

Empaquetado de aplicaciones Java con Docker y Kubernetes

Víctor Orozco - @tuxtor

23 de febrero de 2021

Academik



ACADEMIK

Monólito

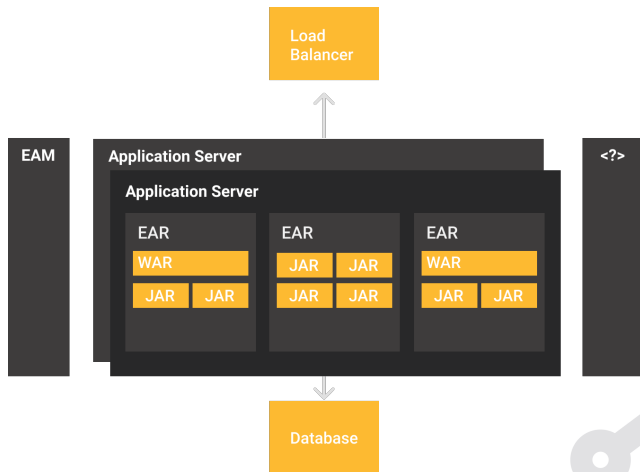
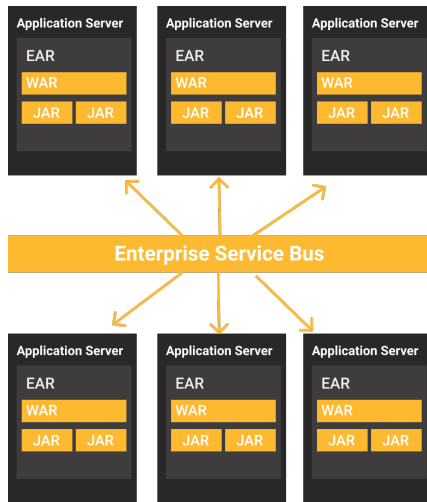


