STREAMING ON KUBERNETES: DOES IT HAVE TO BE THE HARD WAY?

FALL, 2019 / SAN FRANCISCO, CA 2019

@GAMUSSA | #KAFKASUMMIT | @CONFLUENTINC

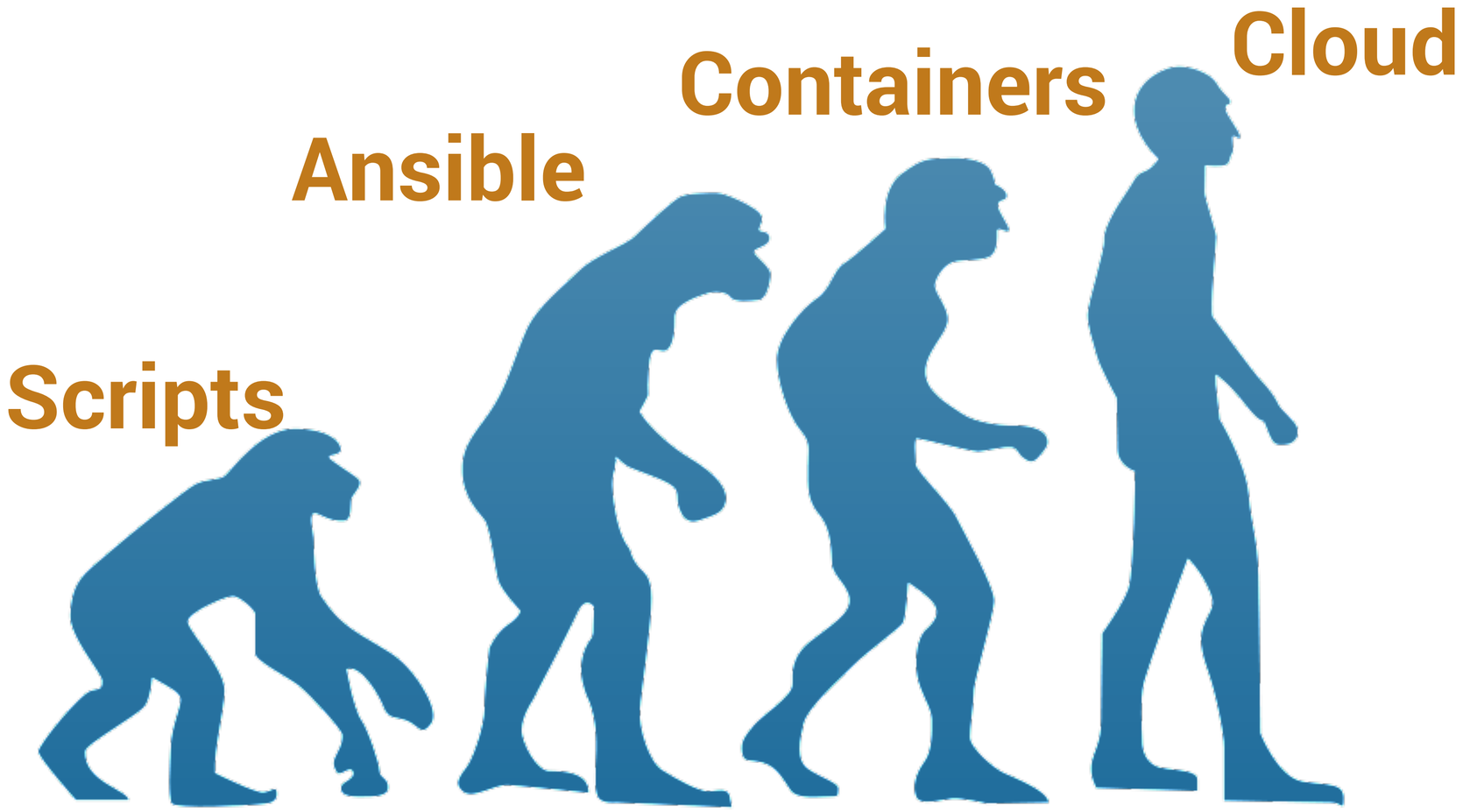

Streams

I BUILD HIGHLY SCALABLE

Hello World

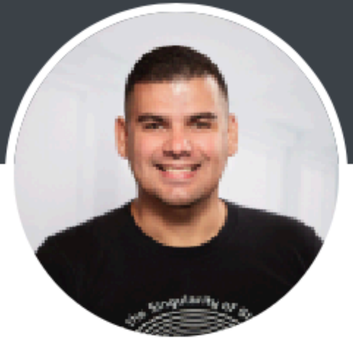
APPS

EVOLUTION OF #DEVKAFKAOPS



CONFLUENT CLOUD

The Rise of Managed Services for Apache Kafka



Ricardo Ferreira

September 20, 2019

As a distributed system for collecting, storing, and processing data at scale, Apache Kafka® comes with its own deployment complexities. Luckily for on-premises scenarios, a myriad of deployment options are available, such as the Confluent Platform which can be deployed on bare metal, virtual

Is the service offered a managed service for Kafka?

Key design decisions abstracted from the user

Requires the user to take the wheel and decide

Claes Mogren
@claesmogren

"Kubernetes is of course named after the Greek god of spending money on cloud services." - @QuinnyPig

3:54 PM · Mar 20, 2019 · Twitter Web Client

68 Retweets **233** Likes https://twitter.com/claesmogren/status/1108456688175079424

Corey Quinn @QuinnyPig · 10h

Replying to @timbray

Proper pronunciations include but are not limited to:
Kubernetes
Kubernetis
Cooper nets
KuBERneties
Kate's
K 8 s
Cubernetes
Kyoober nets
Kuiper Ned's
Kubernandos
Kuberkitties
Cue bernettes
Kubernoots
Koober neats
Kubernetix
Clueberneties

2 4 25

https://twitter.com/QuinnyPig/status/1150927901782499330

WHO RUN STATELESS APPS IN KUBERNETES?

WHO THINKS IT'S A GOOD IDEA?

WHO RUN STATEFUL APPS IN KUBERNETES?

WHO THINKS IT'S A GOOD IDEA?

WELL, IT'S TRICKY ©

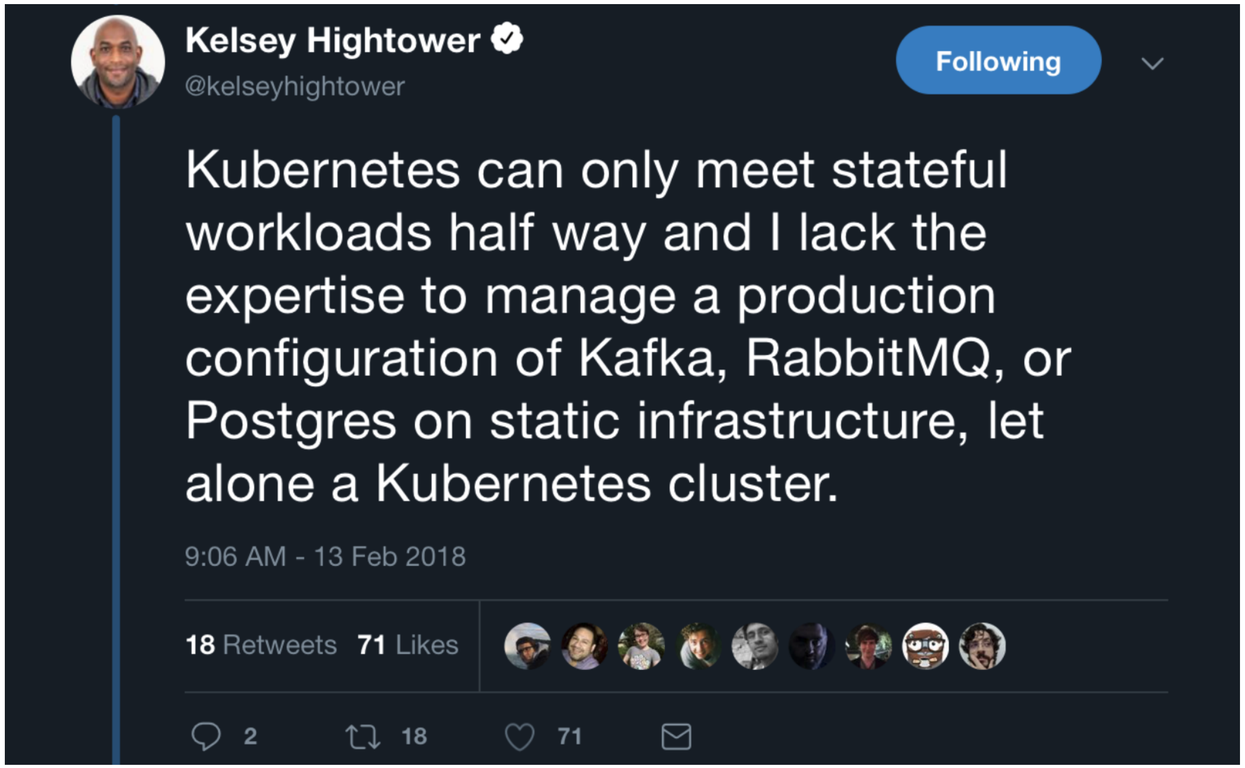
Translating an existing architecture to Kubernetes

