MTC
WEB
SERVICES

РАСШИРЯЕМ ВОЗМОЖНОСТИ CLUSTERAPI

как написать свой infra провайдер и не сойти с ума

Иван Гулаков
DevOps Technical Lead MWS

Иван Гулаков

Техлид DevOps-команды

- Занимаюсь развитием и поддержкой инфраструктуры для облачных сервисов.
- Люблю контейнеры, качу кубы, пилю решения «из коробки».
- Есть 3 кота, которые тоже охотно используют коробочные решения.
- 7 лет в ИТ и ещё не помер.

Инфраструктура MWS

Сервисы облака

MWS

compute | storage | etc

network | paas

Инфраструктура облака

K8s cluster | K8s cluster | K8s cluster

Runtime

Управление инфраструктурой

Pinata platform

Pinata — это...

- Много ClusterAPI
- Много операторов
- Много helm + gitops
- ~~Багов и их героического превозмогания~~

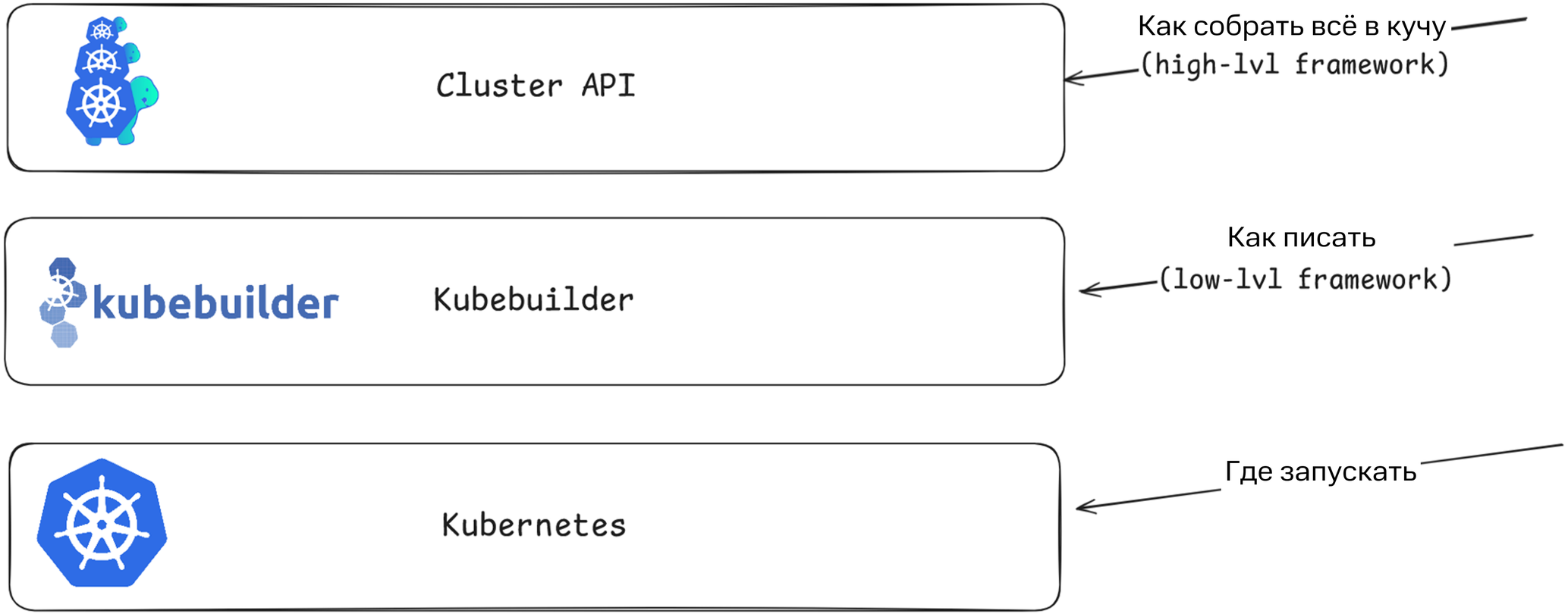
О чём поговорим?

