WHO'S TWEETING ABOUT #DSGTECH AND #KSQL?

USING APACHE KAFKA, KAFKA CONNECT, KSQL AND KUBERNETES

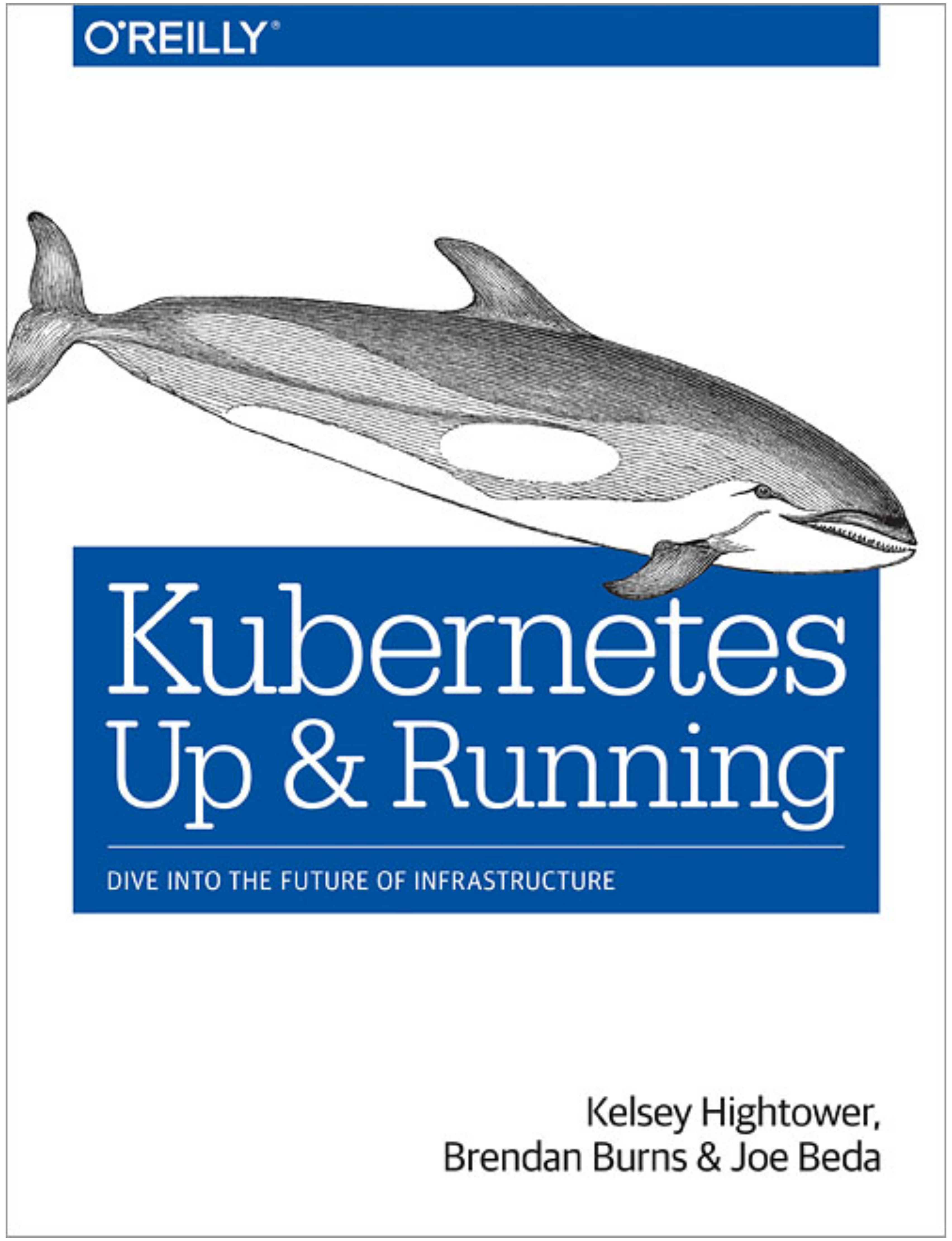
FEBRUARY 2019 | @GAMUSSA

confluent

APACHE kafka

A distributed streaming platform

#devkafkaops


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

Kelsey Hightower
@kelsehightower
Following

Kubernetes has made huge improvements in the ability to run stateful workloads including databases and message queues, but I still prefer not to run them on Kubernetes.

9:04 AM - 13 Feb 2018

308 Retweets 708 Likes

24 308 708

Tweet your reply

Kelsey Hightower @kelseyhightower · Feb 13
Kubernetes can only meet stateful workloads half way and I lack the expertise to manage a production configuration of Kafka, RabbitMQ, or Postgres on static infrastructure, let alone a Kubernetes cluster.

2 18 71

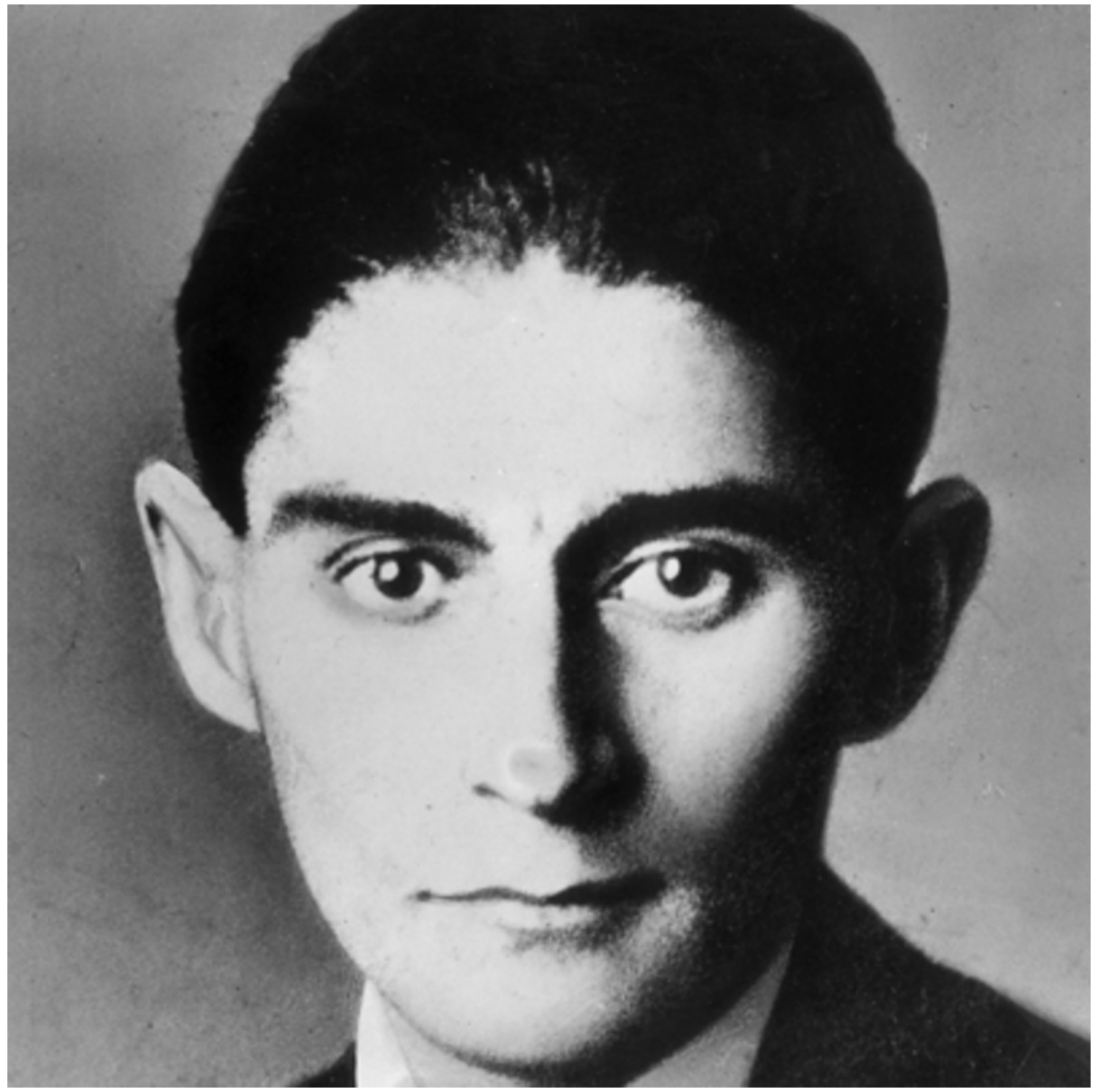
https://twitter.com/kelseyhightower/status/963413508300812295

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

DON'T DESPAIR…

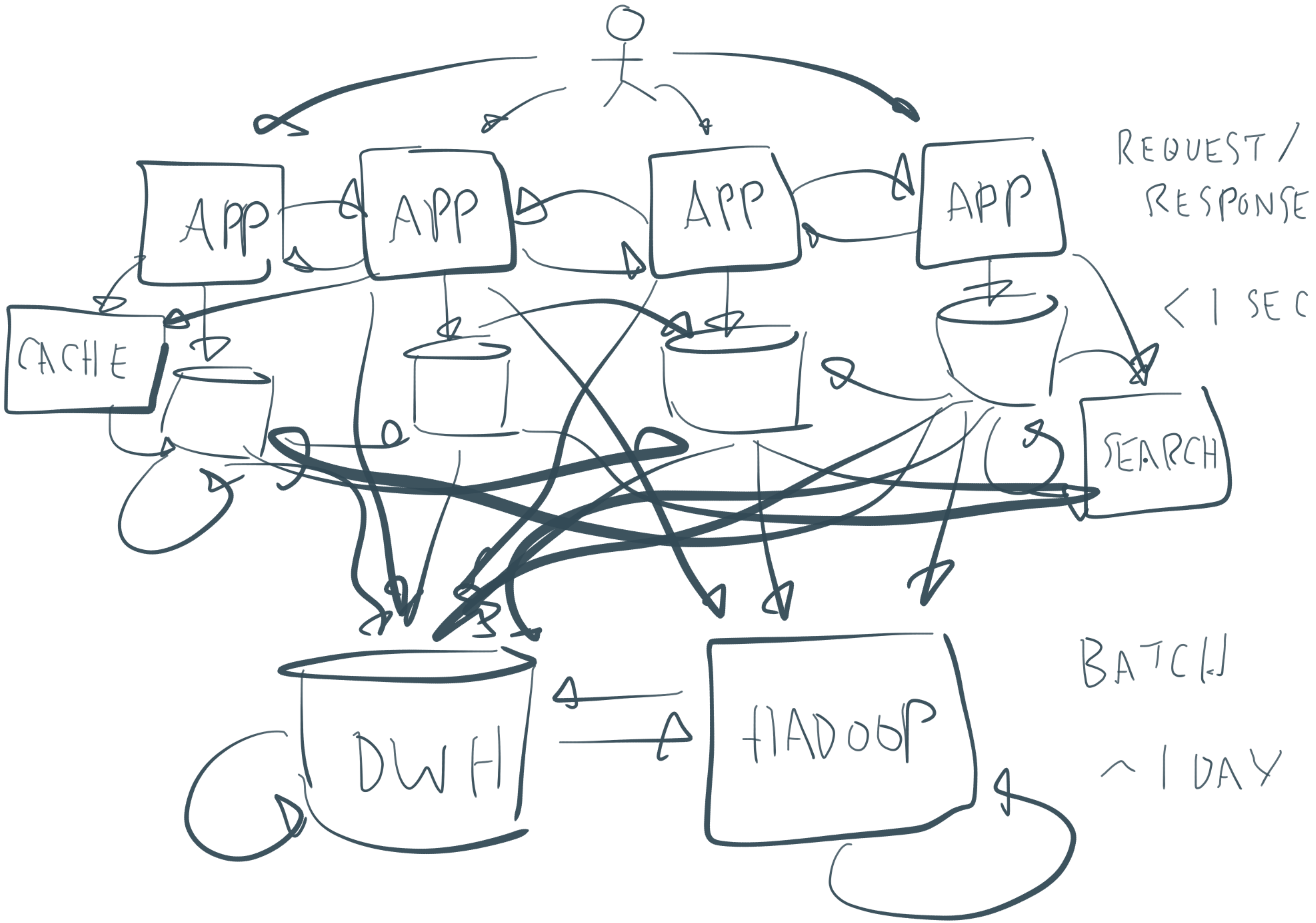
"… not even over the fact that you don't despair.

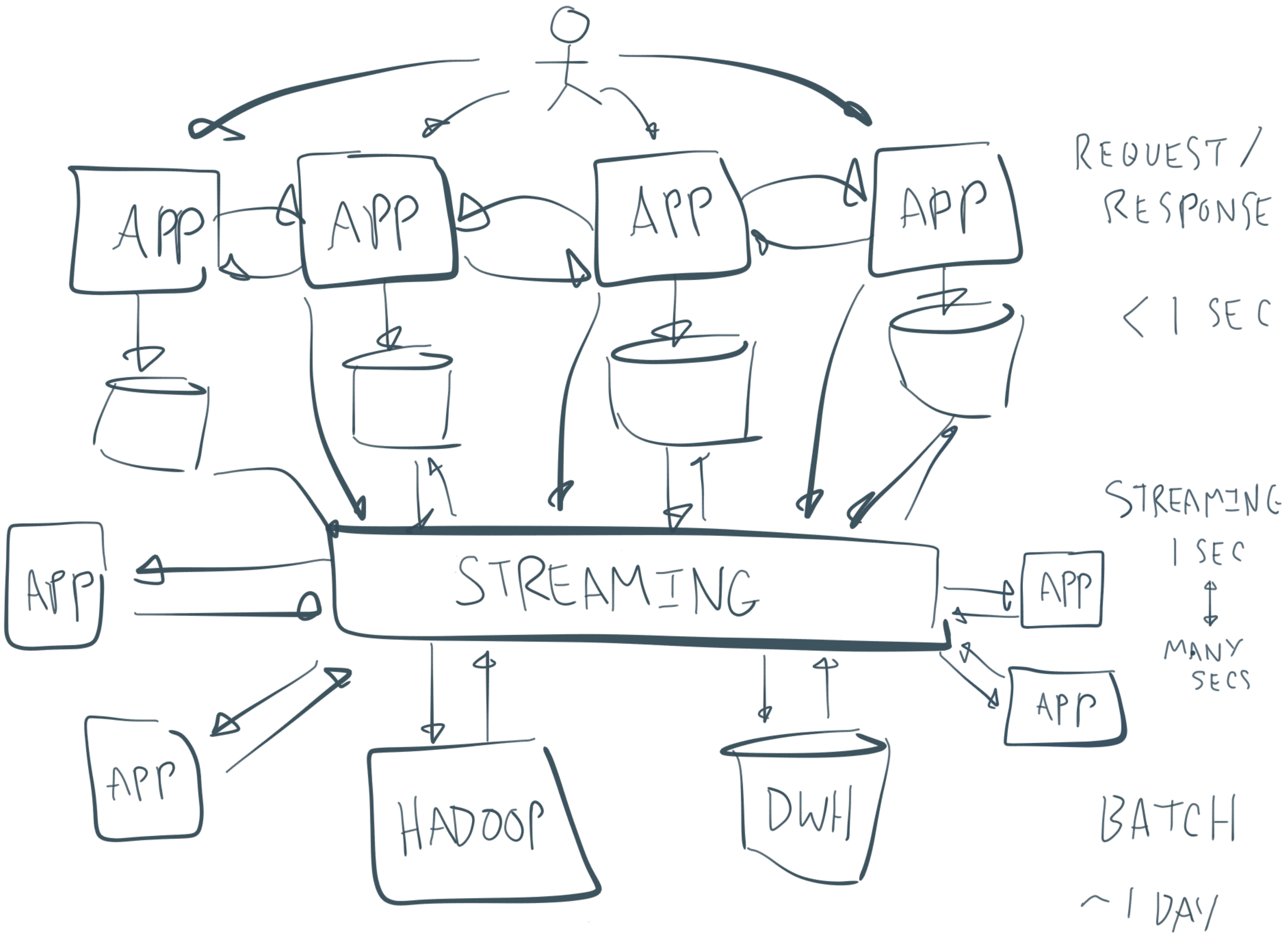
Just when everything seems over with, new forces come marching up, and precisely that means that you are alive"



