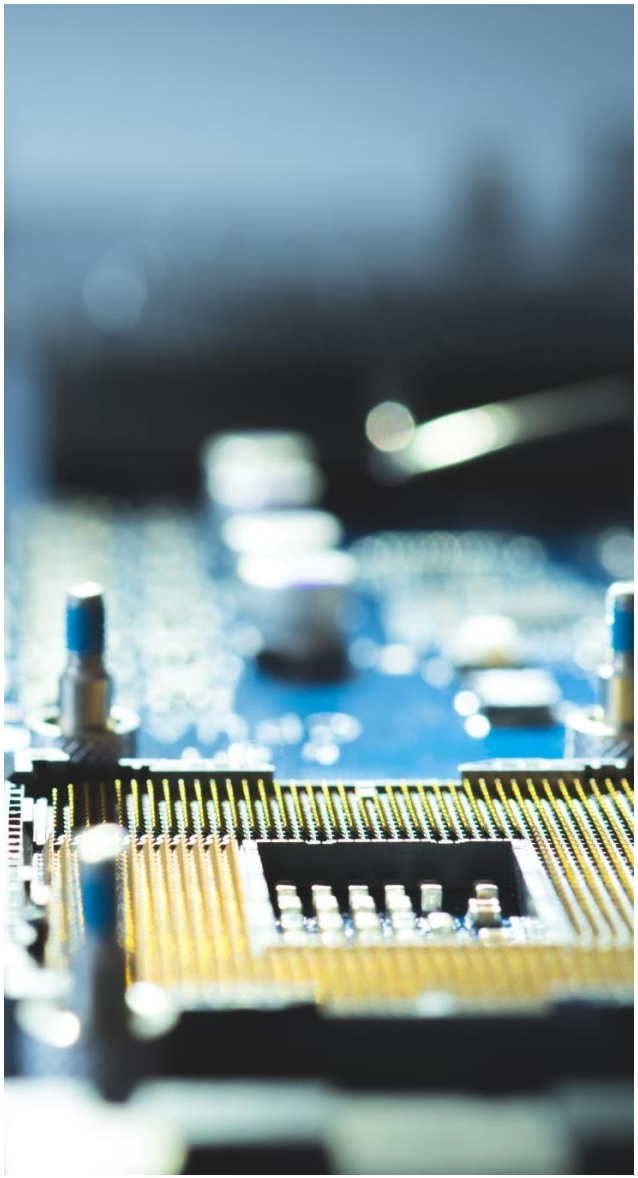
CESTE
Escuela Internacional de Negocios

Kubernetes

Teoría y Prácticas

Diplomatura de Informática Militar

Indice

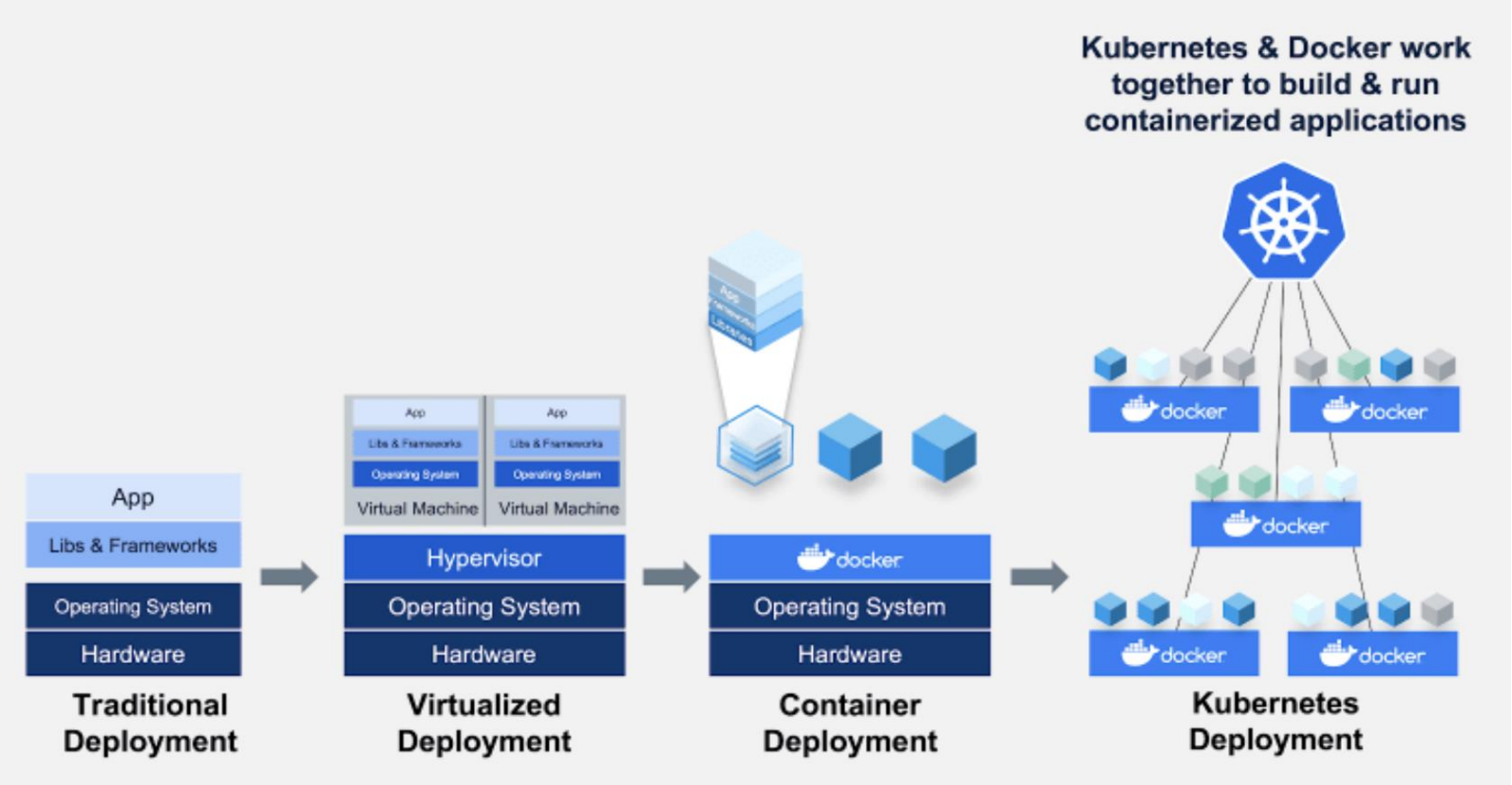


Introducción a Kubernetes

CESTE

Docker y Kubernetes

Docker permite la ejecución de software como **contenedores** en una máquina.
Kubernetes (K8s) permite la **orquestación** de la ejecución de muchos contenedores.

