

# Red Hat Open Tour 2022



---

intel<sup>®</sup>

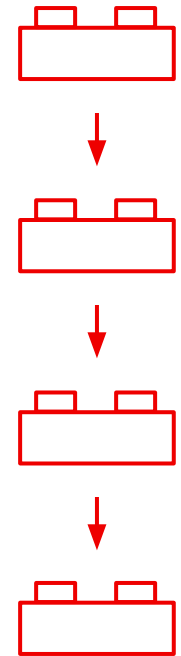
# Gain robust repeatability as self service, by automating the automation

Speaker:  
Solution Architect

In this presentation i am going to talk about

- Standardisation
- Automation
- Collaboration

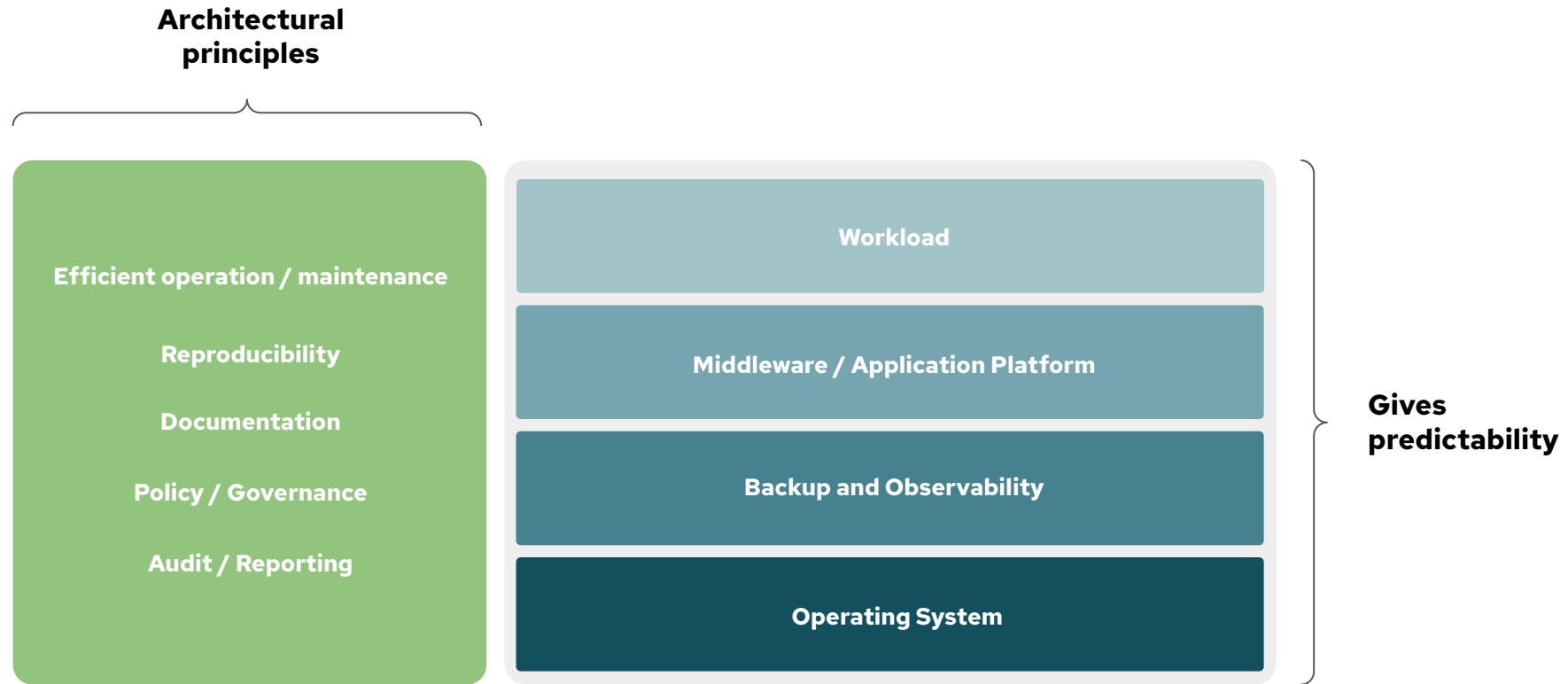
# It all starts with standardisation





# Standardisation in IT

## Example Viewpoint



# Standard Operating Environment (SOE)

## Definition:

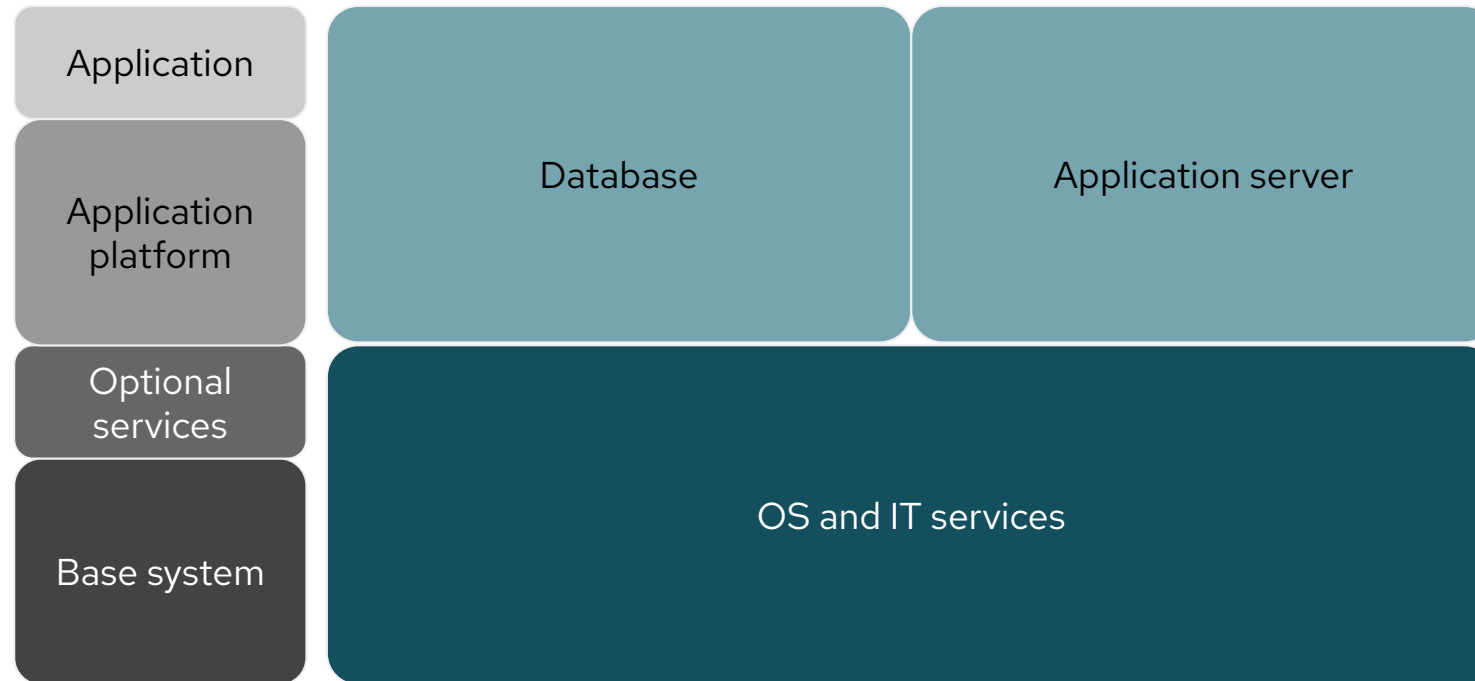
“Provides tools, standards and best practices to manage the lifecycle of an entire, deployed stack – from operating system and infrastructure services through to middleware and applications.”

