

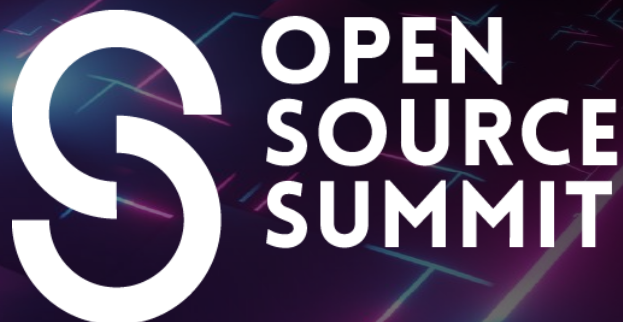


KubeCon



CloudNativeCon

THE LINUX FOUNDATION



AI_dev
Open Source GenAI & ML Summit

China 2024



KubeCon



CloudNativeCon



China 2024

Unlocking Scalability and Simplifying Multi-Cloud Management with Karmada and PipeCD

Hongcai Ren, Huawei, Karmada Community
Khanh Tran, CyberAgent, PipeCD Community

Agenda



China 2024

- Why multi-cluster
- Introduction of Karmada
- Introduction of PipeCD
- Karmada x PipeCD Integration demo

Agenda



China 2024

- **Why multi-cluster**
- Introduction of Karmada
- Introduction of PipeCD
- Karmada x PipeCD Integration demo

Why multi-cluster



China 2024

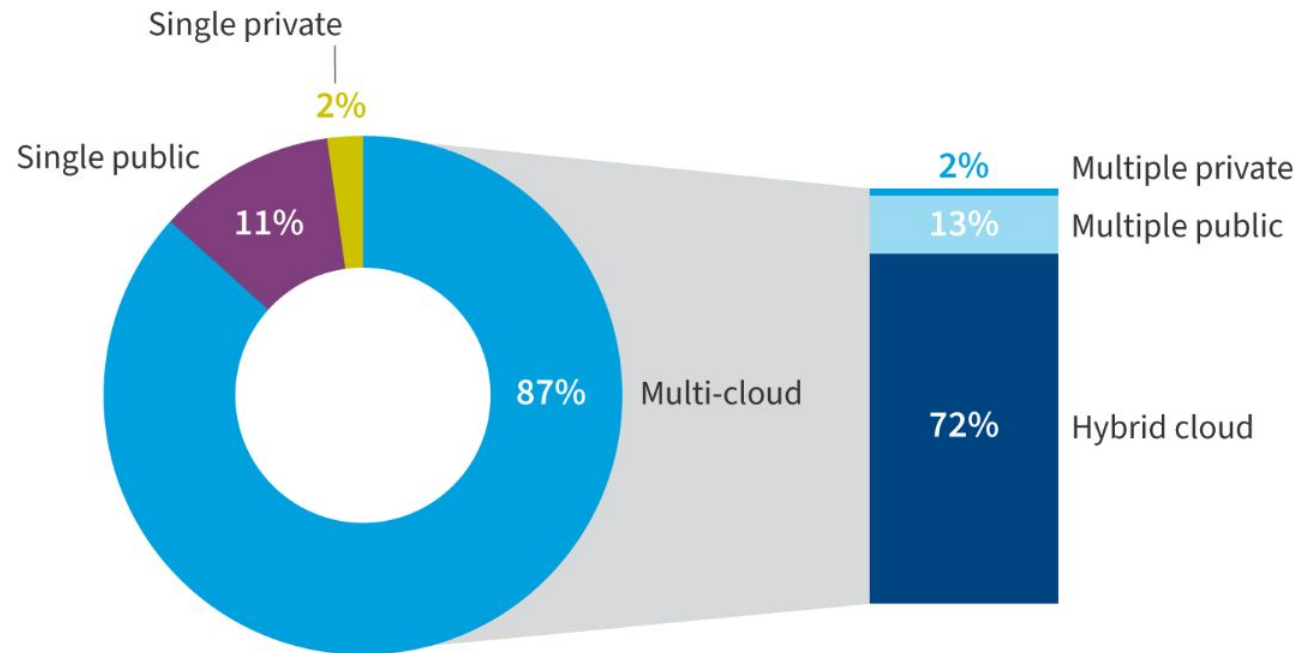
- Location
 - Latency: Deploy apps as close to the customers as possible
 - Data locality: Keep user data in-country
- Isolation
 - Environment (e.g. dev, test, prod)
 - Security isolation: sensitive data must be isolated
 - Organization isolation: Teams have different domains
- Reliability
 - Blast radius: Apps incident in one cluster must not impact the whole system
 - Scale: The apps is too big to fit in a single cluster
- And so on ...

Multi-Cloud and Multi-Cluster Deployment Has Become a Common Practice



China 2024

Organizations embrace multi-cloud



- More than 87% of enterprise respondents are using the services of multiple cloud vendors at the same time.
- Cloud native technologies and the cloud market are maturing, and an era of programmable multi-cloud management services is coming.

Challenges of Being Cloud-Native Multi-Cloud



China 2024

Challenges to multi-cloud container cluster management

Numerous clusters

Complex and repeated cluster configurations

Cluster management varies from vendor to vendor

Fragmented API access entries

Scattered services

Application configuration differs across clusters

Cross-cloud service access

Application synchronization between clusters

Restrictions from clusters

Restricted resource scheduling

Restricted application availability

Restricted auto scaling

Vendor lock-in

Service deployment dependency

Lack of automatic failover

Lack of neutral open-source, multi-cluster orchestration projects

Agenda



China 2024

- Why multi-cluster
- **Introduction of Karmada**
- Introduction of PipeCD
- Karmada x PipeCD Integration demo

Karmada: Open-Source, Cloud-Native Platform for Multi-Cloud Container Orchestration



China 2024

Compatible with Kubernetes native APIs

- Upgrade from single-cluster to multi-cluster deployments without code refactoring
- Seamlessly integrated with the Kubernetes single-cluster tool chain

1

No vendor lock-in & Centralized management

- Support for multi-cloud platforms, auto resource allocation, and free migration
- Not bound to any commercial products from cloud vendors
- Support public clouds, private clouds, and edge clouds

3



2

Out-of-the-box

- Built-in policy sets for multiple scenarios, such as geo-redundancy,
- intra-city active-active, and remote DR

4

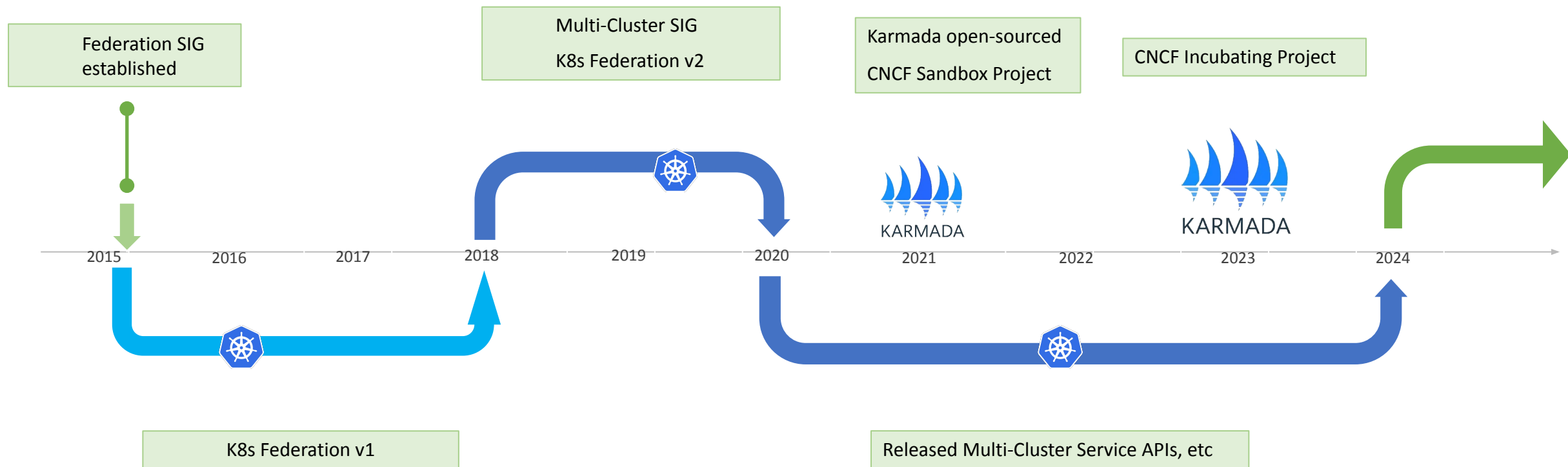
Various multi-cluster scheduling policies

