



# Delivering Progressive Delivery With Service Mesh

Andrew Jenkins | @notthatjenkins

Zach Jory | @zjory

# Speakers

---



Andrew Jenkins is the CTO at Aspen Mesh, where he's building out an enterprise service mesh so organizations can leverage microservices for secure, manageable and rapid innovation.



Zach Jory is the Head of Marketing at Aspen Mesh. He thinks the way cloud-native infrastructure is changing the way the world operates is a generational shift.

# PROGRESSIVE DELIVERY

# Progressive delivery makes it viable to manage feature delivery in modern applications

**Kubernetes** - Declare what you want the world to look like, k8s makes it so



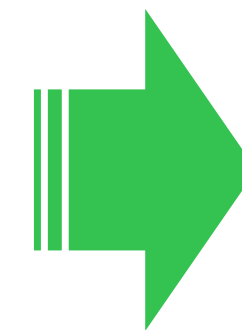
**GitOps** - Treat your world-declarations like code: commit, review, merge, iterate



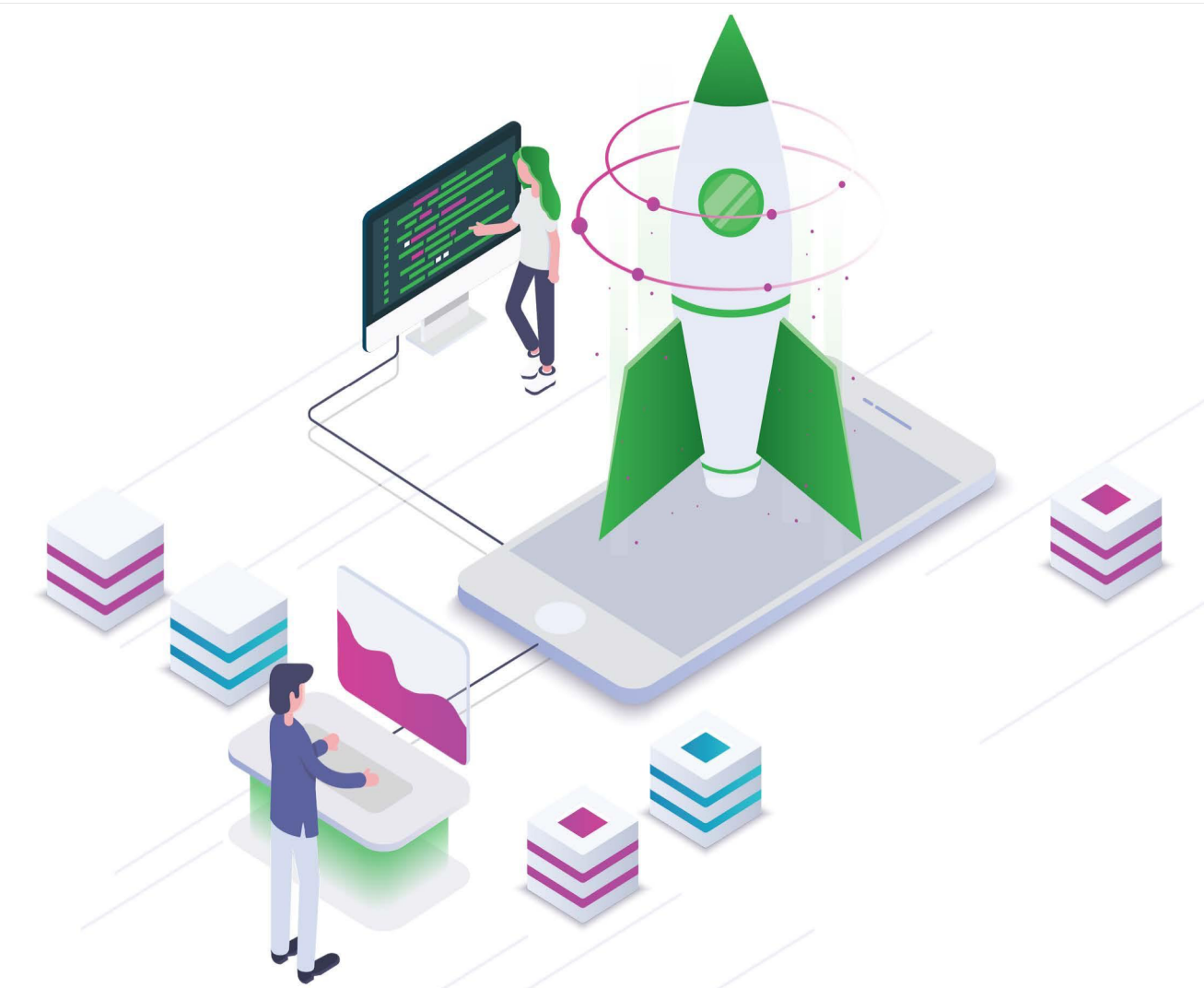
**Service mesh** - Advanced traffic routing (user class, traffic splitting, mirroring)



