

Docker

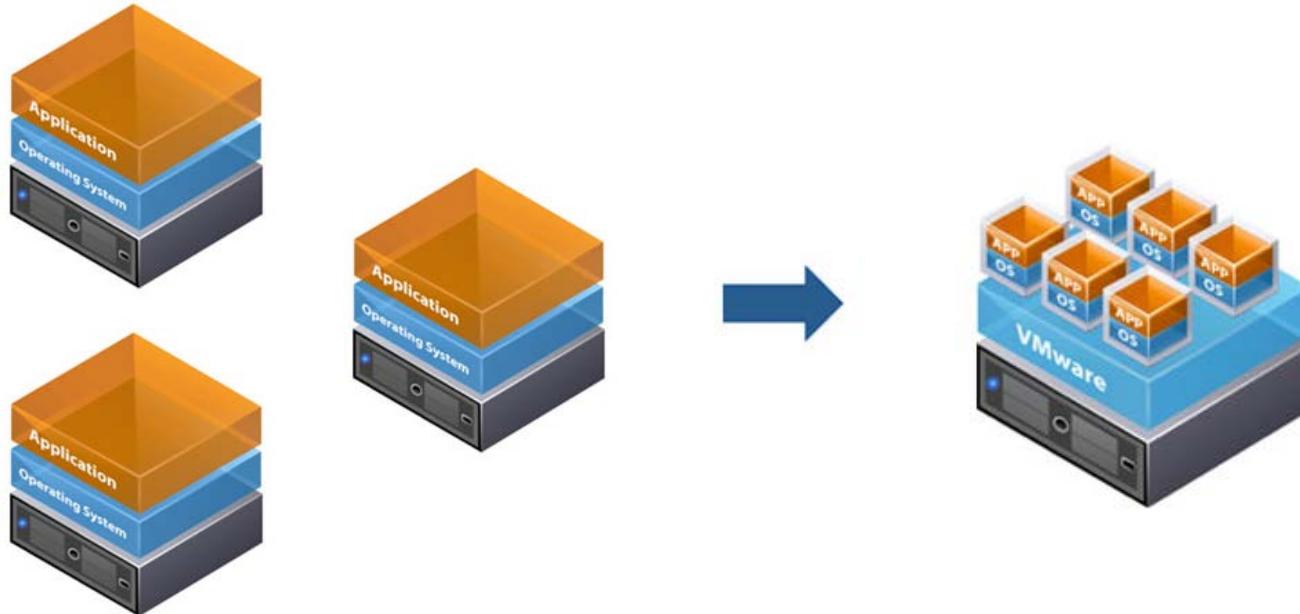
Fernando Agis // I&D

Introducción a la Virtualización con Contenedores

Noviembre 2018

CONATEL C

Virtualización



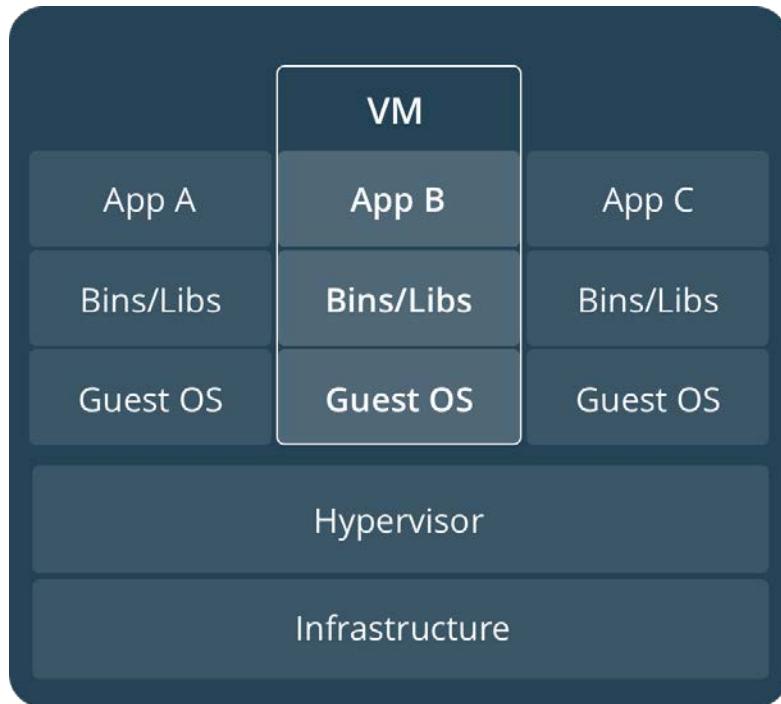
XenServer®
