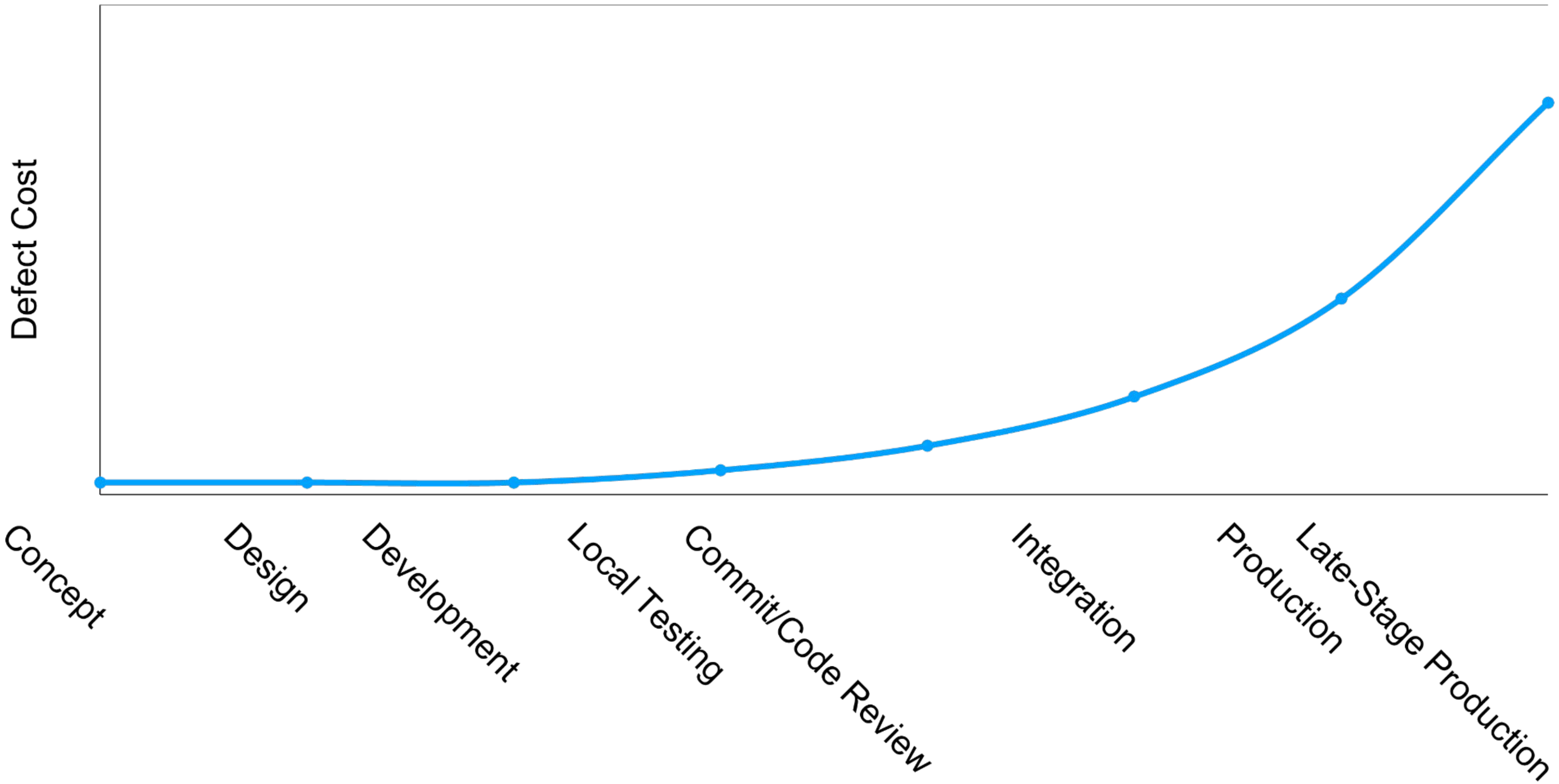
Software Engineering

Part (XII)- Continuous Integration & Continuous Delivery (CI/CD)

