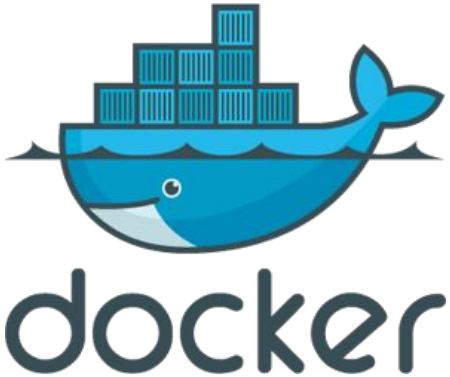
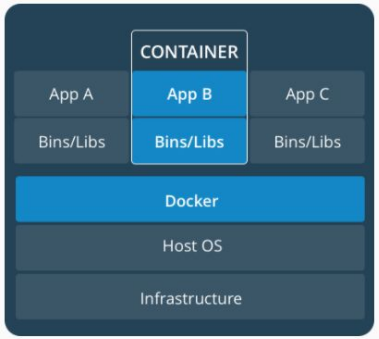
Curs docker/kubernetes

Joan Porta

Docker, tecnología





Docker crea contenedores ligeros y portables para las aplicaciones software que puedan ejecutarse en cualquier máquina con Docker instalado, independientemente del sistema operativo que la máquina tenga por debajo, facilitando así también los despliegues

Docker, avantatges principals

Aventatges developers:

Portabilitat entre entorns, si funciona al desktop, funciona a PRO!

Lleugeresa dels contenidors--> microserveis.

Deployments nous serveis a PRO inmediats! Rollback també inmediat (si DB backwards compatible)

Aventatges Sistemes:

Infraestructure as code! La definció i l'historic de canvis dels nostres servidors està a gitlab!!!!!!

Lleugeresa contenidors: ens permet trindre un servei per contenidor. Ara tenim varis serveis per VM.

Creació d'entorns de manera molt més ràpida. component-reuse

Facil fer deploy's de codi nou, ja que depleguem imatges ja creades previament. Rollbacks rapids.

Self Healing

Si funciona a CPD, funciona a amazon, CGE,...

Diferències amb Virtual Machines

Alta densitat:

Volem crear 10 maquines iguals amb Ubuntu+Java. Cada VM ocupa 10G

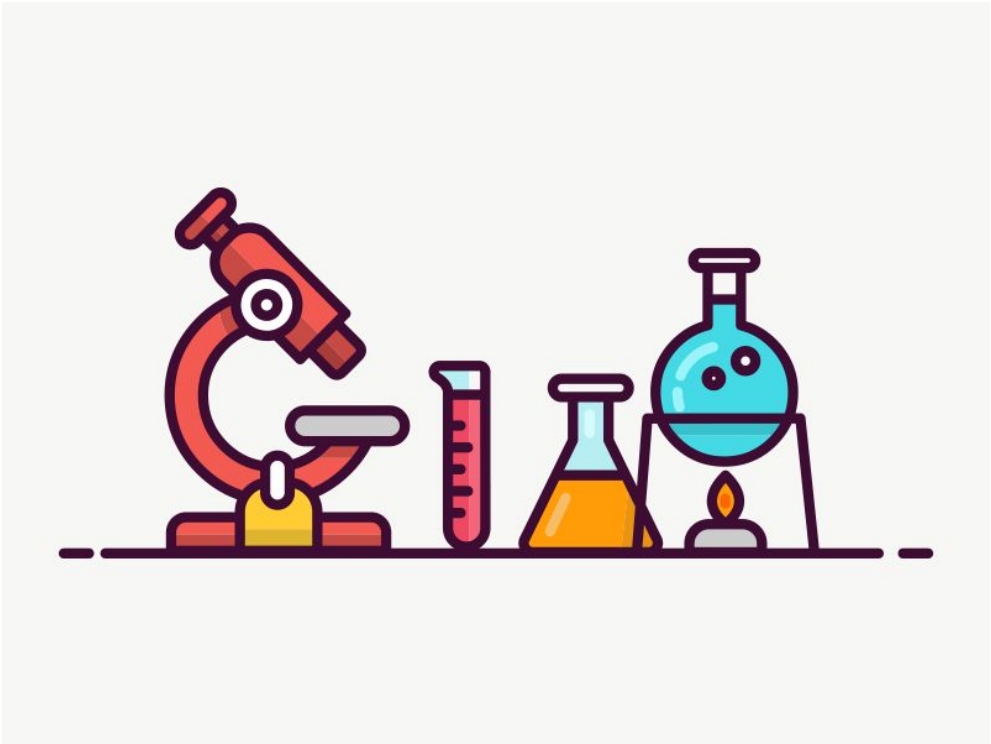
Si ho fem amb VM: TOTAL=100G

Si ho fem amb Containers ~=10G

Infraestructure as code

A gitlab tenim definit exactament com hem creat cada maquina i la seva evolució al llarg del temps.

Get started! Practica 1. Container amb Ubuntu



Creació de dockerfiles, registry

Example: Memcached

```
FROM ubuntu
RUN echo "deb http://archive.ubuntu.com/ubuntu precise main
  universe" > /etc/apt/sources.list
RUN apt-get update
RUN apt-get install -y memcached
```

- Dockerfile: https://github.com/jbarbier/Dockerfile-Basics-Docker-Workshop-2-

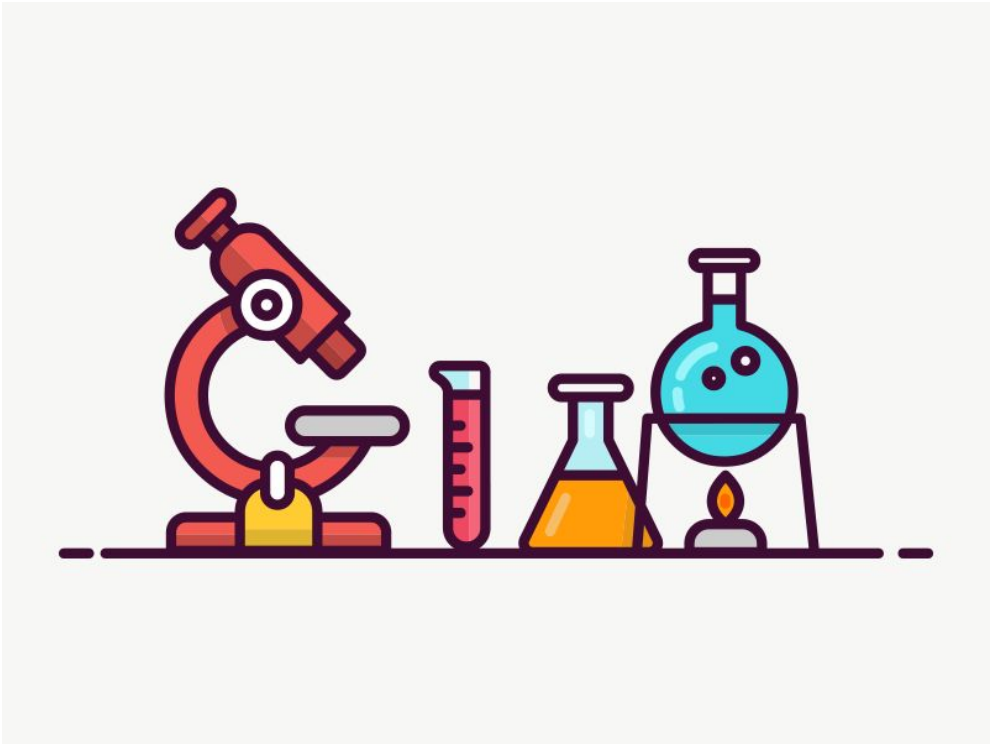
```
$ docker build –t memcached_d1 .
```

- Test it

```
$ docker run -i -t memcached_d1 /bin/bash
root@1f452c9442fb:/# memcached -u daemon -vvv
```

docker

Practica 2, Dockerfile y upload a registry



Gitlab

docker-main/dockerfiles/{prostgres, redis, rabbitmq,...}

docker-main/environments/environments/{int,pre,pro,...}

docker-private/environments/environments/{int,pre,pro,...}

Kubernetes, tecnología

Creada por Google, código abierto 100%. Docker empresa privada. Hay muchísima más contribución a kubernetes que a docker.

Permite montar un cluster de nodos con containers encima que se pueden mover de un nodo a otro. Docker tiene su tecnología de cluster: Swarm.

