

# Data Communications & Networks

## Session 3 – Introduction to K8s

Dr. Jean-Claude Franchitti



## Separation of Responsibilities



## Monolithic vs Microservice

### Monolithic Service:

- Built as a single unit
- Difficult to understand
  - Can become too big to manage (no developer can understand the entire code)
- Fixes are difficult
  - (now you need to re-deploy the whole app)

### Microservice:

- Built as multiple units, separated by logical functionalities
- Easier to understand and manage
- Fixes are easier and only requires deployment of specific microservices

## Containerization

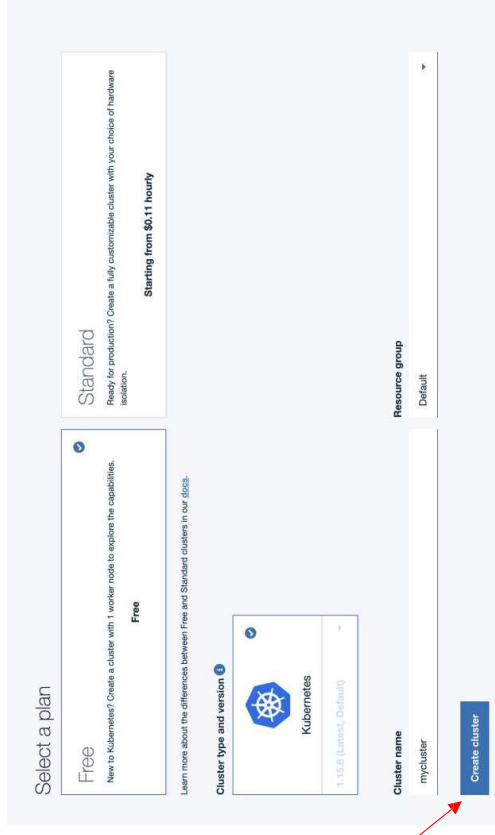
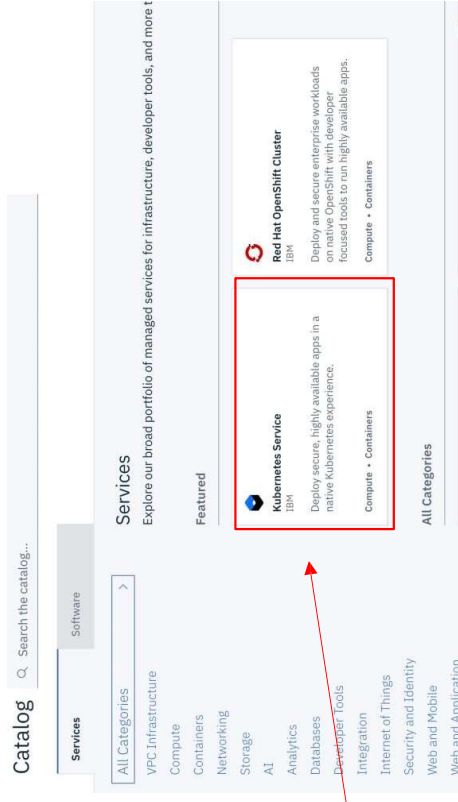
- Traditionally applications are deployed in VMs
  - Difficult to reproduce (configuration change from VM to VM)
  - Any change in VM results in affecting your application
- Containerization allows packaging of your application into a lightweight container with its own operating system
- Container can run in anywhere (no specific VM)
- Docker is the most popular containerization platform
- Containerization paves the way for microservice-based application development and deployment

## Kubernetes (K8s)

- Open Source
- Automates deployment
- Scales up and down
- Facilitates updates
- Facilitates management

# Cluster Creation

- Visit [cloud.ibm.com](https://cloud.ibm.com)
- Click “Catalog” from the top menu
- Select K8s service
- Click “Create” button in the bottom
- Select “Free” and hit “Create Cluster”





- Follow the instructions in the “Access” tab:

Clusters / mycluster

mycluster

Preparing master, workers...  
Expires in 30 days

Access

Overview

Worker Nodes

Worker Pools

Add-ons

DevOps New

Before your cluster provisions, set up your CLI tools

Download and install a few CLI tools and the Kubernetes Service plug-in.

```
curl -sL https://ibm.biz/ist-installer | bash
```

After your cluster provisions, gain access

1. Log in to your IBM Cloud account. Include the `--sso` option if using a federated ID.

```
ibmcloud login -a cloud.ibm.com -r us-south -g <resourceGroupName>
```

