

# **Stop Deploying Blind!**

## **Using Observability and**

## **Argo Rollouts to Light the Way**

ArgoCon NA 2024



# Your hosts today



**Anastasiia Gubska**

SRE/DevOps Engineer  
BT Group  
[gubska2@gmail.com](mailto:gubska2@gmail.com)



**Kostis Kapelonis**

Developer Advocate  
Codefresh by Octopus Deploy  
Argo Team member  
[kostis.kapelonis@octopus.com](mailto:kostis.kapelonis@octopus.com)



# Agenda

- 1 Blind deployments
- 2 How Observability can help
- 3 Argo Rollouts, Metrics and Tools
- 4 Minimum requirements for fully automated deployments
- 5 Best practices for adopting Argo Rollouts
- 6 Common pitfalls and mistakes



# The problem : Blind Deployments



# Learning about failed deployments from customers



Developers pushing new release  
to production



Users find out it doesn't work



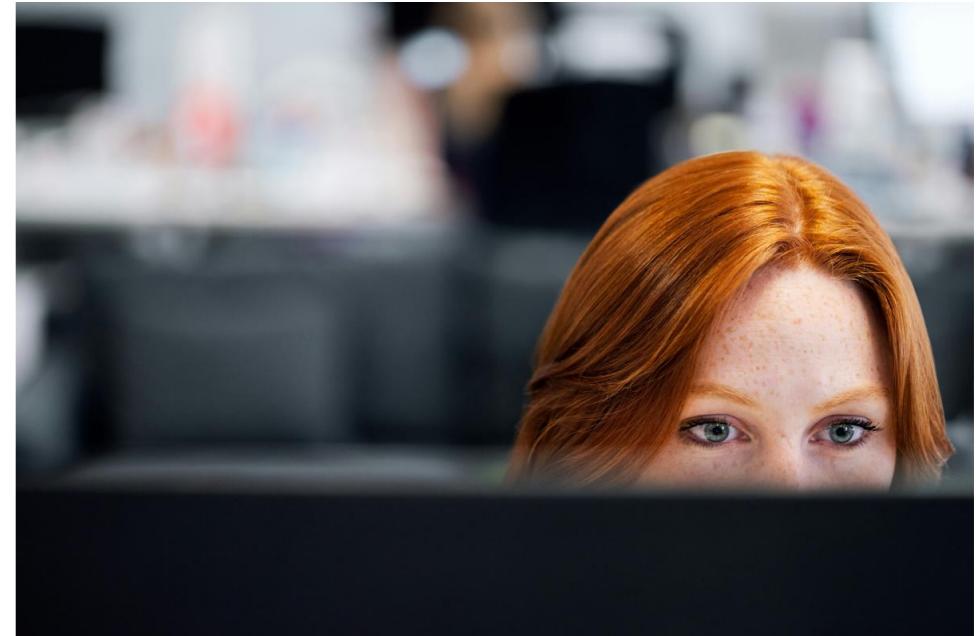
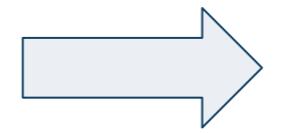
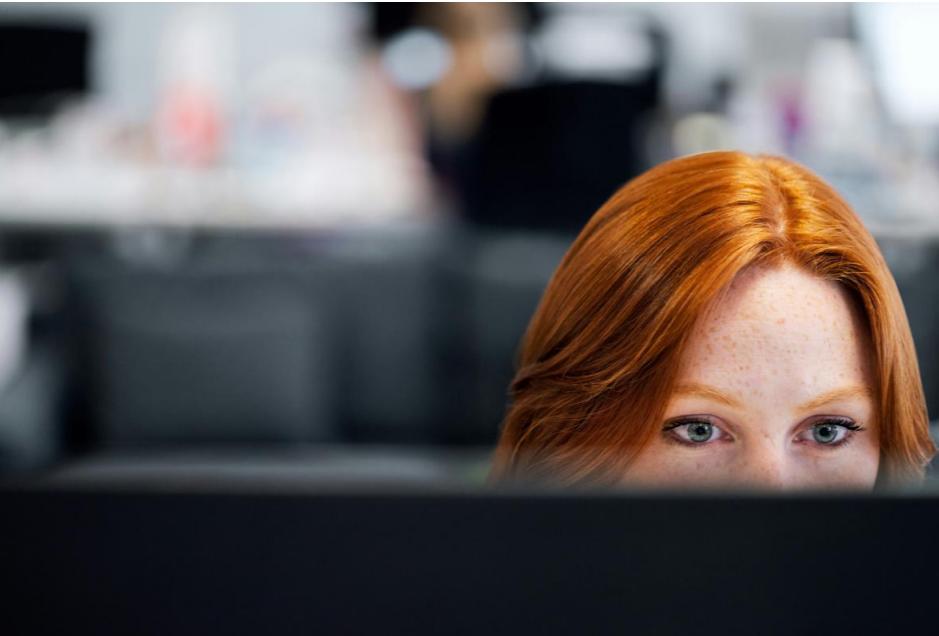
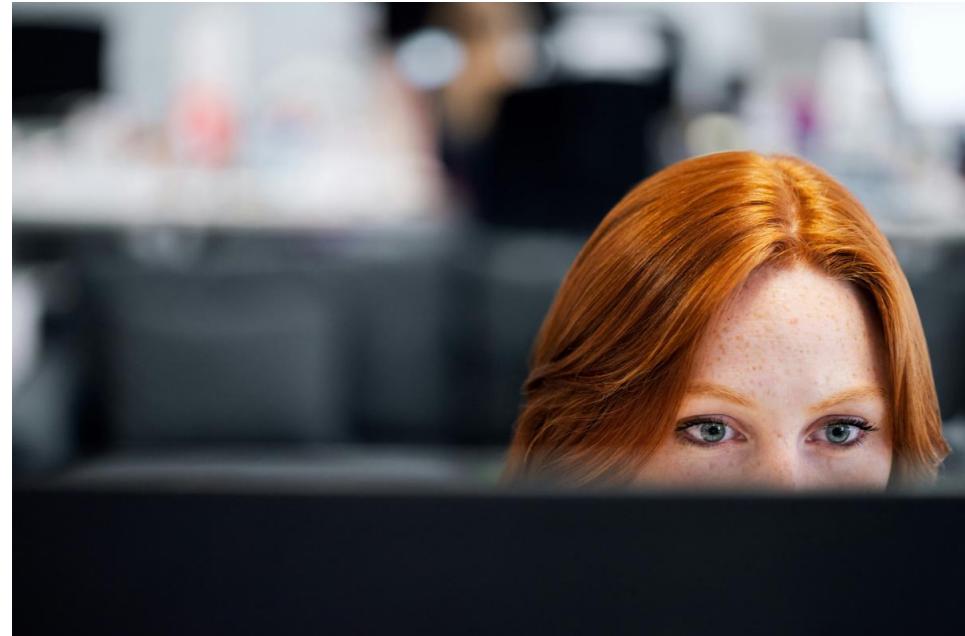
SRE team called in for a rollback



# Let's use metrics, logs and traces



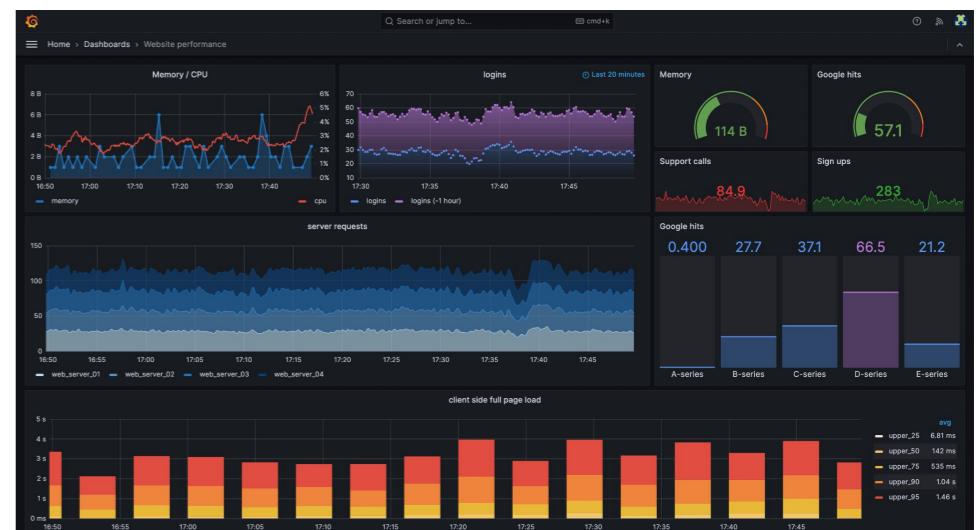
# Metrics checked by humans



3:00 PM  
deployed



3:30 PM  
looking at metrics



4:00 PM  
still looking at metrics



**“I love looking at my metrics for 2 hours after each deployment”**

**- said no one ever**



# How production deployments should happen



Deployment happens at 5.00 pm on Friday



5:15 the whole team is at the pub

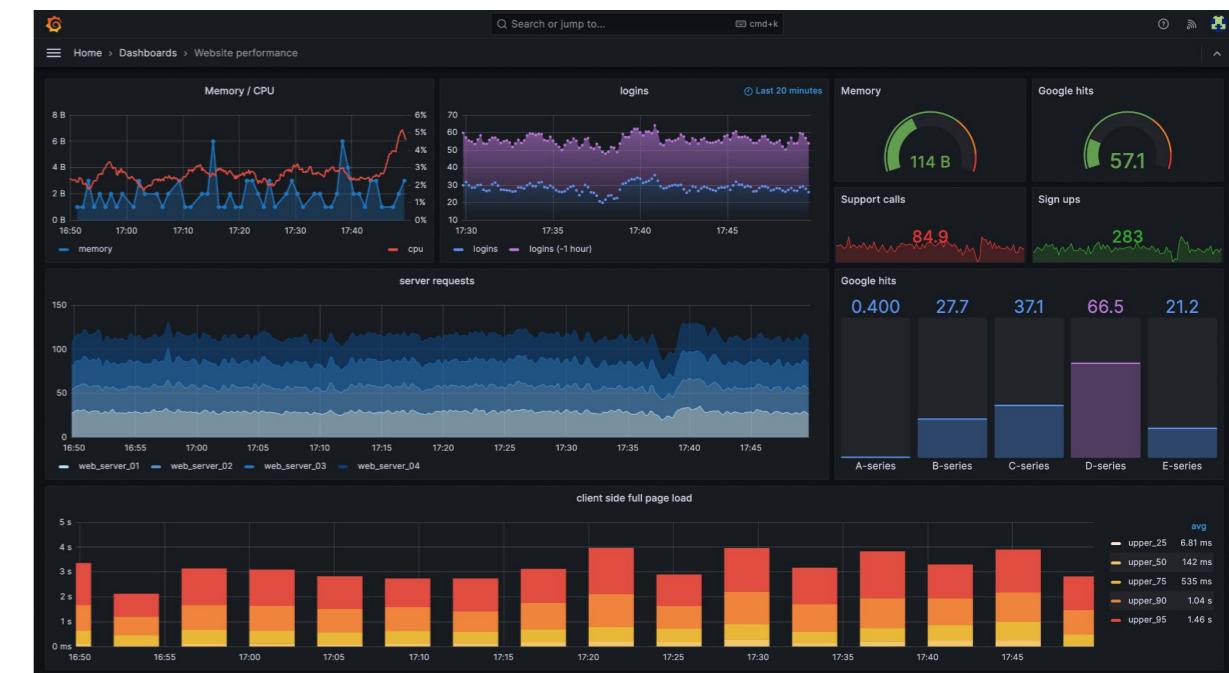


# How observability can help



# Why we need observability

- Learn about failed deployments before your users
- Decide quickly if deployment failed or not
- Compare historical data from previous deployments
- Automated monitoring and alerts even outside of deployments
- **Automated rollbacks WITHOUT human intervention.**



