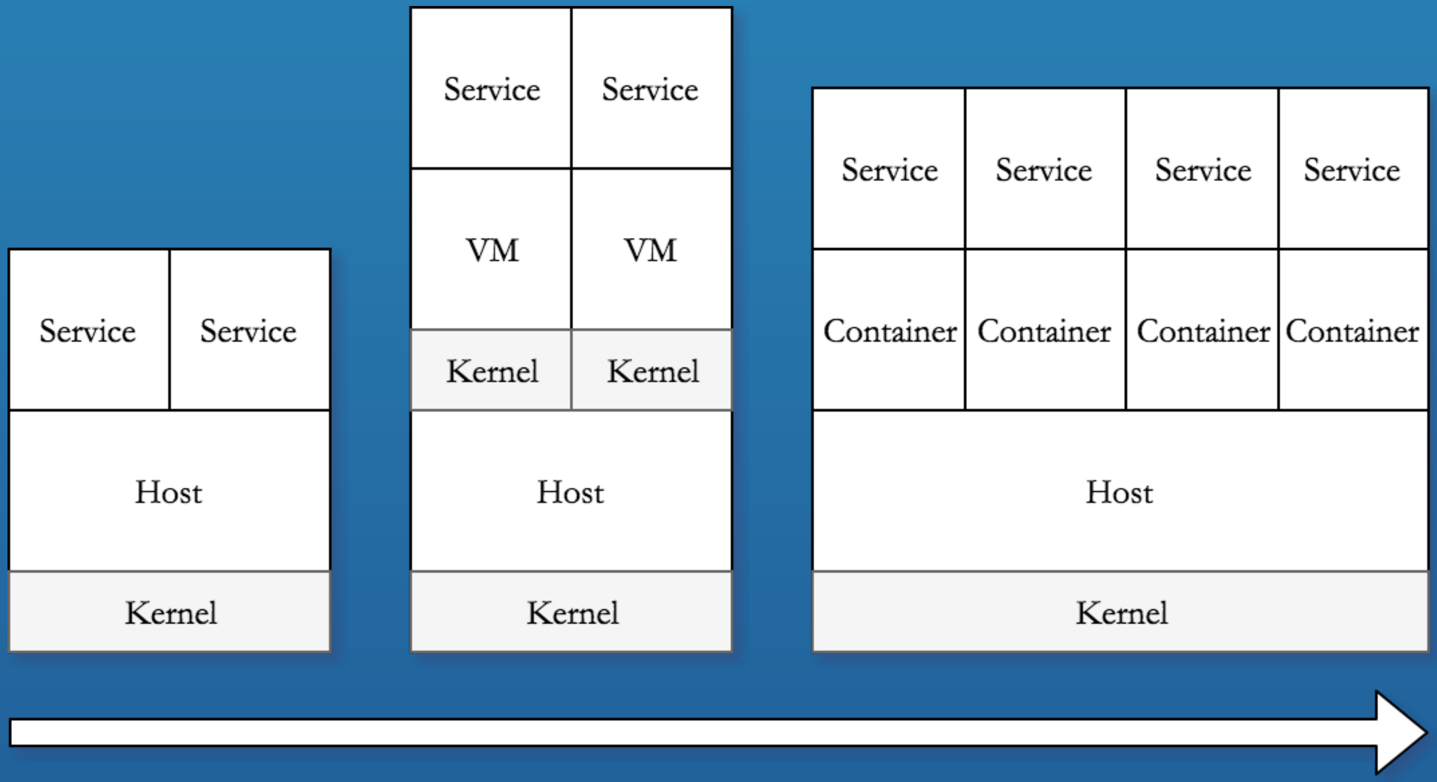
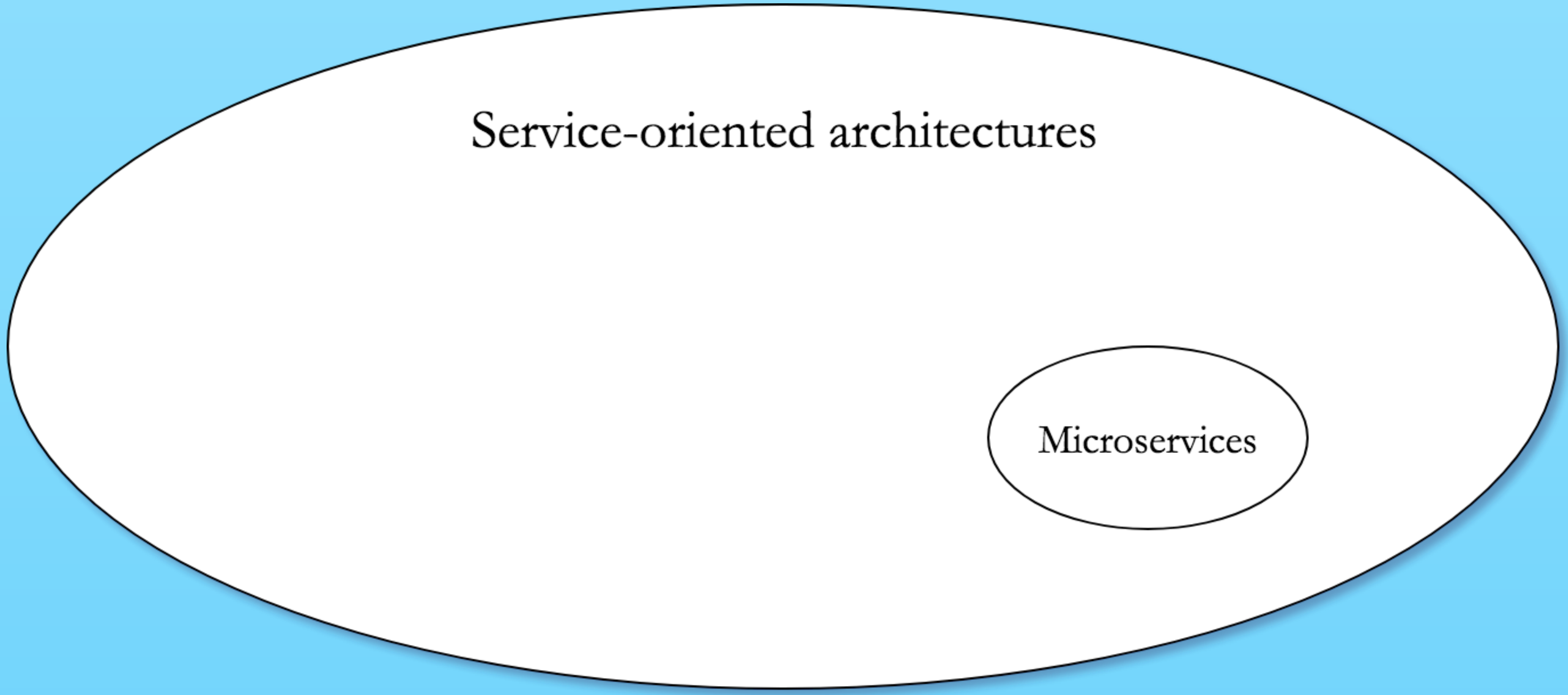
Microservices, containers, orchestration