2. Download the kubeconfig files for your cluster.

```
ibmcloud ks cluster config --cluster botqjd0adq96gludr0
```

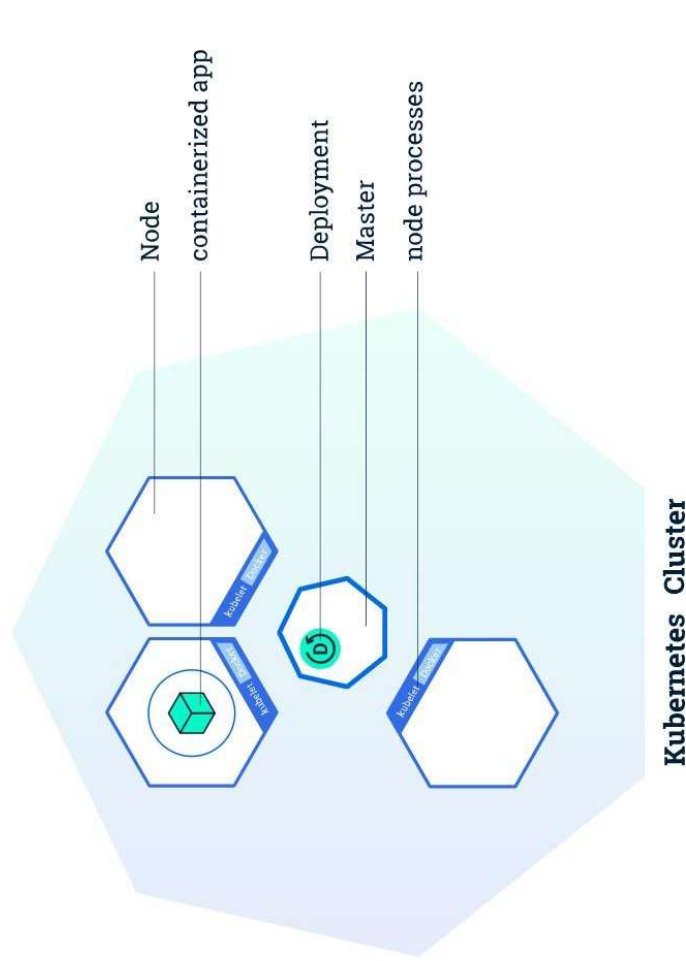
3. Set the KUBECONFIG environment variable. Copy the output from the previous command and paste it in your terminal. The command output looks similar to the following example:

```
export KUBECONFIG=/Users/$USER/.bluemix/plugins/container-service/clusters/botqjd0adq96gludr0/kube-config-hou02-mycluster.yml
```

Alternatively you can directly download your kubeconfig files in manually configure the

## Deployment

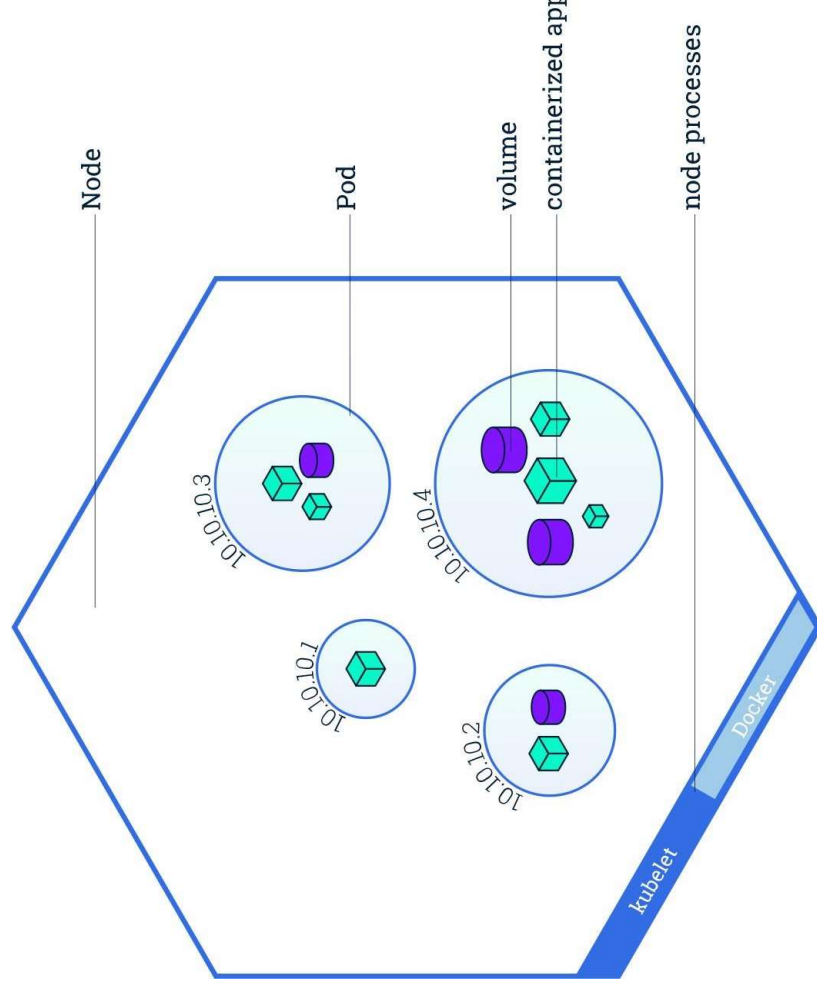
- 
- The diagram illustrates a Kubernetes cluster architecture. It features a central Master node and multiple Worker nodes. The Master node is represented by a hexagon with a green circle containing a white 'D' icon. It is labeled 'Master' and 'node processes'. The Worker nodes are represented by hexagons with a green cube icon. They are labeled 'Node' and 'containerized app'. The Worker nodes are connected to the Master node via a network, indicated by lines. The entire cluster is set against a light blue background with a large, faint, light blue hexagon shape.