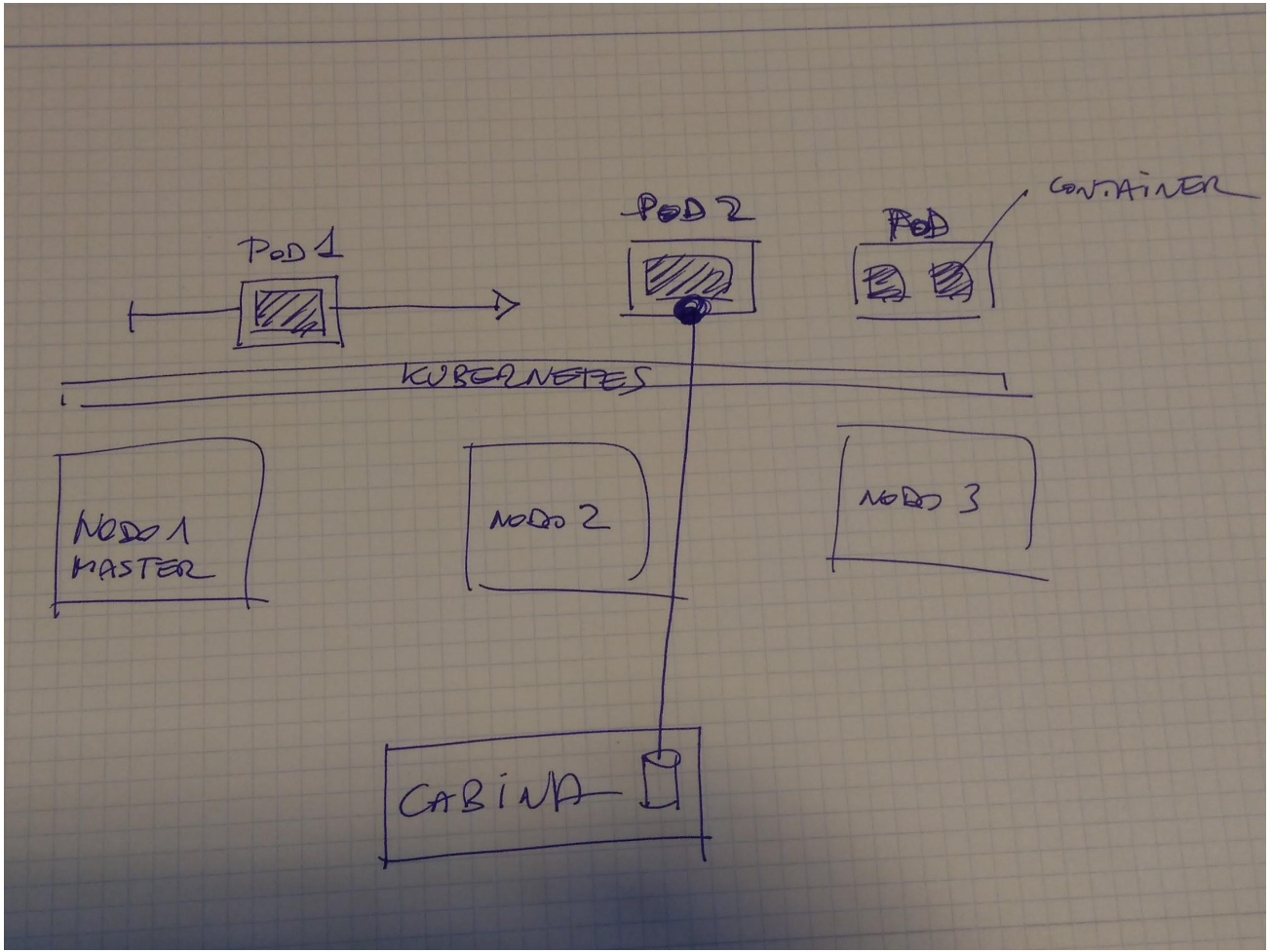
Kubernetes corre encima de nodos docker, él crea sus propios contenedores.

Docker vs kubernetes

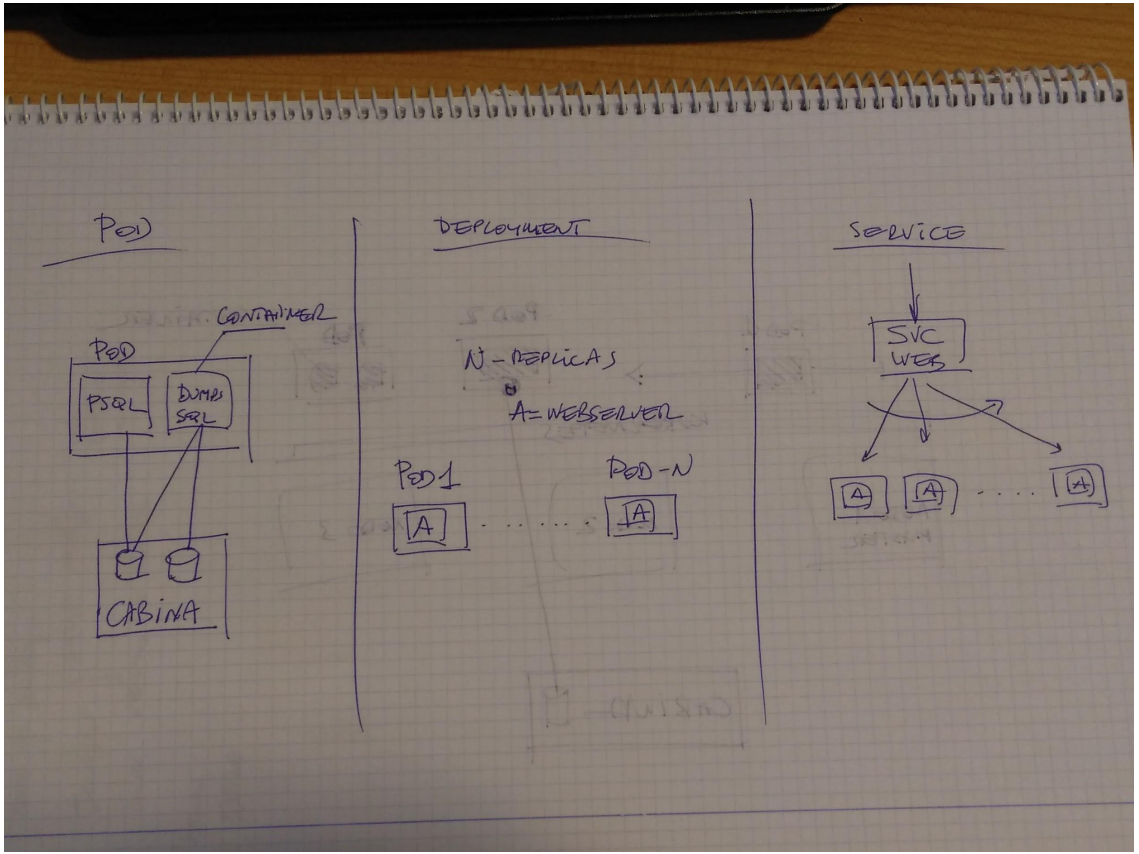
Kubernetes is widely accepted by the community of developers, despite the hard installation process.

The sole reason behind its popularity is the flexibility it offers and also the fact that it is backed by Google, one of the leading tech giants.

Docker, on the other hand, has a small community. It is growing slowly.

