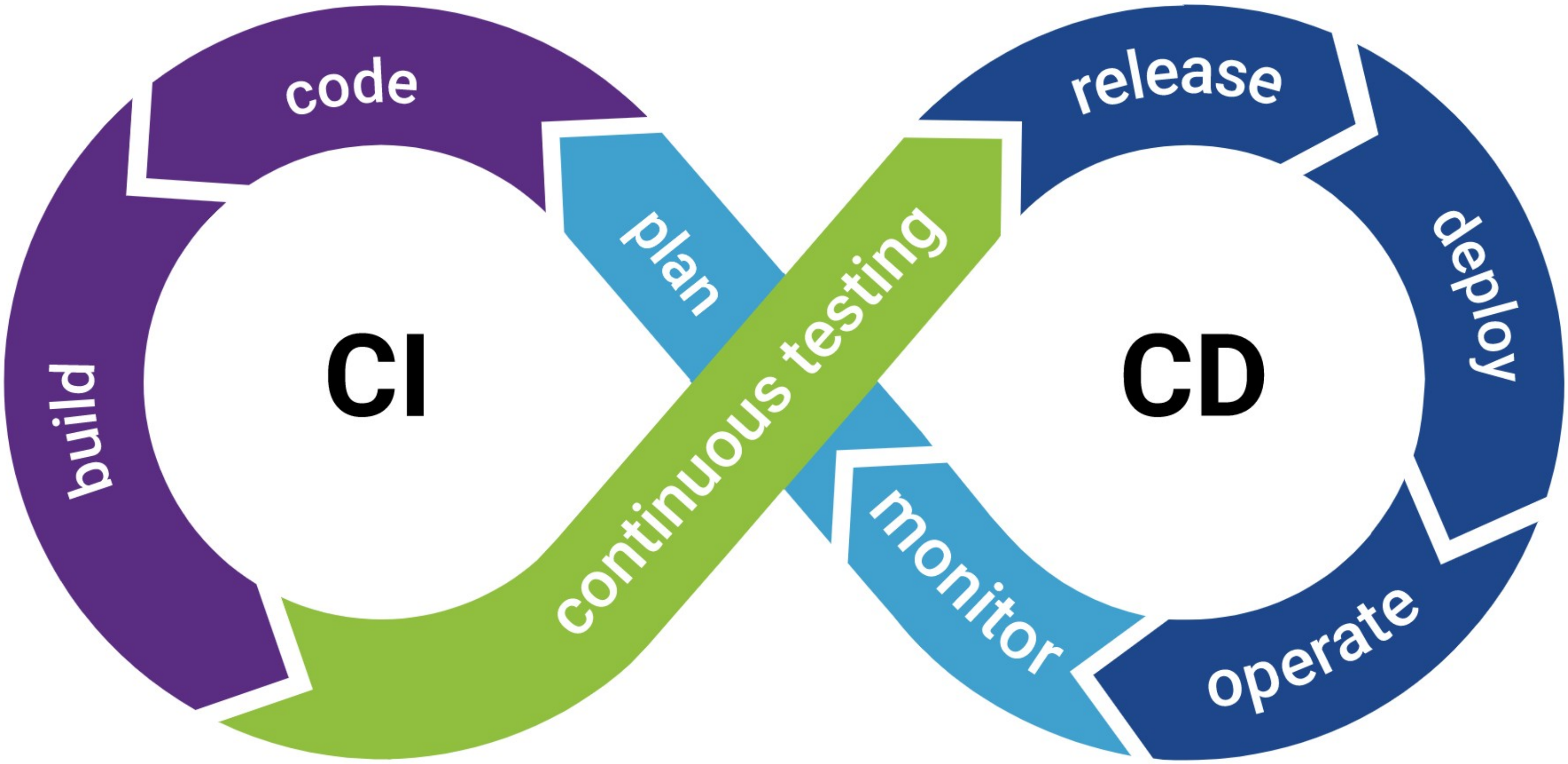
CI/CD

What is CI/CD?


code
build
plan
CI
continuous testing
release
deploy
CD
monitor
operate

What is CI/CD?