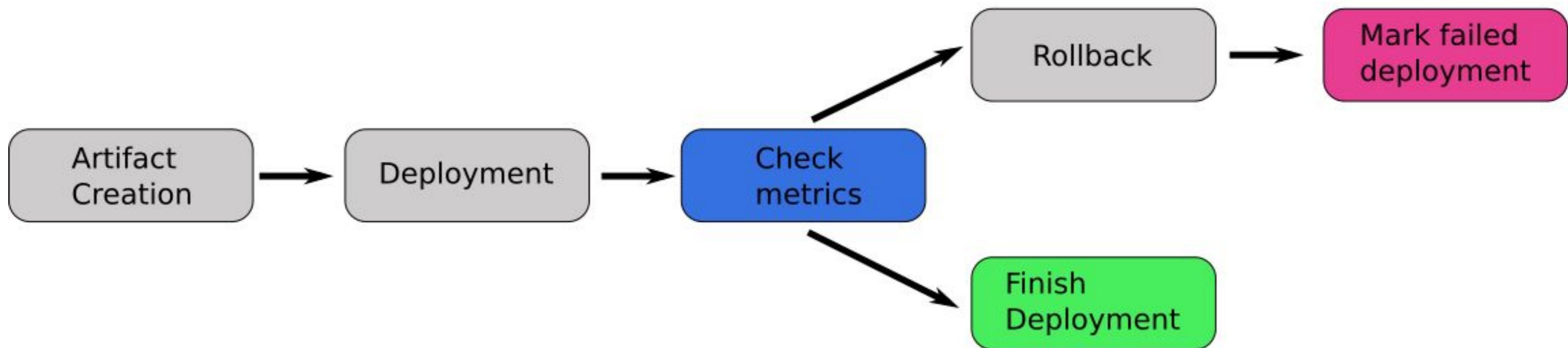
# Make metrics work for you

- Metrics should indicate if a deployment is successful or not.
  - If metrics are ok → **Done**
  - If metrics are not ok → Automatic **rollback**



# Our End Goal

Fully Automated Rollbacks



# Argo Rollouts





# Argo Rollouts

  2405

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

[Learn More](#)

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

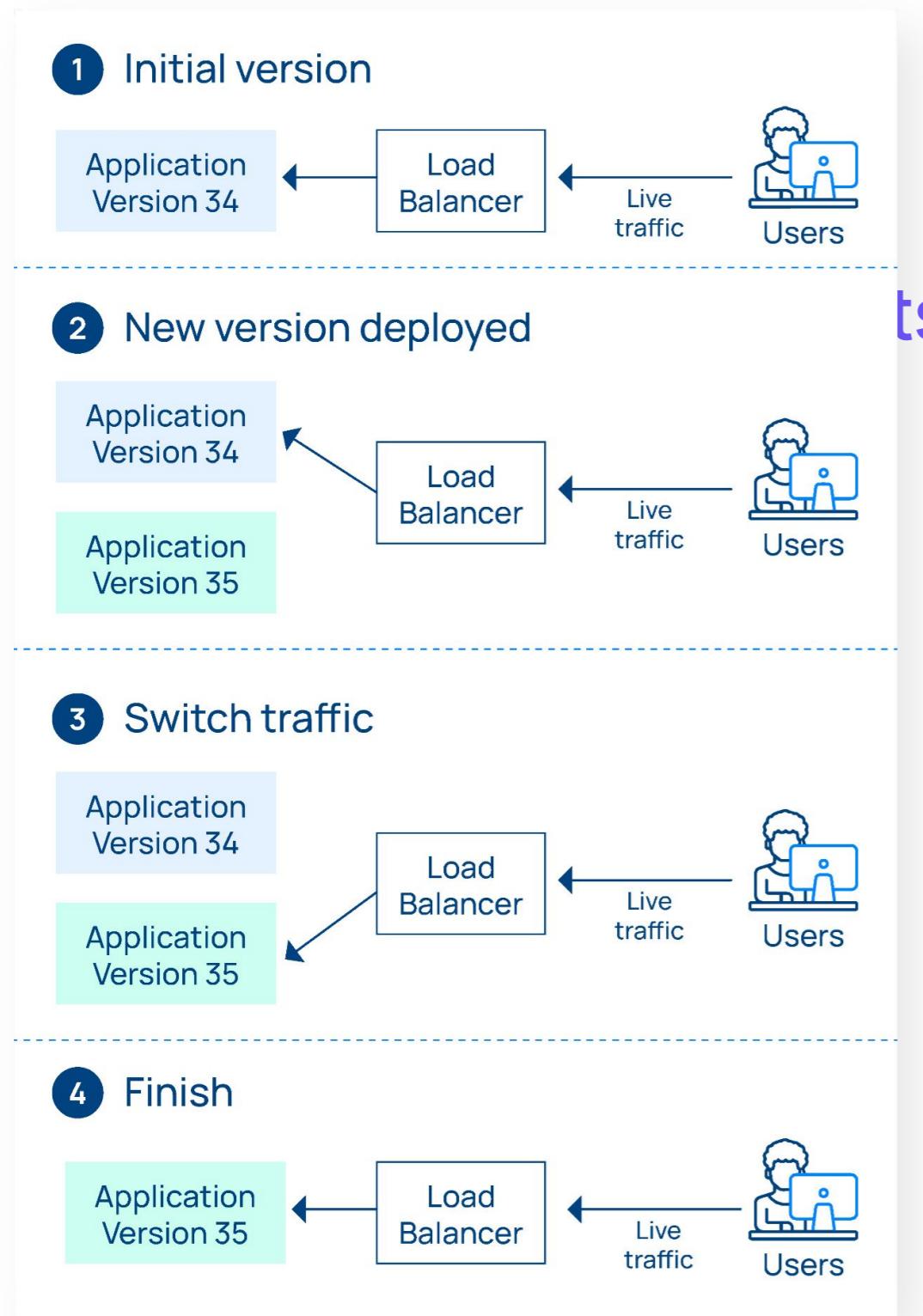


# Progressive delivery with Argo Rollouts

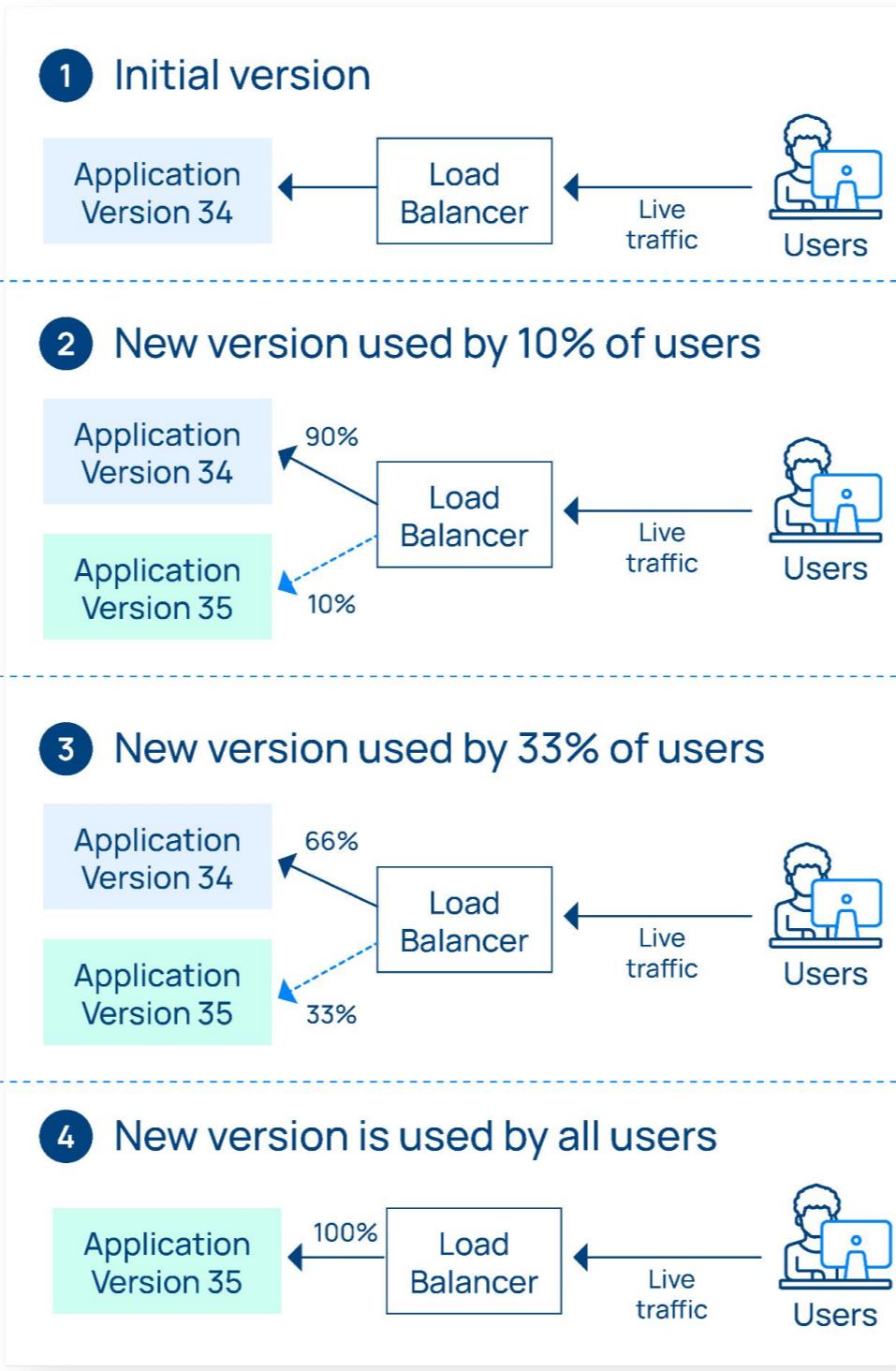
- Kubernetes native
- Standalone project
- Does **NOT** depend on Argo CD
- Blue/Green support
- Canary support
- A/B testing and other Experiments
- Zero downtime
- Automatic rollbacks based on metrics
- Installed on each deployment cluster



