Red Hat Open Tour 2022

Red Hat

intel®

Gain robust repeatability as self service, by automating the automation

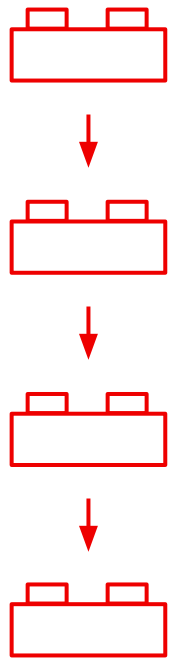
Johan Odell

Solution Architect @ Red Hat

In this presentation i am going to talk about

- Standardisation
- Automation
- Collaboration

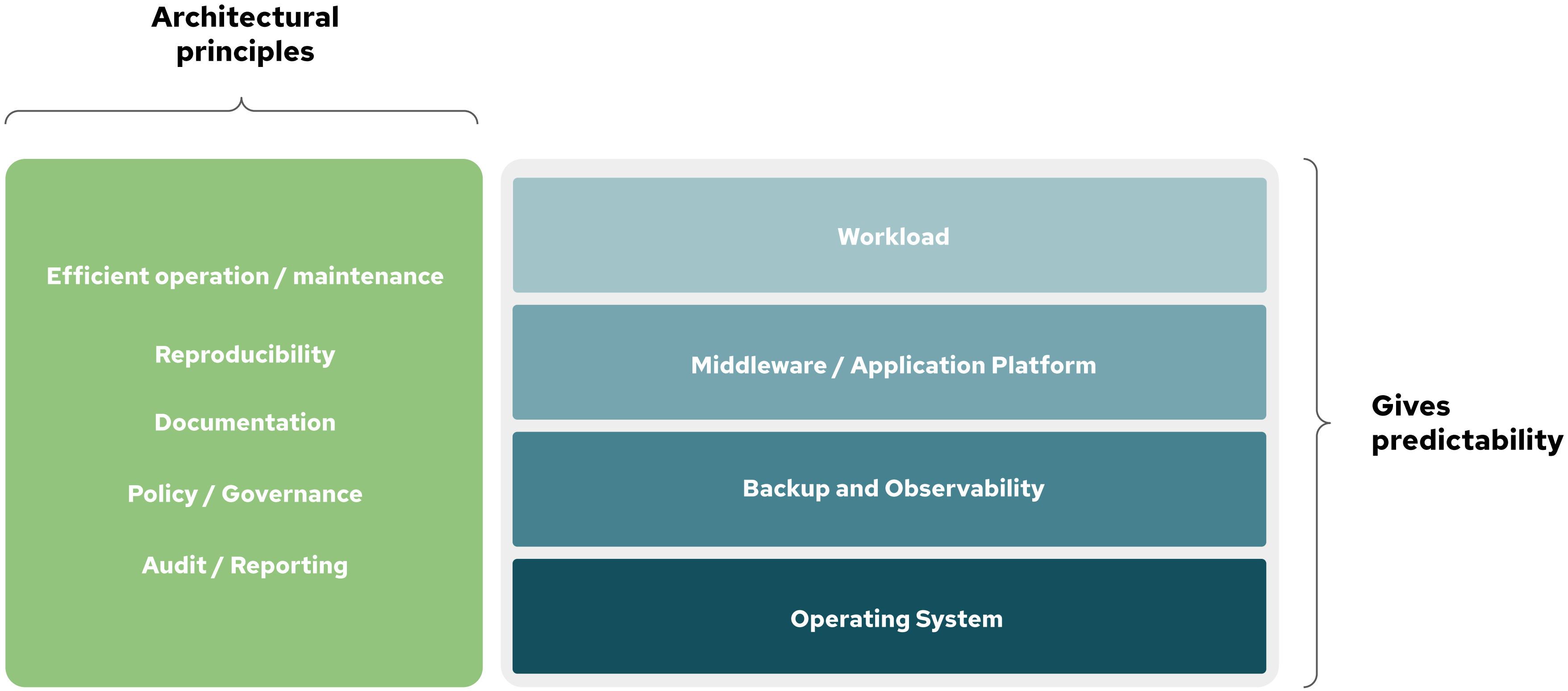
It all starts with standardisation



Picture credit to Association for Advancing Automation (A3)

Standardisation in IT

Example Viewpoint



Standard Operating Environment (SOE)

Definition:

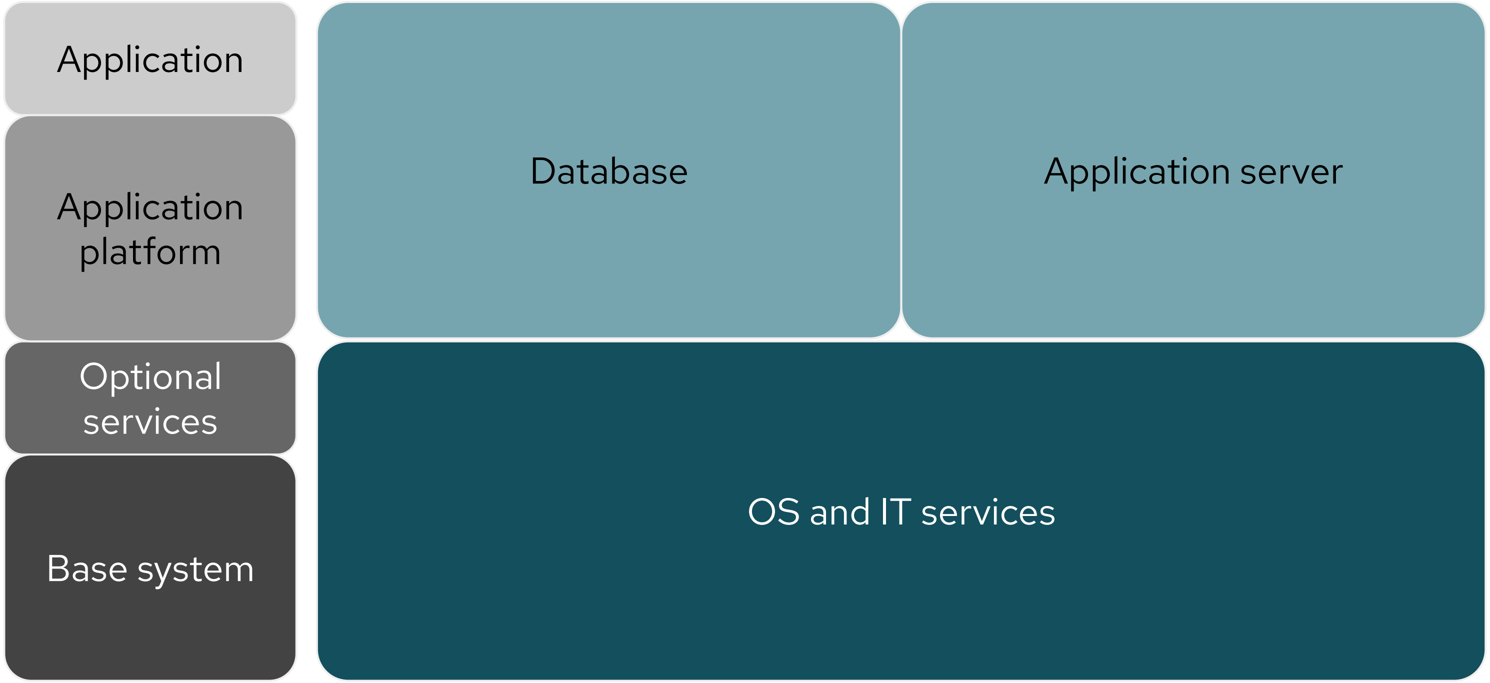
“Provides tools, standards and best practices to manage the lifecycle of an entire, deployed stack – from operating system and infrastructure services through to middleware and applications.”

What areas does it focus on?

- Automation
- Standardisation
- Lifecycle management
- Reporting

Standard Operating Environment (SOE)

