KubeCon CloudNativeCon North America 2026

NORTH AMERICA 2026
SALT LAKE CITY

N Clusters, One kubectl

Multi-Cluster That Just Works

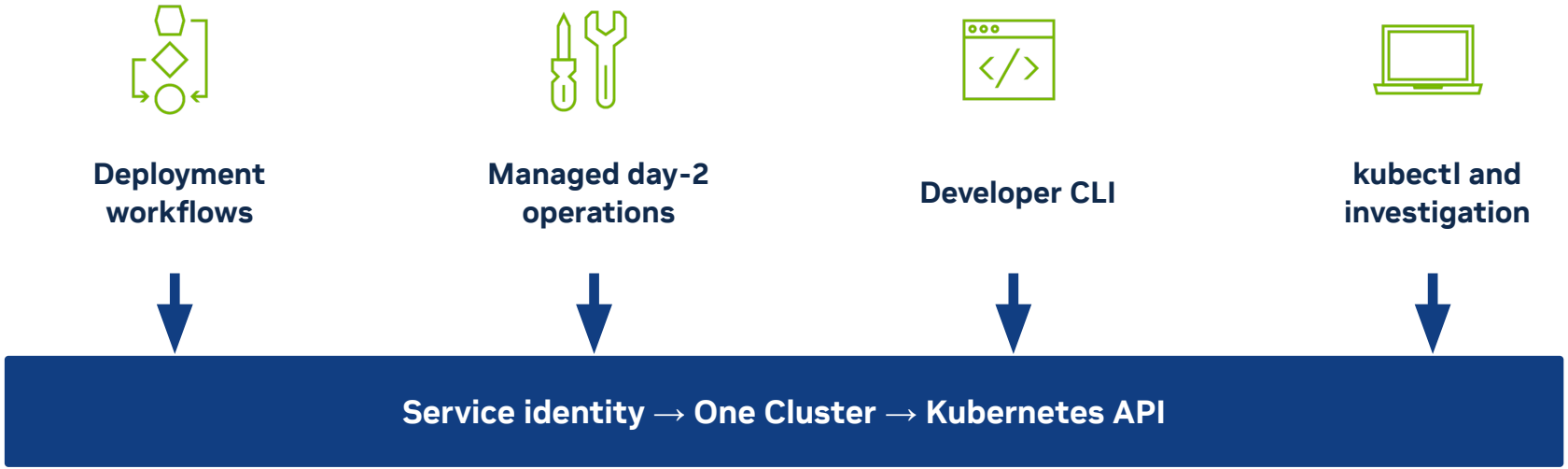
Anne Lau
Salesforce



Scott Brimhall
NVIDIA · built while at Salesforce



One Service. An Ecosystem Of Tools.

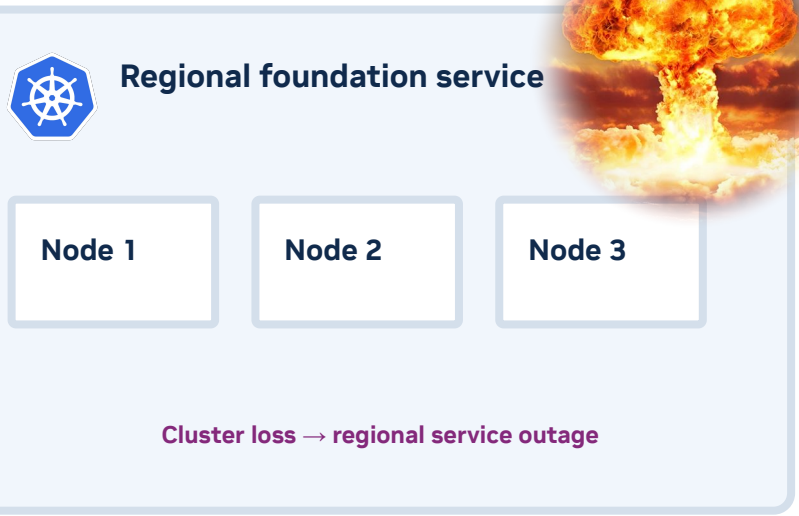


Node Resilience Ends At The Cluster Boundary

24+ hour outage

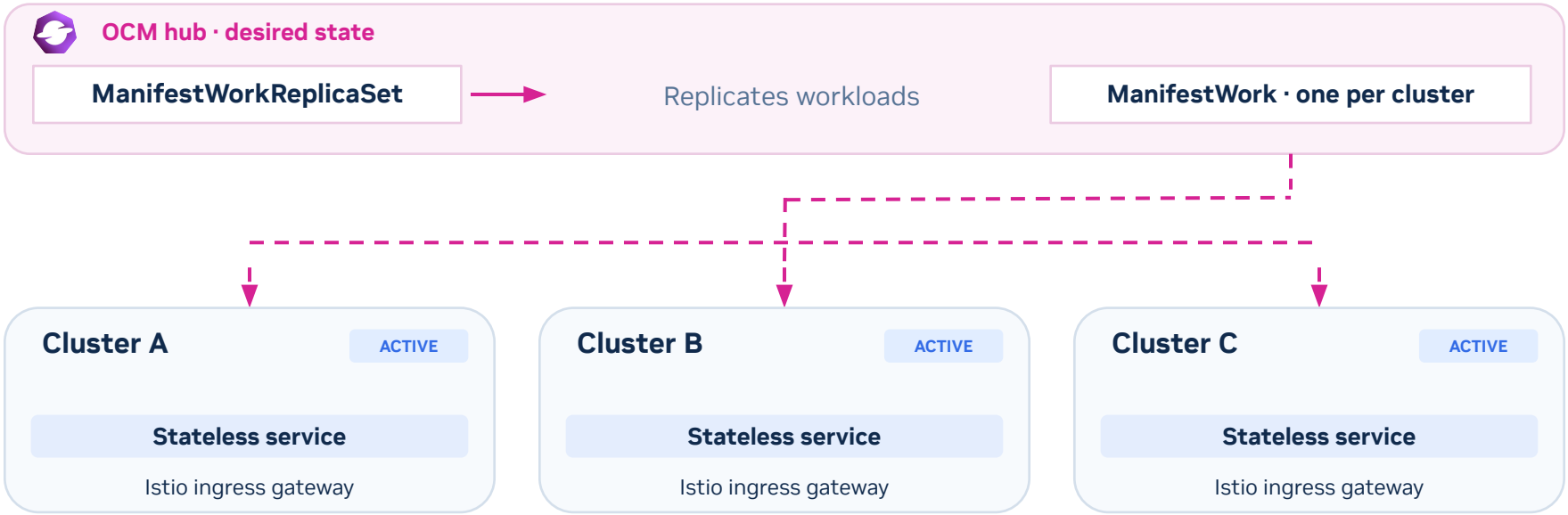
Large-cluster recovery
can outlast the business.

