DevSecOps

Microservicios
Workshop contenedores
Julio 2019
Daniel Andrés Penagos – Arquitecto Consultor de Aplicaciones
Departamento de Arquitectura e Innovación Tecnológica

AGENDA

Contexto de negocio
Que es la tecnología contenedores?
Workshop

AGENDA

Contexto de negocio

Que es la tecnología contenedores?

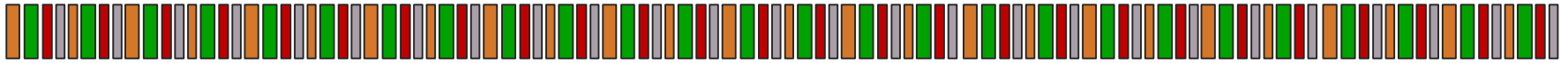
Workshop

Contenedores - Contexto

Waterfall

Agile

DevOps



Idea tomada de: https://image-store.slidesharecdn.com/60384d6c-3998-43ca-8637-c638a308e85d-large.jpeg

Contenedores - Contexto

DASA DEVOPS AGILE SKILLS ASSOCIATION

Cultura

-Derrumba silos
-Abierta
-Detiene el señalamiento de culpables

Procesos

-Agile Dev (TDD, BDD)
-Basado en scrum
-Lean
-Producto mínimo viable

Organización

-Business System Teams
-Platform Teams
-Equipos autónomos
-Gobierno

Automatización

-Construcción, Pruebas, Despliegue
-CI, CD
-Aprovisionamiento
-Automatización de DataCenter

Métrica y Mejora continua

-Monitoreo a todo nivel (negocio, apps, datos, teleco, etc)
-Feedback constante y para todos

Fuente: http://devopscolombia.com/courses/dasa/

Contenedores - Contexto

DASA DEVOPS AGILE SKILLS ASSOCIATION

Cultura

-Derrumba silos
-Abierta
-Detiene el señalamiento de culpables

Procesos

-Agile Dev (TDD, BDD)
-Basado en scrum
-Lean
-Producto mínimo viable

Organización

-Business System Teams
-Platform Teams
-Equipos autónomos
-Gobierno

Automatización

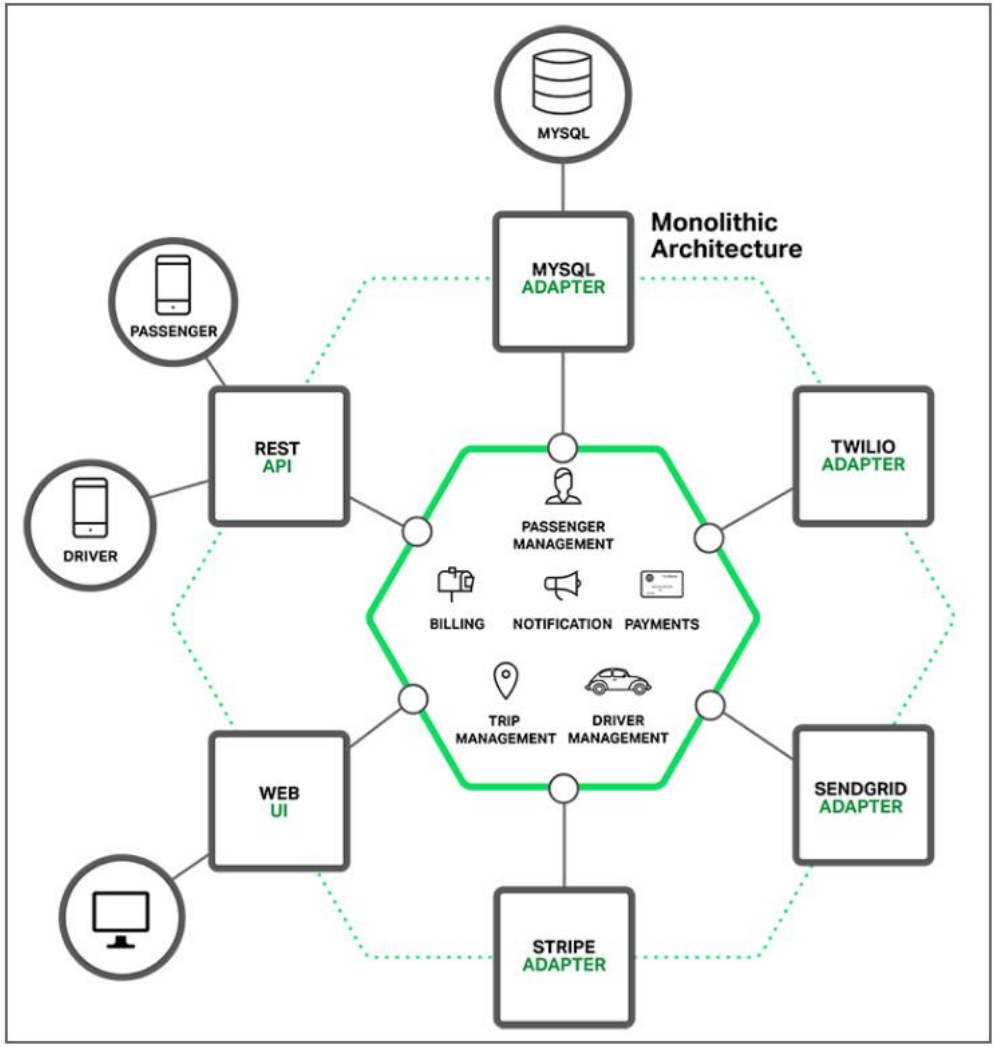
-Construcción, Pruebas, Despliegue
-CI, CD
-Aprovisionamiento
-Automatización de DataCenter

Métrica y Mejora continua

-Monitoreo a todo nivel (negocio, apps, datos, teleco, etc)
-Feedback constante y para todos

Fuente: http://devopscolombia.com/courses/dasa/

Contenedores - Contexto

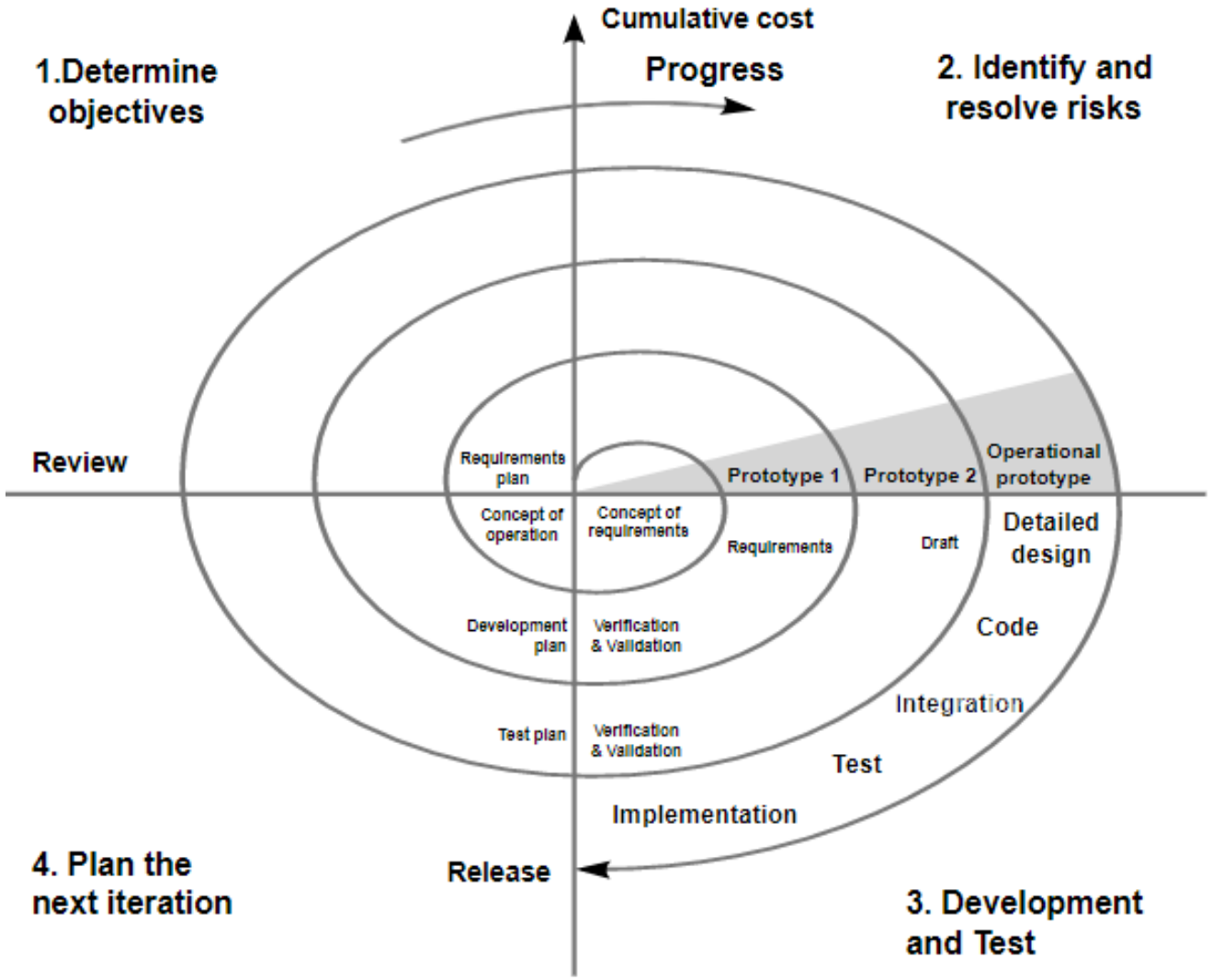


Arquitectura tradicional:

- Core: Lógica de negocio
- Componentes o módulos
- Adaptadores (con el exterior)
- Empaquetado: Monolito

Fuente: https://www.nginx.com/blog/introduction-to-microservices

Contenedores - Contexto



Arquitectura tradicional:

- + LOC
- + Complejidad
- + Pruebas
- + Dificultad para evolucionar
- + Mas lentos los despliegues

Fuente: https://en.wikipedia.org/wiki/Spiral_model

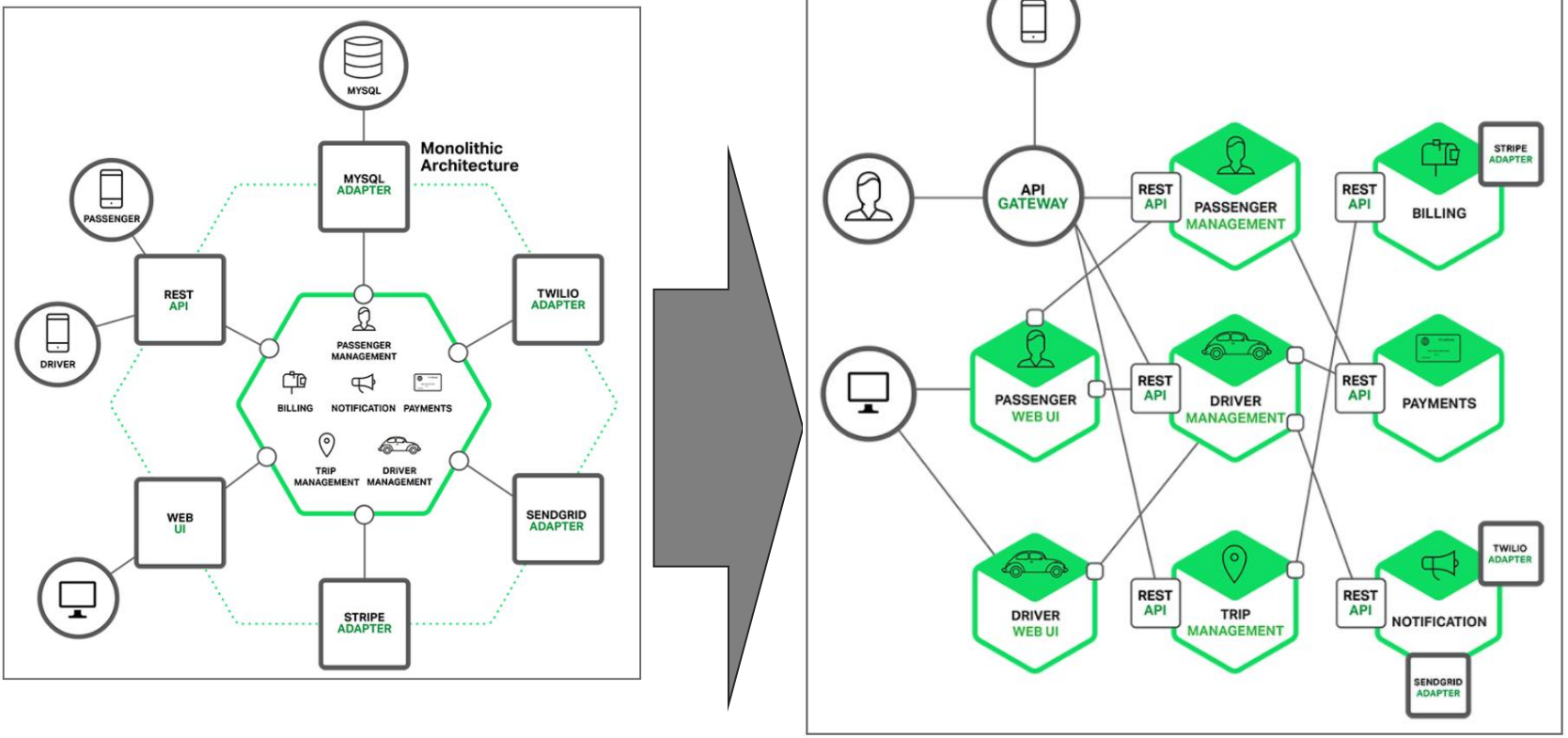
Contenedores - Contexto

Arquitectura tradicional:

Mas oportunidades de mejora…..

- Escalabilidad por funcionalidad
- Estabilidad: Un mismo PID
- Adopción nuevas tecnologías

Contenedores - Contexto

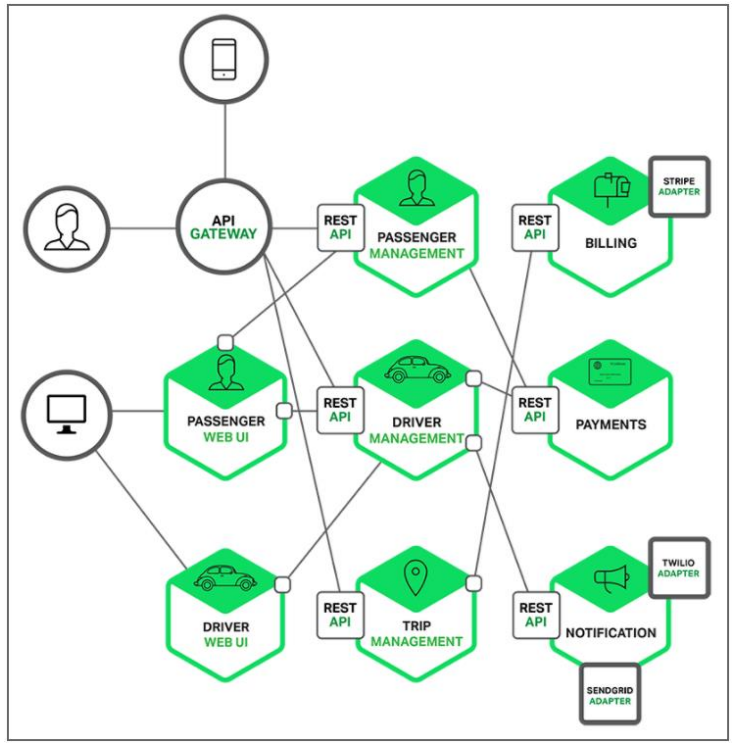


Dividir por funcionalidad

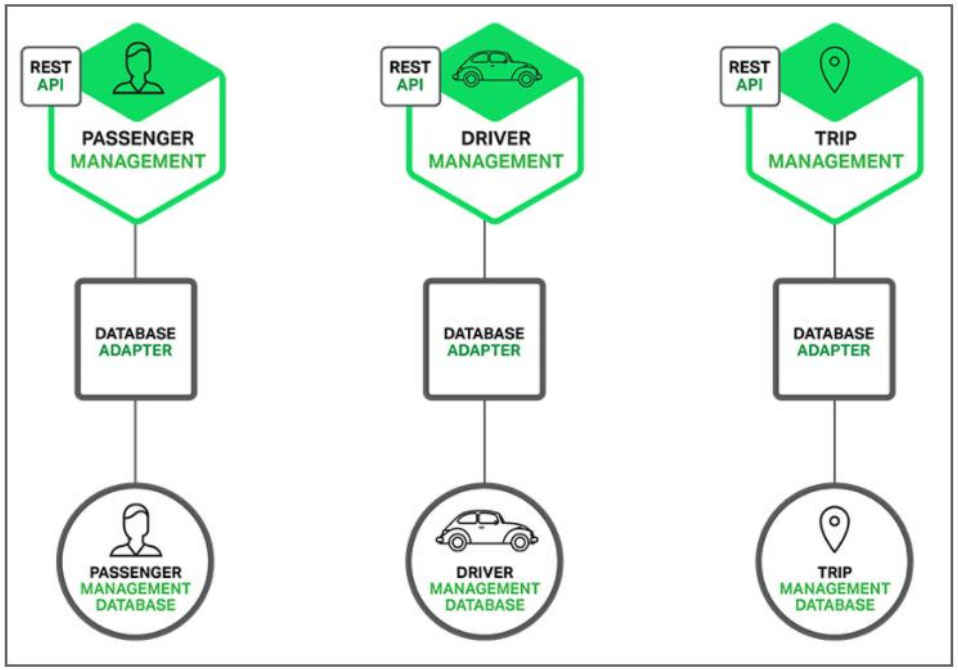
- Alta cohesión
- Bajo acoplamiento

Fuente: https://www.nginx.com/blog/introduction-to-microservices

Contenedores - Contexto



El patrón de arquitectura de microservicios propone independencia, incluso en sus datos!!!



Fuente: https://www.nginx.com/blog/introduction-to-microservices

Contenedores - Contexto

Diferencias con SOA:

- Se rechazan esquemas canónicos
- Utilización de servicios Rest, en lugar de Ws*
- No se usa ESB. Cada microservicio implementa lo que requiere

Ventajas:

- Mantenibilidad
- Mayor facilidad de evolución tecnológica
- Despliegue continuo (en producción)
- Escalabilidad por servicio (esto es muy útil cuando se manejan instancias en la nube)

Contenedores - Contexto

Desventajas:

- Definición del tamaño del microservicio
- Complejidad de tener un sistema distribuido (comunicación entre servicios, fallos de comunicación, etc)
- Manejar la base de datos particionada es un reto. (consistencia)
- El despliegue de microservicios requiere mayor nivel de automatización (ansible, puppet, chef) , o usando mecanismos de virtualización de contenedores (docker)

AGENDA

Contexto de negocio

Que es la tecnología contenedores?

