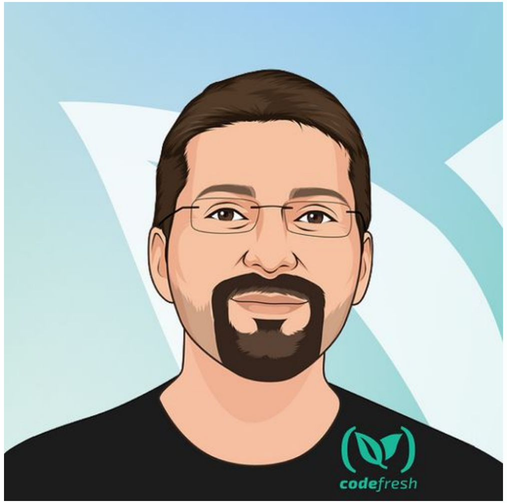
codefresh

Argo Rollouts for Stateful services

ArgoCon EU 2024

Your host



Kostis Kapelonis

kostis@codefresh.io

Developer Advocate - Codefresh

Argo Maintainer

Co-author of GitOps certification with Argo -> **http://learning.codefresh.io**

About Codefresh (acquired by Octopus Deploy)

- ## Modern Deployment Platform

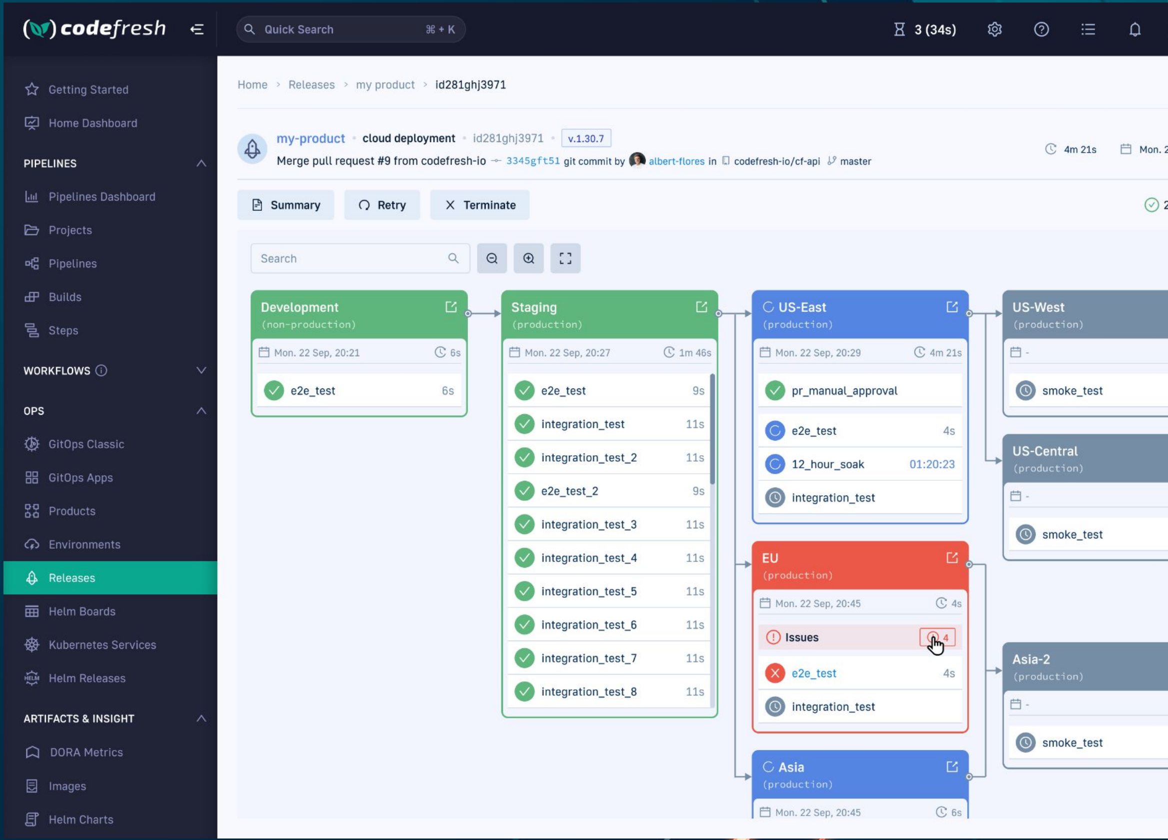
  Comes with CI, CD and GitOps modules

- ## Enterprise Ready

  Code-to-cloud visibility across apps and clusters

- ## Continuous Delivery

  Progressive delivery without compromising stability powered by Argo CD and Argo Rollouts



