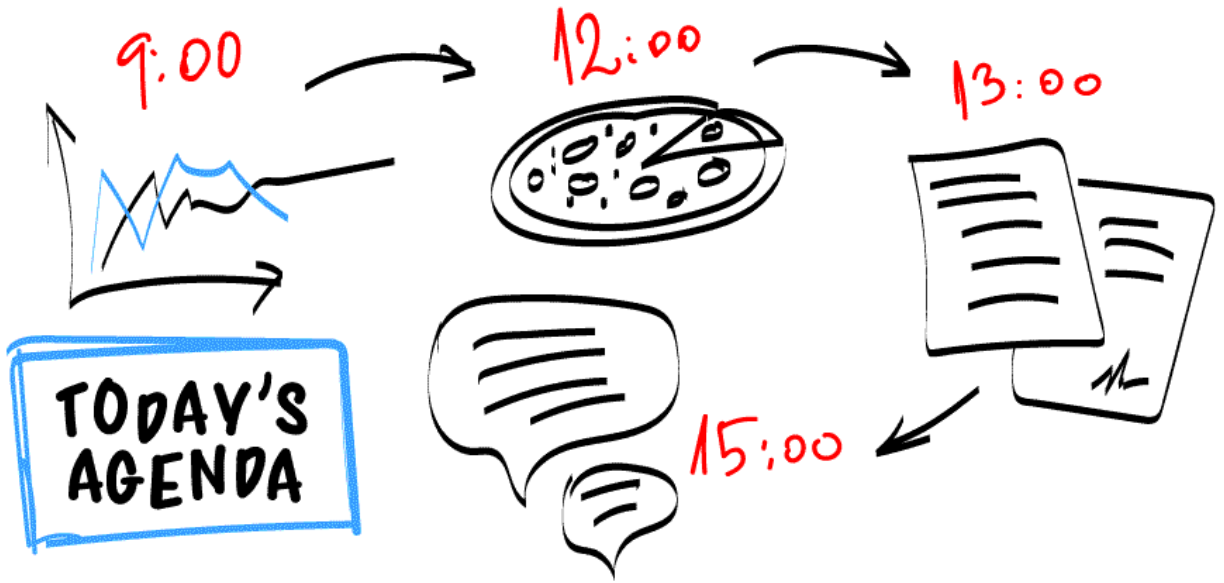

KubeCon
CloudNativeCon
OPEN SOURCE SUMMIT
China 2019

Promoting Kubernetes CI/CD to the next level

Tim Pouyer
@tpouyer
DevOps & SRE Leader
IBM Cloud Private

Agenda

- Introduction
  - Who am I
  - What is IBM Cloud Private
  - How did we get here
- DevOps
  - For ”shrink-wrapped” software
- Focus Areas
  - Automation
  - “Shift-Left”
  - Culture



Introduction

Who am I?

Tim Pouyer
@tpouyer
DevOps & SRE Leader
IBM Cloud Private

- Lead Architect of CI/CD squad
- SRE squad lead
- 14 years experience with CI/CD & DevOps
- World Traveler >60 countries visited
- Boy Scout (Eagle Scout)

Introduction

What is IBM Cloud Private?

- IBM’s on-prem Kubernetes distribution
- Intended to be utilized as a private cloud
- Enterprise security and governance capabilities
- 600 customers in 2 years
- Quarterly releases
- 200+ integrated open source packages/tools
- >180 developers located in 4 countries

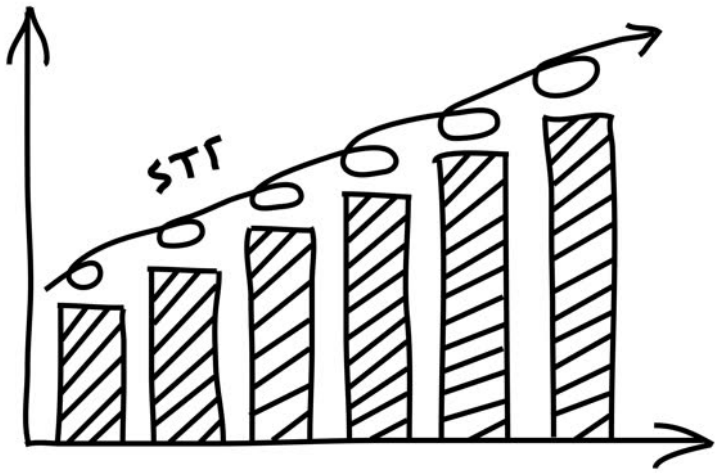
Introduction

How did we get here?

