

# Kubernetes для разработчиков

Ilia Bocharov 2023

# О себе

- 28 лет
- В ИТ с 2015, коммерческий опыт с 2016
- Java, Kotlin, Scala, C#, JavaScript, TypeScript, Yaml
- <https://bocharoviliyav.blog>
- <https://t.me/kindafiction>



# План

- Что такое Kubernetes?
- Почему k8s?
- Как ~~загрузить~~ запустить локально?
- Примеры из жизни
- Если не хватает возможностей из коробки? (CRD, Operator, Istio)
- Можно ли проще? (Helm)

# K8s\*

## Что это такое?

- K8s - Оркестратор контейнерной нагрузки
- OpenSource
- Go
- от греческого - «рулевой»

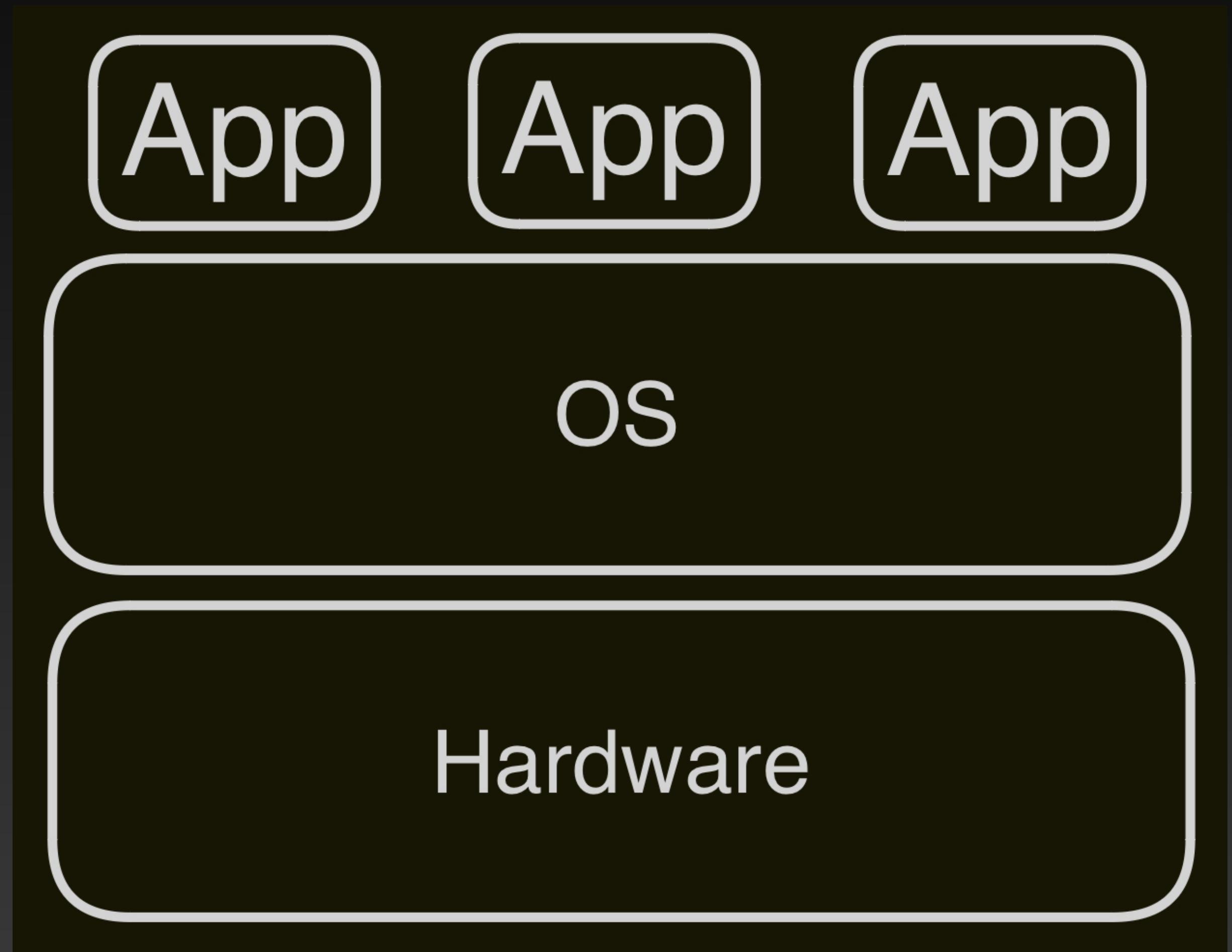


\*k(ubernetes)

**А как до него жили?**

# K8s

## Что было до?



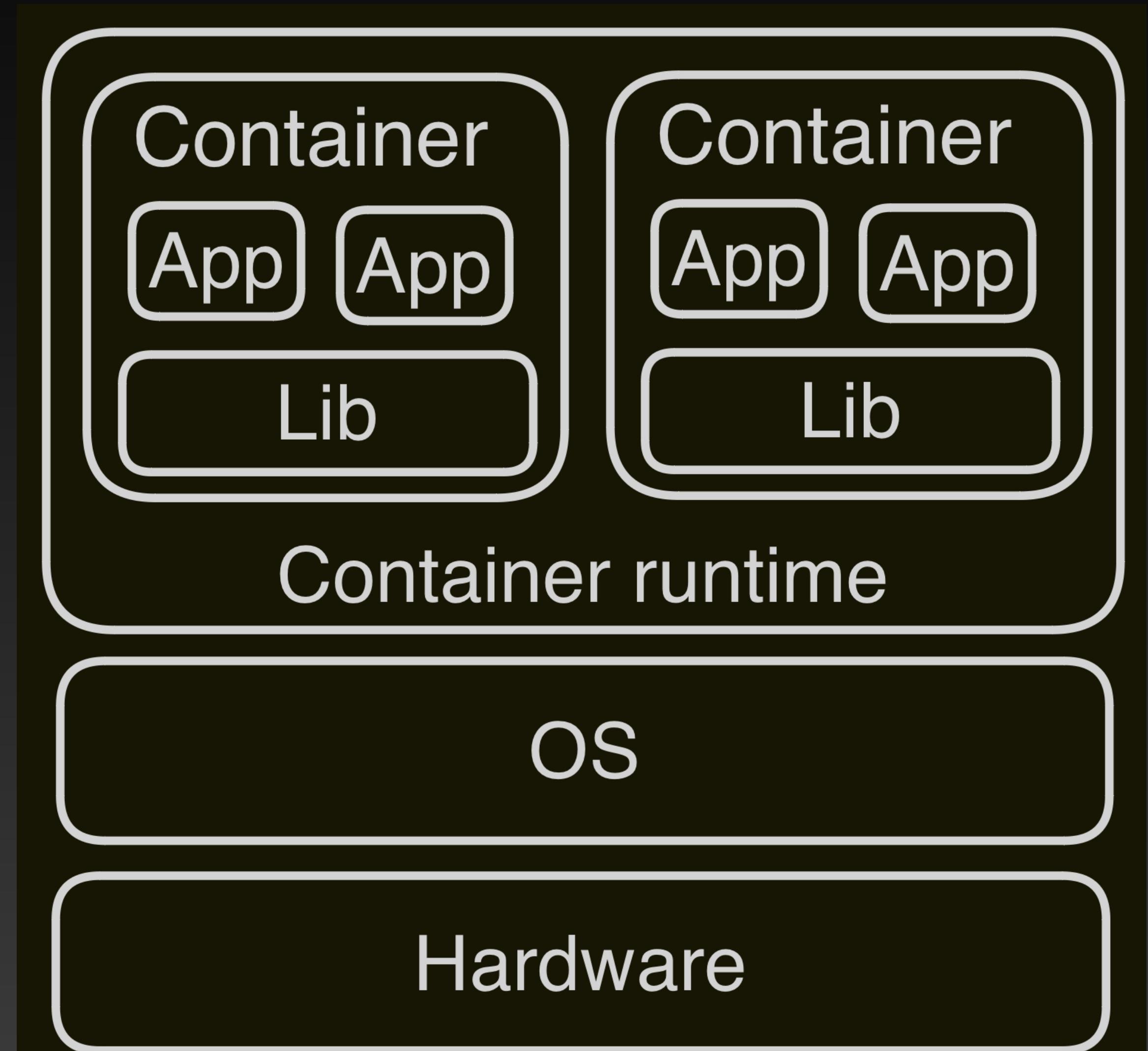
# K8s

Что было до?