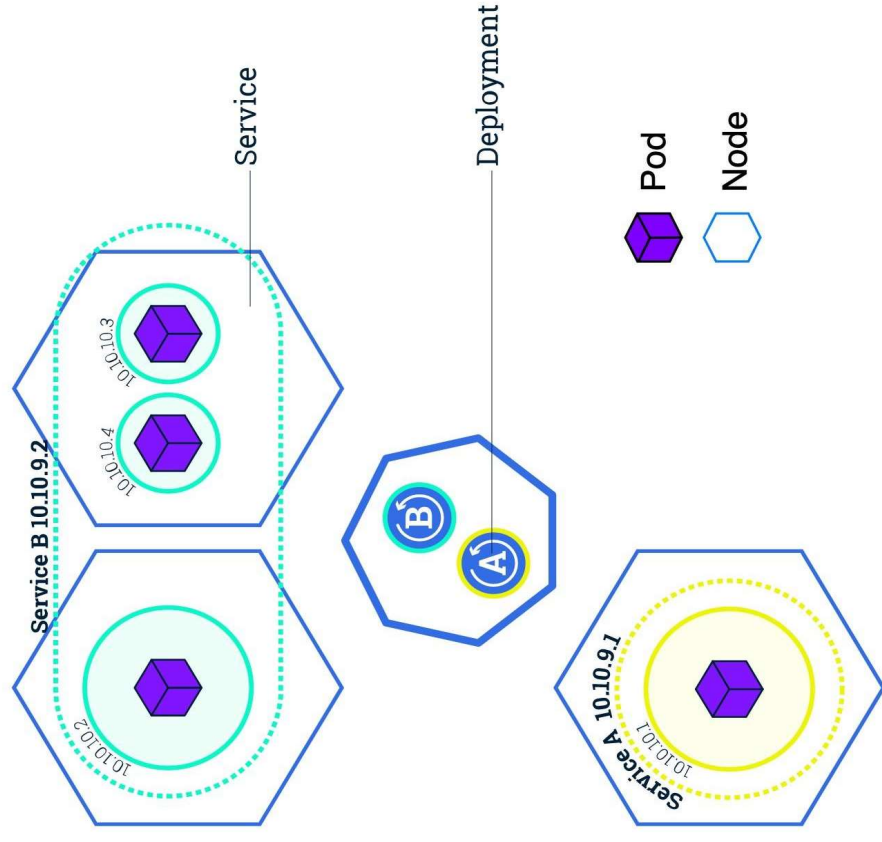
## Exposing Services

- Pods are mortal
- A ReplicaSet is used to create new ones
- However when pod dies, IP will be lost as well
- Services are used to keep things functioning as configured
  - Set of pods and policy to access them as .yaml file



## Services

- *ClusterIP* (default) - Exposes the Service on an internal IP in the cluster.
- *NodePort* - Exposes the Service on the same port of each selected Node in the cluster using NAT. Makes a Service accessible from outside the cluster using `<NodeIP>:<NodePort>` (superset of ClusterIP).
- *LoadBalancer* - Creates an external load balancer in the current cloud (if supported) and assigns a fixed, external IP to the Service (superset of NodePort).
- *ExternalName* - Exposes the Service using an arbitrary name (specified by `externalName` in the spec) by returning a CNAME record with the name. No proxy is used. This type requires v1.7 or higher of kube-dns.





# Data Communications & Networks