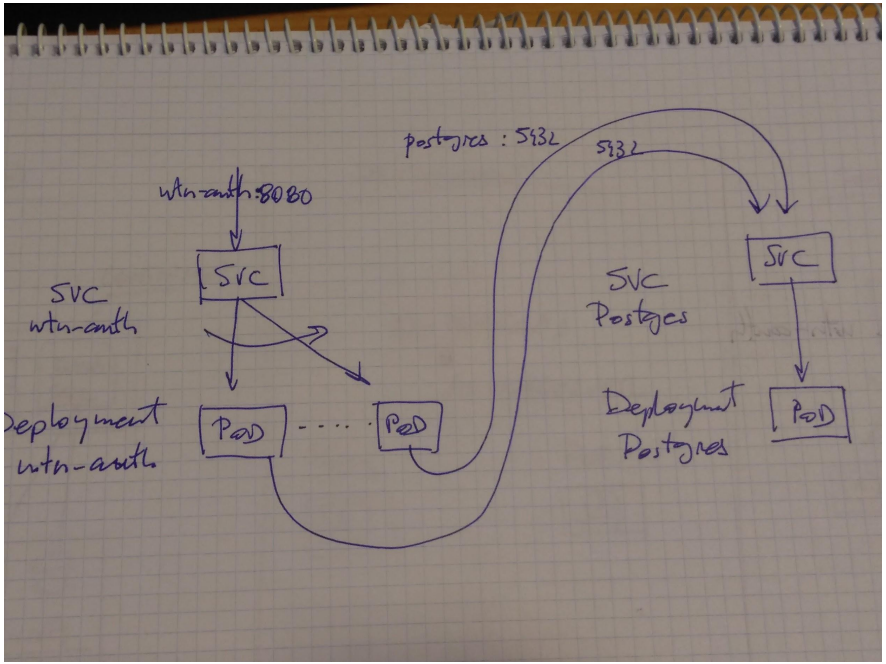

POD 1
POD 2
POD
CONTAINER
KUBERNETES
NODO 1
MASTER
NODO 2
NODO 3
CABINA

Conceptos Kubernetes, POD, Deployment, Service

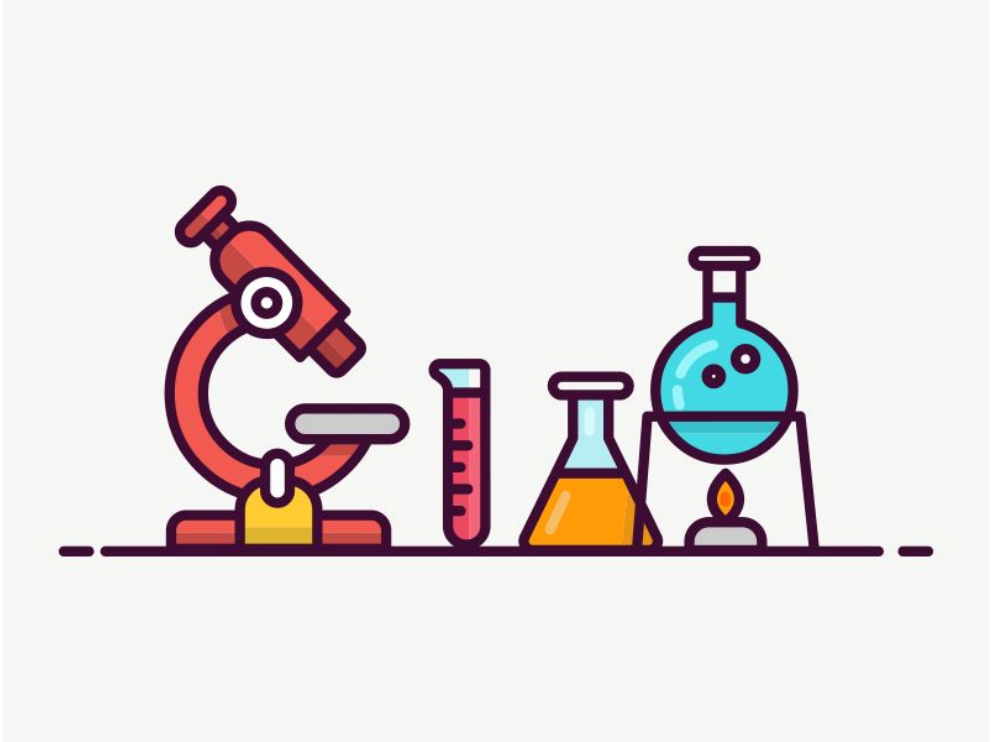


Conceptos k8s: IP’s, DNS

Kubernetes crea un POD automaticamente que es un DNS y registra nombre-IP cada vez que creamos un POD’s. Todo es DHCP! Cuando creamos el cluster, asignamos rango IP. Los POD’s se comunican por nombre (o por IP si queremos)



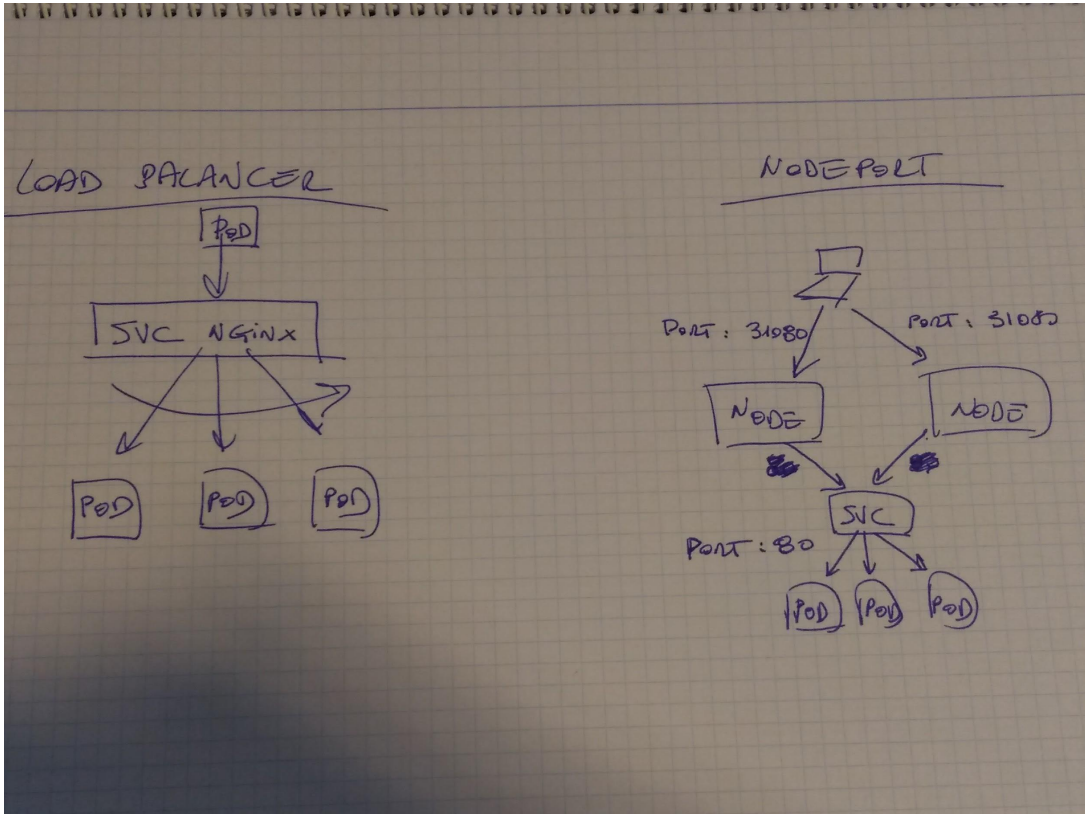
Pràctica 4



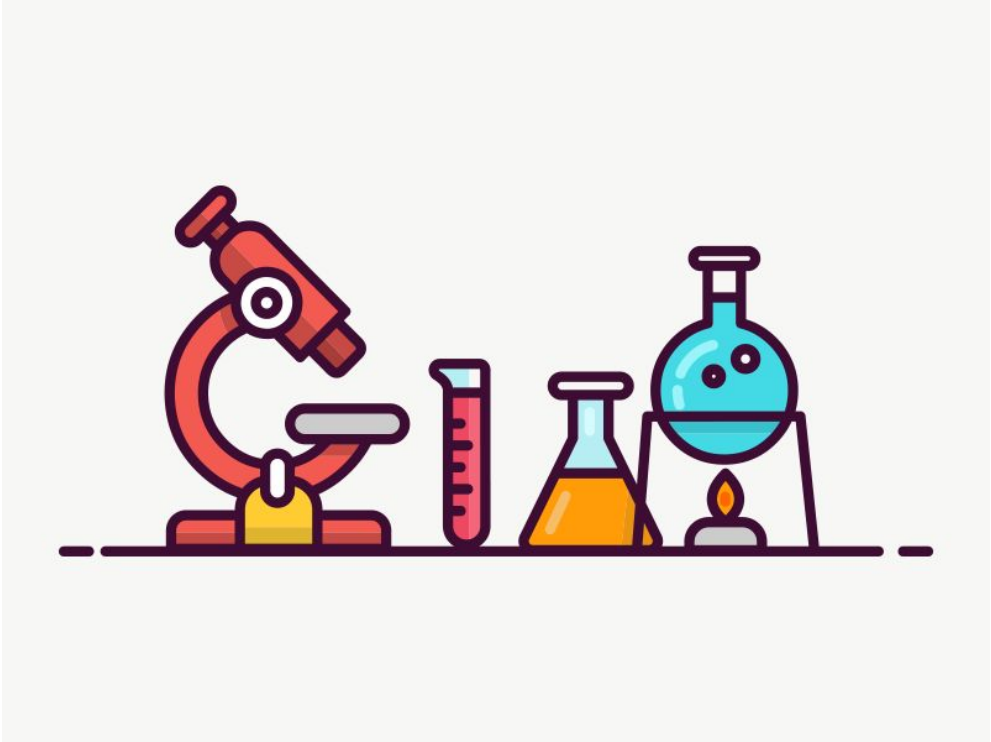
nginx-deployment.yaml

```yaml
apiVersion: apps/v1beta1
kind: Deployment
metadata:
  name: nginx
spec:
  replicas: 2
  template:
    metadata:
      labels:
        lan: frontend
        servicio: nginx
    spec:
      containers:
      - name: nginx
        image: registry.cpd.wtransnet.net:5000/curso/webserver:1.0
        ports:
        - containerPort: 80
      imagePullSecrets:
      - name: registrypullsecret
```

Services: Load balancer, Nodeport

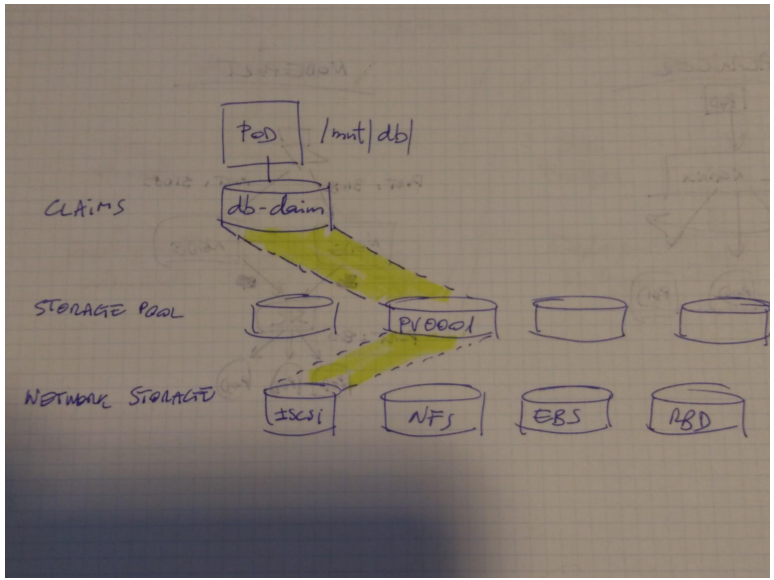


Practica 5



Persistent volumes

1) Volem treure les dades fora del POD per no pèrdreles quan es reinicia (ex: caiguda, o actualitzacio)
2) Volem mantenir el mateix .yaml del deployment per tots els entorns, independentment de l'storage y el medi fisic (ISCSI, NFS, EBS,...)