01 ClusterAPI и infrastructure provider'ы

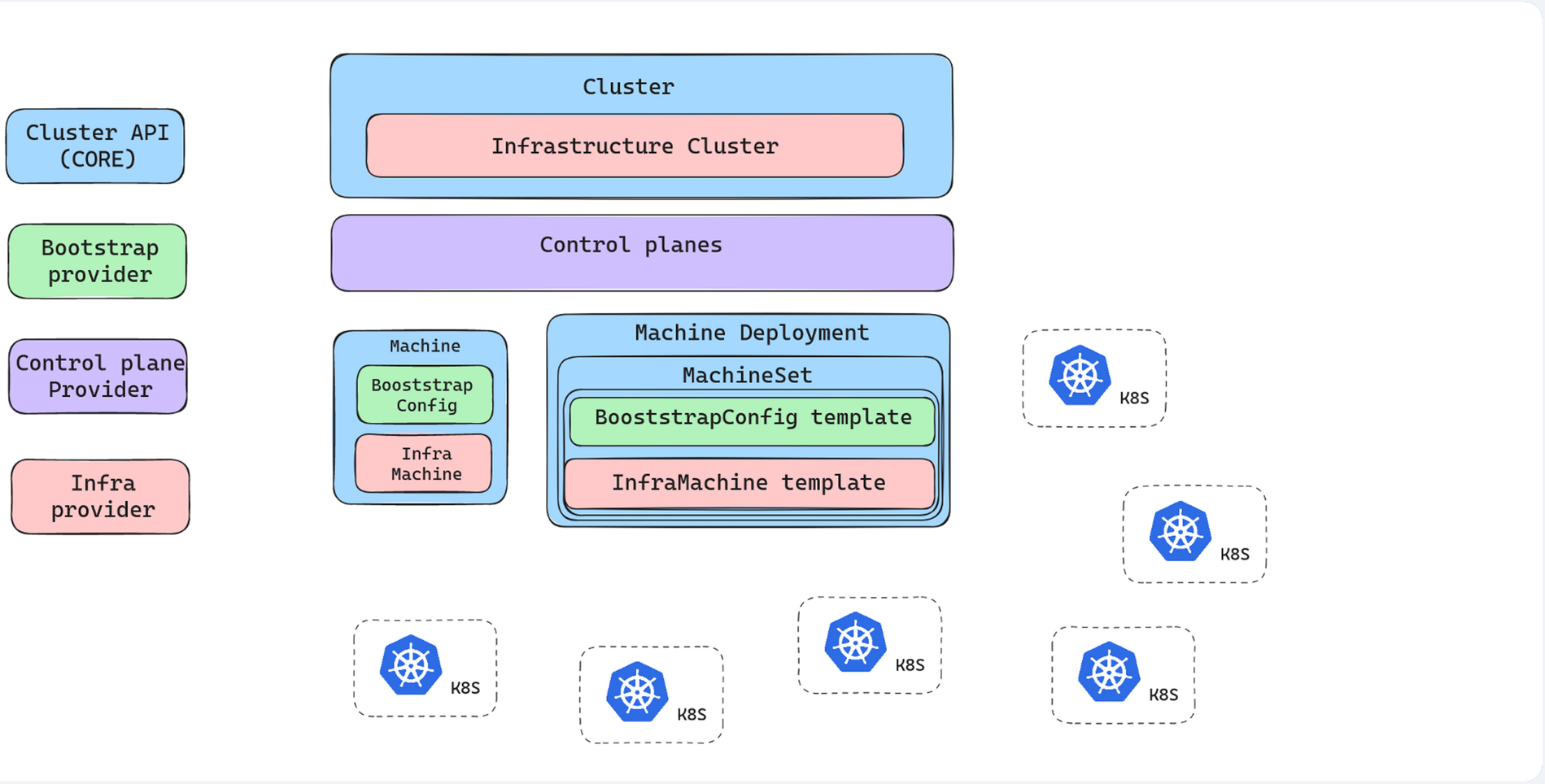
02 Зачем писать своё

03 И сколько боли под этим "писать своё" скрывается

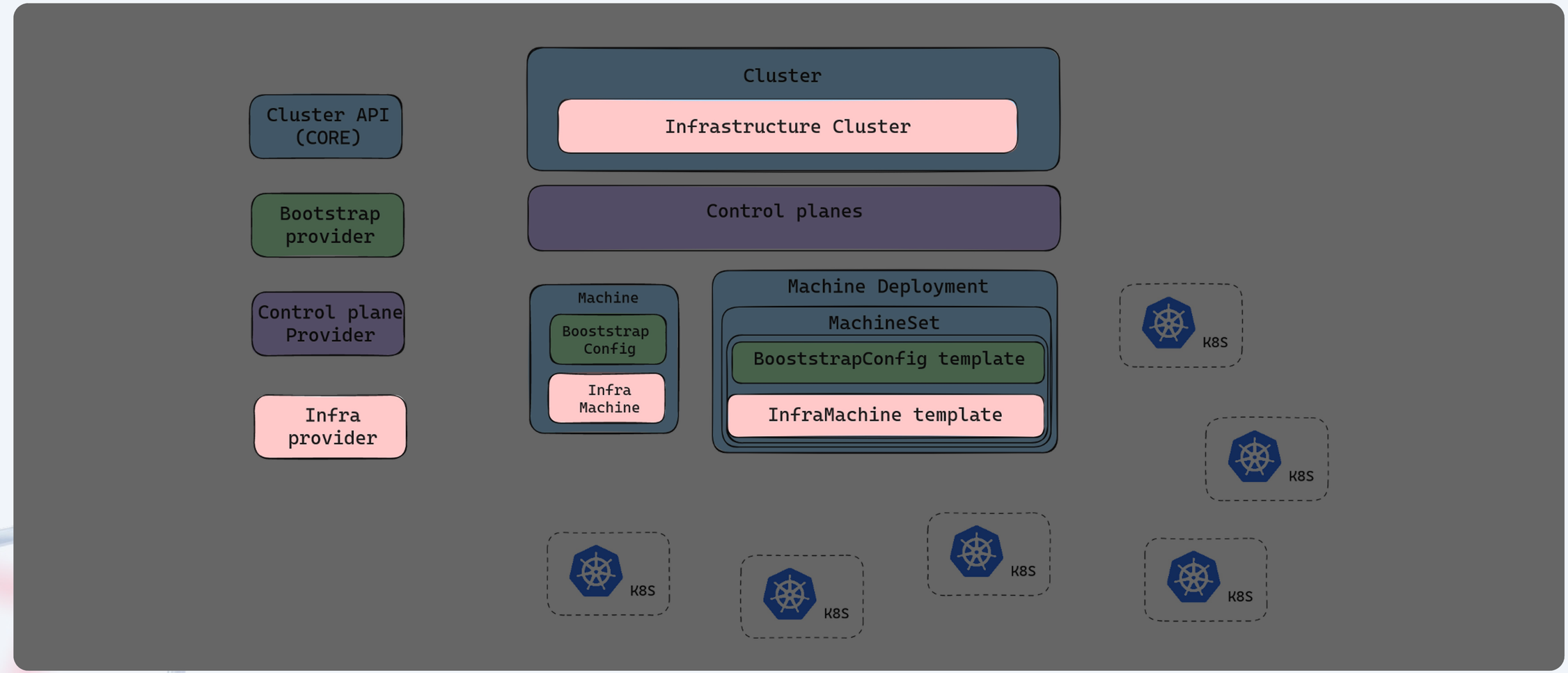
3 кита управления кластерами



ClusterAPI — оркестратор над оркестраторами



Infrastucture Provider — главный трудяга вашего CAPI



Infrastucture Provider

Infra cluster controller

Подготавливает
load balancer для kube api

Репортит статус
в bootstrap provider

https://cluster-api.sigs.k8s.io/developer/providers/cluster-infrastructure

Infrastucture Provider

Infra cluster controller

Подготавливает
load balancer для kube api

Репортит статус
в bootstrap provider

https://cluster-api.sigs.k8s.io/developer/providers/cluster-infrastructure

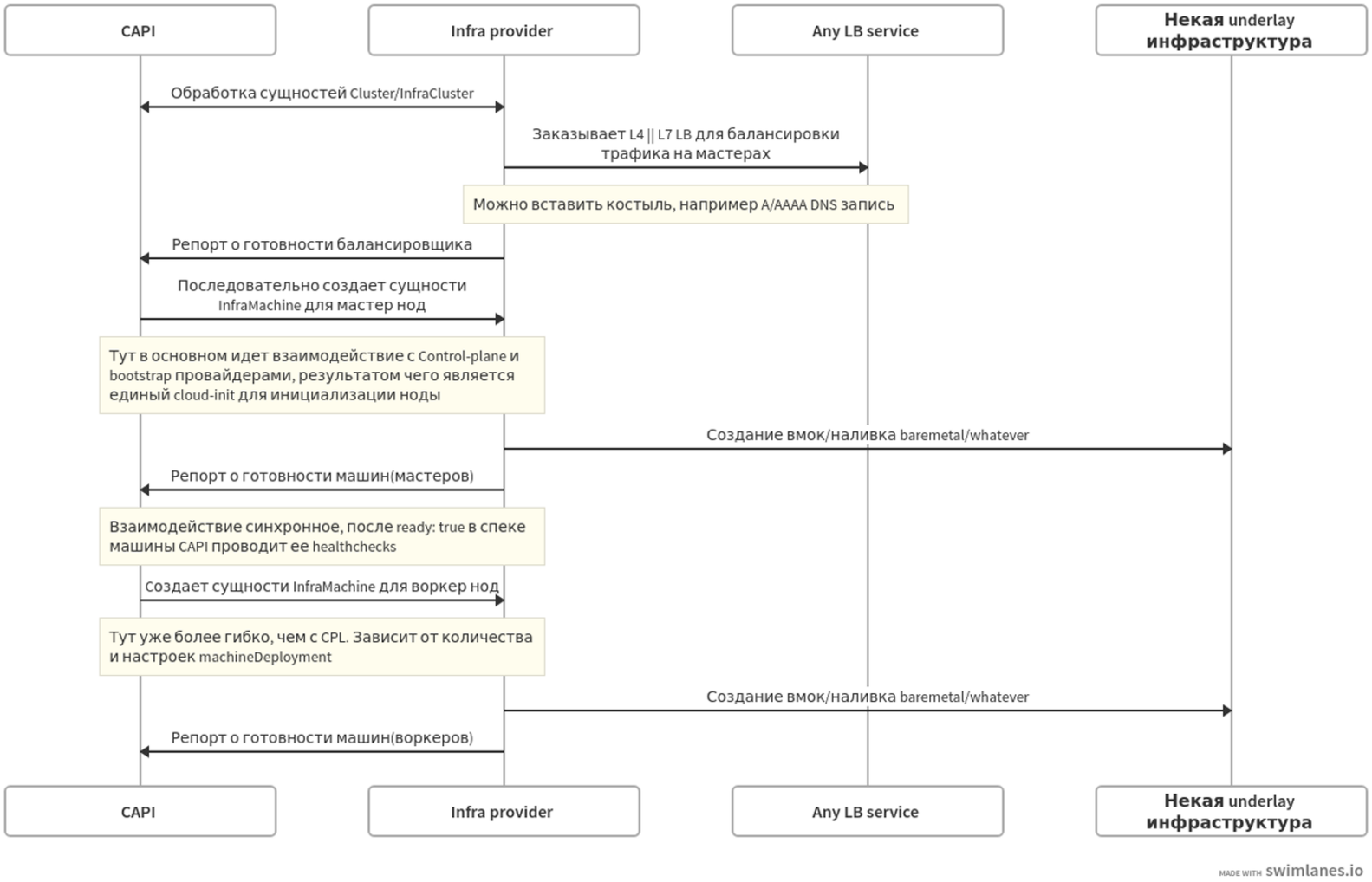
Infra machine controller

Подготавливает
инфраструктуру вплоть
до уровня ОС для запуска k8s

Репортит статус
в bootstrap provider

https://cluster-api.sigs.k8s.io/developer/providers/machine-infrastructure

Infra provider flow



Почему решили писать свой провайдер

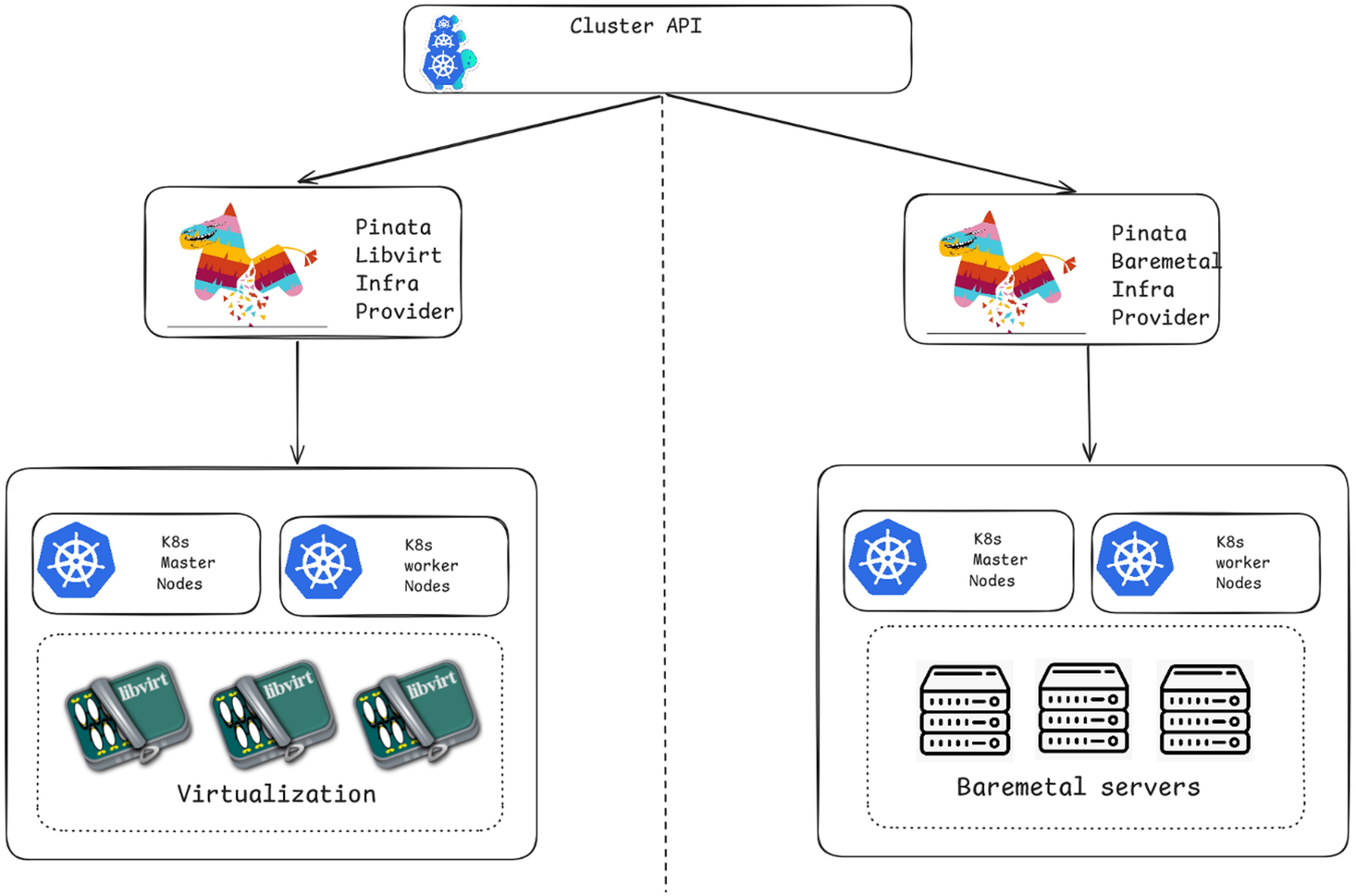
Почему решили писать свой провайдер

01 Наличие собственной системы наливки baremetal серверов VOLK

02 Необходимость интеграции с системой инвентаризации хостов

03 Дополнительные требования по управлению ЖЦ нод кластера от сервисов облака

vO: 2 независимых infra провайдера



v0: 2 независимых infra провайдера

Pinata baremetal infra provider

Работает только с baremetal

Основная разработка, ориентирован на большие prod-like кластера

Pinata libvirt infra provider