my-product • cloud deployment • id281ghj3971 • v.1.30.7

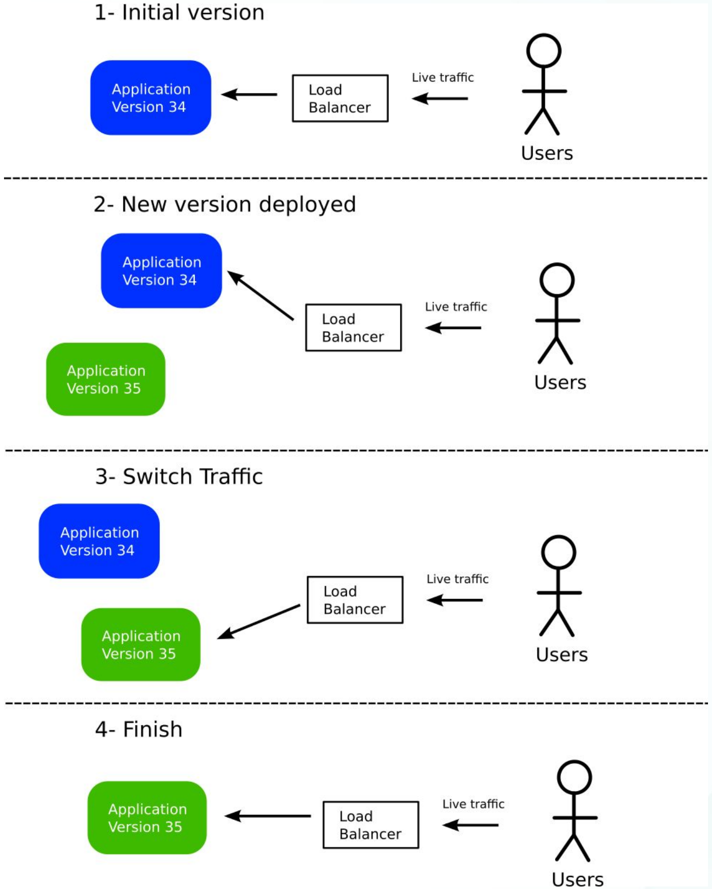
Merge pull request #9 from codefresh-io 3345gft51 git commit by albert-flores in codefresh-io/cf-api master

Agenda

1. Problem statement
2. Kubernetes Downward API
3. Argo Rollouts ephemeral labels
4. Demo with Argo Rollouts and RabbitMQ

