AZURE KUBERNETES SERVICE

Wojciech Barczynski - SMACC.io i Hypatos.ai
Wrzesień 2018

WOJCIECH BARCZYŃSKI

- Lead Software Engineer & System Engineer
- Interests:
  working software
- Hobby:
  teaching software engineering

SMACC

Hypatos

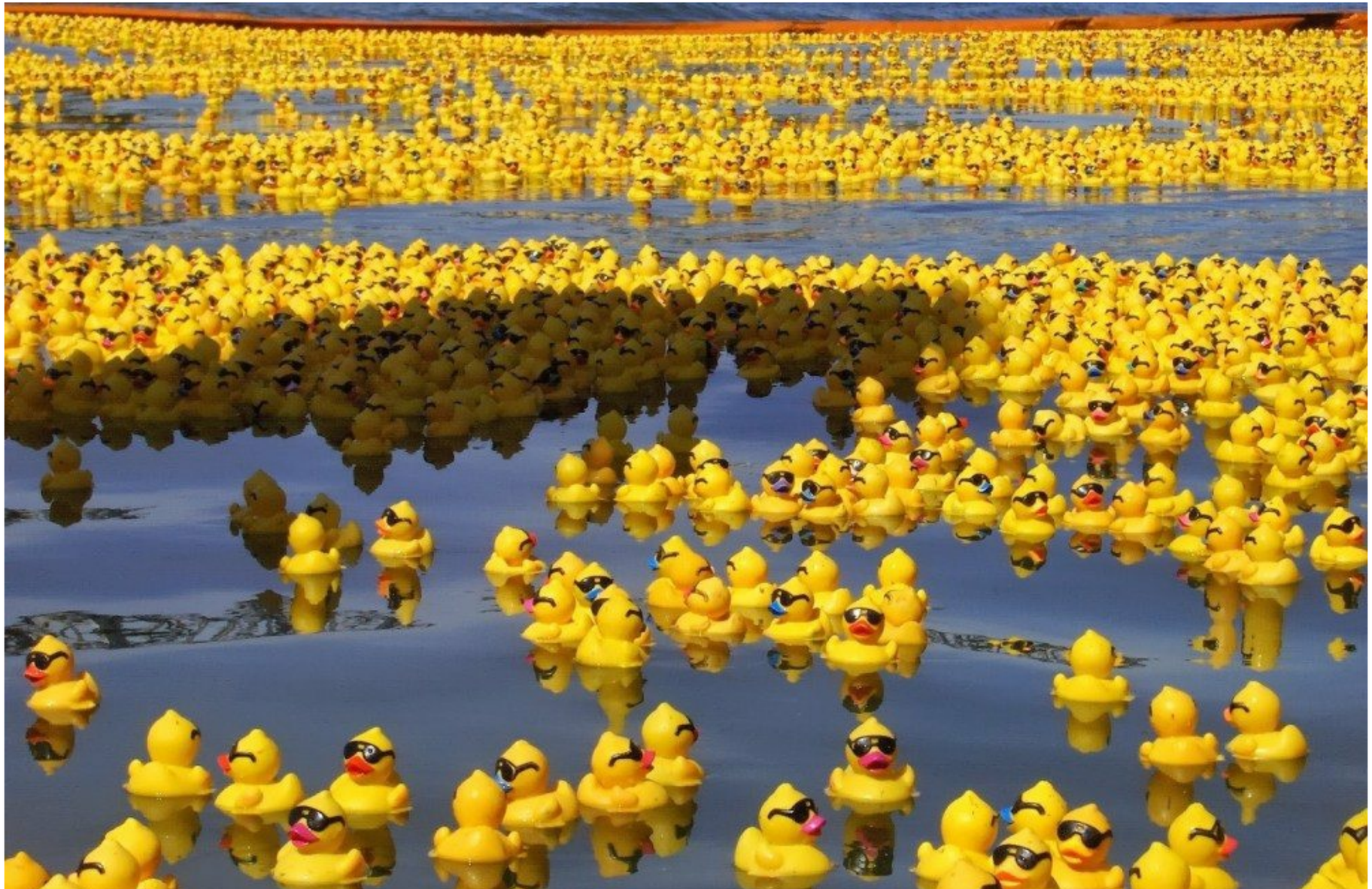
BACKGROUND

- ML FinTech ➡ microservices and k8s
- Before: a top Indonesian mobile ecommerce to k8s
- 3.5y with Openstack, 1000+ nodes, 21 data centers
- Experience with AWS and GCP
- I do not like INFRA :D

DLACZEGO?

- Admistracja jest trudna i kosztowna
- Virtualne Maszyny, ansible, salt, etc.
- Za dużo ruchomych części
- Nie kończąca się standaryzacja

MIKROSERWISY AAA!



DLACZEGO?

- Chmura jednak $$$

IMAGINE

Świat

- bez wiedzy o IaaS
- żadnego konfigurowania na nodzie
- mniej dyskusji o CI / CD ...
- Środowisko jak czarna skrzynka

KUBERNETES

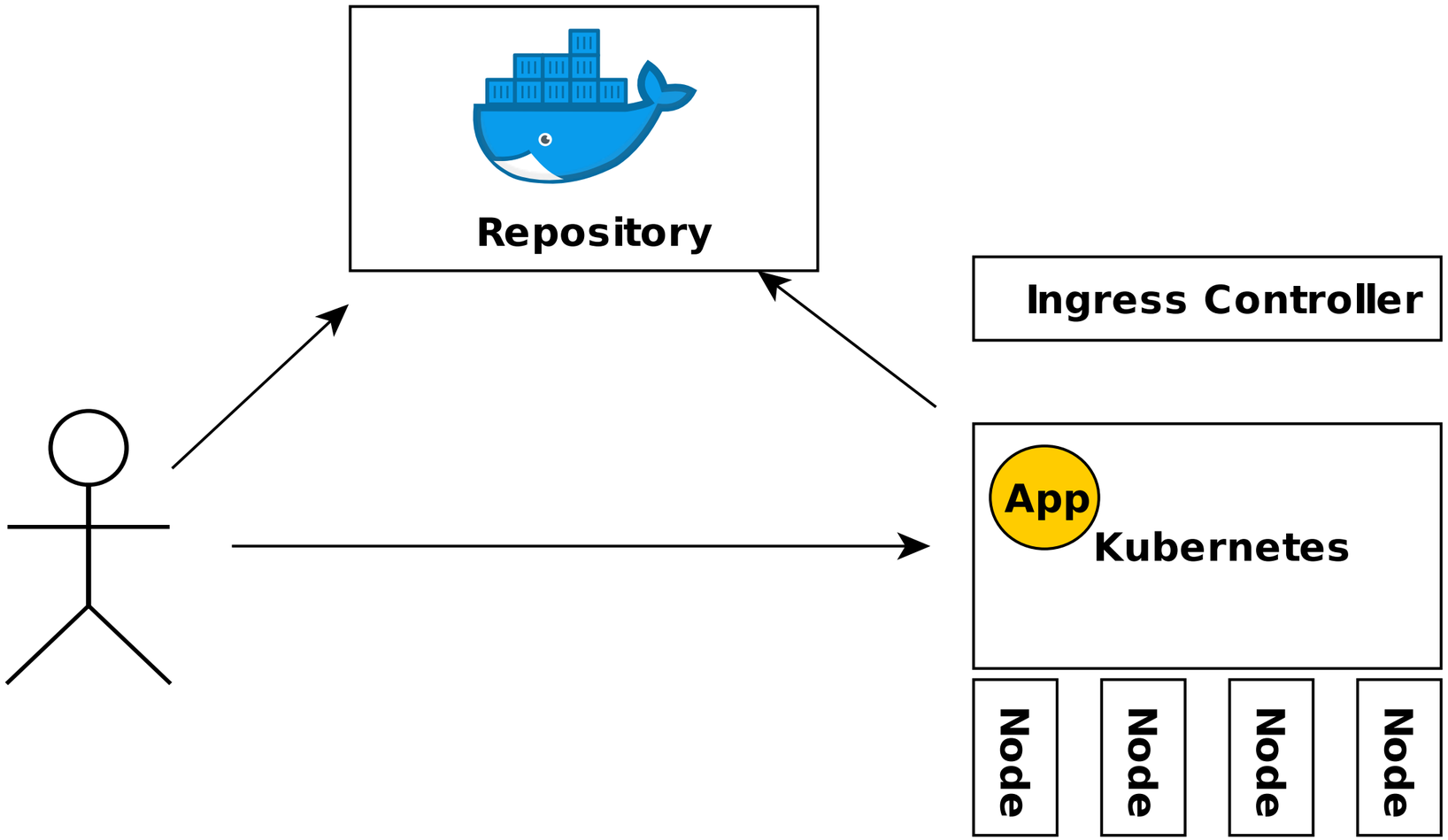
- Simple Semantic*
- Batteries for your 12factory apps
- Service discovery, meta-data support
- Independent from IaaS provider