Работает только с ВМ на libvirt

Побочная разработка, ориентирован на маленькие лабные стенды в dev контуре

Как сделать мастера дешевле

01 Заселять полезную нагрузку на мастера

02 Мастера на ВМ

03 Мастера в кубе

Schedulable мастера

- Высокая вероятность отказа ноды из-за пользовательской нагрузки

- Security — без комментариев

Мастера в кубе

- Не хочется поддерживать свой SDS (CSI)
- Сложная сетевая топология кластера с мастерами
- Комплексная сложность всего решения

Мастера на BM

- Максимально упрощенный
  по конфигурации standalone libvirt
- Простая сетевая схема
- Легко дебажить

Cluster && InfraCluster

```yaml
apiVersion: cluster.x-k8s.io/v1beta1
kind: Cluster
metadata:
  labels:
    cluster.x-k8s.io/cluster-name: devoops
  name: devoops
  namespace: devoops
spec:
  clusterNetwork:
    pods:
      cidrBlocks:
        - ...
    services:
      cidrBlocks:
        - ...
  controlPlaneRef:
    apiVersion: controlplane.cluster.x-k8s.io/v1beta1
    kind: KubeadmControlPlane
    name: devoops
  infrastructureRef:
    apiVersion: infrastructure.cluster.x-k8s.io/v1beta1
    kind: PinataMetalCluster
    name: devoops
```

```yaml
apiVersion: infrastructure.cluster.x-k8s.io/v1beta1
kind: PinataMetalCluster
metadata:
  labels:
    cluster.x-k8s.io/cluster-name: devoops
  name: devoops
  namespace: devoops
spec:
  domainPrefix: dev.cloud.mts.ru
  matchbox:
    namespace: matchbox
  controlPlanePort: 6443
  ipam:
    ipRangeCidr: ...
    netbox:
      domain: ...
      secretKeyRef:
        name: netbox
        key: token
```

