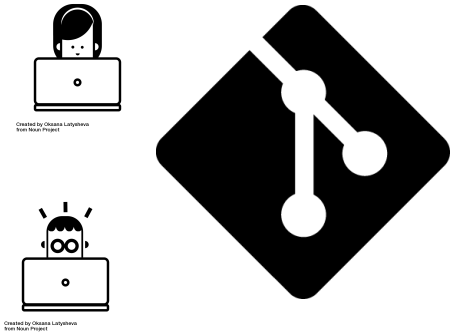
Why GitOps Matters for your business success

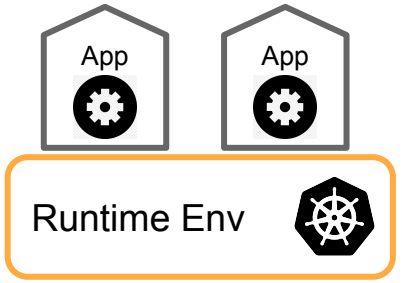
Steve George
COO, Weaveworks
@futurile ▪ steveg@weave.works

Thesis

1. GitOps supports the DevOps approach in a highly effective way
2. GitOps brings DevOps to Cloud Native software operations
3. It can significantly improve the way we operate software

What is GitOps?

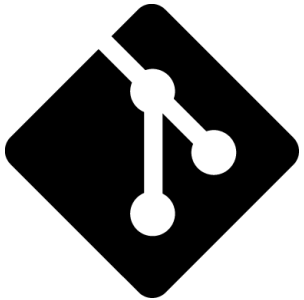

Created by Oksana Latysheva
from Noun Project
Created by Oksana Latysheva
from Noun Project

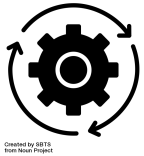

App
App
Runtime Env

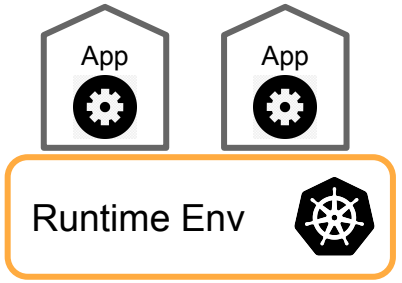
Desired State
Store all code and configuration

Automation

Actual State
Runtime environment



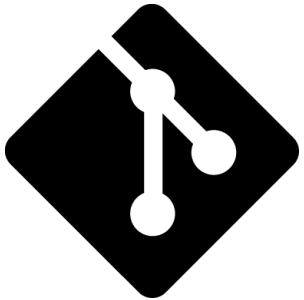


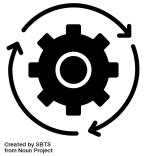


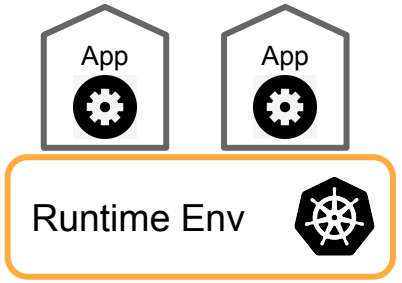
Desired State
Store all code and configuration

Automation

Actual State
Runtime environment





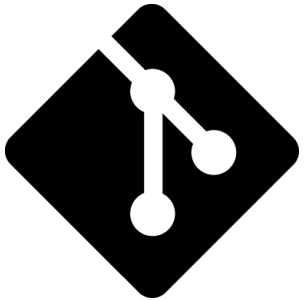


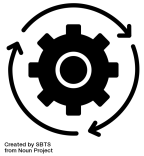
Declarative
Convergent loop

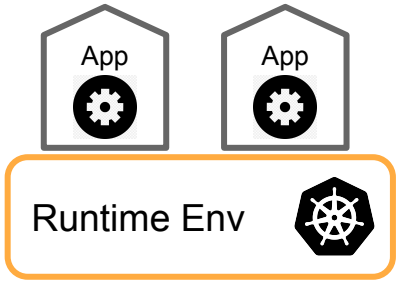
Desired State
Store all code and configuration

Automation

Actual State
Runtime environment







Single interface to operations

CNCF - Principles of GitOps

The entire system is described **declaratively**

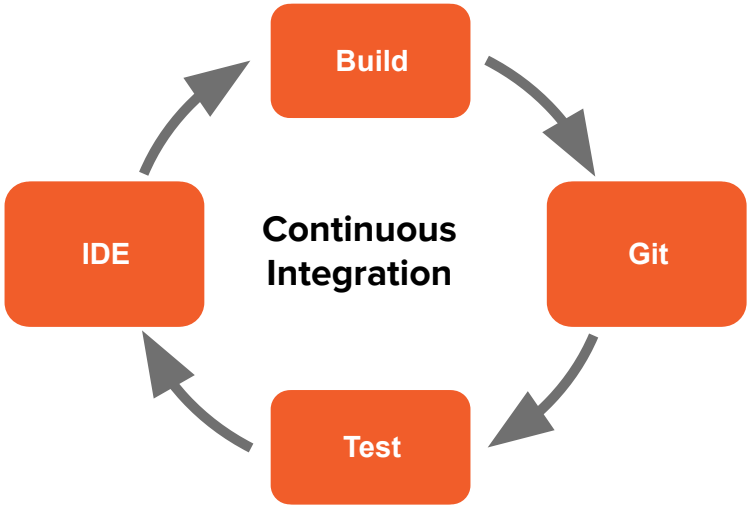
The canonical desired system state is **versioned** in git

Approved changes can be **automatically applied** to the system

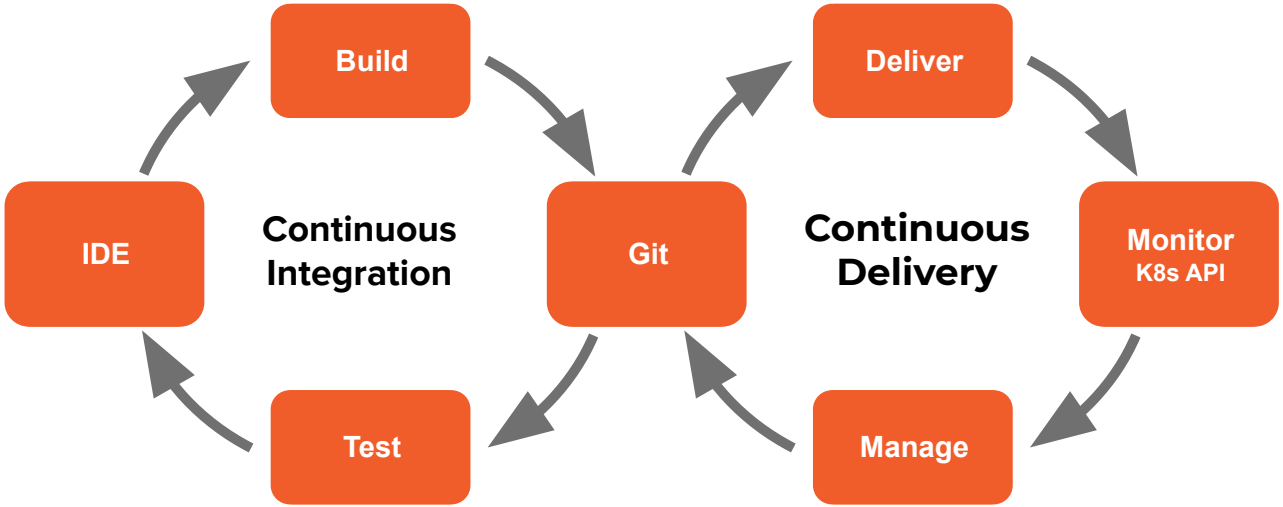
Software agents ensure correctness and alert (diffs & actions)

Actions are performed when the run time state, diverges from the state in git, creating a **closed loop system**

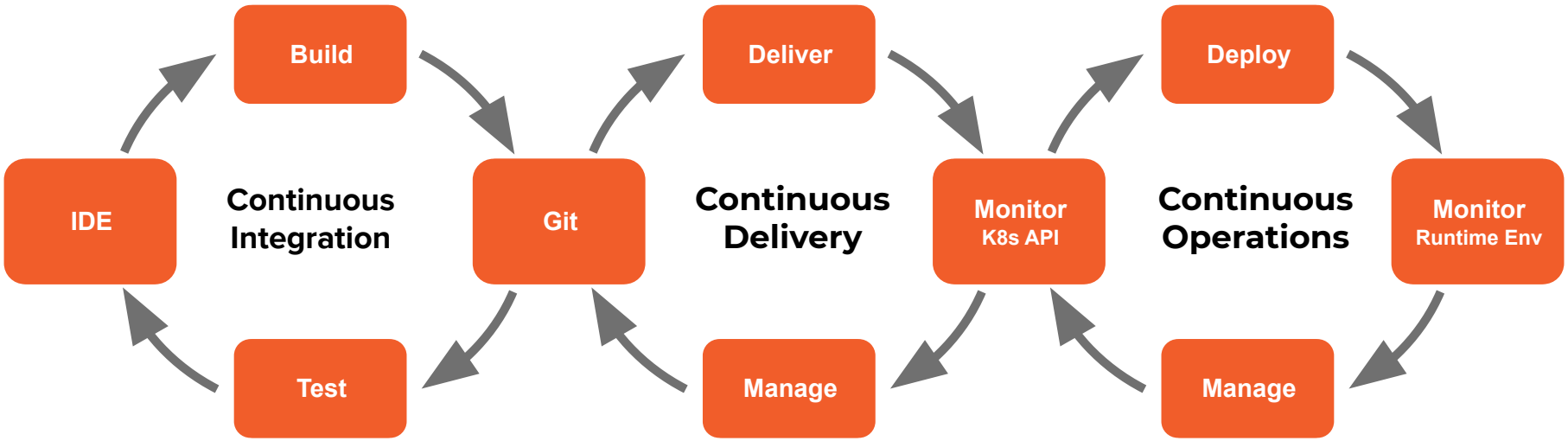
GitOps - Extending how we operate



GitOps - Extending how we operate



GitOps - Extending how we operate



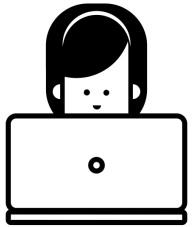
Get better at doing software

Aspect of Software Delivery Performance*	Elite	High	Medium	Low
Deployment frequency For the primary application or service you work on, how often does your organization deploy code to production or release it to end users?	On-demand (multiple deploys per day)	Between once per day and once per week	Between once per week and once per month	Between once per month and once every six months
Lead time for changes For the primary application or service you work on, what is your lead time for changes (i.e., how long does it take to go from code committed to code successfully running in production)?	Less than one day	Between one day and one week	Between one week and one month	Between one month and six months
Time to restore service For the primary application or service you work on, how long does it generally take to restore service when a service incident or a defect that impacts users occurs (e.g., unplanned outage or service impairment)?	Less than one hour	Less than one day[a]	Less than one day[a]	Between one week and one month
Change failure rate For the primary application or service you work on, what percentage of changes to production or released to users result in degraded service (e.g., lead to service impairment or service outage) and subsequently require remediation (e.g., require a hotfix, rollback, fix forward, patch)?	0-15%[b,c]	0-15%[b,d]	0-15%[c,d]	46-60%

https://cloud.google.com/blog/products/devops-sre/the-2019-accelerate-state-of-devops-elite-performance-productivity-and-scaling

- Enable application (DevOps) teams to release more frequently, reduce lead time & operate cloud native applications more effectively

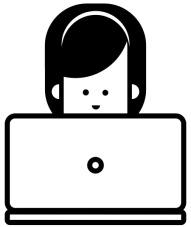
Familiar Tooling
Enables Self-Service



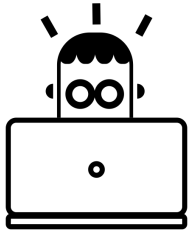
Created by Oksana Latysheva
from Noun Project

- Enable application (DevOps) teams to release more frequently, reduce lead time & operate cloud native applications more effectively