By: Mehran Alidoost Nia

Shahid Beheshti University, Fall 2023

Cost to Fix a Defect



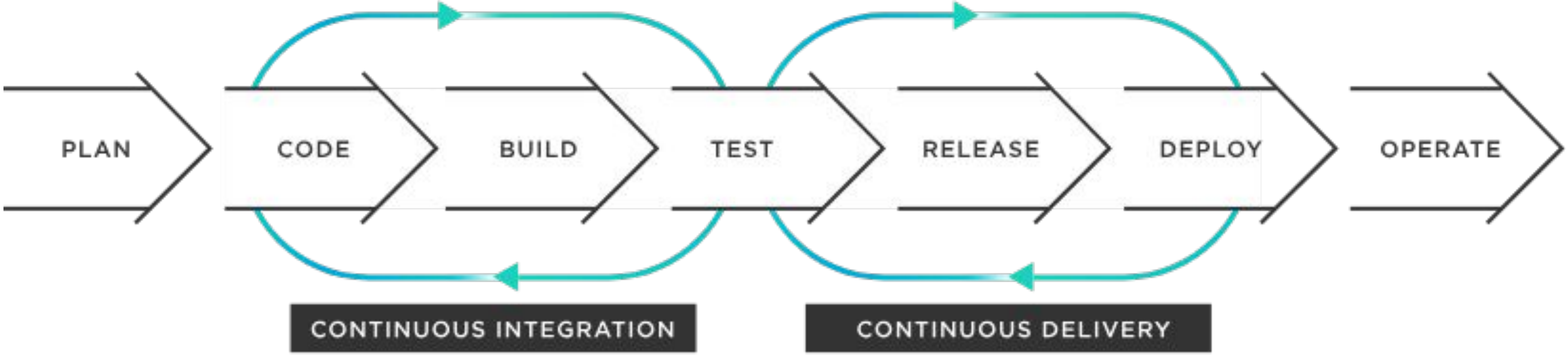
Continuous Integration (CI)

- Continuous integration is the practice of integrating all your code changes into the main branch of a shared source code repository early and often, automatically testing each change when you commit or merge them, and automatically kicking off a build.
- With continuous integration, errors and security issues can be identified and fixed more easily, and much earlier in the development process.

Continuous Delivery (CD)

- Continuous delivery is a software development practice that works in conjunction with CI to automate the infrastructure provisioning and application release process.
- With CD, the software is built so that it can be deployed to production at any time.