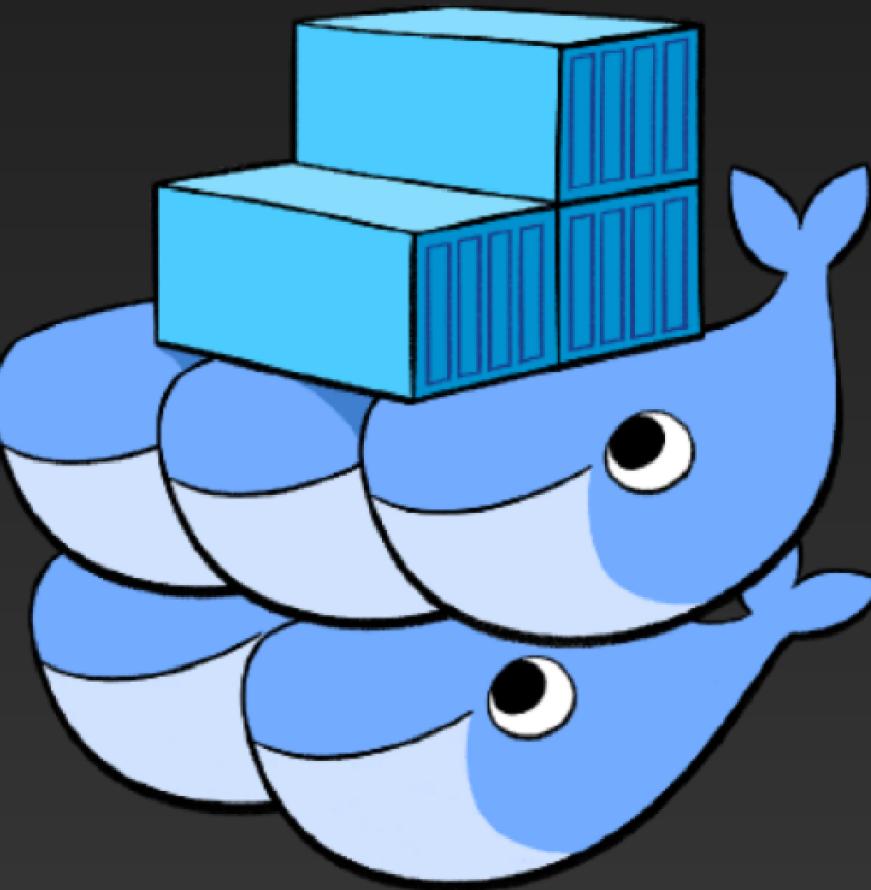
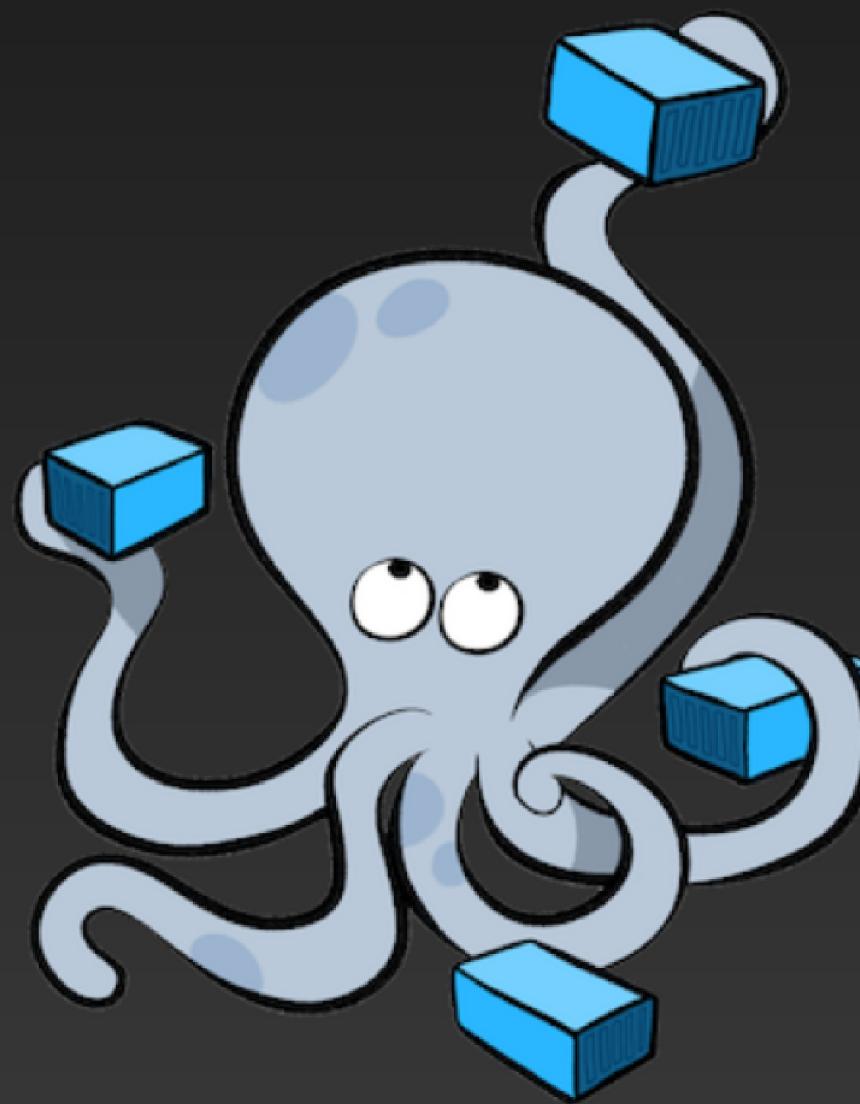
# K8s

Что было до?



# K8s

## Что было до?



# Docker

## Плюсы

- Легковесный
- Низкий порог входления
- Низкий ТТМ

# Docker

## Минусы

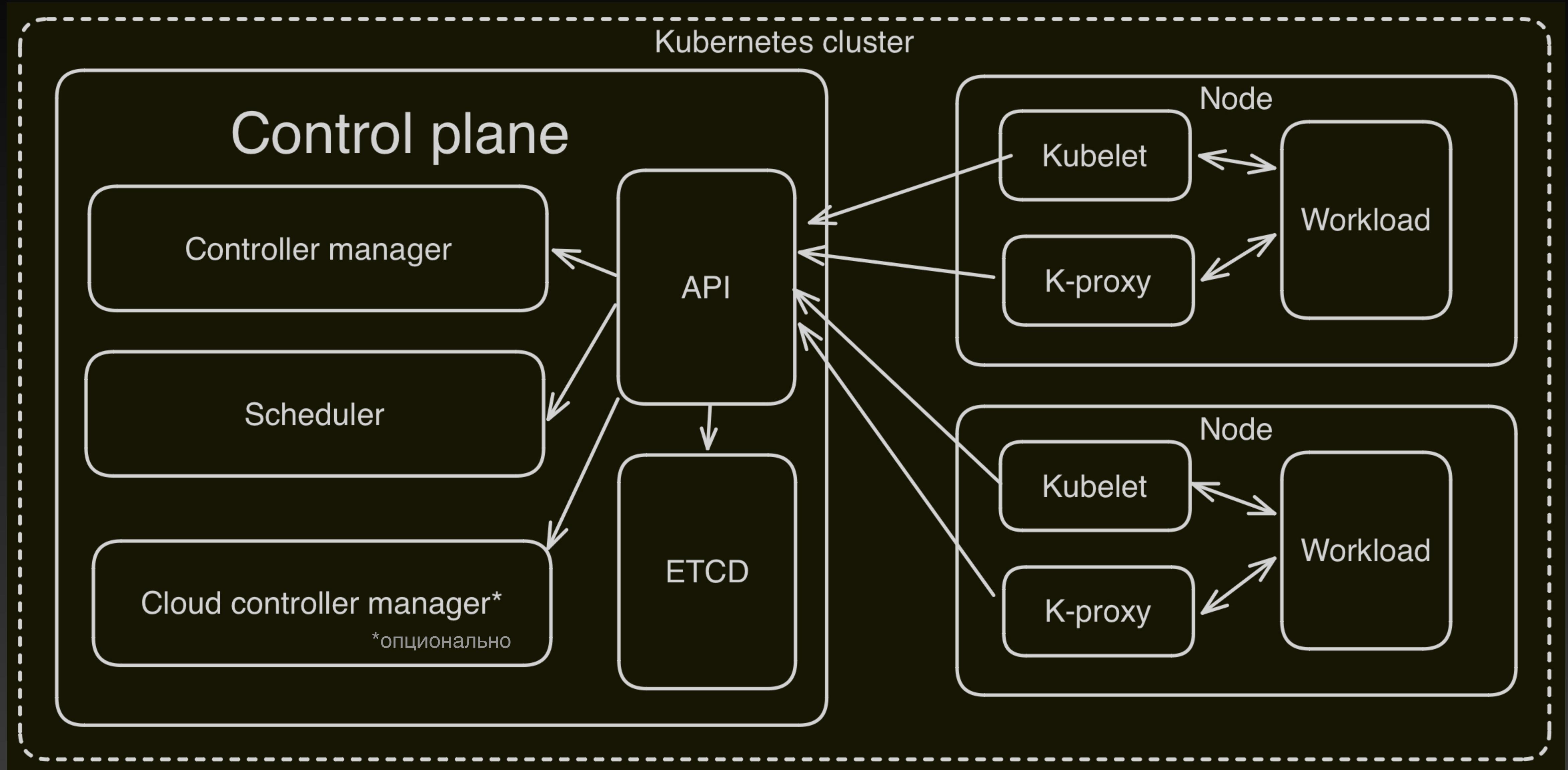
- Экспоненциальный рост сложности поддержки конфигурации с ростом количества сервисов
- Единая точка отказа
- Отсутствие расширяемости функционала

# K8s

## Почему?

- Ответ 1 - примерно константная сложность конфигурации
- Ответ 2 - отказоустойчивость и масштабируемость
- Ответ 3 - почти не ограниченные возможности по расширению стандартного функционала
- Ответ 4 - Google (опыт Borg) + сообщество