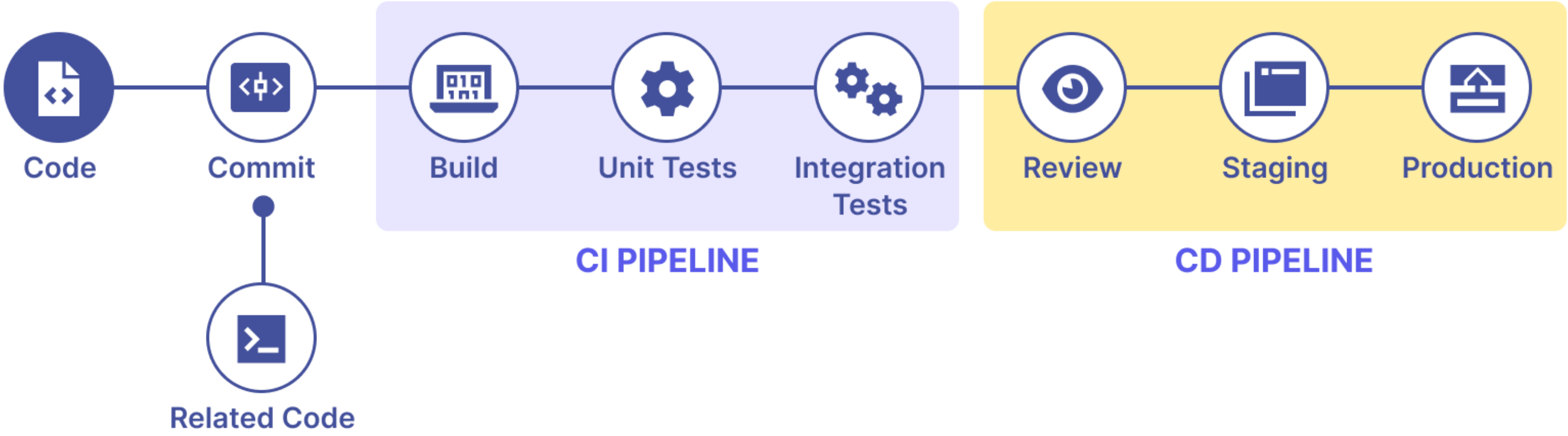
- CI/CD stands for Continuous Integration/Continuous Deployment (or Continuous Delivery).
  - It is a set of practices and tools designed to automate the process of software development, from code changes to production deployment.

- Continuous Integration (CI) involves automating the process of integrating code changes into a shared repository multiple times a day.
  - Each integration is verified by automated tests, allowing teams to detect and fix problems early.

- Continuous Deployment (CD) goes one step further by automating the deployment of code changes to production environments after successful integration and testing.
  - It ensures that software is always deployable, with changes made available to users as soon as they're ready.

- In summary, CI/CD enables teams to deliver software faster, more reliably, and with higher quality, by automating key aspects of the development and deployment process.

