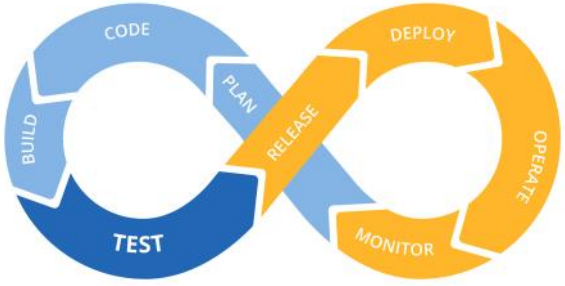
# DevOps - Basics

- **What is DevOps:**
  - A cultural mind-set and practise where development, testers, security & IT Operations teams work as once single entity. (if security is included it is called DevSecOps)
  - DevOps promotes collaboration between Development and Operations team to deploy code to production faster in an automated & repeatable way.
  - DevOps is the union of people, process and technology to continually provide value to customers.
  - **Advantages:**
    - Reduce the disconnection between software developers, quality assurance (QA) engineers, and system administrators.
    - Faster releases / GTM (due to Continuous Integration & Continuous Deployment)
    - Adoption to Market (due to CI/CD)
    - Improved recovery time (due to Continuous Monitoring)
    - Maintaining System Stability & Reliability (thru Automation)
  - **Disadvantages**
    - Working knowledge of DevOps dependency
    - Due to heavy emphasis on automation, tooling knowledge and investment is required.
- **DevOps - Life Cycle & Components/ Phases:**
  - DevOps defines an agile relationship between operations and Development. It is a process that is practiced by the development team and operational engineers together from beginning to the final stage of the product.

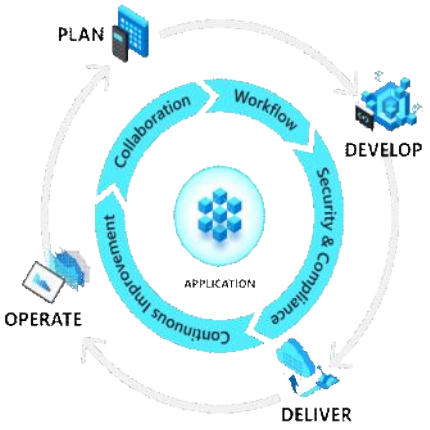



  - Build , code, test & Plan - are associated with Dev Teams
  - Release, Deploy, Operate & Monitor are associated with Ops Teams

| | |
|---|---|
| Plan | : Phase before actual coding. Feedback from previous release & current backlog is taken into consideration to define what needs to go in current release cycle |
| Code | Process of writing the code required for the application<br>Checking it / integrating it into version control system |
| Build | Code integration to the main code , post peer review & approval.<br>Run Integration tests, Unit tests on the merged code. |
| Test | Deploy built code into a staging environment.<br>Tests are conducted<br>Based on Org maturity, this phase could be integrated as Test Driven Deployment or shift left testing. |
| Release | Milestone where decision is made if the build is ready to be pushed into production<br>Usually this could also mean pushing production into a pre-prod environment. |
| Deploy | • Releasing the code into production.<br>Cut-over scenarios vary based on organization maturity. There could be a down time based deployment or live site deployment (blue-green deployment / Canary deployment).<br>• Further read - [Comparing Blue/Green deployment and Canary deployment \| by Brandon Kang \| Medium] |
| Operate | The application is live. Based on configuration the app either scales up/down etc. |

| | |
|---|---|
| | Some means of collecting feedback / issues from customers is in place (like a ticketing system) |
| Monitor | Monitoring & Telemetry is in place to check the health of the application, usage behaviour, potential bottlenecks etc.<br>This is intended to go as an input to the Product Manager (or PO) for the next release cycle. |

- **DevOps Culture:**
  - Collaborate visibility & alignment
  - Shifting and expansion of scope & responsibility .
  - Be agile by releasing software in shorter cycles
  - Continuous learning. Fail fast and adopt learnings.
- **DevOps Principles:**
  - Automation (Of components in DevOps)
  - Collaboration (between Dev & Ops)
  - Integration (of application with other environment in production)
  - Configuration Management
  - Continuous Improvement
- **DevOps Methodology:**