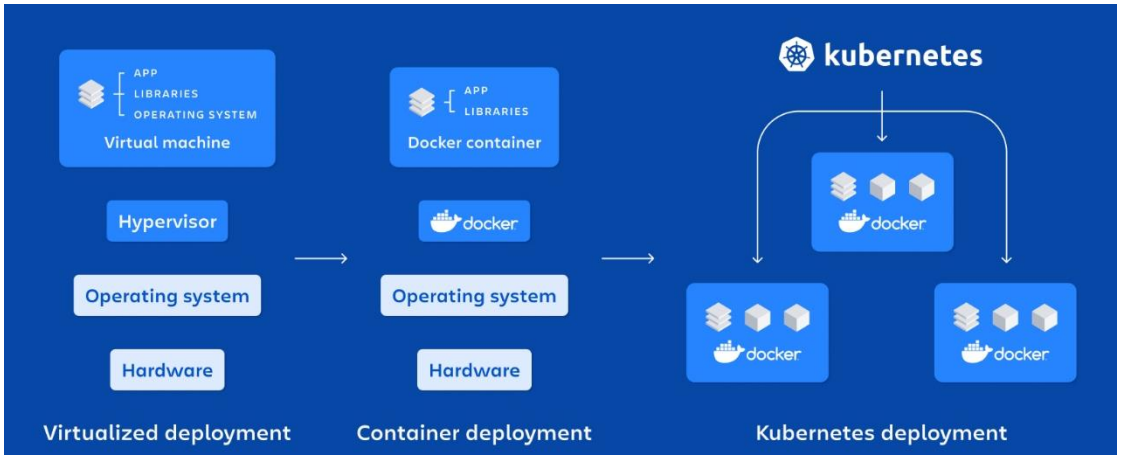


Kubernetes

Si el número de aplicaciones crece en nuestro sistema, es complicado de gestionar. Surge la necesidad de tener un **maestro de orquesta** para coordinar el funcionamiento de un numero de alto de contenedores

Necesitamos una **coordinación** para hacer el despliegue, la supervisión de servicios, el reemplazo, el escalado automático y, en definitiva, la administración de los distintos servicios que componen nuestra arquitectura distribuida.

Kubernetes fue desarrollado **por Google en 2014** y después de su primere reléase en 2015 fue donado a Cloud Native Computing Foundation. Desde entonces todas las versiones estables se publican con Apache Licencing



Kubernetes

Orquestación de conjuntos de contenedores desplegados en diferentes servidores

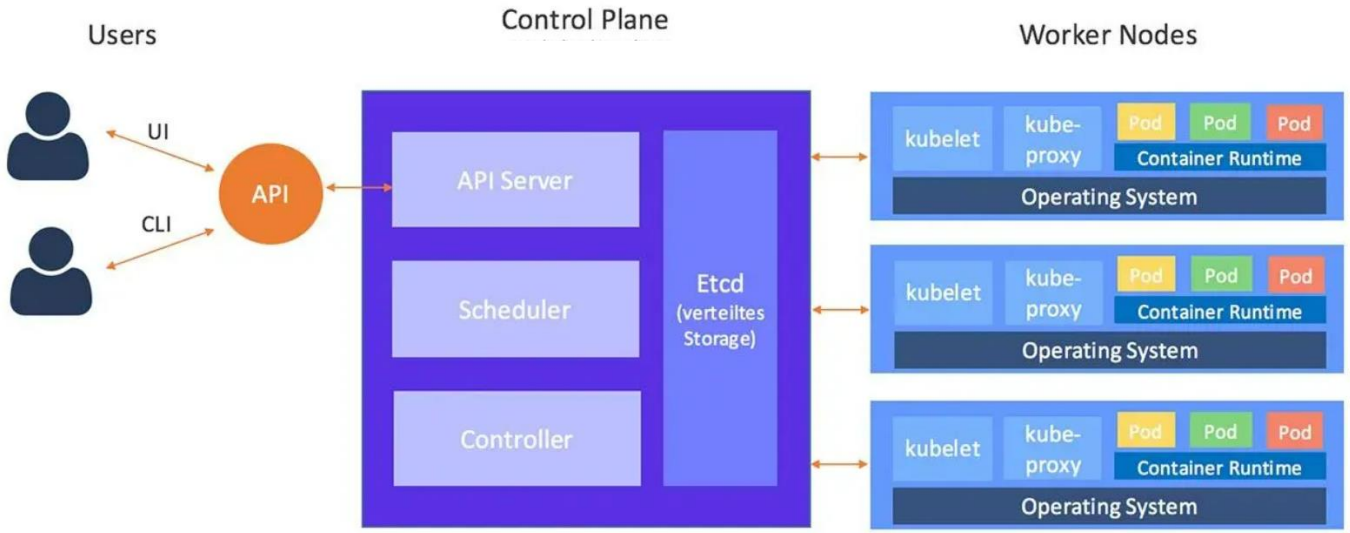
- Colocación automática, red, despliegue, escalado, roll-out/-back, A/B testing

