



Mastering Kubernetes Workflows and Deployments with the Argo Suite

Shipped 2024

Kostis Kapelonis | November 2024

Kostis Kapelonis



Developer Advocate (Octopus Deploy/Codefresh)

Argo Maintainer (Argo CD, Argo Rollouts)

Co-author GitOps certification

<http://learning.codefresh.io>



Topics

- 1 Project Intro
- 2 Argo Workflows
- 3 Argo CD
- 4 Argo Rollouts
- 5 Argo Events
- 6 Use Cases



Introduction



Get More Done with Kubernetes

Open source tools for Kubernetes to run workflows, manage clusters, and do GitOps right.

[View on GitHub](#)



Trusted by



Google



ticketmaster

<https://argoproj.github.io/>



Argo CD

🐱 ⭐ 16997

Declarative continuous delivery with a fully-loaded UI.

[Learn More](#)



Argo Workflows

🐱 ⭐ 14685

Kubernetes-native workflow engine supporting DAG and step-based workflows.

[Learn More](#)

Argo Rollouts

🐱 ⭐ 2619

Advanced Kubernetes deployment strategies such as Canary and Blue-Green made easy.

[Learn More](#)

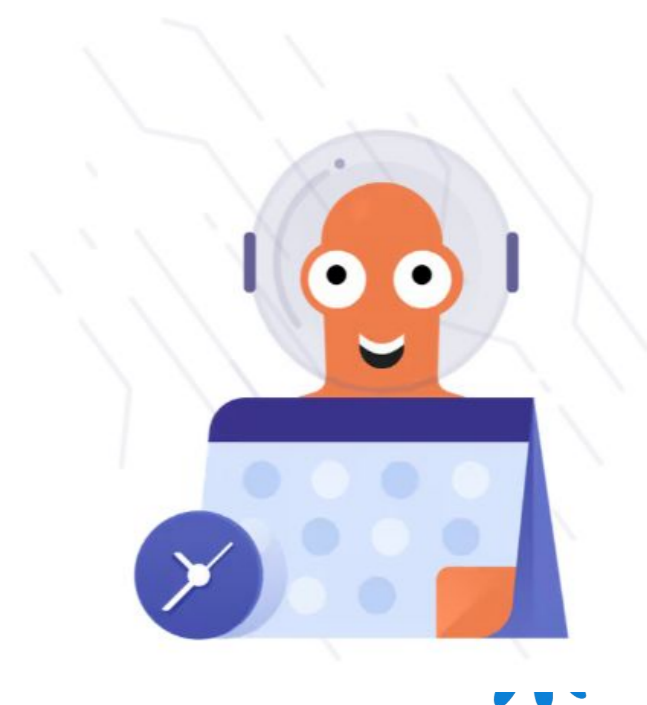


Argo Events

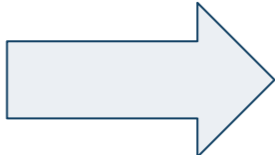
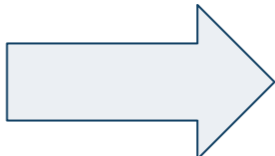
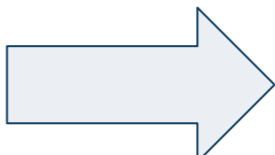

🐱 ⭐ 2299

Event based dependency management for Kubernetes.

[Learn More](#)



What the Argo Projects do

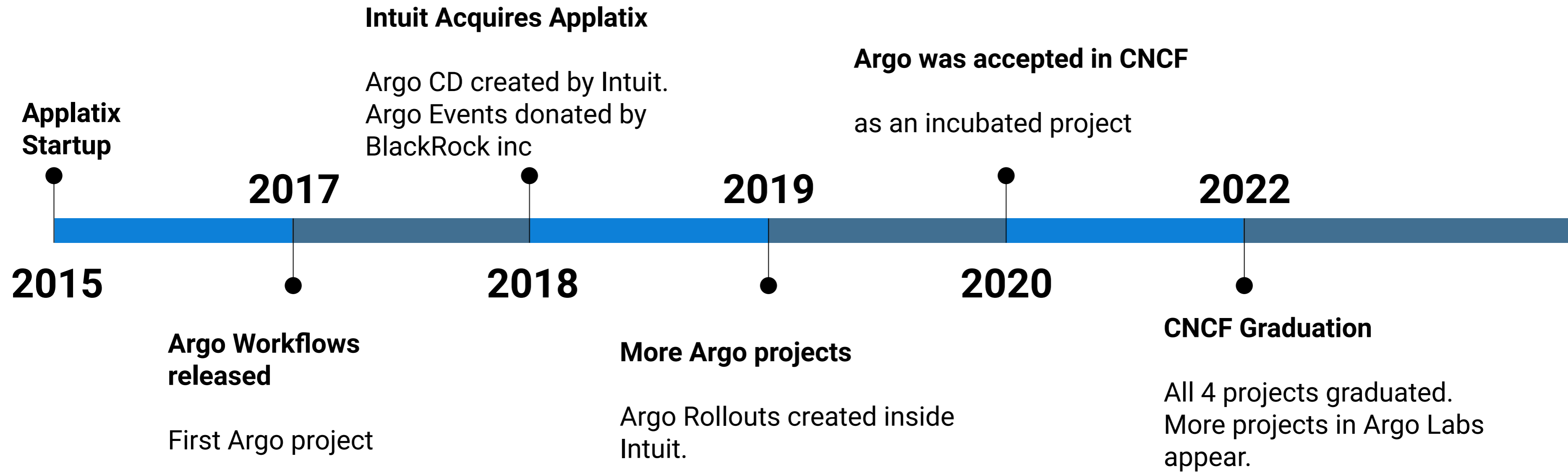
- | | | |
|----------------|---|-------------------------------|
| Argo CD |  | Deploy your App using Gitops |
| Argo Workflows |  | Execute a job/process |
| Argo Events |  | Monitor/create events |
| Argo Rollouts |  | Avoid downtime when deploying |



All 4 projects are self-contained

- There are NO dependencies between the 4 projects
- You can use each project on its own
- There are several common integrations
- Some shared code parts (e.g. notifications, SSO)
- You get extra value by combining them
- It is possible to use all 4 of them (explained later in Use cases)







argoproj-labs

README .md

argoproj-labs

This org is managed by the Argo project maintainers and not part of the CNCF Argo umbrella projects. New repos in this org need to be sponsored and created by one of the Argo project maintainers. The goal is to have a place to collaborate with the community to quickly run experiments, POCs and possibly new features to be later incorporated in one of the Argo projects.

Pinned

 [argocd-image-updater](#) Public

Automatic container image update for Argo CD

 Go  1.2k  249

 [argocd-operator](#) Public

A Kubernetes operator for managing Argo CD clusters.

 Go  612  660

 [community](#) Public

Community documents for argoproj-labs

 12  6

 [argocd-autopilot](#) Public

Argo-CD Autopilot

 Go  873  119

<https://github.com/argoproj-labs>



Created by

INTUIT

Maintained with ❤️ by:

Akuity



BlackRock

BlackRock

Codefresh



Intuit

INTUIT

Pipekit



Red Hat



[Contact us](#) to learn more about corporate maintainers.

Codefresh was acquired by Octopus Deploy in 2024

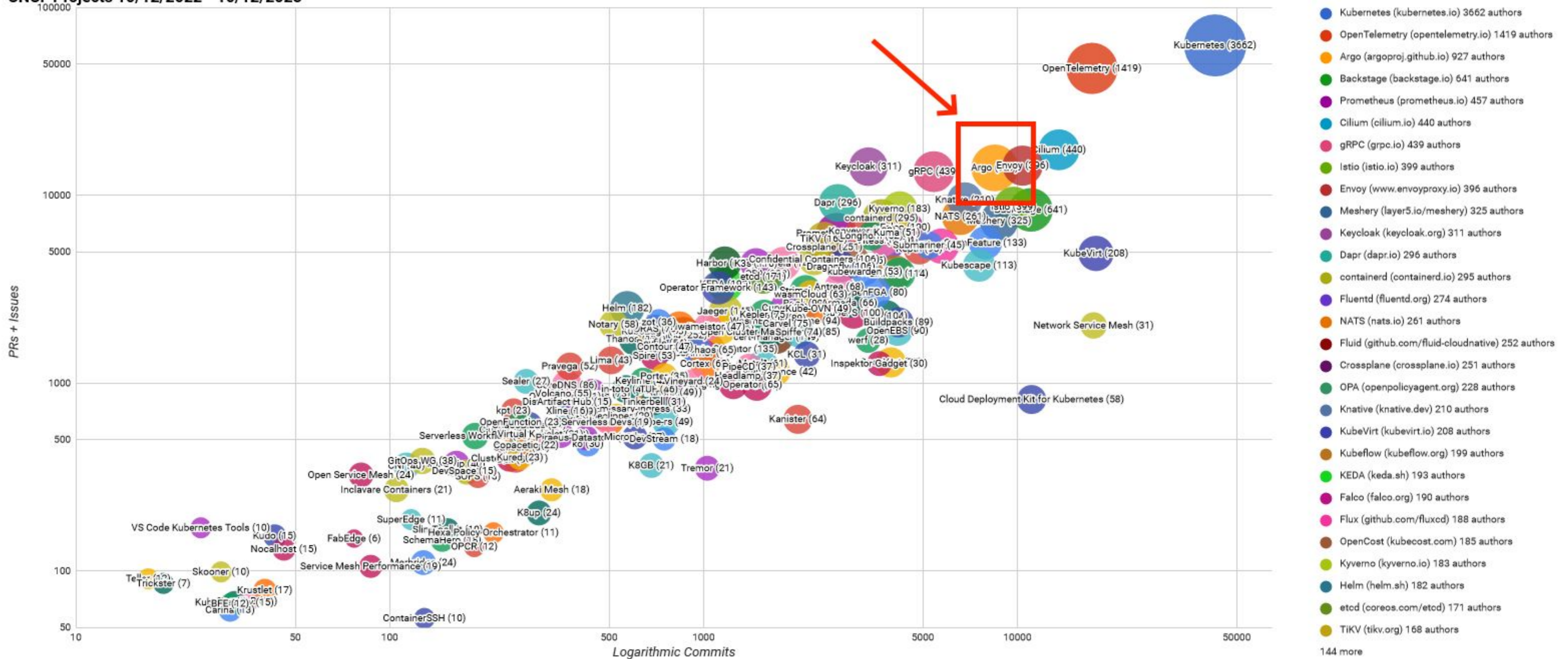


Popularity



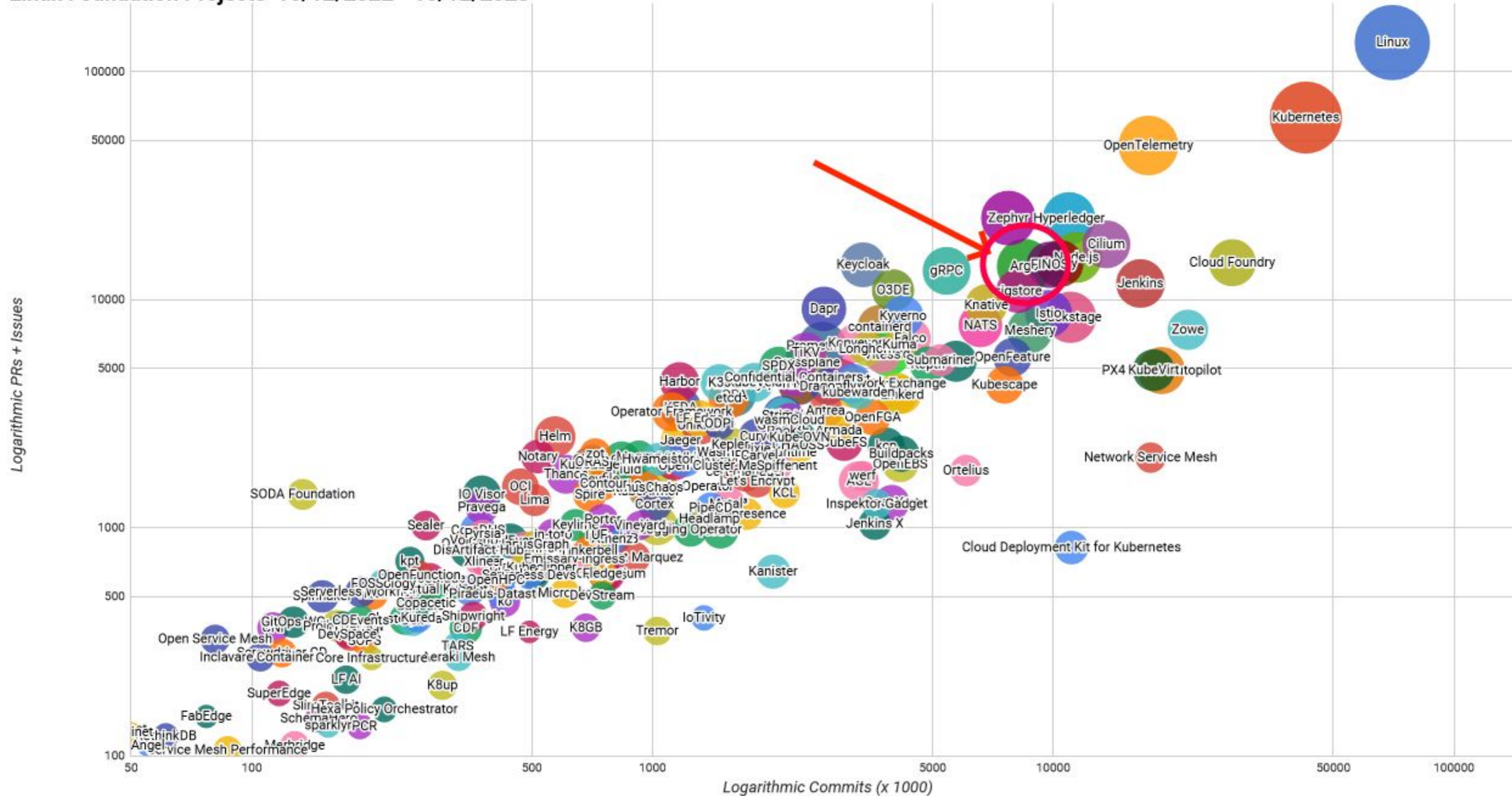
Popular/Active CNCF projects

CNCF Projects 10/12/2022 - 10/12/2023



Popular/Active Linux Foundation projects

Linux Foundation Projects 10/12/2022 - 10/12/2023



- Linux (kernel.org) 4544 authors
- Kubernetes (kubernetes.io) 3662 authors
- OpenTelemetry (opentelemetry.io) 1419 authors
- Argo (argoproj.github.io) 927 authors
- Zephyr (www.zephyrproject.org) 897 authors
- Hyperledger (hyperledger.org) 722 authors
- Backstage (backstage.io) 641 authors
- Node.js (nodejs.org) 606 authors
- Jenkins (jenkins.io) 483 authors
- Prometheus (prometheus.io) 457 authors
- Cilium (cilium.io) 440 authors
- gRPC (grpc.io) 439 authors
- Cloud Foundry (cloudfoundry.org) 400 authors
- Istio (istio.io) 399 authors
- PX4 Drone Autopilot (px4.io) 399 authors
- Envoy (www.envoyproxy.io) 396 authors
- FINOS (finos.org) 365 authors
- Meshery (layer5.io/meshery) 325 authors
- Keycloak (keycloak.org) 311 authors
- Dapr (dapr.io) 296 authors
- containerd (containerd.io) 295 authors
- Fluentd (fluentd.org) 274 authors
- sigstore (sigstore.dev) 266 authors
- NATS (nats.io) 261 authors
- Fluid (github.com/fluid-cloudnative) 252 authors
- Crossplane (crossplane.io) 251 authors
- OPA (openpolicyagent.org) 228 authors
- O3DE (o3de.org) 219 authors
- Knative (knative.dev) 210 authors
- 208 more

Argo Workflows



Argo Workflows

- The original Argo Project
- Workflows/processes
- Kubernetes native
- Alternative to Tekton, Apache Airflow
- Can be used for CI/CD, ML, ETL, Batch jobs etc