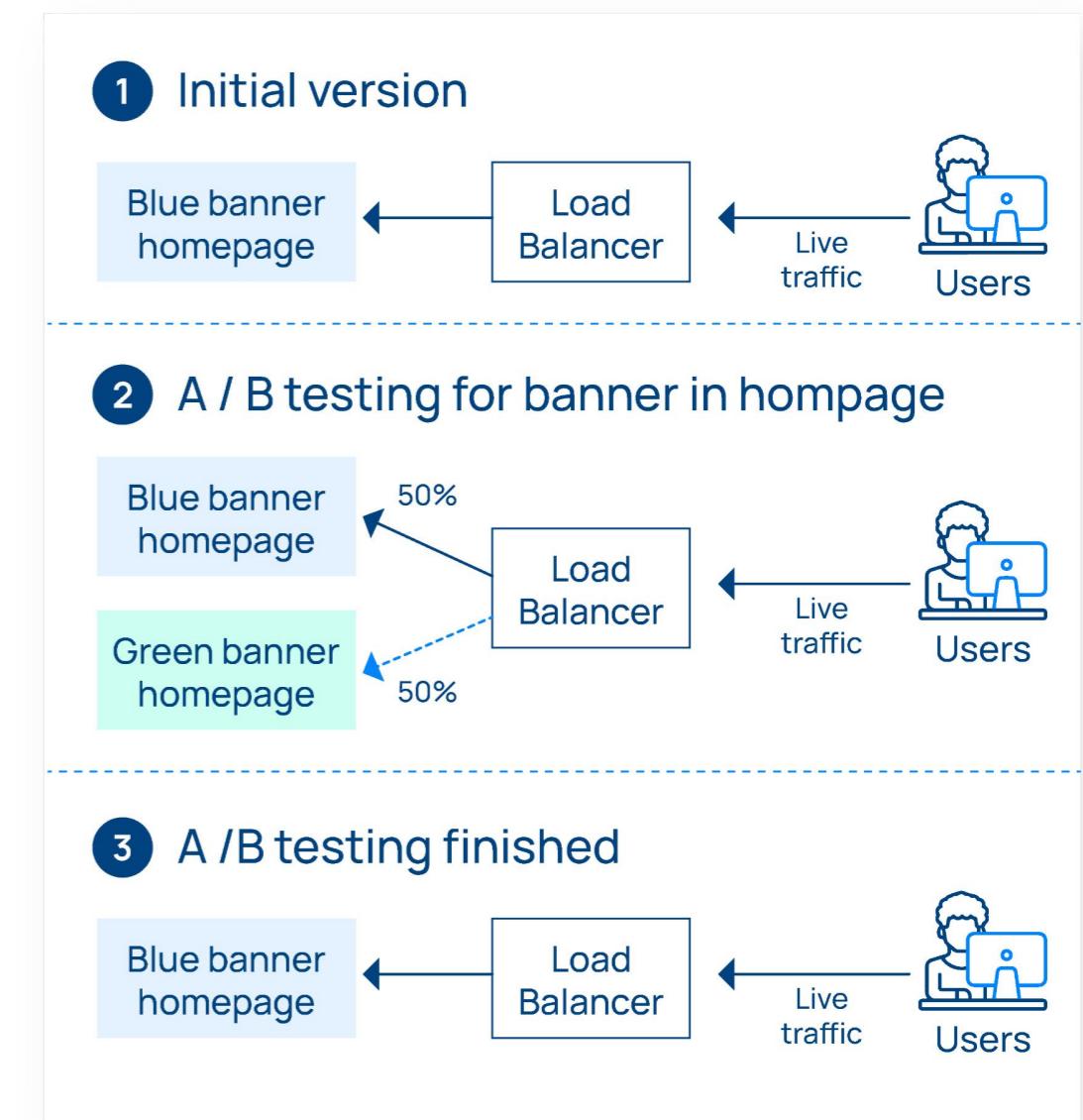
# Blue-Green Deployment



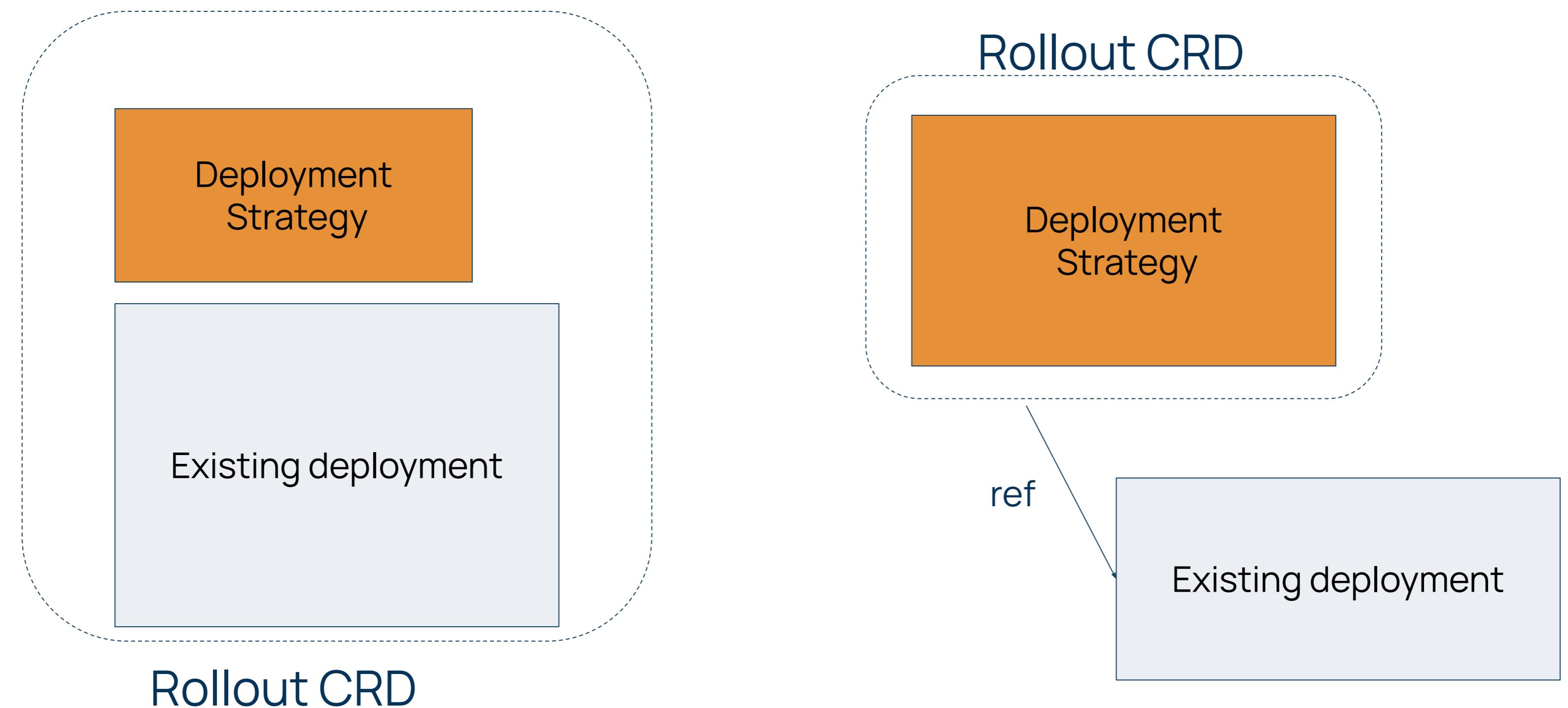
# Canary Release



# A/B testing



# How the Rollout resource works

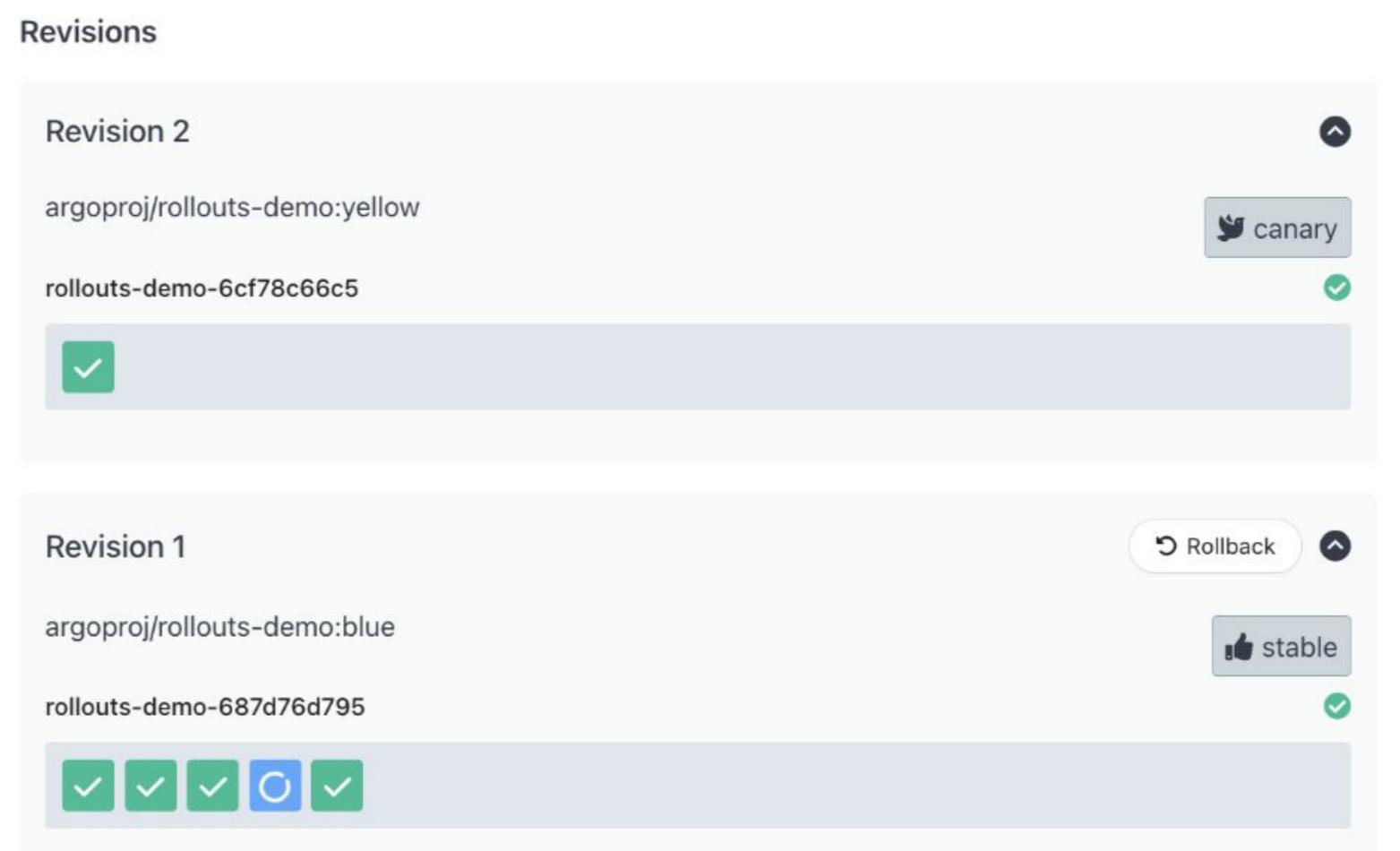
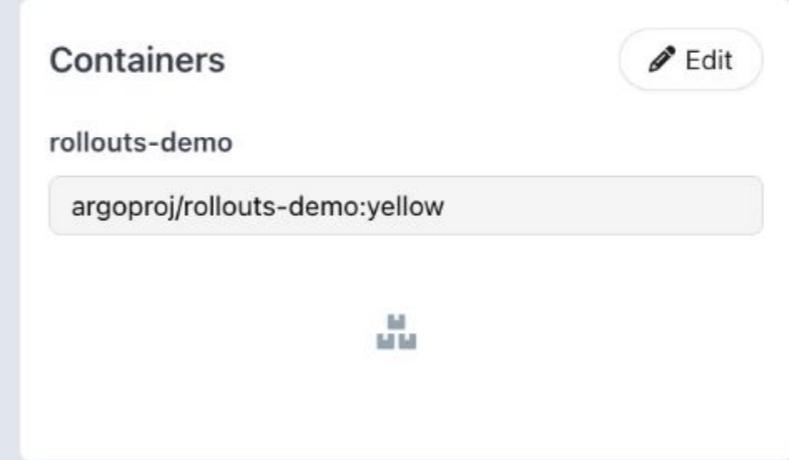
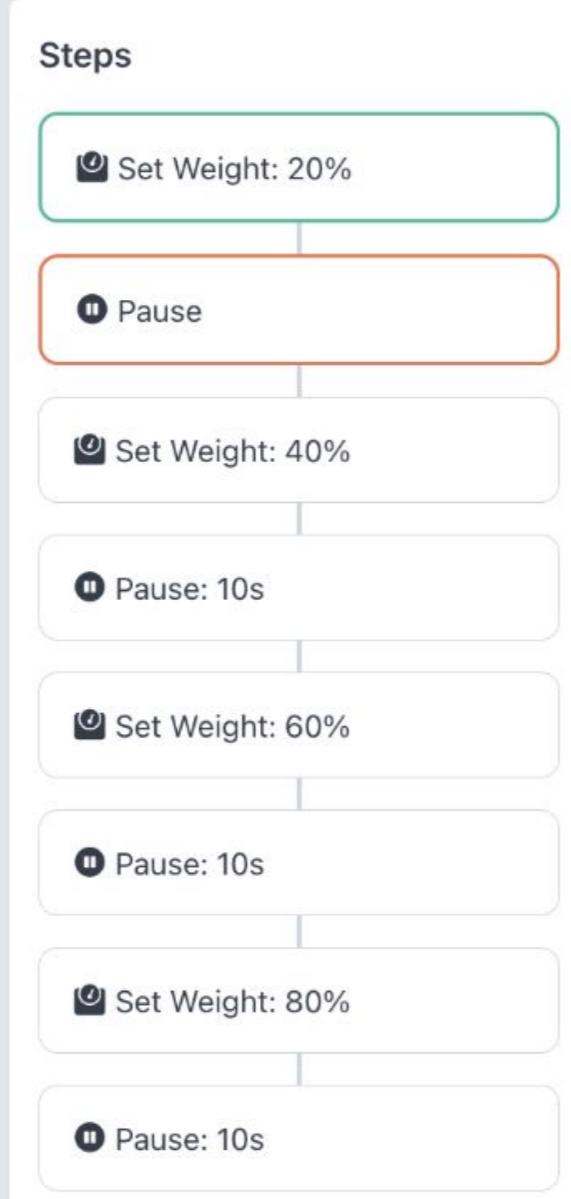


```
apiVersion: argoproj.io/v1alpha1
kind: Rollout
metadata:
  name: example-rollout
spec:
  replicas: 10
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
  spec:
    containers:
      - name: nginx
        image: nginx:1.15.4
        ports:
  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





# Minimum Requirements



# Minimum Requirements

- App capable of running multiple versions at the same time
- App shouldn't use shared/locked resources
- Argo Rollouts controller deployed on every cluster (*if you're using multiple clusters*)
- Avoid using Argo Rollouts for infra apps (*cert-manager, nginx, CoreDNS, sealed-secrets*)
- Metrics to tell if deployment is successful



**Can you tell if a deployment is successful or not within 15 minutes?**

**- WITHOUT a human involved**



# Decide what failed deployment means to you

- **Error rate:** more than 5% of requests have errors ⇒ Failed
- **Request rate:** requests rate falls under 100 rps ⇒ Failed
- **Response time:** 90% of requests complete in under 250ms ⇒ Failed
- Additional criteria:
  - more than 5% of requests have errors OR requests duration increased by more than 40% ⇒ Failed
  - number of errors does not increase by 10% OR requests rate falls under 20 rps ⇒ Failed
- Successful deployment criteria:
  - 98% of requests succeed AND all requests complete in under 100 ms ⇒ Success



# Supported Metric providers



DATADOG



new relic®



Prometheus



**WAVEFRONT**  
by vmware®



Amazon Cloudwatch

- Custom Web call
- Custom Job
- Custom plugin
- Apache SkyWalking



# Analysis example