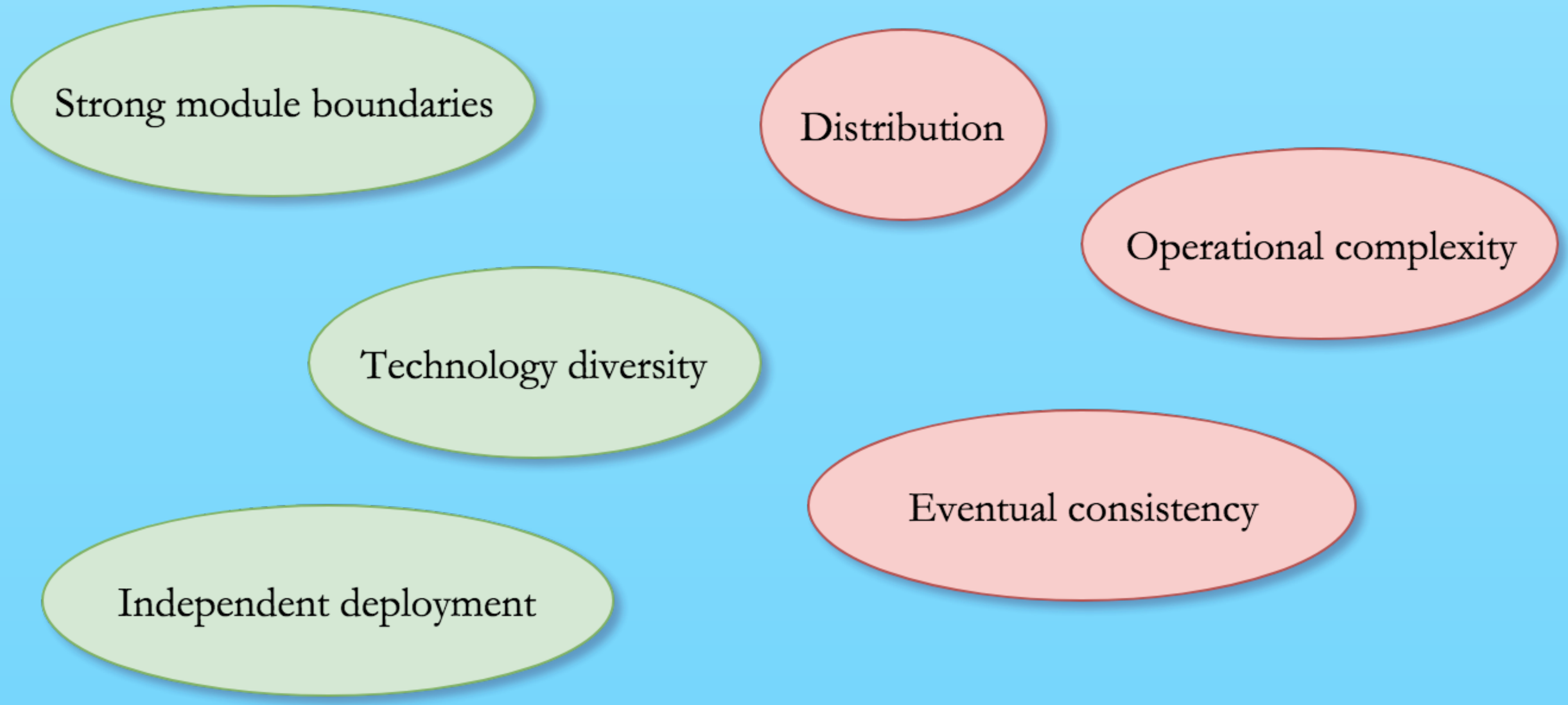


Microservices

A meaningful subset



Benefits and costs



Strong module boundaries

Distribution

Operational complexity

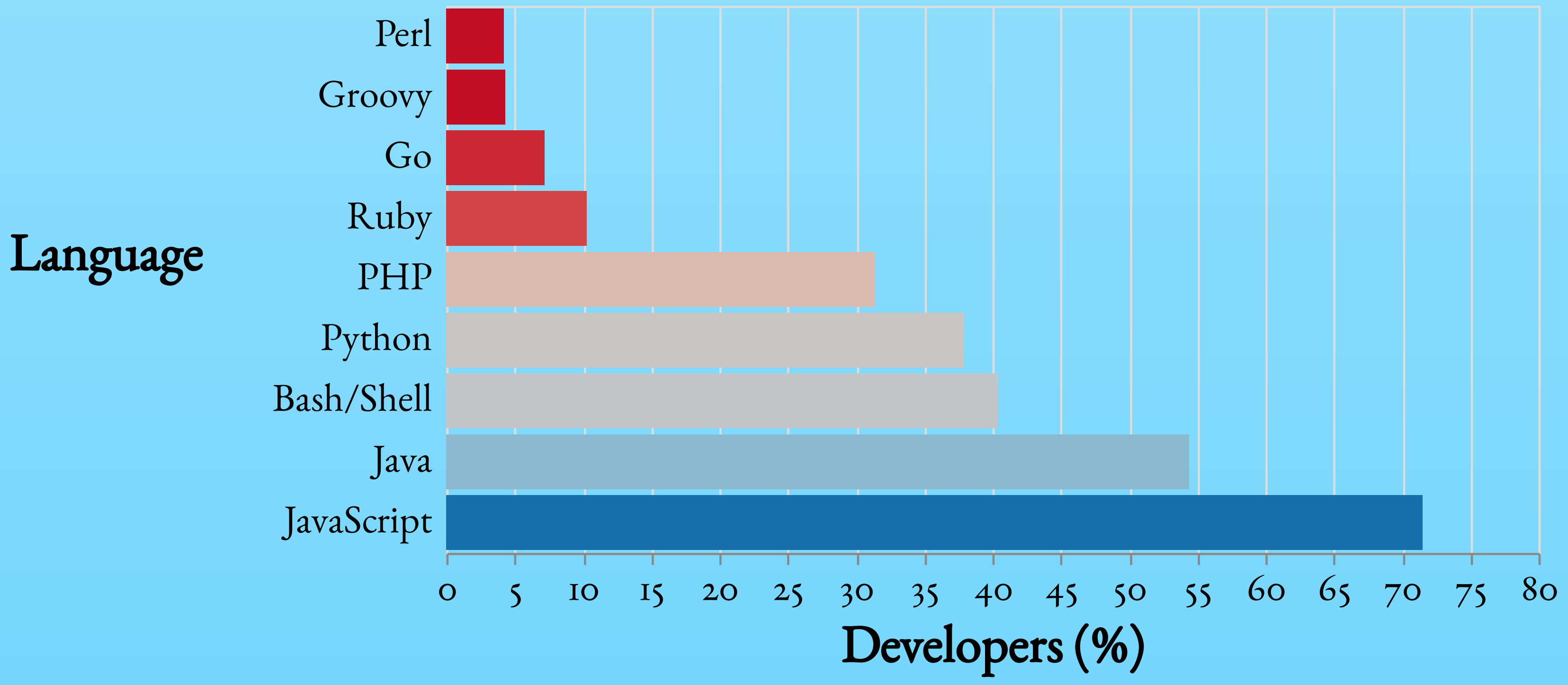
Technology diversity

Eventual consistency

Independent deployment

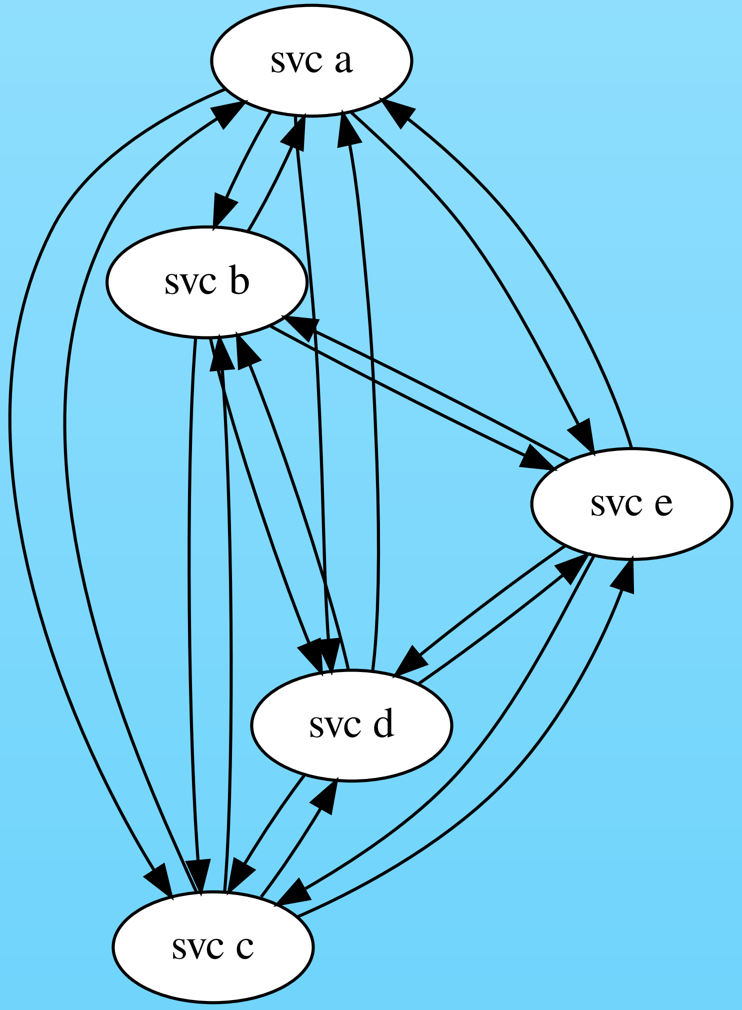
Source: www.martinfowler.com/microservices

Technology diversity



Source: “Programming, scripting and markup languages”, Stack Overflow survey 2018

