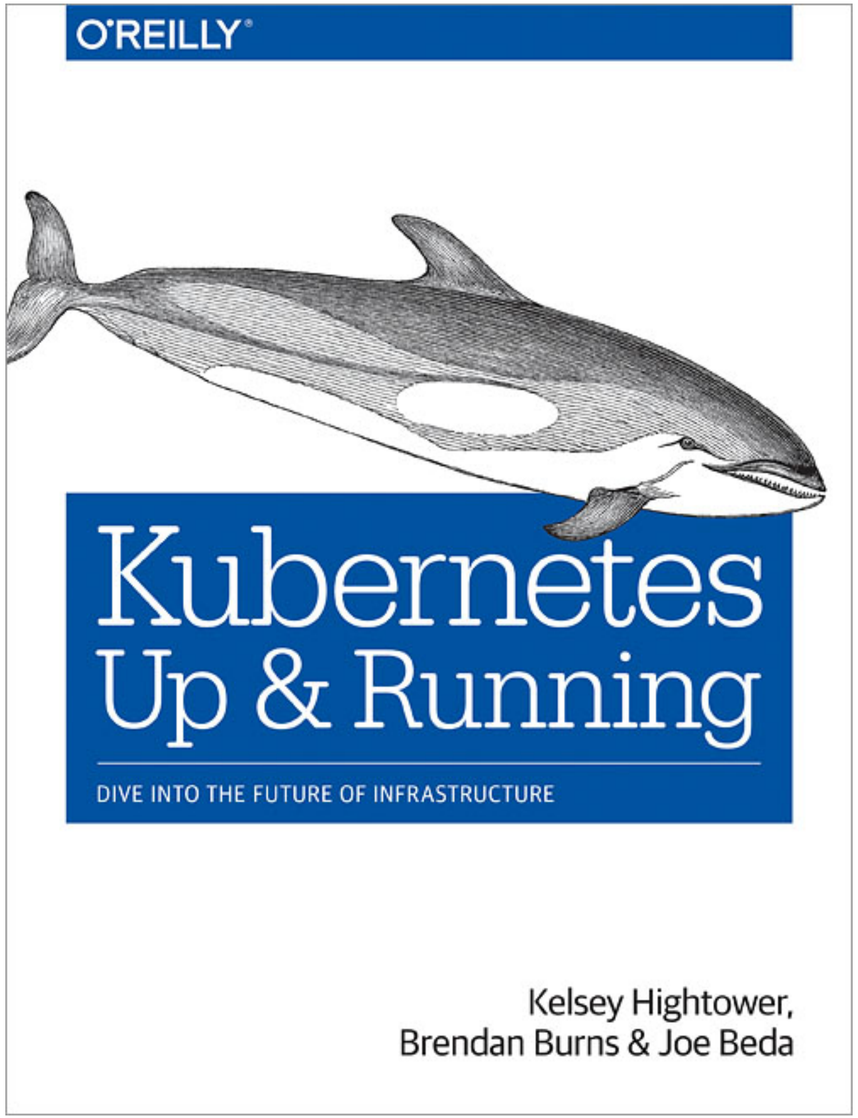

Cloud-Native
Streaming Platform:
Apache Kafka
Meets Kubernetes
apache kafka
@gamussa
#BOSDataDay
@confluentinc

#devkafkaops

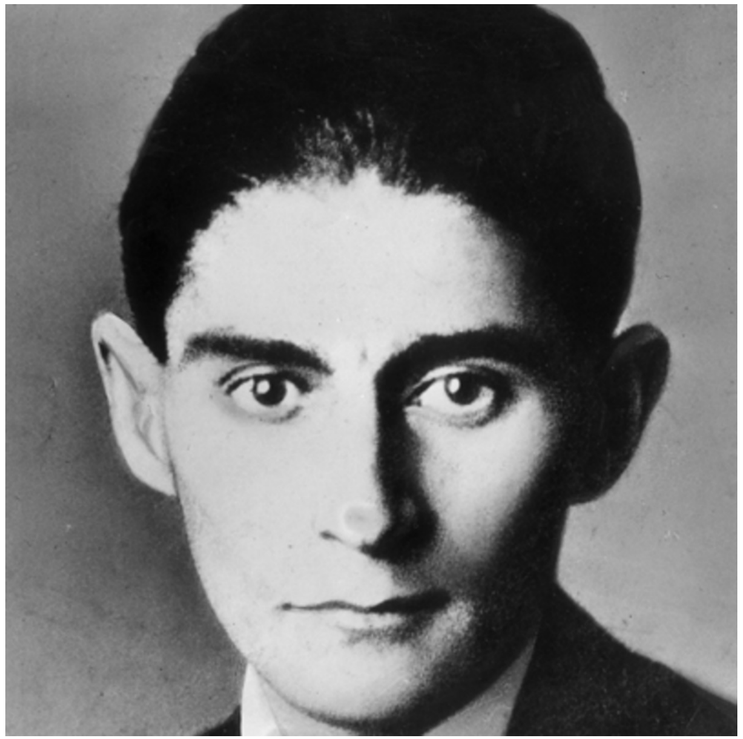

O'REILLY
Kubernetes
Up & Running
DIVE INTO THE FUTURE OF INFRASTRUCTURE
Kelsey Hightower,
Brendan Burns & Joe Beda

https://twitter.com/kelseyhightower/status/963413508300812295

https://twitter.com/kelseyhightower/status/963414038603427840

Don't despair...

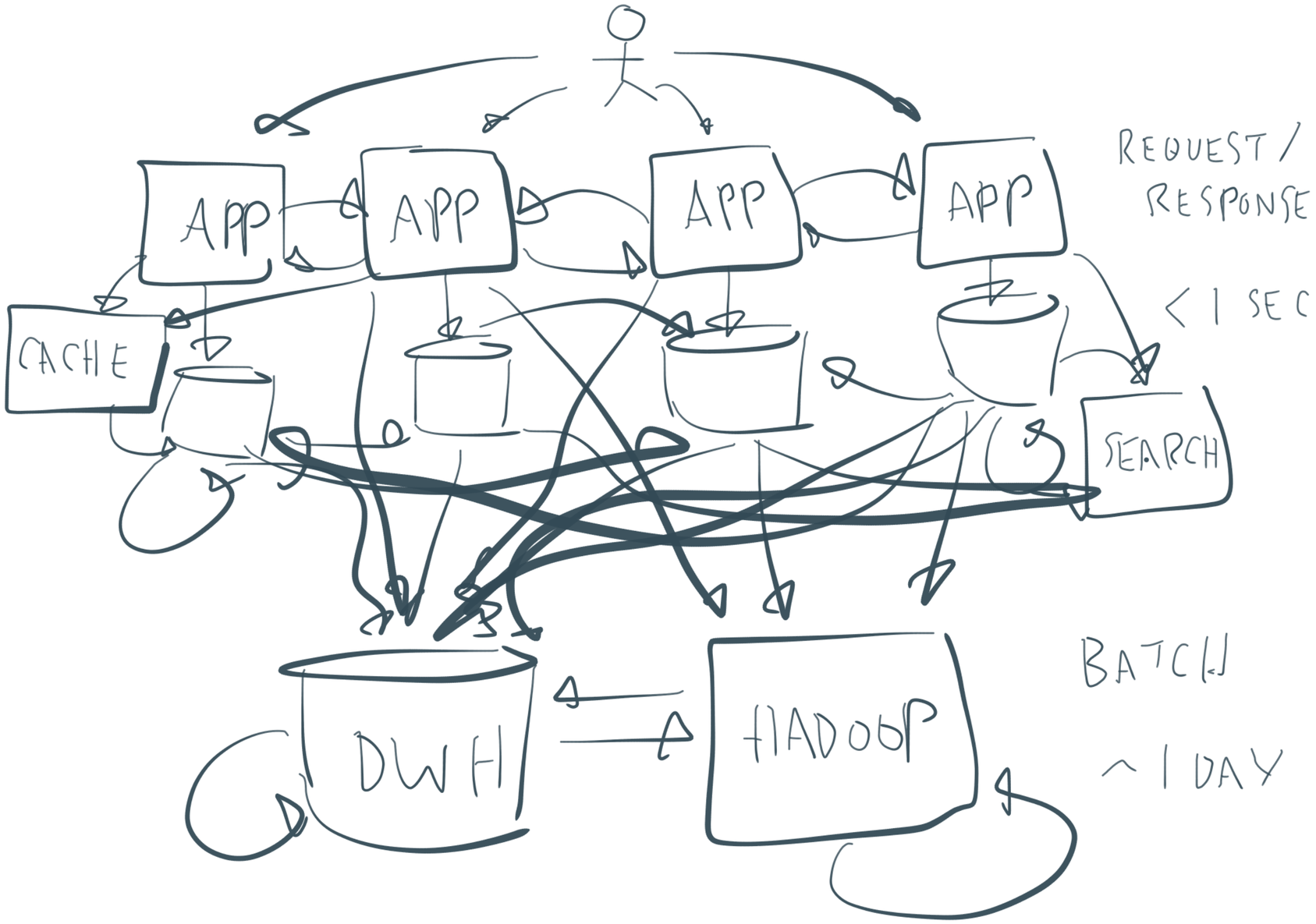
"... not even over the fact that you don't despair. Just when everything seems over with, new forces come marching up, and precisely that means that you are alive"

