OPENSHIFT AT ISU

A New[ish] Platform for Application Delivery

WHAT IS OPENSHIFT?

- Built on Kubernetes
- Container Orchestration

WHAT ARE CONTAINERS?

- Isolated environments
- Based on immutable images
- Cattle not pets

WHY CONTAINER ORCHESTRATION?

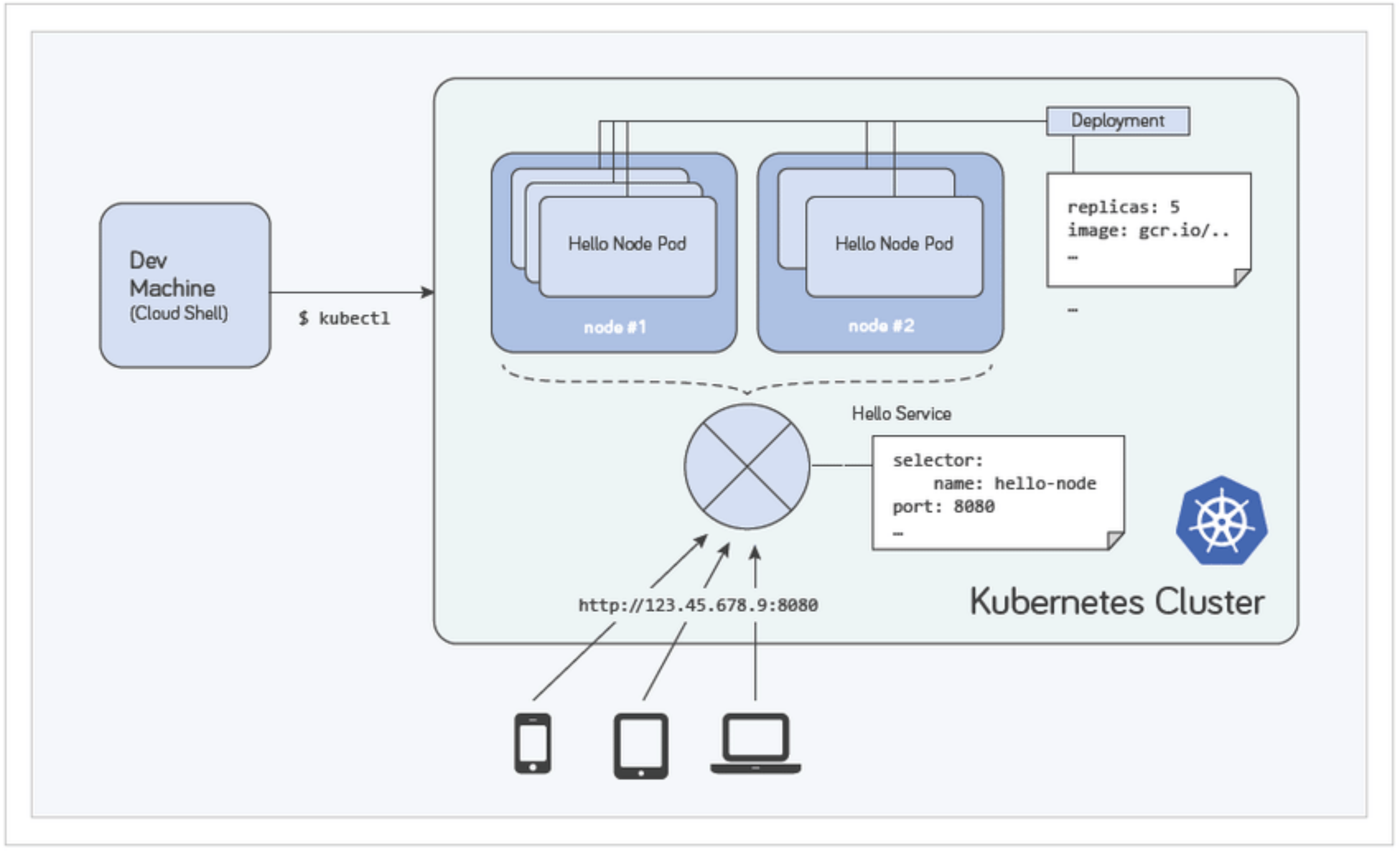
- Self-Service Provisioning
- Infrastructure as Code
- Safe Rolling Deployment
- Language-Agnostic
- Our team once had a dozen servers -> now none
- Hybrid-Cloud Ready
- Centralized Logging / Metrics / Alerting

NOUNS AHOY

- Deployments and Pods
- Services and Routes
- ConfigMaps and Secrets
- Builds and Images

DEPLOYMENTS

Own/manage other related nouns like pods etc.