External access to brokers and other components

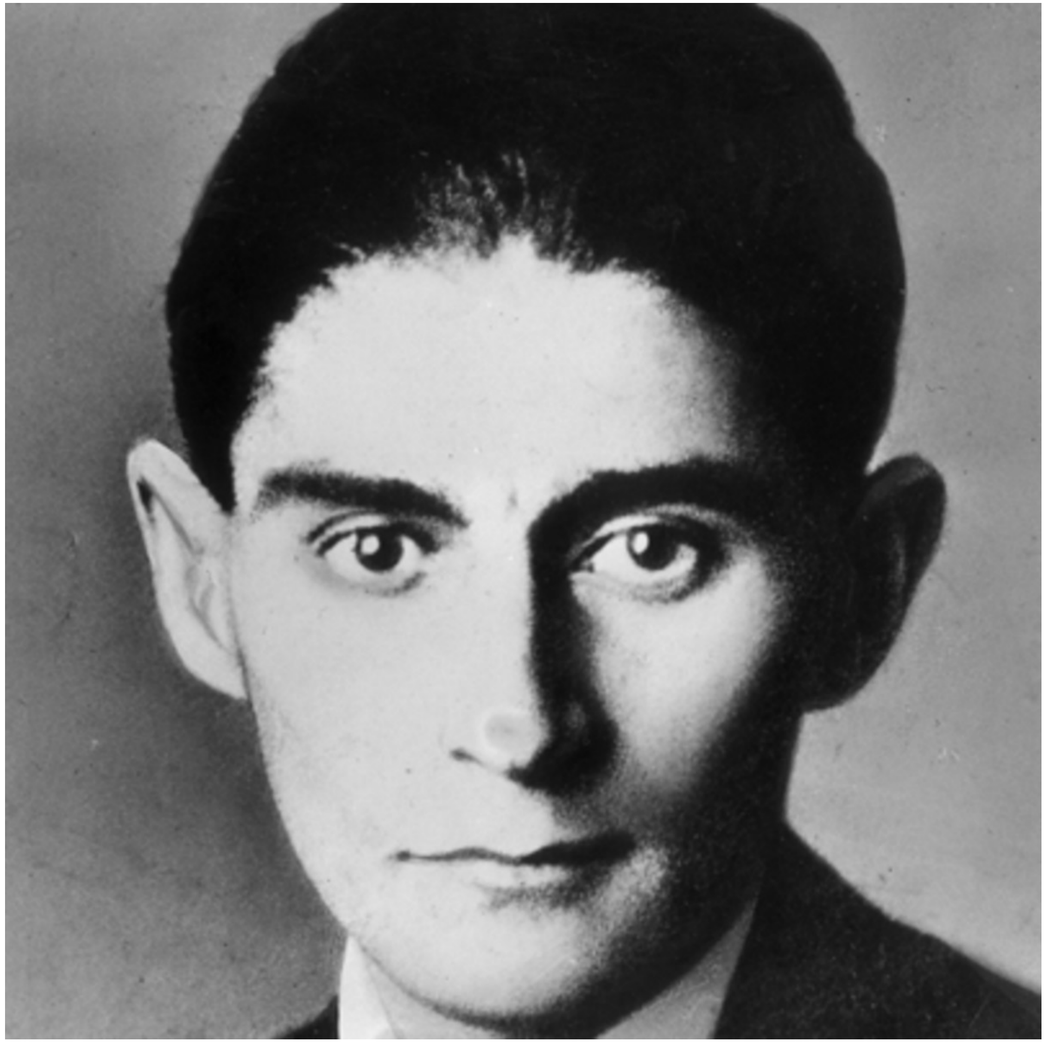
Persistent Storage options on prem and clouds

Security Configuration and Upgrades

KAFKAESQUE WORLD OF KAFKA ON KUBERNETES



DUDE, YOU SAID <<IT DOESN'T HAVE TO BE HARD WAY>>



@Micha8LNg

DO KAFKA ON KUBERNETES DEMO

AND EVERYONE LOOSES THEIR MINDS

confluent

TWEET, YEAH 🚀

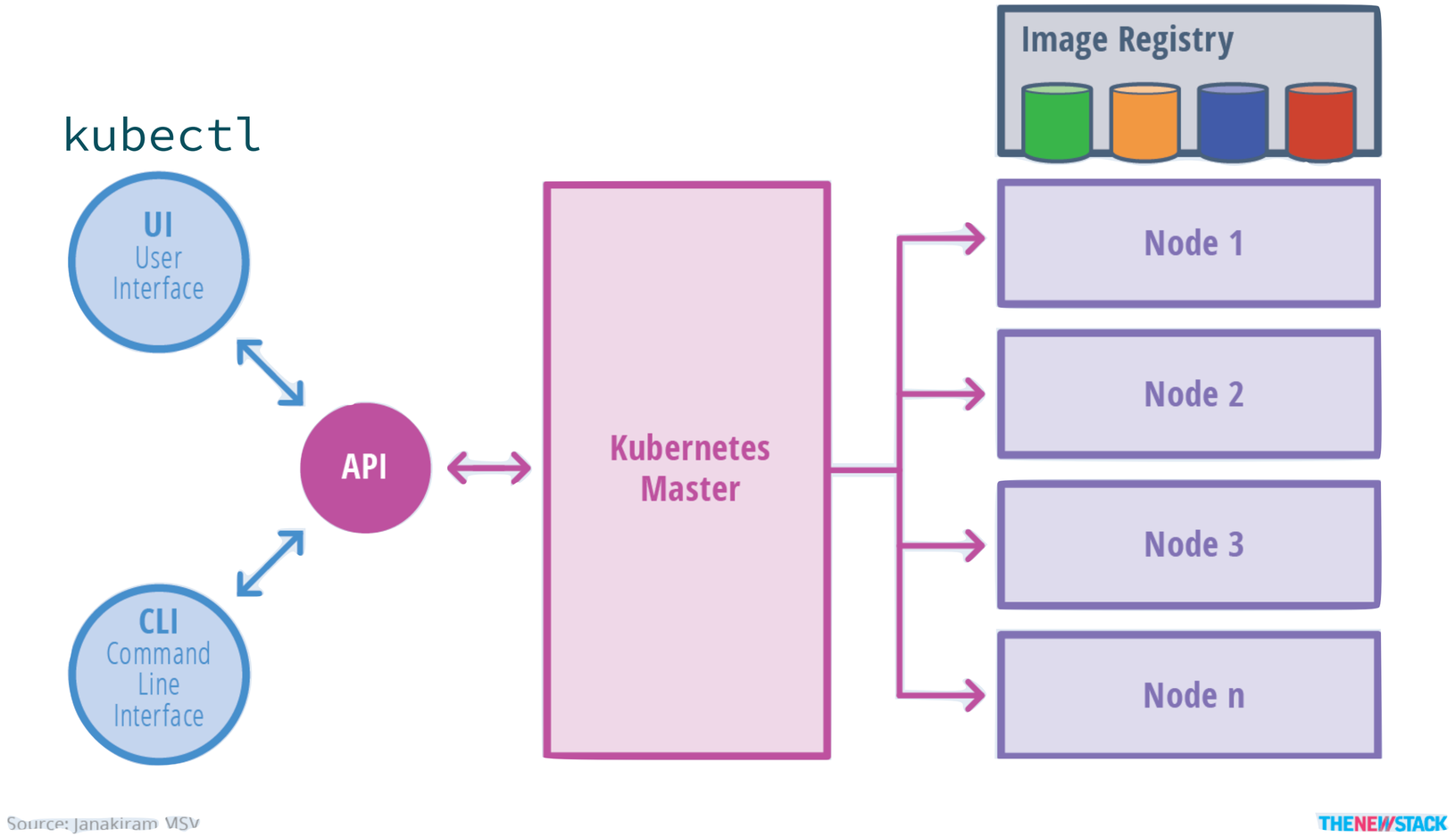
- Follow @gamussa @confluentinc
- 📸🖼️🏋️‍♀️😂
- Tag @gamussa
- With #KafkaSummit