Encara que canvii el POD de node,

segueix estant conectat al Storage

Persistent Volumes, ejemplo: nginx con gluster

```yaml
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: web-nginx
  labels:
    app: web-nginx
spec:
  replicas: 1
  template:
    metadata:
      labels:
        app: web-nginx
    spec:
        containers:
        - name: web-nginx
          image: registry.wtransnet.local:5000/sistemas/web-nginx:latest
          imagePullPolicy: Always
          ports:
          - containerPort: 80
          volumeMounts:
          - mountPath: /mnt/nfs
            name: pvc-web-nginx-nfs
        volumes:
        - name:  pvc-web-nginx-nfs
          persistentVolumeClaim:
            claimName:  pvc-web-nginx-nfs
```

```yaml
apiVersion: v1
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc-web-nginx-nfs
spec:
  accessModes:
    - ReadOnlyMany
  resources:
    requests:
      storage: 200Mi
  selector:
    matchLabels:
      etiqueta: pv-web-nginx-nfs
```

```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv-web-nginx-nfsdata
  labels:
    etiqueta: pv-web-nginx-nfs
spec:
  capacity:
    storage: 200Mi
  accessModes:
    - ReadOnlyMany
  persistentVolumeReclaimPolicy: Recycle
  glusterfs:
    path: vol01
    endpoints: glusterfs-cluster
    readOnly: false
```

Configmaps

Queremos conservar la definicion del POD (.yaml) aunque cambiemos de entorno, le pasamos unas variables o otras dependiendo del entorno.

El configmap:

Inserta variables de entorno en un POD, visbles con un “env” desde el S.O del POD

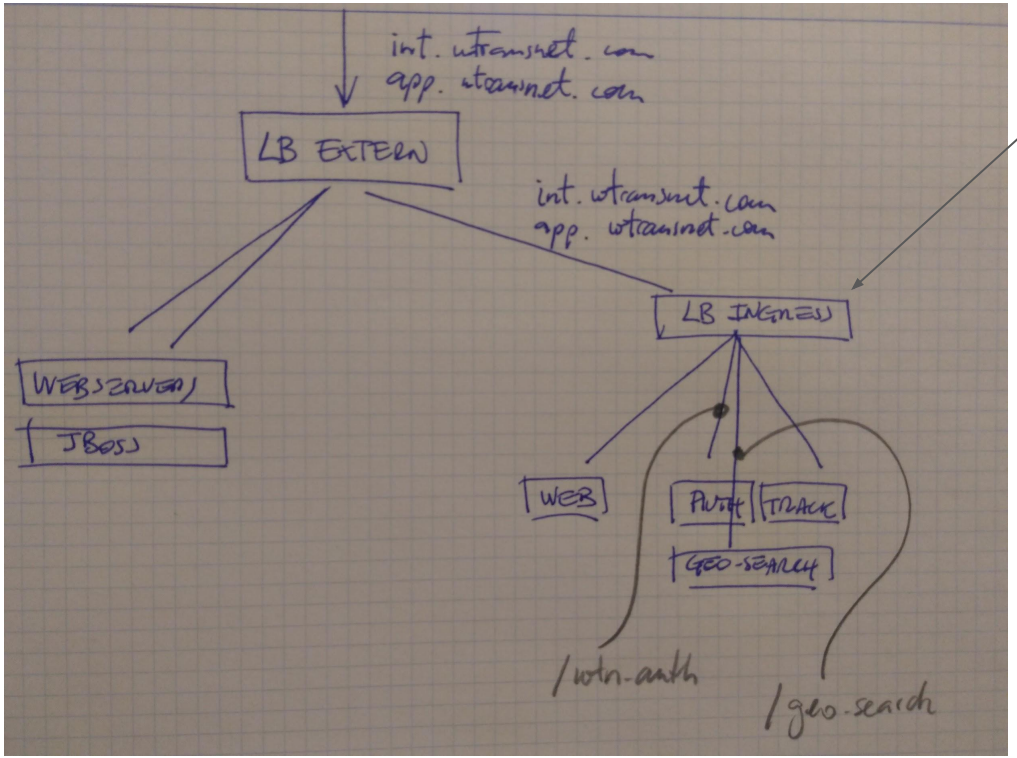
Inserta ficheros: ejemplos: los ficheros del sites enabled de un POD.

Configmaps, ejemplo

shared-configmap.yaml:

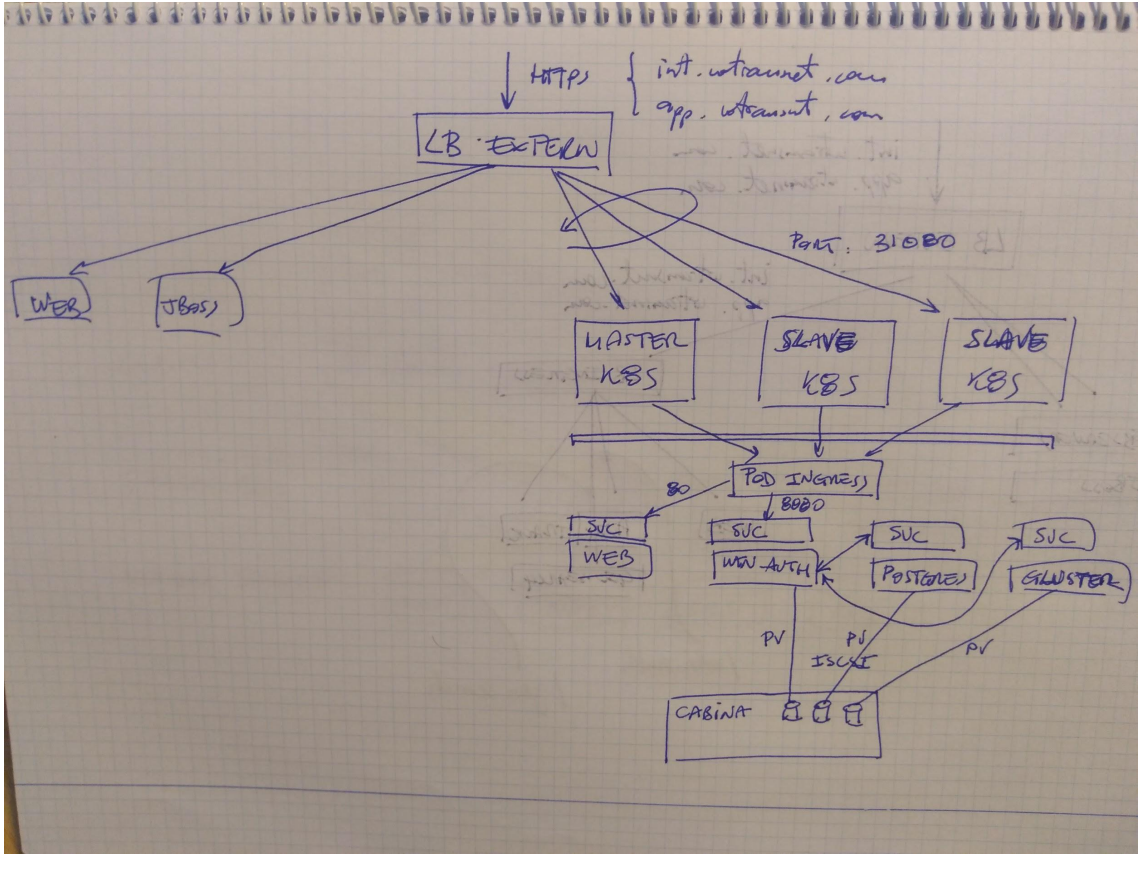
```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: shared-configmap
data:
  I18N_DIR: "/mnt/nfs/v4/app-data/literals/"
  I18N_CACHE_SECONDS: "10"
  REDIS_HOSTNAME: redis
  URL_WTN3: "http://app.wtransnet.local:8080"
  URL_AUTH: "http://wtn-auth:8080"
  RELAYHOST: "zimbra.wtransnet.local"
```