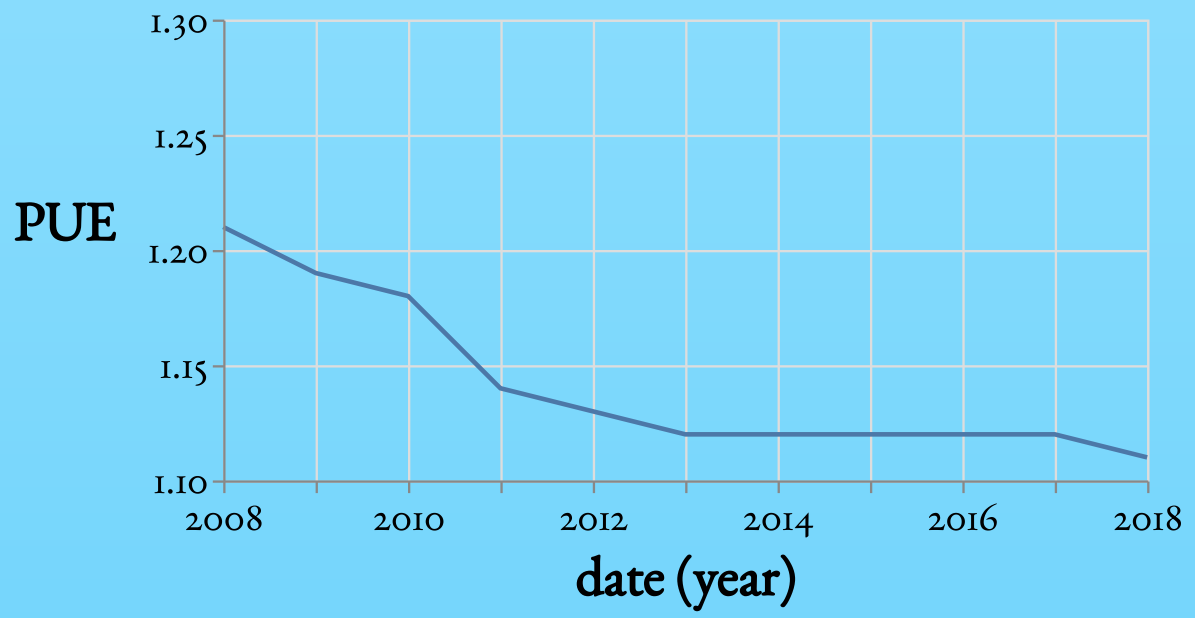
Operational complexity



Containers

Hardware: high efficiency, poor utilisation

$$\text{PUE} = \frac{\text{Watts in}}{\text{IT watts}}$$



Power usage effectiveness data by Google Data Centers

Improving host utilisation

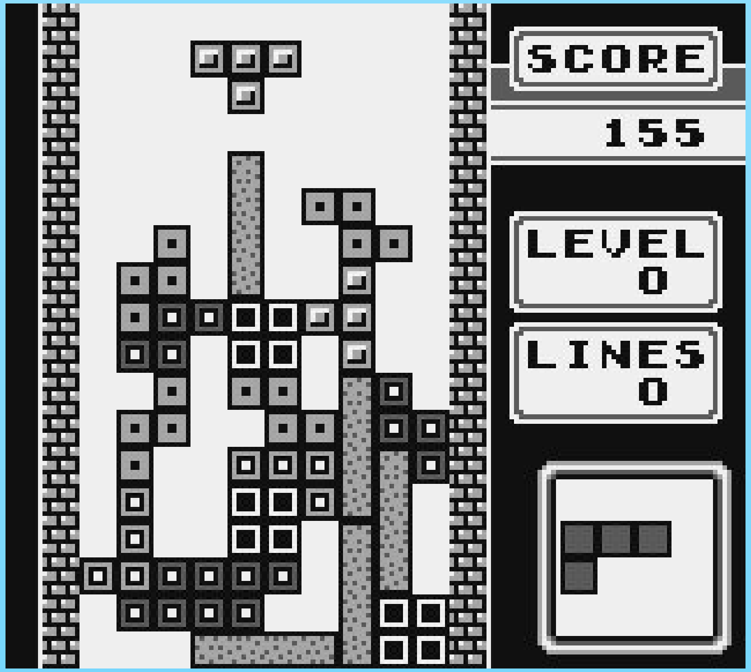


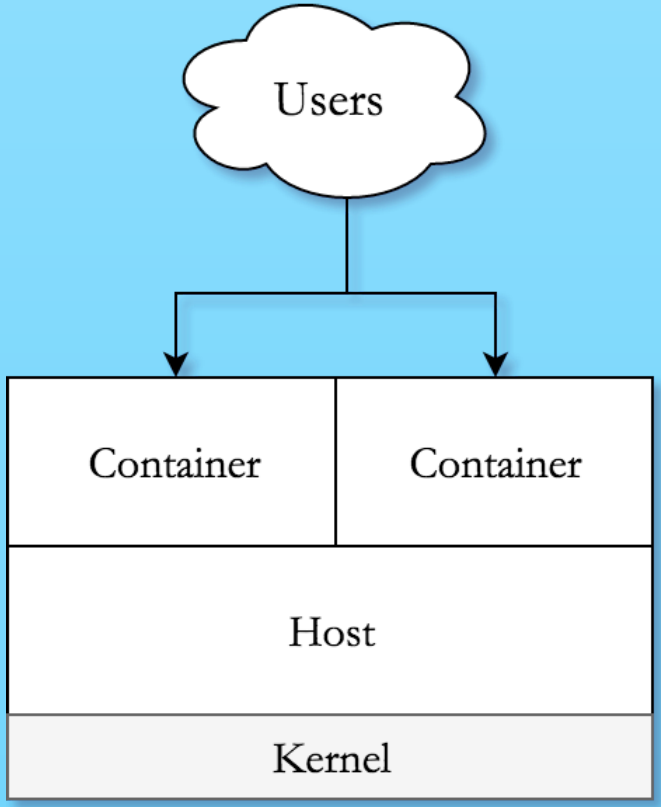
Image source – for a brief history of Kubernetes, see Burns et al., Borg, Omega and Kubernetes