KUBERNETES FUNDAMENTALS

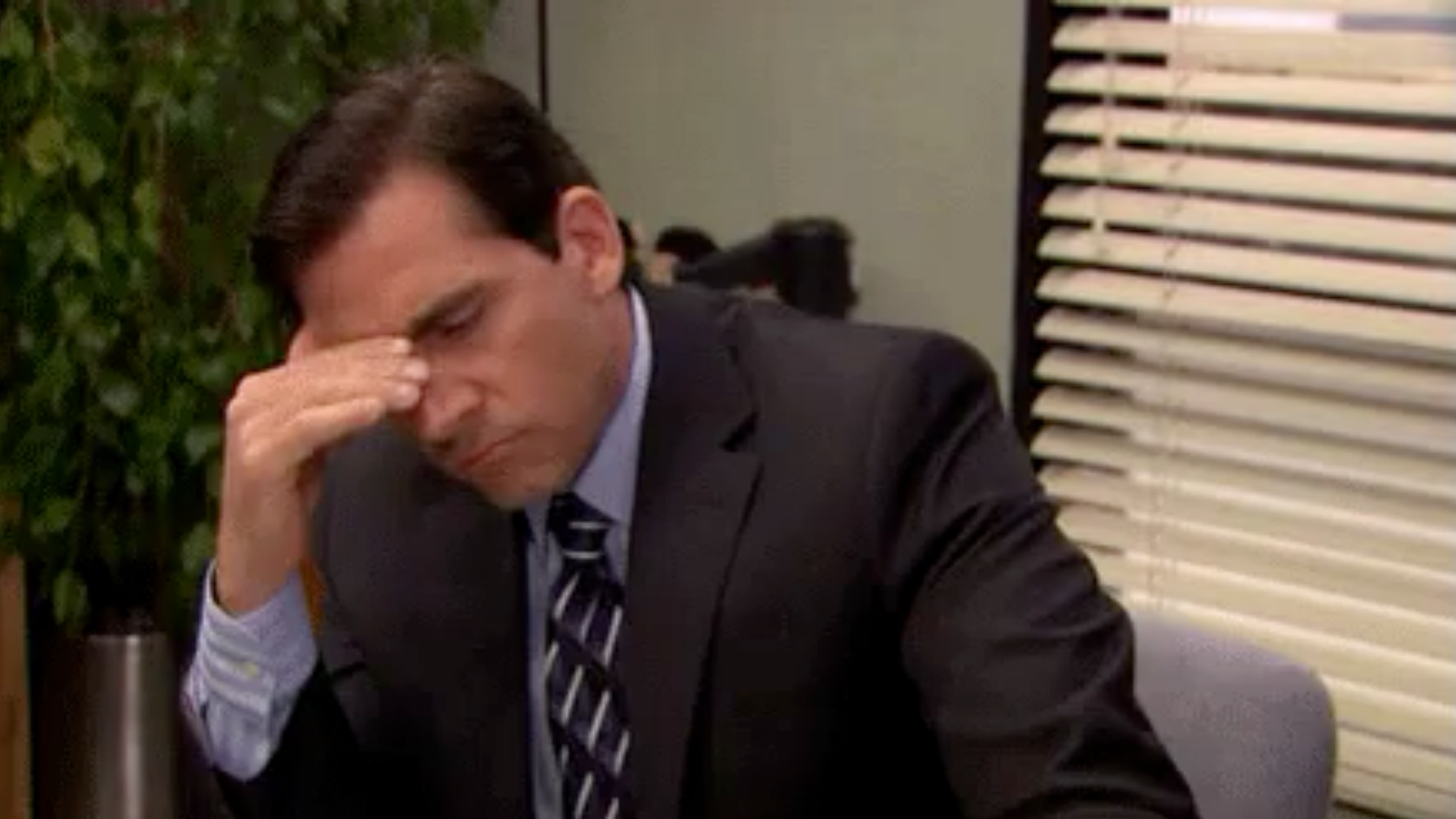
KUBERNETES

- Schedules and allocates resources
- Networking between Pods
- Storage
- Service Discovery