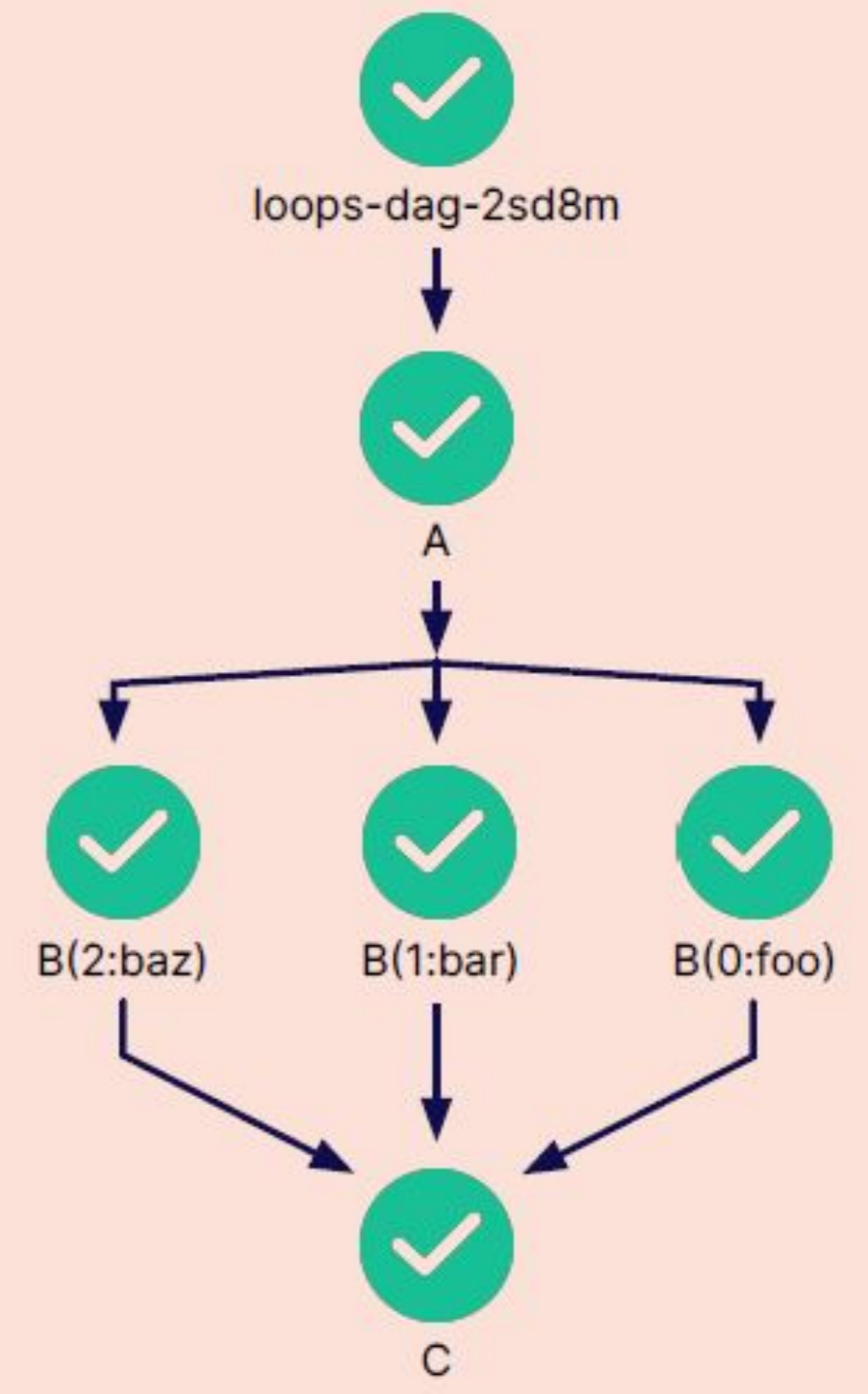
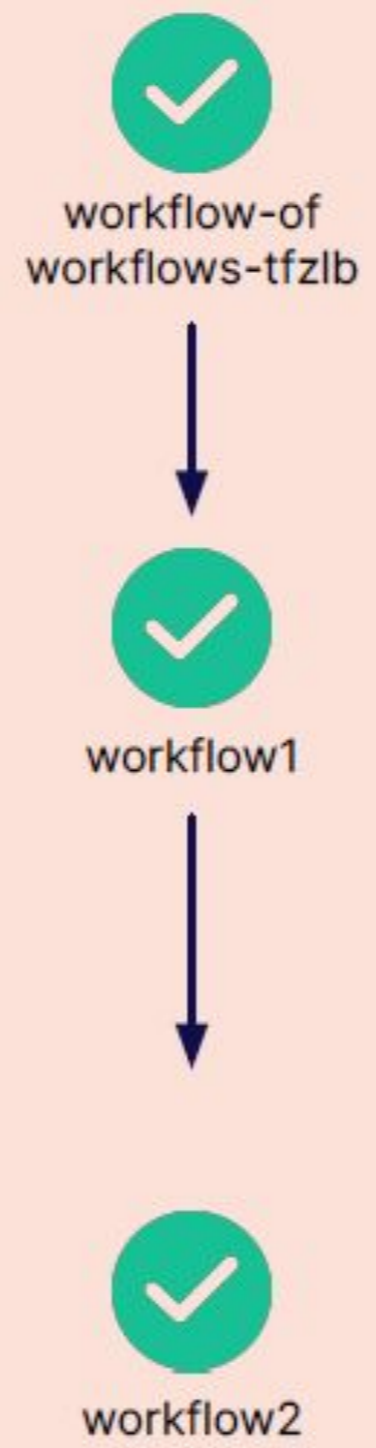
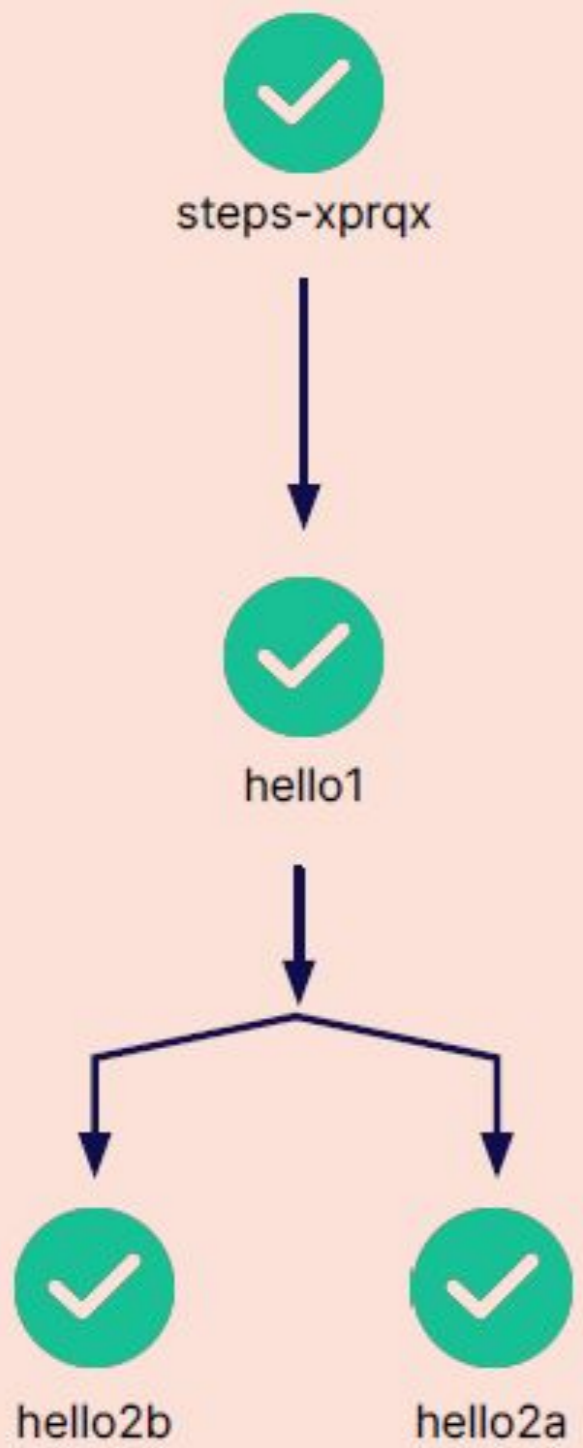
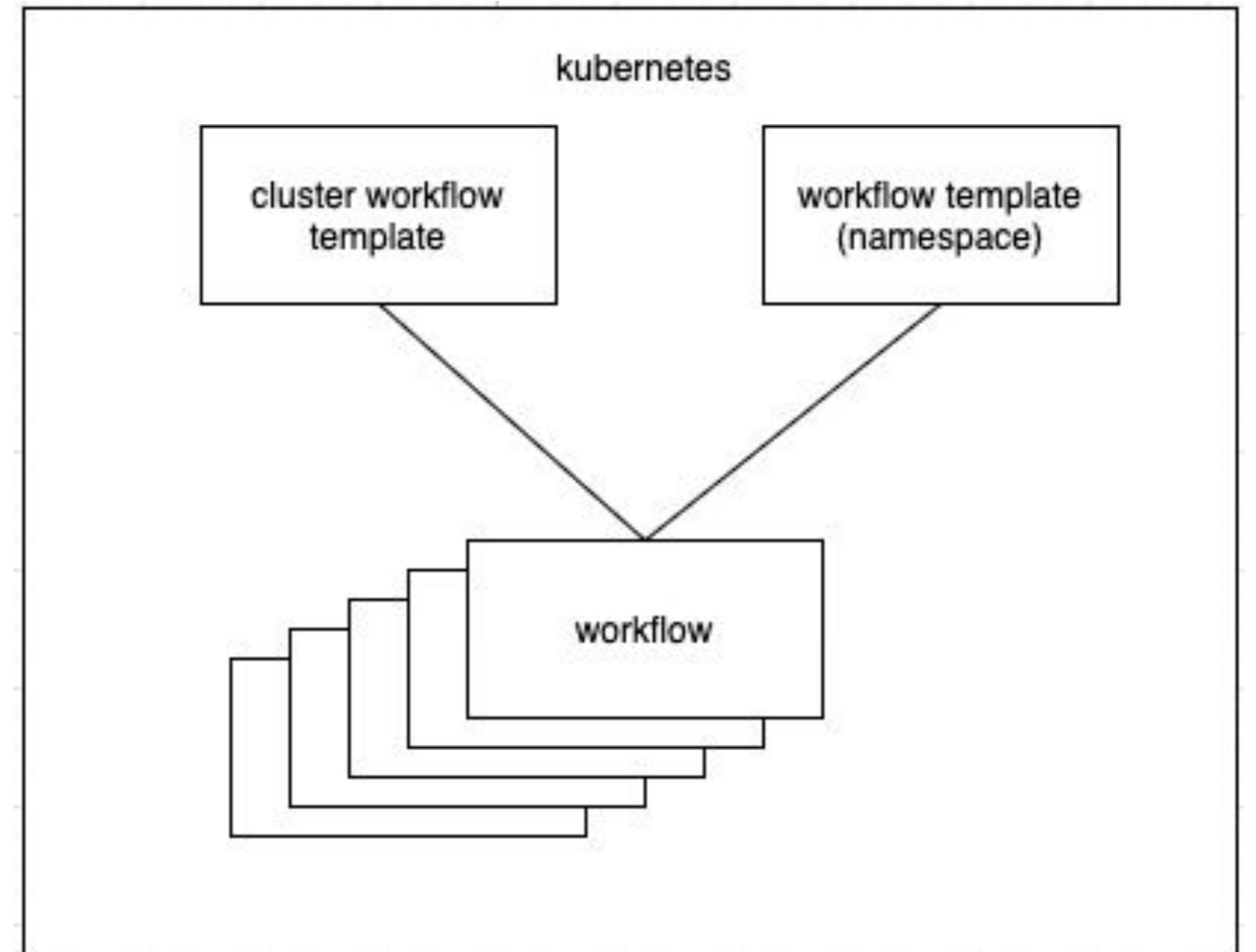


Image credit: pipekit.io



Argo Workflows entities

- **Workflow** - running instance
- **Workflow template** - definition of Workflow
- **CronWorkflows** - on a schedule
- **Cluster Workflow template** - not constrained on a single namespace



```
apiVersion: argoproj.io/v1alpha1
kind: Workflow                                # new type of k8s spec
metadata:
  generateName: hello-world-                # name of the workflow spec
spec:
  entrypoint: hello-world                   # invoke the hello-world template
  templates:
    - name: hello-world                     # name of the template
      container:
        image: busybox
        command: [ echo ]
        args: [ "hello world" ]
        resources: # limit the resources
          limits:
            memory: 32Mi
            cpu: 100m
```



Step-per-pod

- Each step runs on a separate container/pod
- Gain all the advantages of Kubernetes auto-scaling, observability and CRD management

```
apiVersion: argoproj.io/v1alpha1
kind: Workflow
metadata:
  generateName: scripts-bash-
spec:
  entrypoint: bash-script-example
  templates:
  - name: bash-script-example
    steps:
    - - name: generate
      template: gen-random-int-bash
    - - name: print
      template: print-message
      arguments:
        parameters:
        - name: message
          value: "{{steps.generate.outputs.result}}" # The result of the here-script

  - name: gen-random-int-bash
    script:
      image: debian:9.4
      command: [bash]
      source: |
        # Contents of the here-script
        cat /dev/urandom | od -N2 -An -i | awk -v f=1 -v r=100 '{printf "%i\n", f -

  - name: gen-random-int-python
    script:
      image: python:alpine3.6
      command: [python]
      source: |
        import random
        i = random.randint(1, 100)
        print(i)
```

CI/CD example



Workflows / workflow-playground / coinflip-1723111800 WORKFLOW DETAILS

RESUBMIT DELETE LOGS SHARE PREVIOUS RUNS OPEN WORKFLOW TEMPLATE

SUMMARY CONTAINERS INPUTS/OUTPUTS

NAME coinflip-1723111800[0].flip-coin

ID coinflip-1723111800-2322386521

POD NAME coinflip-1723111800-flip-coin-2322386521

HOST NODE NAME gke-argo-demo-apps-default-node-pool-525d6f70-bxjm

TYPE Pod

PHASE ✔ Succeeded

START TIME 08/08/2024, 13:10:00 (53m6s ago)

END TIME 08/08/2024, 13:17:24 (45m42s ago)

DURATION 7m24s

[GET HELP](#)

Workflow Templates / workflow-playground WORKFLOW TEMPLATES

+ CREATE NEW WORKFLOW TEMPLATE

NAMESPACE workflow-playground

LABELS

NAME PATTERN

NAME	NAMESPACE	CREATED
artifacts	workflow-playground	160d10h ago
buildkit	workflow-playground	160d10h ago
ci	workflow-playground	160d10h ago
coinflip	workflow-playground	160d10h ago
distro	workflow-playground	160d10h ago
github-event	workflow-playground	160d10h ago

[GET HELP](#)

Reports / workflow-playground REPORTS

NAMESPACE workflow-playground

LABELS

WORKFLOW TEMPLATE

CRON WORKFLOW

PHASE

Succeeded

Error

Failed

Workflow ID	Duration (seconds)
coinflip-1723112100	~110
coinflip-1723111800	~480
coinflip-1723113900	~30
calendar-w4kb6	~0
calendar-zswm	~0
artifacts-1723114500	~10
coinflip-1723114500	188
ci-1723114500	~550

[GET HELP](#)

