Chaos Testing and Testing

What is Testing?

Asserting the expectation of the input and outputs of functions.

```
require "test/unit/assertions"
include Test::Unit::Assertions

hello = 'world'

assert_equal 'world', hello, "hello function should return 'world'"
```

What is Chaos Testing?

Building a system to withstand and tolerant any kind of failure by **purposely introducing random failures** in a **production system**.

ChaosKube

TestKube

ChaosMonkey

Helm Chart Directory Structure

A helm chart is a collection files within a **directory**

```
wordpress/
  Chart.yaml          # A YAML file containing information about the chart
  LICENSE             # OPTIONAL: A plain text file containing the license for the chart
  README.md           # OPTIONAL: A human-readable README file
  values.yaml         # The default configuration values for this chart
  values.schema.json  # OPTIONAL: A JSON Schema for imposing a structure on the values.yaml file
  charts/             # A directory containing any charts upon which this chart depends.
  crds/               # Custom Resource Definitions
  templates/          # A directory of templates that, when combined with values
                      # will generate valid Kubernetes manifest files.
  templates/NOTES.txt # OPTIONAL: A plain text file containing short usage notes
```

Helm reserves use of the charts/, crds/, and templates/ directories, and of the listed file names. Other files will be left as they are.

Helm Chart File Structure

The Helm Chart describes the contents of the package such as the type of chart, dependencies

```yaml
annotations:
  category: CMS
apiVersion: v2
appVersion: 5.8.3
dependencies:
  - condition: mariadb.enabled
    name: mariadb
    repository: https://charts.bitnami.com/bitnami
    version: 9.x.x
  - condition: memcached.enabled
    name: memcached
    repository: https://charts.bitnami.com/bitnami
    version: 5.x.x
  - name: common
    repository: https://charts.bitnami.com/bitnami
    tags:
      - bitnami-common
    version: 1.x.x
description: Web publishing platform for building blogs and websites.
engine: gotpl
home: https://github.com/bitnami/charts/tree/master/bitnami/wordpress
icon: https://bitnami.com/assets/stacks/wordpress/img/wordpress-stack-220x234.png
keywords:
  - blog
  - cms
  - php
  - wordpress
maintainers:
  - email: containers@bitnami.com
    name: Bitnami
name: wordpress
sources:
  - https://github.com/bitnami/bitnami-docker-wordpress
  - https://wordpress.org/
version: 12.3.3
```

Helm Packaging and Installing

To package a chart directory in a versioned chart archive the “helm package” command is used.

```
helm package --sign ./mychart --key mykey --keyring ~/.gnupg/secring.gpg
```

Versioned chart archives are used by Helm package repositories

Artifact Hub is where we can find published Helm charts

https://artifacthub.io/

```
helm repo add nicholaswilde https://nicholaswilde.github.io/helm-charts/
helm repo update
helm install postgres nicholaswilde/postgres
```

Installing a helm chart

Kustomize

Kustomize provides more flexibility when writing Kubernetes configuration files by allowing you to overlay (override) to “patch” configurations.

Kustomize is built into KubeCTL

```
├── base
│   ├── kustomization.yaml
│   ├── deployment.yaml
│   └── service.yaml
└── patches
    ├── dev
    │   ├── kustomization.yaml
    │   ├── deployment.yaml
    │   └── namespace.yaml
    └── prod
        ├── kustomization.yaml
        ├── deployment.yaml
        └── namespace.yaml
```

Kustomization file defined what will be overwritten in the base components.

What is Infrastructure as Code?

The Problem with Manual Configuration

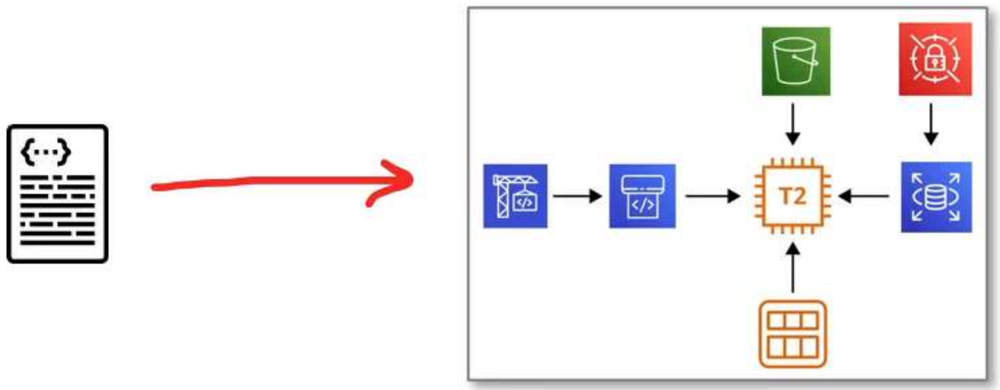
Manually configuring your cloud infrastructure allows you easily start using new service offerings to quickly prototype architectures however it comes with many downsides:

- Its easy to mis-configure a service though human error
- Its hard to manage the expected state of configuration for compliance
- Its hard to transfer configuration knowledge to other team members

Infrastructure as Code (IaC)

You write a configuration script to **automate** **creating, updating or destroying** cloud infrastructure.

- IaC is a **blueprint** of your infrastructure.
- IaC allows you to easily **share, version or inventory** your cloud infrastructure.



Popular Infrastructure as Code tools (IaC)

Declarative

- What you see is what you get. ***Explicit***
- More verbose, but zero chance of mis-configuration
- Uses scripting languages eg. JSON, YAML, XML

ARM Templates
Supports only Azure

Azure Blueprints
Supports only Azure
Manages relationship between services

CloudFormation
Only for AWS

Cloud Deployment Manager
Supports on Google Cloud

Terraform
Supports many cloud service providers (CSPs) and cloud services.

Imperative

- You say what you want, and the rest is filled in. ***Implict***
- Less verbose, you could end up with misconfiguration
- Does more than Declarative
- Uses programming languages eg. Python, Ruby, JavaScript

AWS Cloud Development Kit (CDK)
Supports only AWS
Many built-in templates for opinionated best practices

Pulumi
Supports AWS, Azure, GCP, K8

IaC for Kubernetes

Managing Infrastructure **of the Cluster**

Terraform (or any IaC) tool can be used to provision the cluster and managed services to be used alongside the cluster eg. Manage Database.

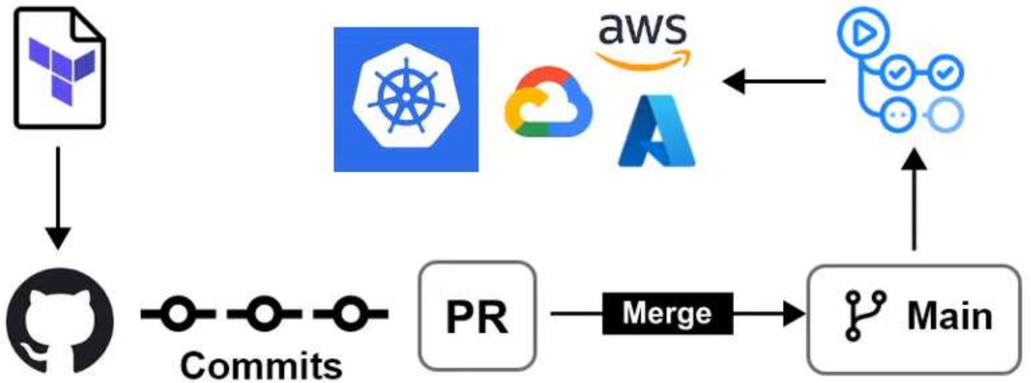
Terraform can technically manage cluster components via their Manifest module
So you can *benefit from the state management provided by Terraform.
Kubernetes already manages state with the Controller Manager and etcd

Managing Infrastructure **in the Cluster**

For managing (application) infrastructure within the cluster eg. Pods, Services, Ingres
It is recommended to package as Helm charts and use that in your CI/CD.