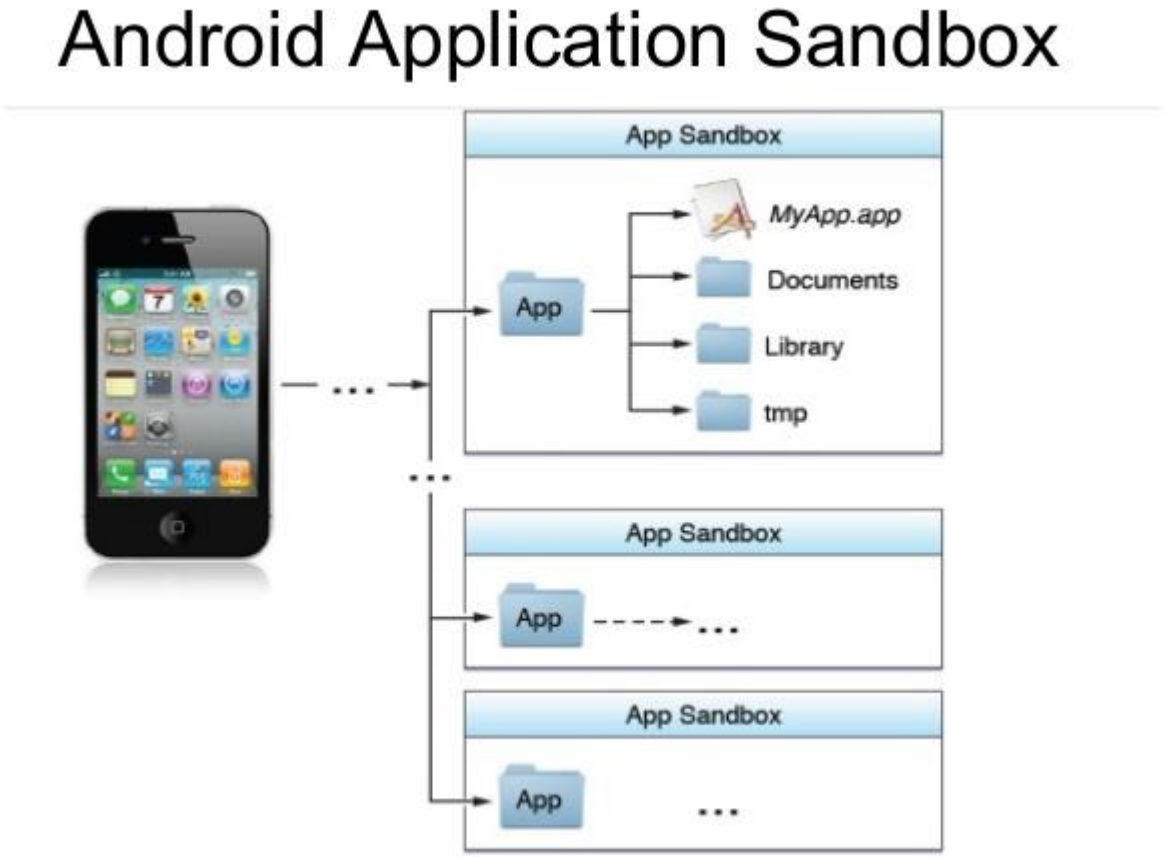
Workshop

Contenedores - Contexto

Sandbox

Fuente de la imagen: https://commons.wikimedia.org/wiki/File:Sandbox-2013.jpg

Contenedores - Contexto



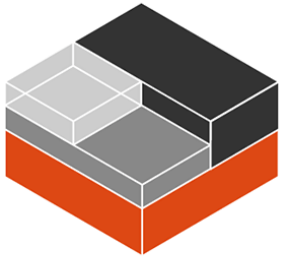
Sandbox

Fuente de la imagen: https://www.slideshare.net/ravishankarkumar963/analysis-and-research-of-system-security-based-on-android

Contenedores - Contexto

2000: FreeBSD Jail –subsistemas
+filesystem, users, network

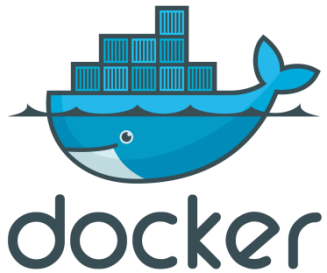
2000: Vserver
“Virtual Private Server”
Uso de “Security Contexts”



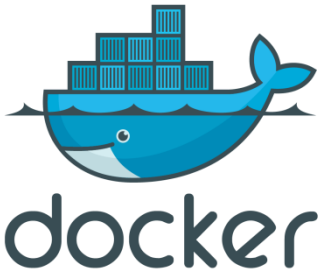
LXC

2008: Linux Containers
“Linux Virtual Environments”
Uso de “cgroups” by google-2006

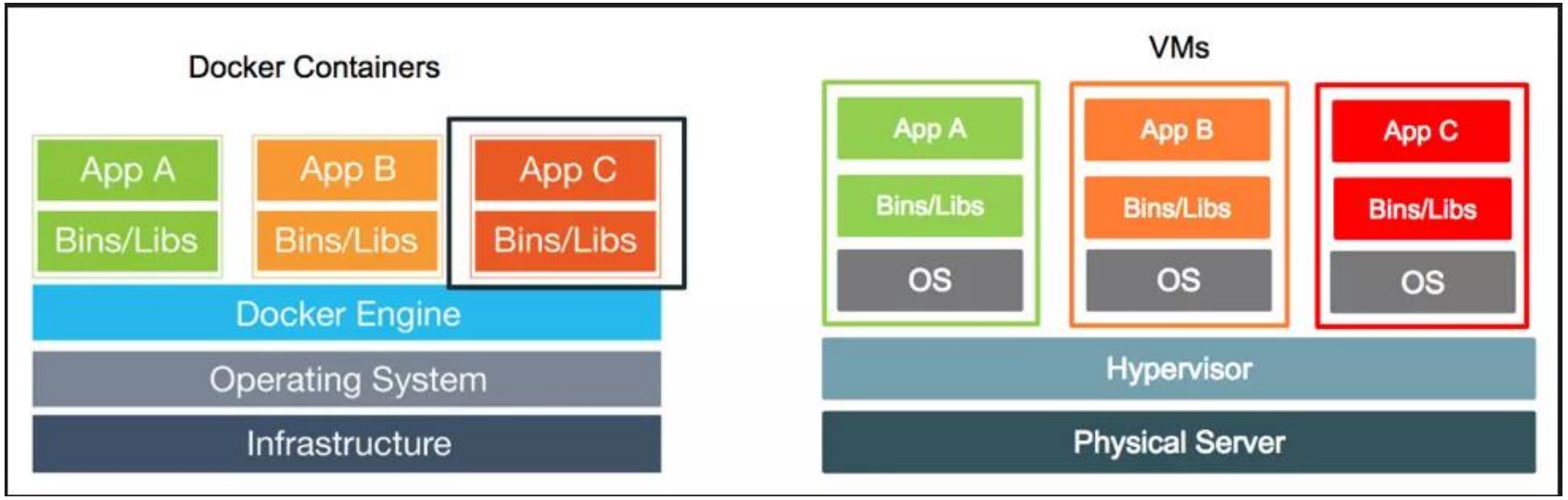
2010(dotcloud)
2013: Docker -opensource
Uso de LXC + cgroups + API