Figura 1: Monólito



Microservicios

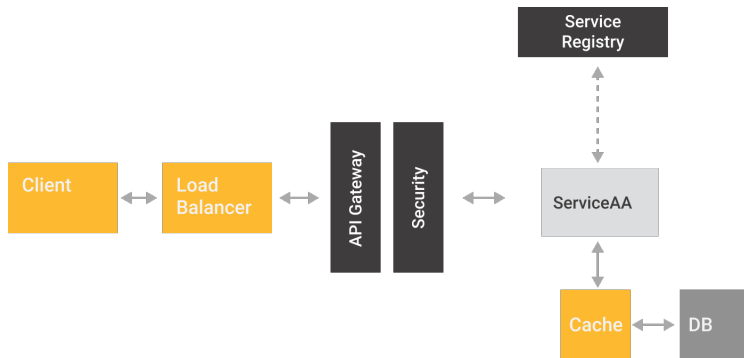


Figura 3: Microservicios

Monolito vs. Microservicios



**JUST
CALL
ME
AN
OPEN
SOURCE
COWBOY**

40 WORDS & 5 MATRICES



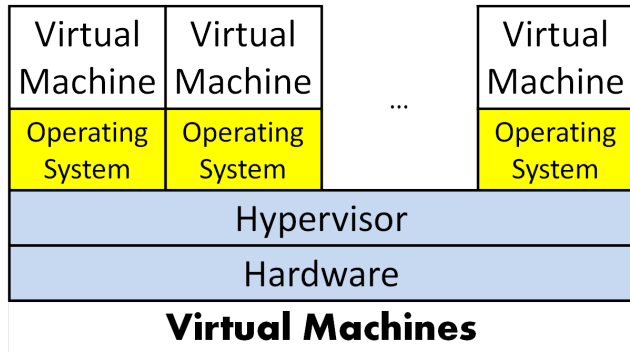
ACADEMIK

Introducción a Docker



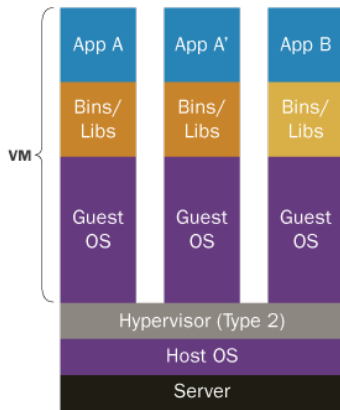


Hipervisores

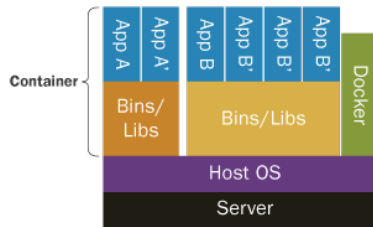


Despliegue contenedores

Containers vs. VMs

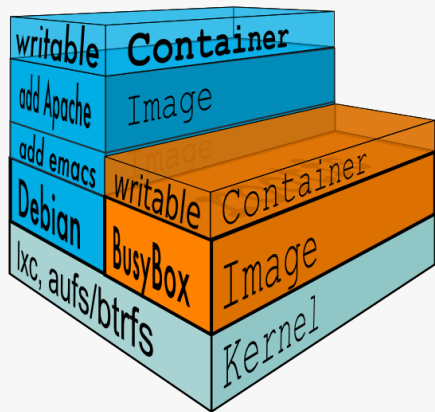


Containers are isolated, but share OS and, where appropriate, bins/libraries



Contenedor

Contenedor = bibliotecas + app + shell



Contenedor

Contenedor = bibliotecas + app + shell

Ideas generales:

- Distribución
- Volatilidad
- Automatización

- Cgroups + Namespace: Isolamiento de recursos de un grupo y visibilidad entre procesos
- libcontainer (LXC/Libvirt/systemd-nspawn)
- SELinux, AppArmor, Netfilter
- Ventajas: Boot time, menos overhead
- Desventajas: Deprecated en Kubernetes, Linux

Demo 1

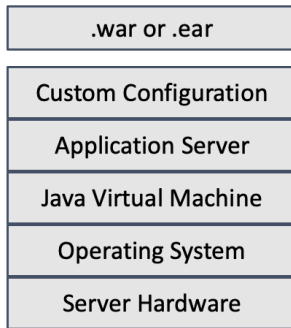
- Imagen base (ubuntu)
- Ejecución
- Agregar paquete
- Commit
- Ejecución

Demo 1

```
1 docker pull ubuntu
2 docker run ubuntu echo "Hola ubuntu"
3 docker run -it ubuntu /bin/bash
4 apt-get update && apt-get install nginx
5 docker ps
6 docker commit -id- tuxtor/nginx
7 docker run -d -p 81:80 tuxtor/nginx
```

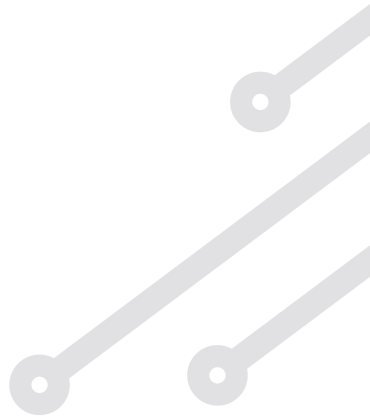
Aplicaciones Java

Java tradicional



Formas de despliegue

- War sobre un contenedor o app server
- Microservicio como UberJar o FatJar
- Microservicio en Docker



Características

- Todos los app servers tienen imágenes listas
- Configuración debe ser externalizada
- Archivos descriptores (en imagen)
- Archivos de configuración de dominio (en imagen)
- Archivos properties (en war)



Demo 1

App Server completo

Microservicio como UberJar/FatJar

- El jar no requiere entorno para ejecutarse, solo JVM
- Configuración debe ser externalizada
- Archivos de configuración de dominio (en imagen)
- **Archivos descriptores** (en war)
- **Configuración mediante archivos properties** (en war)



Microservicio como FatJar + Docker + Config + Payara Micro

- ThinJar
- ThinJar - Mi código
- Container - Dependencias/runtime
- Archivos de configuración de dominio (en imagen)
- **Archivos descriptores** (en war)
- **Configuración mediante archivos properties** (en war)