Business objective:
near-zero service RTO



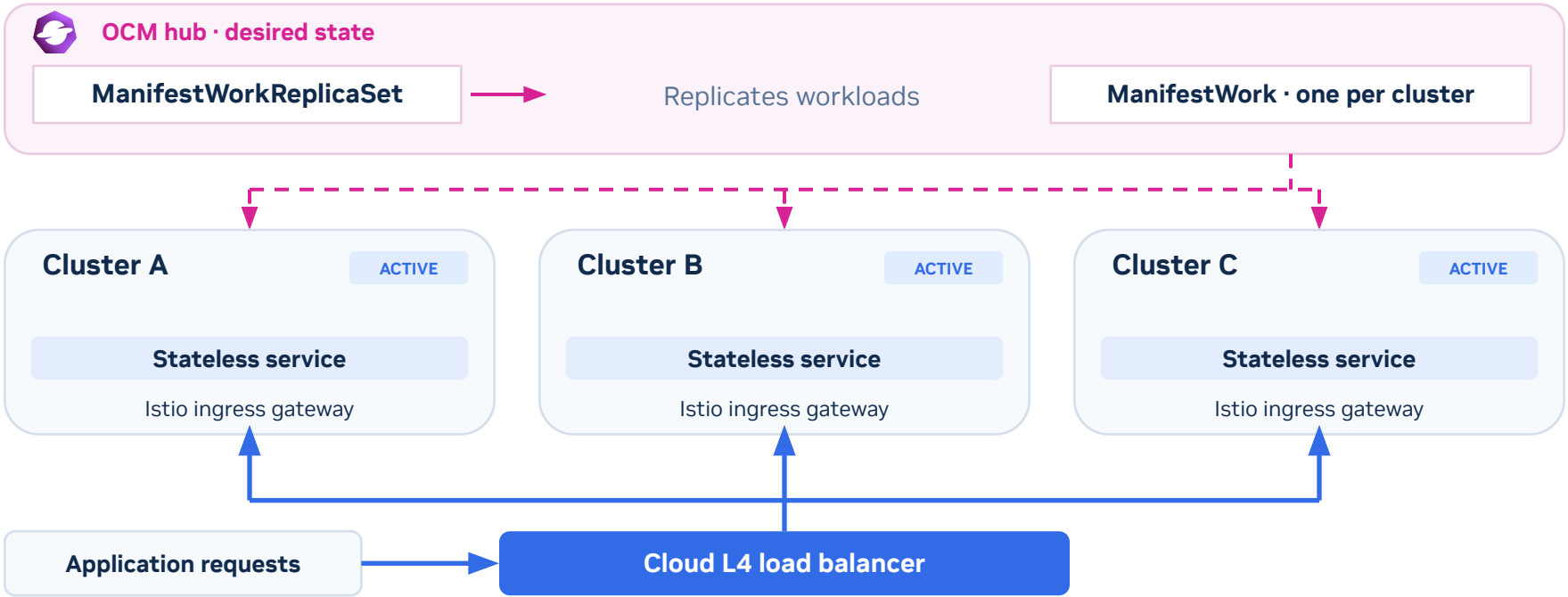
Cluster Resiliency with Open Cluster Management

The same service runs across clusters. Traffic continues when one fails.



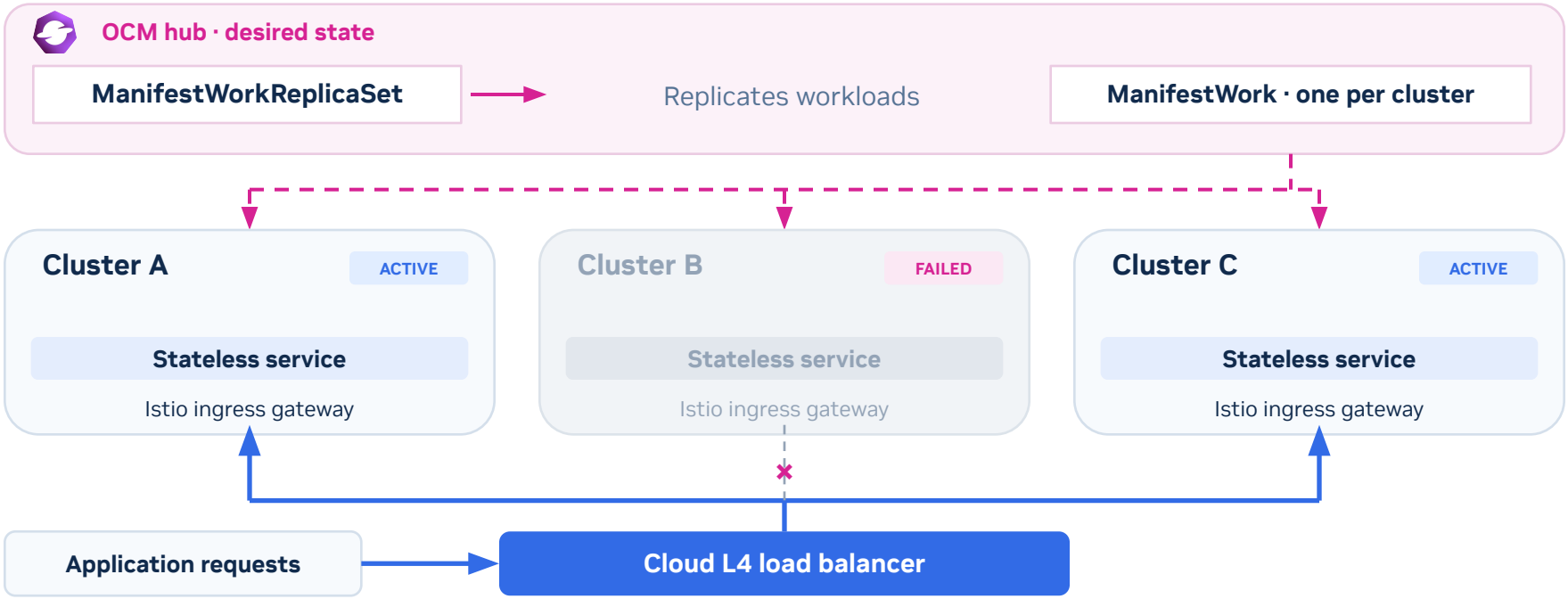
Cluster Resiliency with Open Cluster Management

The same service runs across clusters. Traffic continues when one fails.