**Feature flags** - Activate new application behavior without changing code



**Progressive delivery** - Software deployment does not equal user activation



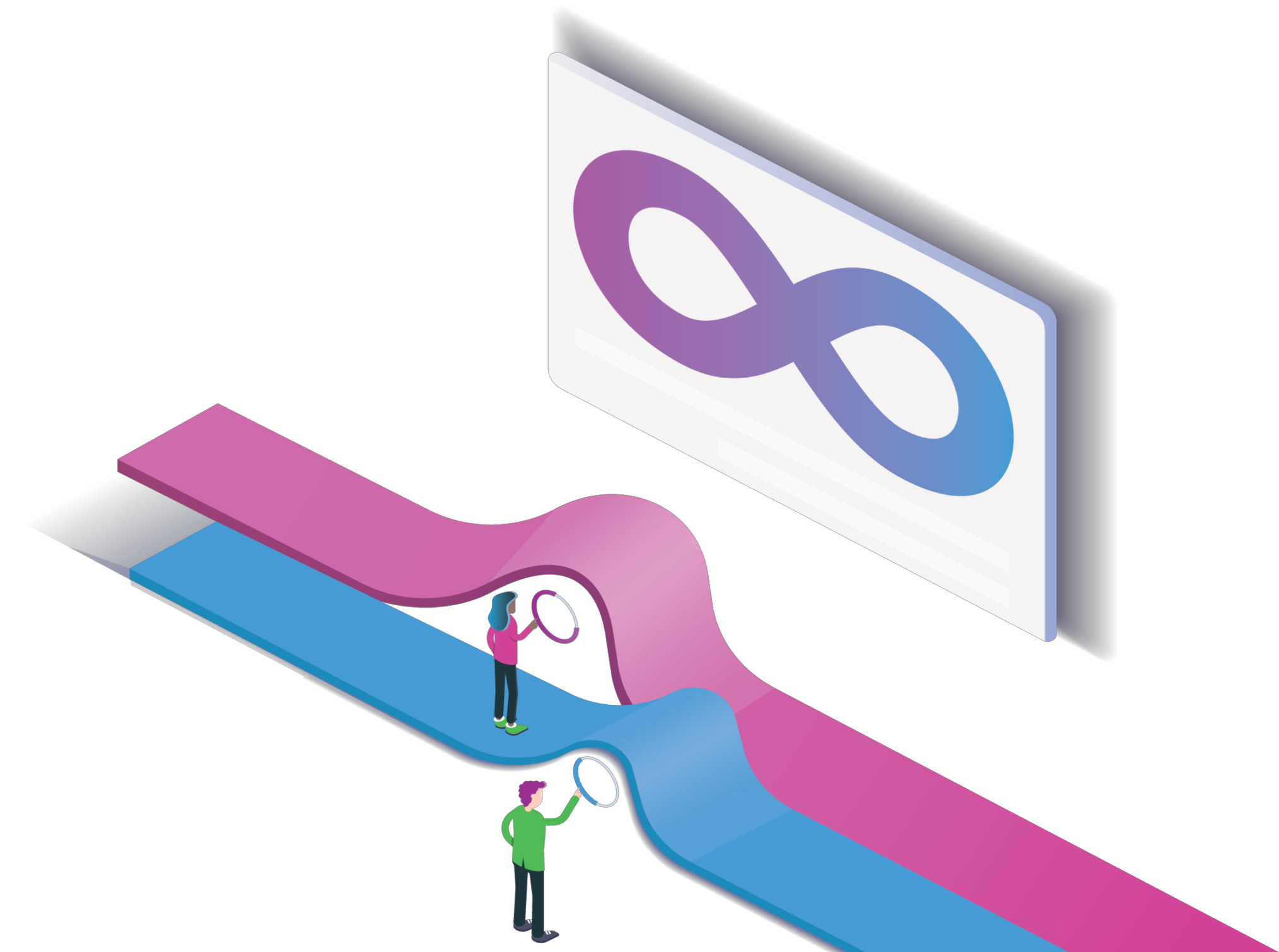


# Progressive delivery builds on the success of CI

---

**CI** provides a way to distribute risk, but doesn't actually limit the risk.

**Progressive delivery** limits the blast radius when rolling out new software.



# Progressive delivery = controlled empowerment

---

Progressive delivery allows the business to say when it's acceptable for new code to hit the customer.

With progressive delivery:

- Deployment is not the same as release
- Service activation is not the same as deployment
- Developers can deploy a service, the service can ship, but that doesn't mean it's activated for all users



# Progressive delivery means deployment is not the same as release

Deployments are rapid and iterative but decoupled from application changes.