codefresh

Argo Rollouts with stateless services



1- Initial version

Application Version 34 ← Load Balancer ← Live traffic ← Users

2- New version deployed

Application Version 34 ← Load Balancer ← Live traffic ← Users

Application Version 35

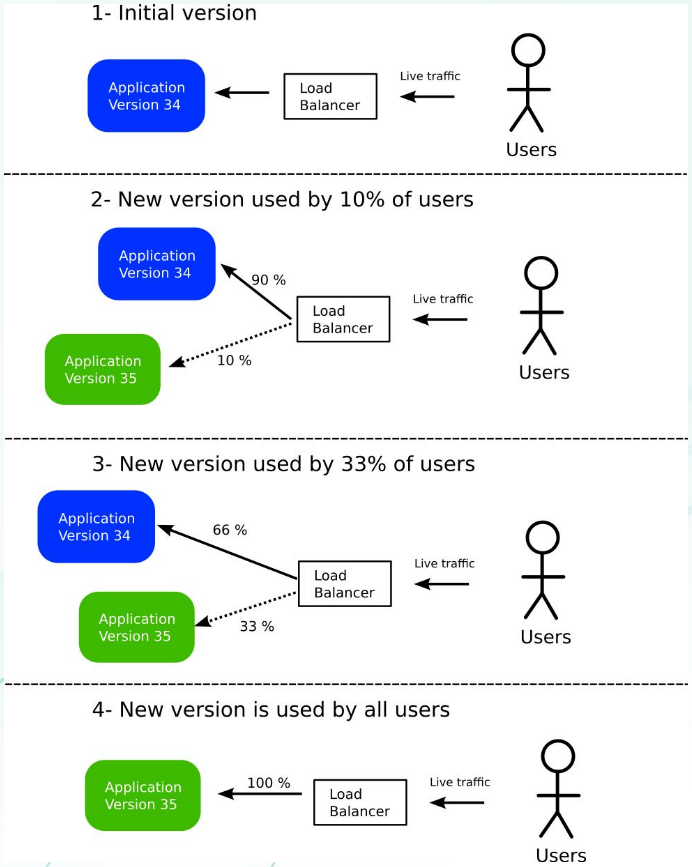
3- Switch Traffic

Application Version 34

Application Version 35 ← Load Balancer ← Live traffic ← Users

4- Finish

Application Version 35 ← Load Balancer ← Live traffic ← Users



1- Initial version

Application Version 34 ← Load Balancer ← Live traffic ← Users

2- New version used by 10% of users

Application Version 34 ← 90 % ← Load Balancer ← Live traffic ← Users

Application Version 35 ← 10 % ← Load Balancer

3- New version used by 33% of users

Application Version 34 ← 66 % ← Load Balancer ← Live traffic ← Users

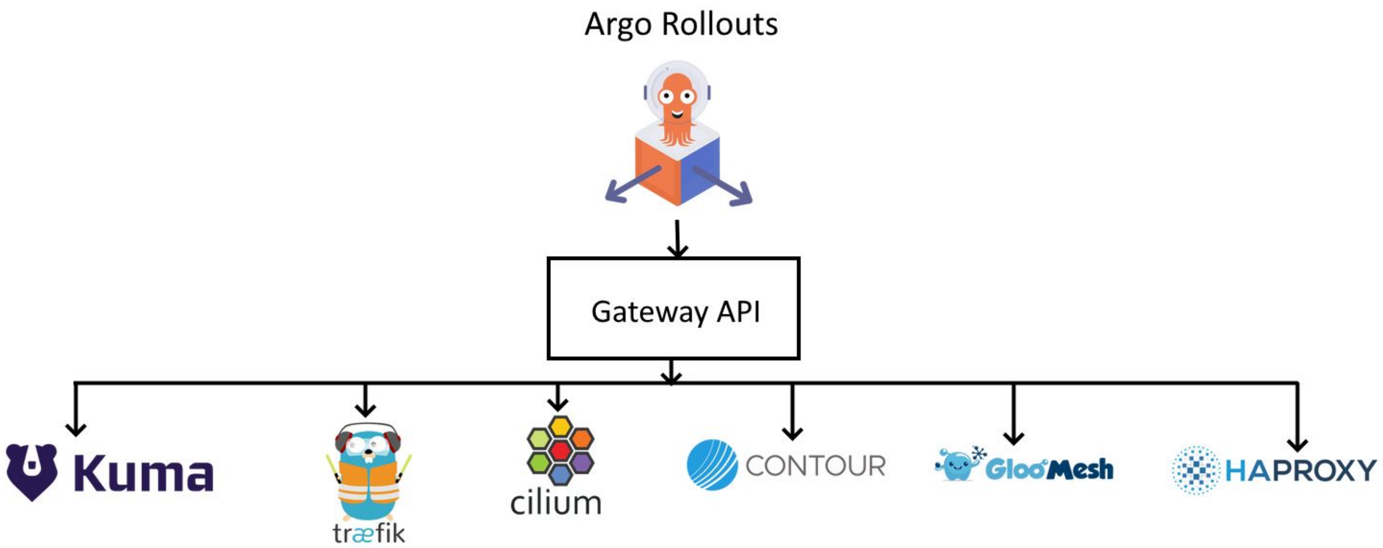
Application Version 35 ← 33 % ← Load Balancer

4- New version is used by all users

Application Version 35 ← 100 % ← Load Balancer ← Live traffic ← Users

codefresh

Many traffic providers (including Gateway API)

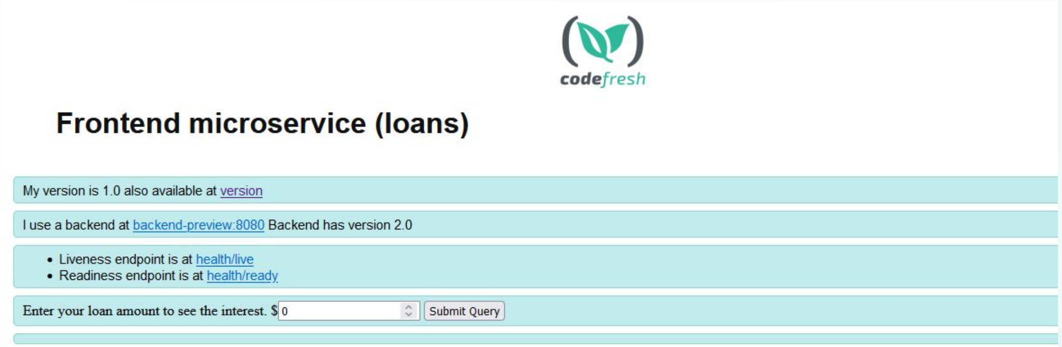


Argo Rollouts

Gateway API

https://github.com/argoproj-labs/rollouts-plugin-trafficrouter-gatewayapi

Argo Rollouts assumes http connections



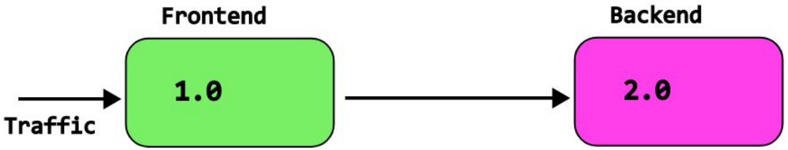
Frontend microservice (loans)

