

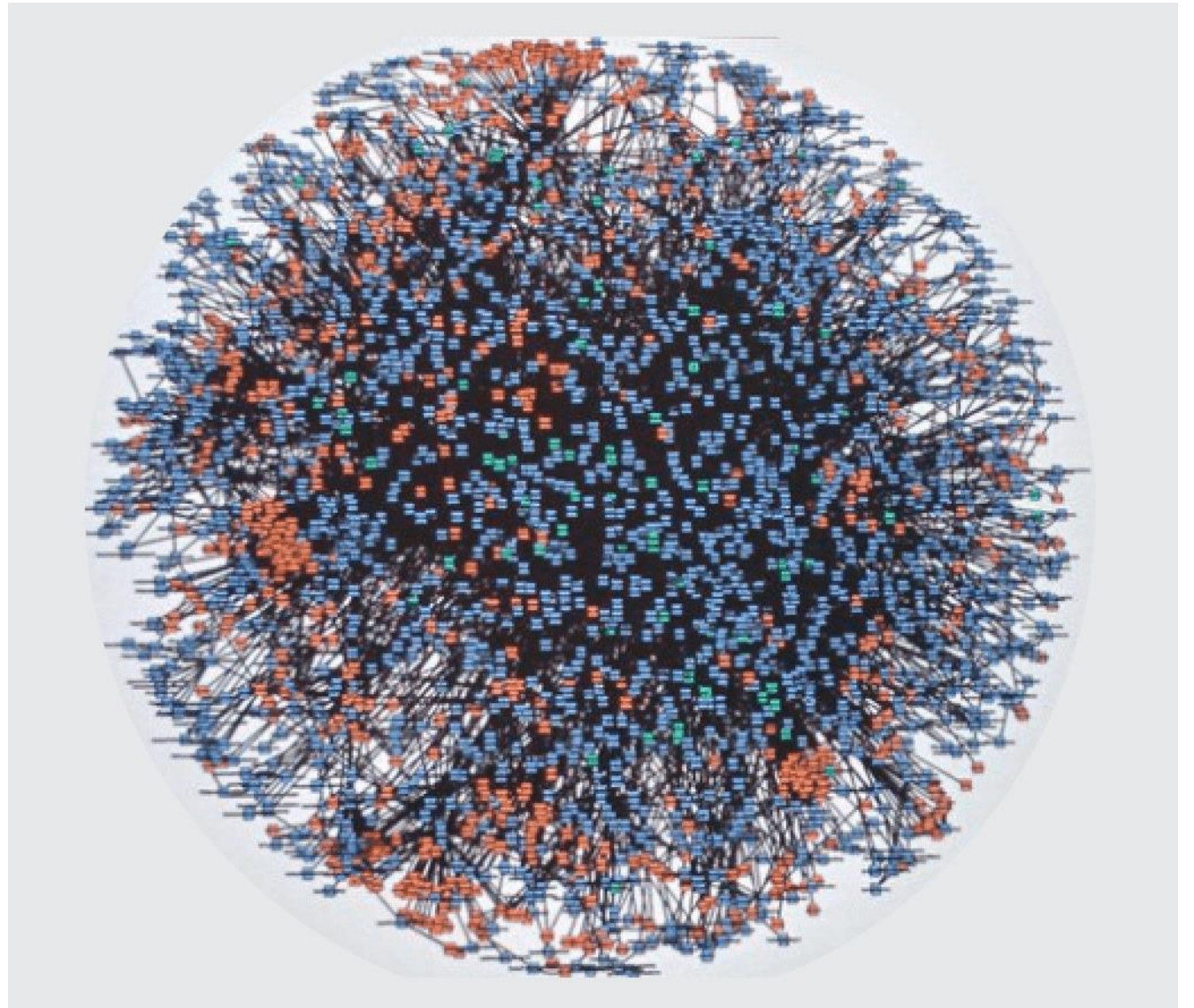
Kubernetes

Семинар 9

Микросервисы

Сложно рассчитать
потребление ресурсов

Сложно аллоцировать
приложения



Pets VS Cattle



- Уникален и неповторим
- Если болеет, то лечим
- Имеет особое имя (например, Осирис)



- Все одинаковые
- Если болеет, то избавляемся и заменяем на такого же
- Нет имени, только идентификаторы (например, d433fsd2)

Оркестратор



Что делает оркестратор?

- Аллокация – куда кого поставить
- Реплицирование / масштабирование
- Проверка готовности сервиса
- Воскрешение
- Перепланирование (rescheduling)
- Управление сервисами
 - Service discovery
 - Load balancing

Какие есть оркестраторы?

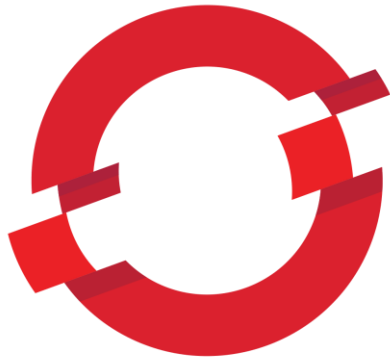


kubernetes

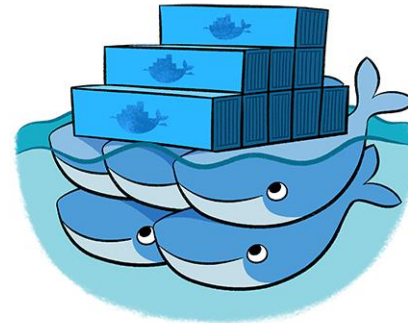


HashiCorp


























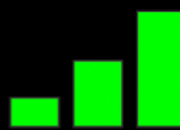
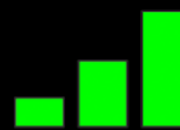