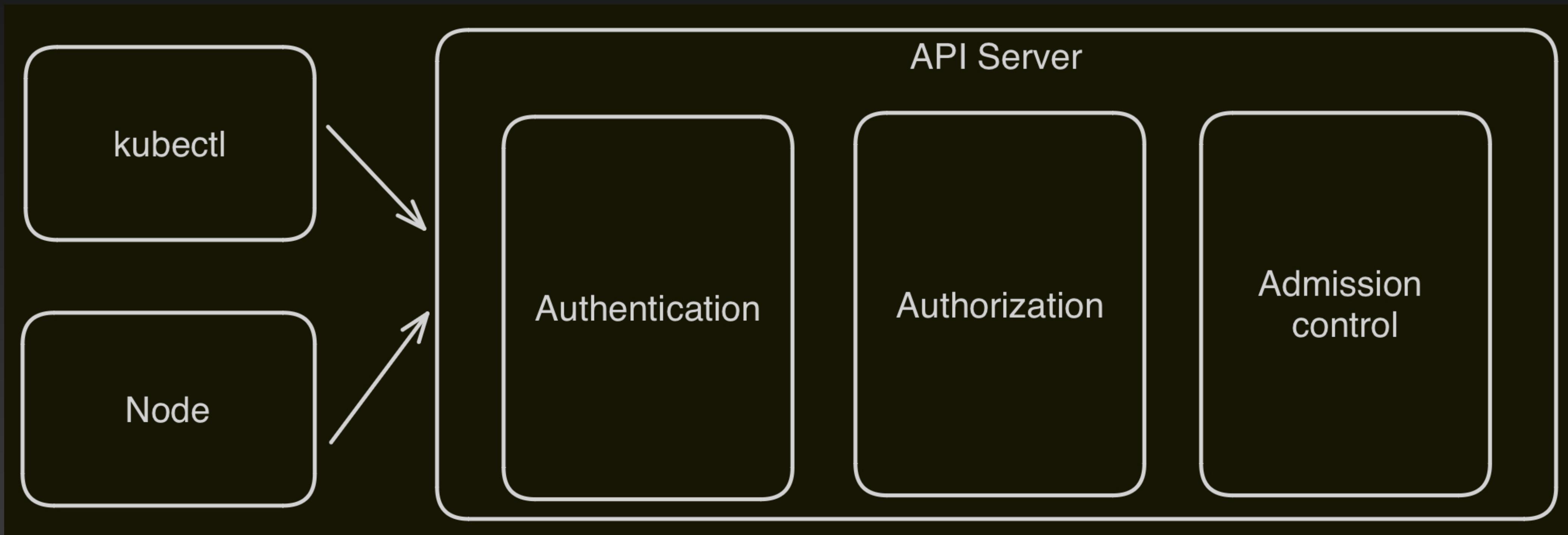
# K8s



# Control plane

## API

- API сервер
- Узел взаимодействия Control plane, рабочих узлов и пользователей



# Control plane

## Controller manager

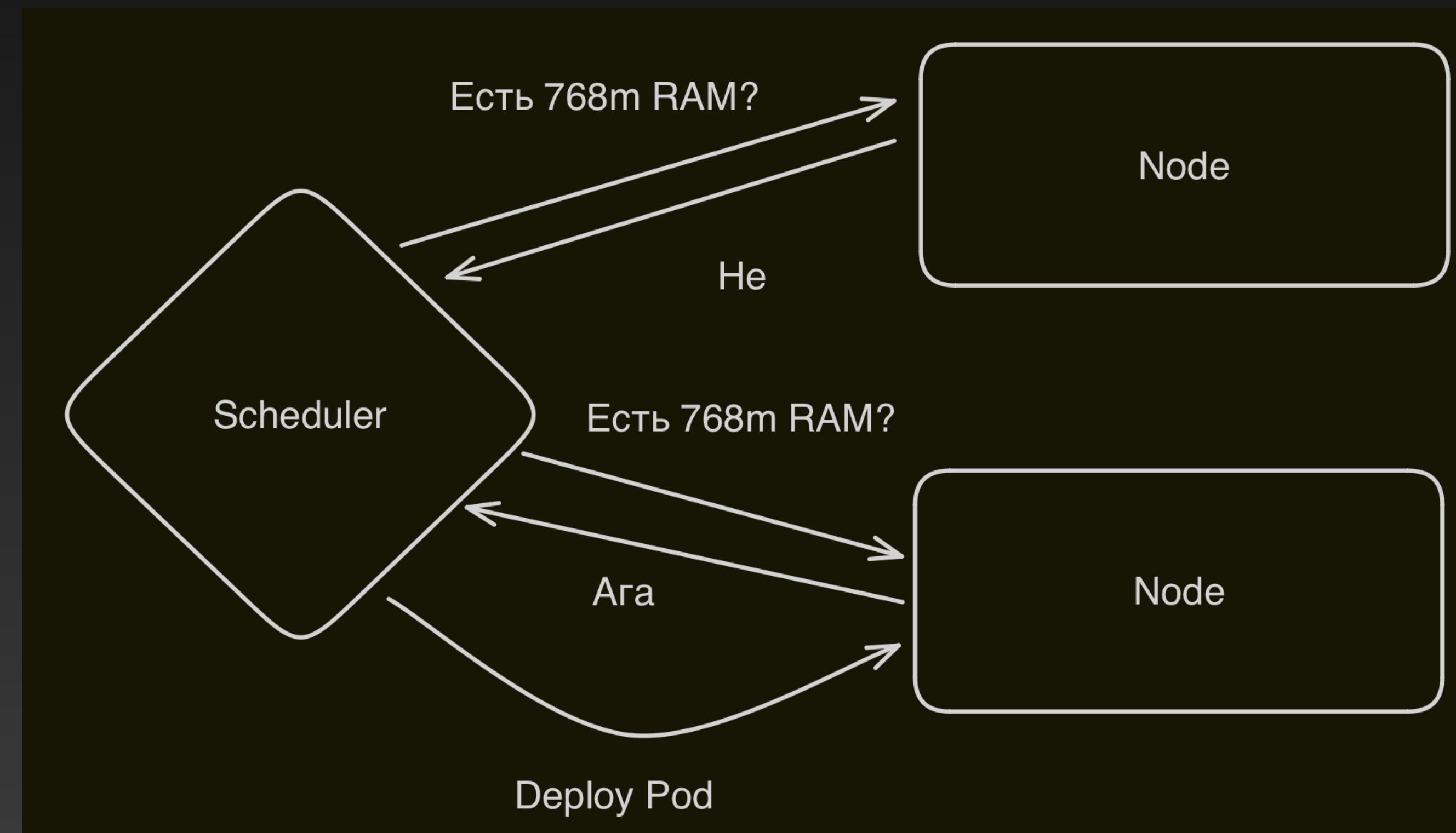
- Логическое объединение нескольких управляющих сущностей (Job controller, Node controller)



# Control plane

## Scheduler

- Назначение узла на котором будет работать Pod, по большому количеству параметров



# Control plane

## ETCD

- Распределенное, высокодоступное KV хранилище



# **Control plane**

## Cloud controller manager

- Взаимодействие с API Cloud провайдера
- Разделение зон ответственности



# Node Kubelet

- Агент, задача которого убедится что контейнеры в Pod'ах выполняются корректно



# Node Kube-proxy

- Отвечает за сетевое взаимодействие узлов



**И как это все запустить  
локально?**

# Local K8s

	KOs	K3s/K3d	Kind	MicroK8s	Minikube	CRC - Code Ready Containers
Разработчик/ компания	Mirantis	Rancher Labs	kubernetes-sigs	Canonical	Kubernetes	Red Hat
Требования к железу	Низкие	Низкие	Никие	Средние	Средние	Средние
Сложность установки	Просто	Просто	Просто	Средне	Средне	Средне
Особенности	Крайне легковесный, минимальный функционал	Поддержка нескольких нод	Kubernetes IN Docker	Аддоны и поддержка нескольких нод	Аддоны, быстрый старт	Локальный OpenShift

**Запустили, что дальше?**

# Объекты k8s

## Философия объектов

- Что и на какой ноде запускаем?
- Какие выделяем ресурсы?
- Политики отказоустойчивости и жизненного цикла

