Modern Continuous Delivery

A journey in four acts

ThoughtWorks®

WHO AM I?
KEN MUGRAGE

PRINCIPAL TECHNOLOGIST -
THOUGHTWORKS OFFICE OF THE CTO

Some Definitions

DevOps: A culture where people, regardless of title or background, work together to imagine, develop, deploy and operate a system.

- me

https://kenmugrage.com/2017/05/05/my-new-definition-of-devops/

@kmugrage

Some Definitions

Continuous Delivery: The ability to get changes of all types—including new features, configuration changes, bug fixes and experiments—into production, or into the hands of users, safely and quickly in a sustainable way.

- Jez Humble

DISCLAIMER: You’re about to get a lot of information which is designed to make you want more

The Origins of Continuous Delivery

UNITED KINGDOM – 2005

We knew everything

- Signed Agile Manifesto
- Defined Continuous Integration
- Created the first (or second) CI server
- Created Selenium

Java – Write once, run anywhere

- Developed a system on Windows laptops to be deployed to a Solaris cluster
- Did all the right Continuous Integration things
- One small issue…

It didn’t work in production

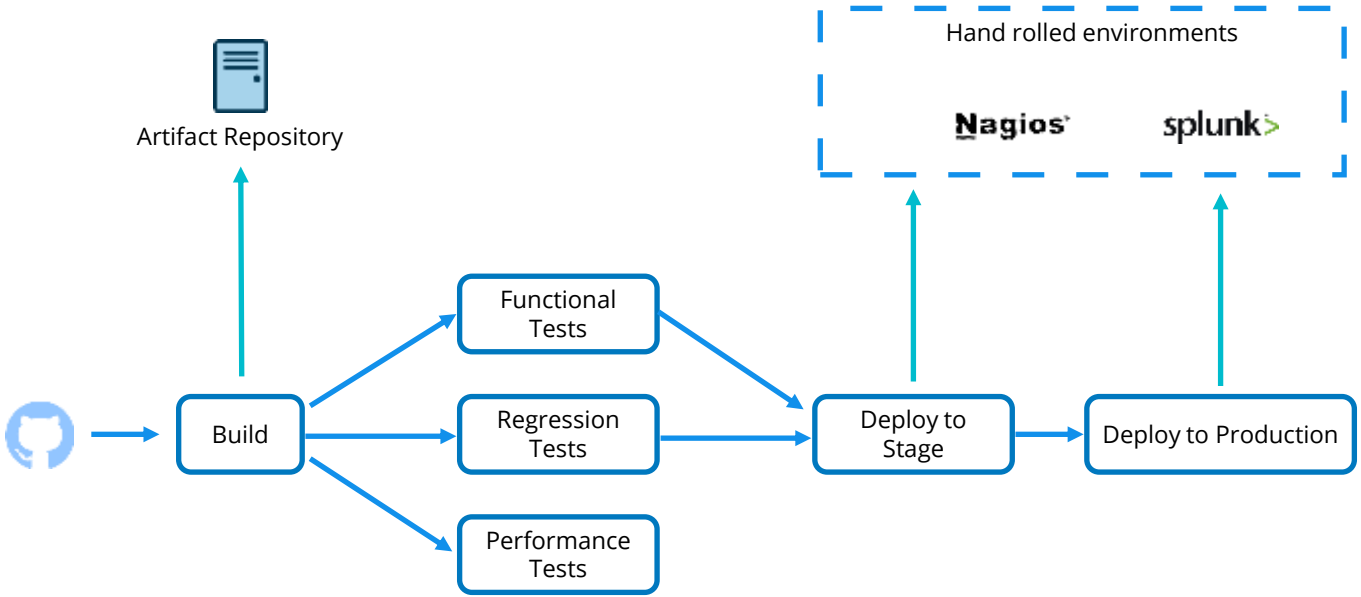
Conan The Deployer

- Massive shell script
- Automated deployment to a cluster after every successful CI run
- Deployment became a non-issue

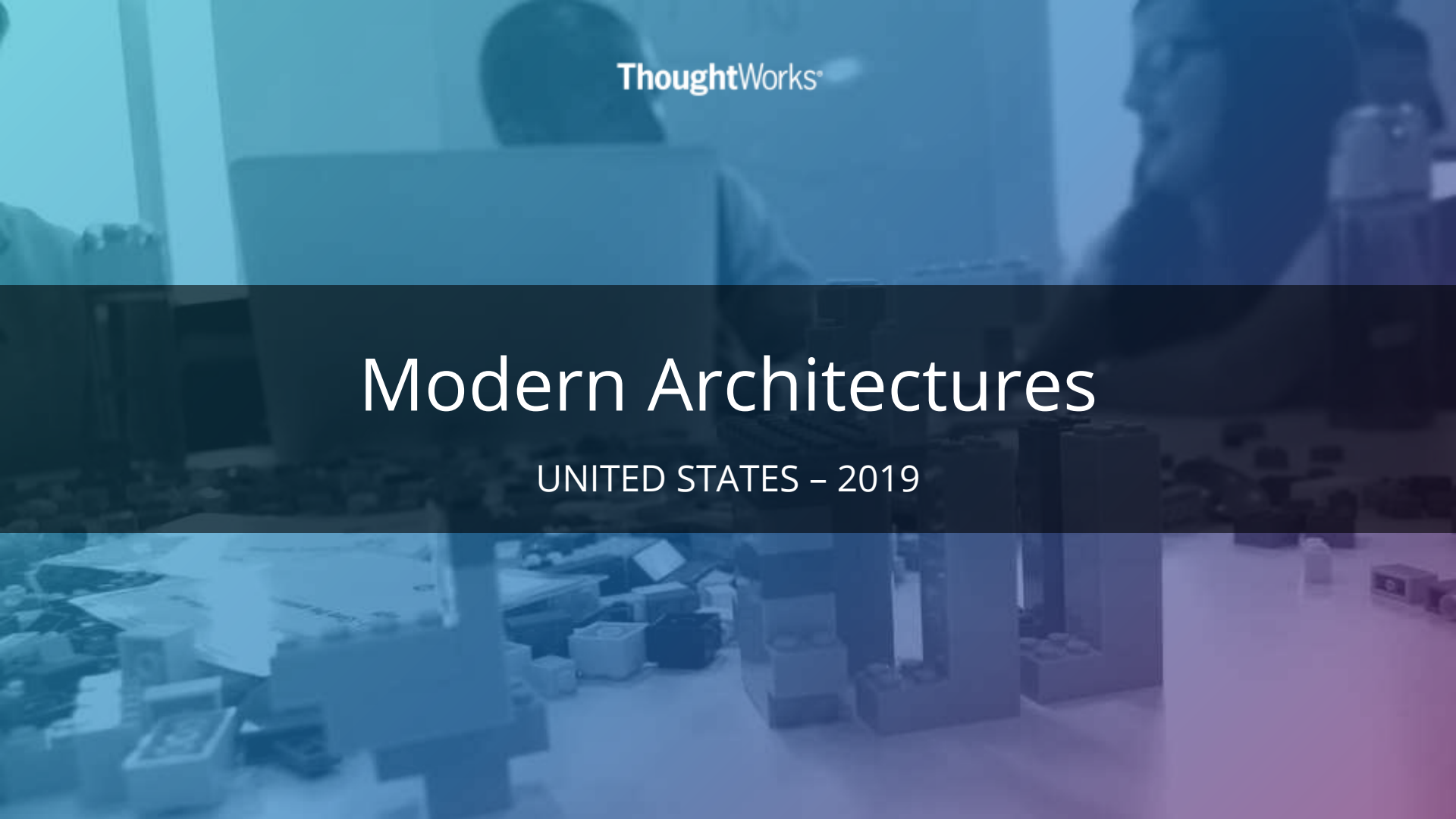
The Output

- Jez Humble, Dan North and Chris Read write “The Deployment Production Line”
- Jez Humble and David Farley release the book “Continuous Delivery“
- Continuous Delivery becomes an expectation