Nomad

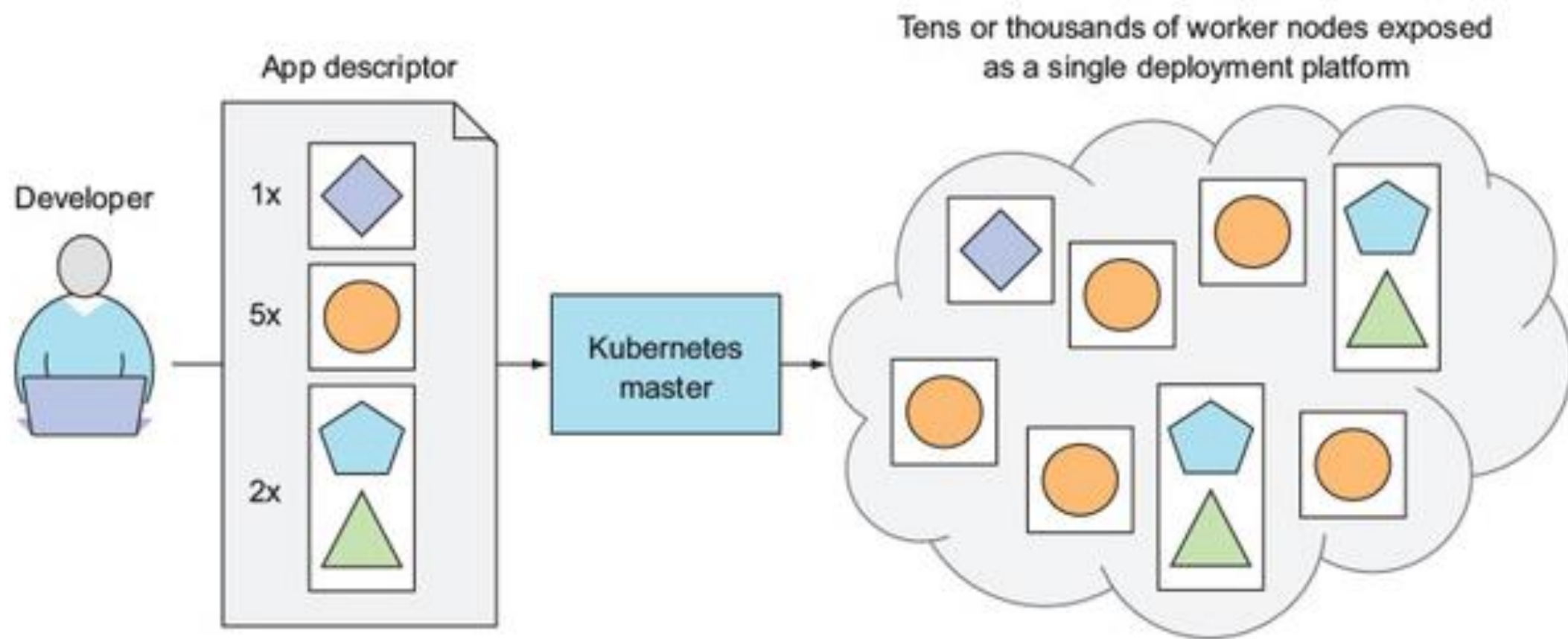


OPENSIFT



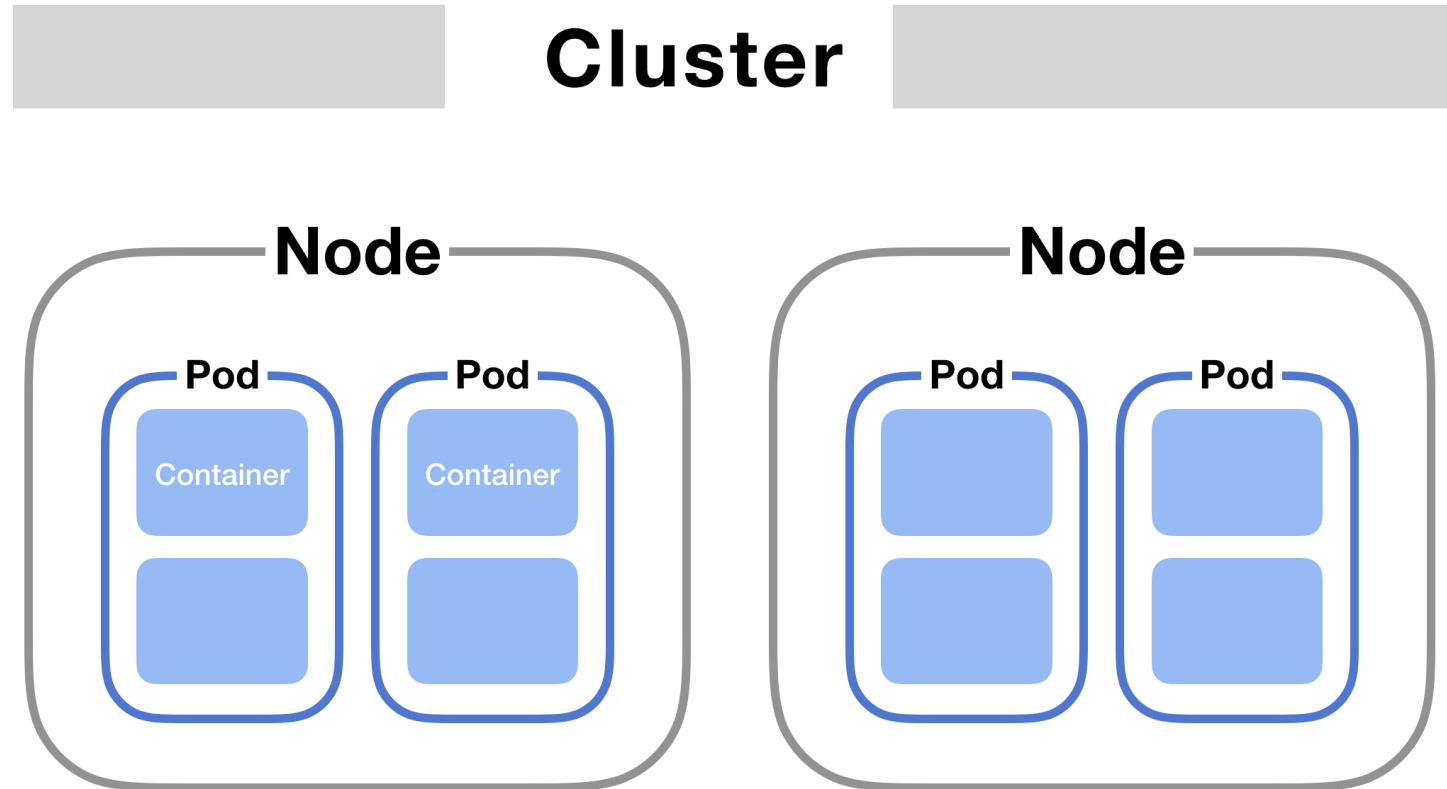
Почему кubernetes?

	Простота вхождения	Enterprise?	Функциона л из коробки	Расширяе мость	Отказоуст ойчивость	Доставка кода	IaC	Сообщество
								
								
