Progress®

Beyond the Code: Implementing DevOps and CI/CD Techniques for Cloud Apps

Edsel Garcia
Cameron Wright

OpenEdge Development

October 2019

Agenda

- Introduction to CI/CD
- CI/CD Pipeline
- Getting Started
- Demo
- Q&A

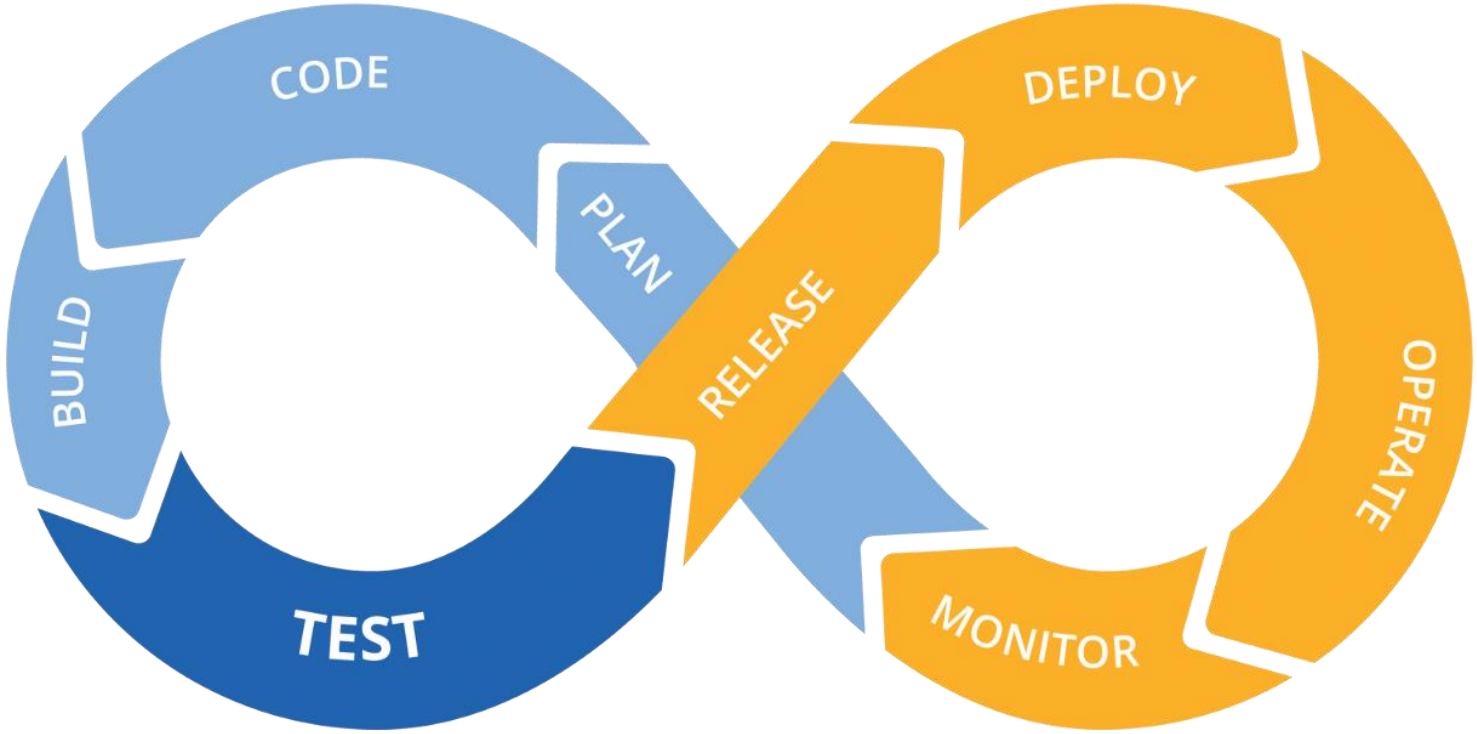
Progress®

CI/CD

Continuous Integration is a software development practice in which you build and test software every time a developer pushes code to the application.

https://dzone.com/articles/9-bene-ts-of-continuous-integration

Continuous Development Cycle



CI/CD Concepts

Continuous Integration

Continuous Integration is the process of checking in code frequently and having **builds triggered by those check-ins that run lightweight tests and code scans that provide quick feedback to engineers** on the code they recently checked in. Continuous Integration is the responsibility of the development team, supported by configurations that were set up by the systems team.

Continuous Validation

Continuous Validation is the process of running **automated functional, acceptance and other relevant tests, which are triggered after successful completion of a Continuous Integration cycle**. The systems team owns the configuration of this environment, but the content of the tests and how they should work comes from development.