Control plane && Machine Deployment

```yaml
apiVersion: controlplane.cluster.x-k8s.io/v1beta1
kind: KubeadmControlPlane
metadata:
  labels:
    cluster.x-k8s.io/cluster-name: devoops
  name: devoops
  namespace: devoops
spec:
  kubeadmConfigSpec:
    initConfiguration:
      nodeRegistration:
        kubeletExtraArgs:
        name: "{{ .pinatametal.name }}"
    joinConfiguration:
      controlPlane: {}
      nodeRegistration:
        kubeletExtraArgs:
        name: "{{ .pinatametal.name }}"
  machineTemplate:
    infrastructureRef:
      apiVersion: infrastructure.cluster.x-k8s.io/v1beta1
      kind: PinataMetalMachineTemplate
      name: devoops-cpl
  replicas: 1
  rolloutStrategy:
    rollingUpdate:
      maxSurge: 1
    type: RollingUpdate
  version: v1.29.2
```

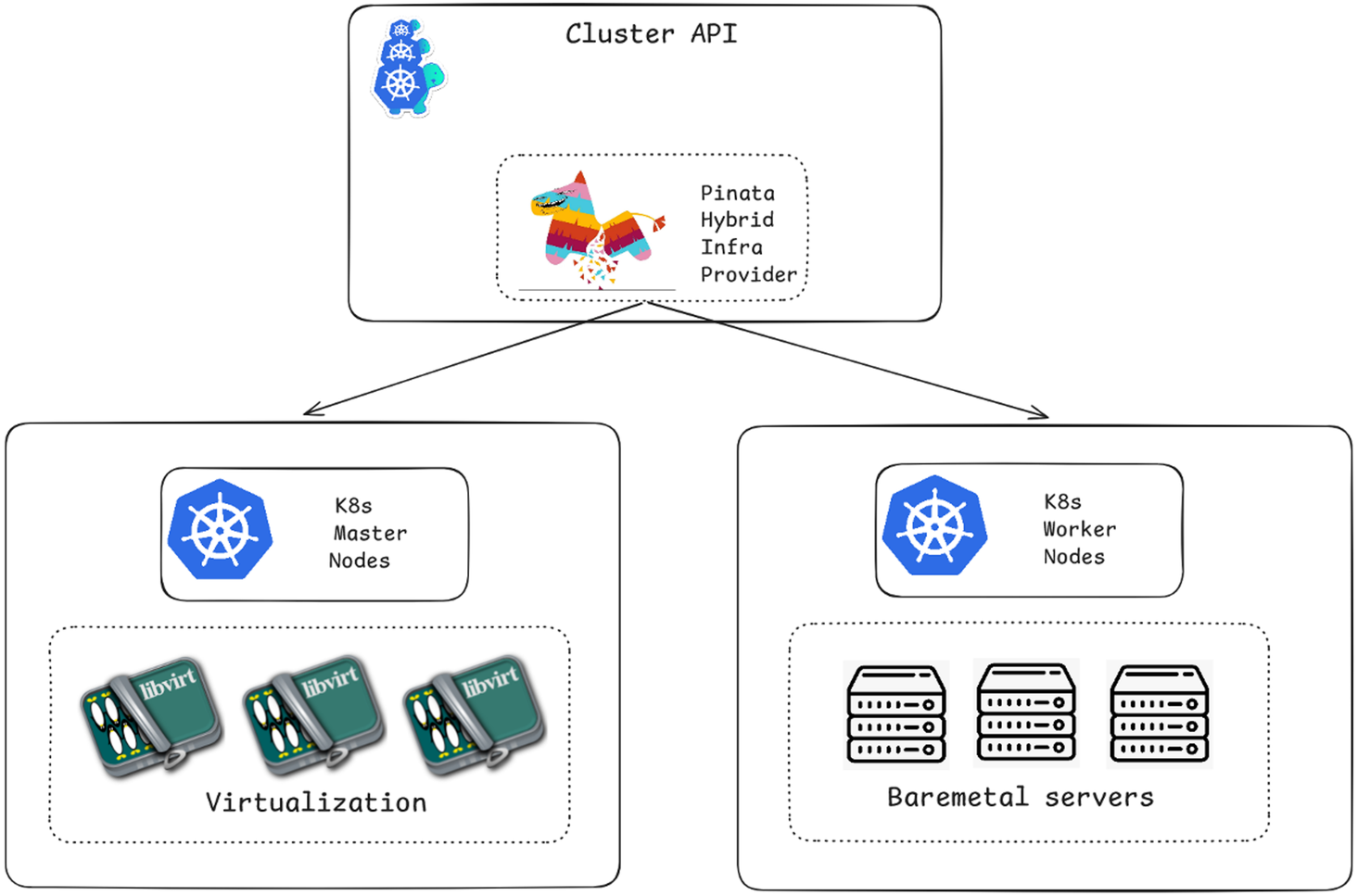
```yaml
apiVersion: cluster.x-k8s.io/v1beta1
kind: MachineDeployment
metadata:
  labels:
    cluster.x-k8s.io/cluster-name: devoops
  name: devoops-md-0
  namespace: devoops
spec:
  clusterName: devoops
  replicas: 1
  selector:
    matchLabels: {}
  template:
    spec:
      bootstrap:
        configRef:
          apiVersion: bootstrap.cluster.x-k8s.io/v1beta1
          kind: KubeadmConfigTemplate
          name: devoops-md-0
      clusterName: devoops
      infrastructureRef:
        apiVersion: infrastructure.cluster.x-k8s.io/v1beta1
        kind: PinataMetalMachineTemplate
        name: devoops-machine
      version: v1.29.2
```