 OPENSIFT								
		 						

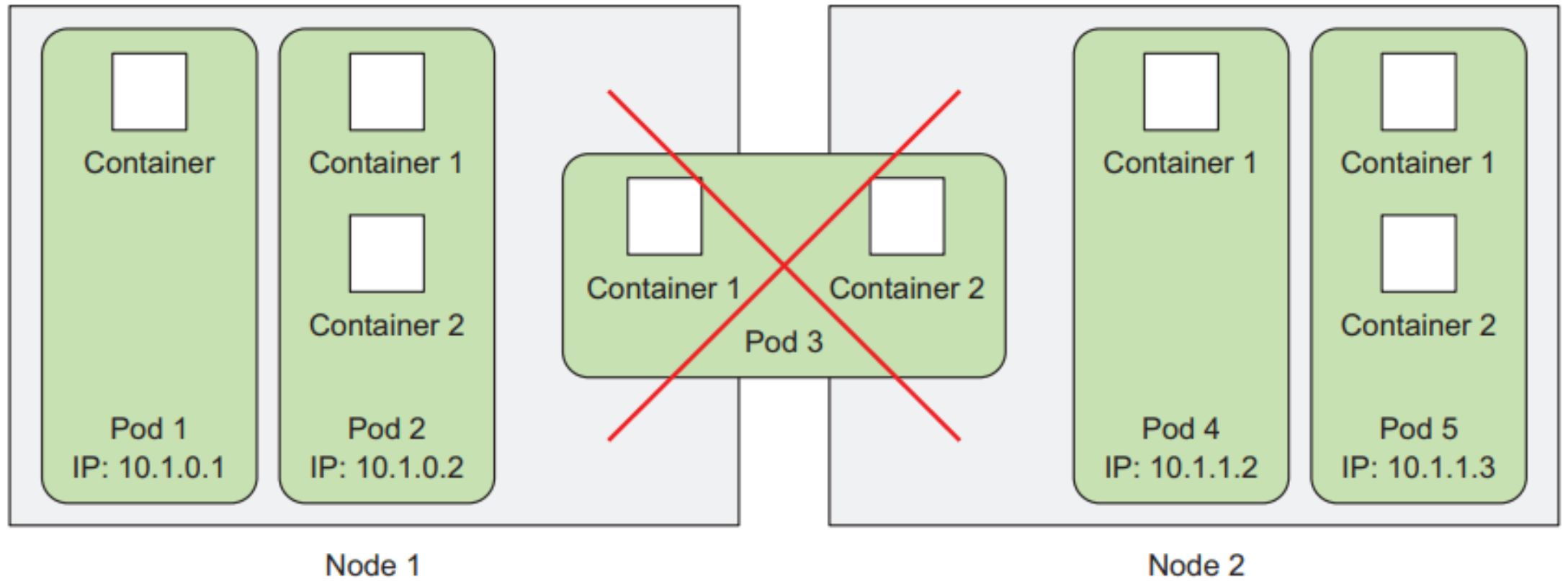


POD

- Минимальная сущность, управляемая Kubernetes
- Группа контейнеров (один или несколько)
- У всех контейнеров внутри одного POD общие network, PID, namespaces