## 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

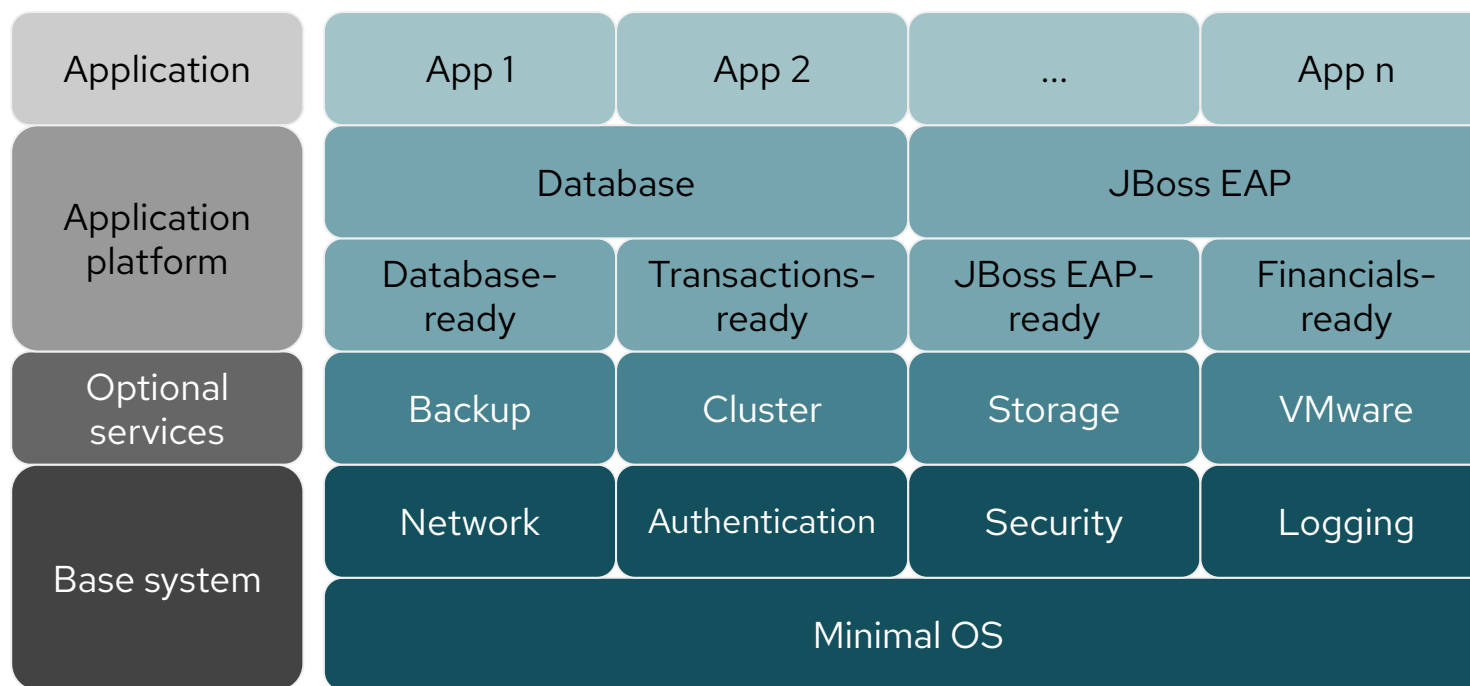


Basic approach



# Standard Operating Environment (SOE)

Concept: Independent yet compatible and interchangeable components

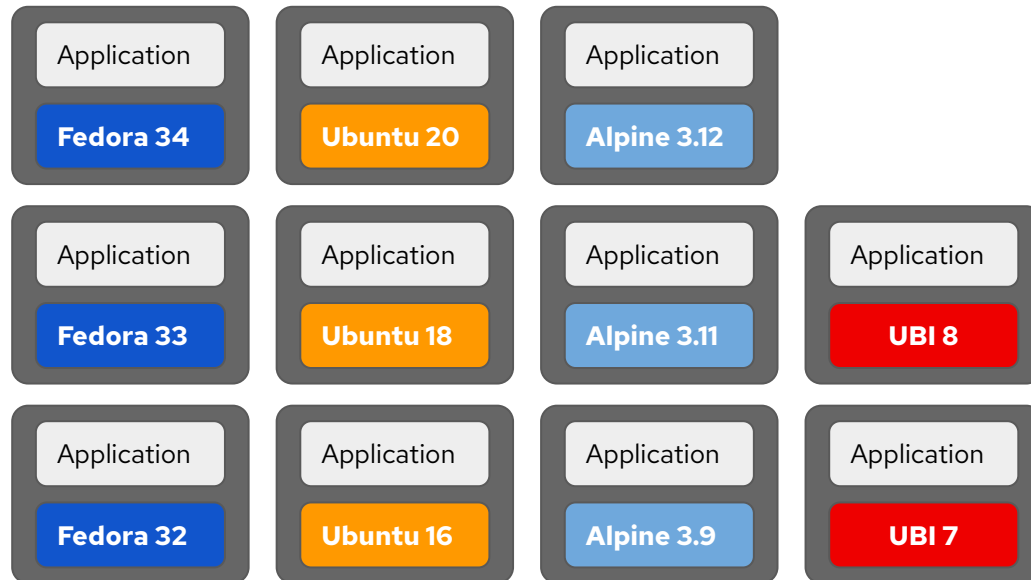


## Adaptive SOE approach

# With no standard

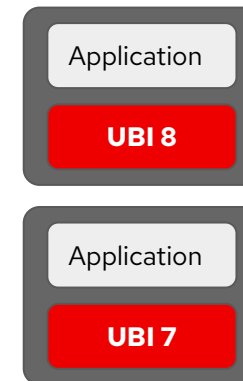
## Everyone will make their own choice

### No Standard Operating Environment



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

### Standard based on UBI 7 & UBI 8



- 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  $\Rightarrow$  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

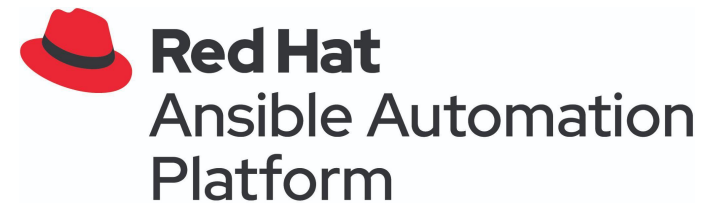
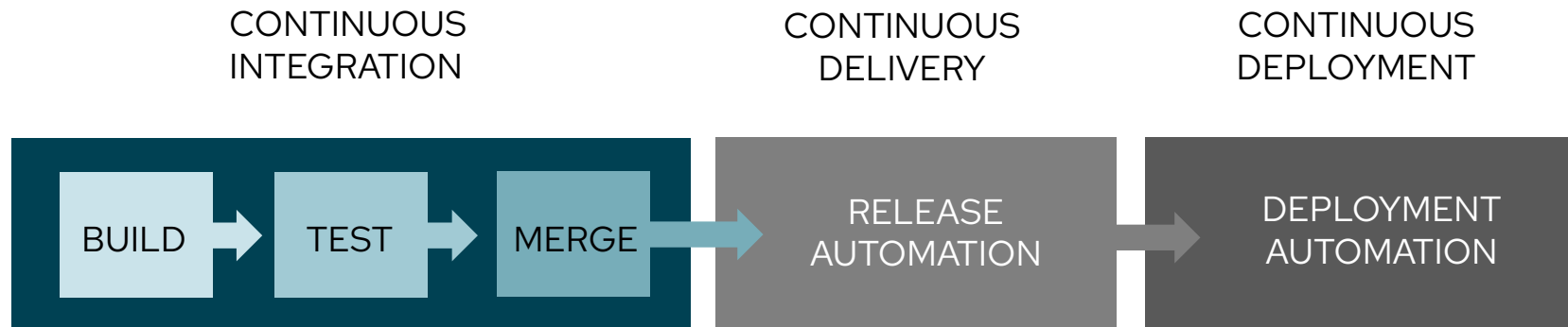
CONFIG  
MANAGEMENT