One-size-fits-most, generic servers with functional application blocks



Application

Application platform

Optional services

Base system

Database

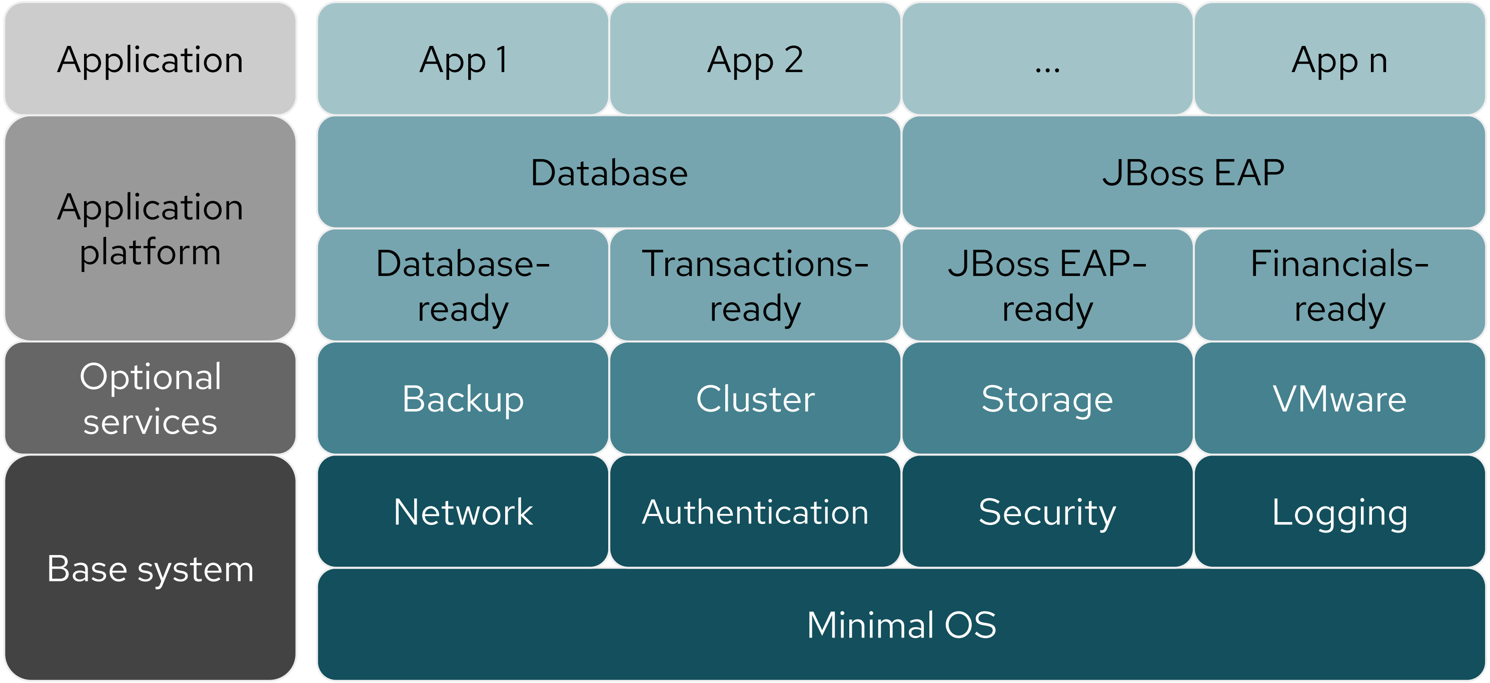
Application server

OS and IT services

Basic approach

Standard Operating Environment (SOE)

Concept: Independent yet compatible and interchangeable components



Application

App 1

App 2

...

App n

Application platform

Database

JBoss EAP

Database-ready

Transactions-ready

JBoss EAP-ready

Financials-ready

Optional services

Backup

Cluster

Storage

VMware

Base system

Network

Authentication

Security

Logging

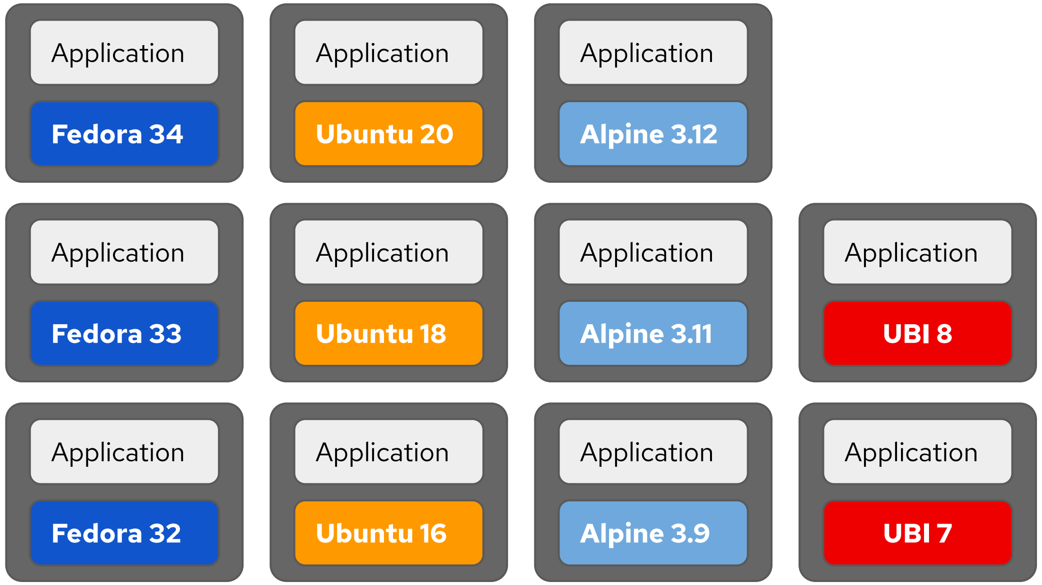
Minimal OS

Adaptive SOE approach

With no standard

Everyone will make their own choice

No Standard Operating Environment



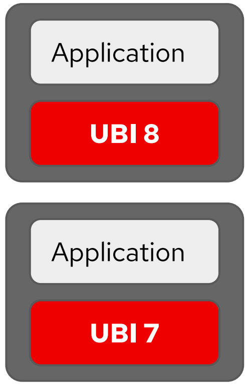
Application / Fedora 34 — Application / Ubuntu 20 — Application / Alpine 3.12

Application / Fedora 33 — Application / Ubuntu 18 — Application / Alpine 3.11 — Application / UBI 8

Application / Fedora 32 — Application / Ubuntu 16 — Application / Alpine 3.9 — Application / UBI 7

- 8 different versions of glibc
- 3 different versions of muslc
- 11 different versions of OpenSSL

Standard based on UBI 7 & UBI 8



Application / UBI 8

Application / UBI 7

- 2 different versions of glibc
- 2 different versions of OpenSSL

Efficiency Through Automation

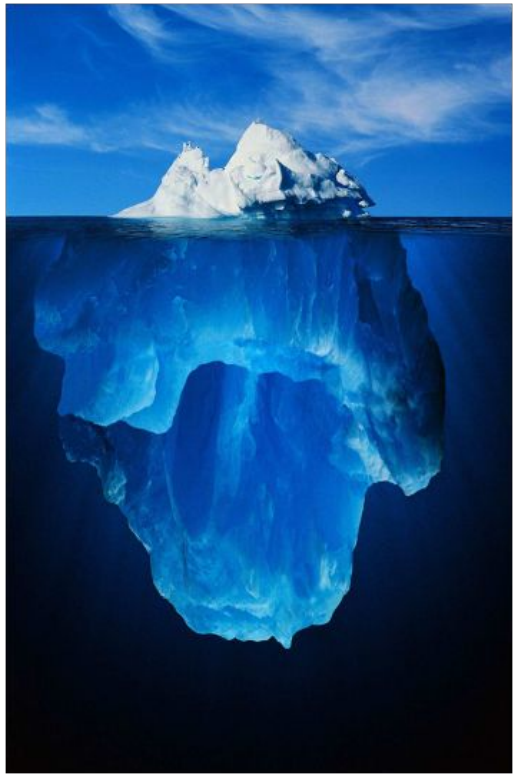
Ok, standards are great, but:

- only define point-in-time snapshots of the environment
- take time to maintain in a complex environment
- By automating the process of implementing standards we achieve:
- higher flexibility to accommodate change ⇒ higher agility
- higher operational efficiency
- eases lifecycle management

What kinds of automation?

IT Automation

your stack



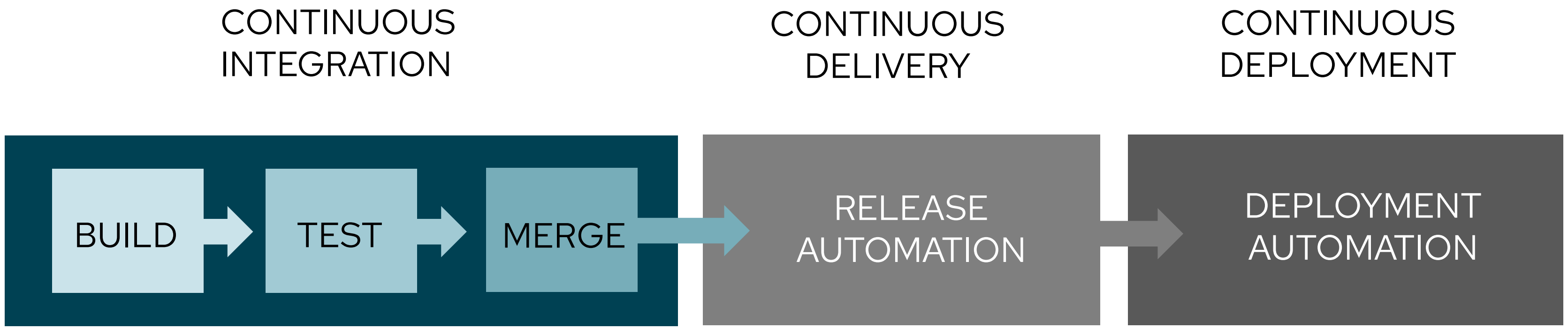
Business value

Lots of tech

use cases

- FULLY AUTOMATED PROVISIONING
- SECURITY/ COMPLIANCE
- CONFIG MANAGEMENT
- CONTINUOUS DELIVERY
- ORCHESTRATION
- APP DEPLOYMENT