- Cluster scheduling based on affinity and multi-cluster
- HA deployment across regions, AZs, clusters, and vendors

Development of Multi-Cluster Container Orchestration



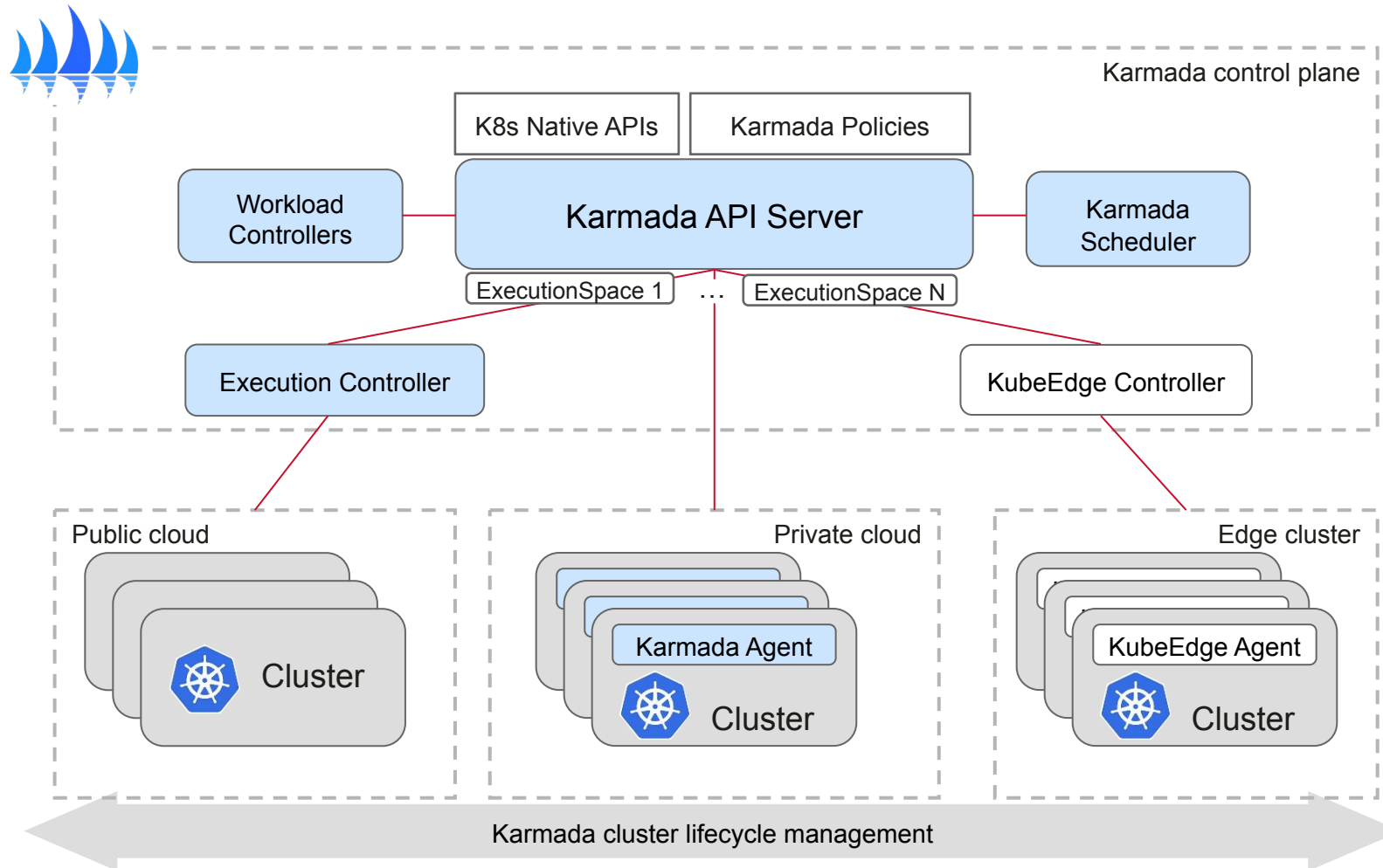
China 2024



Karmada Overview



China 2024



Core Features



China 2024

Multi-cluster management

1. Support cluster in Public cloud, on-prem or edge
2. Unified management
3. Multi-Vendor support including Huawei Cloud CCE, AWS EKS, and Other K8s clusters.
4. Push and Pull mode support
5. X86 & ARM64 support

Cross-cloud deployment

1. Use Kubernetes native API, including CRD
2. Facilitating cross-cloud application deployment, release, and operations capabilities for users.
3. Use built-in policy sets for scenarios, including: Active-active, Remote Disaster Recovery, etc

Cross-cloud App failover

Provide the ability for failover and address disaster recovery concerns in multi-cloud applications.

Centralized Management

Access cluster's resource from Karmada control plane (get/describe/logs/exec, etc)

Multi-cluster service and LB

Deploy service across cluster by leveraging MCS-API

Zero Refactoring: Using Kubernetes Native APIs to Deploy a Multi-Cluster Application



China 2024

Widely applicable propagation policy

```
apiVersion: policy.karmada.io/v1alpha1
kind: PropagationPolicy
metadata:
  name: multi-zone-replication
spec:
  resourceSelectors:
  - apiVersion: apps/v1
    kind: Deployment
    labelSelector:
      matchLabels:
        ha-mode: multi-zone-replication
  placement:
    spreadConstraints:
    - spreadByField: zone
      maxGroups: 2
      minGroups: 2
    - spreadByField: cluster
      maxGroups: 3
      minGroups: 3
```