GOALS

- Utilzie resources to early 100%
- Application and services mindset

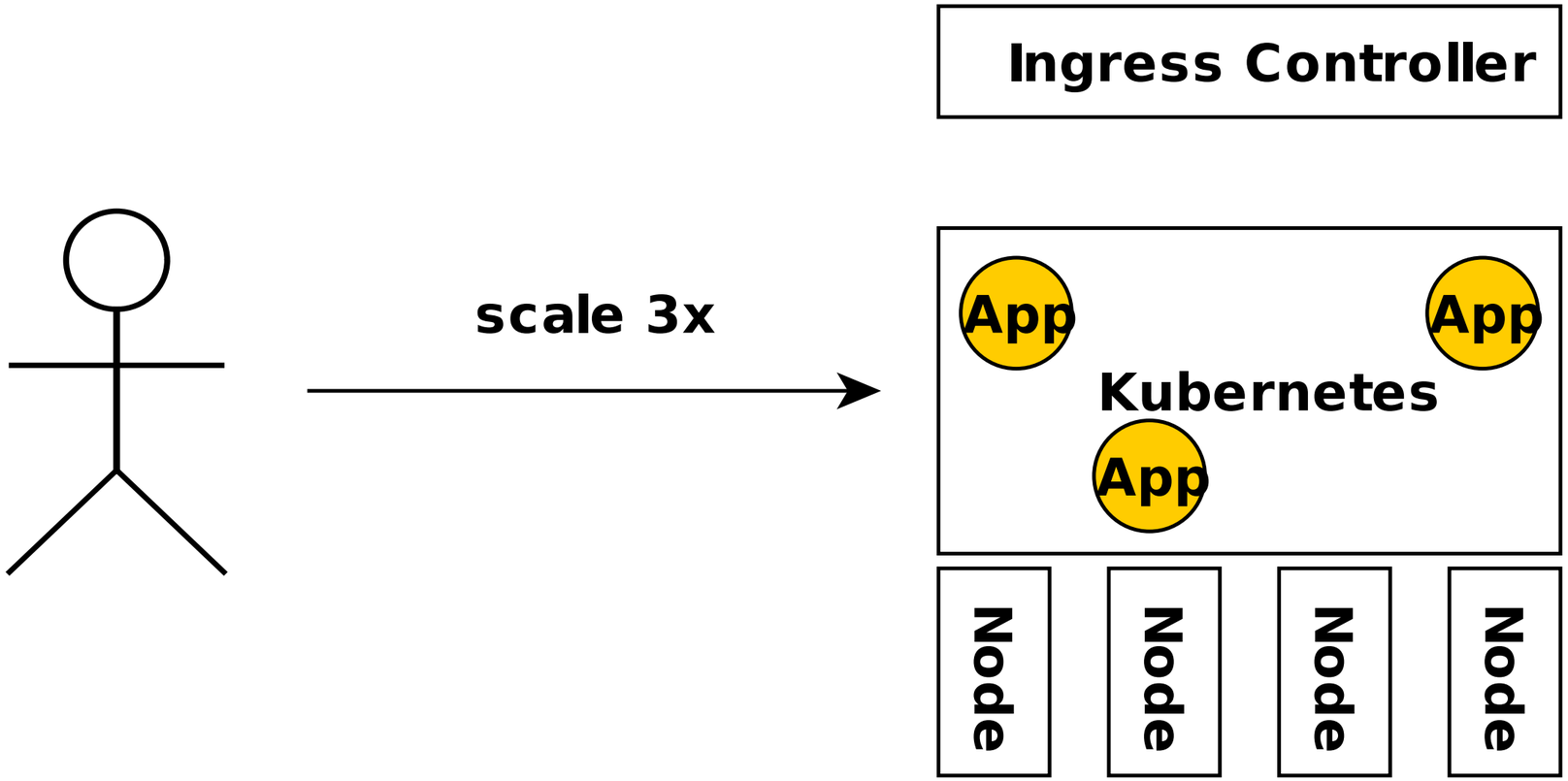
KUBERNETES

KUBERNETES



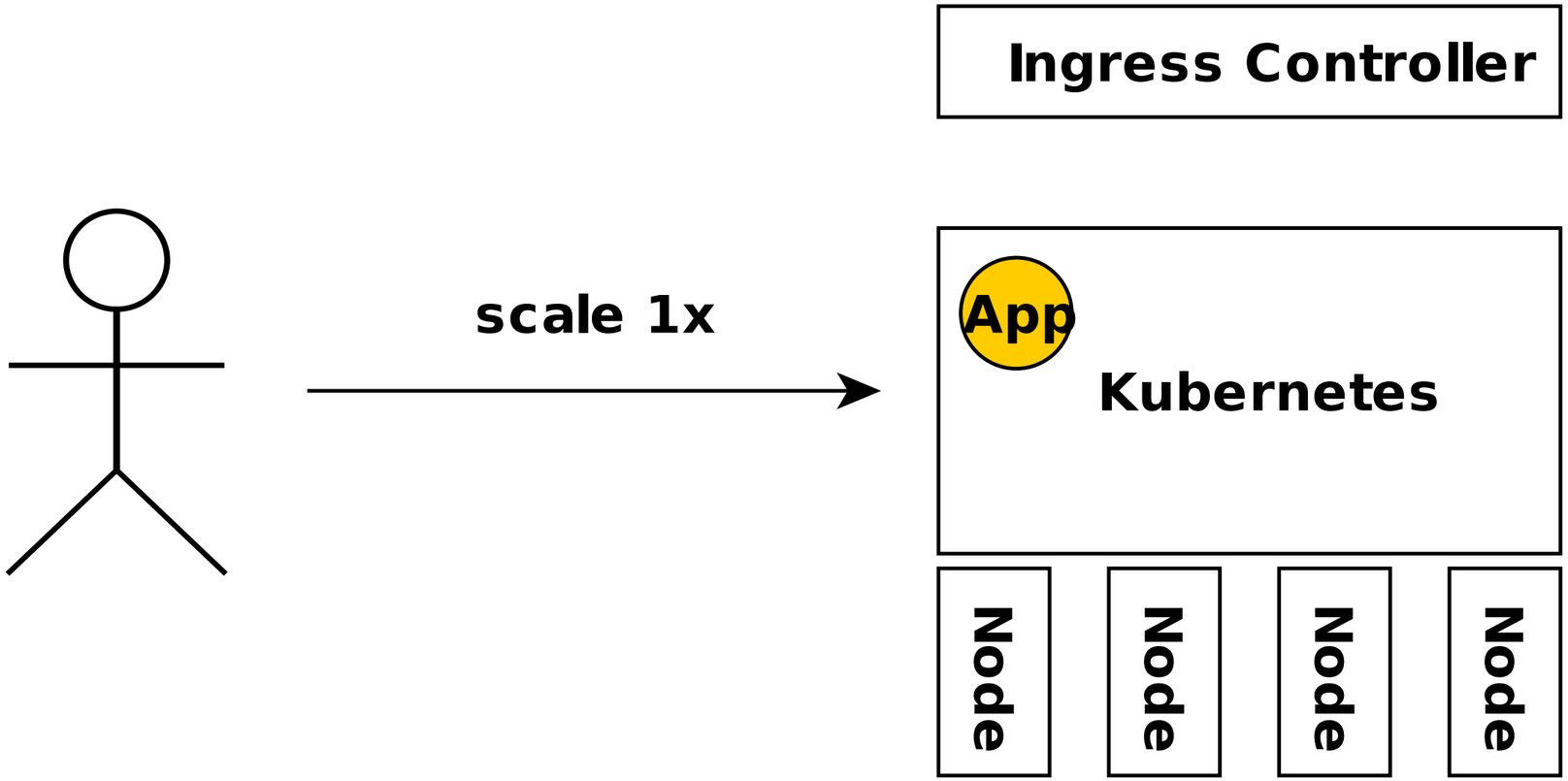
make docker_push; kubectl create -f app-srv-dpl.yaml

SCALE UP! SCALE DOWN!



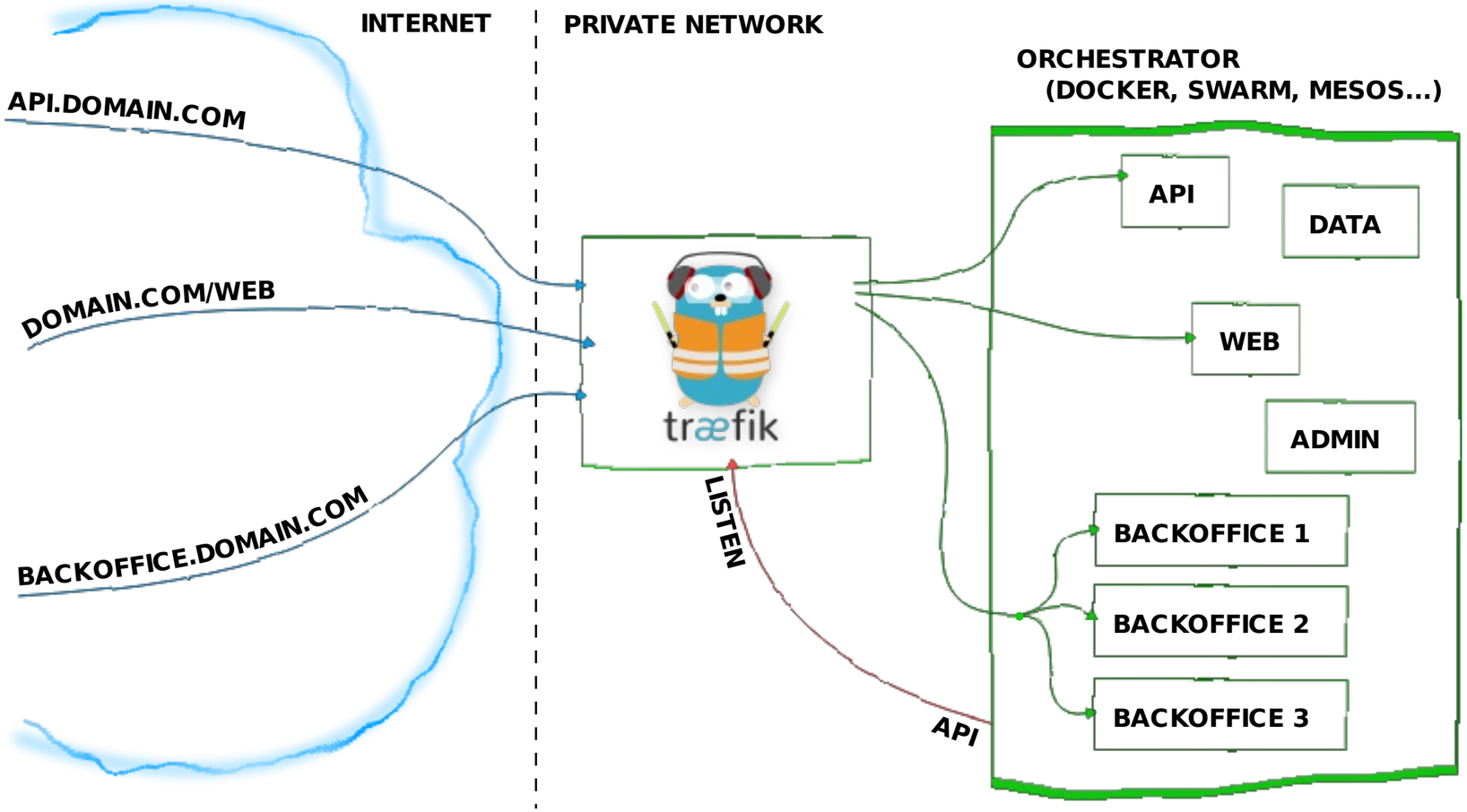
kubectl --replicas=3 -f app-srv-dpl.yaml

SCALE UP! SCALE DOWN!



kubectl --replicas=1 -f app-srv-dpl.yaml

HOW GET USER REQUESTS?