**Объект =  
желаемое  
состояние**

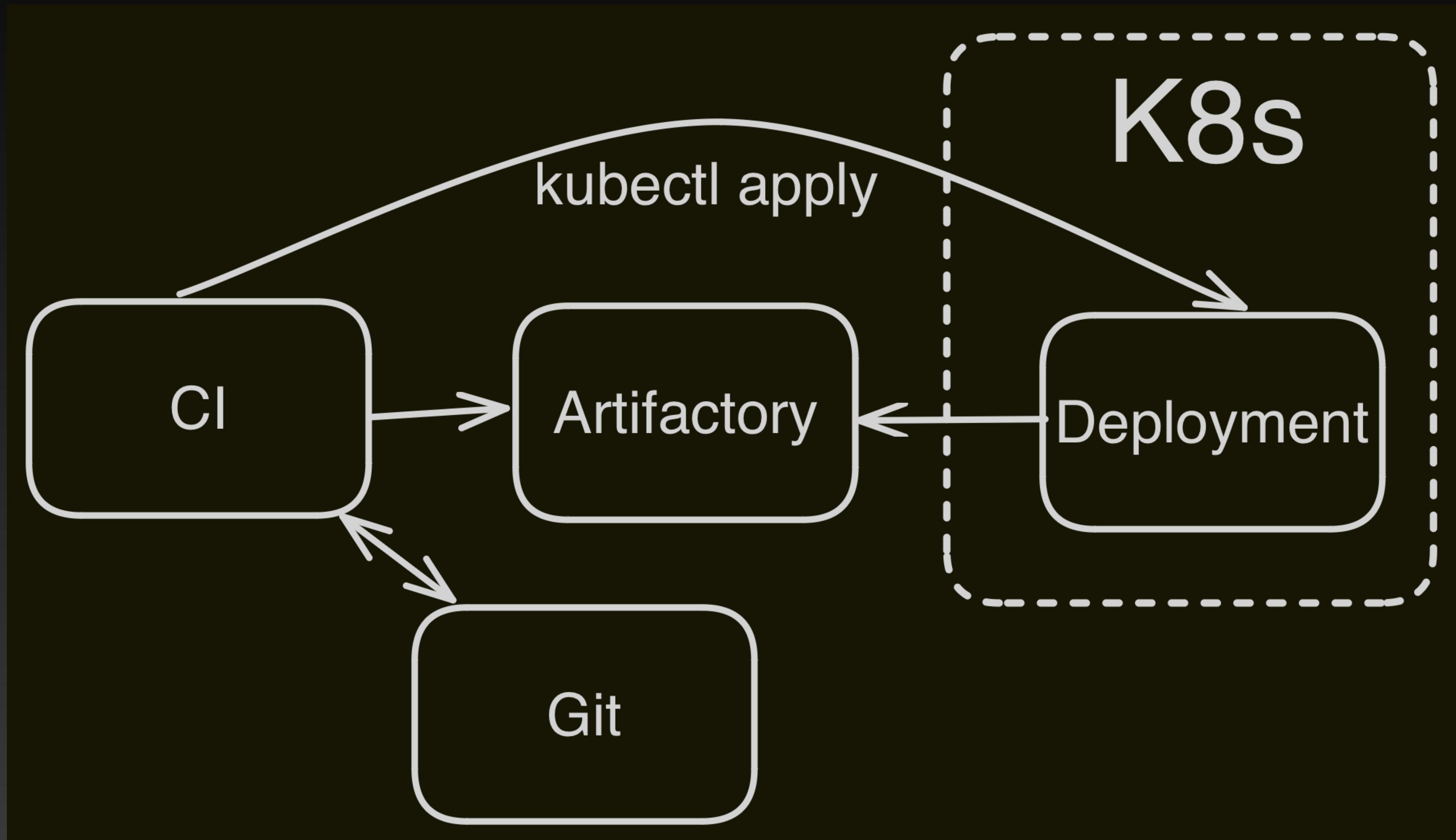
# Объекты K8s

## Манифест

```
kind: Service
apiVersion: v1
metadata:
  name: back-for-front
spec:
```

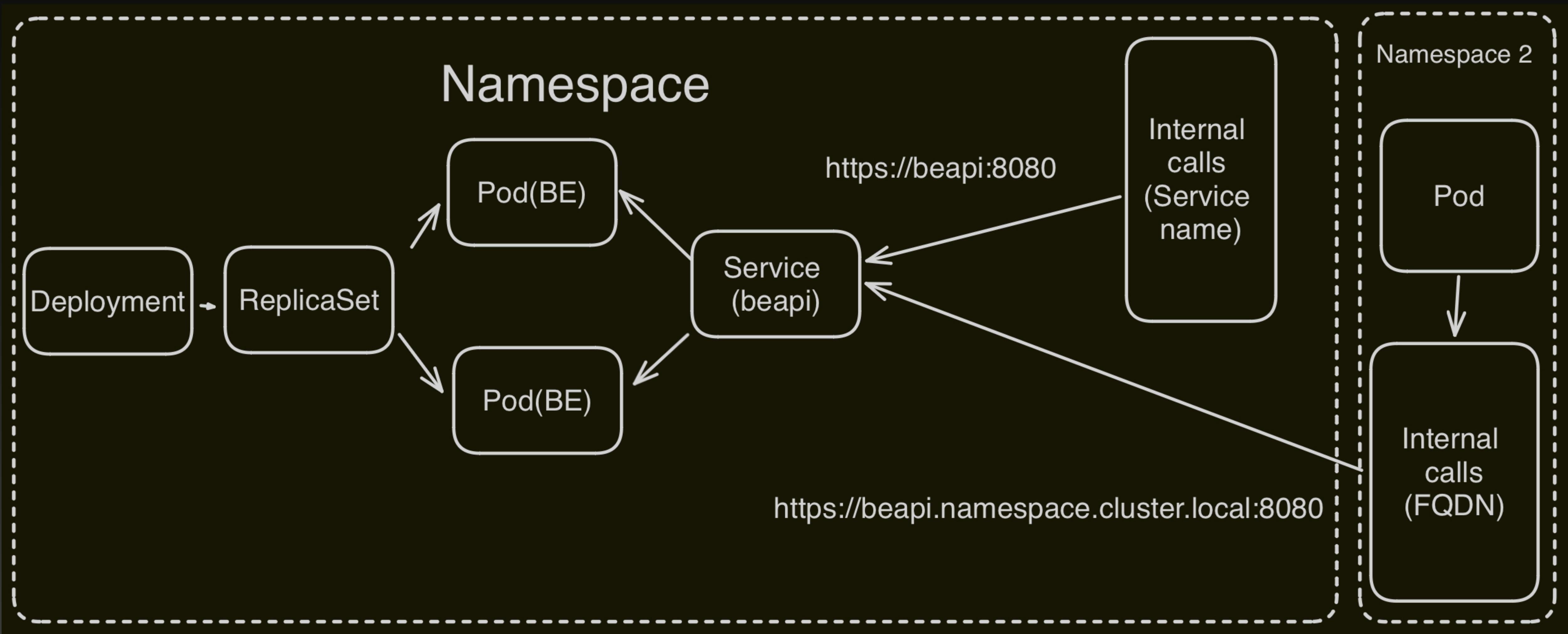
# Объекты K8s

## Deployment



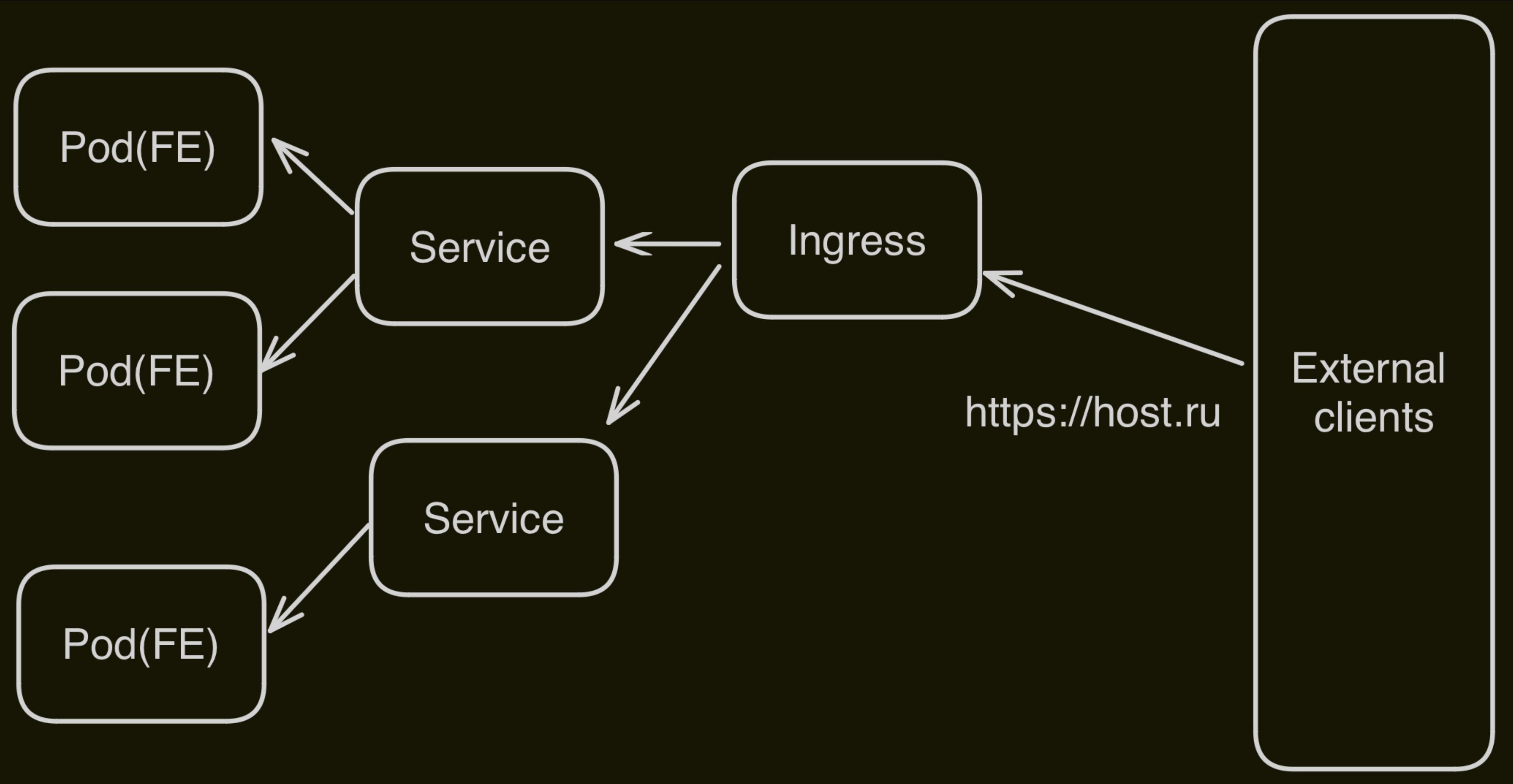
# Объекты K8s

## Backend



# Объекты K8s

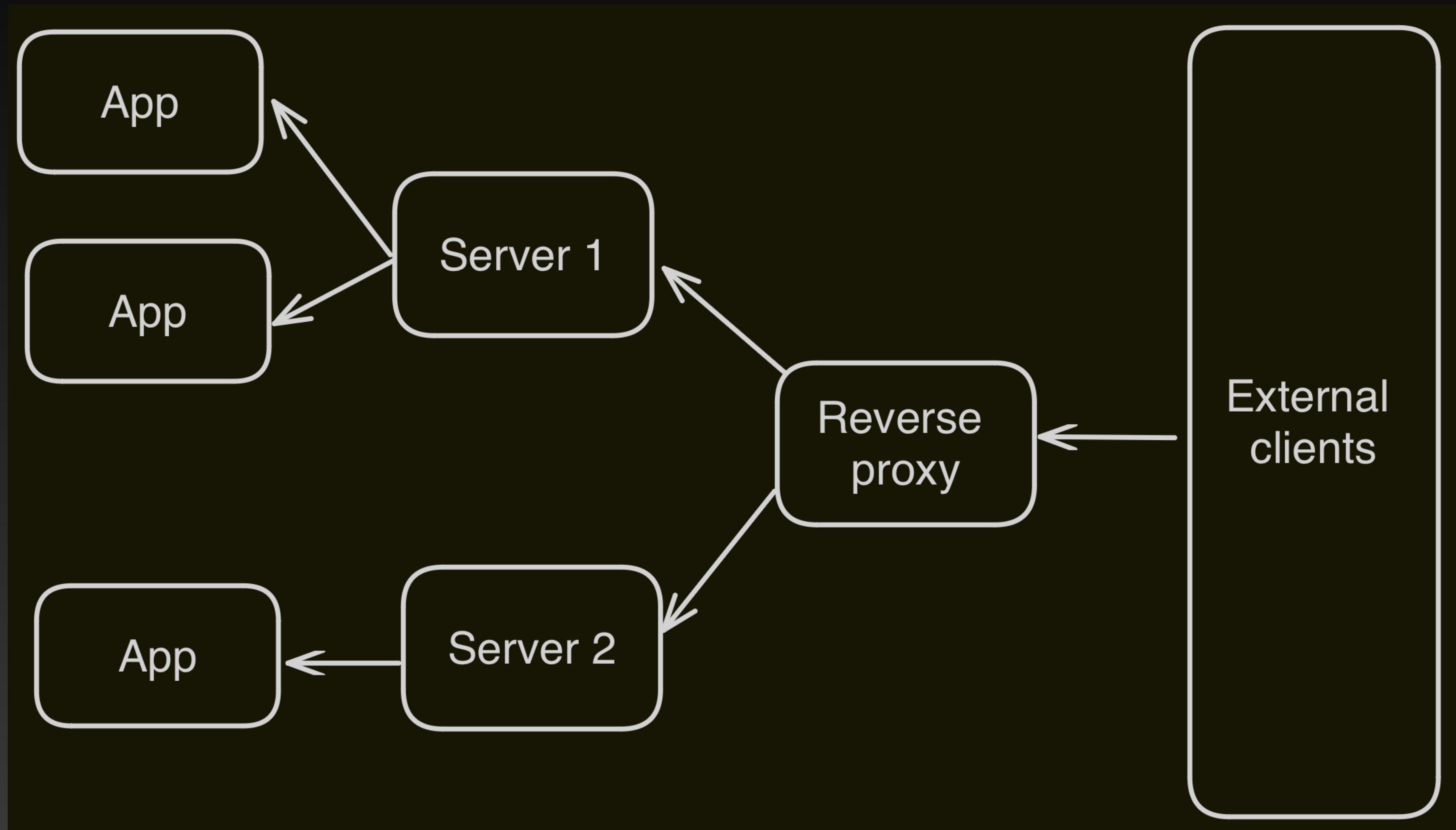
## Frontend



**Ничего не напоминает?**

# Объекты K8s

## Reverse proxy?



# Reverse proxy

- Nginx:
  - Web Server который прекрасно справляется с балансировкой нагрузки и проксированием
- Envoy:
  - Динамический API
- Traefik:
  - Cloud Native Application Proxy
- HAProxy:
  - Load Balancer / Reverse Proxy

<https://nginx.org/ru/>

<https://www.envoyproxy.io/>

<https://www.haproxy.org/>

<https://traefik.io>

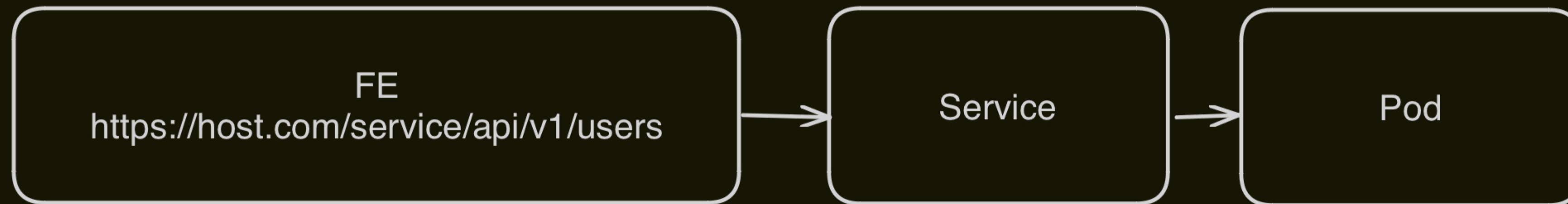
<https://osmosx.github.io/posts/nginx-envoy-haproxy-traefik/>