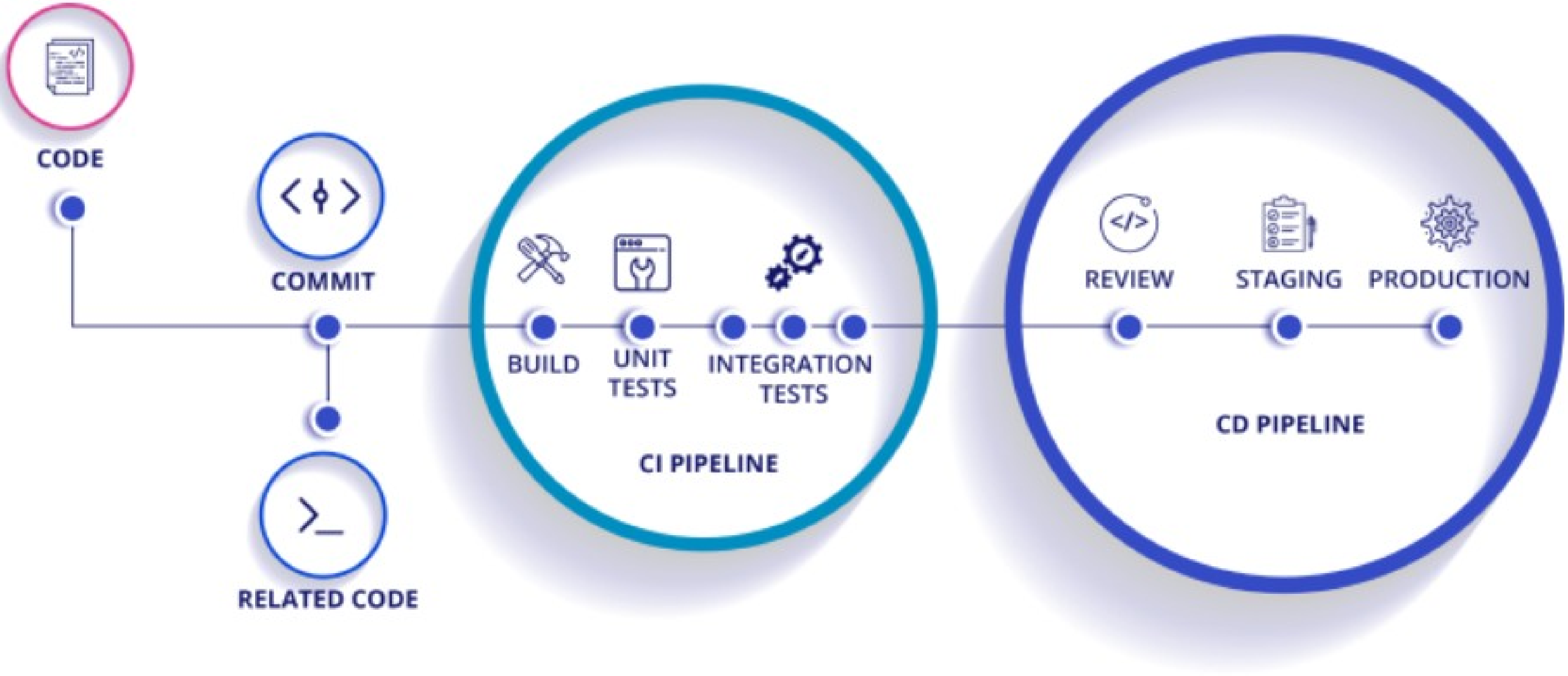
What is CI/CD?


Code
Commit
Related Code
Build
Unit Tests
Integration Tests
CI PIPELINE
Review
Staging
Production
CD PIPELINE