KUBERNETES ARCHITECTURE



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

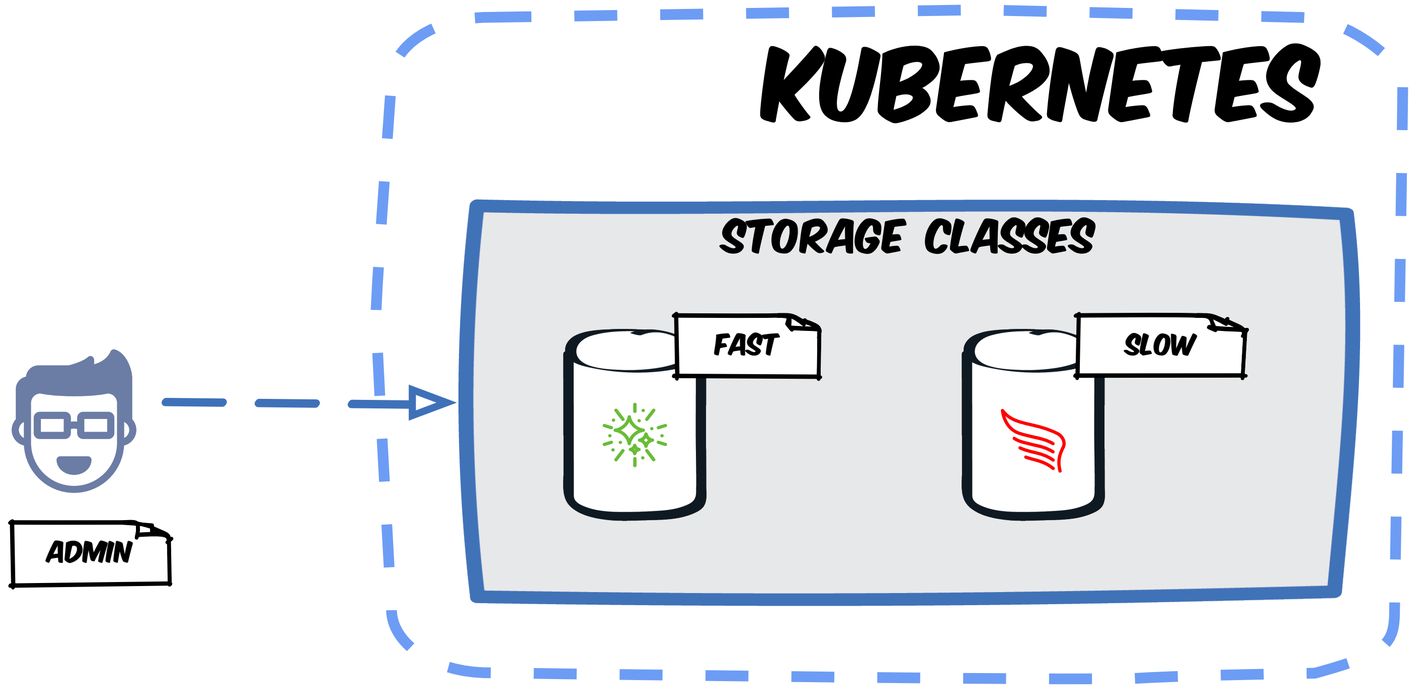


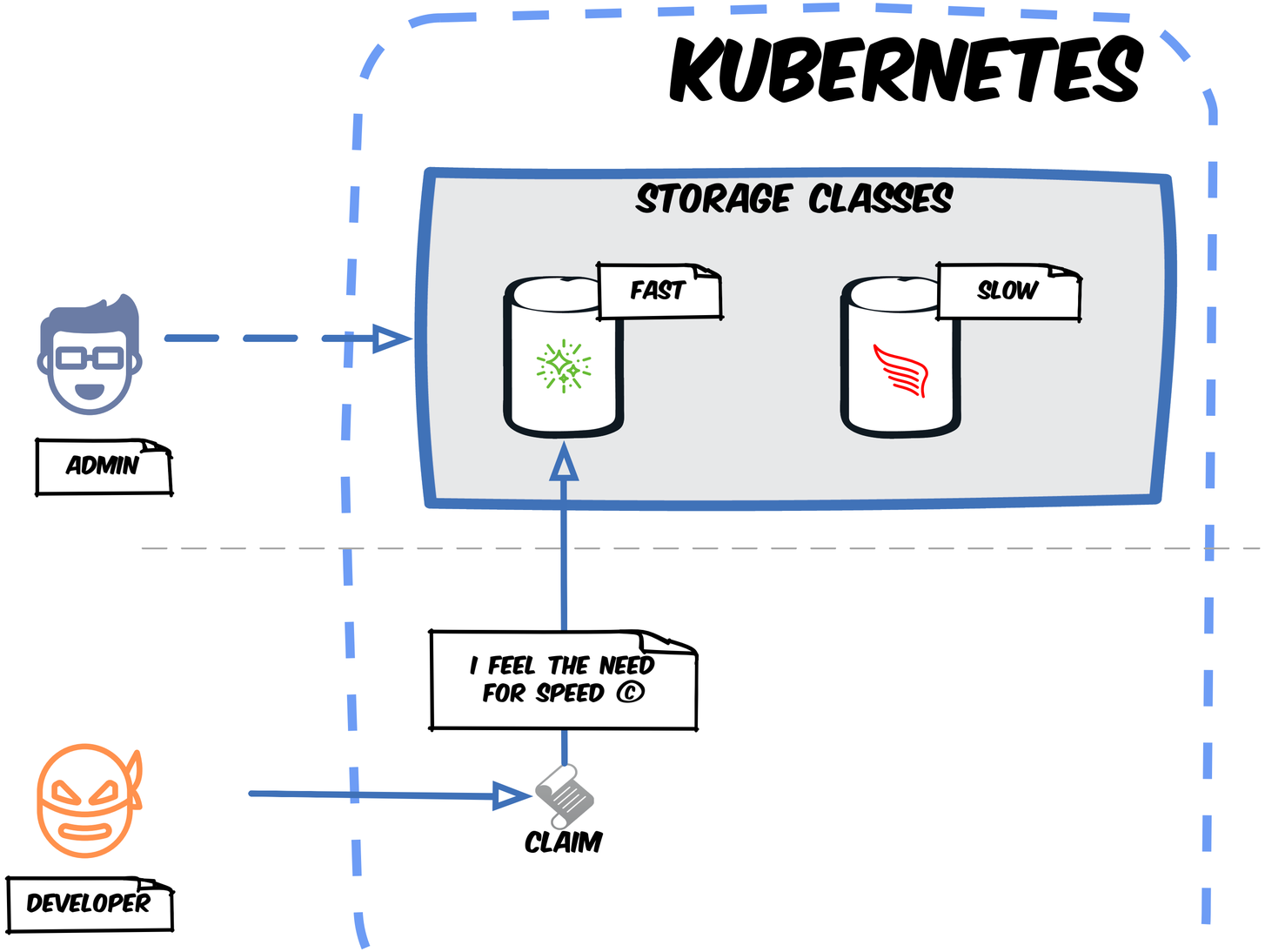
POD

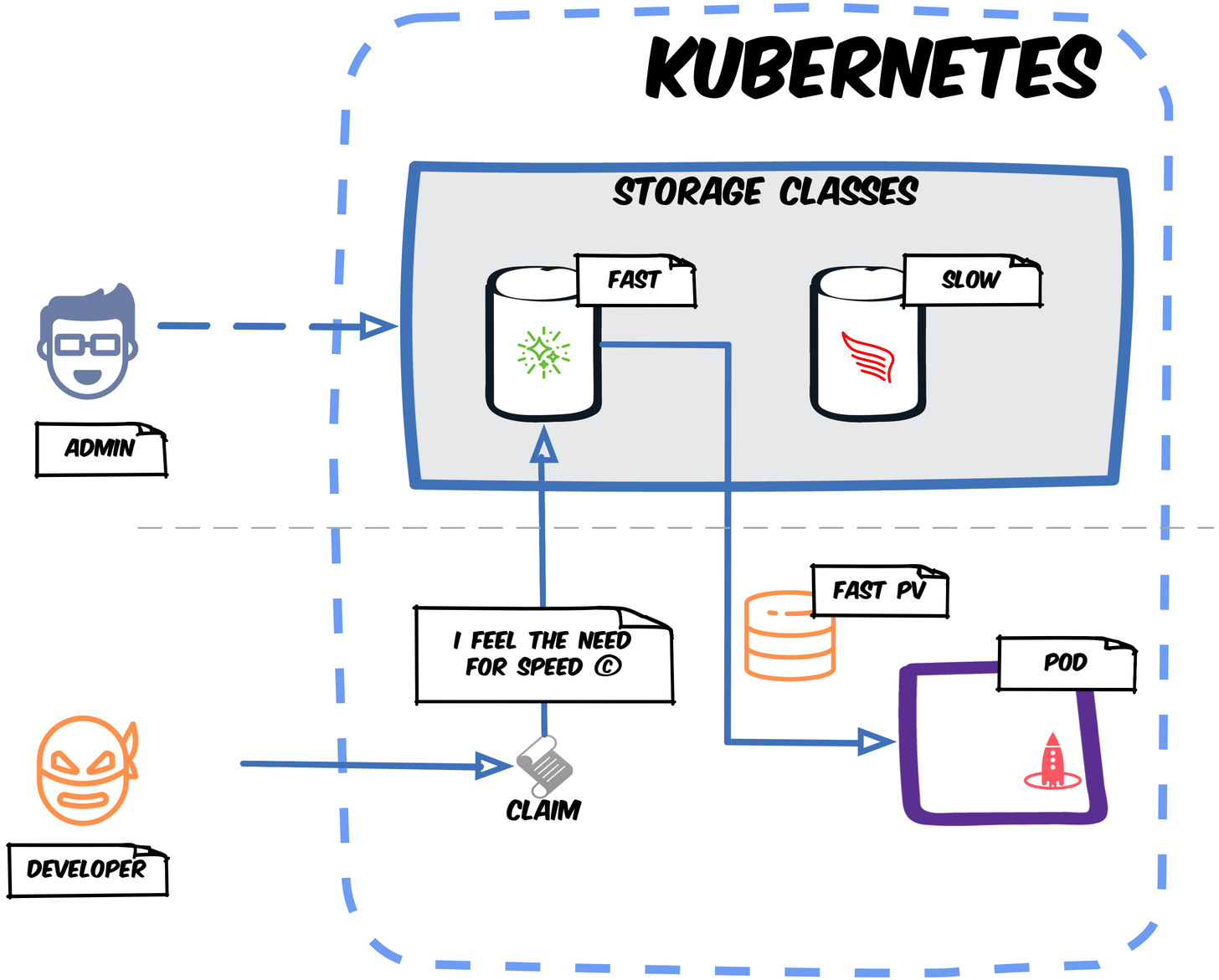
- Basic Unit of Deployment in Kubernetes
- A collection of containers sharing:
  - Namespace
  - Network
  - Volumes