POD



Манифест пода

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx
spec:
  containers:
  - name: nginx
    image: nginx:1.14.2
    ports:
    - containerPort: 80
```

apiVersion – используемая для создания объекта версия API Kubernetes

kind – тип объекта, который хотим создать

metadata – данные, позволяющие идентифицировать объект

spec – требуемое состояние объекта

Metadata

- Имя создаваемого объекта
- Метки (labels)
- Namespace
- Аннотации (annotations)

```
metadata:  
  name: annotations-demo  
  annotations:  
    imageregistry: "https://hub.docker.com/"
```

“

Метки предназначены для Kubernetes,
а аннотации — для людей!

”

Спец

- Что запускать
- С какими параметрами

```
spec:  
  containers:  
    - name: nginx  
      image: nginx:1.14.2  
      ports:  
        - containerPort: 80
```

Способы взаимодействия с k8s

- kubectl (command line tool)
- webUI
- API

Демо работы с POD

Работа с ресурсами

Реквесты – ресурсы, необходимые для работы контейнера

Суммарное значение всех реквестов не может выходить за физические ресурсы хоста

При определении хоста для контейнера учитываются реквесты

Если на хосте не хватит ресурсов, то kube-scheduler 100% не выберет данных хост

Работа с ресурсами

Лимиты – пороговые значения, которые может потреблять контейнер

При превышении заданных значений:

- Перезапуск в случае превышения по RAM
- Ограничение в потреблении в случае превышения по CPU

Суммарное значение limits всех pod на хосте может быть большим, чем количество физических ресурсов хоста

Демо работы с ресурсами

Readiness/liveness пробы

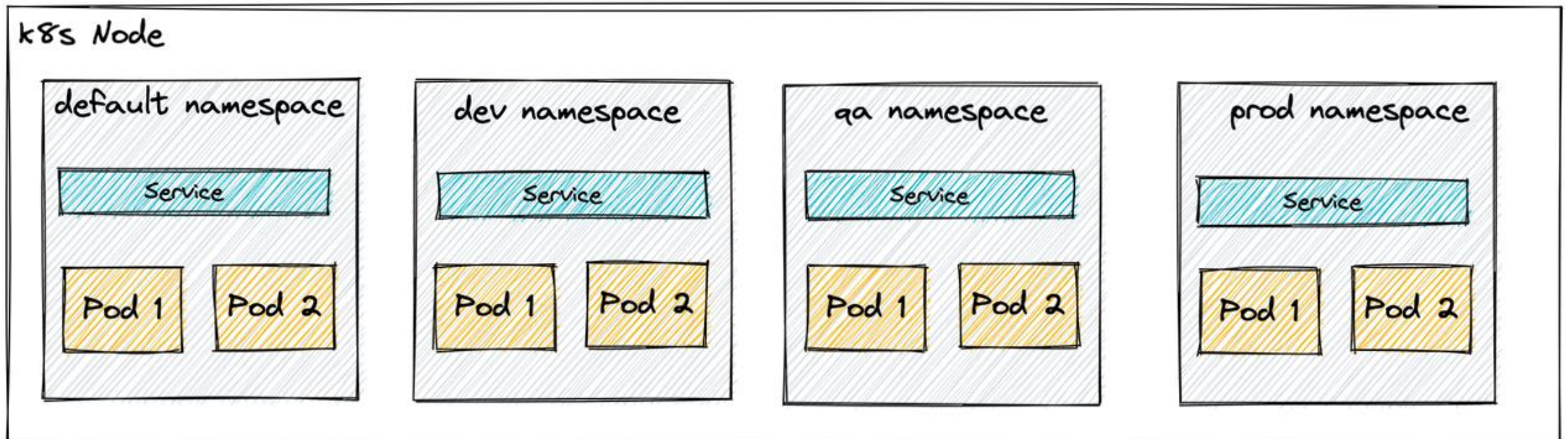
Механизм, позволяющий контролировать готовность приложения к работе

Демо readiness

Namespaces

Можно создать несколько виртуальных кластеров в рамках одного физического кластера

Namespace – один виртуальный кластер (одно окружение)



Namespaces

Применяется для:

1. Разграничения прав между командами
2. Лимитирования ресурсов на проект с помощью квот (cpu, memory)

Namespaces

- **default** – для объектов, у которых явно не определена принадлежность к namespace
- **kube-system** – для системных объектов k8s
- **kube-public** – для объектов, к которым нужен доступ из любой точки кластера

Демо namespaces

Конфигурирование приложений