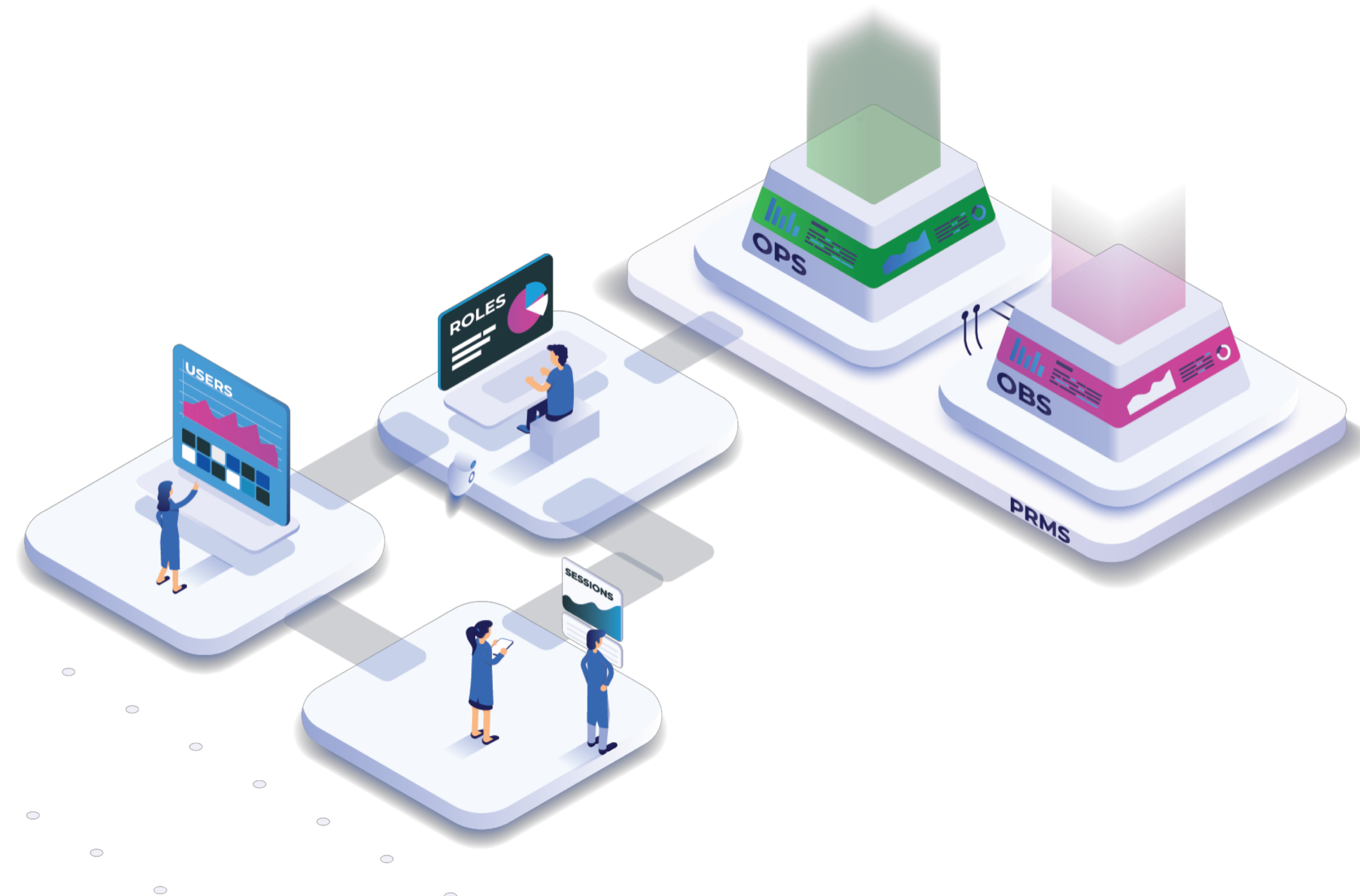
Can deploy multiple times without being forced to roll out new features to customers.



# Service mesh enables progressive delivery

Service mesh makes it possible to do progressive experimentation when rolling out new services through:

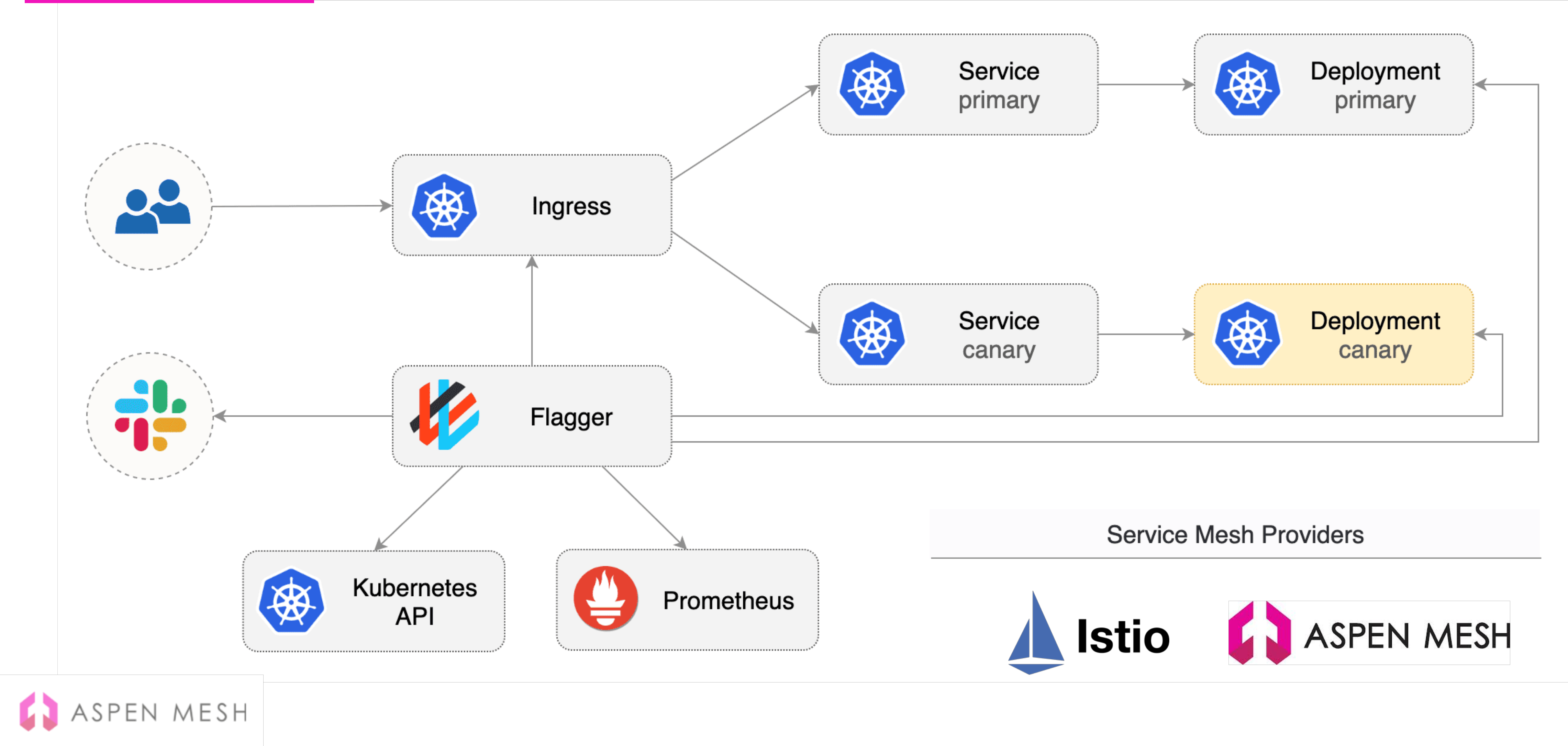
- Feature flags
- Rolling deploys
- **Canary**
- **Traffic mirroring**