Familiar Tooling
Enables Self-Service



Created by Oksana Latysheva
from Noun Project



Created by Oksana Latysheva
from Noun Project

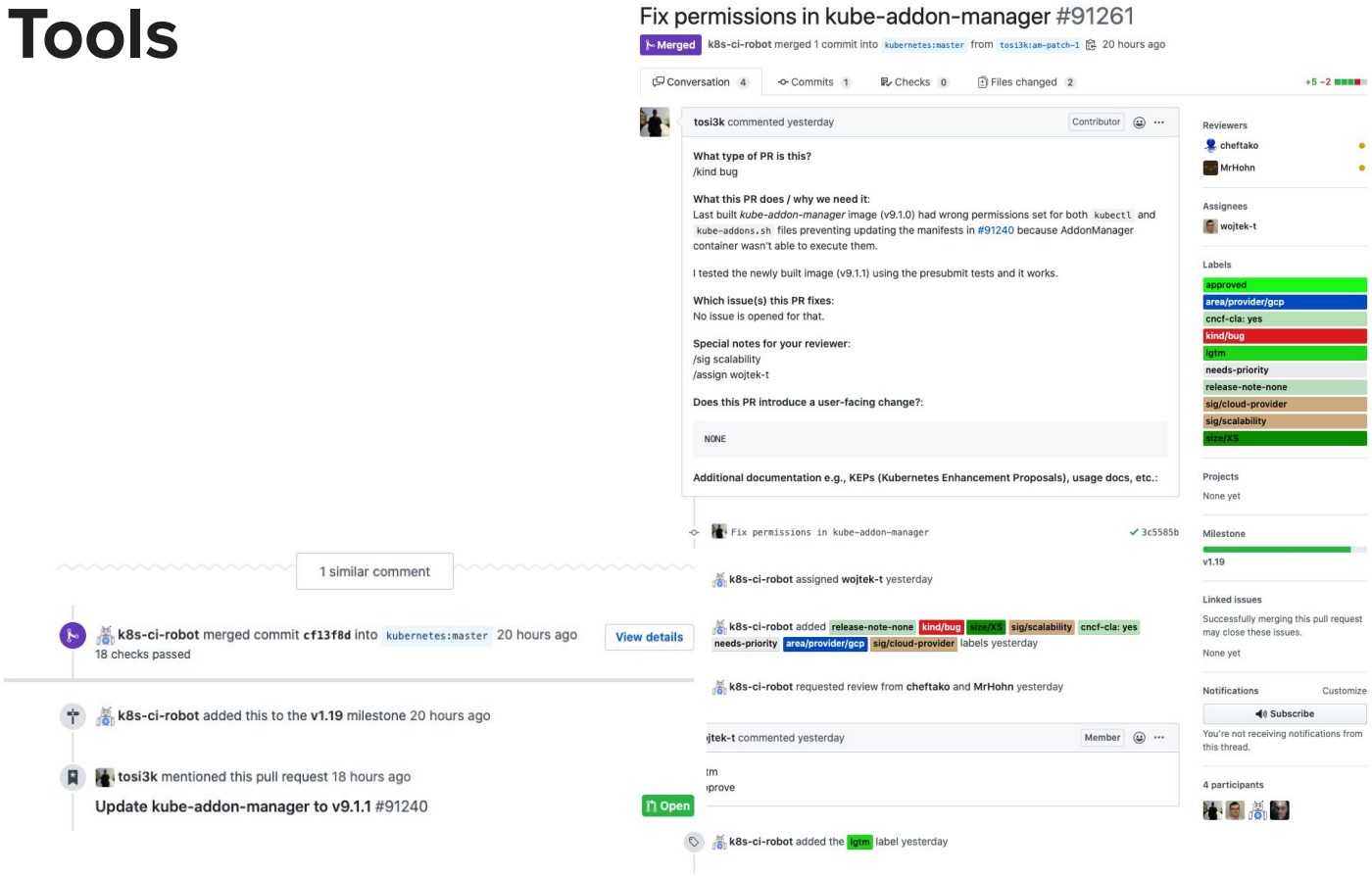
- Enable Platform teams to maintain reliability, security, compliance and cost management

Enables Resilience
Security and compliance built-in

Aspect of Software Delivery Performance*	Elite	High	Medium	Low
Deployment frequency For the primary application or service you work on, how often does your organization deploy code to production or release it to end users?	On-demand (multiple deploys per day)	Between once per day and once per week	Between once per week and once per month	Between once per month and once every six months
Lead time for changes For the primary application or service you work on, what is your lead time for changes (i.e., how long does it take to go from code committed to code successfully running in production)?	Less than one day	Between one day and one week	Between one week and one month	Between one month and six months
Time to restore service For the primary application or service you work on, how long does it generally take to restore service when a service incident or a defect that impacts users occurs (e.g., unplanned outage or service impairment)?	Less than one hour	Less than one day[a]	Less than one day[a]	Between one week and one month
Change failure rate For the primary application or service you work on, what percentage of changes to production or released to users result in degraded service (e.g., lead to service impairment or service outage) and subsequently require remediation (e.g., require a hotfix, rollback, fix forward, patch)?	0-15%[b,c]	0-15%[b,d]	0-15%[c,d]	46-60%

https://cloud.google.com/blog/products/devops-sre/the-2019-accelerate-state-of-devops-elite-performance-productivity-and-scaling

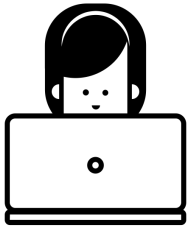
Familiar Tools



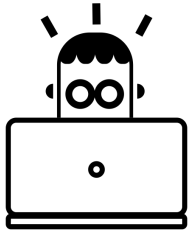
Self Service Ops
(not self-service infra)

- Enable application (DevOps) teams to release more frequently, reduce lead time & operate cloud native applications more effectively

Familiar Tooling
Enables Self-Service



Created by Oksana Latysheva
from Noun Project



Created by Oksana Latysheva
from Noun Project

- Enable Platform teams to maintain reliability, security, compliance and cost management

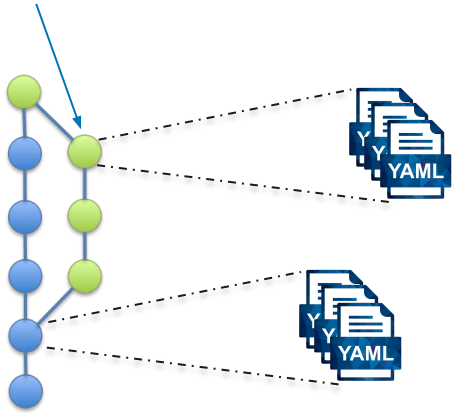
Enables Resilience
Security and compliance built-in

Resilience through Git Semantics

versioned

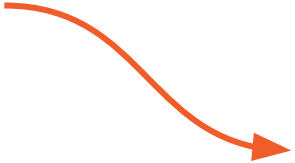
```
less                                                    ⌥⌘2
commit 119c3905b3c0a975e313556f7b2049492a8252ac (HEAD -> master, origin/master, origin/HEAD)
Merge: 5433c28d 7377aae0
Author: Hidde Beydals <hiddeco@users.noreply.github.com>
Date:   Mon May 18 21:00:34 2020 +0200

    Merge pull request #3060 from circa10a/docs/helm-operator-integration

commit 7377aae0c1e3e01021c818a1d4dd8b8487e51b5e
Author: Caleb Lemoine <calebLemoine@gmail.com>
Date:   Fri May 15 19:48:40 2020 -0500

    docs: update helm operator integration glob patterns

commit 5433c28dd96585ee9db160f9ea4068ee4ed7b713
Merge: f6b18042 83c084e3
Author: Hidde Beydals <hiddeco@users.noreply.github.com>
Date:   Wed May 13 12:42:36 2020 +0200

    Merge pull request #3053 from maruina/secret-annotations

commit 83c084e3e3ddb99d6b81e1e932ebcf15379f150b
Author: Matteo Ruina <matteo.ruina@skyscanner.net>
Date:   Mon May 11 16:38:33 2020 +0200

    Support annotations in secret

    Do not bump chart

    Fix value used

commit f6b1804221699e38af85ef6ff5da1474d4317c83
Merge: 0ad6a83e e969cd71
Author: Hidde Beydals <hiddeco@users.noreply.github.com>
Date:   Tue May 12 17:50:37 2020 +0200

    Merge pull request #3008 from fluxcd/non-preferred-resources
:
```

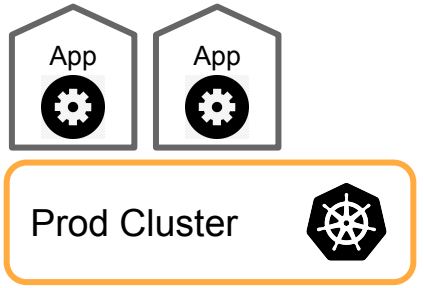
each version is a complete representation