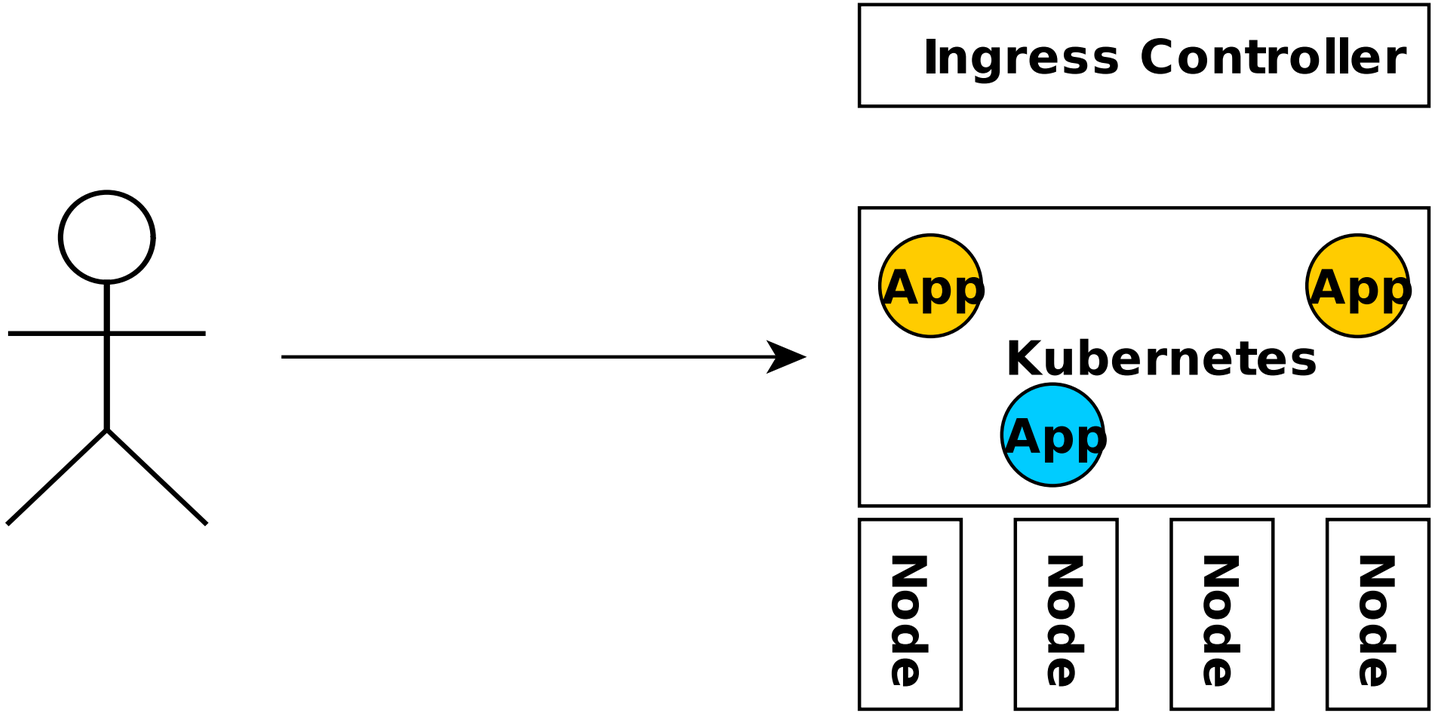


Ingress Controller

INGRESS

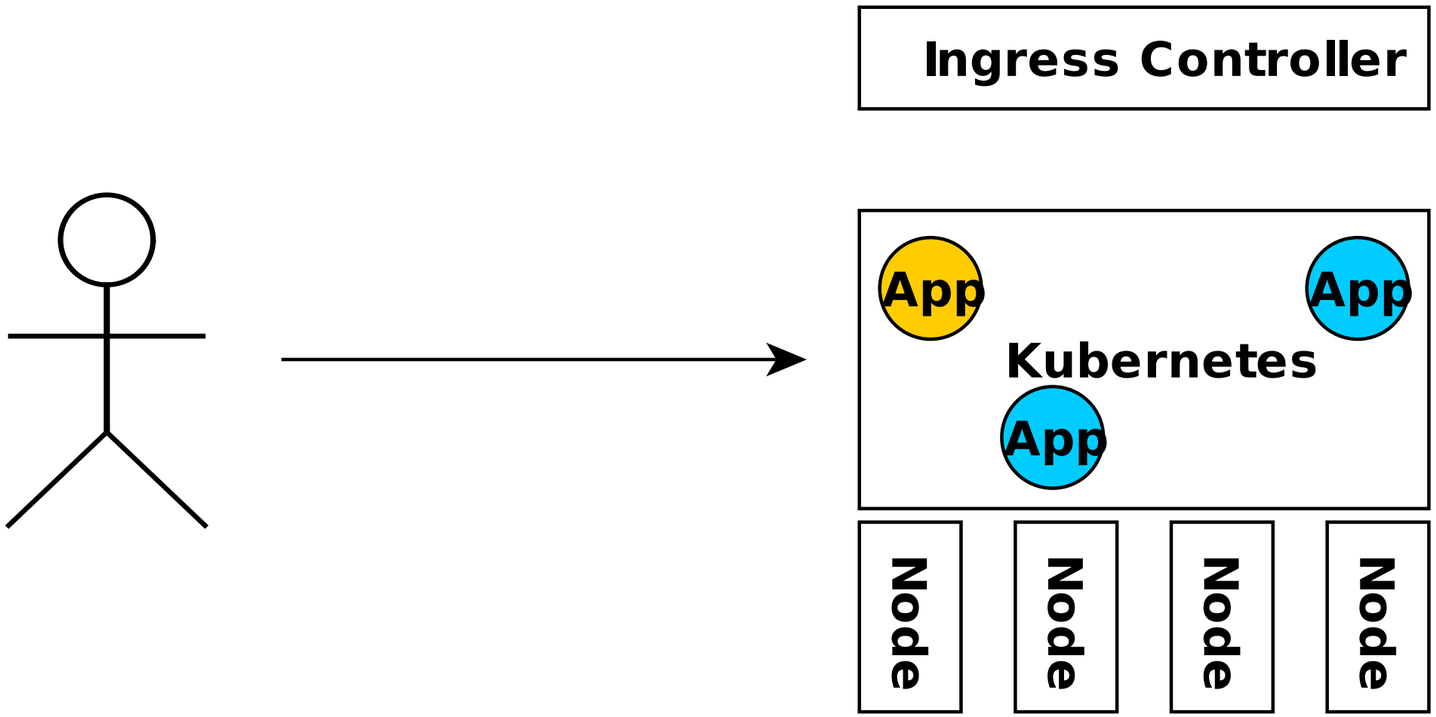
Pattern	Target App Service
api.smacc.io/v1/users	users-v1
api.smacc.io/v2/users	users-v2
smacc.io	web

ROLLING UPDATES!