Traditional Continuous Delivery



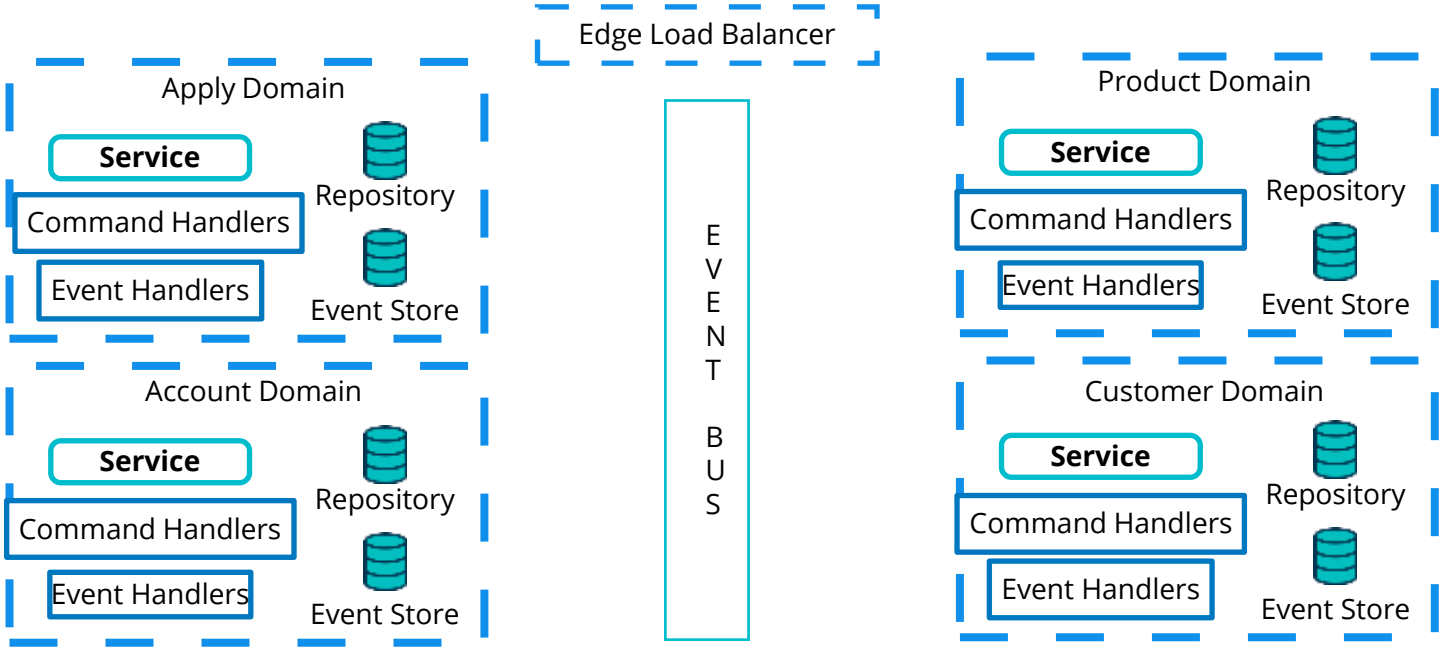
@kmugrage



Modern Architectures

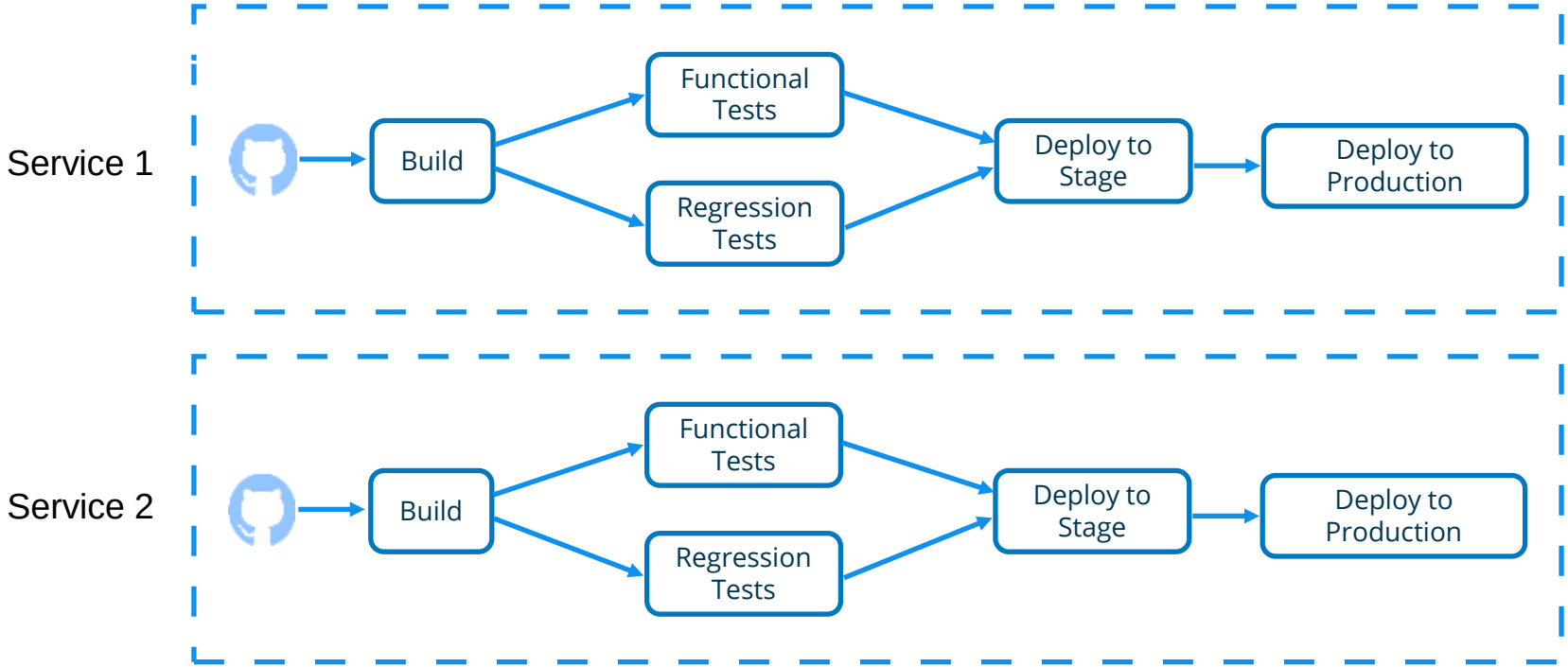
UNITED STATES – 2019

Financial Services Platform



@kmugrage

Smaller, Faster Pipelines