Application Development and Deployment



CONTINUOUS INTEGRATION

CONTINUOUS DELIVERY

CONTINUOUS DEPLOYMENT

BUILD → TEST → MERGE → RELEASE AUTOMATION → DEPLOYMENT AUTOMATION

Red Hat OpenShift

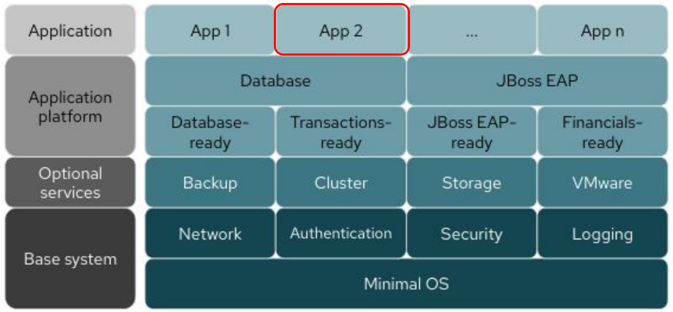
Red Hat Ansible Automation Platform

Git* & Ansible Automation Platform Application upgrade via CI/CD

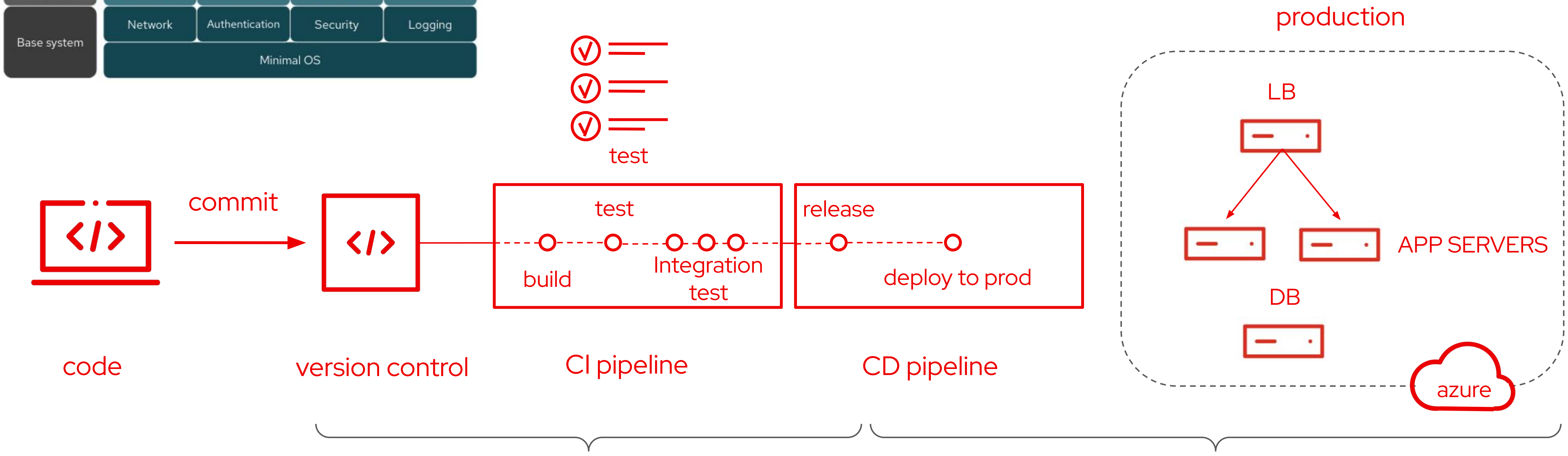
In this demo you will see

Seamless upgrade of application

- Commit application change to git
- Triggers pipeline run with tests
- Deploy change to production
- Remove 1st appserver from load balancer
- Update 1st appserver and enable in load balancer
- Remove 2nd appserver from load balancer
- Update 2nd appserver and enable in load balancer.



Application	App 1	App 2	...	App n
Application platform	Database		JBoss EAP	
	Database-ready	Transactions-ready	JBoss EAP-ready	Financials-ready
Optional services	Backup	Cluster	Storage	VMware
Base system	Network	Authentication	Security	Logging
	Minimal OS			



GitLab

Red Hat
Ansible Automation
Platform

DEMO

Red Hat

What did we see in the demo

Seamless upgrade of application

- ✓ No manual steps
- ✓ No human errors
- ✓ Predictable outcomes
- ✓ Higher efficiency
- ✓ Faster time to market
- ✓ Less stress

Continuous Integration

Simplified collaborative application development

- Self service application platform
- Build and test your application automatically
- Standardised native tools
- Everything as code
  - Application
  - Deployment
  - Build and test
- Collaborative workflow

OpenShift Pipelines

Open source, standardised, cloud-native

based on TEKTON

Why OpenShift Pipelines?

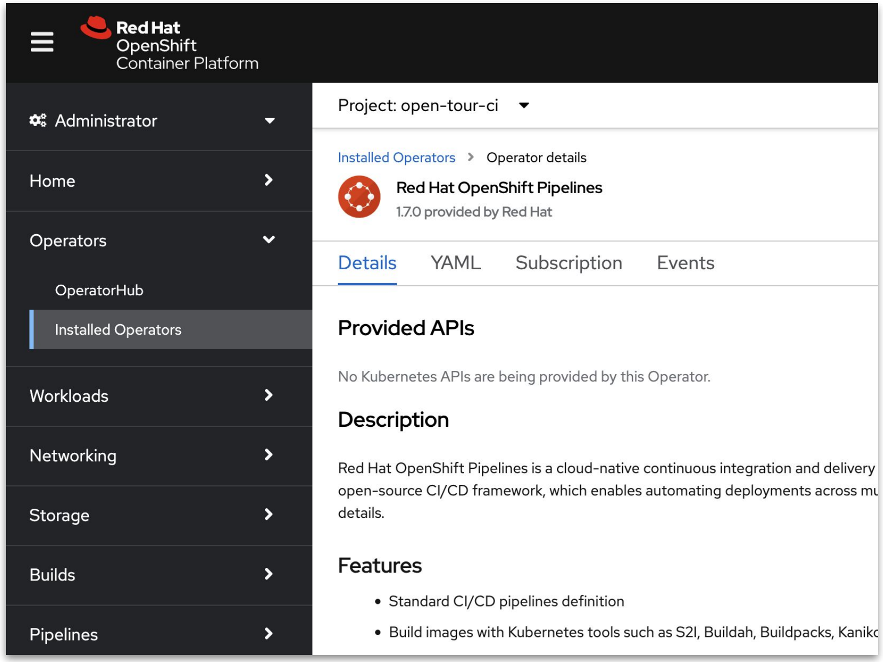
Built for Cloud-native

Scale on-demand

Secure pipeline execution

Flexible and powerful

Pipelines as a service



Red Hat OpenShift Pipelines

1.7.0 provided by Red Hat

Provided APIs

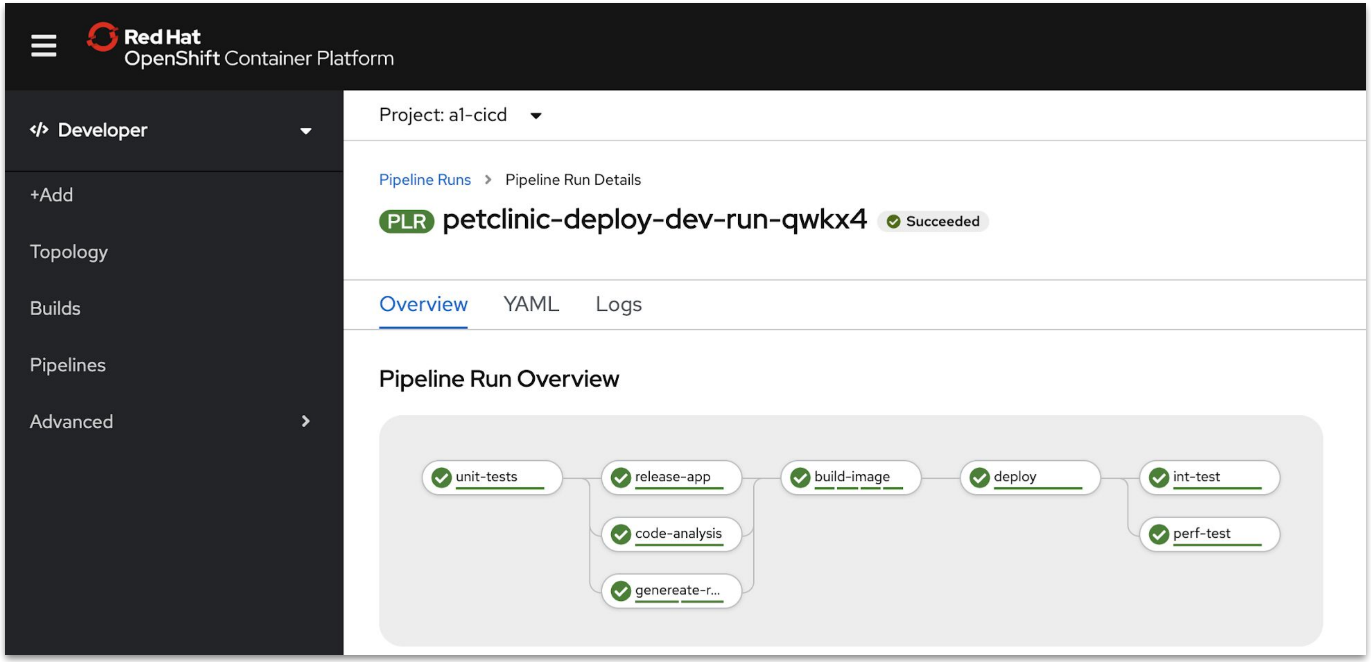
No Kubernetes APIs are being provided by this Operator.

Description

Red Hat OpenShift Pipelines is a cloud-native continuous integration and delivery open-source CI/CD framework, which enables automating deployments across mu details.