CONTINUOUS  
DELIVERY

ORCHESTRATION

APP  
DEPLOYMENT

# Application Development and Deployment



# Git\* & Ansible Automation Platform

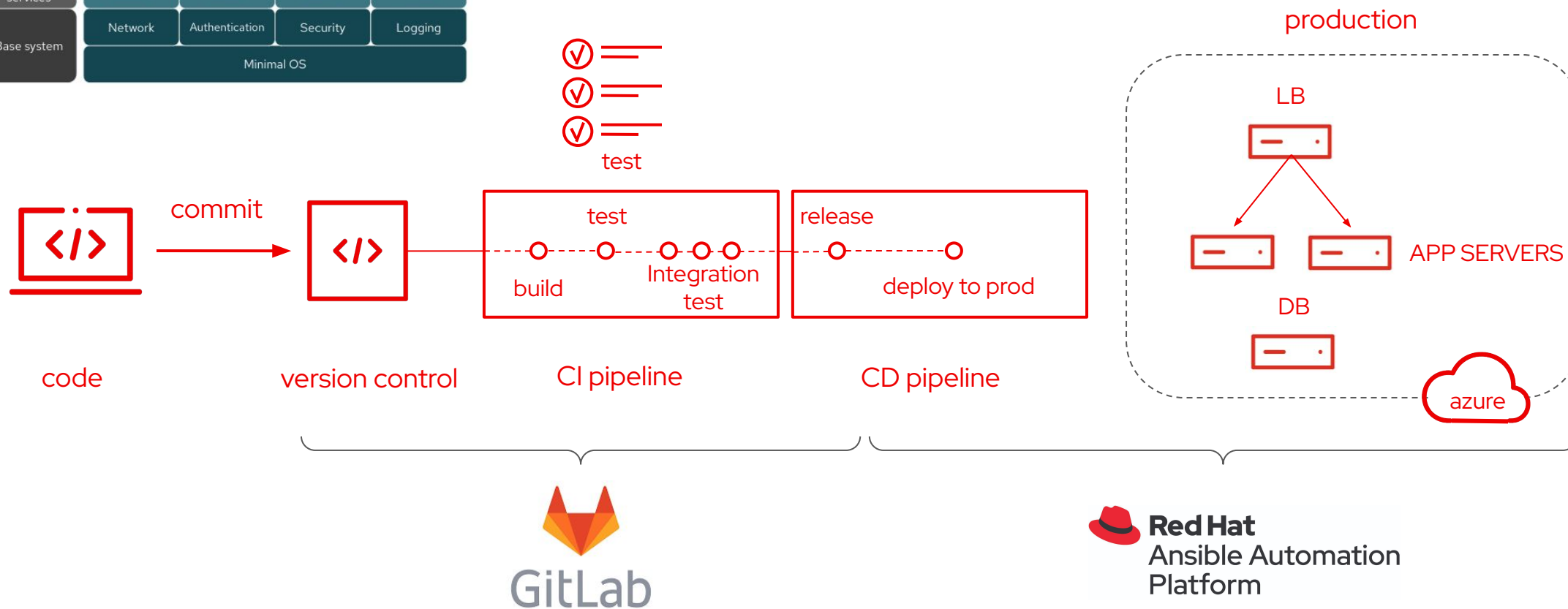
## Application upgrade via CI/CD

- 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.