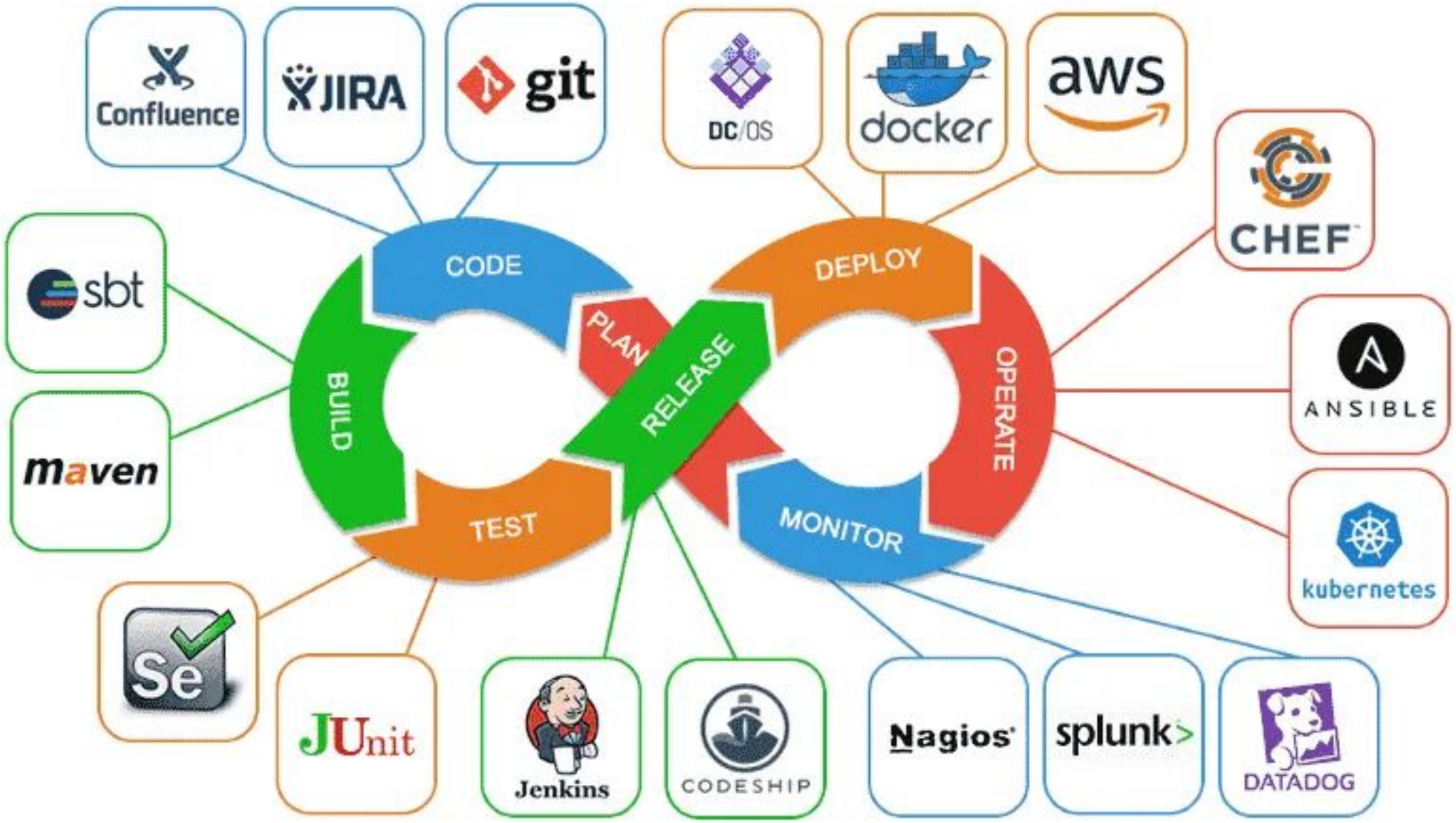
CI/CD Overview



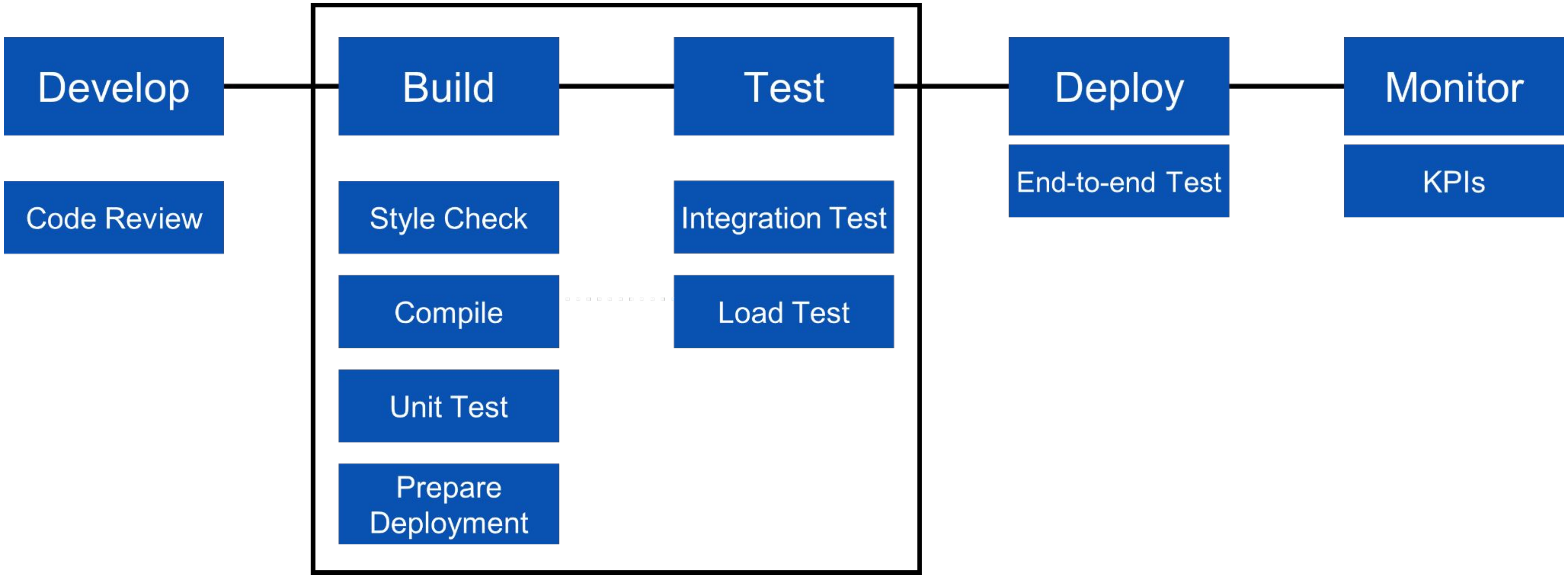
CI/CD Fundamentals

- A single source repository
- Frequent check-ins to main branch
- Automated builds
- Self-testing builds
- Frequent iterations
- Stable testing environments
- Maximum visibility
- Predictable deployments anytime

Some CI/CD Tools



CI/CD Steps



Develop
Code Review
Build