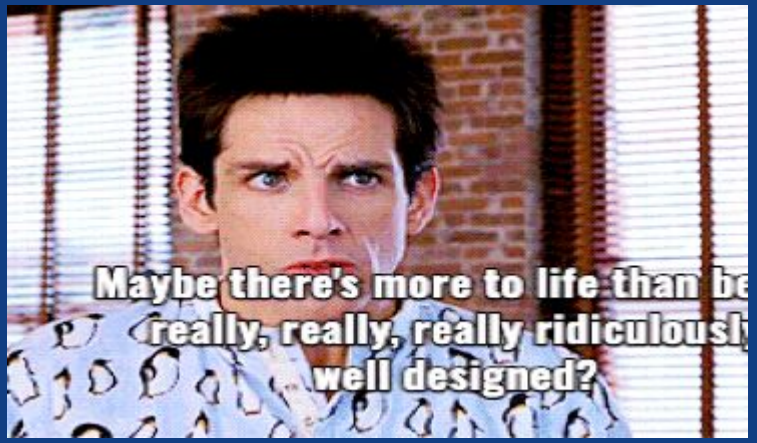


Cluster Resiliency with Open Cluster Management

The same service runs across clusters. Traffic continues when one fails.



Start with the developer experience.



Work backward to the technology.

Experience → API contract → architecture

Resiliency Improves. Familiar Workflows Break.

KUBECTL

```
$ kubectl get pods
No resources found in default namespace.

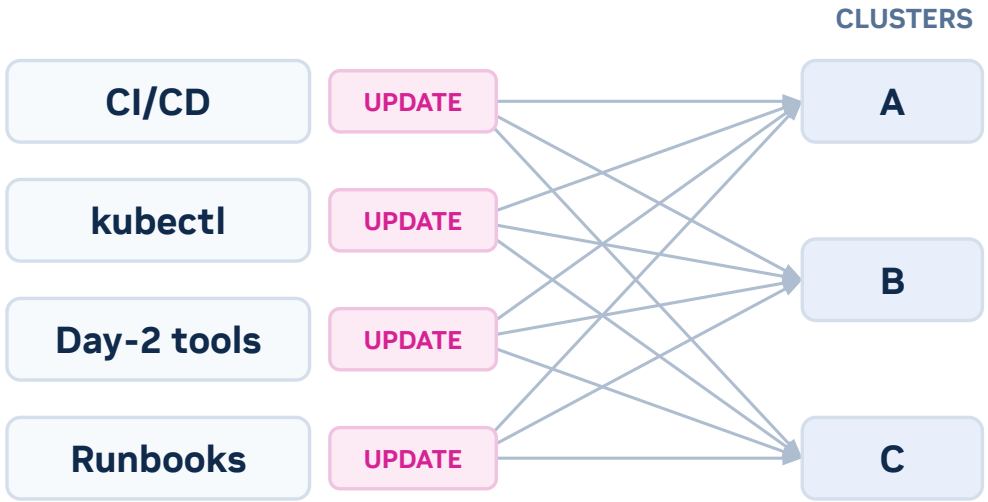
# Where are my resources? Which cluster?
```



Choose Where The Complexity Lives

OPTION 1

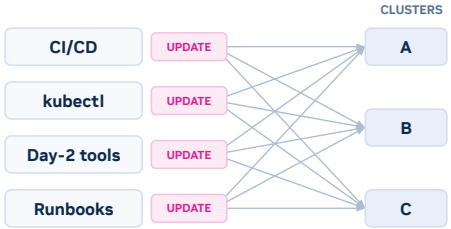
Change every client



Choose Where The Complexity Lives

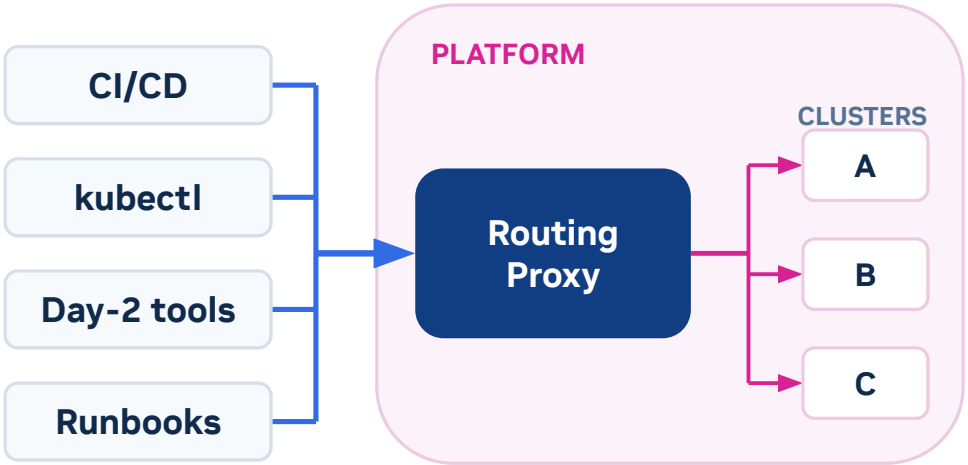
OPTION 1

Change every client



OUR CHOICE

Keep workflows familiar



Before: one service, one cluster

One service context resolves to one Kubernetes API.