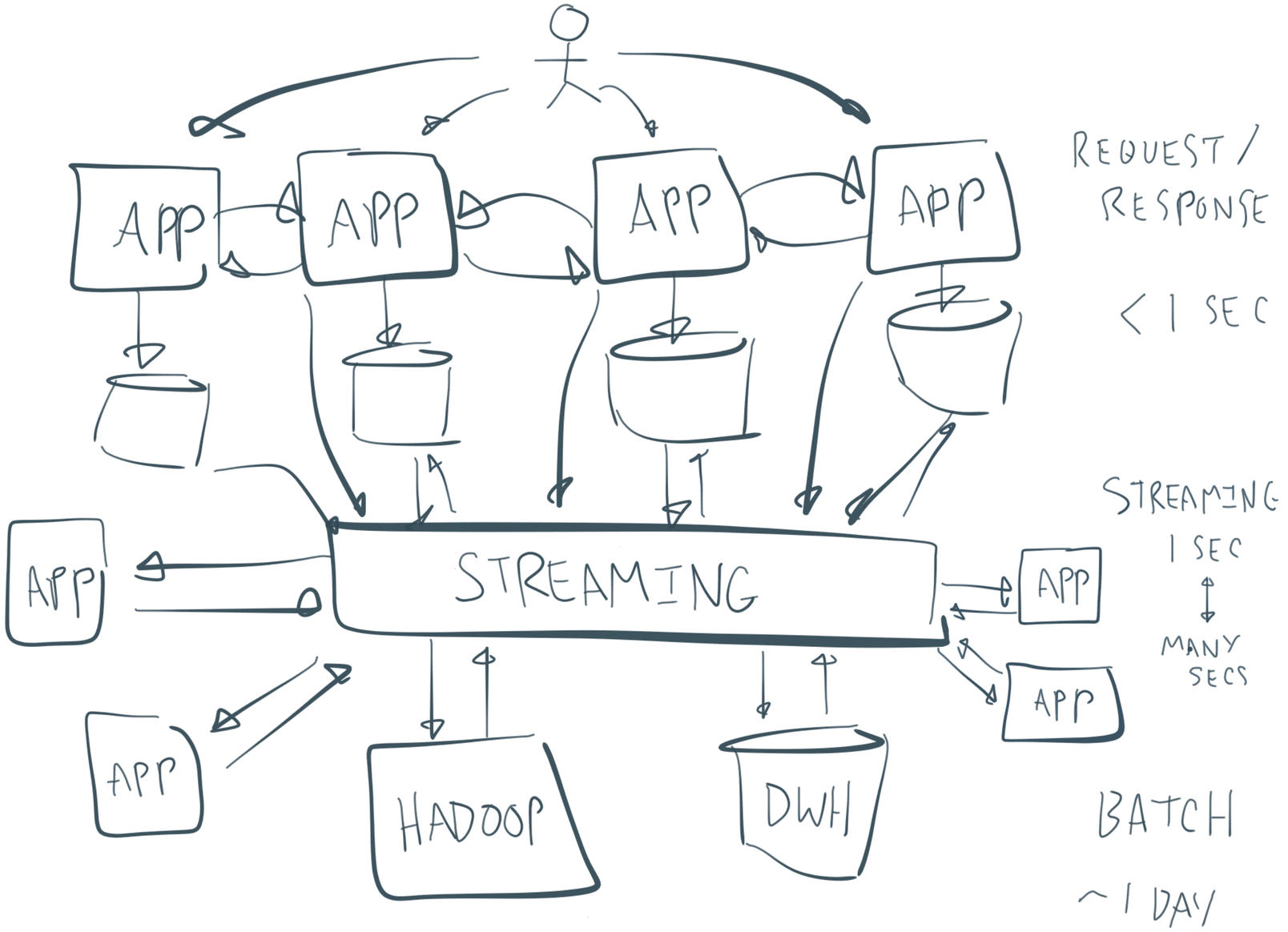


Franz Kafka

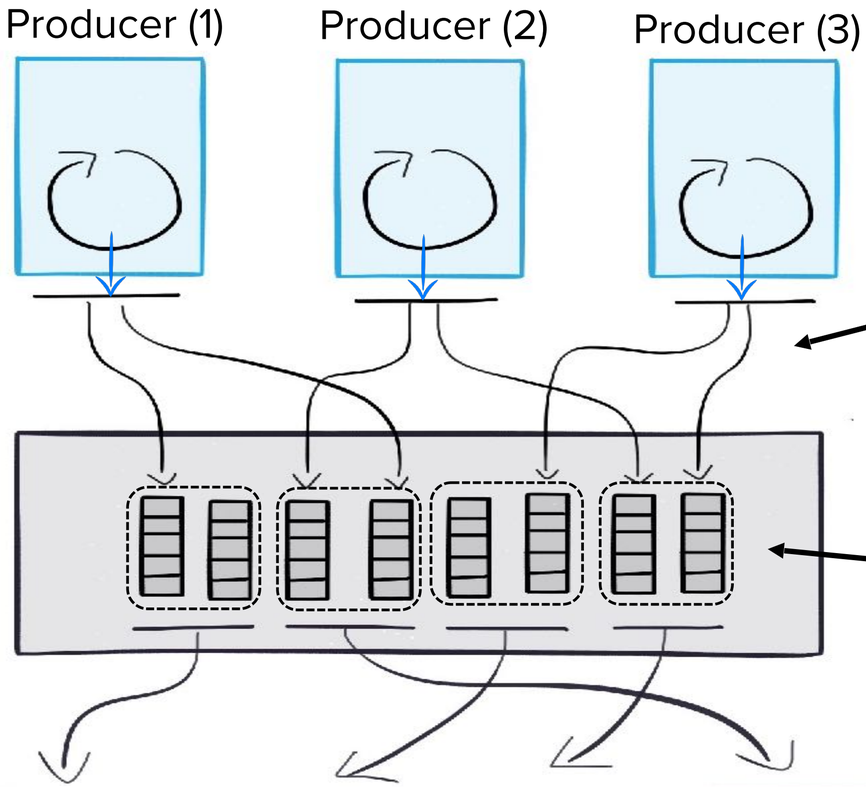
confluent

Kafka Streaming Architecture Fundamentals


APP
APP
APP
APP
REQUEST / RESPONSE
< 1 SEC
CACHE
SEARCH
DWH
HADOOP
BATCH
~ 1 DAY


APP
APP
APP
APP
APP
STREAMING
APP
APP
APP
HADOOP
DWH
REQUEST / RESPONSE
< 1 SEC
STREAMING
1 SEC
MANY SECS
BATCH
~ 1 DAY