Contenedores - Contexto

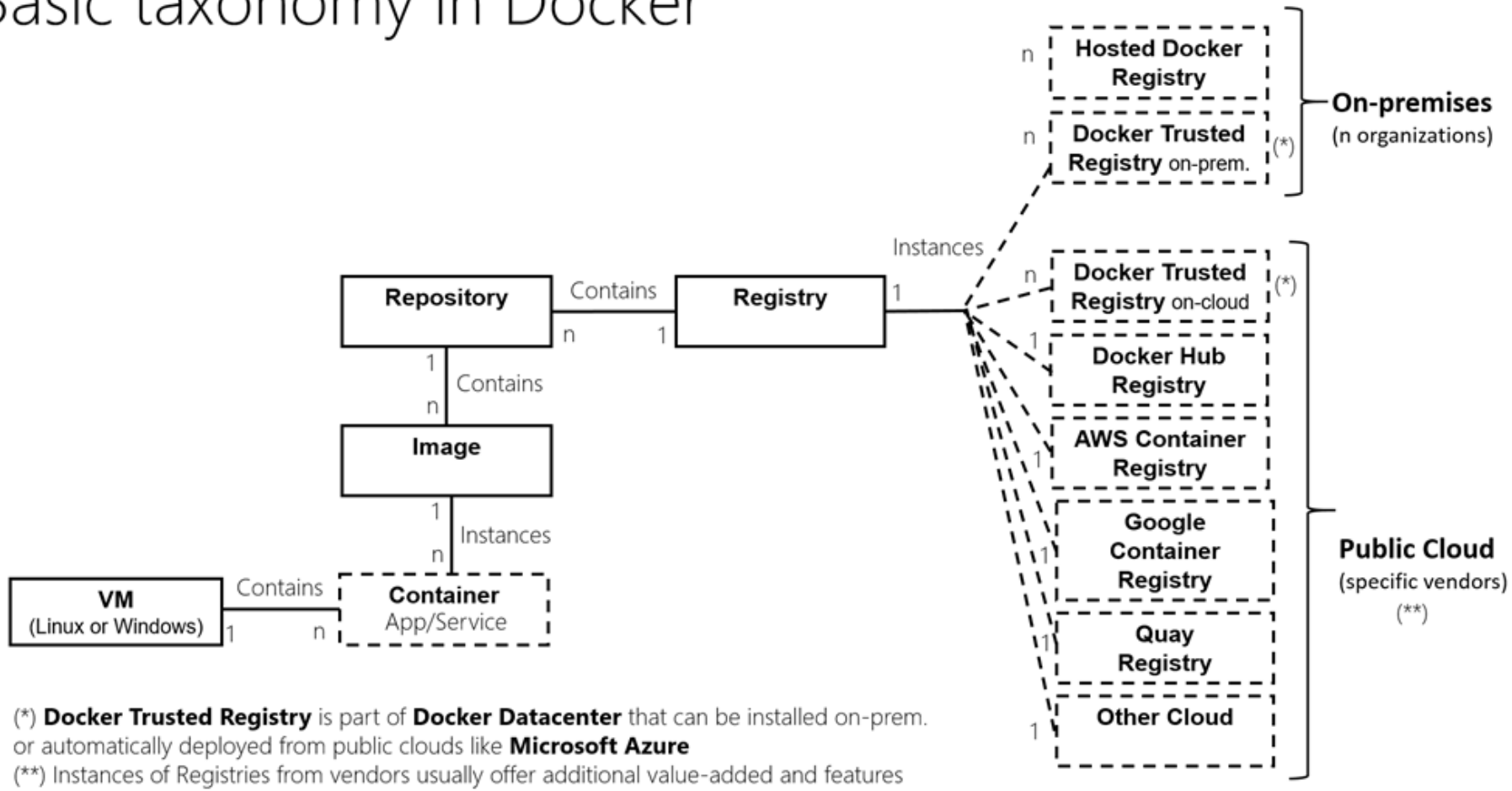


Contenedores - Contexto



Contenedores - Contexto

Basic taxonomy in Docker



(*) **Docker Trusted Registry** is part of **Docker Datacenter** that can be installed on-prem. or automatically deployed from public clouds like **Microsoft Azure**
(**) Instances of Registries from vendors usually offer additional value-added and features compared to a plain docker-registry.

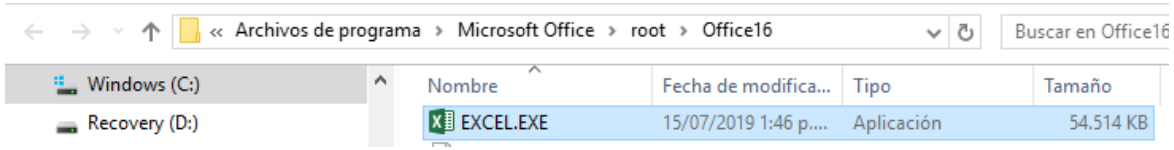
Fuente: https://mitra.computa.asia/article/msdn-docker-and-containers-basic-elements-and-taxonomy

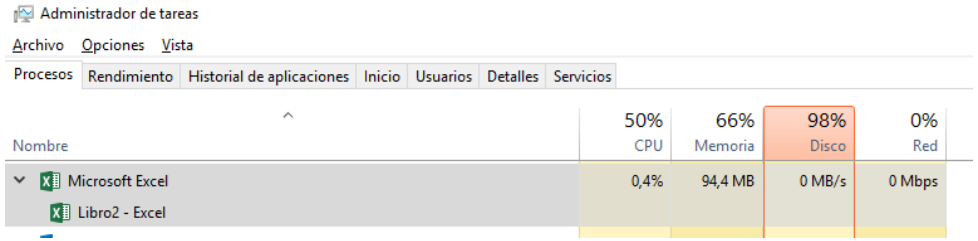
Contenedores - Contexto

Imagen:
Archivo binario

Contenedor
En ejecución

Símil: *"el símil permite conectar diferentes elementos de una forma simple y eficaz para ofrecer una nueva forma de ver o entender una cosa determinada, pues opera trasladando las características o rasgos, simbólicos o evidentes, de una cosa a la otra"*