Example policy: Configure a multi-AZ HA deployment scheme for all Deployments

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  app: nginx
  ha-mode: multi-zone-replication
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx
        ports:
        - containerPort: 80
```

Deploy apps with standard Kubernetes API definitions
kubectl create -f nginx-deployment.yaml

Agenda



China 2024

- Why multi-cluster
- Introduction of Karmada
- **Introduction of PipeCD**
- Karmada x PipeCD Integration demo

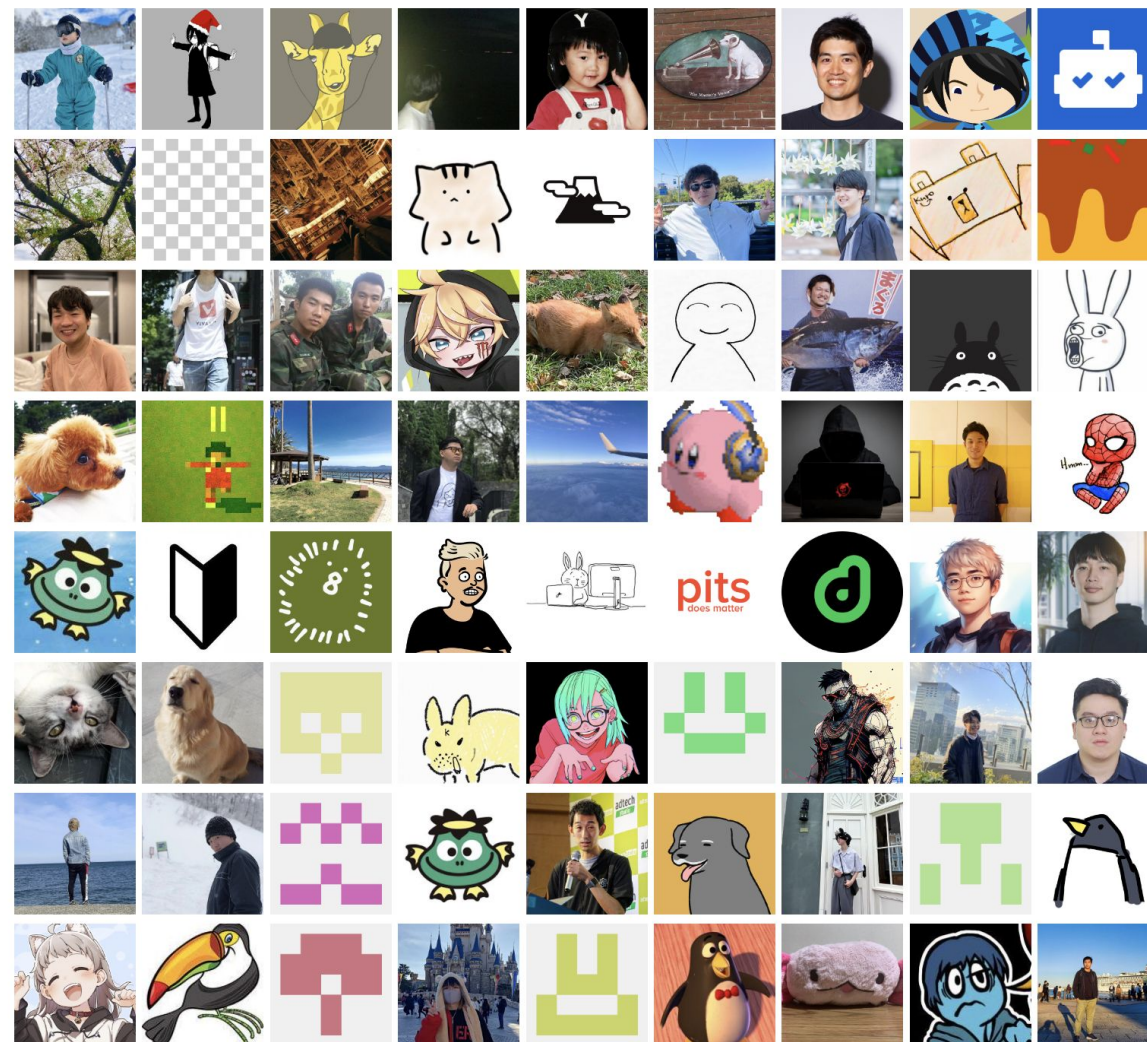
Introduction of PipeCD



China 2024



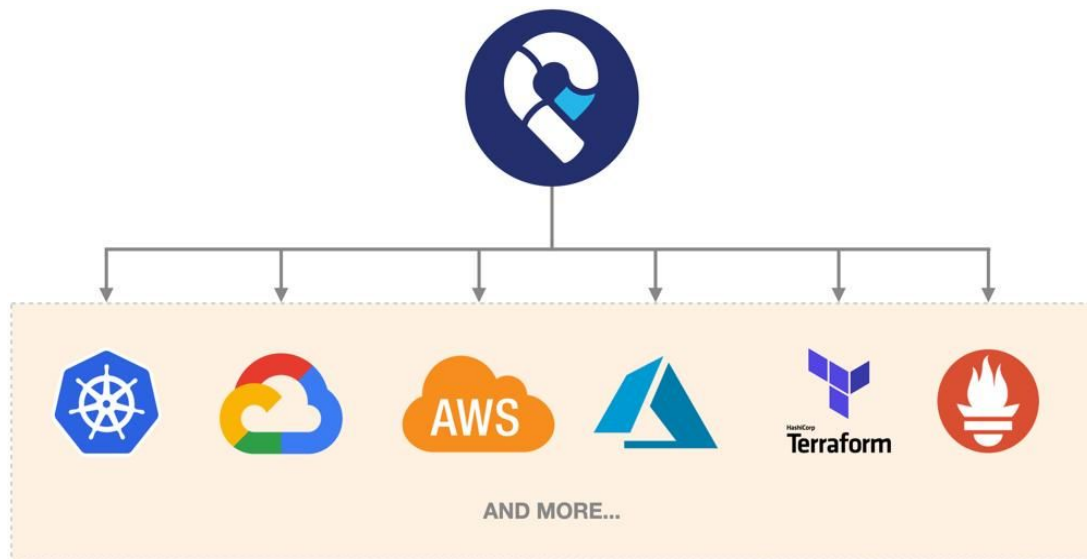
- 4000s commits / 90 contributors
- CNCF Sandbox from 2023



What is PipeCD



China 2024



The One CD for All {applications, platforms, operations}

- A GitOps style continuous delivery platform that provides consistent deployment and operations experience for any applications
- Support multi-clouds providers, multi-analysis providers
- Built-in secret management/attachment for manifests manipulating
- Event-based communication mechanism to work with whatever CI
- **And it works for not just Kubernetes workloads but other as well**

Build your own Progressive Delivery Pipeline

- Built In support for Progressive Delivery (Canary, Blue/Green...)
- Automatically analyze, rollout with metrics

FAILURE nakabonne-dev 4 months ago

Failed while executing stage stage-1

Application podinfo-2

Piped nakabonne-cluster

Summary Sync progressively because of updating image podinfo from 4.0.2 to 4.0.0

Commit Update to 4.0.0 (2f8ed99)

Triggered by nakabonne

K8S_CANARY_ROLLOUT ANALYSIS K8S_PRIMARY_ROLLOUT K8S_CANARY_CLEAN ROLLBACK

ANALYSIS

```
1 [2020-11-24 10:43:38 +09:00] Preparing deploy source at running commit (ae32a73ec9b24470e83b9ec8d263de6936ca34d6)
2 [2020-11-24 10:43:40 +09:00] Successfully cloned the running commit ae32a73ec9b24470e83b9ec8d263de6936ca34d6 of the repository delivery
3 [2020-11-24 10:43:40 +09:00] Successfully loaded the deployment configuration file
4 [2020-11-24 10:43:40 +09:00] Successfully prepared deploy source at running commit (ae32a73ec9b24470e83b9ec8d263de6936ca34d6)
5 [2020-11-24 10:43:40 +09:00] [metrics-2] Start analysis for Prometheus
6 [2020-11-24 10:43:40 +09:00] [metrics-0] Start analysis for Prometheus
7 [2020-11-24 10:43:40 +09:00] [metrics-1] Start analysis for Prometheus
8 [2020-11-24 10:48:40 +09:00] [metrics-1] Run query against Prometheus: "sum without(path, status) (rate(http_request_duration_seconds_sum(job=\"podinfo-canary\") [5m]))
    /
    sum without(path, status) (rate(http_request_duration_seconds_count(job=\"podinfo-canary\") [5m]))
    /
9 [2020-11-24 10:48:40 +09:00] [metrics-0] Run query against Prometheus: "rate(http_requests_total(status=~\"5.*\", job=\"podinfo-canary\") [5m])
    /
    rate(http_requests_total(job=\"podinfo-canary\") [5m])"
```

Zero Refactoring: Using your Kubernetes manifests to perform Progressive Delivery



China 2024

```
1  apiVersion: pipecd.dev/v1beta1
2  kind: KubernetesApp
3  spec:
4    name: wait-approval
5    labels:
6      env: example
7      team: product
8    pipeline:
9      stages:
10       - name: K8S_CANARY_ROLLOUT
11         with:
12           replicas: 10%
13       - name: WAIT_APPROVAL
14         with:
15           approvers:
16             - khanhtc1202
17       - name: K8S_PRIMARY_ROLLOUT
18       - name: K8S_CANARY_CLEAN
```

▼ wait-approval

app.pipecd.yaml

deployment.yaml

service.yaml



✓ K8S_CANARY_ROLLOUT

✓ WAIT_APPROVAL
Approved by nghialv

✓ K8S_PRIMARY_ROLLOUT

✓ K8S_CANARY_CLEAN

Zero Refactoring: Using your Kubernetes manifests to perform Progressive Delivery



China 2024