immutabile

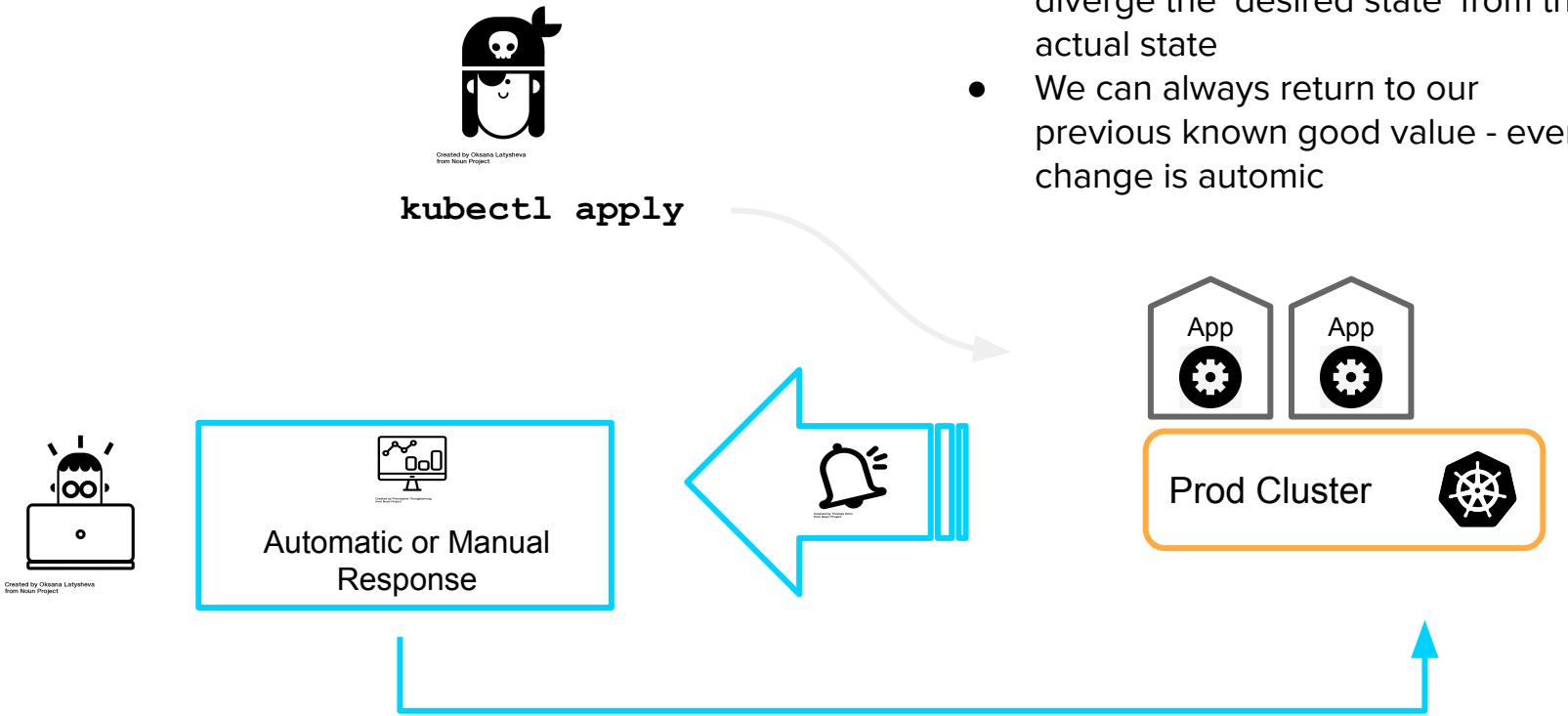
`kubectl apply`





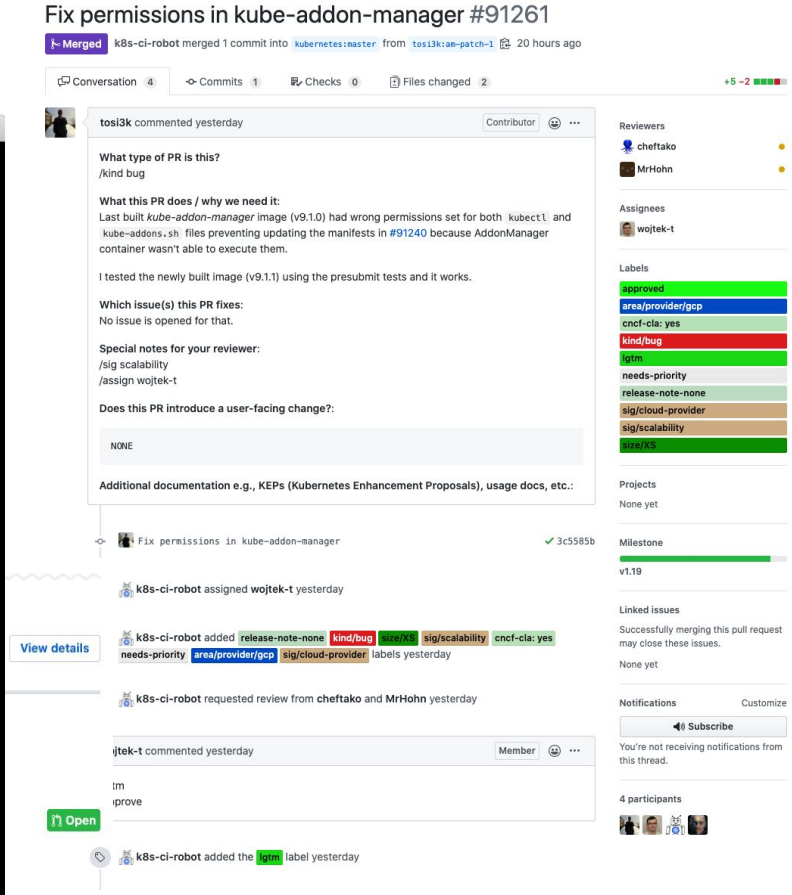
Revert Any change

- Any change to the system will diverge the ‘desired state’ from the actual state
- We can always return to our previous known good value - every change is automic



Compliance ❤️ GitOps

```
less                                                    ⌥⌘2
commit 119c3905b3c0a975e313556f7b2049492a8252ac (HEAD -> master, origin/master, origin/HEAD)
Merge: 5433c28d 7377aae0
Author: Hidde Beydals <hiddeco@users.noreply.github.com>
Date:   Mon May 18 21:00:34 2020 +0200

    Merge pull request #3060 from circa10a/docs/helm-operator-integration

commit 7377aae0c1e3e01021c818a1d4dd8b8487e51b5e
Author: Caleb Lemoine <caleblemoine@gmail.com>
Date:   Fri May 15 19:48:40 2020 -0500

    docs: update helm operator integration glob patterns

commit 5433c28dd96585ee9db160f9ea4068ee4ed7b713
Merge: f6b18042 83c084e3
Author: Hidde Beydals <hiddeco@users.noreply.github.com>
Date:   Wed May 13 12:42:36 2020 +0200

    Merge pull request #3053 from maruina/secret-annotations

commit 83c084e3e3ddb99d6b81e1e932ebcf15379f150b
Author: Matteo Ruina <matteo.ruina@skyscanner.net>
Date:   Mon May 11 16:38:33 2020 +0200

    Support annotations in secret

    Do not bump chart

    Fix value used

commit f6b1804221699e38af85ef6ff5da1474d4317c83
Merge: 0ad6a83e e969cd71
Author: Hidde Beydals <hiddeco@users.noreply.github.com>
Date:   Tue May 12 17:50:37 2020 +0200

    Merge pull request #3008 from fluxcd/non-preferred-resources
:
```

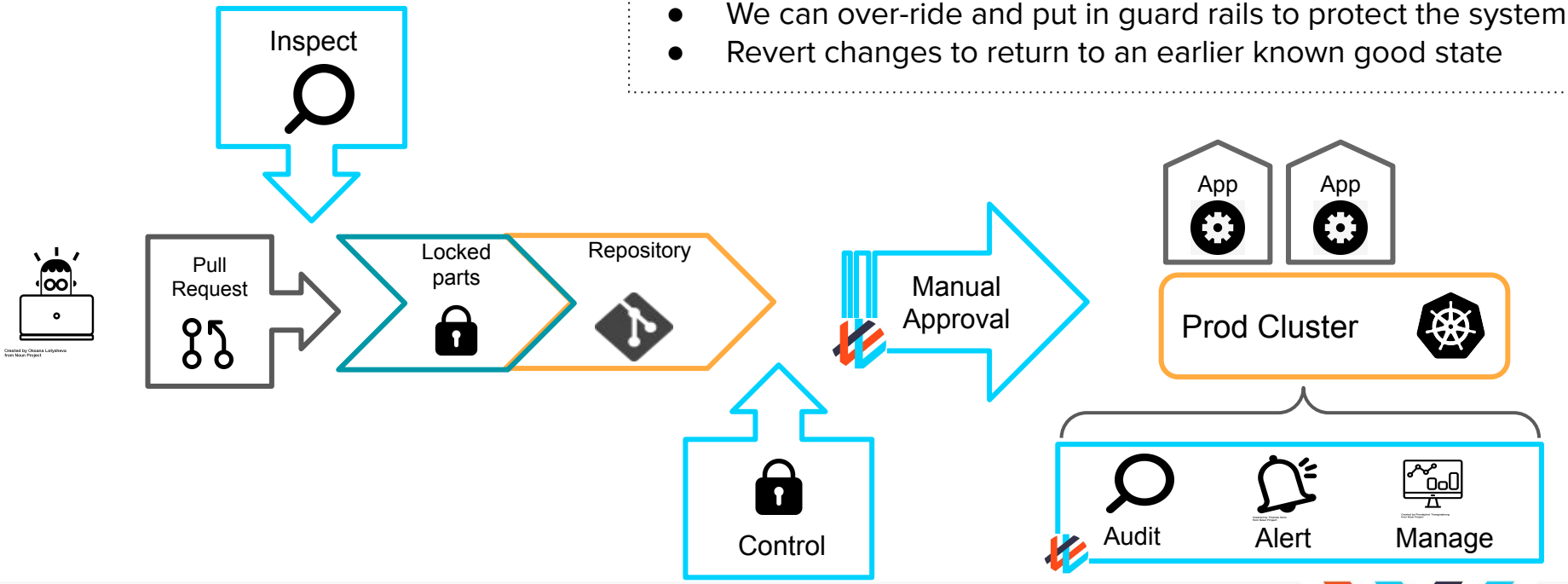


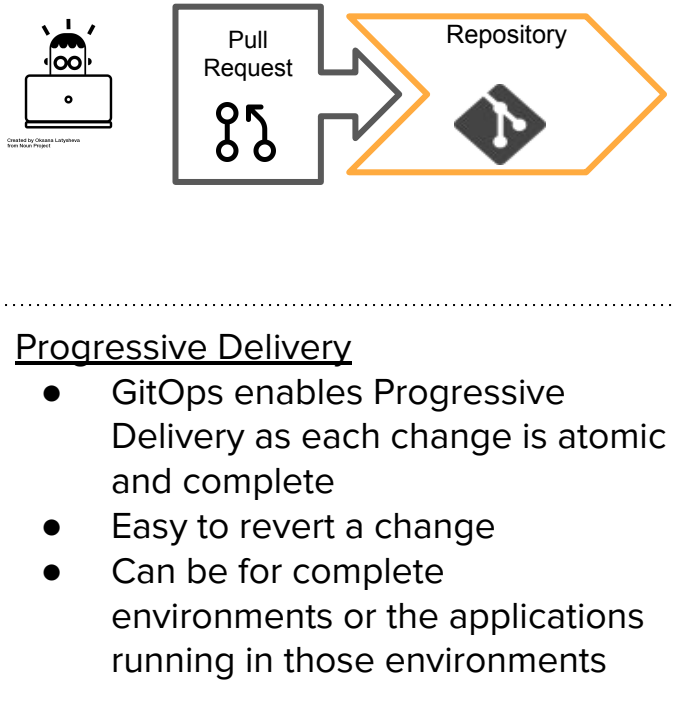
Aspect of Software Delivery Performance*	Elite	High	Medium	Low
Deployment frequency For the primary application or service you work on, how often does your organization deploy code to production or release it to end users?	On-demand (multiple deploys per day)	Between once per day and once per week	Between once per week and once per month	Between once per month and once every six months
Lead time for changes For the primary application or service you work on, what is your lead time for changes (i.e., how long does it take to go from code committed to code successfully running in production)?	Less than one day	Between one day and one week	Between one week and one month	Between one month and six months
Time to restore service For the primary application or service you work on, how long does it generally take to restore service when a service incident or a defect that impacts users occurs (e.g., unplanned outage or service impairment)?	Less than one hour	Less than one day[a]	Less than one day[a]	Between one week and one month
Change failure rate For the primary application or service you work on, what percentage of changes to production or released to users result in degraded service (e.g., lead to service impairment or service outage) and subsequently require remediation (e.g., require a hotfix, rollback, fix forward, patch)?	0-15%[b,c]	0-15%[b,d]	0-15%[c,d]	46-60%

https://cloud.google.com/blog/products/devops-sre/the-2019-accelerate-state-of-devops-elite-performance-productivity-and-scaling

Move fast and don't break things

- Represents the entire state of the system - platform, services and applications
- We can easily inspect changes ensuring they meet standards
- We can over-ride and put in guard rails to protect the system
- Revert changes to return to an earlier known good state





Progressive Delivery

- GitOps enables Progressive Delivery as each change is atomic and complete
- Easy to revert a change
- Can be for complete environments or the applications running in those environments

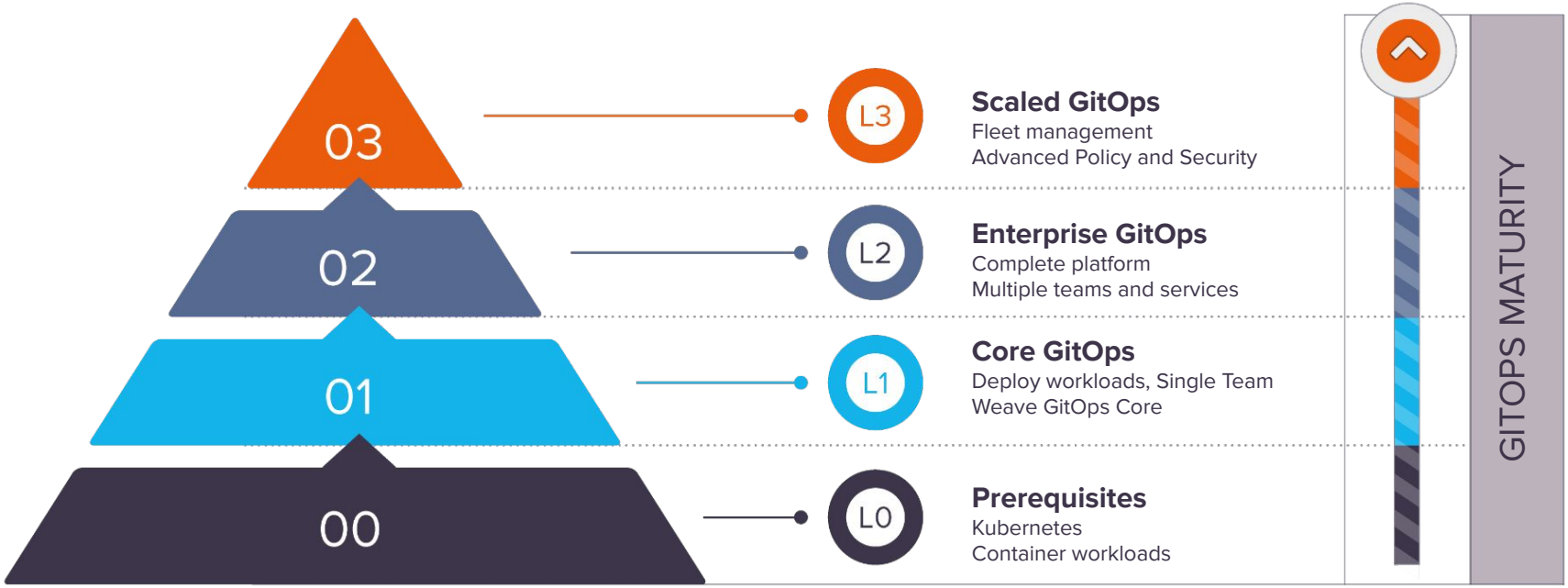
Aspect of Software Delivery Performance*	Elite	High	Medium	Low
Deployment frequency For the primary application or service you work on, how often does your organization deploy code to production or release it to end users?	On-demand (multiple deploys per day)	Between once per day and once per week	Between once per week and once per month	Between once per month and once per six months
Lead time for changes For the primary application or service you work on, what is your lead time for changes (i.e., how long does it take to go from code committed to code successfully running in production)?	Less than one day	Between one day and one week	Between one week and one month	Between one month and six months
Time to restore service For the primary application or service you work on, how long does it generally take to restore service when a service incident or a defect that impacts users occurs (e.g., unplanned outage or service impairment)?	Less than one hour	Less than one day[a]	Less than one day[a]	Between one week and one month
Change failure rate For the primary application or service you work on, what percentage of changes to production or released to users result in degraded service (e.g., lead to service impairment or service outage) and subsequently require remediation (e.g., require a hotfix, rollback, fix forward, patch)?	0-15%[b,c]	0-15%[b,d]	0-15%[c,d]	46-60%