CLIENT SETUP	Kubeconfig	One cluster context + existing credentials

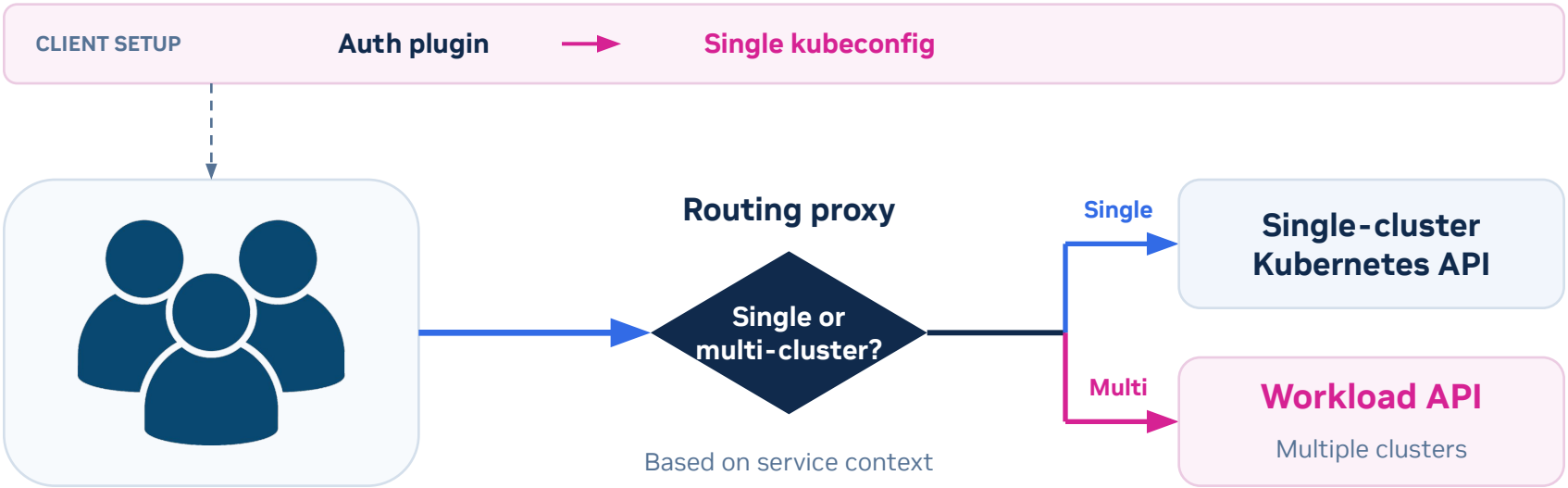


A Thin Routing API Proxy Layer



After: one endpoint, either path

The service context selects the single-cluster or Workload API path.



Workload API: A Smart Routing Proxy

One kubeconfig. One service context. No client-side cluster selection.

FAMILIAR WORKLOAD OPERATIONS

Service context: foo-bar

READ	GET · LIST · WATCH
WRITE	POST · PUT · PATCH · DELETE
CONNECT	logs · exec · port-forward

One service endpoint

Cluster routing stays behind the interface.

Workload API: A Smart Routing Proxy

One kubeconfig. One service context. No client-side cluster selection.

FAMILIAR WORKLOAD OPERATIONS

Service context: foo-bar

READ	GET · LIST · WATCH
WRITE	POST · PUT · PATCH · DELETE
CONNECT	logs · exec · port-forward

One service endpoint

Cluster routing stays behind the interface.

EXISTING CREDENTIALS, PASSED THROUGH

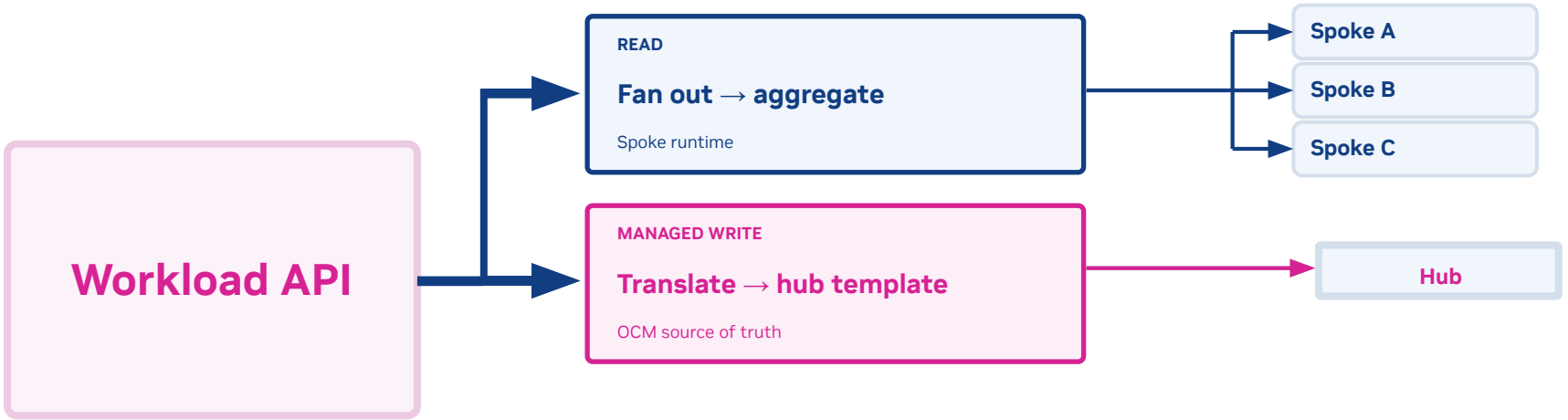
Caller → proxy → Workload API → hub / managed clusters

The proxy forwards existing user credentials; it does not generate them.