Style Check
Compile
Unit Test
Prepare Deployment
Test
Integration Test
Load Test
Deploy
End-to-end Test
Monitor
KPIs

CI/CD Fundamentals

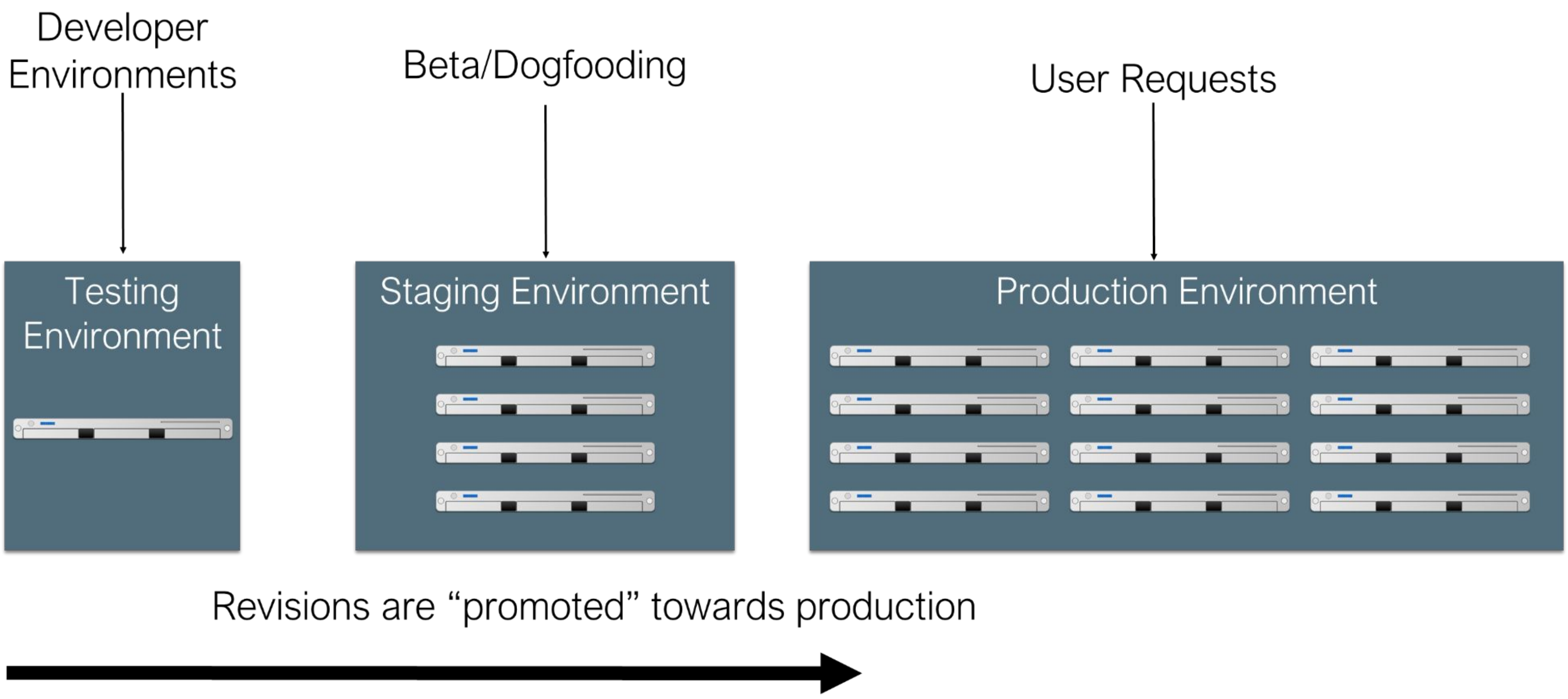
- A single source repository
- Frequent check-ins to main branch
- Automated builds
- Self-testing builds
- Frequent iterations
- Stable testing environments
- Maximum visibility
- Predictable deployments anytime