Isolated processes

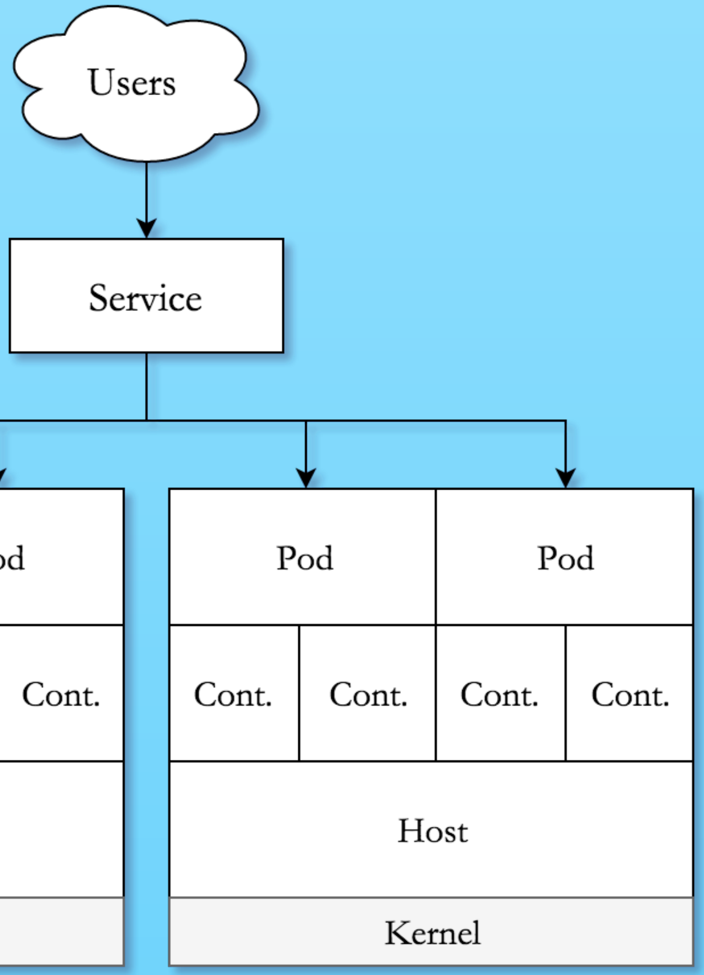
```
$ docker history `docker images | grep hello-openshift | awk '{print $3}'`
IMAGE               CREATED             SIZE                COMMENT
7af3297a3fb4        3 months ago        6.09MB
<missing>           3 months ago        0B                  Imported from -
```