```
x-airflow-common:
  &airflow-common
  build:
    context: .
    dockerfile: DockerfileAirflow
  depends_on:
    - postgres
  environment:
    - LOAD_EX=n
    - EXECUTOR=Local
    - AIRFLOW__CORE__EXECUTOR=LocalExecutor
    - AIRFLOW__CORE__SQL_ALCHEMY_CONN=postgresql+psycopg2://airflow:airflow@postgres:5432/airflow
    - AIRFLOW__WEBSERVER__SECRET_KEY=FB0o_zt4e3Ziq3LdUU07F2Z95cvFFx16hU8jTeR1ASM=
    - AIRFLOW__CORE__FERNET_KEY=FB0o_zt4e3Ziq3LdUU07F2Z95cvFFx16hU8jTeR1ASM=
    - AIRFLOW__CORE__LOAD_EXAMPLES=True
    - AIRFLOW__CORE__LOGGING_LEVEL=INFO
```

Демо конфигурирования

Конфигмапа

Стандартный способ хранения конфигураций в k8s

Смотрим демо

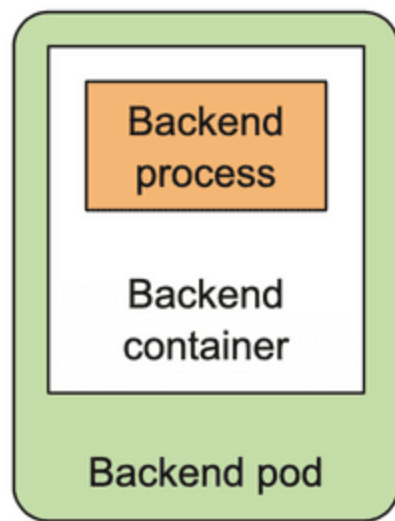
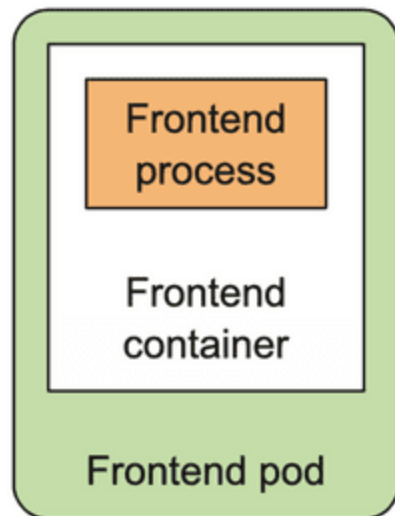
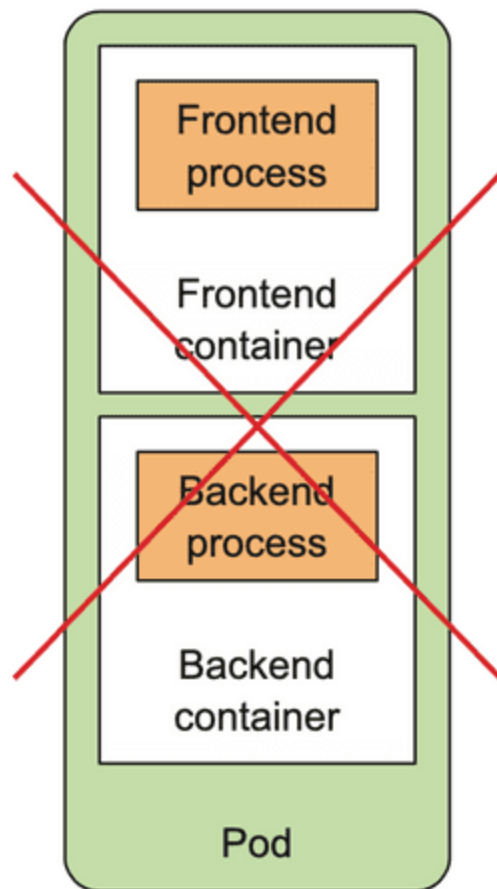
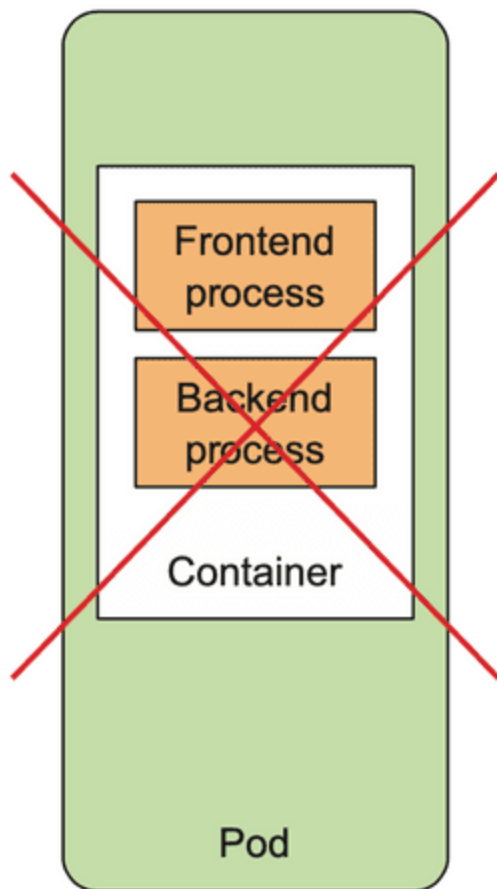
POD – много контейнеров

*Значит, это что-то типа
docker-compose?*

**Docker-compose –
эмуляция прод среды**

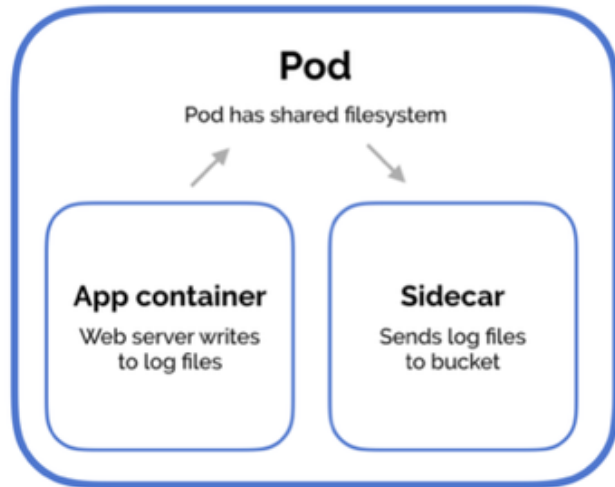
```
s3:
  image: minio/minio
  container_name: mlflow_s3
  ports:
    - 9000:9000
    - 9001:9001
  command: server /data --console-address ':9001'
  environment:
    - MINIO_ROOT_USER=mlflow
    - MINIO_ROOT_PASSWORD=password