- Familiar Tools - git for collaboration
- Self Service
  - Pull-based (security)
  - Implicit audit log (compliance)
- Resilience
  - Versioned, immutable store
  - Complete representation
  - Drift detection and remediation
- Config reviews in Git
- Progressive Delivery

https://cloud.google.com/blog/products/devops-sre/the-2019-accelerate-state-of-devops-elite-performance-productivity-and-scaling

Adopting GitOps

GitOps Maturity Model - Summary

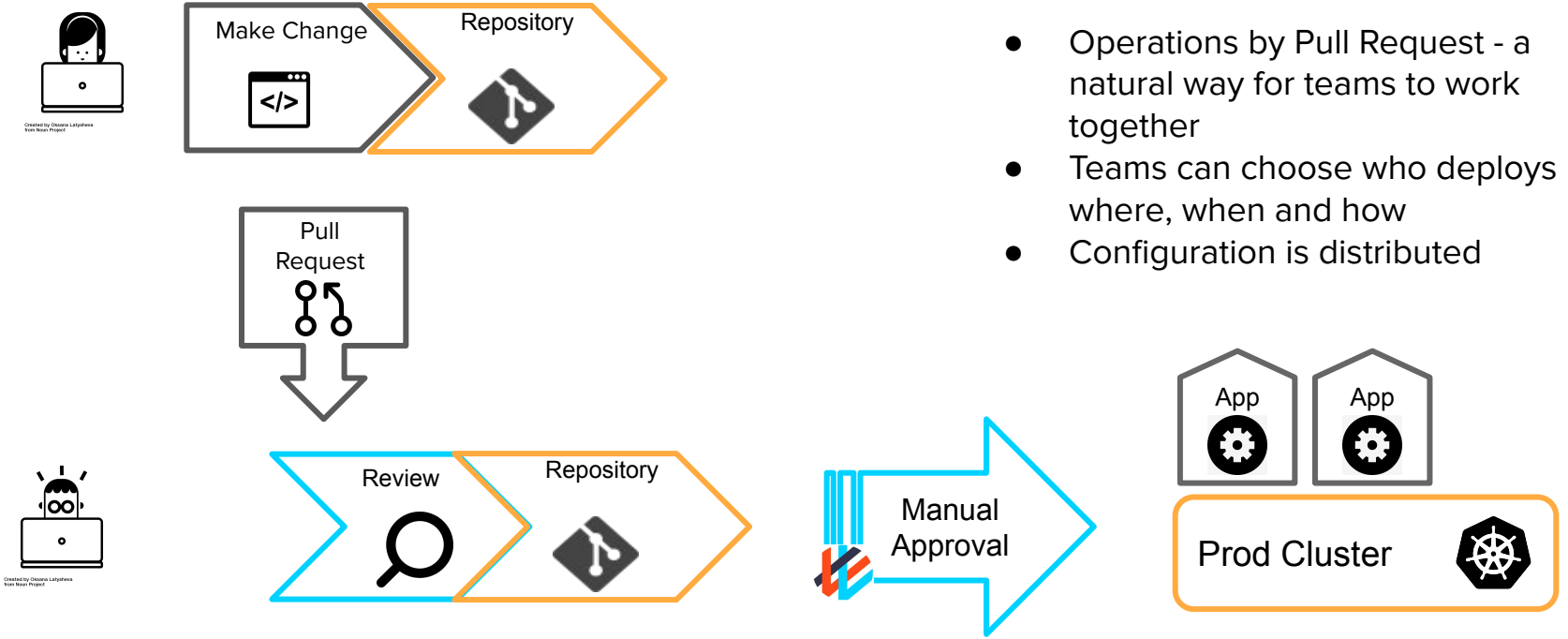


谢谢
Thank You
Danke
Obrigado
Спасибо!
Merci
شكرا

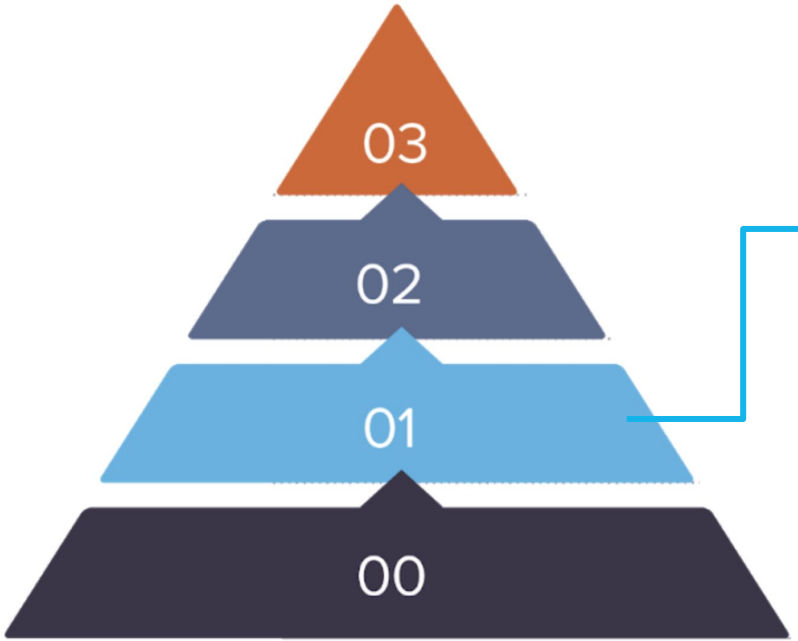
Steve George
COO, Weaveworks
@futurile ▪ steveg@weave.works

A point of collaboration

- Operations by Pull Request - a natural way for teams to work together
- Teams can choose who deploys where, when and how
- Configuration is distributed



Level 1 - Core GitOps



Capabilities

- Start with simple applications and services - few dependencies
- GitOps for the Workload / Application layer
  - Automated reconciliation / drift detection
- Declarative infrastructure as code but without automatic reconciliation
- Environment & Secrets Management

Open Community - GitOps Working Group

- Defining a vendor neutral, principle-led meaning for GitOps
  - Interoperability and clarity
  - Exploring and developing the use-cases
- Industry wide collaboration
  - Amazon, Codefresh, GitHub, Microsoft and Weaveworks are founding members
  - User community
- GitOps Principles
  - Declarative Configuration
  - Version controlled, immutable storage
  - Automated delivery
  - Software agents reconcile
  - Closed loop

https://github.com/gitops-working-group/gitops-working-group

Thesis:

GitOps supports the DevOps agenda in a particularly effective manner. (where Git is the interface for operational actions.)

Evolution of DevOps to GitOps

DevOps		GitOps
Infrastructure as Code	→	Platform as Code
Imperative Automation	→	Declarative Automation
Mutation	→	Immutability
Single State	→	Reconciliation / Convergence
Deployment	→	Deployment and Operations

Principles of GitOps

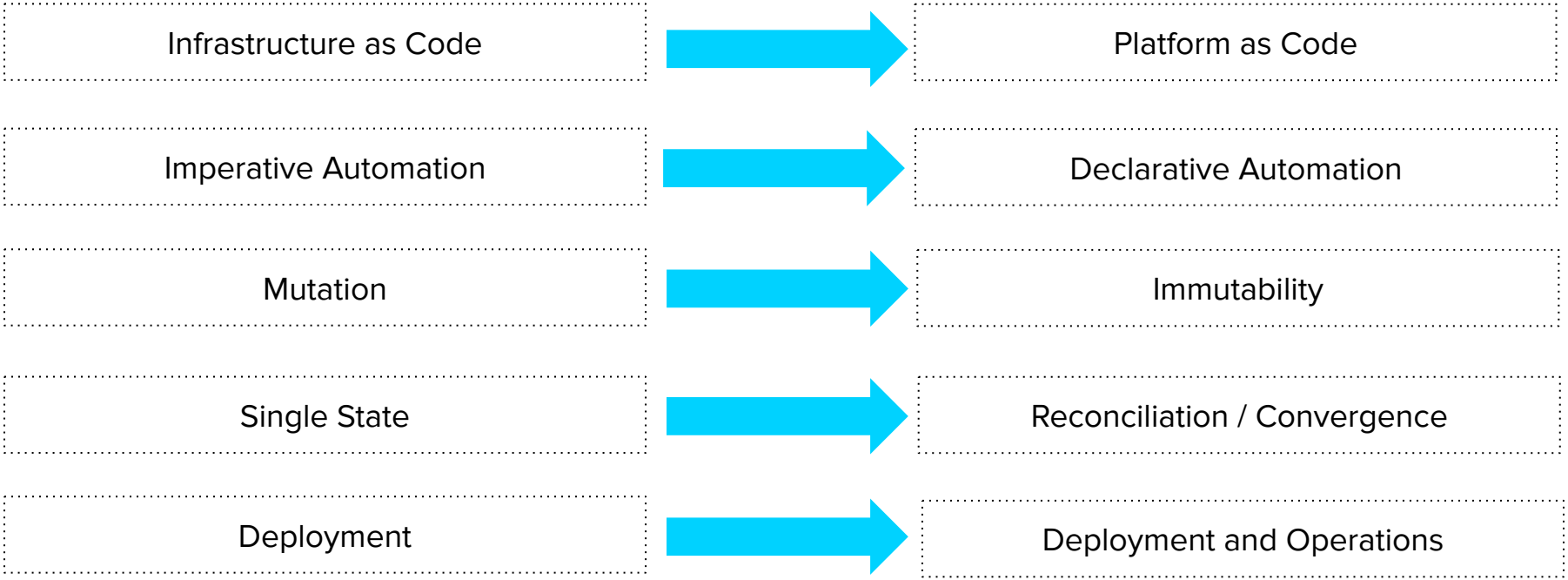
The entire system is described **declaratively**

The canonical desired system state is **versioned** in git

Approved changes an be **automatically applied** to the system

Software agents ensure correctness and alert (diffs & actions)

GitOps Elements



https://github.com/gitops-working-group/gitops-working-group

1. **Declarative Configuration**: All resources managed through a GitOps process must be completely expressed declaratively.
2. **Version controlled, immutable storage**: Declarative descriptions are stored in a repository that supports immutability, versioning and version history. For example, git.
3. **Automated delivery**: Delivery of the declarative descriptions, from the repository to runtime environment, is fully automated.
4. **Software Agents**: Reconcilers maintain system state and apply the resources described in the declarative configuration.
5. **Closed loop**: Actions are performed on divergence between the version controlled declarative configuration and the actual state of the target system.

