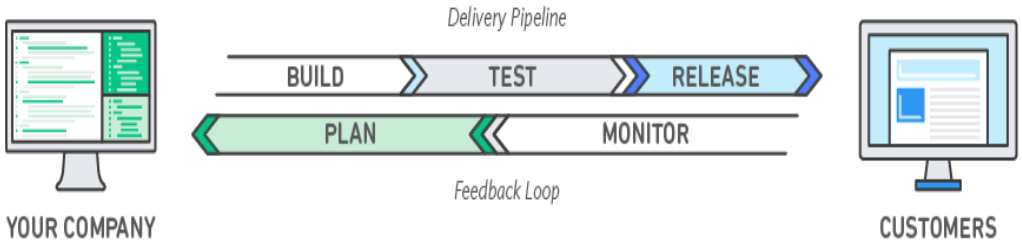
- By adopting continuous improvement practices, teams are able to build in increased stability and reliability of the products and services they deploy. These practices help reduce failures and risk.

IMPROVE THE MEAN TIME TO RECOVERY:

- The mean time to recovery metric indicates how long it takes to recover from a failure or breach. To manage software failures, security breaches, and continuous improvement plans, teams should measure and work to improve this metric.

➢ DEVOPS & THE APPLICATION LIFECYCLE:

- DevOps influences the application lifecycle throughout its planning, development, delivery, and operations phases. Each phase relies on the other phases, and the phases aren't role-specific. A DevOps culture involves all roles in each phase to some extent.



PLAN:

- DevOps teams ideate, define, and describe features and capabilities of the applications and systems they are building.

DEVELOP:

- It includes all aspects of coding writing, testing, reviewing and the integration of code by team members as well as building that code into build artifacts that can be deployed into various environments.

DELIVER:

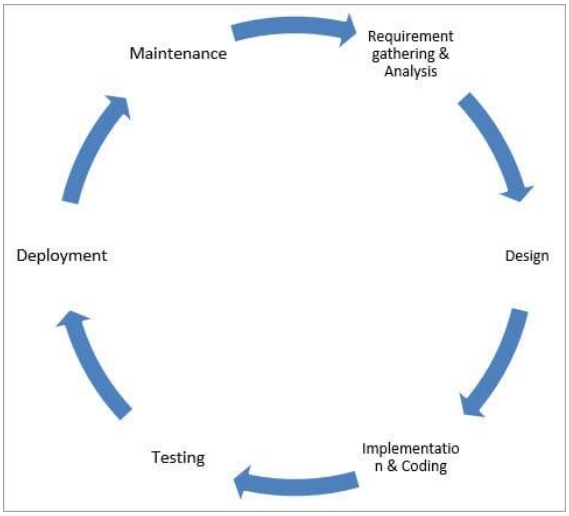
- It is a process of deploying applications into production environments in a consistent and reliable way.
- In this phase, teams define a release management process with clear manual approval stages.

OPERATE / MONITOR:

- The phase involves maintaining, monitoring and troubleshooting applications in production environments.

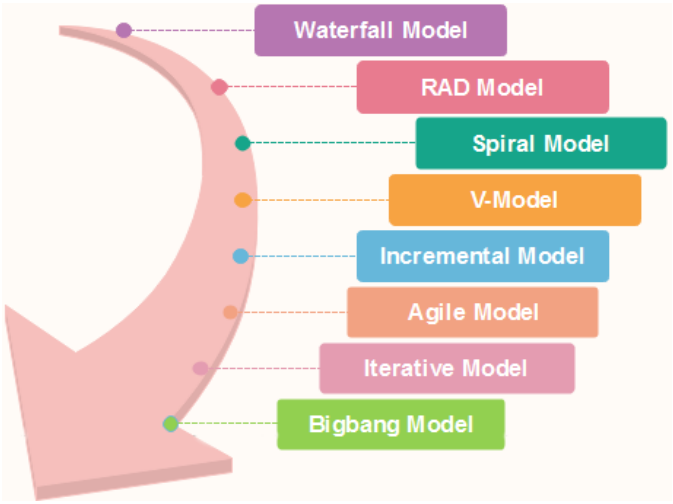
➢ SOFTWARE DEVELOPMENT LIFE CYCLE (SDLC):

- The SDLC provides a framework for managing the software development process, which helps to ensure that all necessary steps are taken and that the final product meets the requirements.
- SDLC Methodologies are processes and practices used by software development teams in order to successfully navigate the SDLC.
- The SDLC typically includes the following phases:



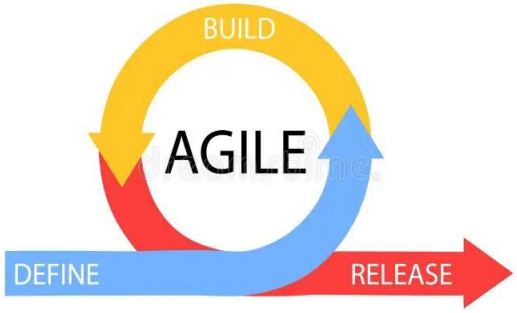
SDLC MODELS:

- There are different software development life cycle models specify and design, which are followed during the software development phase.
- Each process model follows a series of phase unique to its type to ensure success in the step of software development.