**Нужно писать nginx/envoy....  
конфигурации?**

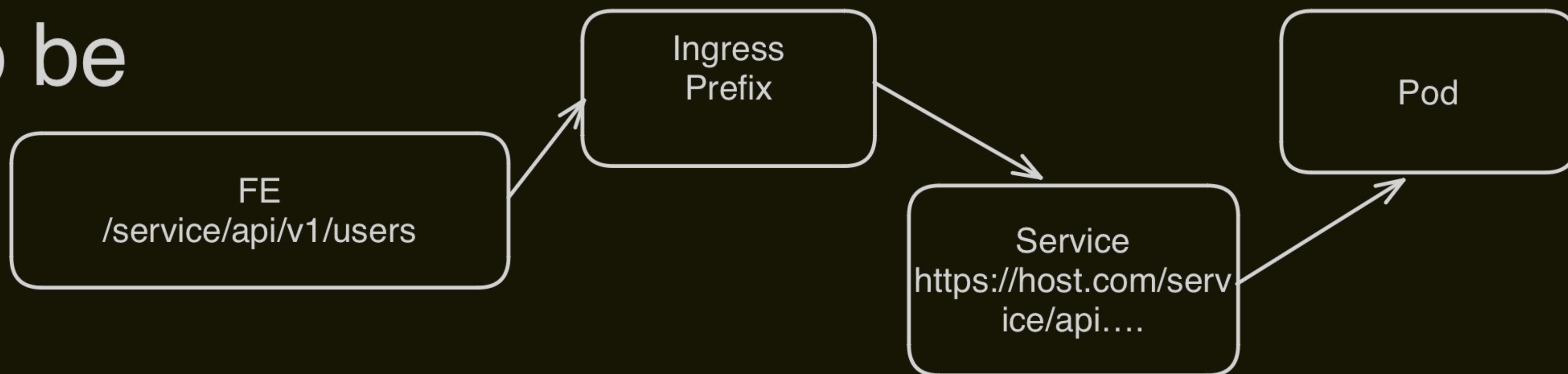
# Примеры из рабочей практики

## Роутинг запросов

As is



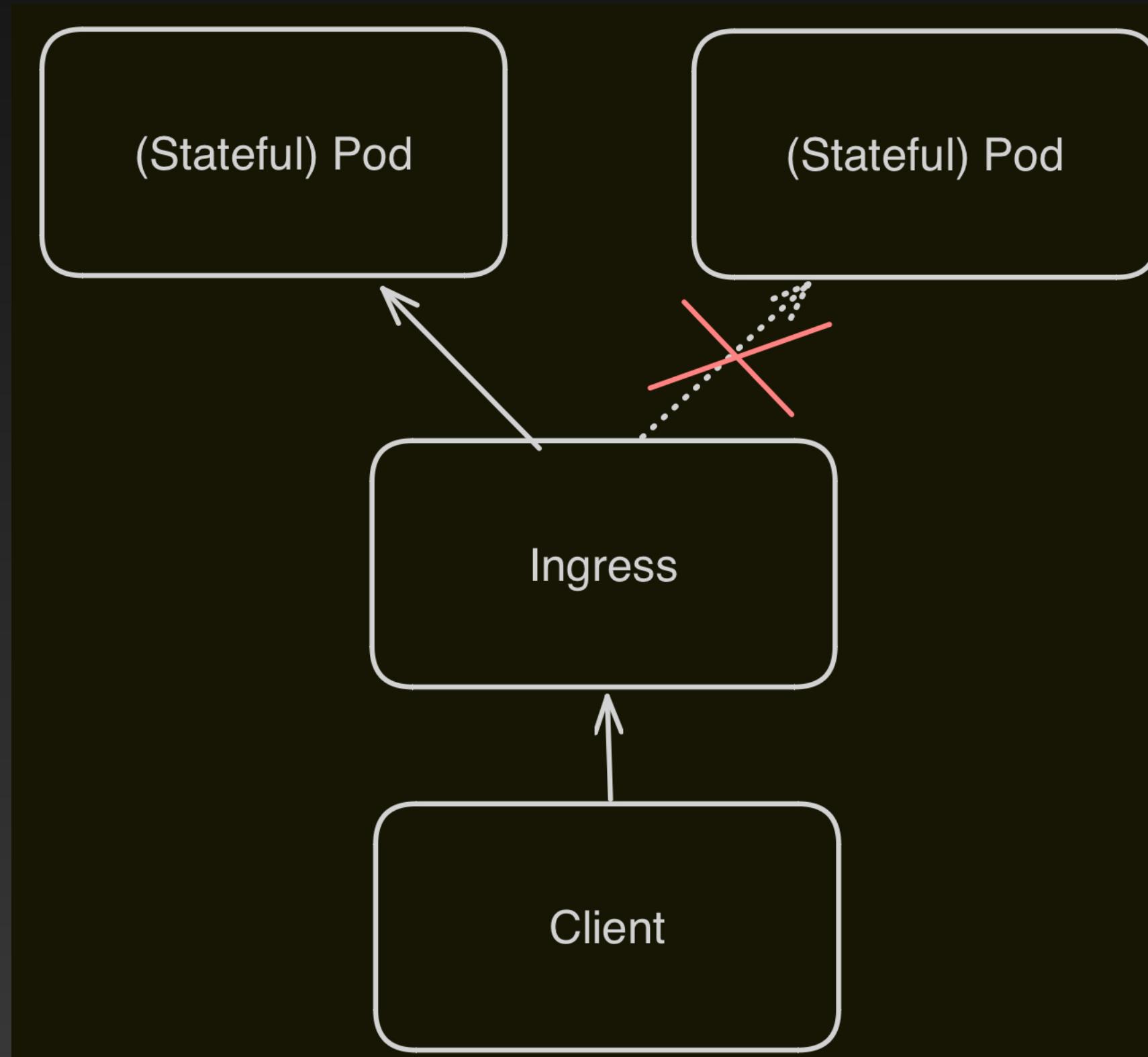
To be



# Примеры из рабочей практики

## Sticky-session

```
nginx.ingress.kubernetes.io/affinity: "true"  
nginx.ingress.kubernetes.io/affinity-mode: "persistent"  
nginx.ingress.kubernetes.io/session-cookie-name: "my-cookie"
```