The Principles of GitOps

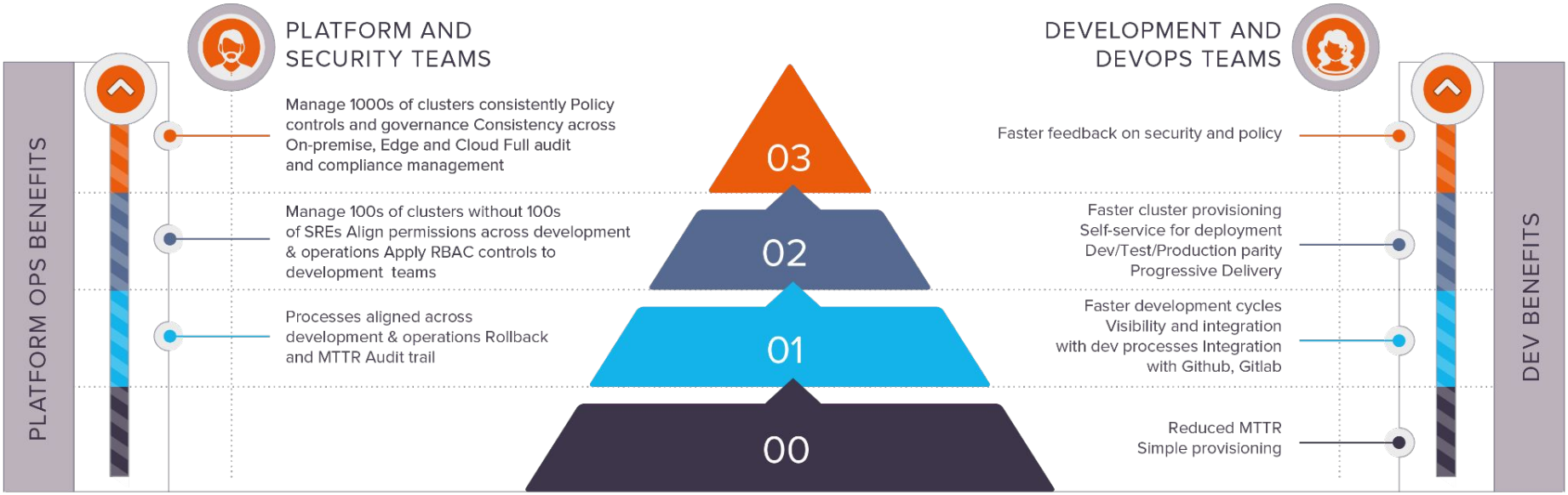
The entire system is described **declaratively**

The canonical desired system state is versioned in **git**

Approved changes can be **automatically applied** to the system

Software agents ensure correctness and perform actions on divergence in a closed loop

GitOps Maturity Model - Benefits for different teams



GitOps Maturity Model - Details

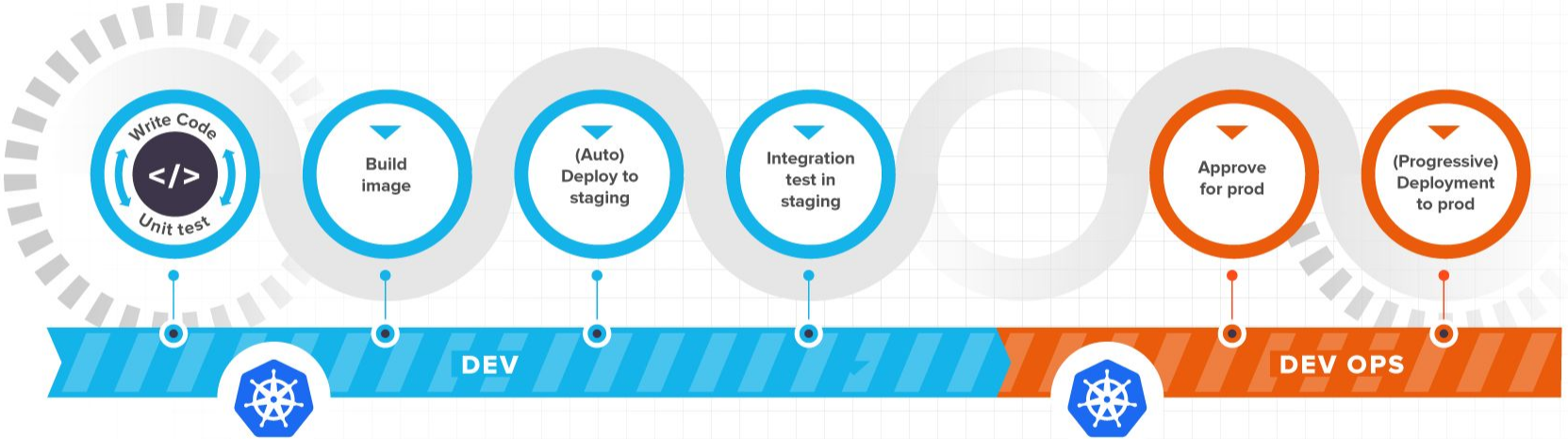
	Capabilities	Business Benefit
Level 3 Scaled GitOps	Fleet management and fleet reconciliation Mass customization Advanced Policy Advanced Cross-Enterprise Governance Multi-Cloud and Hybrid	Enables Support for 5G/IoT/Edge deployment Significant cost reduction for fleet management Supports massive scale Enables Enterprise Governance and Compliance Empowers cross-vendor choice, visibility and control
Level 2 Enterprise GitOps	GitOps for the whole environment - Infra, Cluster, Config, Workloads Separation of Cluster from Workloads Security / RBAC / Audit / Governance / Policy Progressive Delivery Package / Template Customization & Management	Improves agility across teams Further reductions in cost as you scale Enables Governance & Compliance validation & audit Improves security Enables Corporate Policy enforcement
Level 1 Core GitOps	Declarative IaC without reconciliation GitOps for the Workload / App - Automated reconciliation / drift detection Environment & Secrets Management Pull-based approach Package Management (helm or kustomize)	Increases frequency of deployment Reduces cost of operations Reduces MTTR Increases visibility and audit of operations Introduces observability Introduces basic Governance
Level 0 Prerequisites for GitOps	Declarative Infrastructure as Code (IaC) Declarative Workloads Version control, Immutable images Suitable infrastructure (e.g. Kubernetes)	“Cloud-native” Faster provisioning of infrastructures Repeatability Immutability

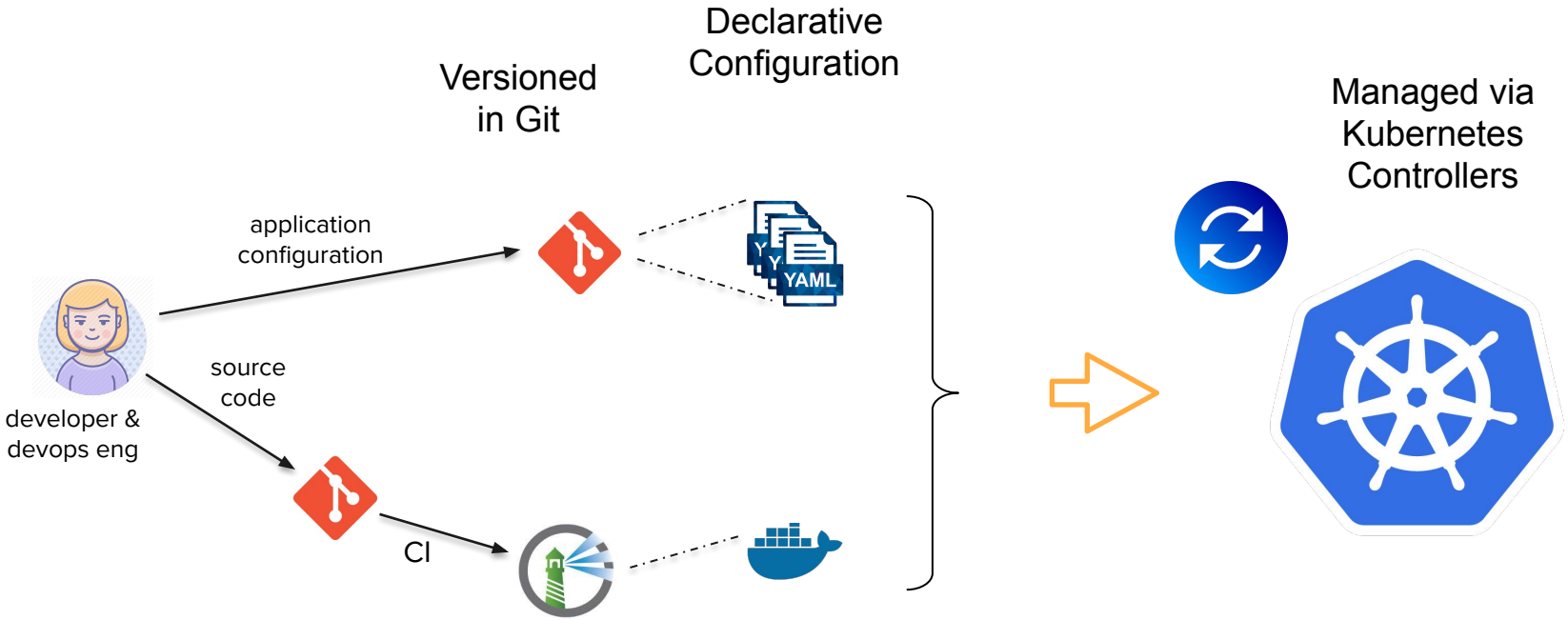
GitOps – An Operating Model for Cloud Native

Unifying Deployment, Monitoring and Management.

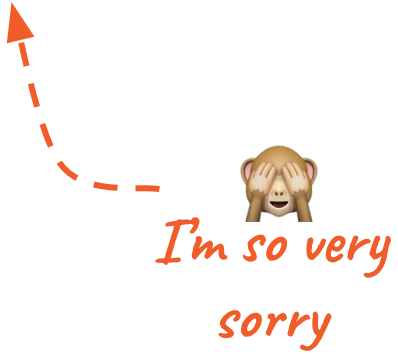
Git as the single source of truth of a system's desired state

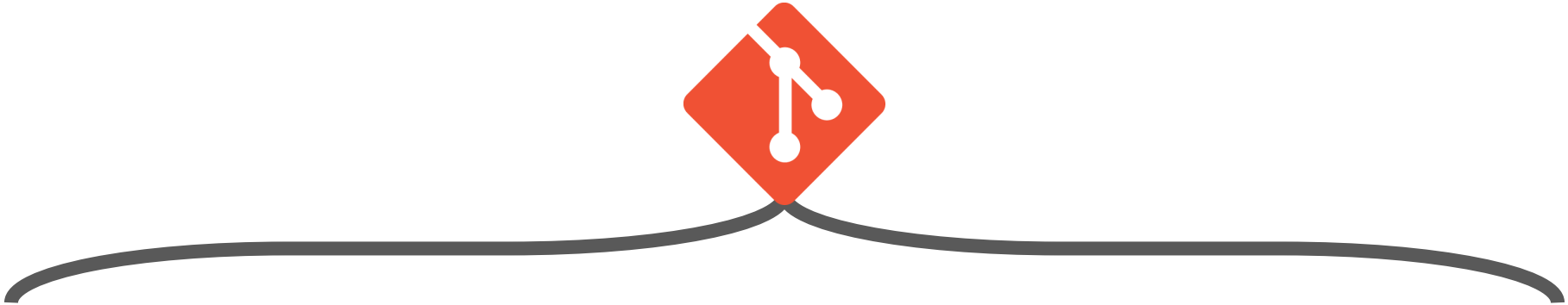
- **ALL** intended operations are committed by pull request
- **ALL** diffs between intended and observed state with automatic convergence
- **ALL** changes are observable, verifiable and auditable



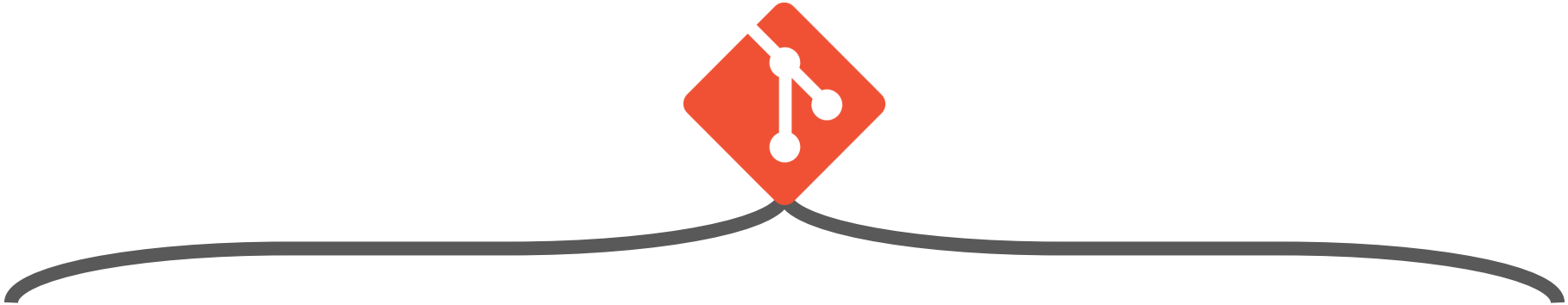

Versioned
in Git
Declarative
Configuration
Managed via
Kubernetes
Controllers
application
configuration
source
code
developer &
devops eng
CI
YAML

What should be GitOps'ed?





?