Open Source Virtualization



CITRIX®
ORACLE®
VM

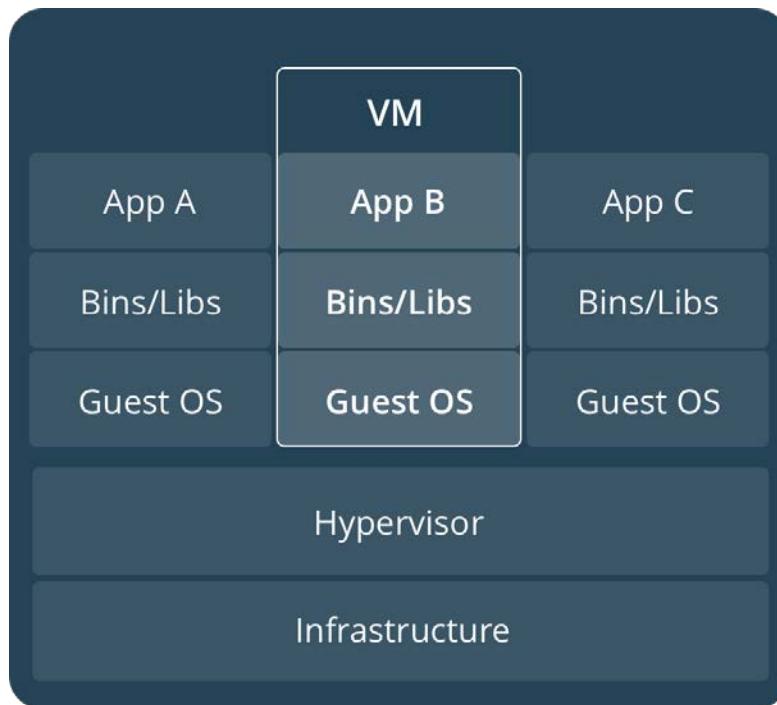
Arquitectura

Máquinas Virtuales



Arquitectura

Máquinas Virtuales

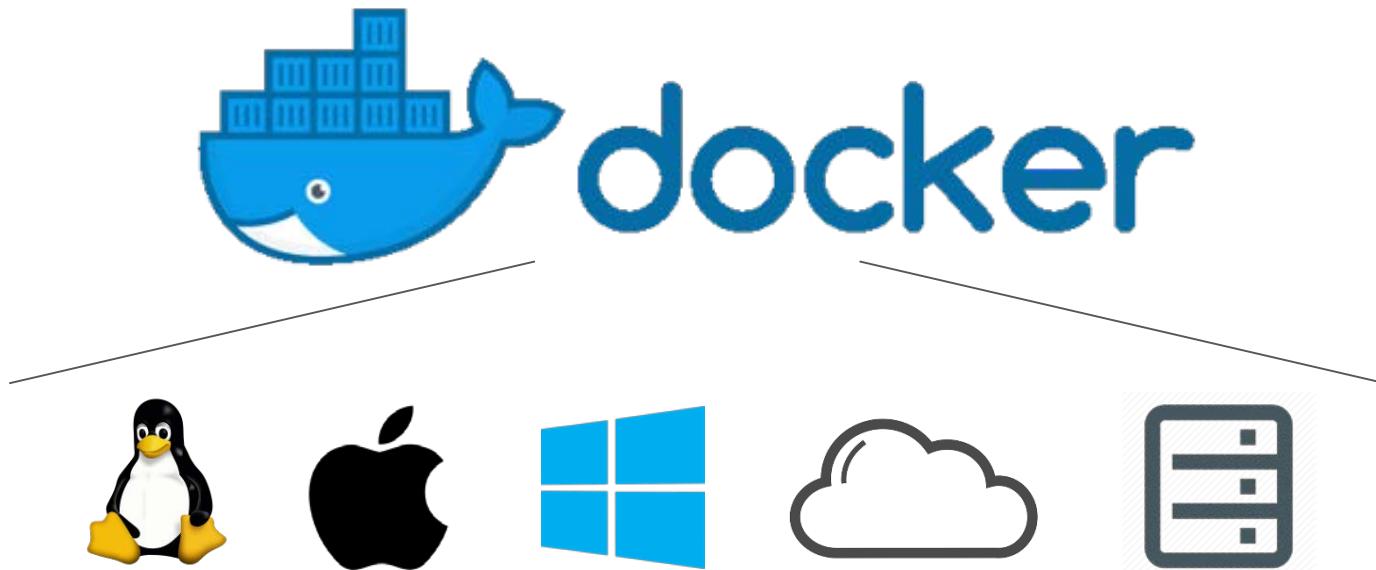