Cron Workflows / workflow-playground / coinflip CRON WORKFLOW DETAILS

SUBMIT UPDATE SUSPEND DELETE SHARE OPEN WORKFLOW TEMPLATE

STATUS MANIFEST CRON METADATA WORKFLOW WORKFLOW METADATA

JSON/YAML METADATA SPEC STATUS

```

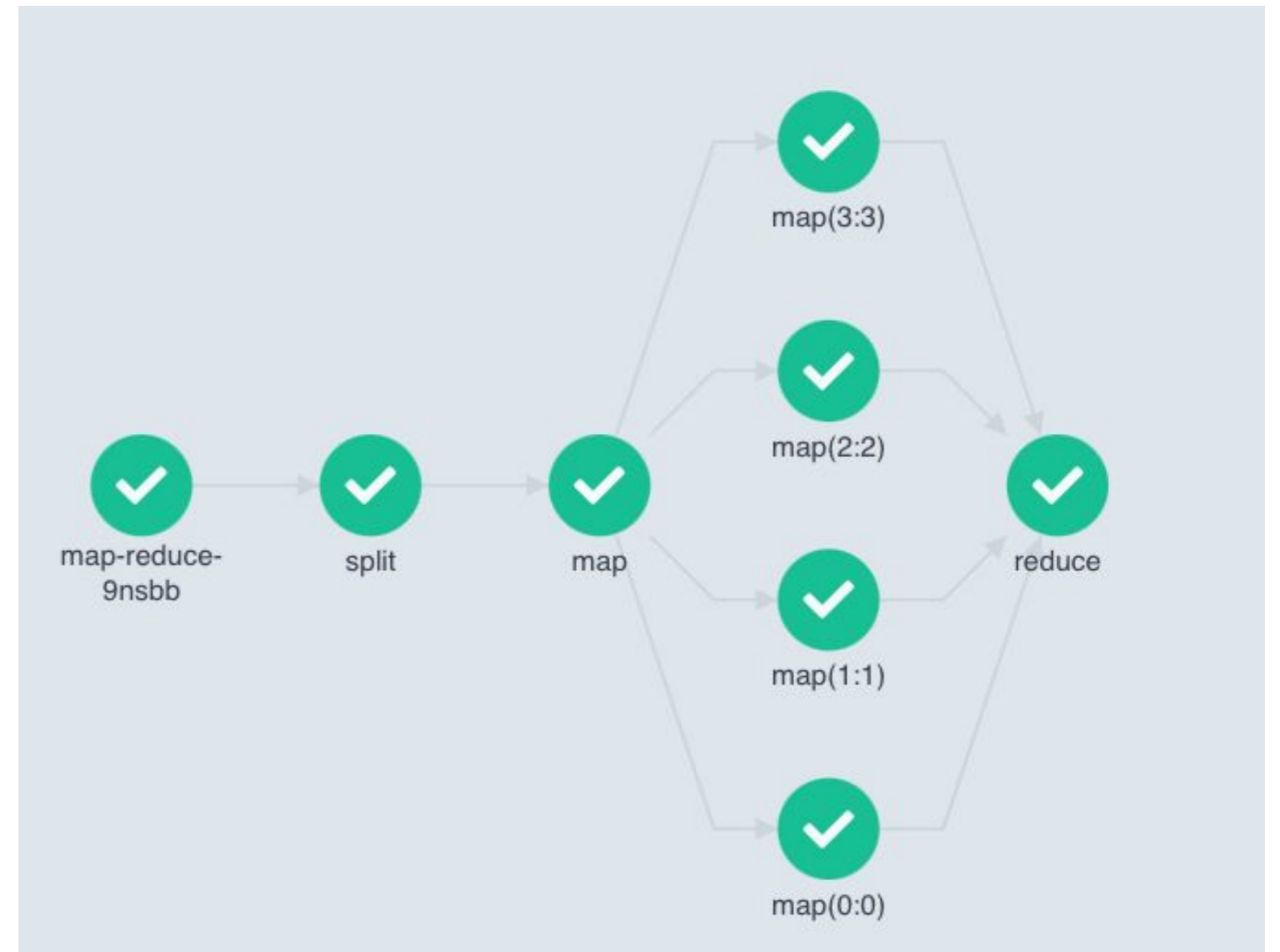
1 metadata:
2   name: coinflip
3   namespace: workflow-playground
4   uid: 9a782e6f-9aef-4c91-8d6e-e4bad46705a1
5   resourceVersion: '215319447'
6   generation: 92367
7   creationTimestamp: '2024-03-01T00:28:43Z'
8   annotations:
9     argocd.argoproj.io/tracking-id: workflow-examples:argoproj.io/CronWorkflow:workflow-playground/coinflip
10    cronworkflows.argoproj.io/last-used-schedule: '*/* * * * *'
11    kubernetes.io/last-applied-configuration: >
12      {"apiVersion":"argoproj.io/v1alpha1","kind":"CronWorkflow","metadata":{"annotations":{"argocd.argoproj.io/tracking-id":"workflow-examples:argoproj.io/CronWorkflow:workflow
13      * * *
14      *","workflowspec":{"serviceAccountName":"workflow","workflowTemplateRef":{"name":"coinflip"}}}}
15 managedFields:
16 - manager: argocd-controller
17   operation: Update
18   apiVersion: argoproj.io/v1alpha1
19   time: '2024-03-01T00:28:43Z'
20   fieldType: FieldsV1
21   fieldsV1:
22     f:metadata:
23       .: {}
24       f:argocd.argoproj.io/tracking-id: {}
25

```

[GET HELP](#)

Argo Workflows - other features

- Artifact storage/retrieval
- Workflow Archiving
- CLI/API and Analytics
- Retry mechanism/ Timeouts
- Suspend/resume
- Loops/Conditionals
- SSO/RBAC



Argo CD




Argo CD


- Deploys applications
- Kubernetes native
- Supports Helm/Kustomize
- Health status analysis
- Multi-tenant/RBAC


The screenshot displays the Argo CD web interface for an application named 'dex'. The top navigation bar includes 'Applications / Q dex' and a search icon. Below the navigation bar are several action buttons: 'DETAILS', 'DIFF', 'SYNC', 'SYNC STATUS', 'HISTORY AND ROLLBACK', 'DELETE', and 'REFRESH'. The main content area is divided into three sections: 'APP HEALTH' showing 'Healthy' with a green heart icon, 'SYNC STATUS' showing 'Synced to HEAD (b018629)' with a green checkmark and 'Auto sync is enabled.', and 'LAST SYNC' showing 'Sync OK to c94542d' with a green checkmark and 'Succeeded 5 months ago (Thu Feb 29 2024 20:09:17 GMT+0200)'. Below these sections is a detailed view of the application's resources, including 'dex svc', 'dex sa', 'dex deploy', 'dex-cert certificate', 'dex-issuer issuer', 'dex ep', 'dex-d66zh endpointslice', 'dex-5fc8545f77 rs', 'dex-cert-1 certificaterequest', 'dex-cert-2 certificaterequest', and 'dex-cert-3 certificaterequest'. Each resource card shows its name, icon, status (green checkmarks), and last sync time (e.g., '6 months', '5 months', '3 months', 'a month').





Argo CD UI


 **argo**
v2.11.0+1cffa15

 **Applications**


 Settings


 User Info


 Documentation

 Favorites Only


SYNC STATUS


 Unknown 2


 **Synced** 20


 OutOfSync 0


HEALTH STATUS


 Unknown 0

 Progressing 6

 Suspended 0


 **Healthy** 20

 Degraded 2

 Missing 0



Applications APPLICATIONS TILES

Previous 1 2 Next Sort: name ▾ Items per page: 10 ▾

 **argo-events**

Project: default

Labels:

Status:  Healthy  Synced

Reposito... https://github.com/argoproj/argoproj-de...

Target R... HEAD


Path: argo-events

Destinati... in-cluster

Namesp... workflow-playground



Created ... 02/22/2024 21:32:40 (6 months ago)

Last Sync: 07/13/2024 18:06:35 (a month ago)

 **argo-rollouts**

Project: default

Labels:

Status:  Healthy  Synced

Reposito... https://github.com/argoproj/argoproj-de...

Target R... HEAD


Path: argo-rollouts

Destinati... in-cluster

Namesp... argo-rollouts



Created ... 02/22/2024 21:32:41 (6 months ago)

Last Sync: 02/23/2024 19:36:19 (6 months ago)

 **argo-workflows**

Project: default

Labels:

Status:  Healthy  Synced

Reposito... https://github.com/argoproj/argoproj-de...

Target R... HEAD


Path: argo-workflows

Destinati... in-cluster

Namesp... argo



Created ... 02/22/2024 21:32:41 (6 months ago)

Last Sync: 03/11/2024 13:40:15 (5 months ago)

 **argocd-image-updater**

Project: default

Labels:


Status:  Healthy  Synced

Reposito... https://github.com/argoproj/argoproj-de...

Target R... HEAD



Path: argocd-image-updater

Destinati... in-cluster

 **dex**

Project: default

Labels:


Status:  Healthy  Synced

Reposito... https://github.com/argoproj/argoproj-de...

Target R... HEAD



Path: dex

Destinati... in-cluster

 **example.guestbook**

Project: default

Labels:

Status:  Healthy  Synced


Reposito... https://github.com/agaudreault/argocd-...

Target R... sync-from-demo

Path: guestbook

Destinati... in-cluster

Argo CD UI

 **argo**
v2.11.0+1cffa15

- Applications
- Settings
- User Info
- Documentation

NAME

KINDS

SYNC STATUS

- Synced 2
- OutOfSync 0

HEALTH STATUS

- Healthy 4
- Progressing 0
- Degraded 0

Applications /

APPLICATION DETAILS TREE

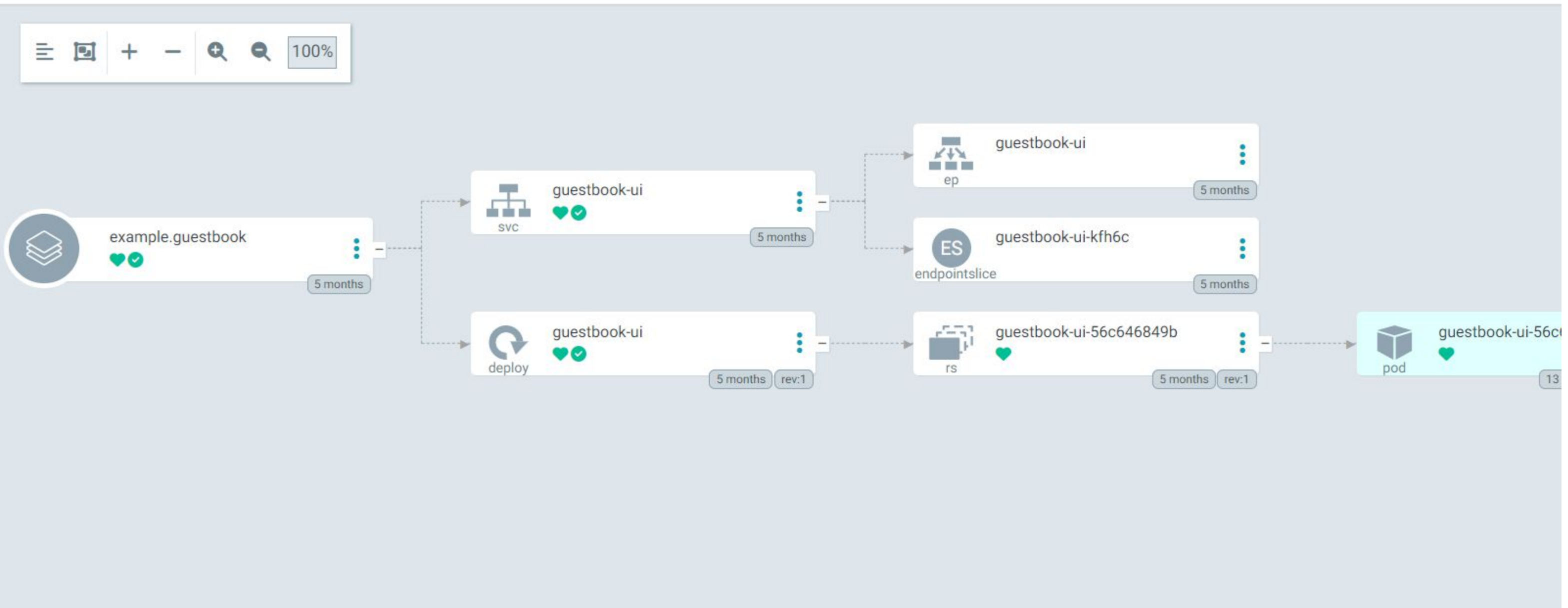
- DETAILS
- DIFF
- SYNC
- SYNC STATUS
- HISTORY AND ROLLBACK
- DELETE
- REFRESH

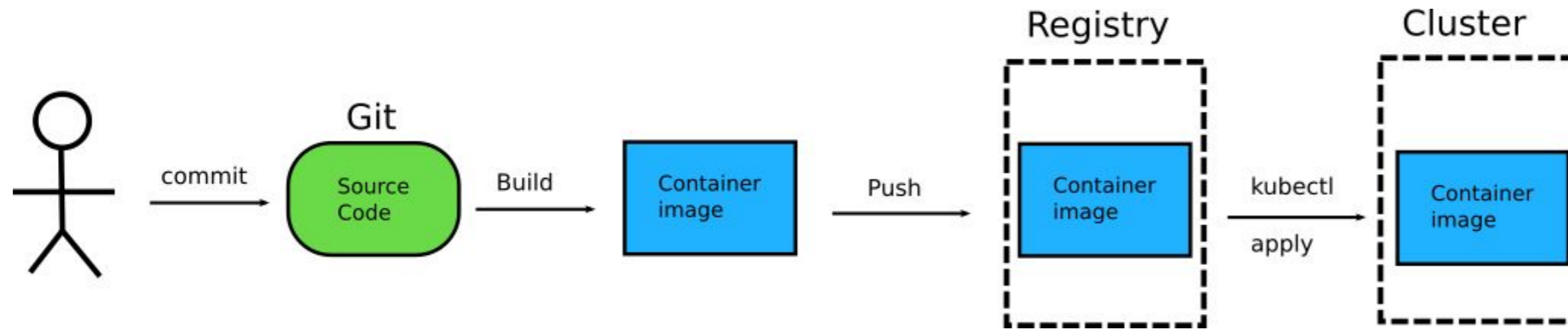
    Log in

APP HEALTH **Healthy**

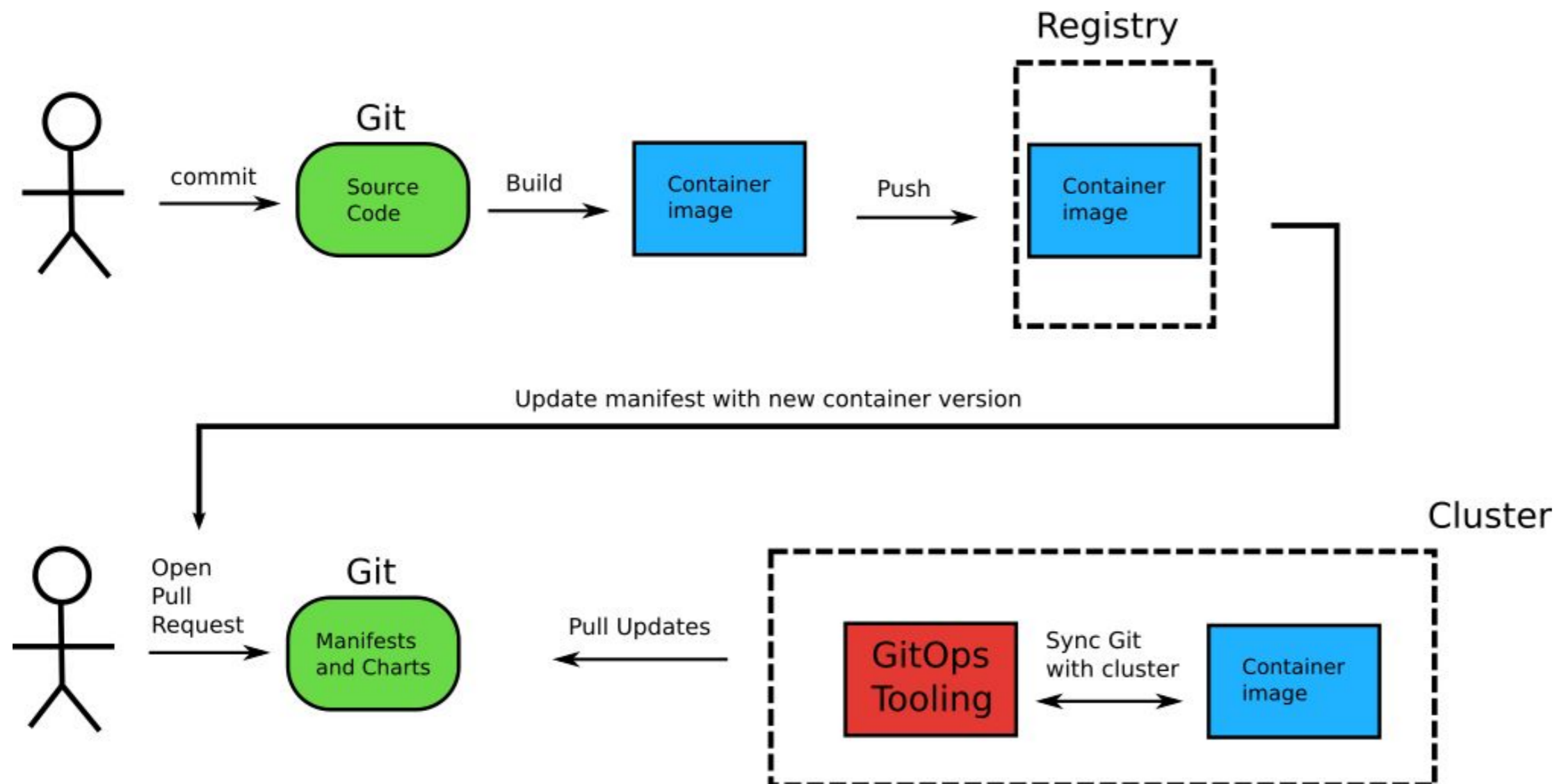
SYNC STATUS **Synced** to sync-from-demo (0d6bc4b)
Auto sync is enabled.
Author: Alexandre Gaudreault <alexandre_gaudreault@intuit...>
Comment: update readme

LAST SYNC **Sync OK** to d7927a2
Succeeded 5 months ago (Mon Mar 11 2024 20:23:46 GMT+0200)
Author: Anand Francis Joseph <anandfrancis.joseph@gmail...>
Comment: Template variable for container port (#251)





Abusing CI as CD



With Argo CD



Avoid Configuration Drift

SUMMARY PARAMETERS MANIFEST **DIFF** EVENTS

Compact diff Inline Diff

```
 /Service/default/guestbook-ui  
  
 1 apiVersion: v1  
 2 kind: Service  
 3 metadata:  
 4   labels:  
 5     app.kubernetes.io/instance: guestbook  
 6   name: guestbook-ui  
 7 spec:  
 8   ports:  
  
 9     - port: 8080  
10     targetPort: 80  
11 selector:  
12   app: guestbook-ui  
  
 1 apiVersion: v1  
 2 kind: Service  
 3 metadata:  
 4   labels:  
 5     app.kubernetes.io/instance: guestbook  
 6   name: guestbook-ui  
 7 spec:  
 8   ports:  
 9     - port: 80  
10     targetPort: 80  
11     - port: 8080  
12     targetPort: 80  
13 selector:  
14   app: guestbook-ui
```

Applications / guestbook

APP DETAILS APP DIFF SYNC SYNC STATUS HISTORY AND ROLLBACK DELETE REFRESH

Healthy **OutOfSync** Sync OK

From HEAD (6bed858)
Authored by Alex Collins <alexec...>
Updates examples to better reflec...