PinataMetalCluster vs PinataLibvirtCluster

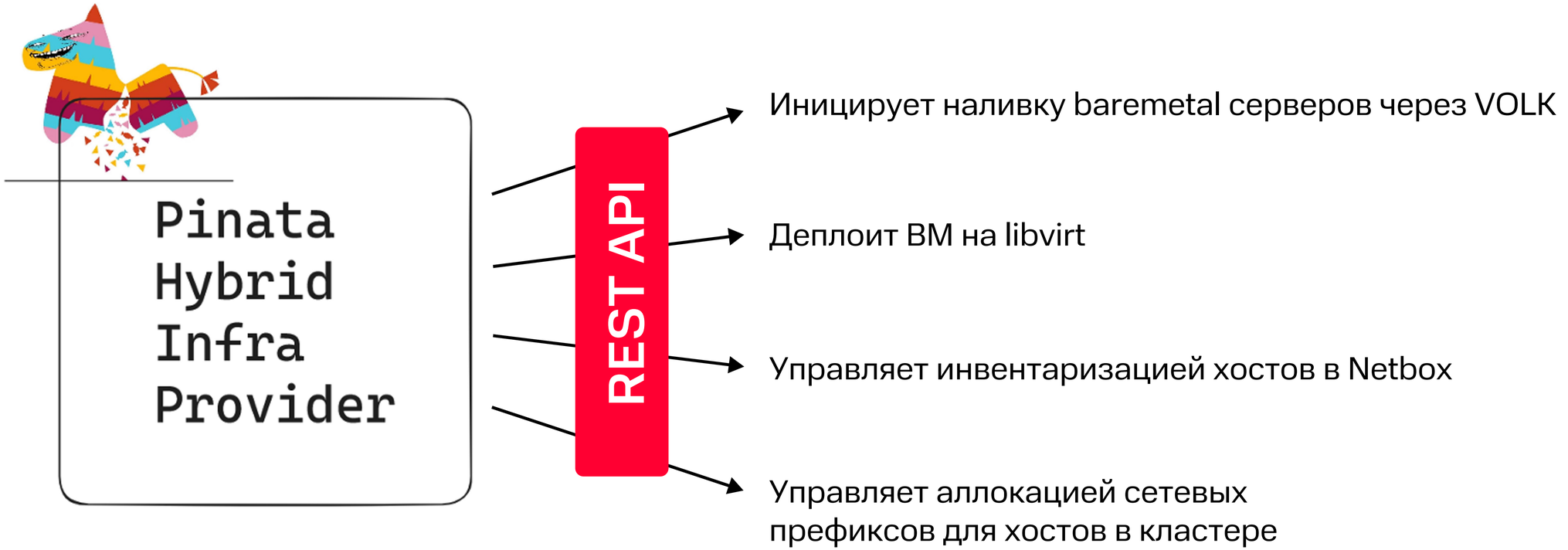
```yaml
apiVersion: infrastructure.cluster.x-k8s.io/v1beta1
kind: PinataMetalCluster
metadata:
  labels:
    cluster.x-k8s.io/cluster-name: devoops
  name: devoops
  namespace: devoops
spec:
  domainPrefix: dev.cloud.mts.ru
  matchbox:
    namespace: matchbox
  controlPlanePort: 6443
  ipam:
    ipRangeCidr: ...
    netbox:
      domain: ...
      secretKeyRef:
        name: netbox
        key: token
```

```yaml
apiVersion: infrastructure.cluster.x-k8s.io/v1alpha1
kind: PinataLibvirtCluster
metadata:
  labels:
    cluster.x-k8s.io/cluster-name: devoops
  name: devoops
  namespace: devoops
spec:
  hypervisorNodes:
    - name: libvirt1
      conf:
        host: ...
        user: ...
        keyfileSecretRef:
          name: libvirt-ssh-key
          key: id_ed25519_libvirt
        remoteSocket: "/var/run/libvirt/libvirt-sock"
        port: ...
  ipamIPRange: ...
  ipamCluster: local
  ipamDomain: ...
  dnsNameservers:
    - ...
```

v1: Гибридные кластера



v1: Гибридные кластера



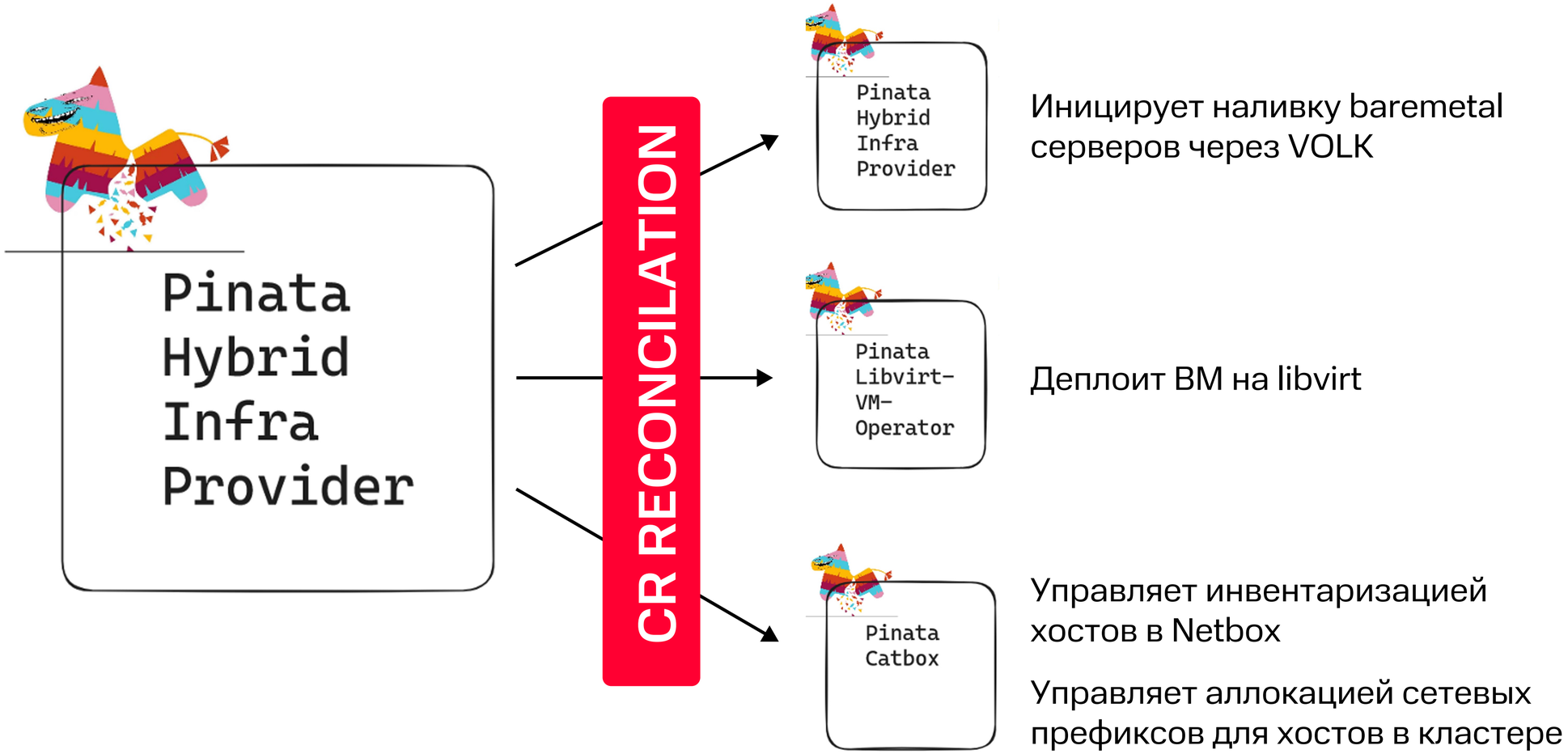
Бизнес доволен

- Экономия в десятки серверов на мастерах
- Развертывание кластеров значительно ускорилось
- Сроки не продолбаны

А мы — не очень...

- Сложно понять, где застрял деплой
- Сложно тестировать
- Высокий порог входа в кодовую базу провайдера
- Огромные “портянки” информации в статусах → сложный FSM