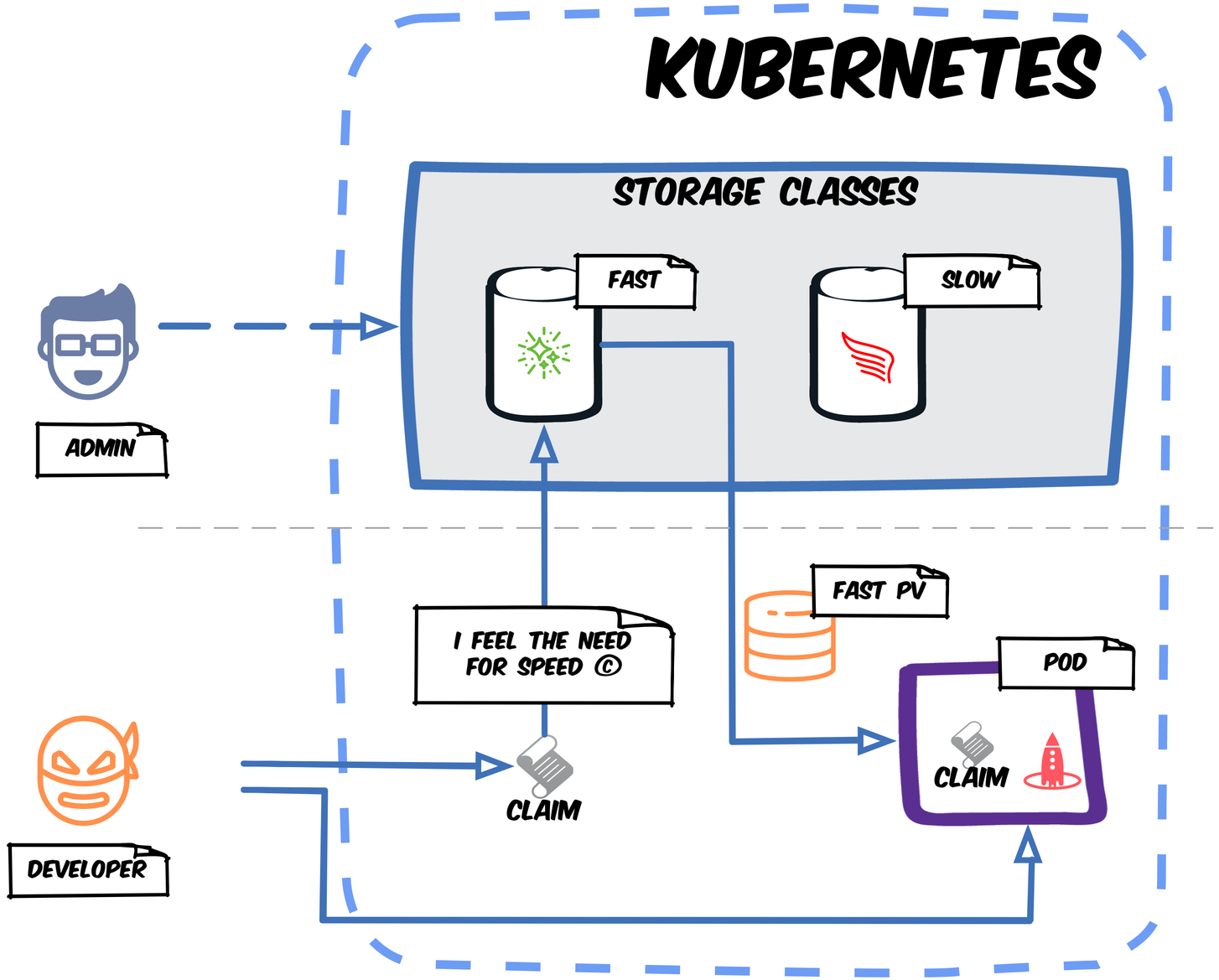
STORAGE

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

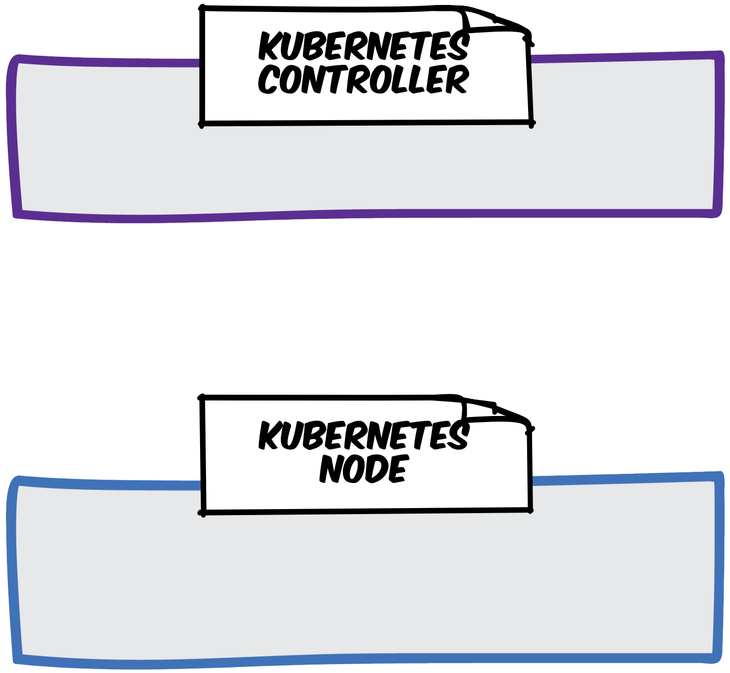

KUBERNETES
STORAGE CLASSES
FAST
SLOW
ADMIN

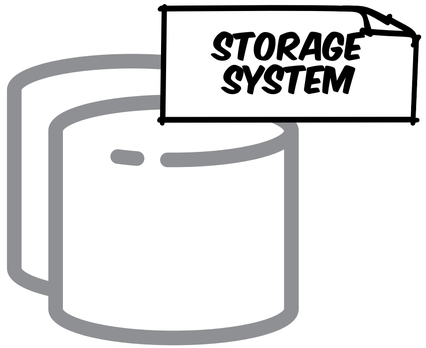

KUBERNETES
STORAGE CLASSES
FAST
SLOW
ADMIN
I FEEL THE NEED
FOR SPEED ©
CLAIM
DEVELOPER


KUBERNETES
STORAGE CLASSES
FAST
SLOW
ADMIN
I FEEL THE NEED
FOR SPEED ©
FAST PV
POD
CLAIM
DEVELOPER


KUBERNETES
STORAGE CLASSES
FAST
SLOW
ADMIN
I FEEL THE NEED
FOR SPEED ©