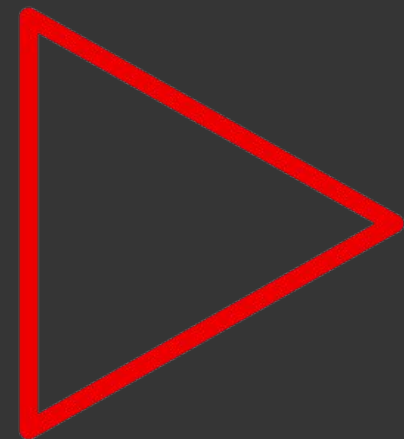
**Seamless  
upgrade of  
application**



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			



# DEMO



## Seamless upgrade of application

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

# Continuous Integration

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

**Simplified  
collaborative  
application  
development**

# OpenShift Pipelines

Open source, standardised,  
cloud-native



based on TEKTON

# Why OpenShift Pipelines?



Built for  
Cloud-native



Scale  
on-demand

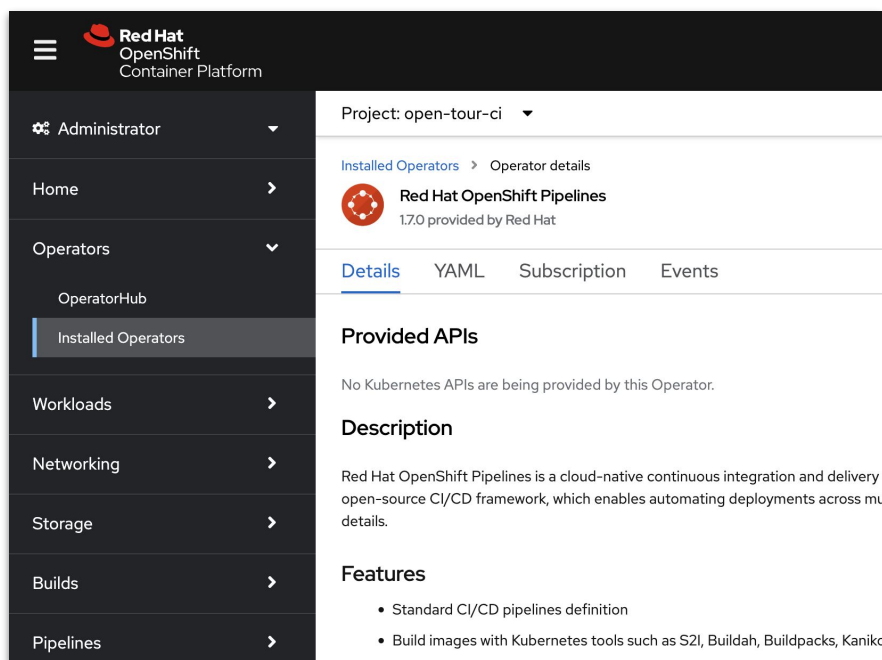


Secure pipeline  
execution

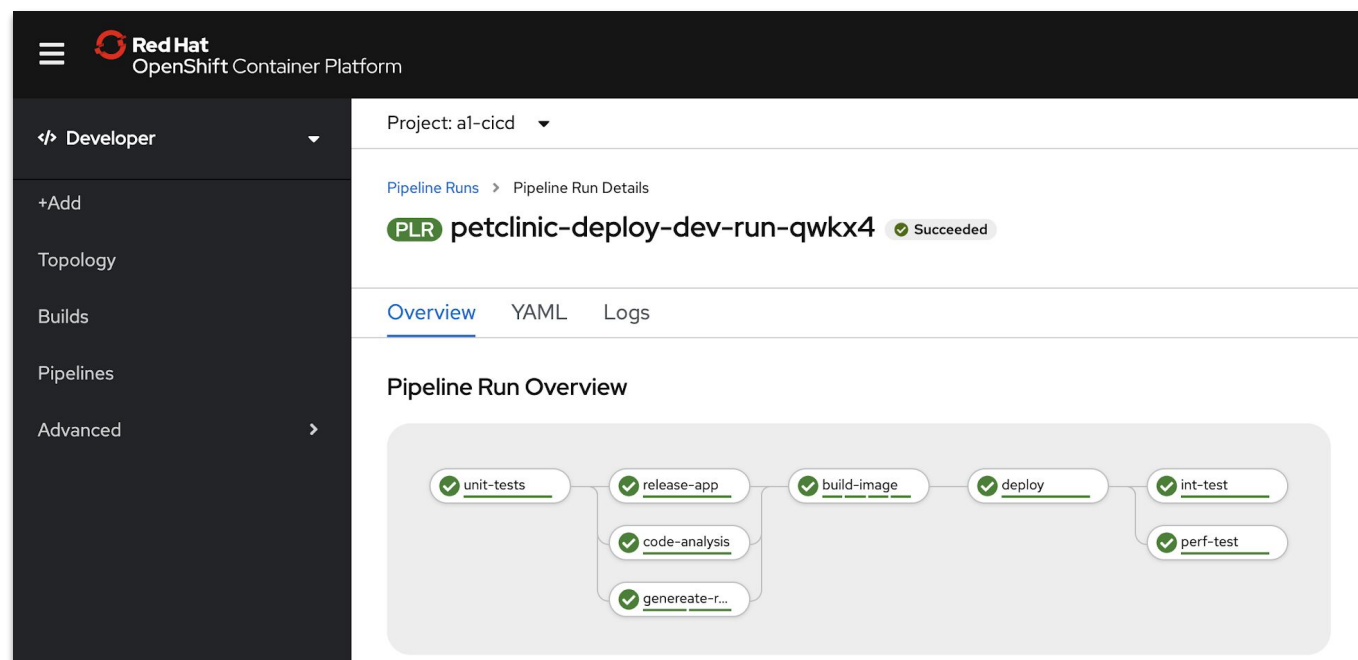


Flexible and  
powerful

# Pipelines as a service



The screenshot shows the Red Hat OpenShift Container Platform Administrator interface. The left sidebar contains navigation links: Administrator, Home, Operators, OperatorHub, Installed Operators, Workloads, Networking, Storage, Builds, and Pipelines. The main content area displays the details for the 'Red Hat OpenShift Pipelines' operator (version 1.7.0). It