Declarativo – not procedimental

- Declarar el estado deseado, reconciliar con él
- Se auto repara

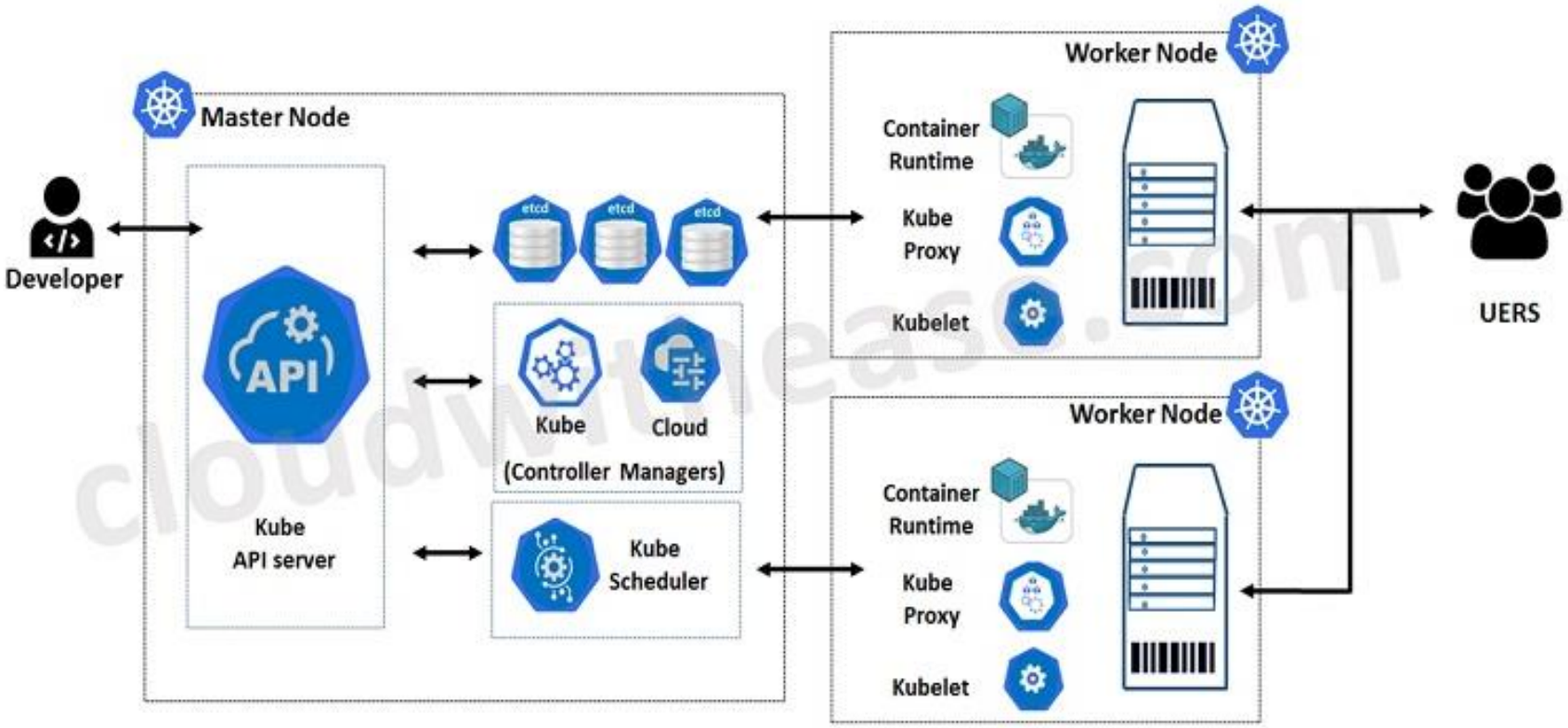
Portabilidad de las cargas de trabajo

- Abstracción de las peculiaridades de los proveedores de cloud
- Multiples entoros de ejecución para los contenedores



Kubernetes

Arquitectura de la plataforma Kubernetes



Cluster de Kubernetes

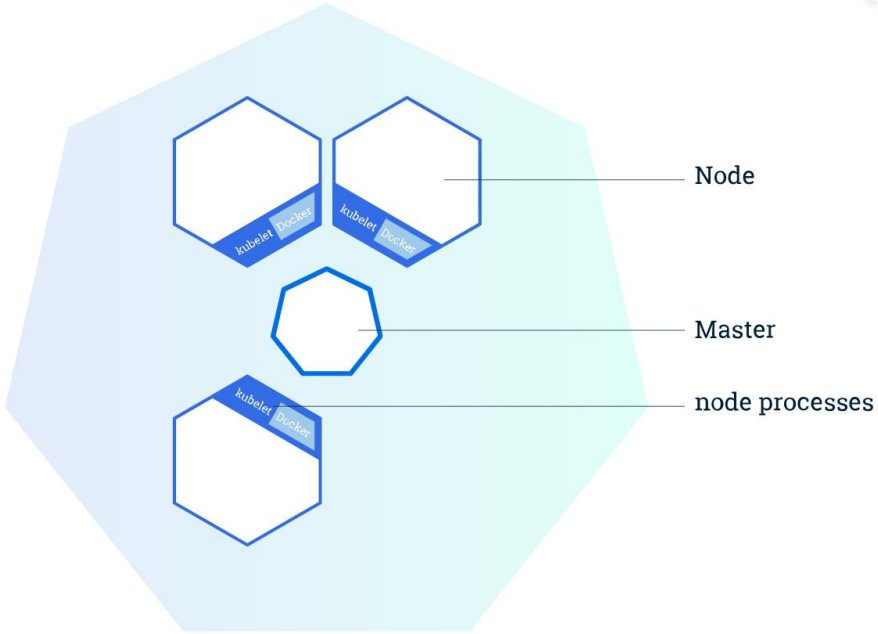
CESTE

Kubernetes

El cluster de Kubernetes

Los **maestros** administran el clúster y los **nodos** se utilizan para alojar las aplicaciones en ejecución

Un clúster de Kubernetes que maneja el tráfico de producción debe tener un mínimo de tres nodos.



El **Maestro** es responsable de gestionar el clúster. Los nodos maestros coordinarán toda la actividad que sucede en su clúster, como programar aplicaciones, mantener el estado deseado, escalar aplicaciones e implementar nuevas actualizaciones.

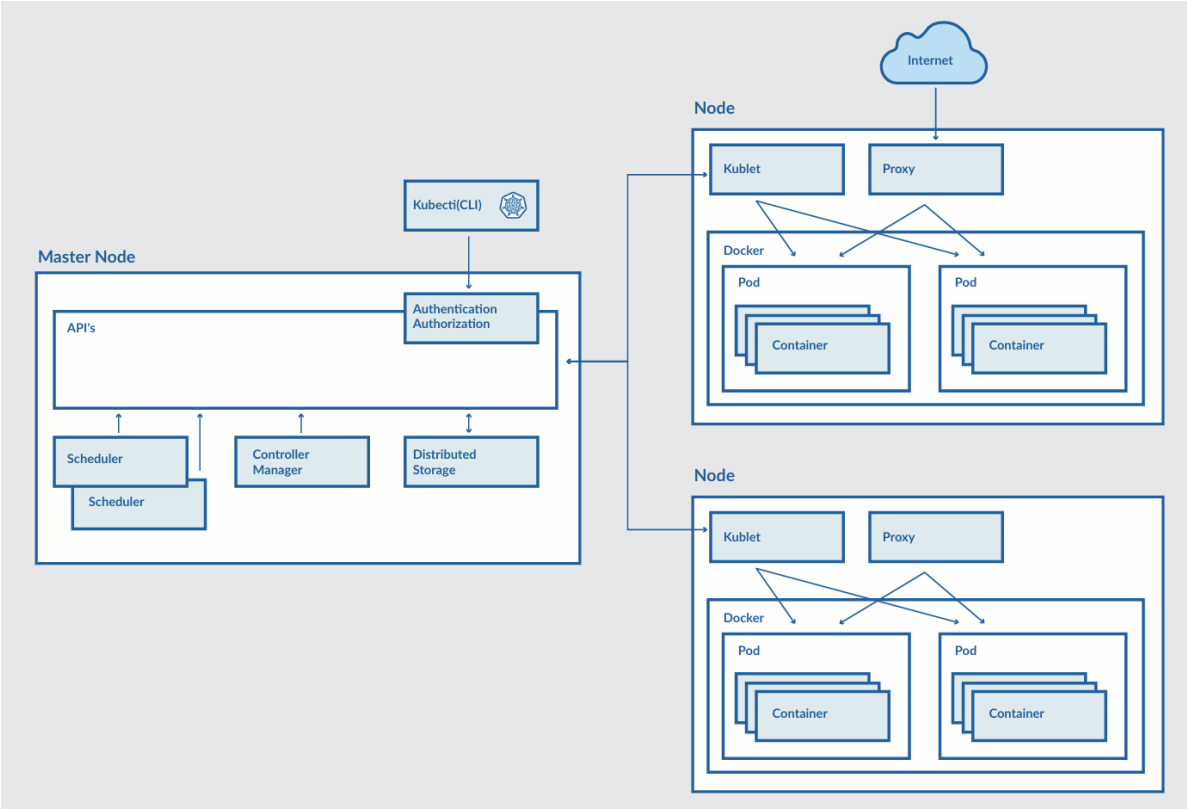
Un **nodo** es una máquina virtual o una computadora física que se utiliza como máquina de trabajo en un clúster de Kubernetes. Cada nodo del clúster es administrado por el maestro.