Ingress

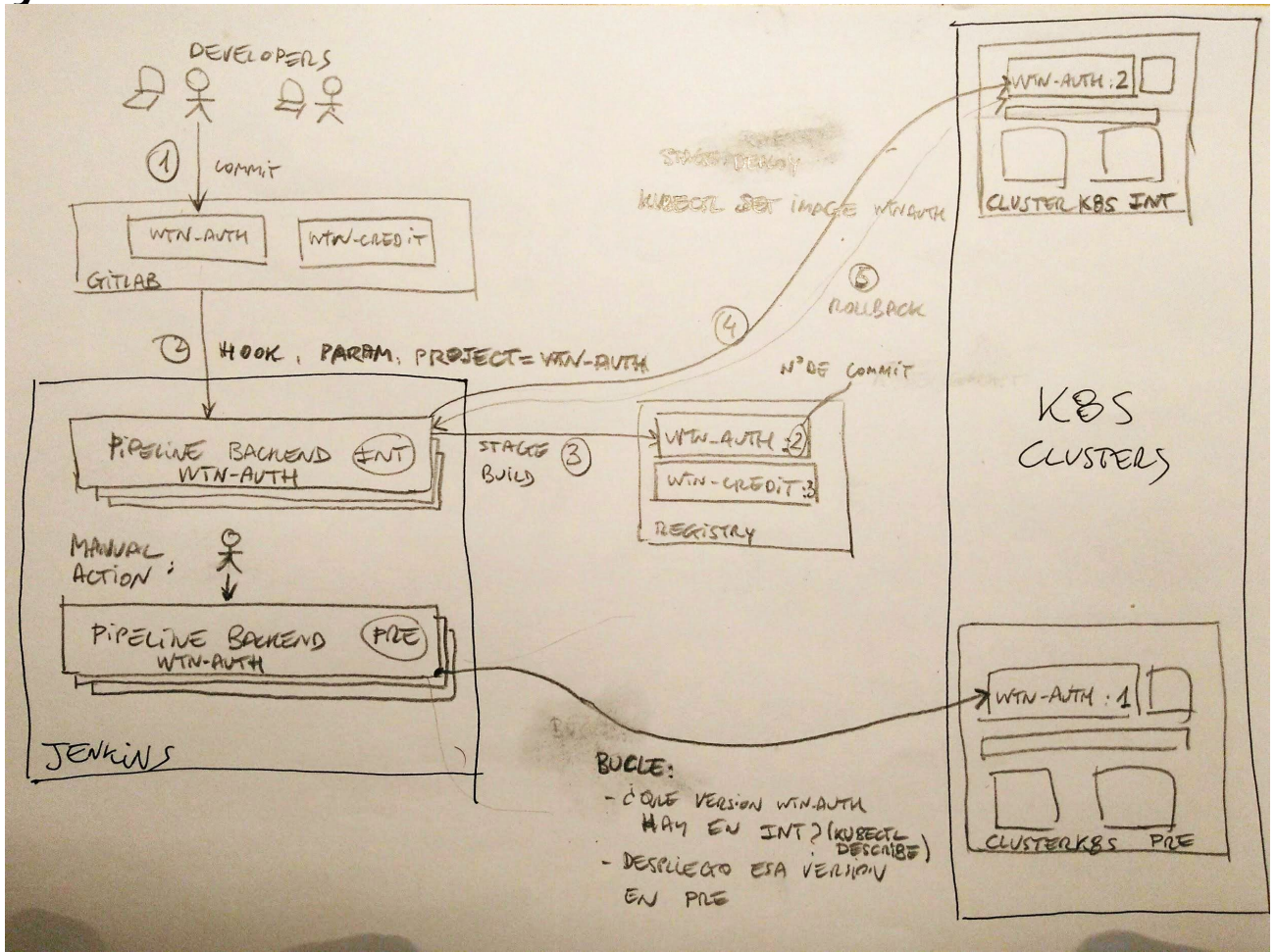



```yaml
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: nginx-ingress-lb
  namespace: default
spec:
  rules:
  - host: int.wtransnet.com
    http:
      paths:
      - path: /
        backend:
          serviceName: web-nginx
          servicePort: 80
  - host: app-int.wtransnet.com
    http:
      paths:
      - path: /
        backend:
          serviceName: web-nginx
          servicePort: 80
      - path: /bridge-wtn3
        backend:
          serviceName: wtn-bridge-wtn3
          servicePort: 8080
      - path:  /tracking
        backend:
          serviceName: wtn-tracking
          servicePort: 8080
```

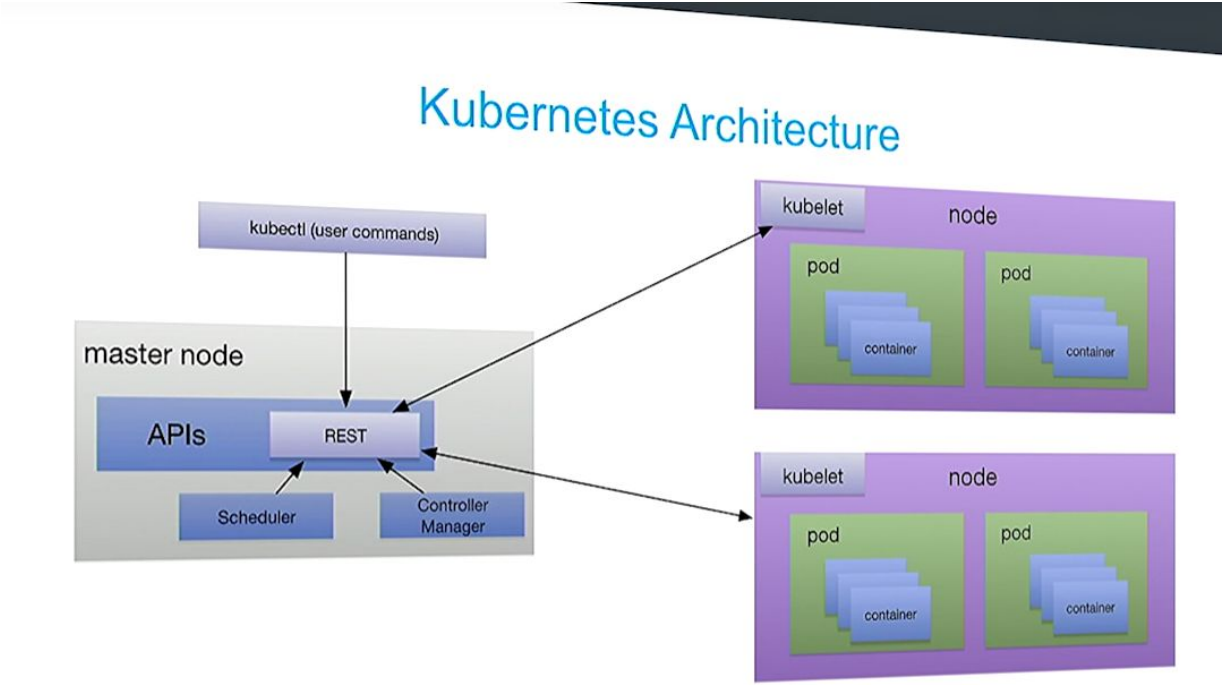
Esquema entorno completo wtransnet



Deployments desde Jenkins

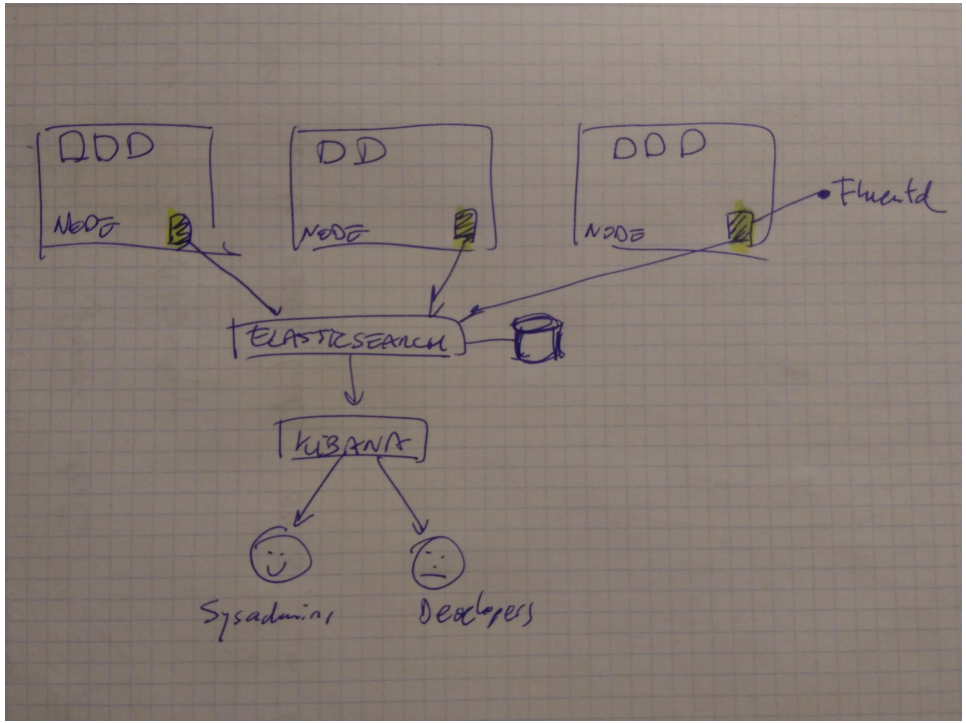


Arquitectura cluster K8S

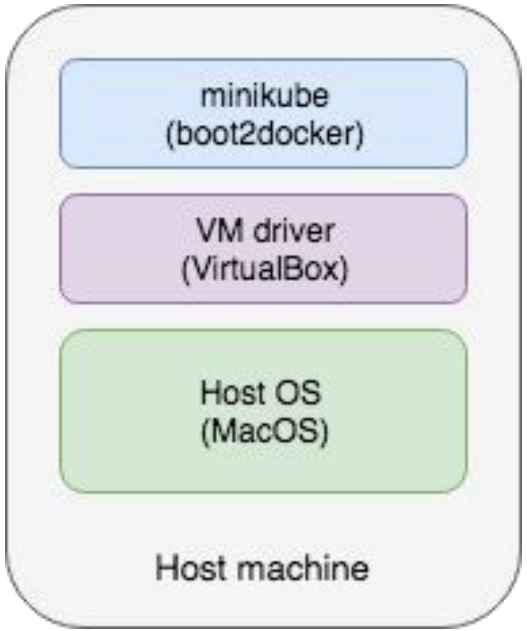


Logs. Centralització.