To 6bed858
Succeeded 7 days ago (Mon Aug 17 2020 19:27:35 GMT+0300)
Authored by Alex Collins <alexec@users.noreply.github.com>
Updates examples to better reflect hook usage today (#41)



Applications / guestbook

APP DETAILS APP DIFF SYNC SYNC STATUS HISTORY AND ROLLBACK DELETE REFRESH

Healthy Synced Sync OK

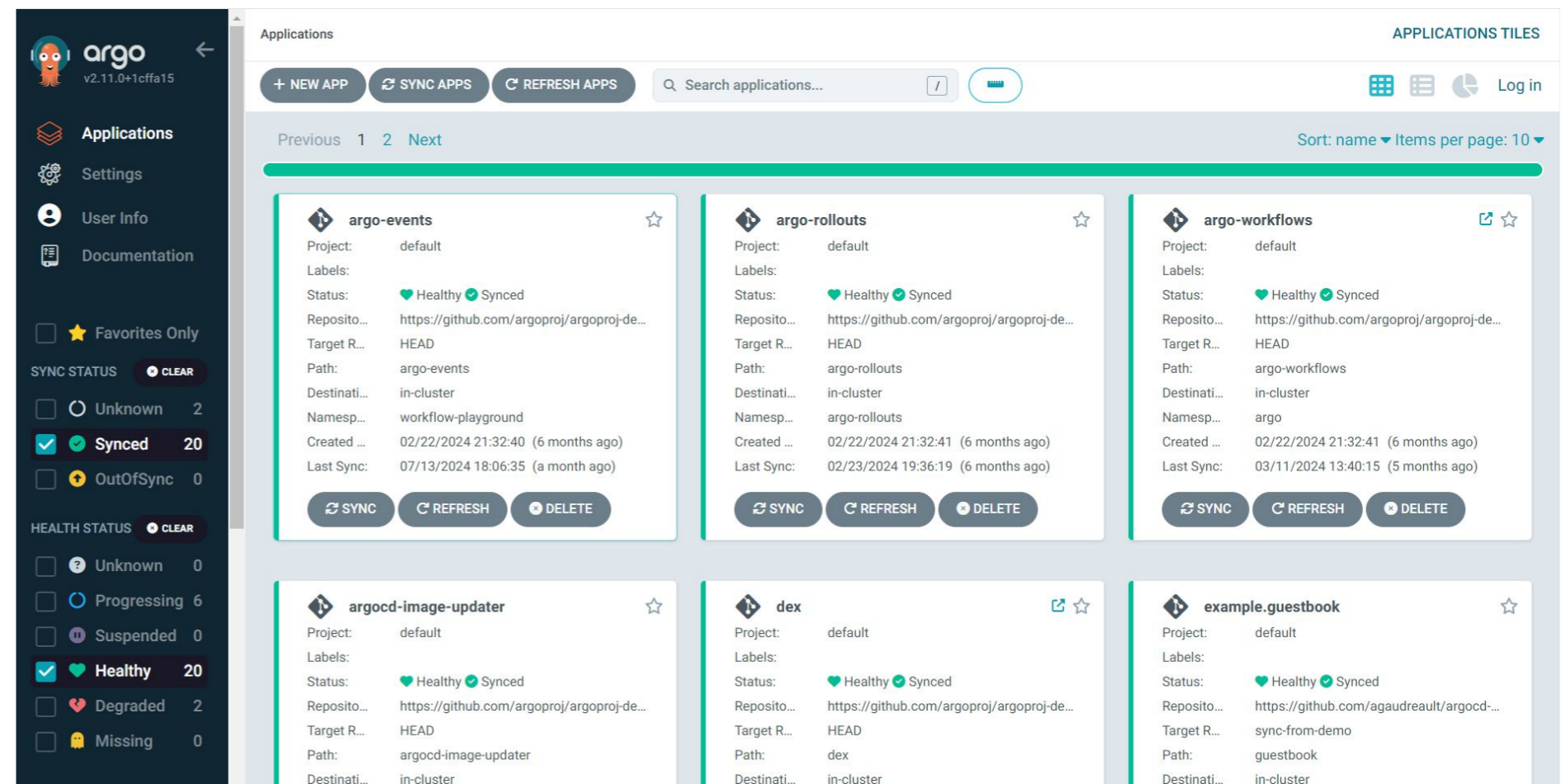
To HEAD (6bed858)
Authored by Alex Collins <alexec...>
Updates examples to better reflec...

To 6bed858
Succeeded 7 days ago (Mon Aug 17 2020 19:27:35 GMT+0300)
Authored by Alex Collins <alexec@users.noreply.github.com>
Updates examples to better reflect hook usage today (#41)



Argo CD entities

- **Application**- Link between a cluster and Git repo
- **Project** - RBAC for Applications
- **ApplicationSet**- Generator/grouping for applications



The screenshot displays the Argo CD web interface. On the left is a dark sidebar with navigation options: Applications, Settings, User Info, and Documentation. Below these are filters for SYNC STATUS (Unknown: 2, Synced: 20, OutOfSync: 0) and HEALTH STATUS (Unknown: 0, Progressing: 6, Suspended: 0, Healthy: 20, Degraded: 2, Missing: 0). The main content area, titled 'Applications', shows a grid of application cards. Each card includes the application name, project, labels, status (Healthy/Synced), repository URL, target revision, path, destination, namespace, creation time, and last sync time. Action buttons for SYNC, REFRESH, and DELETE are provided for each application. The applications shown are: argo-events, argo-rollouts, argo-workflows, argocd-image-updater, dex, and example.guestbook.



Sync manifests to Cluster

```
apiVersion: argoproj.io/v1alpha1
```

```
kind: Application
```

```
metadata:
```

```
name: guestbook Name of application
```

```
namespace: argocd
```

```
spec:
```

```
project: default
```

```
source:
```

Where to read the Kubernetes manifest

```
repoURL: https://github.com/argoproj/argocd-example-apps.git
```

```
targetRevision: HEAD
```

```
path: guestbook
```

```
destination:
```

Which cluster to deploy the application to

```
server: https://kubernetes.default.svc
```

```
namespace: guestbook
```



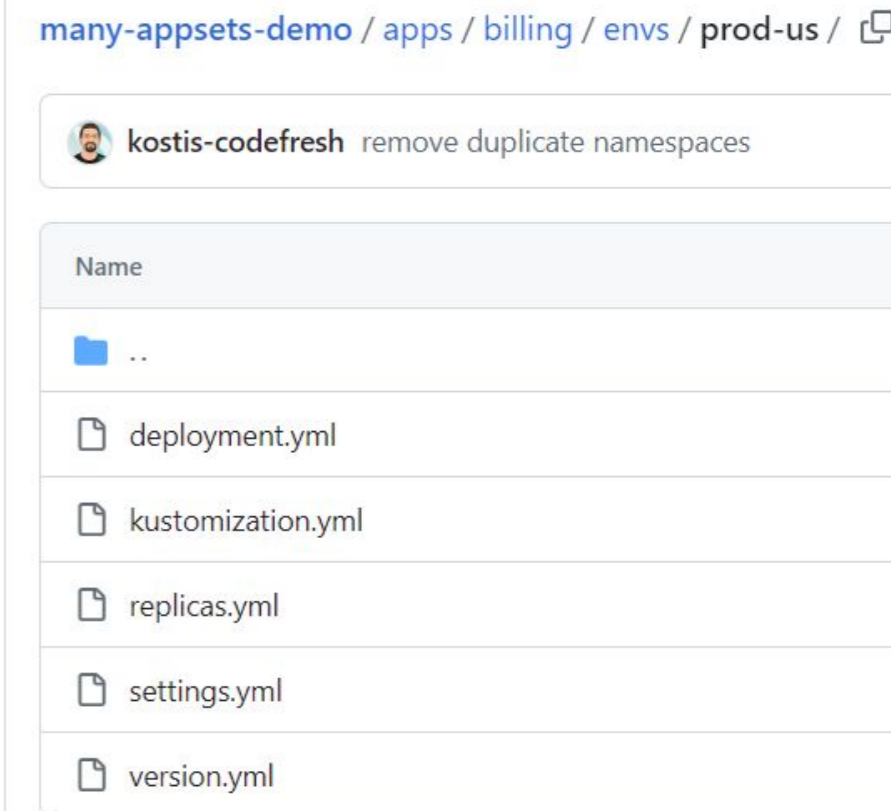
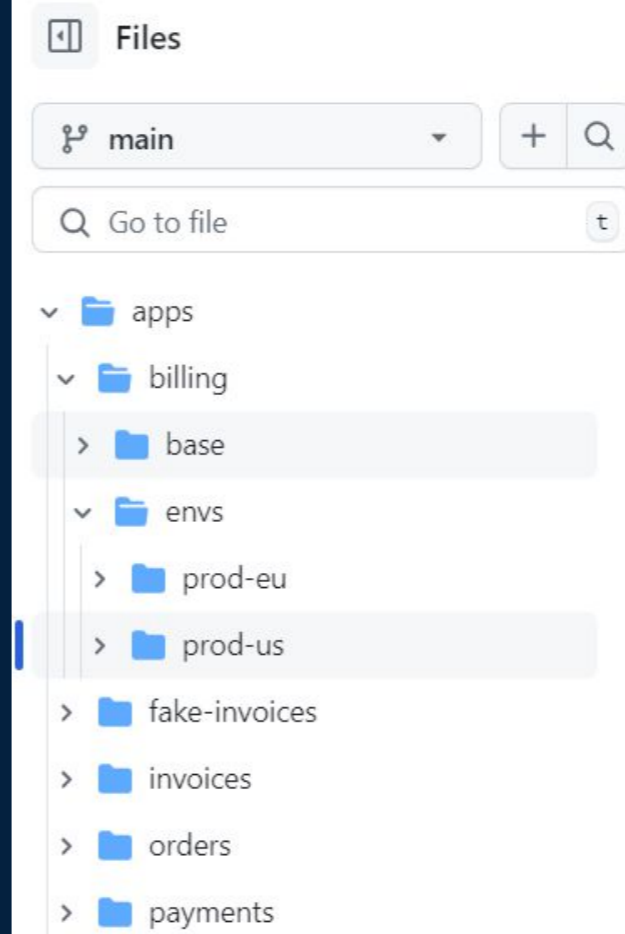
```

apiVersion: argoproj.io/v1alpha1
kind: ApplicationSet
metadata:
  name: my-qa-appset
  namespace: argocd
spec:
  goTemplate: true
  goTemplateOptions: ["missingkey=error"]
  generators:
    - git:
        repoURL: https://github.com/kostis-codefresh/many-appsets-demo.git
        revision: HEAD
        directories:
          - path: apps/*/envs/qa
  template:
    metadata:
      name: '{{index .path.segments 1}}-{{index .path.segments 3}}'
    spec:
      # The project the application belongs to.
      project: default

      # Source of the application manifests
      source:
        repoURL: https://github.com/kostis-codefresh/many-appsets-demo.git
        targetRevision: HEAD
        path: '{{.path.path}}'

      # Destination cluster and namespace to deploy the application
      destination:
        server: https://kubernetes.default.svc
        namespace: '{{index .path.segments 1}}-{{index .path.segments 3}}'

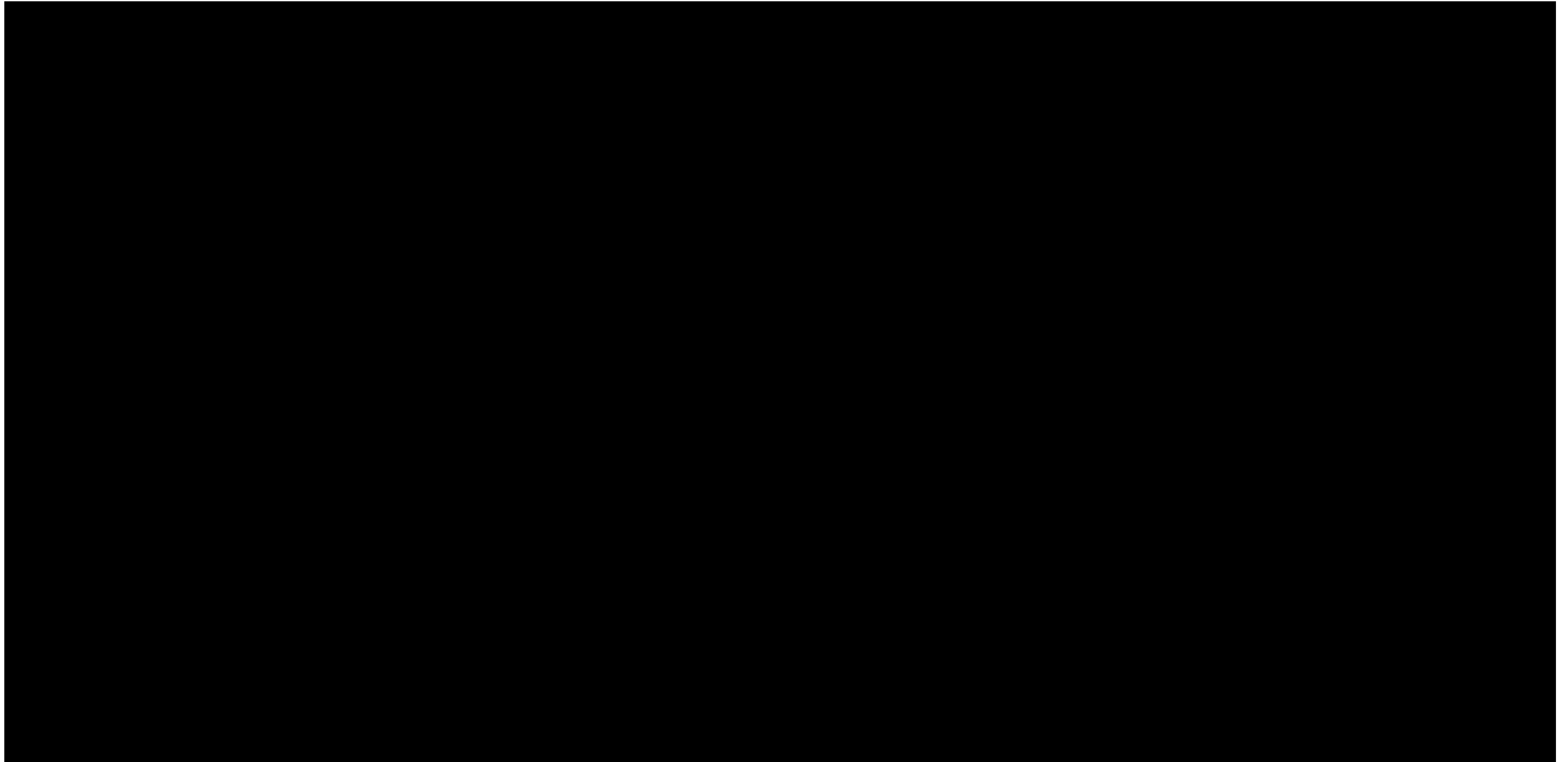
```



