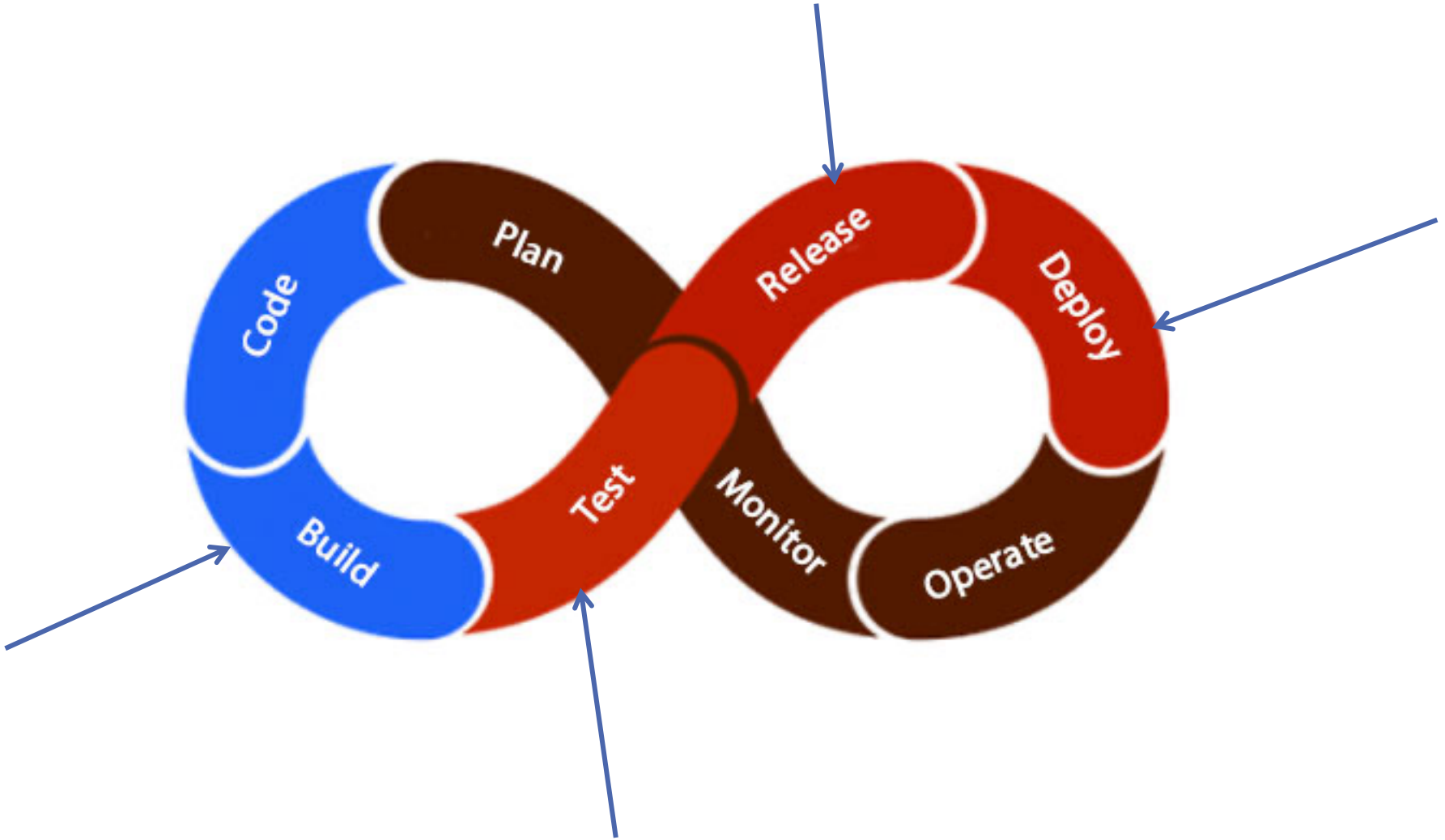