En un nodo típico, tendrá herramientas para manejar operaciones de contenedores (como Docker, rkt) y Kubelet, un agente para administrar el nodo.

Kubernetes

El cluster de Kubernetes

Los nodos son servidores ‘bare-metal’, máquinas virtuales locales o máquinas virtuales en un proveedor de nube.



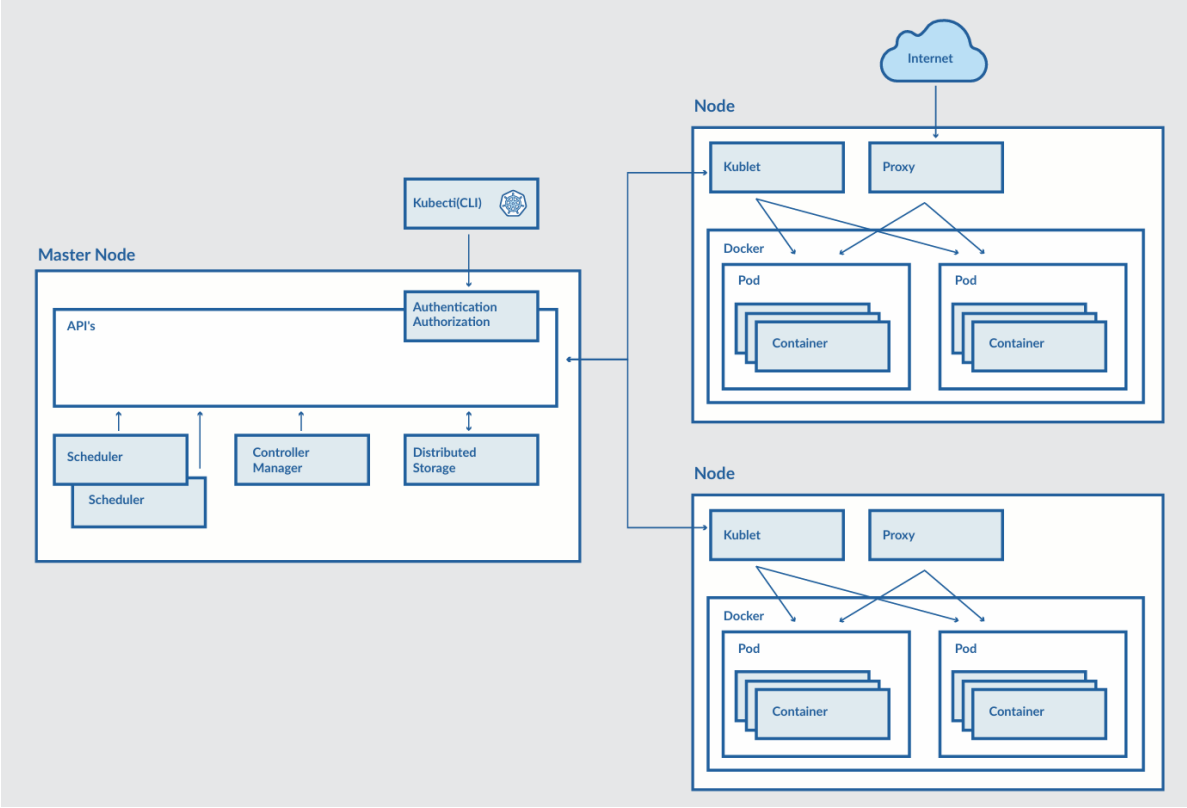
Cada nodo contiene un **runtime** de ejecución de contenedor (por ejemplo, Docker Engine),

kubelet (responsable de iniciar, detener y administrar contenedores individuales según las solicitudes de Kubernetes) plano de control) y kube-proxy (responsable de las redes y el equilibrio de carga).

Kubernetes

El cluster de Kubernetes

Un clúster de Kubernetes también contiene **uno o más nodos maestros** que ejecutan el plano de control de Kubernetes.



El plano de control consta de diferentes procesos:

- un **servidor API** (proporciona JSON a través de API HTTP)
- **Scheduler** (programador) que selecciona los nodos para ejecutar contenedores
- **Controller Manager** (administrador de controladores) que ejecuta controladores
- y **etcd,** un almacén de configuración disponible globalmente