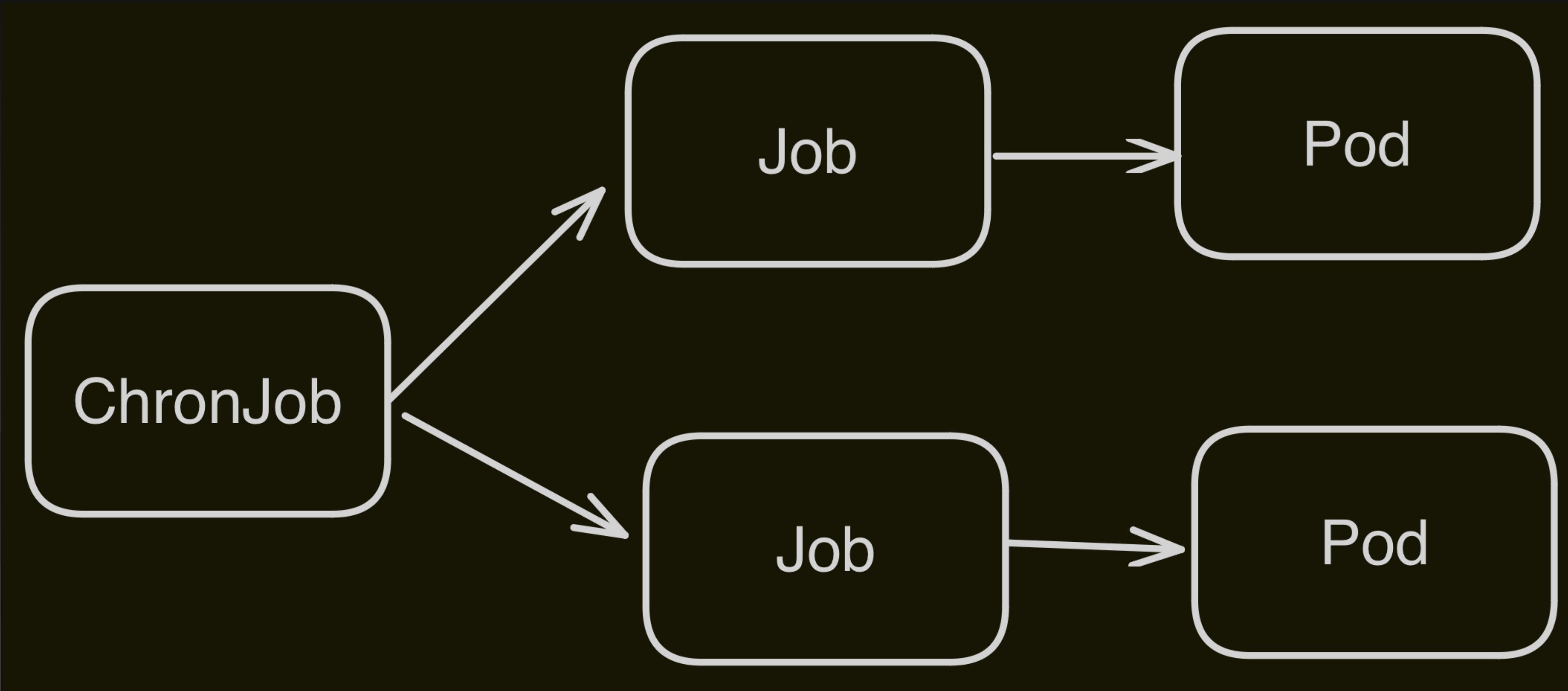


**У меня есть несколько приложений на  
Windows Server, запускаемых  
планировщиком. Может ли K8s так?**

# Примеры из рабочей практики

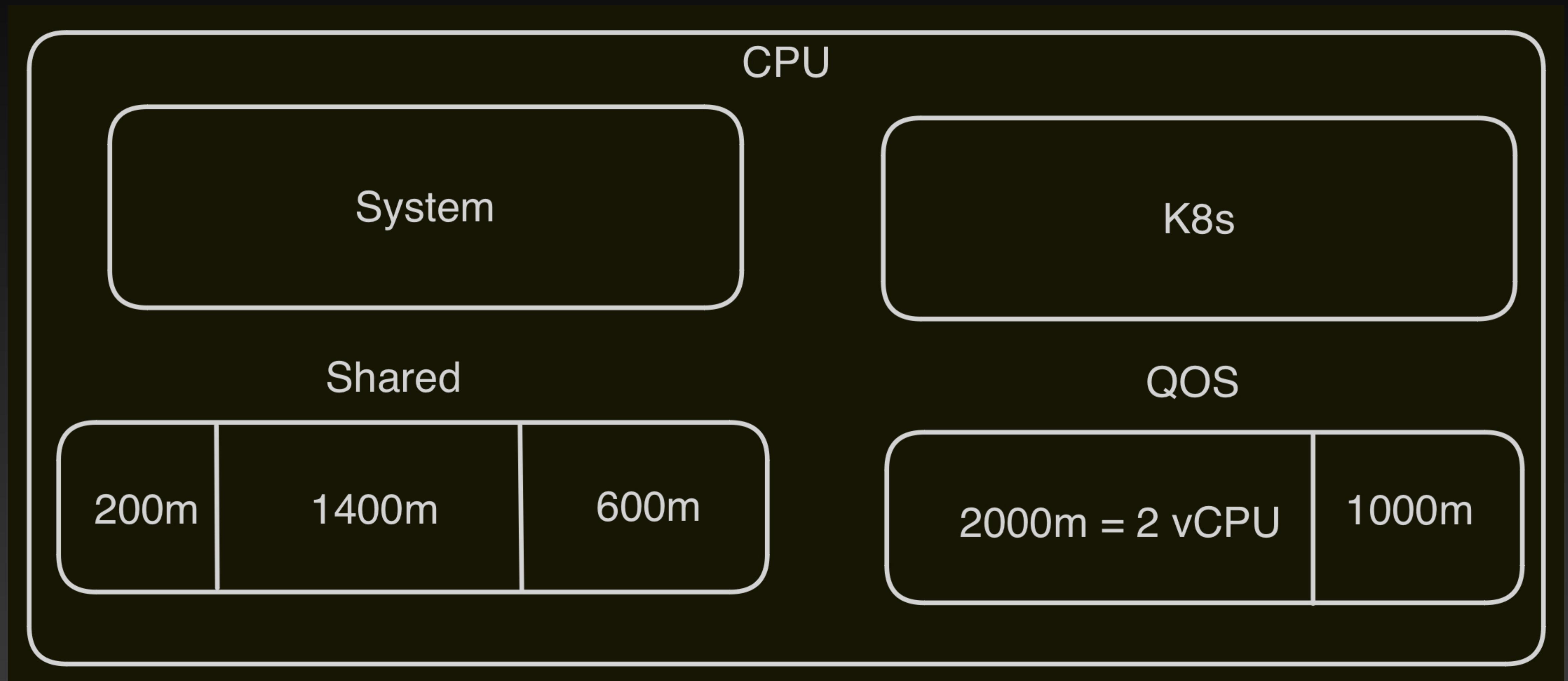
## Job

```
schedule: "*/5 * * * *"
```



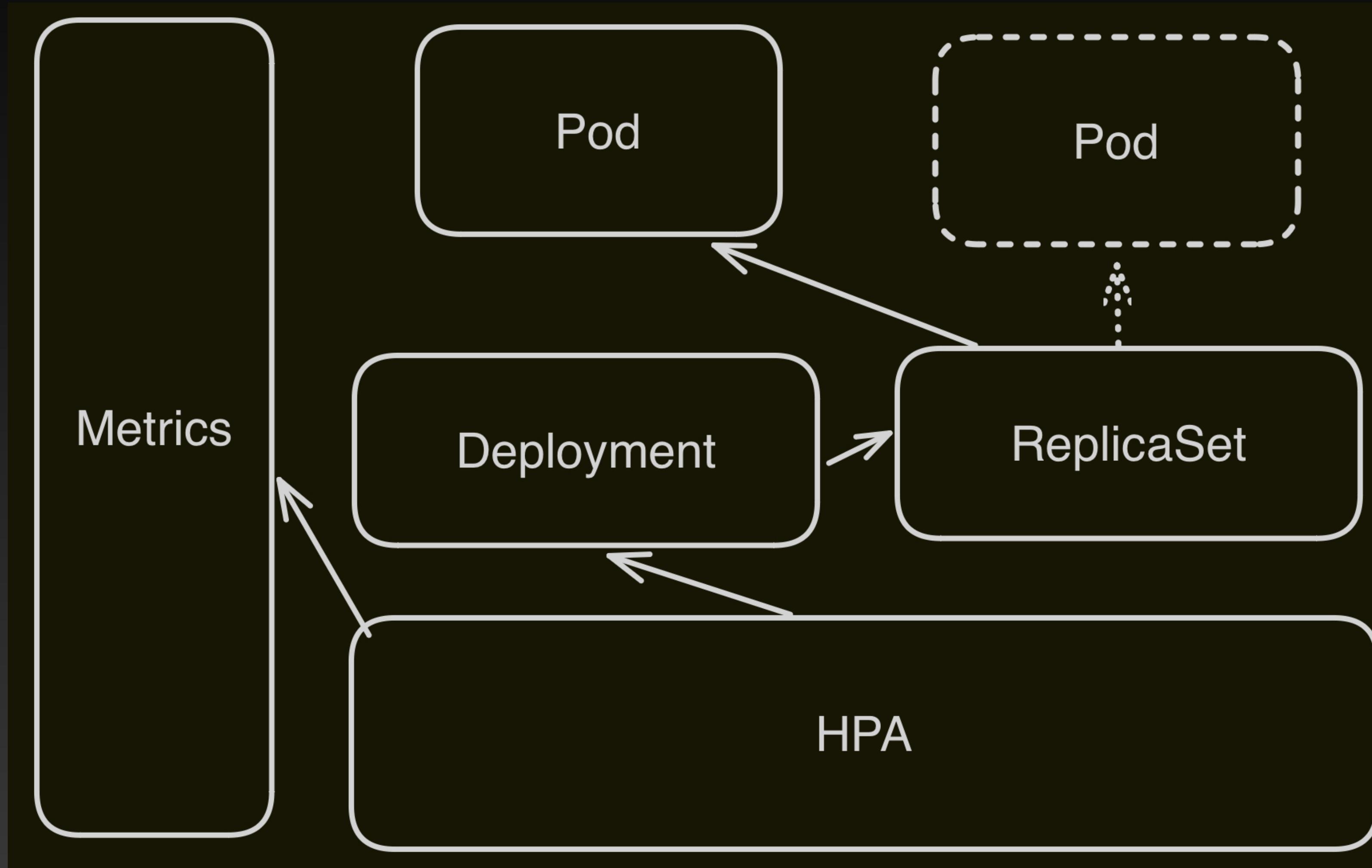
**А как управлять ресурсами?**

# CPU в K8s



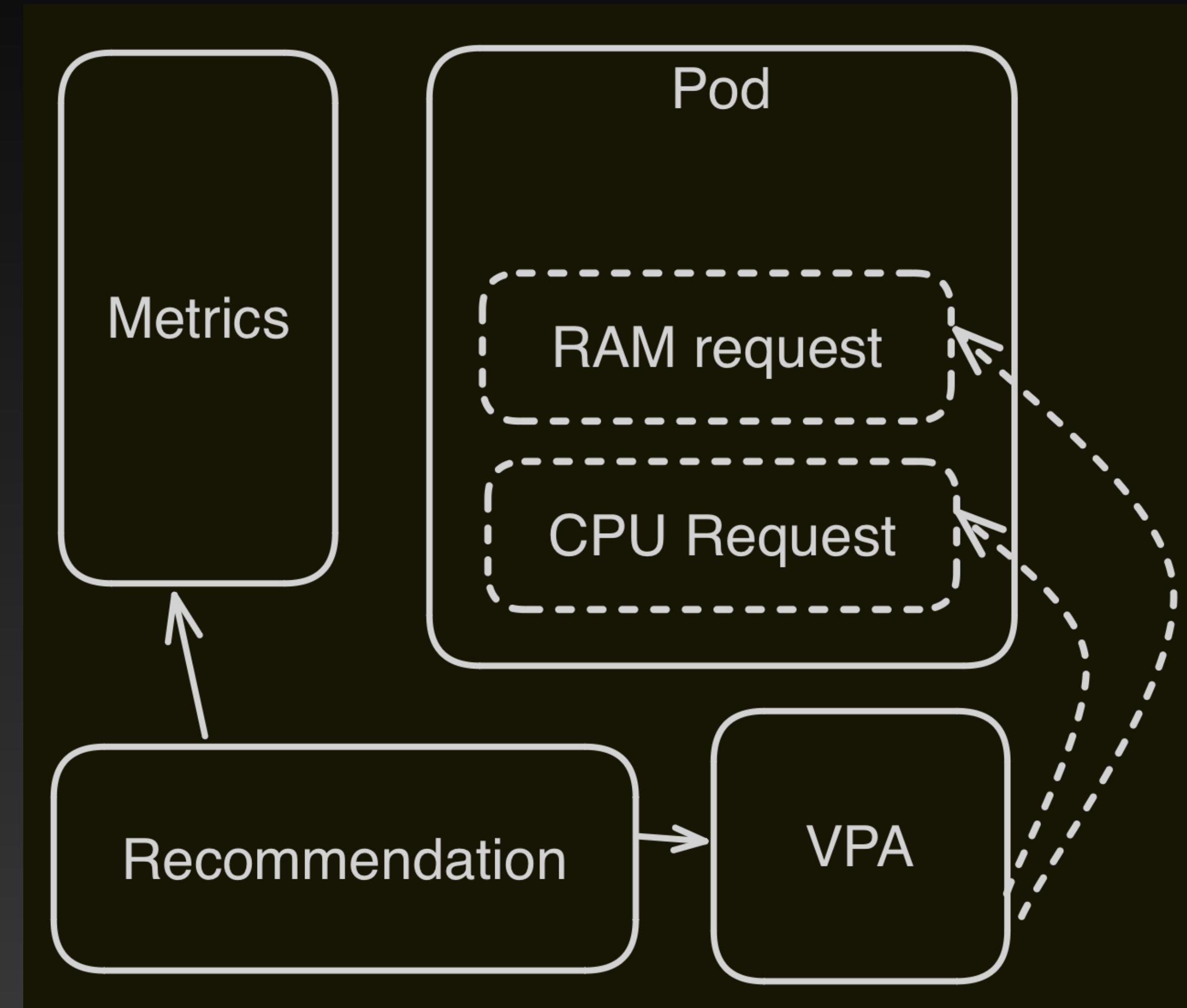