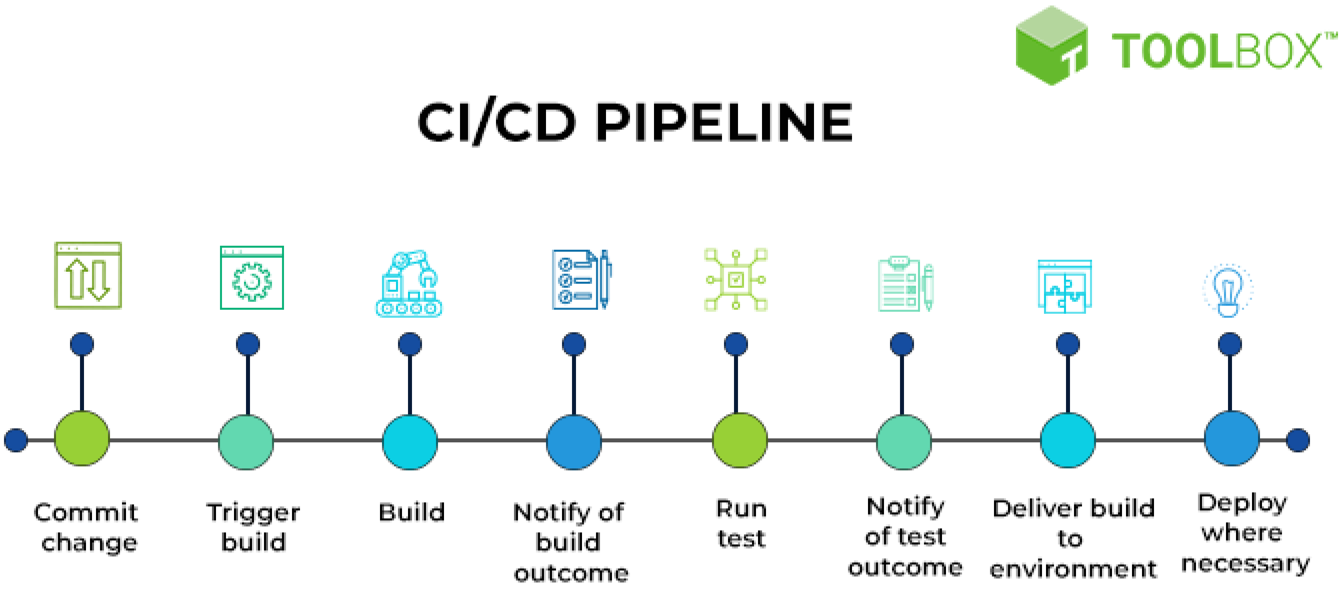
Why CI/CD?

- **Faster Time to Market:** CI/CD automates the software delivery process, allowing teams to release updates more frequently and respond quickly to market demands.
- **Increased Reliability:** Automated testing and deployment reduce the risk of errors and ensure that code changes are thoroughly validated before reaching production environments.
- **Better Collaboration:** CI/CD encourages collaboration between developers, testers, and operations teams, leading to improved communication and faster feedback loops.
- **Continuous Improvement:** By continuously integrating and deploying code changes, teams can identify and address issues earlier in the development cycle, leading to faster iterations and improved software quality.
- **Cost Savings:** Streamlining the development and deployment process with CI/CD reduces manual effort, minimizes downtime, and lowers the risk of costly errors, ultimately saving time and resources.

Why CI/CD?