My version is 1.0 also available at version

I use a backend at backend-preview:8080 Backend has version 2.0

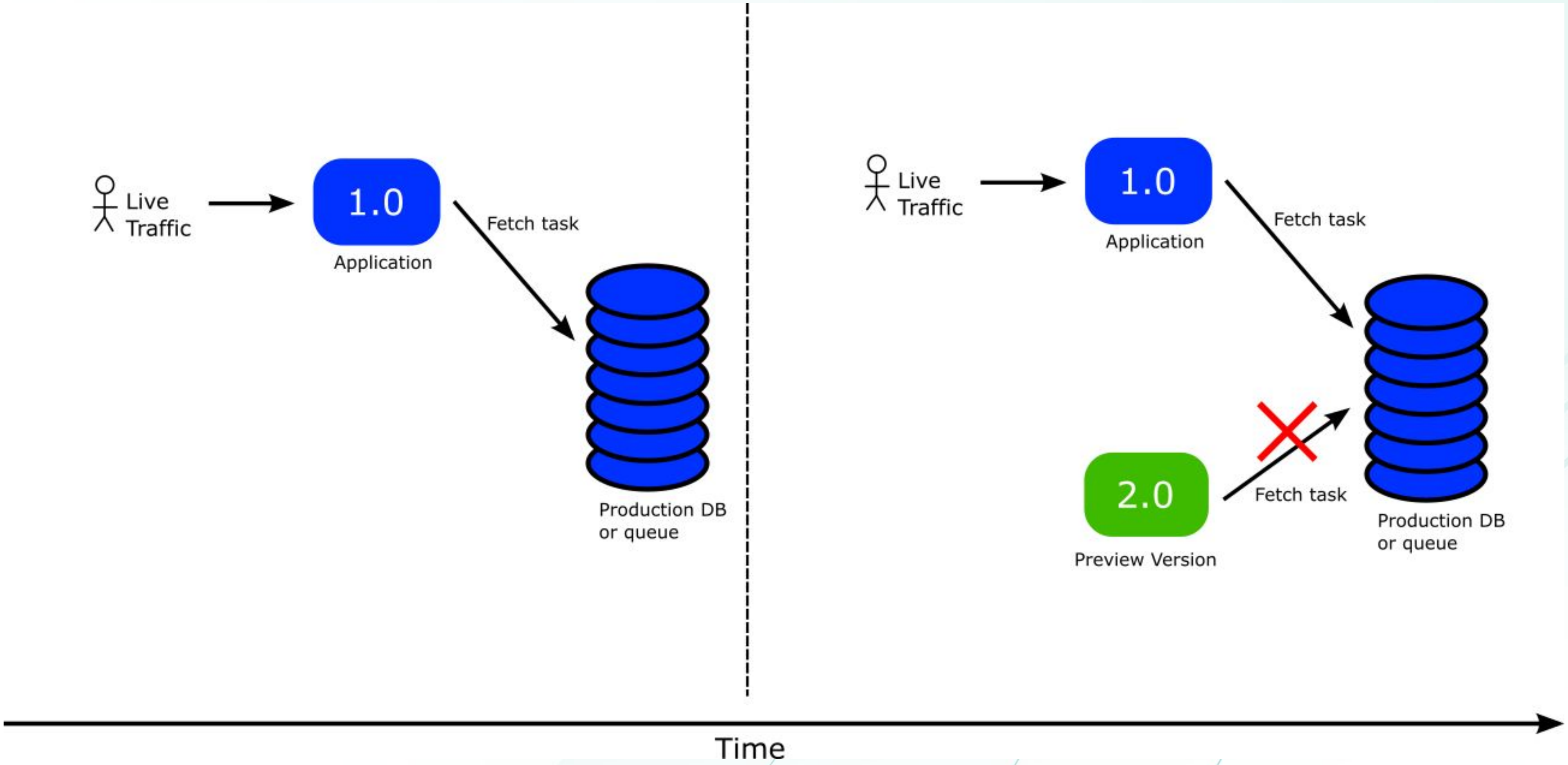
- Liveness endpoint is at health/live
- Readiness endpoint is at health/ready

Enter your loan amount to see the interest. $ 0 Submit Query



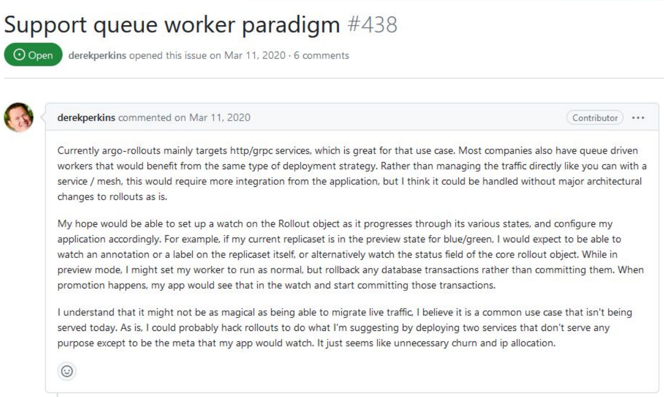
https://codefresh.io/blog/multi-service-progressive-delivery-with-argo-rollouts/

Argo Rollouts with stateful services


Live Traffic
1.0
Application
Fetch task
Production DB or queue
Live Traffic
1.0
Application
Fetch task
2.0
Preview Version
Fetch task
Production DB or queue
Time

codefresh

Constant question for new Argo Rollouts users



Support queue worker paradigm #438

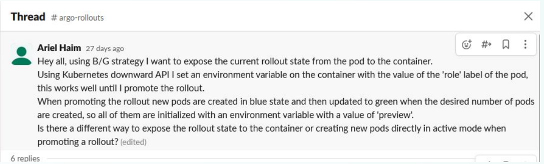
Open · derekperkins opened this issue on Mar 11, 2020 · 6 comments

derekperkins commented on Mar 11, 2020 · Contributor

Currently argo-rollouts mainly targets http/grpc services, which is great for that use case. Most companies also have queue driven workers that would benefit from the same type of deployment strategy. Rather than managing the traffic directly like you can with a service / mesh, this would require more integration from the application, but I think it could be handled without major architectural changes to rollouts as is.