```
1  apiVersion: pipecd.dev/v1beta1
2  kind: KubernetesApp
3  spec:
4    name: wait-approval
5    labels:
6      env: example
7      team: product
8    pipeline:
9      stages:
10       - name: K8S_CANARY_ROLLOUT
11         with:
12           replicas: 10%
13       - name: WAIT_APPROVAL
14         with:
15           approvers:
16             - khanhtc1202
17       - name: K8S_PRIMARY_ROLLOUT
18       - name: K8S_CANARY_CLEAN
```

▼ wait-approval

app.pipecd.yaml

deployment.yaml

service.yaml



 kustomize.io



✓ K8S_CANARY_ROLLOUT

✓ WAIT_APPROVAL
Approved by nghialv

✓ K8S_PRIMARY_ROLLOUT

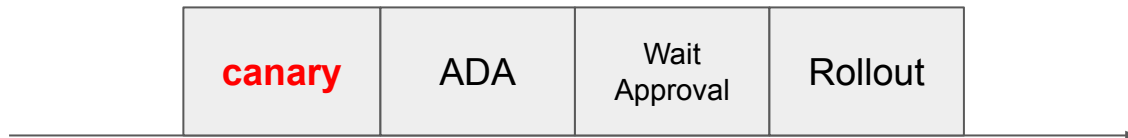
✓ K8S_CANARY_CLEAN

Less effort to Progressive Delivery



China 2024

pipecd: PIPELINE_SYNC



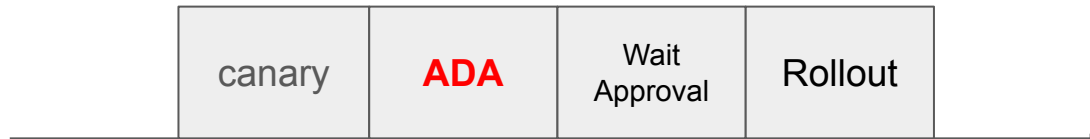
```
pipeline:
  stages:
    - name: K8S_CANARY_ROLLOUT
      with:
        replicas: 10%
    - name: K8S_BASELINE_ROLLOUT
      with:
        replicas: 10%
    - name: ANALYSIS
      with:
        duration: 15m
        metrics:
          - template:
              name: http_error_rate_canary_baseline
              appArgs:
                podNamePrefix: 
            # - template:
            #   name: http_4xx_error_rate_threshold
            #   appArgs:
            #     podNamePrefix: 
            #   threshold: "0.025"
    - name: K8S_BASELINE_CLEAN
    - name: K8S_CANARY_CLEAN
    - name: WAIT_APPROVAL
      with:
        timeout: 1h
    - name: K8S_PRIMARY_ROLLOUT
```

Less effort to Progressive Delivery



China 2024

pipecd: PIPELINE_SYNC



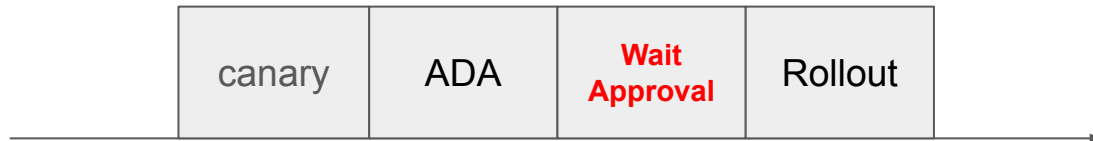
```
pipeline:
  stages:
    - name: K8S_CANARY_ROLLOUT
      with:
        replicas: 10%
    - name: K8S_BASELINE_ROLLOUT
      with:
        replicas: 10%
    - name: ANALYSIS
      with:
        duration: 15m
        metrics:
          - template:
              name: http_error_rate_canary_baseline
              appArgs:
                podNamePrefix: 
          # - template:
          #   name: http_4xx_error_rate_threshold
          #   appArgs:
          #     podNamePrefix: 
          #   threshold: "0.025"
    - name: K8S_BASELINE_CLEAN
    - name: K8S_CANARY_CLEAN
    - name: WAIT_APPROVAL
      with:
        timeout: 1h
    - name: K8S_PRIMARY_ROLLOUT
```


Less effort to Progressive Delivery



China 2024

pipecd: PIPELINE_SYNC



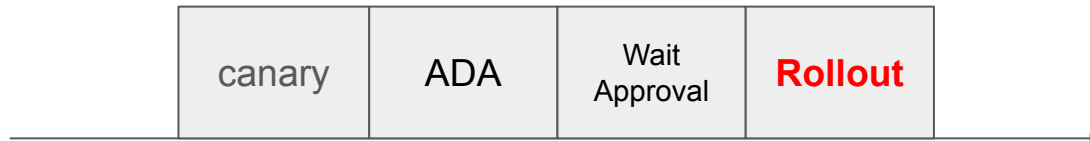
```
pipeline:
  stages:
    - name: K8S_CANARY_ROLLOUT
      with:
        replicas: 10%
    - name: K8S_BASELINE_ROLLOUT
      with:
        replicas: 10%
    - name: ANALYSIS
      with:
        duration: 15m
        metrics:
          - template:
              name: http_error_rate_canary_baseline
              appArgs:
                podNamePrefix: 
          # - template:
          #   name: http_4xx_error_rate_threshold
          #   appArgs:
          #     podNamePrefix: 
          #   threshold: "0.025"
    - name: K8S_BASELINE_CLEAN
    - name: K8S_CANARY_CLEAN
    - name: WAIT_APPROVAL
      with:
        timeout: 1h
    - name: K8S_PRIMARY_ROLLOUT
```

Less effort to Progressive Delivery



China 2024

pipecd: PIPELINE_SYNC

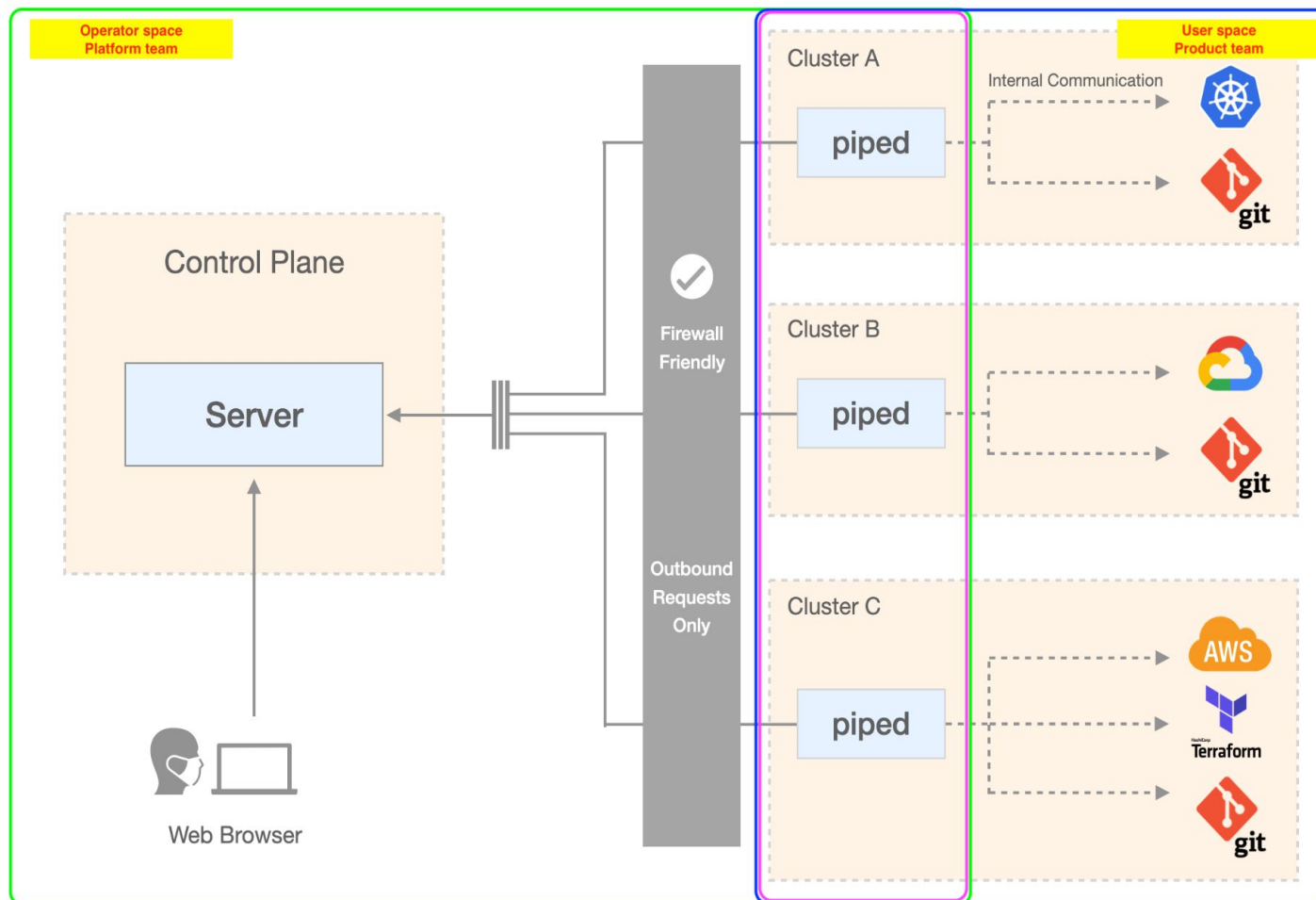