Kubernetes Manifests

Application configuration

Provisioning config

Dashboards

Alerts

Operator Runbooks

Application checklists

Monitoring Data Pipeline

Sealed Secrets

Use Cases

Different use-cases for Weave GitOps

1. Application Delivery and Operations

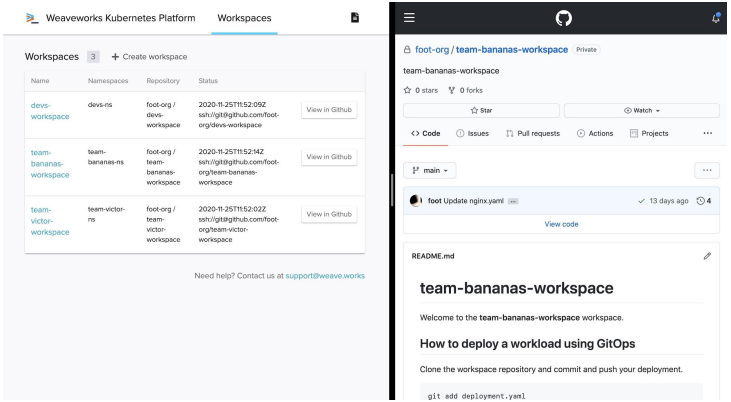
Enabling application development teams to do DevOps at scale. Deploying applications through a dev/staging/prod pipeline.

2. Platform as code

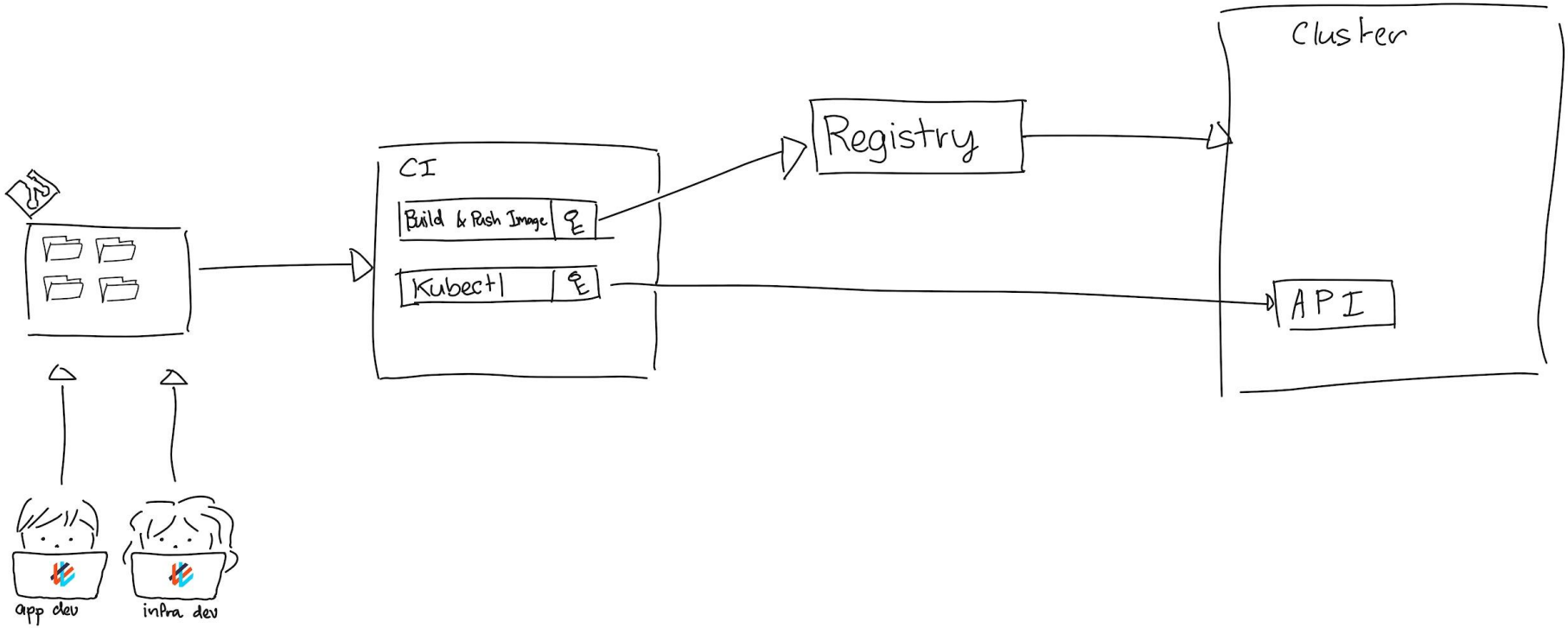
Deploying the same ‘standard platform’ using a platform-as-code approach. Whether on-premise or in the cloud. Deploying and managing multiple different Kubernetes clusters.

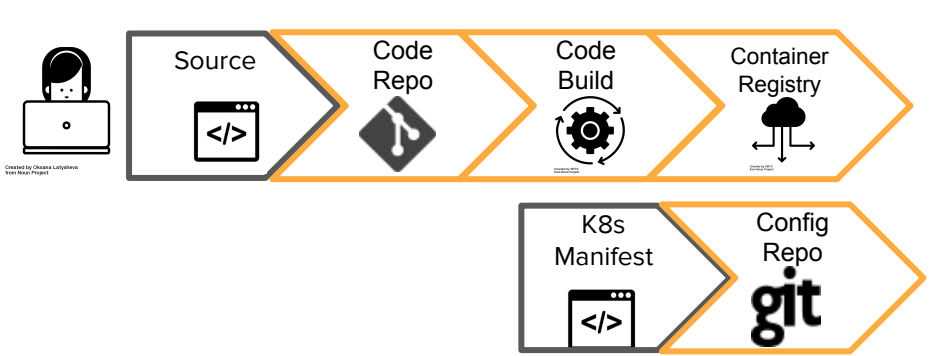
3. Security and controls

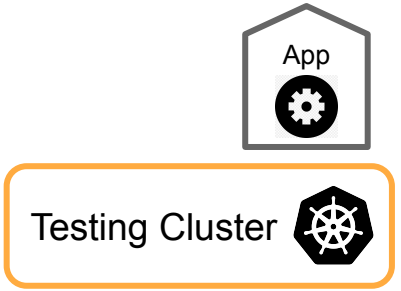
Using GitOps-based security controls to secure the deployment for both Operators and DevOps teams.