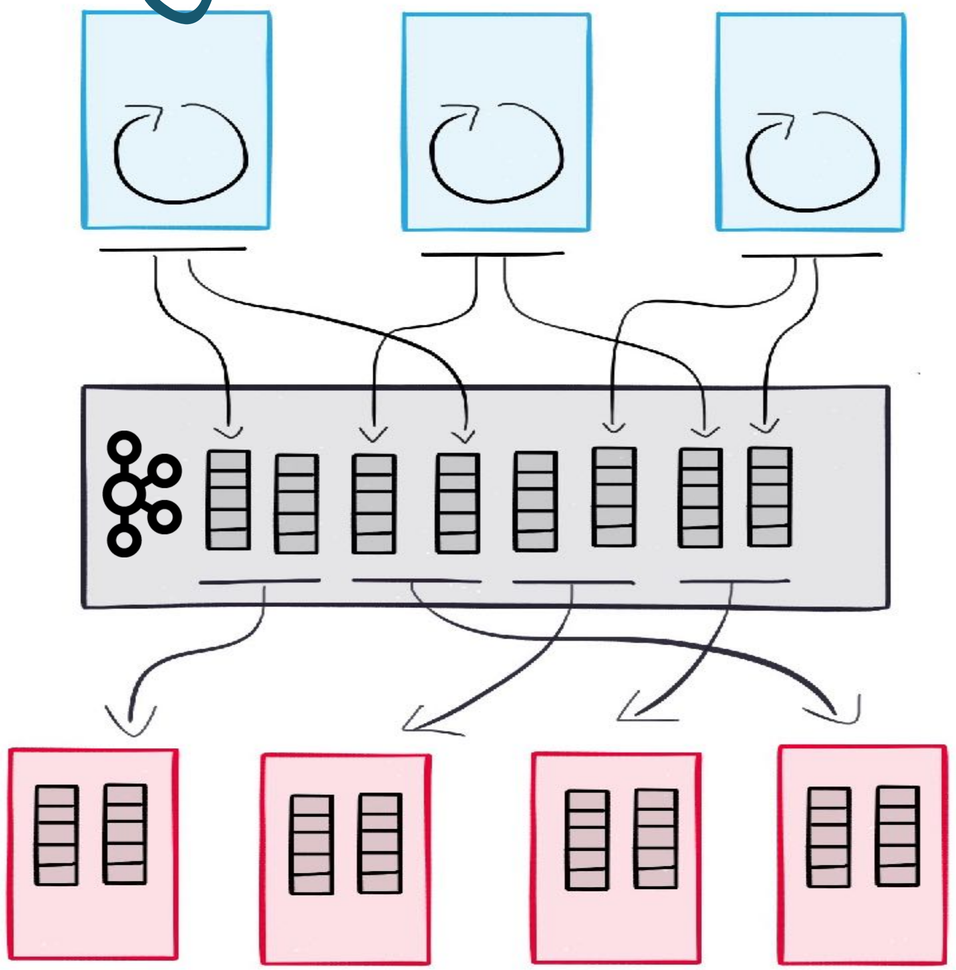
Shard data to get scalability



Messages are sent to different partitions

Partitions live on different machines

Linearly Scalable Architecture

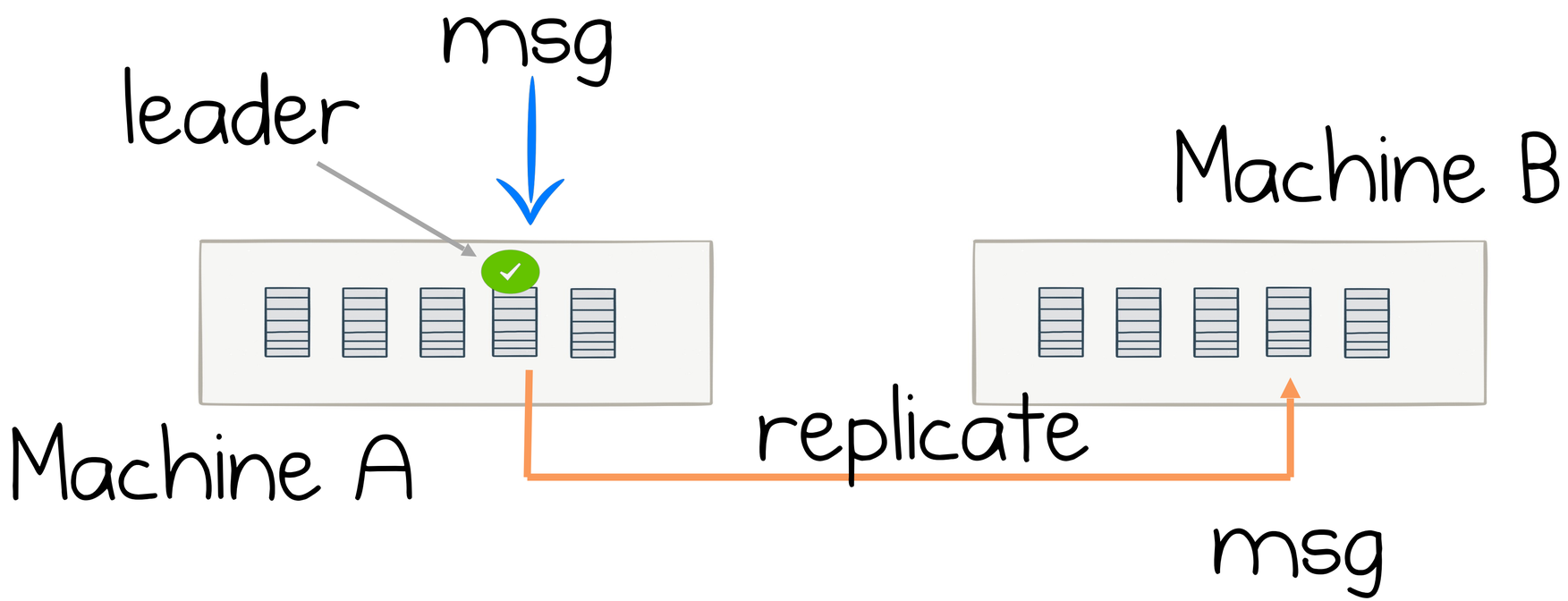


Single topic:

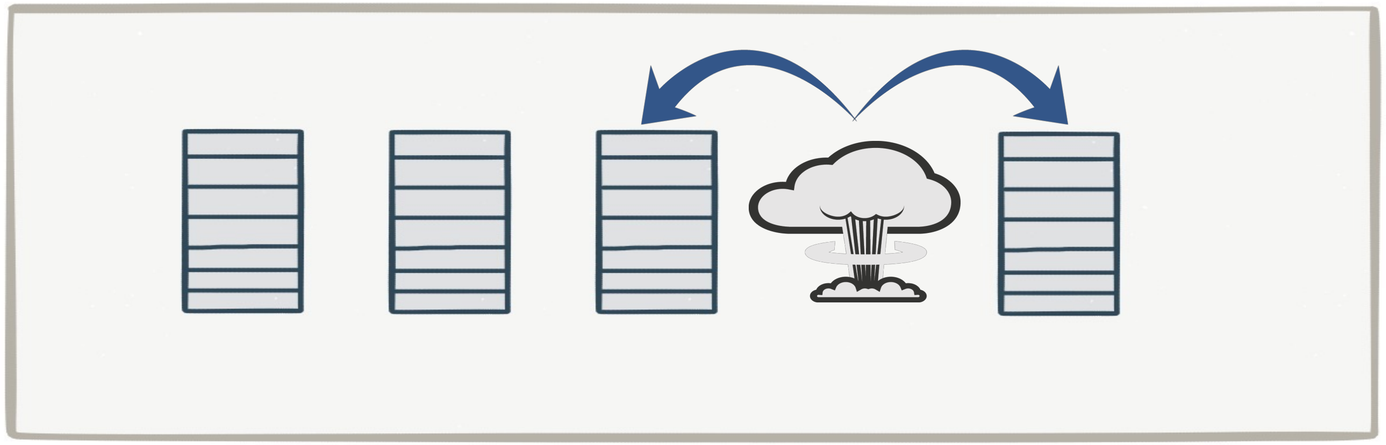
- Many producers machines
- Many consumer machines
- Many Broker machines

No Bottleneck!!

