Openshift

Maciej Żarczyński

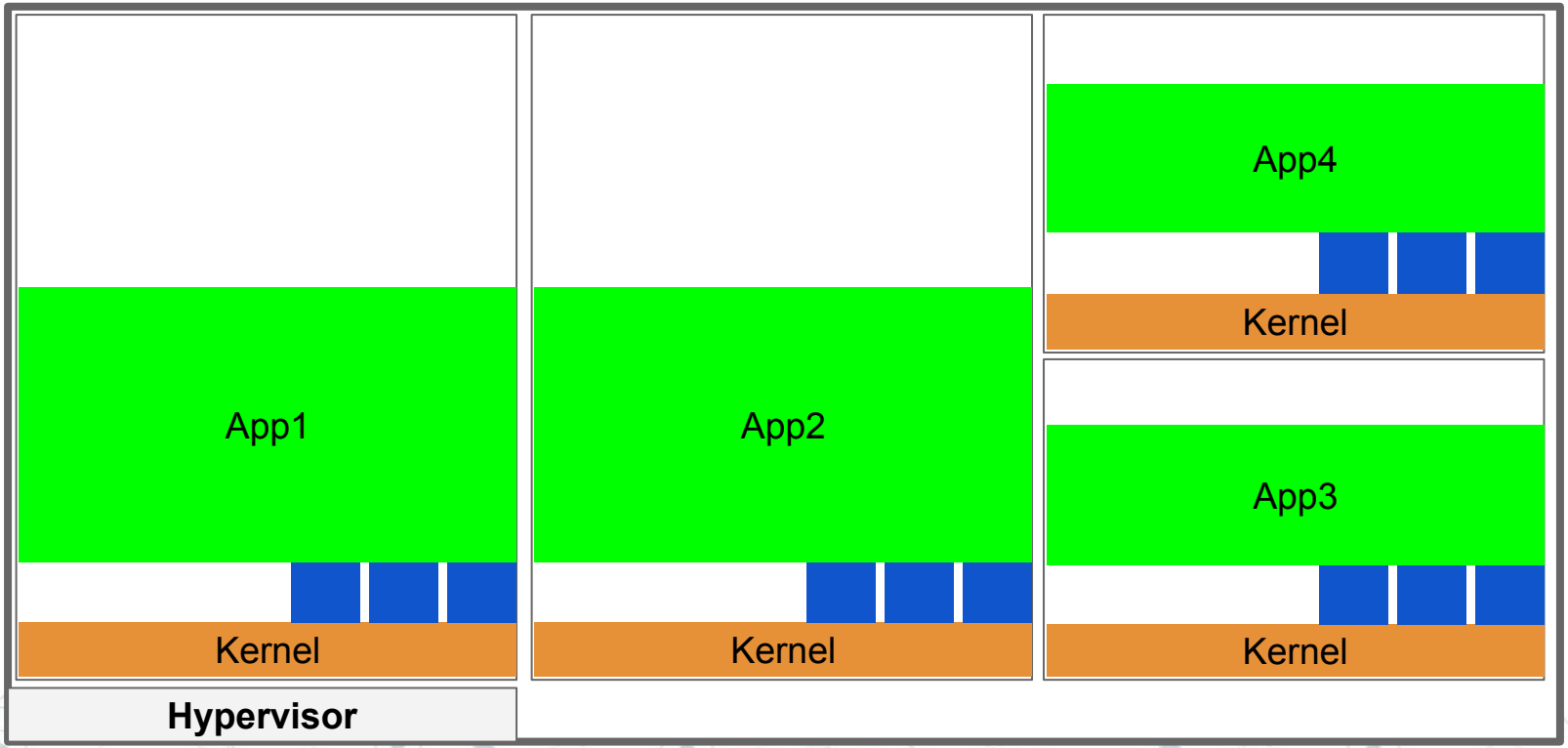
Agenda

- Wprowadzenie
- Wybrane obiekty API i ich właściwości
- Demo

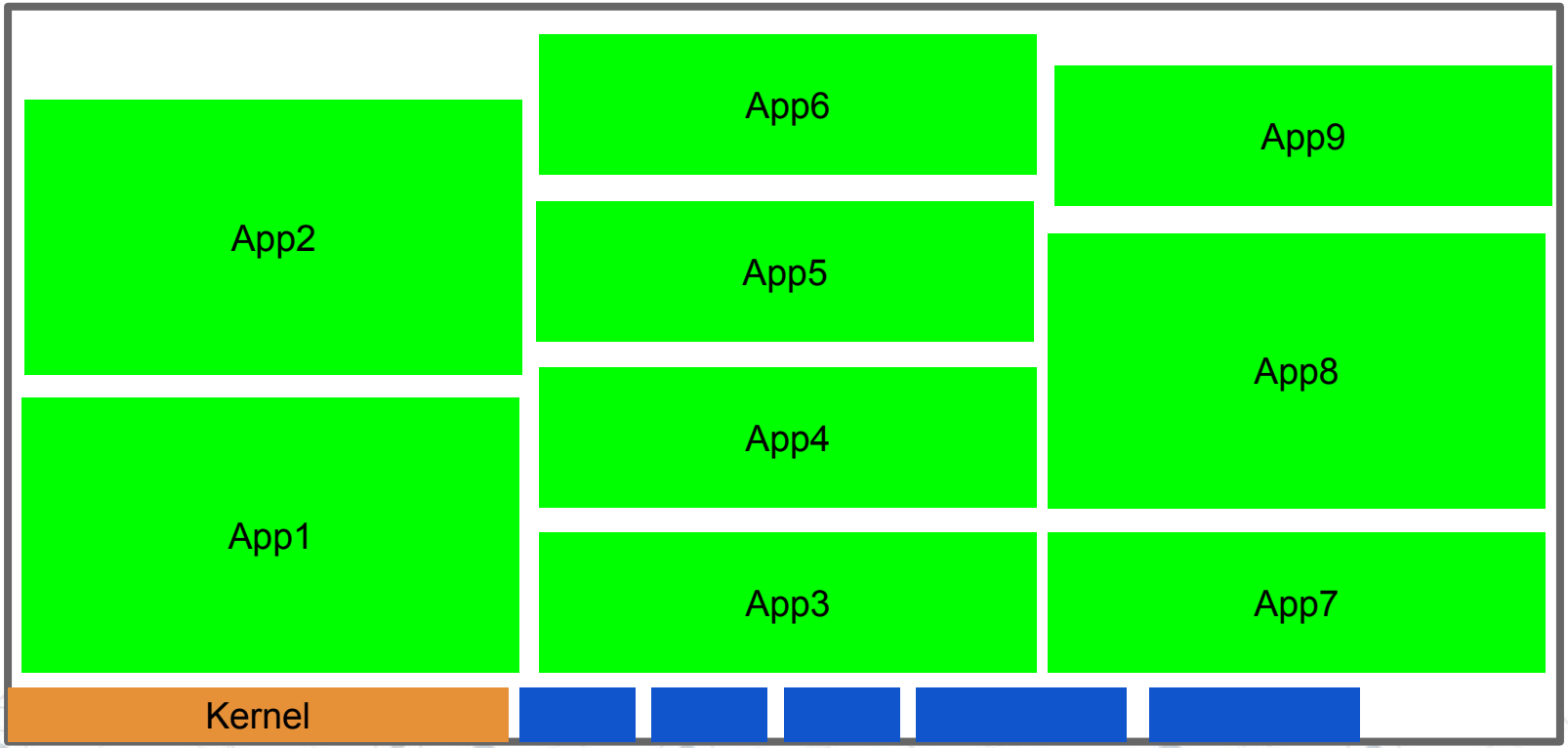
Optymalizacja zasobów

Aspekt sprzętowy

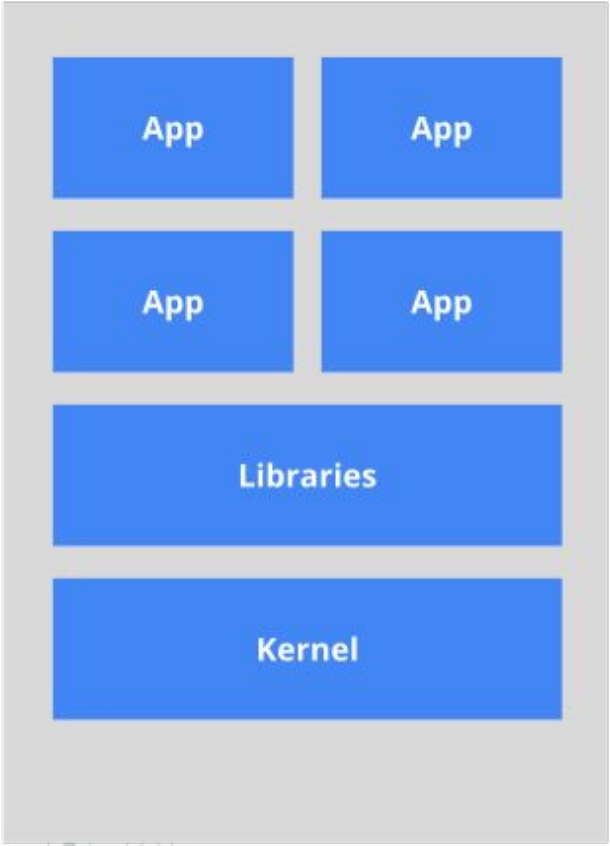
Klasyczna wirtualizacja (Xen, VMWare, KVM, etc.)