Microservicio como ThinJar + Docker + Payara Micro

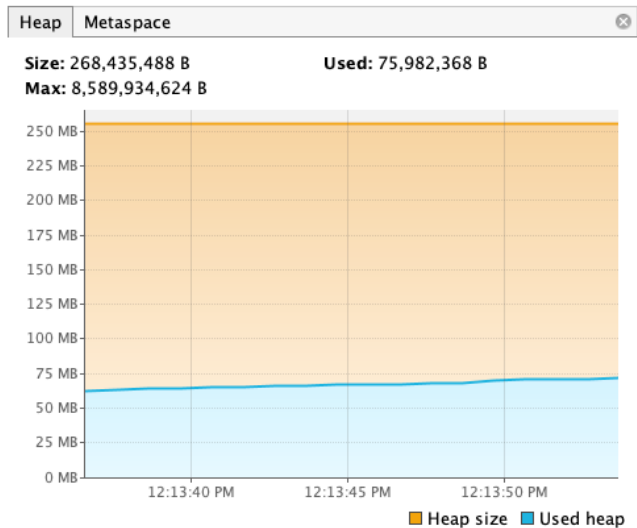
Límites de memoria

```
1. tuxtor@millenium-falcon-2: ~ (zsh)
➔ ~ docker run -it -m=100M --memory-swap=100M ubuntu free -h
```

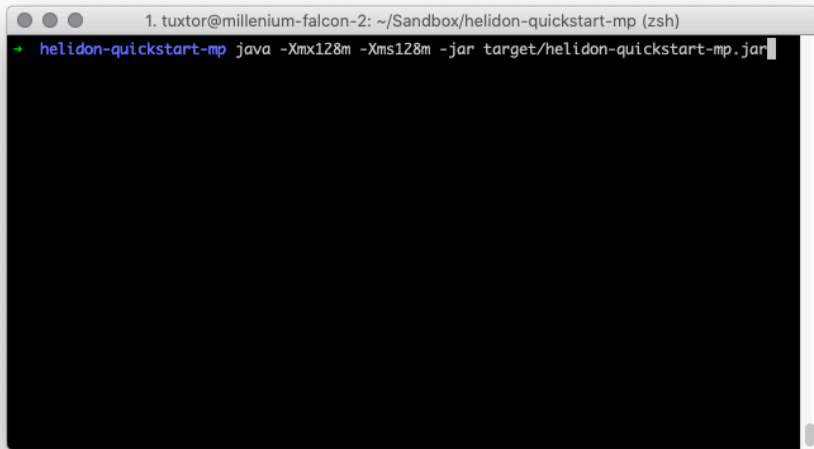
	total	used	free	shared	buff/cache	available
Mem:	1.9G	289M	1.1G	744K	530M	1.5G
Swap:	1.0G	0B	1.0G			

```
➔ ~
```

Límites de memoria - JVM

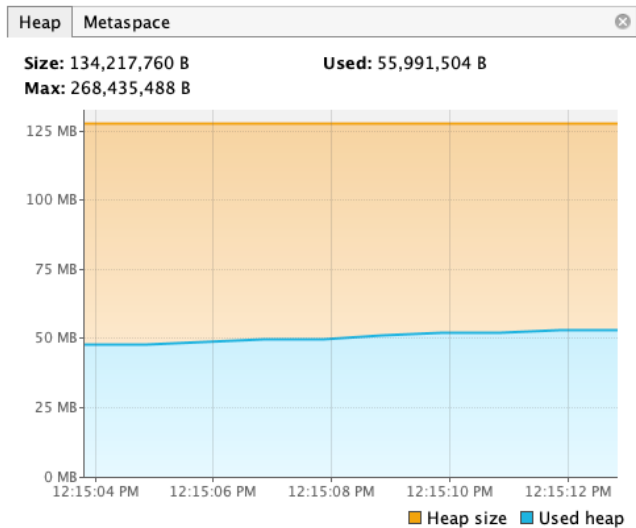


Límites de memoria - JVM

A terminal window with a title bar showing the user '1. tuxtor@millenium-falcon-2' and the directory '~/Sandbox/helidon-quickstart-mp' in 'zsh' shell. The command 'helidon-quickstart-mp java -Xmx128m -Xms128m -jar target/helidon-quickstart-mp.jar' is entered. The terminal has a black background and a white cursor at the end of the command line.

```
1. tuxtor@millenium-falcon-2: ~/Sandbox/helidon-quickstart-mp (zsh)
→ helidon-quickstart-mp java -Xmx128m -Xms128m -jar target/helidon-quickstart-mp.jar
```