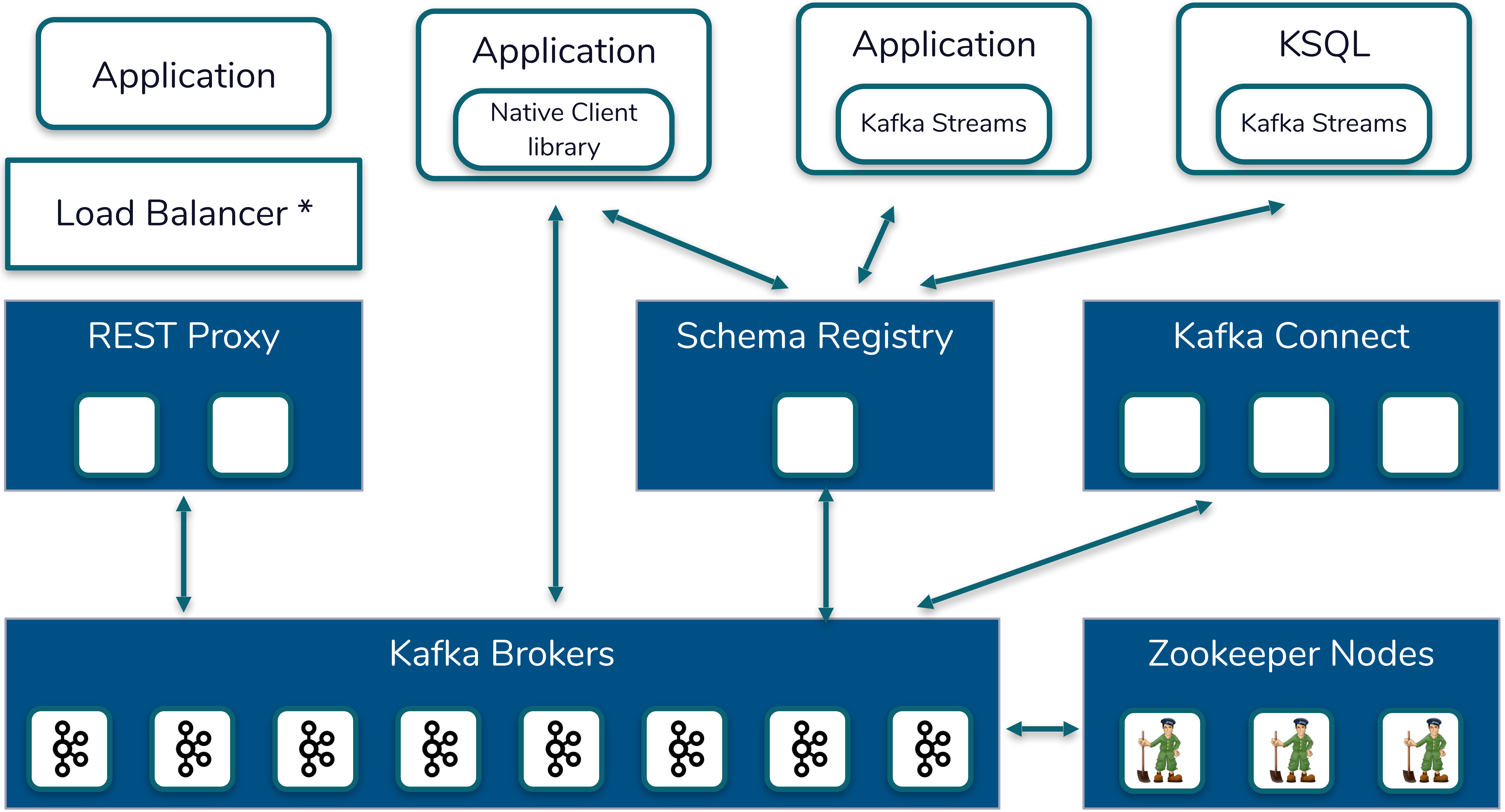
Franz Kafka

KAFKA STREAMING ARCHITECTURE FUNDAMENTALS

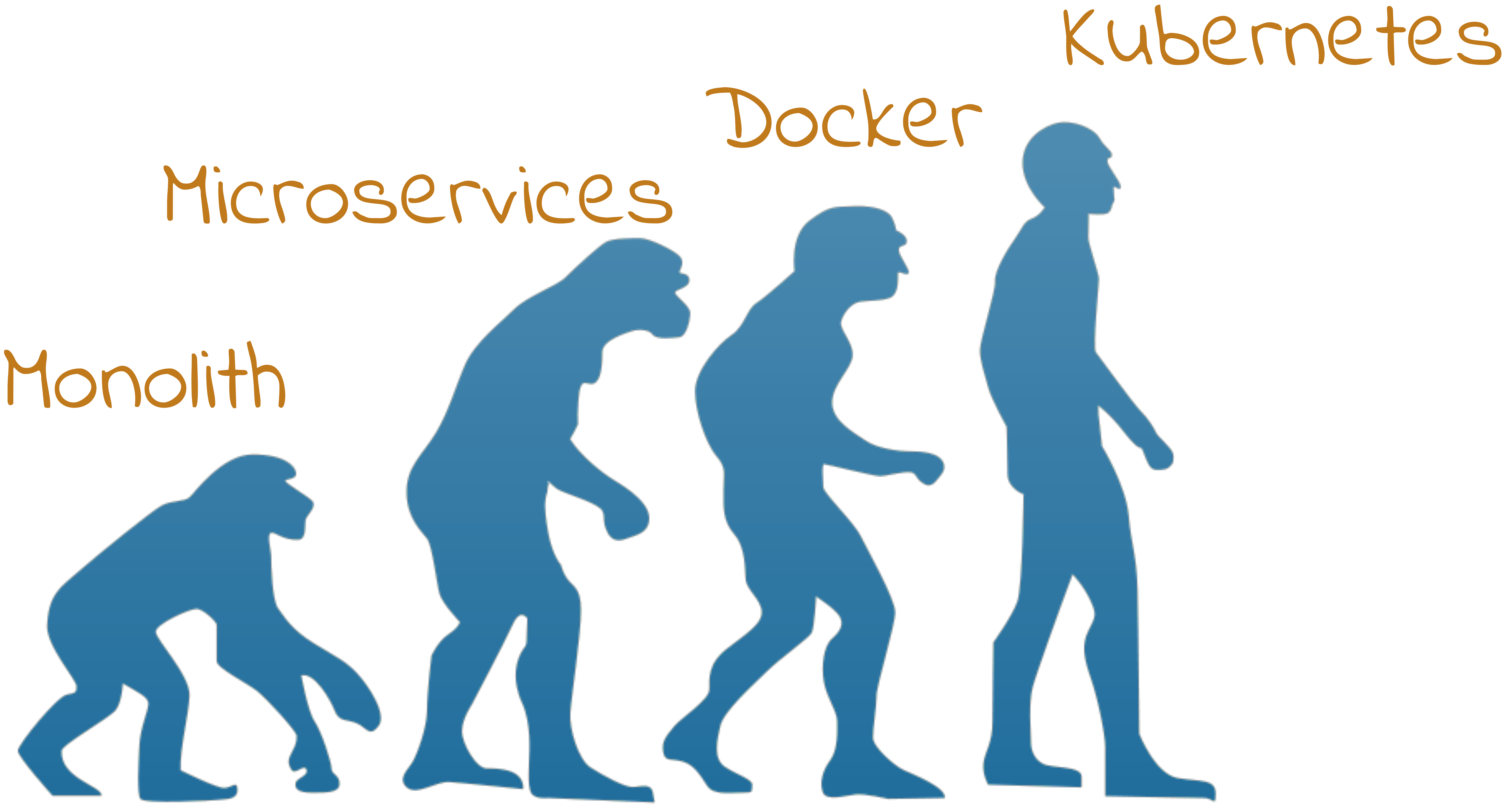

APP
APP
APP
APP
REQUEST / RESPONSE
< 1 SEC
CACHE
SEARCH
DWH
HADOOP
BATCH
~ 1 DAY


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

EVENT STREAMING PLATFORM ARCHITECTURE



KUBERNETES FUNDAMENTALS


Monolith
Microservices
Docker
Kubernetes

Syed Aqueel Haider
@sahrizv

Follow

2014 - We must adopt #microservices to solve all problems with monoliths
2016 - We must adopt #docker to solve all problems with microservices
2018 - We must adopt #kubernetes to solve all problems with docker

1:25 PM - 14 Jul 2018

1,622 Retweets **2,263** Likes

https://twitter.com/sahrizv/status/1018184792611827712

ORCHESTRATION

- Compute
- Networking
- Storage
- Service Discovery

KUBERNETES

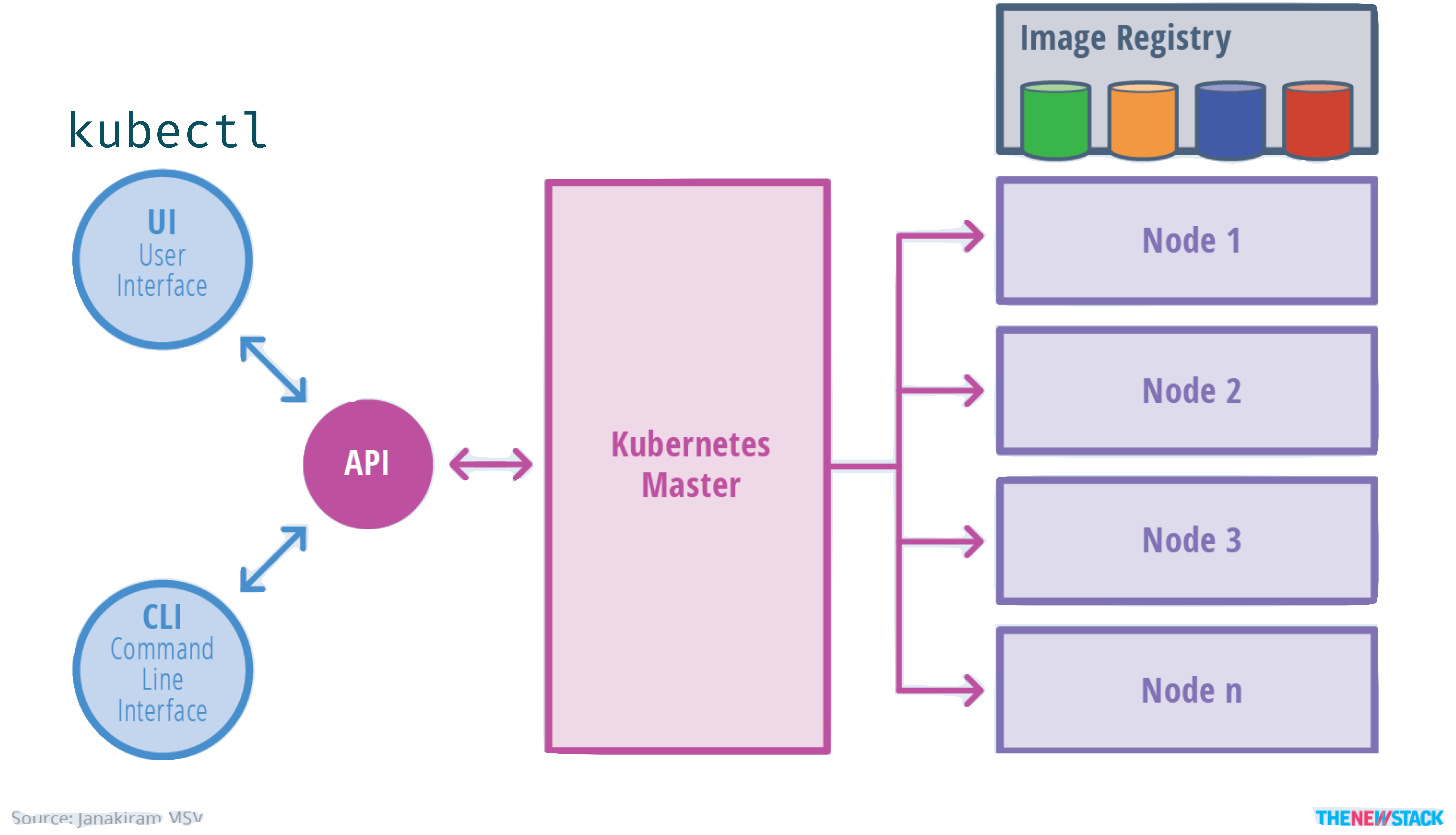
Schedules and allocates resources

Networking between Pods

Storage

Service Discovery

REFRESHER - KUBERNETES ARCHITECTURE



https://thenewstack.io/kubernetes-an-overview/

POD

Basic Unit of Deployment
in Kubernetes

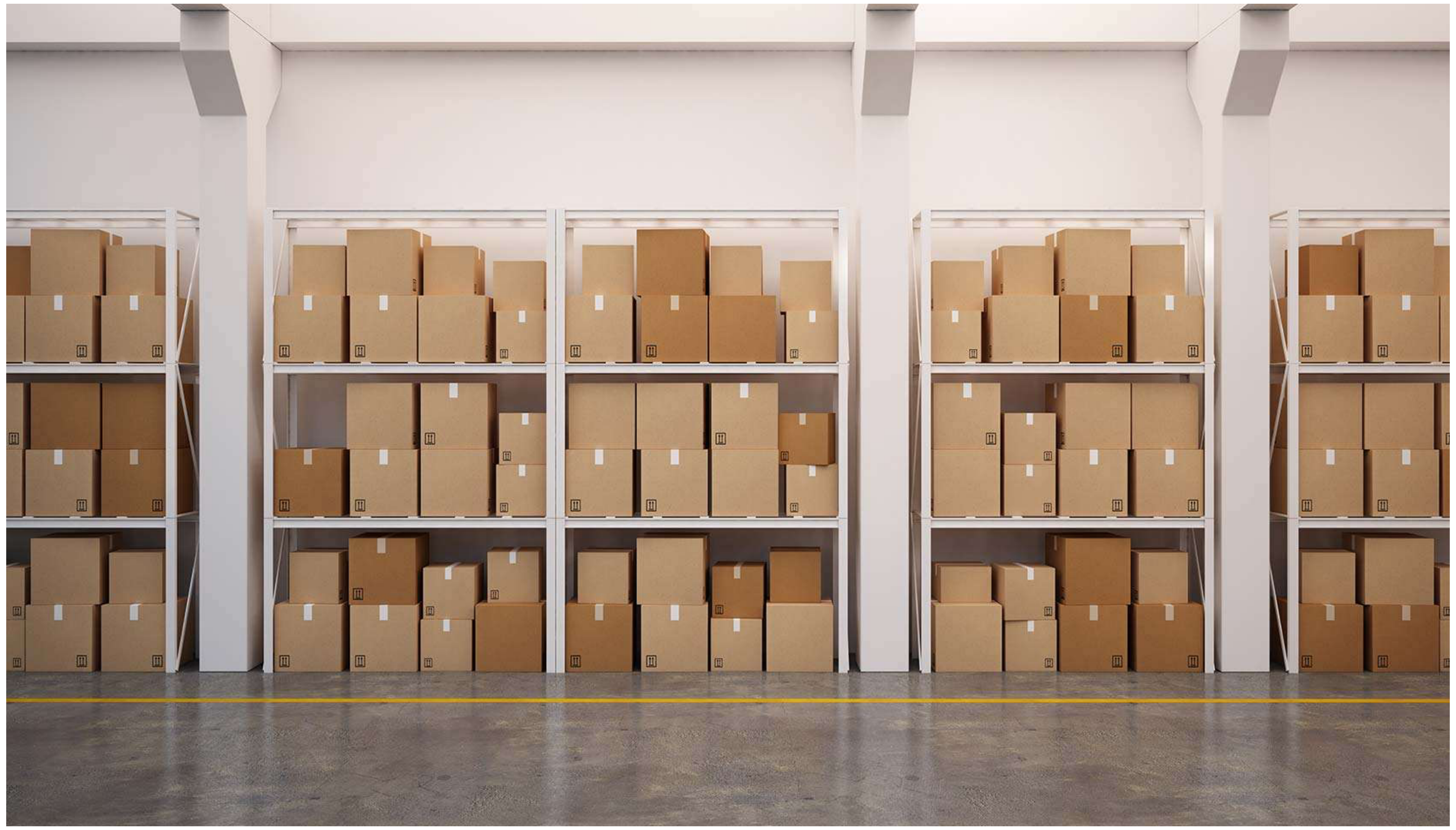
A collection of containers sharing:
- Namespace
- Network
- Volumes

STORAGE

Persistent Volume (PV) &

Persistent Volume Claim (PVC)