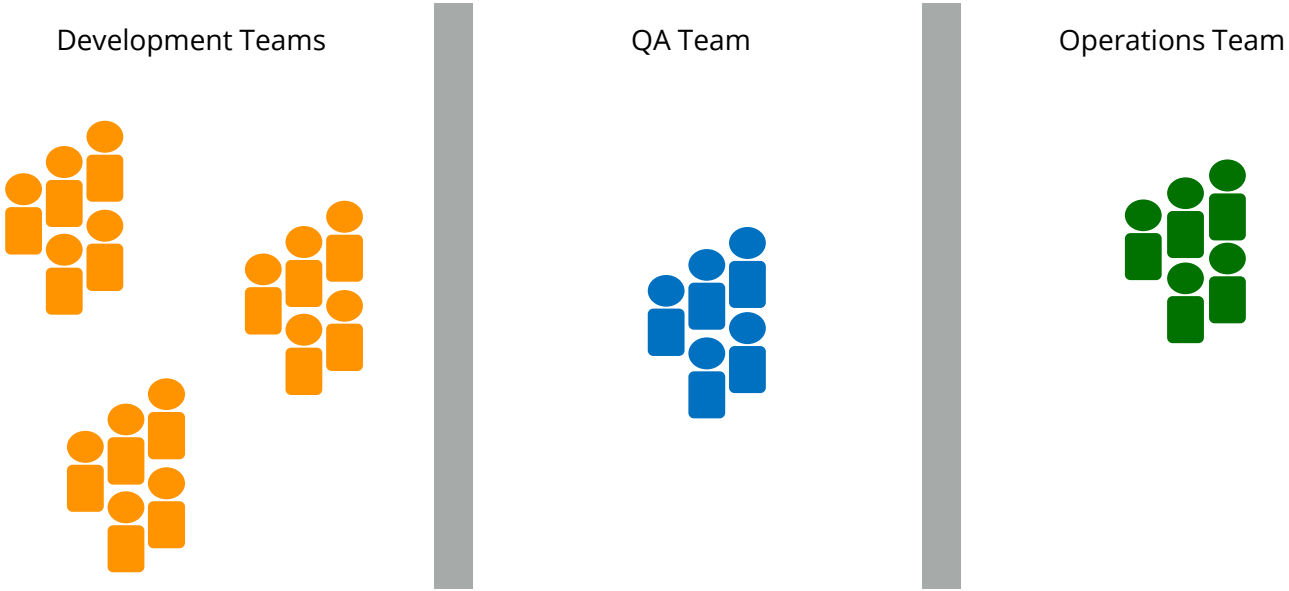


@kmugrage

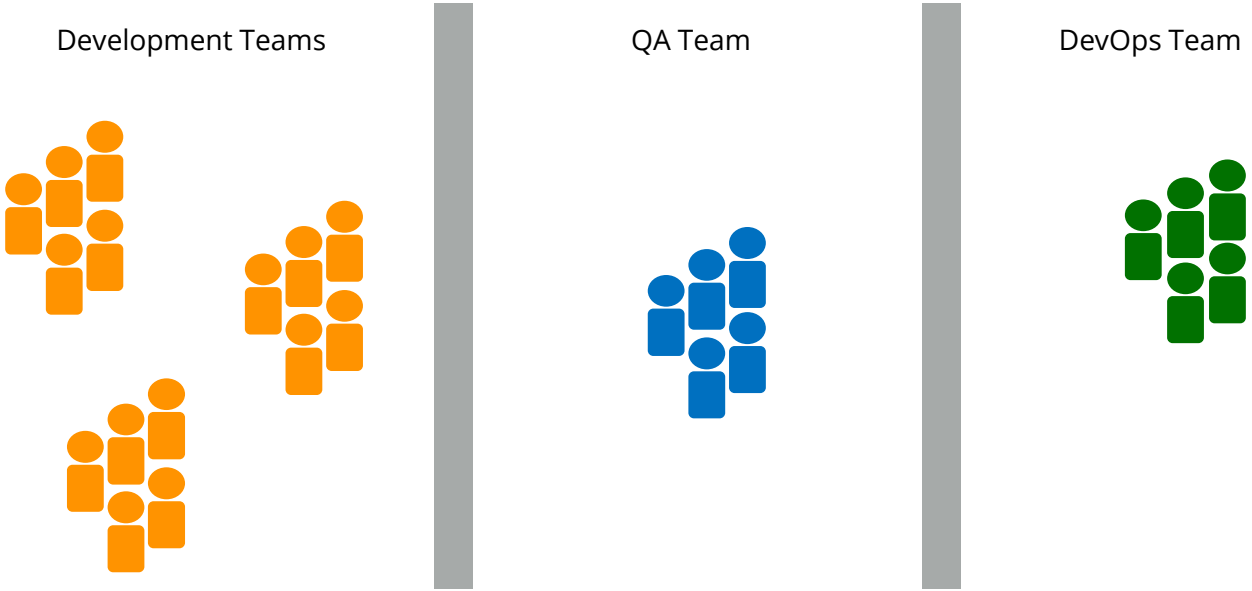
Everything Changes

THE DETAILS

Traditional Structure



Renaming Ops isn’t a solution



@kmugrage

Products over Projects (do the DevOps)