Konteneryzacja (Docker)

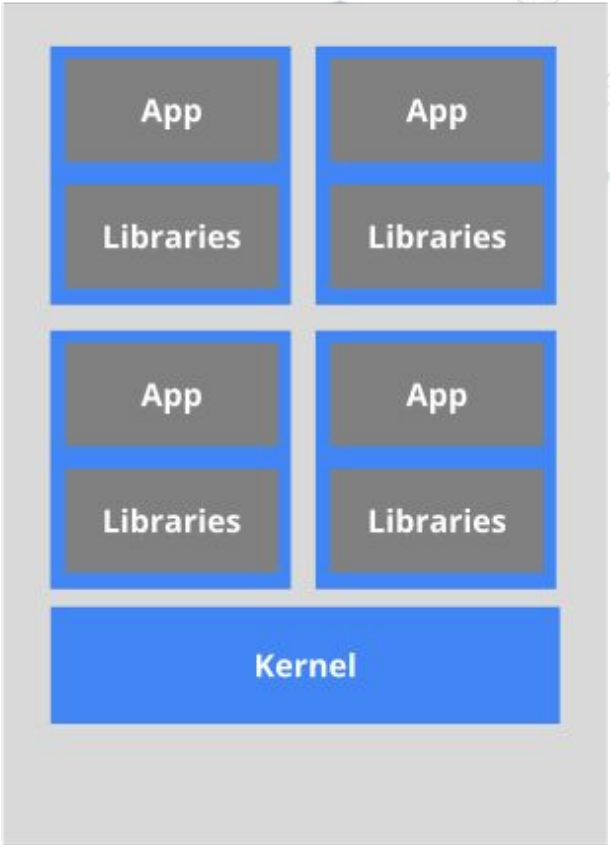


The old way: Applications on host



Heavyweight, non-portable
Relies on OS package manager

The new way: Deploy containers



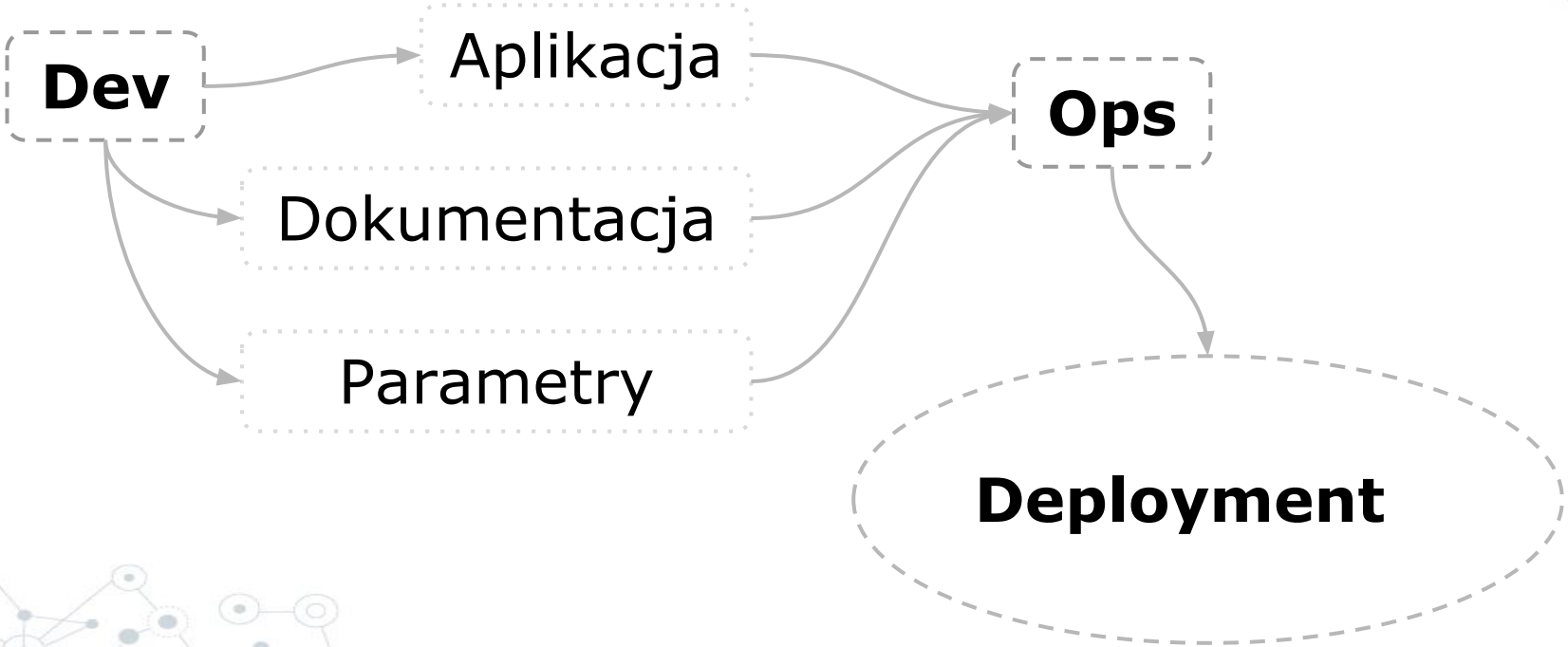
Small and fast, portable
Uses OS-level virtualization