Replicate to get fault tolerance

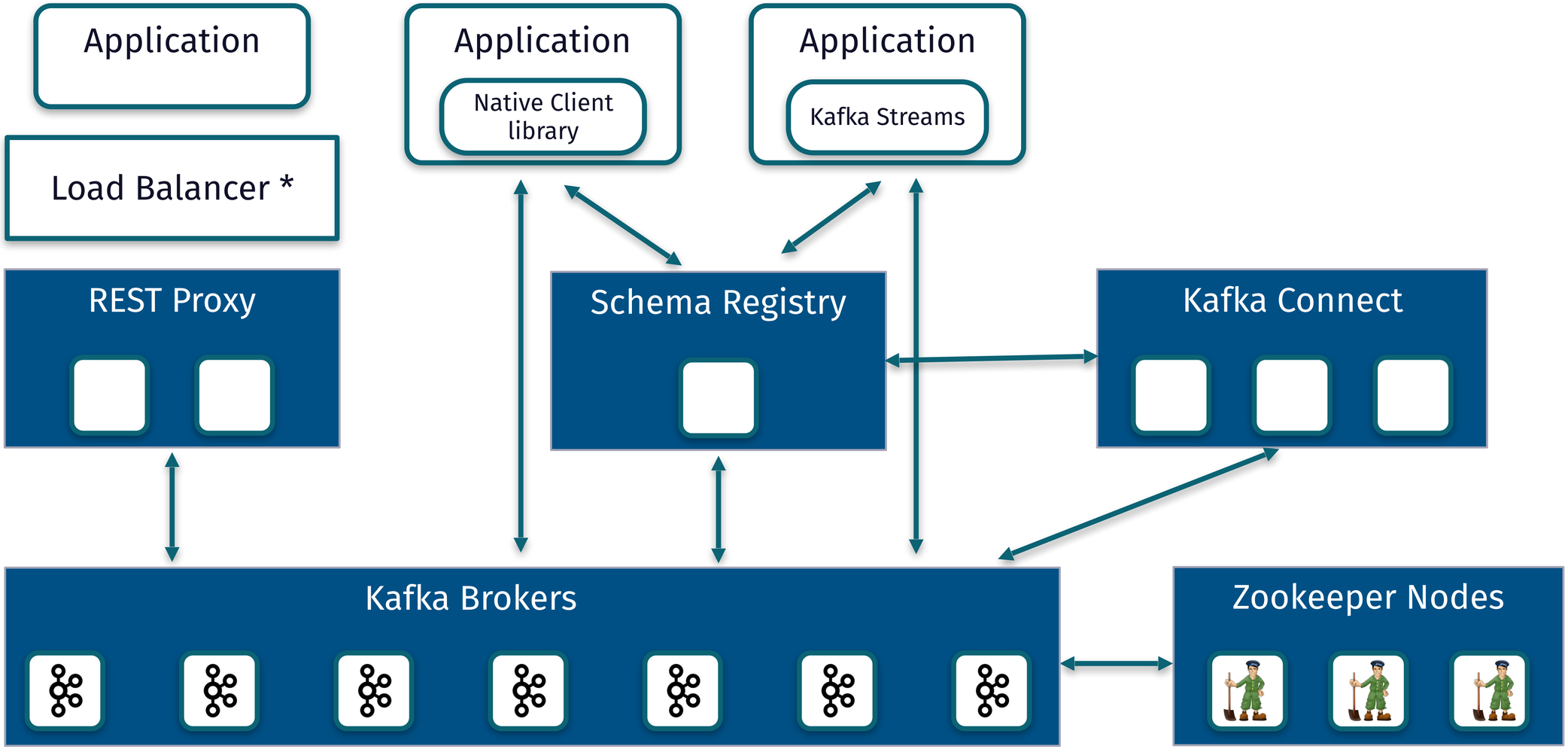


Replication provides resiliency

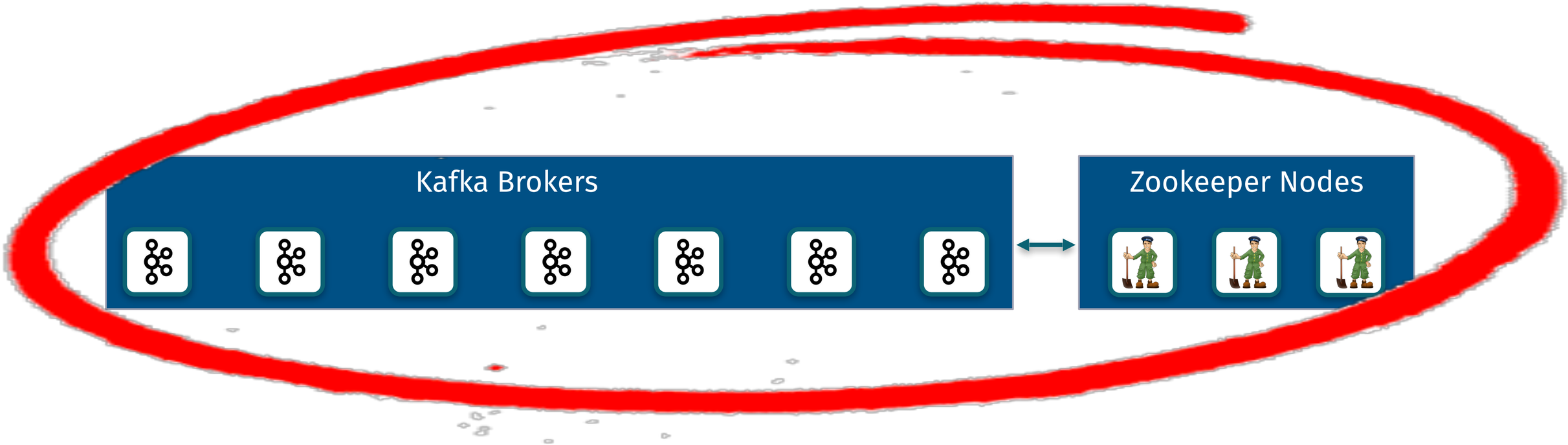


A 'replica' takes over on machine failure

High-level Architecture

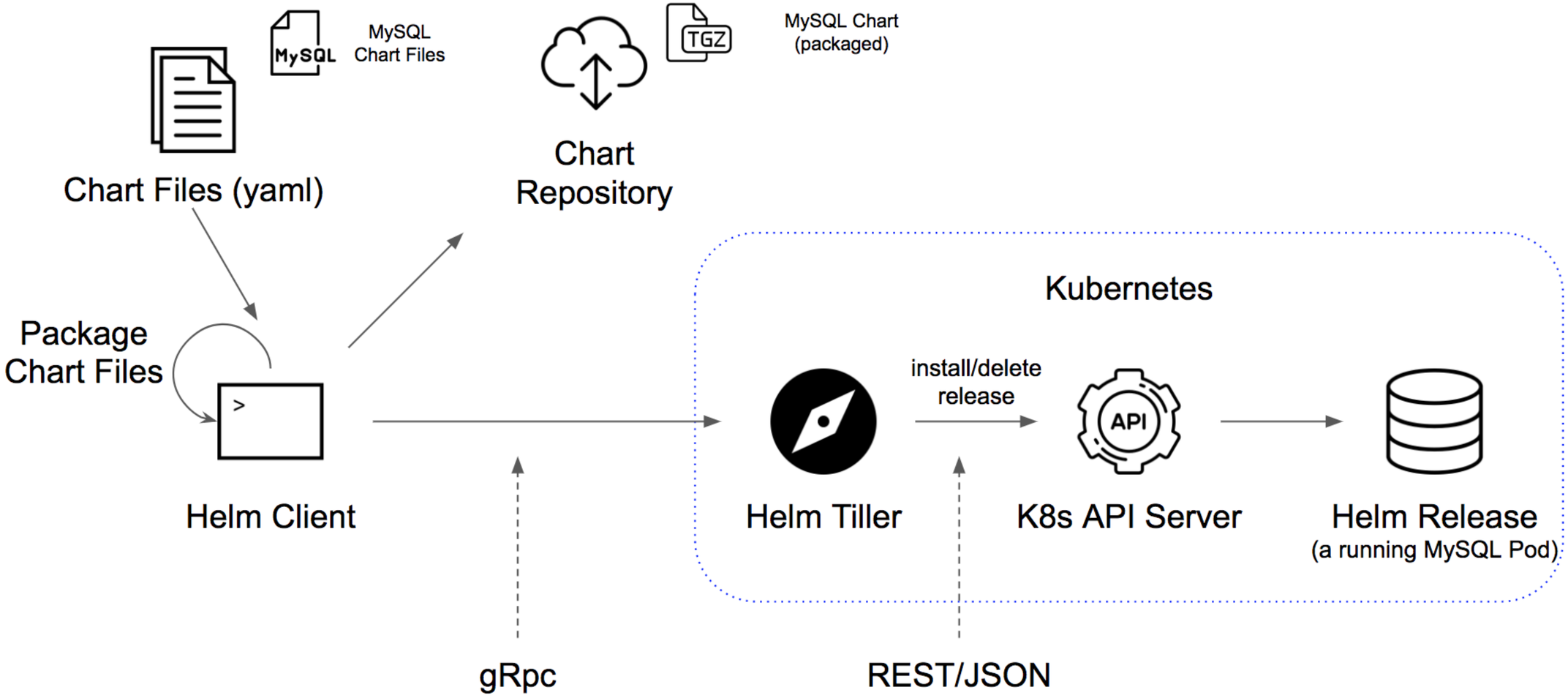


Bare minimum

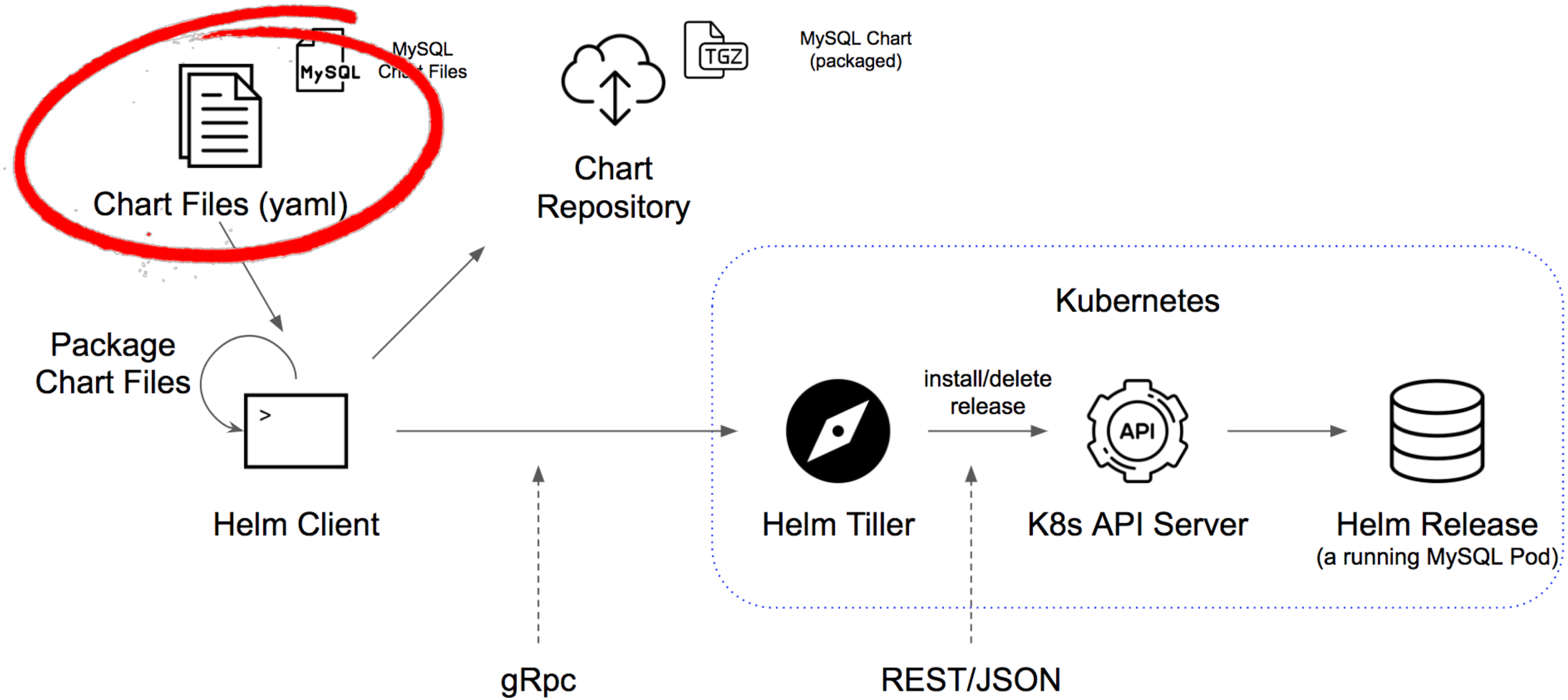


Workloads Deployment

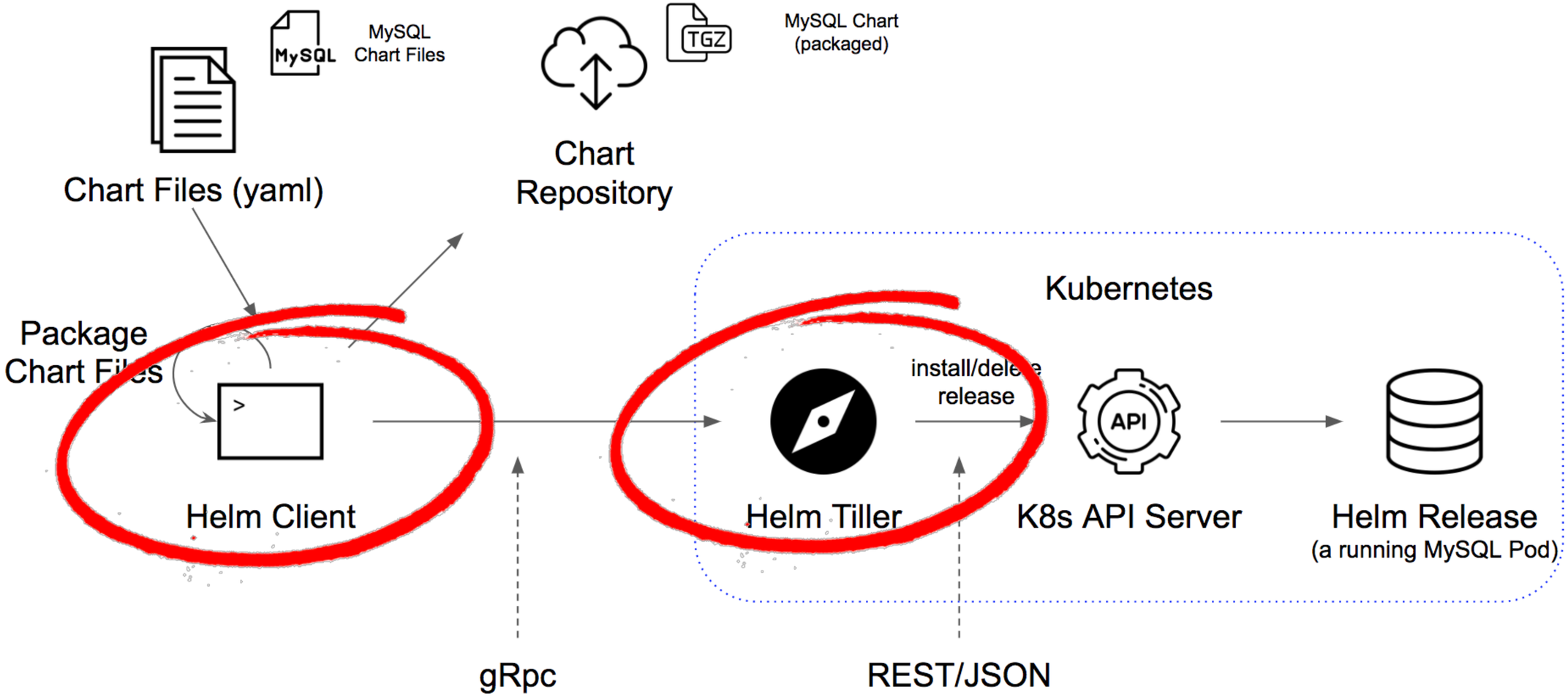
Helm Charts

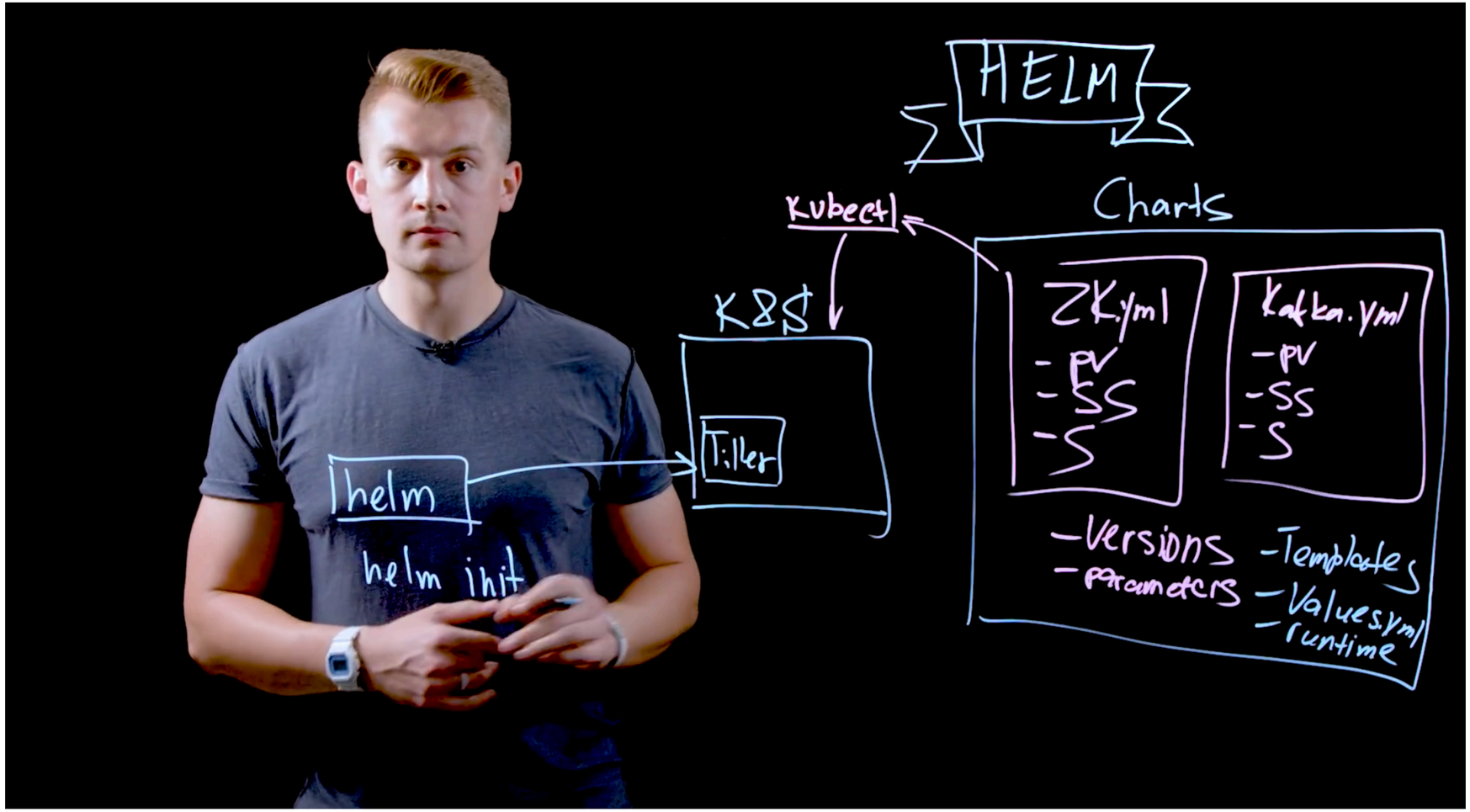

MySQL
Chart Files
TGZ
MySQL Chart
(packaged)
Chart Files (yaml)
Chart
Repository
Kubernetes
Package
Chart Files
>
install/delete
release
API
Helm Client
Helm Tiller
K8s API Server
Helm Release
(a running MySQL Pod)
gRpc
REST/JSON

Helm Charts


MySQL
Chart Files
Chart Files (yaml)
Chart
Repository
TGZ
MySQL Chart
(packaged)
Package
Chart Files
>
Helm Client
Kubernetes
Helm Tiller
install/delete
release
API
K8s API Server
Helm Release
(a running MySQL Pod)
gRpc
REST/JSON