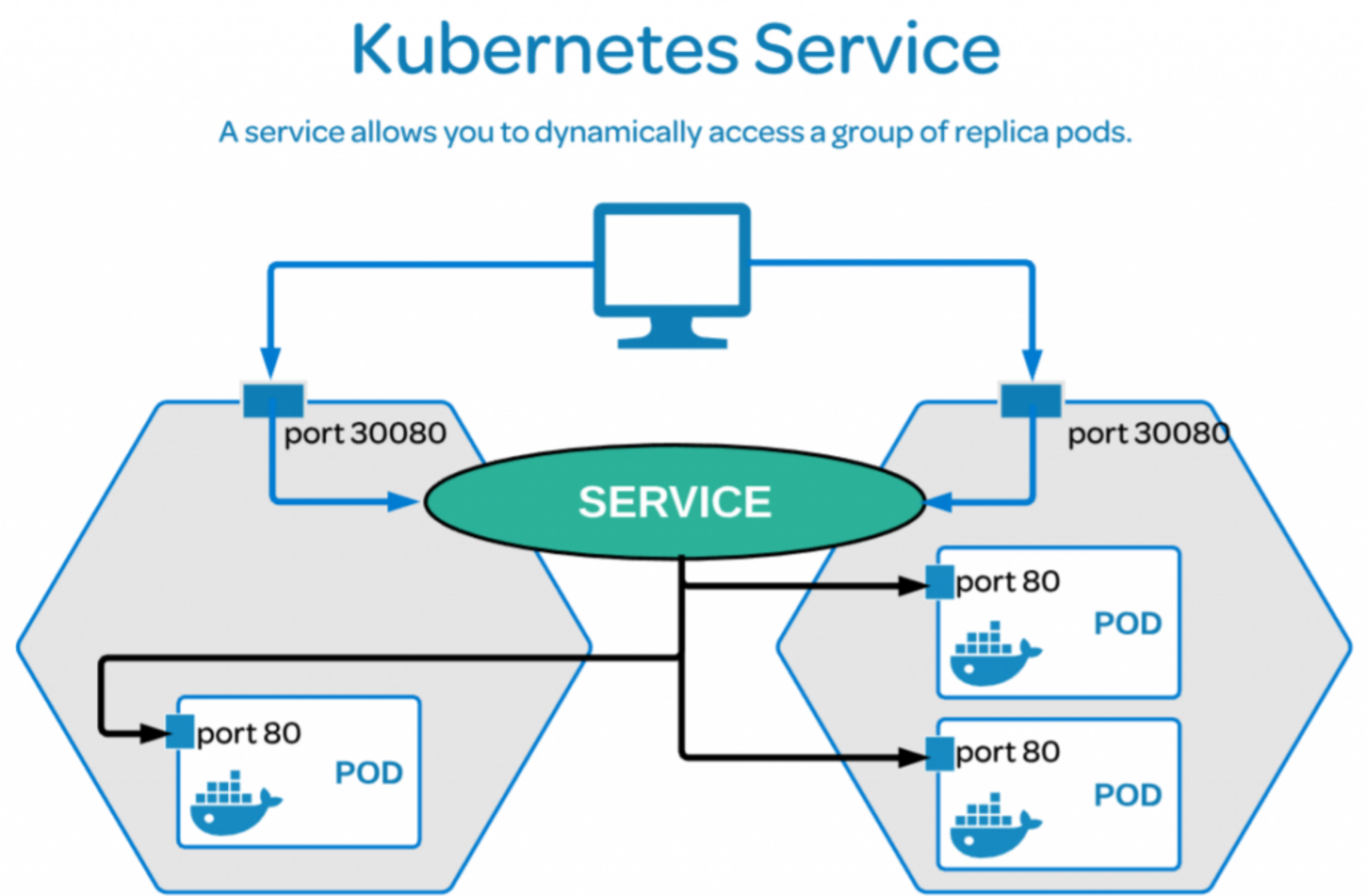


```
replicas: 5
image: gcr.io/..
…
```

…

```
selector:
    name: hello-node
port: 8080
…
```

SERVICES AND ROUTES



Kubernetes Service

A service allows you to dynamically access a group of replica pods.

CONFIGMAPS AND SECRETS

- Managed separately from code
- Can provide config files, env vars, etc.
- Secrets are just more polite ConfigMaps

BUILDS AND IMAGES

- Containers come from (Docker) Images
- Images come from Builds
- Images can also come from 3rd party repos

SOURCE-TO-IMAGE (S2I)

- Insert your code into a base image environment that's ready to run that code
- RH Delivered S2I images for
- ■ .NET
- ■ Java
- ■ Go
- ■ Node
- ■ Perl
- ■ PHP
- ■ Python
- ■ Ruby

S2I BUILD PROCESS

```
- Copies source code to defined location in the builder image
- Runs an assemble script in the builder image
- Sets up the run script
- Outputs a final container image
```

SOURCE-TO-IMAGE (S2I)

- Our apps do not directly generate a container image
- No Dockerfile required

SOURCE-TO-IMAGE (S2I)