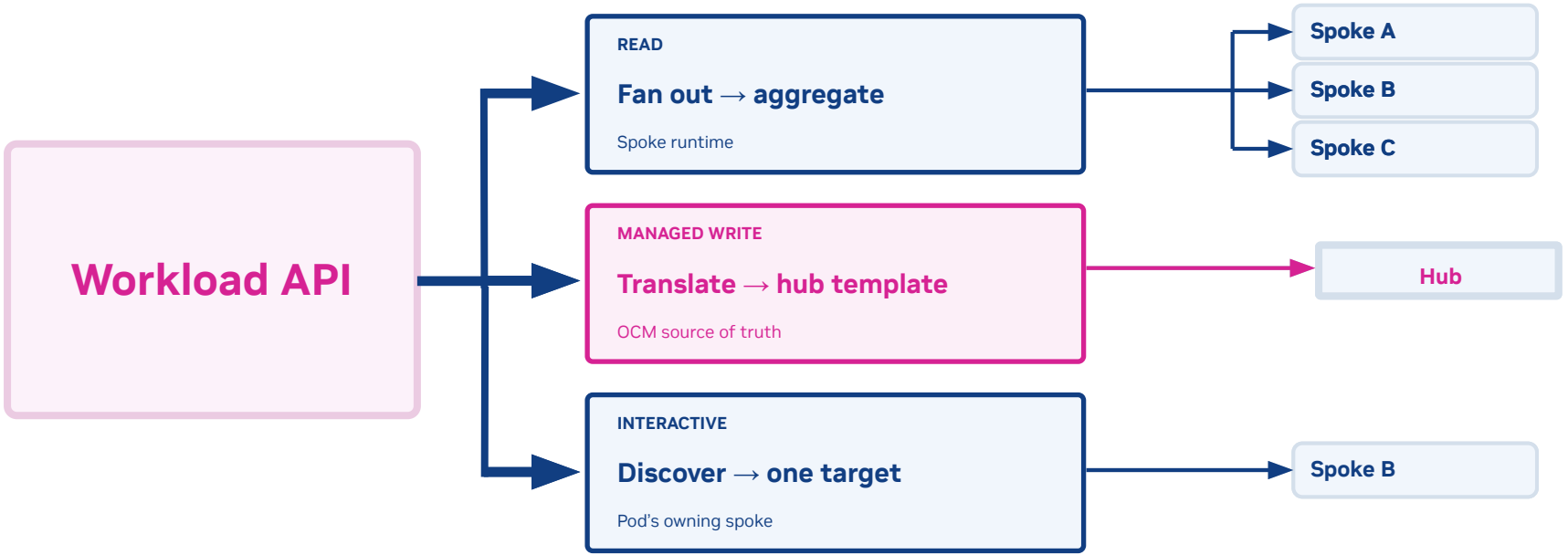
Workload API: Three Request Paths



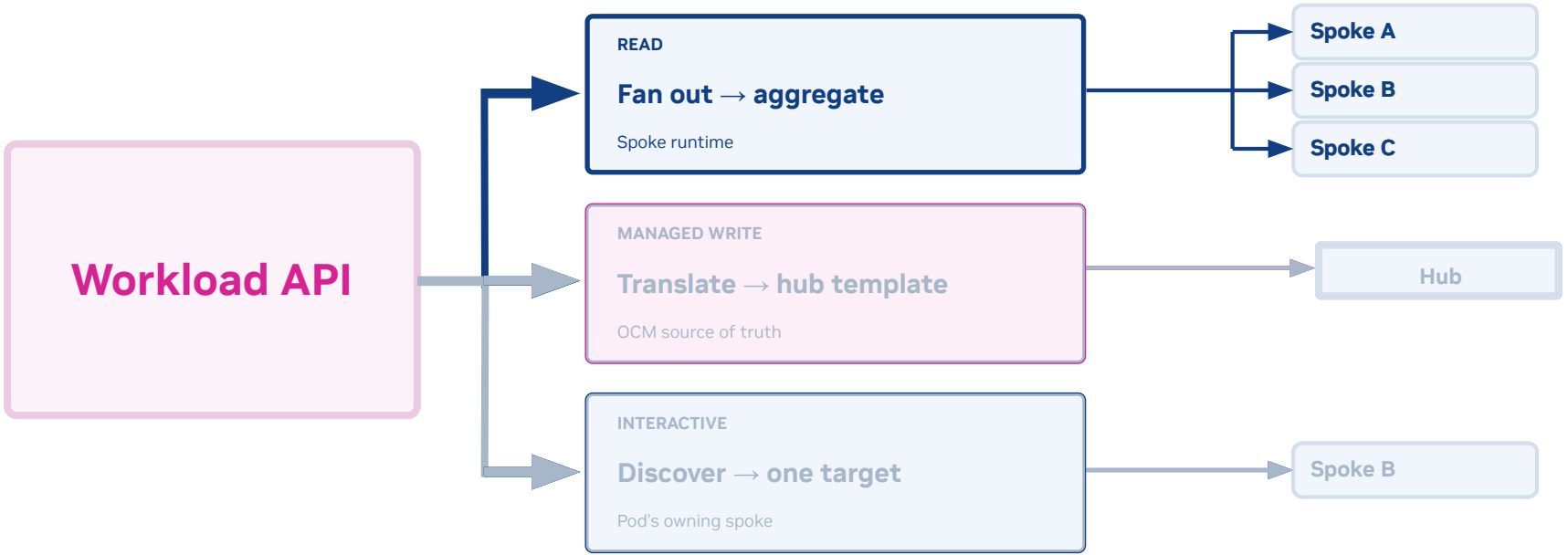
Workload API: Three Request Paths



Workload API: Three Request Paths



Reads inspect every selected spoke



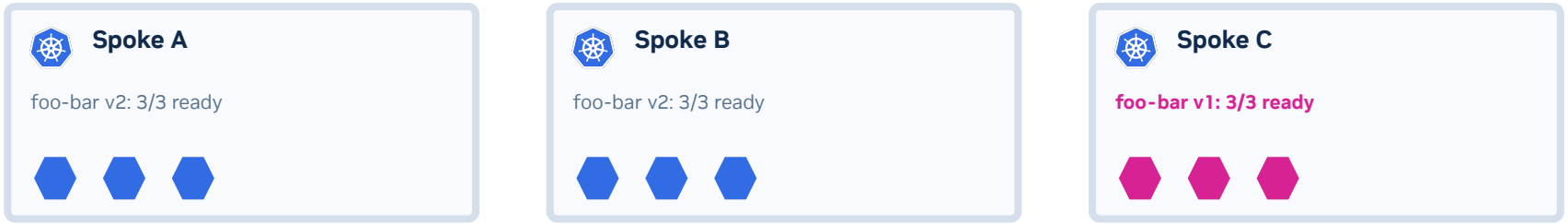
LIST: Nine pods, with their origins intact

KUBECTL

```
$ kubectl get pods -L multi-cluster.k8s.io/origin
NAME         READY   STATUS    ORIGIN
foo-bar-a1    1/1    Running   spoke-a
foo-bar-a2    1/1    Running   spoke-a
foo-bar-a3    1/1    Running   spoke-a
foo-bar-b1    0/1    Error     spoke-b
foo-bar-b2    1/1    Running   spoke-b
foo-bar-b3    1/1    Running   spoke-b
foo-bar-c1    1/1    Running   spoke-c
foo-bar-c2    1/1    Running   spoke-c
foo-bar-c3    1/1    Running   spoke-c
```