Generate applications from Git folders



Cluster bootstrapping

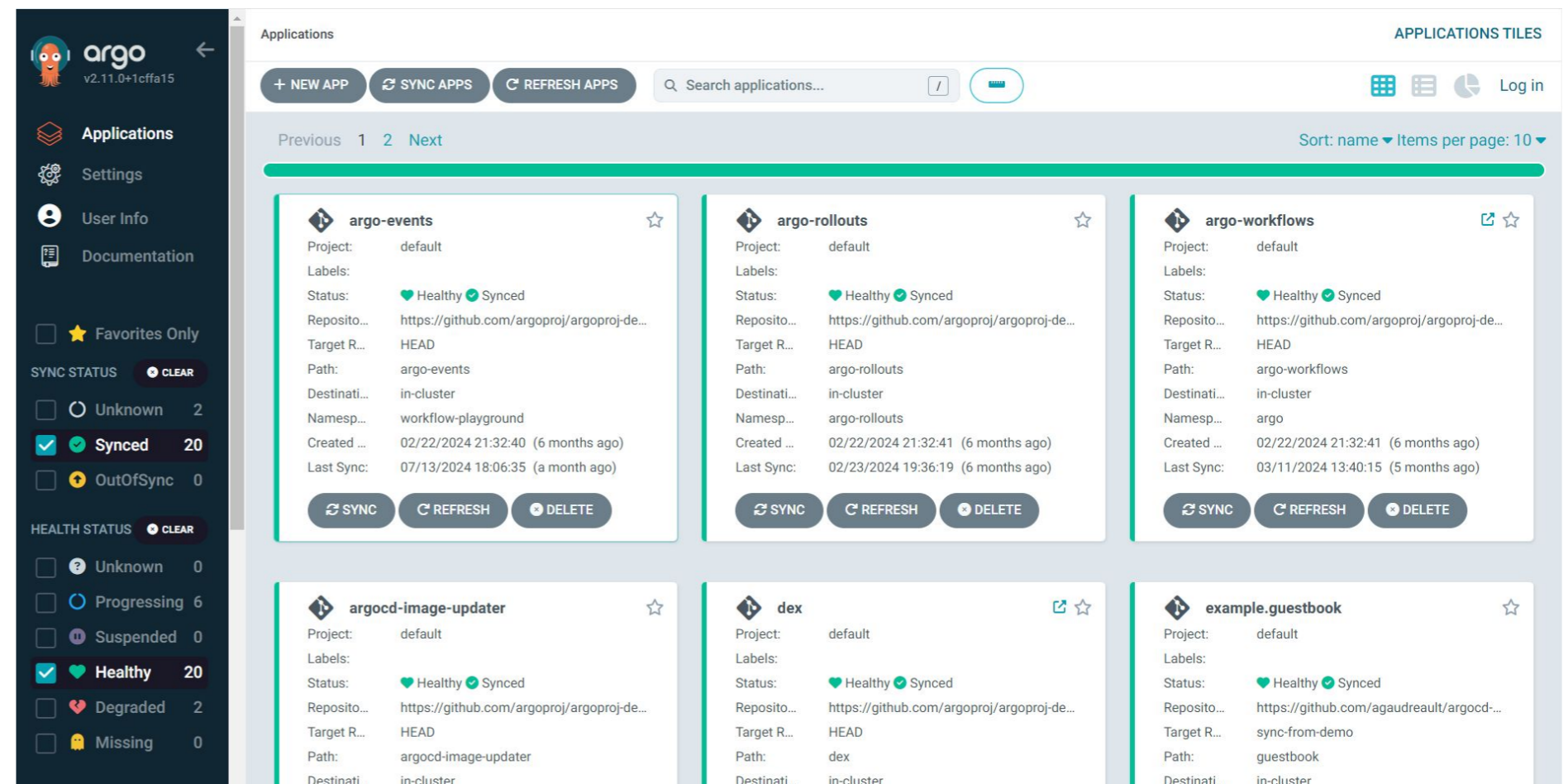


Argo CD topologies



Argo CD other features

- Sync policies
- Sync waves/phases/windows
- Git webhooks
- CLI/API
- SSO/RBAC
- Plugins
- Notifications



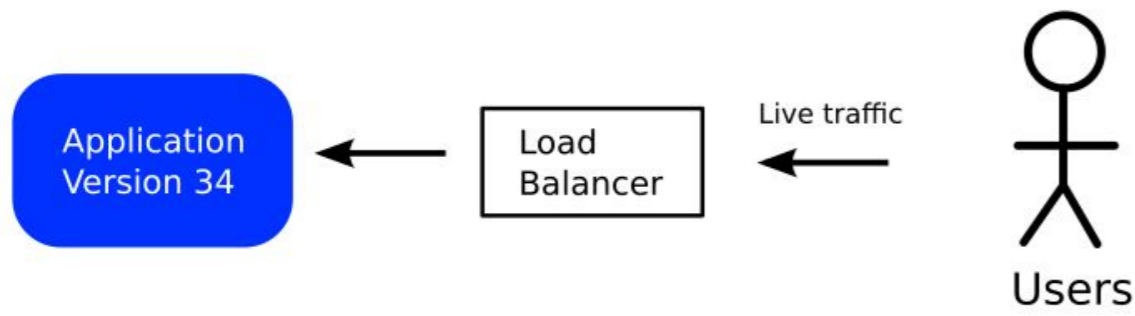
The screenshot displays the Argo CD web interface. On the left is a dark sidebar with navigation options: Applications, Settings, User Info, and Documentation. Below these are filters for 'Favorites Only', 'SYNC STATUS' (Unknown: 2, Synced: 20, OutOfSync: 0), and 'HEALTH STATUS' (Unknown: 0, Progressing: 6, Suspended: 0, Healthy: 20, Degraded: 2, Missing: 0). The main content area, titled 'Applications', shows a grid of application tiles. Each tile includes the application name, project, labels, status (Healthy/Synced), repository URL, target revision, path, destination, namespace, creation time, and last sync time. Action buttons for SYNC, REFRESH, and DELETE are provided for each application. The applications shown are argo-events, argo-rollouts, argo-workflows, argocd-image-updater, dex, and example.guestbook.