Źródło: https://kubernetes.io

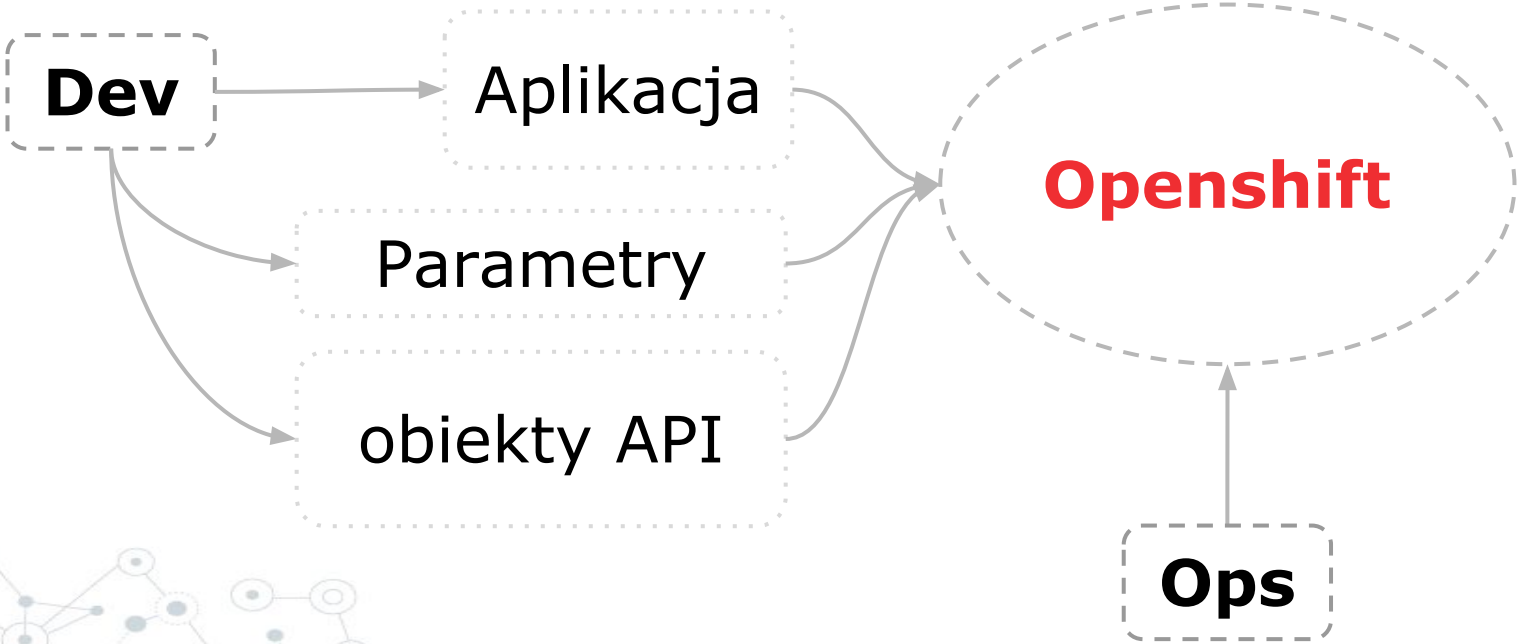
Optymalizacja zasobów

Aspekt ludzki / czasowy

Stara szkoła



Zrównoleglenie procesu /rozdzielenie ról



CI/CD/CD w firmach jest jak seks w liceum

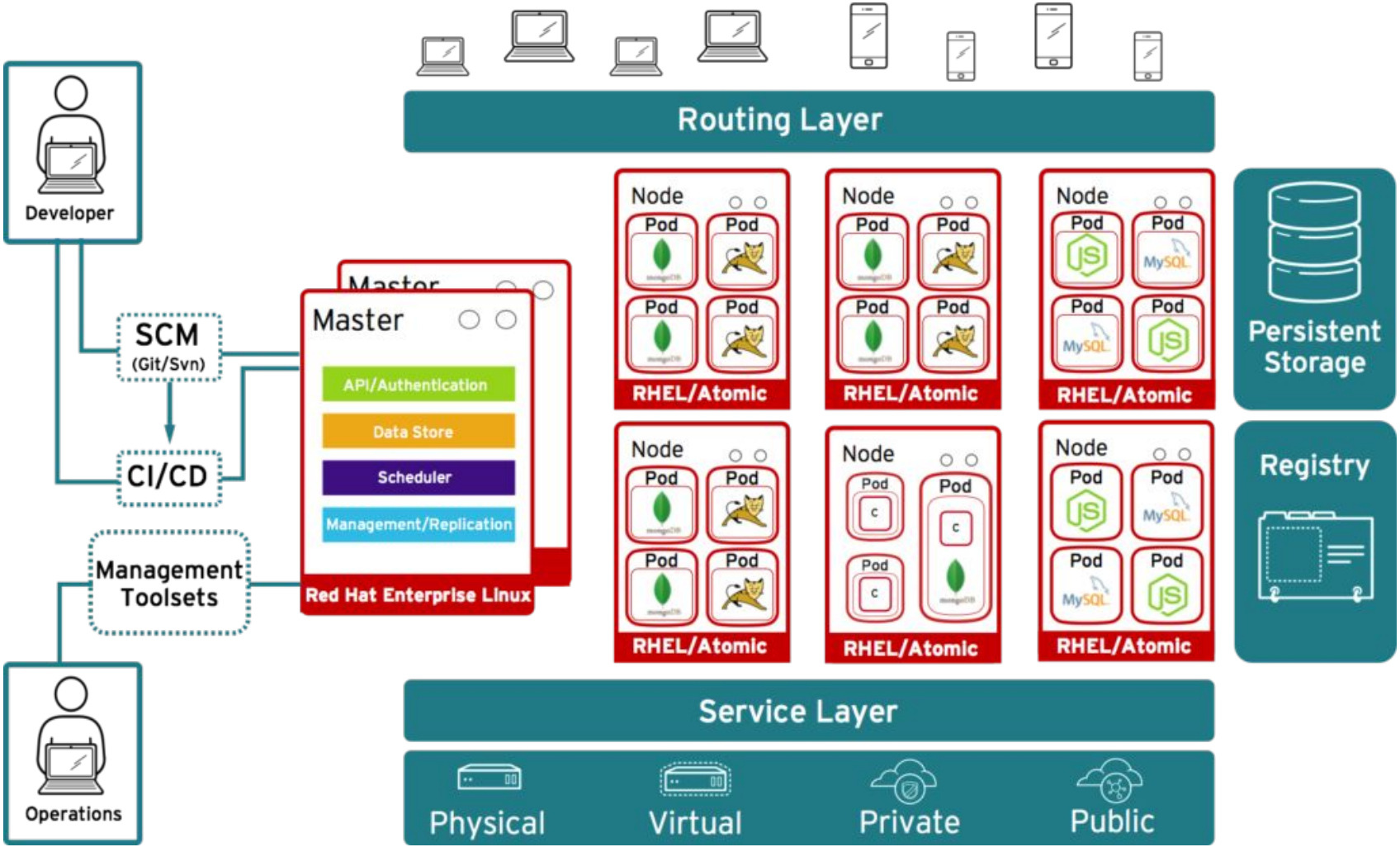
PaaS

kubernetes

by Google

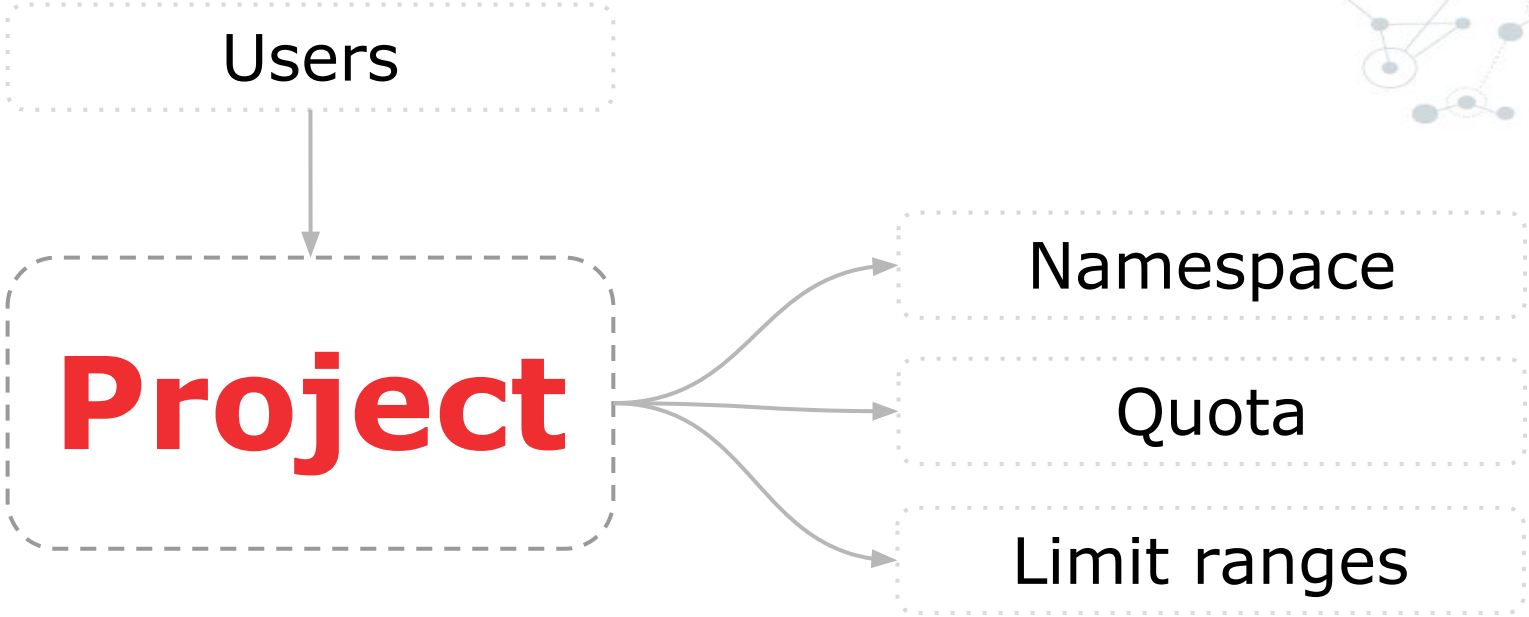
Źródło: https://kubernetes.io

Źródło: https://blog.openshift.com/building-kubernetes-bringing-google-scale-container-orchestration-to-the-enterprise/



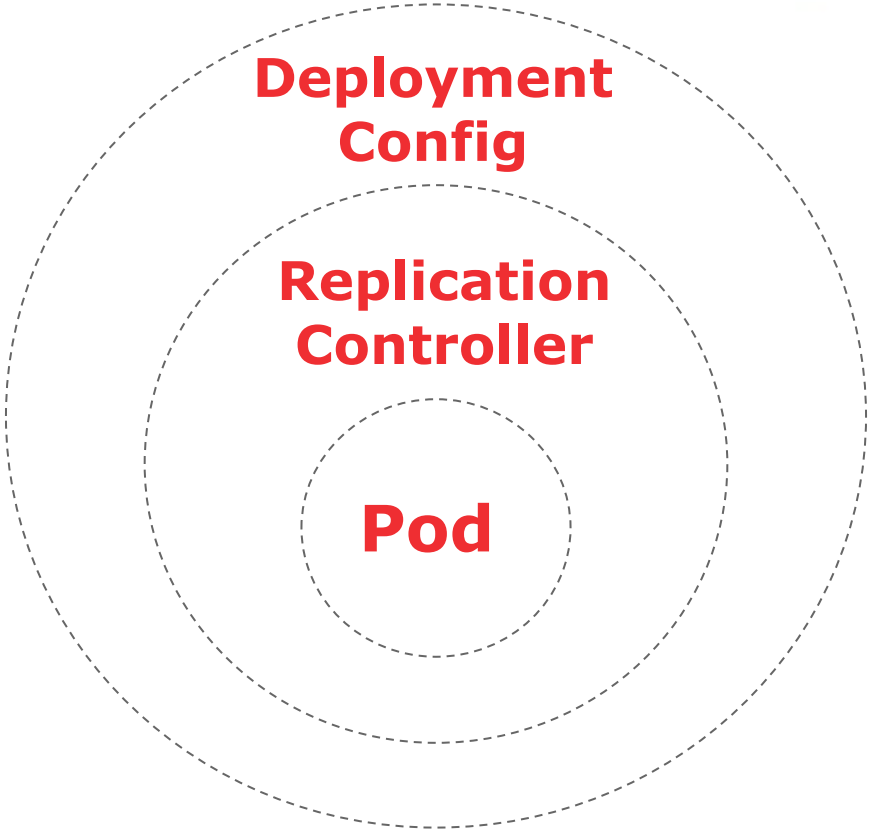
Źródło: https://www.cncf.io/blog/2016/08/23/deploying-1000-nodes-of-openshift-on-the-cncf-cluster-part-1/