v2: Флот операторов



Инициирует наливку baremetal серверов через VOLK

Деплоит ВМ на libvirt

Управляет инвентаризацией хостов в Netbox

Управляет аллокацией сетевых префиксов для хостов в кластере

Взаимодействие с Netbox через клиент

internal/service/interface.go +5 −10 View file @b7da7516

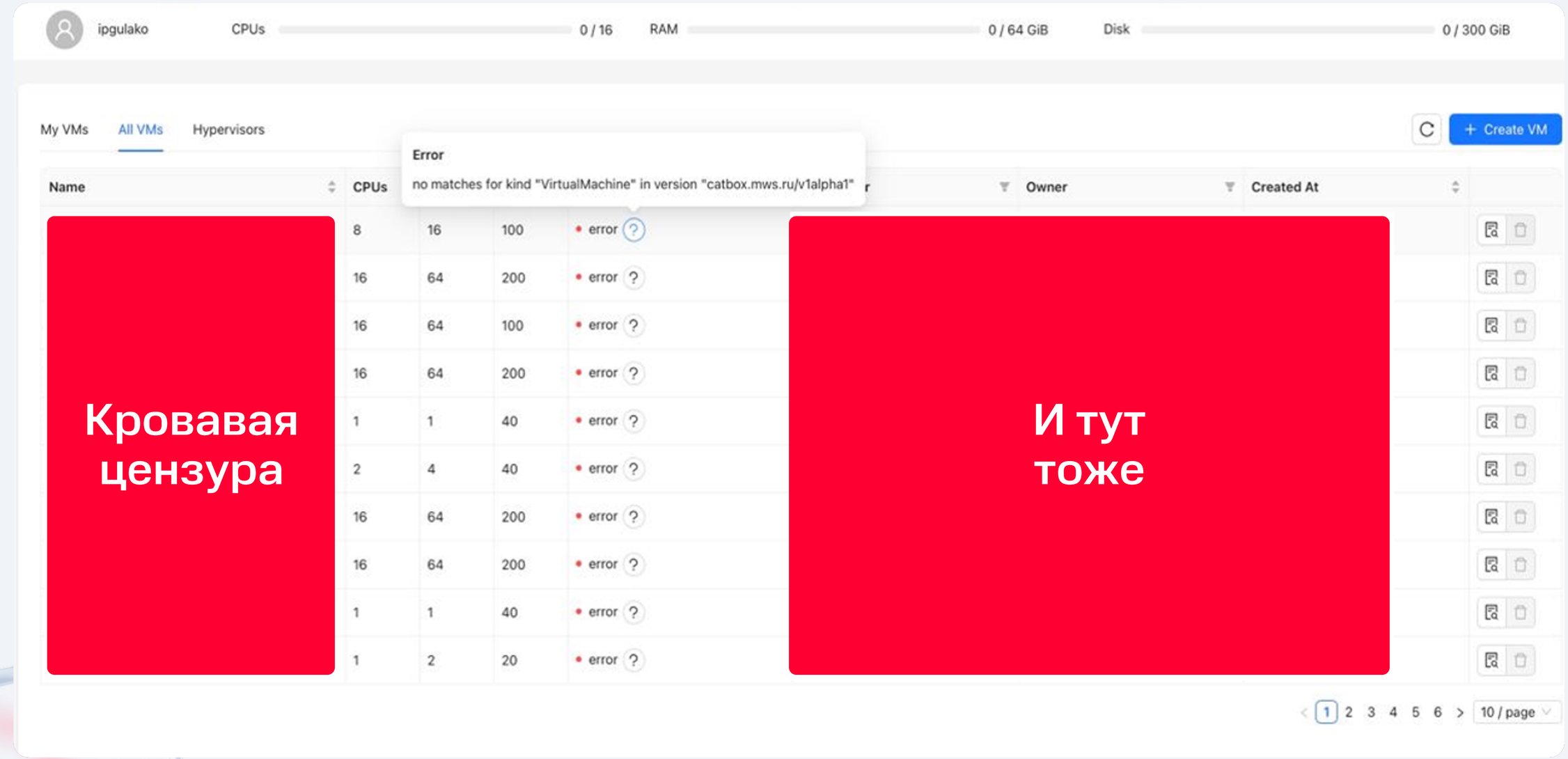
```
@@ -36,7 +36,9 @@ type PinataMetal interface {
 }

 type Cluster interface {
+        MachineWithRef
         GetName() string
+        GetNamespace() string
         GetHostname() string
         GetDomain() string
         SetControlPlaneEndpoint(hostname string, port int)
@@ -46,12 +48,6 @@ type Cluster interface {
         GetServiceCIDRBlocks() []string
         GetPodCIDRBlocks() []string

-        AddPodCidrID(id int32, cidr string)
-        GetPodCidrIDs() []int32
-        GetPodCidrMap() map[string]int32
-        SetClusterID(id int32)
-        GetClusterID() int32
-
         GetTenant() string
         GetSite() string
         GetGroup() string
@@ -69,7 +65,6 @@ type IPA interface {

 type BaremetalMachine interface {
         BaseMachine
-        MachineWithRef
         ListAvailablePinataMetals() ([]PinataMetal, error)
         GetOwnedPinataMetal() (PinataMetal, error)
         GetIpxeID() string
@@ -88,9 +83,6 @@ type LibvirtMachine interface {
         GetVcpus() uint
         GetRam() uint
         GetDisks() []infrav1beta1.Disk
-        GetCluster() string
-        SetVMId(int)
-        GetVMId() int

         GetHypervisorHost() string
         GetHypervisorName() string
@@ -114,6 +106,7 @@ type VolkMachine interface {
 }
```

Взаимодействие с Netbox через доп. оператор

```
func (m *MachinePinataVolkService) getCatboxDeviceStatus(ctx context.Context) (*catboxv1alpha1.DeviceStatus, error)
{
    device, err := m.getOrCreateCatboxDevice(ctx)
    if err != nil {
        return nil, err
    }

    if !device.Status.Ready {
        return nil, fmt.Errorf("%w: catbox device is not ready", ErrTemporary)
    }

    return &device.Status, nil
}

func (m *MachinePinataVolkService) getOrCreateCatboxDevice(ctx context.Context) (*catboxv1alpha1.Device, error) {
    device := &catboxv1alpha1.Device{}
    ns := types.NamespacedName{
        Name:      m.machine.GetName(),
        Namespace: m.machine.GetNamespace(),
    }
    if err := m.client.Get(ctx, ns, device); err != nil {
        if k8sErrors.IsNotFound(err) {
            return m.createCatboxDevice(ctx)
        }
        return nil, fmt.Errorf("cannot get catbox device %s: %w", m.machine.GetName(), err)
    }

    return device, nil
}

func (m *MachinePinataVolkService) createCatboxDevice(ctx context.Context) (*catboxv1alpha1.Device, error) {
    host, err := m.getOrReserveVolkHost()
    if err != nil {
        return nil, fmt.Errorf("cannot get volk host: %w", err)
    }

    device := &catboxv1alpha1.Device{
        ObjectMeta: metav1.ObjectMeta{
            Name:      m.machine.GetName(),
            Namespace: m.machine.GetNamespace(),
            OwnerReferences: []metav1.OwnerReference{
                m.machine.GetOwnerRef(),
            },
        },
        Spec: catboxv1alpha1.DeviceSpec{
            Selector: catboxv1alpha1.DeviceSelector{
                OutOfBandIP: host.IpmiIP,
            },
            ClusterName: m.cluster.GetName(),
            Tenant:      m.cluster.GetTenant(),
        },
    }

    if err := m.client.Create(ctx, device); err != nil {
        return nil, fmt.Errorf("cannot create catbox device %s: %w", m.cluster.GetName(), err)
    }

    return device, nil
}
```

Победа?...