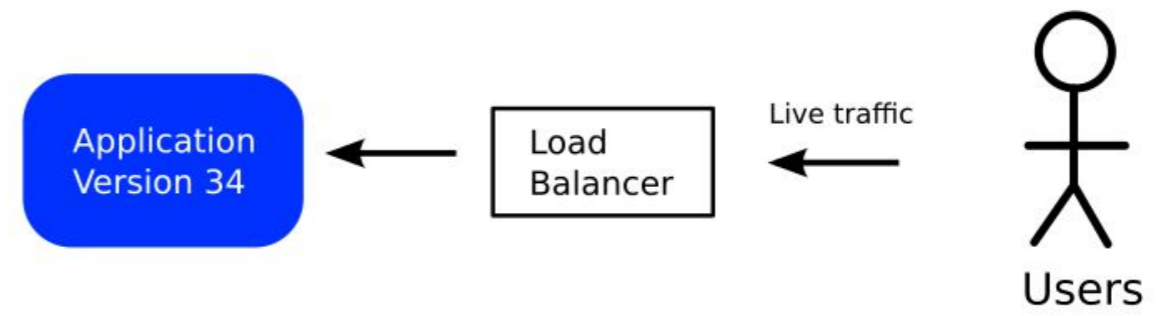
Argo Rollouts



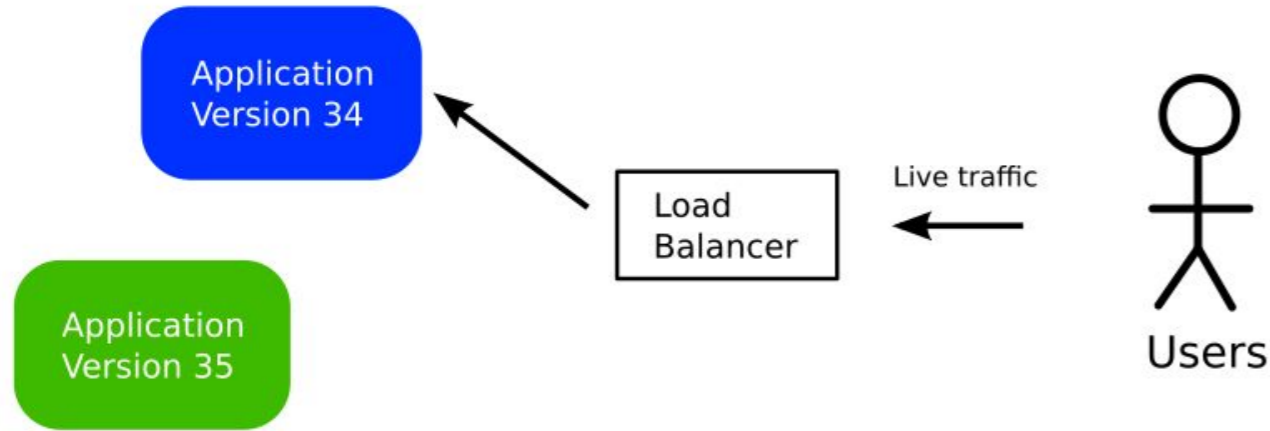
1- Initial version



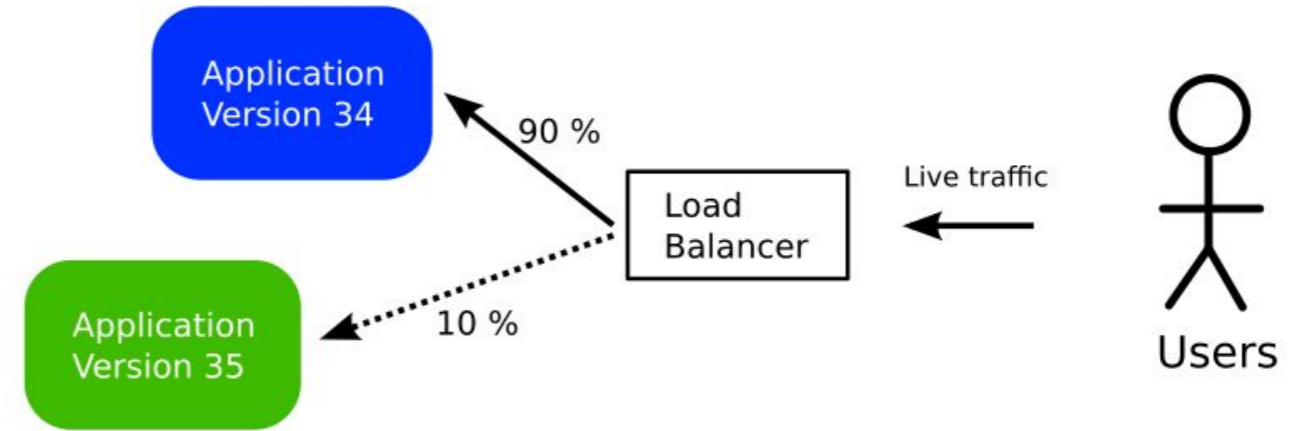
1- Initial version



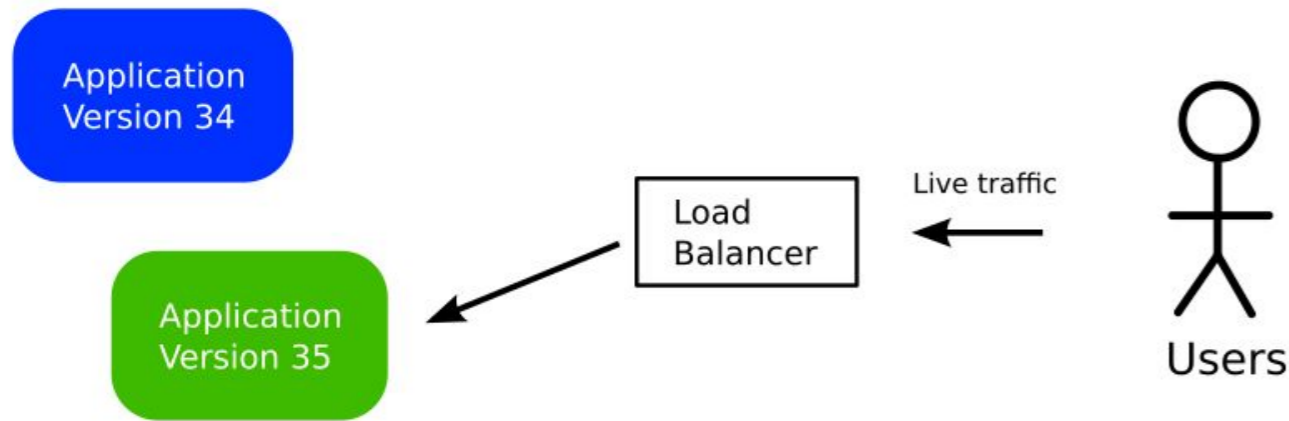
2- New version deployed



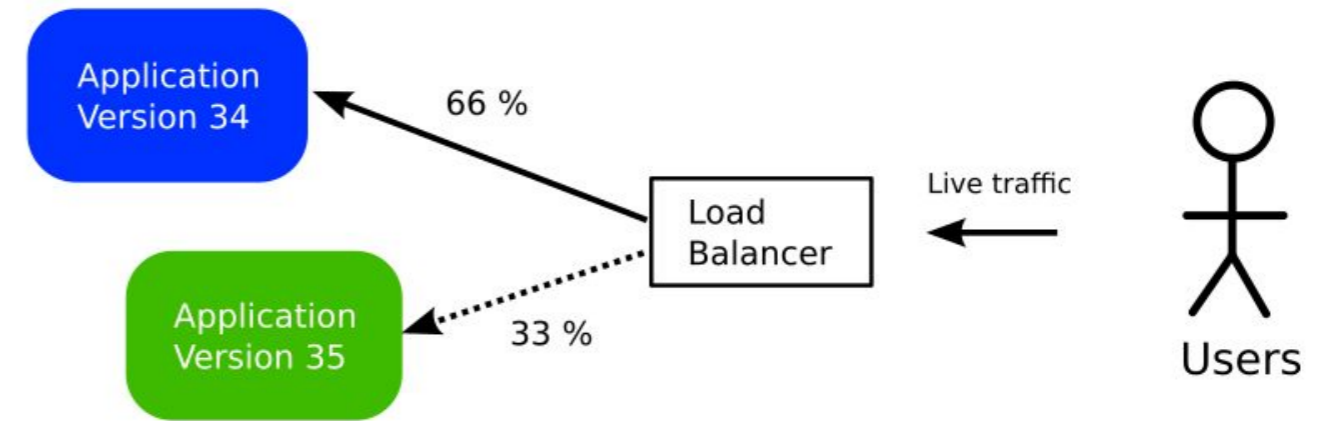
2- New version used by 10% of users



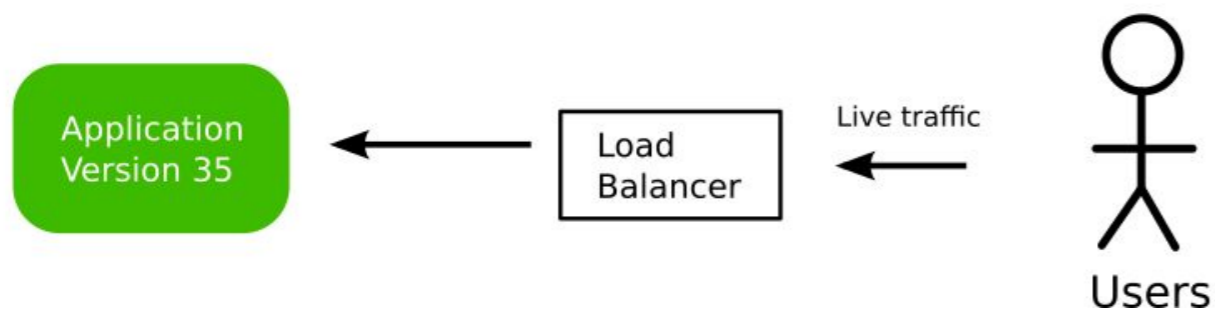
3- Switch Traffic



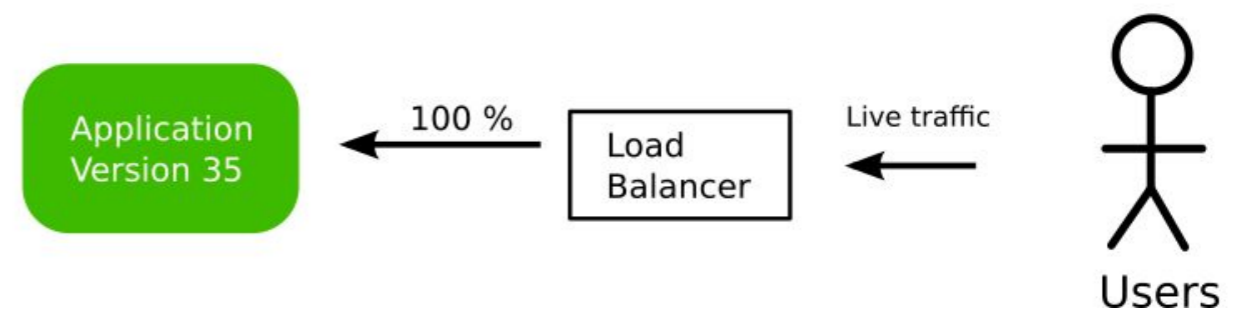
3- New version used by 33% of users



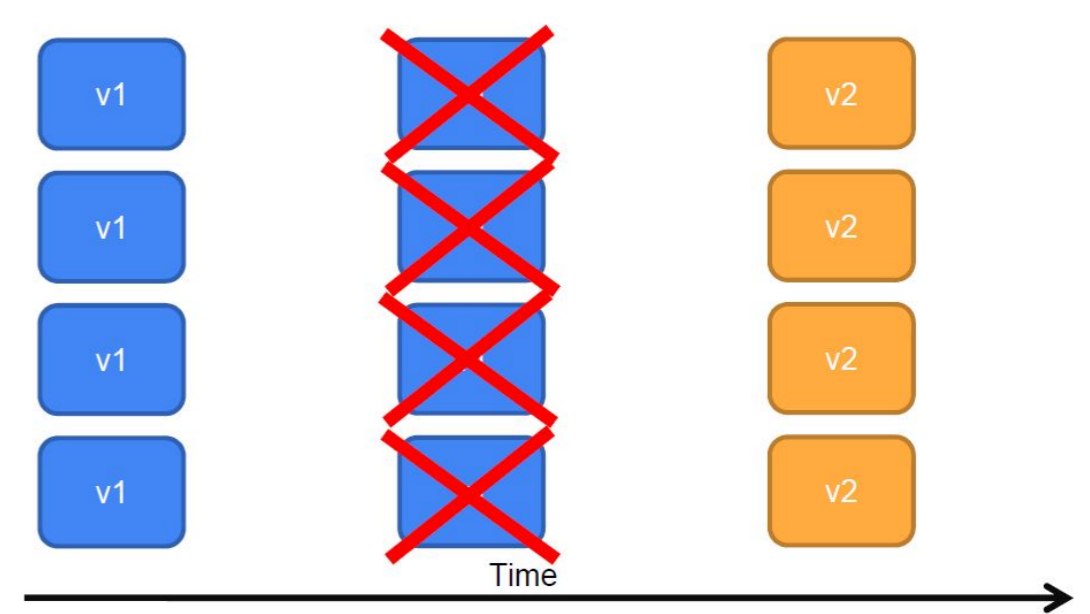
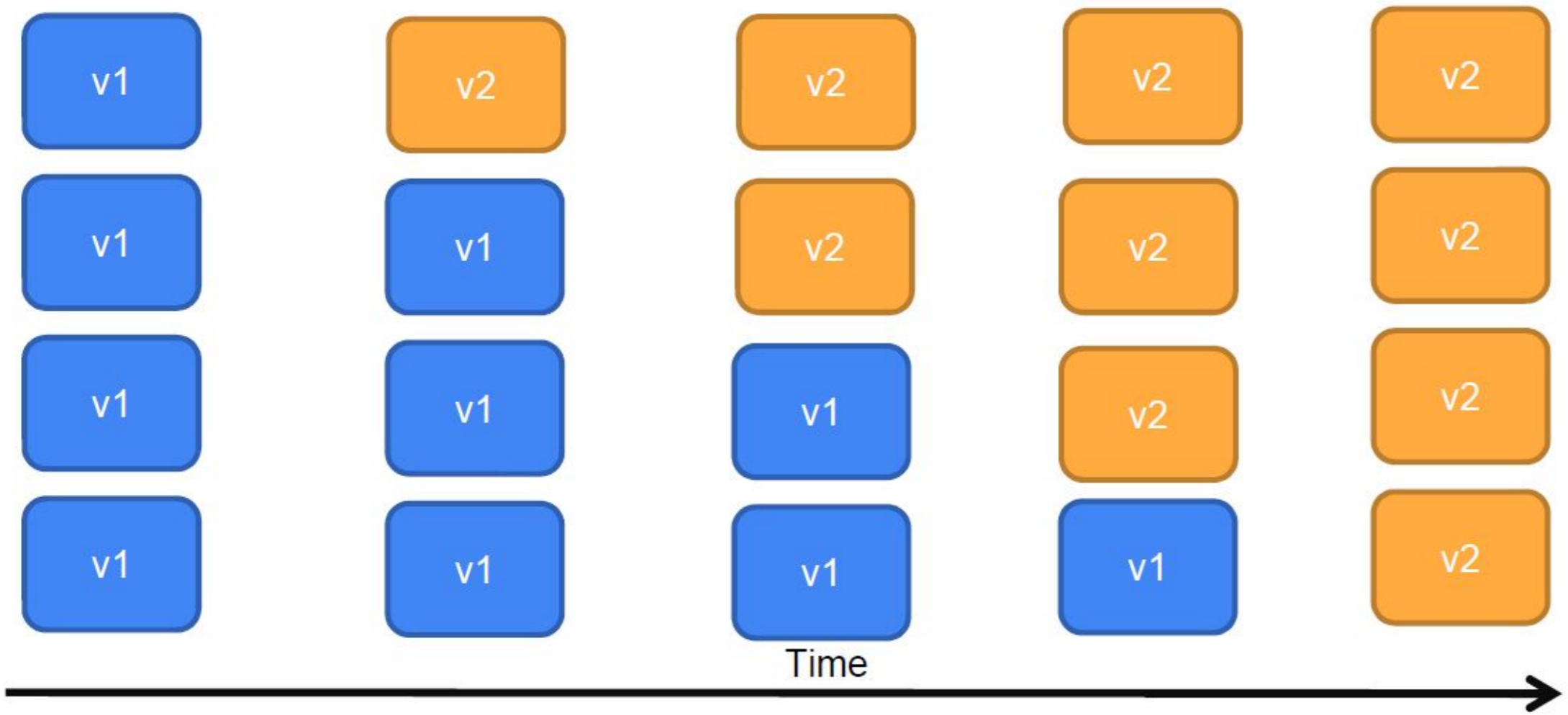
4- Finish



4- New version is used by all users



Default Kubernetes deployments



Argo Rollouts









- Rollouts (new CRD)
- Extends Deployment
- Blue/Green/Canaries
- Minimal dashboard
- Pre/Post checks

The screenshot shows the Argo Rollouts dashboard for a rollout named "rollouts-demo". At the top right, there are control buttons: Restart, Retry, Abort, Promote, and PromoteFull. The dashboard is divided into several sections:

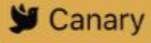
- Steps:** A vertical sequence of steps: Set Weight: 20%, Pause, Set Weight: 40%, Pause: 10s, Set Weight: 60%, Pause: 10s, Set Weight: 80%, and Pause: 10s. The "Pause" step is currently active and highlighted with a red border.
- Summary:** Shows the rollout strategy as "Canary" (indicated by a bird icon). It also shows "Step" as 1/8, "Set Weight" as 20, and "Actual Weight" as 20.
- Containers:** Lists the containers for the rollout: "rollouts-demo" and "argoproj/rollouts-demo:yellow".
- Revisions:** Shows two revisions:
 - Revision 2:** "argoproj/rollouts-demo:yellow" with revision ID "rollouts-demo-6cf78c66c5". It is marked as "canary" and has a green checkmark.
 - Revision 1:** "argoproj/rollouts-demo:blue" with revision ID "rollouts-demo-687d76d795". It is marked as "stable" and has five green checkmarks, indicating it is the previous stable revision.





Steps


-  Set Weight: 20%
-  Pause
-  Set Weight: 40%
-  Pause: 10s
-  Set Weight: 60%
-  Pause: 10s
-  Set Weight: 80%
-  Pause: 10s

Summary

Strategy  Canary

Step  1/8

Set Weight  20


Actual Weight  20

Containers

[Edit](#)

rollouts-demo

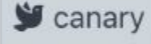

argoproj/rollouts-demo:yellow




Revisions

Revision 2



argoproj/rollouts-demo:yellow






rollouts-demo-6cf78c66c5  canary 



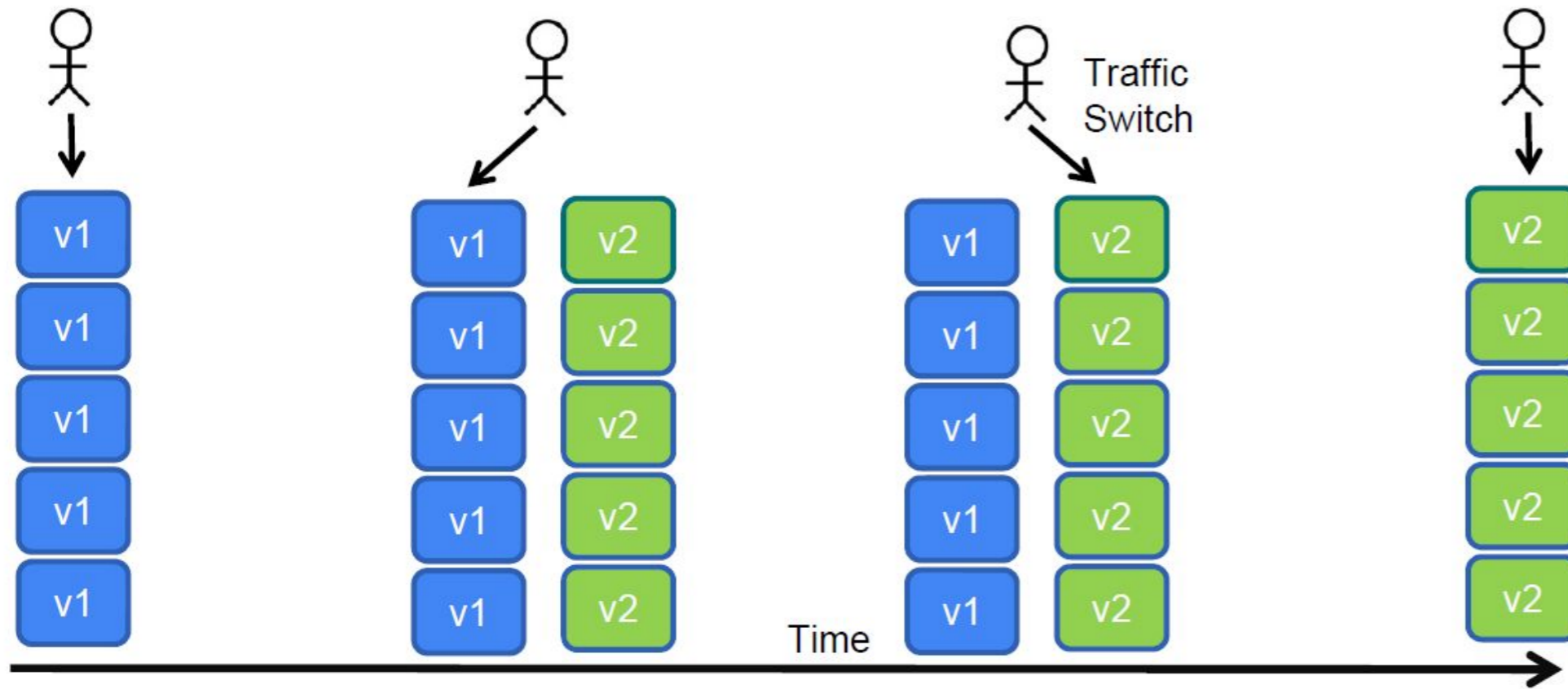
Revision 1

argoproj/rollouts-demo:blue

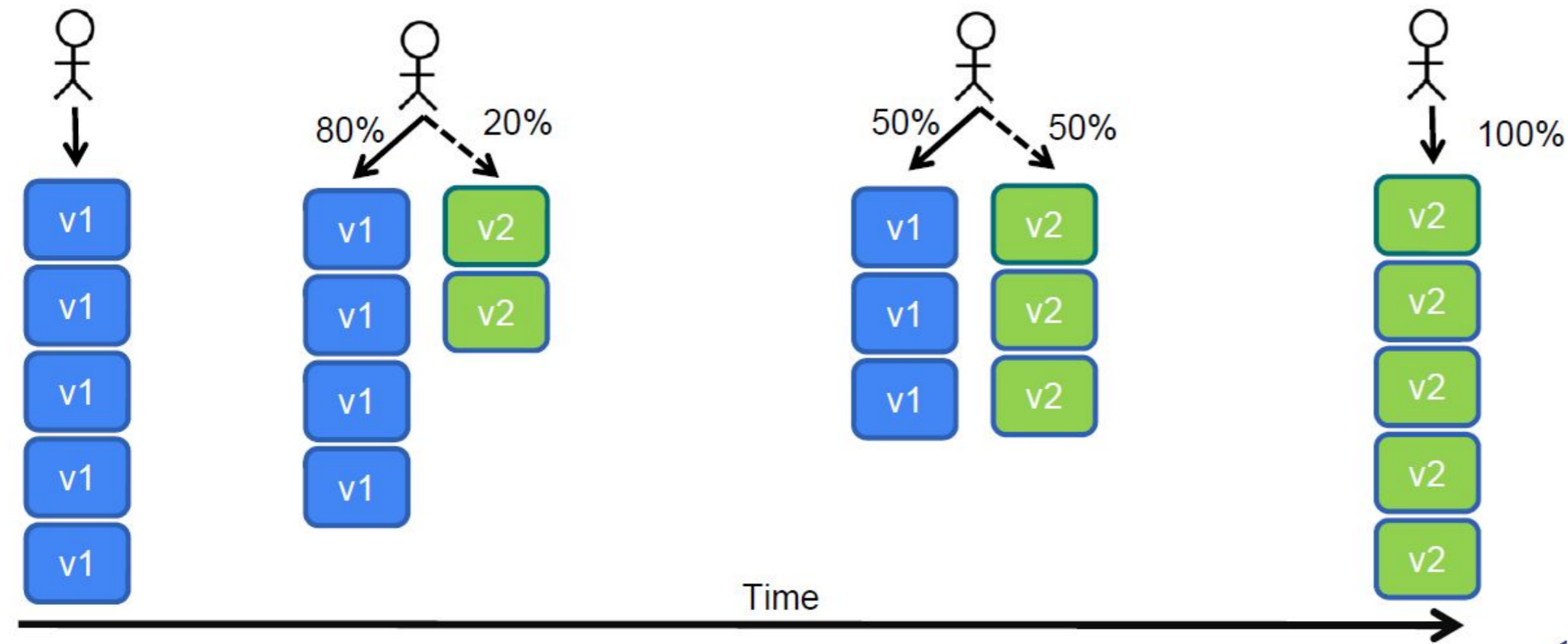
rollouts-demo-687d76d795  Rollback 





Kubernetes Progressive Delivery



Argo Rollouts Entities

- **Rollout** - main spec
- **AnalysisTemplate** - define pre/post checks
- **ClusterAnalysisTemplate** - clusterwide
- **AnalysisRun** - result of check
- **Experiment** - a/b testing

The screenshot shows the Argo Rollouts UI for a rollout named 'rollouts-demo'. The interface is divided into several sections:

- Steps:** A vertical list of steps in a rollout process. The steps are: 'Set Weight: 20%' (highlighted in green), 'Pause' (highlighted in orange), 'Set Weight: 40%', 'Pause: 10s', 'Set Weight: 60%', 'Pause: 10s', 'Set Weight: 80%', and 'Pause: 10s'.
- Summary:** A section showing the current strategy as 'Canary', the current step as '1/8', and the current set weight as '20%'. It also shows the actual weight as '20%'.
- Containers:** A section showing the rollout name 'rollouts-demo' and the container name 'argoproj/rollouts-demo:yellow'.
- Revisions:** A section showing two revisions. Revision 2 is the current revision, named 'argoproj/rollouts-demo:yellow', with a 'canary' strategy and a green checkmark. Revision 1 is named 'argoproj/rollouts-demo:blue', with a 'stable' strategy and a green checkmark. A 'Rollback' button is visible next to Revision 1.