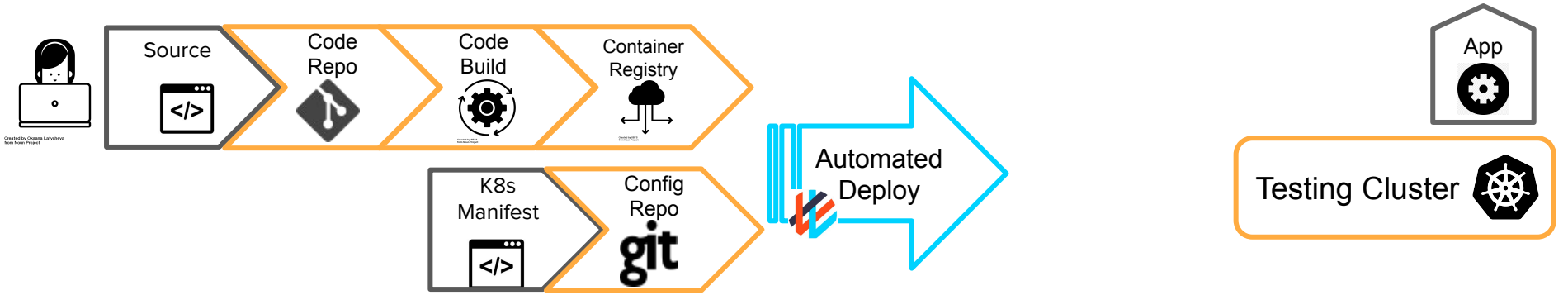


Traditional CI/CD

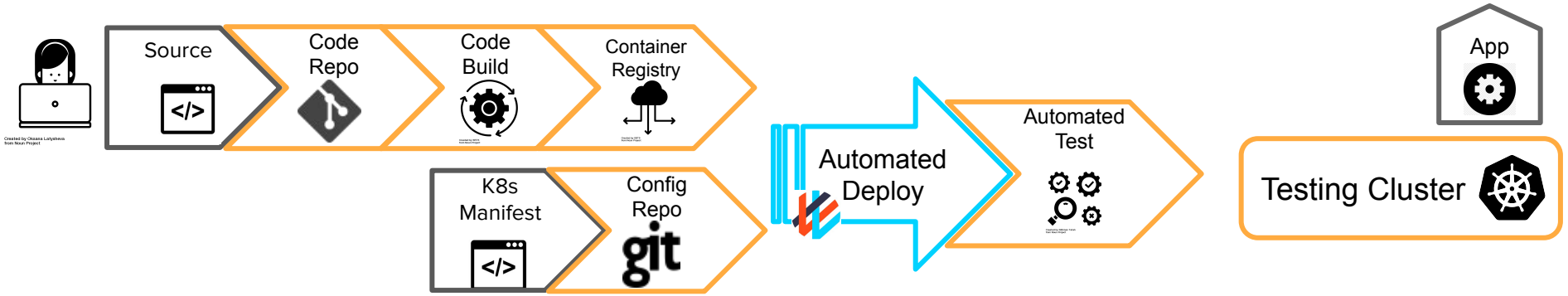



Source
Code Repo
Code Build
Container Registry
K8s Manifest
Config Repo
git

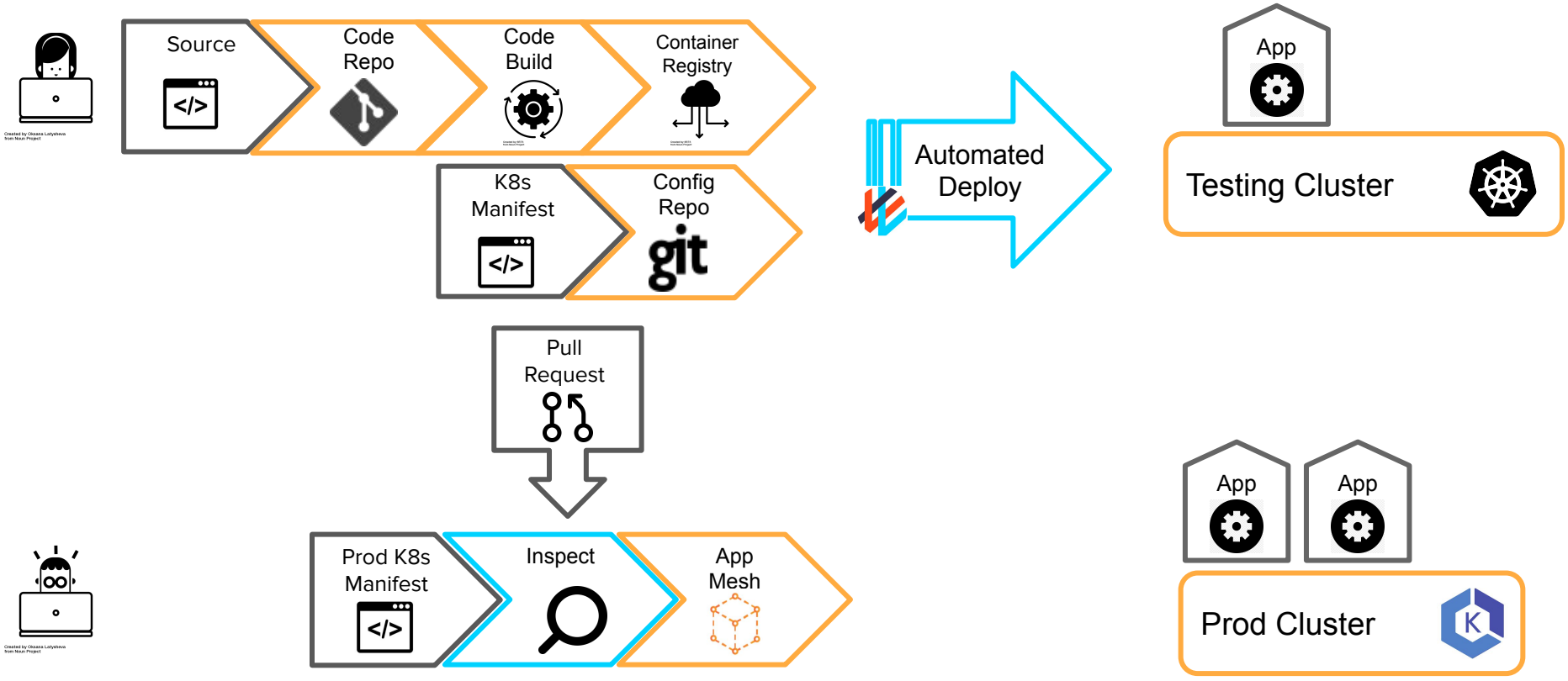

App
Testing Cluster

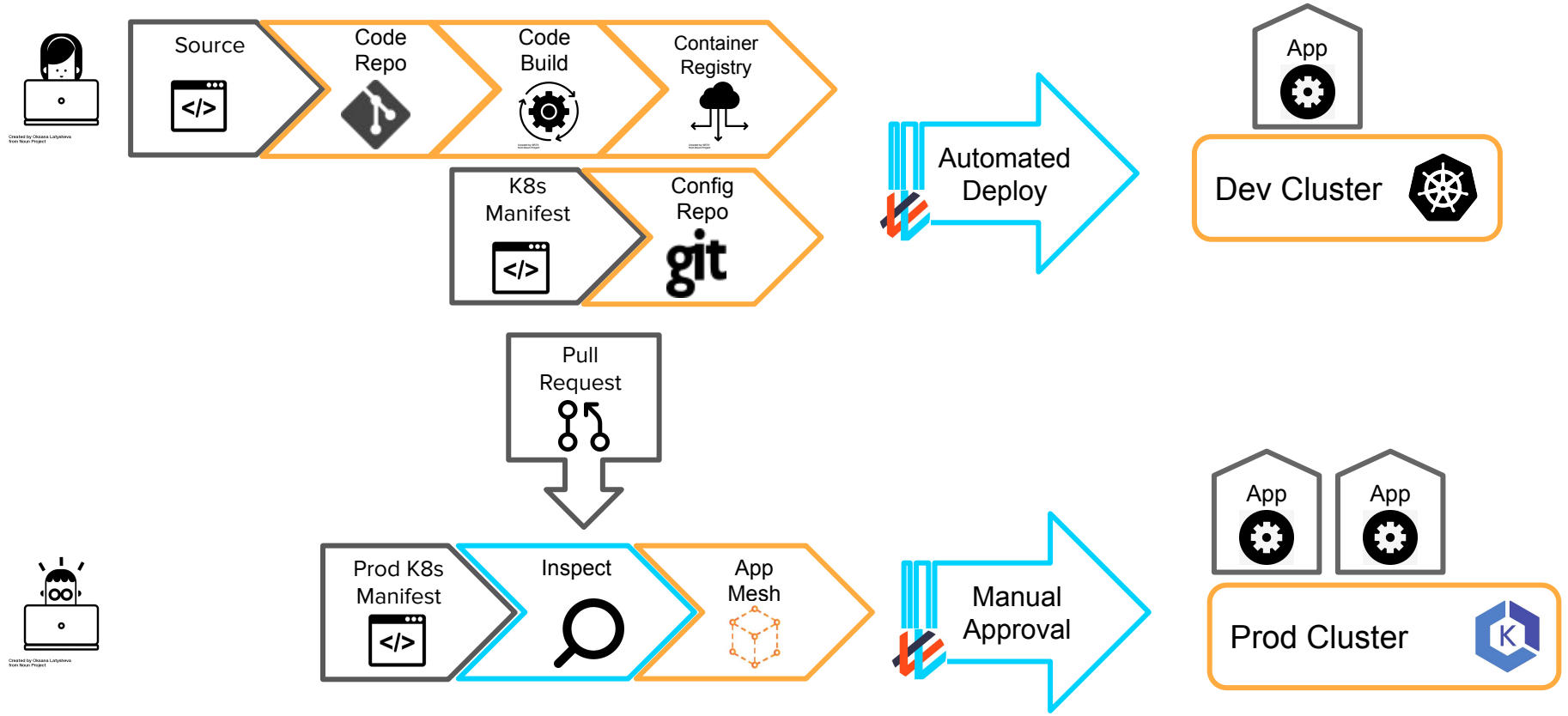


Automated deployment



Automated Testing as part of the pipeline

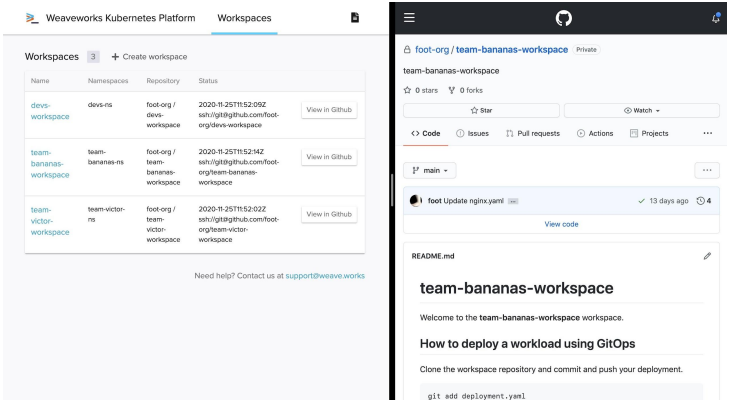

Source
Code Repo
Code Build
Container Registry
K8s Manifest
Config Repo
git
Automated Deploy
App
Testing Cluster
Pull Request
Prod K8s Manifest
Inspect
App Mesh
App
App
Prod Cluster

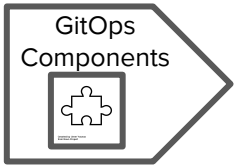

Source
Code Repo
Code Build
Container Registry
K8s Manifest
Config Repo
git
Automated Deploy
App
Dev Cluster
Pull Request
Prod K8s Manifest
Inspect
App Mesh
Manual Approval
App
App
Prod Cluster
K

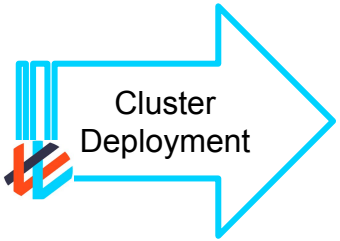
Use Case: Cluster Operations

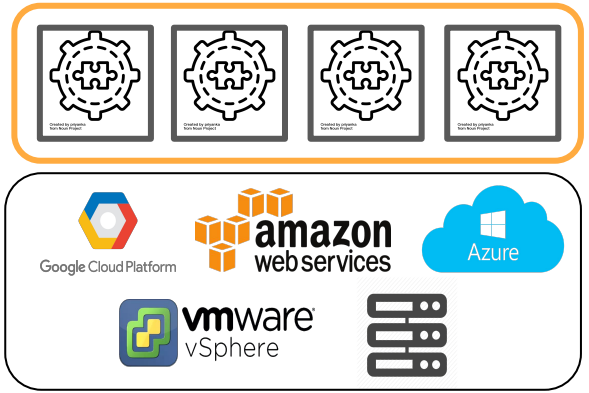
Weave GitOps Enterprise - Cluster Operations

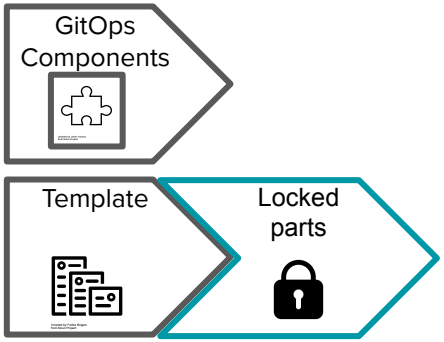
- Standard Cluster components to deploy a platform-as-code
- Multiple different K8s backends
- Templating to provide configuration and multi-cluster management

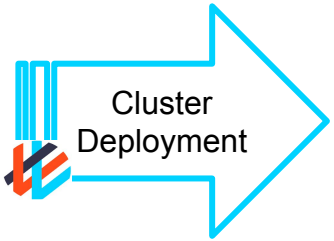


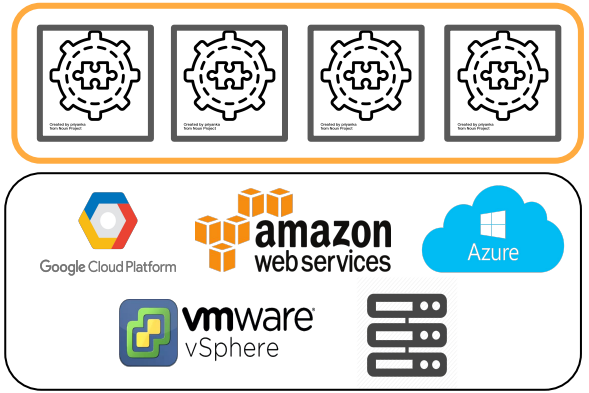

GitOps
Components

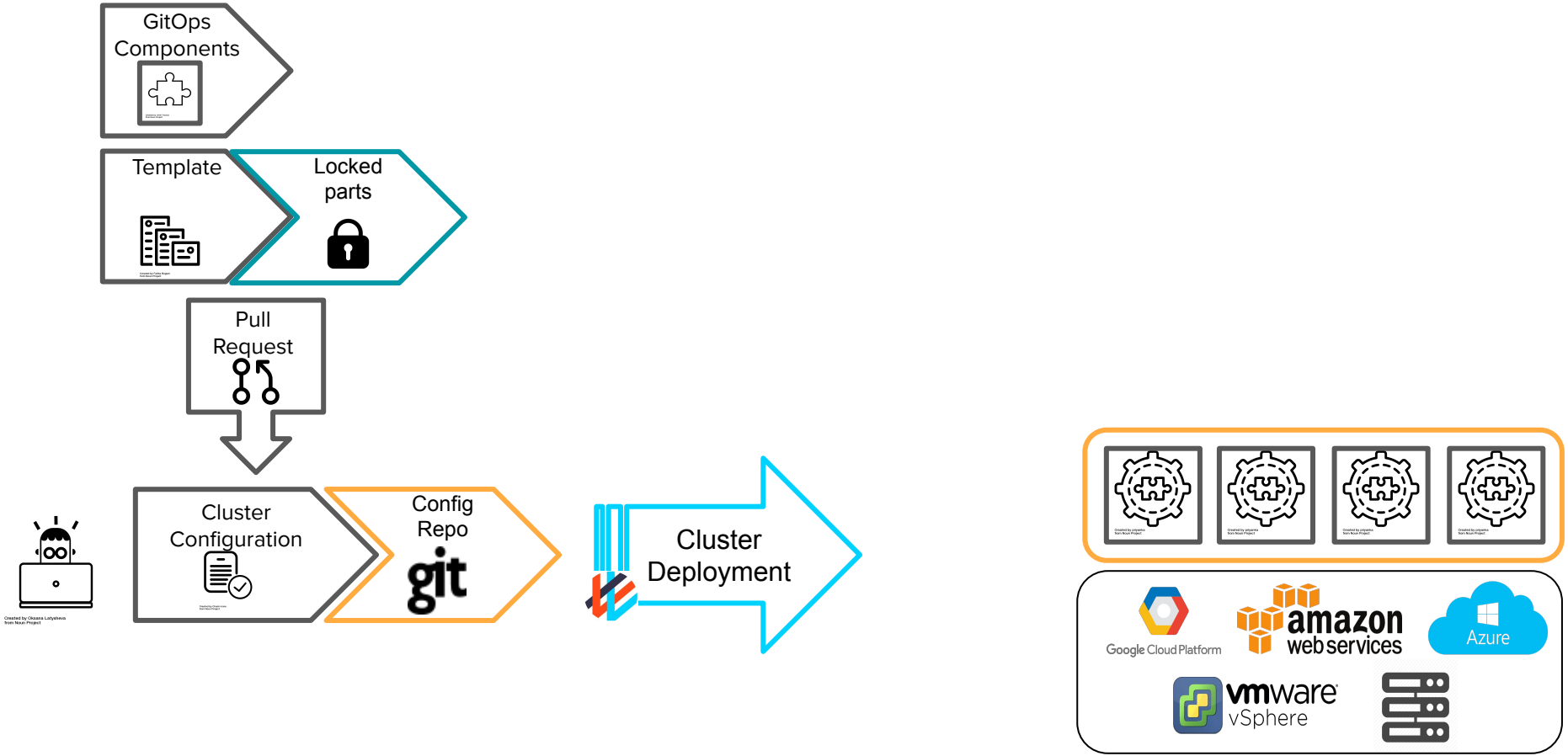

Cluster
Deployment


Google Cloud Platform
amazon
web services
Azure
vmware
vSphere


GitOps
Components
Template
Locked
parts


Cluster
Deployment


Google Cloud Platform
amazon
web services
Azure
vmware
vSphere


GitOps Components
Template
Locked parts
Pull Request
Cluster Configuration
Config Repo
git
Cluster Deployment
Google Cloud Platform
amazon web services
Azure
vmware vSphere