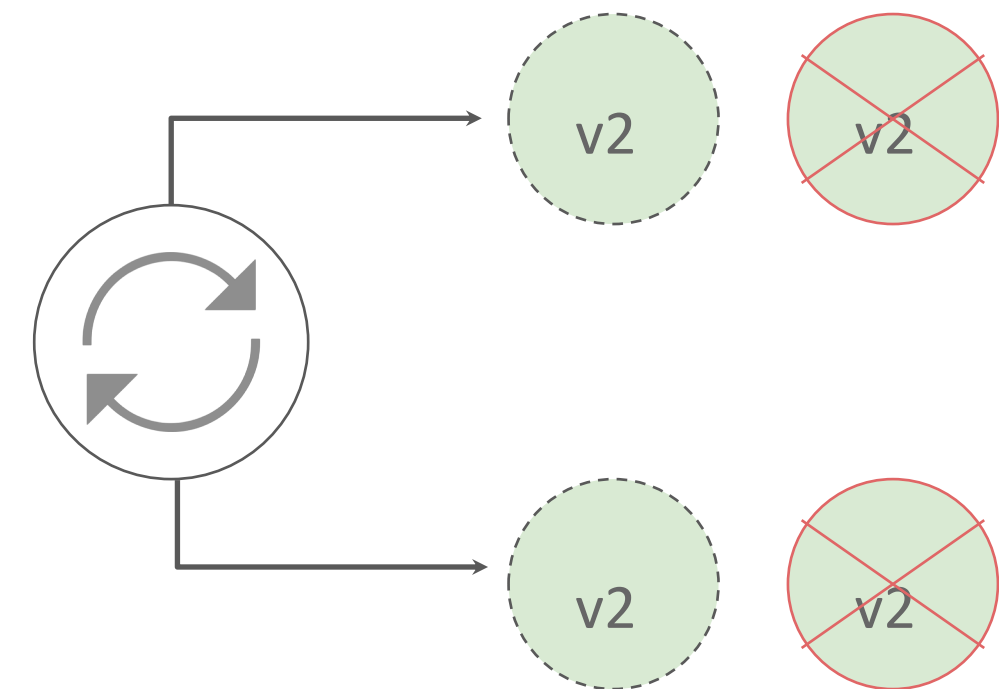
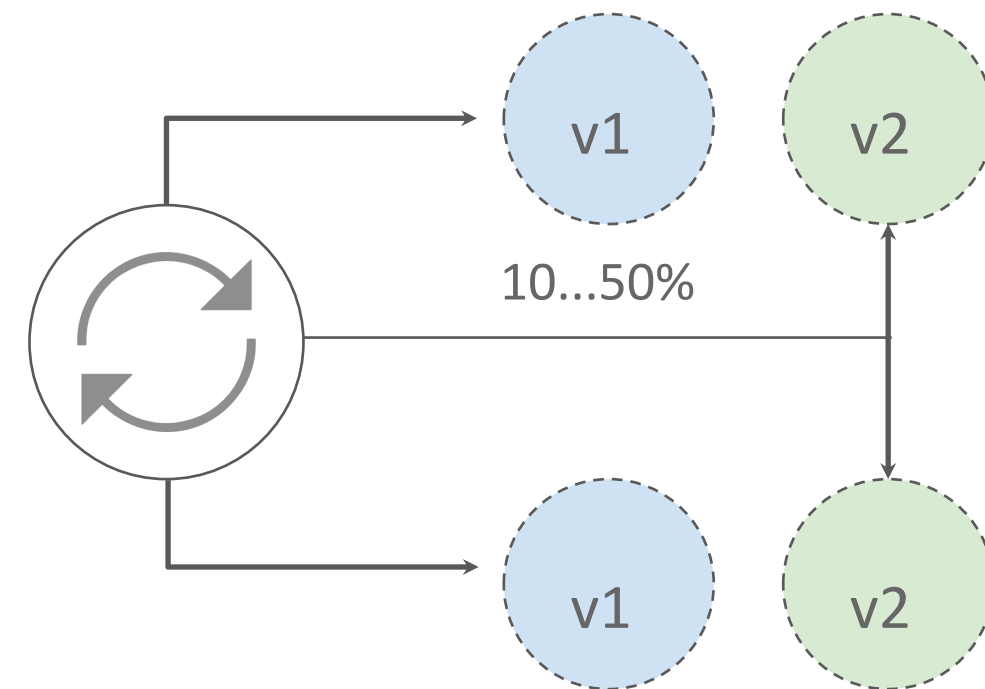