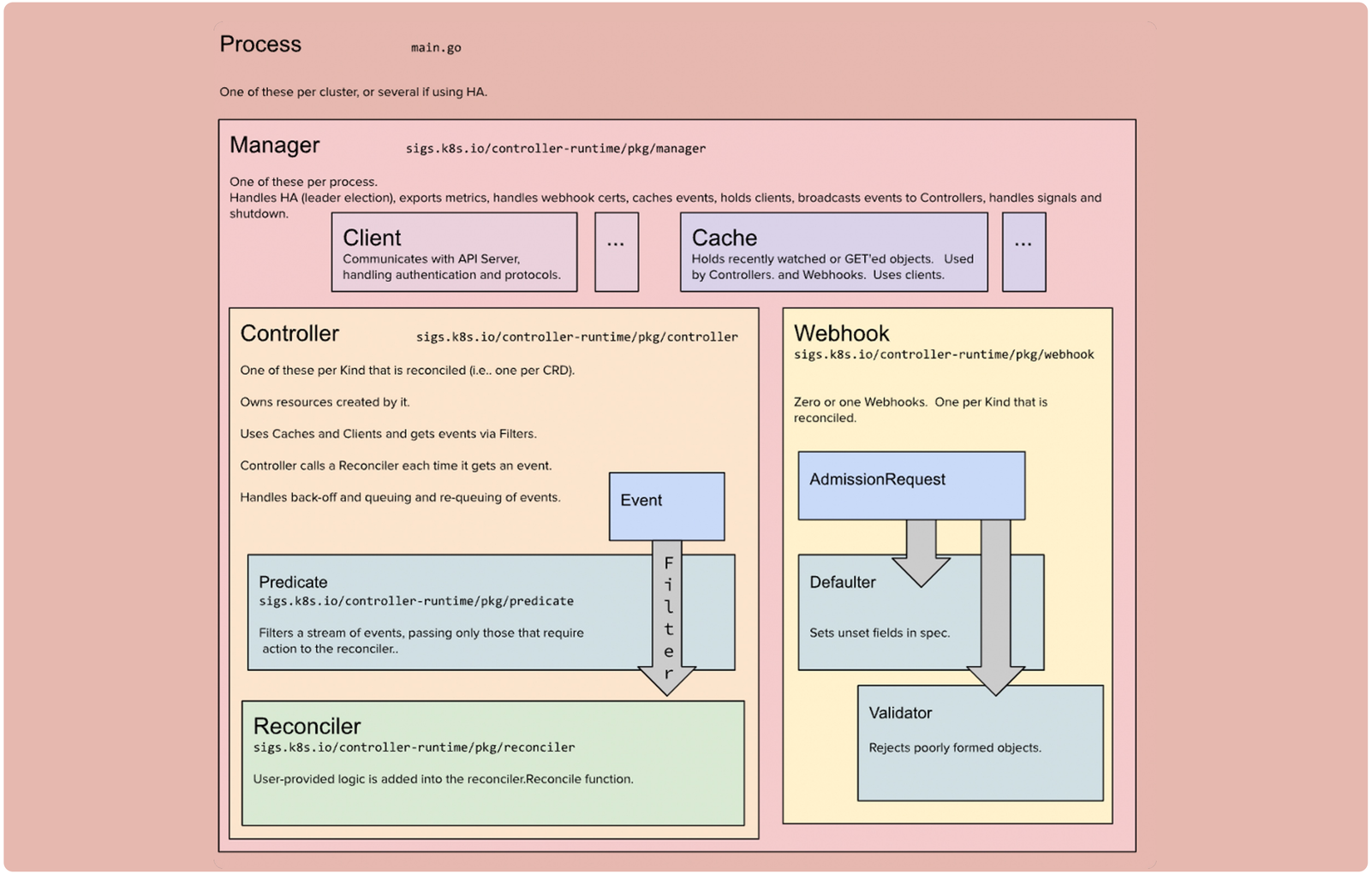
- Ретраи и обработка ошибок теперь на стороне отдельных операторов
- Гораздо проще тестировать кусочки бизнес логики а не весь ее пайплайн
- Отдельные операторы проще дорабатывать и поддерживать
- Статусы CR самого провайдера заметно "похудели"


ipgulako
CPUs 0 / 16
RAM 0 / 64 GiB
Disk 0 / 300 GiB
My VMs
All VMs
Hypervisors
Create VM
Error
no matches for kind "VirtualMachine" in version "catbox.mws.ru/v1alpha1"
Name
CPUs
Owner
Created At
Кровавая цензура
И тут тоже
error
10 / page

v3: Наводим красоту

- Перерабатываем FSM
- Чистим мусор за собой при удалении
- Фильтруем ненужные события

Как устроен оператор



Из чего состоит CR

```
type PinataHybridMachine struct {
    metav1.TypeMeta   `json:",inline"`
    metav1.ObjectMeta `json:"metadata,omitempty"`

    Spec   PinataHybridMachineSpec   `json:"spec,omitempty"`
    Status PinataHybridMachineStatus `json:"status,omitempty"`
}
```

Метадата объекта:

- тип/версия
- лейблы/аннотации
- история обновлений
- ...

Тело объекта:

Содержит основную информацию
для обработки контроллером

Статус объекта:

Вспомогательная сущность
для обработки промежуточных состояний

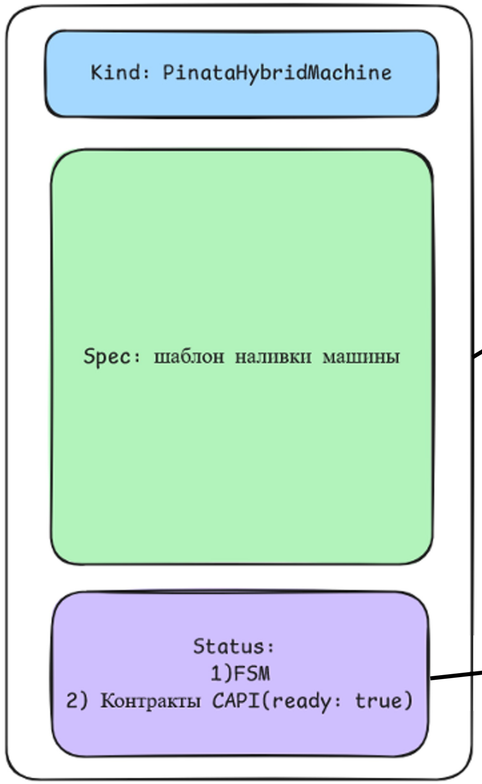
Ограничения clusterctl move

> **Warning**
>
> Before running `clusterctl move`, the user should take care of preparing the target management cluster, including also installing all the required provider using `clusterctl init`.
>
> The version of the providers installed in the target management cluster should be at least the same version of the corresponding provider in the source cluster.

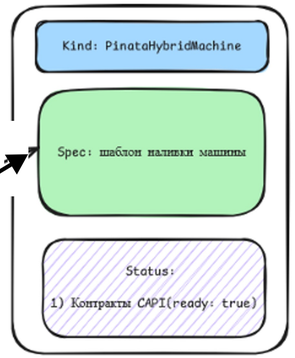
> **Warning: Status subresource is never restored**
>
> Every object's `Status` subresource, including every nested field (e.g. `Status.Conditions`), is never restored during a `move` operation. A `Status` subresource should never contain fields that cannot be recreated or derived from information in spec, metadata, or external systems. Provider implementers should not store non-ephemeral data in the `Status`. `Status` should be able to be fully rebuilt by controllers by observing the current state of resources.