Staging Environment

As software gets more complex with more dependencies, it's impossible to simulate the whole thing when testing

- Idea: Deploy to a complete production-like environment, but don't have everyone use it
- Lower risk if a problem occurs in staging than in production

Test-Stage-Production



Developer Environments

Beta/Dogfooding

User Requests

Testing Environment

Staging Environment

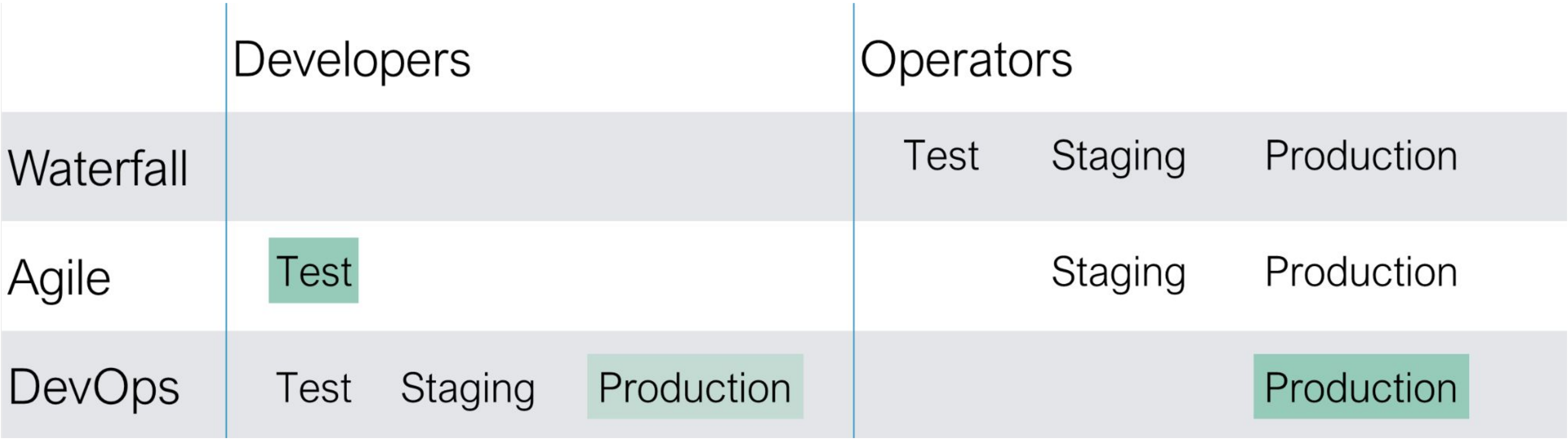
Production Environment

Revisions are “promoted” towards production

Q/A takes place in each stage (including production!)