My hope would be able to set up a watch on the Rollout object as it progresses through its various states, and configure my application accordingly. For example, if my current replicaset is in the preview state for blue/green, I would expect to be able to watch an annotation or a label on the replicaset itself, or alternatively watch the status field of the core rollout object. While in preview mode, I might set my worker to run as normal, but rollback any database transactions rather than committing them. When promotion happens, my app would see that in the watch and start committing those transactions.

I understand that it might not be as magical as being able to migrate live traffic, I believe it is a common use case that isn't being served today. As is, I could probably hack rollouts to do what I'm suggesting by deploying two services that don't serve any purpose except to be the meta that my app would watch. It just seems like unnecessary churn and ip allocation.

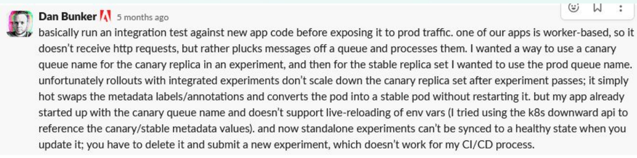


Thread # argo-rollouts

Ariel Haim 27 days ago

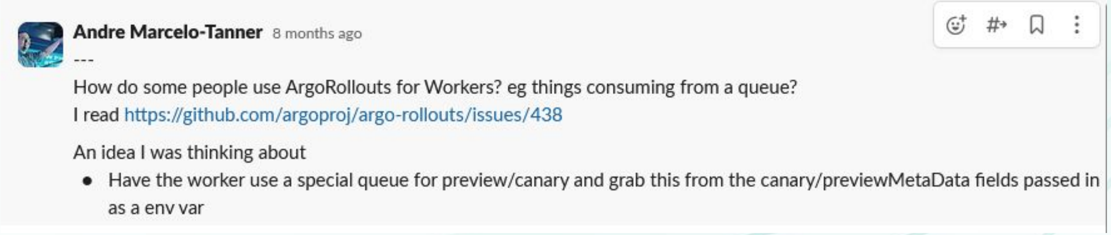
Hey all, using B/G strategy I want to expose the current rollout state from the pod to the container.
Using Kubernetes downward API I set an environment variable on the container with the value of the 'role' label of the pod, this works well until I promote the rollout.
When promoting the rollout new pods are created in blue state and then updated to green when the desired number of pods are created, so all of them are initialized with an environment variable with a value of 'preview'.
Is there a different way to expose the rollout state to the container or creating new pods directly in active mode when promoting a rollout? (edited)

6 replies



Dan Bunker 5 months ago

basically run an integration test against new app code before exposing it to prod traffic. one of our apps is worker-based, so it doesn't receive http requests, but rather plucks messages off a queue and processes them. I wanted a way to use a canary queue name for the canary replica in an experiment, and then for the stable replica set I wanted to use the prod queue name. unfortunately rollouts with integrated experiments don't scale down the canary replica set after experiment passes; it simply hot swaps the metadata labels/annotations and converts the pod into a stable pod without restarting it. but my app already started up with the canary queue name and doesn't support live-reloading of env vars (I tried using the k8s downward api to reference the canary/stable metadata values). and now standalone experiments can't be synced to a healthy state when you update it; you have to delete it and submit a new experiment, which doesn't work for my CI/CD process.

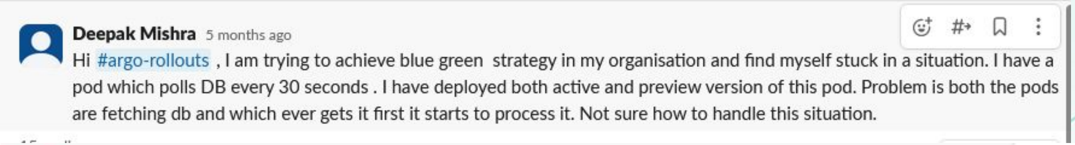


Andre Marcelo-Tanner 8 months ago

How do some people use ArgoRollouts for Workers? eg things consuming from a queue?
I read https://github.com/argoproj/argo-rollouts/issues/438

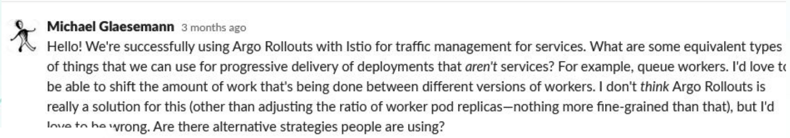
An idea I was thinking about

- Have the worker use a special queue for preview/canary and grab this from the canary/previewMetaData fields passed in as a env var



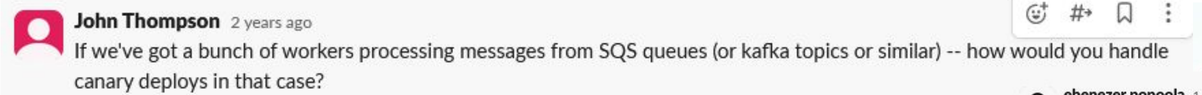
Deepak Mishra 5 months ago

Hi #argo-rollouts , I am trying to achieve blue green strategy in my organisation and find myself stuck in a situation. I have a pod which polls DB every 30 seconds . I have deployed both active and preview version of this pod. Problem is both the pods are fetching db and which ever gets it first it starts to process it. Not sure how to handle this situation.