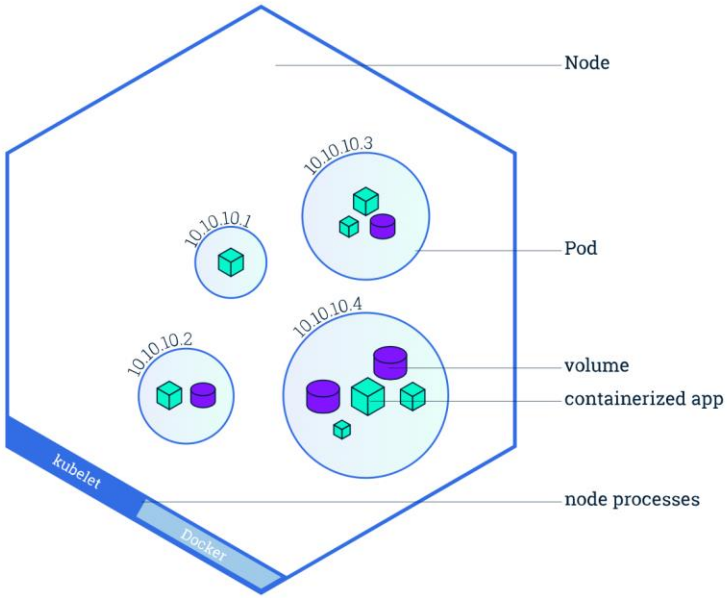
Elementos de Kubernetes

PODs

POD

Un pod es la unidad desplegable más pequeña que Kubernetes puede gestionar., un "host lógico".
Un pod es un grupo lógico de uno o más contenedores, comparten la dirección IP y rango puertos.
Los pods son efímeros (frente a fallos de ejecucion o de nodo) y no pueden moverse de servidor

Los pods siempre se ejecutan en nodos y es administrado por el Master. En un nodo puede haber varios pods.



La programación (scheduler) de Pods la realiza automáticamente el Master y esto tiene en cuenta los recursos disponibles en los Nodos.

Cada nodo de Kubernetes ejecuta al menos:

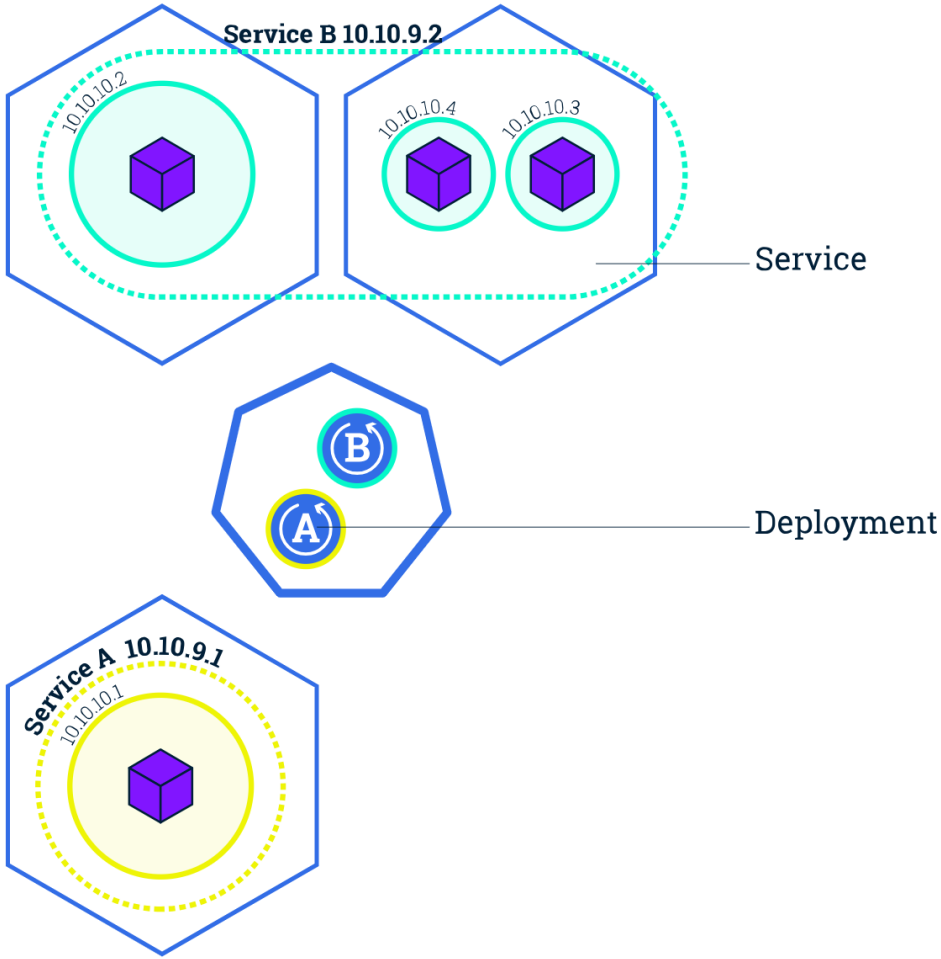
- Un run time de contenedor (como Docker, rkt) que se encargará de extraer todos sus contenedores de un registro.
- Kubelet, que actúa como puente entre el Kubernetes Master y los Nodos; gestiona los Pods y los contenedores que se ejecutan en una máquina.servidor

CESTE

SERVICIOS

SERVICIO

Un servicio es una abstracción que define un conjunto lógico de Pods y la política de acceso a los mismos. El conjunto de Pods involucrados en un servicio se determina mediante un selector de etiquetas



Tipos de servicios :

- **ClusterIP** expone los pods a conexiones desde el interior del cluster
- **NodePort** expone los pods al tráfico externo, reenviando el tráfico desde un puerto en cada nodo del clúster al puerto del contenedor
- **LoadBalancer**, también expone pods al tráfico externo, como lo hace el servicio NodePort, sin embargo, también proporciona un balanceo de carga

SERVICIOS

SERVICE DISCOVERY

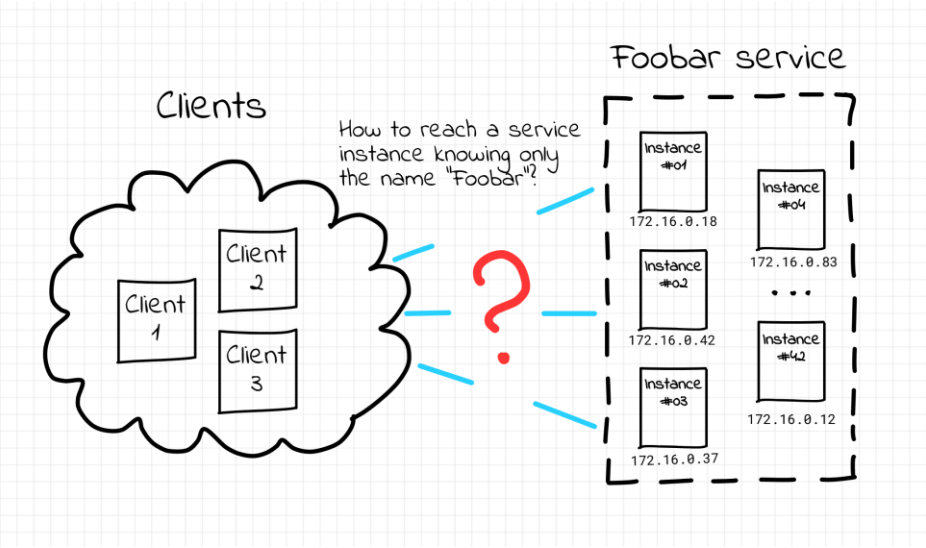
Permite a diferentes aplicaciones Kubernetes conectarse sin que necesiten conocer las direcciones IP. Un cliente que utiliza uno o más contenedores dentro de un pod no debería necesitar saber con qué pod específico funciona, especialmente si hay varios pods (réplicas).