Operations Responsibility (DevOps)

- Once we deploy, someone has to monitor software, make sure it’s running OK, no bugs, etc
- Assume 3 environments:
  - Test, Staging, Production
- Whose job is it?

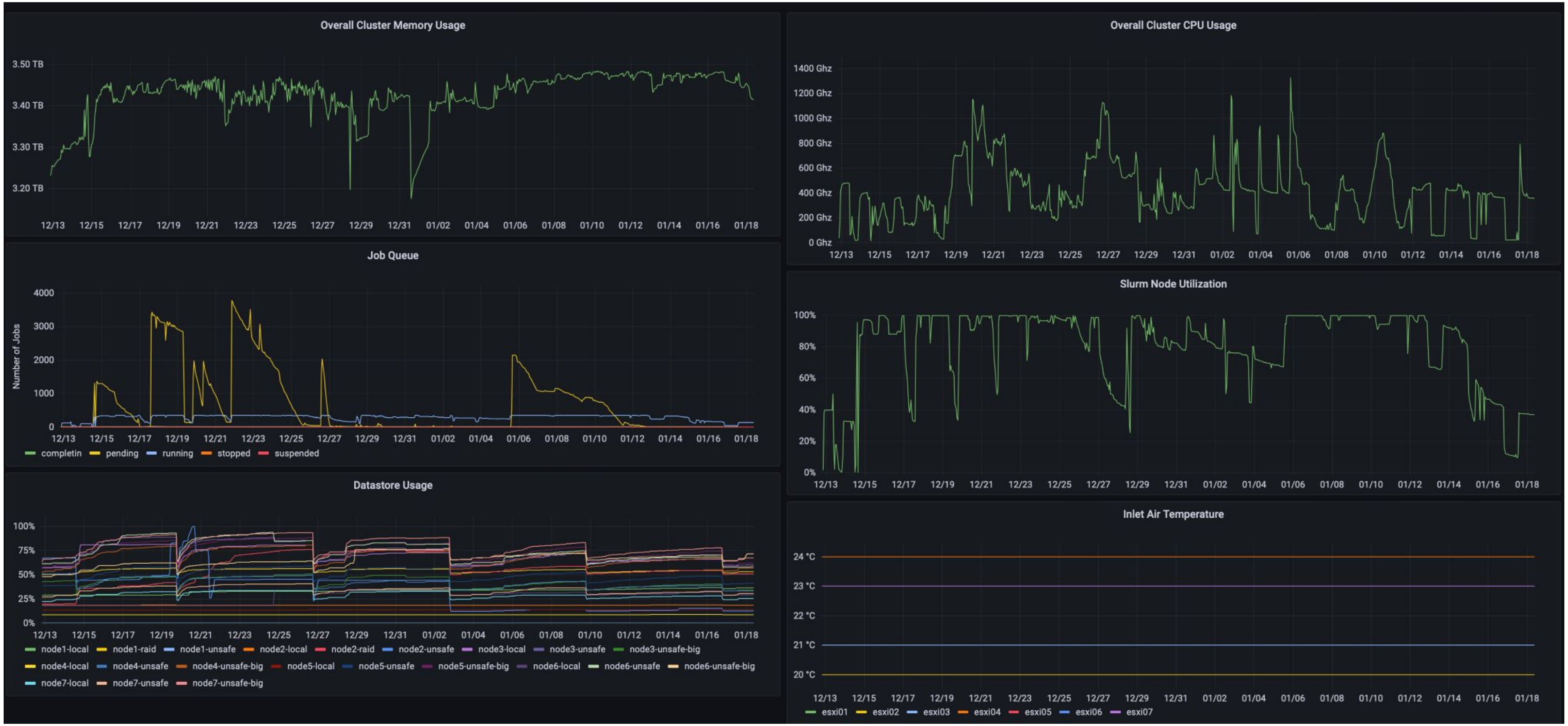


	Developers	Operators
Waterfall		Test Staging Production
Agile	Test	Staging Production
DevOps	Test Staging Production	Production