Orchestration

Single-host deployment



Distributed deployment



Users

Service

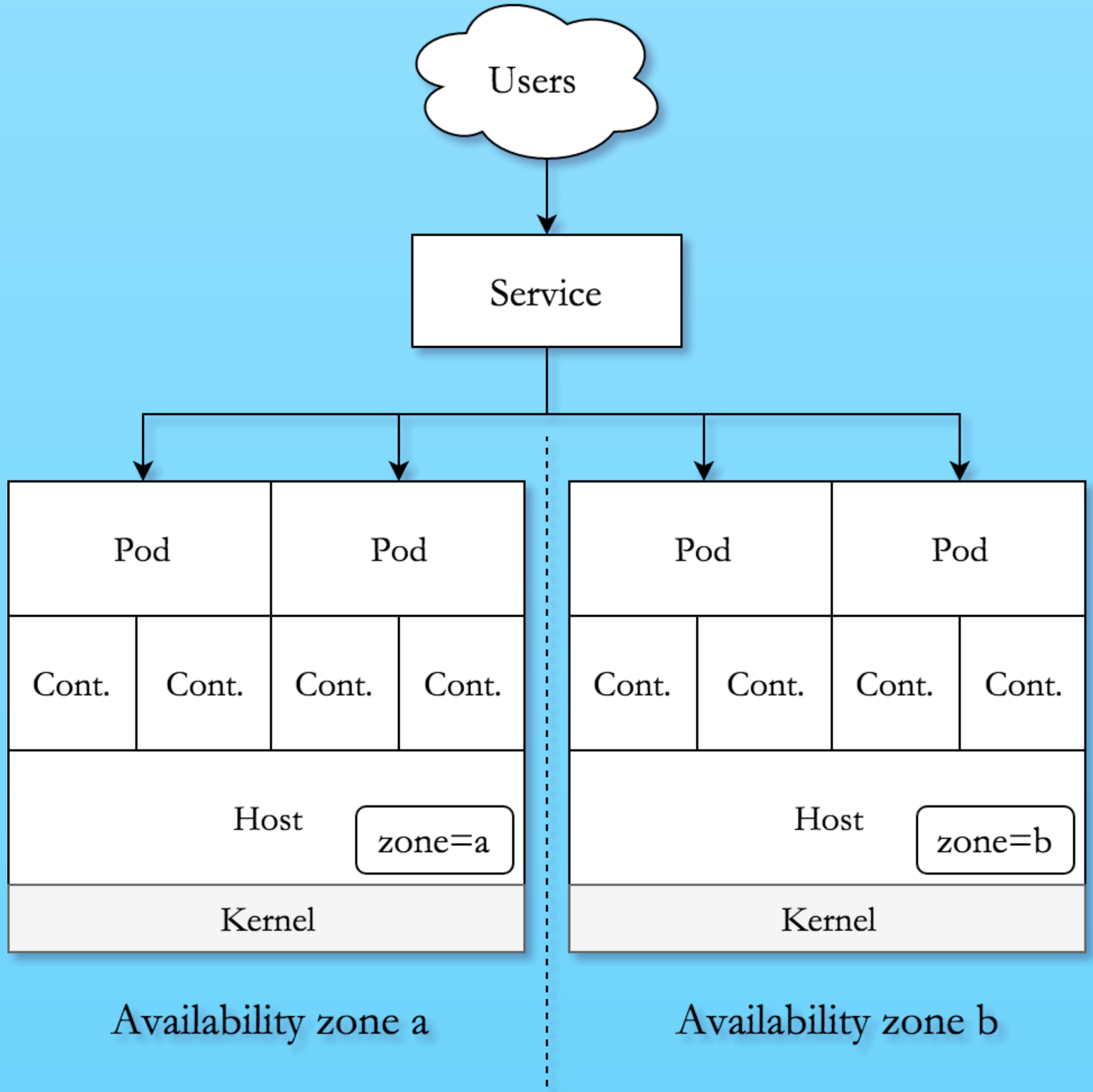
Pod Pod Pod Pod

Cont. Cont. Cont. Cont. Cont. Cont. Cont. Cont.