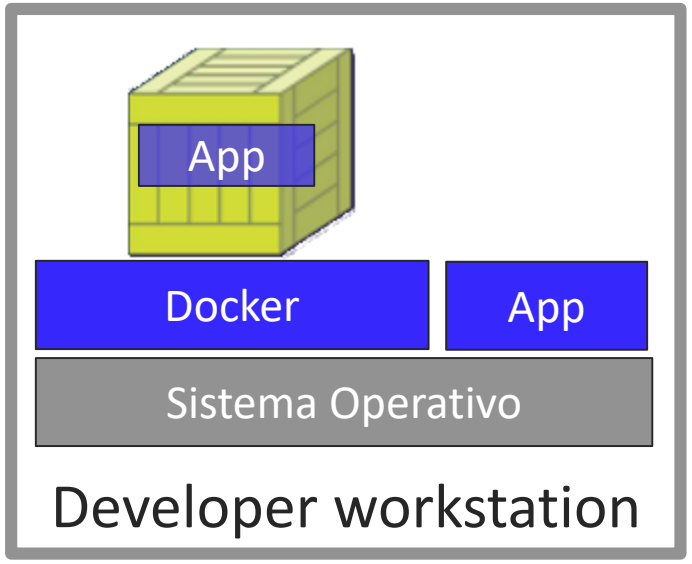




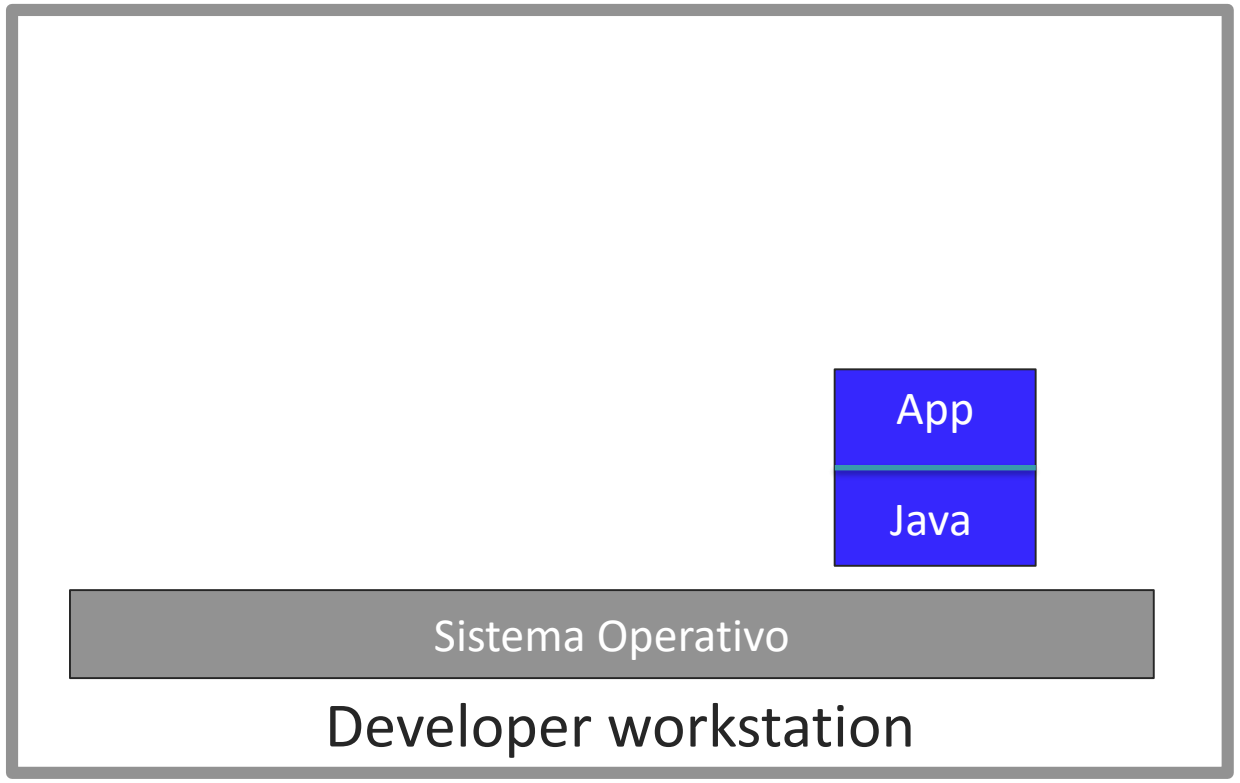
Fuente: Codeschool –trydocker

Contenedores - Contexto

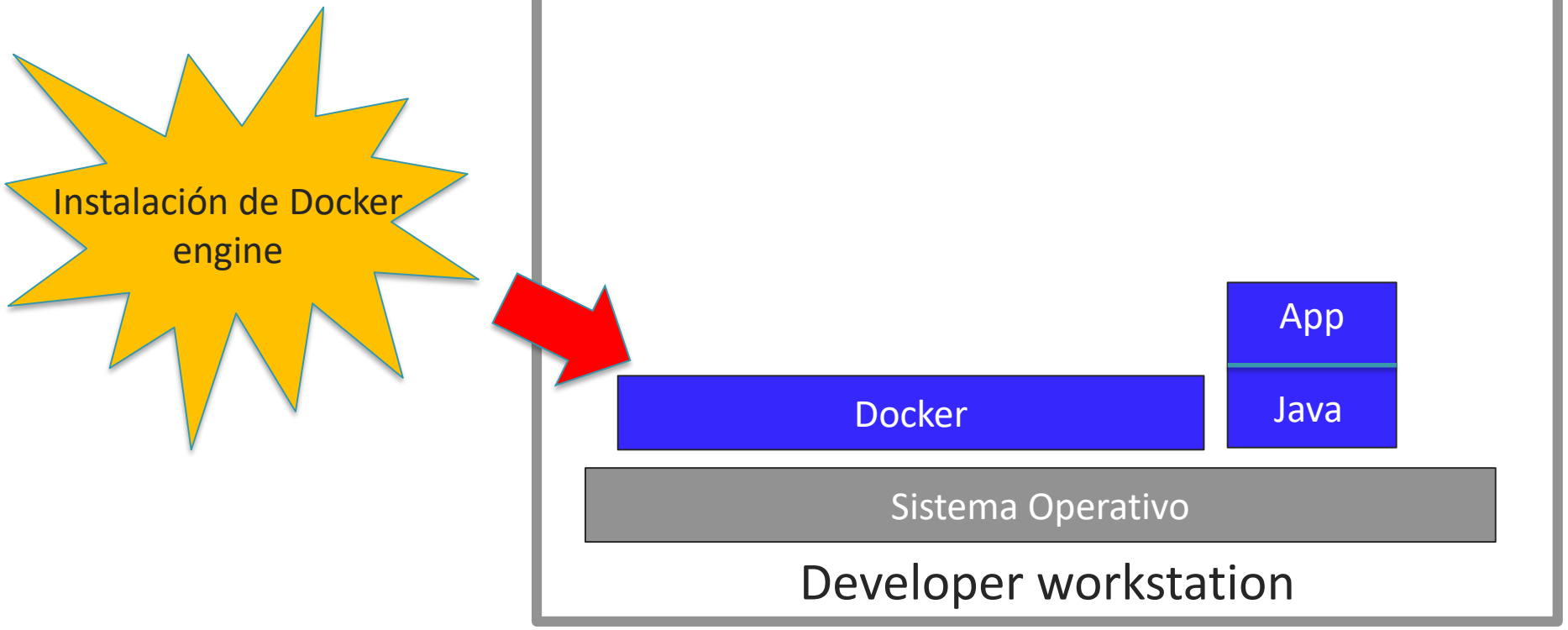
Contenedores - Contexto



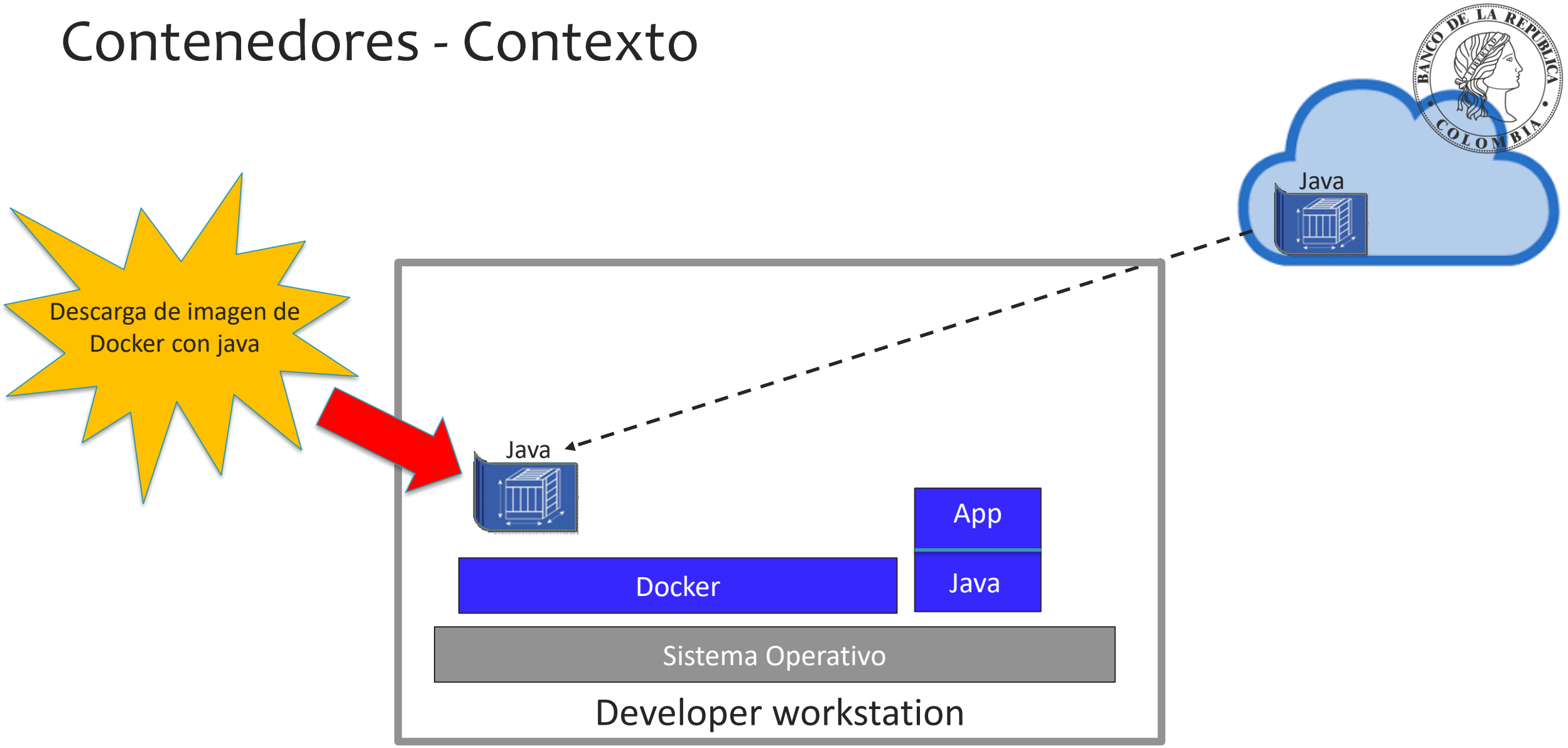
Contenedores - Contexto



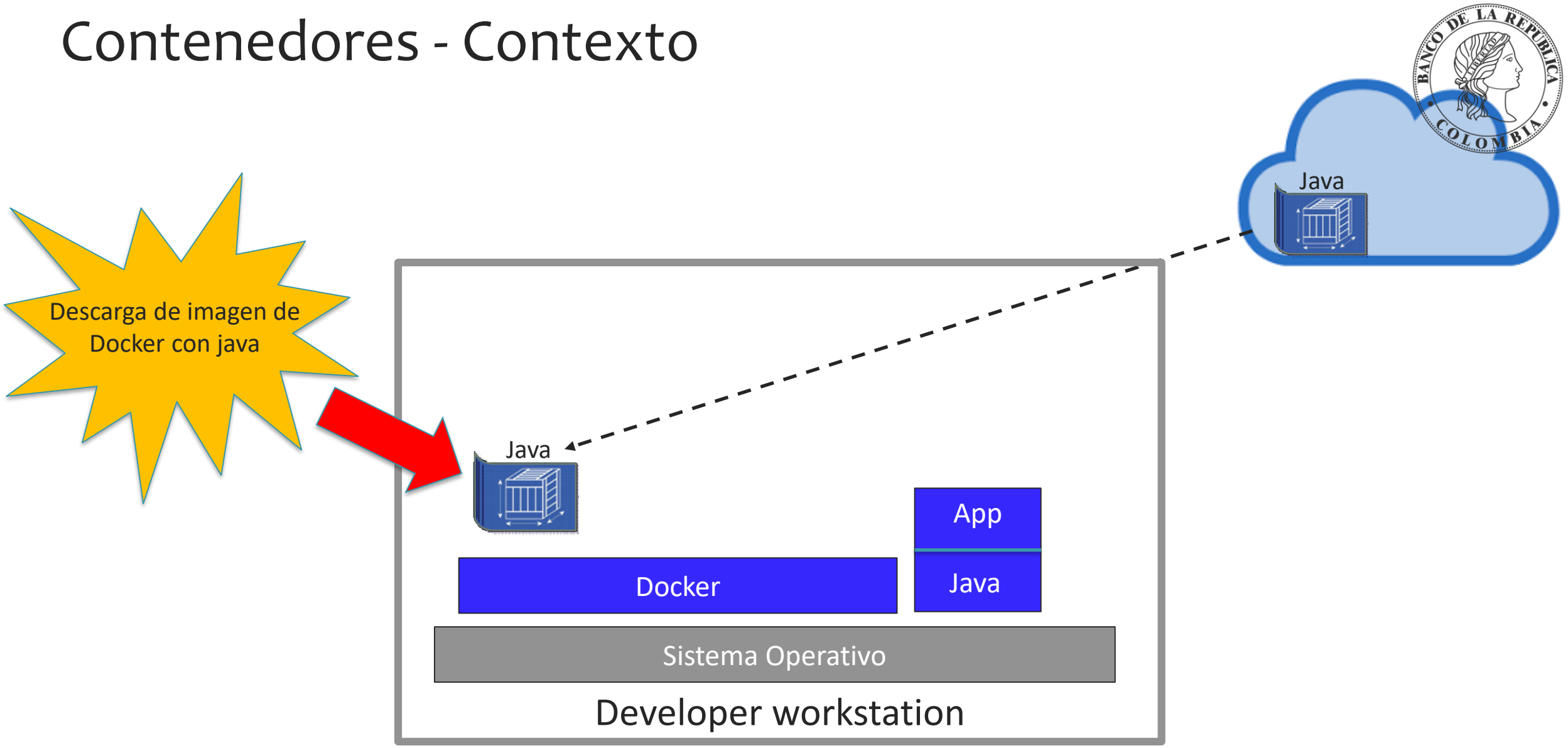
Contenedores - Contexto



