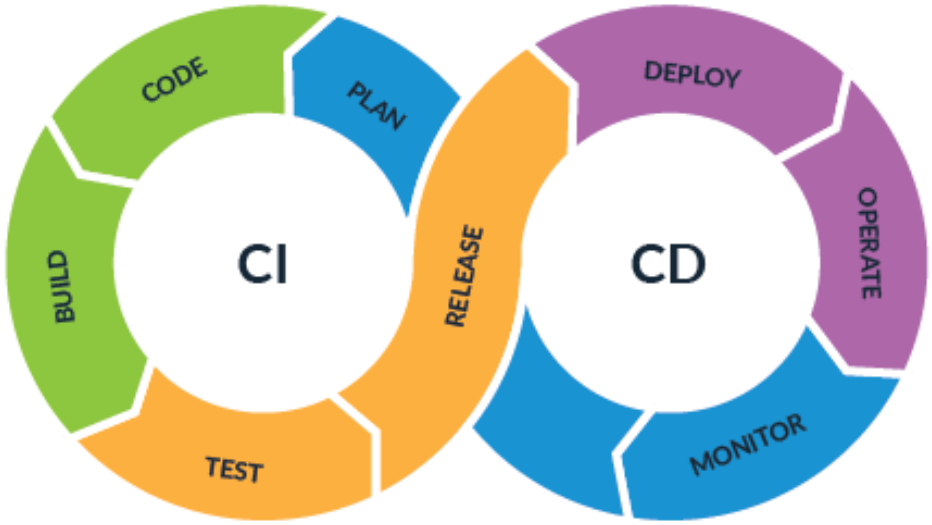
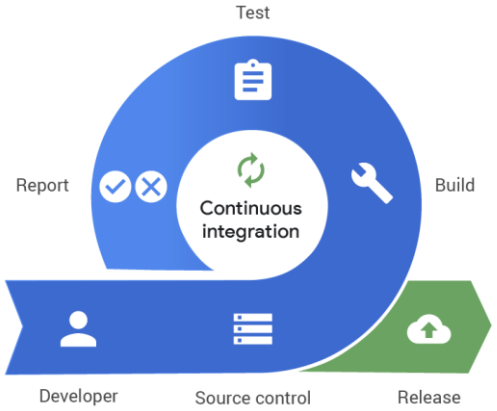
CI/CD
THE BACKBONE OF THE MODERN DEVOPS ENVIRONMENT

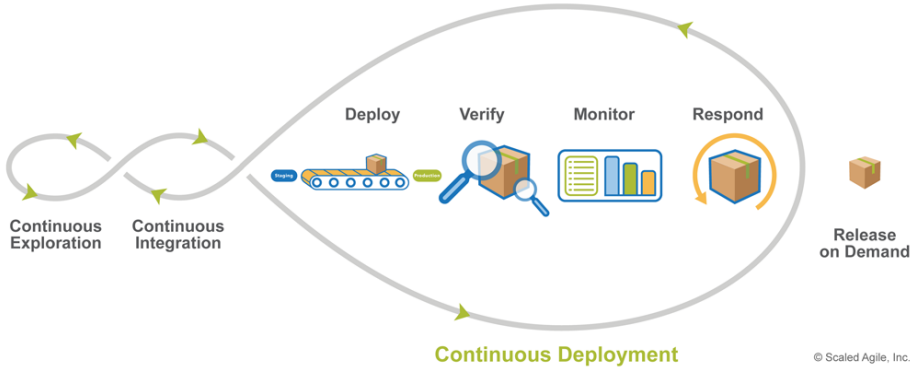


CONTINUES INTEGRATION



- Continuous integration (CI) is a fundamental part of any DevOps team's toolchain. While often mentioned in the context of the modern DevOps movement. The basic idea is that code is integrated continuously, throughout a software project's development, rather than at the end of the development cycle.
- Prior to the adoption of continuous integration, it was not uncommon for software engineers to work in relative isolation, developing code independently.
- At the end of a project would then come a tedious, painful, and occasionally disastrous integration phase in which the developers would work to combine, or integrate, their independent code into a single functioning program. This type of workflow has an obvious drawback–miscommunications, errors, or incompatibilities are worked on in an extremely costly manner after coding has been completed.
- So In software engineering, continuous integration (CI) is the practice of merging all developers' working copies to shared mainline several times a day.
- Main aim is to produce a high-quality deployable artifact.

CONTINUES DEPLOYMENT



- CD is an enabler for the business as a whole to be faster, more responsive, and more accurate in generating customer digital experiences. It is not a new set of tools for the techies – rather it is a new way of working across a business.
- CD has the ability to unleash a business to achieve success at an unprecedented scale and speed, but like any new venture, the benefits and outcomes should be measured against the core mission and objectives of an organization.
- Continuous Delivery (CD) aims to reduce the lead time for reliably releasing software at any time. The key enablers are:
- 1- Decomposing work into very small deliverables
- 2- Automation of builds, deploys, and testing
- 3- Monitoring everything
- CD is all about Deploying the artifact automatically

CI/CD BENEFITS

- It is much easier for more than one team member to work on the same feature.
- The cost of merging small changes is much lower than large changes.
- Team members can evaluate their peer's changes in a system that has passed automated testing rather than the moving target of a feature branch.
- It is much easier for QA engineers to validate small changes iteratively in a consistent system, so detecting the source of errors becomes much simpler.
- Features can get into the hands of customers in an earlier state. This means that if we're on the wrong track, we can change direction after only a small investment.

CI/CD BENEFITS

Technical Language	Value	Translation
Catch Compile Errors After Merge	Reduce Cost	Less developer time on issues from new developer code
Catch Unit Test Failures	Avoid Cost	Less bugs in production and less time in testing
Detect Security Vulnerabilities	Avoid Cost	Prevent embarrassing or costly security holes
Automate Infrastructure Creation	Avoid Cost	Less human error, Faster deployments
Automate Infrastructure Cleanup	Reduce Cost	Less infrastructure costs from unused resources
Faster and More Frequent Production Deployments	Increase Revenue	New value-generating features released more quickly
Deploy to Production Without Manual Checks	Increase Revenue	Less time to market
Automated Smoke Tests	Protect Revenue	Reduced downtime from a deploy-related crash or major bug
Automated Rollback Triggered by Job Failure	Protect Revenue	Quick undo to return production to working state