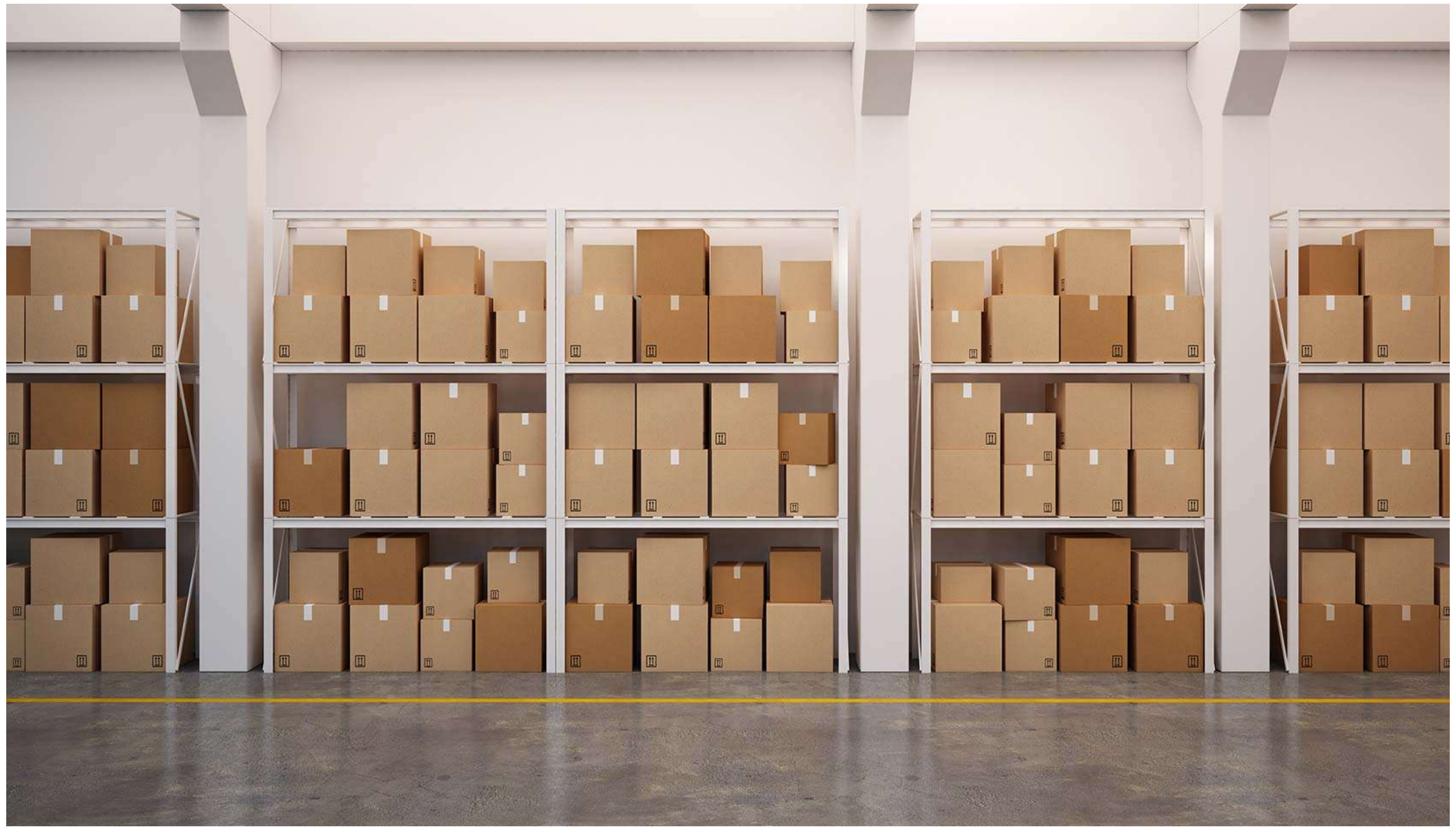
Both PV and PVC are 'resources'



STORAGE

Persistent Volume (PV) &

- Persistent Volume Claim (PVC)
  - PV is a piece of storage that is provisioned dynamic or static of any individual pod that uses the PV

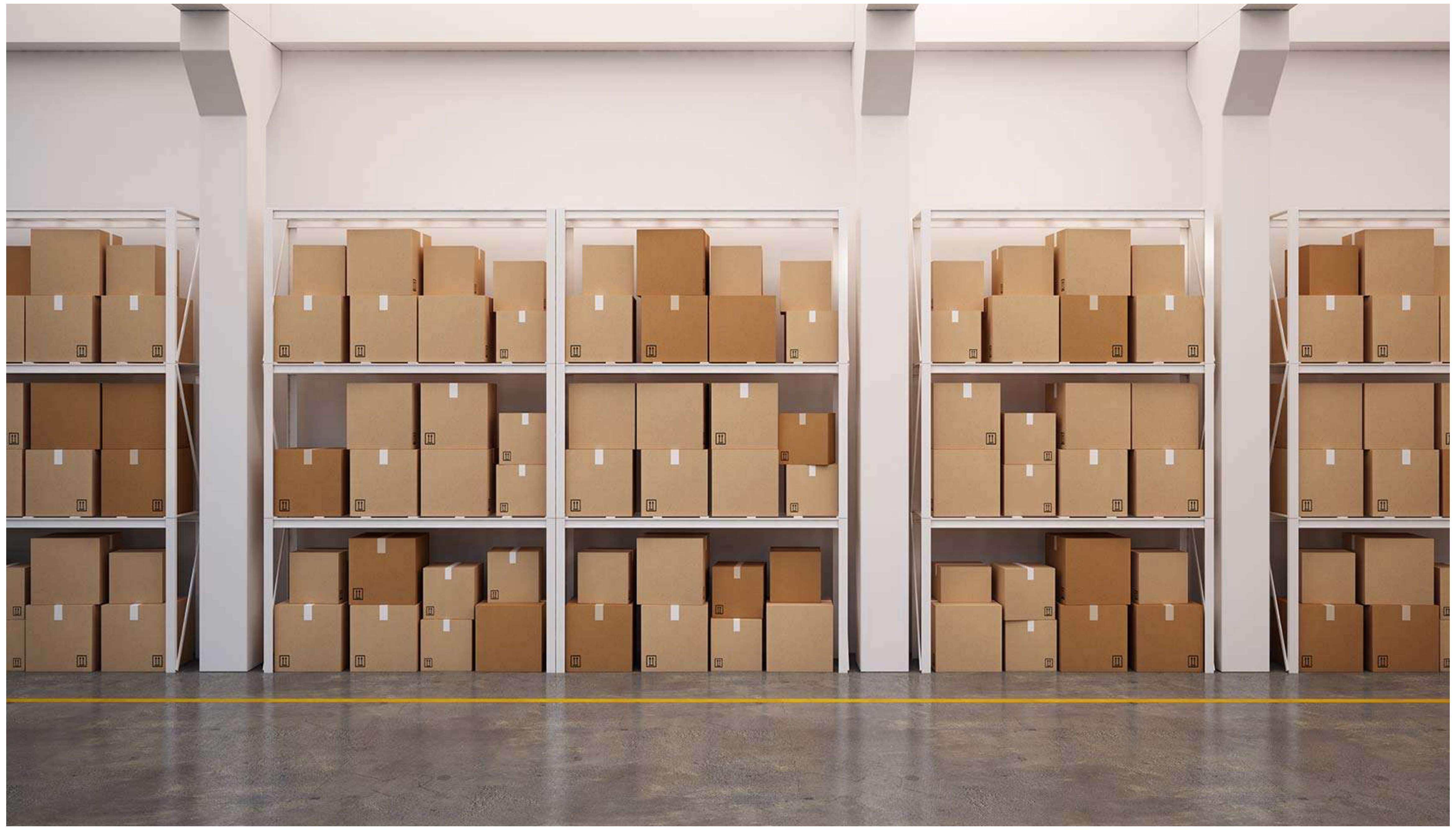


STORAGE

Persistent Volume (PV) &

Persistent Volume Claim (PVC)

PVC is a request for storage by a User

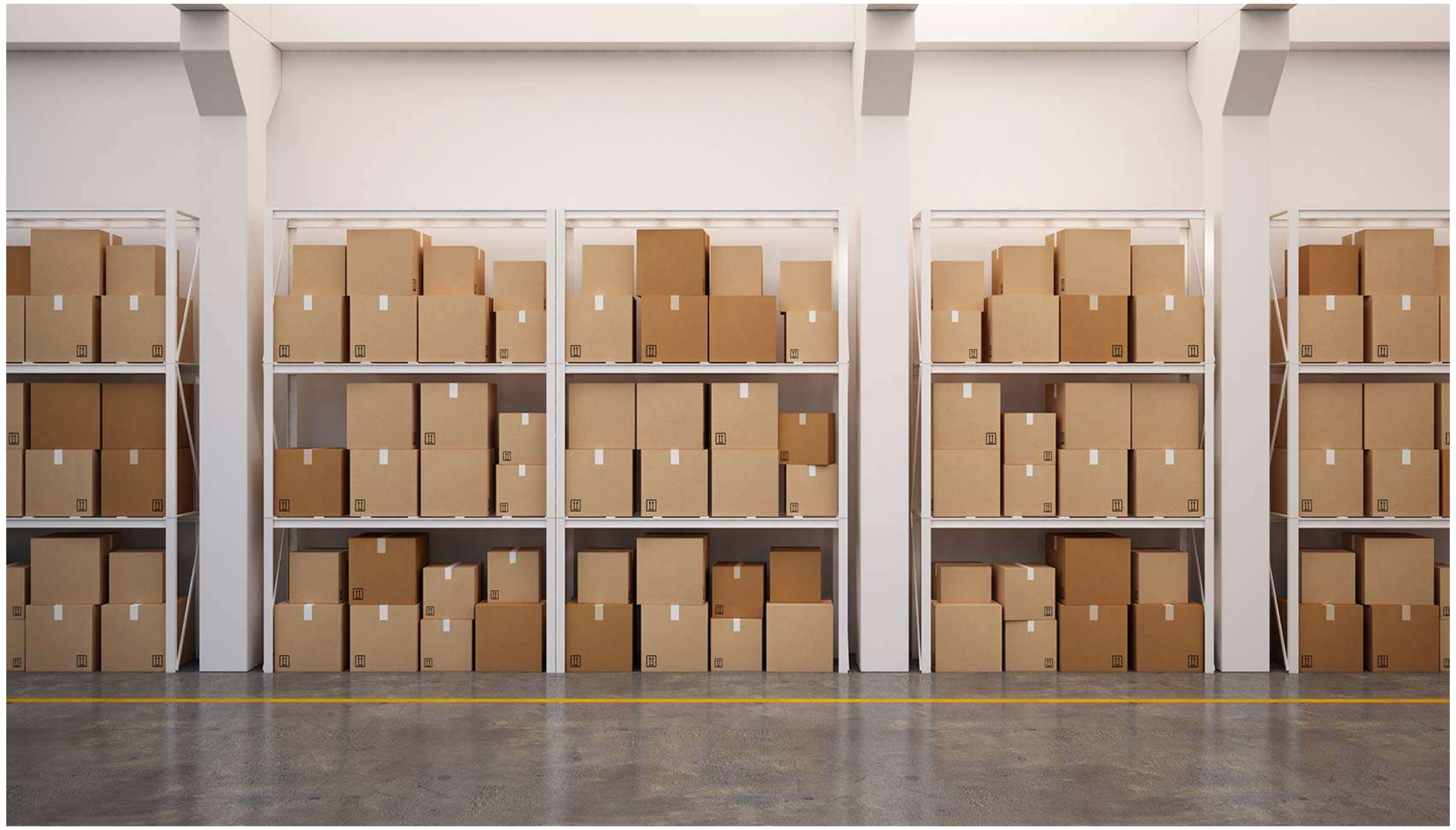


STORAGE

Persistent Volume (PV) &

Persistent Volume Claim (PVC)

PVCs consume PV

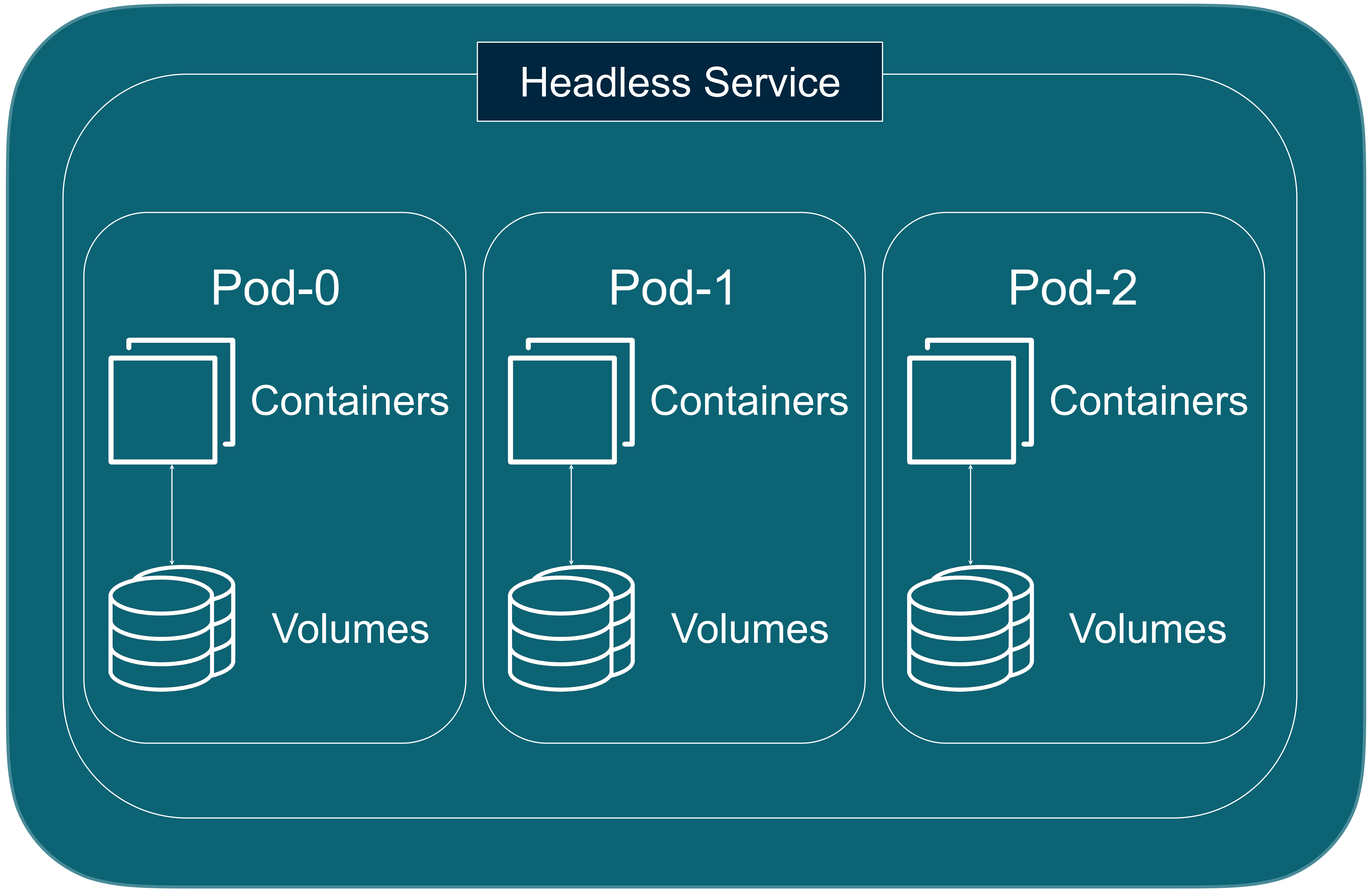


STATEFUL WORKLOADS

STATEFULSET

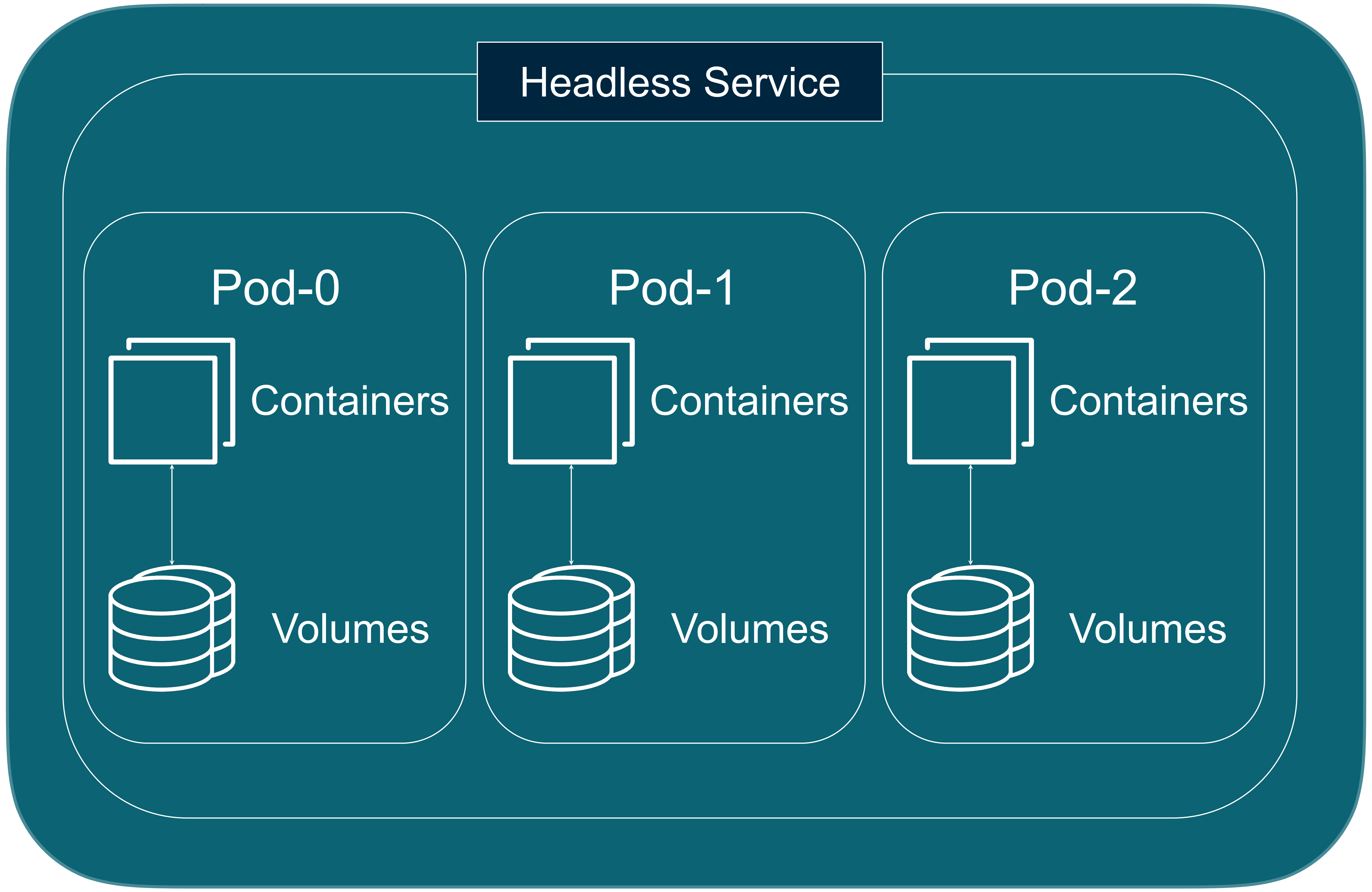
Rely on Headless Service to provide network identity