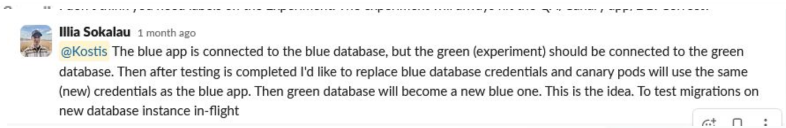
Michael Glaesemann 3 months ago

Hello! We're successfully using Argo Rollouts with Istio for traffic management for services. What are some equivalent types of things that we can use for progressive delivery of deployments that *aren't* services? For example, queue workers. I'd love to be able to shift the amount of work that's being done between different versions of workers. I don't *think* Argo Rollouts is really a solution for this (other than adjusting the ratio of worker pod replicas—nothing more fine-grained than that), but I'd love to be wrong. Are there alternative strategies people are using?



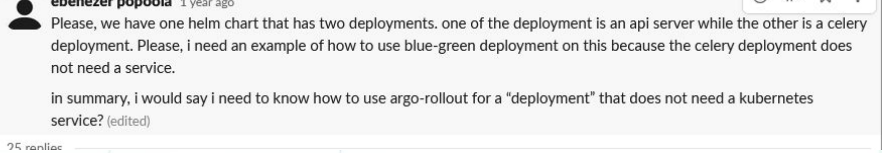
John Thompson 2 years ago

If we've got a bunch of workers processing messages from SQS queues (or kafka topics or similar) -- how would you handle canary deploys in that case?



Illia Sokalau 1 month ago

@Kostis The blue app is connected to the blue database, but the green (experiment) should be connected to the green database. Then after testing is completed I'd like to replace blue database credentials and canary pods will use the same (new) credentials as the blue app. Then green database will become a new blue one. This is the idea. To test migrations on new database instance in-flight