Helm Charts


MySQL
Chart Files
TGZ
MySQL Chart
(packaged)
Chart Files (yaml)
Chart
Repository
Package
Chart Files
>
Kubernetes
install/delete
release
API
Helm Client
Helm Tiller
K8s API Server
Helm Release
(a running MySQL Pod)
gRpc
REST/JSON



https://cnfl.io/helm_video

Kafka deployment checklist

PVC for Storage

StatefulSet for 3-node zk

Optional Pod Anti-Affinity to spread the ZK ensemble across nodes

Headless Service

ConfigMap for Prometheus JMX exporter

Uses ZK Headless Svc

PVC for Storage

StatefulSet for n-node Kafka

A group of NodePort Services for external traffic

ConfigMap for Prometheus JMX exporter

Meet Kubernetes operator

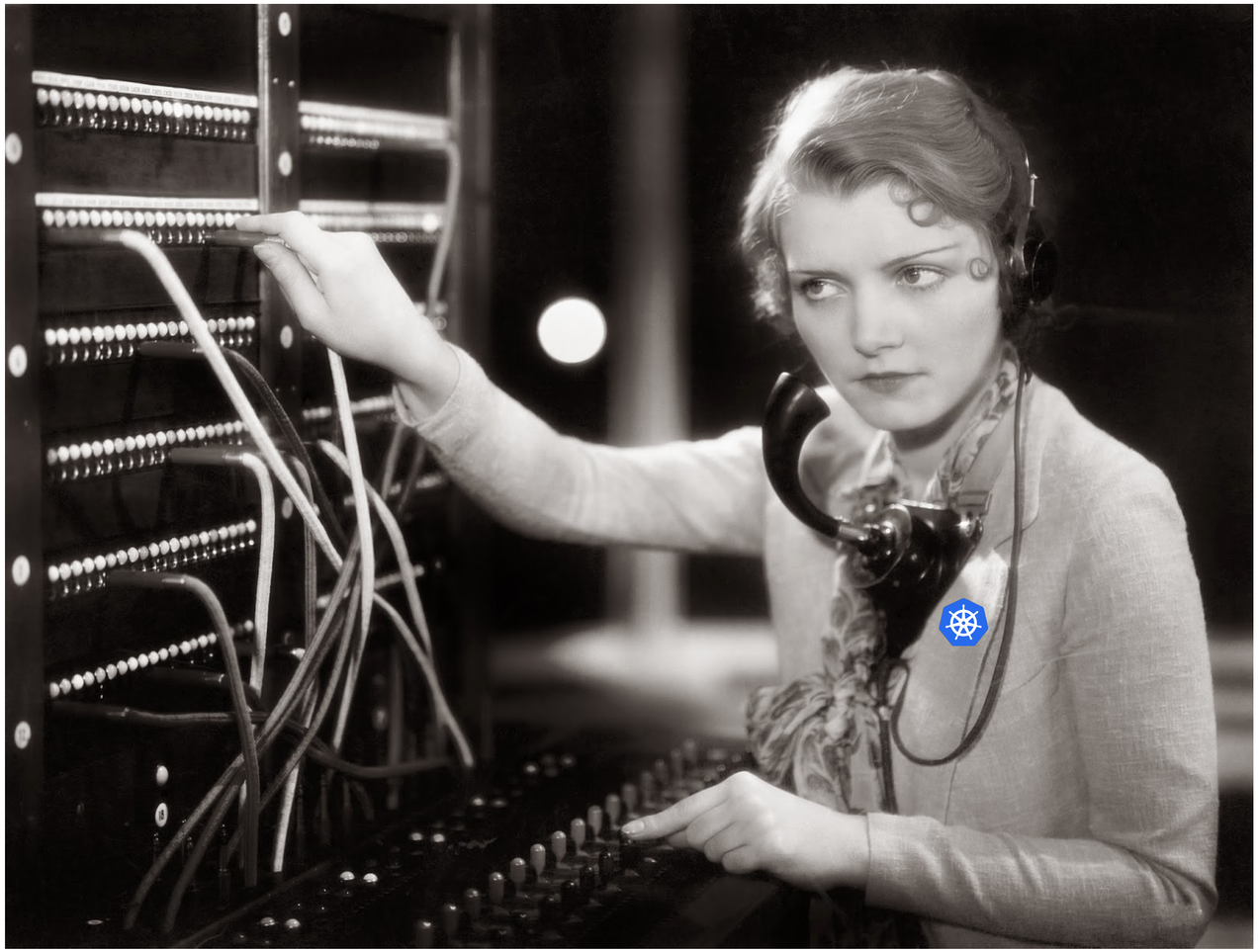
Kubernetes operator

Embedded with operational knowledge

of both data software and

Kubernetes

- Backup/restore
- Scale up/down
- Rebalance data
- Regular health checks



Controller

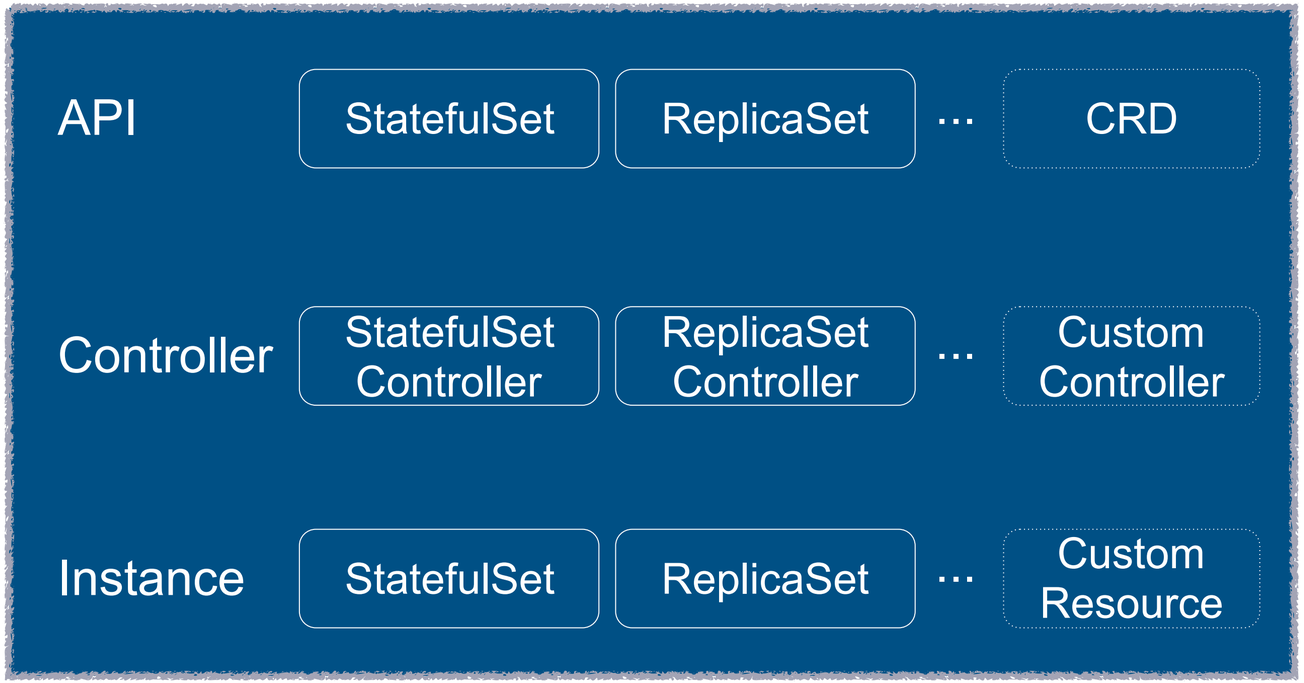
Brain behind Kubernetes

resources

e.g. replication controller, namespace controller etc.

Custom Resource Definition(CRD)

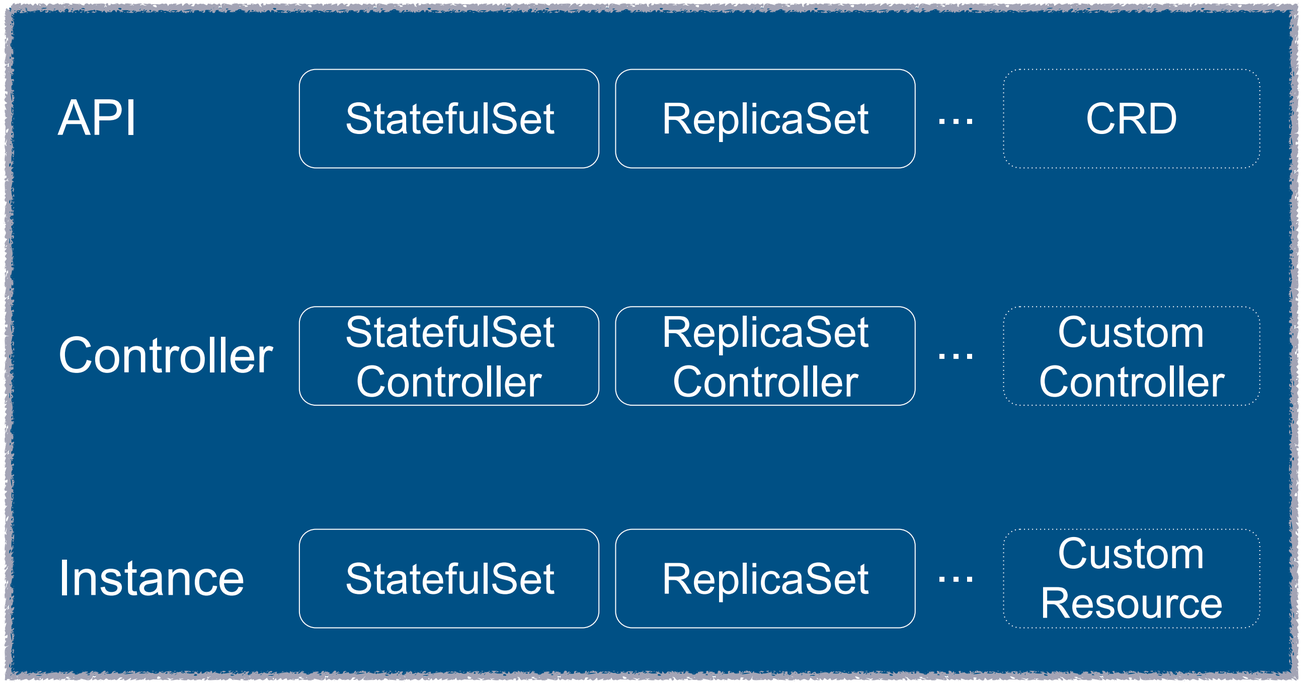
Extend existing
Kubernetes API



Custom Resource Definition(CRD)