Features

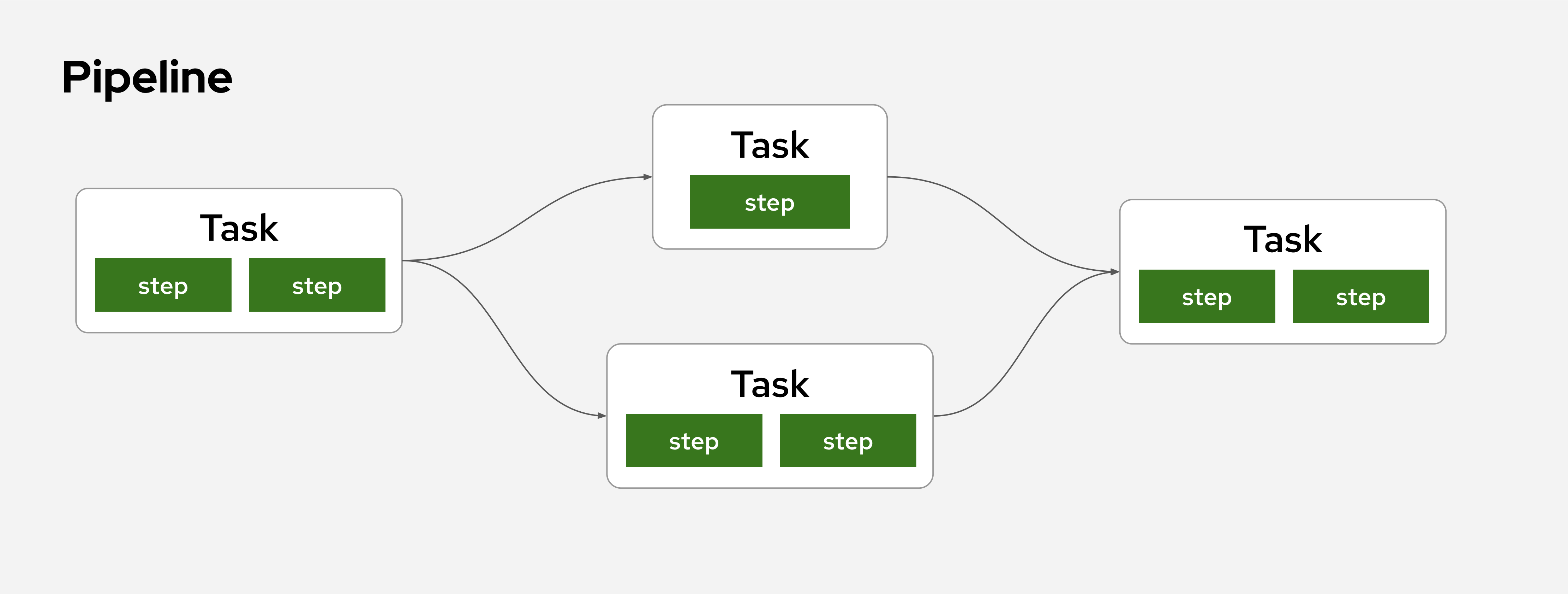
- Standard CI/CD pipelines definition
- Build images with Kubernetes tools such as S2I, Buildah, Buildpacks, Kanikc



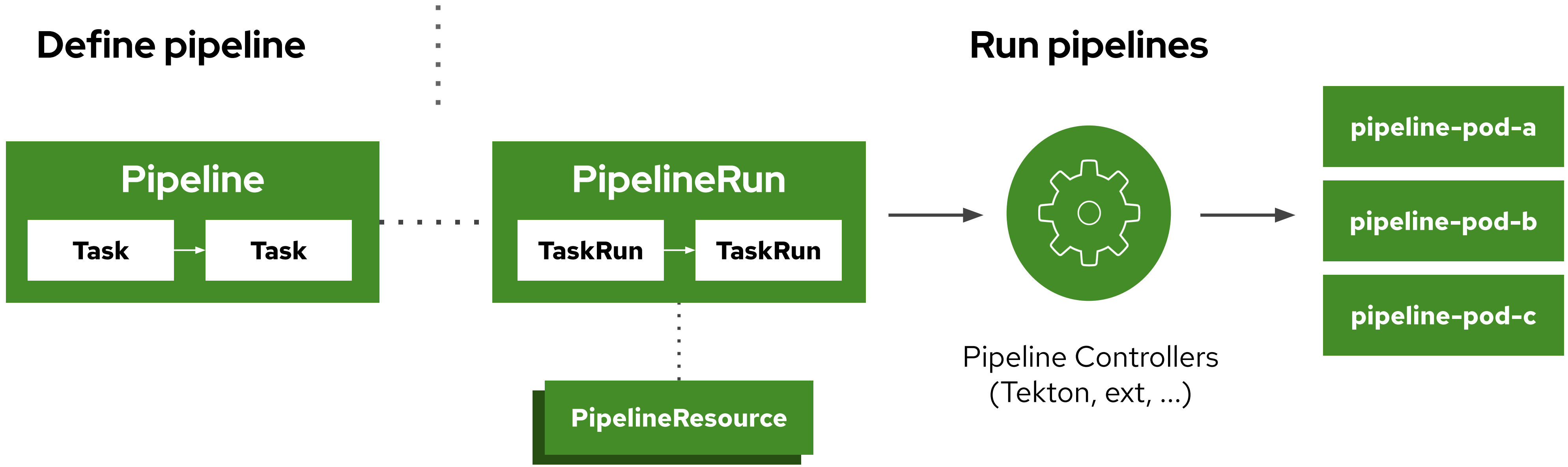
PLR petclinic-deploy-dev-run-qwkx4 ✓ Succeeded

Pipeline Run Overview

OpenShift Pipelines - Tekton concepts



OpenShift Pipelines - Architecture



Define pipeline

Run pipelines

Pipeline

Task

Task

PipelineRun

TaskRun

TaskRun

PipelineResource

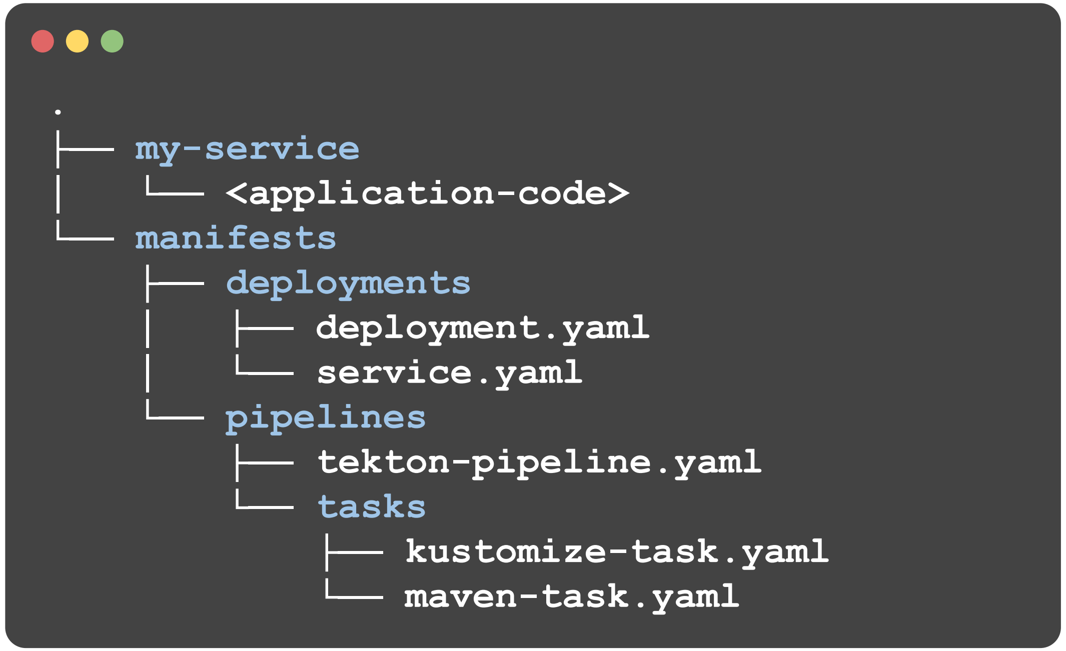
Pipeline Controllers
(Tekton, ext, ...)

pipeline-pod-a

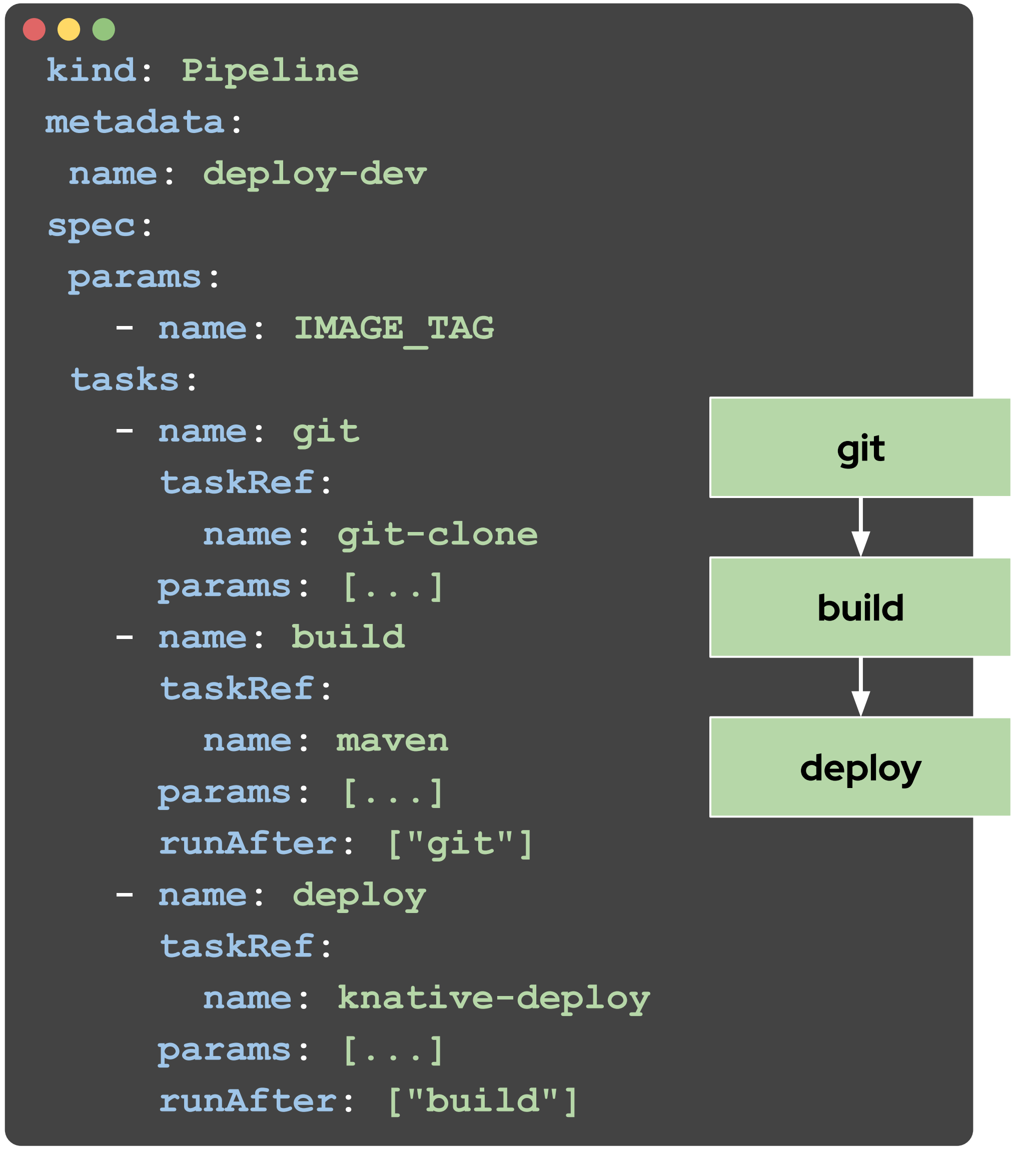
pipeline-pod-b

pipeline-pod-c

Everything as code



```
.
├── my-service
│   └── <application-code>
└── manifests
    ├── deployments
    │   ├── deployment.yaml
    │   └── service.yaml
    └── pipelines
        ├── tekton-pipeline.yaml
        └── tasks
            ├── kustomize-task.yaml
            └── maven-task.yaml
```