Contenedores - Contexto



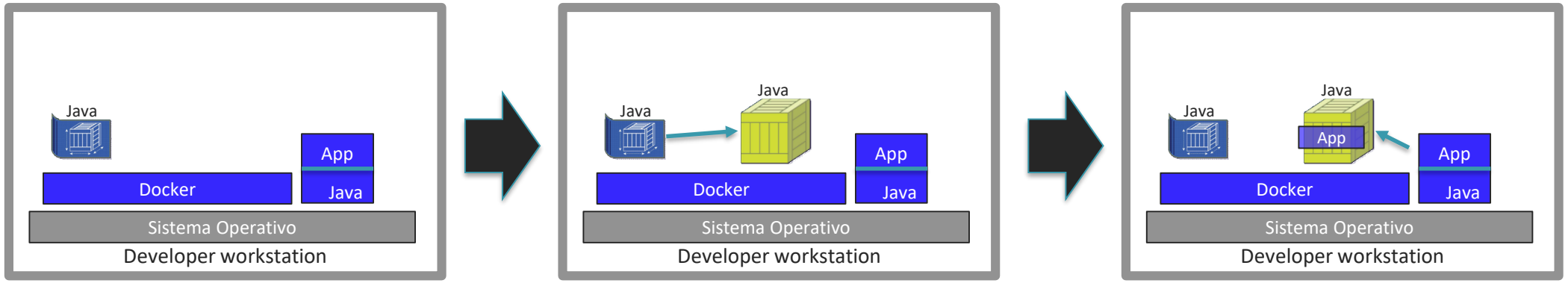
Contenedores - Contexto



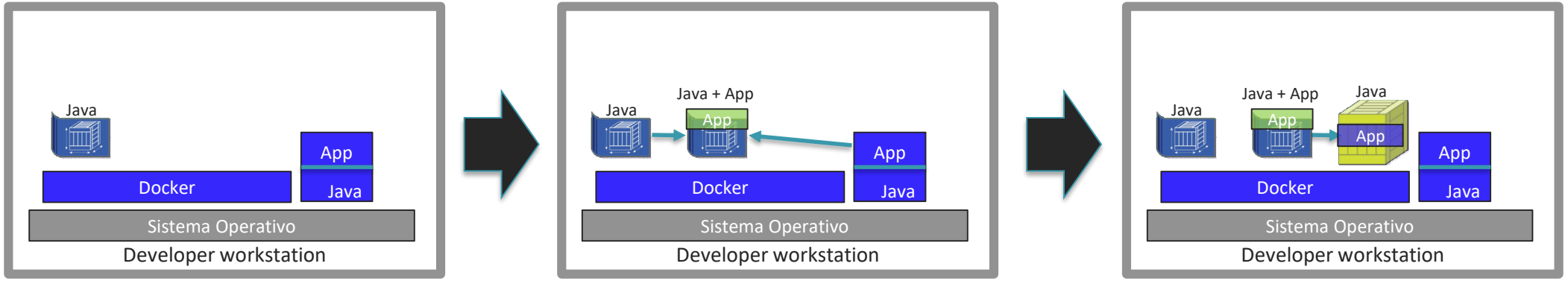
Contenedores - Contexto

Alternativa 1:

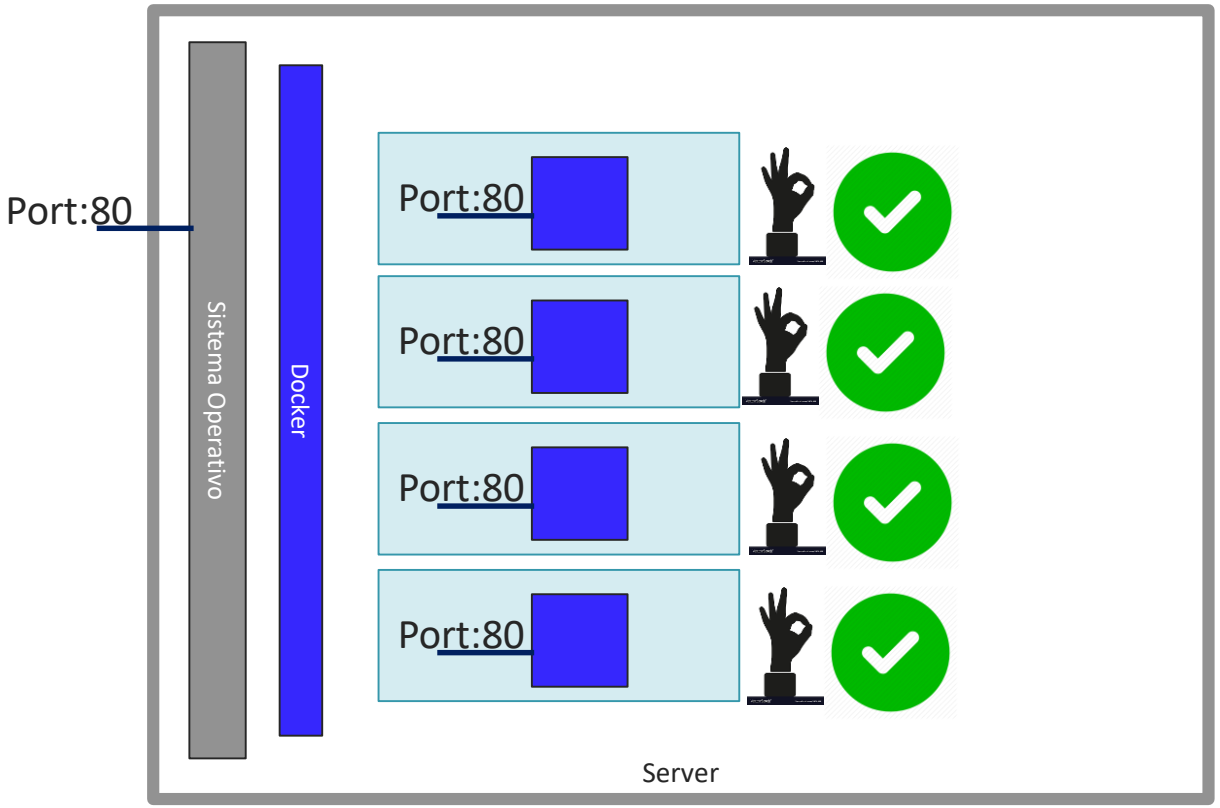
Cuando se recree el contenedor, se pierden los cambios



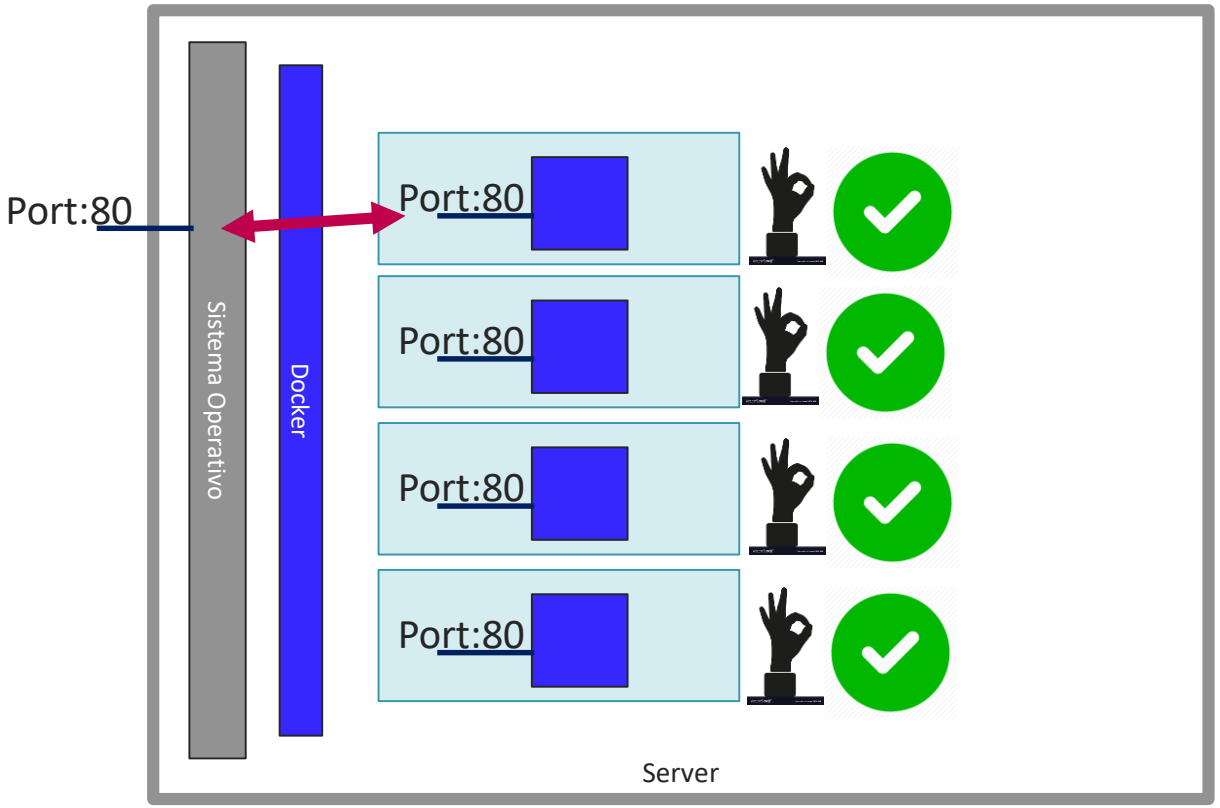
Alternativa 2:



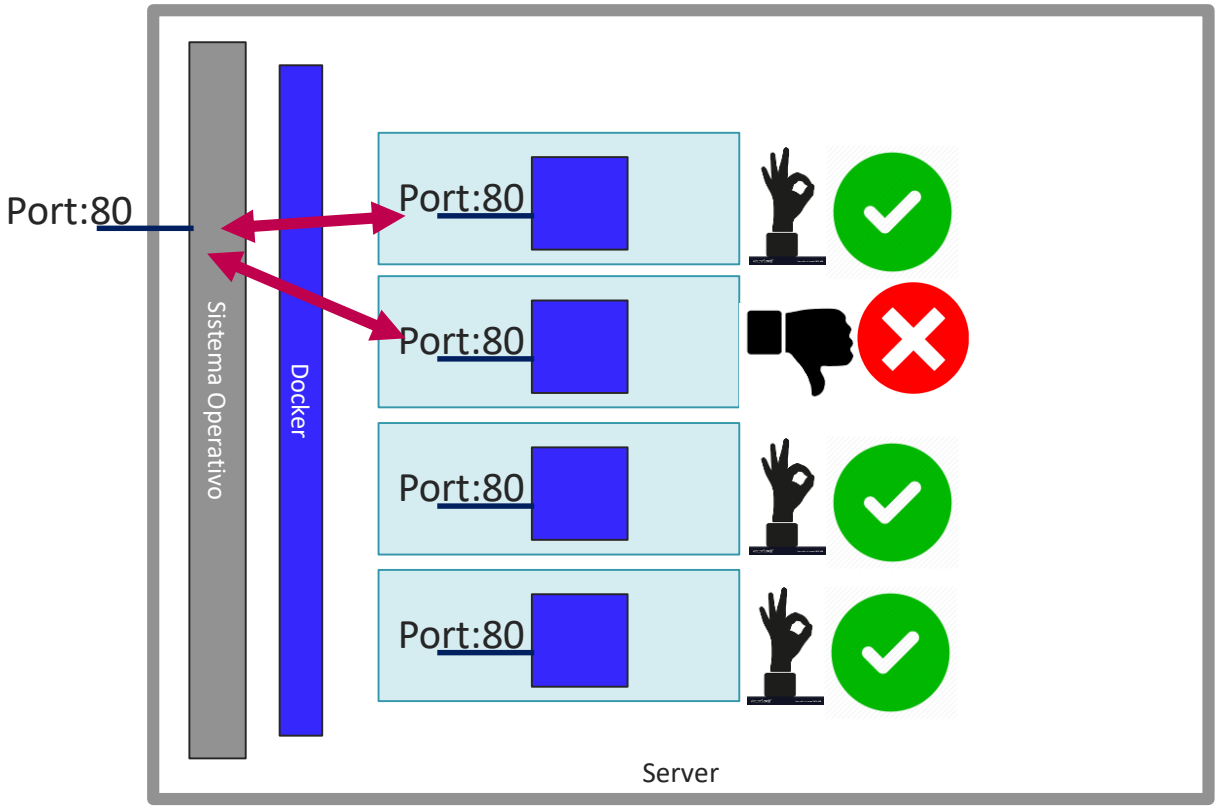
Como funciona el networking en Docker / Docker-compose



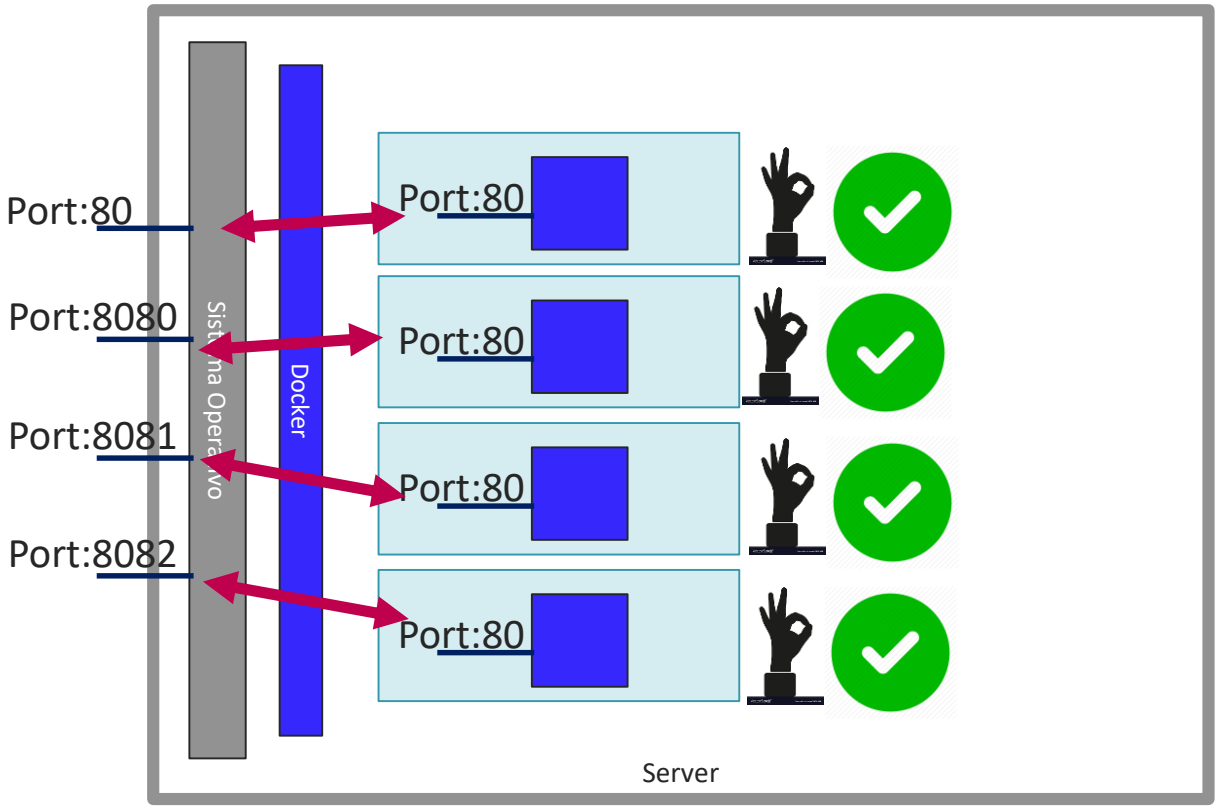
Como funciona el networking en Docker / Docker-compose