CODE
COMMIT
RELATED CODE
BUILD
UNIT TESTS
INTEGRATION TESTS
CI PIPELINE
REVIEW
STAGING
PRODUCTION
CD PIPELINE

CI/CD Lifecycle

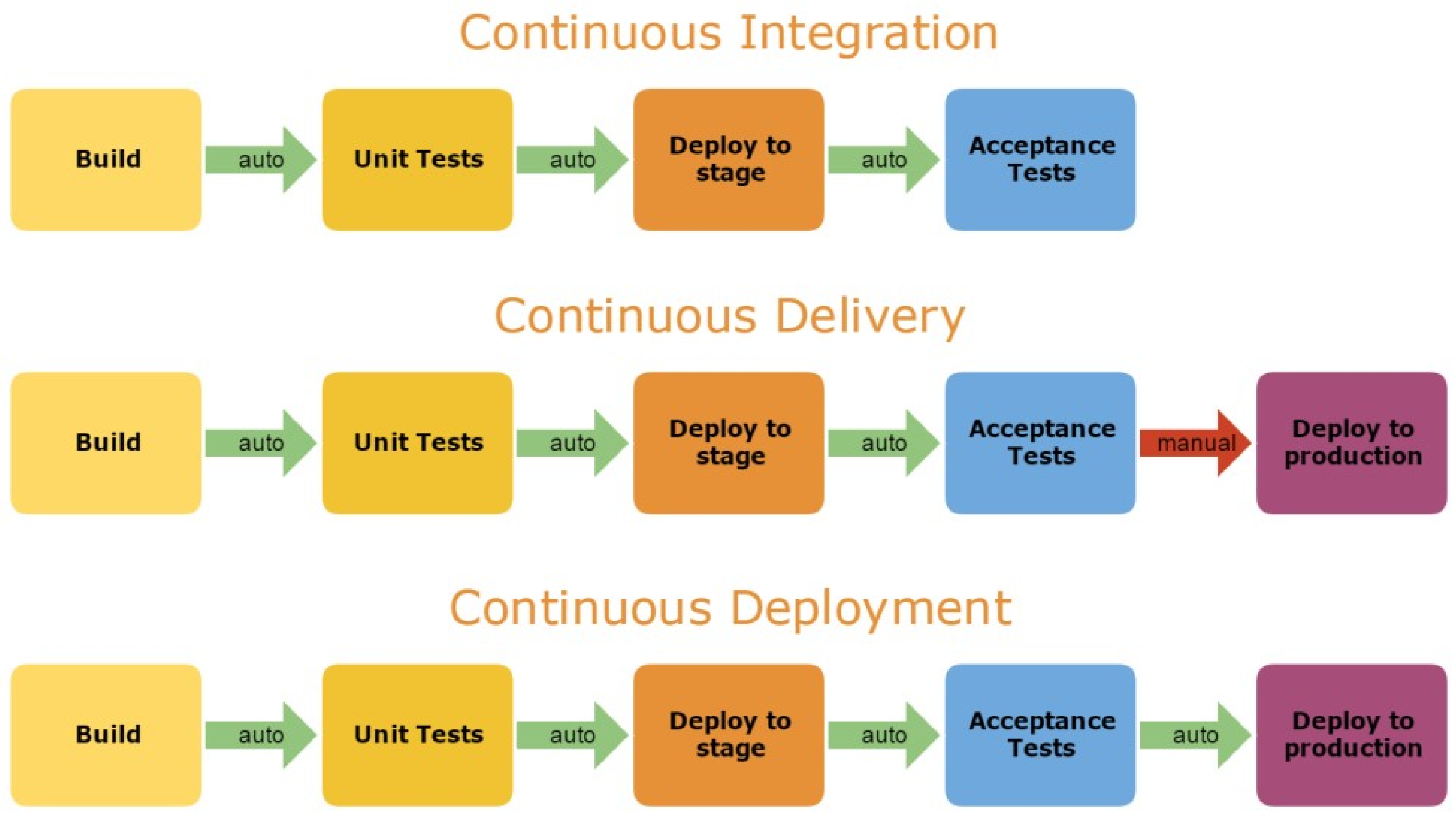


Continuous Delivery vs Continuous Deployment

- **Continuous Delivery:**
  - Involves automatically building, testing, and deploying code changes to a staging environment.
  - Allows for manual approval before deploying changes to production.
  - Provides a consistent and reliable process for releasing software updates.

- **Continuous Deployment:**
  - Goes a step further by automatically deploying code changes to production after passing automated tests.
  - Eliminates the need for manual approval, resulting in faster delivery of features and bug fixes.
  - Requires a high level of confidence in automated testing and deployment processes to ensure production stability.

Continuous Delivery vs Continuous Deployment



Continuous Integration

Build

auto

Unit Tests

auto

Deploy to stage

auto

Acceptance Tests

Continuous Delivery

Build

auto

Unit Tests

auto

Deploy to stage

auto

Acceptance Tests

manual

Deploy to production

Continuous Deployment

Build

auto

Unit Tests

auto

Deploy to stage

auto

Acceptance Tests

auto

Deploy to production

CI/CD vs DevOps

- **CI/CD (Continuous Integration/Continuous Delivery):**
  - Focuses on automating the software delivery process, from code integration to deployment.
  - Enhances collaboration among development, testing, and operations teams.
  - Improves software quality, speed, and reliability through automation of testing and deployment.

- **DevOps:**
  - A culture and set of practices that emphasizes collaboration and communication between development and operations teams.
  - Encompasses CI/CD practices but also extends to cultural aspects, such as shared responsibilities, transparency, and feedback loops.
  - Aims to break down organizational silos and streamline the entire software delivery lifecycle for faster and more reliable releases.