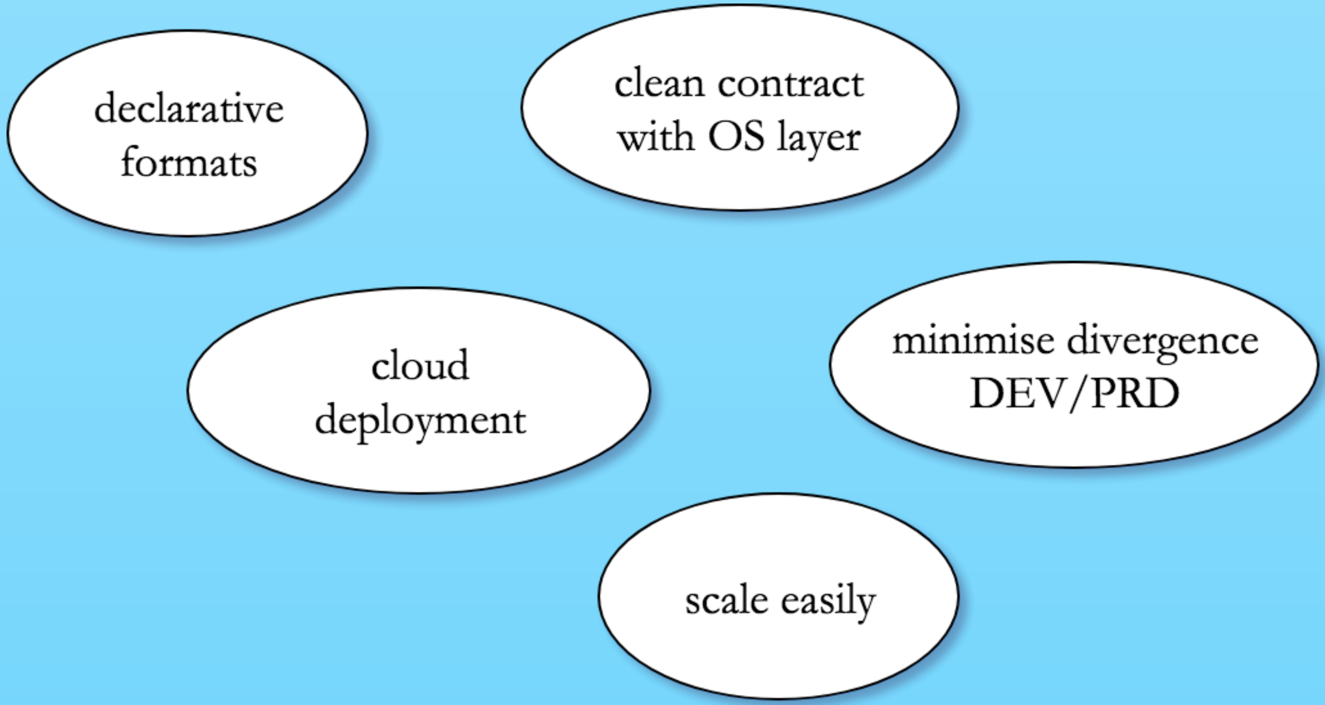
Host Host

Kernel Kernel

Highly available deployment



~~Twelve~~Five factors



declarative formats

clean contract with OS layer

cloud deployment

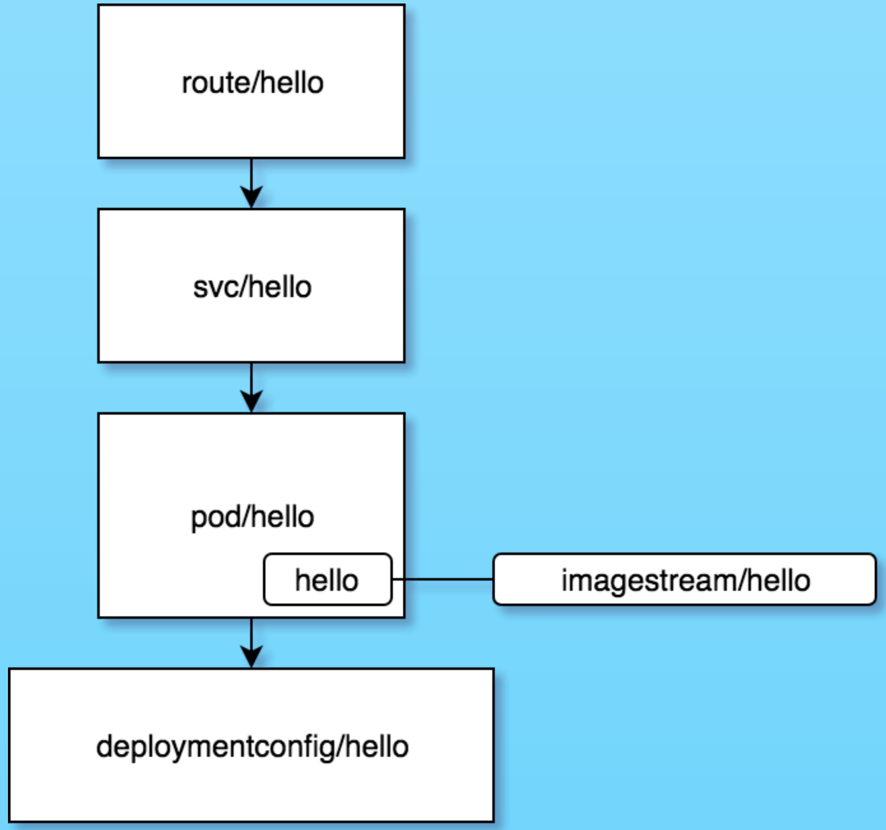
minimise divergence DEV/PRD

scale easily

Source: 12factor.net

Lab 1

A minimal deployment



Steps

```
$ oc new-project `whoami`
$ oc new-app --name hello --docker-image=openshift/hello-openshift
$ oc expose svc/hello
$ oc describe route/hello
$ curl hello-`whoami`.apps.cc-openshift.de
Hello OpenShift!