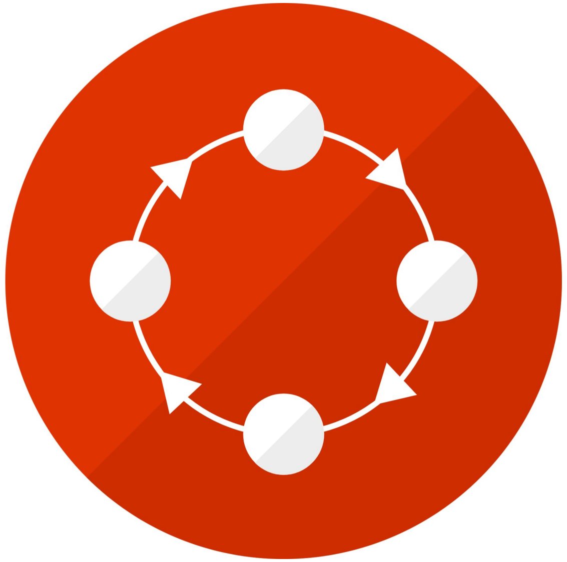
```
kind: Pipeline
metadata:
 name: deploy-dev
spec:
 params:
   - name: IMAGE_TAG
 tasks:
   - name: git
     taskRef:
       name: git-clone
     params: [...]
   - name: build
     taskRef:
       name: maven
     params: [...]
     runAfter: ["git"]
   - name: deploy
     taskRef:
       name: knative-deploy
     params: [...]
     runAfter: ["build"]
```

git

build

deploy

Pipelines as code



- GitOps enabled - git-centric workflow
- Integrated with Git provider
  - Events, actions
- Pipelines run in cluster
  - No pre-configured infrastructure

Automate the automation

Standardise your CI - cloud-native style

1. Pipelines as a (cluster) service
2. CI resources - cloud-native
3. Git-centric workflow

DEMO

What did we see in the demo

- ✓ No manual steps
- ✓ No human errors
- ✓ Predictable outcomes
- ✓ Higher efficiency
- ✓ Faster time to market
- ✓ Less stress

Simplified collaborative application development

Continuous Delivery

Hybrid cloud pattern: Multicloud GitOps

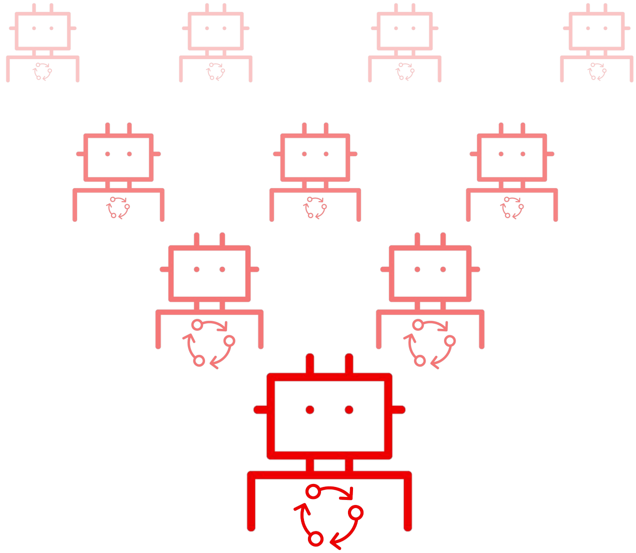
Automated business continuity

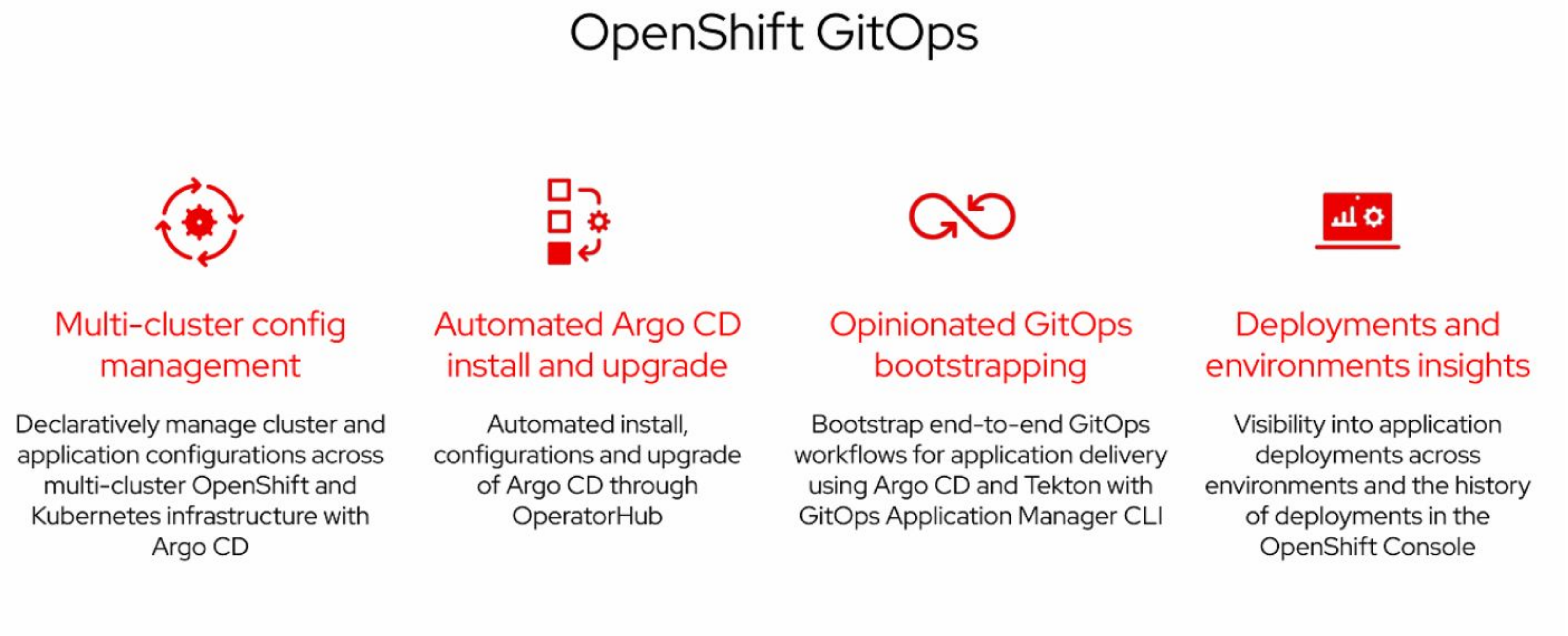
- Keep delivering no matter of location
- Automate introduction of new features
- Manage risks by replication and scaling out environments
- Everything automated

PURPOSE

We want everything as code.

Applications, configurations and secrets delivered to autonomous environments. Visible change history. Comes with self healing.





OpenShift GitOps

Multi-cluster config management

Declaratively manage cluster and application configurations across multi-cluster OpenShift and Kubernetes infrastructure with Argo CD