Tots els pods treuen logs a stdout/stderr



Minikube I



VIRTUALBOX:

```
apt-get install -y virtualbox
```

KUBECTL:

```
wget
https://storage.googleapis.com/kubernetes-release/release/$KUBECTL_RELEASE/bin/linux/amd64/kubectl.....
```

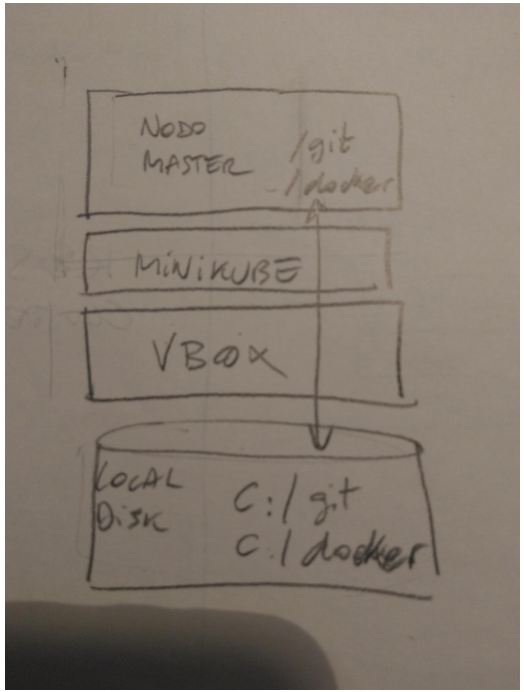
MINIKUBE:

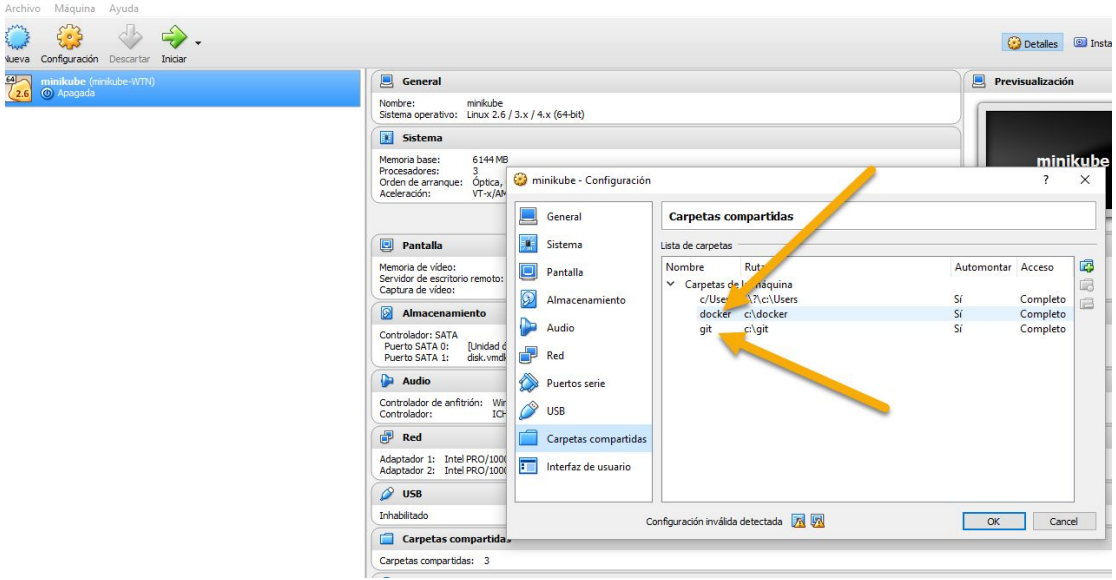
```
curl -Lo minikube
https://storage.googleapis.com/minikube/releases/$MINIKUBE_RELEASE/minikube-linux-amd64 &&.........
```

```
minikube start --kubernetes-version=$KUBERNETES_RELEASE
--insecure-registry=registry.wtransnet.local:5000 --memory 6144
--cpus 3
```

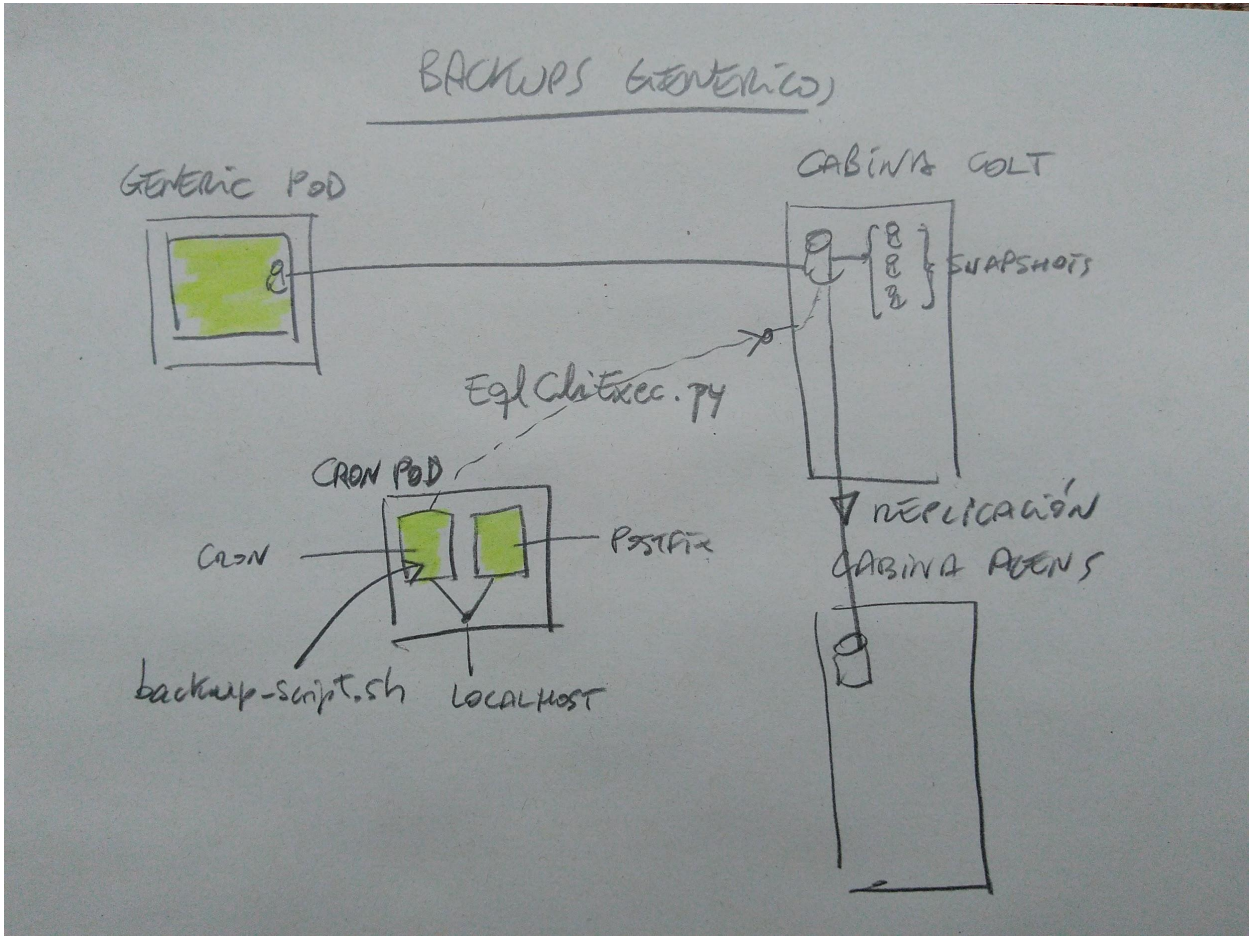
Minikube II

```
vboxmanage sharedfolder add "minikube" --name "docker" --hostpath /docker --automount
vboxmanage sharedfolder add "minikube" --name "git"    --hostpath /git --automount
```