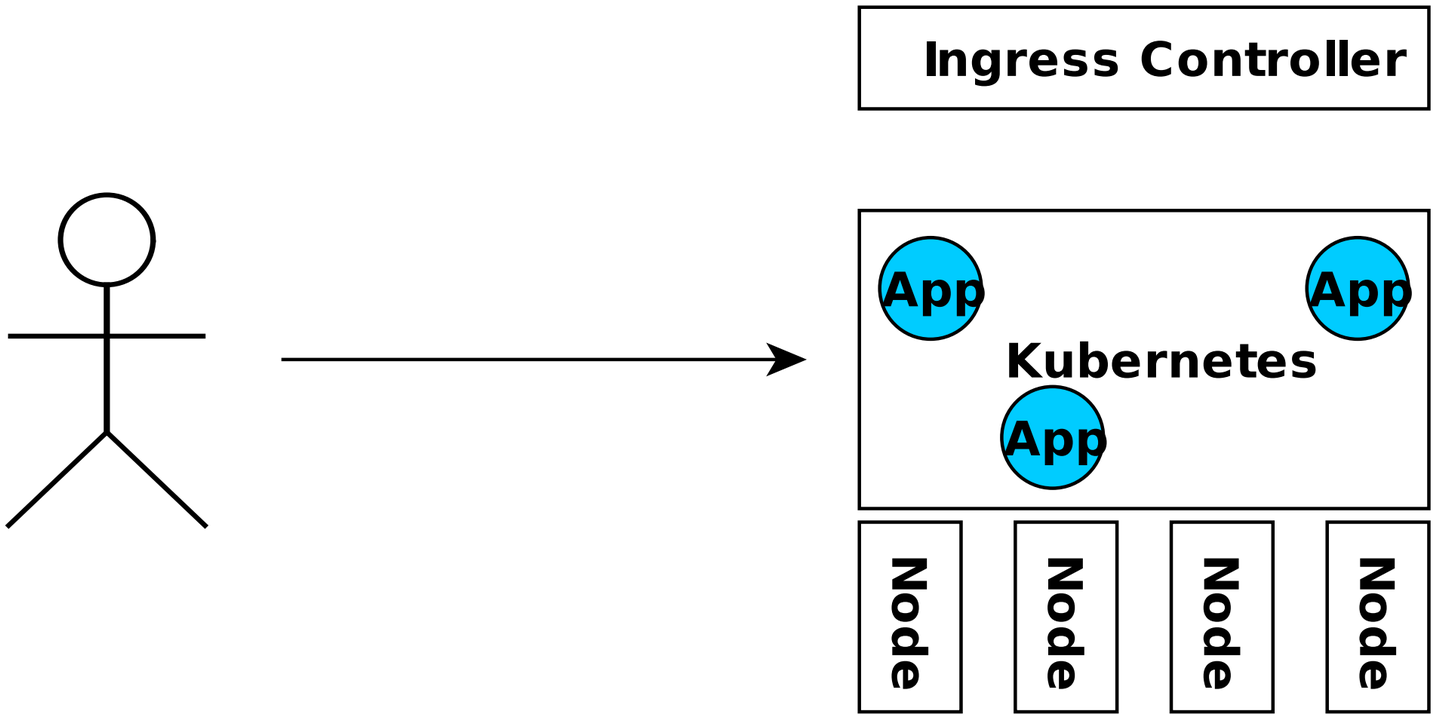


kubectl set image deployment/app app=app:v2.0.0

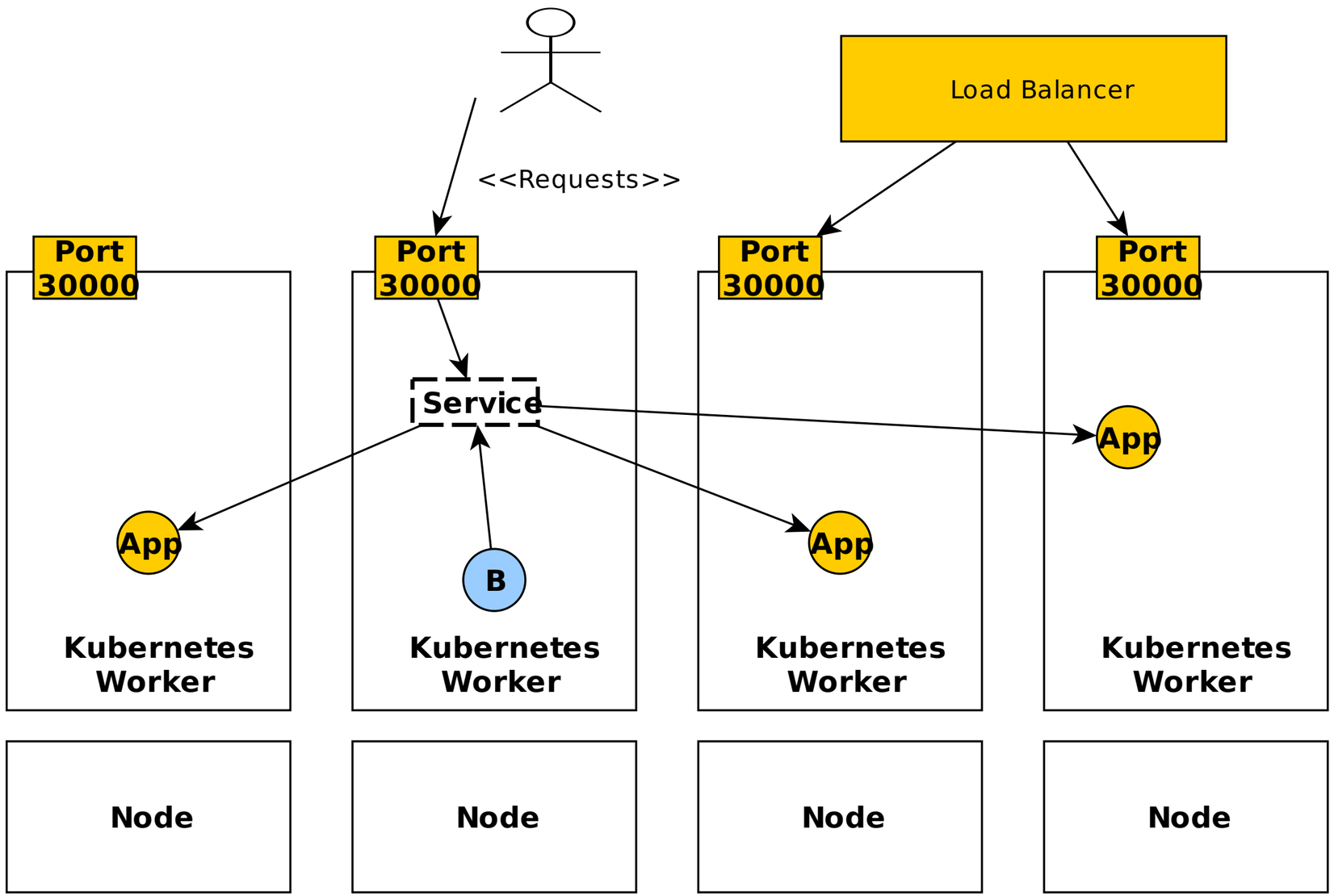
ROLLING UPDATES!



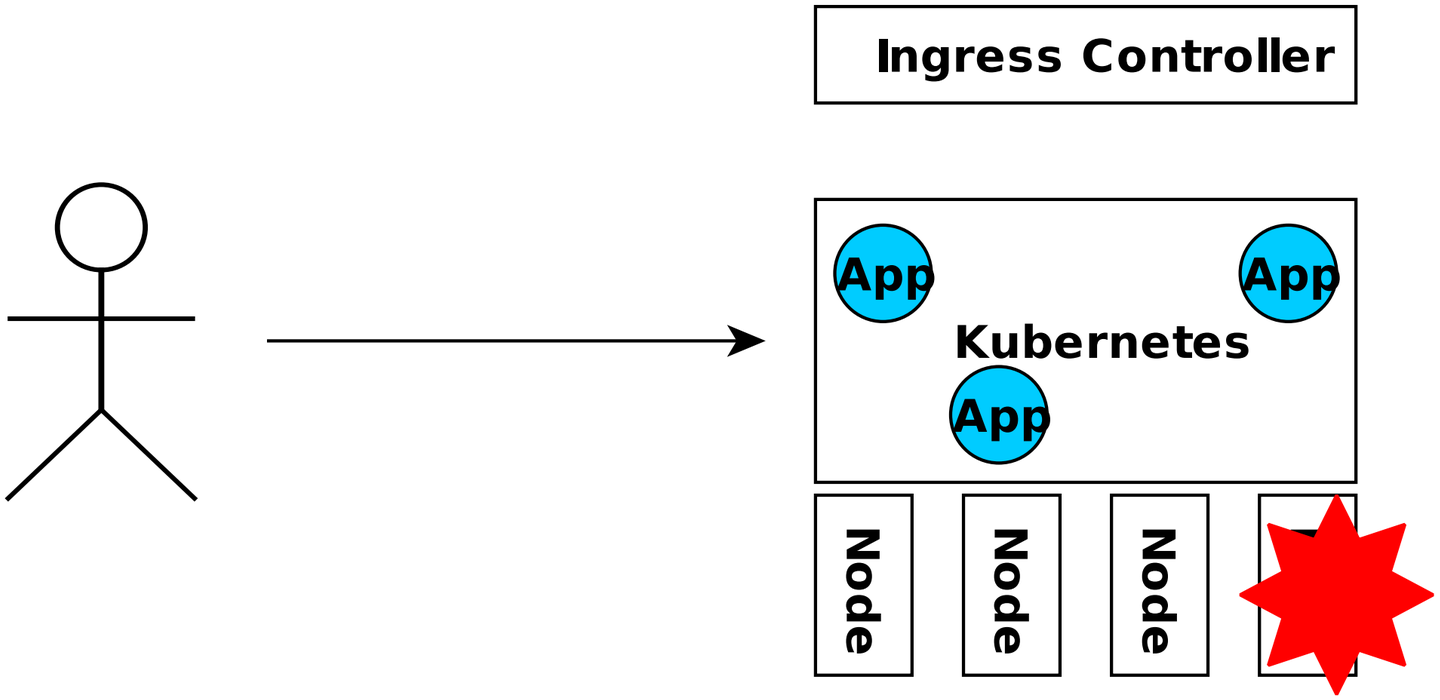
ROLLING UPDATES!