Automated Argo CD install and upgrade

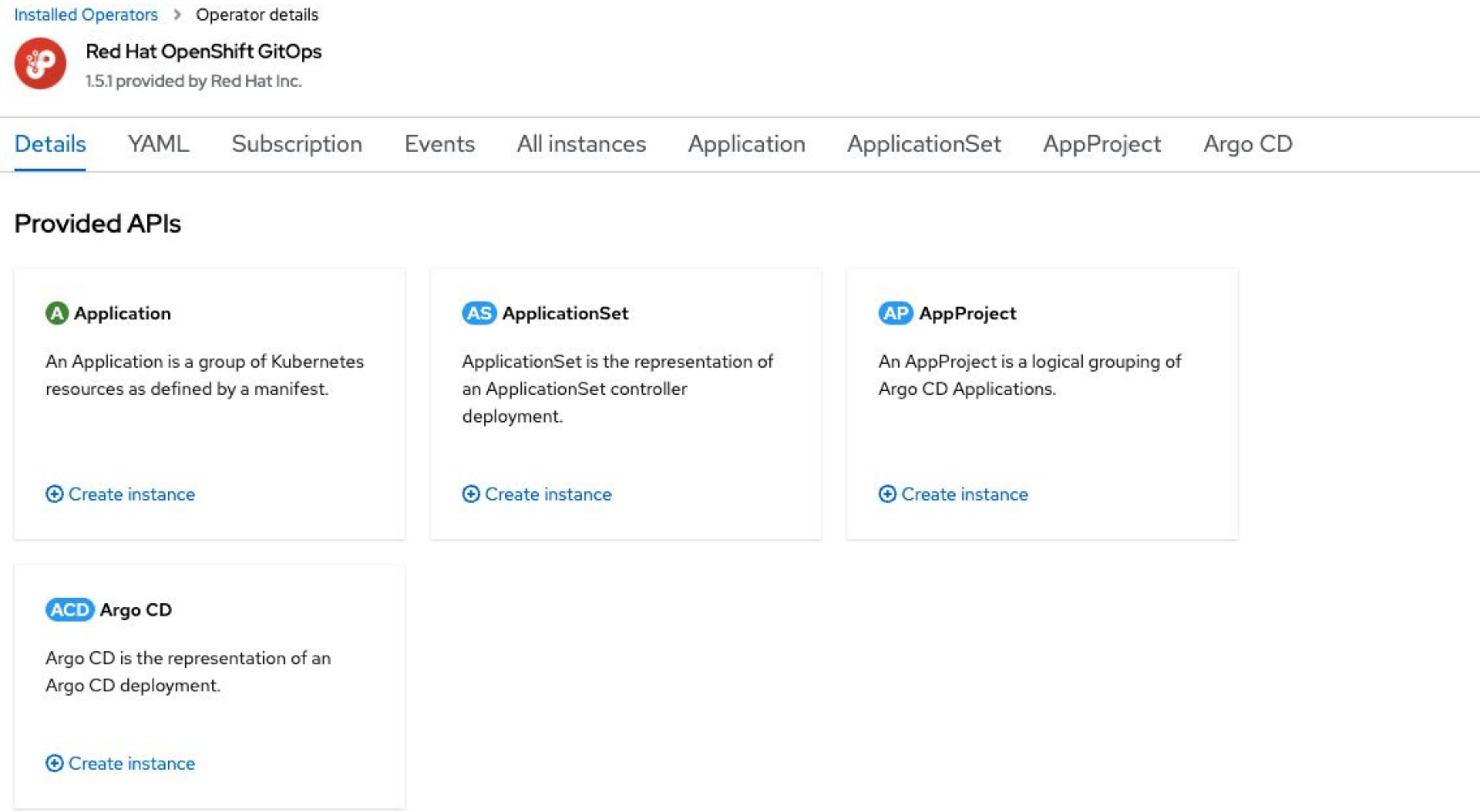
Automated install, configurations and upgrade of Argo CD through OperatorHub

Opinionated GitOps bootstrapping

Bootstrap end-to-end GitOps workflows for application delivery using Argo CD and Tekton with GitOps Application Manager CLI

Deployments and environments insights

Visibility into application deployments across environments and the history of deployments in the OpenShift Console



Red Hat OpenShift GitOps

1.5.1 provided by Red Hat Inc.

Provided APIs

A Application

An Application is a group of Kubernetes resources as defined by a manifest.

⊕ Create instance

AS ApplicationSet

ApplicationSet is the representation of an ApplicationSet controller deployment.

⊕ Create instance

AP AppProject

An AppProject is a logical grouping of Argo CD Applications.

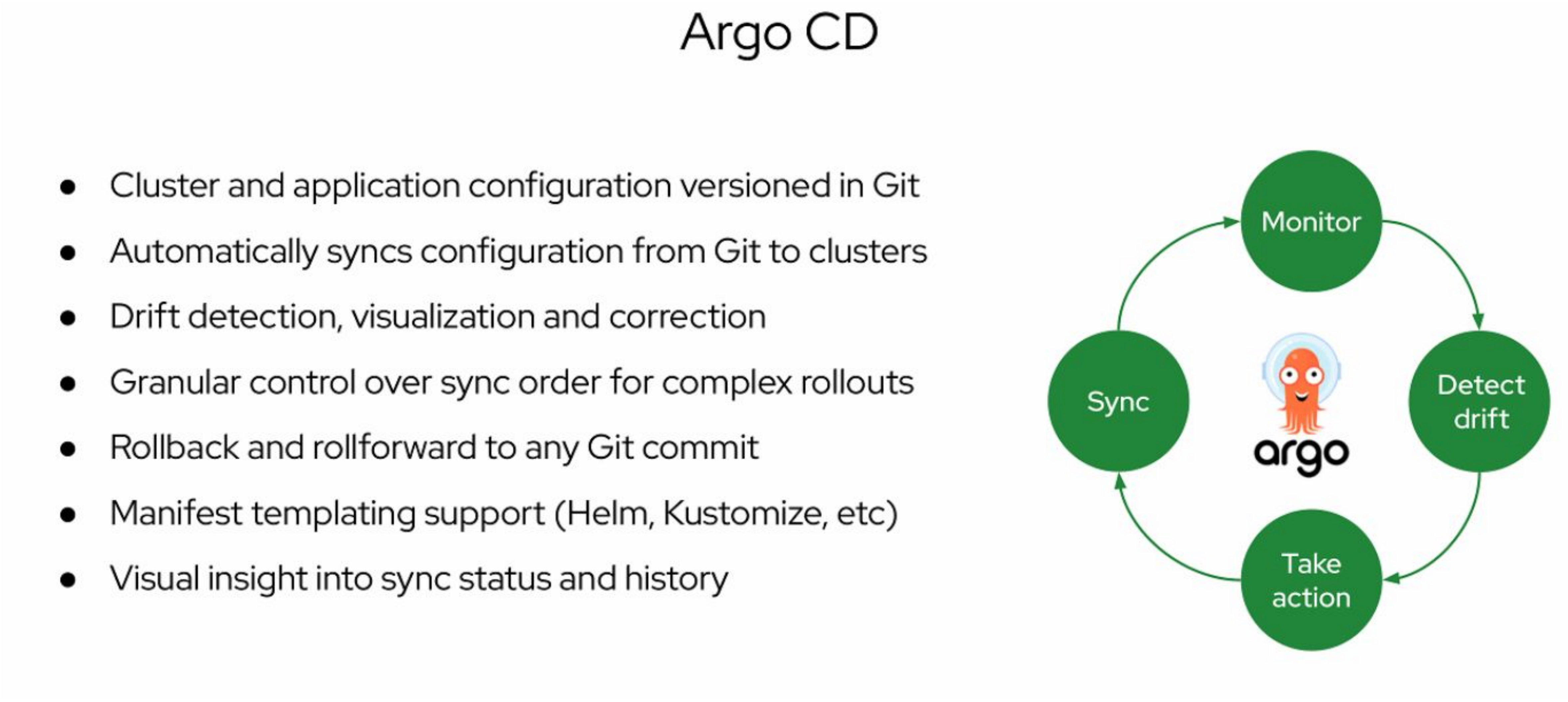
⊕ Create instance

ACD Argo CD

Argo CD is the representation of an Argo CD deployment.

⊕ Create instance

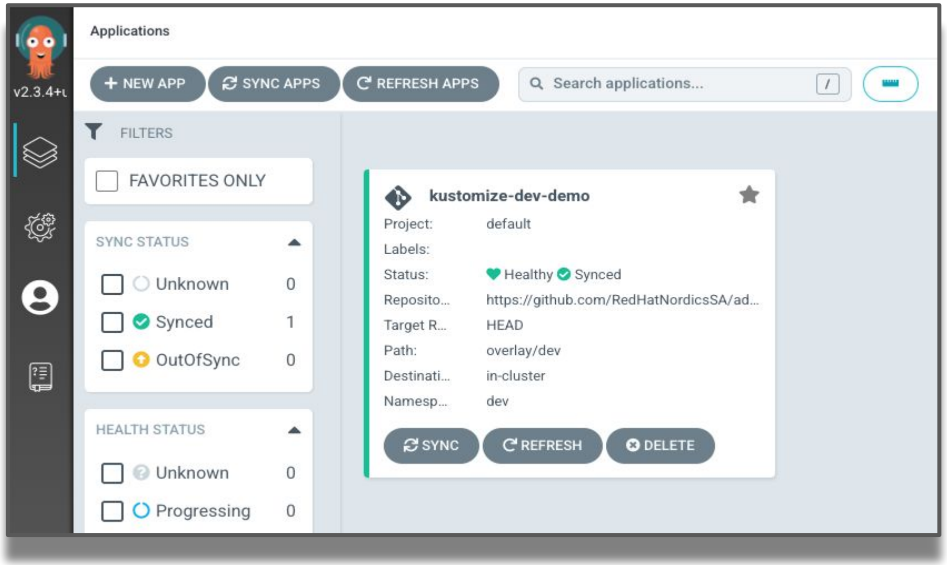
Operator based



Argo CD

- Cluster and application configuration versioned in Git
- Automatically syncs configuration from Git to clusters
- Drift detection, visualization and correction
- Granular control over sync order for complex rollouts
- Rollback and rollforward to any Git commit
- Manifest templating support (Helm, Kustomize, etc)
- Visual insight into sync status and history