Contenedores



Qué es Docker?

- Implementación open-source de Linux Containers
- Es la implementación #1
- Repositorio de imágenes públicas: docker hub

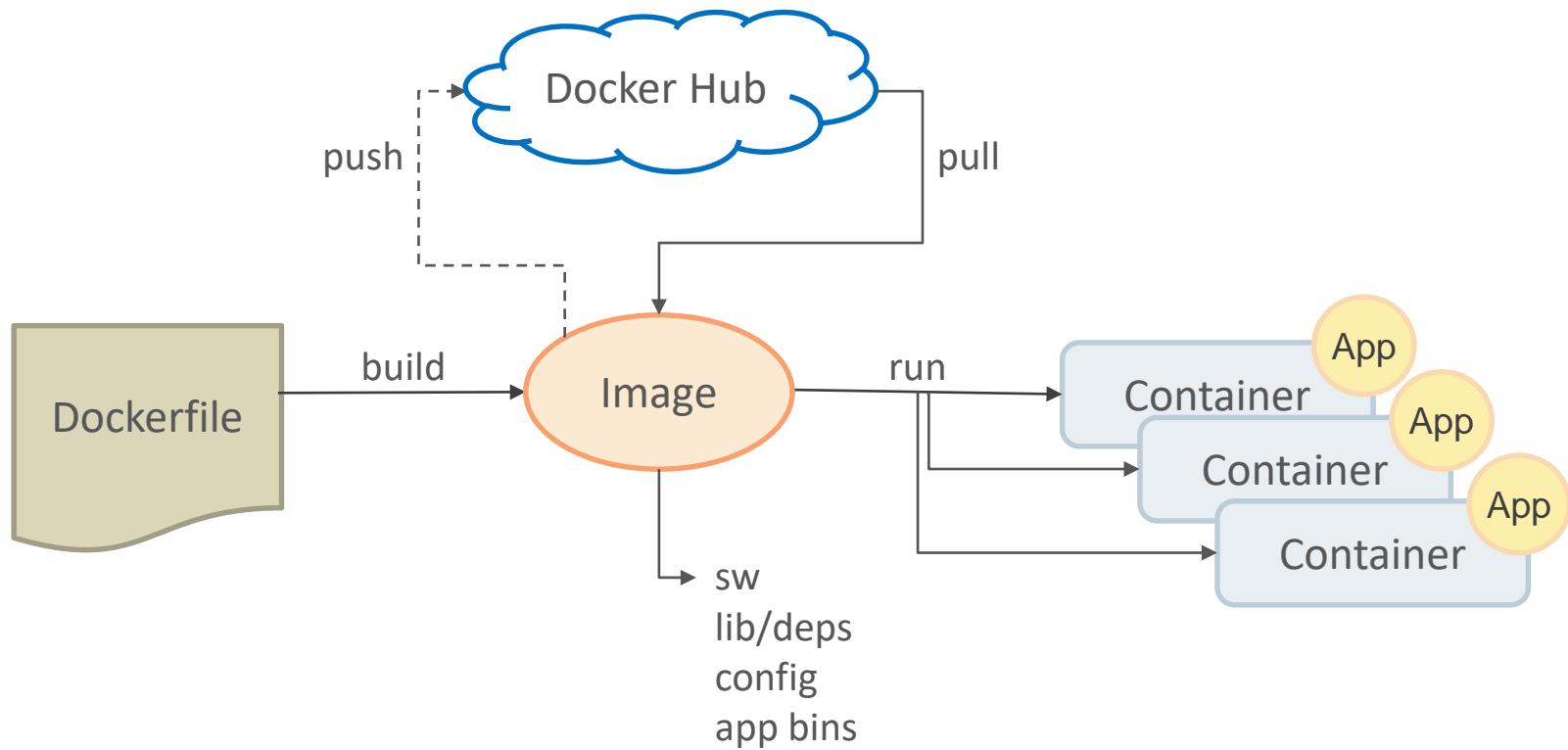


Cuales son las ventajas?