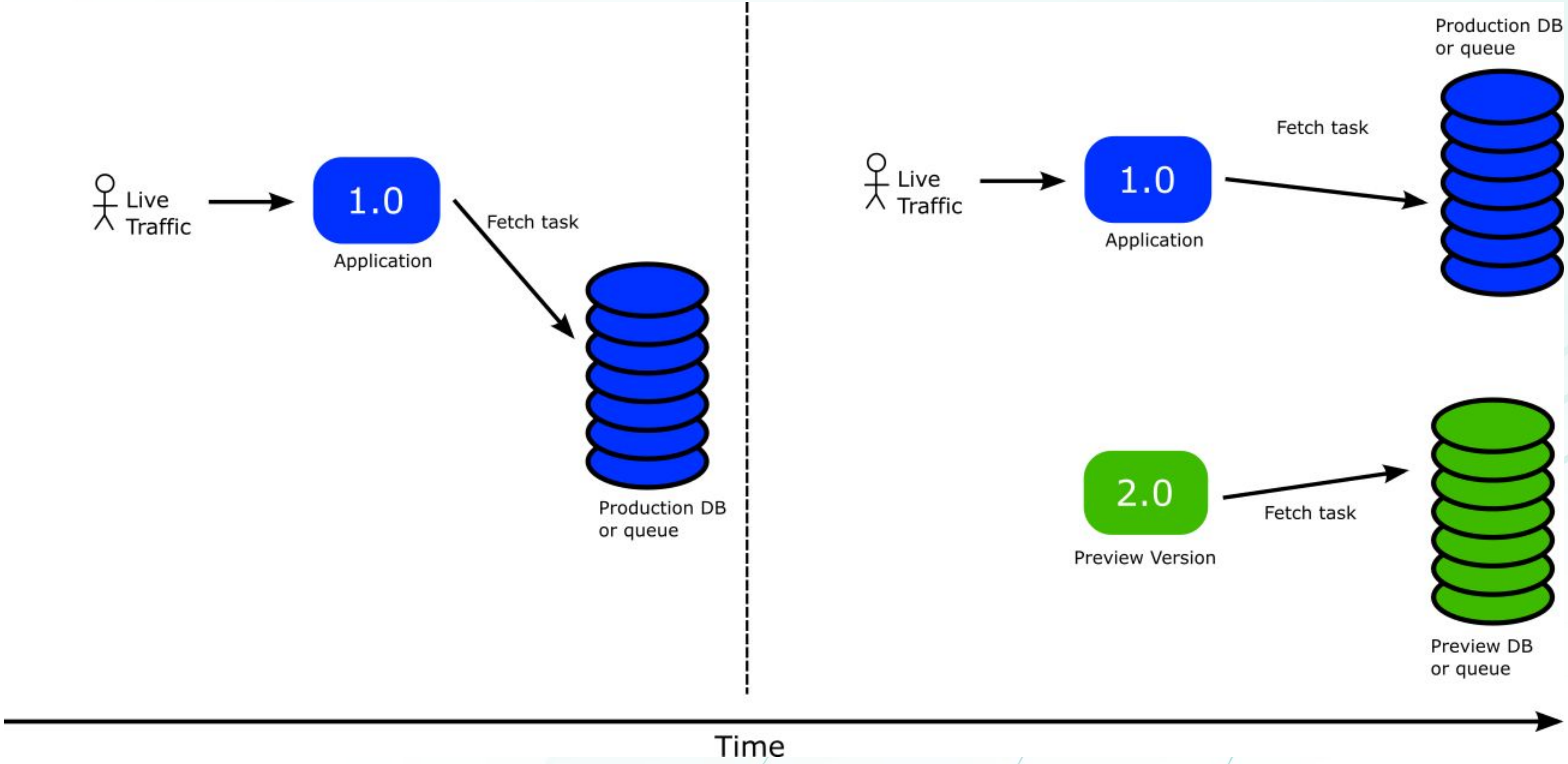


ebenezer popoola 1 year ago

Please, we have one helm chart that has two deployments. one of the deployment is an api server while the other is a celery deployment. Please, i need an example of how to use blue-green deployment on this because the celery deployment does not need a service.

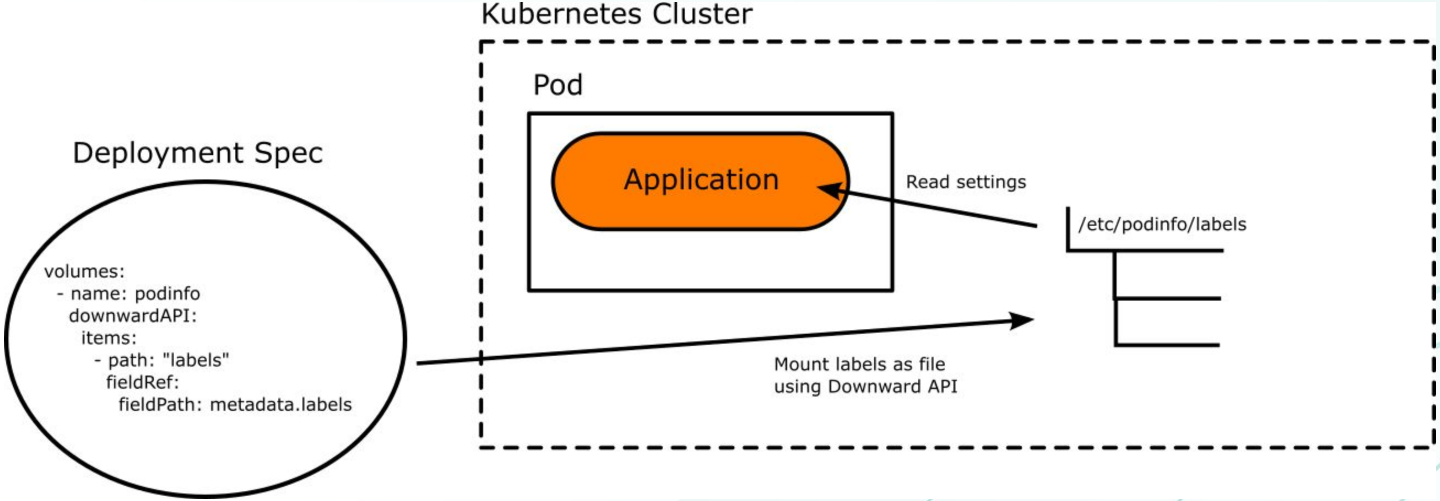
in summary, i would say i need to know how to use argo-rollout for a “deployment” that does not need a kubernetes service? (edited)

25 replies


Live Traffic
1.0
Application
Fetch task
Production DB or queue
Production DB or queue
Fetch task
Live Traffic
1.0
Application
2.0
Preview Version
Fetch task
Preview DB or queue
Time

The Kubernetes Downward API

Mount your labels as files (or env vars)



```
volumes:
  - name: podinfo
    downwardAPI:
      items:
        - path: "labels"
          fieldRef:
            fieldPath: metadata.labels
```

https://kubernetes.io/docs/concepts/workloads/pods/downward-api/

Make your configuration smarter

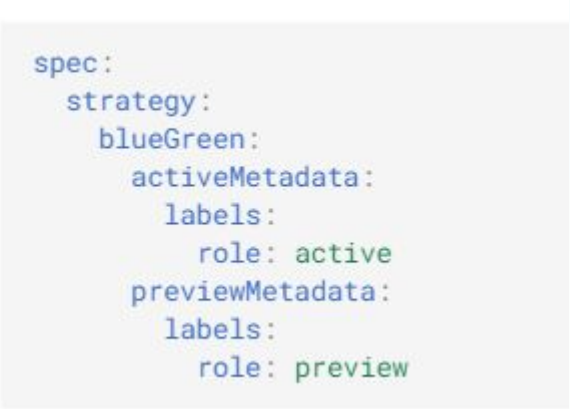
1. Have labels that denote role (stable or canary)
2. Mount these labels to your application
3. Have the application source code read them
4. Do NOT use environment variables. Load from files

codefresh

Argo Rollouts Ephemeral labels

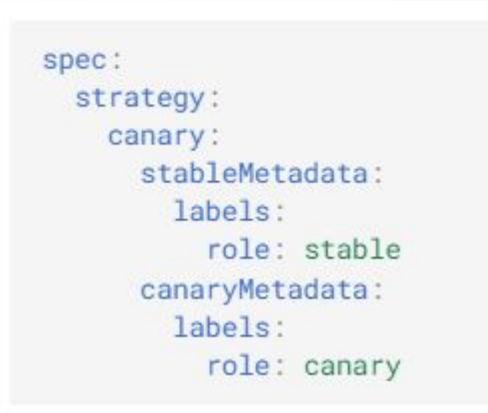
Let Argo Rollouts instruct the app automatically