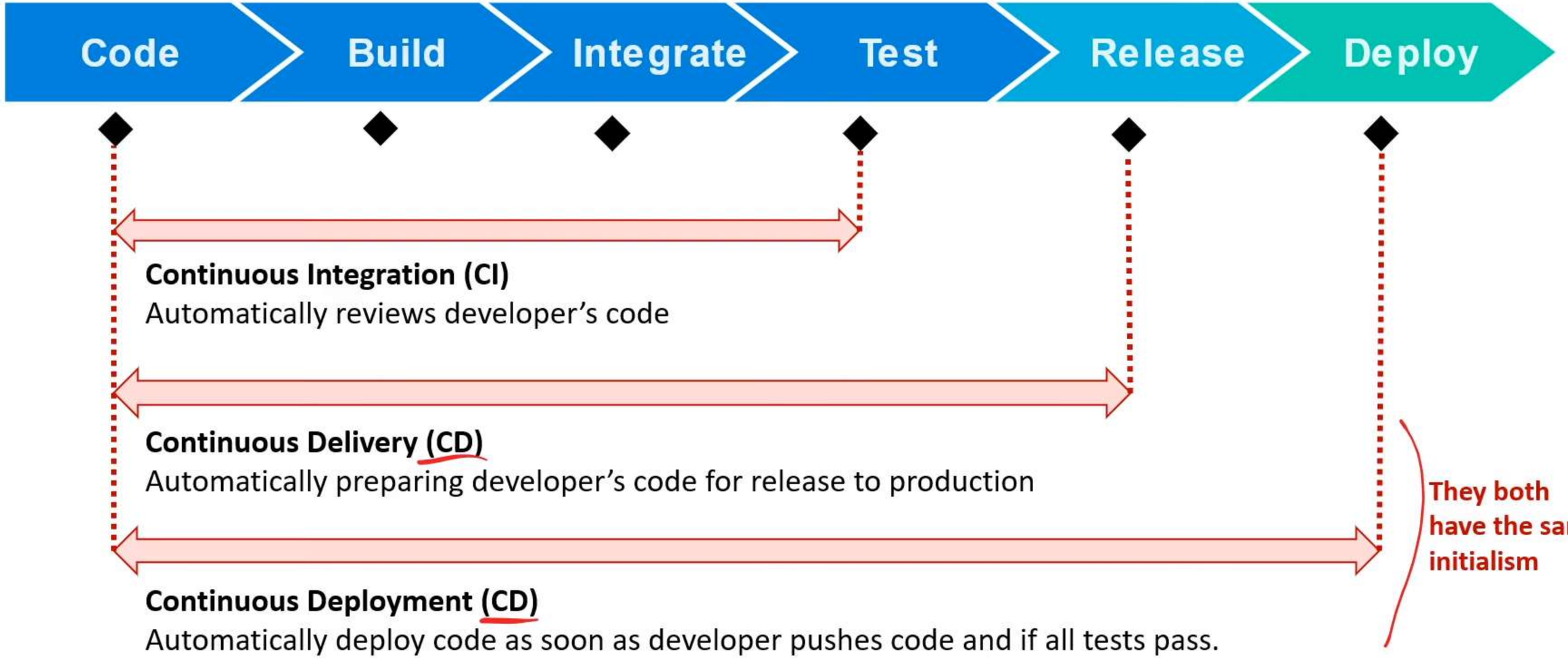
What is GitOps?

GitOps is when you take Infrastructure as Code (IaC) and you use a git repository to **introduce a formal process to review and accept changes to infrastructure code**, once that code is accepted, it automatically triggers a deploy



CI/CD Models

Production (prod) is the live server where real paying users are using the platform
Staging is a private server developers do a final manual tests as a customer (QA) before deploying to prod



Argo vs Flux

CNCF have two CI/CD projects that serve the same purpose but take a different approach.

- Originally developed by Weaveworks
- CLI-first approach
- ExperimentalWeb UI as a plugin
- Supports Role-based access controls (RBAC)
- Supports multitenancy (Flux 2)
- Supports Helm and Kustomization
- Automation of Container Updates

- Both a CLI and Web UI
- Supports Role-based access controls (RBAC)
- Supports Single Sign On (SSO)
- Supports multitenancy
- Helm, Kustomization, ksonnet and jsonnet
- Manual commit and sync to update containers

Generally Flux is simpler in design and CLI focused

Jenkins and Jenkins X and CloudBees

Jenkins is an open-source popular and mature CI/CD tool **for any kind of workload**.
Jenkins can be used to deploy applications on to Kubernetes.
Jenkins is written in Java, and have many plugins for any use case.

Jenkins X is an open-source CI/CD tool for modern cloud applications on Kubernetes.
Compared to Jenkins its suppose to be much easier to use.

Jenkins X may replace or merge with Jenkins one day to only have a single CI/CD tool for all use cases.

CloudBees is the commercial distribution of Jenkins and Jenkins X for large and compliance first organizations
CloudBees acquired InfraDNA, InfraDNA organization created Jenkins.

CircleCI

CircleCI is a propriety fully managed CI/CD service to make deployments easy and seamless
CircleCI can support deploying applications to Kubernetes