FAST PV
POD
CLAIM
CLAIM
DEVELOPER

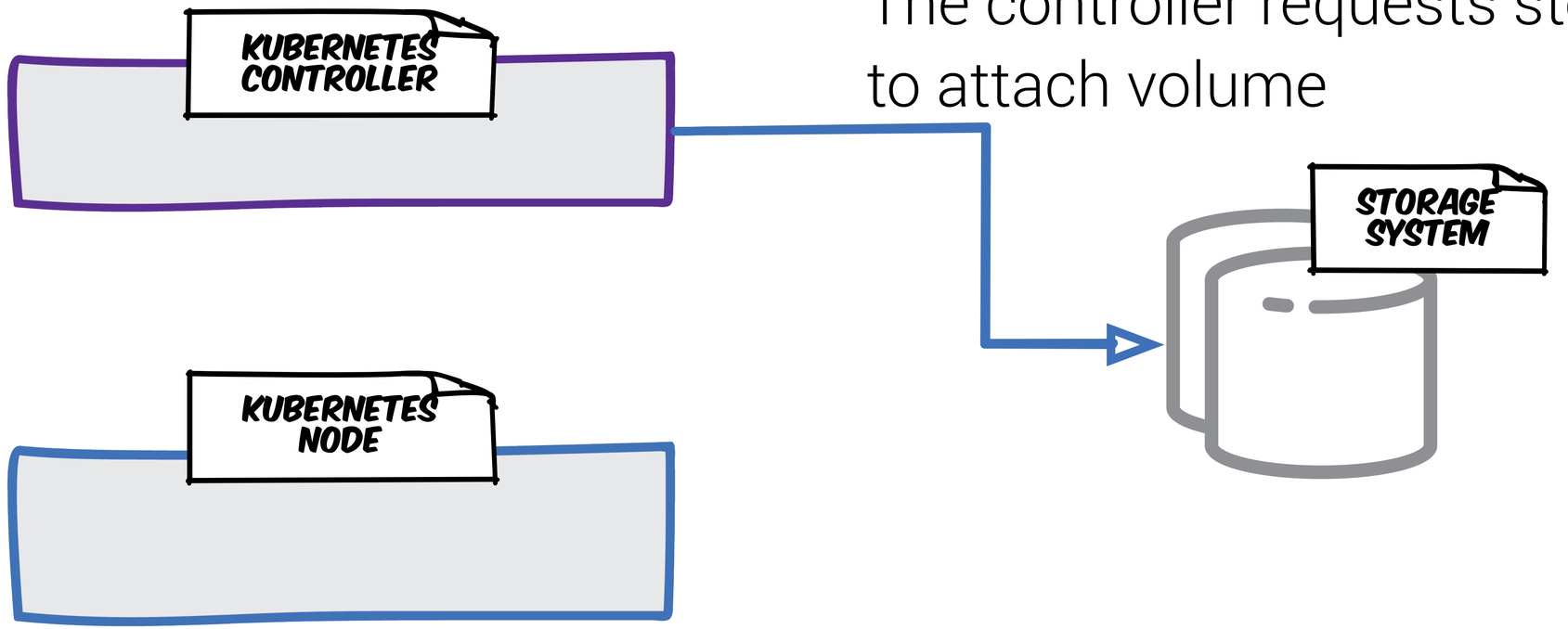
PROVISIONING STORAGE IN KUBERNETES





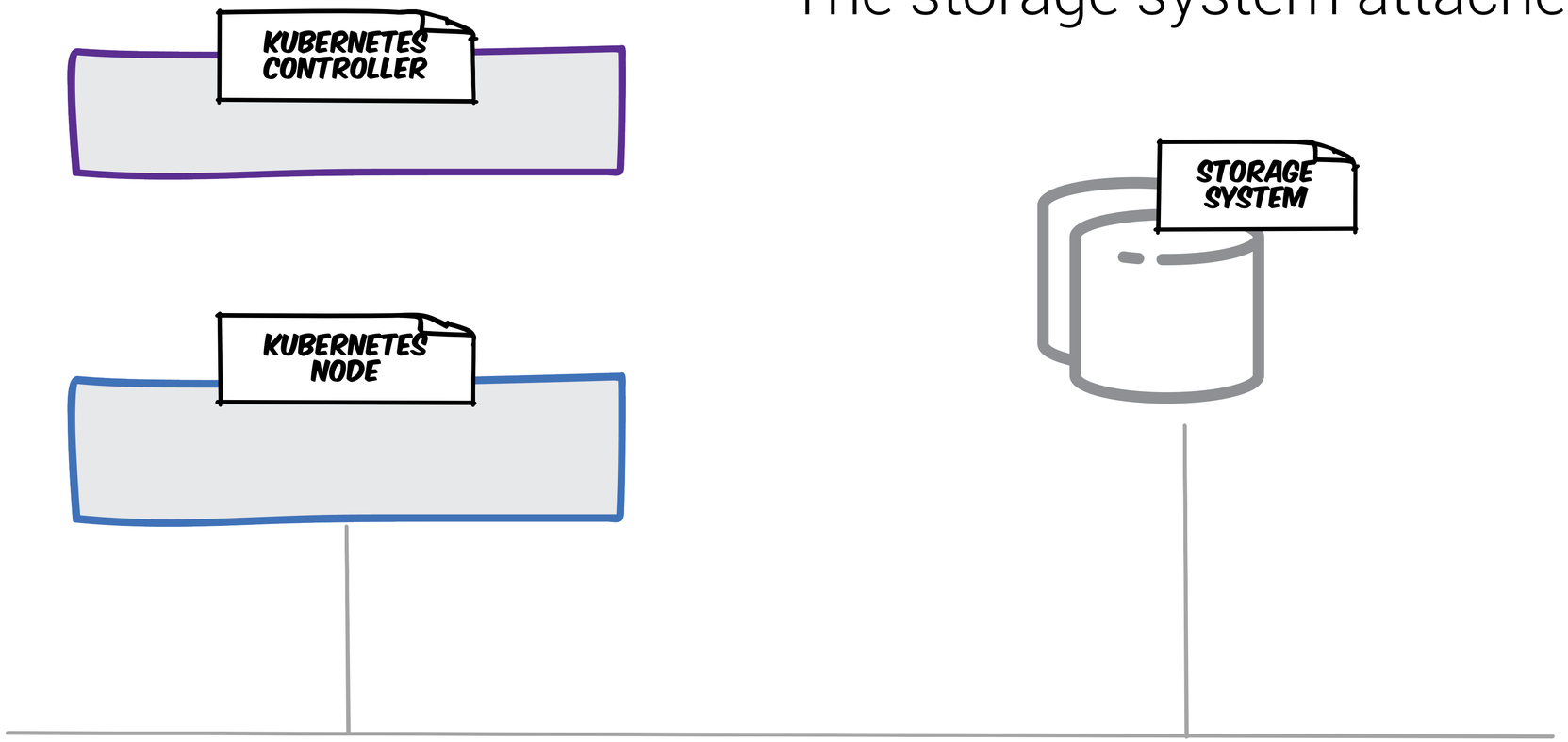
PROVISIONING STORAGE IN KUBERNETES

The controller requests storage system to attach volume



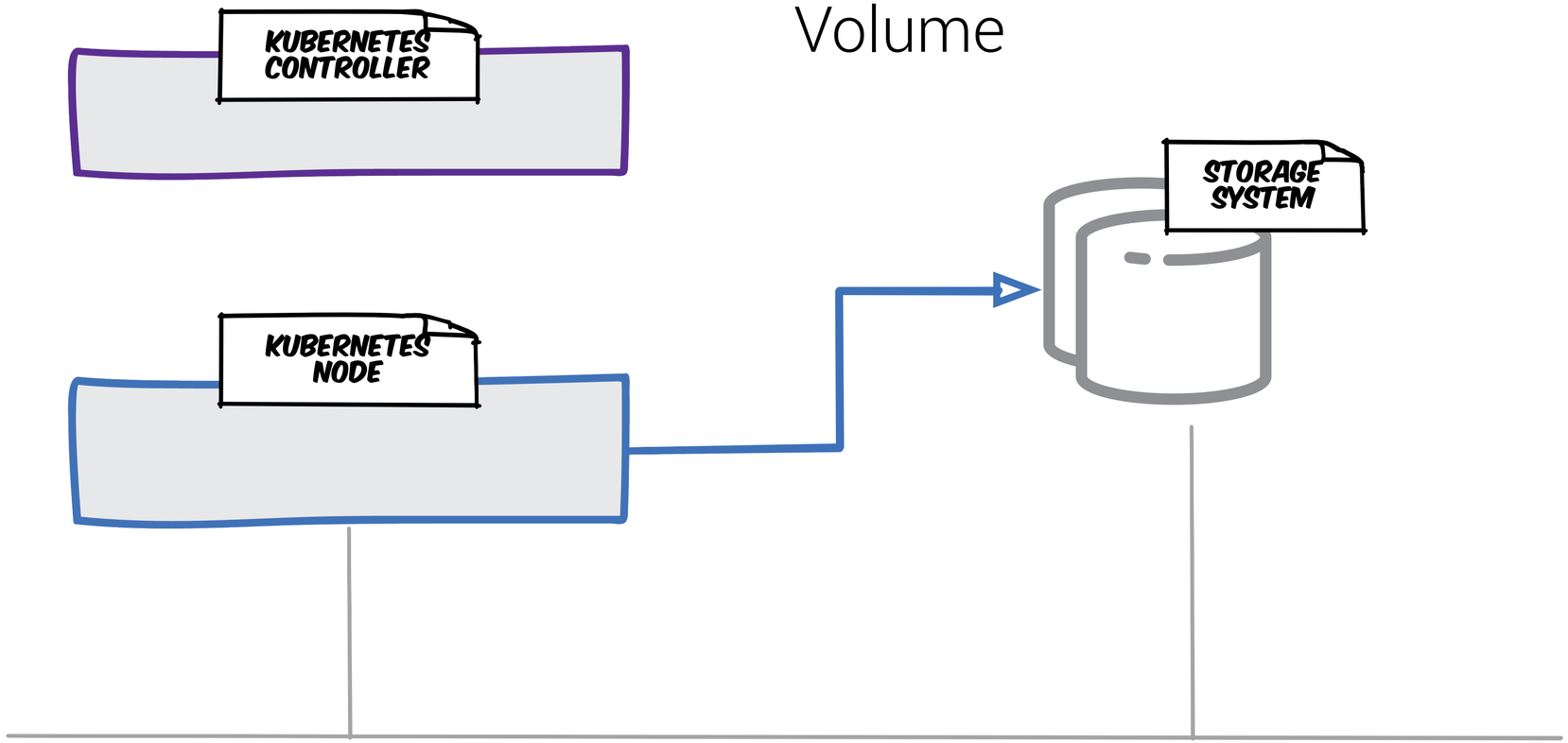
PROVISIONING STORAGE IN KUBERNETES

The storage system attaches the volume



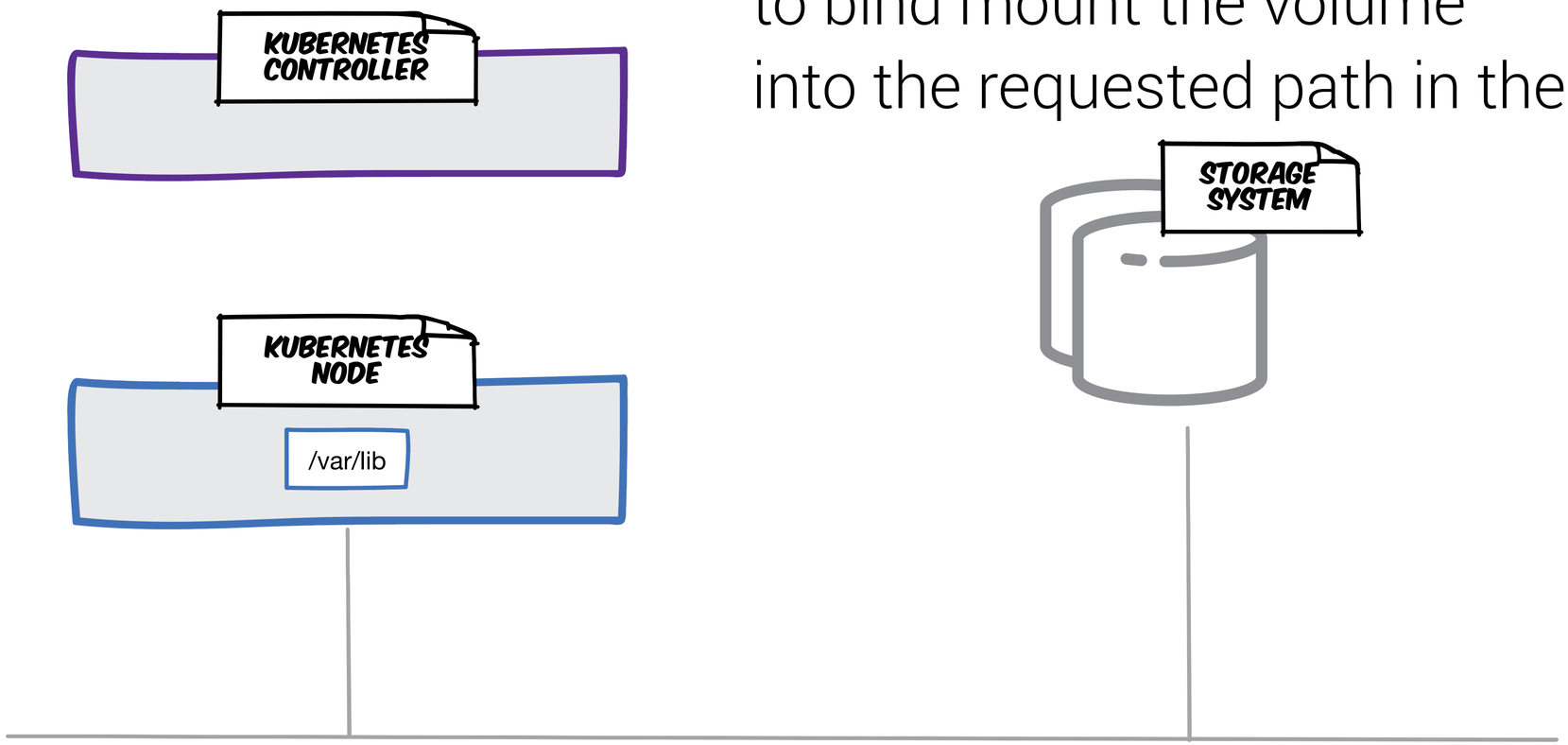
PROVISIONING STORAGE IN KUBERNETES

Kubelet requests storage system to mount Volume