includes tabs for Details, YAML, Subscription, and Events. The 'Description' section states: 'Red Hat OpenShift Pipelines is a cloud-native continuous integration and delivery open-source CI/CD framework, which enables automating deployments across multiple environments.' The 'Features' section lists: 'Standard CI/CD pipelines definition' and 'Build images with Kubernetes tools such as S2I, Buildah, Buildpacks, Kaniko'.

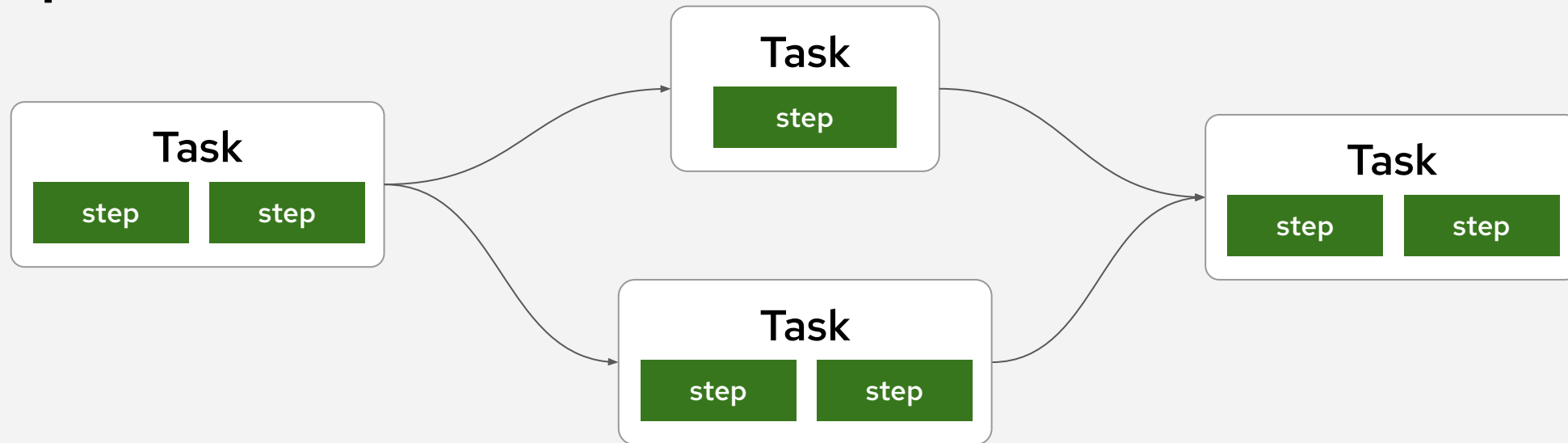


The screenshot shows the Red Hat OpenShift Container Platform Developer interface. The left sidebar contains navigation links: Developer, +Add, Topology, Builds, Pipelines, and Advanced. The main content area displays the details for a pipeline run named 'petclinic-deploy-dev-run-qwkx4' (PLR) with a 'Succeeded' status. It includes tabs for Overview, YAML, and Logs. The 'Pipeline Run Overview' section shows a flowchart of the pipeline steps: unit-tests, release-app, build-image, deploy, int-test, and perf-test. The 'release-app' step is expanded, showing sub-steps: code-analysis and generate-r...