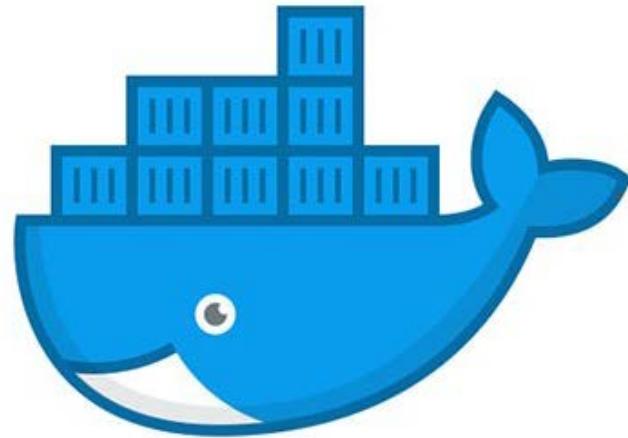
- No necesita Hypervisor
- Uso eficiente de recursos (CPU, Memoria)
- Menor consumo de disco (mucho menor)
- Levantan muy rápido (milisegundos)
- Ideales para Microservicios y DevOps
- Fácilmente automatizables
- Son muy fáciles de transportar y recrear (text file)
- Permiten tratar a la infraestructura como código



Cómo funciona?



Llevándolo a la práctica...