Grew from a few dozen devs incorporating tens of Docker images to a few hundred devs incorporating hundreds of Docker images.

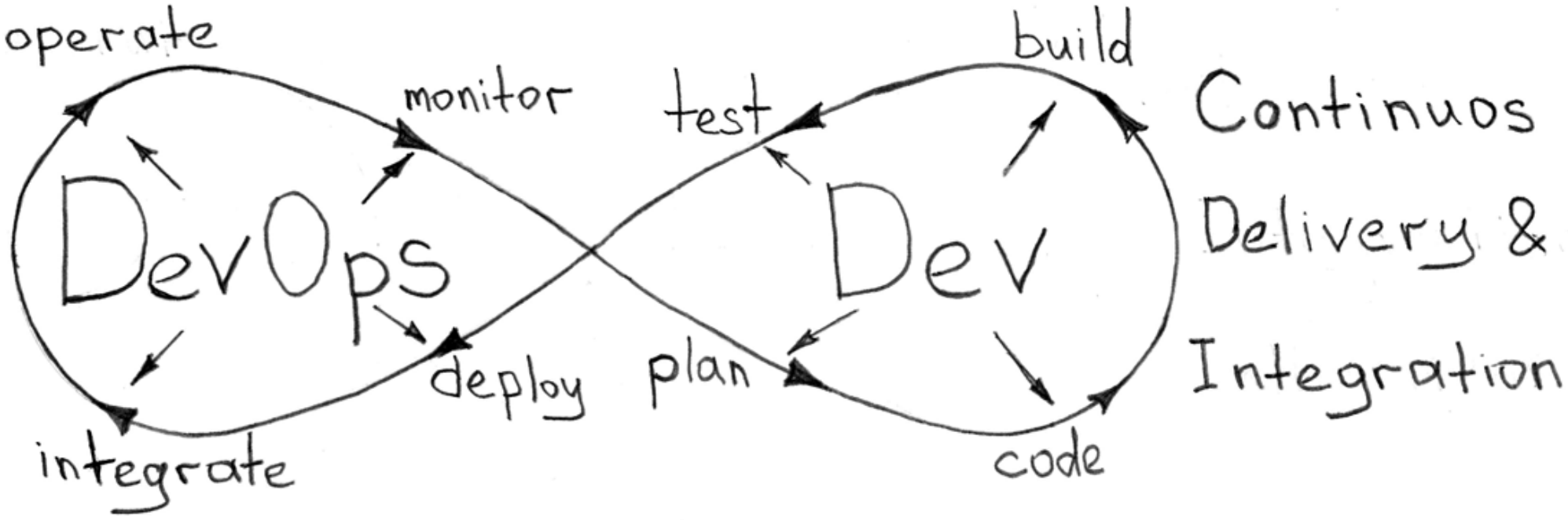
Added support for 2 additional platforms (ppc64le & s390x) in addition to x86

>600 customers in 2 years



DevOps

For ”shrink-wrapped” software?