mlflow:
  build:
    context: .
    dockerfile: Dockerfile
  ports:
    - 5050:5000
  environment:
    - MLFLOW_S3_ENDPOINT_URL=http://s3:9000
    - AWS_ACCESS_KEY_ID=mlflow
    - AWS_SECRET_ACCESS_KEY=password
```

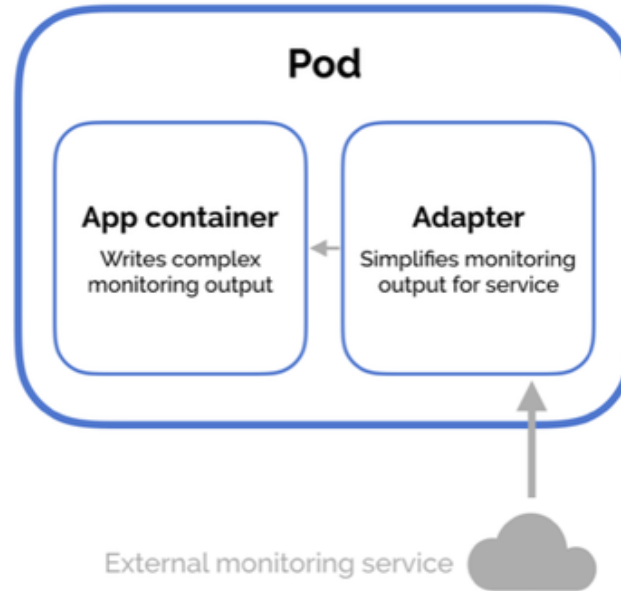


Паттерны использования POD

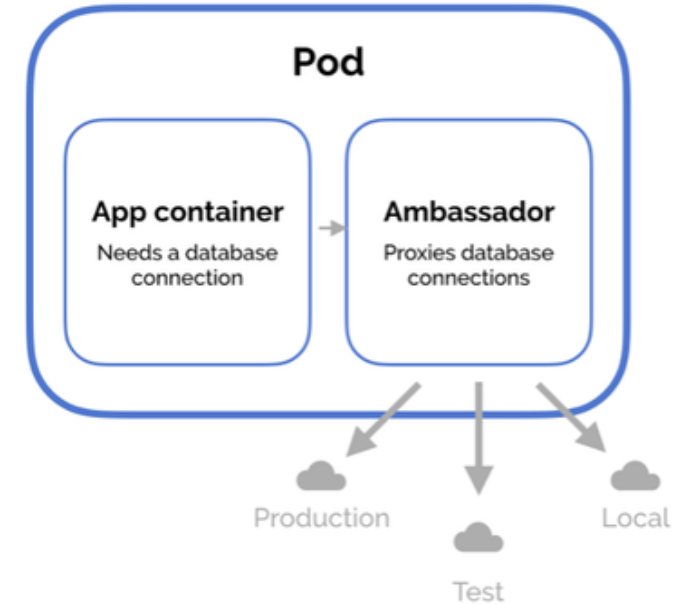
Sidecar



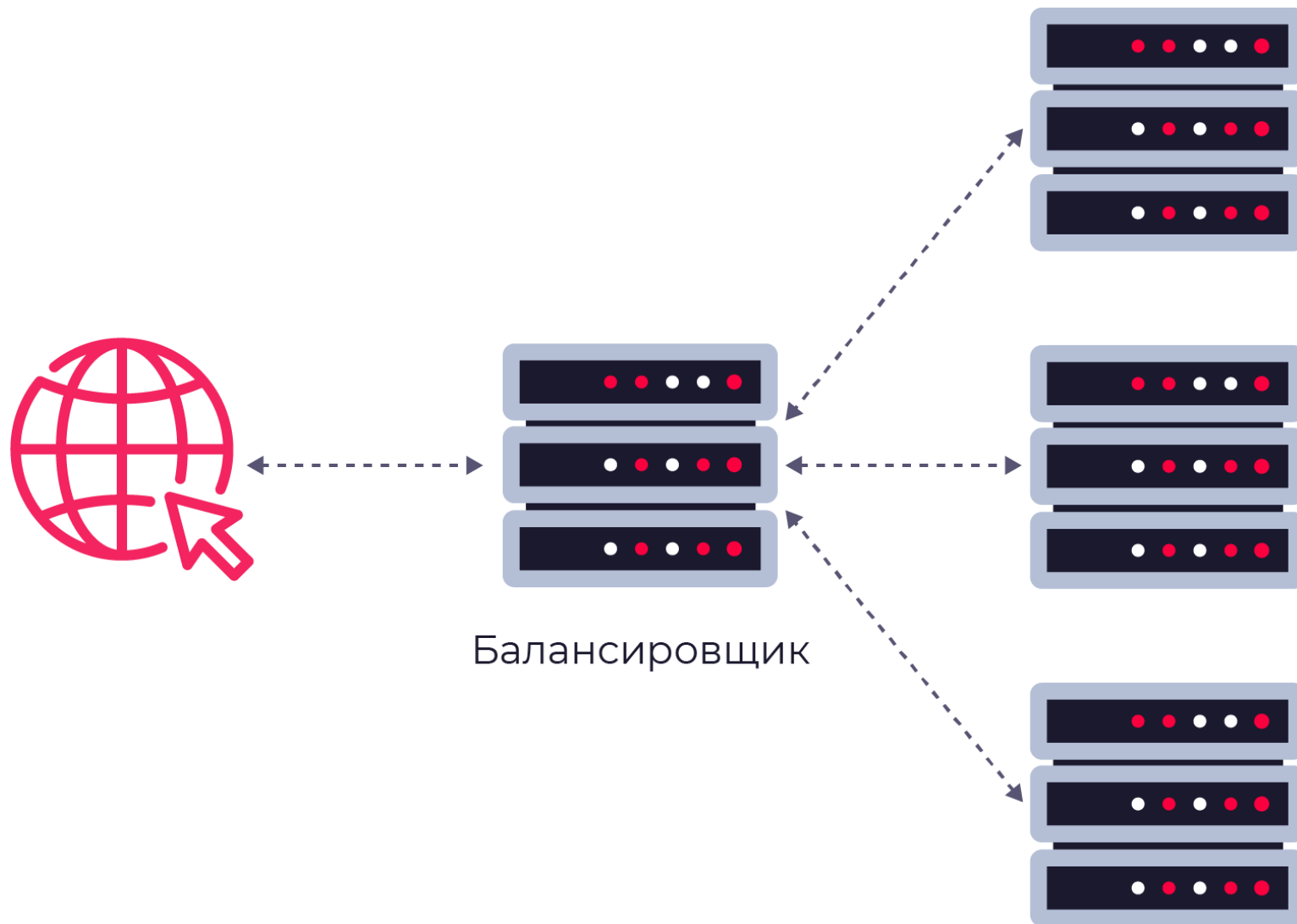
Adapter



Ambassador



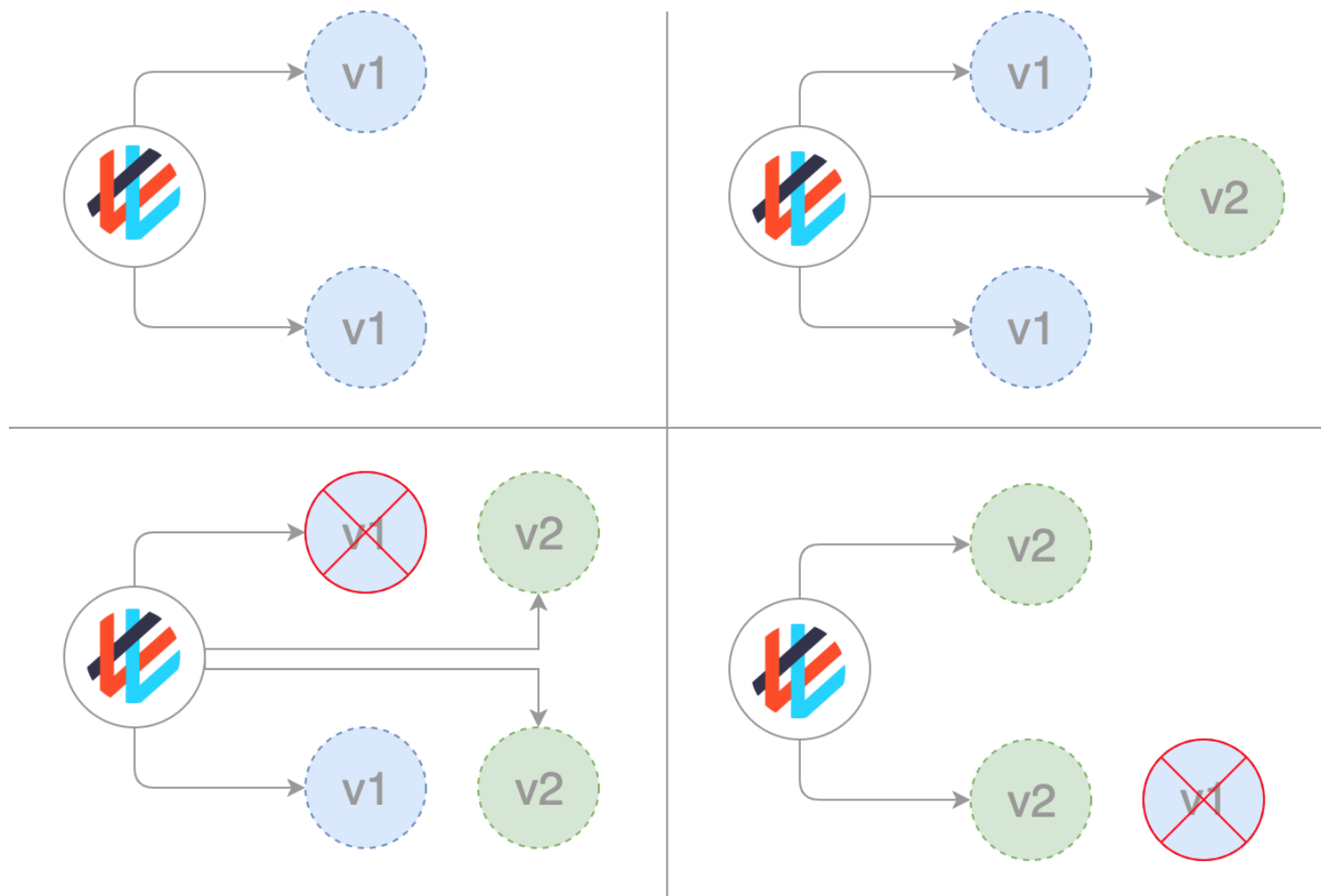
Масштабирование



Сервис