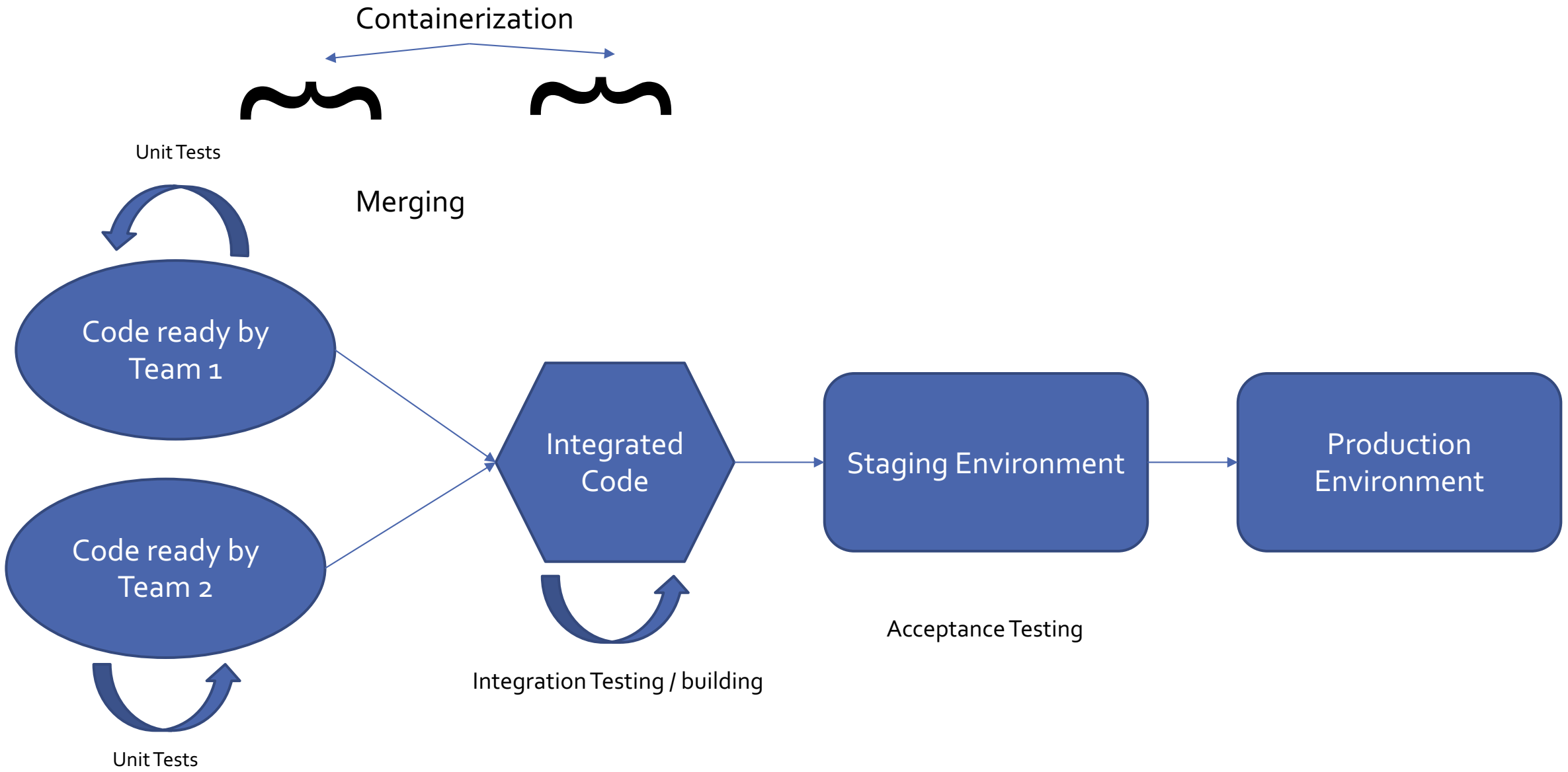
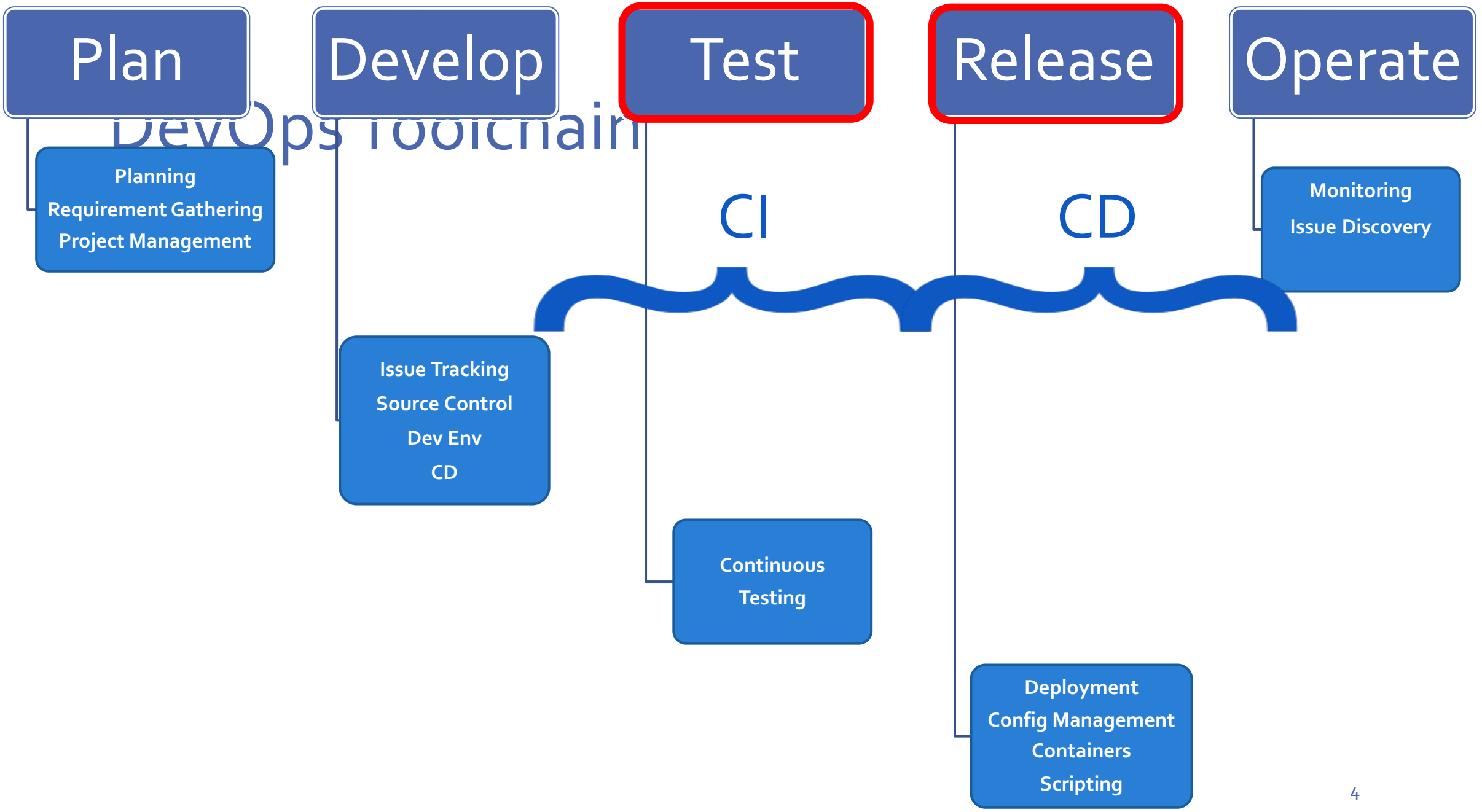
CONTINUOUS INTEGRATION & CONTINUOUS DELIVERY