```
apiVersion: argoproj.io/v1alpha1
kind: AnalysisTemplate
metadata:
  name: success-rate
spec:
  args:
  - name: service-name
  metrics:
  - name: success-rate
    interval: 2m
    count: 2
    # 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
    # Success mean
    # (number of requests that return 2xx HTTP status divided by all requests) returns over 95%
    successCondition: result[0] >= 0.95
  provider:
    prometheus:
      address: http://prom-release-prometheus-server.prom.svc.cluster.local:80
      query: sum(response_status{app="{{args.service-name}}",role="canary",status=~"2.*"})/
sum(response_status{app="{{args.service-name}}",role="canary"})
```



# **What to measure**



# USE/RED metrics

## USE METHOD

Utilization (% time that service was busy)

Saturation (queue length)

Errors (count)

## RED METHOD

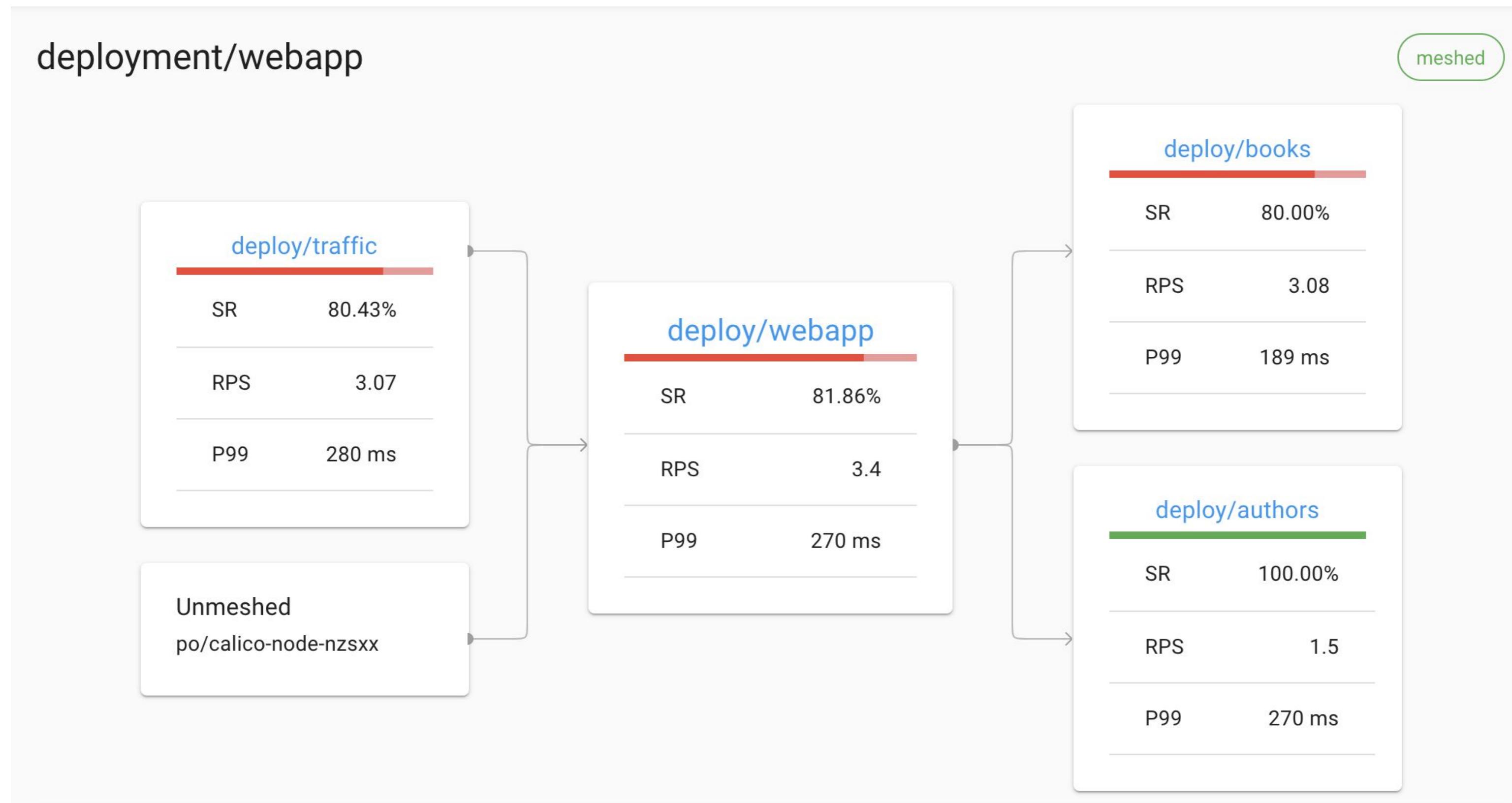
Rate (requests per second)

Errors (number of failed requests)

Duration (how much time requests take)



# RED metrics for free with a service mesh (e.g. Linkerd)



# For End-User Applications look at End User Metrics

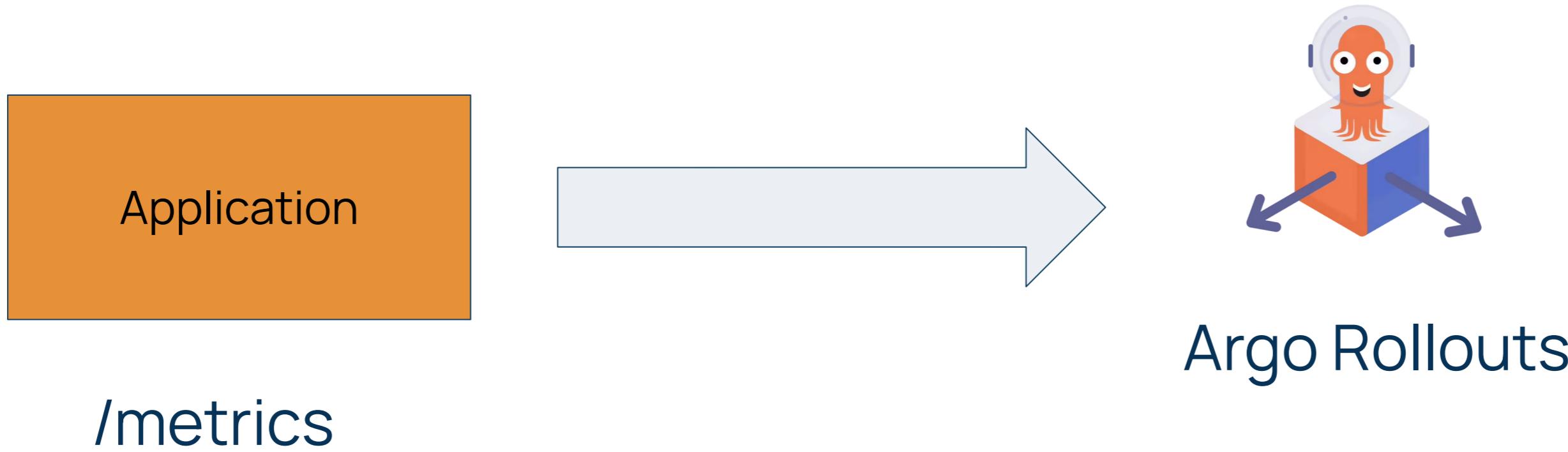
- Number of logins