Service 1

Service 4

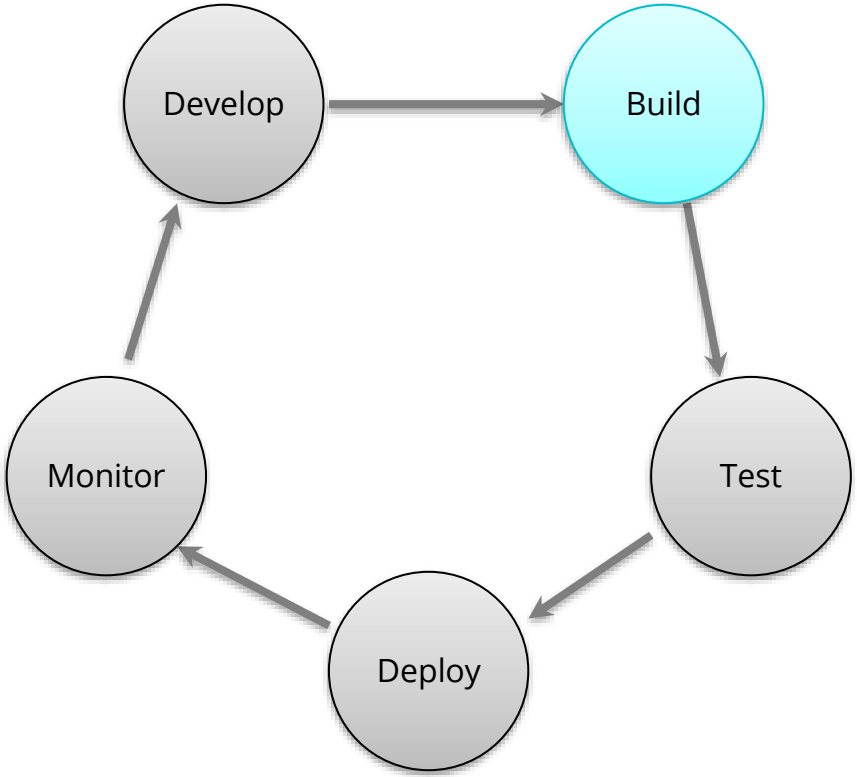
Service 2

Service 5

Service 6

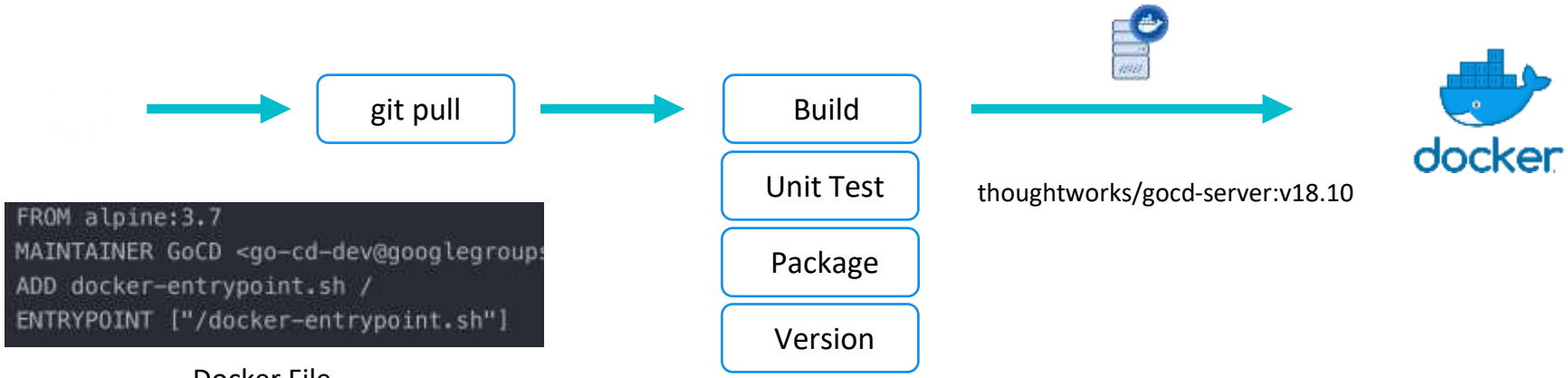
Service 3

@kmugrage



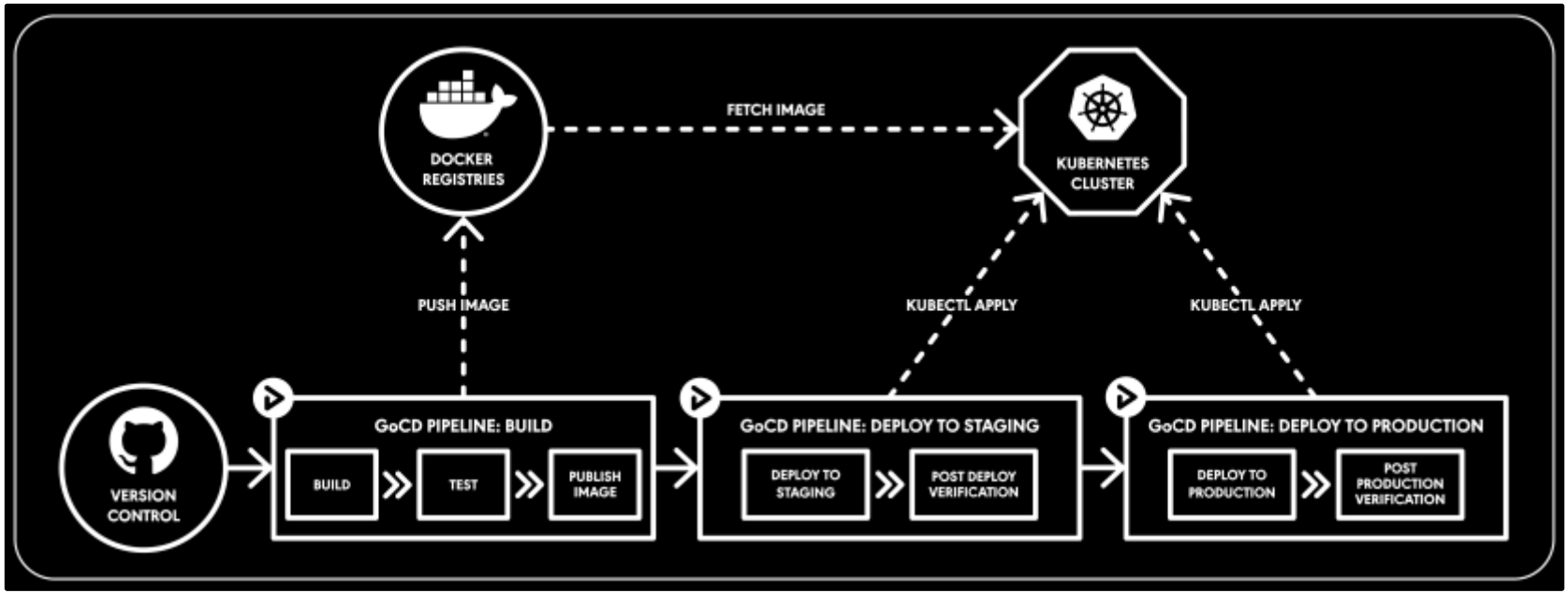
@kmugrage

The New Build Artifact



@kmugrage

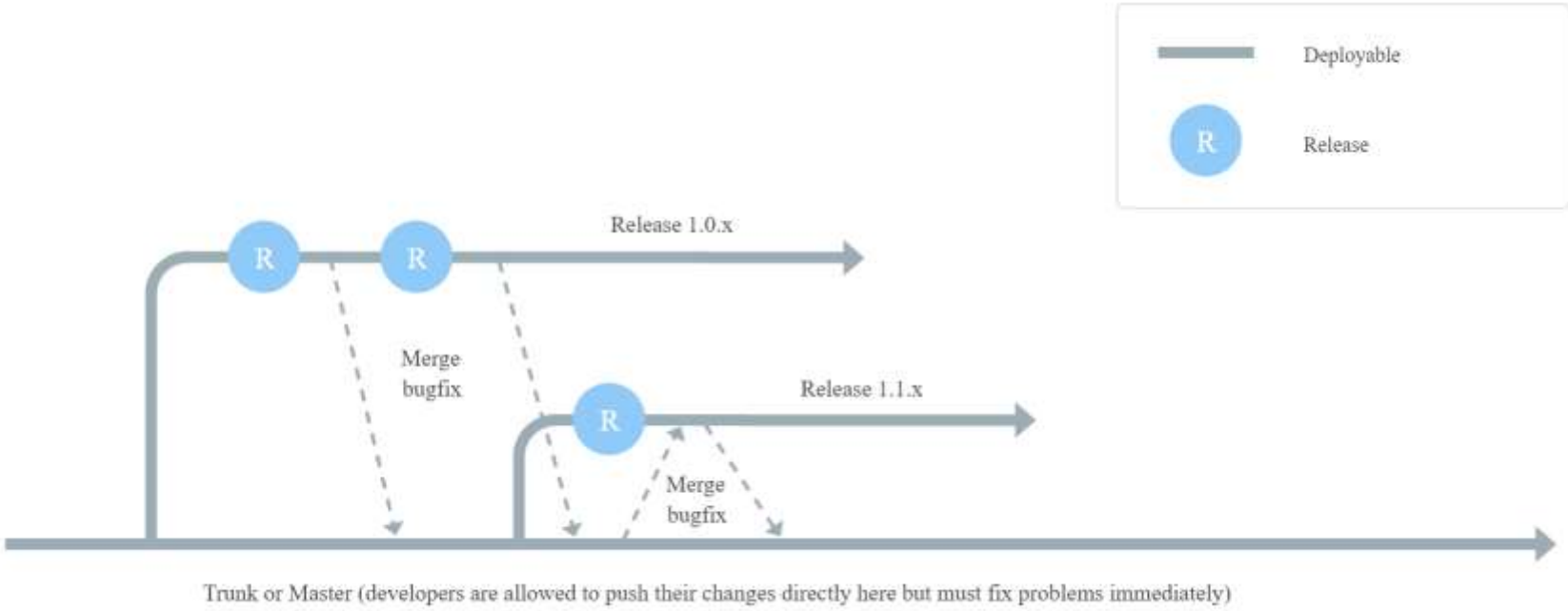
Modern pipeline with declarative deployments