Red Hat



□ FAVORITES ONLY

SYNC STATUS

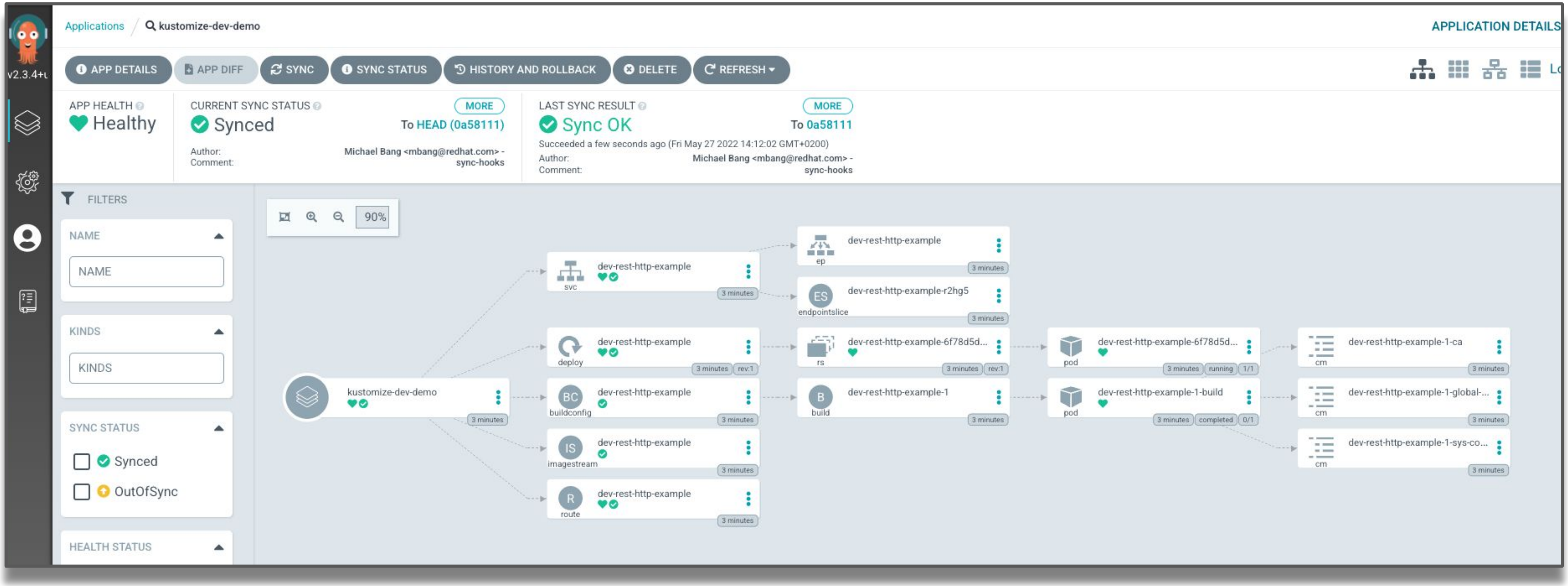
□ Unknown	0
□ Synced	1
□ OutOfSync	0

HEALTH STATUS

□ Unknown	0
□ Progressing	0

kustomize-dev-demo

Project:	default
Labels:	
Status:	Healthy ✓ Synced
Reposito...	https://github.com/RedHatNordicsSA/ad...
Target R...	HEAD
Path:	overlay/dev
Destinati...	in-cluster
Namesp...	dev



APP HEALTH
Healthy

CURRENT SYNC STATUS
Synced
To HEAD (0a58111)
Author: Michael Bang <mbang@redhat.com> -
Comment: sync-hooks

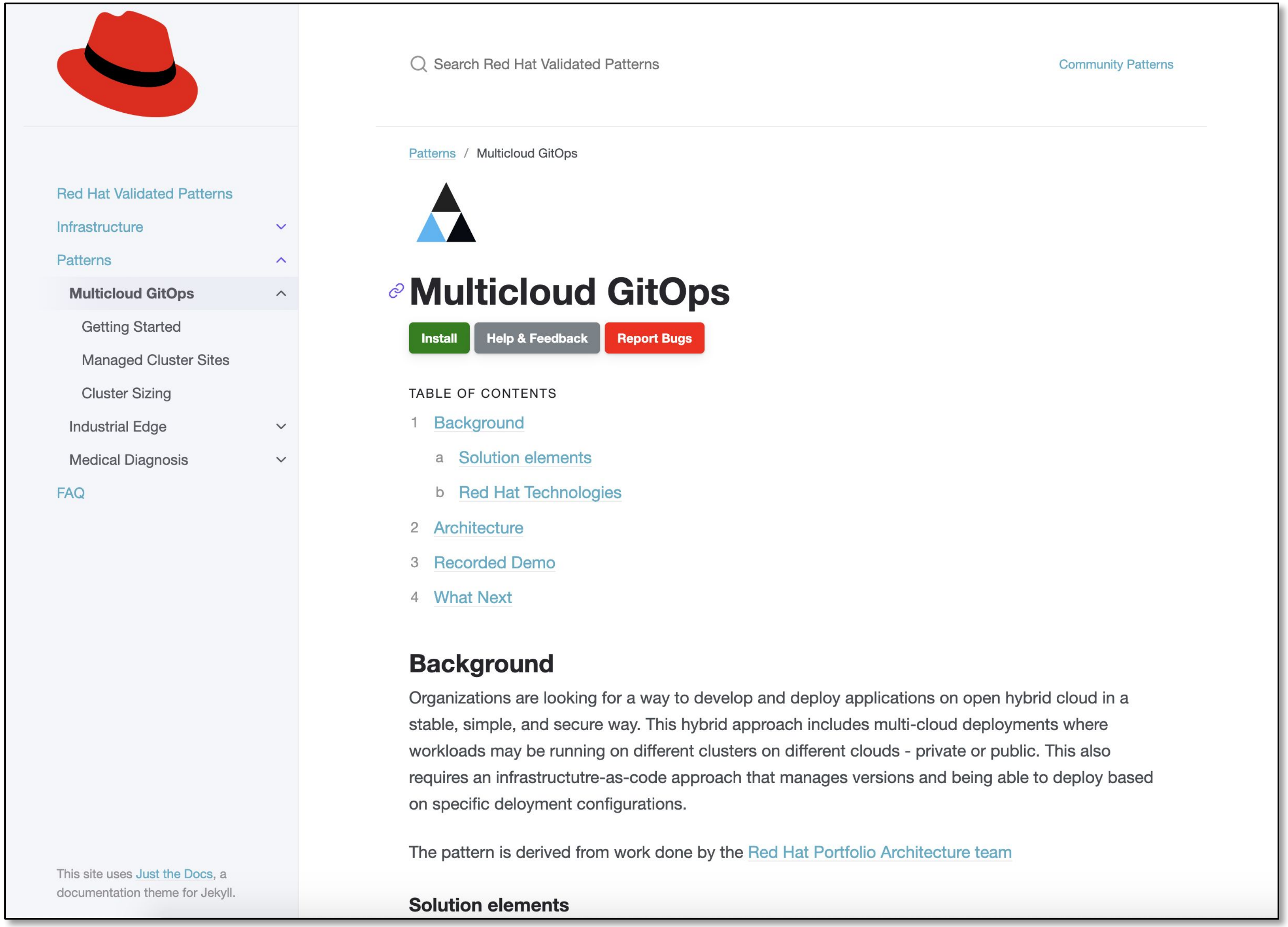
LAST SYNC RESULT
Sync OK
To 0a58111
Succeeded a few seconds ago (Fri May 27 2022 14:12:02 GMT+0200)
Author: Michael Bang <mbang@redhat.com> -
Comment: sync-hooks

SYNC STATUS

□ Synced

□ OutOfSync

TAKE IDEAS FROM HYBRID CLOUD PATTERNS



Search Red Hat Validated Patterns

Community Patterns

Red Hat Validated Patterns
Infrastructure
Patterns
Multicloud GitOps
Getting Started
Managed Cluster Sites
Cluster Sizing
Industrial Edge
Medical Diagnosis
FAQ

This site uses Just the Docs, a documentation theme for Jekyll.

Patterns / Multicloud GitOps

Multicloud GitOps

Install | Help & Feedback | Report Bugs

TABLE OF CONTENTS

Background

Organizations are looking for a way to develop and deploy applications on open hybrid cloud in a stable, simple, and secure way. This hybrid approach includes multi-cloud deployments where workloads may be running on different clusters on different clouds - private or public. This also requires an infrastructutre-as-code approach that manages versions and being able to deploy based on specific deloyment configurations.

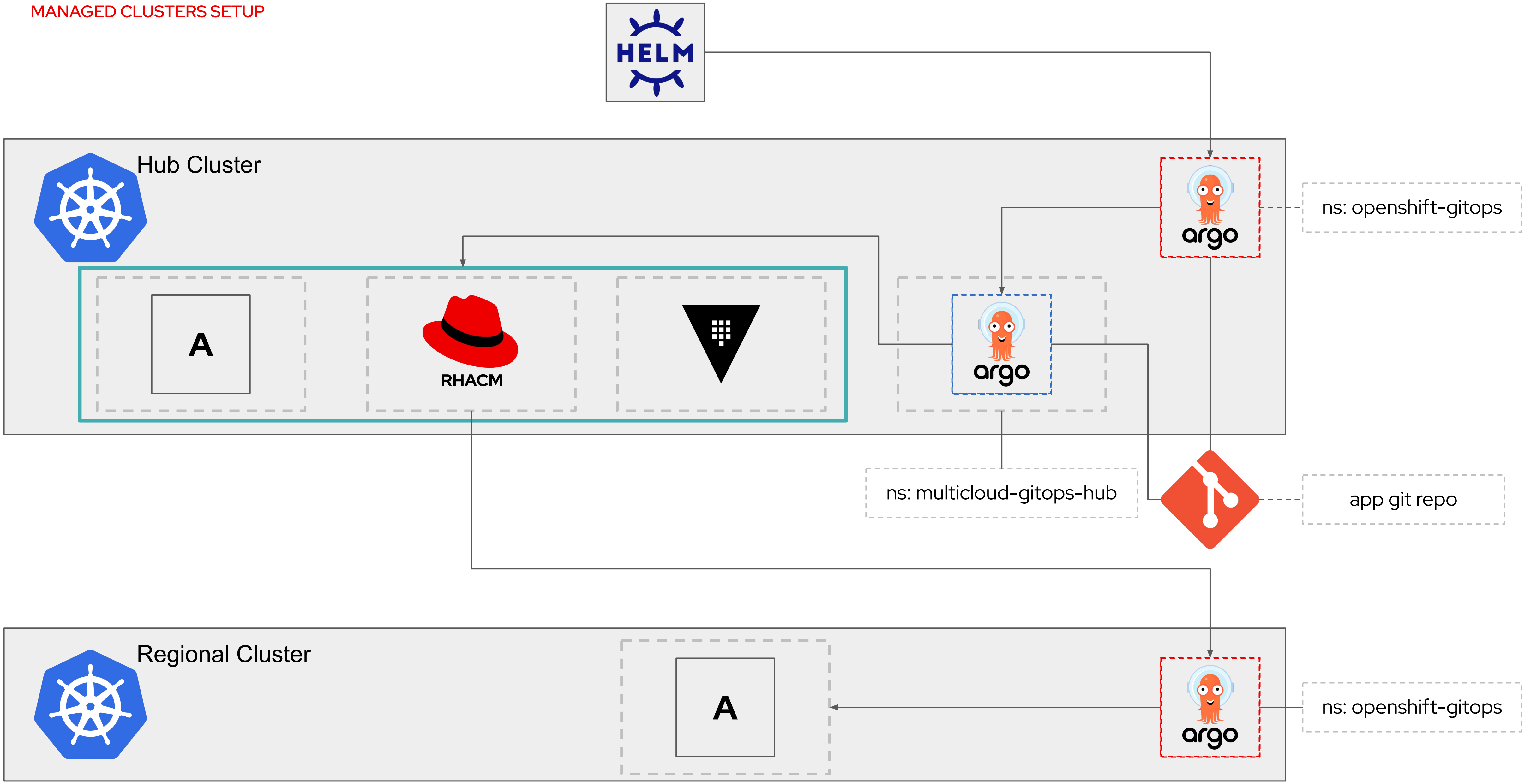
The pattern is derived from work done by the Red Hat Portfolio Architecture team