```
pipeline:
  stages:
    - name: K8S_CANARY_ROLLOUT
      with:
        replicas: 10%
    - name: K8S_BASELINE_ROLLOUT
      with:
        replicas: 10%
    - name: ANALYSIS
      with:
        duration: 15m
        metrics:
          - template:
              name: http_error_rate_canary_baseline
              appArgs:
                podNamePrefix: 
          # - template:
          #   name: http_4xx_error_rate_threshold
          #   appArgs:
          #     podNamePrefix: 
          #   threshold: "0.025"
    - name: K8S_BASELINE_CLEAN
    - name: K8S_CANARY_CLEAN
    - name: WAIT_APPROVAL
      with:
        timeout: 1h
    - name: K8S_PRIMARY_ROLLOUT
```

Why PipeCD for Multi-cluster/cloud



China 2024



Support Multi-clusters/clouds at mind

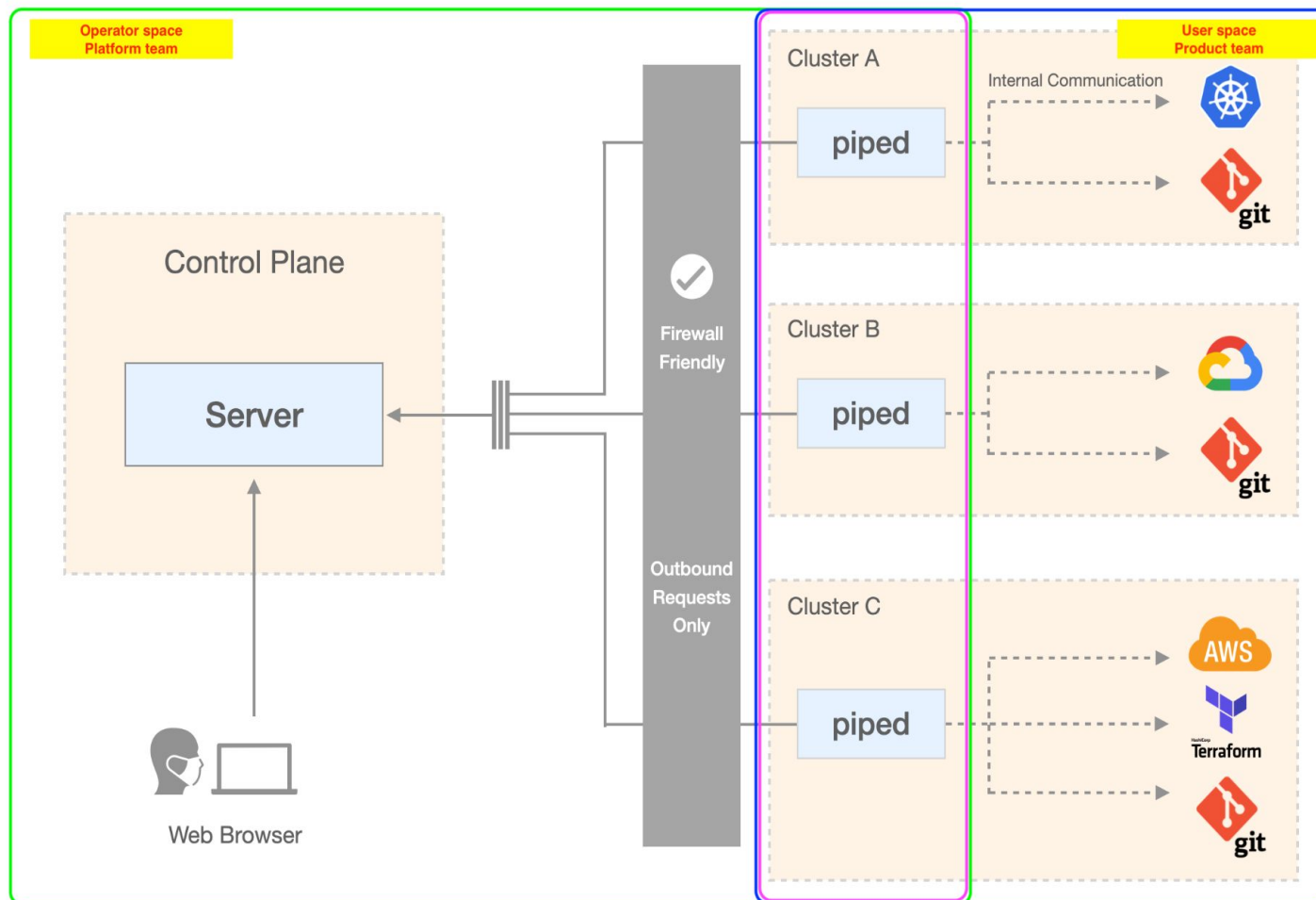
Easy to operate multi-cluster, multi-tenancy by separating control-plane and pipcd

- Single Control-Plane for centric state management
- Agents installed freely next to your applications in clusters (or not)

Why PipeCD for Multi-cluster/cloud



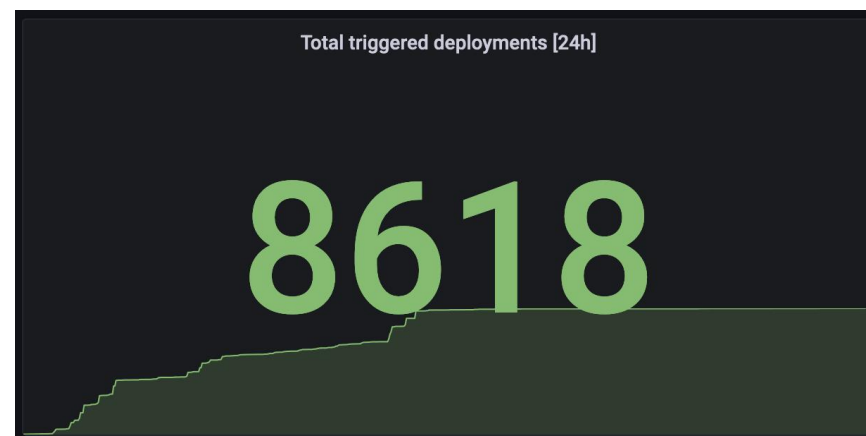
China 2024



Support Multi-clusters/clouds at mind

Easy to operate multi-cluster, multi-tenancy by separating control-plane and pipcd

- Single Control-Plane for centric state management
- Agents installed freely next to your applications in clusters (or not)

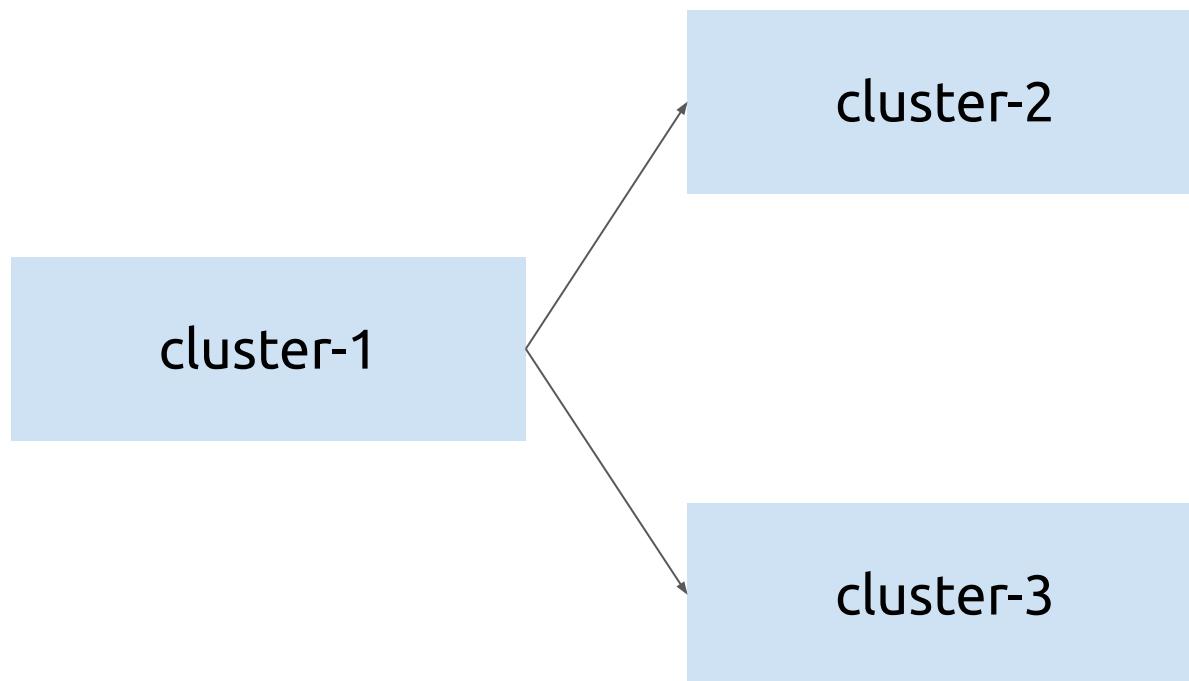


Why PipeCD for Multi-cluster/cloud



China 2024

pipecd: Deployment Chain