DevOps

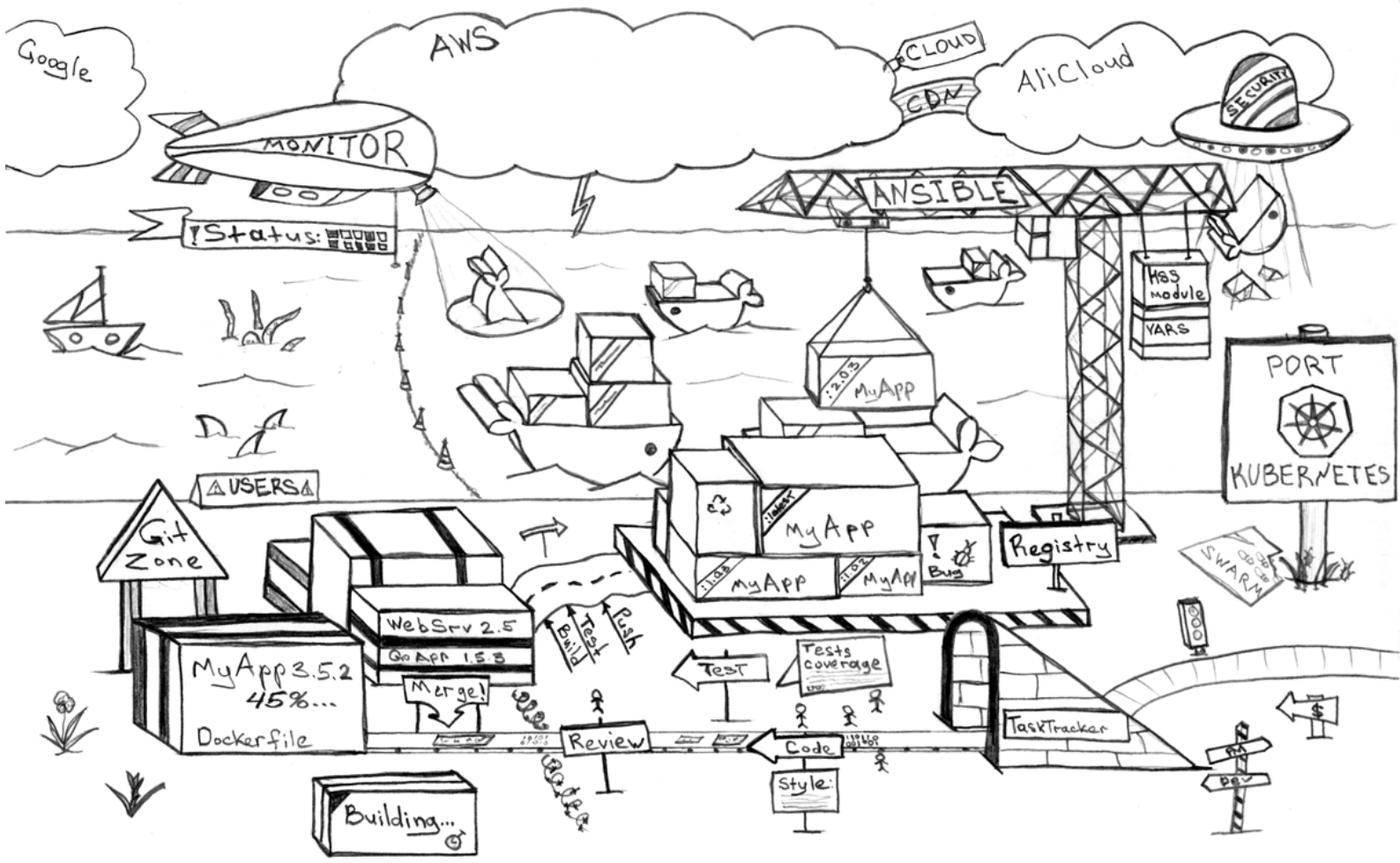
Too many topics to cover

This presentation will focus on 3 major areas

1 - Automation

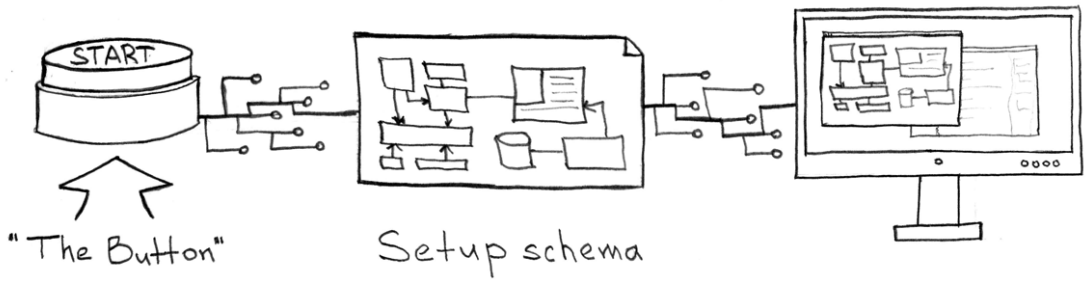
2 - “Shift-Left”