Solution elements

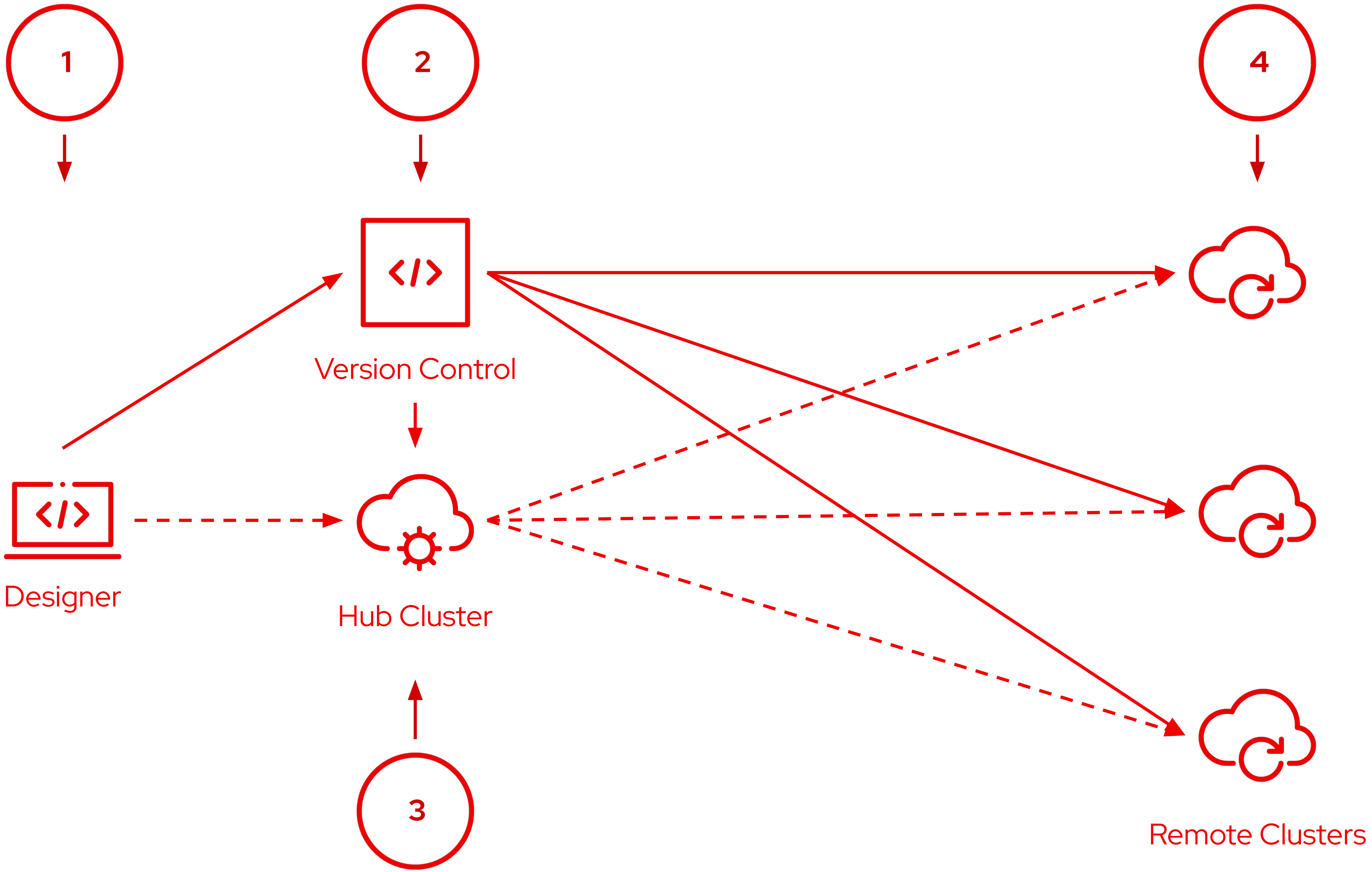
Let’s look at the multicloud gitops pattern

https://redhat-gitops-patterns.io/
https://hybrid-cloud-patterns.io/multicloud-gitops/

MANAGED CLUSTERS SETUP



Sugar & Sparrow



One repository to control delivery versions

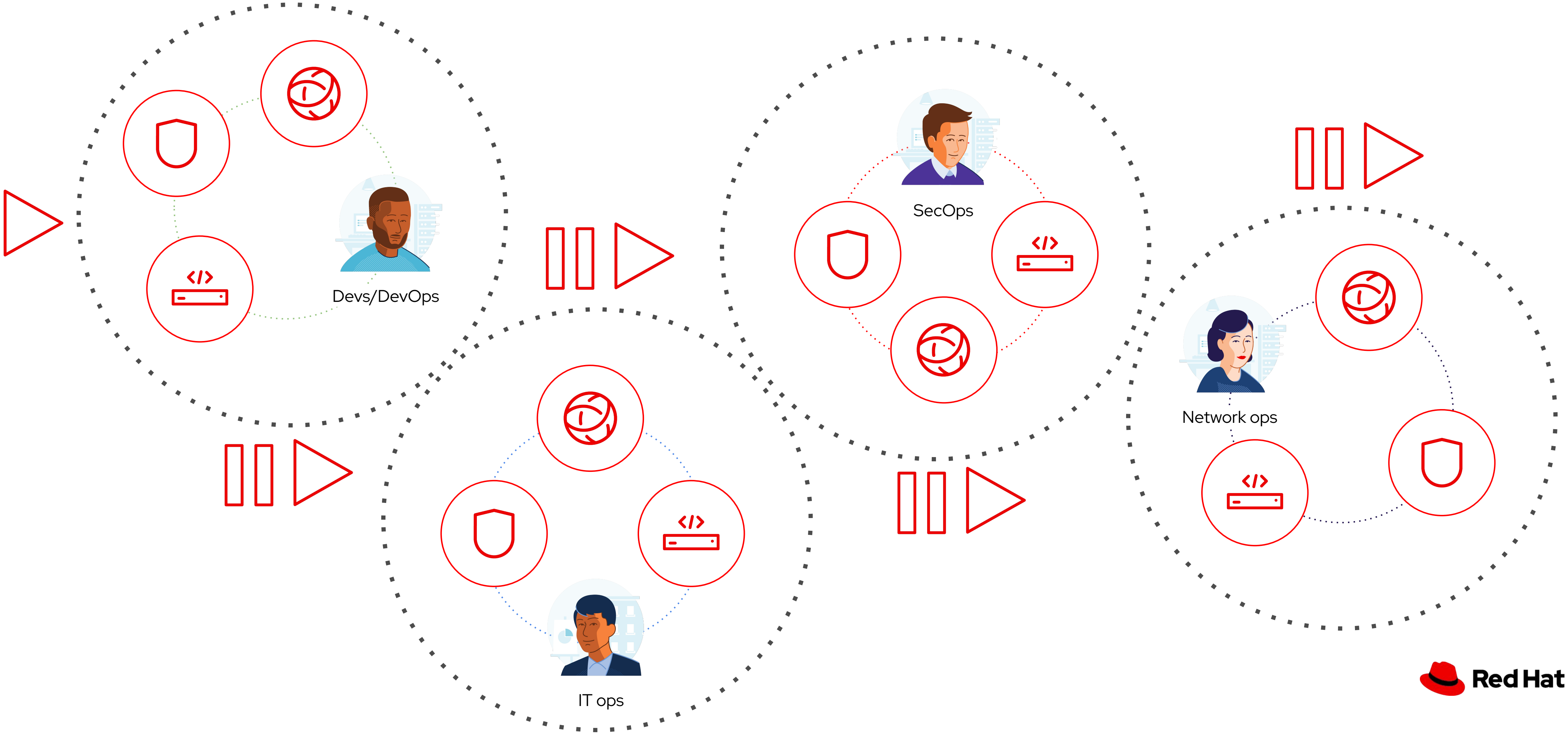
Several environments in hybrid clouds to automatically adapt to configuration or application changes.

1. Create
2. Commit
3. Automate - - - -
4. Keep Delivering ——

Provide business value through collaboration

But many organizations have a common problem...

Too many unintegrated, domain-specific tools, limited collaboration and scale



In this presentation you learned about

- Standardisation
- Automation
- Collaboration

To gain robust repeatability as self service, by automating the automation

Red Hat Services get you going!