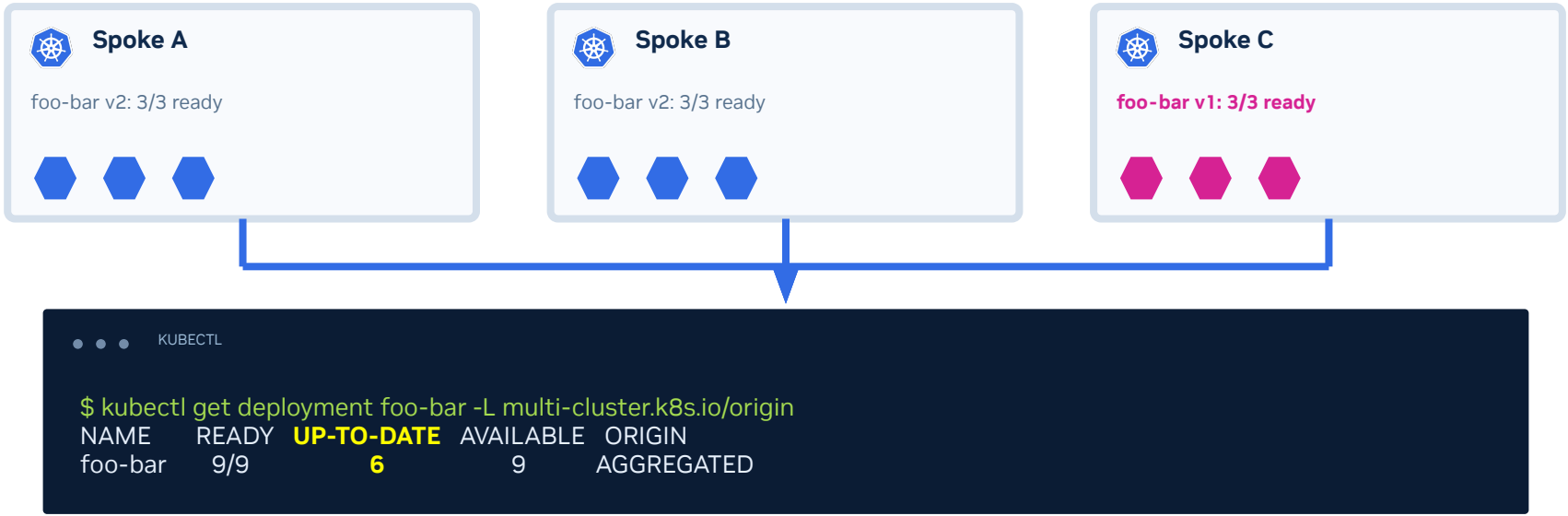
GET: Healthy totals can hide an unfinished rollout

All nine replicas are ready. Spoke C still runs the previous revision.



GET: Healthy totals can hide an unfinished rollout

All nine replicas are ready. Spoke C still runs the previous revision.



GET: Additional conditions for multi-cluster

KUBECTL

```
$ kubectl get deployment foo-bar -oyaml
apiVersion: apps/v1
kind: Deployment
```

```
status:
  conditions:
  - type: ImageInSync
    status: "True"
  - type: GenerationInSync
    status: "False"
    reason: GenerationMismatch
    message: >-
      status.observedGeneration:
      spoke-a=12, spoke-b=12, spoke-c=11
```

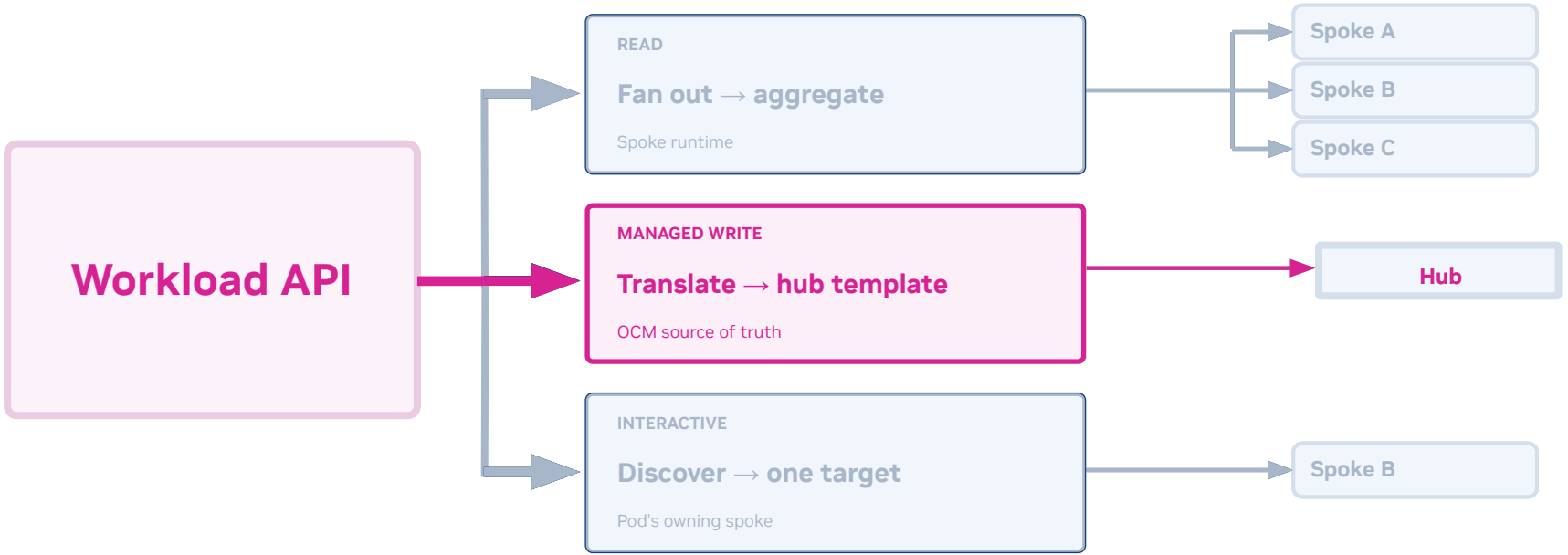
ImageInSync = True

Spoke image specs agree.

