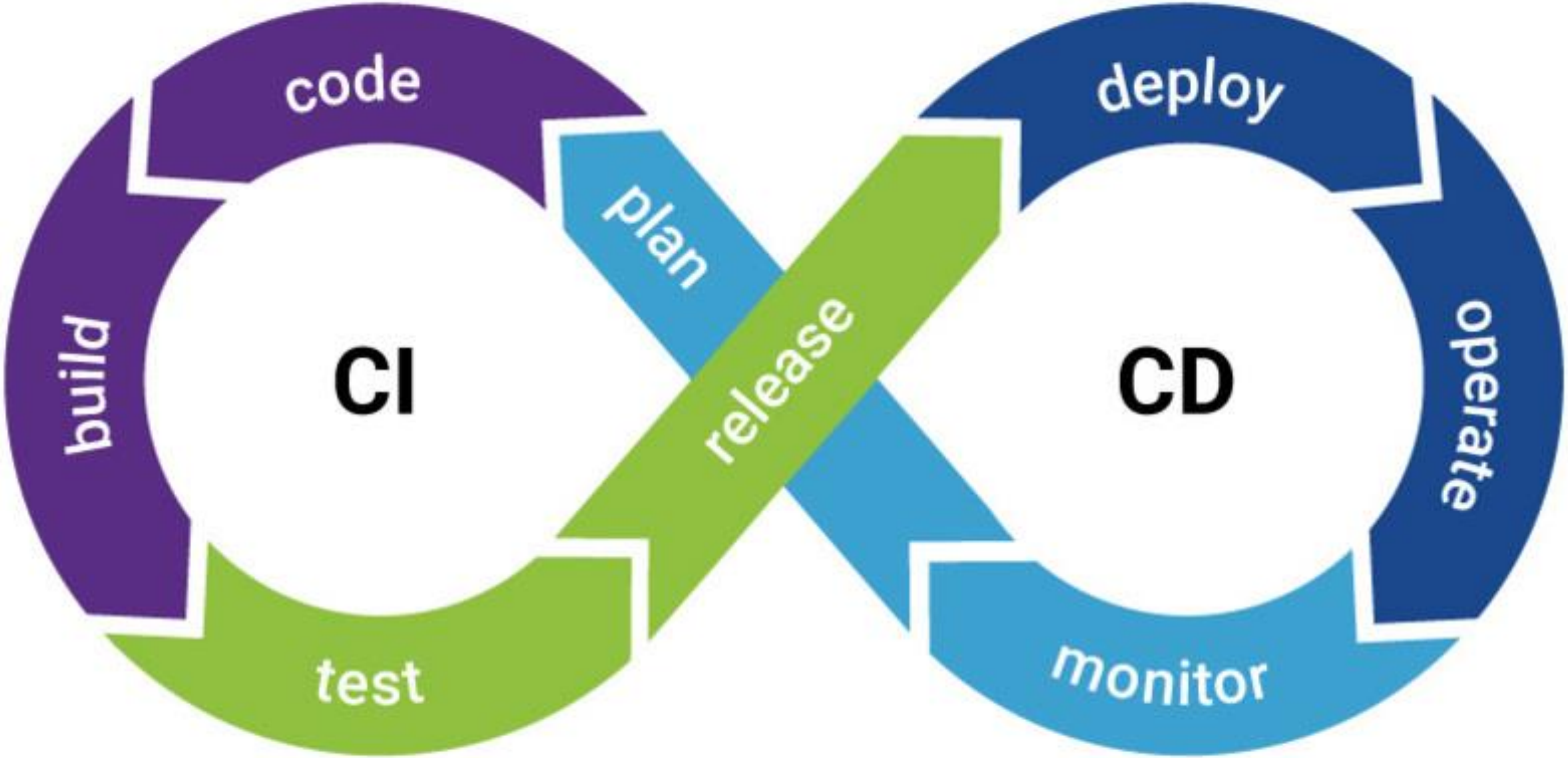

# CI/CD - A Revolutionary Tool for Delivery
By Gaurav Singh Verma

# Current Challenges of Deployment Life Cycle

| | | | |
|---|---|---|---|
| | Merging/ compiling and identifying errors are time consuming task, which takes developer time | | Manually creation of infrastructure leads to human errors and configuration failures |
| | Manual unit testing is sometimes time consuming and prone to vulnerability | | Manual Post implementation checks are not affective as the down time is increased due to that |
| | Manual Deployment is a time consuming task which means more time to market new features | | Response time is low while doing rollback if any bug is found in post implementation checks |

# Effects of the Current Challenges

- Delays in development life cycle.
- Unit testing are sometimes ignored due to development delays.
- Deployment challenges, as the operations implementation team might not have thorough knowledge of the application.
- Infrastructure changes and configuration issues between the non-production and production environment.
- Manual post implementation checks/ status takes time for a huge application.
- Errors/ delays in rollback if the implementation team faces issues.

# Continuous Integration/Continuous Deployment approach

**Continuous Integration (CI): is a process of automatic merging and compiling of code whenever a code is checked-out by multiple developers. We can also automate and configure unit testing**

**Continuous Deployment (CD): is an automated process of deploying the code to production with some automated steps for configurations and deployment**

**CI/ CD approach: focuses on software-development and delivery life cycles underlining tools that emphasize automation**

# CI/CD Benefits

- Automated integration means less time to develop and no issues from developer codes
- Bugs/ issues can be avoid in production as we use automated unit testing
- Less time to market, as the deployment time gets reduced due to automated steps for deployment.
- Infrastructure creation/ configuration automation helps in Zero issues and no configuration drift.
- Down time gets reduced and early detection of issues while automated smoke testing is in place
- This approach can quickly redo all the changes and the infrastructure is back to its original state without affecting the users