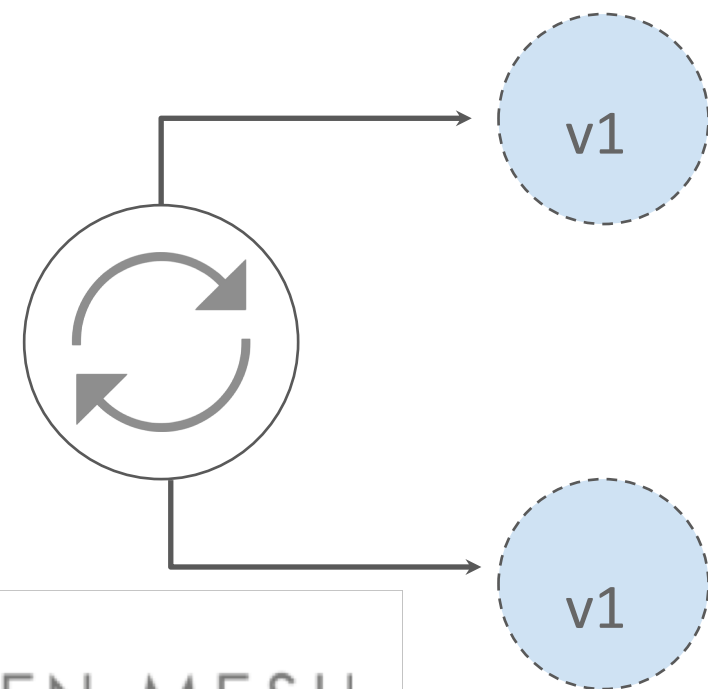
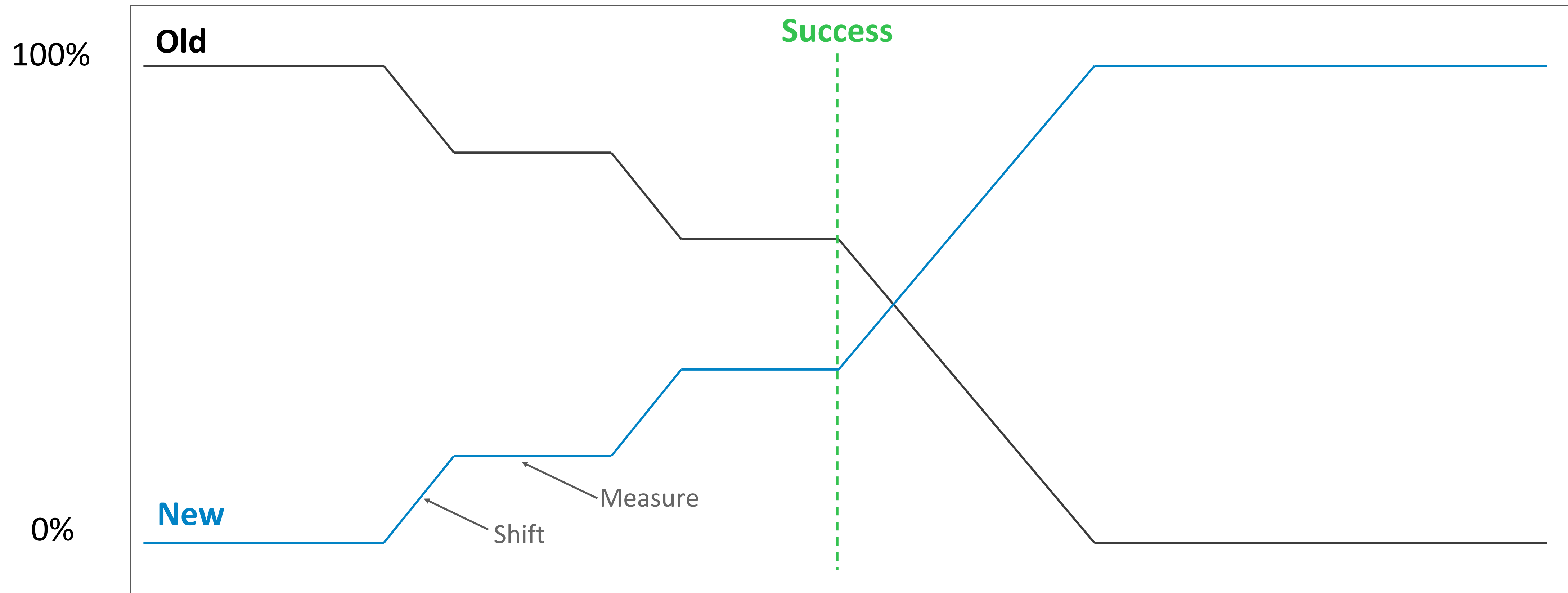