GenerationInSync = False

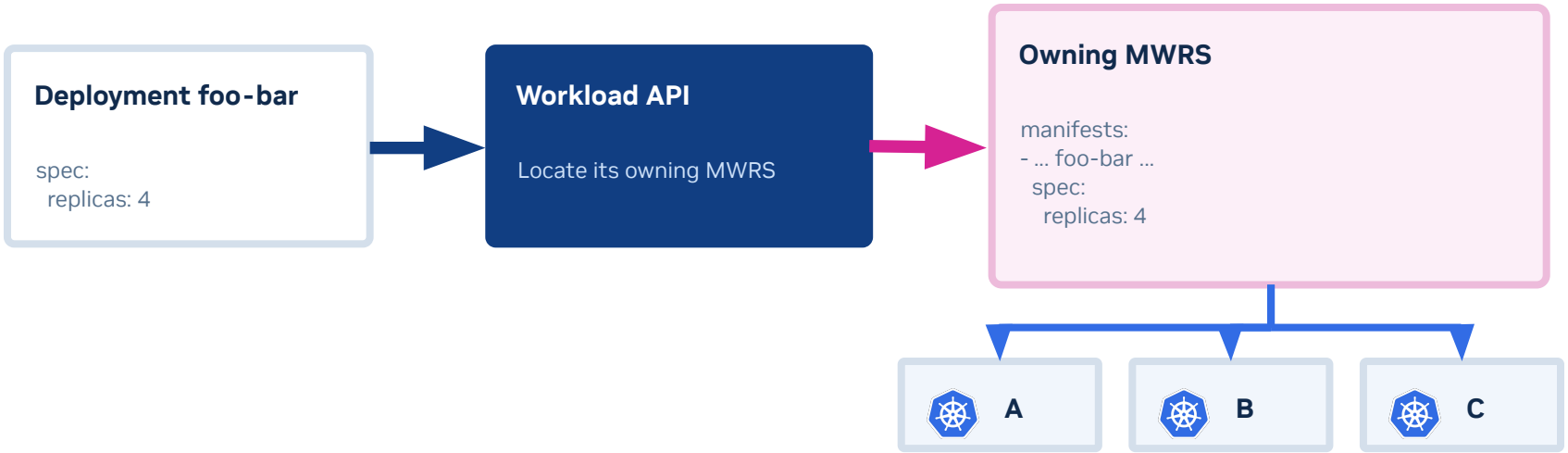
Spoke C has not caught up.

Managed writes update desired state on the hub



One Deployment edit, one owning manifest

The user edits foo-bar. The platform updates its desired state on the hub.