```
1  apiVersion: pipecd.dev/v1beta1
2  kind: KubernetesApp
3  spec:
4    name: app-1
5    labels:
6      env: cluster-1
7    pipeline:
8      stages:
9        ...
10   postSync:
11     chain:
12       applications:
13         - name: app-1
14           kind: KUBERNETES
15           labels:
16             env: cluster-2
17         - name: app-1
18           kind: KUBERNETES
19           labels:
20             env: cluster-3
21
```

Agenda



China 2024

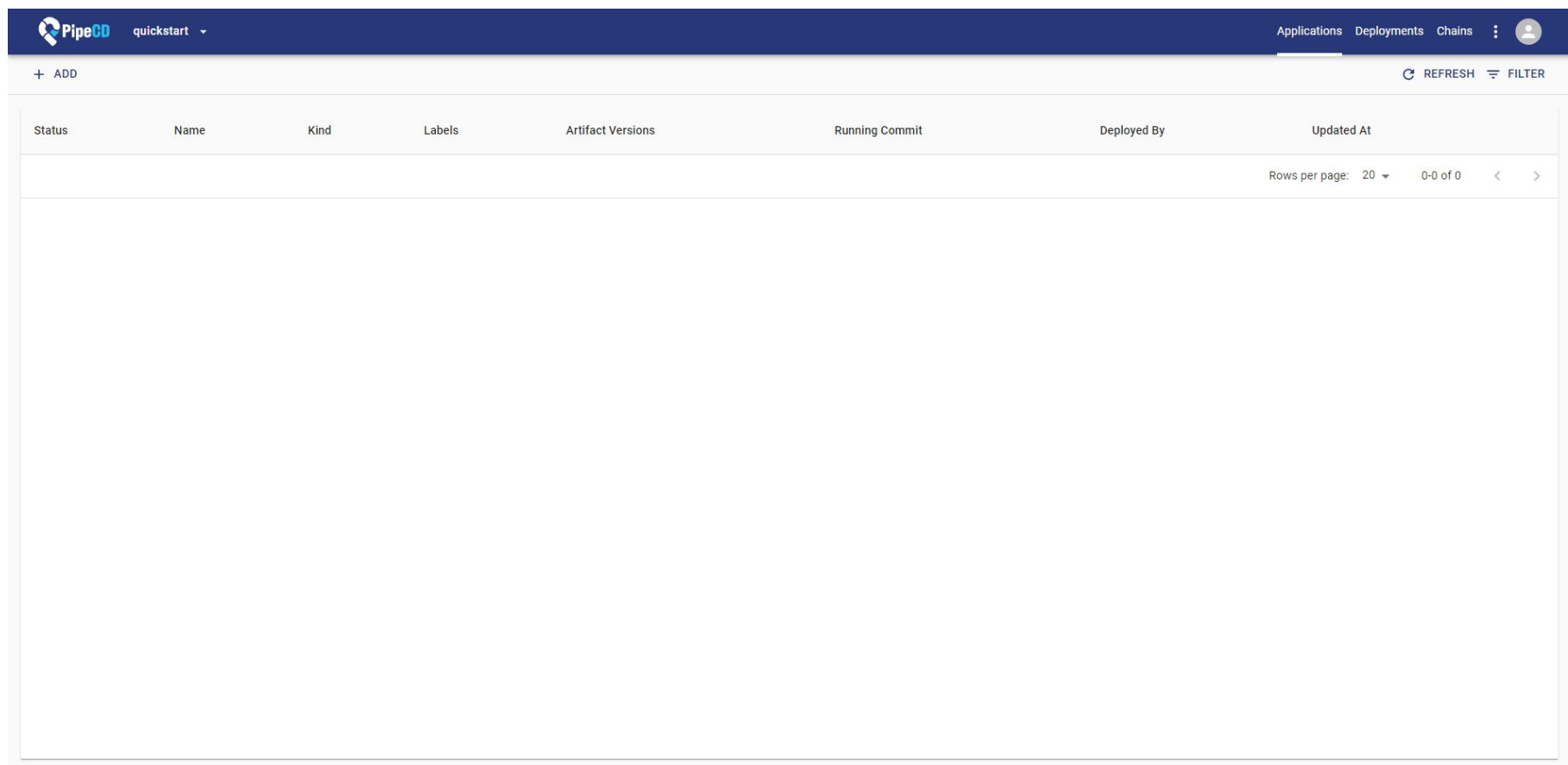
- Why multi-cluster
- Introduction of Karmada
- Introduction of PipeCD
- **Karmada x PipeCD Integration demo**

Demo of PipeCD



China 2024

Install and launch the PipeCD

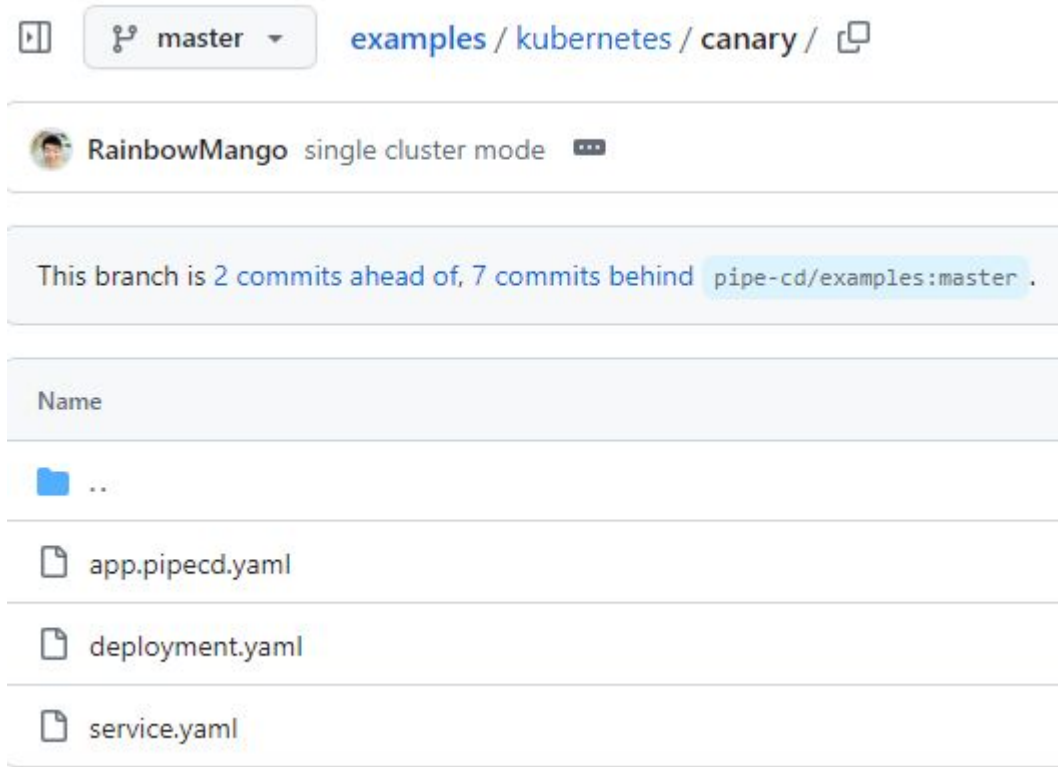


Demo of PipeCD



China 2024

Prepare application as well as PipeCD configuration

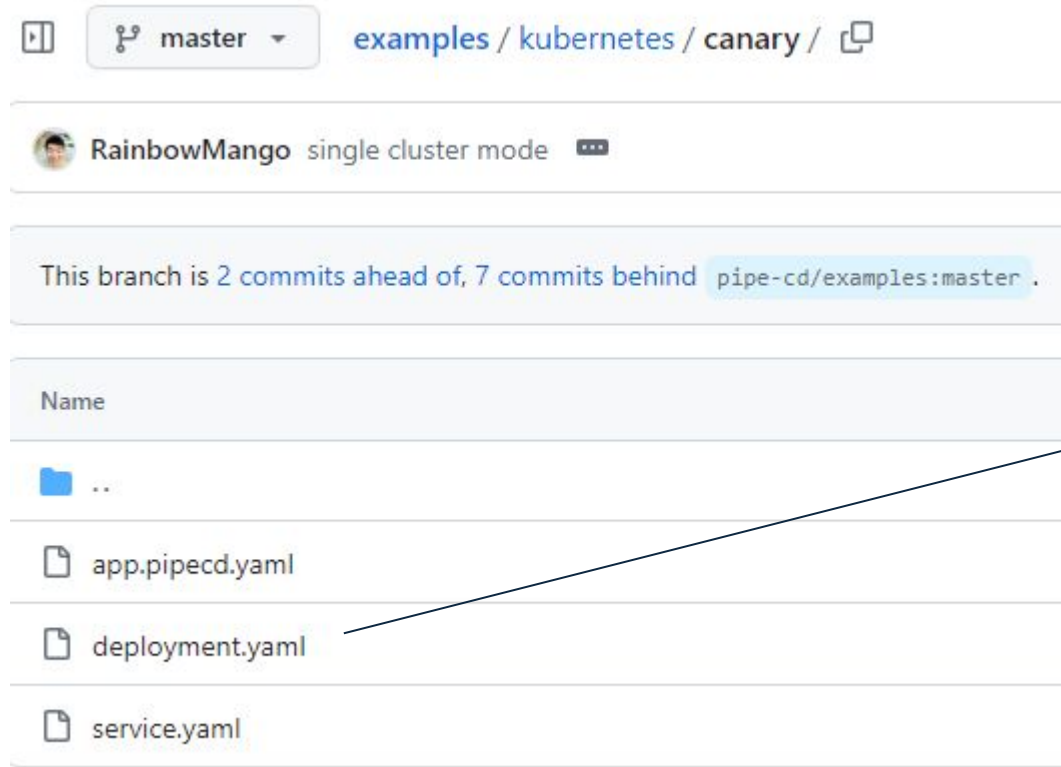


Demo of PipeCD



China 2024

Prepare application as well as PipeCD configuration



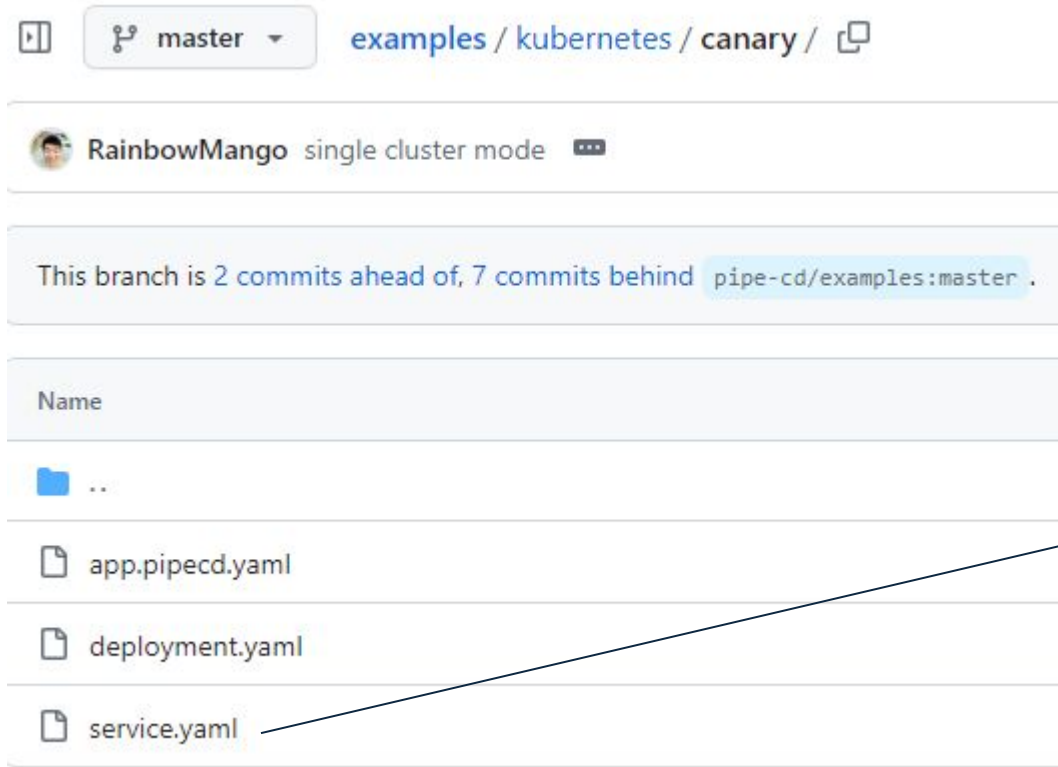
```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: canary
  labels:
    app: canary
spec:
  replicas: 2
  revisionHistoryLimit: 2
  selector:
    matchLabels:
      app: canary
      pipecd.dev/variant: primary
  template:
    metadata:
      labels:
        app: canary
        pipecd.dev/variant: primary
    spec:
      containers:
        - name: helloworld
          image: ghcr.io/pipe-cd/helloworld:v0.32.0
          args:
            - server
          ports:
            - containerPort: 9085
```

Demo of PipeCD



China 2024

Prepare application as well as PipeCD configuration



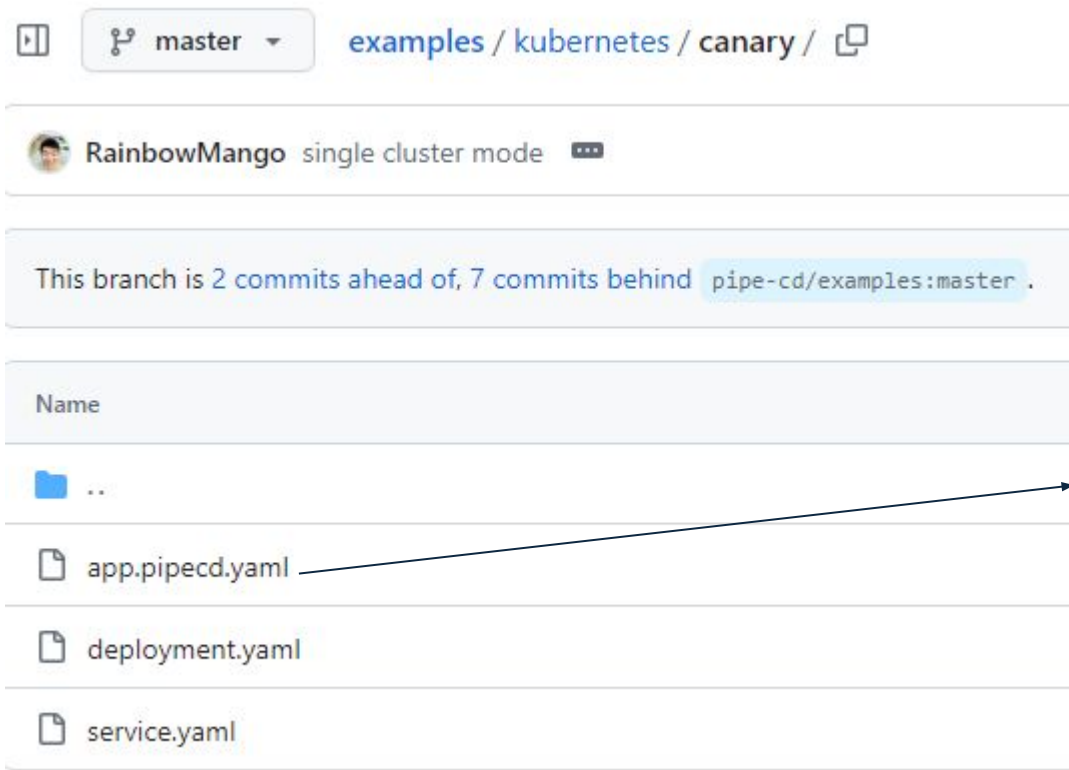
```
apiVersion: v1
kind: Service
metadata:
  name: canary
spec:
  selector:
    app: canary
  ports:
    - protocol: TCP
      port: 9085
      targetPort: 9085
```

Demo of PipeCD



China 2024

Prepare application as well as PipeCD configuration