- Number of items put in the basket
- Rate of payments that succeed
- Rejected payments
- Search queries
- Duration of user session



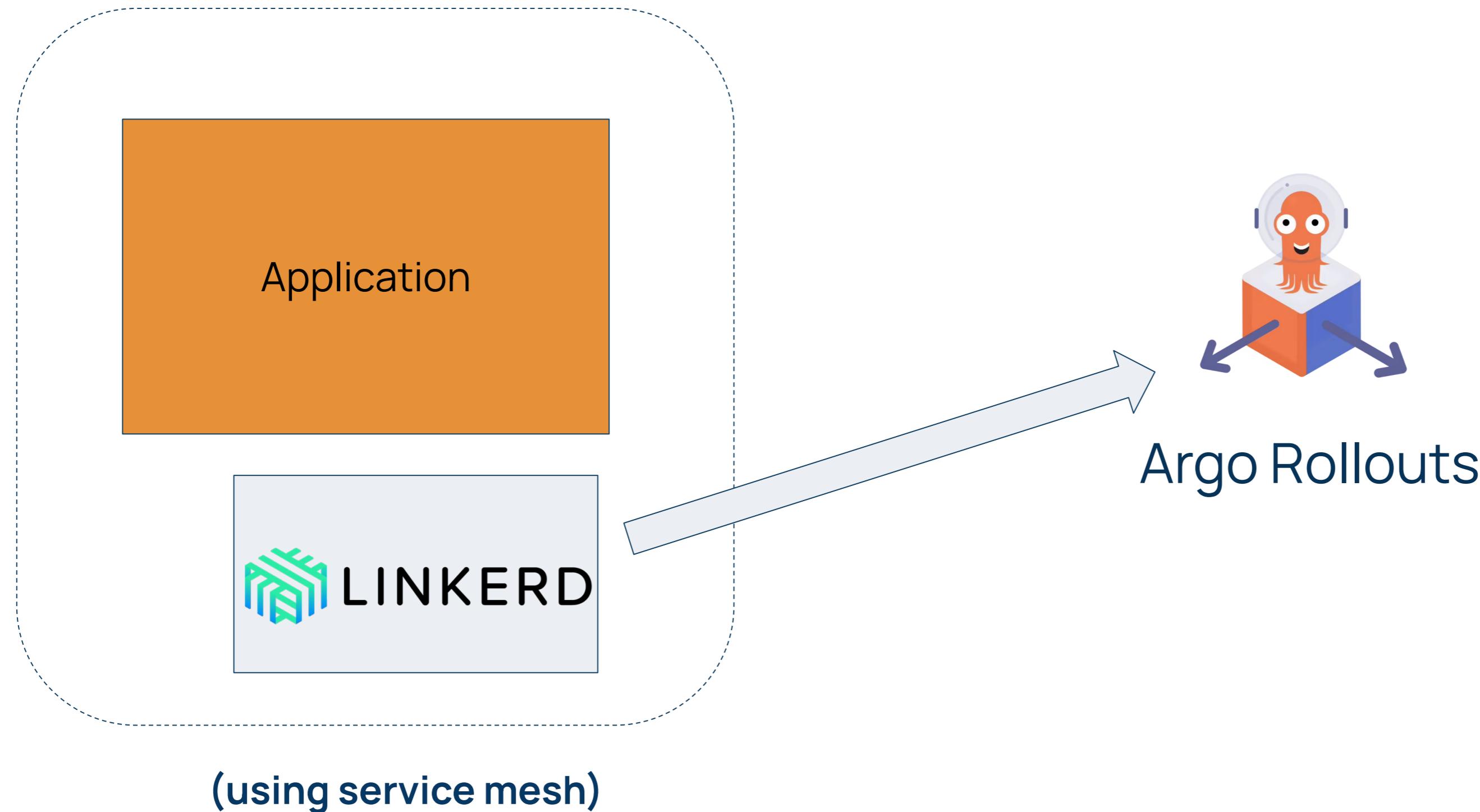
# Use cases



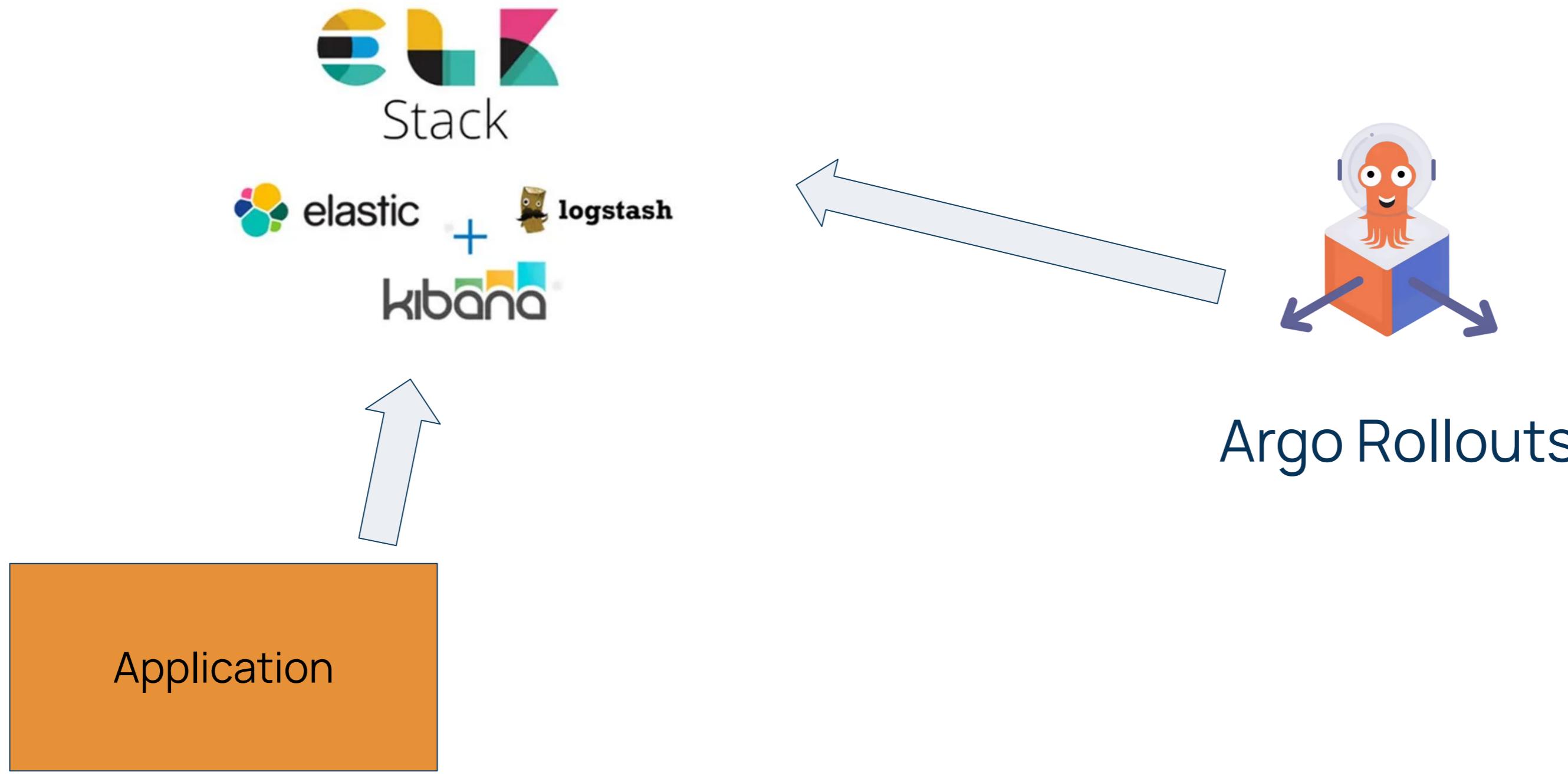
# Get ad-hoc metrics from the application itself



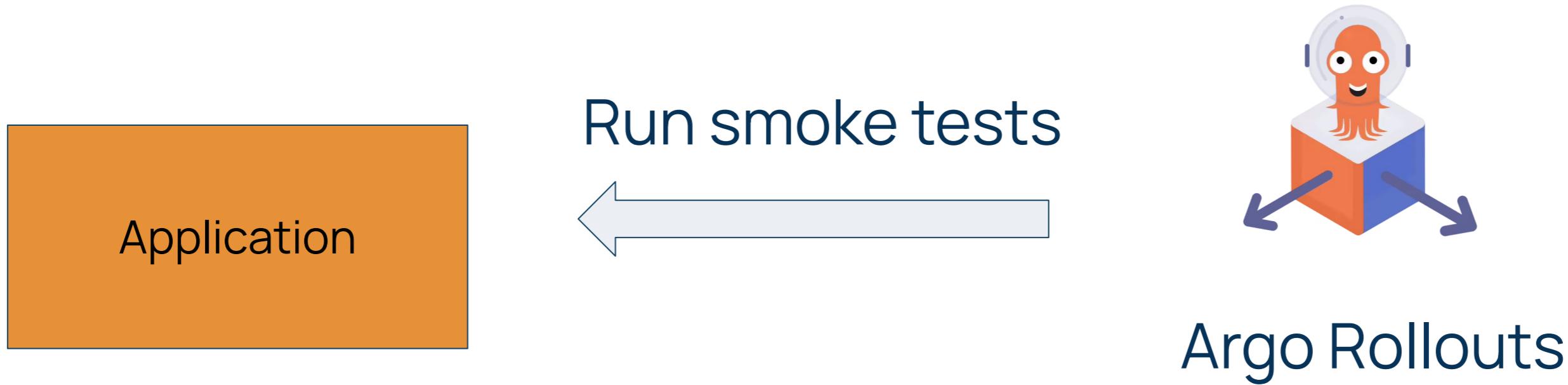
# Get metrics from an intermediate application



# Consult an external application for deployment status



# Make ad-hoc decision by a custom call or job



# Common pitfalls



# Common pitfalls

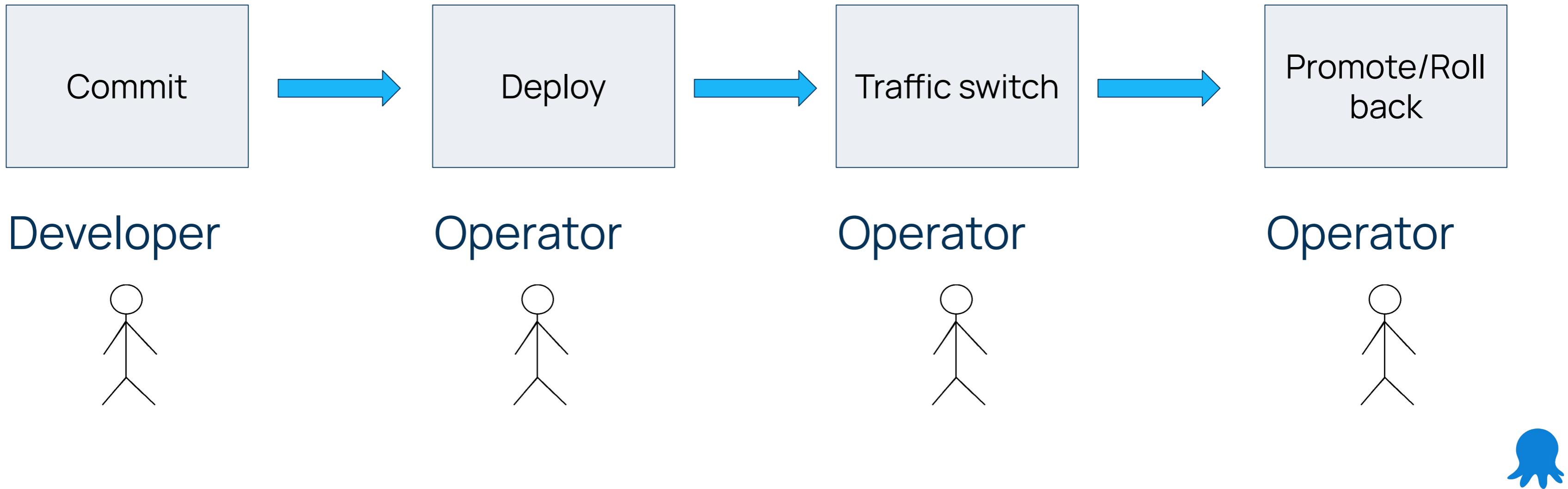
- Not having metrics
- Not having enough metrics
- Not having relevant Metrics
- Looking manually at metrics
- Not trusting metrics
- Not checking the requirements of Argo

## Rollouts

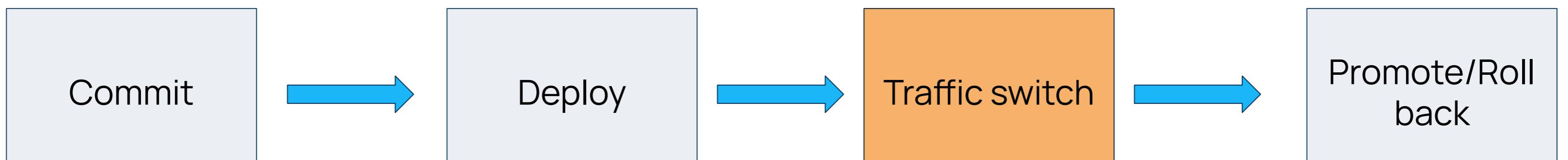
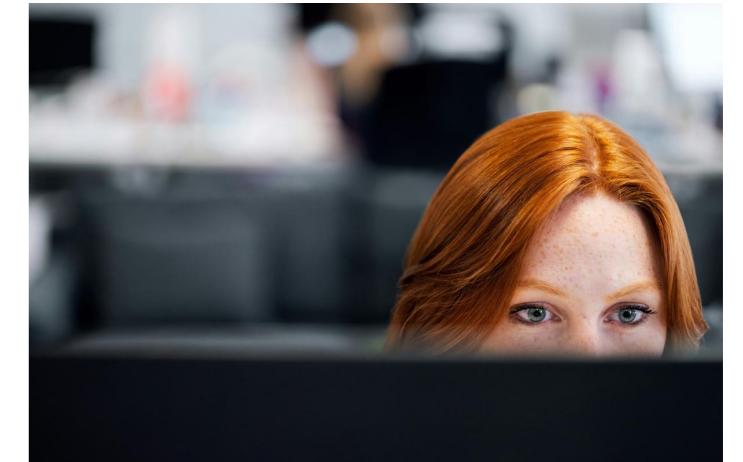
- Not automating the full process



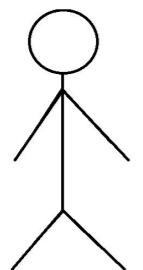
# Before Argo Rollouts



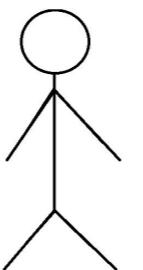
# Adopting Argo Rollouts partially



Developer



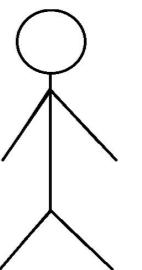
Operator



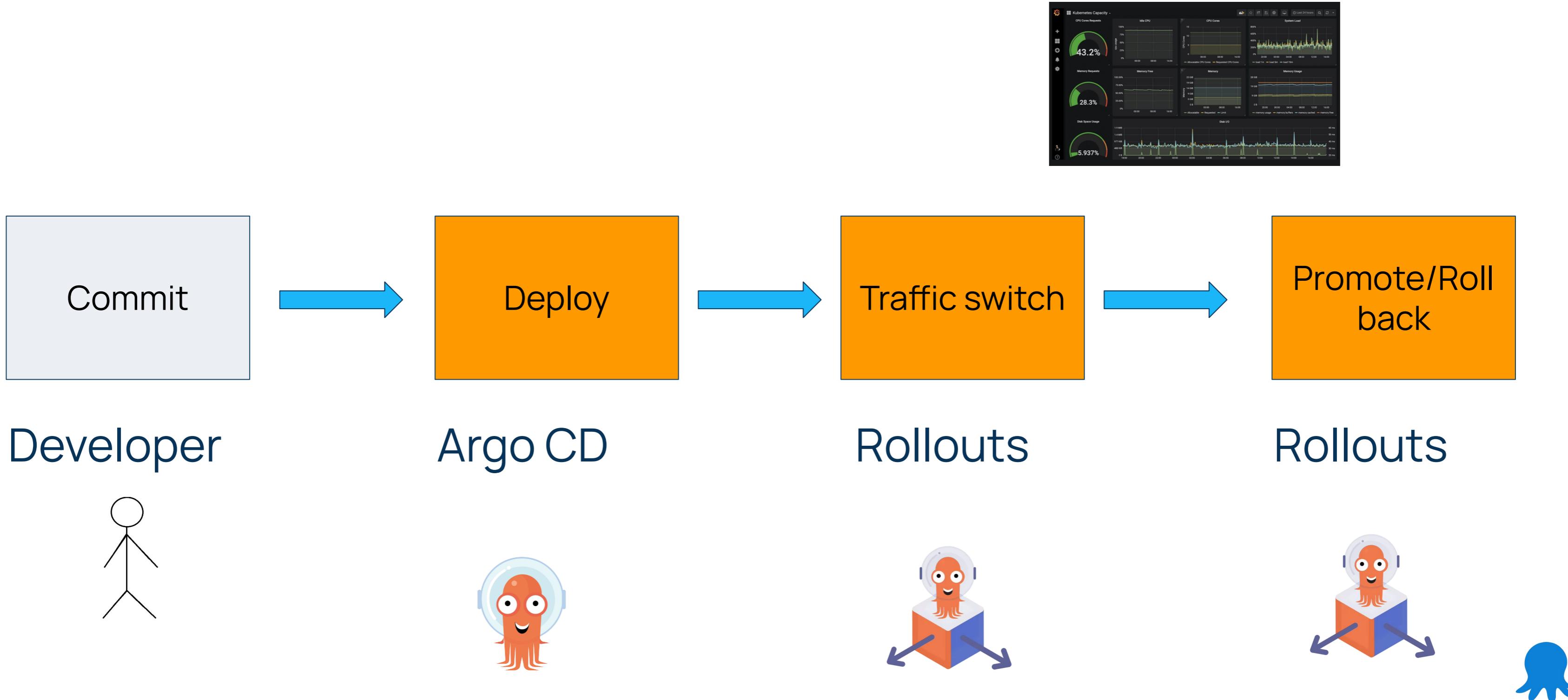
Rollouts



Operator