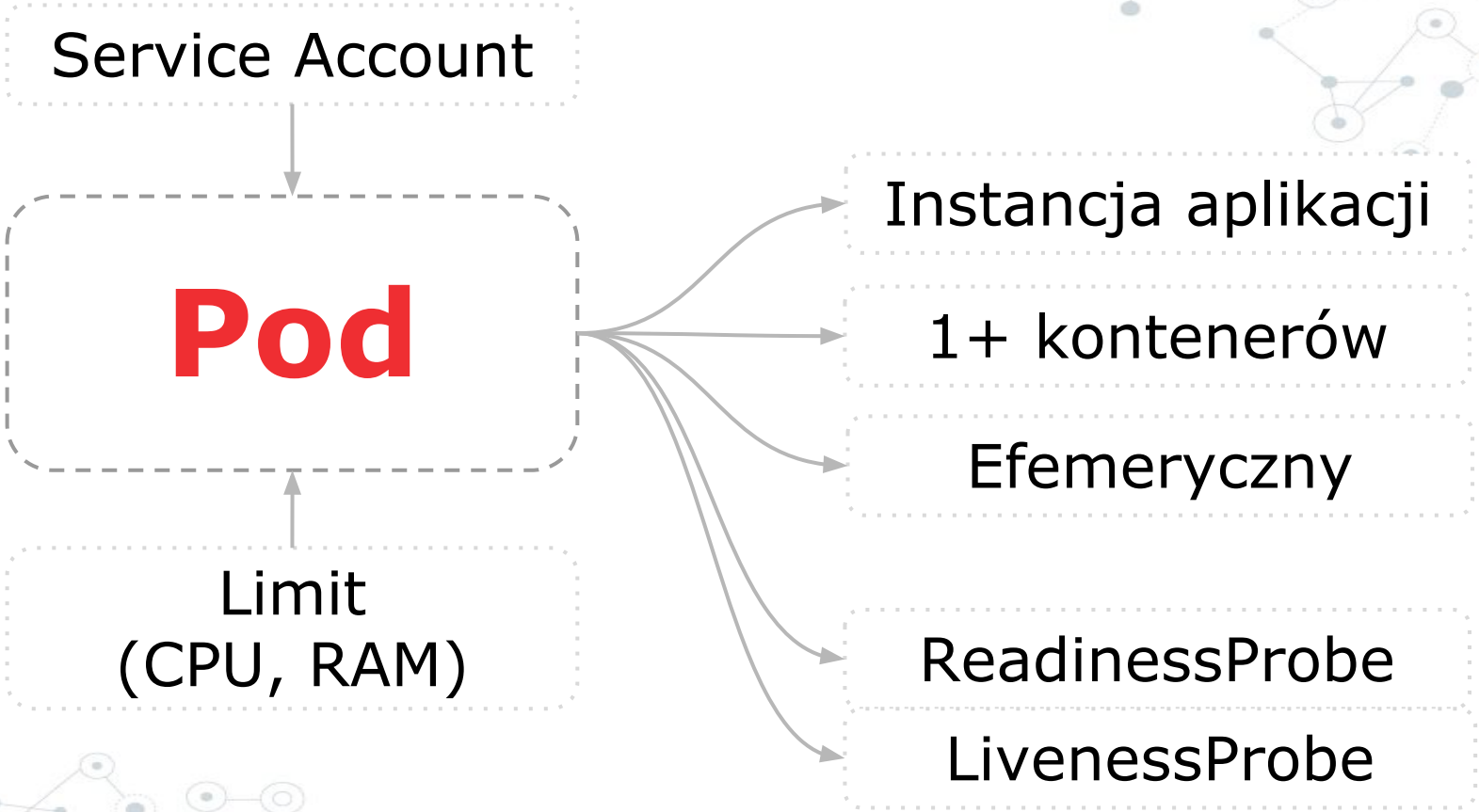
Obiekty API


Users
Project
Namespace
Quota
Limit ranges

ImageStream

- isTag
- isImage
- Reprezentacja zbioru image-y dockera
- Trigger
- Promocja
- Import


Deployment Config
Replication Controller
Pod


Service Account
Pod
Limit
(CPU, RAM)
Instancja aplikacji
1+ kontenerów
Efemeryczny
ReadinessProbe
LivenessProbe

Pod

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: jug-app-pod
  labels:
    app: jug
spec:
  containers:
  - name: jug-app
    image: 172.30.1.1:5000/dev/jug@sha256:7123433f5deb3451e5c3898f6fe…
    ports:
    - containerPort: 8080
      protocol: TCP
```

Pilnowanie zadanej liczby
Pod-ów (replik)

ReplicationController

“Zatrzaśnięta” wersja
obrazu aplikacji
(@sha256)

ReplicationController