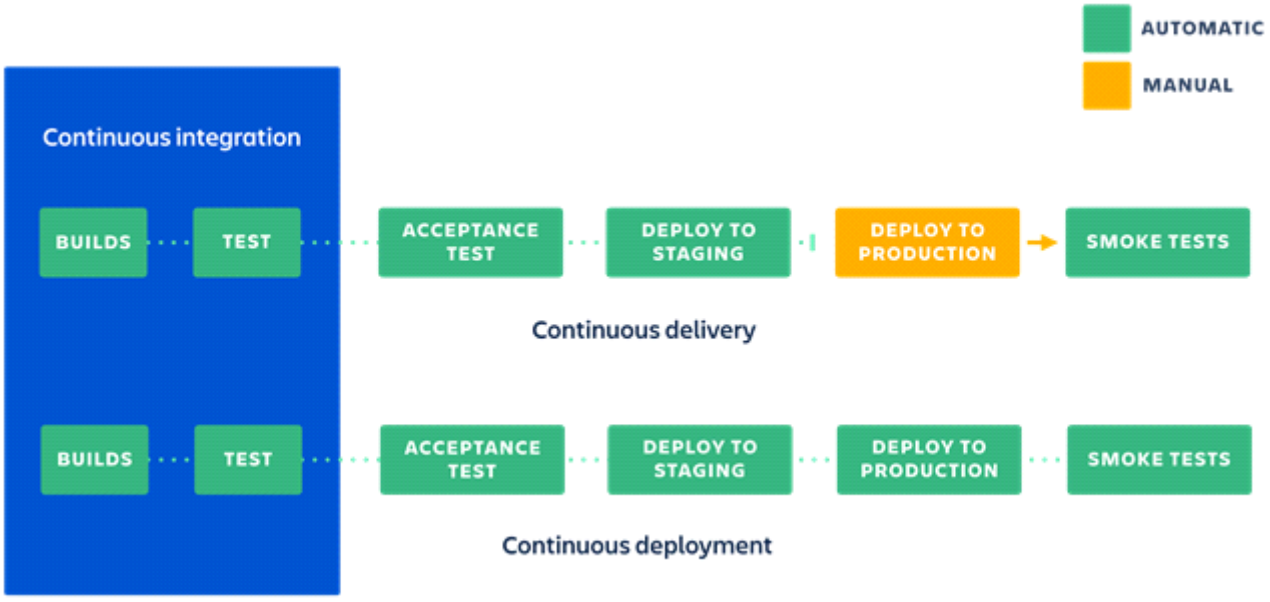



  - Each phase relies on the others and the phases are not role-specific. In a true DevOps culture, each role is involved in each phase to some extent.
  - Comprises of
    - Development
    - Integration
    - Testing
    - Monitoring
    - Feedback
    - Deployment
    - Operations
  - Automation of each of these phases in life cycle is considered as Continuous Everything.
    - Continuous Development
    - **Continuous Integration**
    - Continuous Testing
    - **Continuous Monitoring**
    - **Continuous Feedback**
    - **Continuous Deployment**
    - Continuous Operations
  - Dev more responsible for code in prod.
  - Ops more responsible / providing view points during plan & deploy (feedback provided to dev on security, governance & quality)
- **DevOps Implementation Practises:**
  - **CI/ CD: Continuous Integration & Continuous Delivery:**
    **Continuous integration** is a software development practice in which developers merge code

changes frequently into the main code branch. Continuous integration employs automated testing, which runs every time new code is committed so the code in the main branch is always stable.

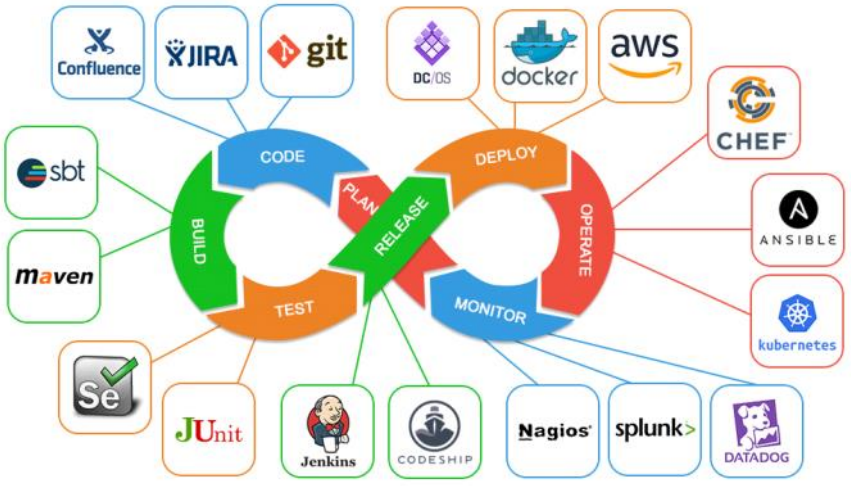
**Continuous delivery** is the frequent, automated deployment of new application versions into a production environment. By automating the steps required for deployment, teams reduce issues that may occur upon deployment and enable more frequent updates.

**Continuous deployment** takes continuous delivery further and it the process of making the build available to customers in an automated process.




- **Infrastructure as a Code:**
  - The practise of defining Systems and their topology in a version control system and treating it as code.
  - Helps teams deploy system resources in a reliable, repeatable and controlled way.
  - Helps automate deployment and reduces the risk of human error, especially for complex large environments.
- The above two practises depend on the below:
  - Version Control Systems
  - Configuration Management
  - Continuous Monitoring
  - Agile development approach

- **DevOps Tool Chain:**
  - There are many tools now available in market that support DevOps Tools.
  - Each tool focusing on a particular stage(s).
  - Below is one such example :




- **Other terms**:

| | |
|---|---|
| Mutable | - Something that keeps changing or fluctuating. |

| | |
|---|---|
| Immutable | - Something that doesn't change (over time or steps) |
| Ephemeral | - Short Lived |
| Idempotent | denoting an element of a set which is unchanged in value when multiplied or otherwise operated on by itself. |

- **DevOps Extensions** (while there are many , the below are popular):

| | | |
|---|---|---|
| GitOps | DevOps practises for Infrastructure | What is GitOps? \| GitLab |
| AIOps | IT Operations management using Artificial Intelligence | What is AIOps? \| IBM |
| BizDevOps | Business inputs in the DevOps cycle. Decisions in the DevOps cycle are based on inputs from Biz reps. | https://www.devopsinstitute.com/what-is-bizdevops/ |
| DevSecOps | Security throughout the DevOps Cycle, instead of a phase in the application life cycle | https://www.ibm.com/cloud/learn/devsecops |
| NoOps | NoOps is a fully automated approach that eliminates the operations team, aside from an integrated team focused on life cycle. The self-managed system requires resiliency and a high level of maturity with fail points that have already been addressed. | https://www.bmc.com/blogs/itops-devops-and-noops-oh-my/ |