# The proper approach - automate everything

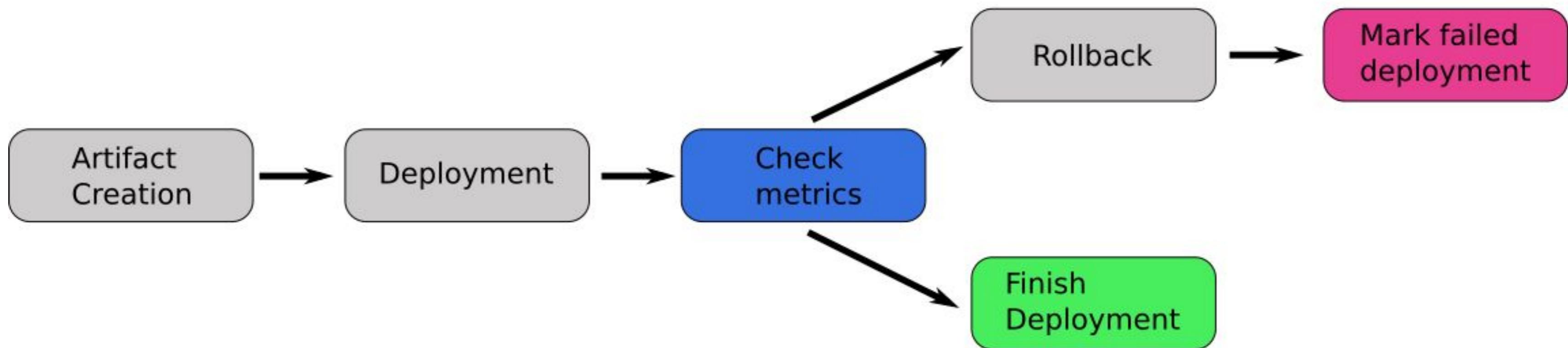


# Conclusion



# Our end goal

Fully Automated Rollbacks

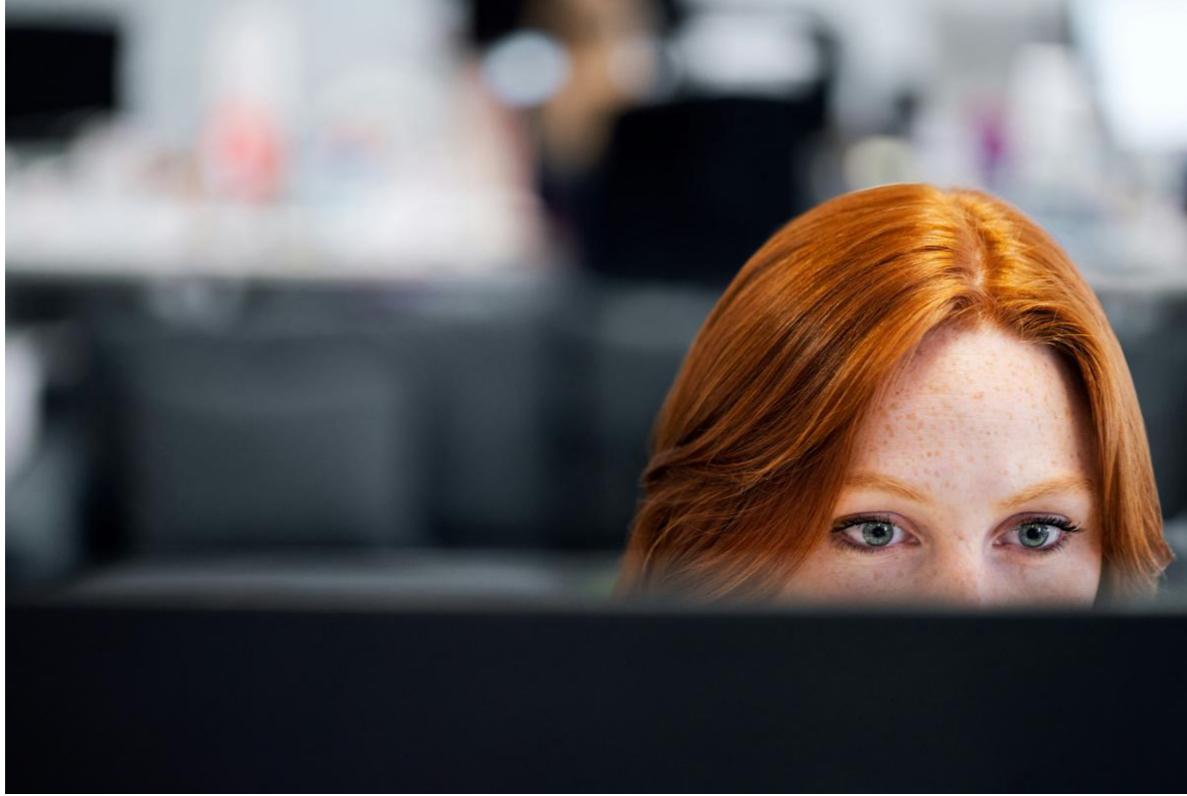


# What we have seen today

- Learn the Requirements of Progressive Delivery
- Have Metrics in your apps
- Employ **Relevant** Metrics
- Automate deployment/promotions
- Automate rollbacks
- Use Argo Rollouts for Kubernetes applications



# How production deployments should happen



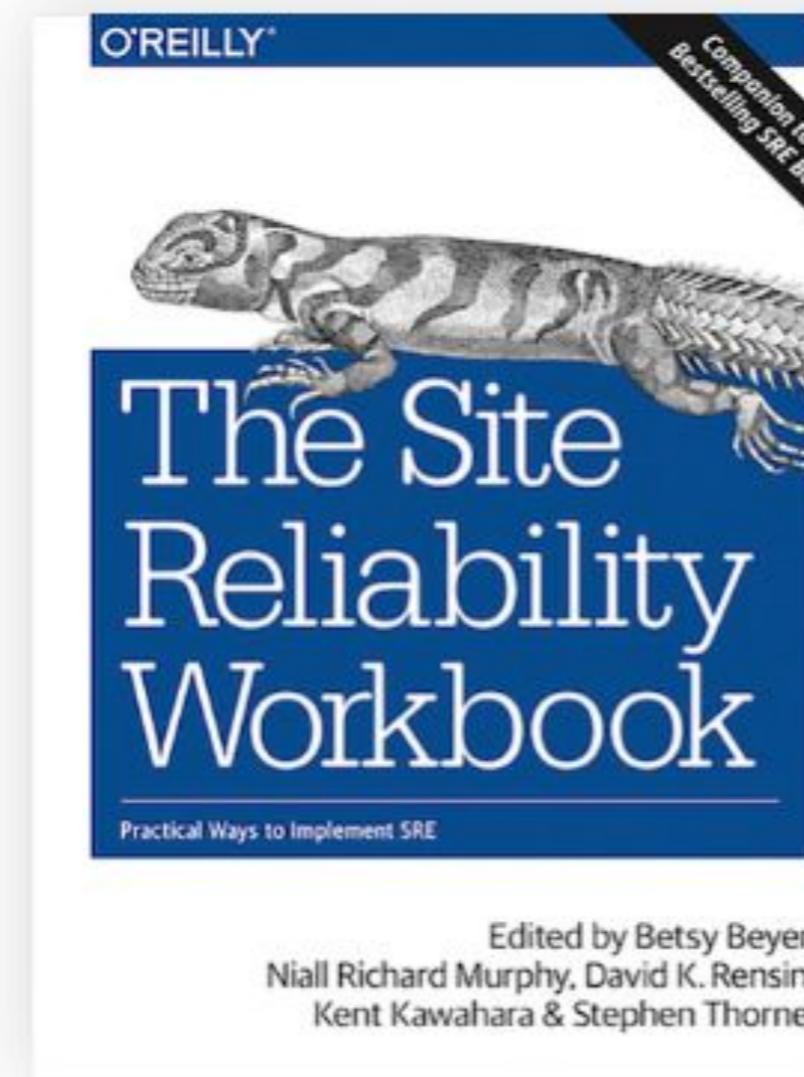
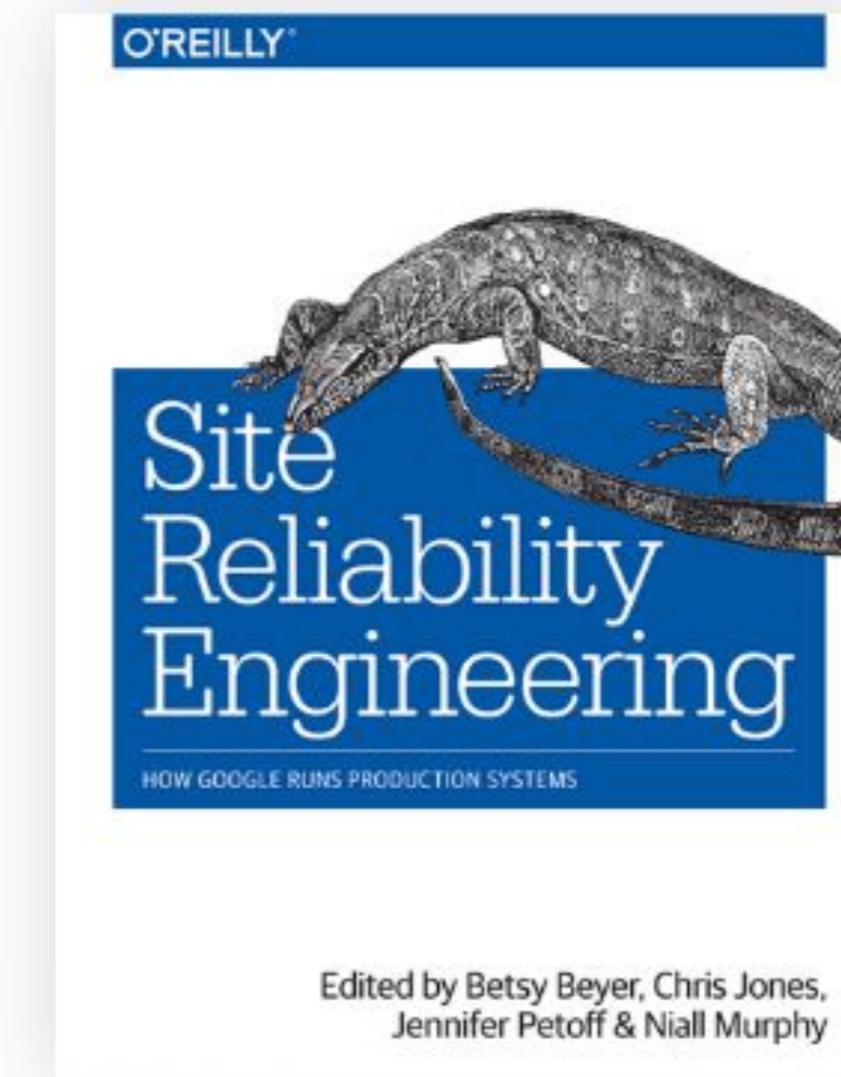
Deployment happens at 5.00 pm on Friday



5:15 the whole team is at the pub



# Books about monitoring and metrics

[Read online](#)[View details](#)[Read online](#)[View details](#)[Book updates](#)[Read online](#)

<https://sre.google/books/>



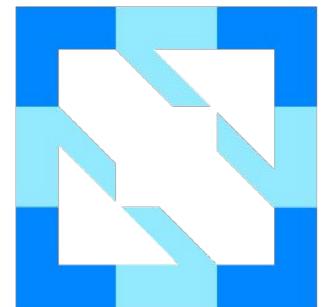
# Anastasiia Gubska

SRE/DevOps Engineer, BT Group



CNCF's First  
Deaf Ambassador  
Breaking barriers

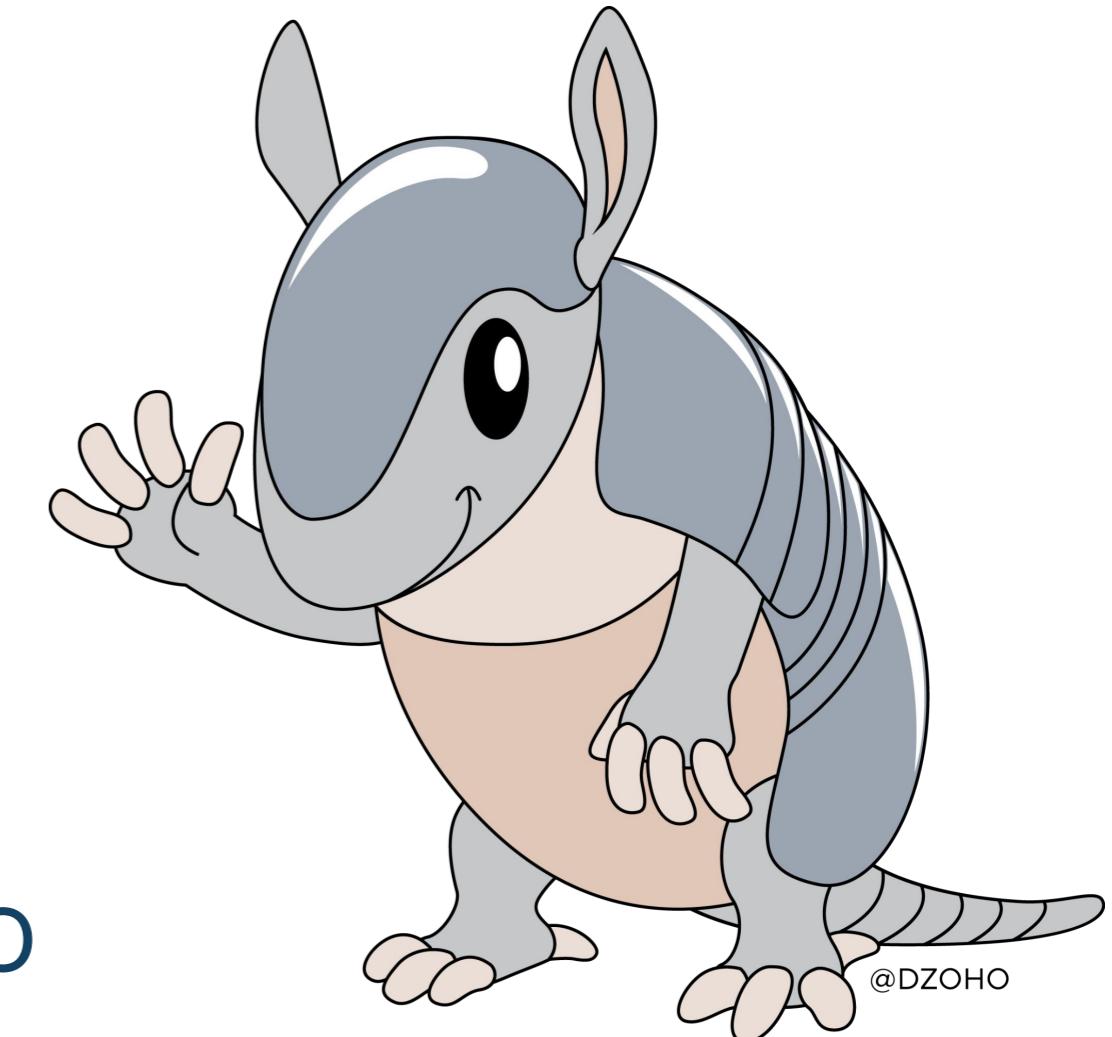




**TAG Contributor Strategy**  
**DEAF & HARD OF HEARING**  
W O R K I N G   G R O U P

Wondering what it is like to be **deaf in tech?**

Want to know what our community can do to  