# OpenShift Pipelines - Tekton concepts

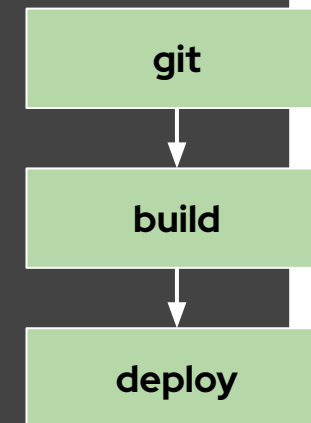
## Pipeline



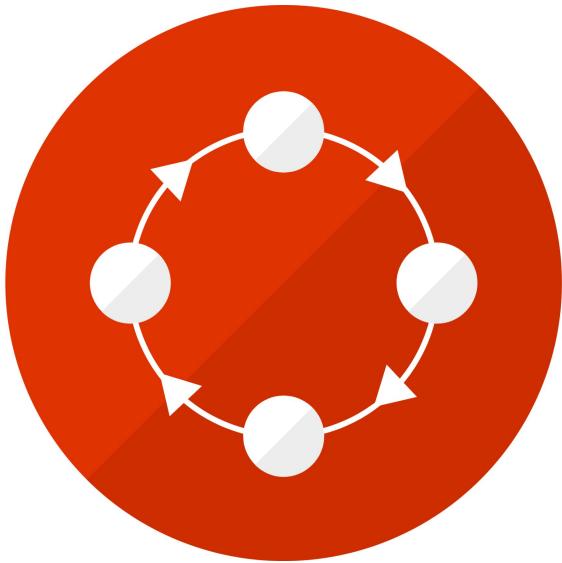
# Everything as code

```
.
├── 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"]
```

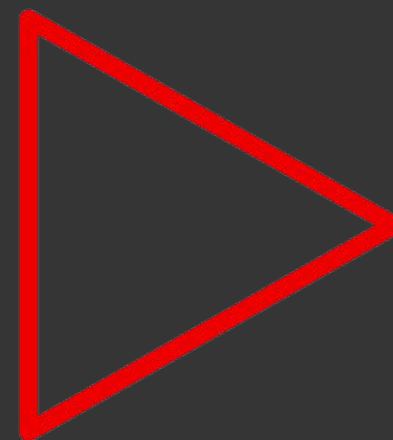


# Pipelines as code



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

# 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

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

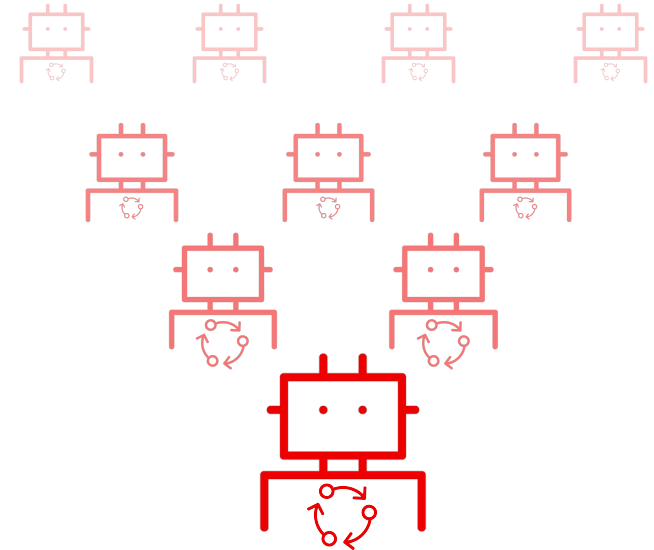
**Automated  
business  
continuity**

We want everything as code.

Applications, configurations  
and secrets delivered to  
autonomous environments.

Visible change history.

Comes with self healing.







## 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



# 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

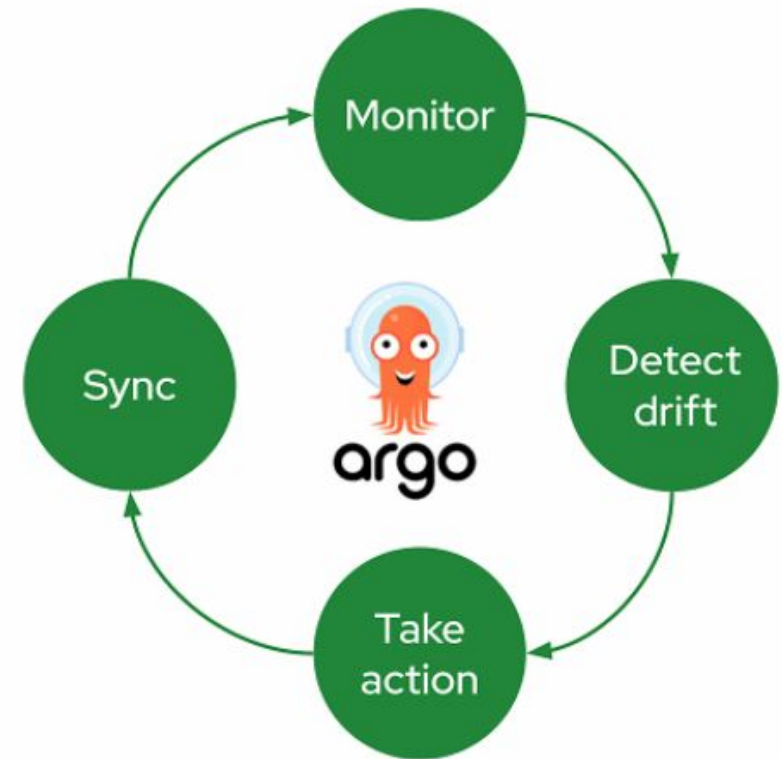


## Deployments and environments insights

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

# 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



v2.3.4+

Applications

+ NEW APP

SYNC APPS

REFRESH APPS

Search applications...