# Примеры из рабочей практики

## НРА (Horizontal Pod Autoscaler)



# Примеры из рабочей практики

## VPA (Vertical Pod Autoscaler)



**С ресурсами понятно, а есть ли возможность  
управлять жизненным циклом?**

# Примеры из рабочей практики

## Graceful shutdown

```
lifecycle:
```

```
  preStop:
```

```
    exec:
```

```
      command:
```

```
        - sh
```

```
        - ' -c '
```

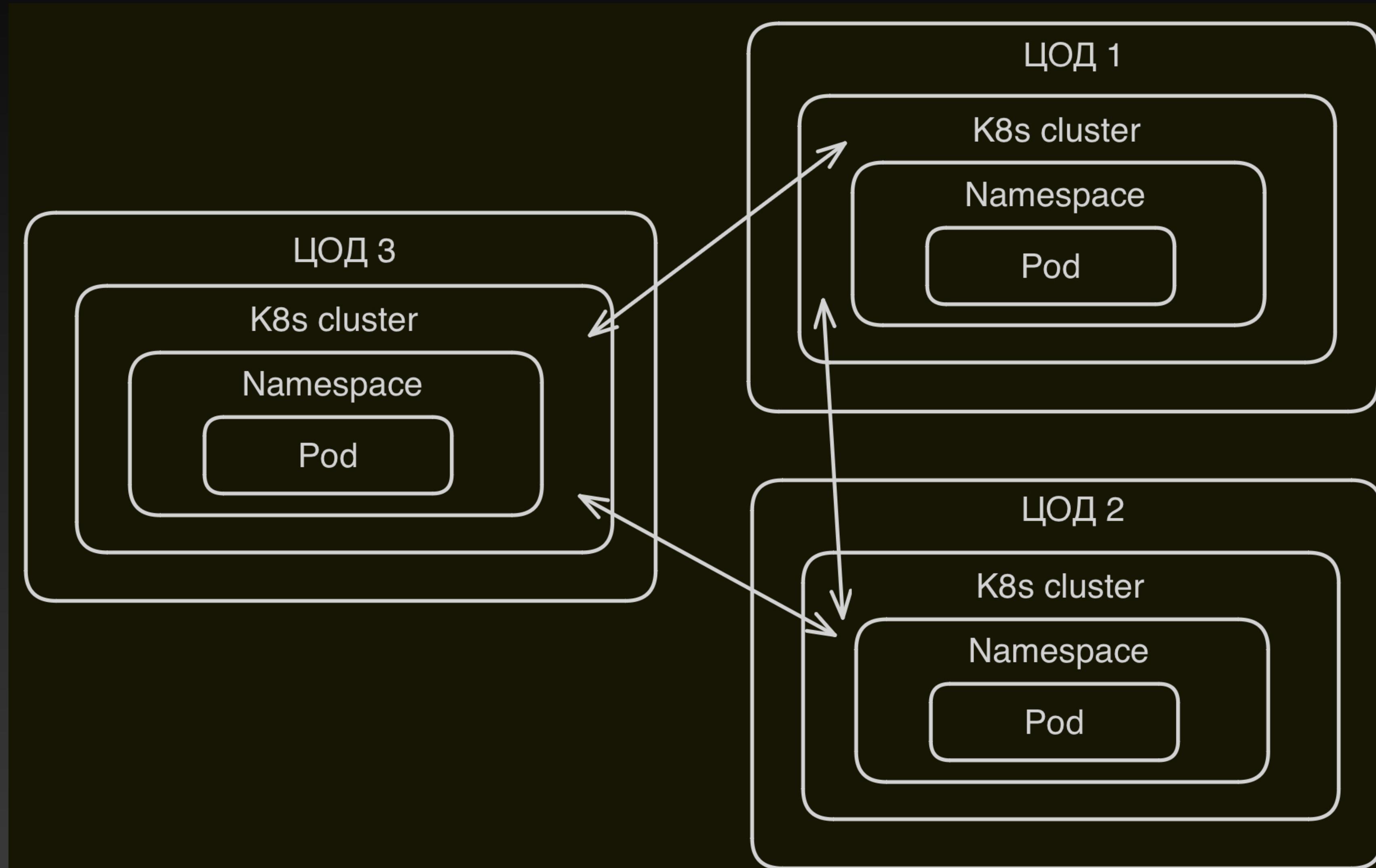
```
        - sleep 30
```

```
terminationGracePeriodSeconds: 30
```