Ideal for highly available stateful workloads



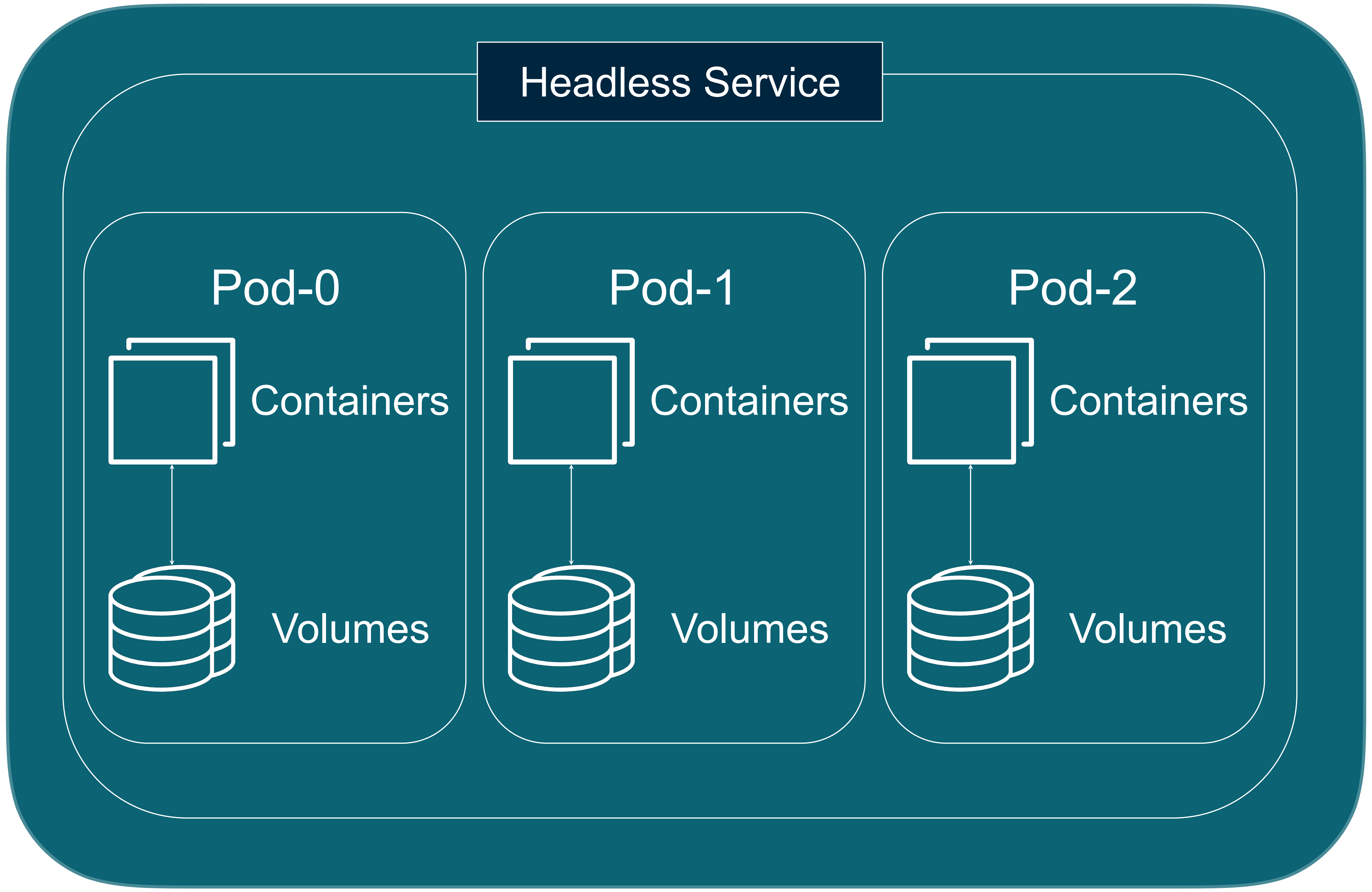
STATEFULSET

Rely on **Headless Service** to provide network identity



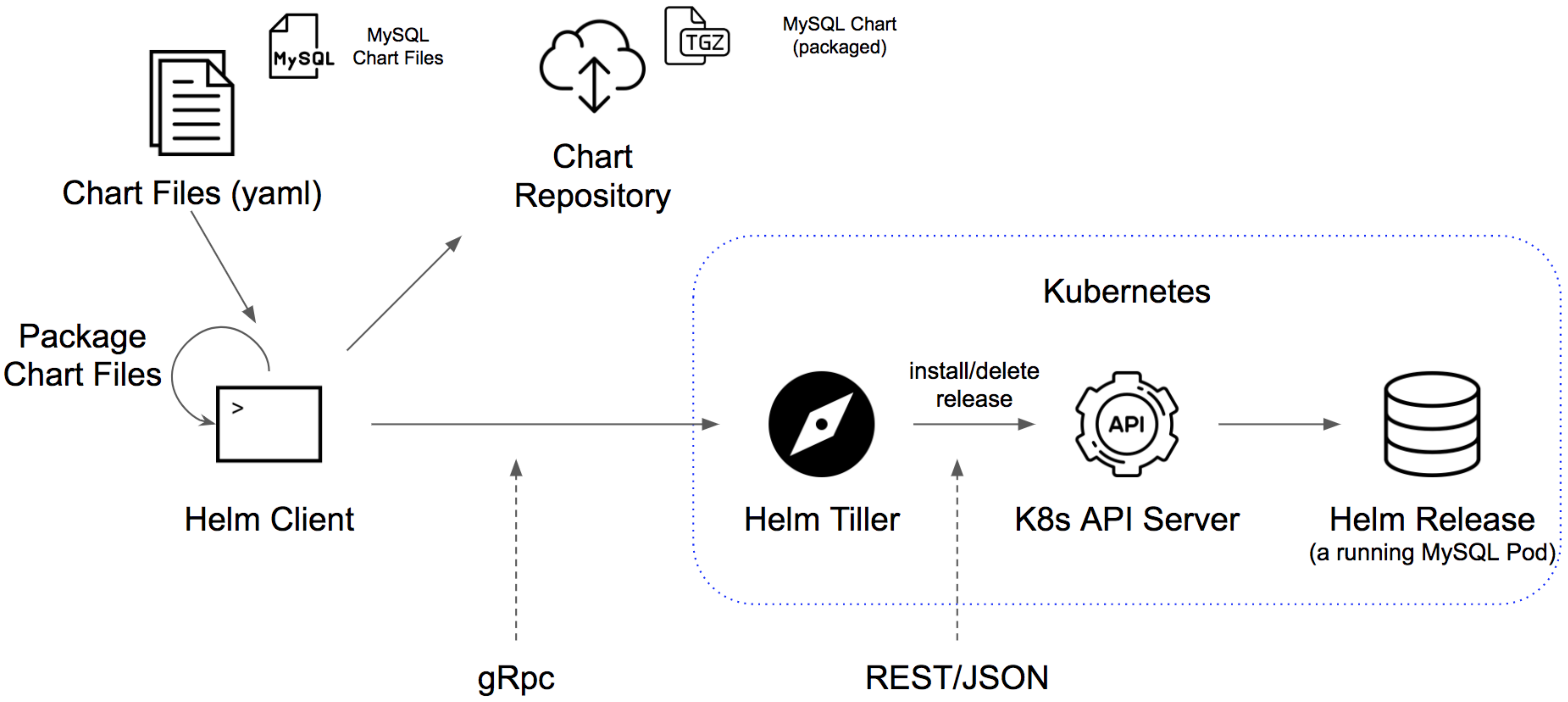
STATEFULSET

Ideal for highly available **stateful workloads**

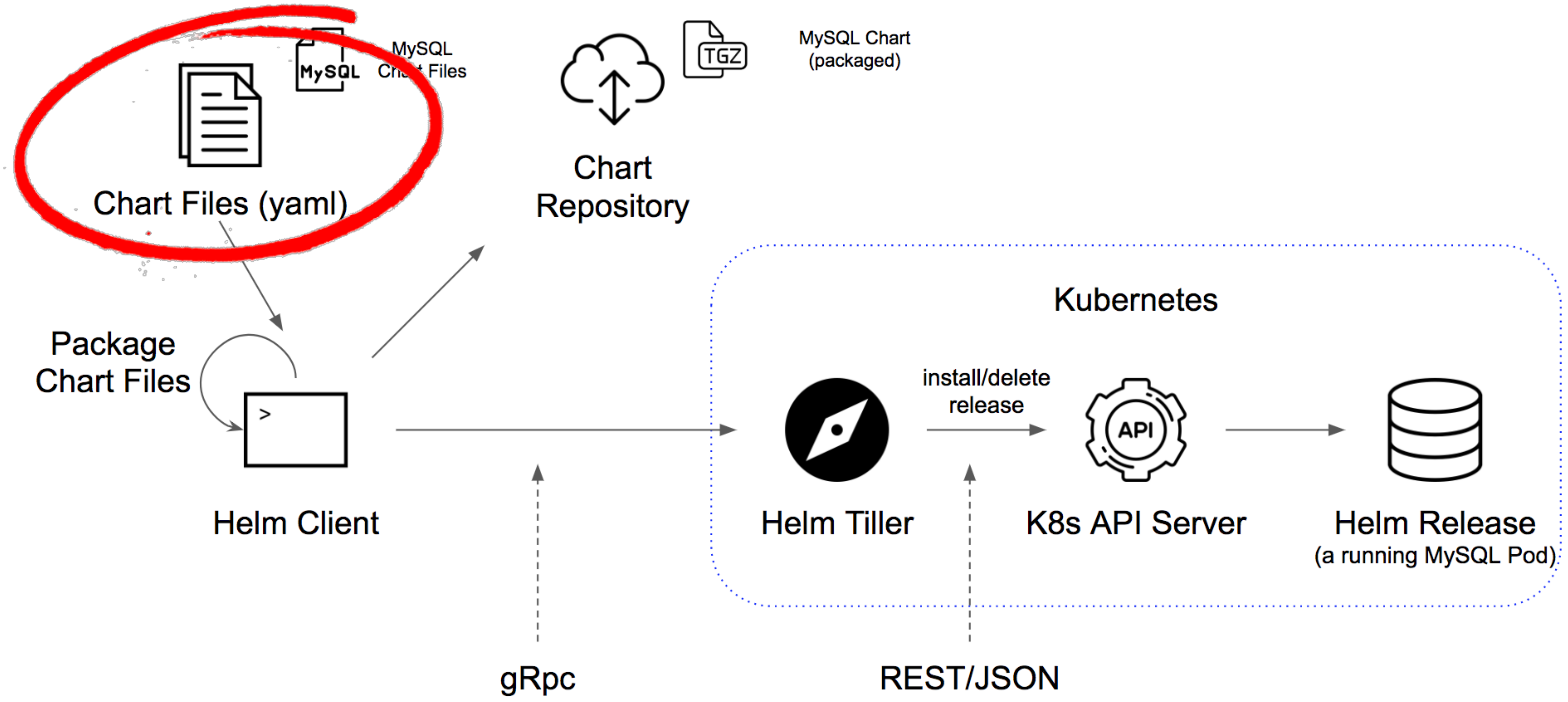


WORKLOADS DEPLOYMENT

HELM CHARTS

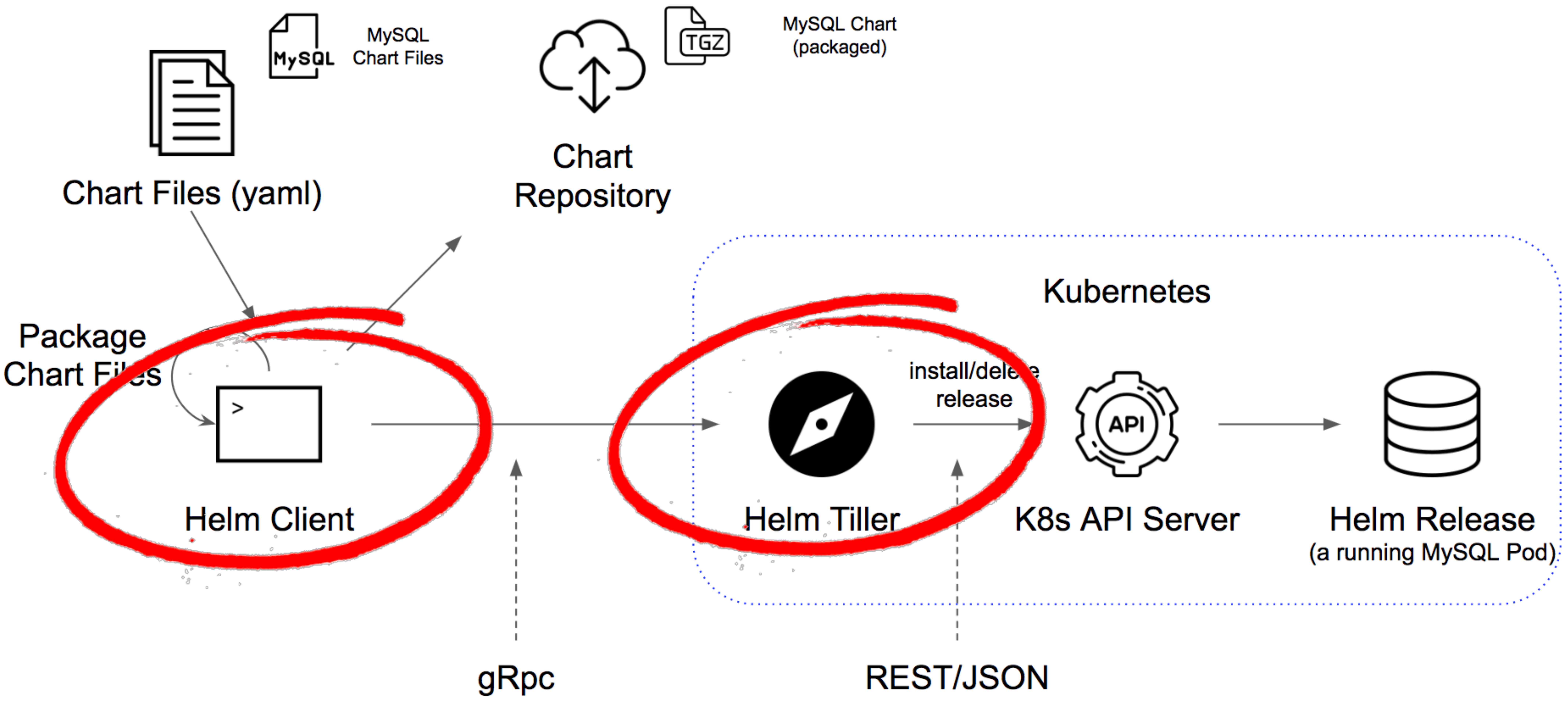

MySQL
MySQL
Chart Files
TGZ
MySQL Chart
(packaged)
Chart Files (yaml)
Chart
Repository
Package
Chart Files
Helm Client
Kubernetes
install/delete
release
API
Helm Tiller
K8s API Server
Helm Release
(a running MySQL Pod)
gRpc
REST/JSON