```

Service

```yaml
apiVersion: v1
kind: Service
metadata:
  labels:
    app: hello
  name: hello
spec:
  ports:
  - name: 8080-tcp
    port: 8080
    protocol: TCP
    targetPort: 8080
  selector:
    app: hello
    deploymentconfig: hello
```

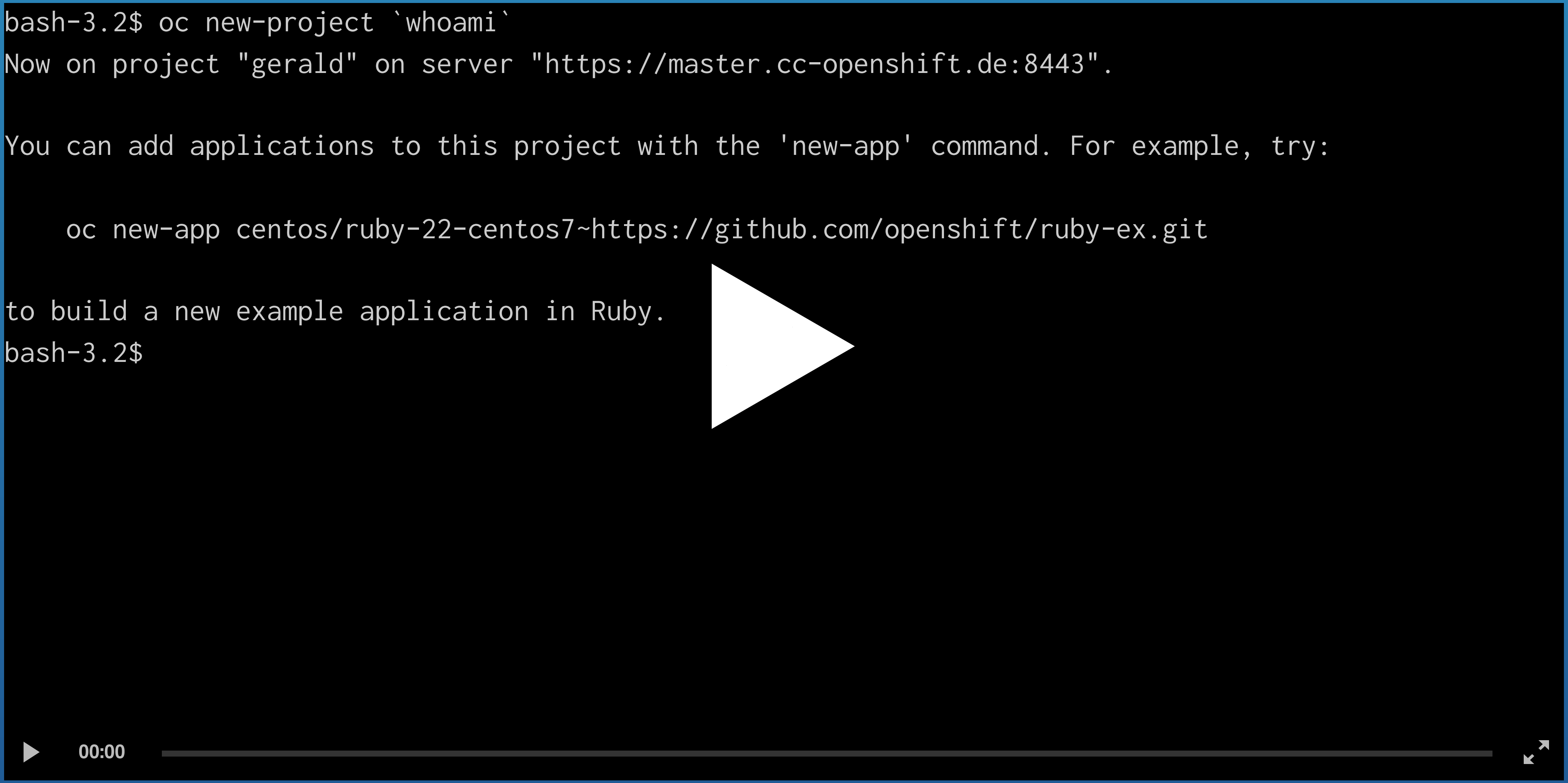
Route

```
apiVersion: v1
kind: Route
metadata:
  labels:
    app: hello
  name: hello
spec:
  host: hello-gerald.apps.cc-openshift.de
  port:
    targetPort: 8080-tcp
  to:
    kind: Service
    name: hello
    weight: 100
  wildcardPolicy: None
```