3 - Culture

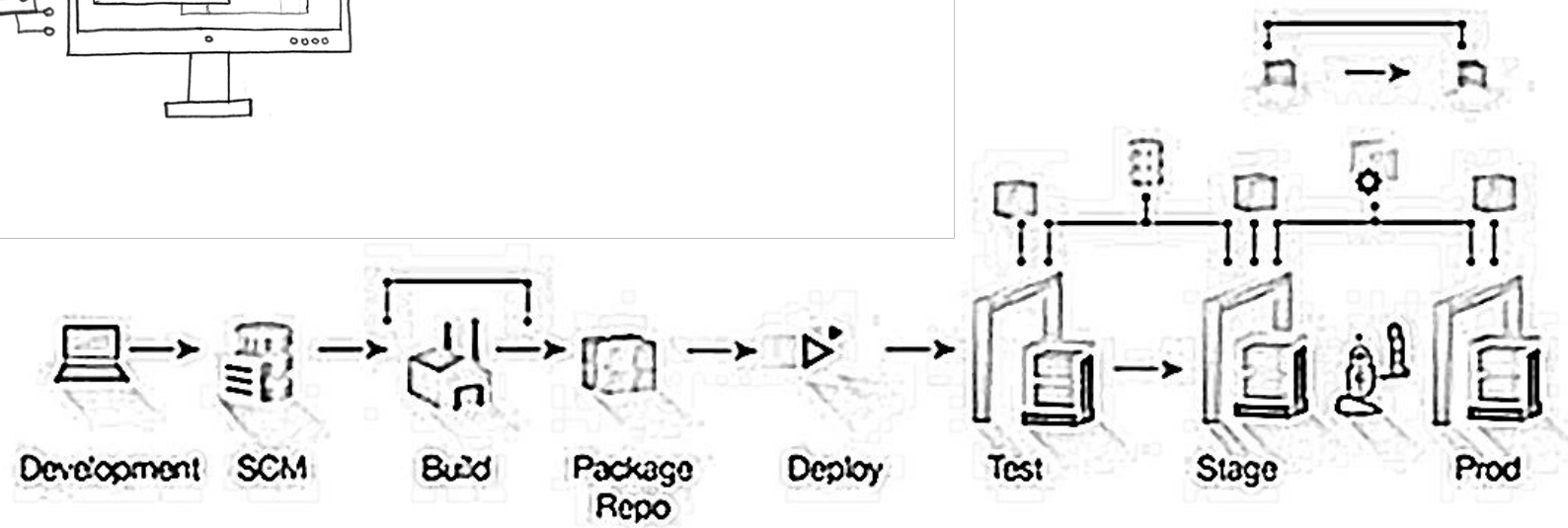


Focus Areas

Automation: guiding principles



- Push button
- Full-stack
- Easily shared



- Easy to use
- Familiar tooling
- Work across all archs

Focus Areas

Automation: cross squad, cross org shared assets

Built on the build-harness project from Cloud Posse
https://github.com/cloudposse/build-harness
Forked for internal use and customization, working to Open Source our fork soon

Shared Makefile targets stored in github and easily included into your custom Makefile

```
40
41  -include $(shell curl -so .build-harness -H "Authorization: token $(GITHUB_TOKEN)" -H "Accept: application/vnd.github.v3.raw"
42
```

Utilize Terraform to provision infrastructure across supported cloud providers

- Makefile.azure
- Makefile.fyre
- Makefile.icp_aws
- Makefile.nutanix
- Makefile.openshift_aws

- All terraforms exposed via build-harness
- Common set of targets for linting, building, deploying, testing, etc…

```
11  .PHONY: deploy\:azure
12  ## Deploy icp on azure
13  deploy\:azure:
14          @$(GIT) clone -b $(AZURE_GIT_TF_DEPLOY_BRANCH) $(AZURE_GIT_TF_DEPLOY_PROJECT) $(AZURE_DEPLOY_DIR)
15          @$(SELF) terraform:apply TERRAFORM_VARS_FILE=$(AZURE_TERRAFORM_VARS_FILE) TERRAFORM_DIR=$(AZURE_DEPLOY_DIR)/templates/i
```

Focus Areas

Automation: supported archs, cloud providers, topologies

Automated deployment for each cloud provider

Use specific docker image for requested arch

Use override files and resource types in terraform to support different deployment topologies

HA *multi-az* *ceph-rook*

non-HA *glusterfs*

mixed arch (x86 w/ ppc64le)