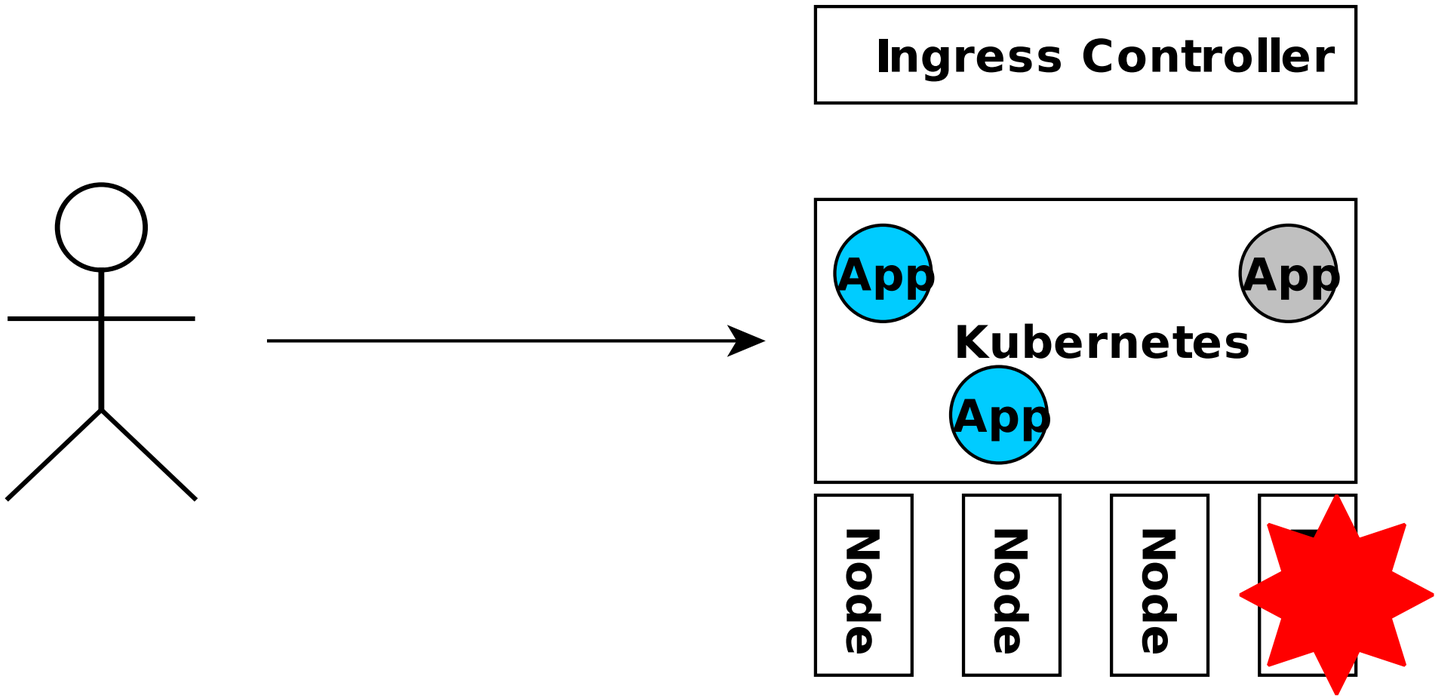
LOAD BALANCING



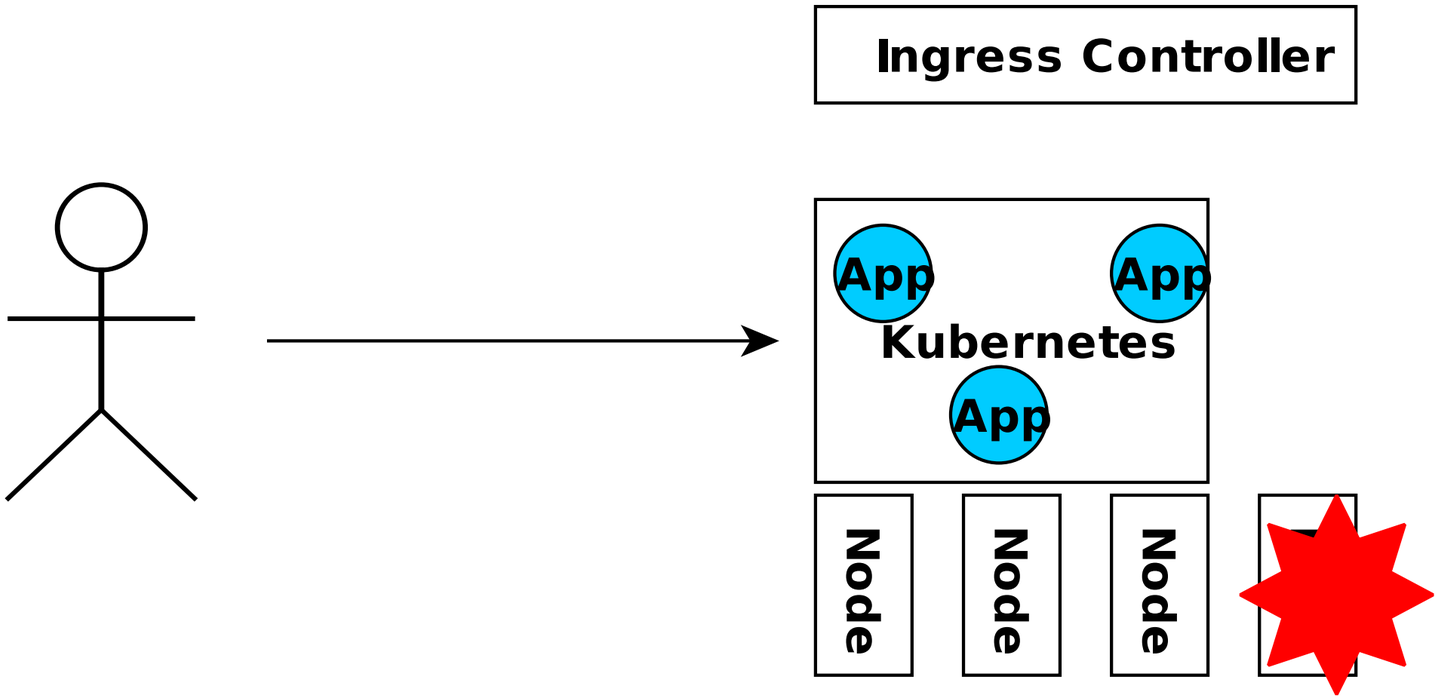
RESISTANCE!



RESISTANCE!



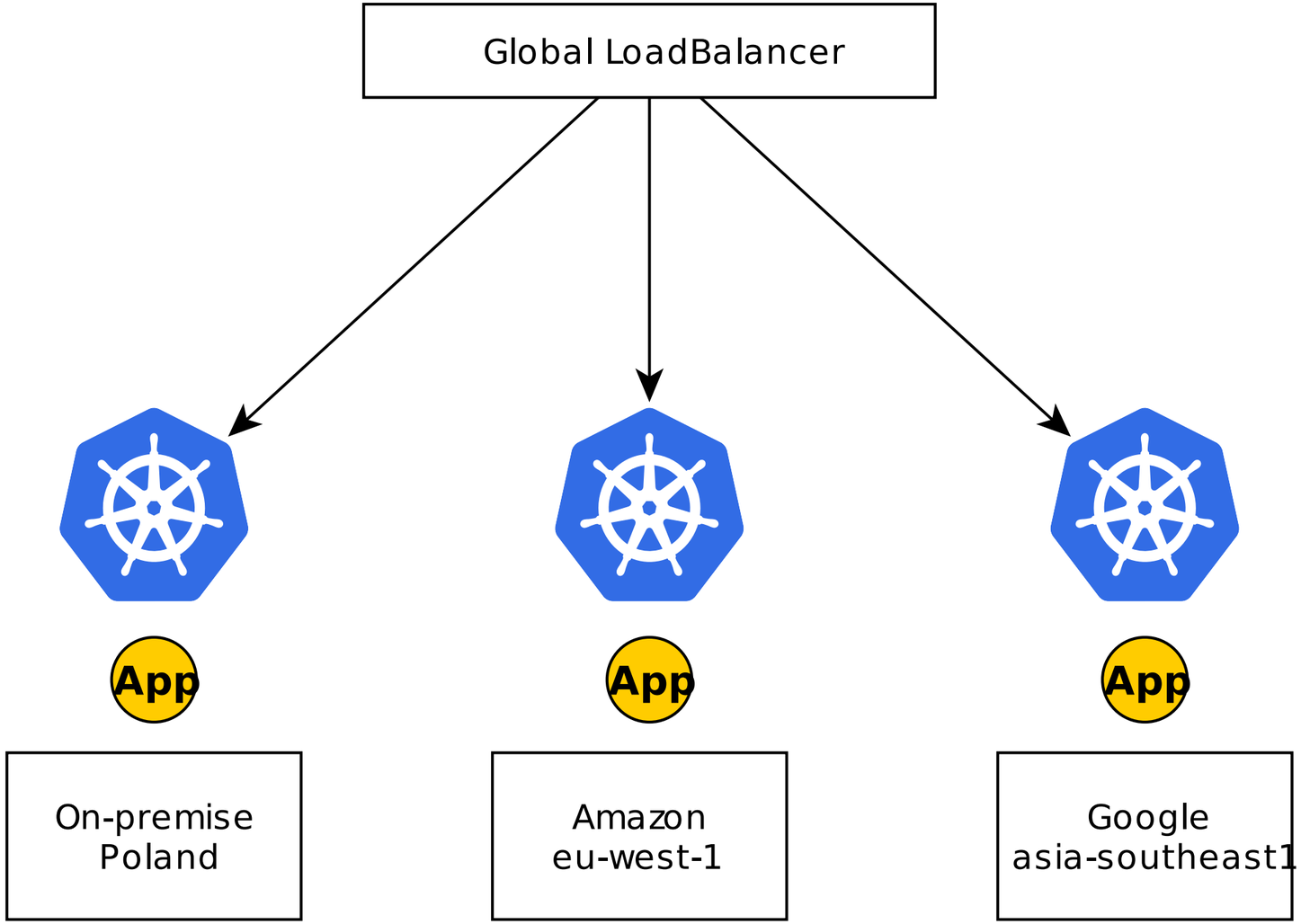
RESISTANCE!



RESISTANCE!

- When a node dies
- When other apps eats all memory
- When we drain nodes before upgrade
- You can easily scale up, create machine and join it to cluster (easier with kops or on GCE)

FEDERATION


Global LoadBalancer
App
App
App
On-premise
Poland
Amazon
eu-west-1
Google
asia-southeast1

SERVICE DISCOVERY

- names in DNS:
  curl http://users/list
- labels:
  name=value
- annotations:
  prometheus.io/scrape: "true"

SERVICE DISCOVERY

- loosely couple components
- auto-wiring with logging and monitoring

CONFIGURATION FILES

- Yaml
- easy to generate and work with

KUBERNETES @ AZURE

OPTIONS

- AKS - managed
- ACS - installation wizard
- Your own installation with Installer

AKS

- GKE for Google
- EKS or Fargate for Amazon

AKS

- Independent from IaaS
- Our OnPrem = Our OnCloud
- Consolidation of our micro-services
- Plug end play, e.g., monitoring

AKS

- You: k8s workers
- Azure: k8s masters

AKS

- You: upgrade your k8s
- Azure: update your kube-system pods, k8s config, and nodes

AZURE UPDATES

Bumpy road ahead

- Kube-Systen pods
- Kubernetes configuration - ☠
- System: on the node restart applied
- System: Memory-preserving updates - ☠

AZURE UPDATES - NODES

```
NAME                    VERSION   OS-IMAGE          KERNEL-VERSION     CON
aks-nodepool1-27173880-0   v1.10.3   Ubuntu 16.04.4 LTS   4.15.0-1018-a
aks-nodepool1-27173880-1   v1.10.3   Ubuntu 16.04.4 LTS   4.15.0-1018-a
aks-nodepool1-27173880-2   v1.10.3   Ubuntu 16.04.4 LTS   4.15.0-1018-a
aks-nodepool1-27173880-3   v1.10.3   Ubuntu 16.04.4 LTS   4.15.0-1018-a
aks-nodepool1-27173880-5   v1.10.3   Ubuntu 16.04.4 LTS   4.15.0-1019-a
```

kubectl get nodes -o wide

AZURE UPDATES - K8S

- Scalling down / up your cluster applies the newest k8s config changes

AZURE K8S INTEGRATION

- Load Balancers
- Persistence Volumes
- Graphic Cards Support
- Authentication with oauth
- Monitoring?