```
apiVersion: pipedcd.dev/v1beta1
kind: KubernetesApp
spec:
  name: canary
  labels:
    env: example
    team: product
  pipeline:
    stages:
      # Deploy the workloads of CANARY variant. In this case, the number of
      # workload replicas of CANARY variant is 10% of the replicas number of PRIMARY variant.
      - name: K8S_CANARY_ROLLOUT
        with:
          replicas: 10%
      # Wait 10 seconds before going to the next stage.
      - name: WAIT
        with:
          duration: 10s
      # Update the workload of PRIMARY variant to the new version.
      - name: K8S_PRIMARY_ROLLOUT
      # Destroy all workloads of CANARY variant.
      - name: K8S_CANARY_CLEAN
```

Demo of PipeCD



China 2024

Deploy application to a single cluster

quickstart

+ ADD

Status	Name	Kind	Labels	Artifact Versions	Running Commit
--------	------	------	--------	-------------------	----------------

ADD FROM SUGGESTIONS ?ADD MANUALLY

Add a new application to "quickstart" project

✓ Select piped and platform provider

Piped
horen (504e9ee7-7f2d-4ad...

Platform Provider
kubernetes-default

✓ Select application to add

Application
name: canary, repo: examples

3 Confirm information before adding

Kind
KUBERNETES

Path
kubernetes/canary

Config Filename
app.piped.yaml

Label 0
env: example

Label 1
team: product

SAVE

CANCEL

Demo of PipeCD + Karmada



China 2024

Add Karmada configuration

📁 master [examples / kubernetes / canary /](#)

RainbowMango add Karmada propagation policy

This branch is 1 commit ahead of, 7 commits behind pipe-cd/examples:master .

Name
..
app.piecd.yaml
deployment.yaml
propagationpolicy.yaml
service.yaml