Focus Areas

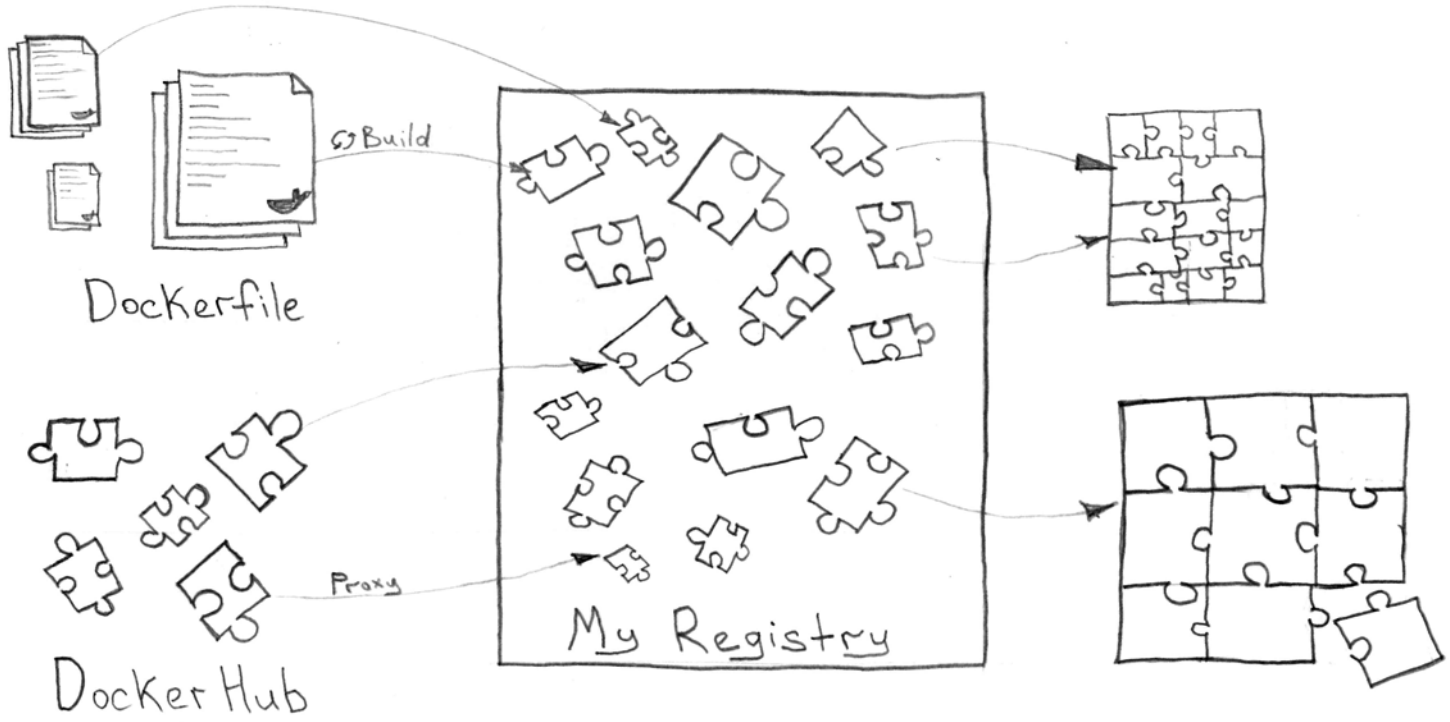
Automation: package assembly, version coordination