**improve accessibility?**



**Come chat with us!**

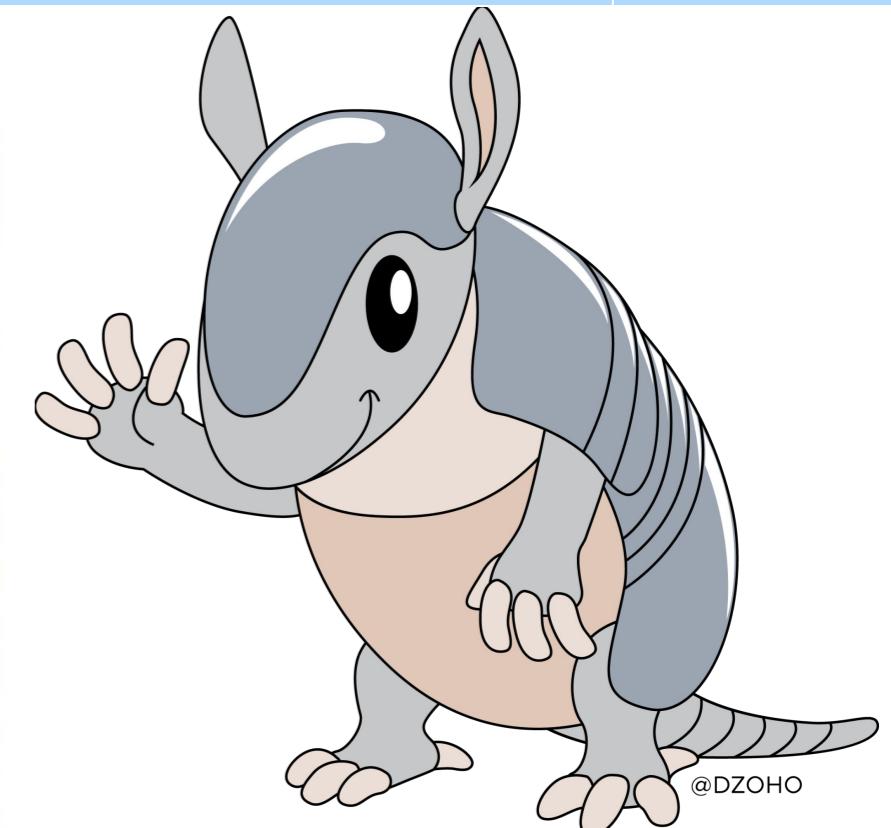
#deaf-and-hard-of-hearing (CNCF Slack)



# Our Team on Stage

# Community Activities

When?	Talk	Room #	When?	Talk	Room #
Tue, 3:46pm	Beyond the Checkbox: Humanizing Accessibility	Regency Ballroom B	Thu, 4 pm	Deaf and Hard of Hearing Advocacy Discussion	DEI Community Hub
Tue, 4:30 pm	Stop Deploying Blind! Using Observability and Argo Rollouts to Light the Way	ArgoCon	Thu, 5 pm	👉 Sign Language Crash Course	DEI Community Hub
Wed, 12:10 pm	AI and ML: Let's Talk About the Boring (yet Critical!) Operational Side	Level 2   255 B			
Wed, 3:25 pm	How to Get Started Contributing in the CNCF	Level 2   Ballroom C			
Thu, 3:25 pm	TLS and MTLS: Introduction to Modern Security	Level 2   251 AD			
Fri, 11:55am	Accessibility at KubeCon: Deaf Voices in Cloud Native	Level 1   Ballroom B			



# Thank you!

- <https://argoproj.github.io/rollouts/>
- <https://www.brendangregg.com/usemethod.html>
- <https://grafana.com/blog/2018/08/02/the-red-method-how-to-instrument-your-services/>
- <https://contribute.cncf.io/about/deaf-and-hard-of-hearing/>
- <https://sre.google/books/>