```
apiVersion: policy.karmada.io/v1alpha1
kind: PropagationPolicy
metadata:
  name: canary
spec:
  resourceSelectors:
    - apiVersion: apps/v1
      kind: Deployment
      name: canary
    - apiVersion: v1
      kind: Service
      name: canary
  placement:
    clusterAffinity:
      clusterNames:
        - member1
        - member2
  replicaScheduling:
    replicaSchedulingType: Divided
    replicaDivisionPreference: Weighted
    weightPreference:
      staticWeightList:
        - targetCluster:
            clusterNames:
              - member1
            weight: 1
        - targetCluster:
            clusterNames:
              - member2
            weight: 1
```

Demo of PipeCD + Karmada



China 2024

Deploy application to Karmada

PipeCD quickstart

+ ADD

Status	Name	Kind	Labels	Artifact Versions	Running Commit	Deployed By	Up
--------	------	------	--------	-------------------	----------------	-------------	----

Rows per page: 20

ADD FROM SUGGESTIONS ?

ADD MANUALLY

Add a new application to "quickstart" project

✓ Select piped and platform provider

Piped
horen (504e9ee7-7f2d-4ad...

Platform Provider
karmada-dev

✓ Select application to add

Application
name: canary, repo: examples

3 Confirm information before adding

Kind
KUBERNETES

Path
kubernetes/canary

Config Filename
app.piecd.yaml

Label 0
env: example

Label 1
team: product

SAVE

CANCEL

Demo of PipeCD + Karmada



China 2024

Show status of applications

quickstart ▾

Applications | Deployments | Chains |

canary env: example team: product ↻ SYNC ▾

✓ Synced Healthy

Application ID654ed105-eaf5-41a8-81bd-6864cffdb618

KindKUBERNETES

Pipedhoren

Platform Providerkarmada-dev

Configuration Directorykubernetes/canary

Latest Deployment [more details](#)

Deployed At2 minutes ago

Artifact Versionshelloworld:v0.32.0

SummaryQuick sync by applying all manifests

This app demonstrates how to deploy a Kubernetes app by Canary strategy without requiring any mesh.
References: [adding a new app](#), [app configuration](#)

PropagationPolicy
 canary

Service
 canary

Deployment
 canary

ResourceBinding
 canary-service

ResourceBinding
 canary-deployment

FILTER

Demo of PipeCD + Karmada



KubeCon



CloudNativeCon



China 2024

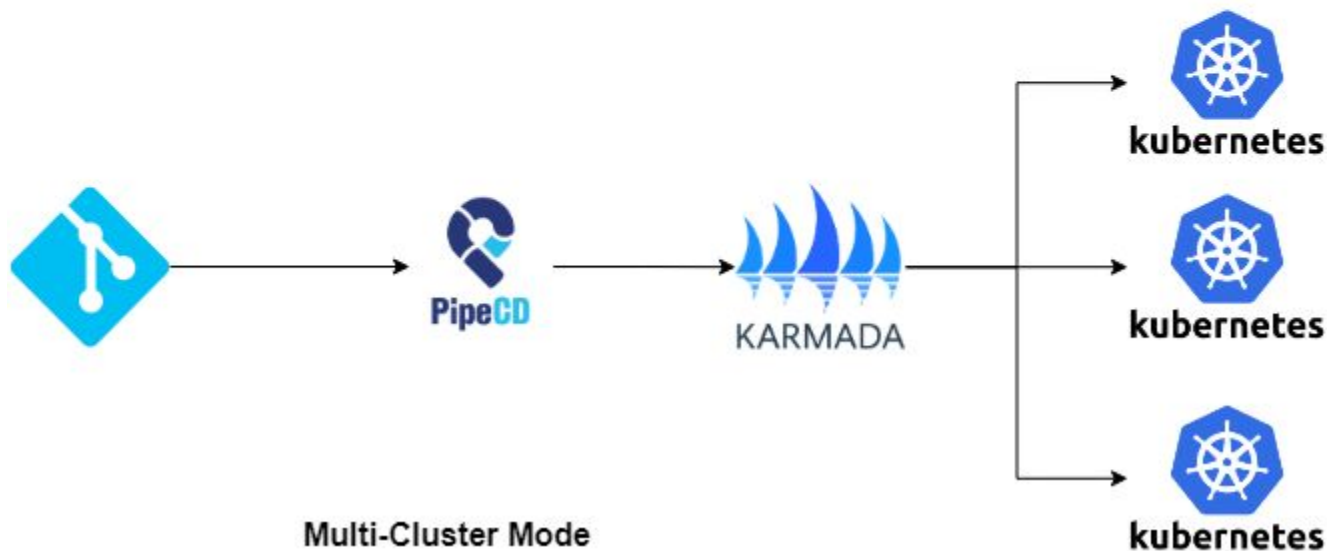


AI_dev

Single Cluster Mode



Multi-Cluster Mode



Take Away



China 2024

- PipeCD, a GitOps tool that enables deploy application by pull request on Git
 - Easy to manage / operate for both dev & operator
 - Manage multi applications across multiple clusters
 - Same simple but powerful interface for many kinds of applications (not just Kubernetes)
- Karmada, multi-cluster container orchestration platform
 - Speaks Kubernetes API
 - Easy to integrate with Kubernetes ecosystem tool-chain
 - Manage application across multiple clusters

Join Us



China 2024



<https://karmada.io>



<https://github.com/karmada-io/karmada>



<https://slack.cncf.io> (#karmada)



<https://pipecd.dev>



<https://github.com/pipe-cd/pipecd>



<https://slack.cncf.io> (#pipecd)



KubeCon



CloudNativeCon



China 2024

Thanks!