Cleanup

```
$ oc delete all -l app=hello
deploymentconfig "hello" deleted
imagestream "hello" deleted
route "hello" deleted
pod "hello-1-q42wr" deleted
service "hello" deleted
```

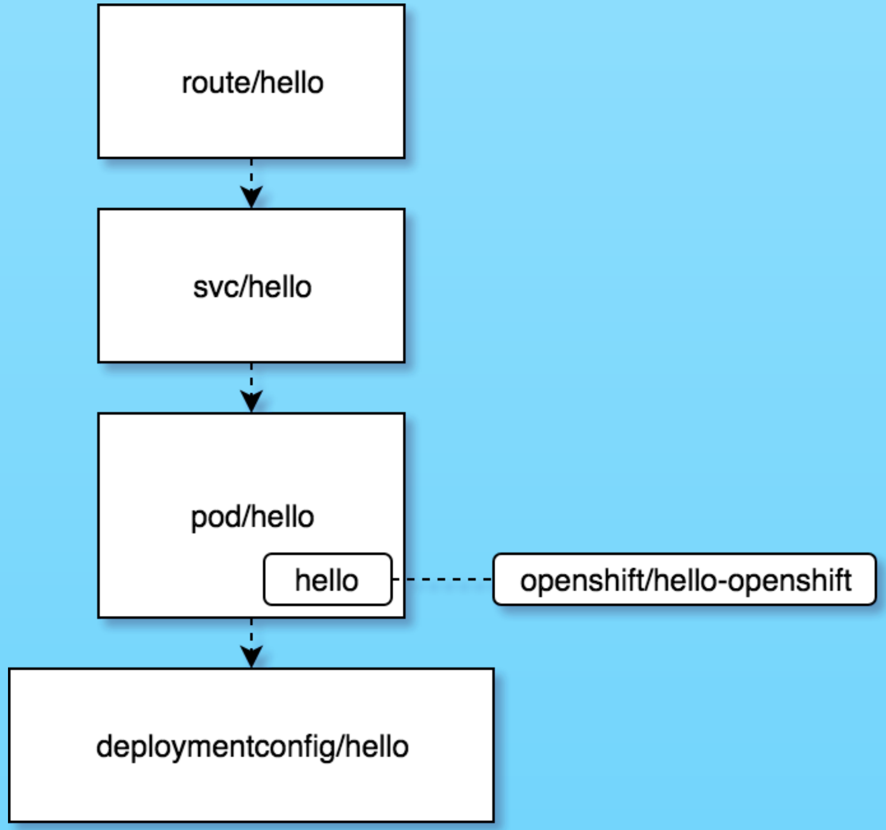
Recap



```
bash-3.2$ oc new-project `whoami`
Now on project "gerald" on server "https://master.cc-openshift.de:8443".

You can add applications to this project with the 'new-app' command. For example, try:

    oc new-app centos/ruby-22-centos7~https://github.com/openshift/ruby-ex.git

to build a new example application in Ruby.
bash-3.2$
```

Lab 2

A broken deployment



Steps

```
$ git clone \
  https://github.com/gerald1248/microservices-containers-orchestration
$ cd microservices-containers-orchestration/
$ oc new-app -f k8s/template-hello-broken.yml
$ oc expose svc/hello
$ oc get po
NAME             READY     STATUS             RESTARTS   AGE
hello-1-deploy   1/1       Running            0          13s
hello-1-r6l72    0/1       ImagePullBackOff   0          11s
```

Faulty DeploymentConfig (excerpt)

```yaml
spec:
  containers:
  - image: openshift/hullo-openshift
    imagePullPolicy: Always
    name: hello
    ports:
    - containerPort: 8080
      protocol: TCP
    - containerPort: 8888
      protocol: TCP
    resources: {}
```

DeploymentConfig fixed

```
$ oc edit dc/hello
:%s/hullo-openshift/hello-openshift/
:x
deploymentconfig "hello" edited
```

Interim result

```
$ curl hello-`whoami`.apps.cc-openshift.de
...
  <body>
    <div>
      <h1>Application is not available</h1>
      ...
    </div>
  </body>
```

Check the pod is working

```
$ oc get po
NAME            READY     STATUS    RESTARTS   AGE
hello-2-897mg   1/1       Running   0          6m
$ oc port-forward hello-2-897mg 58080:8080
Forwarding from 127.0.0.1:58080 -> 8080
```

Then, in a separate terminal:

```
$ curl localhost:58080
Hello, OpenShift!
```

The pod works as intended.

Faulty service (excerpt)

```yaml
apiVersion: v1
kind: Service
metadata:
  labels:
    app: hello
  name: hello
spec:
  ports:
  - name: 8080-tcp
    port: 8080
    protocol: TCP
    targetPort: 8080
  selector:
    app: hello-openshift
    deploymentconfig: hello
```

Service fixed

```
$ oc edit svc/hello
:%s/hello-openshift/hello/
:x
service "hello" edited
```

Result

```
$ curl hello-`whoami`.apps.cc-openshift.de
Hello OpenShift!
```

Cleanup

```
$ oc delete all -l app=hello
deploymentconfig "hello" deleted
route "hello" deleted
pod "hello-1-q42wr" deleted
service "hello" deleted
```

Finally

```
bash-3.2$ oc new-app -f k8s/template-hello-fixed.yml
```

Thank you

Clone at gerald1248/microservices-containers-orchestration

Slides courtesy of markdeck by @arnehilmann arnehilmann/markdeck