Continuous Delivery

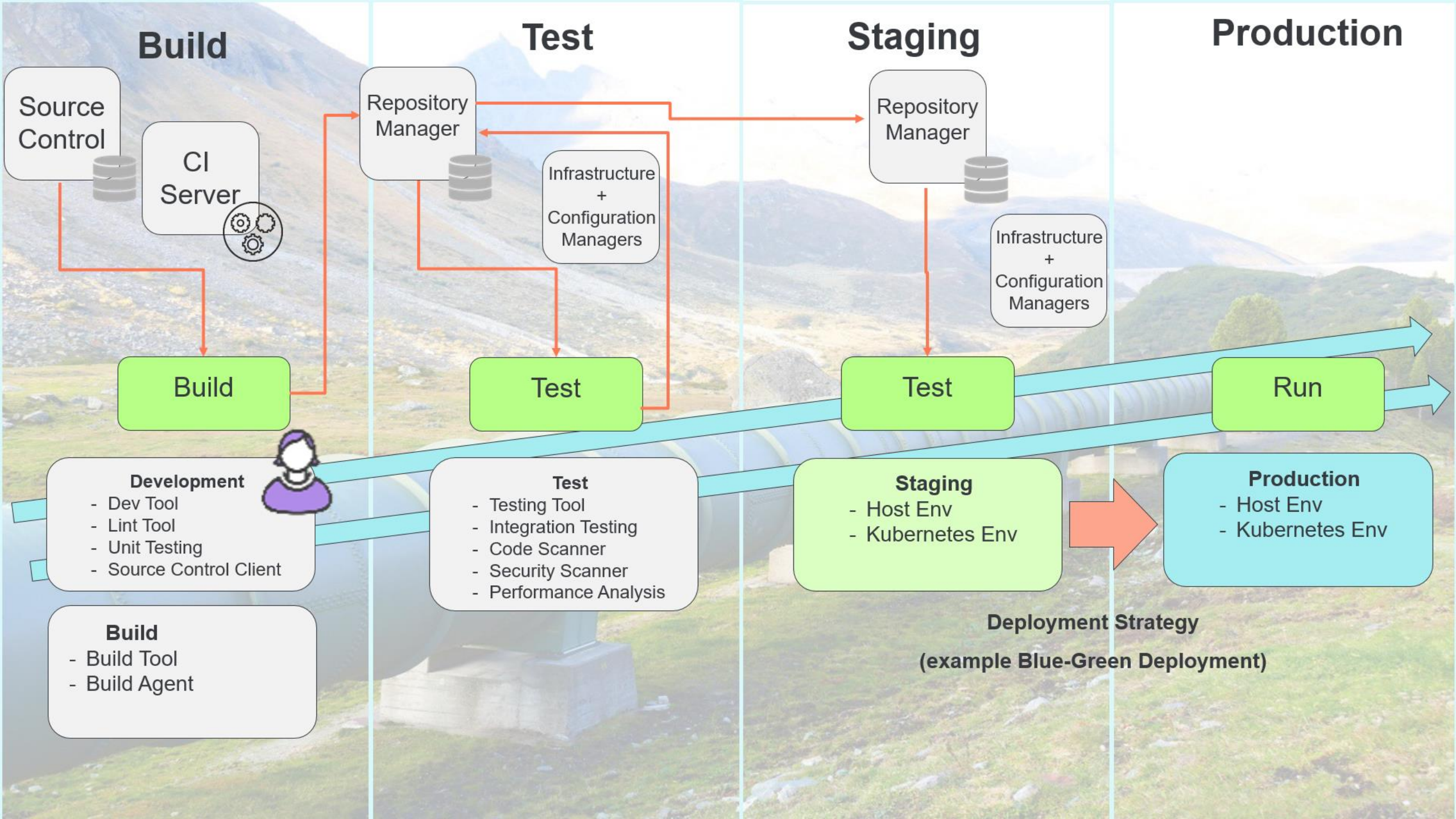
Continuous Delivery is an extension of Continuous Integration and Validation, which ensures that the steps of the previous processes are configured into a **segmented pipeline that results in a potentially ship-able deliverable** at the end of the pipeline. The configuration of this pipeline is mostly owned by the systems team, with input from/usage by development.

Benefits of CI/CD

- Reduced Deployment Risk
- Continuous Feedback
  - Build/Test Feedback
  - User Feedback
- Code Coverage
- High Quality and Confidence
- Productivity

Progress®

CI/CD Pipeline


Build
Test
Staging
Production
Source Control
CI Server
Repository Manager
Repository Manager
Infrastructure + Configuration Managers
Infrastructure + Configuration Managers
Build
Test
Test
Run
Development
- Dev Tool
- Lint Tool
- Unit Testing
- Source Control Client
Test
- Testing Tool
- Integration Testing
- Code Scanner
- Security Scanner
- Performance Analysis
Staging
- Host Env
- Kubernetes Env
Production
- Host Env
- Kubernetes Env
Build
- Build Tool
- Build Agent
Deployment Strategy
(example Blue-Green Deployment)

Progress®

Getting Started

Getting Started

- Build a basic automated pipeline then iterate
  - Codify / Automate and use Infrastructure as Code principles
  - Code Coverage and Quality is key
- Focus on areas of the pipeline based on organization requirements
- A Maturity Model can help to understand the state of CI/CD:
  - http://bekkopen.github.io/maturity-model/
  - https://dzone.com/articles/continuous-delivery-the-holy-grail-of-cloud-app-ma

Getting Started

- Best Practices
  - Automate the Build / IaC
  - Everyone Commits to the Mainline Every Day
  - Isolate and Secure Your CI/CD Environment
  - Build Only Once and Promote the Result Through the Pipeline
- Resources:
  - https://www.martinfowler.com/articles/continuousIntegration.html
  - https://www.digitalocean.com/community/tutorials/an-introduction-to-ci-cd-best-practicesGeneral

Summary

- Become Better Developers
- Automation, Quality, Best Practices
- Use a Maturity Model to understand where you are
- Focus on areas of the pipeline based on requirements and iterate

Progress®

Demo

Progress®

Q&A


Progress®