

## Taylorism

- Adoption of command and control management.
- Organizations divided into functional silos
- Decision-making is separated from work.
- ★ Working devops means pushing small releases faster in order to get feedback, minimize risk, & maximise learning
- ★ Taylorism was designed for factory work while software dev is like craft work.

## Operations view of dev

- Dev teams throw code over the wall
- Manually implemented changes
- Lack of back-out plans
- Lack of testing
- Env that don't look like production.

## Dev view of ops:-

- All or nothing changes
- Change windows in the dead of night
- Implemented by people furthest away from the application.
- ops just cuts & pastes from "runbooks".
- ★ → Infrastructure as code describe infrastructure in an executable textual format.
- Ephemeral infrastructure can be used & then discarded. Servers are built on demand, via automation.

Rather than patching a running container, immutable delivery is making changes to the container image, then redeploying a new container.

## Continuous Integration vs Continuous Delivery

- CI/CD is not one thing.
- Continuous Integration (CI)
  - continuously building, testing & merging to master.
- Continuous Delivery (CD):
  - continuously deploying to a production like environment.

### CI

- Devs integrate code often
- Devs work in short lived feature branches
- Each check-in is verified by an automated build.

### Changes are kept small:

- working in small batches
- committing regularly.
- using pull requests
- committing all changes daily.

### CI automation:

- Builds & Tests every pull request.
- Use CI tools that monitor version control
- Tests should run after each build

## Benefits of CI:

- Faster reaction times to changes
- Reduced code integration risk
- Higher code quality
- The code in version control works.

## Continuous Delivery:

- A software discipline where you can build software in such a way that the software can be released to production at any time.

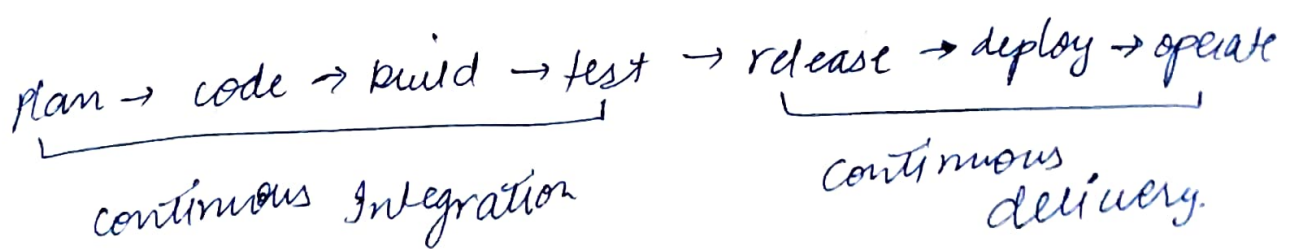
## CI/CD Pipeline

Automated gates that create a pipeline of checks:

- unit testing
- code quality checks
- Vulnerability Scanning
- Integration testing
- Security testing
- Package signing.

## CI/CD pipeline needs:

- A code repository.
- A build server.
- An integration server.
- An artifact repository.
- Automatic configuration & deployment.



## Five Key principles

- build quality in
- work in small batches
- computers perform repetitive tasks, people solve problems
- Relentlessly pursue continuous improvement.
- Everyone is responsible.

## How DevOps manages risk

- Deployment is king.
- Deployment is decoupled from activation
- Deployment is not one size fits all.