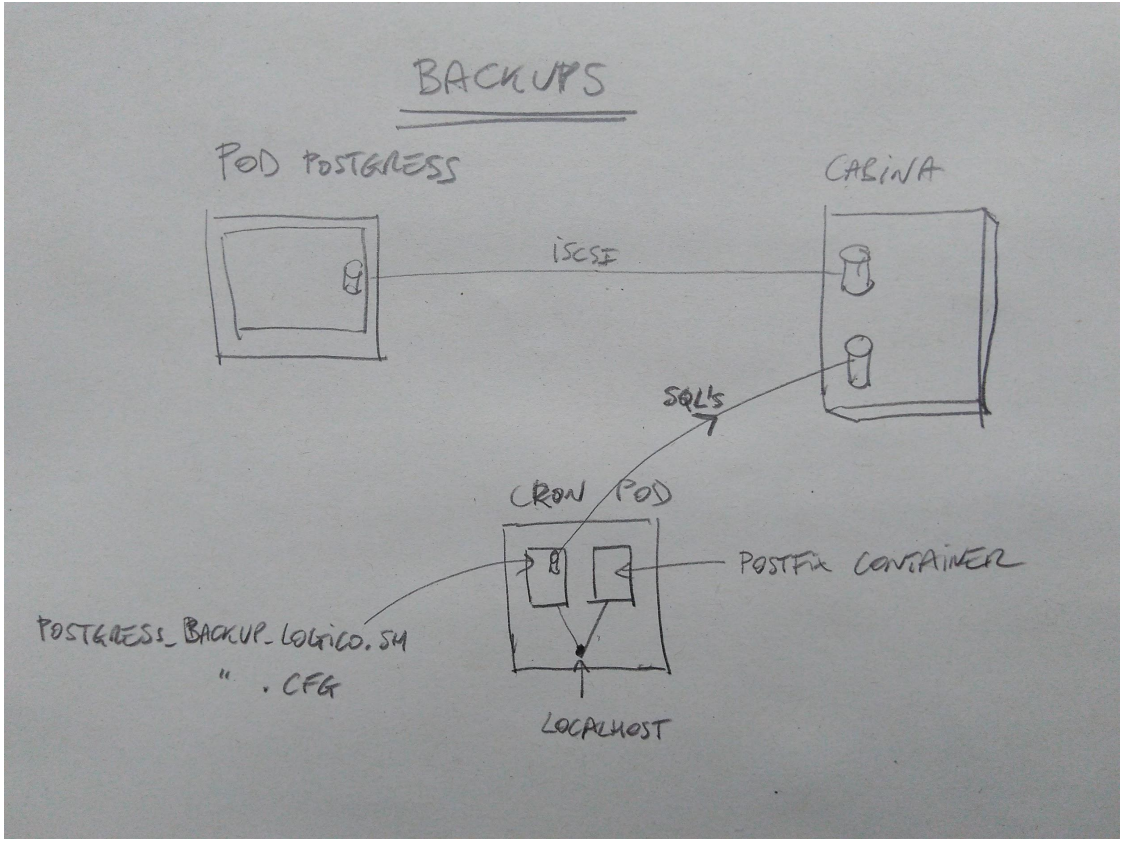




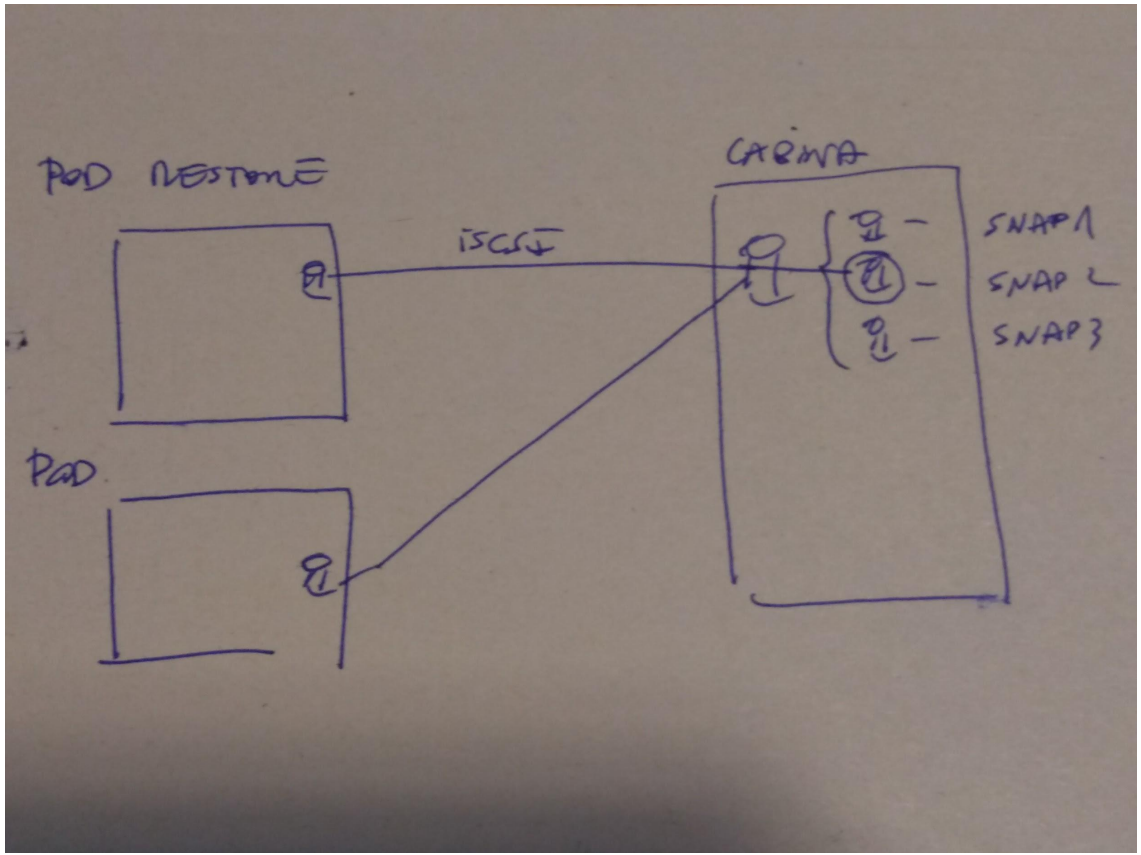
BACKUPS I


BACKUPS GENERICOS
GENERIC POD
CABINA COLT
SNAPSHOTS
EqlCliExec.py
CRON POD
CRON
POSTFIX
backup-script.sh
LOCALHOST
REPLICACIÓN
CABINA PROMS