Разделяем стейты

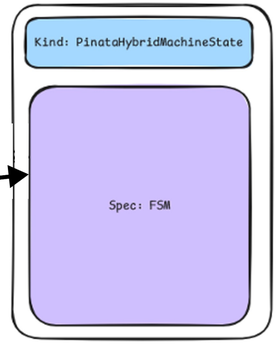
Полное управление
ЖЦ ноды



Только информация о наливке (шаблон)



Только информация о состоянии (FSM)



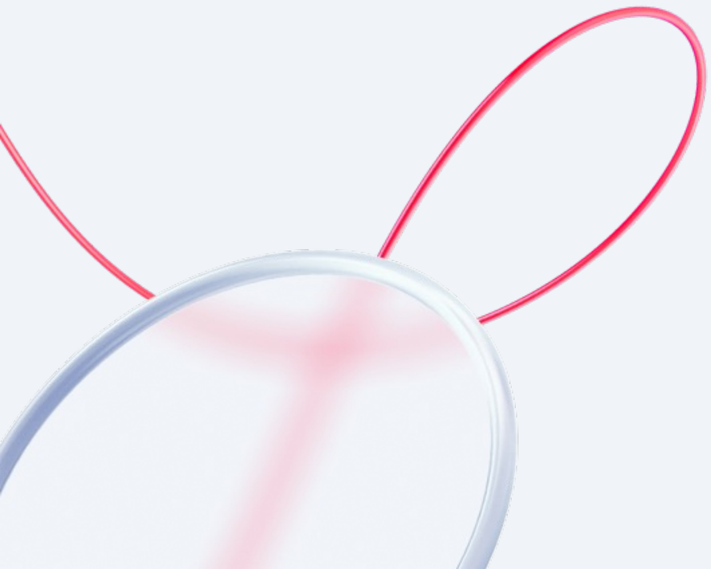
Разделяем стейты

```yaml
apiVersion: infrastructure.cluster.x-k8s.io/v1beta1
kind: PinataHybridMachine
metadata:
  annotations:
    cluster.x-k8s.io/cloned-from-groupkind: PinataHybridMachineTemplate.infrastructure.cluster.x-k8s.io
    cluster.x-k8s.io/cloned-from-name: msk-avg-stg-iaas-capi-mesh-demo-1-controlplane
  creationTimestamp: '2024-07-18T17:15:34Z'
  finalizers:
    - pinata-hybrid.machine.x-k8s.io/finalizer
  generation: 2
  labels:
    cluster.x-k8s.io/cluster-name: msk-avg-stg-iaas-capi-mesh-demo-1
    cluster.x-k8s.io/control-plane: ''
    cluster.x-k8s.io/control-plane-name: msk-avg-stg-iaas-capi-mesh-demo-1
  name: msk-avg-stg-iaas-capi-mesh-demo-1-7pxb4
  namespace: msk-avg-stg-iaas-capi-mesh-demo-1
status:
  ready: true
  state: ''
spec:
  ...
```

```yaml
apiVersion: infrastructure.cluster.x-k8s.io/v1beta1
kind: PinataHybridMachineState
metadata:
  creationTimestamp: '2024-07-18T17:15:35Z'
  generation: 5
  name: msk-avg-stg-iaas-capi-mesh-demo-1-7pxb4
  namespace: msk-avg-stg-iaas-capi-mesh-demo-1
  ownerReferences:
    - apiVersion: infrastructure.cluster.x-k8s.io/v1beta
      blockOwnerDeletion: true
      controller: true
      kind: PinataHybridMachine
      name: msk-avg-stg-iaas-capi-mesh-demo-1-7pxb4
      uid: 3e8bbb21-1336-4215-8c96-2b667b8e7d8d
spec:
  hypervisorName: libvirt1
  ipamStatus:
    addresses:
      - cidr: ...
        id: ...
        mac: ...
        name: ens3
    vmid: 976
  state: Running
```

Чистим за собой CR

```go
ownerRef := metav1.NewControllerRef(s.vm, v1alpha1.GroupVersion.WithKind(s.vm.Kind))
catvm := catboxv1alpha1.VirtualMachine{
    ObjectMeta: v1.ObjectMeta{
    Name:            s.GetName(),
    Namespace:       s.GetNamespace(),
    OwnerReferences: []metav1.OwnerReference{*ownerRef},
},
}
```



```yaml
apiVersion: catbox.mws.ru/v1alpha1
kind: VirtualMachine
metadata:
    ownerReferences:
    - apiVersion: pinata.cloud.mts.ru/v1alpha1
      blockOwnerDeletion: true
      controller: true
      kind: LibvirtVM
      name: ipgulako-test
      uid: 2d28d089-2ee5-4020-bece-cc8e7bda5659
```

Custom predicates

- **Predicate** — механизм фильтрации ивентов с CR для оператора

- Позволяет контроллеру реагировать только на нужные ивенты

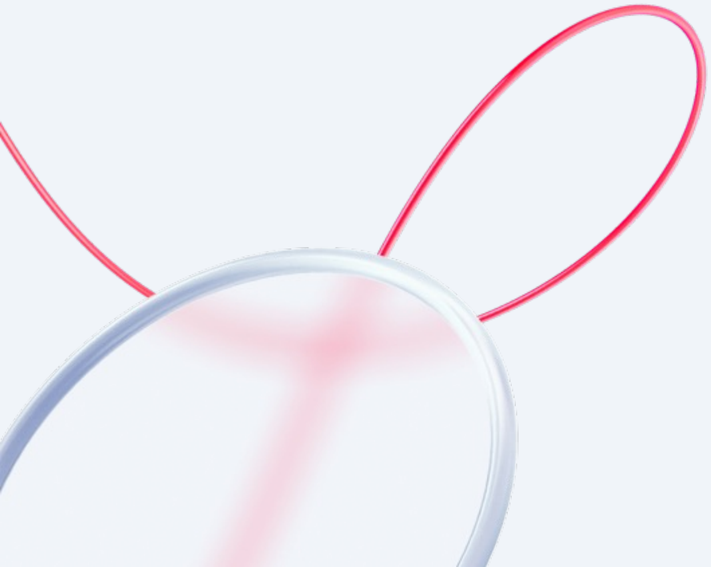
Custom predicates

- **Create** — первичное создание CR (kubectl create),
либо перезапуск оператора
- **Update** — любые действия по изменению
- **Delete** — удаление CR
- **Generic** — взаимодействие с CR другими контроллерами*

Custom predicates

```
// SetupWithManager sets up the controller with the Manager.
func (r *NodeConfigReconciler) SetupWithManager(mgr ctrl.Manager) error {
    customPredicates := predicate.Funcs{
        CreateFunc: func(e event.CreateEvent) bool {
            labels := e.Object.GetLabels()
            _, ok := labels[nodeLabelKey]
            if ok {
                return labels[nodeLabelKey] == r.NodeLabelValue
            }
            return false
        },
        UpdateFunc: func(e event.UpdateEvent) bool {
            labels := e.ObjectNew.GetLabels()
            _, ok := labels[nodeLabelKey]
            if ok {
                if len(e.ObjectOld.GetFinalizers()) != len(e.ObjectNew.GetFinalizers()) {
                    return true
                }
                // Дополнительно фильтруем ивенты на апдейт статуса, чтобы не было двойных применений и не ломался
дифф на старые значения
                if e.ObjectNew.GetGeneration() == e.ObjectOld.GetGeneration() {
                    return false
                }
                return labels[nodeLabelKey] == r.NodeLabelValue
            }
            return false
        },
        DeleteFunc: func(e event.DeleteEvent) bool {
            return true
        },
        // GenericFunc: func(e event.GenericEvent) bool {
        //  return false
        // },
    }

    return ctrl.NewControllerManagedBy(mgr).
        For(&sysctlv1alpha1.NodeConfig{}).
        WithEventFilter(customPredicates).
        Complete(r)
}
```

Заглушка ordering/immutability



Выводы:

- Много маленьких операторов лучше, чем один сложный
- Не перегружайте Status в ваших CRD кучей логики
- Не бойтесь писать свои инструменты

Спасибо за внимание

Есть вопросы?
Буду рад пообщаться с вами

Иван Гулаков

DevOps
Technical Lead MWS

https://t.me/kepiukik

Полезные ссылки

Kubebuilder

CAPI