FILTERS

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

SYNC

REFRESH

DELETE

APPLICATION DETAILS

MORE

To 0a58111

(GMT+0200)

g@redhat.com> - sync-hooks

dev-rest-http-example

svc

3 minutes

dev-rest-http-example

ep

3 minutes

dev-rest-http-example-r2hg5

ES endpointslice

3 minutes

dev-rest-http-example-6f78d5d...

rs

3 minutes rev:1

dev-rest-http-example-1

B build

3 minutes

dev-rest-http-example-6f78d5d...

pod

3 minutes running 1/1

dev-rest-http-example-1-ca

cm

3 minutes

dev-rest-http-example-1-global...

cm

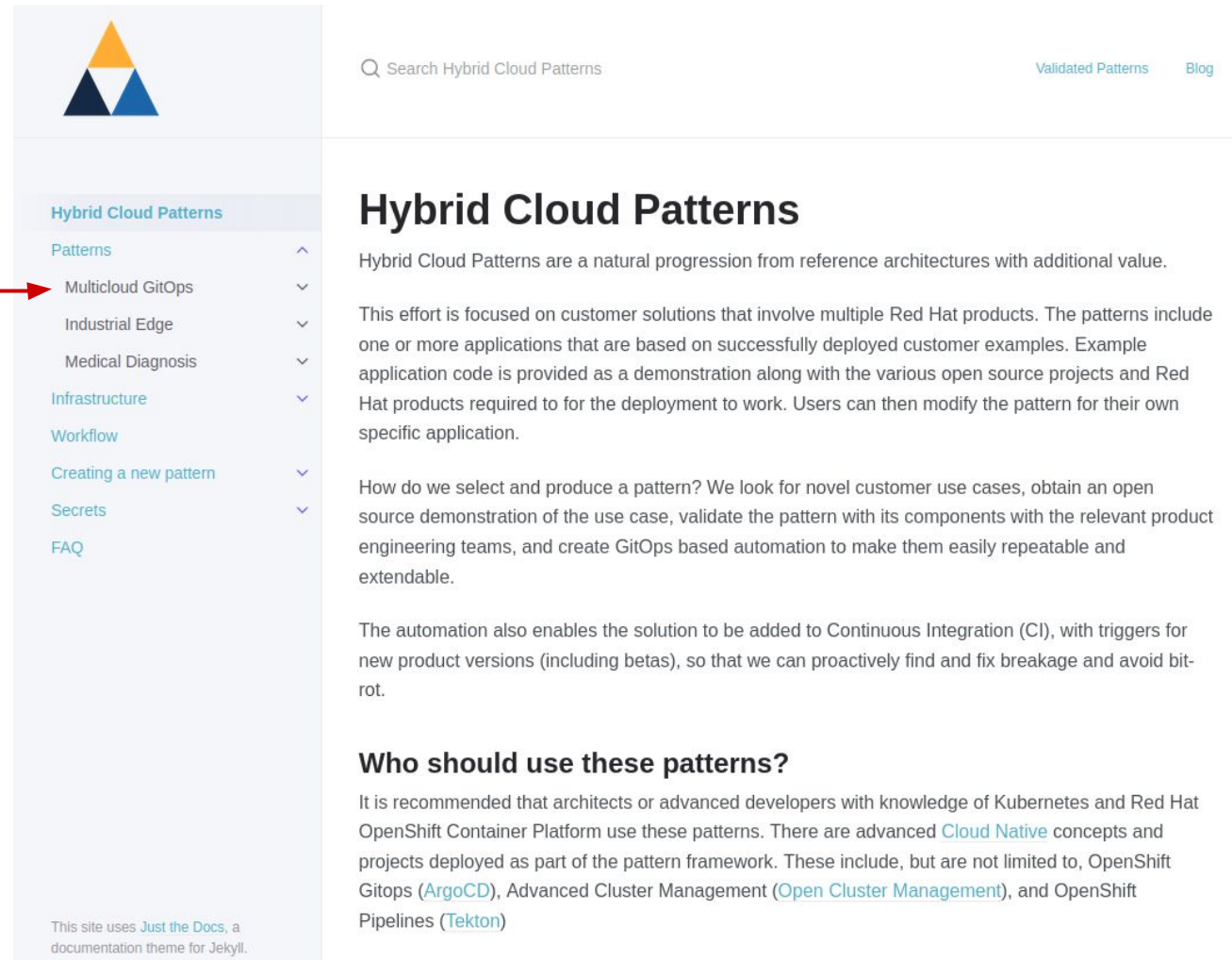
3 minutes

dev-rest-http-example-1-sys-co...