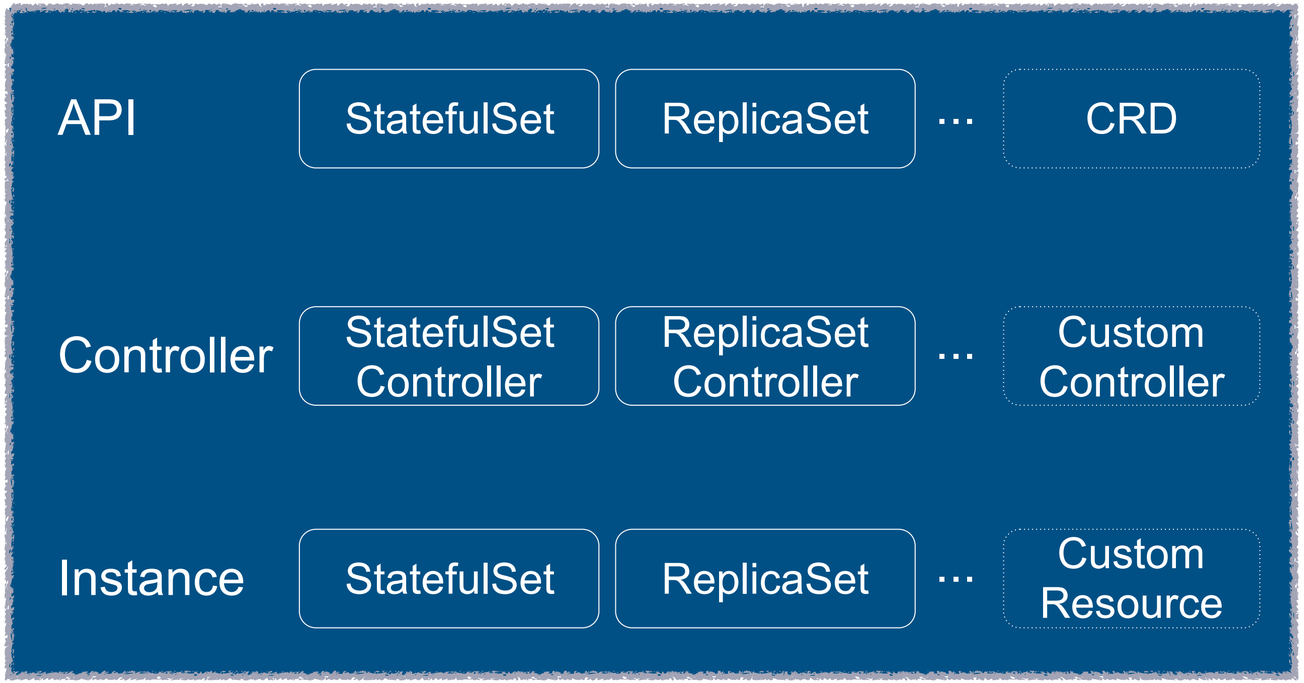
Usually works together

Custom Controller



Custom Resource Definition(CRD)

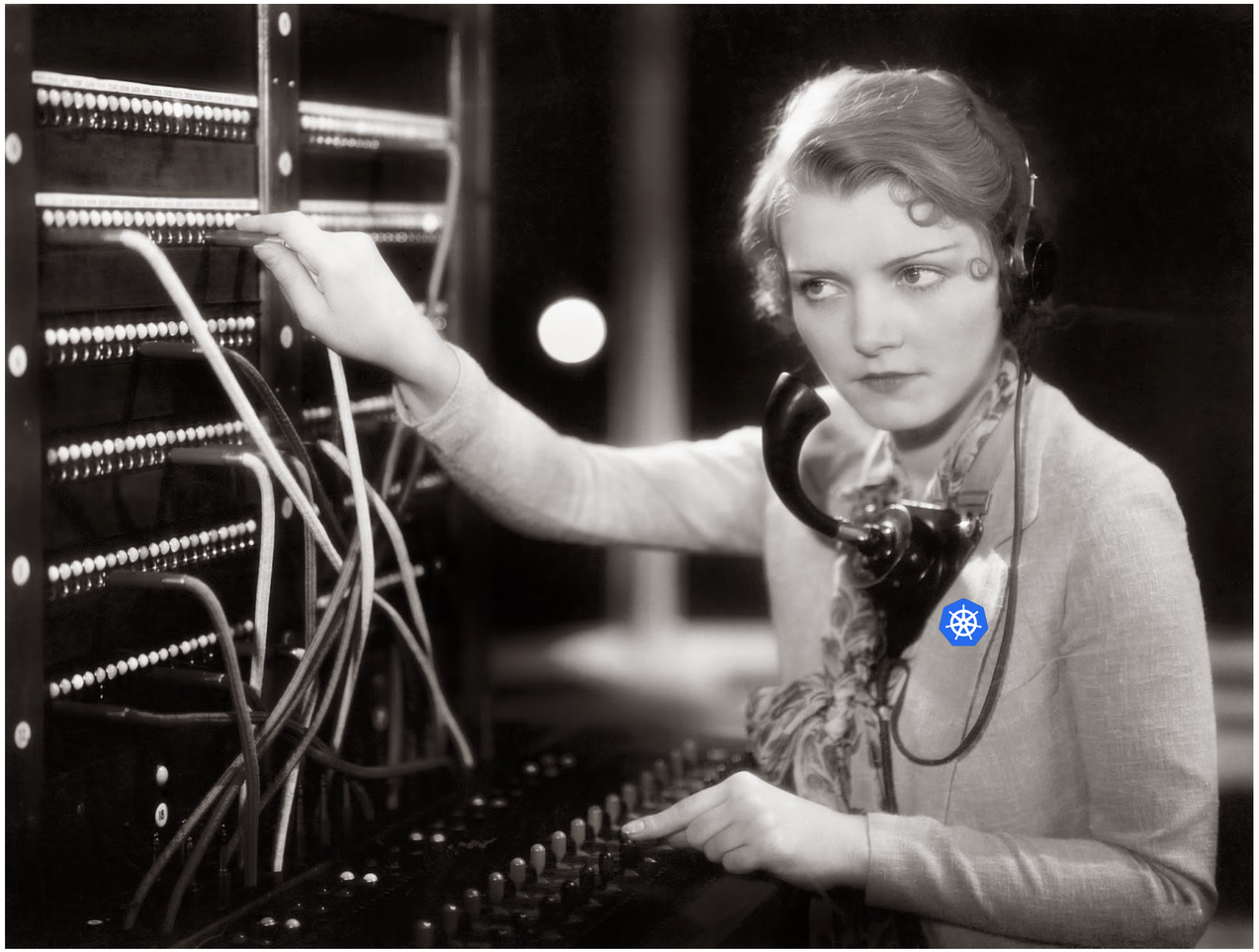
Users can create and access Customer Resources with kubectl, just as they do for built-in resources like pods.



Operator

Deploy and Manage your production streaming platform with Confluent operator.