```
1  apiVersion: v1
2  kind: Service
3  metadata:
4    name: ml-service
5  spec:
6    selector:
7      app: fastapi-ml
8    type: LoadBalancer
9    ports:
10     - protocol: TCP
11       port: 80
12       targetPort: 80
```


Обновления



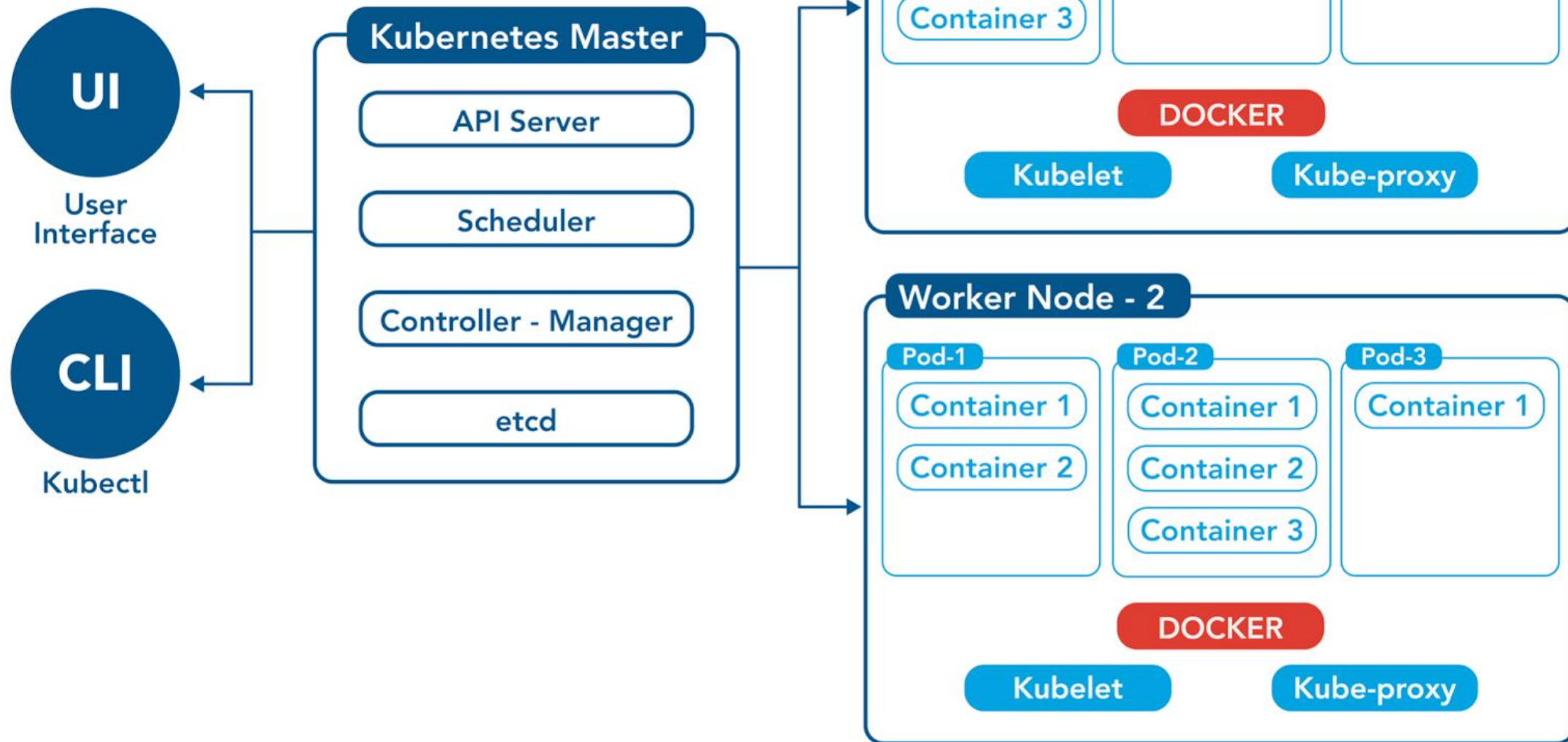
Job

```
1  apiVersion: batch/v1
2  kind: Job
3  metadata:
4    name: pi
5  spec:
6    template:
7      spec:
8        containers:
9          - name: pi
10           image: perl:5.34.0
11           command: ["perl", "-Mbignum=bpi", "-wle", "print bpi(2000)"]
12           restartPolicy: Never
13   backoffLimit: 4
```

CronJob

```
1  apiVersion: batch/v1
2  kind: CronJob
3  metadata:
4    name: hello
5  spec:
6    schedule: "* * * * *"
7    jobTemplate:
8      spec:
9        template:
10          spec:
11            containers:
12              - name: hello
13                image: busybox:1.28
14                imagePullPolicy: IfNotPresent
15                command:
16                  - /bin/sh
17                  - -c
18                  - date; echo Hello from the Kubernetes cluster
19            restartPolicy: OnFailure
```

Kubernetes Architecture



Мемы вместо выводов

<https://www.youtube.com/watch?v=LeVULLqWwcg>