```yaml
apiVersion: v1
kind: ReplicationController
metadata:
  labels:
    app: jug
  name: jug-8
spec:
  replicas: 4
  selector:
    app: jug
  template:
    metadata:
      labels:
        app: jug
    spec:
      containers:
        <DEFINICJA POD-a>
```

Wersjonowanie aplikacji

DeploymentConfig

Strategie

Rolling update

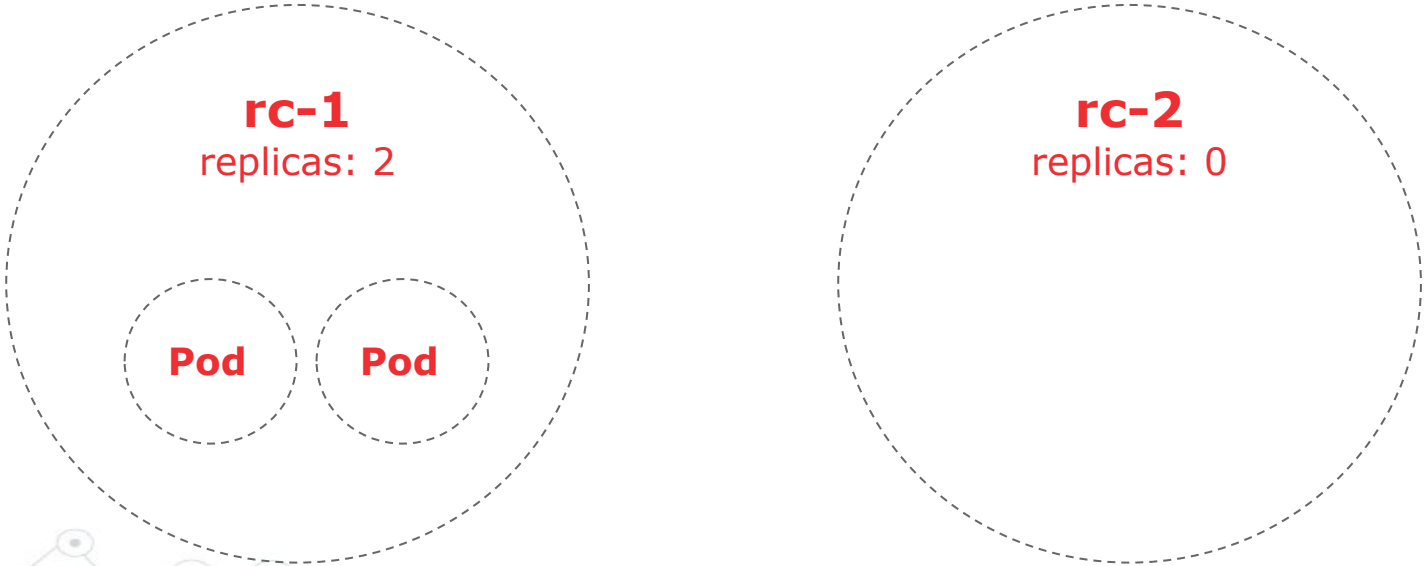
Recreate

Custom

Hook

Trigger

DeploymentConfig - strategia: Recreate

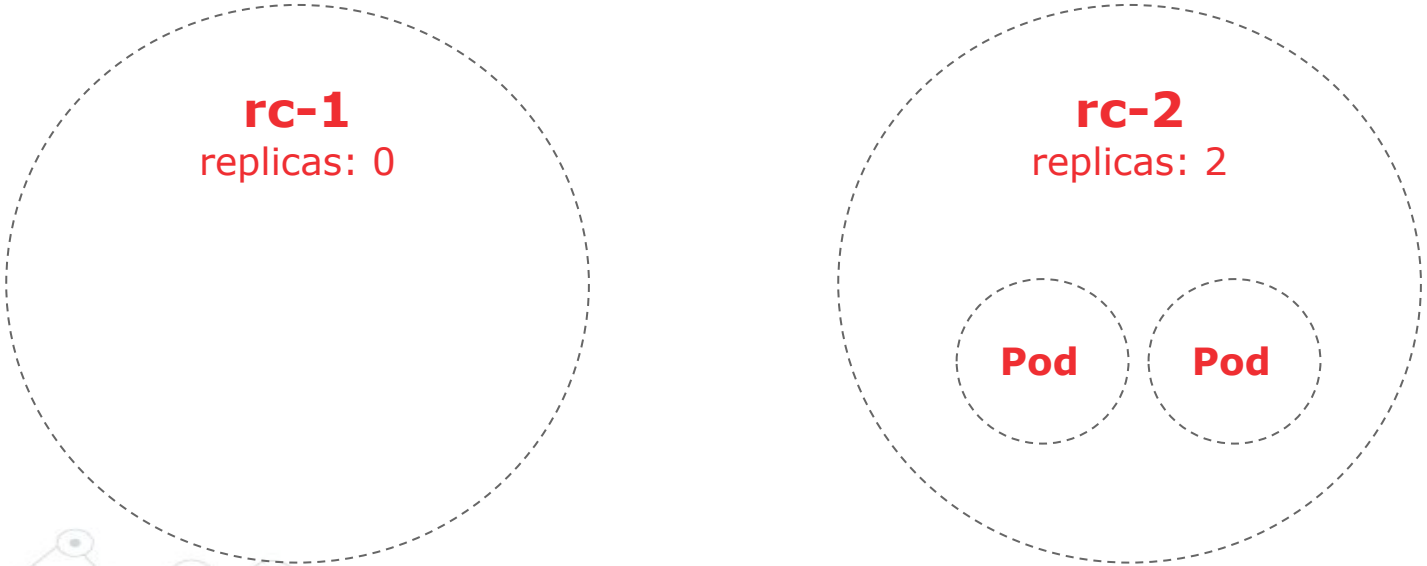