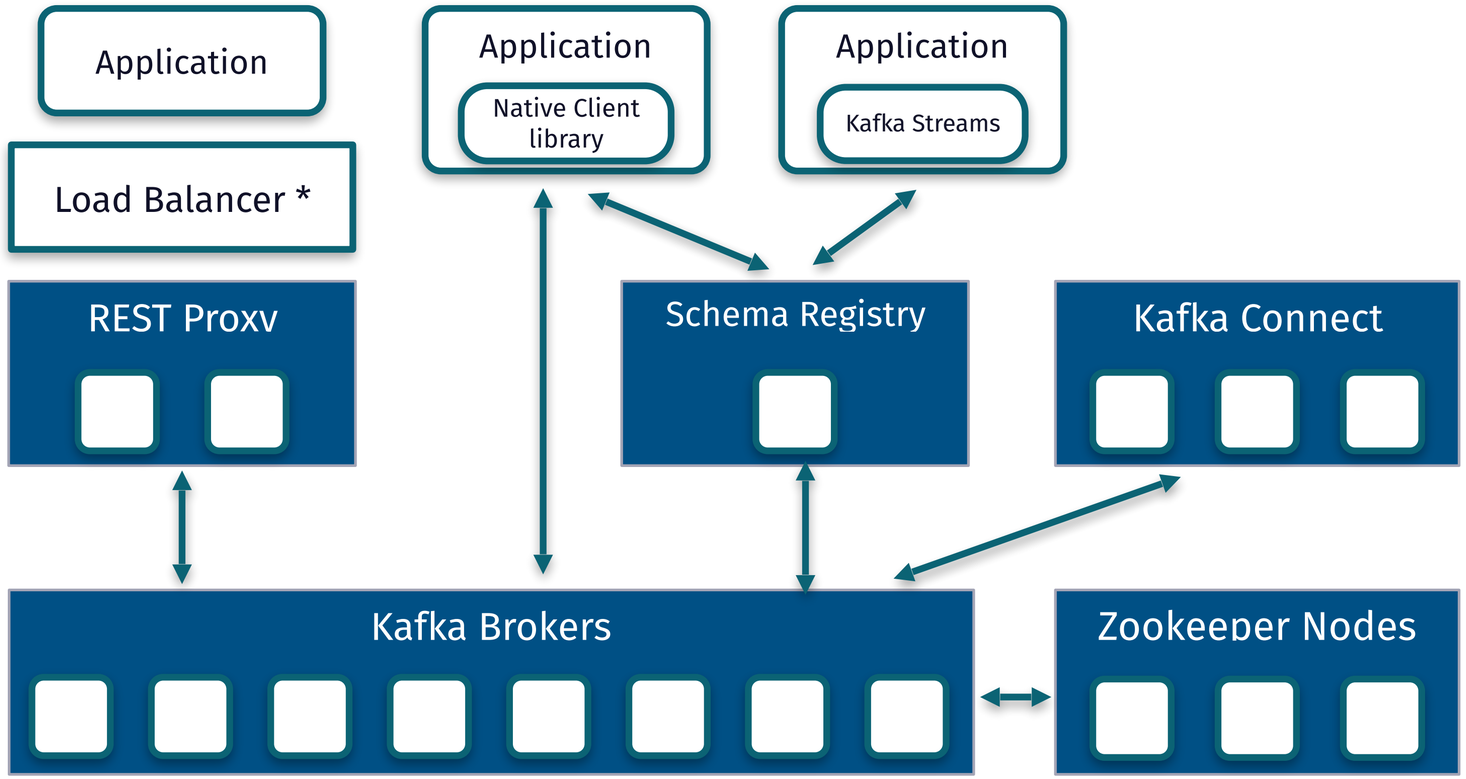
- Automated Provisioning
- Platform Operations
- Resiliency
- Monitoring



Confluent Platform Reference Architecture

Each Confluent Platform component has specific characteristics:

- Security (SSL certificates)
- DNS names and zones
- Host selection
- Fault tolerance
- Scaling

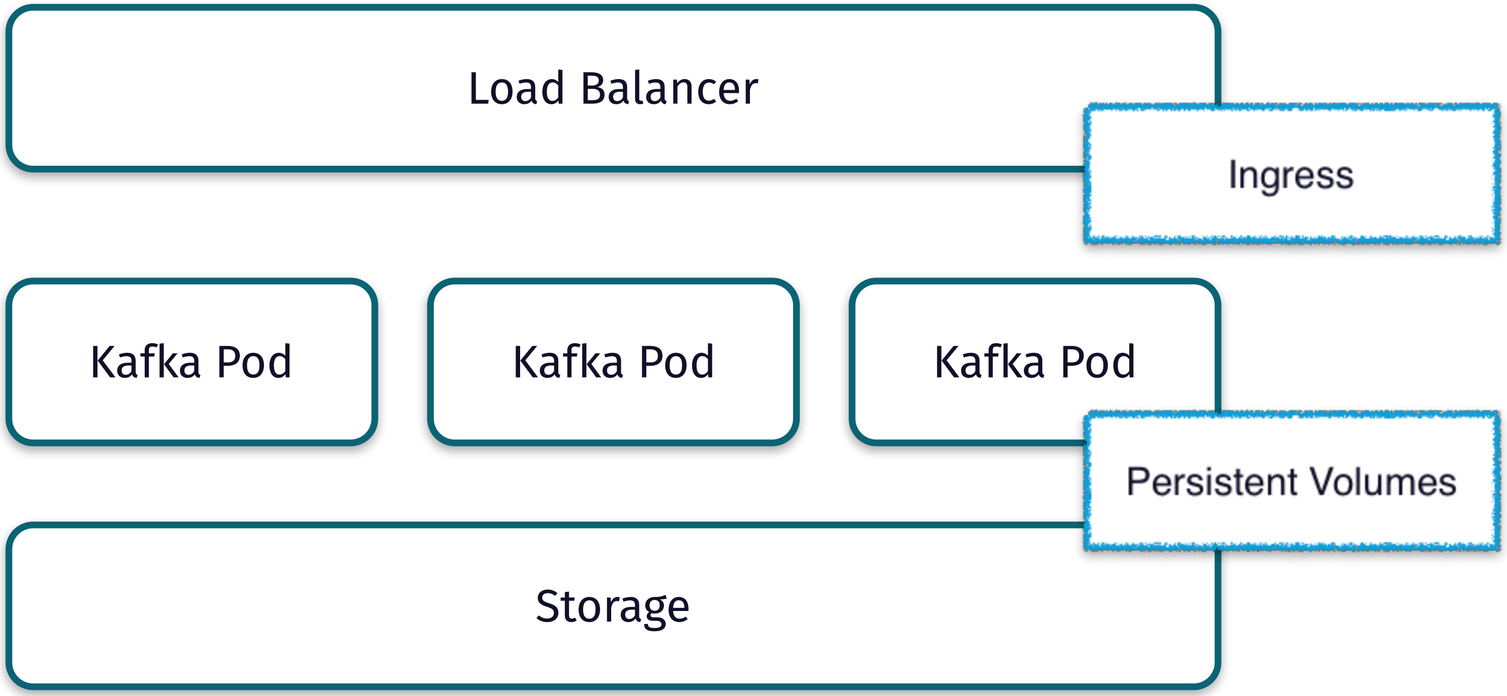


Confluent Operator: Automated Provisioning

```
> cat kafka.yml

## Kafka Cluster
##
kafka:
  name: kafka
  replicas: 3
  resources:
    cpu: 200m
    memory: 1Gi
  external:
    enabled: false
    domain: ""
  tls:
    enabled: true
    domain: devoops.ru
    fullchain: |-
    privkey: |-


> helm install -f kafka.yml --name kafka
```



Confluent Operator: Scale Horizontally

Automate scaling:

- Spin up new broker pod(s)
- Distribute partitions to the new broker(s)
  - Determine balancing plan
  - Execute balancing plan
  - Monitor resources

```
> cat kafka_new.yml

## Kafka Cluster
##
kafka:
  name: kafka
  replicas: 5
  version: 5.0.0

> helm upgrade -f kafka_new.yml --name kafka
```

Confluent Operator: Rolling Upgrade

Automated rolling upgrade with no downtime for Kafka.

- Stop broker
- wait for leader election to complete
- Start broker with new version
- wait for zero under-replicated-partitions
- Repeat

```
> cat kafka_new.yml

## Kafka Cluster
##
kafka:
  name: kafka
  replicas: 5
  version: 5.1.0

> helm upgrade -f kafka_new.yml --name kafka
```

Will it fly? Let's see

Confluent operator

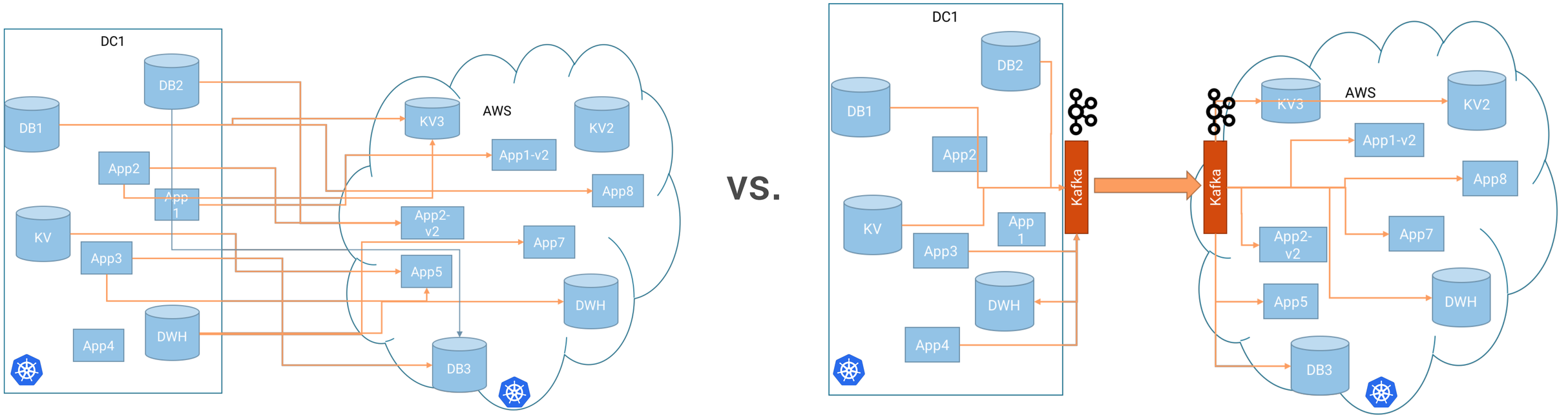
Automate provisioning

Scale your Kafkas and CP clusters elastically

Monitor SLAs through Confluent Control Center or Prometheus

Operate at scale with enterprise support from Confluent

Advanced use cases



Don't despair!

Resources and Next Steps

https://cnfl.io/helm_video

https://cnfl.io/cp-helm

https://cnfl.io/k8s

https://slackpass.io/confluentcommunity
#kubernetes

THANKS!

@gamussa
viktor@confluent.io