**А как дела с  
отказоустойчивостью?**

# Примеры из рабочей практики

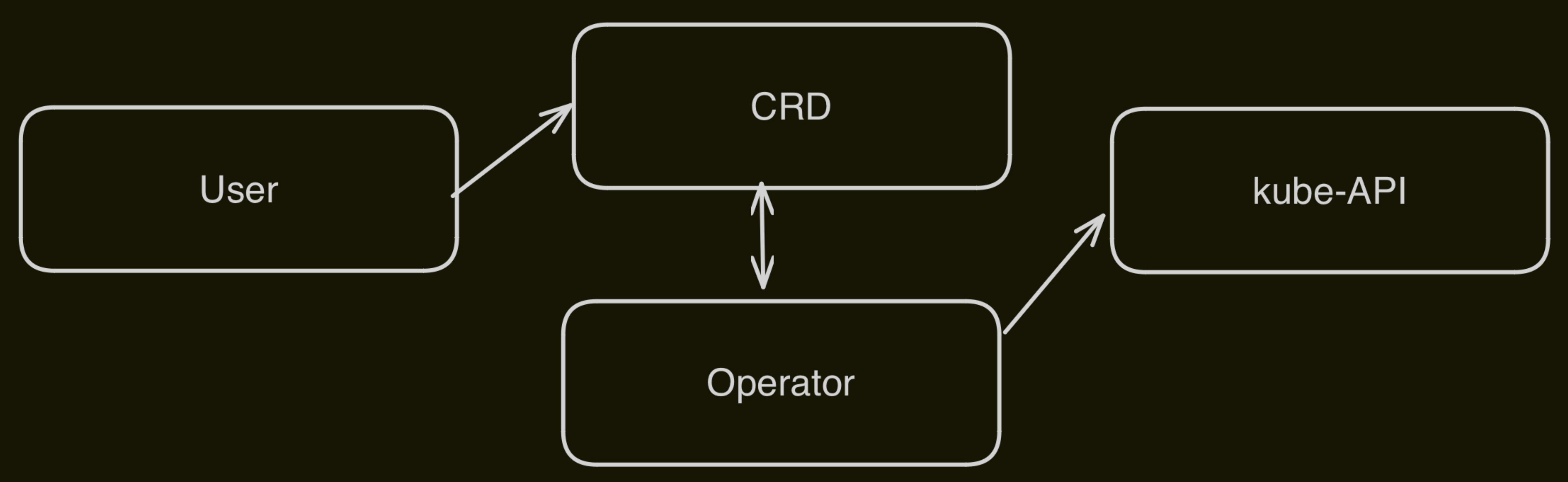
## Геораспределение



**Не хватает функционала  
встроенных объектов?**

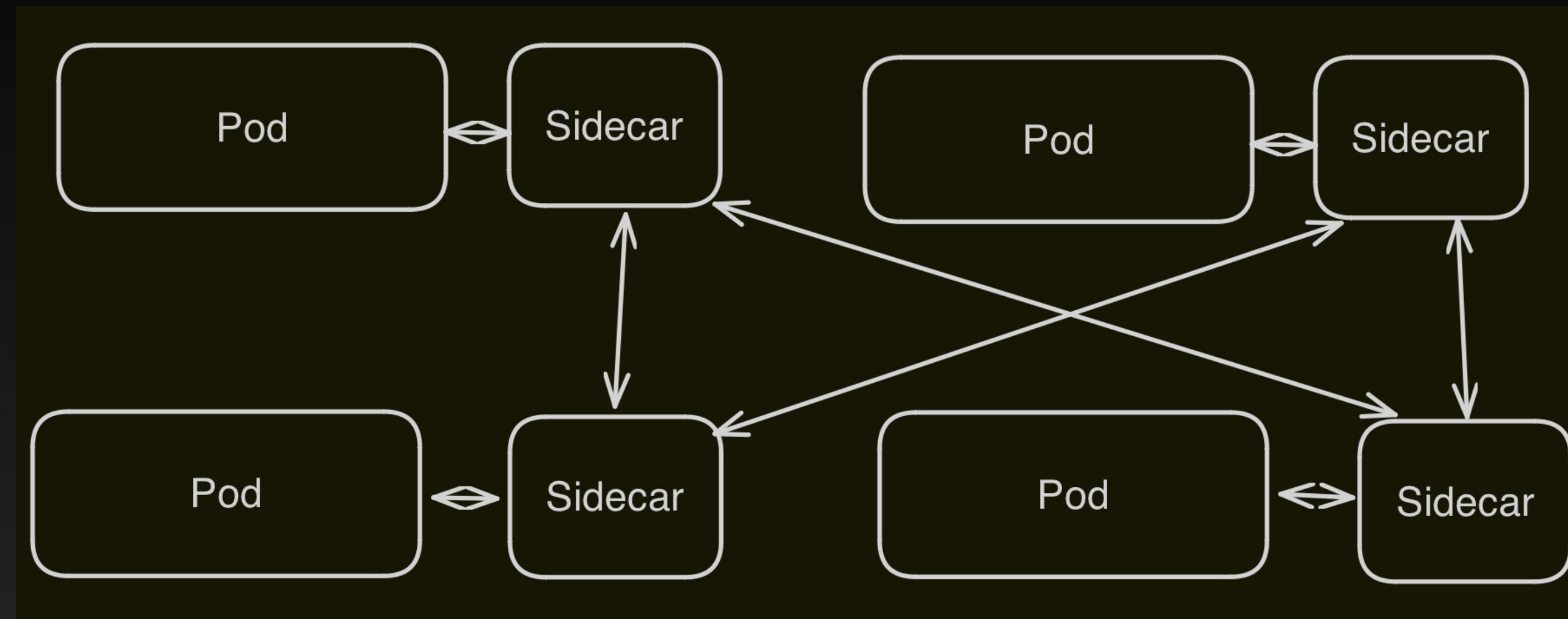
пишем своё

# CRD (Custom Resource Definitions) + Operator



**Пользуемся готовыми  
решениями**

# Service mesh



<https://www.youtube.com/watch?v=9CUfaeT3T-A>  
<https://www.youtube.com/watch?v=S-WxeCDDels>  
[https://www.youtube.com/watch?v=8oj3rKBR\\_Rs](https://www.youtube.com/watch?v=8oj3rKBR_Rs)

# Istio

## Дополнительные возможности

- Timeout
- Retry
- Circuit breaker
- Blue-green
- Canary deployment
- AB tests
- Multi-cluster

<https://habr.com/ru/companies/oleg-bunin/articles/493026/>

[https://blog.skillfactory.ru/glossary/a-b-testirovanie/.](https://blog.skillfactory.ru/glossary/a-b-testirovanie/)

<https://medium.com/@kirill.sereda/стратегии-обработки-ошибок-circuit-breaker-pattern-650232944e37>

# Istio

## Плата за них

- Не самая простая конфигурация
- Каждый sidecar потребляет CPU и RAM
- Sidecar «хранит всю таблицу маршрутизации»

**Сложно как-то, можно  
попроще?**



# Helm

## Менеджер пакетов для k8s

```
containers:  
- name: deis-database  
  image: {{ .Values.imageRegistry }}/postgres:{{ .Values.dockerTag }}  
  imagePullPolicy: {{ .Values.pullPolicy }}  
  ports:  
    - containerPort: 5432  
  env:  
    - name: DATABASE_STORAGE  
      value: {{ default "minio" .Values.storage }}
```

```
imageRegistry: "quay.io/deis"  
dockerTag: "latest"  
pullPolicy: "Always"  
storage: "s3"
```

values.yaml

Template

# Итого

- K8s - не только для DevOps
- K8s != запустил приложение и забыл
- Знание инструментов позволяет решать более широкий спектр задач и даёт конкурентное преимущество на рынке
- Helm облегчает работу с k8s
- Istio - нужно взвесить все за и против

Спасибо за внимание!