LIMITS

- No node-pool support
- RBAC?
- No limited centralized logging
- Federation support

PAIN POINTS

- Memory preserving updates
- AKS team changes configuration

ANNOYING

- Slow deletes
- Slow attaching and detaching volumes
- You are not able to delete a pod without --force

LOVE

- Openess on github: AKS issues

CREATE

```
az aks create --name portal-production \
--resource-group MYCOMPANY \
--node-vm-size 'Standard_D4_v2' \
--node-count 4 \
--generate-ssh-keys
```

CREATE

GO TO PORTAL

ssh to nodes

READY TO GO!

- az aks get-credentials -g MYCOMP -n portal-prod
- kubectl get pods

UPDATE

```
Name    MasterVersion    NodePoolVersion    Upgrades
-------  ---------------  -----------------  ----------------------------------------
default          1.10.3          1.10.3           1.10.5, 1.10.6, 1.10.7, 1.11.1, 1.1
```

az aks get-upgrades -g MYCOMP --name portal-dev -o table

UPDATE

```
az aks upgrade --name portal-dev\
--resource-group MYCOMP \
--output table \
--kubernetes-version 1.10.3
```

Do not rush!

UPDATE

```
az aks upgrade --name portal-dev\
--resource-group MYCOMP \
--output table \
--kubernetes-version 1.10.3
```

Do not rush!

AKS @ SMACC

SETUP AZURE

- az aks CLI for setting k8s
- Terraform for everything else

TF also sets our AWS

KUBERNETES

- Pure, generated, kubernetes config
- 2x kubernetes operators

CONTINUOUS DEPLOYMENT

- Github
- TravisCI
- hub.docker.com
- Kubernetes

In spirit similar to the Kelsey Hightower approach

BACKUP

- Ark
- CronJobs for some components

ENVIORNEMNTS

Env	Number of Nodes
Prod	7
Staging	5
Dev	4
Tools	1

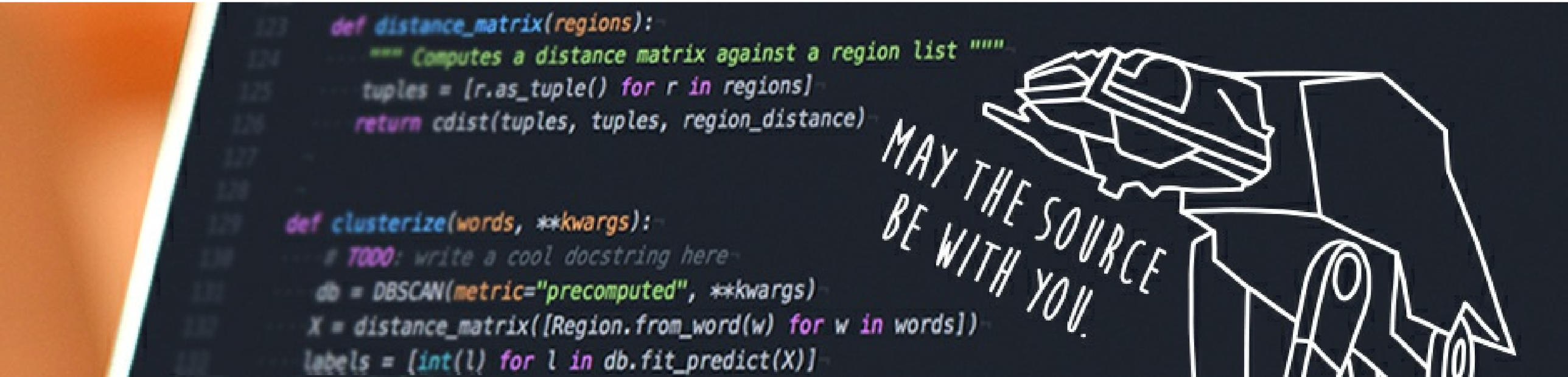
We also have short-living ML clusters

SUMMARY

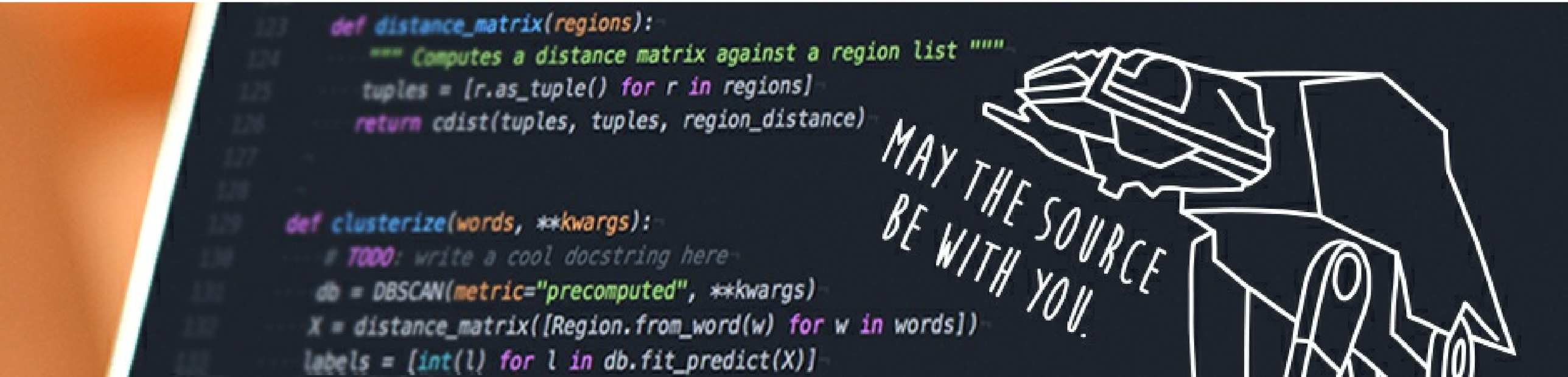
- Kubernetes not a silver bullet, but damn close
- AKS the easiest way to start with k8s in Azure
- Still bumpy period - see github issues

DZIĘKUJĘ. PYTANIA?

ps. We are hiring.



BACKUP SLIDES



0.1 ➡ 1.0

1. CLEAN UP

- Single script for repo - Makefile [1]
- Resurrect the README

[1] With zsh or bash auto-completion plugin in your terminal.

2. GET BACK ALL THE KNOWLEDGE

- Puppet, Chef, ... ➡ Dockerfile
- Check the instances ➡ Dockerfile, README.rst
- Nagios, ... ➡ README.rst, checks/

3. INTRODUCE RUN_LOCAL

- make run_local
- A nice section on how to run in README.rst
- Use: docker-compose

The most crucial point.

4. GET TO KUBERNETES

- make kube_create_config
- make kube_apply
- Generate the yaml files if your envs differ

5. CONTINUOUS DEPLOYMENT

Travis:

- test code, build docker, push to docker repo
- only run the rolling update:
  kubectl set image deployment/api-status nginx=nginx:1.9.1
- did not create any kubernetes artificats [*]

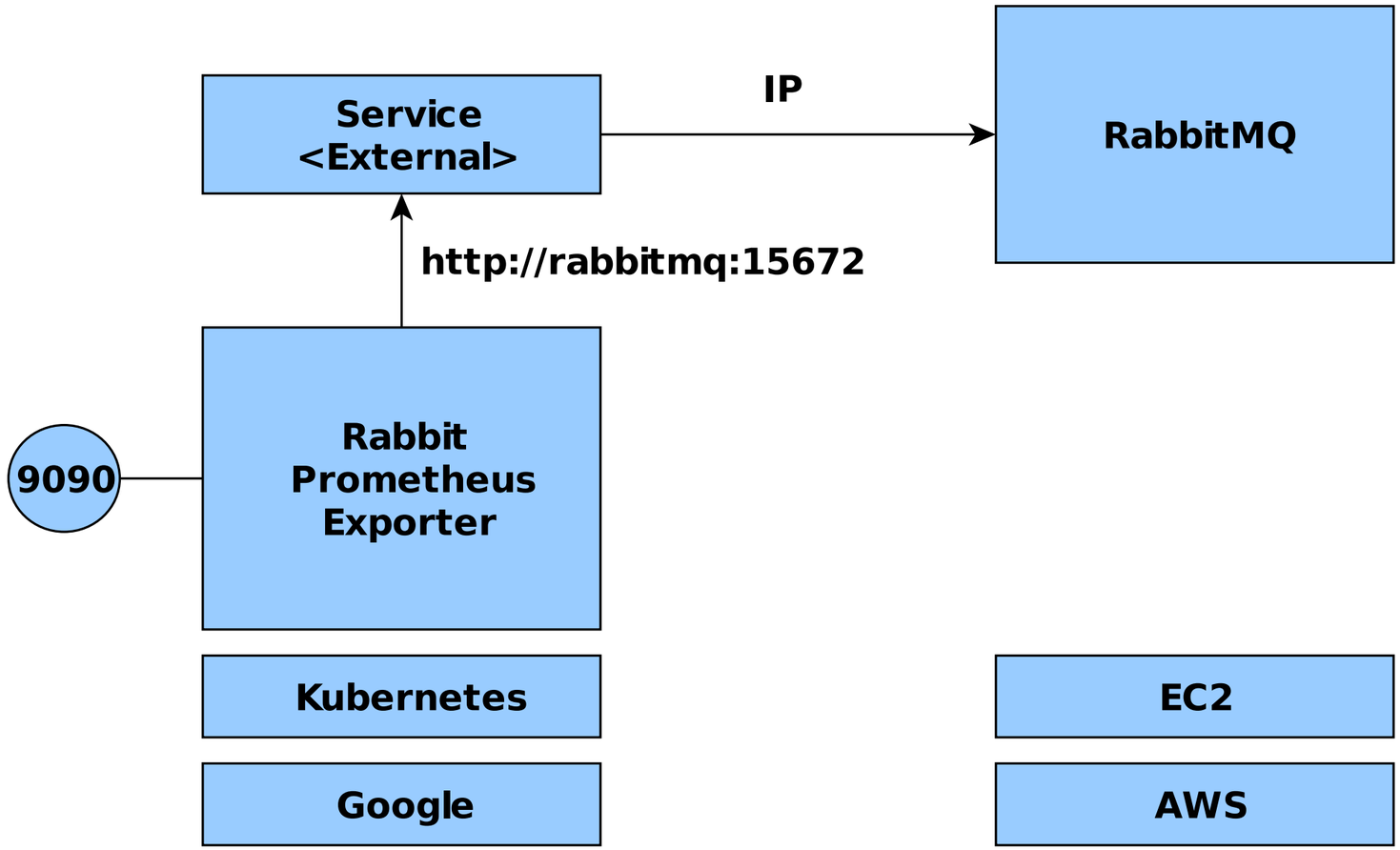
[*]