Instruct Blue/green app of its "color"



```
spec:
  strategy:
    blueGreen:
      activeMetadata:
        labels:
          role: active
      previewMetadata:
        labels:
          role: preview
```

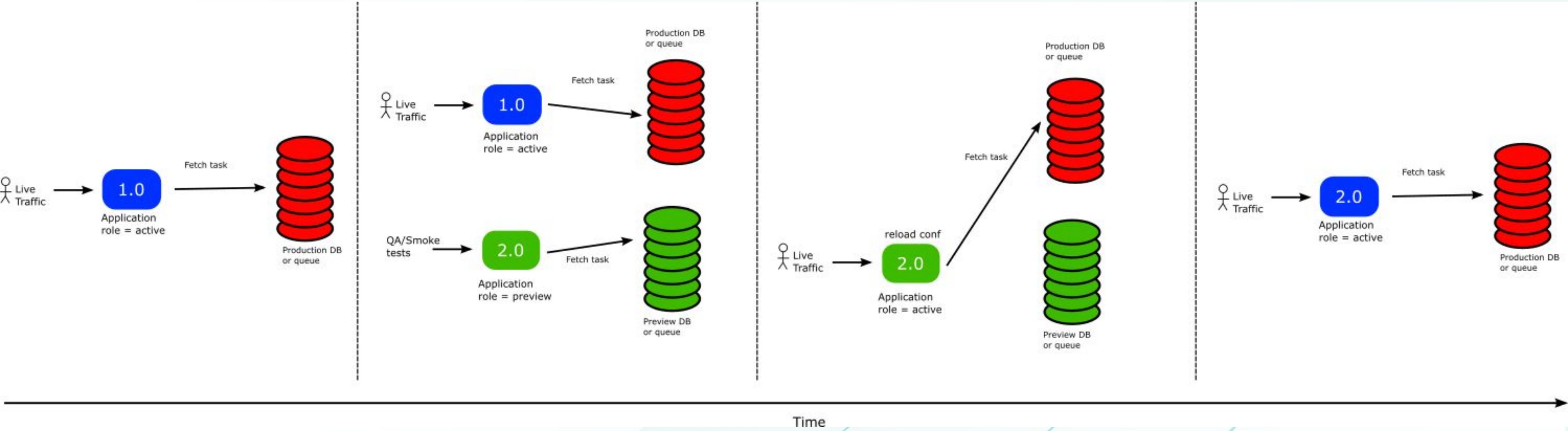
Instruct Canary application for its promotion "status"



```
spec:
  strategy:
    canary:
      stableMetadata:
        labels:
          role: stable
      canaryMetadata:
        labels:
          role: canary
```

https://argo-rollouts.readthedocs.io/en/stable/features/ephemeral-metadata/

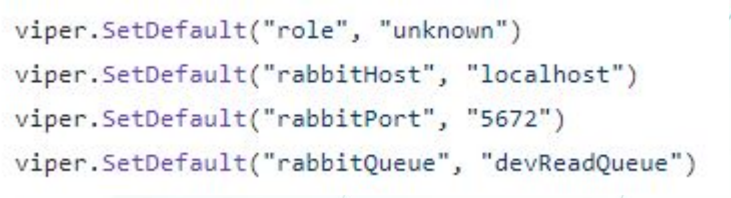
Full Process



Auto-reloading of Configuration

Make your application smarter

1. The application should read conf from files
2. DB/Queue URL must be configurable
3. Application should auto-reload conf on its own
4. You need to coordinate with your developers for this



```
viper.SetDefault("role", "unknown")
viper.SetDefault("rabbitHost", "localhost")
viper.SetDefault("rabbitPort", "5672")
viper.SetDefault("rabbitQueue", "devReadQueue")
```

Popular languages support

1. Viper Conf (Golang)
2. RefreshScope (Spring/Java)
3. chokidar/config (Node.js)
4. configparser/watchdog (Python)
5. yaml/listen (Ruby)
6. config/watchservice (Kotlin)
7. config/config-watch (Rust)
8. symfony/config-filesystem (PHP)

Coordinate with the Developers

- Photo by Sylvain Mauroux on Unsplash

codefresh

Demo Time

Argo Rollouts, Downward API, RabbitMQ, golang viper autoreload

https://github.com/kostis-codefresh/argo-rollouts-stateful-example

Argo Rollouts for stateful services

1. Use Kubernetes Downward API
2. Use Argo Rollouts ephemeral labels
3. Application should read configuration from files
4. Application should auto-reload its configuration
5. Enjoy !

Questions?

kostis@codefresh.io

codefresh

Argo Rollouts

Downward API

https://codefresh.io/blog/progressive-delivery-for-stateful-services-using-argo-rollouts/