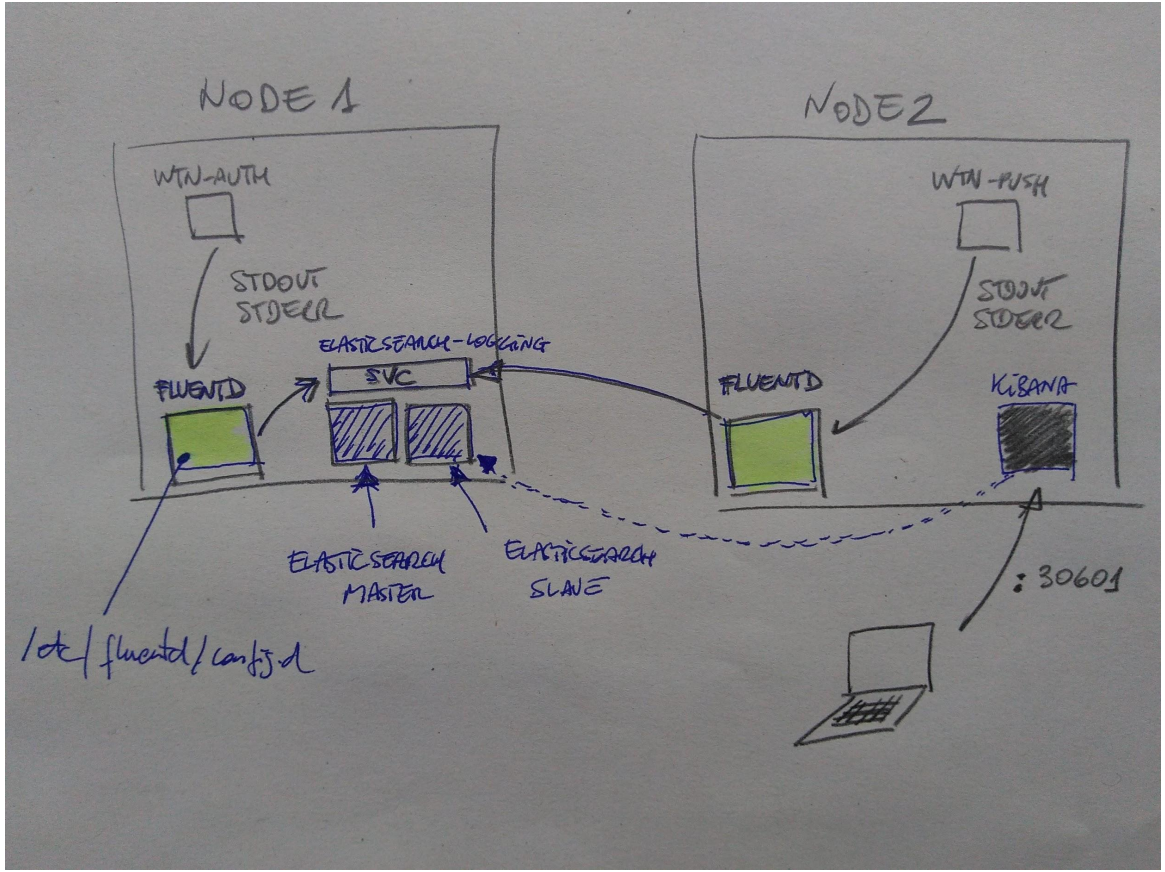
BACKUPS POSTGRES



RESTORE

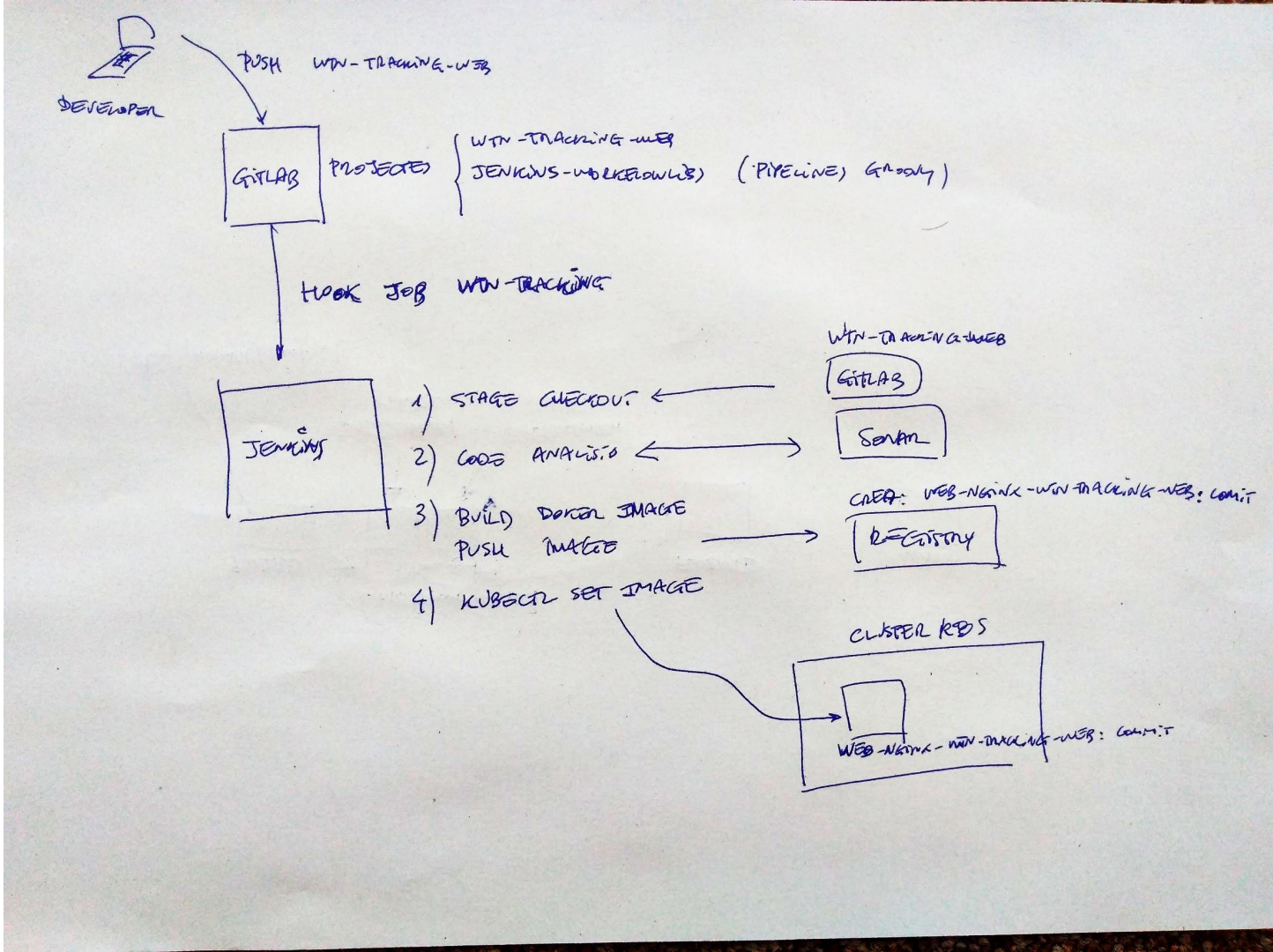


Logs

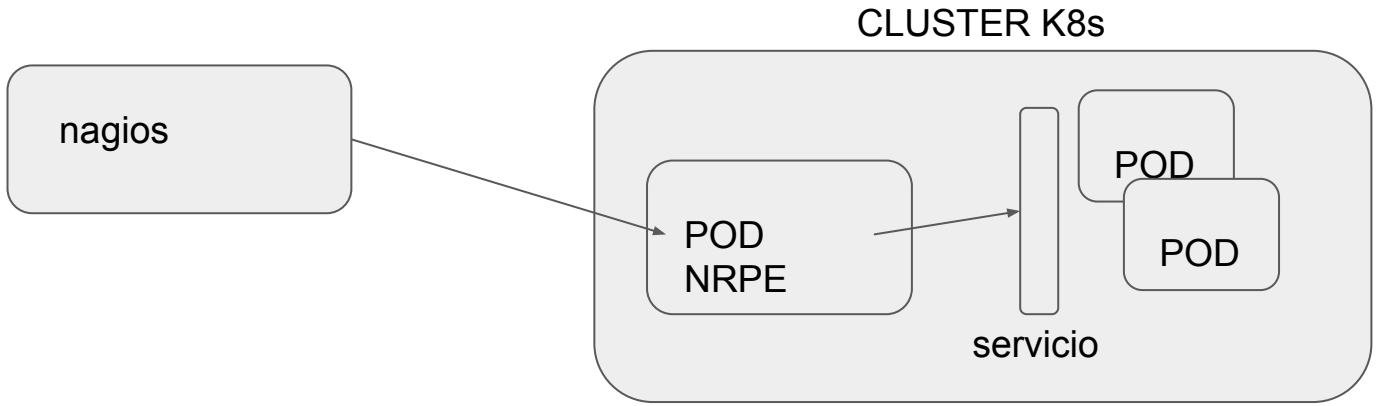


Cada node deixa logs dels seus POD’s a /var/log/containers

Jenkins Frontend Pipeline

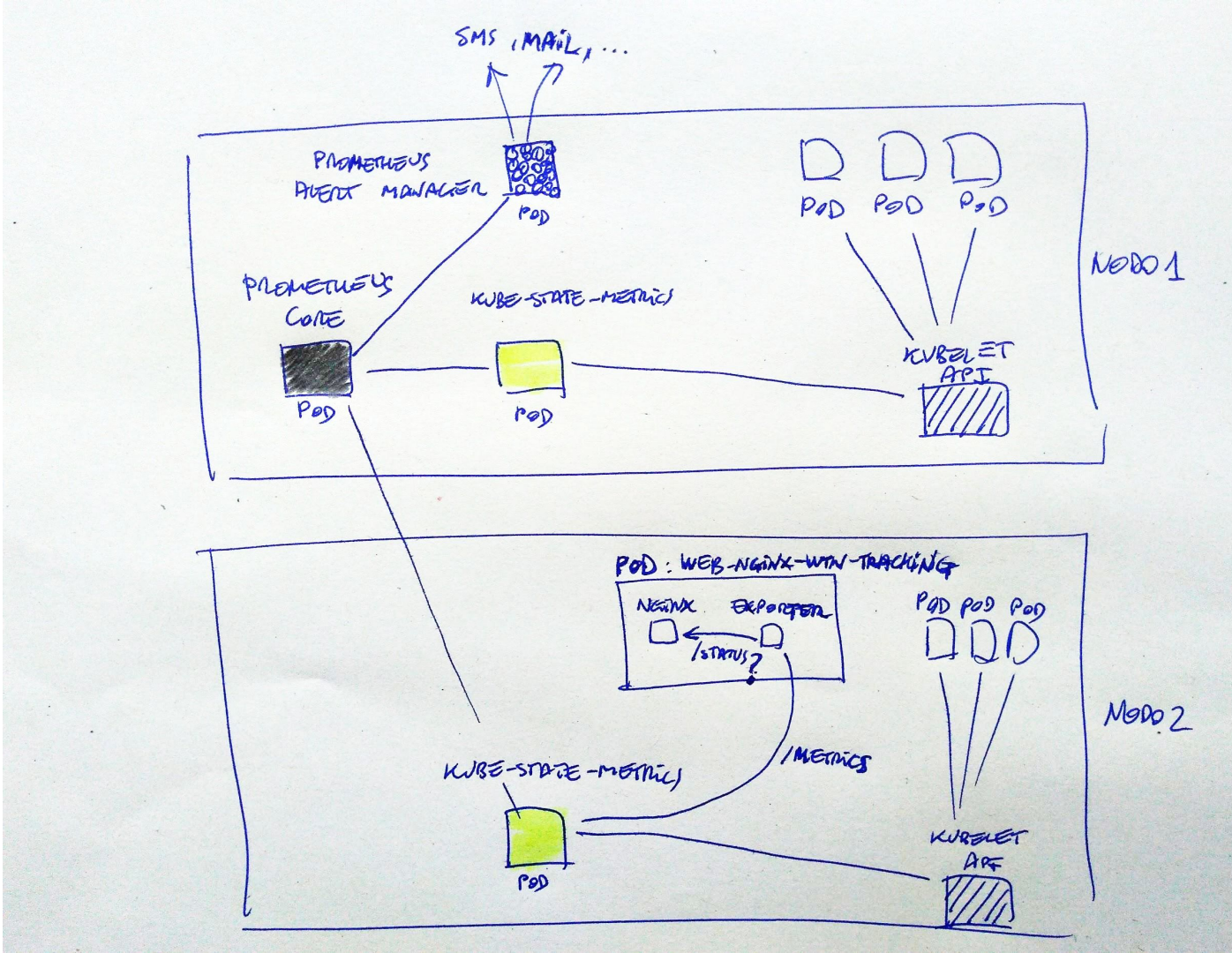


Monitorització, alarmes, opció Nagios fora cluster



Inconvenient: metriques multidemensionals. Ex: vull saber la suma de RAM que consumeixen tots els PODS que tenen imatge=X

Prometheus +grafana



Prometheus exporters de metricas

https://github.com/prometheus/docs/blob/master/content/docs/instrumenting/exporters.md