@kmugrage

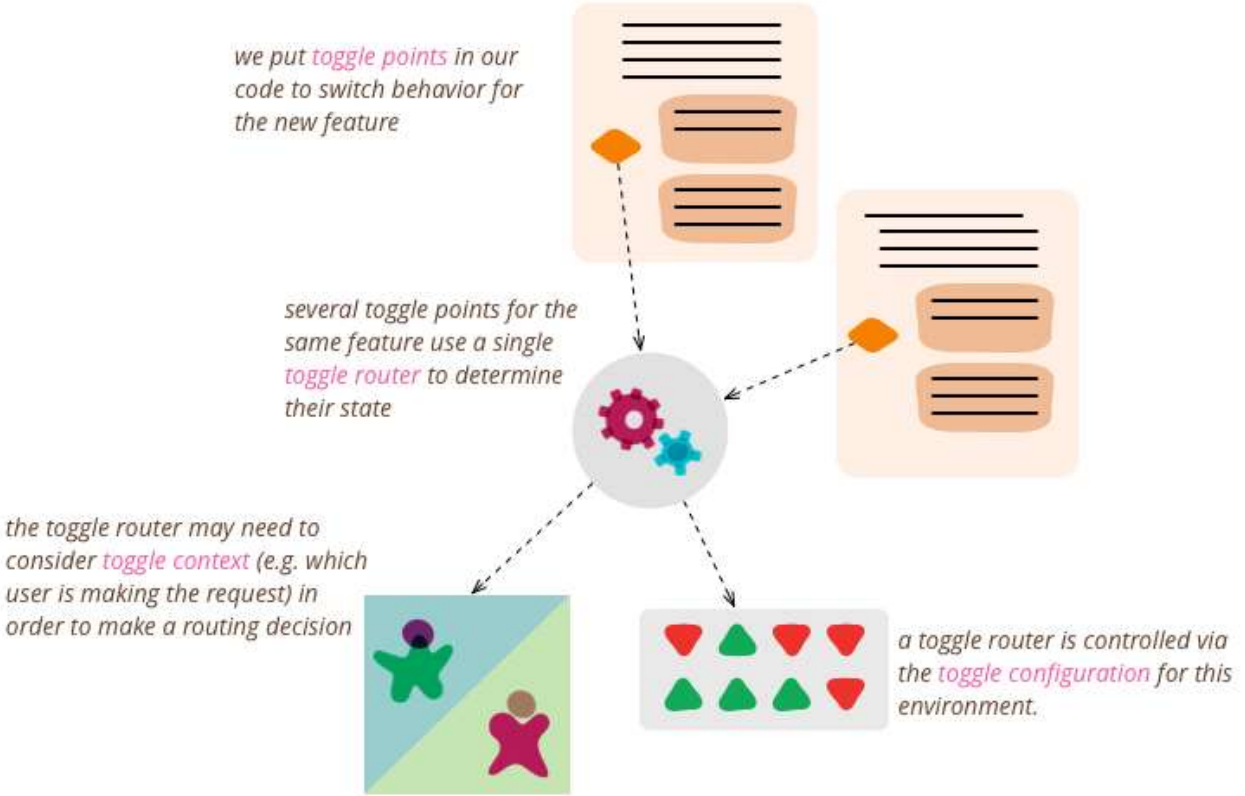
Trunk Based Development

Everyone pushes to master

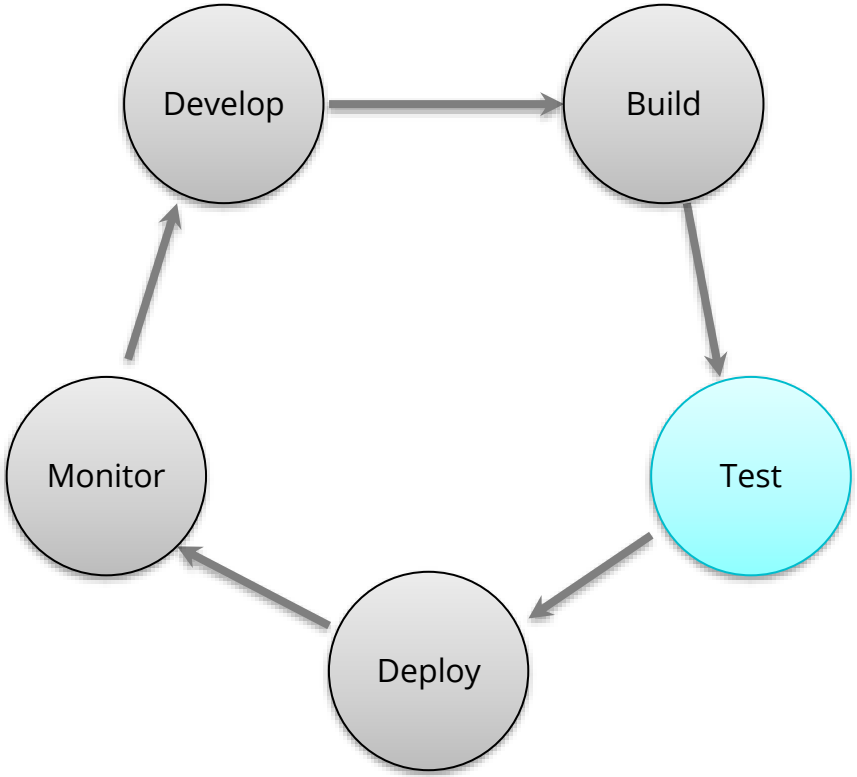


@kmugrage

Feature Toggles

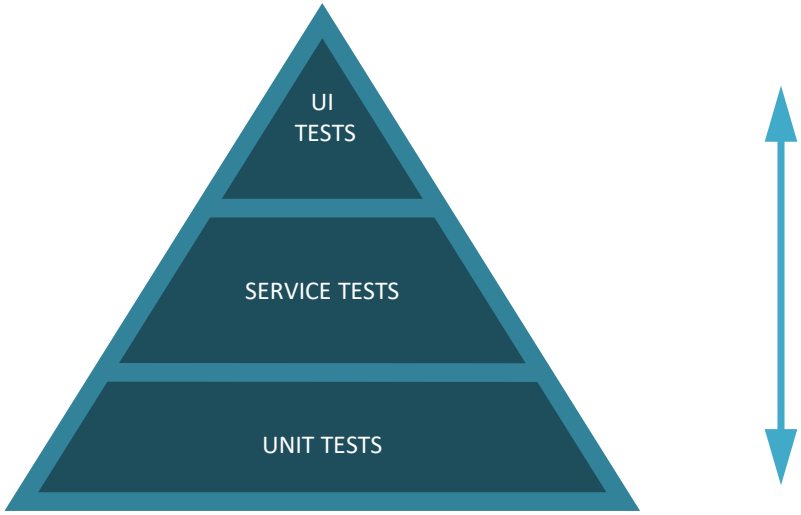


@kmugrage

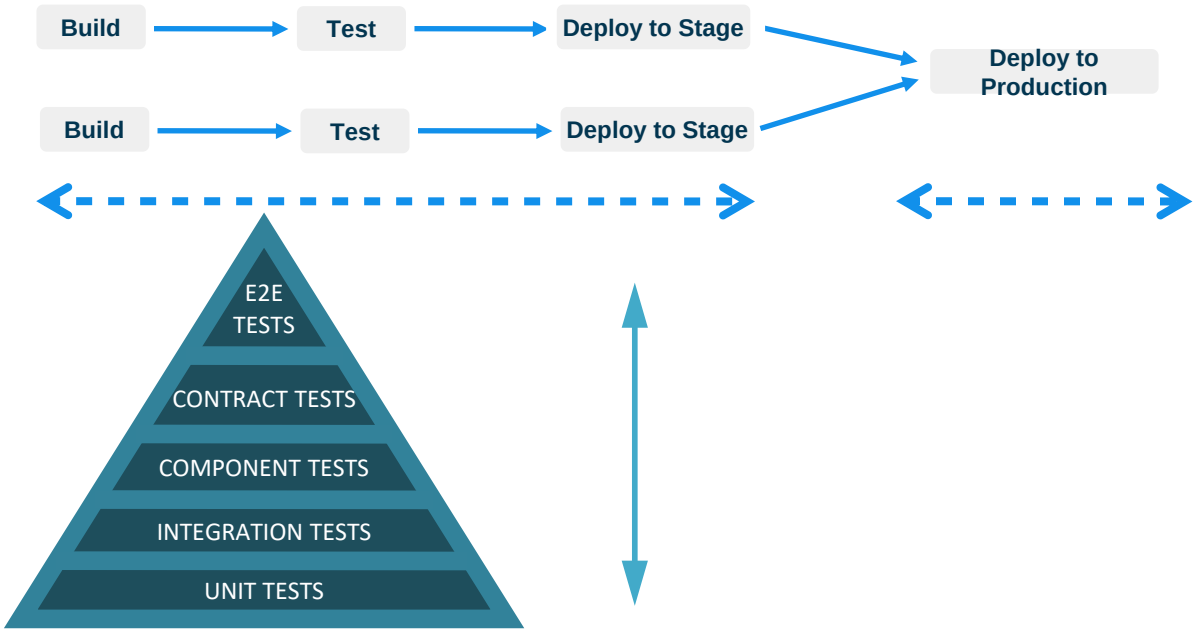


@kmugrage

The Test Pyramid

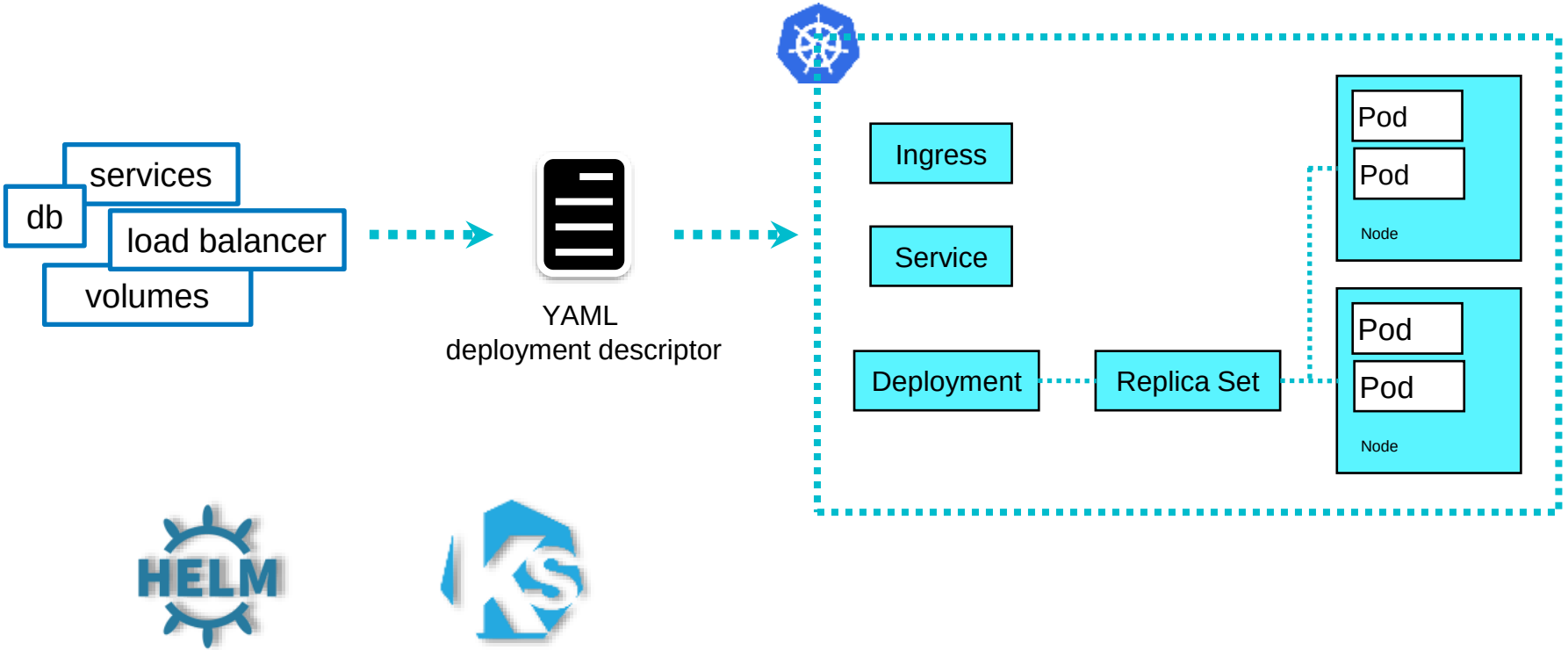


The Test Pyramid In Context



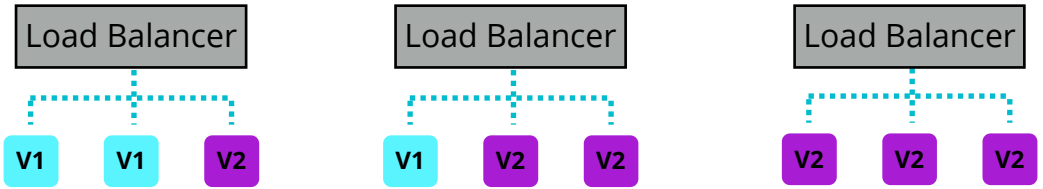
@kmugrage

Declarative Deployments