Límites de memoria - JVM



Límites de memoria

```
1. tuxtor@millenium-falcon-2: ~ (zsh)
➔ ~ docker run -it -m=100M --memory-swap=100M ubuntu free -h
```

	total	used	free	shared	buff/cache	available
Mem:	1.9G	289M	1.1G	744K	530M	1.5G
Swap:	1.0G	0B	1.0G			

```
➔ ~
```


Java Options

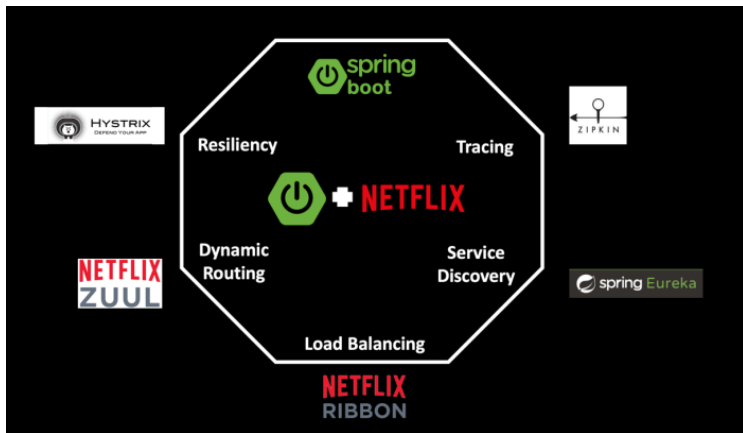
1 | `CMD java -XX:+PrintFlagsFinal -XX:+PrintGCDetails
$JAVA_OPTIONS -jar java-container.jar`



ACADEMIK

APIs vs Service Mesh







- REST API
- Configuration
- Discovery
- REST Client
- Monitoring
- Authentication
- Logging
- Resilience



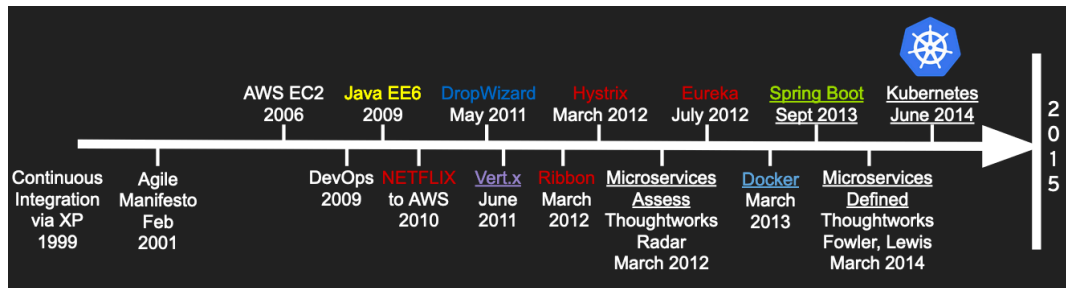
- REST API - Spring Rest
- Configuration - Spring Config
- Discovery - Eureka, Zuul
- REST Client - Spring RestTemplate
- Monitoring y tracing - Zipkin
- Authentication - Spring security
- Logging - Kibana
- Resilience - Hystrix

- REST API - JAX-RS
- Configuration - MicroProfile Config
- Discovery - Ad-hoc/Zookeeper
- REST Client - MicroProfile Client
- Monitoring y tracing - MicroProfile Healthcheck
- Authentication - MicroProfile JWT
- Logging - Kibana
- Resilience - MicroProfile Fault Tolerance