DeploymentConfig - strategia: Recreate

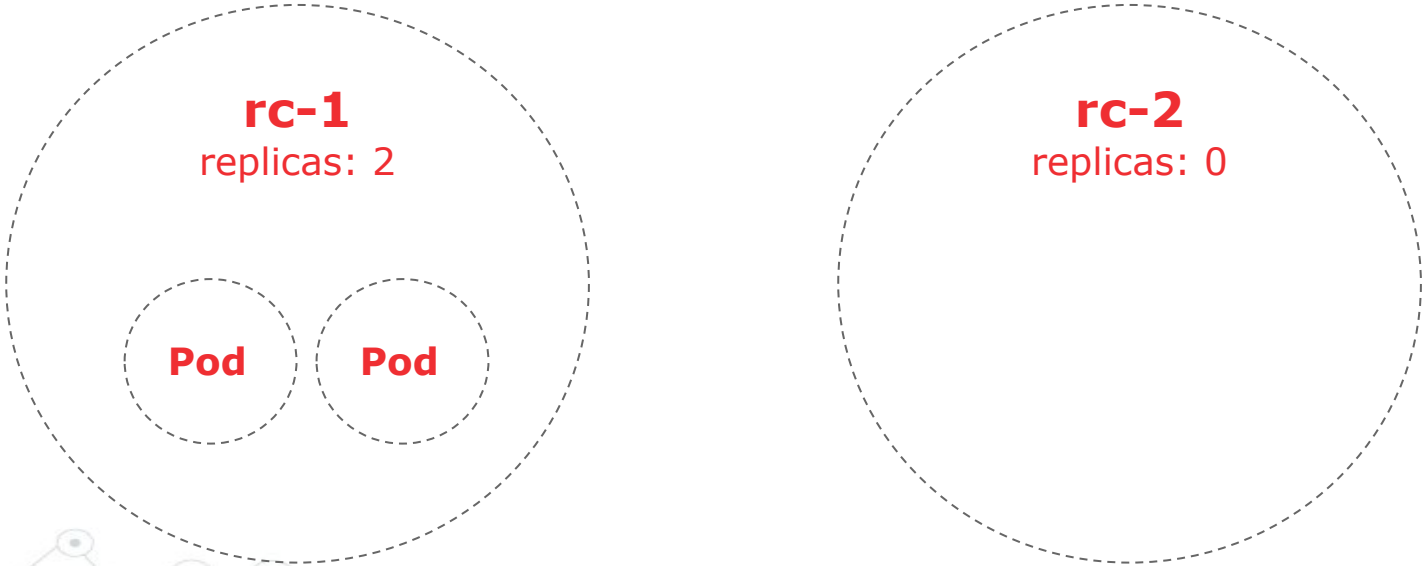
rc-1
replicas: 0

rc-2
replicas: 0

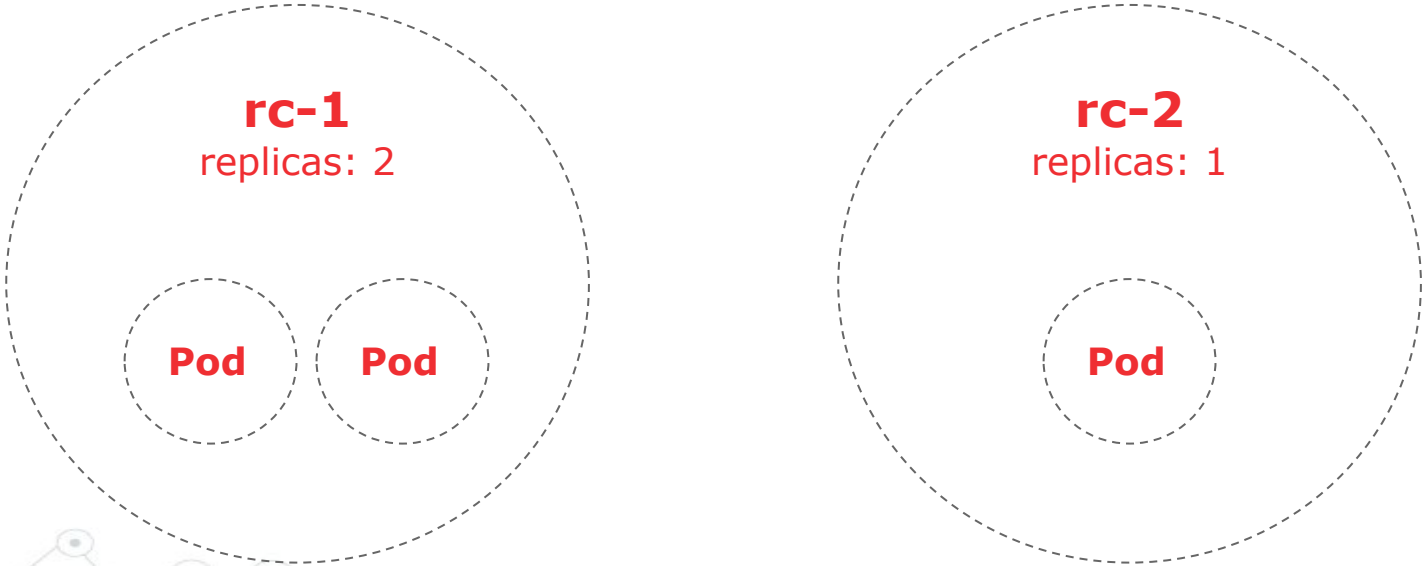
DeploymentConfig - strategia: Recreate



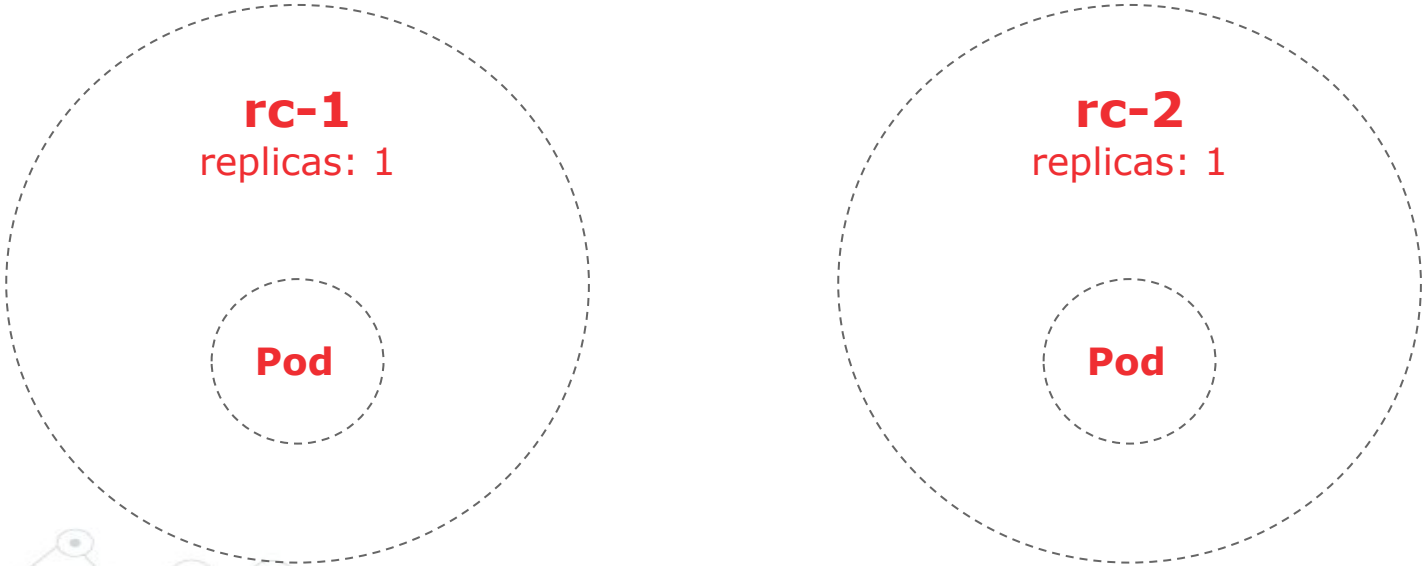
DeploymentConfig - strategia: Rolling



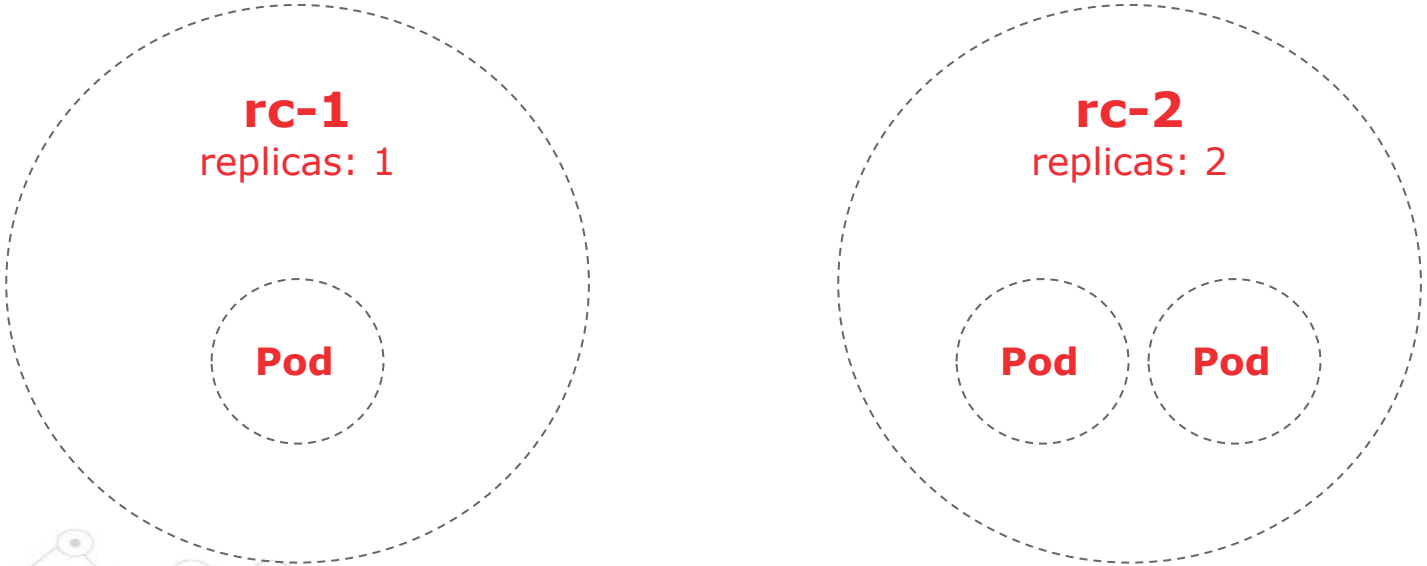
DeploymentConfig - strategia: Rolling



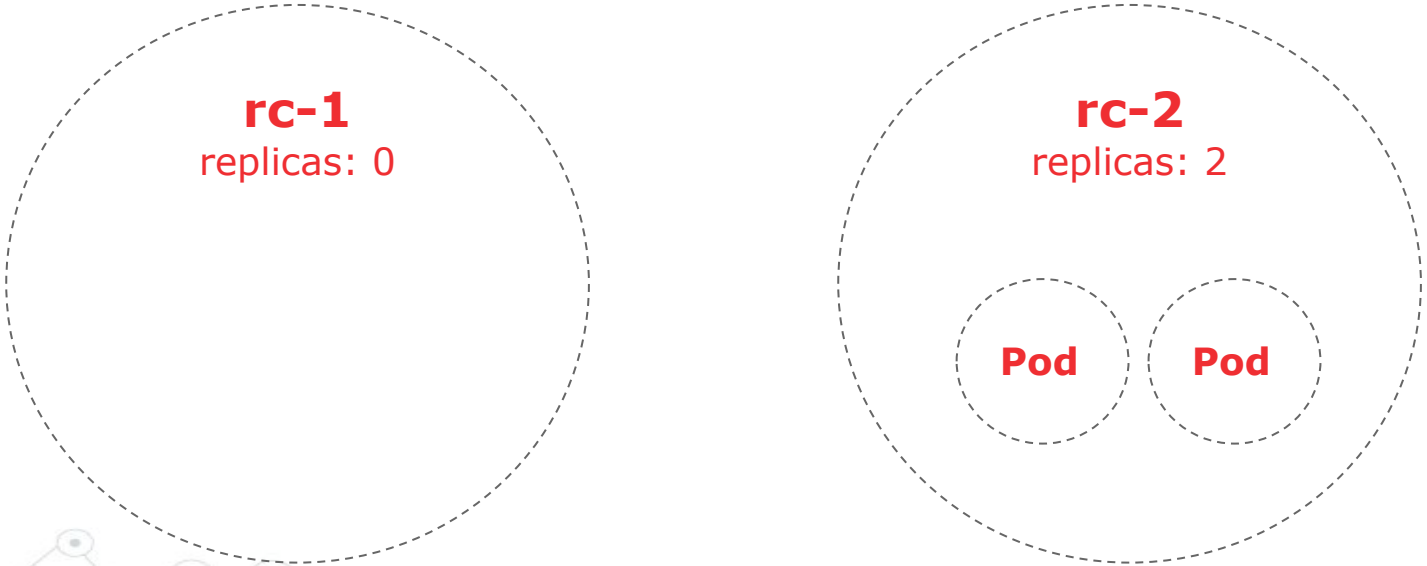
DeploymentConfig - strategia: Rolling