6. KEEP IT RUNNING

Bridge the new with old:

- Use external services in Kubernetes
- Add Kubernetes services to your Service Discovery [1]

[1] I evaluated feeding K8S events to HashiCorp consul


Service
<External>
IP
RabbitMQ
http://rabbitmq:15672
9090
Rabbit
Prometheus
Exporter
Kubernetes
EC2
Google
AWS

7. INTRODUCE SMOKE-TEST

```
TARGET_URL=127.0.0 make smoke_test
TARGET_URL=api.example.com/users make smoke_test
```

8. GOT FIRST MICRO-SERVICES

To offload the biggest components:

- Keep the light on of the old components
- New functionality delegated to micro-services

9. GET PERFORMANCE TESTING

- introduce wrk for evaluating performance (more like a check for dockers)
- load test the real system

SERVICE SELF-CONSCIOUSNESS

Add to old services:

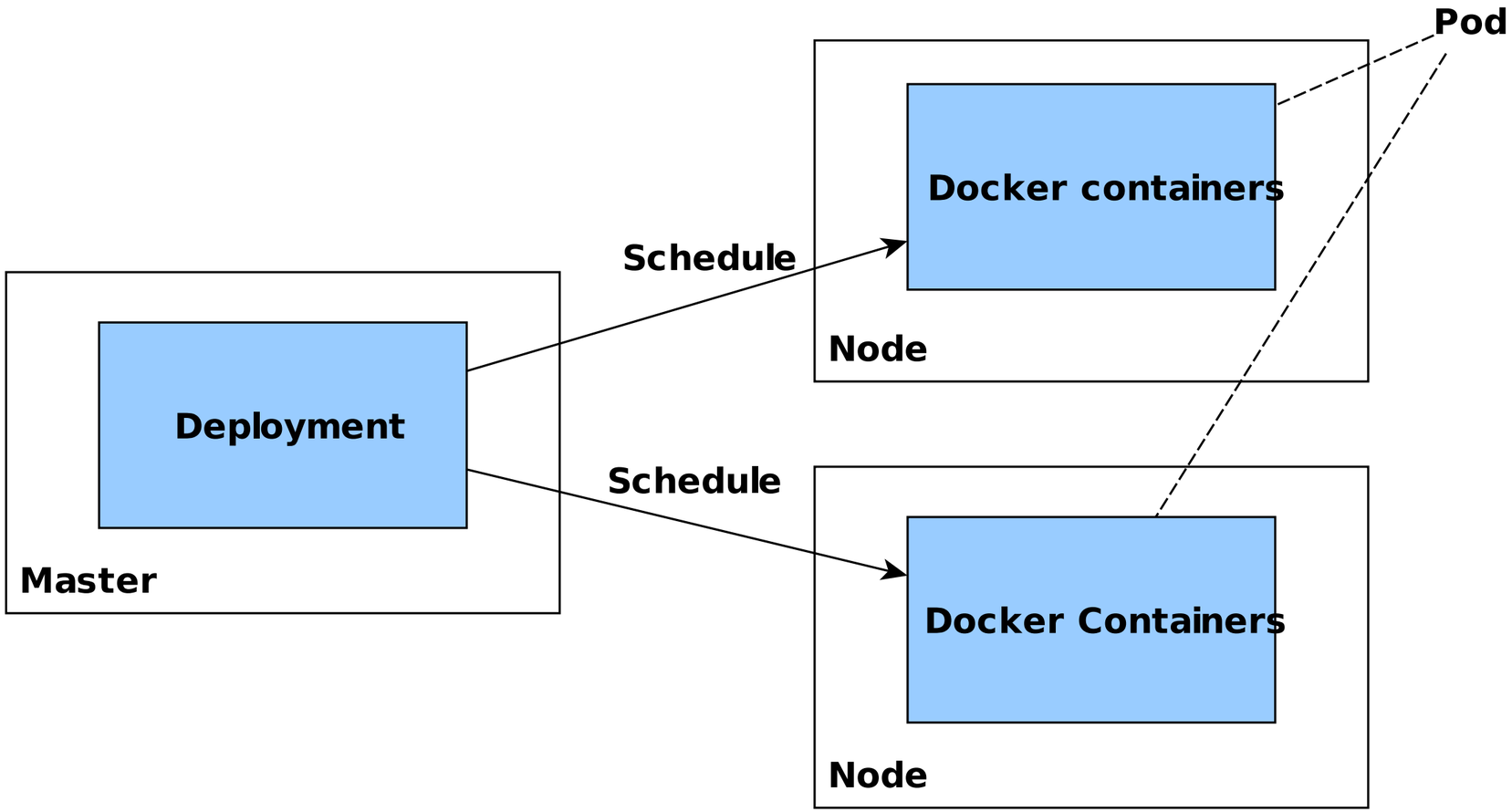
1. *metrics/*
2. *health/*
3. *info/*
4. *alertrules/ - PoC*