Use Artifactory

Use Helm Charts

JFrog Artifactory

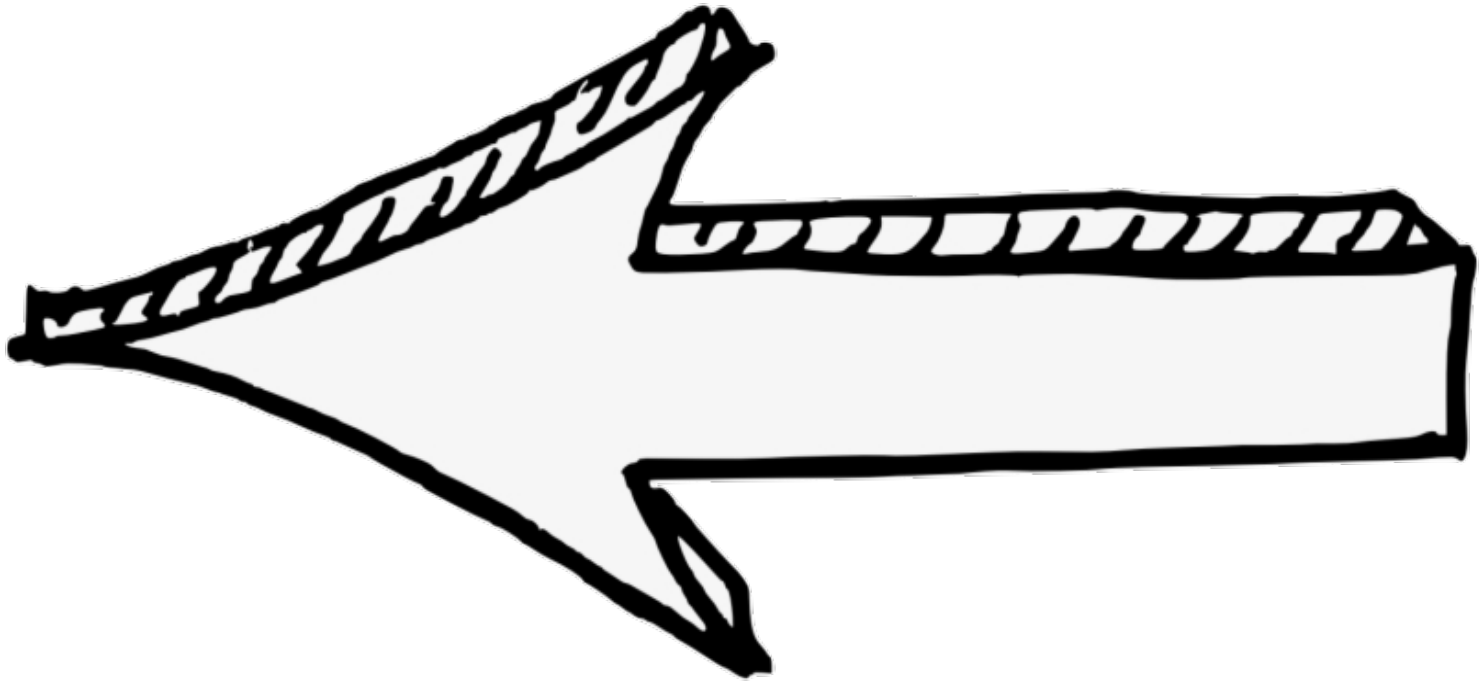
HELM



Focus Areas

“Shift-Left”