At the top right of the UI, there are several control buttons: 'Restart', 'Retry', 'Abort', 'Promote', and 'PromoteFull'.



```
apiVersion: argoproj.io/v1alpha1
kind: Rollout
metadata:
  name: example-rollout
spec:
  replicas: 10
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx
          image: nginx:1.15.4
          ports:
            - containerPort: 80
  minReadySeconds: 30
  revisionHistoryLimit: 3
```

```
strategy:
  canary: #Indicates that the rollout should use the Canary strategy
    maxSurge: "25%"
    maxUnavailable: 0
    steps:
      - setWeight: 10
      - pause:
          duration: 1h # 1 hour
      - setWeight: 20
      - pause: {} # pause indefinitely
```

Strategy

Rollout extends K8s deployment



Without/With traffic management

Prod
50%



Canary
50%



Linkerd



Prod
95%



Canary
5%



Prod
75%



Canary
25%



Linkerd



Prod
70%



Canary
30%



Prod
90%



Canary
10%



Linkerd



Prod
20%



Canary
80%



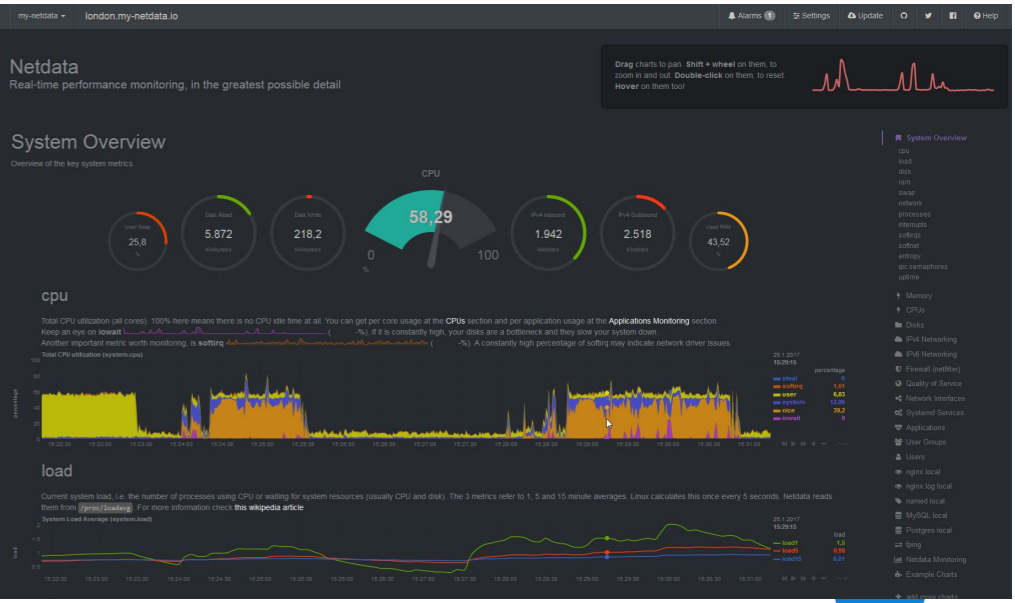
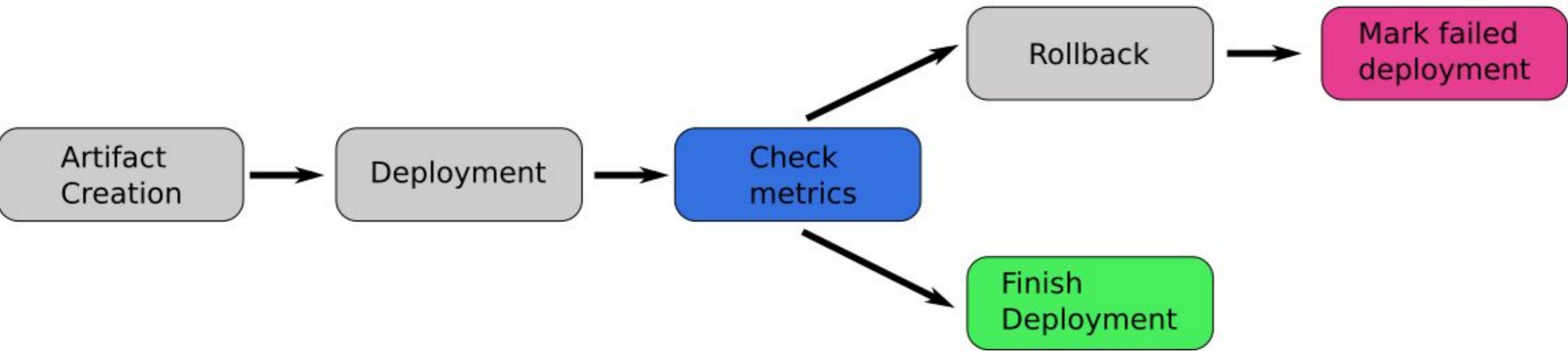
Supported Traffic managers

- AWS Ingress Controller
- Ambassador Labs
- Apache APISIX
- Linkerd
- Istio
- Kong
- Nginx
- Traefik
- Openshift Routes
- Gloo Gateway
- Contour
- Cilium
- Envoy Gateway
- Gateway API



Pre/Post checks

Fully Automated Rollbacks



```
apiVersion: argoproj.io/v1alpha1
kind: AnalysisTemplate
metadata:
  name: success-rate
spec:
  args:
  - name: service-name
  metrics:
  - name: success-rate
    interval: 5m
    # NOTE: prometheus queries return results in the form of a vector.
    # So it is common to access the index 0 of the returned array to obtain the value
    successCondition: result[0] >= 0.95
    failureLimit: 3
  provider:
    prometheus:
      address: http://prometheus.example.com:9090
      query: |
        sum(irate(
          istio_requests_total{reporter="source",destination_service=~"{{args.service-
name}}",response_code!~"5.*"}[5m]
        )) /
        sum(irate(
          istio_requests_total{reporter="source",destination_service=~"{{args.service-name}}"}[5m]
        ))|
```

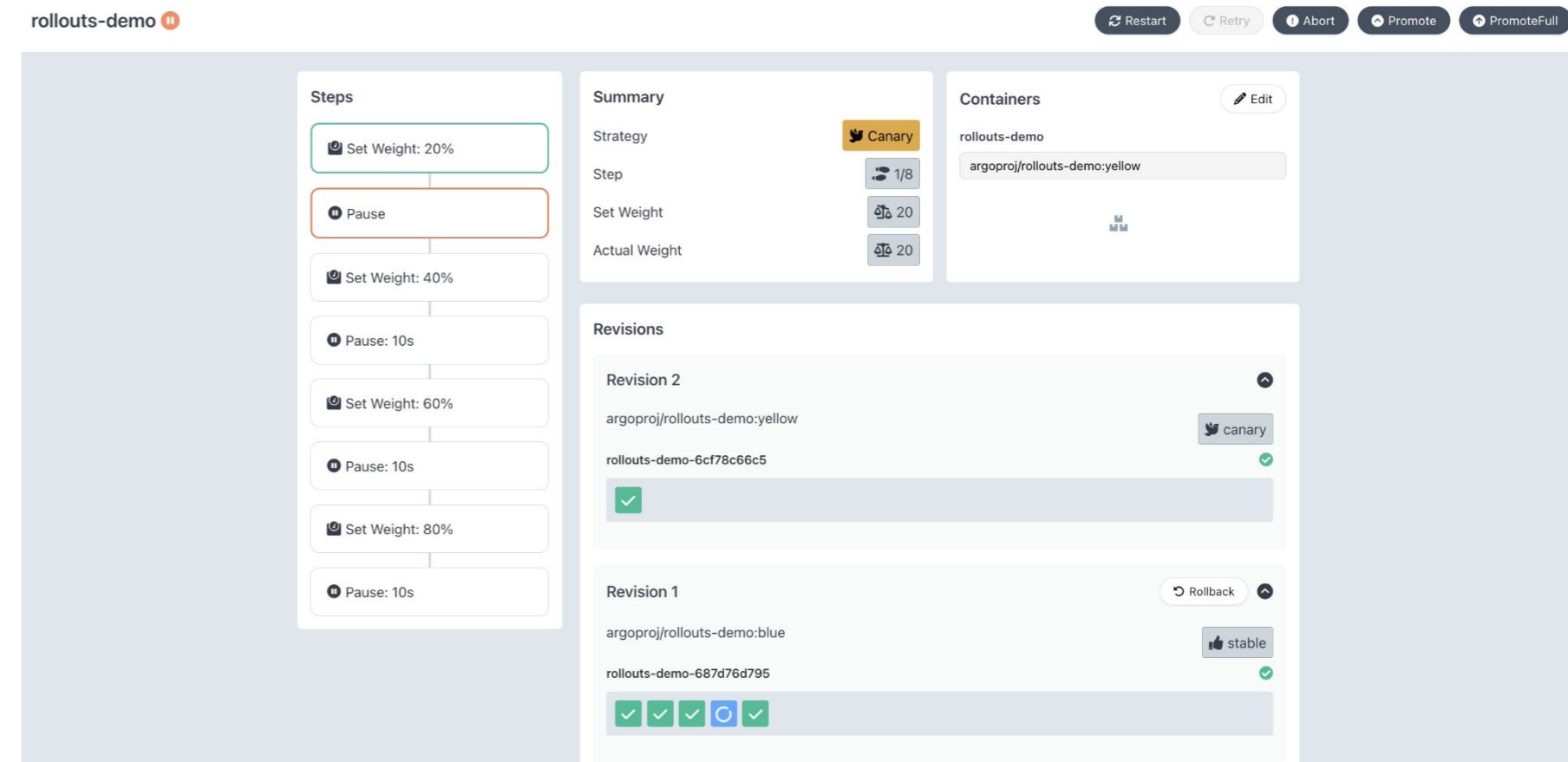
Supported Metric providers

- Prometheus
- Datadog
- New Relic
- Wavefront
- CloudWatch
- Apache SkyWalking
- Graphite
- Custom Web call
- Custom Job
- Custom plugin



Argo Rollouts - Other features

- A/B testing
- Header based routing
- Argo CD UI extension
- Notifications
- Plugins
- CLI/Metrics



The screenshot displays the Argo Rollouts UI for a rollout named 'rollouts-demo'. At the top right, there are control buttons: Restart, Retry, Abort, Promote, and PromoteFull. The main interface is divided into several panels:

- Steps:** A vertical sequence of steps: 'Set Weight: 20%' (green border), 'Pause' (orange border), 'Set Weight: 40%', 'Pause: 10s', 'Set Weight: 60%', 'Pause: 10s', 'Set Weight: 80%', and 'Pause: 10s'.
- Summary:** Shows the current strategy as 'Canary' (yellow icon), 'Step' as '1/8', and 'Set Weight' and 'Actual Weight' both at '20%'.
- Containers:** Lists the rollout name 'rollouts-demo' and the specific container 'argoproj/rollouts-demo:yellow'.
- Revisions:** Shows two revisions:
 - Revision 2:** 'argoproj/rollouts-demo:yellow' with ID 'rollouts-demo-6cf78c66c5', marked as 'canary' with a green checkmark.
 - Revision 1:** 'argoproj/rollouts-demo:blue' with ID 'rollouts-demo-687d76d795', marked as 'stable' with a thumbs-up icon and a green checkmark. It includes a 'Rollback' button.



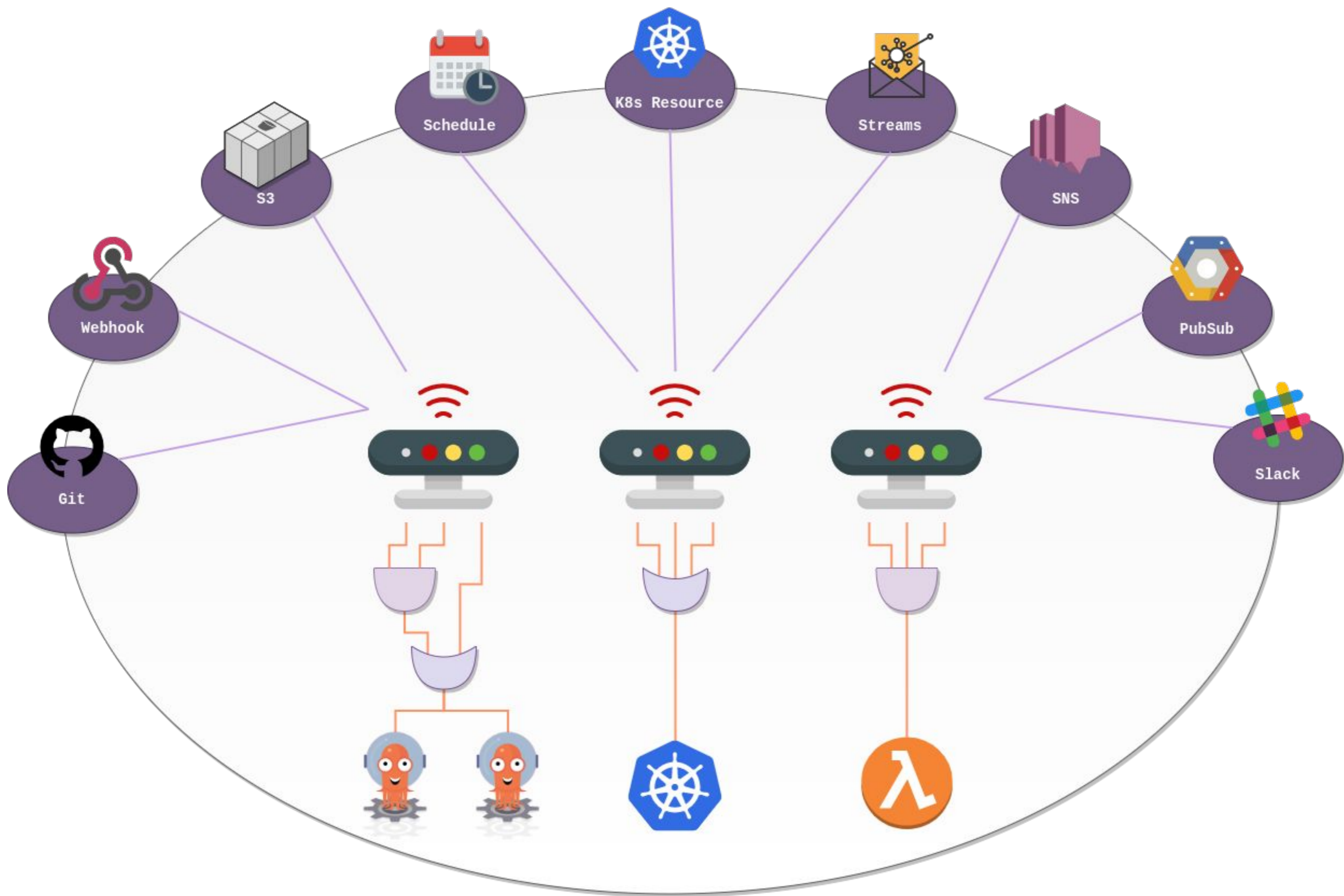
Argo Events



Argo Events

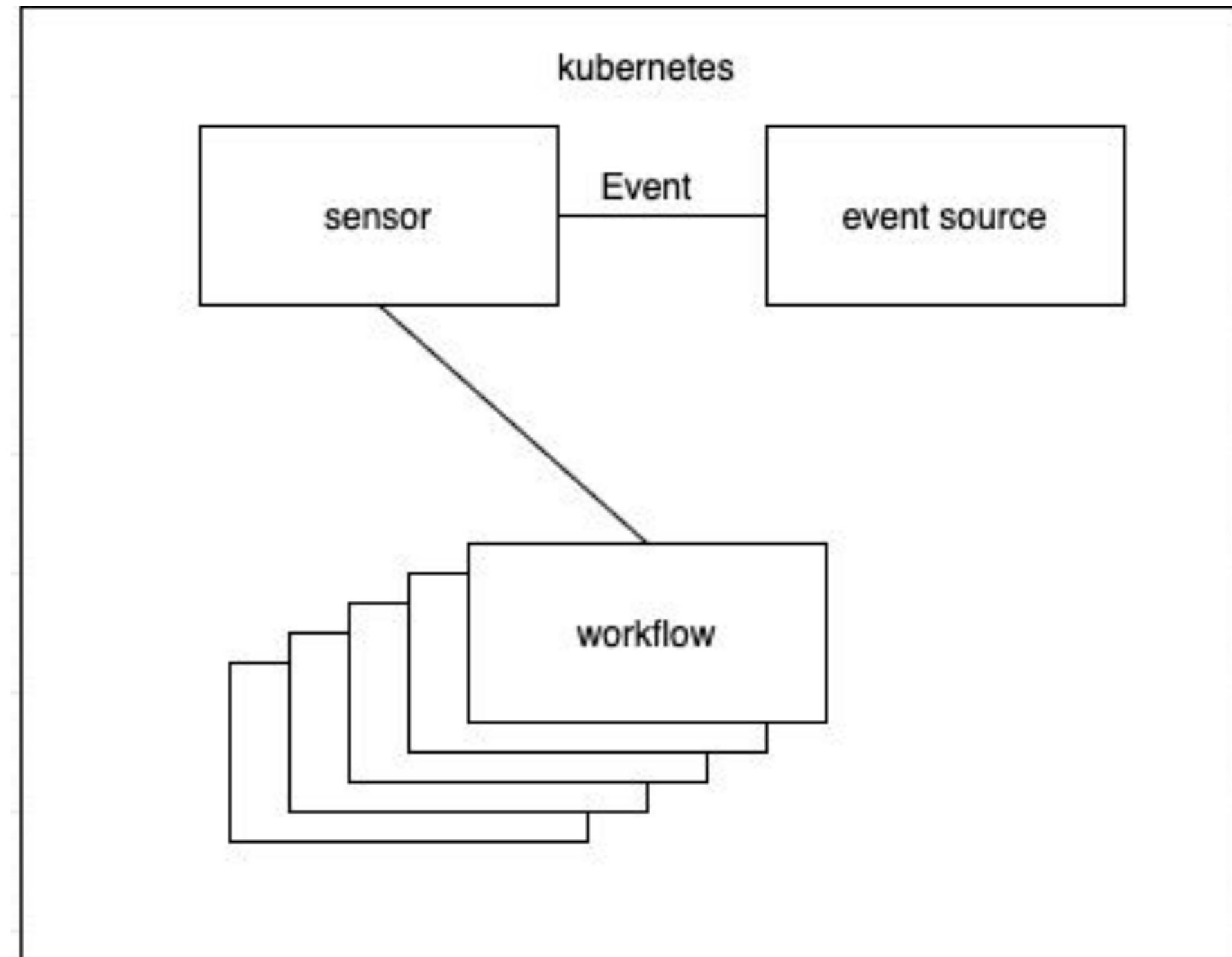
- Generic Event mechanism
- Kubernetes native
- Connects several sources such as AMPQ, SQS, PubSub, Kafka, MQTT, Slack, Webhooks
- cloudevents.io compliant