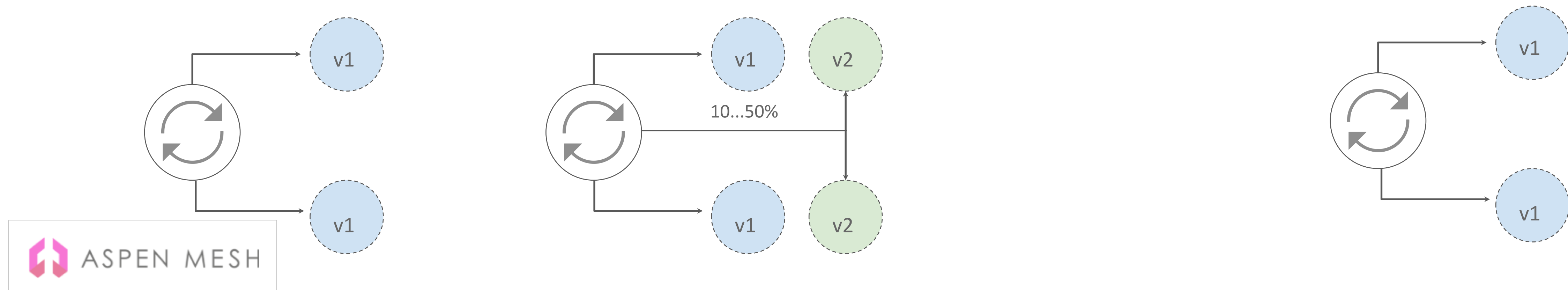
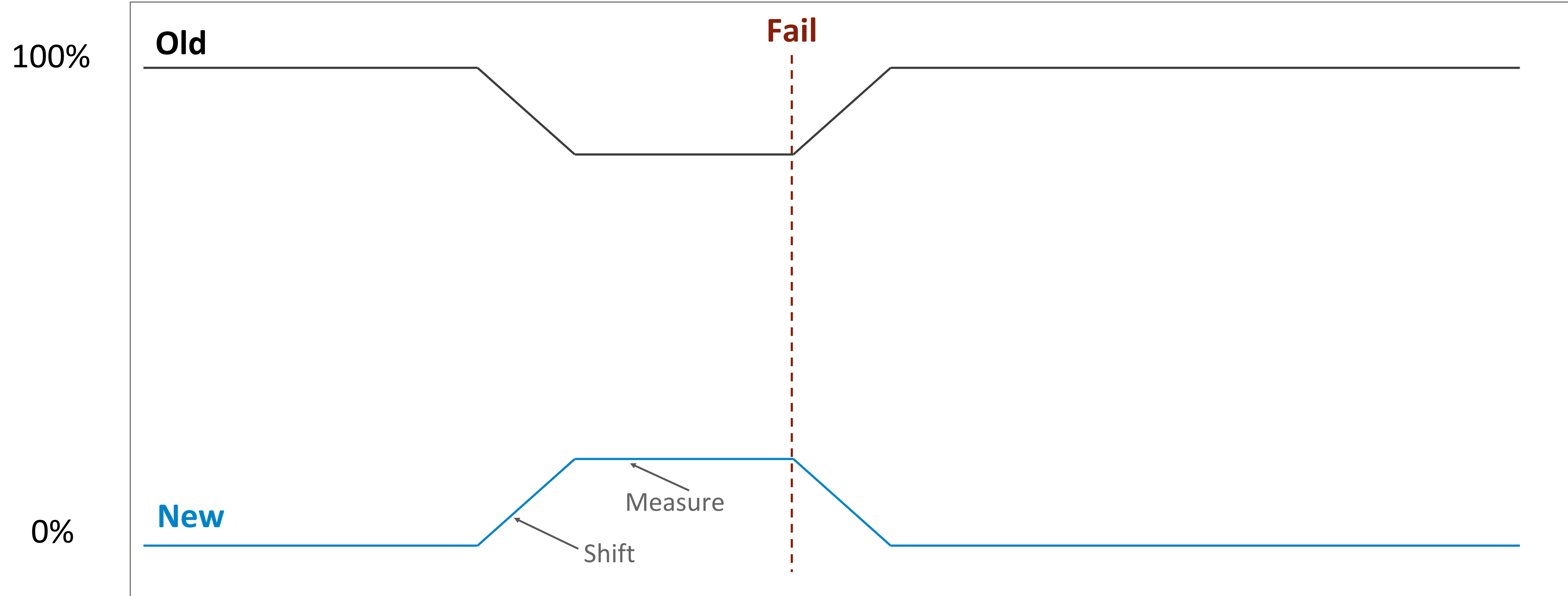