The earlier we can perform these tasks
the easier to fix identified issues



canary deploys

integration tests

functional tests

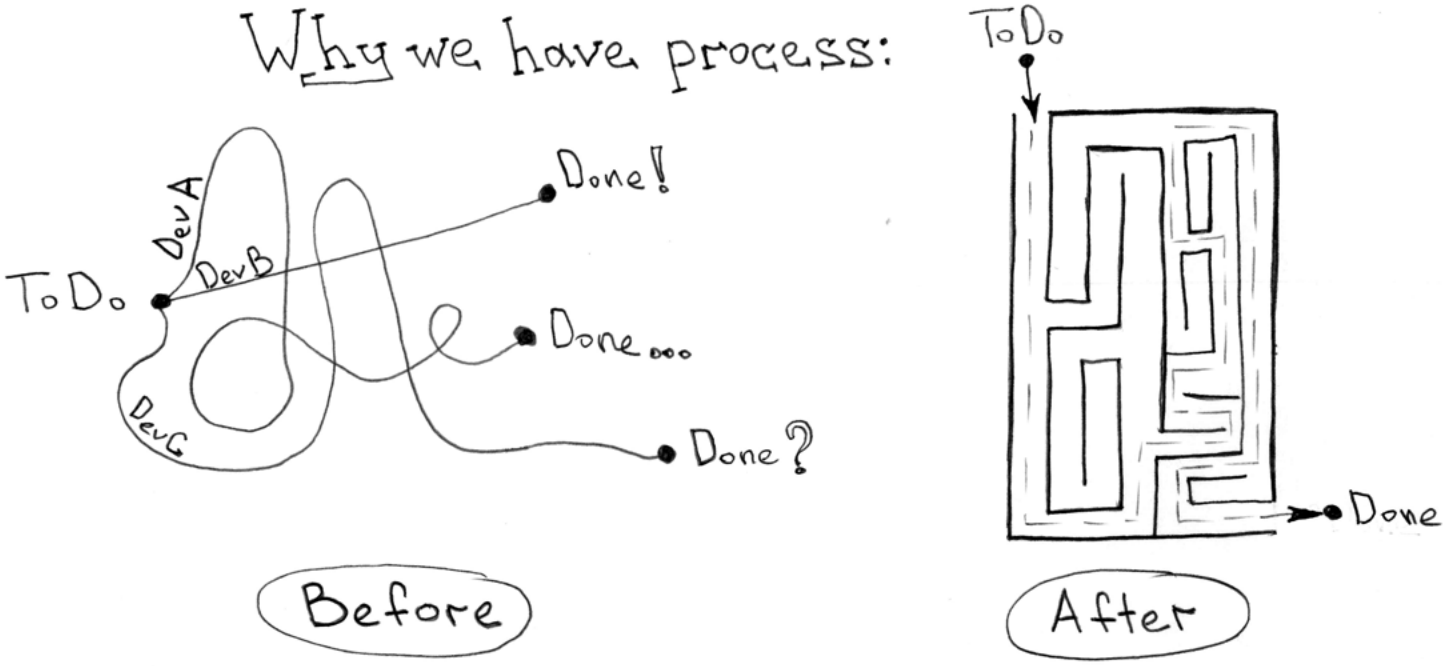
vulnerability scans

docker image security scans

Focus Areas

“Shift-Left”: promotion stages

Don’t run a service?



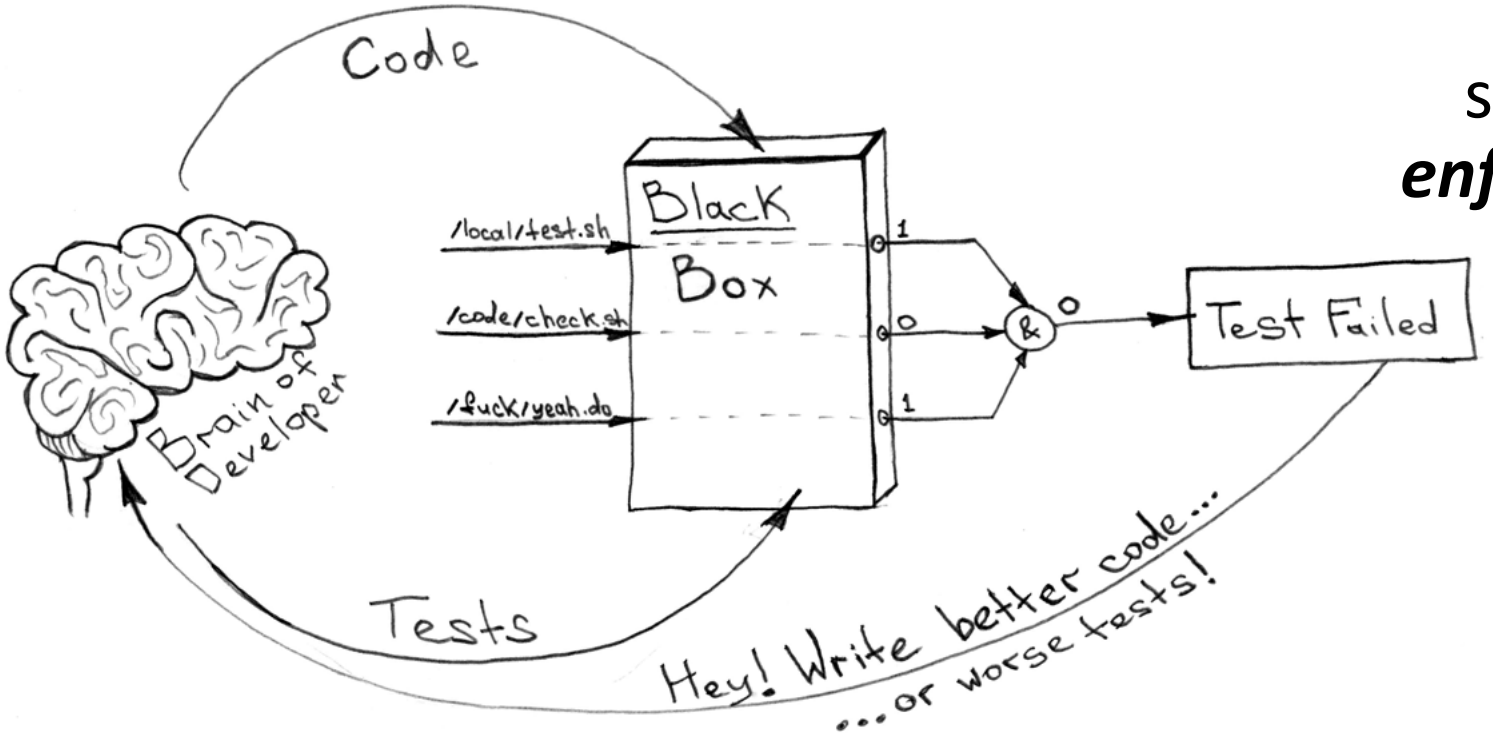
Use Artifactory docker & helm repos as Stages