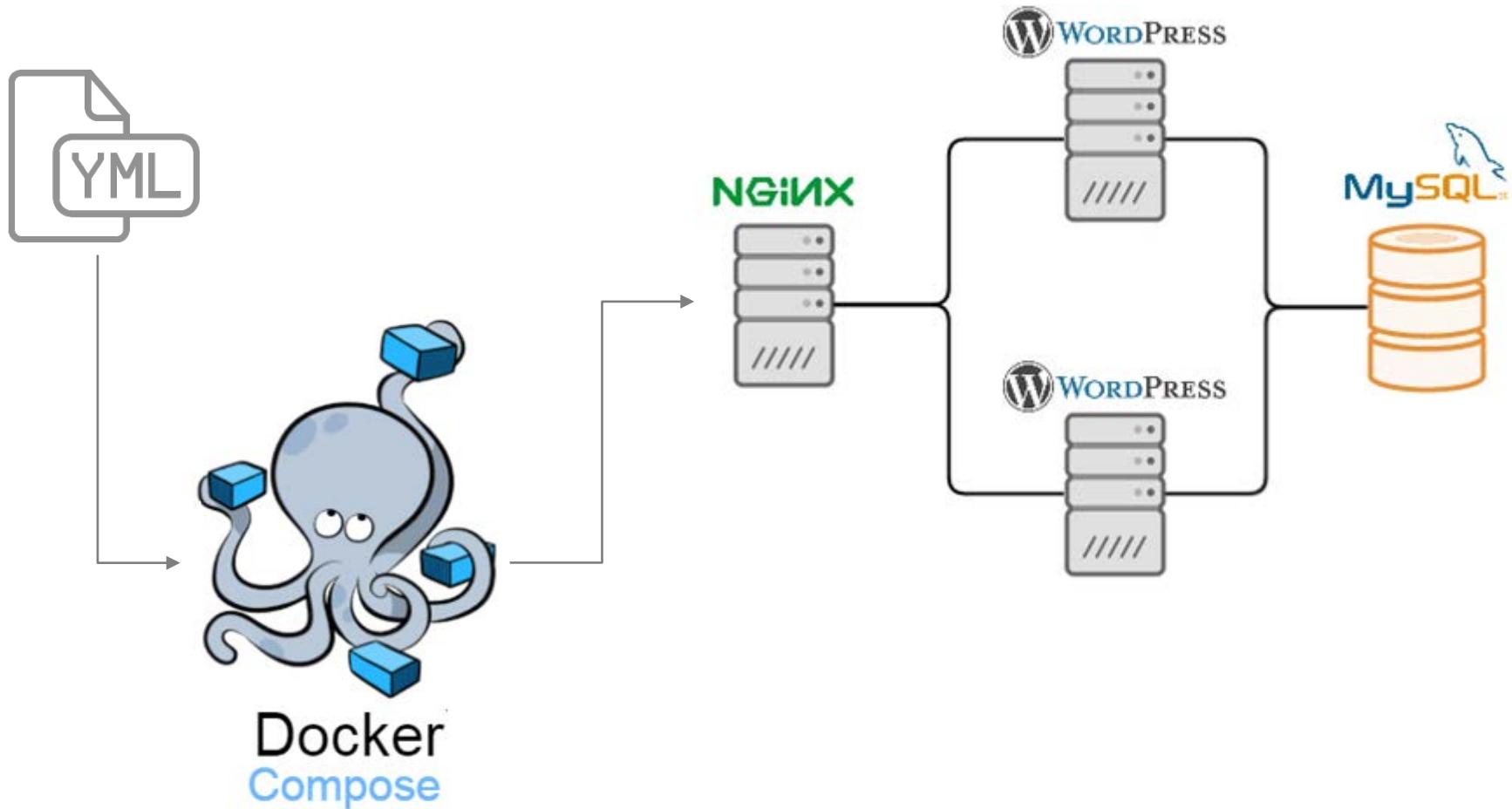


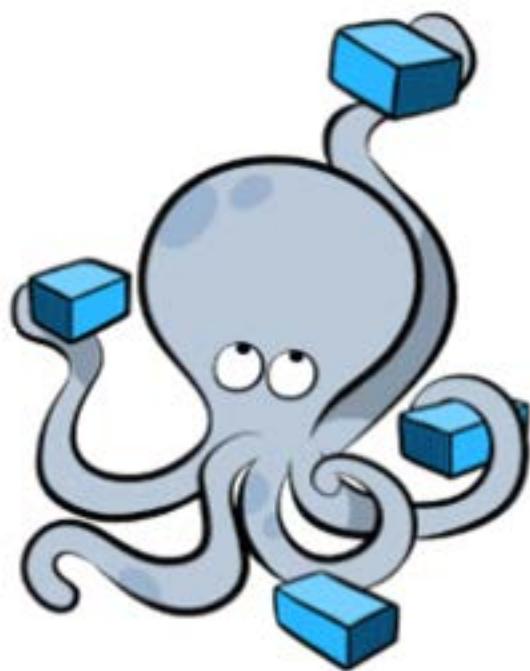
3 servidores Ubuntu en (menos) de 2 minutos

Llevándolo a la práctica...



A crear Contenedores...





Docker
Compose

```
fagis@ubuntu:~/dockers/wordpress-2$ █
```



Terminal #1 Terminal #2

Orquestando Contenedores

- Agrupamiento lógico (aplicación)
- Clustering de servidores
- Deploy automático
- Balanceo de carga
- Escalamiento automático
- Self-healing
- Rollouts y rollback
- Run Anywhere



kubernetes

Para qué se utiliza?

65%

use Docker to deliver development agility.

48%

use Docker to control app environments.

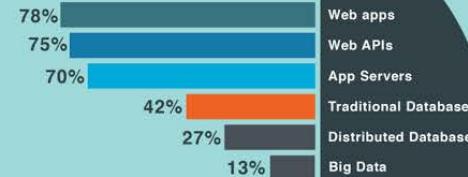
41%

use Docker to achieve app portability.

90%

use Docker for apps in development.

Docker Workloads



58%

use Docker for apps in production.



90%

plan dev environments around Docker.



80%

plan DevOps around Docker.



Que pasó en 5 años?