# DRIVING PROGRESSIVE DELIVERY WITH CANARY AND TRAFFIC MIRRORING

# Flagger









# Mirroring

---

A special pre-stage where requests are duplicated: one copy goes to primary and one copy to canary.

Users see only the response from the primary. The service mesh collects metrics and traces on the canary response, then discards.

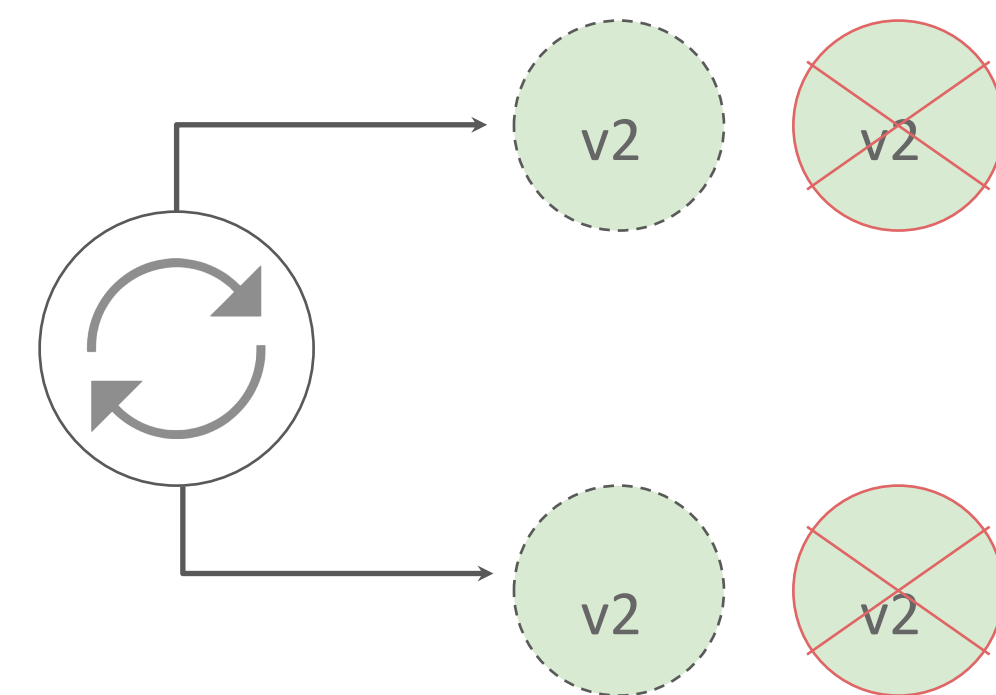
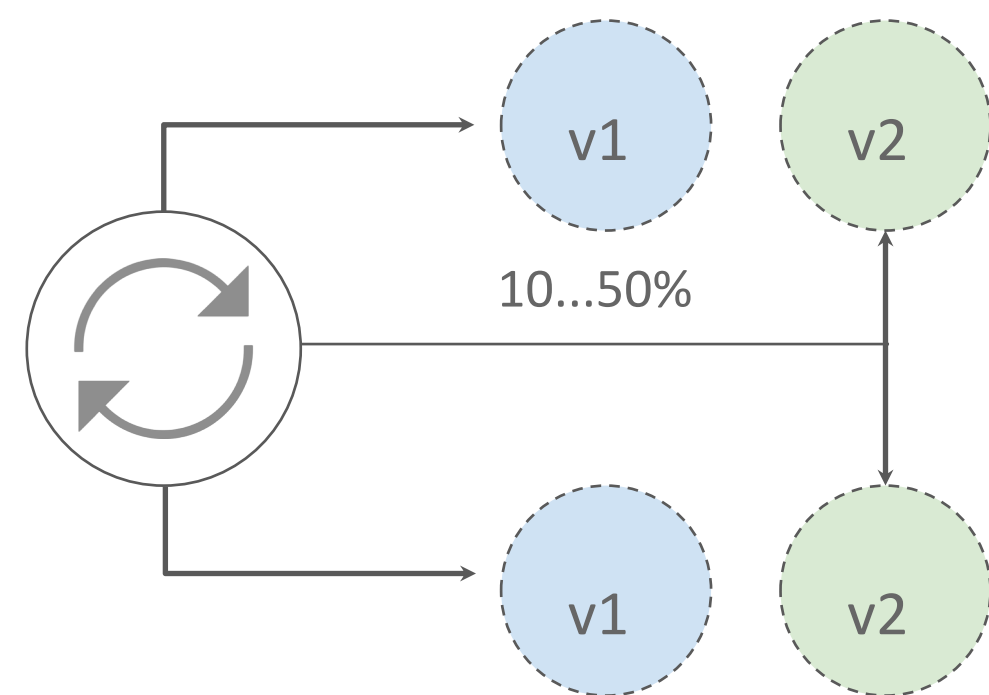