scratch -> integration -> daily -> edge -> stable -> release

JFrog Artifactory

Focus Areas

“Shift-Left”: quality gates



set criteria for each ”stage”

enforce criteria, no exceptions!

no pass… no proceed

Focus Areas

“Shift-Left”: shortening feedback loop

get builds in front of consumers sooner in dev cycle

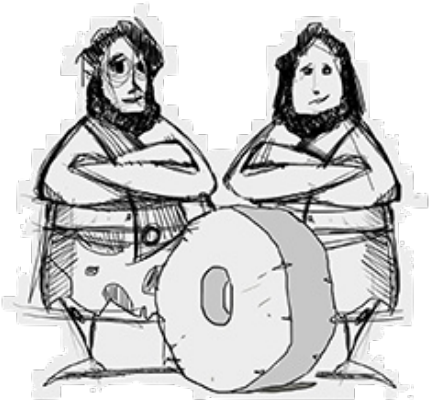
gather feedback quickly

act on that feedback immediately

slack

JFrog Artifactory

good idea



great idea

Focus Areas

Culture

were not talking about differences in nationalities

change is hard

developers don’t like to give up control

developers don’t like to change their process

managers want to ship product

nobody wants a release date to slip

Focus Areas

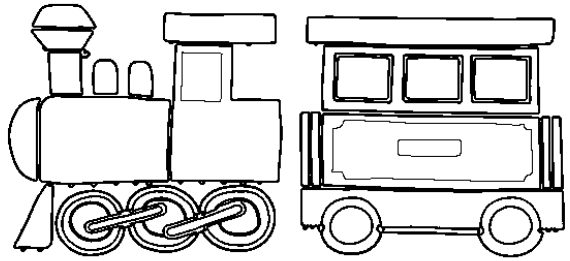
Culture: dev controlled builds -> ci/cd controlled builds

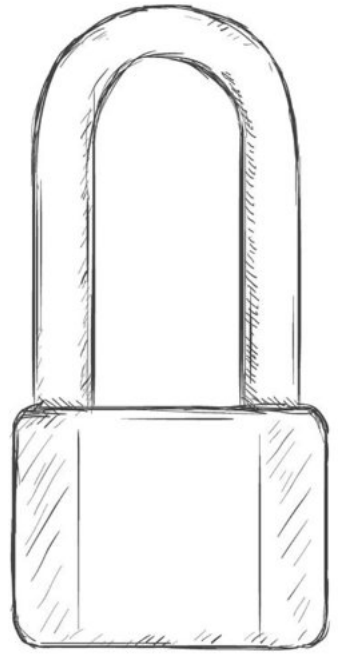
dev builds can work in small teams

unmanageable in large teams

squads own their build processes

ci/cd owns the combined process





ci/cd conducts the train, all components are passengers

get on board the release train or get off

Focus Areas

Culture: understand the process

spend time to educate

explain the “why” not just the “how”

make sure everyone understands… not just the devs

Focus Areas

Culture: trust in the process

it will be tough

it will challenge you

be strong in your resolve

know that the ends will justify the means

Credits

KubeCon CloudNativeCon OPEN SOURCE SUMMIT China 2019

Couldn’t have done it without these people

Andrew McNamara

ICP DevOps Specialist

IBM Cloud Private