Argo Events entities

- **EventSource** - where to read events from
- **Trigger** - what to do when an event happens
- **Sensor** - connects sources and triggers
- **EventBus** - connects Sources and Sensors together



Creating events from webhooks

```
apiVersion: argoproj.io/v1alpha1
kind: EventSource
metadata:
  name: webhook
spec:
  service:
    ports:
      - port: 12000
        targetPort: 12000
  webhook:
    # event-source can run multiple HTTP servers. Simply define a unique port to start a new HTTP server
    # example:
    # port to run HTTP server on
    port: "12000"
    # endpoint to listen to
    endpoint: /example
    # HTTP request method to allow. In this case, only POST requests are accepted
    method: POST
```



```
apiVersion: argoproj.io/v1alpha1
kind: Sensor
metadata:
  name: webhook
spec:
  template:
    serviceAccountName: operate-workflow-sa
  dependencies:
    - name: test-dep
      eventSourceName: webhook
      eventName: example
  triggers:
    - template:
        name: webhook-workflow-trigger
      k8s:
        operation: create
        source:
          resource:
            apiVersion: argoproj.io/v1alpha1
            kind: Workflow
            metadata:
              generateName: webhook-
            spec:
              entrypoint: whalesay
              arguments:
                parameters:
                  - name: message
                    # the value will get overridden by event payload from test-dep
                    value: hello world
```

Starting a workflow from a webhook event



Argo Workflows UI also works for Argo Events

The screenshot displays the Argo Workflows UI interface for an event flow. The top navigation bar includes the text "Event Flow / workflow-playground" and a series of action buttons: "CREATE EVENT SOURCE", "CREATE SENSOR", "SHOW EVENT-FLOW", "SHOW WORKFLOWS", and "COLLAPSE/EXPAND HIDDEN NODES". Below the navigation bar is a search bar with a "Search" input field and several icons for navigation and search.

The main area shows a flow diagram for an event source named "example-with-interval". The flow starts with a "calendar" node (blue icon with a clock) which triggers two "sensor" nodes (blue icons with a signal tower) via "test-dep" dependencies. The top sensor triggers a "log" node (blue icon with a document), which then triggers a "log-trigger" node (blue icon with a document). The bottom sensor triggers an "argo-workflow-trigger" node (blue icon with a list). Both "log-trigger" and "argo-workflow-trigger" nodes trigger three "workflow" nodes (red circles with a white 'X') via "test-dep" dependencies. The workflow nodes are "calendar-2d2pp", "calendar-q66lm", and "calendar-hm7z9".

```
graph LR;
    calendar[calendar] -- test-dep --> sensor1[sensor];
    calendar -- test-dep --> sensor2[sensor];
    sensor1 -- test-dep --> log[log];
    sensor2 -- test-dep --> argo_workflow_trigger[argo-workflow-trigger];
    log -- test-dep --> workflow1[calendar-2d2pp];
    argo_workflow_trigger -- test-dep --> workflow2[calendar-q66lm];
    argo_workflow_trigger -- test-dep --> workflow3[calendar-hm7z9];
```



Use Cases



Argo CD and Argo Rollouts



QA



Staging



Production US



UAT



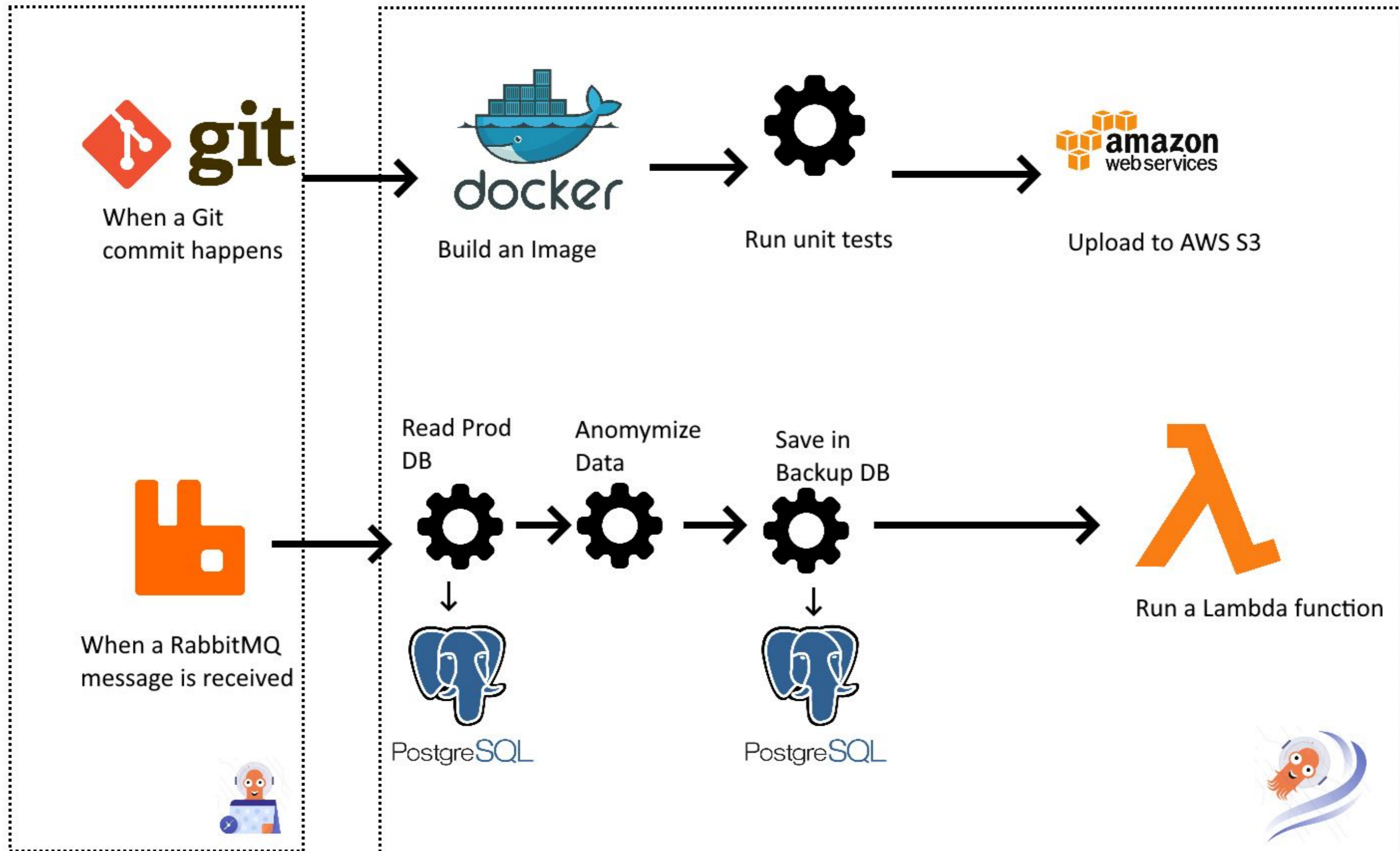
Load Testing



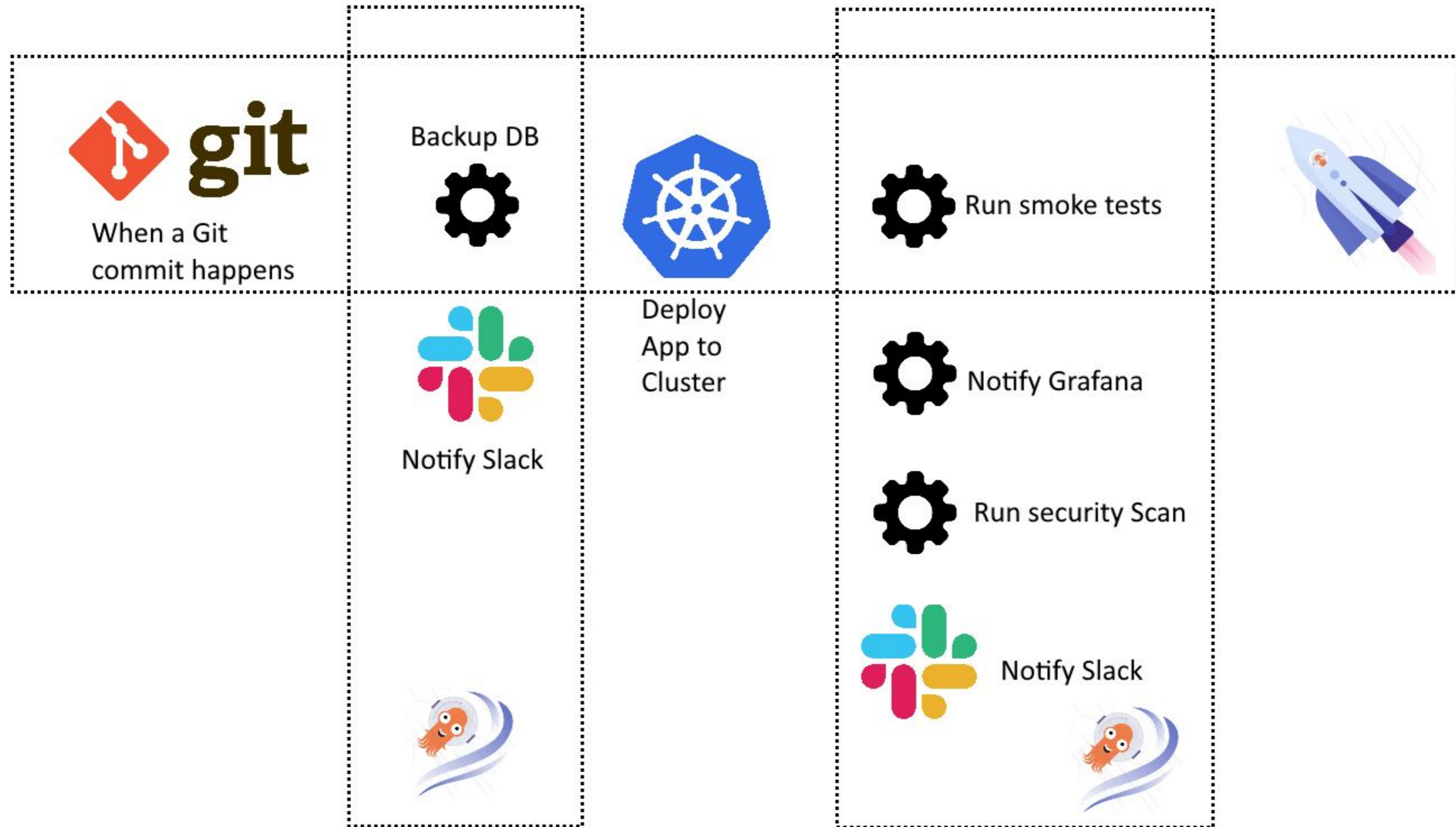
Production EU



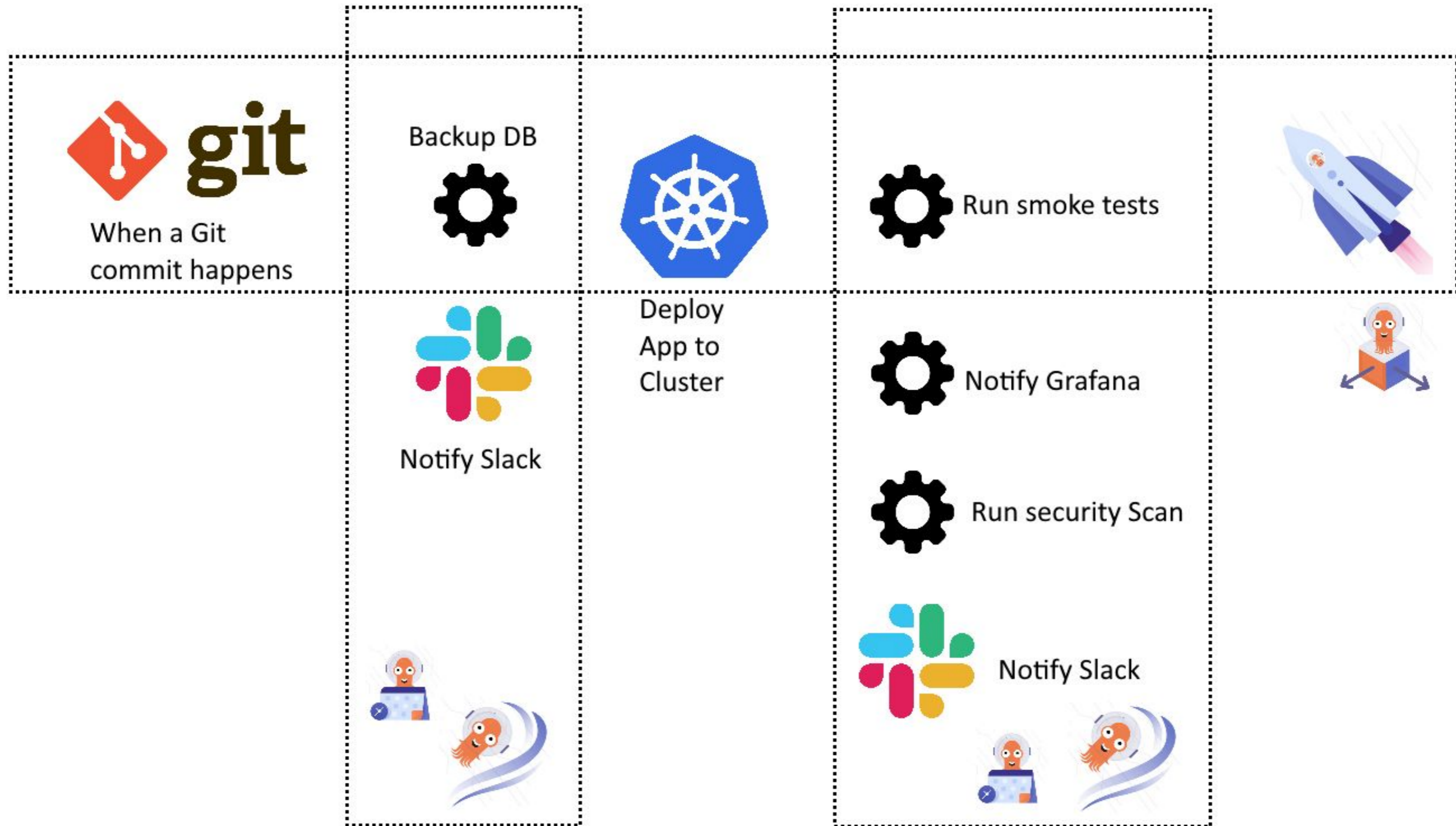
Argo Workflows and Argo Events



Argo CD and Argo Workflows



All 4 Argo projects (developer portal)





Thank you!

Questions: kostis.kapelonis@octopus.com

GitOps/Argo CD certification learning.codefresh.io

CNCF Slack <https://slack.cncf.io/>

Blog <https://blog.argoproj.io/>

 Octopus Deploy

Backup Slides



GitOps Principles

v1.0.0

1 Declarative

A **system** managed by GitOps must have its desired state expressed **declaratively**.

2 Versioned and Immutable

Desired state is **stored** in a way that enforces immutability, versioning and retains a complete version history.

3 Pulled Automatically

Software agents automatically pull the desired state declarations from the source.

4 Continuously Reconciled

Software agents **continuously** observe actual system state and **attempt to apply** the desired state.

From [OpenGitOps.dev](https://open-gitops.dev)

Project history

1. Startup Applatix was formed (2015)
2. Argo Workflows was released by Applatix (2017)
3. Applatix was acquired by Intuit (2018)
4. Argo CD and Argo Rollouts were created by Intuit (2018 and 2019)
5. Argo Events was donated by Blackrock Inc (2018)
6. Incubating open source software of the CNCF (accepted in 2020)
7. Graduated from CNCF in 2022 🧐