HELM CHARTS

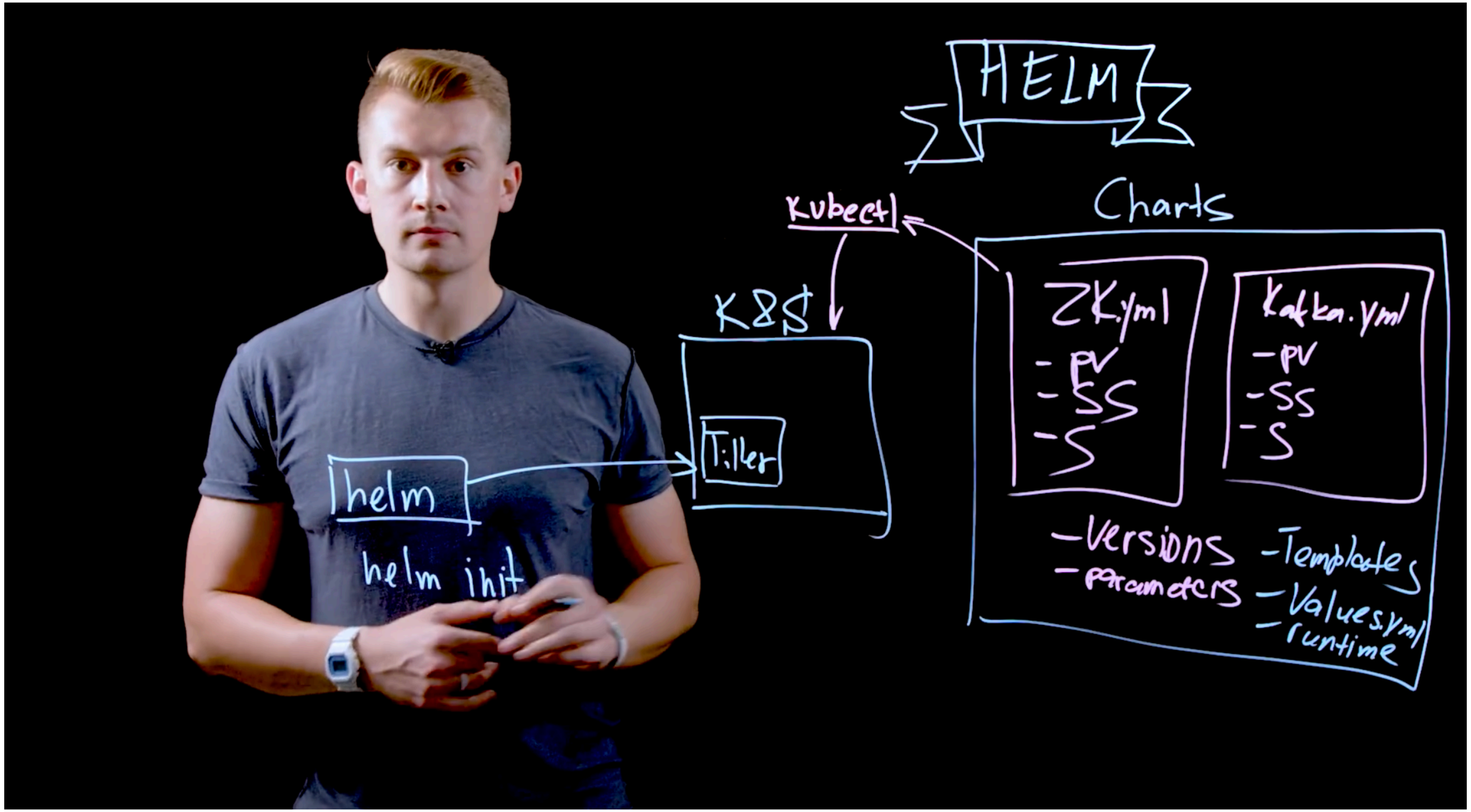


MySQL
MySQL Chart Files
Chart Files (yaml)
Chart Repository
TGZ
MySQL Chart (packaged)
Package Chart Files
Helm Client
Kubernetes
install/delete release
API
Helm Tiller
K8s API Server
Helm Release
(a running MySQL Pod)
gRpc
REST/JSON

HELM CHARTS



MySQL
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

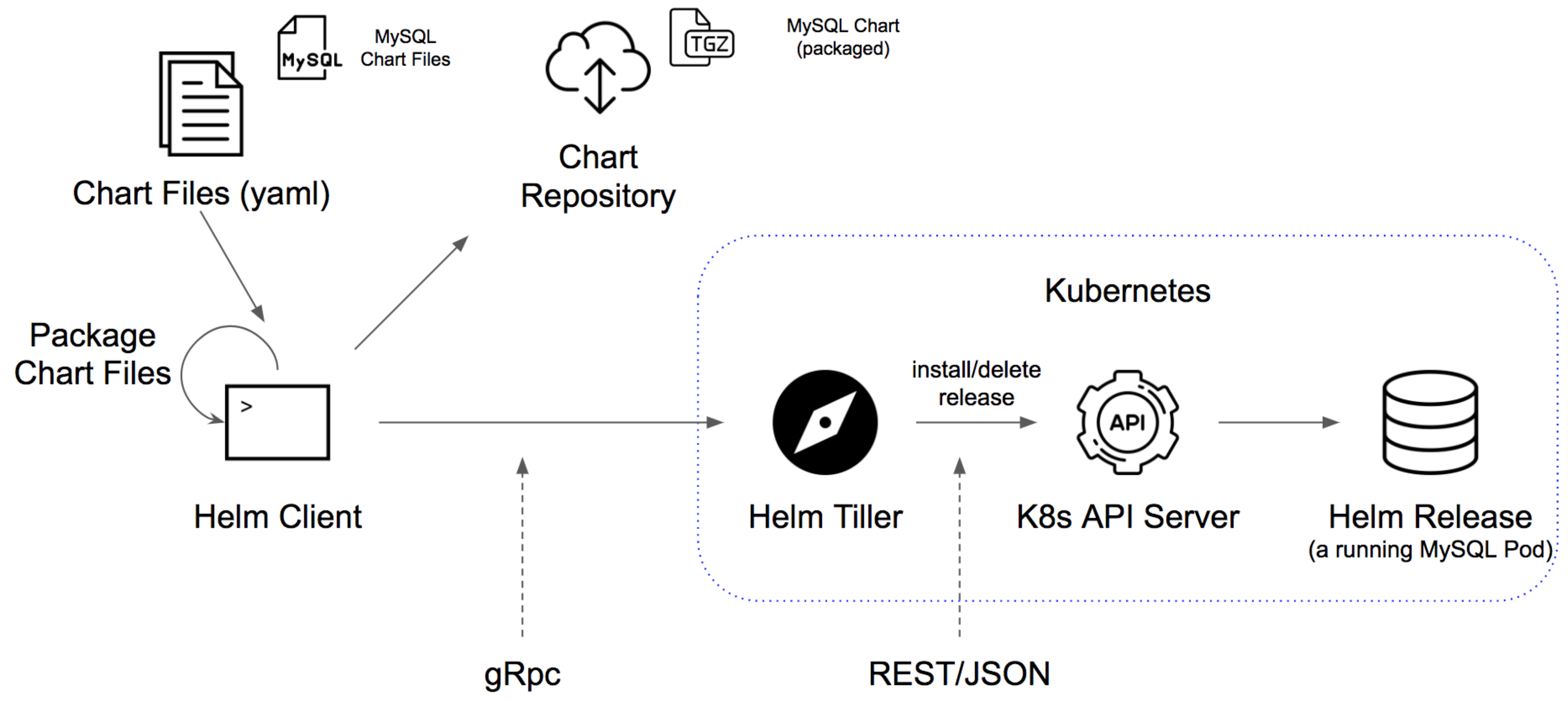


https://cnfl.io/helm_video

HELM CHARTS

Package Manager

Package multiple K8s resources into one deployment unit: **Chart**



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

kafka

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

BASIC COMPONENTS ARE NOT ENOUGH

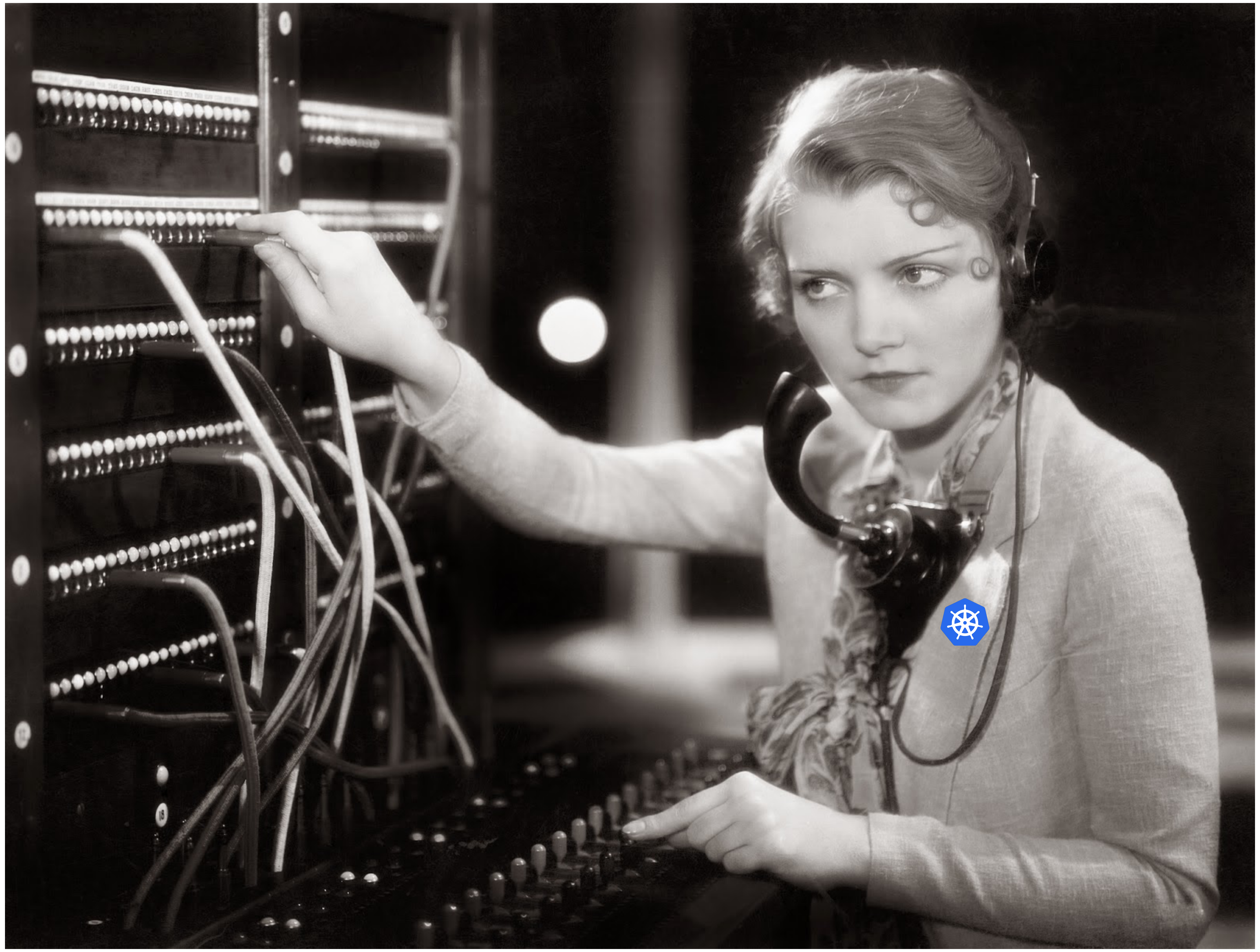
MEET KUBERNETES OPERATOR

confluent

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.

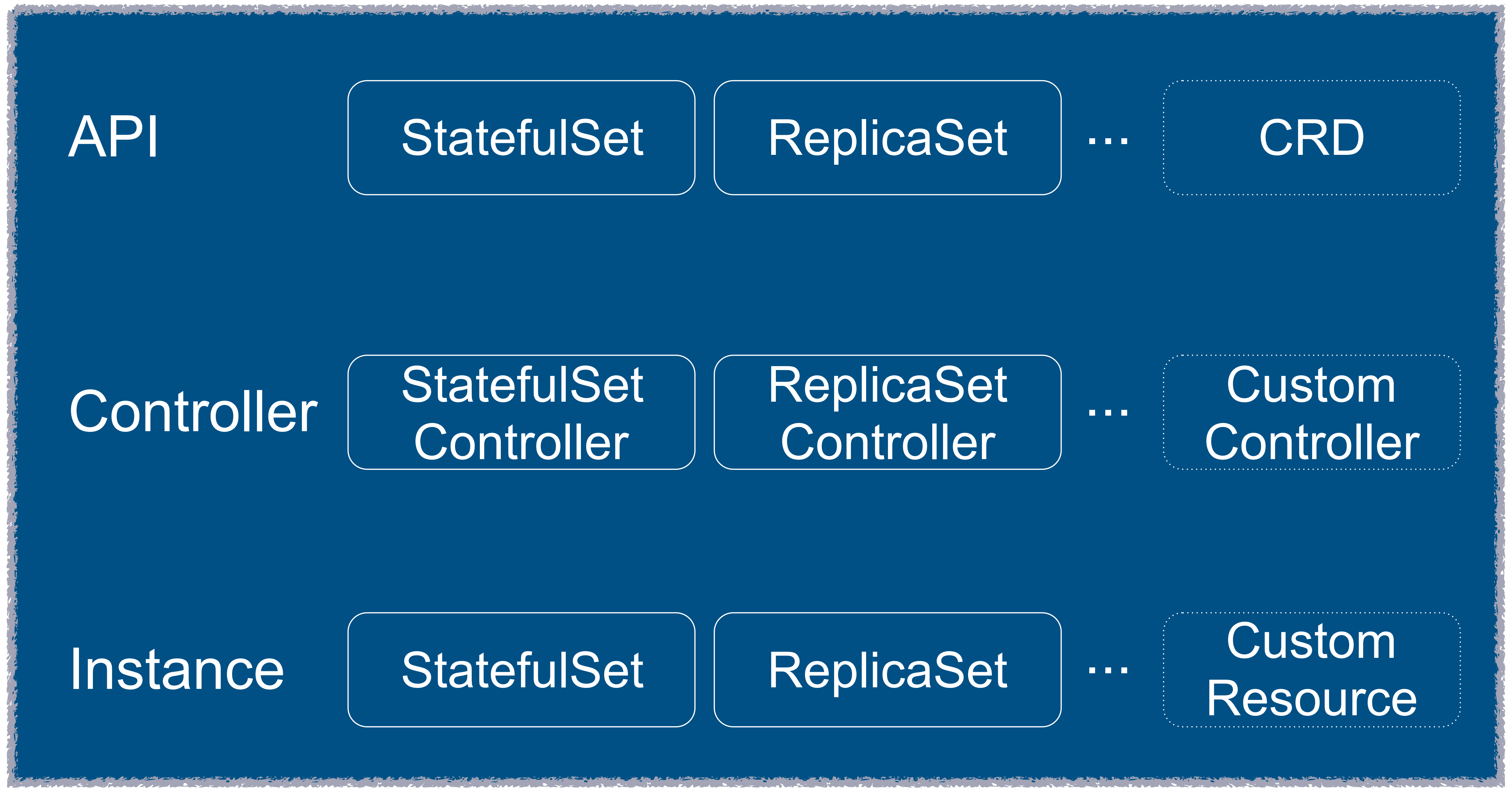
CUSTOM RESOURCE DEFINITION(CRD)

Extend existing Kubernetes API

API	StatefulSet	ReplicaSet	…	CRD
Controller	StatefulSet Controller	ReplicaSet Controller	…	Custom Controller
Instance	StatefulSet	ReplicaSet	…	Custom Resource