Tipos de service discovery:

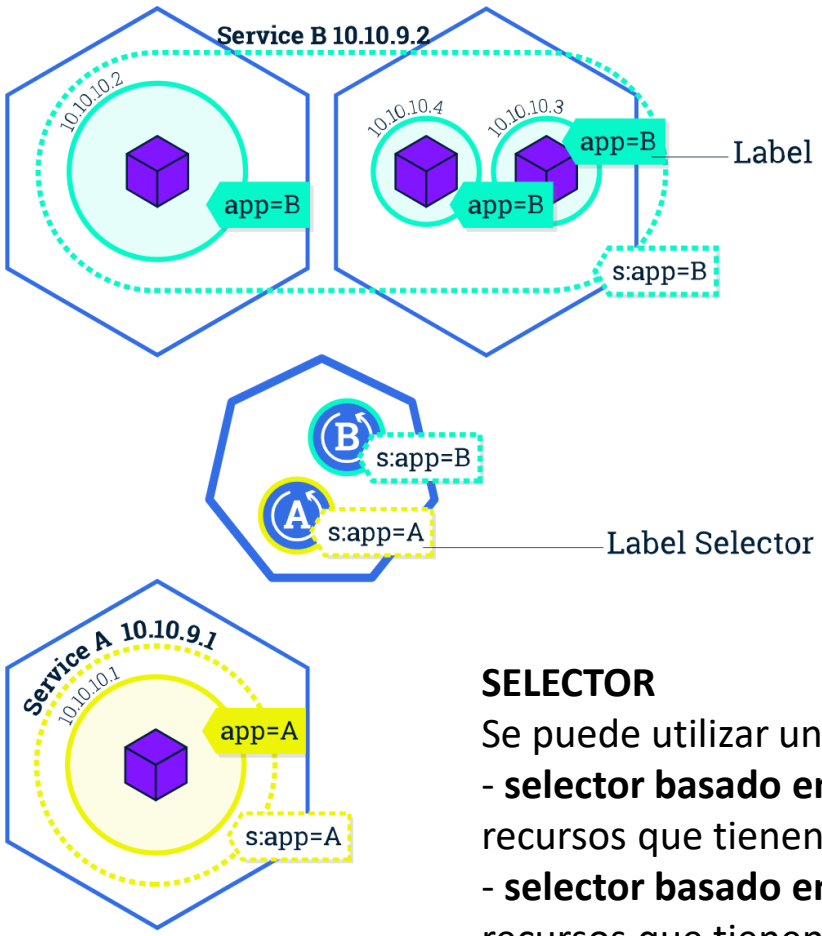
- **Variables de entorno** que se inyectan en el despliegue del pod que pueden usar las aplicaciones

```
MYSQL_SERVICE_HOST=10.0.150.150
MYSQL_SERVICE_PORT=3306
```

- **DNS** se configura a nivel de cluster y permite servicio de DNS para los pods



LABELs y SELECTORs



LABEL

Una etiqueta es un par clave/valor que se adjunta a recurso, por ejemplo, un pod. Etiquetas se puede adjuntar a los recursos en el momento de la creación, en cualquier momento posterior. Un recurso puede tener múltiples etiquetas

```
release: stable
environment: dev
```

```
environment = dev
environment != live
environment in (dev, test)
environment notin (live)
release = stable, environment = dev
```

SELECTOR

Se puede utilizar un selector de etiquetas para organizar los recursos:

- **selector basado en la igualdad**. Define una condición para seleccionar recursos que tienen el valor de etiqueta especificado.
- **selector basado en conjuntos** define una condición para seleccionar recursos que tienen una etiqueta con su valor dentro del conjunto de valores especificado.

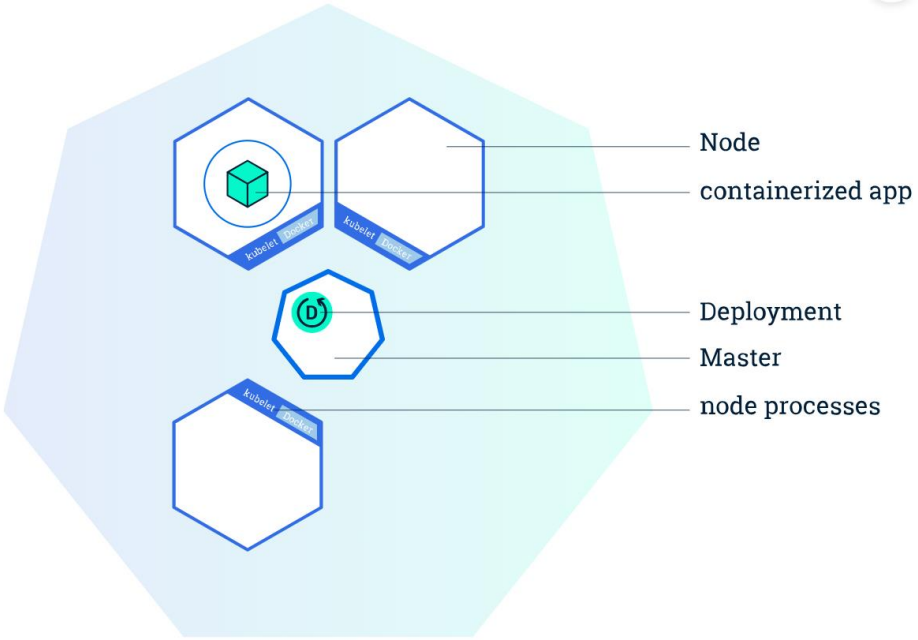
CESTE

GESTOR DE CONTROLADORES

CONTROLADOR

Un controlador gestiona un conjunto de pods y garantiza que el clúster está en un estado especificado.

A diferencia de pods creados manualmente, los pods mantenidos por una replicación se reemplazan automáticamente si falla, se eliminan o se cancelan.