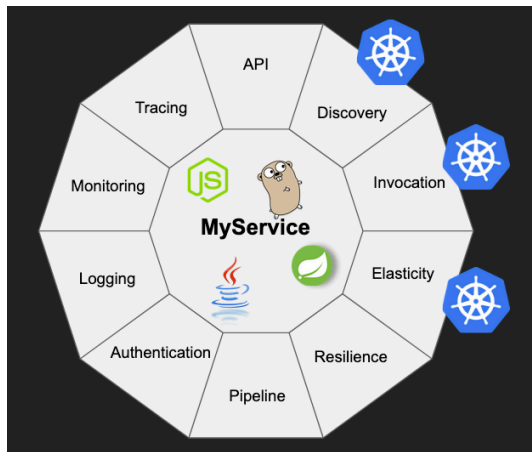
El camino a Kubernetes



El camino a Kubernetes



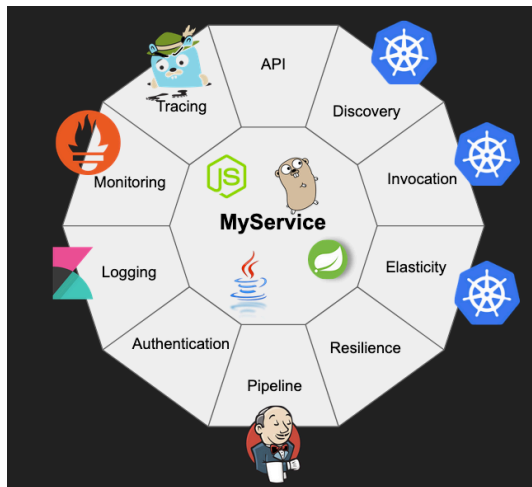
Créditos: Rafael Benevides



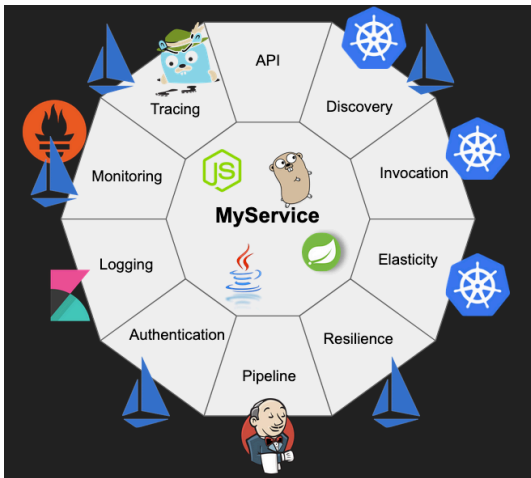
Créditos: Rafael Benevides

@tuxtor

Kubernetes



Kubernetes





@tuxtor

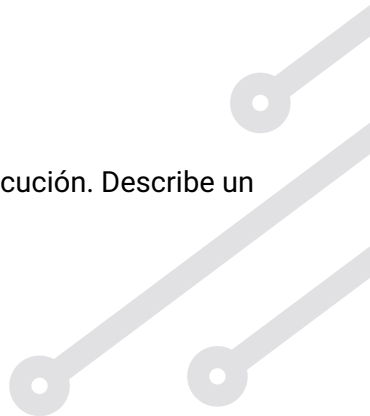
¿Que es un pod?

- Uno o más contenedores
- IP Compartida
- Caen todos o ninguno (ciclo de vida)
- Storage compartido
- Recursos compartidos



¿Que es un deployment?

Descriptor que mantiene el número de PODs replicas en ejecución. Describe un estado



¿Que es un servicio?

Agrupación de PODs, -e.g. una IP estable virtual y un nombre de DNS si se requiere-



- vorozco@nabenik.com
- @tuxtor
- <http://voroeco.com>
- <http://tuxtor.shekalug.org>



This work is licensed under
Creative Commons Attribution-
NonCommercial-ShareAlike 3.0
Guatemala (CC BY-NC-SA 3.0 GT).



ACADEMIK

Escríbenos a cursos@academik.io

www.academik.io

(CC BY-NC-SA3.0 GT)