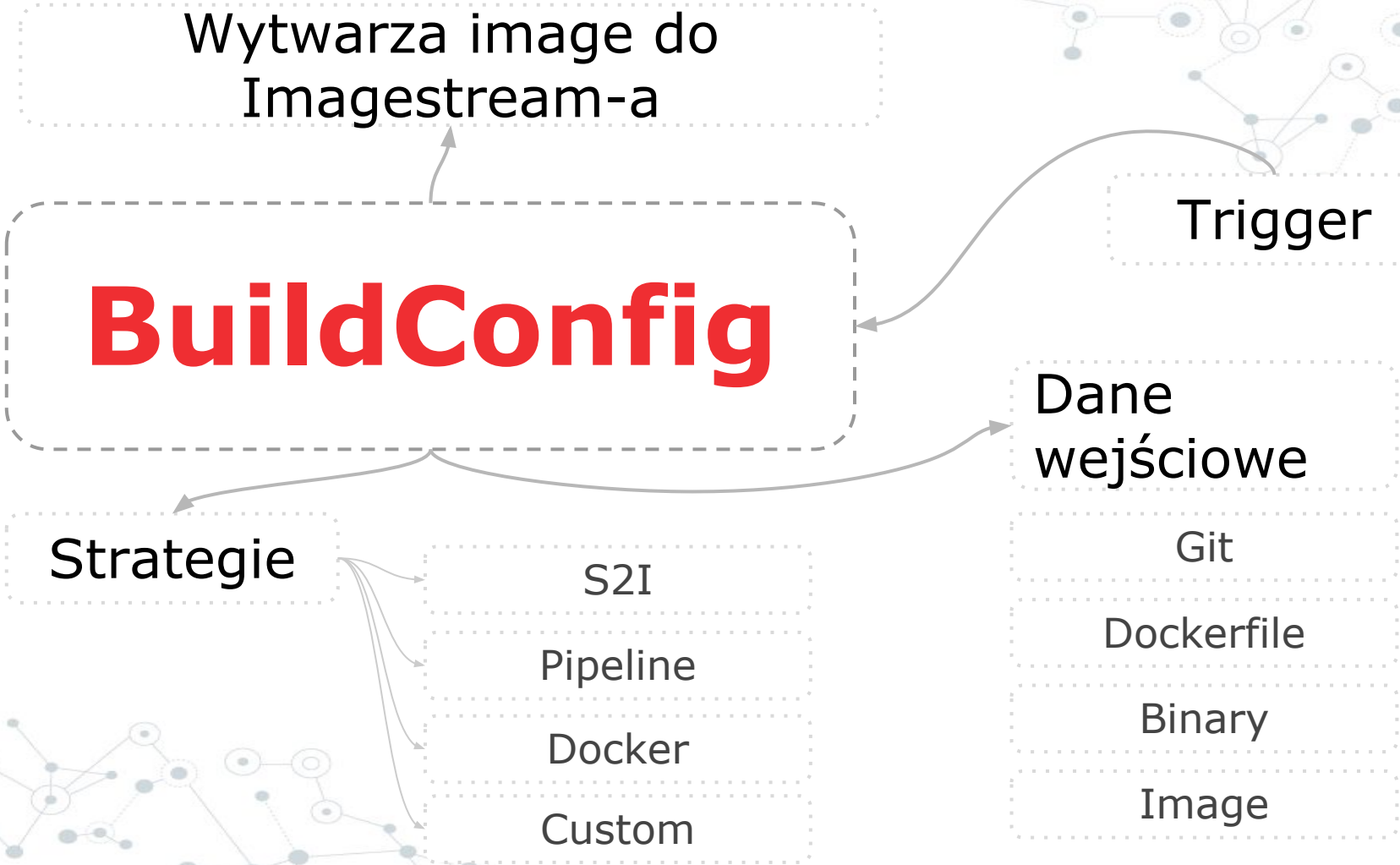


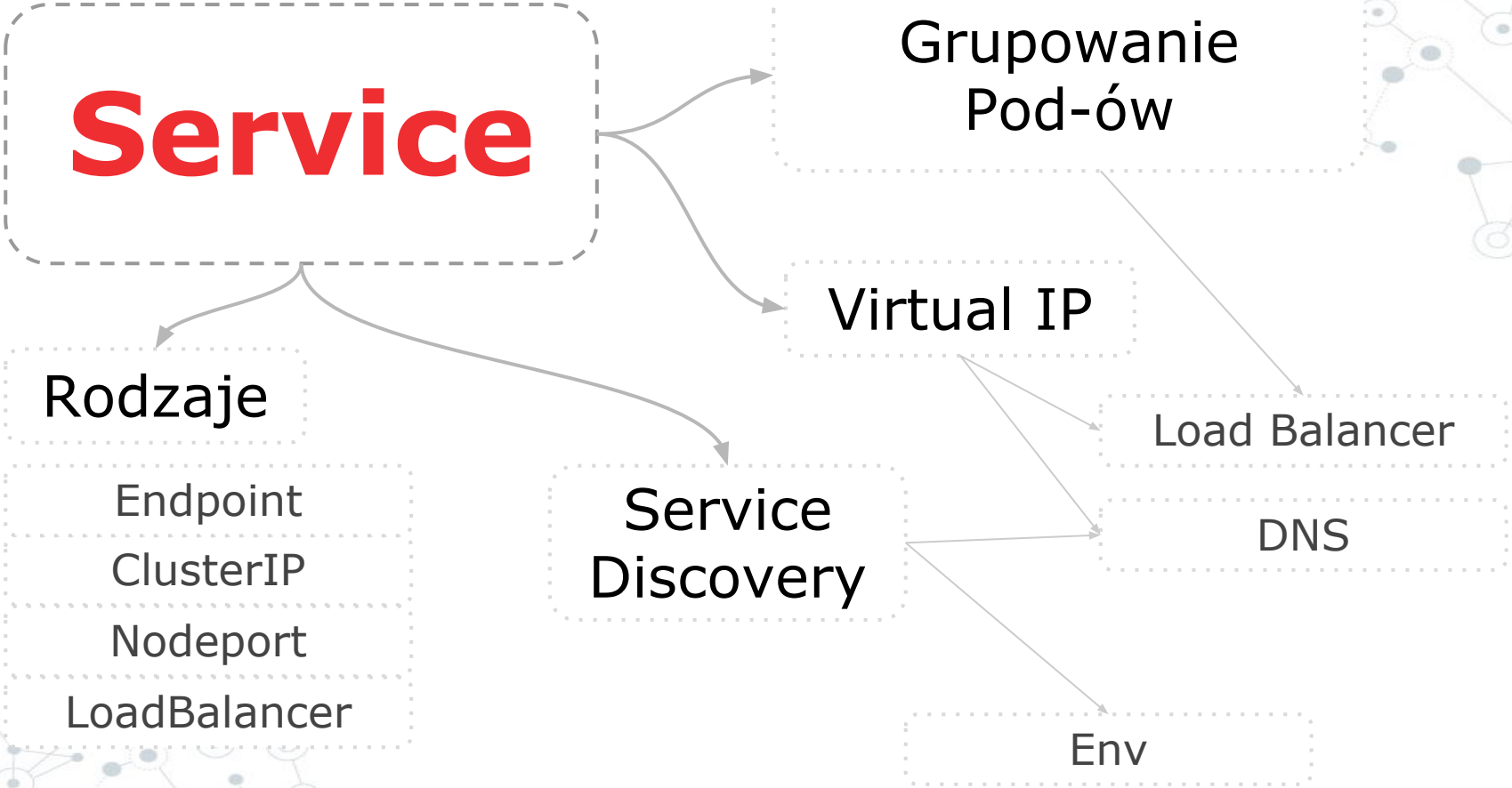
DeploymentConfig - strategia: Rolling



DeploymentConfig - strategia: Rolling



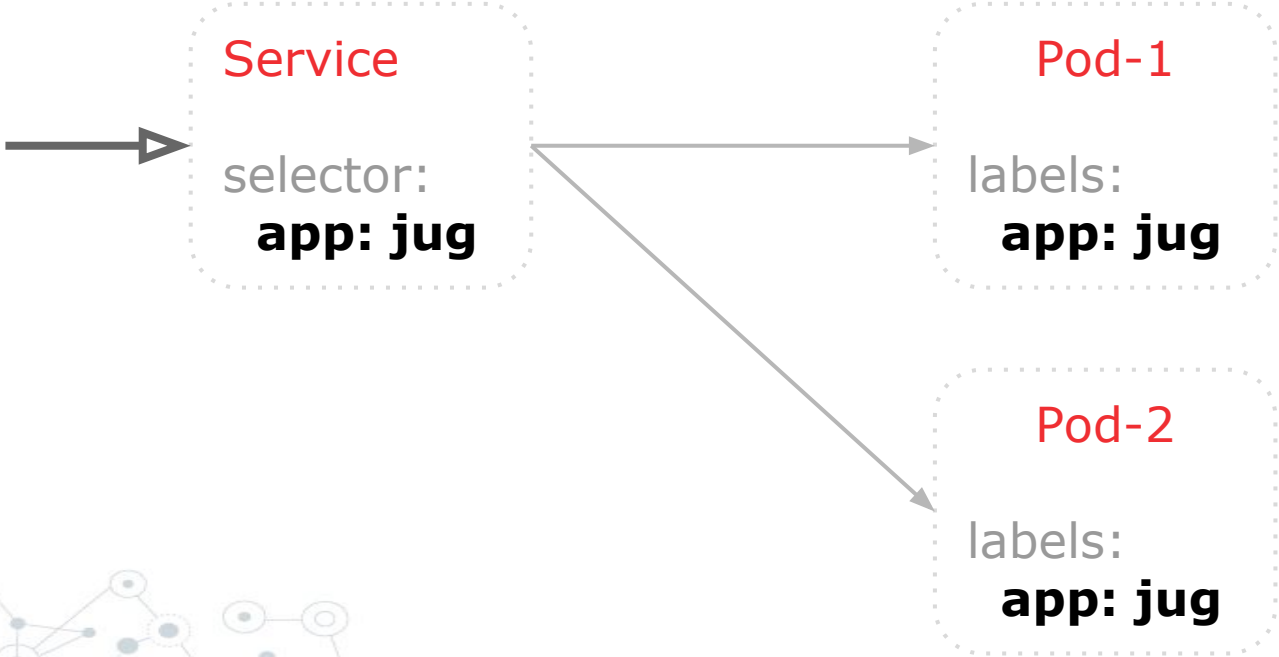

Wytwarza image do Imagestream-a
Trigger
BuildConfig
Dane wejściowe
Strategie
S2I
Pipeline
Docker
Custom
Git
Dockerfile
Binary
Image


Service
Grupowanie Pod-ów
Virtual IP
Rodzaje
Endpoint
ClusterIP
Nodeport
LoadBalancer
Service Discovery
Load Balancer
DNS
Env

Service