PROVISIONING STORAGE IN KUBERNETES

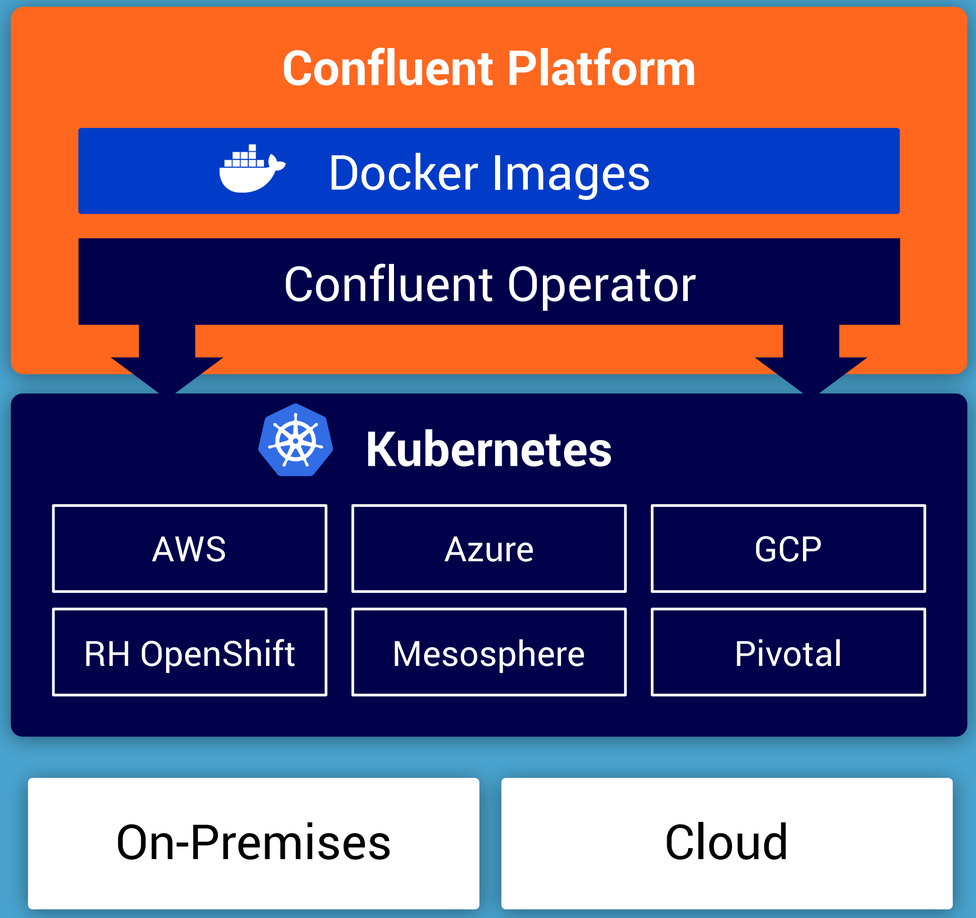
Kubelet can now request container runtime
to bind mount the volume
into the requested path in the container.



BUT I JUST WANT TO DEPLOY KAFKA

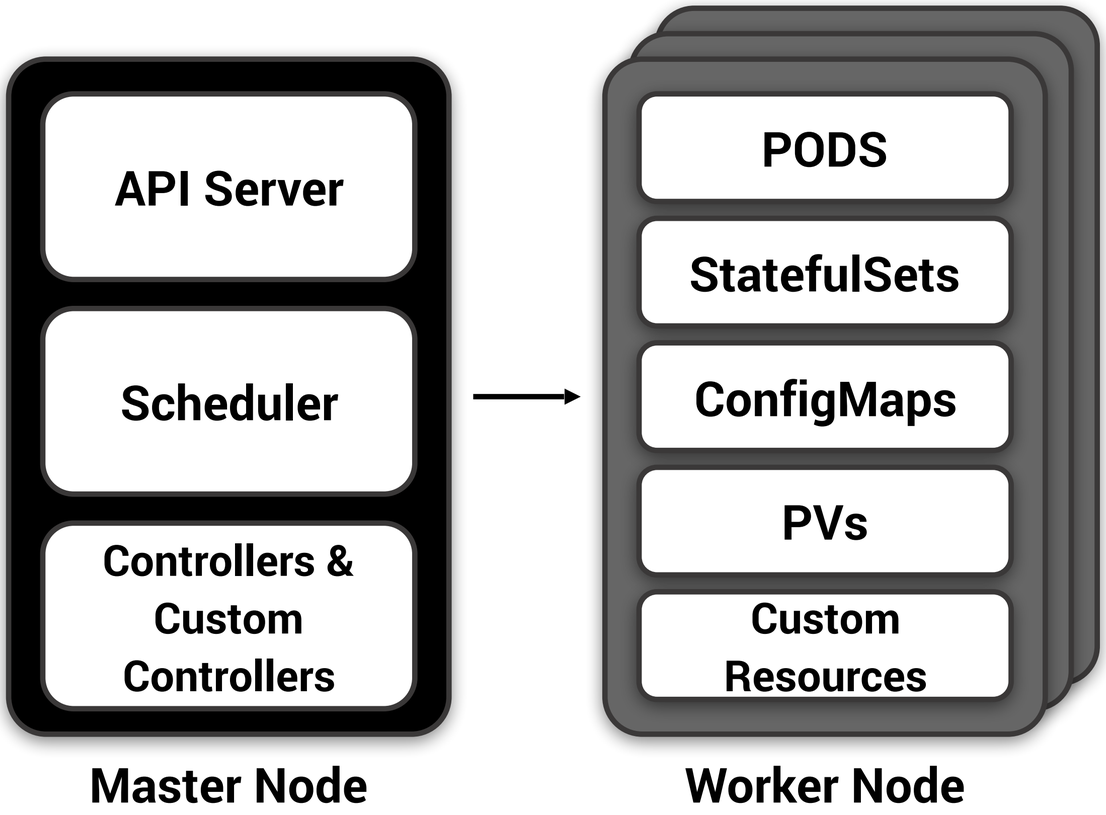
CONFLUENT OPERATOR: APACHE KAFKA ON KUBERNETES MADE SIMPLE

Run Apache Kafka and Confluent Platform as a cloud-native application on Kubernetes to **minimize operating complexity** and **increase developer agility**