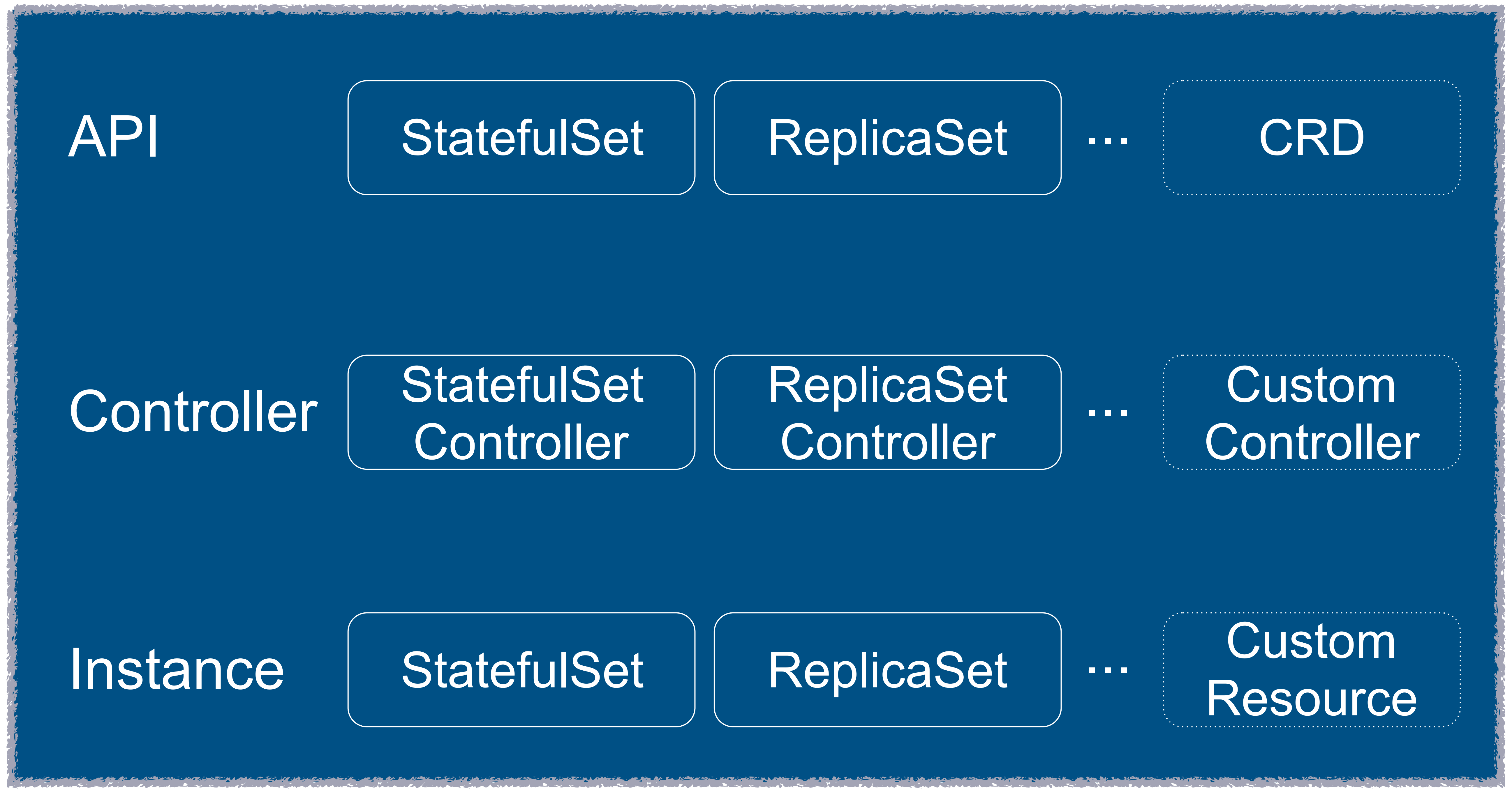
CUSTOM RESOURCE DEFINITION(CRD)

Usually works together **Custom Controller**



CUSTOM RESOURCE DEFINITION(CRD)

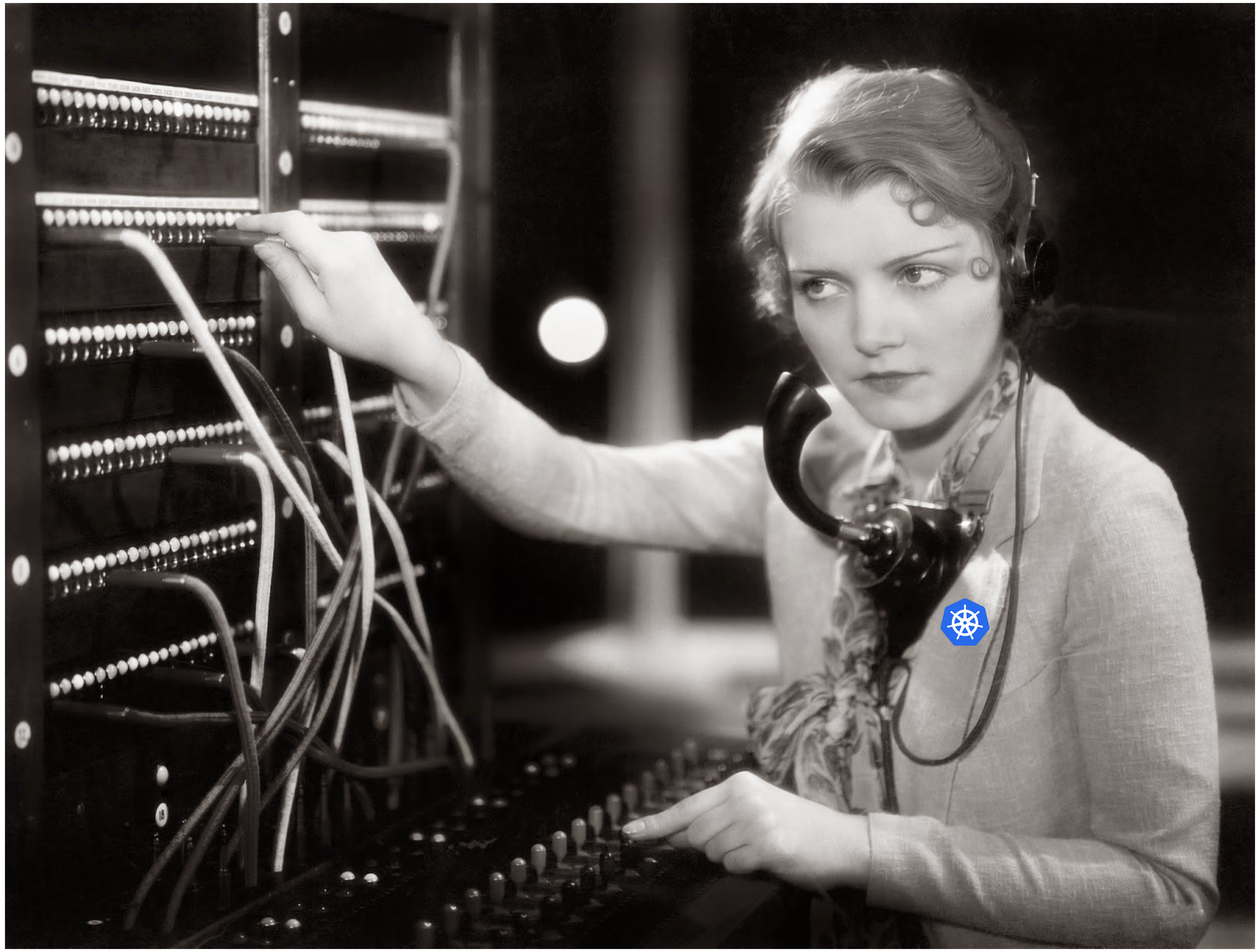
Users can create and access Customer Resources with ***kubectl***, just as they do for built-in resources like **pods**.



OPERATOR

Deploy and Manage your production streaming platform with Confluent Operator.

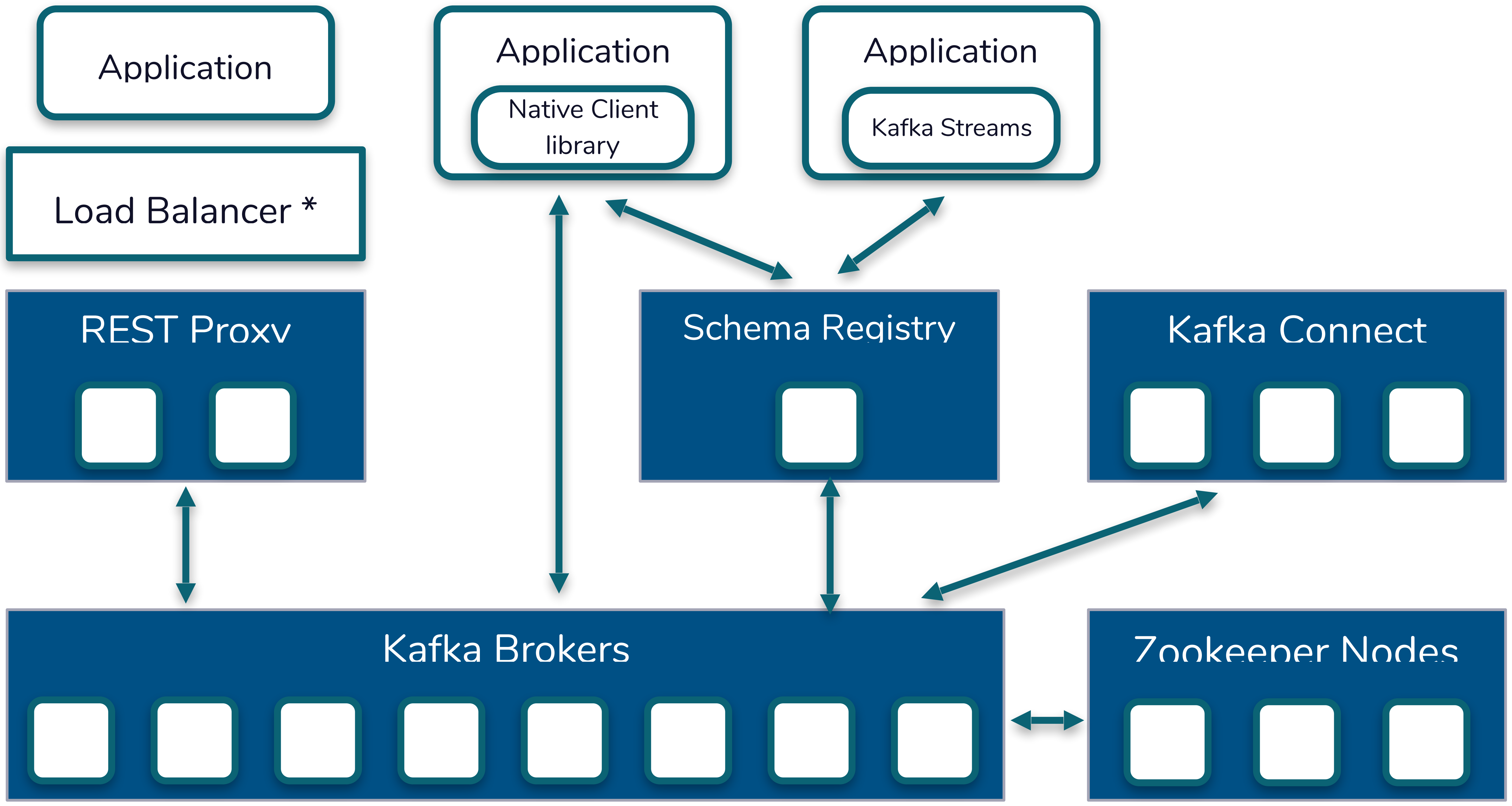
- Automated Provisioning
- Platform Operations
- Resiliency
- Monitoring



CONFLUENT PLATFORM REFERENCE ARCHITECTURE

Each Confluent Platform component has specific characteristics:

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

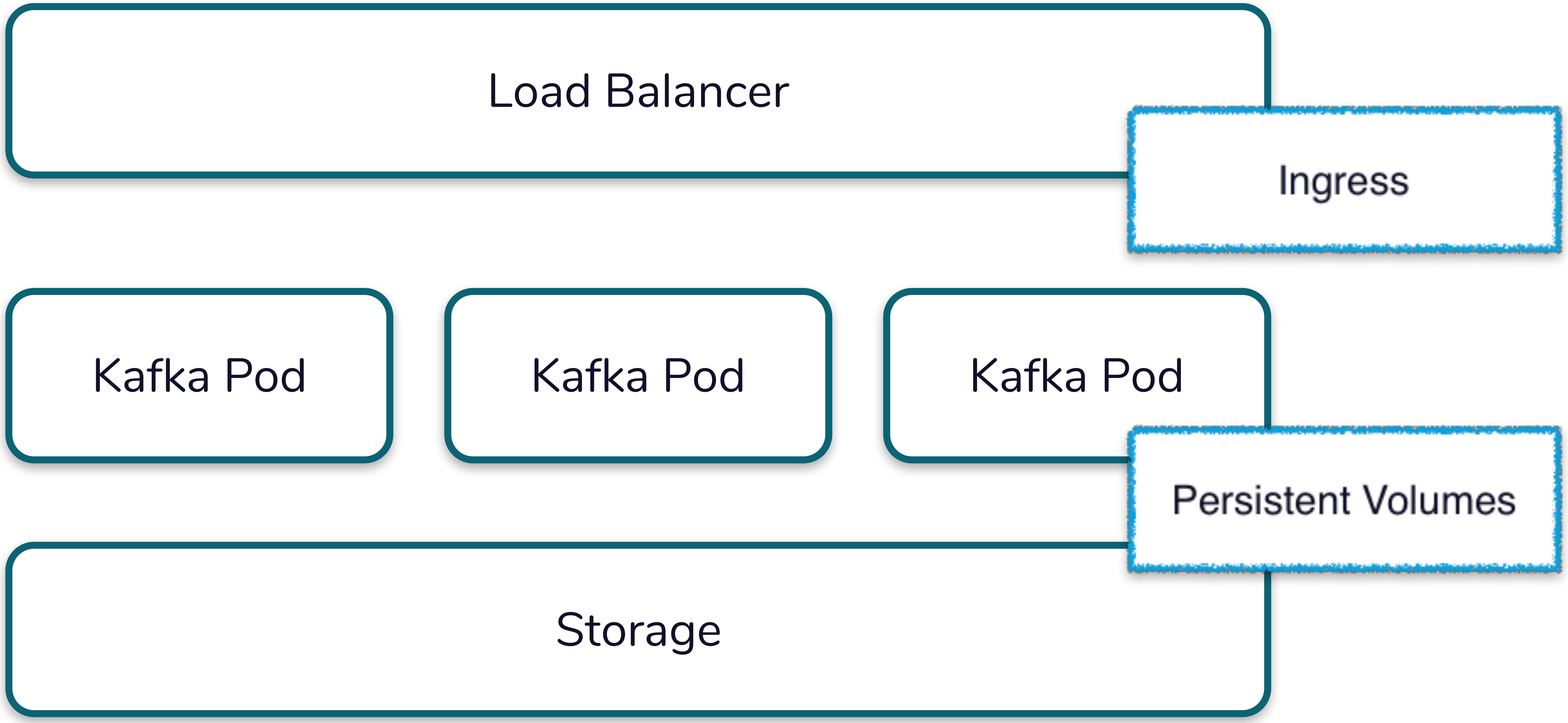


CONFLUENT OPERATOR: AUTOMATED PROVISIONING

```
> 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

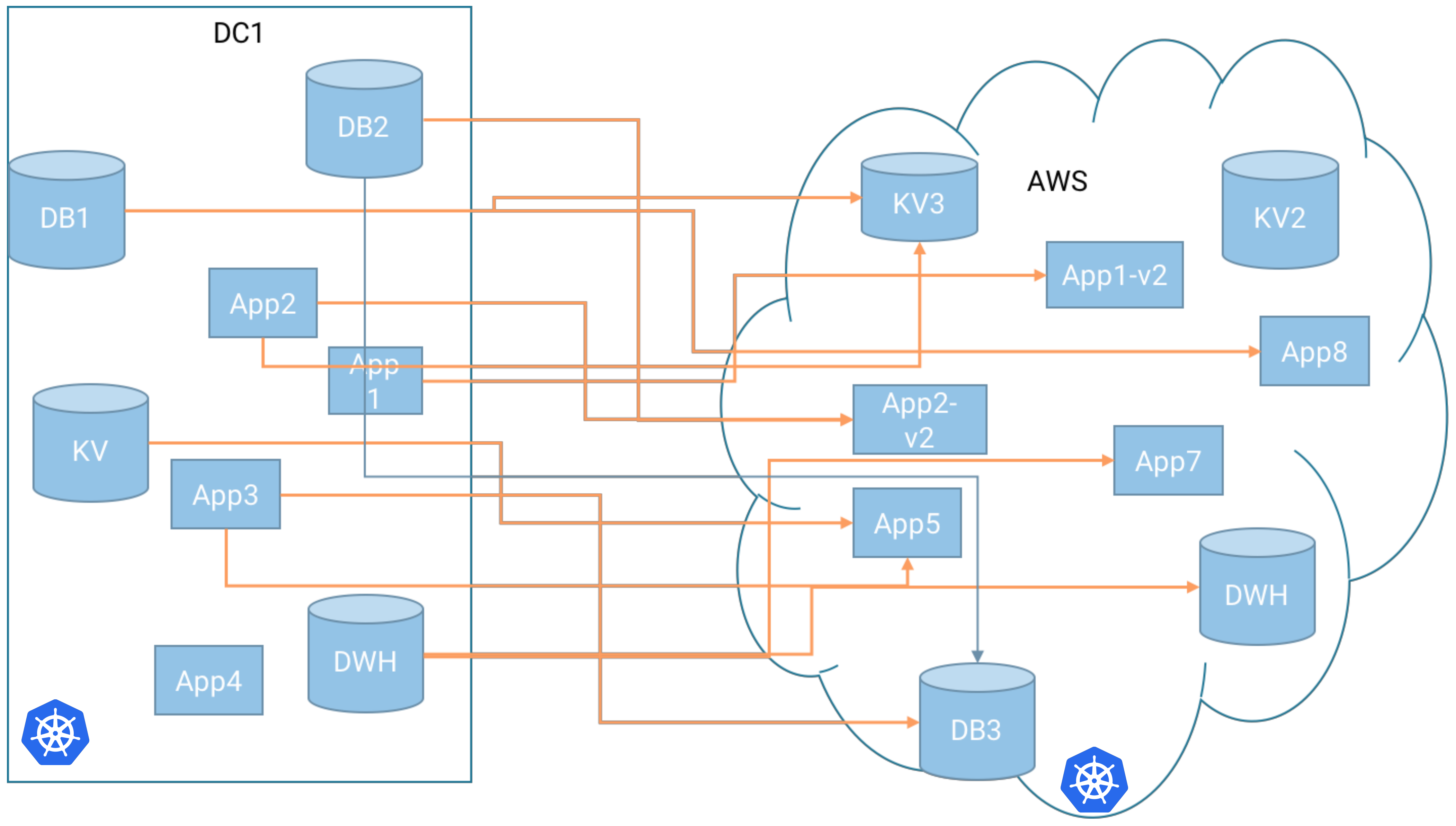
Automate provisioning

Scale your Kafkas and CP clusters elastically

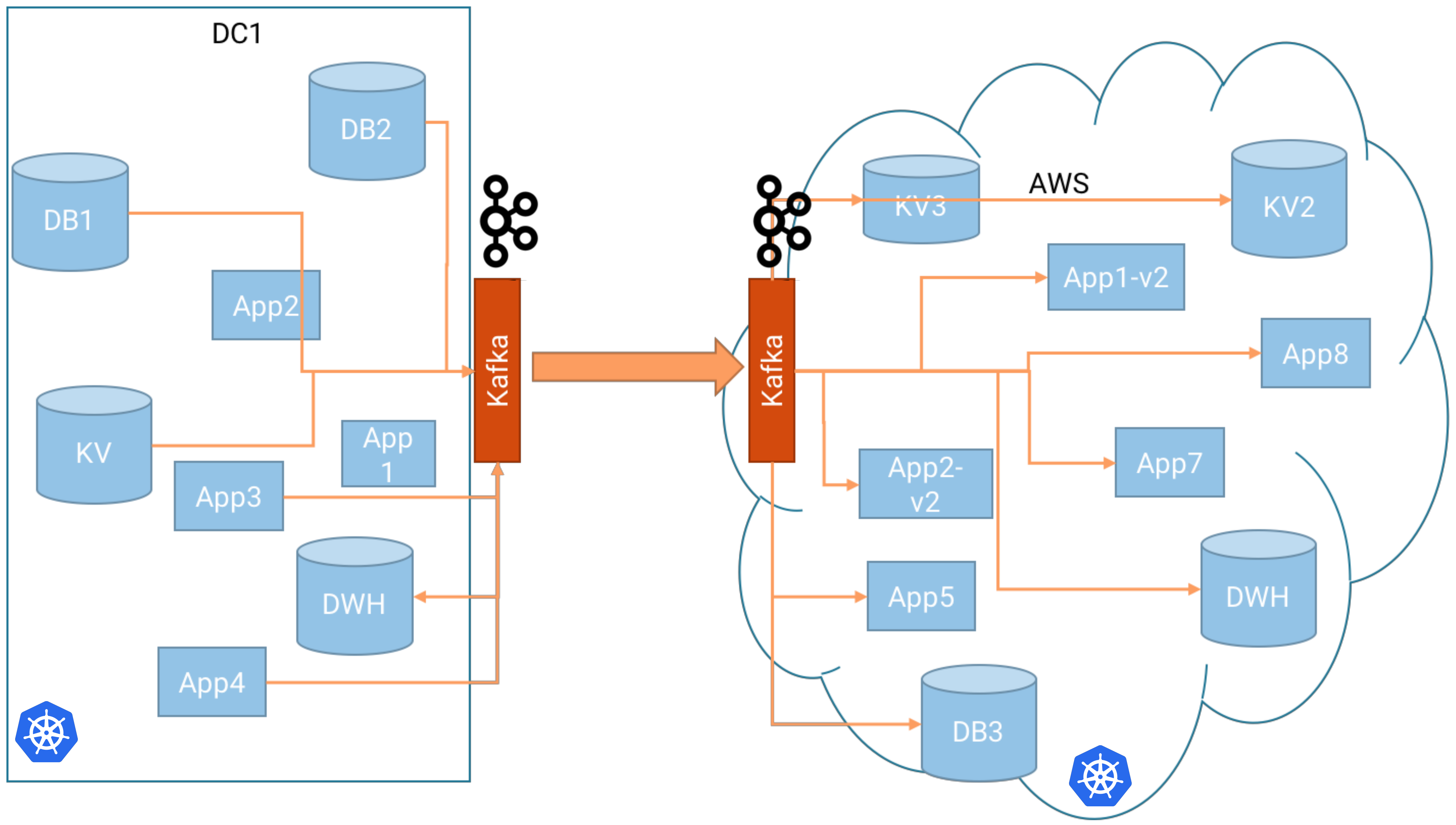
Monitor SLAs through Confluent Control Center or Prometheus

Operate at scale with enterprise support from Confluent

ADVANCED USE CASES

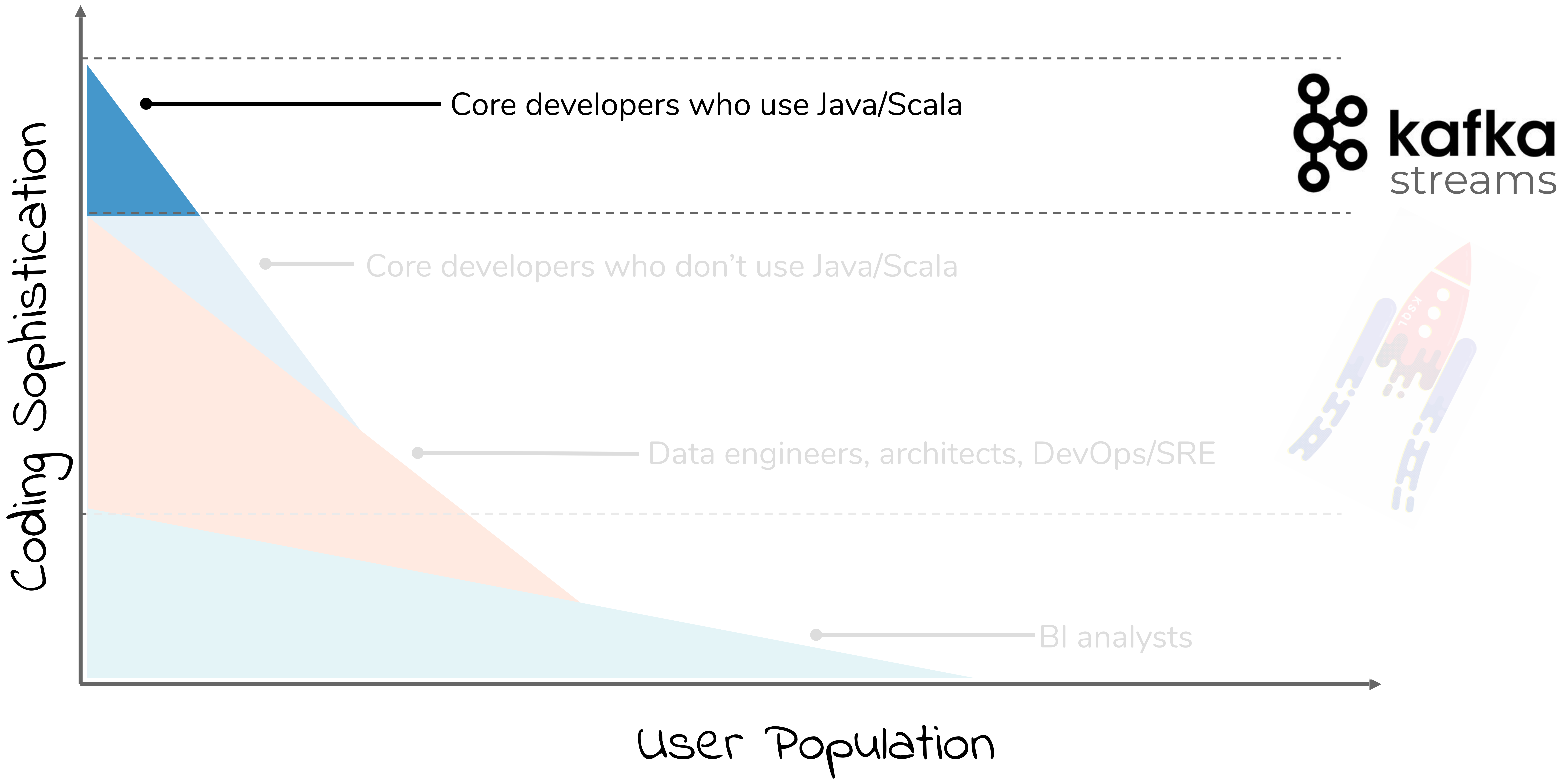


vs.

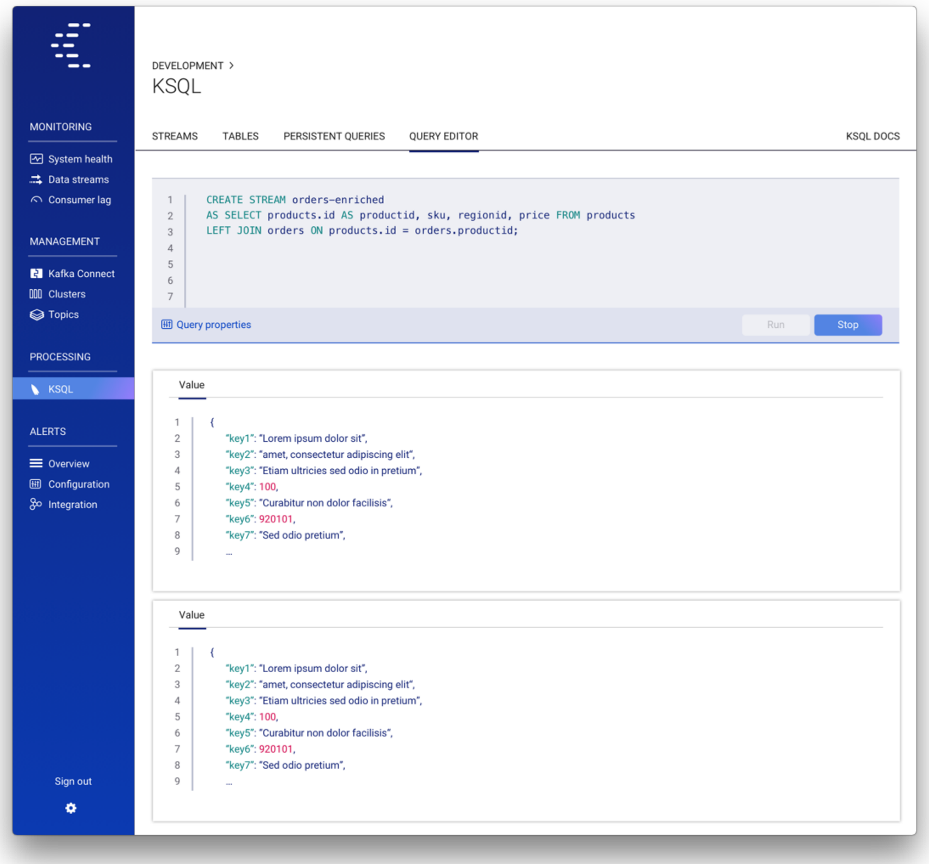


DON'T DESPAIR!

LOWER THE BAR TO ENTER THE WORLD OF STREAMING



KSQL #FTW



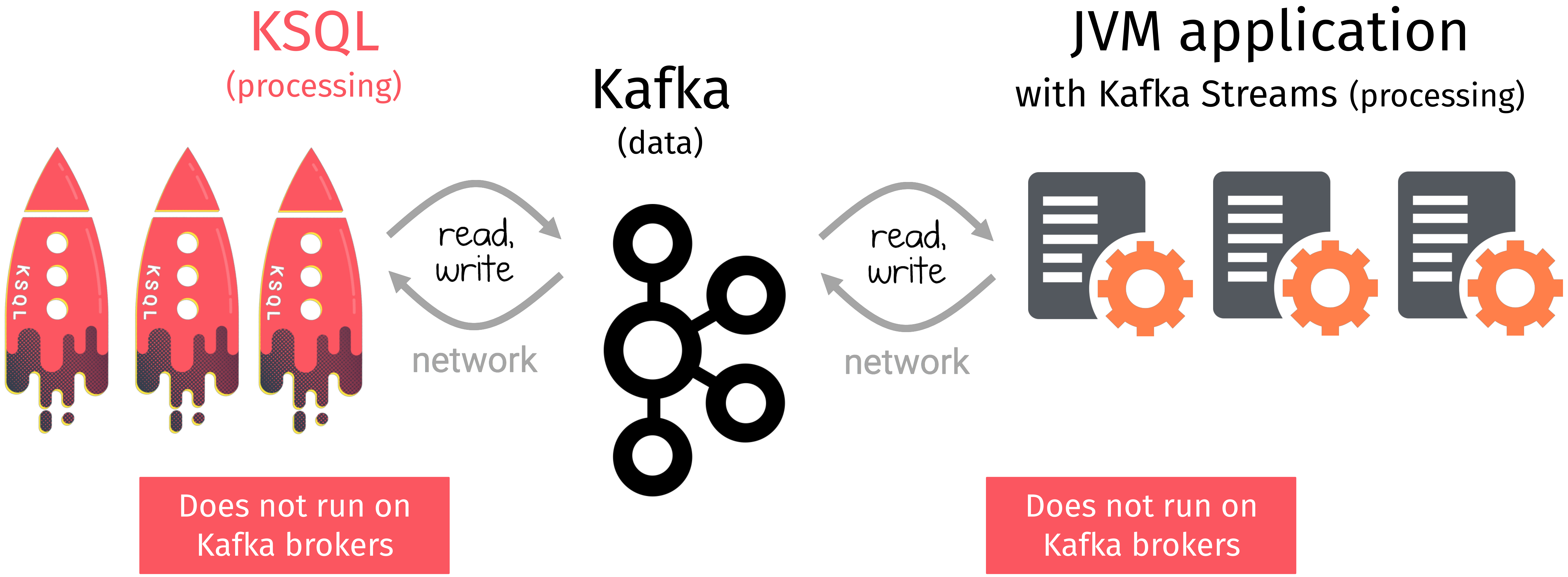
1 UI

2 CLI

3 REST

4 Headless

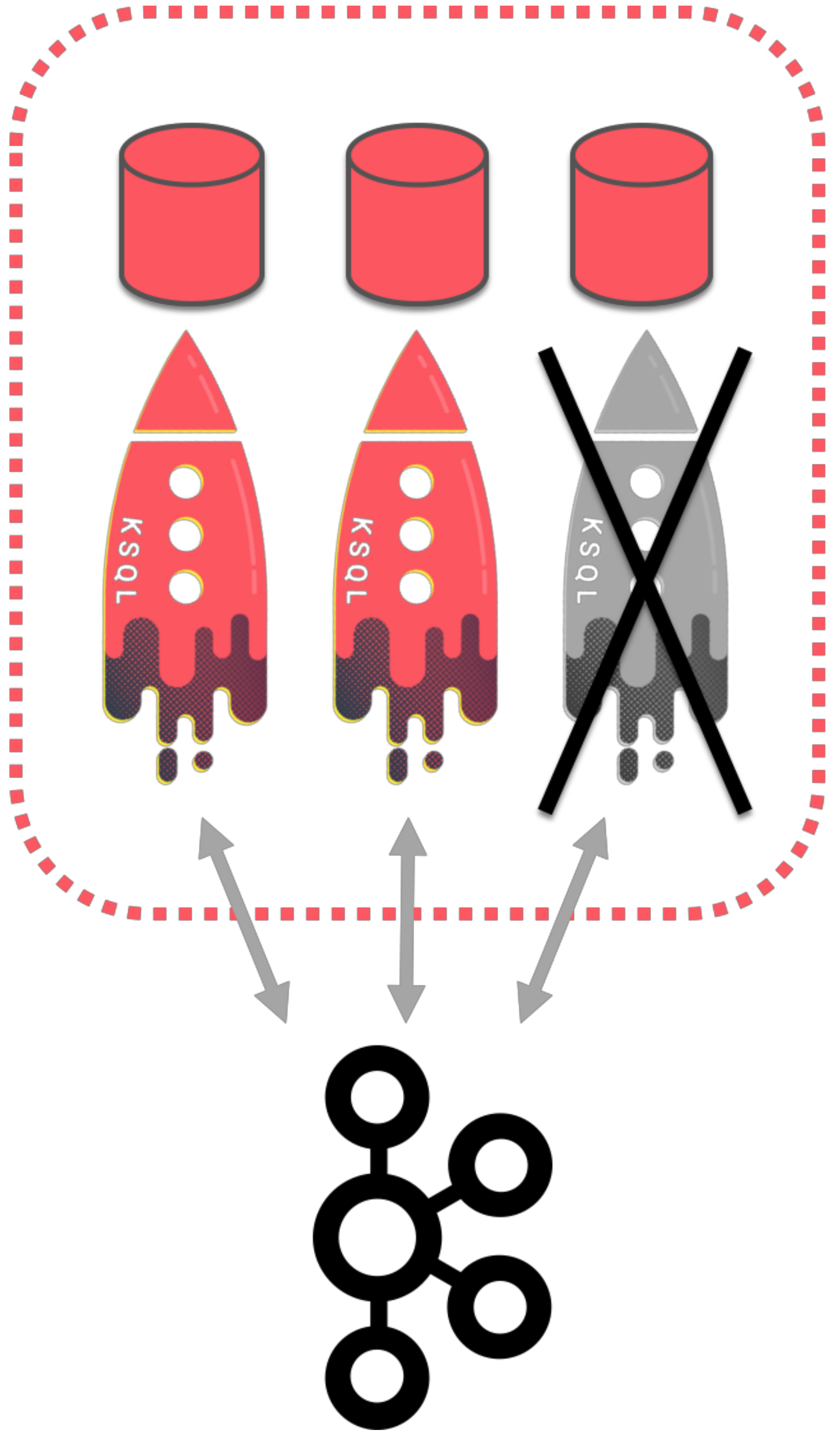
INTERACTION WITH KAFKA

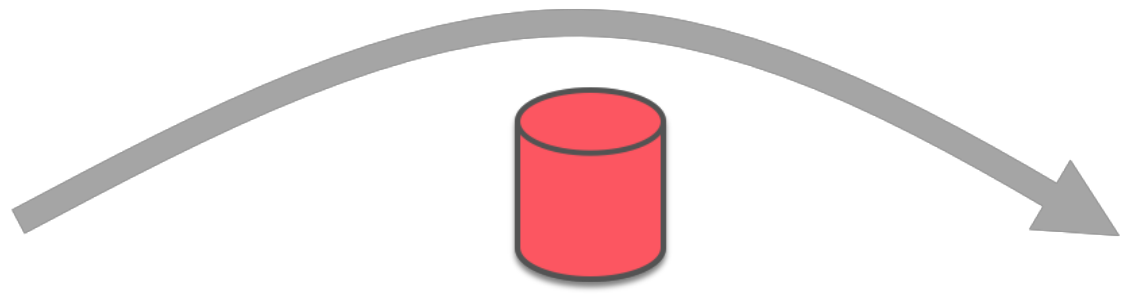


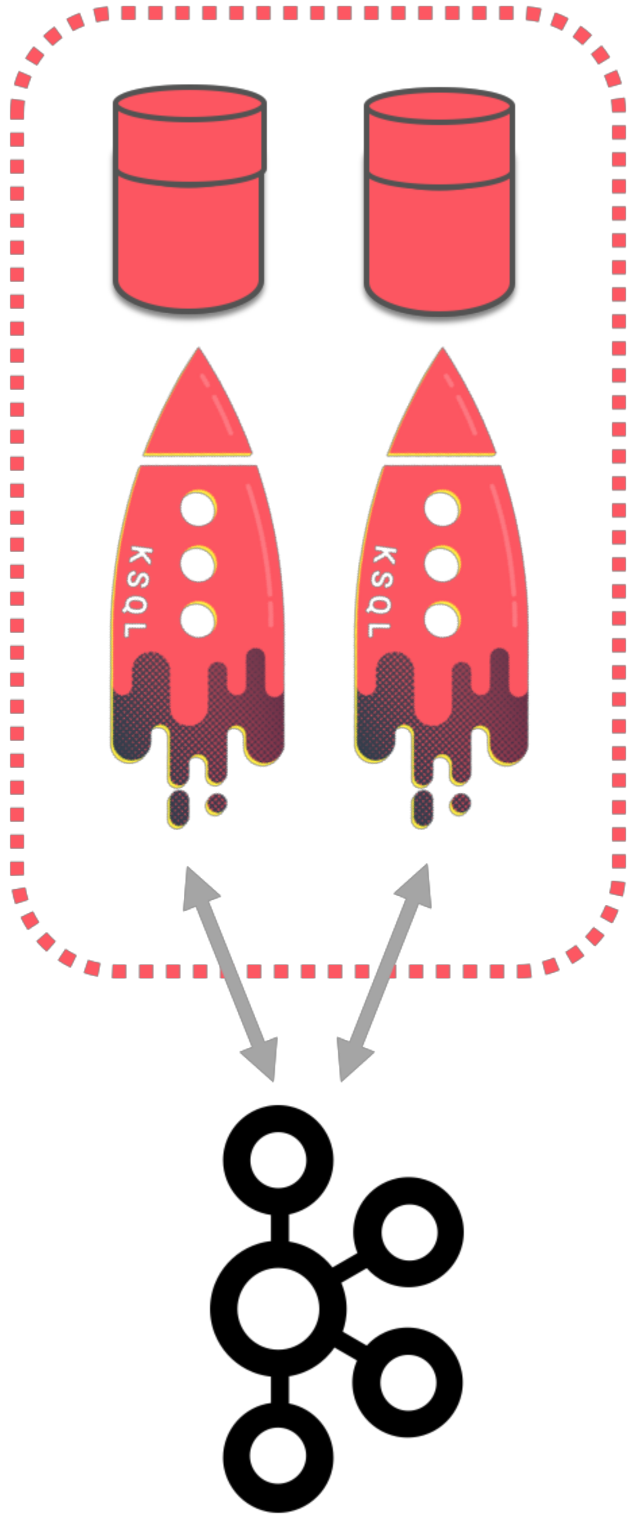
FAULT-TOLERANCE, POWERED BY KAFKA

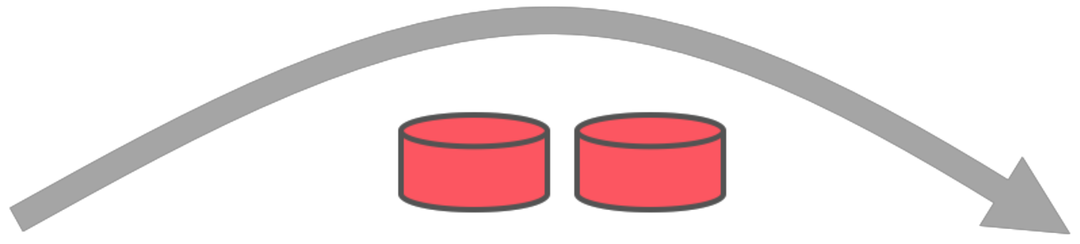
#3 died so #1 and #2 take over

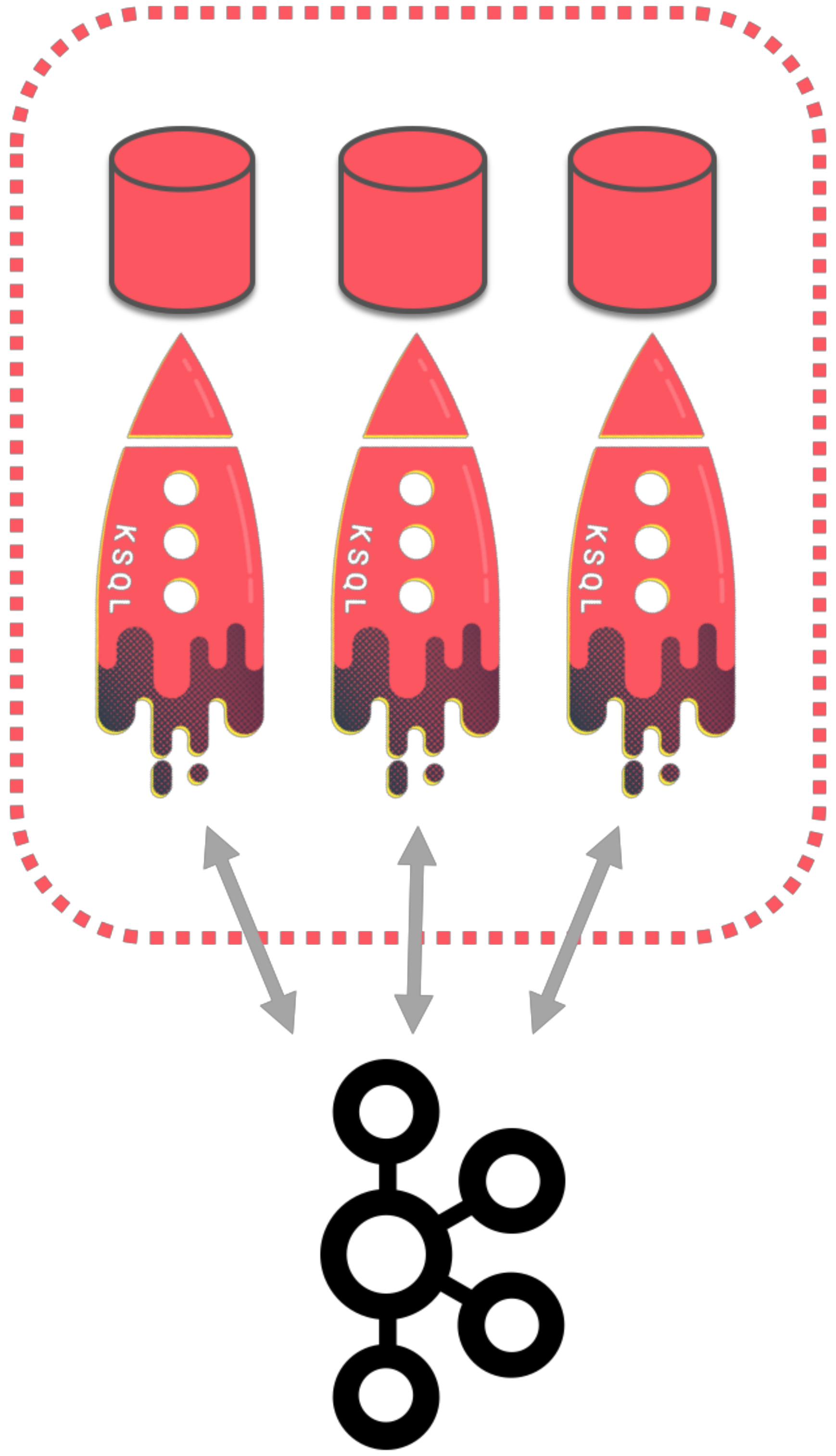
#3 is back so the work is split again











1. Kafka consumer group rebalance is triggered
2. Processing and state of #3 is migrated via Kafka to remaining servers #1 + #2

1. Kafka consumer group rebalance is triggered