cm

3 minutes

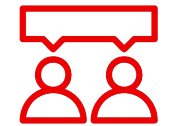
38



The screenshot shows the 'Hybrid Cloud Patterns' website. The left sidebar contains a navigation menu with the following items: 'Hybrid Cloud Patterns' (highlighted), 'Patterns', 'Multicloud GitOps', 'Industrial Edge', 'Medical Diagnosis', 'Infrastructure', 'Workflow', 'Creating a new pattern', 'Secrets', and 'FAQ'. A red arrow points from the 'Multicloud GitOps' link in the sidebar to the main content area. The main content area has a search bar at the top right with the text 'Search Hybrid Cloud Patterns', and links for 'Validated Patterns' and 'Blog'. The main heading is 'Hybrid Cloud Patterns'. Below it, there is a paragraph: 'Hybrid Cloud Patterns are a natural progression from reference architectures with additional value.' This is followed by another paragraph: 'This effort is focused on customer solutions that involve multiple Red Hat products. The patterns include one or more applications that are based on successfully deployed customer examples. Example application code is provided as a demonstration along with the various open source projects and Red Hat products required to for the deployment to work. Users can then modify the pattern for their own specific application.' Then, another paragraph: 'How do we select and produce a pattern? We look for novel customer use cases, obtain an open source demonstration of the use case, validate the pattern with its components with the relevant product engineering teams, and create GitOps based automation to make them easily repeatable and extendable.' This is followed by a paragraph: 'The automation also enables the solution to be added to Continuous Integration (CI), with triggers for new product versions (including betas), so that we can proactively find and fix breakage and avoid bit-rot.' Below this is a section titled 'Who should use these patterns?' with the text: 'It is recommended that architects or advanced developers with knowledge of Kubernetes and Red Hat OpenShift Container Platform use these patterns. There are advanced Cloud Native concepts and projects deployed as part of the pattern framework. These include, but are not limited to, OpenShift Gitops (ArgoCD), Advanced Cluster Management (Open Cluster Management), and OpenShift Pipelines (Tekton)'. At the bottom left of the sidebar, there is a small note: 'This site uses Just the Docs, a documentation theme for Jekyll.'

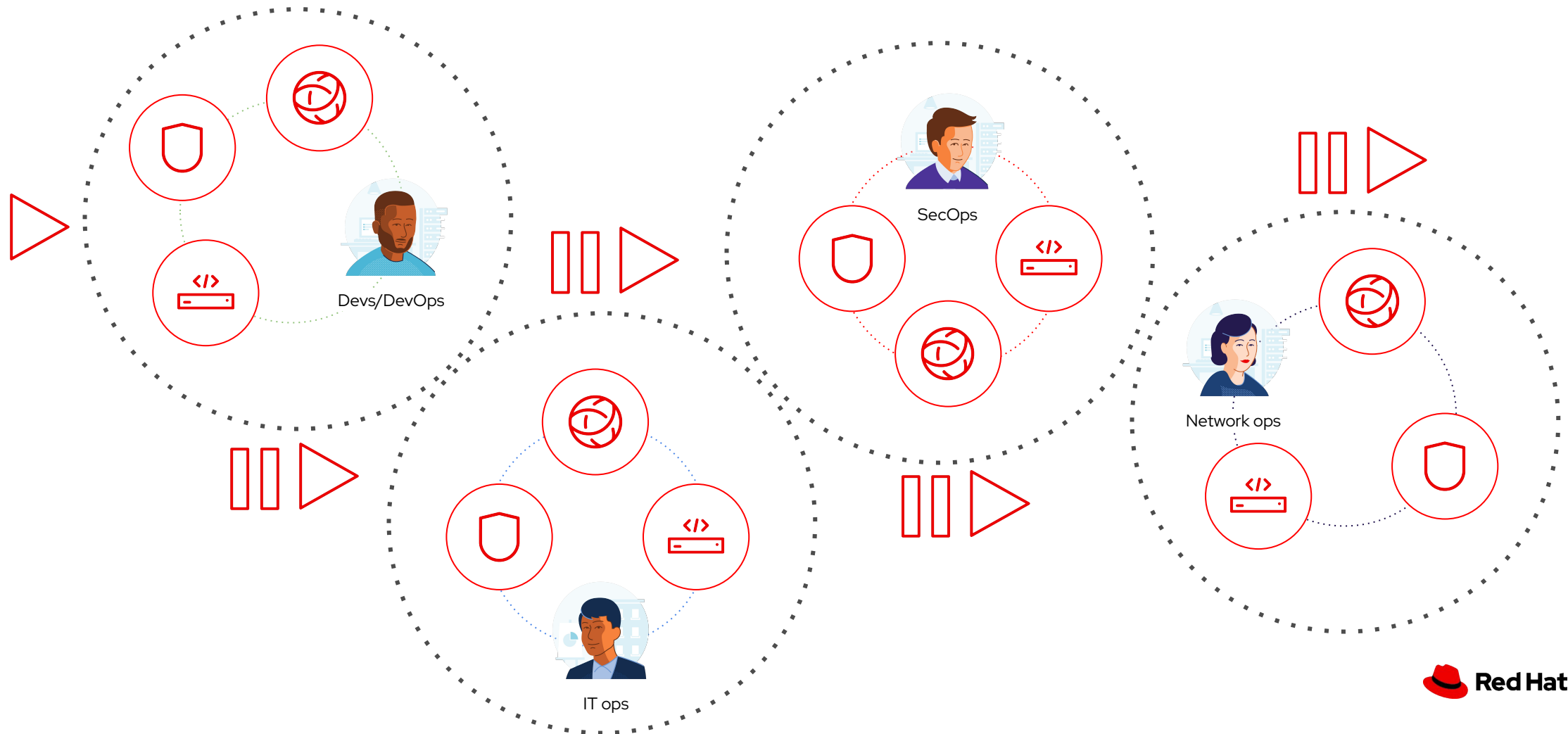
Let's look at the multicloud gitops pattern today

# 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!