Schematic manifest excerpt; surrounding fields omitted

Two edits, one parent resourceVersion

CONCURRENT MANAGED EDITS

Edit A: foo-bar replicas / Edit B: foo-bar ConfigMap change in the same MWRS



1 **BOTH READ**

A and B read
MWRS version 42

2 **A WRITES**

A updates foo-bar
Parent becomes 43

3 **B IS STALE**

B submits its array
with version 42

4 **CONFLICT**

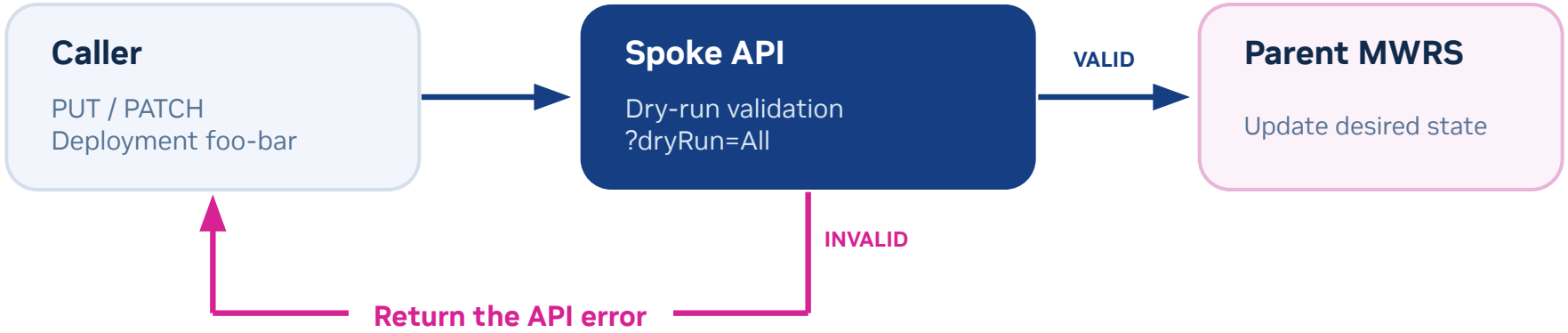
Hub rejects B
A stays intact

5 **REFRESH**

B re-reads before
resubmitting

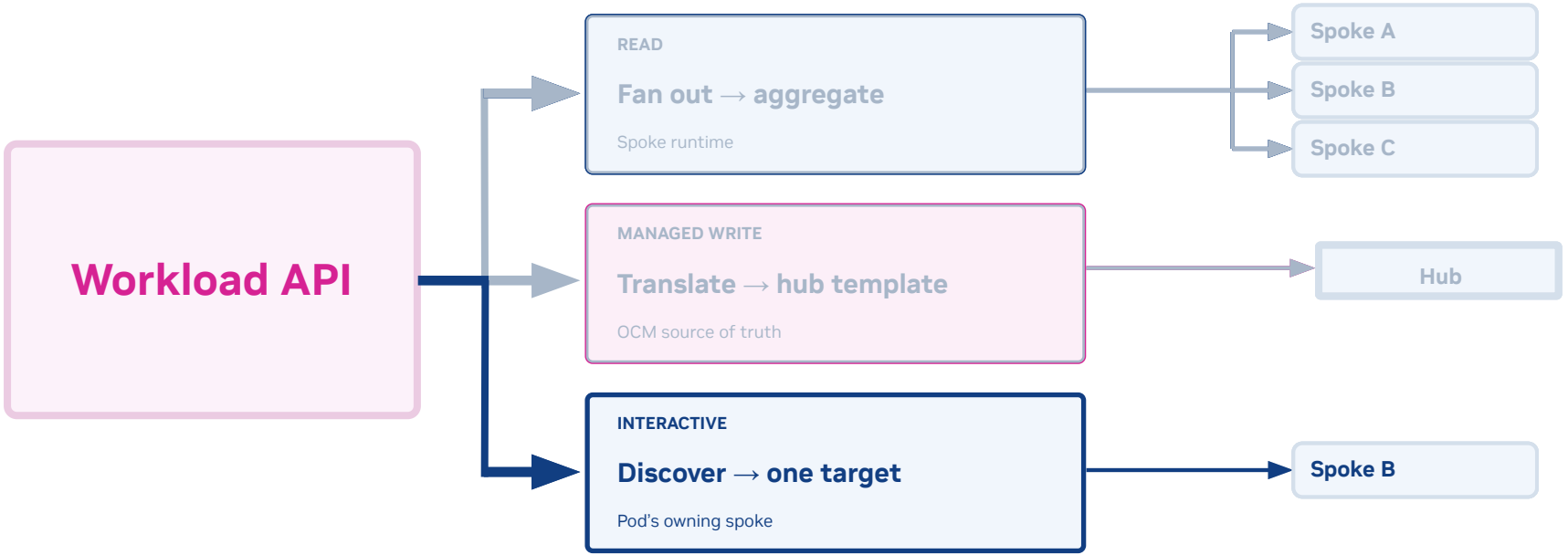
PUT / PATCH: errors reach the caller

Validate changes to an existing Deployment before updating its parent MWRS.



The caller gets the error.

Interactive requests target the pod's owning spoke



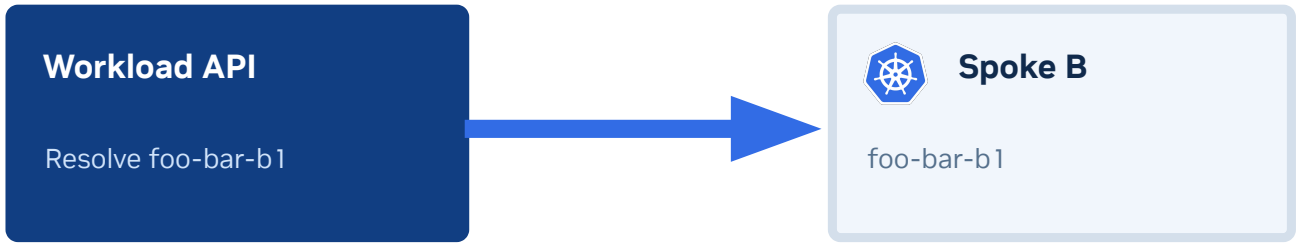
Interactive: Discover the pod, route to its spoke.

The pod origin from the list identifies one concrete destination.

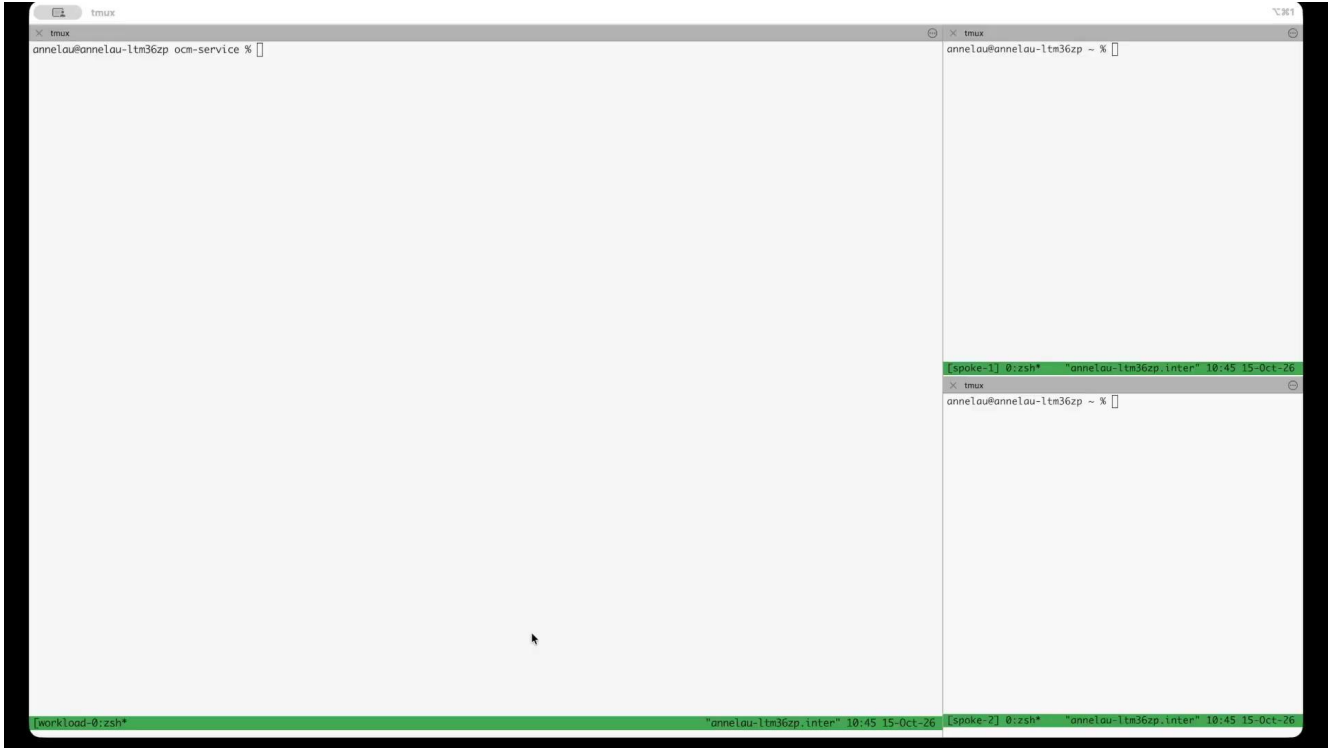
KUBECTL

```
$ kubectl logs -f foo-bar-b1

... log events continue on the same connection ...
```



One kubectl. Watch It Work.



Your developers are your customers.

Start with their experience.

Do cool stuff for them.

End with their experience.



KubeCon CloudNativeCon North America 2026

Questions?

Reach out on Kubernetes Slack!

Anne Lau

Salesforce



Scott Brimhall

NVIDIA · built while at Salesforce



Meet the Experts

KubeCon CloudNativeCon North America 2026

Meet NVIDIA developers and maintainers in informal Q&A sessions. Get introduced, share ideas, and ask questions about Kubernetes and AI infrastructure.

November 10–12
NVIDIA booth #170



Scan for the schedule

NVIDIA