2. Part of processing incl. state is migrated via Kafka from #1 + #2 to server #3

DIFFERENCES

	KSQL	kafka streams
You write…	**KSQL statements**	**JVM applications**
UI included for human interaction	**Yes**, in Confluent Platform	No
CLI included for human interaction	**Yes**	No
Data formats	Avro, JSON, CSV (today)	**Any data format**, including Avro, JSON, CSV, Protobuf, XML
REST API included	**Yes**	No, but you can DIY
Runtime included	**Yes**, the KSQL server	**Not needed**, applications run as standard JVM processes
Queryable state	Not yet	**Yes**

STANDING ON THE SHOULDERS OF STREAMING GIANTS

KSQL

Powered by

Kafka Streams

Powered by

Producer, Consumer APIs

Ease of use

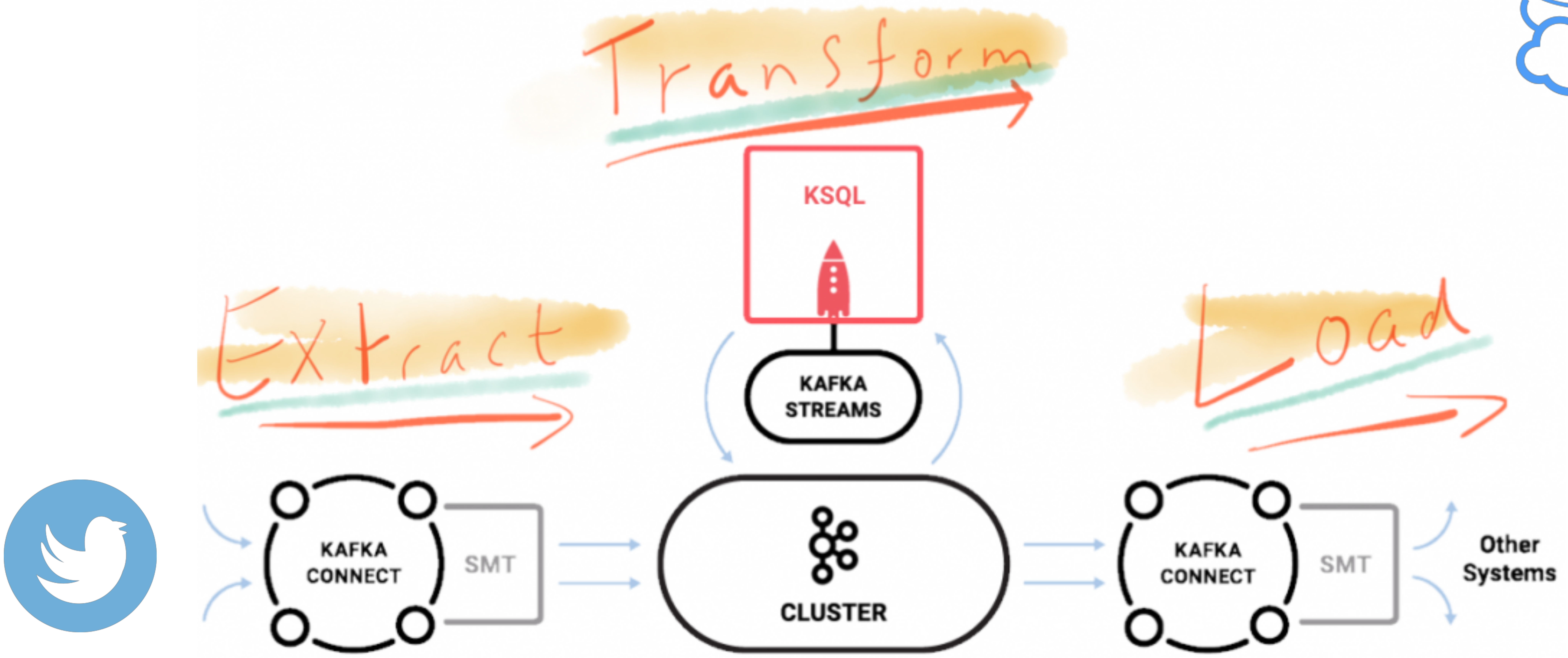
Flexibility

```
CREATE STREAM,
CREATE TABLE, SELECT, JOIN,
GROUP BY, SUM, …
```

KSQL UDFs

```
KStream<>, KTable<>, filter(), map(),
flatMap(), join(), aggregate(),
transform(), …
```

```
subscribe(), poll(), send(),
flush(), beginTransaction(), …
```


Transform
KSQL
KAFKA STREAMS
Extract
Load
KAFKA CONNECT
SMT
CLUSTER
KAFKA CONNECT
SMT
Other Systems

ONE LAST THING...

kafka summit

https://kafka-summit.org

Gamov30

APRIL 2, 2019

NEW YORK CITY

REGISTER

LEARN MORE

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