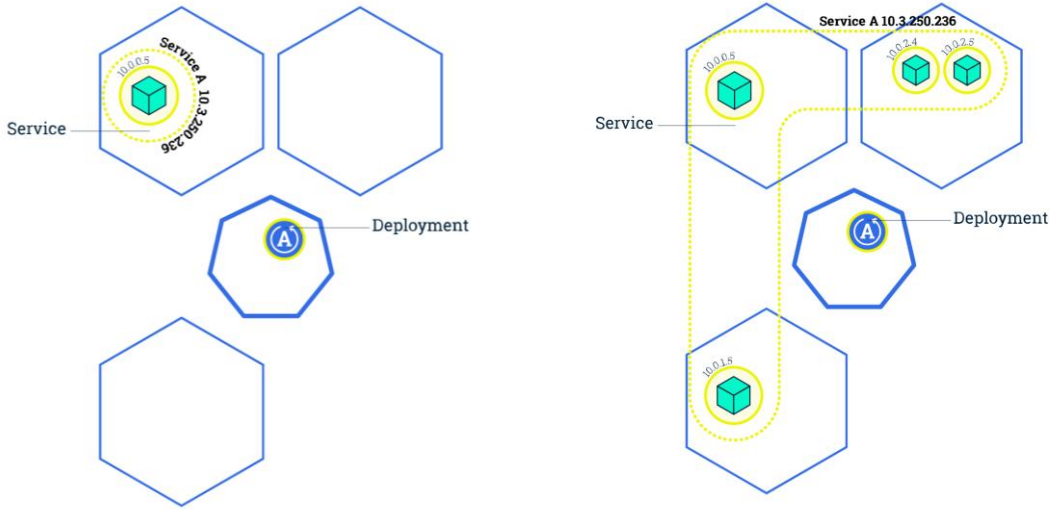
GESTOR DE CONTROLADORES

TIPOS DE CONTROLADORES

- **Despliegue o implemenatción, u**na implementación define un estado deseado para la lógica. grupo de pods y conjuntos de réplicas. Crea nuevos recursos o reemplaza los recursos existentes, si es necesario con despliegues incrementales

- **Replicas, u**n controlador de replicación es responsable de ejecutar el número especificado de copias de pods (réplicas) en todo el grupo. Un controlador de replicación solo admite selectores basados en igualdad

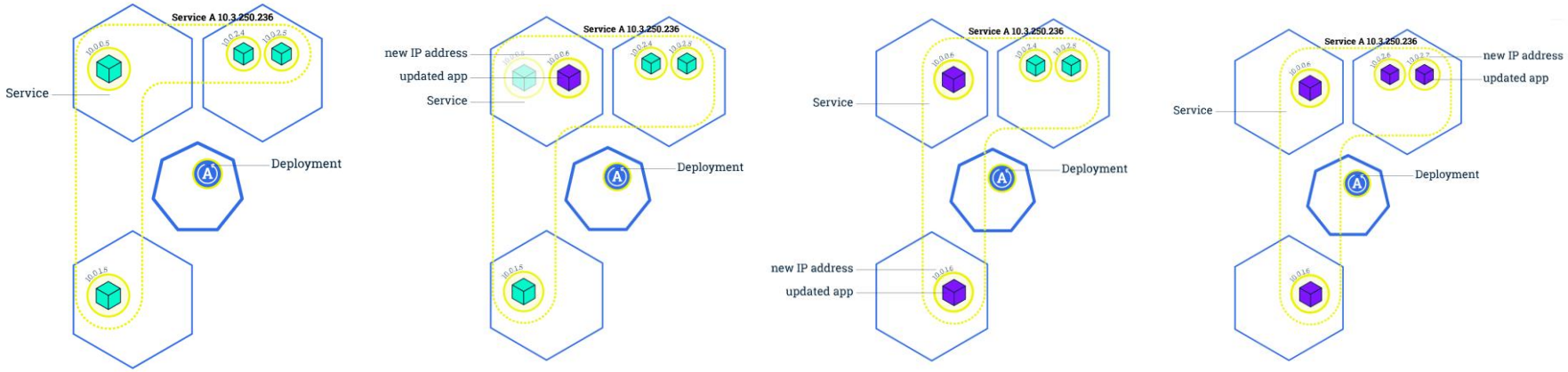
- **Replica sets**, este controlador es como el anterior pero también admite selectores basados en



GESTOR DE CONTROLADORES

TIPOS DE CONTROLADORES: Despliegues incrementales

una implementación se puede actualizar, implementar con roll-out o rollback. Las **actualizaciones continuas** permiten la actualización de las implementaciones sin tiempo de inactividad al actualizar incrementalmente las instancias de Pods con otras nuevas.



Las actualizaciones continuas permiten las siguientes acciones:

- Promocionar una aplicación de un entorno a otro (con actualizaciones de imágenes de contenedor)
- Retroceder a versiones anteriores
- Integración continua y entrega continua de aplicaciones sin tiempo de inactividad

GESTOR DE CONTROLADORES

Dos tipos de orquestación

CONTROL IMPERATIVO
Usar solo pods y controladores de replicación implementar una aplicación es, al menos en parte, una forma imperativa de gestionar el software, porque por lo general, requiere pasos manuales.

Implica varios pasos o tareas, algunas de las cuales son manuales. Cuando se trabaja en equipo, generalmente se requiere que estos pasos se documenten y, en un caso ideal, se automaticen. Esto puede no ser tarea trivial, incluso si cada uno de los pasos es simple.

CONTROL DECLARATIVO
Un enfoque declarativo para tareas administrativas es más apropiado para tareas no triviales.

En este caso un administrador define un estado objetivo para un sistema (aplicación, servidor o clúster). Normalmente en un lenguaje específico de dominio (DSL) se utiliza para describir el estado objetivo.

Una herramienta administrativa, como Kubernetes, toma esta definición como entrada y se encarga de cómo alcanzar el estado objetivo desde el estado observable actual.

Volúmenes de almacenamiento

Volumenes

VOLMEN

El volumen se define a nivel de pod y se utiliza para preservar datos en caso de fallos de contenedores y para utilizar un volumen para compartir datos entre contenedores en un pod.

Un volumen tiene el mismo ciclo de vida que el pod que lo contiene, cuando se elimina un pod, el volumen también se elimina. Kubernetes admite diferentes tipos de volúmenes, que son implementados como complementos.

VOLMEN PERSISTENTE

Un volumen persistente representa una unidad de almacenamiento de HW real en red en el clúster que ha sido aprovisionada por un administrador. El almacenamiento persistente tiene un ciclo de vida independiente de cualquier pod individual.

Admite diferentes modos de acceso y montaje como lectura-escritura por un solo nodo, montaje como solo lectura por muchos nodos y montaje como lectura-escritura por muchos nodos. Kubernetes admite diferentes tipos de volúmenes persistentes, que se implementan como Complementos

El acceso (binding) a volumen persistente se define con un **PERSISTENT VOLUME CLAIM** que define una cantidad específica de almacenamiento solicitado y modos de acceso específicos.