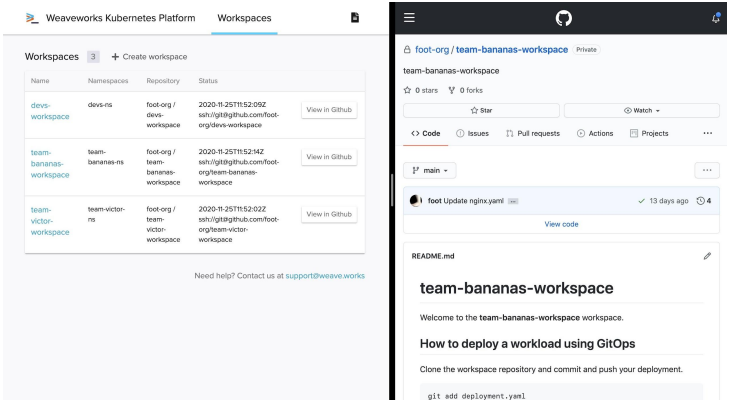
Use Case: Security and Controls

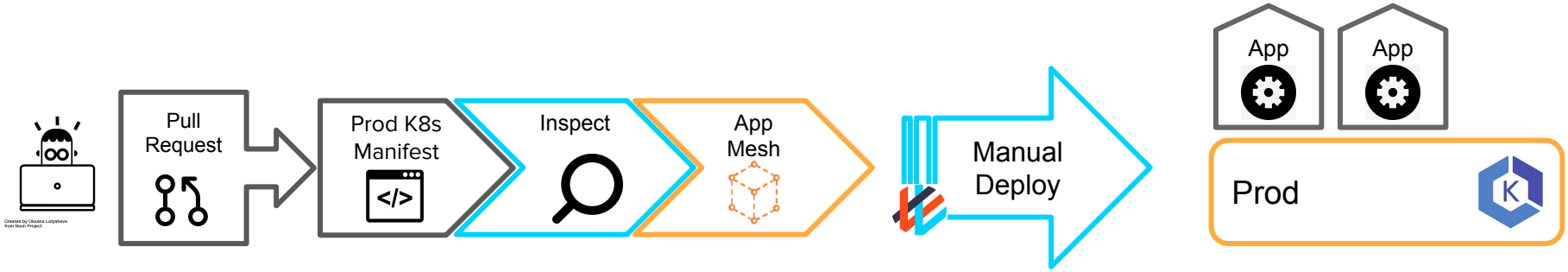
Security and controls

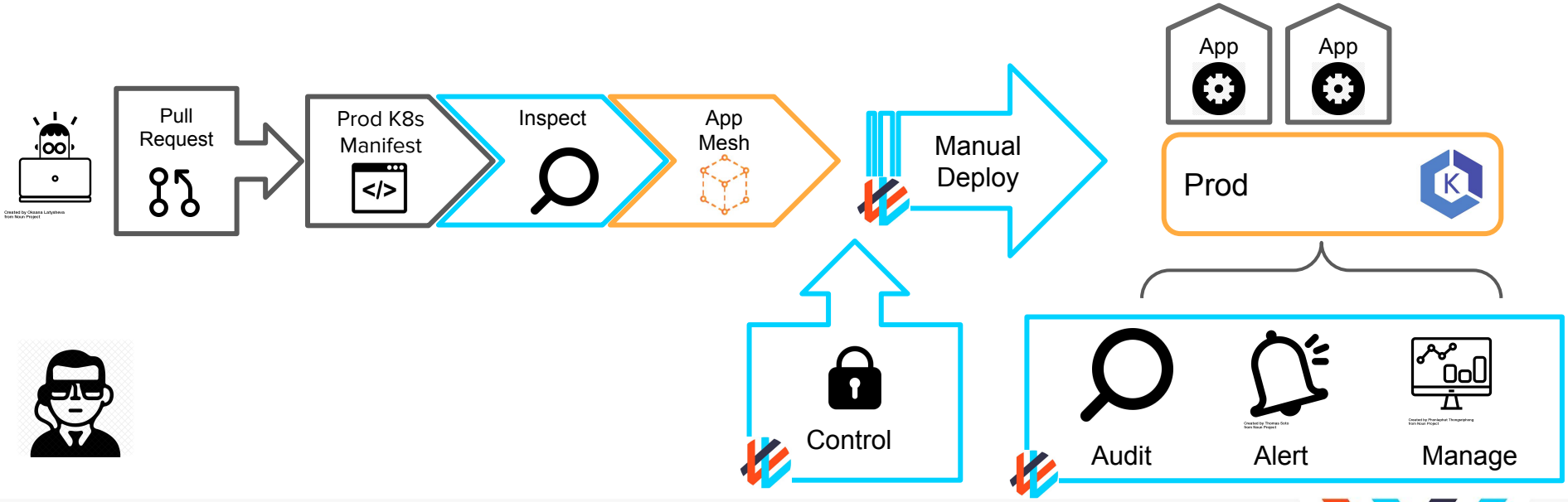
Using GitOps-based security controls to secure the deployment for both Operators and DevOps teams.

Adding additional guard rails to prevent bad deployments.

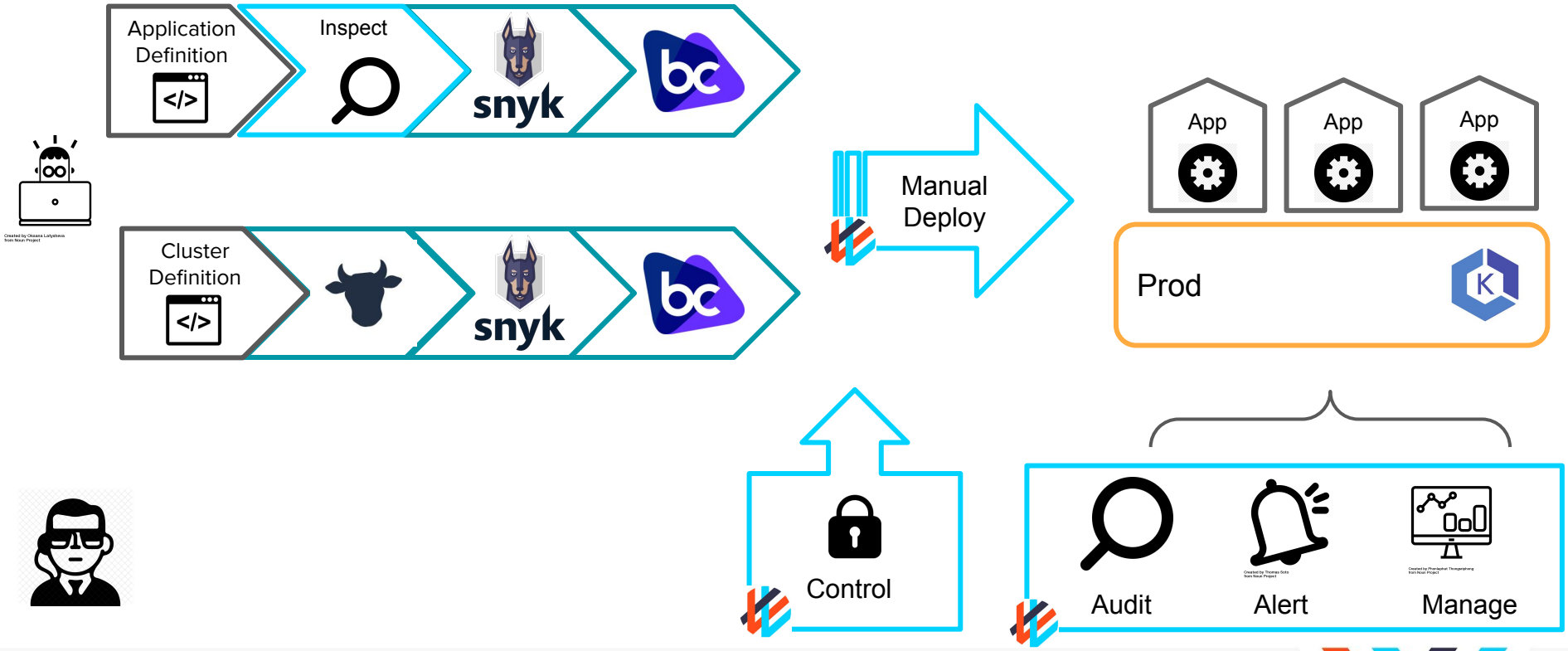
Operating and inspecting applications in production.



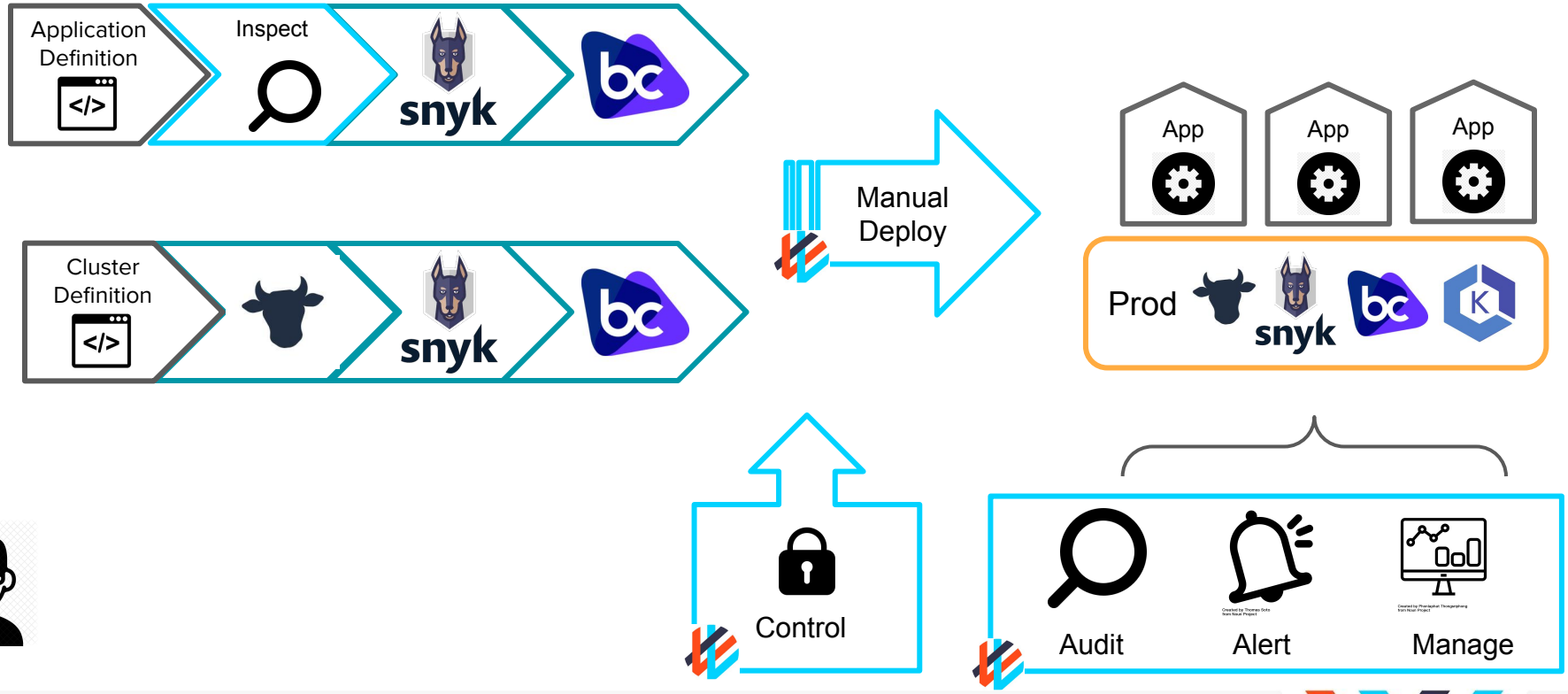

Pull
Request
Prod K8s
Manifest
</>
Inspect
App
Mesh
Manual
Deploy
App
App
Prod
K


Pull Request
Prod K8s Manifest
Inspect
App Mesh
Manual Deploy
App
App
Prod
Control
Audit
Alert
Manage

Cluster and Application Control



Cluster and Application Control



GitOps Maturity Model - Summary