Monitoring

- Hardware
  - Voltages, temperatures, fan speeds, component health
- OS
  - Memory usage, swap usage, disk space, CPU load
- Middleware
  - Memory, thread/db connection pools, connections, response time
- Applications
  - Business transactions, conversion rate, status of 3rd party components

Monitoring Dashboard



Overall Cluster Memory Usage

Overall Cluster CPU Usage

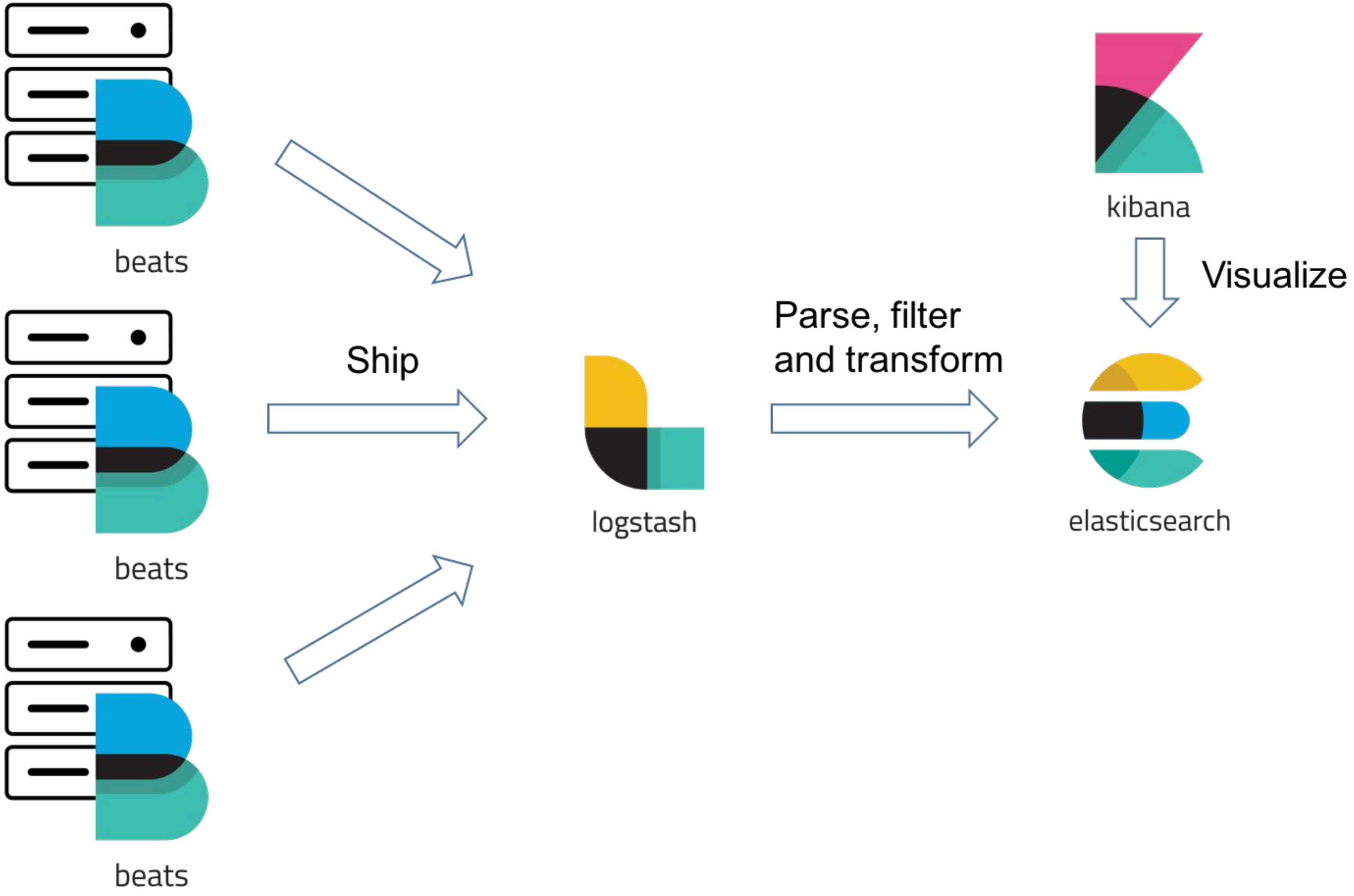