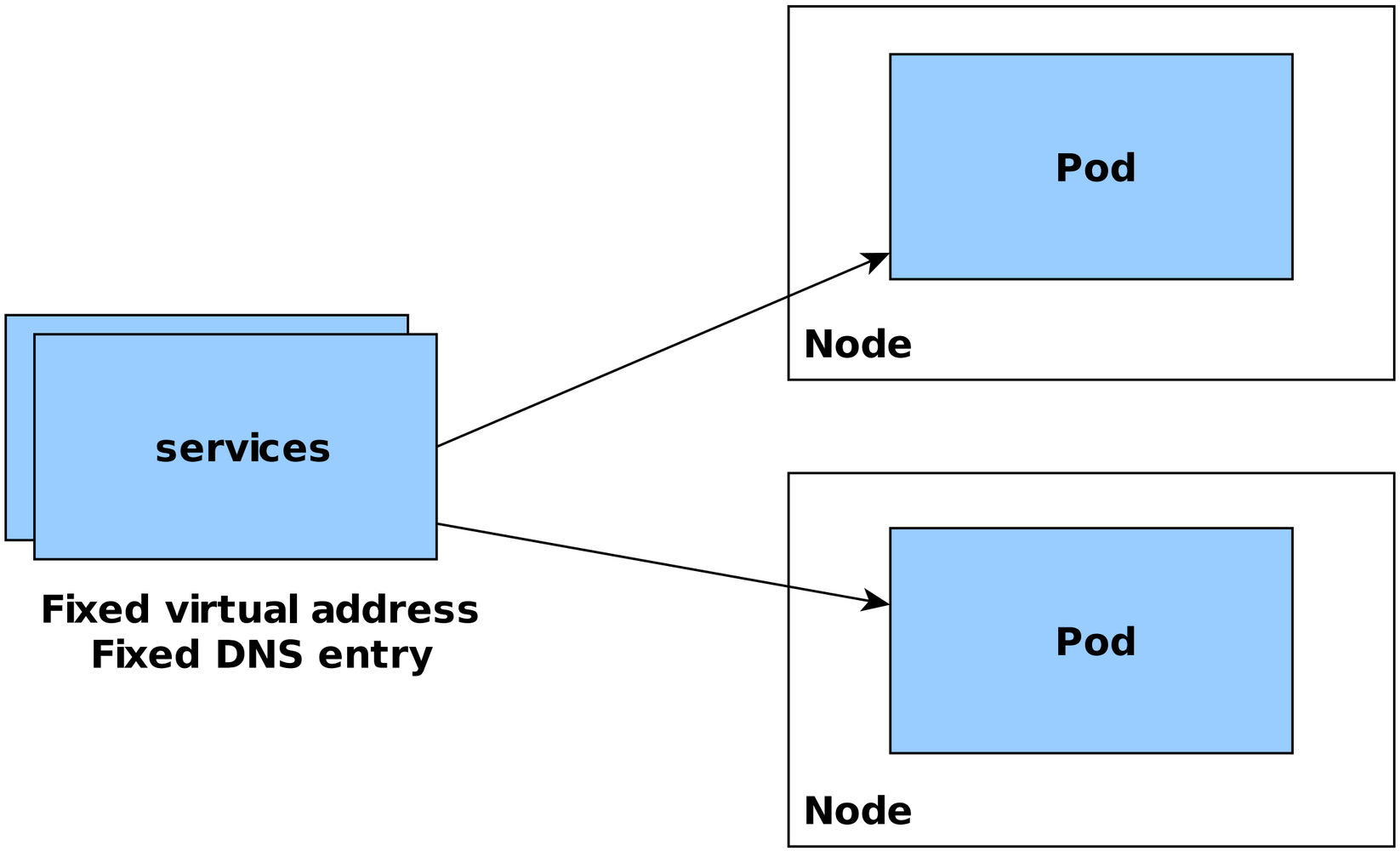
CHANGE THE WORK ORGANIZATION

- From Scrum
- To Kanban

For the next talk

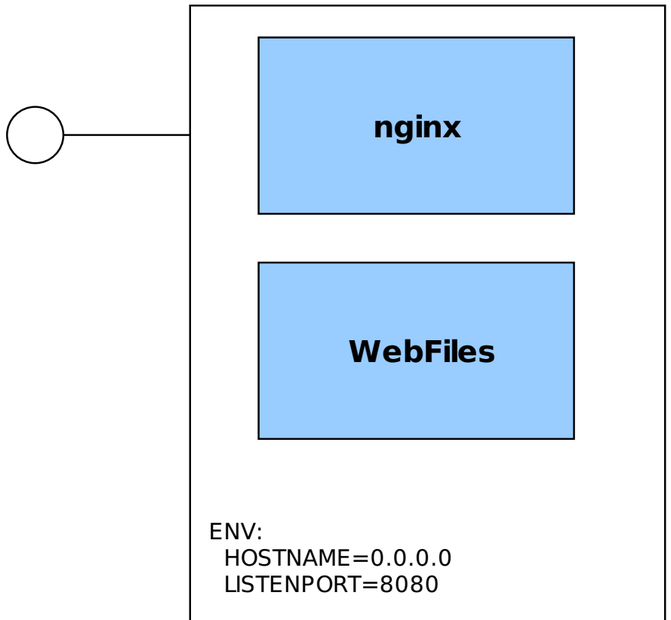
KUBERNETES CONCEPTS


Pod
Docker containers
Schedule
Node
Deployment
Schedule
Master
Docker Containers
Node

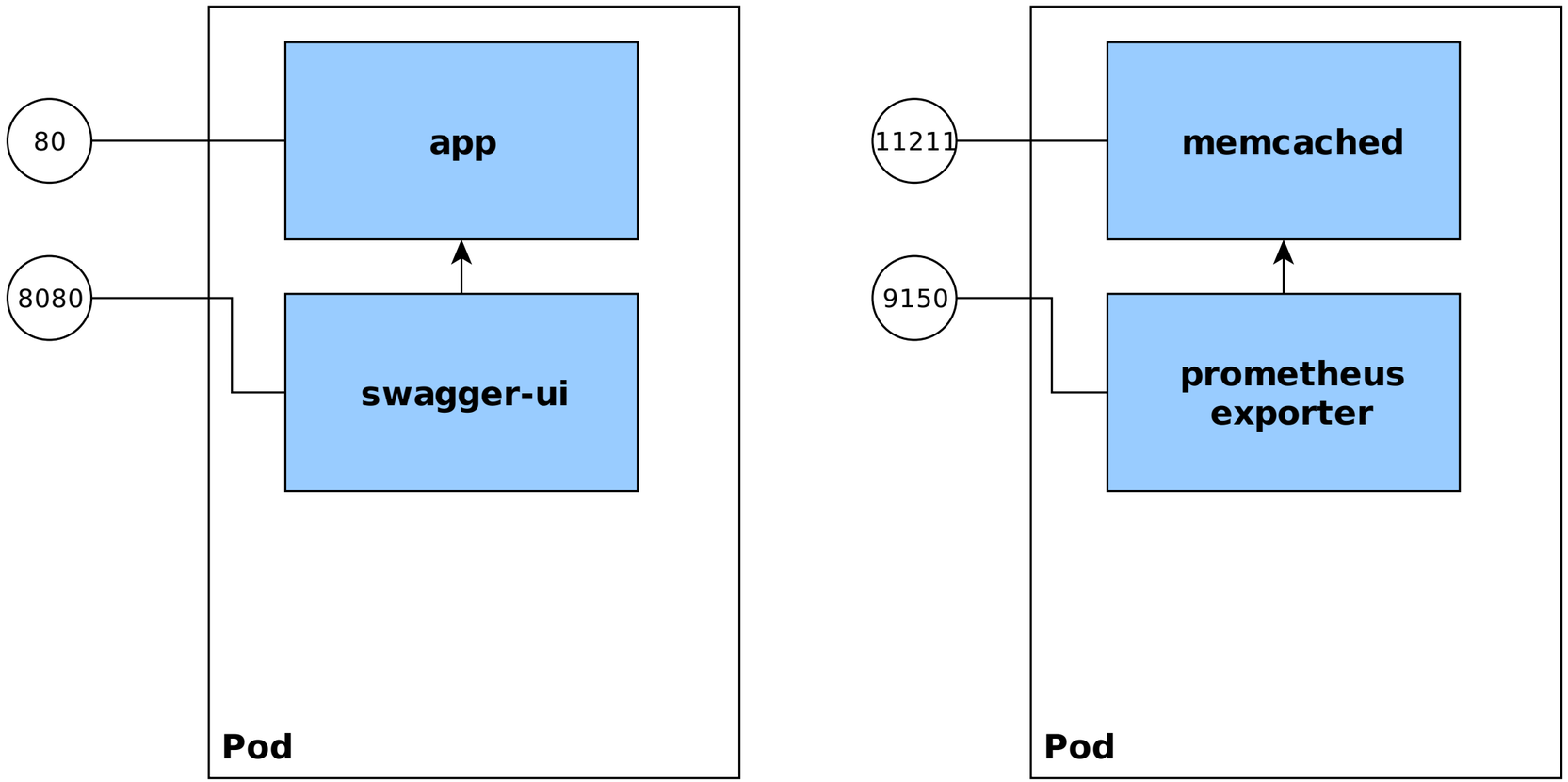

services
Fixed virtual address
Fixed DNS entry
Pod
Node
Pod
Node

PODS

- See each other on localhost
- Live and die together
- Can expose multiple ports



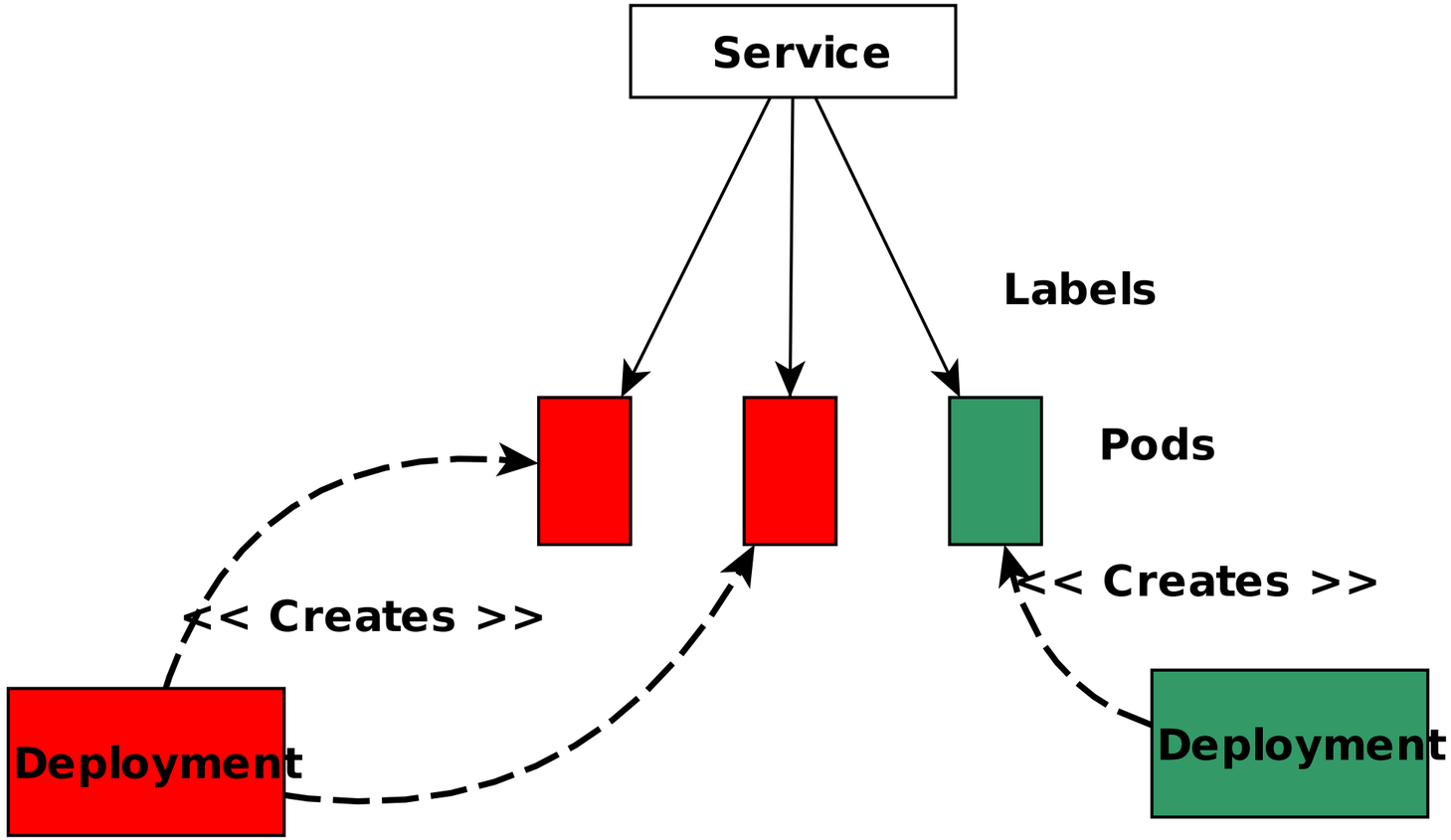
SIDE-CARS


80
app
8080
swagger-ui
Pod
11211
memcached
9150
prometheus exporter
Pod

BASIC CONCEPTS

Name	Purpose	
Service	Interface	Entry point (Service Name)
Deployment	Factory	How many pods, which pods
Pod	Implementation	1+ docker running

ROLLING RELEASE WITH DEPLOYMENTS



Also possible