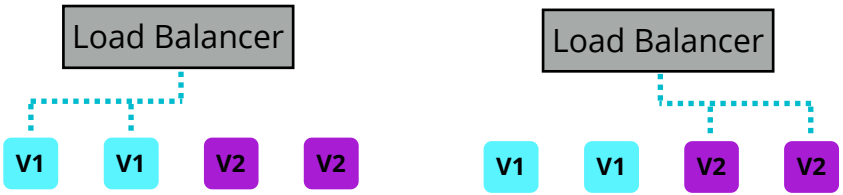


@kmugrage

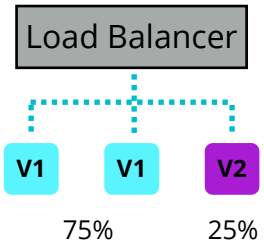
Deployment Strategies



Rolling Update



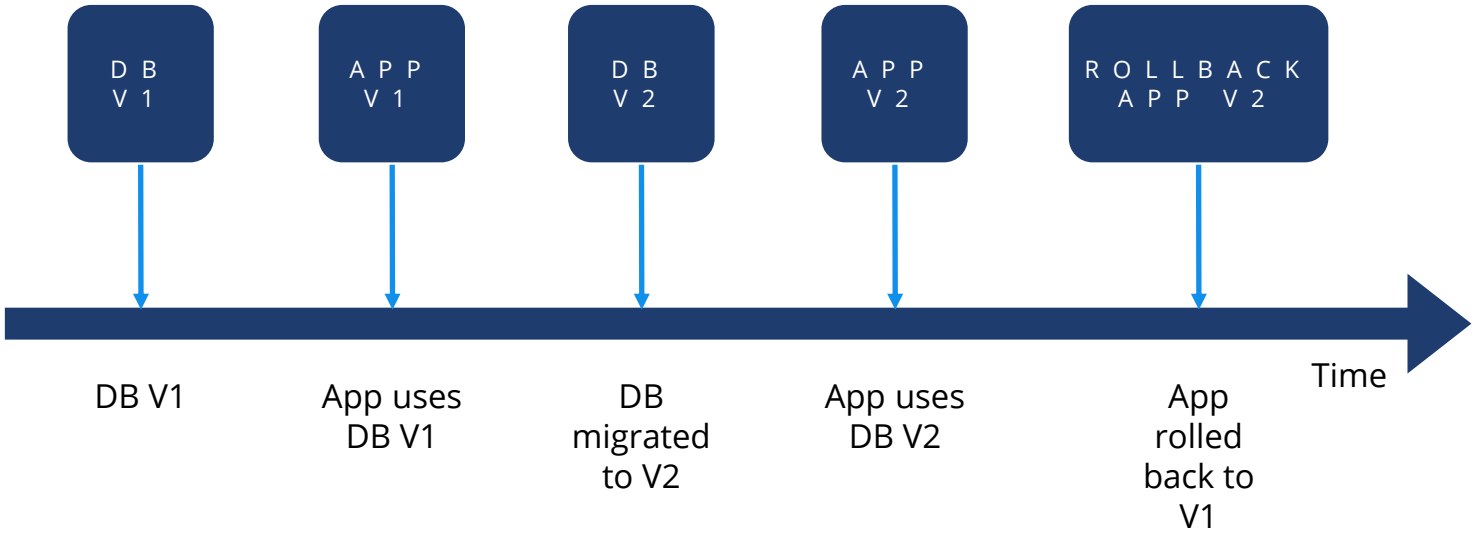
Blue Green Deployment



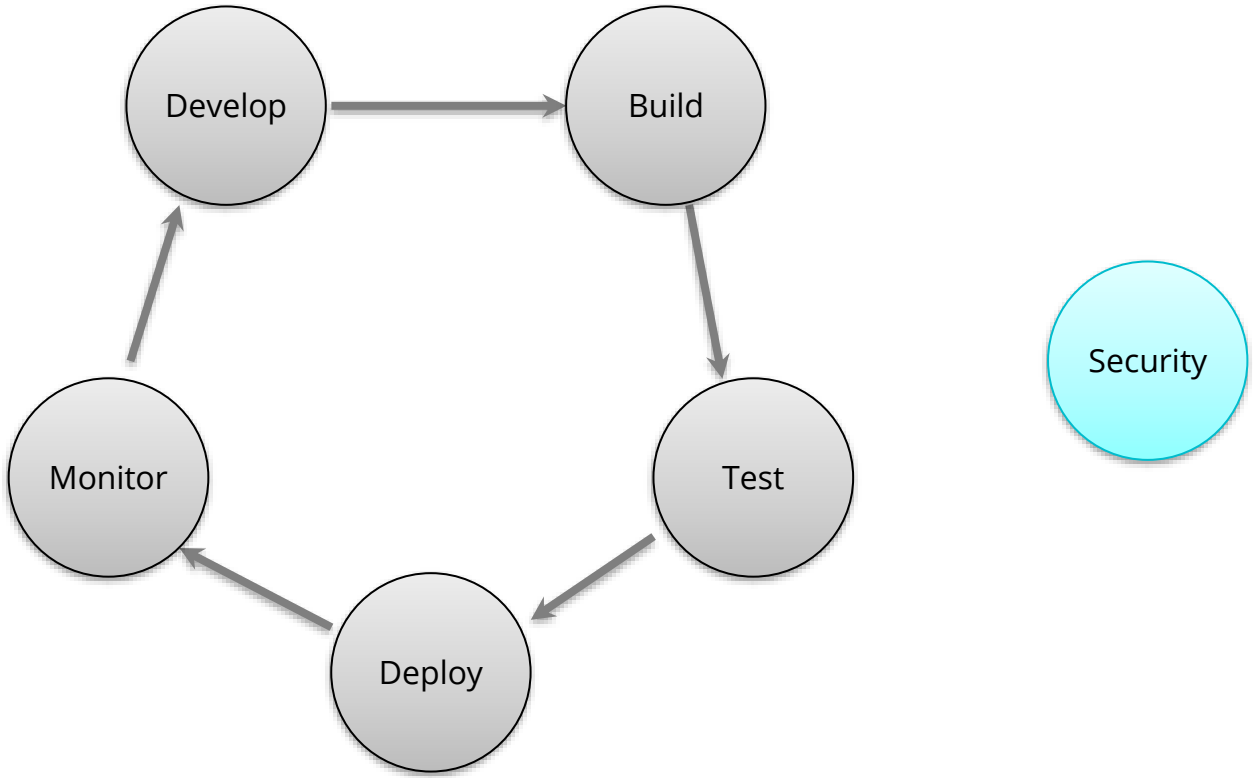
Canary Deployment

Release DB Changes Out Of Band

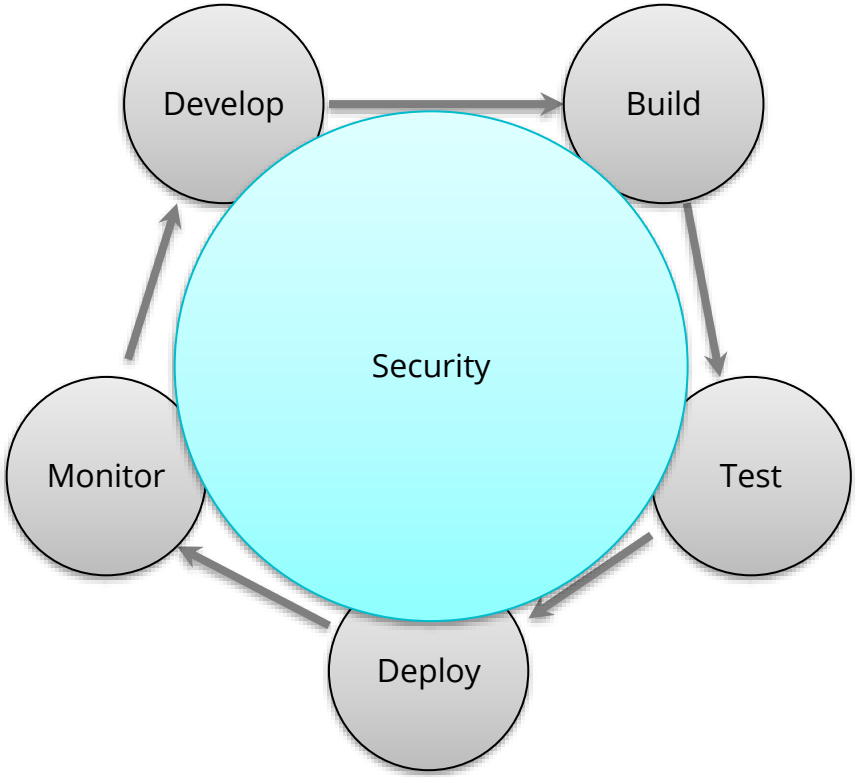
Expand / Contract Pattern



@kmugrage

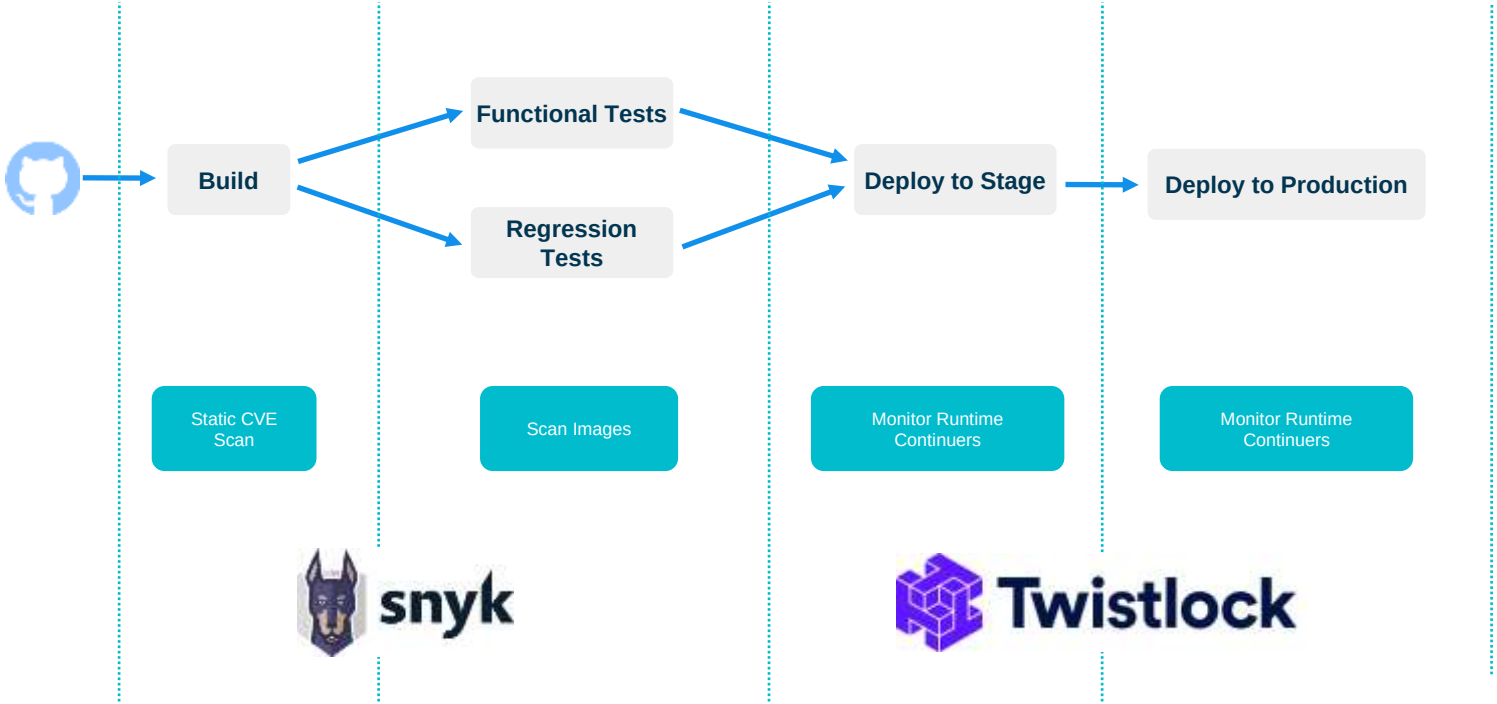


@kmugrage

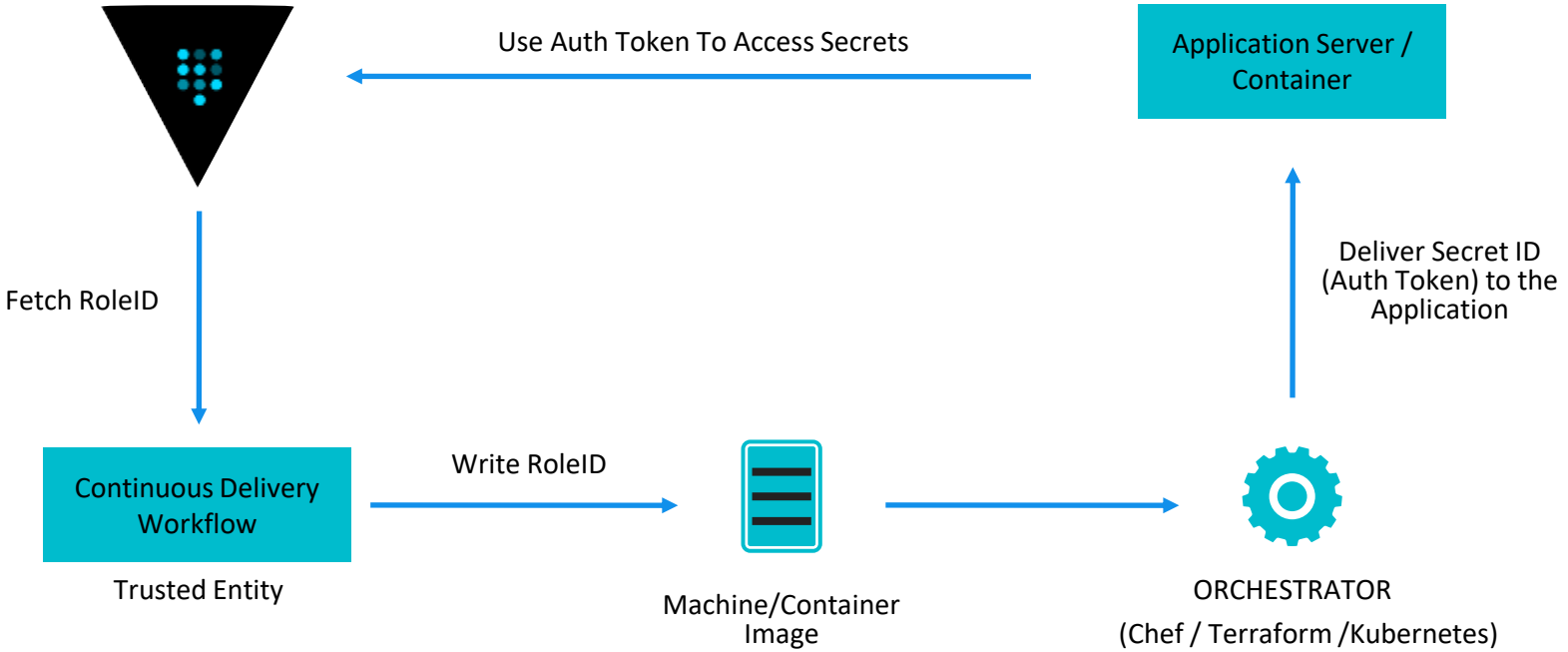


@kmugrage

Vulnerability Planning



Secrets Management



@kmugrage

EPILOGUE

In the future

- Teams must be structured to own small pieces
- Docker and Kubernetes are here to stay... until they aren't
- You must test in production. Own it and architect for it
- Security is everybody's job

Resources & Citations

- http://agilemanifesto.org/
- https://martinfowler.com/articles/originalContinuousIntegration.html
- https://continuousdelivery.com/wp-content/uploads/2011/04/deployment_production_line.pdf
- https://martinfowler.com/articles/microservice-trade-offs.html
- https://martinfowler.com/articles/practical-test-pyramid.html
- https://trunkbaseddevelopment.com/
- https://martinfowler.com/articles/feature-toggles.html
- https://docs.honeycomb.io/learning-about-observability/intro-to-observability/
- https://martinfowler.com/articles/201701-event-driven.html

Thank You

Ken Mugrage - @kmugrage

ThoughtWorks®