Job Queue

Slurm Node Utilization

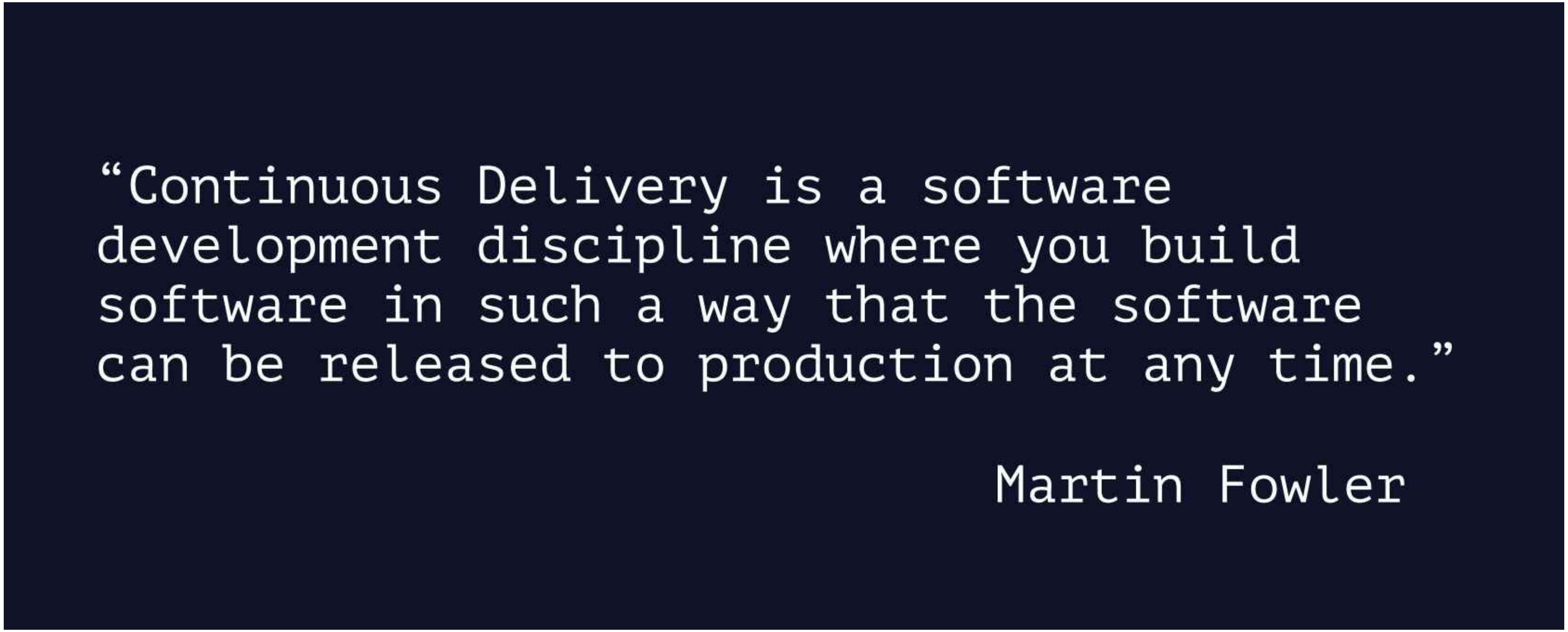
Datastore Usage

Inlet Air Temperature

ELK Stack


beats
beats
beats
Ship
logstash
Parse, filter
and transform
elasticsearch
kibana
Visualize

Quote of the Day



“Continuous Delivery is a software development discipline where you build software in such a way that the software can be released to production at any time.”

Martin Fowler