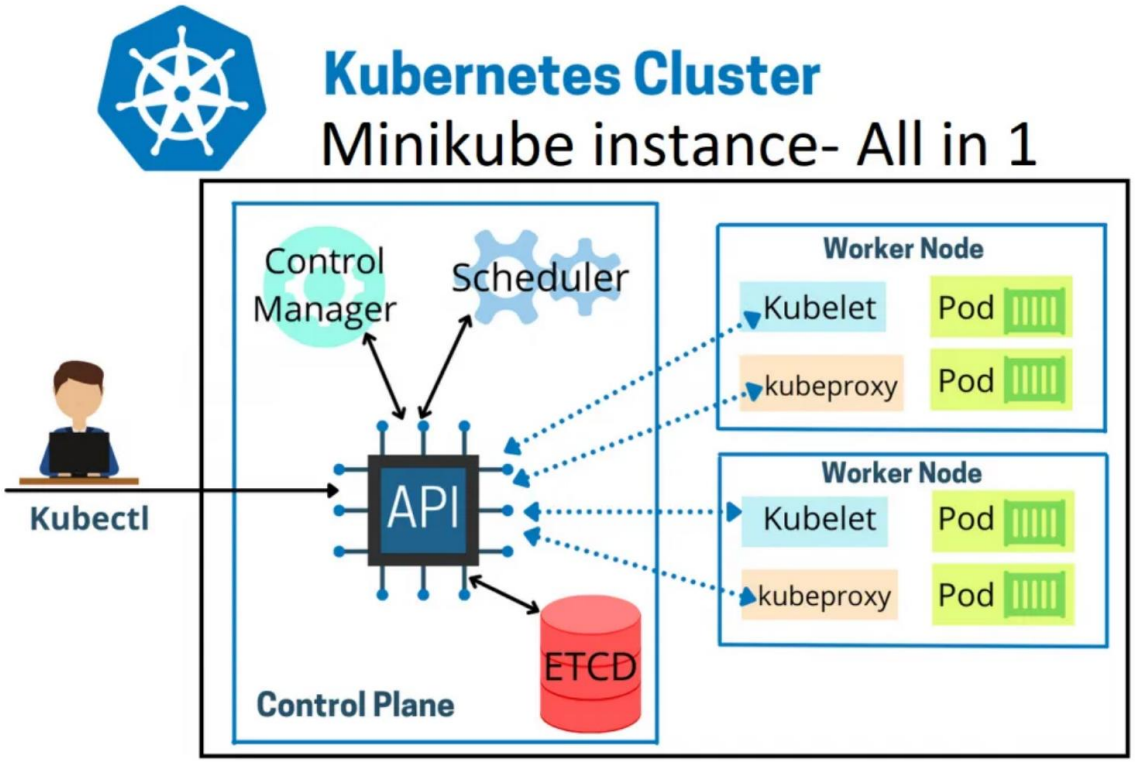
Prácticas

Practicas

Minikube

(https://minikube.sigs.k8s.io/docs/)

1. La forma recomendada de iniciar un clúster de Kubernetes con fines de desarrollo es mediante minikube .
2. Minikube crea una VM en la máquina local e implementa un clúster simple que contiene solo un nodo. Minikube está disponible para sistemas Linux, Mac OS y Windows.
3. La CLI de minikube proporciona operaciones básicas de arranque, como: iniciar, detener, estado y eliminar.
4. Usaremos una terminal en línea con minikube preinstalado.



Practicas

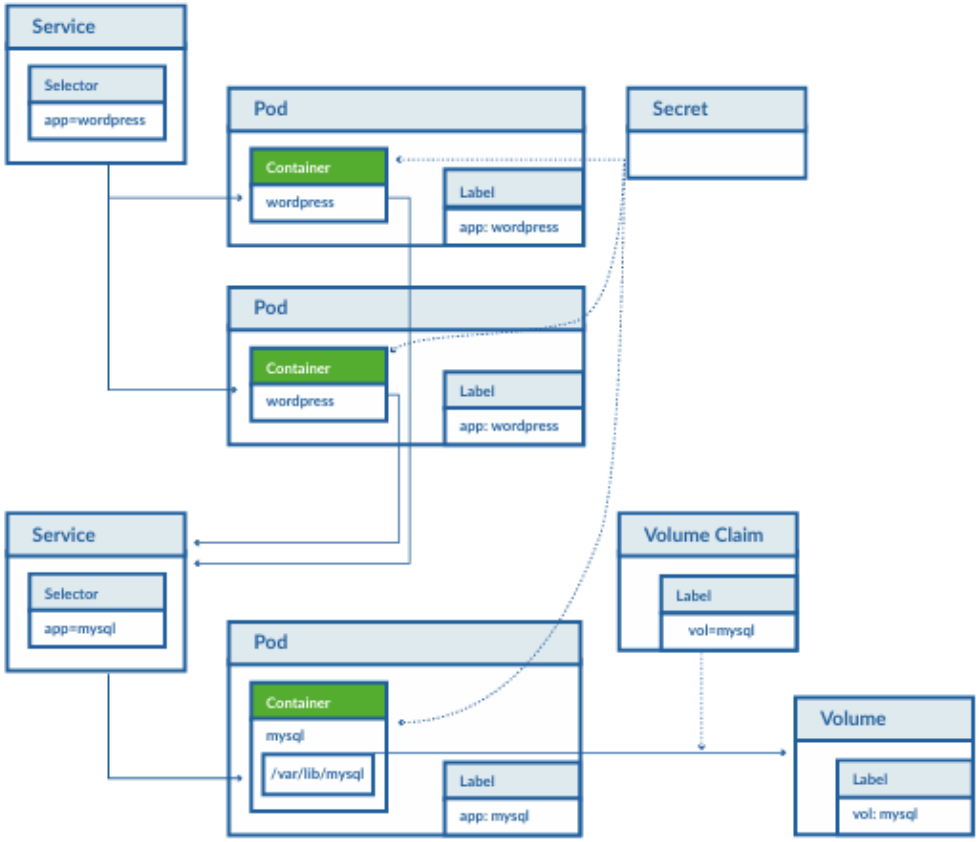
GitHub Repo
https://github.com/cpl-ceste/kubernetes_exercices

EJERICIO 1:

Desplegar una aplicación en Kubernetes
Explorar los componentes básicos

Practicas

EJERICIO 2:
Despleigar una WordPress con MySQL



CESTE

Knowledge Check

CESTE

www.ceste.es