Code
Plan
Release
Deploy
Build
Test
Monitor
Operate


Containerization
Unit Tests
Merging
Code ready by Team 1
Code ready by Team 2
Integrated Code
Staging Environment
Production Environment
Acceptance Testing
Integration Testing / building
Unit Tests

DevOps Toolchain



Continuous Integration

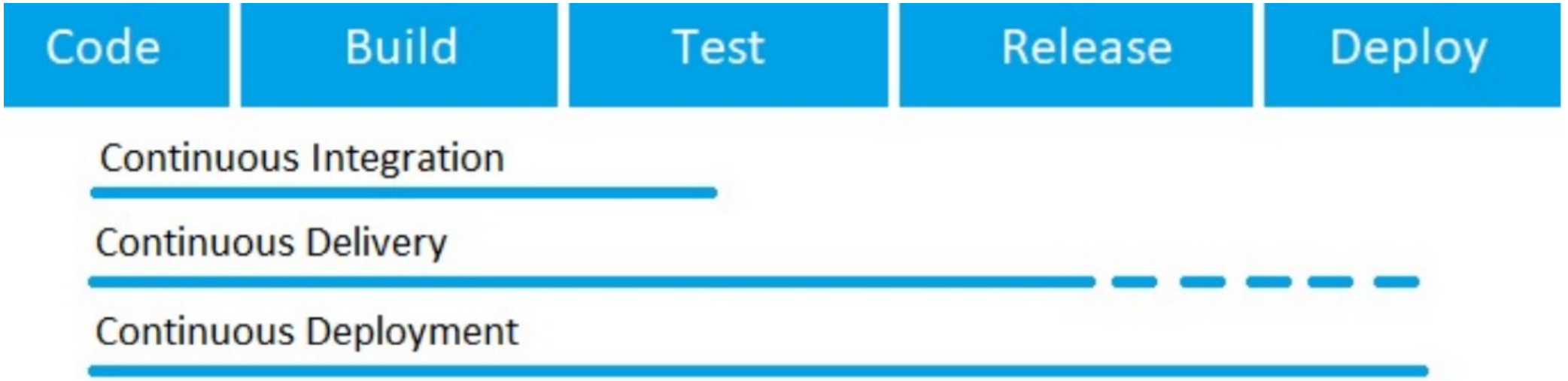
- It is about automating build and test processes to make sure the resulting software is in a good state, ideally every time a developer changes code.
- It is critical in multiple person dev teams on a project.
- It is merging all code from all developers to one central branch of the repo many times a day trying to avoid conflicts in the code in the future.
- It helps development teams avoid “integration hell” where the software works on individual developers’ machines, but it fails when all developers combine their code.

Continuous Delivery

- Continuous Delivery represent the end-to-end chain of regularly delivering code to production.
- It is a small build cycle with short sprints.
- The code is always ready to be released but isn't pushed to production unless you make the decision to do so.
- It requires building, testing, and releasing faster and more frequently.
- It ensures that each change is releasable, with complete automation of the release process.