Apps can be deployed through Web Console or with

```
oc apply
```

CONTINUOUS INTEGRATION

- GitLab
- GitLab CI
- GitLab Runner (Operator)

.GITLAB-CI.YML

```
stages:
  - build
  - test
  - deploy

build-job:
  stage: build
  script:
    - echo "Hello, $GITLAB_USER_LOGIN!"

test-job1:
  stage: test
  script:
    - echo "This job does something"

test-job2:
```

CI DEPLOYMENT PIPELINE

- Git Commit Hook (.gitlab-ci.yml)
- OpenShift Runner initiates a Build
- Build results in container image
- Deployment, Config, Volumes, Services and Routes created (kustomize)
- Deployment spawns Pod based on our built image
- Application is available

KUSTOMIZE

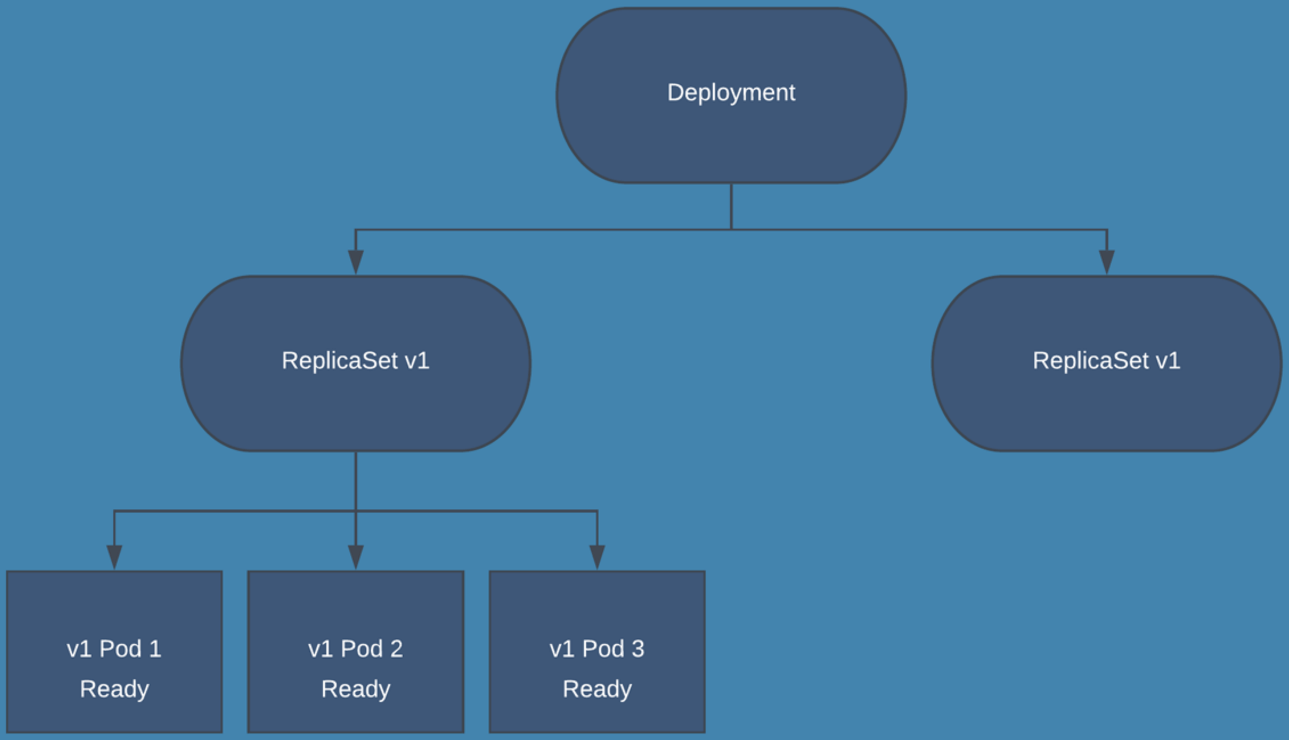
- Template-free way to customize K8S YAML resources
- Declarative approach to customization.
- Supports the idea of bases and overlays.
- Emits a YAML stream that can be piped into `oc apply`

DEPLOYMENT PIPELINE

- Nothing is modified directly in OpenShift
- Everything is configured and fully stored in GitLab

EASY ROLLING DEPLOYMENTS

1. Build new image from source code
2. New pods are started from the new image
3. Traffic is still routed to the old pods until new reports ready
4. Traffic is redirected to new pods
5. Old pods are destroyed


Deployment
ReplicaSet v1
ReplicaSet v1
v1 Pod 1
Ready
v1 Pod 2
Ready
v1 Pod 3
Ready

DEMO TIME!

Instantiating a new app from a git repo

OpenShift Console Link

OPERATOR HUB

Candy store of deployables (demo)

Operator Hub Link

CENTRALIZED LOGGING

Kibana

FIN.

Questions?