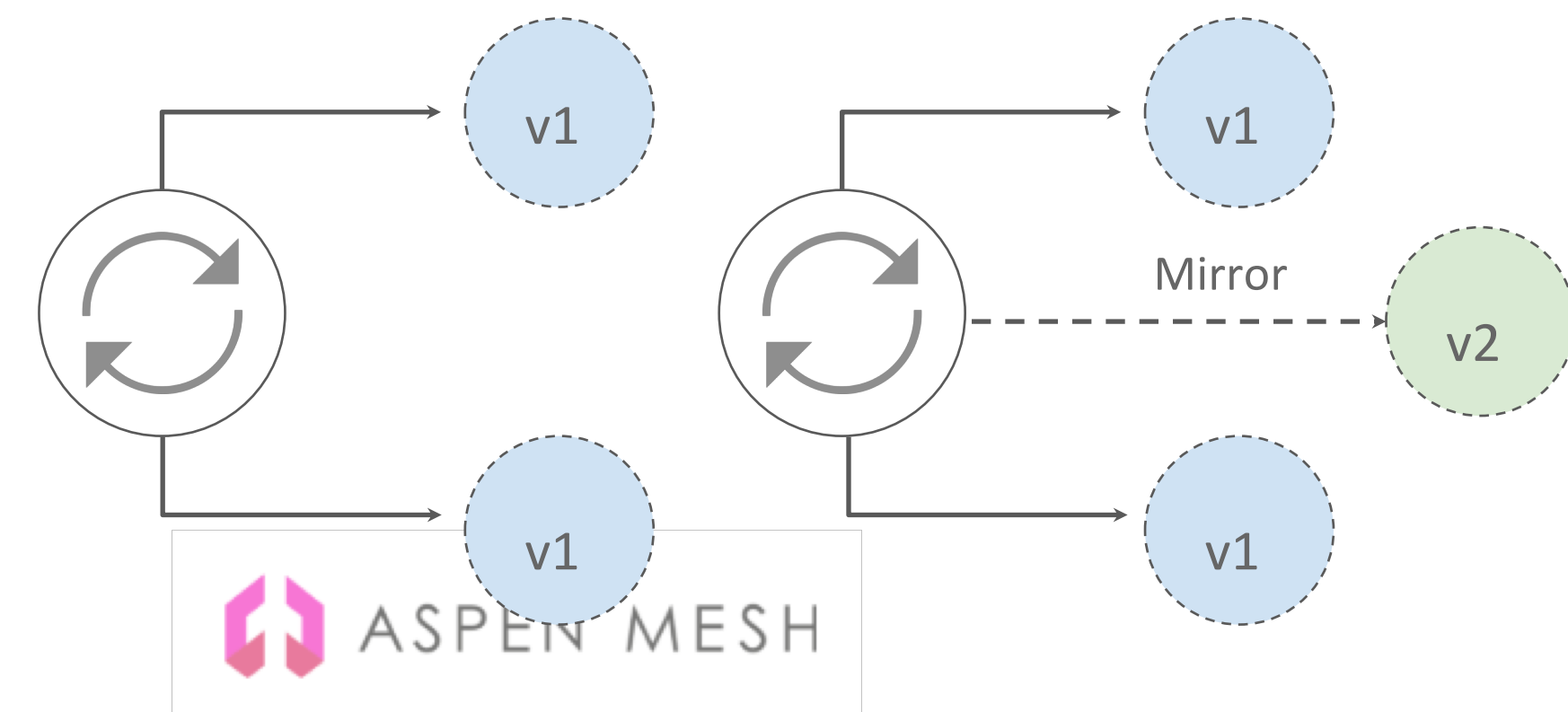
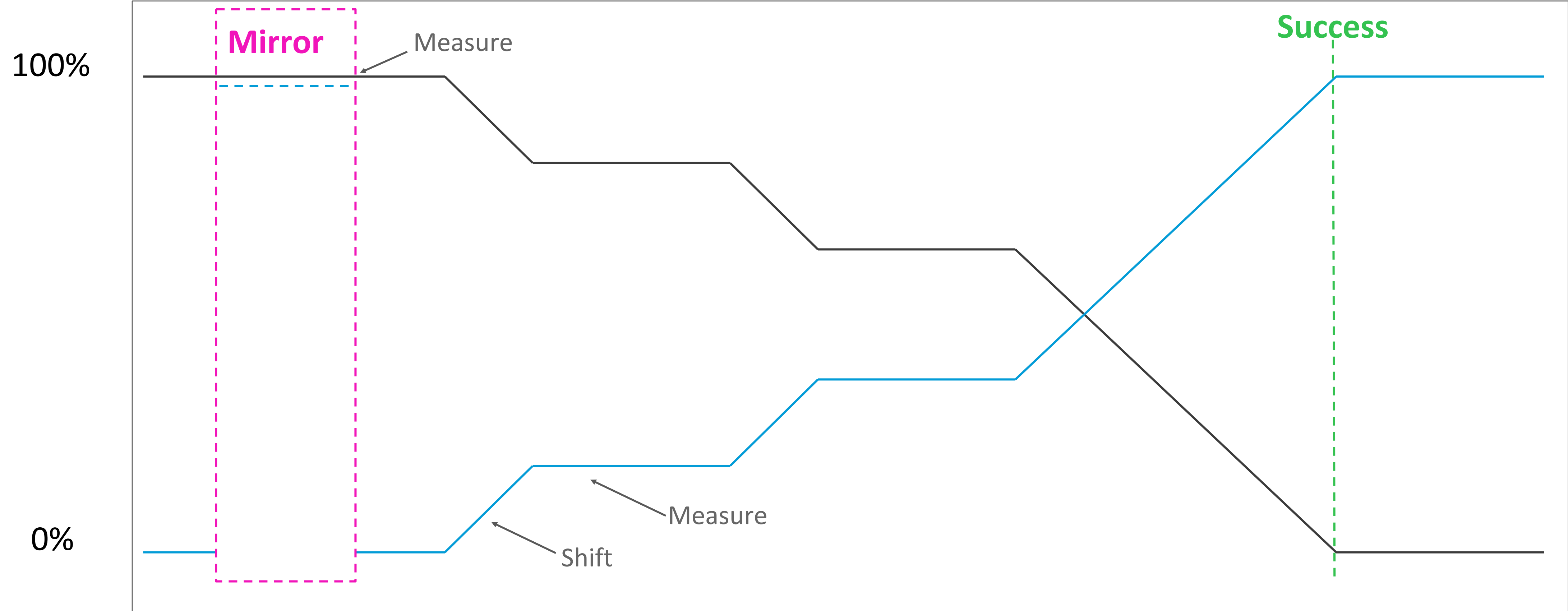
**Pros:** If the canary is a bad release, you can detect it while sending 0% bad responses to users (instead of 10%)

**Cons:** Mirrored requests **must be *idempotent*** (GETs and special kinds of writes).

~~Deduct \$100 from andrews\_account~~

Get andrews\_account balance

Set andrews\_timezone to MDT



# DEMOS

# Thank You



[aspenmesh.io](https://aspenmesh.io)