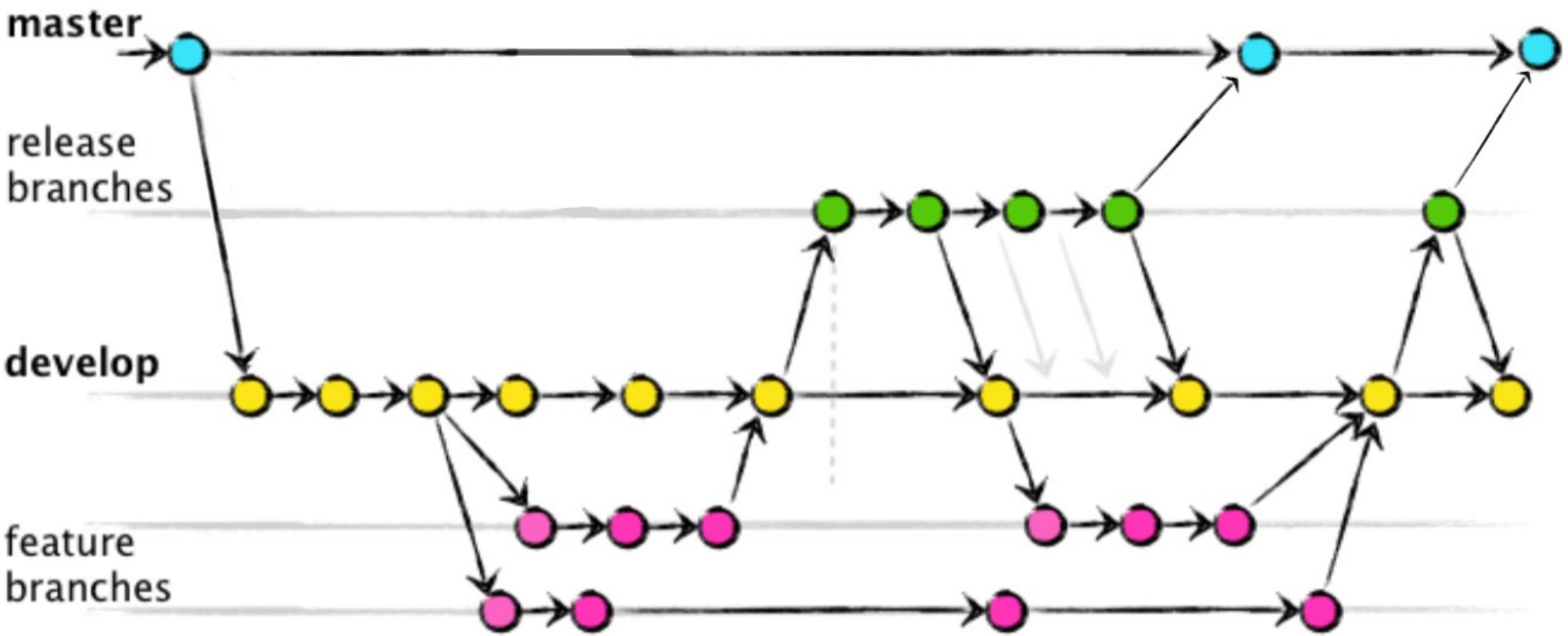
Continuous Delivery

- There are basically three types of deployment:
  1. Manual
  2. Automated with human interaction:
  3. Full automated:
- The automated deployment that still needs some human interaction is what we call Continuous Delivery.
- Basically, when you have your release files ready to be deployed at any time, you are practicing Continuous Delivery.



- With continuous delivery we should always have a release package that is ready to be deployed at any time, but the deployment step itself is still manual or manually triggered.
- Even if it is as easy as clicking a button.

Version Control Branching for CD



Continuous Deployment

- With Continuous Deployment, every change that is made is automatically deployed to production.
- It is the process by which qualified changes in software code or architecture are deployed to production as soon as they are ready and without human intervention.

Continuous Deployment

- Continuous Deployment differs from continuous delivery with the step of immediate autonomous deployment to a production environment.
- The automated deployment doesn’t need human interaction.
- When the checks pass the package is automatically deployed to production. Continuous deployment is the full end to end, automated software deployment pipeline.

Continuous Deployment

- Advantages
  - High productivity benefits for modern software businesses by allowing quick response to changing market demands
  - Rapidly deploy and validate new ideas and features.
  - React to customer feedback in real time
- Disadvantages
  - Expensive initial engineering cost.
  - High accuracy testing is required