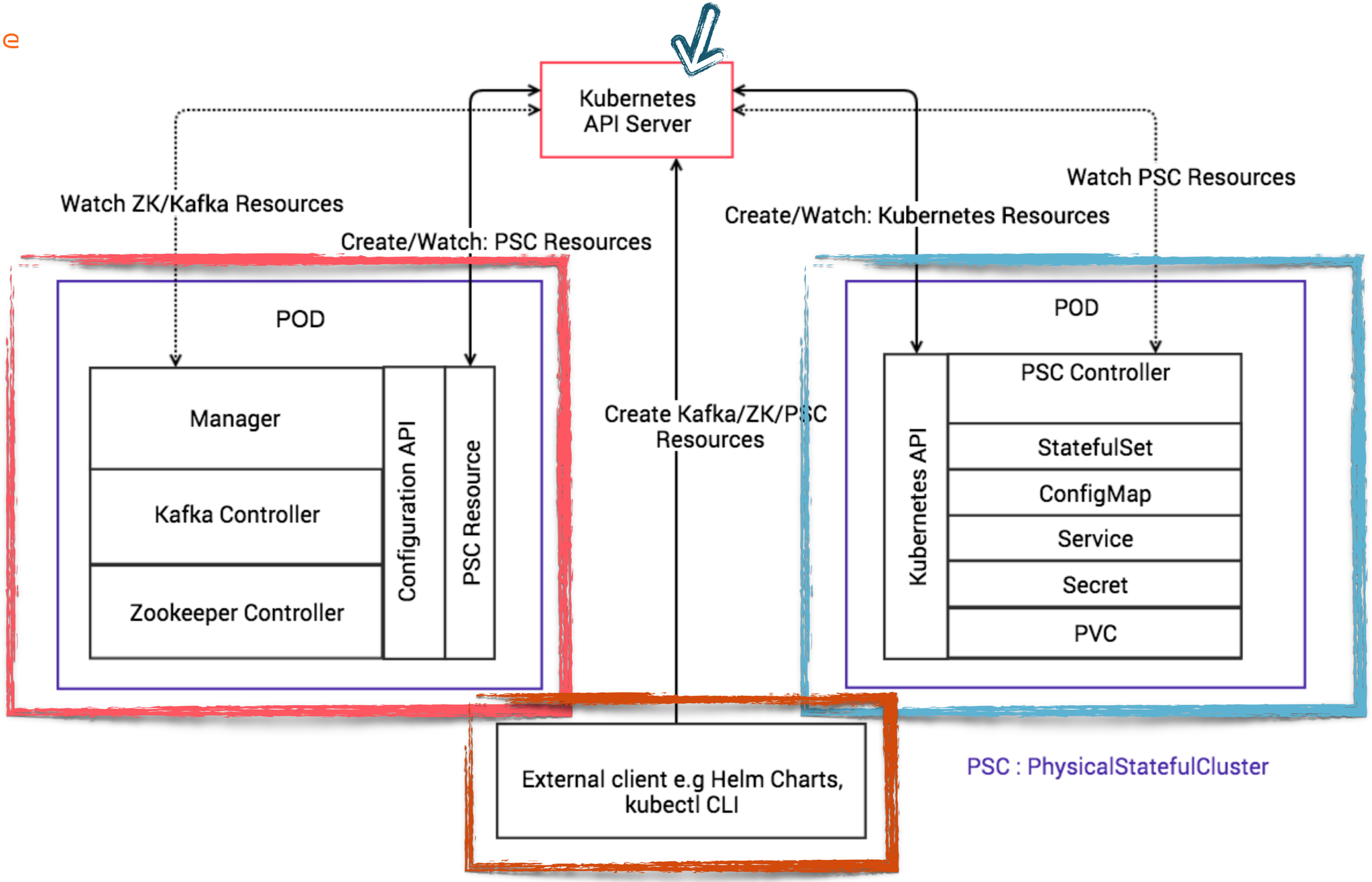



CUSTOM RESOURCE
CONTROLLER

A CUSTOM KUBERNETES CONTROLLER



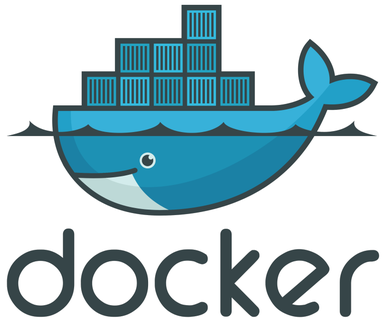
- Nodes and pods are where Applications run on Kubernetes
- Applications use objects like StatefulSets, Configmaps, PVs
- Custom Controllers create custom resources that provide unique application functionality:
  - Upgrades, elasticity, Kafka Operational Logic


Kubernetes API Server
Watch ZK/Kafka Resources
Create/Watch: PSC Resources
Watch PSC Resources
Create/Watch: Kubernetes Resources
POD
Manager
Kafka Controller
Zookeeper Controller
Configuration API
PSC Resource
Create Kafka/ZK/PSC Resources
POD
Kubernetes API
PSC Controller
StatefulSet
ConfigMap
Service
Secret
PVC
External client e.g Helm Charts, kubectl CLI
PSC : PhysicalStatefulCluster

WORKLOADS DEPLOYMENT

USUAL SUSPECTS

Certified container images



Operator Helm Charts

confluent

ROLLING UPGRADE

Kafka Broker Upgrades:

1. Stop the broker, upgrade Kafka
2. Wait for Partition Leader reassignment
3. Start the upgraded broker
4. Wait for zero under-replicated partitions
5. Upgrade the next broker

```
> 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 SCALE

Spin up new brokers, connect workers easily

Manual Rebalance required in v1.0

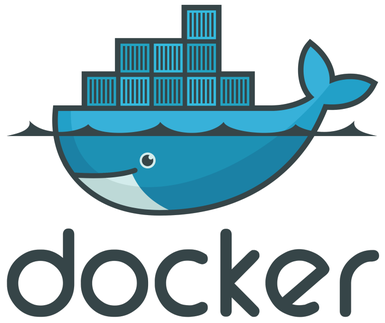
- Determine balancing plan
- Execute balancing plan
- Monitor Resources

```
> cat kafka_new.yml

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

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

One More Thing…

USUAL SUSPECTS

Certified container images



Pivotal Container Service

RED HAT OPENSHIFT Container Platform

Google Cloud

amazon web services

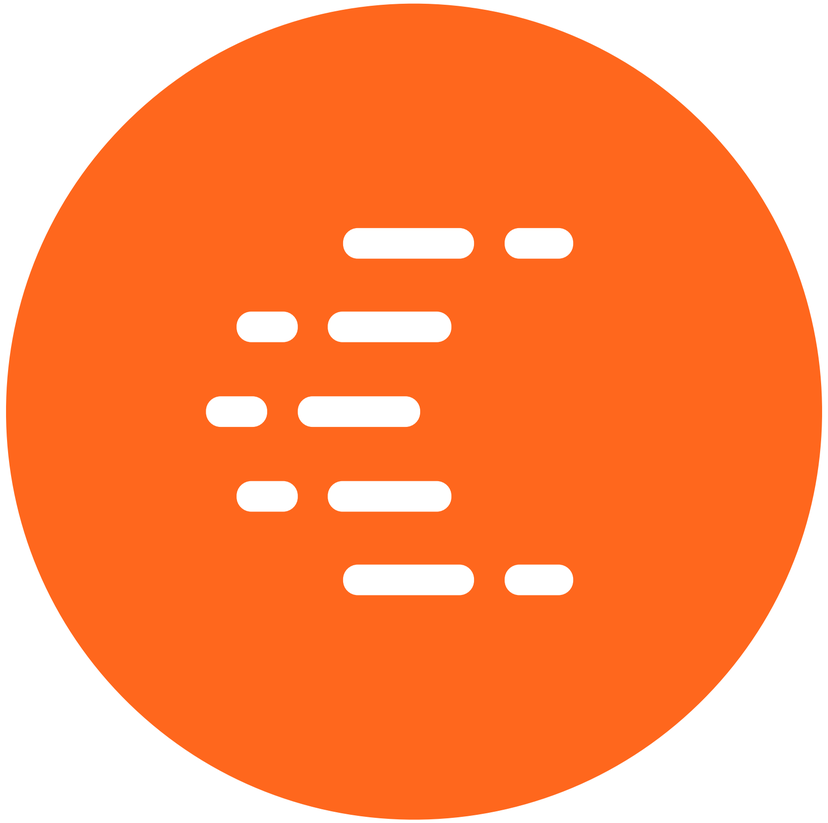
Microsoft Azure

KOTLIN DSL FOR KUBERNETES

```
fun main() {
    val client = DefaultKafkaClient().inNamespace("operator")
    println(client.kafkaClusters().list())
    client.kafkaClusters().create(
        newKafkaCluster {
            metadata {
                name = "kafka"
            }
            spec {
                replicas = 3
            }
        }
    )
    println(client.kafkaClusters().list())
}
```

KOTLIN DSL FOR KUBERNETES

```
fun main() {
    val client = DefaultKafkaClient().inNamespace("operator")
    println(client.kafkaClusters().list())
    client.kafkaClusters().create(
        newKafkaCluster {
            metadata {
                name = "kafka"
            }
            spec {
                replicas = 3
            }
        }
    )
    println(client.kafkaClusters().list())
}
```



THANKS!

@gamussa
viktor@confluent.io

@Micha8LNg
michael.ng@confluent.io

https://slackpass.io/confluentcommunity

#kubernetes


confluent