```yaml
apiVersion: v1
kind: Service
metadata:
  labels:
    app: jug
  name: jug
spec:
  ports:
  - name: czesiek
    port: 8080
    protocol: TCP
    targetPort: 8080
  selector:
    app: jug
  sessionAffinity: None
  type: ClusterIP
```

Label / Selector



Service Discovery

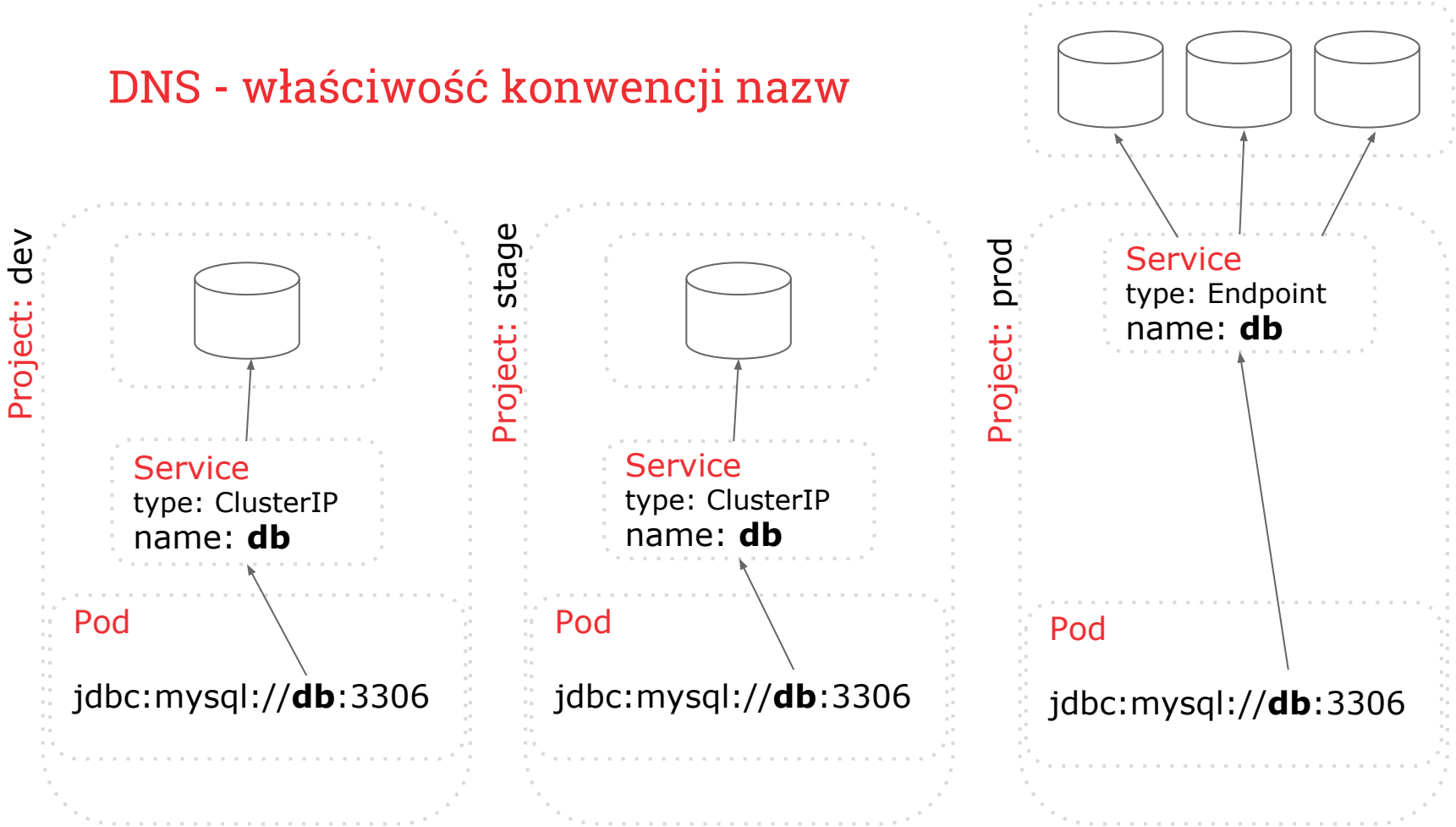
Project
name: dev

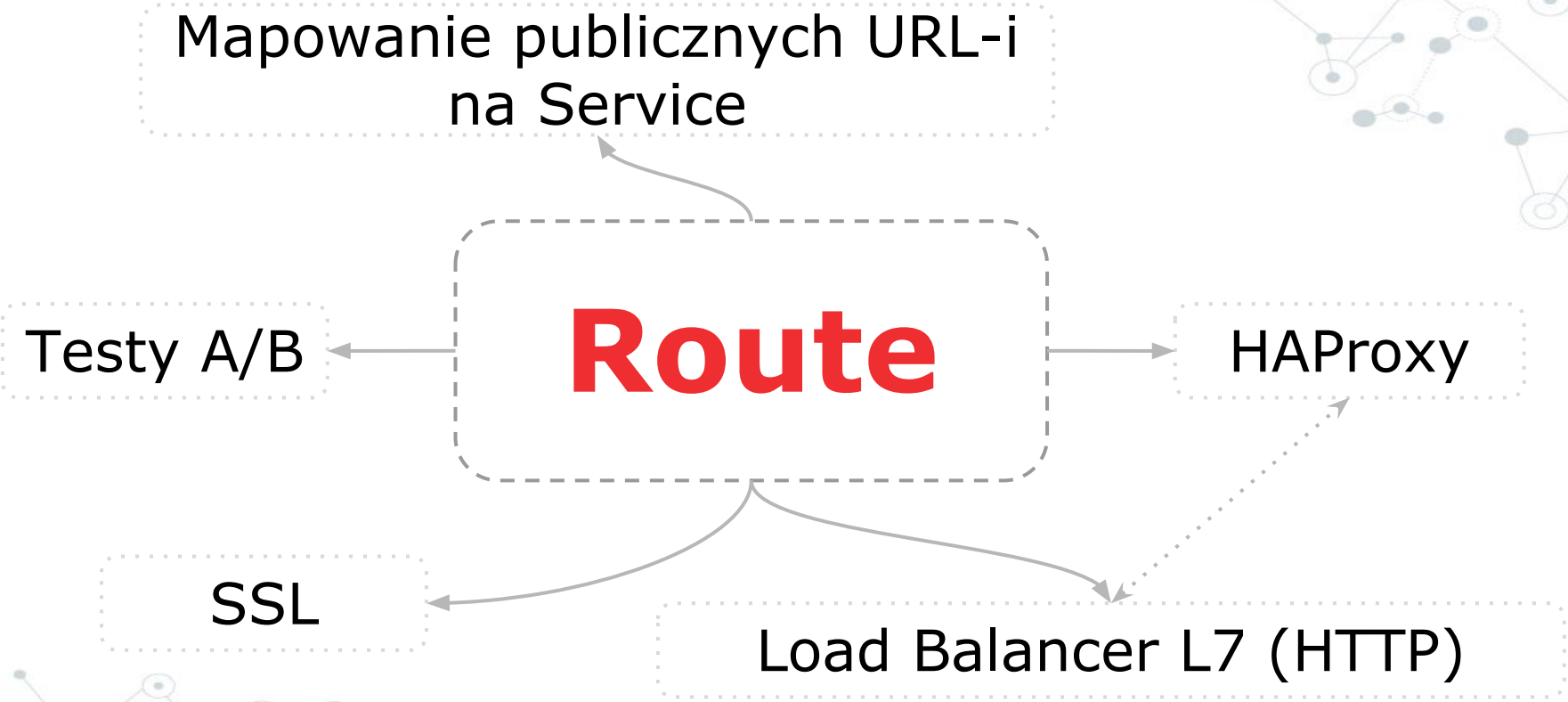
Service
name: jug

DNS (wewnątrz klastra):
jug.dev.svc

ENV:
JUG_PORT_8080_TCP_ADDR=172.30.241.153
JUG_PORT_8080_TCP=tcp://172.30.241.153:8080
JUG_PORT=tcp://172.30.241.153:8080
JUG_SERVICE_HOST=172.30.241.153
JUG_SERVICE_PORT=8080
JUG_PORT_8080_TCP_PORT=8080
JUG_PORT_8080_TCP_PROTO=tcp
JUG_SERVICE_PORT_CZESIEK=8080

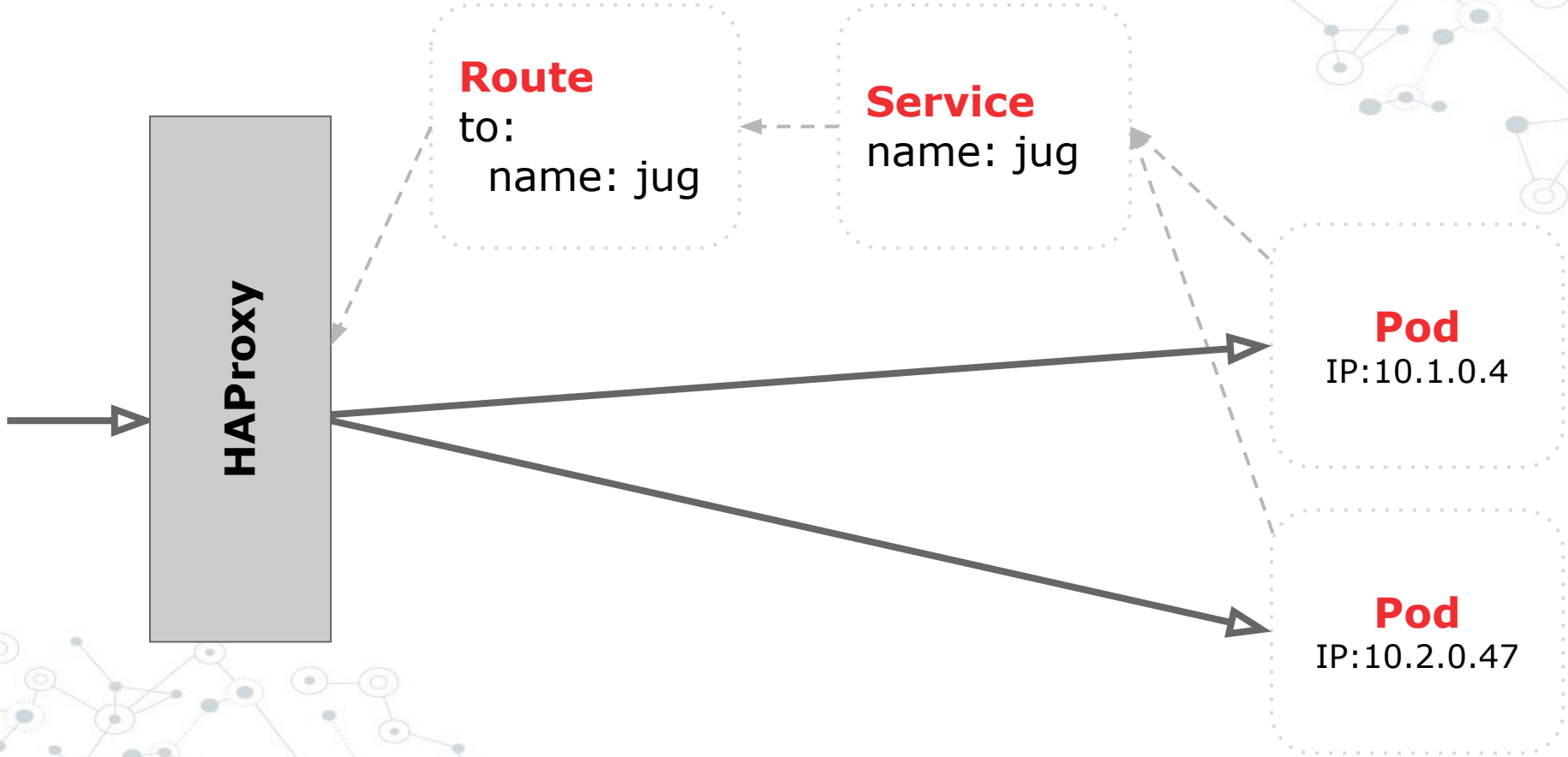
DNS - właściwość konwencji nazw




Mapowanie publicznych URL-i na Service
Testy A/B
Route
HAProxy
SSL
Load Balancer L7 (HTTP)

Route

```yaml
apiVersion: v1
kind: Route
metadata:
  labels:
    app: jug
  name: jug
spec:
  host: jug-dev.192.168.99.100.nip.io
  port:
    targetPort: czesiek
  to:
    kind: Service
    name: jug
    weight: 100
  wildcardPolicy: None
```


Route
to:
name: jug
Service
name: jug
HAProxy
Pod
IP:10.1.0.4
Pod
IP:10.2.0.47

Demo

Credits

Presentation template by SlidesCarnival