AGILE MODEL:

- Agile is a software development approach that emphasizes team collaboration, customer and user feedback, and high adaptability to change through short release cycles.
- Teams that practice Agile provide continual changes and improvements to customers, collect their feedback, then learn and adjust based on customer wants and needs.
- Agile is substantially different from other more traditional frameworks such as waterfall, which includes long release cycles defined by sequential phases. Kanban and Scrum are two popular frameworks associated with Agile.



DEVOPS:

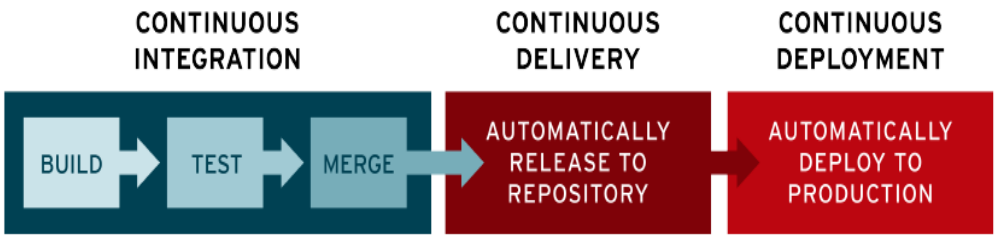
- DevOps is not technically an SDLC methodology but it does share the goal of maximizing software project success and includes Agile-inspired concepts.

ADVANTAGES OF DEVOPS:

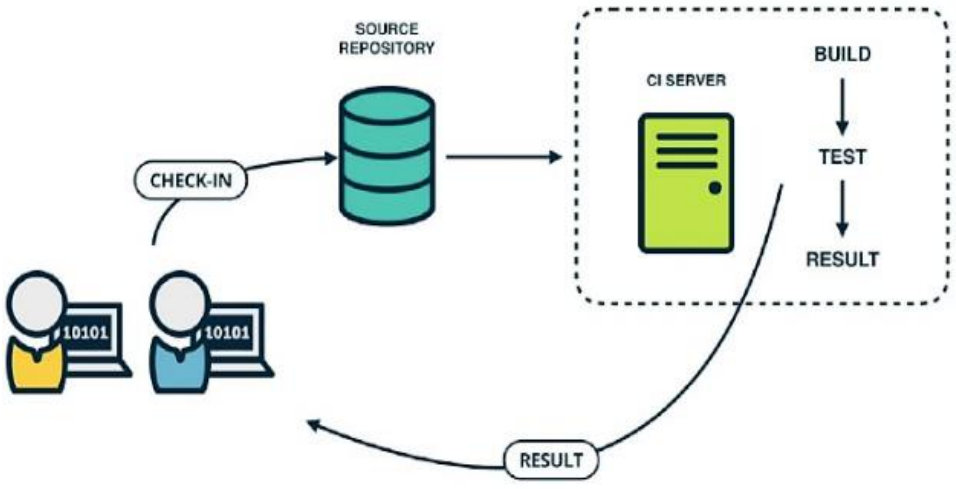
- Software development teams are self-sufficient; shipping and maintaining software without depending on the IT or technical operations teams.
- The deployment process is automated and optimized. A junior developer can learn to safely deploy, with less effort.
- Teams implement **Continuous Integration / Continuous Delivery (CI/CD).**
- Using the right tools, engineers save time on deployment so they can focus on coding.
- Feedback loops integrated throughout the entire process.

➢ CI AND CD (AND OTHER CD) PROCESS:

- CI/CD is a method to frequently deliver apps to customers by automation into the stages of app development.
- Main concepts of CI/CD are continuous integration, continuous delivery, and continuous deployment.
- CI/CD is a solution to the problems integrating new code can cause for development and operations teams.
- Continuous delivery and/or deployment (CD) is a 2 part process that refers to the integration, testing, and delivery of code changes.
- CI/CD helps organizations avoid bugs and code failures while maintaining a continuous cycle of software development and updates.



CONTINUOUS INTEGRATION (CI):



- The "CI" in CI/CD always refers to continuous integration, an automation process for developers that facilitates more frequent merging of code changes back to a shared branch, or “trunk.” As these updates are made, automated testing steps are triggered to ensure the reliability of merged code changes.
- Developers regularly merge their code changes into a central repository, after which automated builds and tests are run.
- CI refers to the build or integration stage of the software release process and entails both an automation component and a cultural component.
- The key goals of CI are to find and address bugs quicker, improve software quality, and reduce the time it takes to validate and release new software updates.

CONTINUOUS DELIVERY (CD):

- CD is the automated delivery of completed code to environments like testing and development.
- CD provides an automated and consistent way for code to be delivered to these environments.
- Continuous delivery usually means a developer’s changes to an application are automatically bug tested and uploaded to a repository (like GitHub or a container registry), where they can then be deployed to a live production environment by the operations team. It’s an answer to the problem of poor visibility and communication between dev and business teams.

CONTINUOUS DEPLOYMENT (CD):

- CD is an extension of continuous delivery, and can refer to automating the release of a developer’s changes from the repository to production, where it is usable by customers.
- In practice, continuous deployment means that a developer’s change to a cloud application could go live within minutes of writing it (assuming it passes automated testing). This makes it much easier to continuously receive and incorporate user feedback.