14M

Docker
Hosts

900K

Docker
apps

77K%

Growth in Docker
job listings

12B

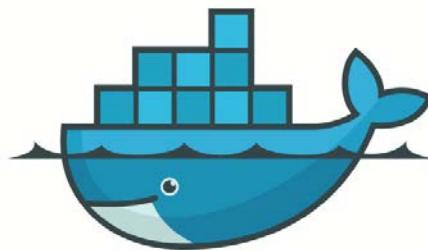
Image pulls
Over 390K%
Growth

3300

Project
Contributors



Plataforma como Servicio



docker



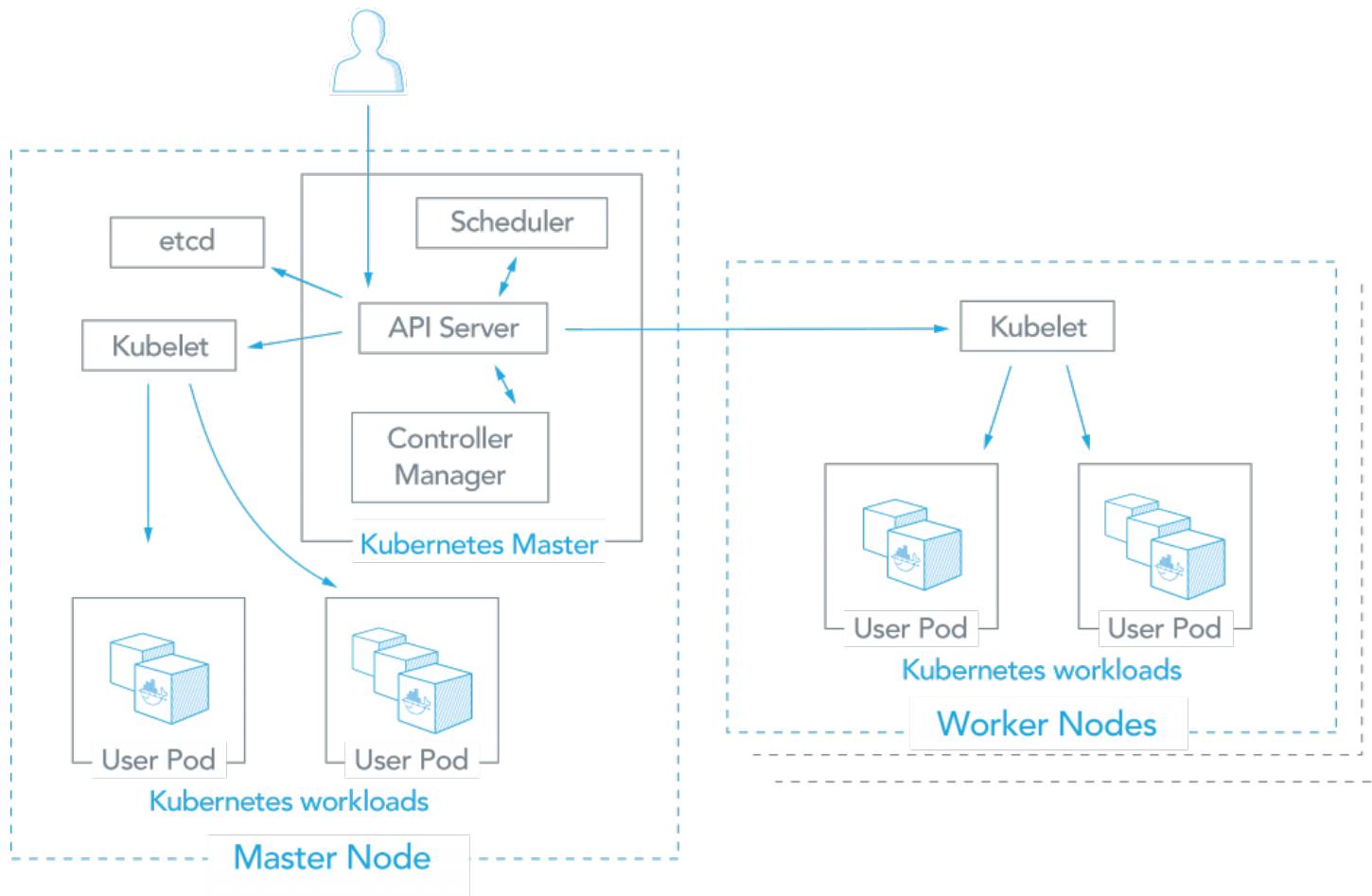
kubernetes



OPENSIFT

Y luego...?

Kubernetes



API Server: management hub for Kubernetes
 Scheduler: places a workload on the appropriate Node
 Controller Manager: scales workloads up/down
 etcd: stores configuration data which can be accessed by API Server

Kubelet: Receives pod specifications from API Server, updates Nodes
 Master Node: places workloads on Nodes
 Worker Nodes: receives requests from Master Nodes and dispatches them
 User Pod: a group of containers with shared resources

Docker Swarm