Como funciona el networking en Docker / Docker-compose

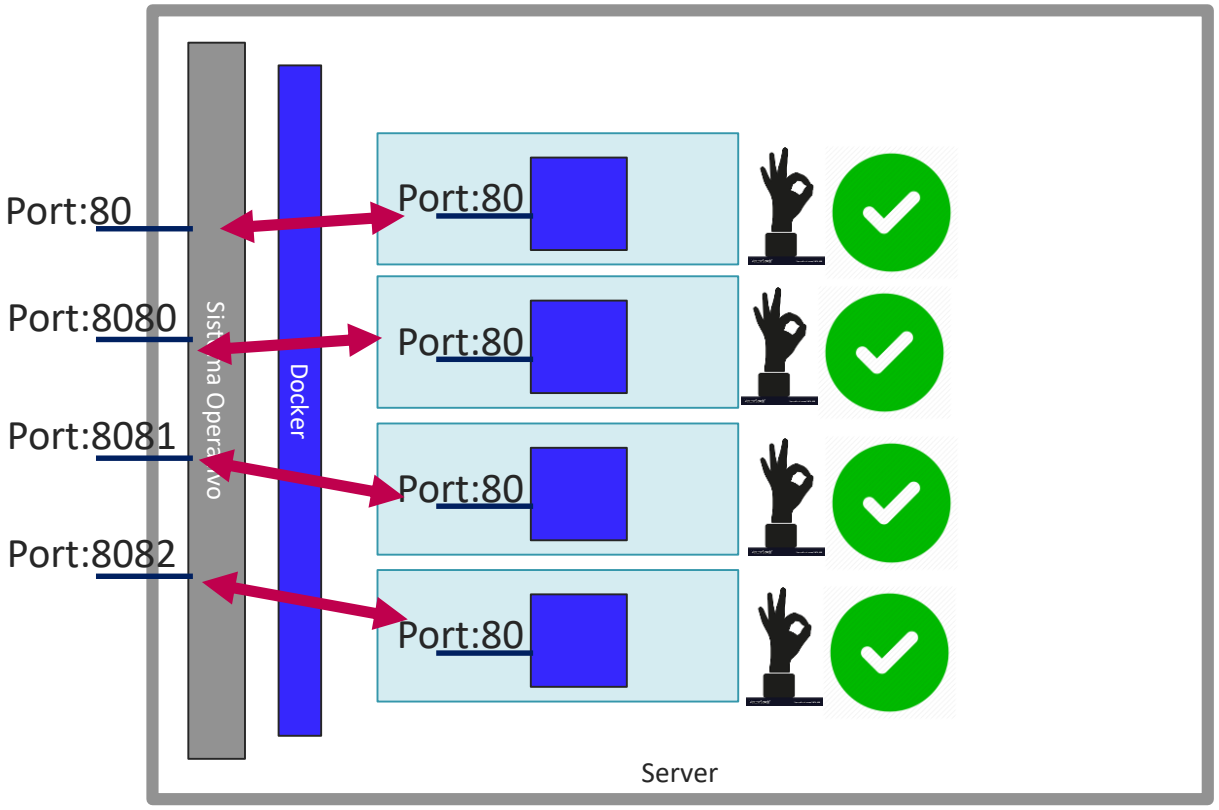


Como funciona el networking en Docker / Docker-compose

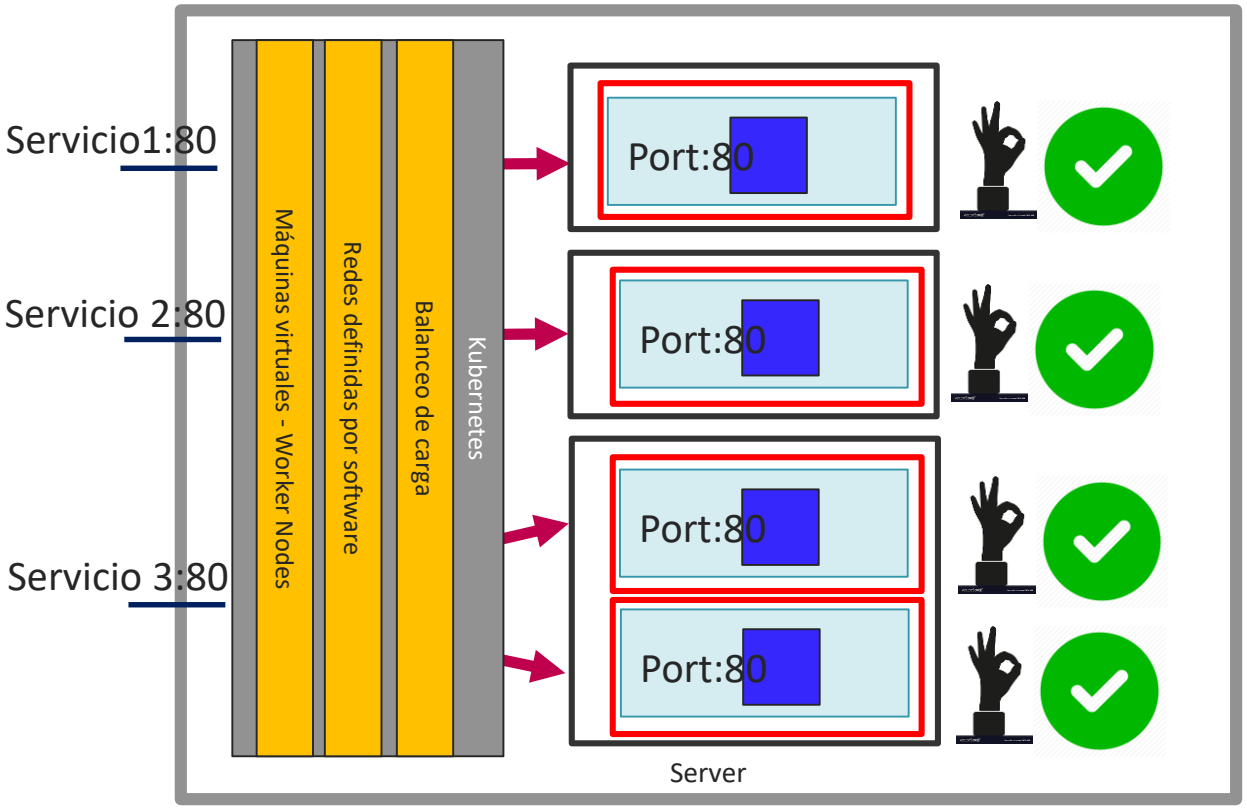


Como funciona el networking en Kubernetes- Qué es un POD

Contenedores (peladitos)



Kubernetes



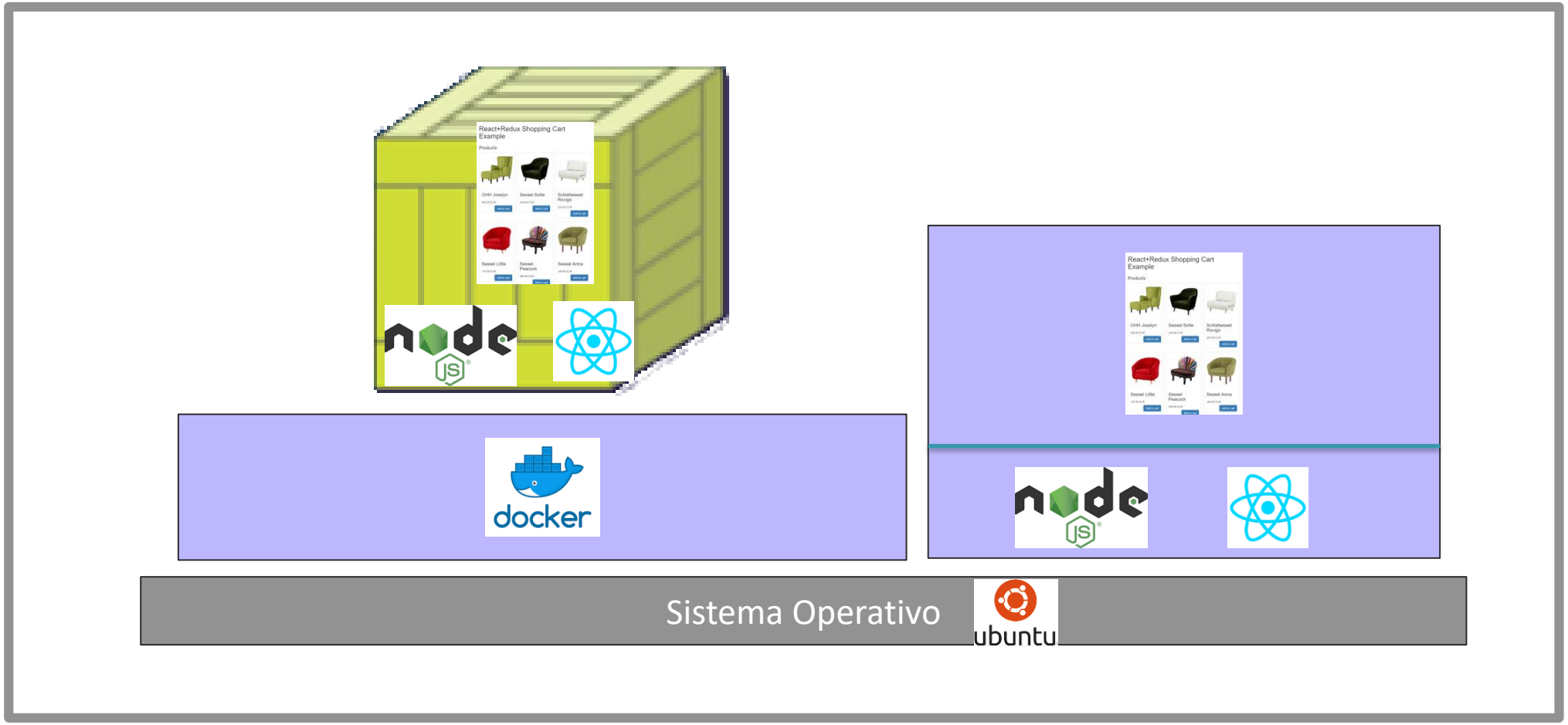
AGENDA

Contexto de negocio

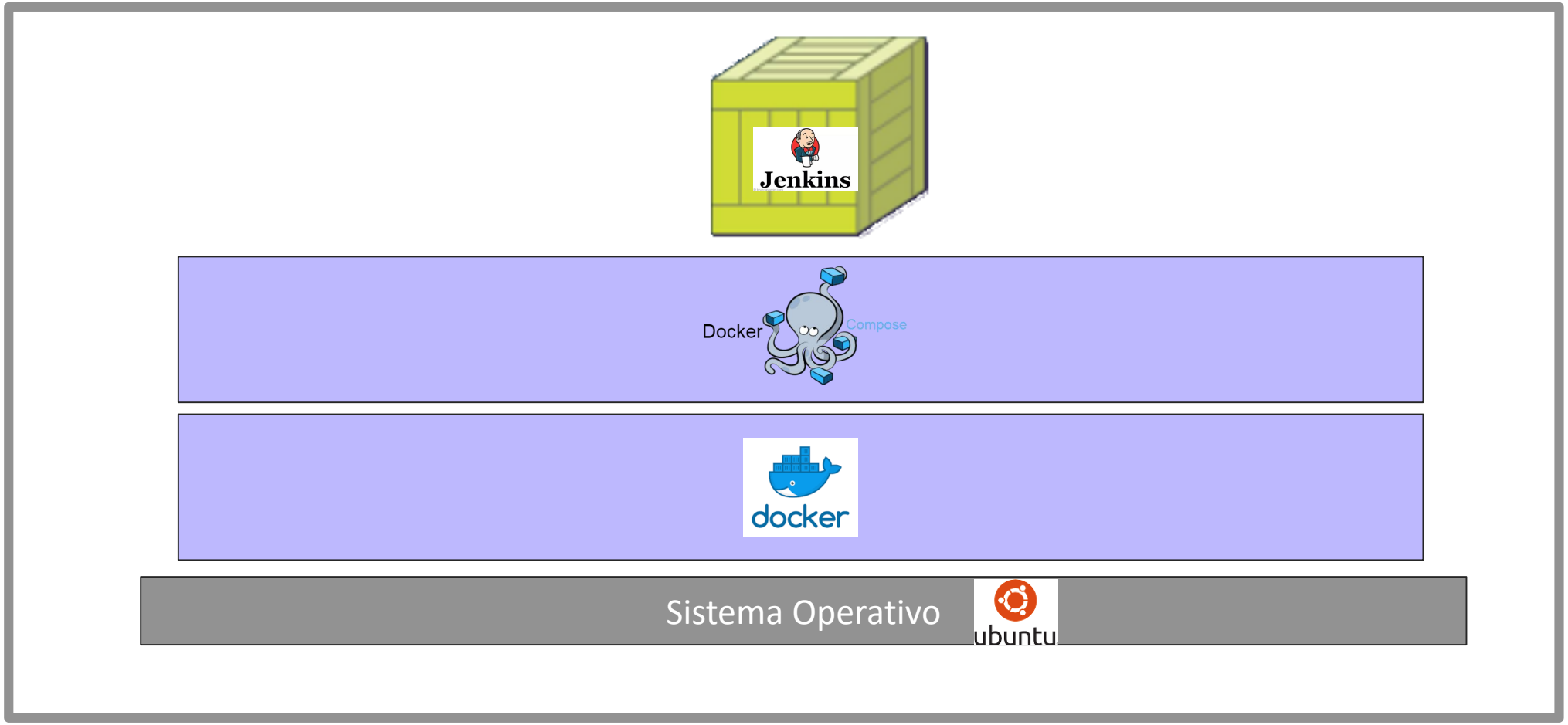
Que es la tecnología contenedores?

Workshop

Contenedores – Workshop - Parte 1 (contenerizar una app)

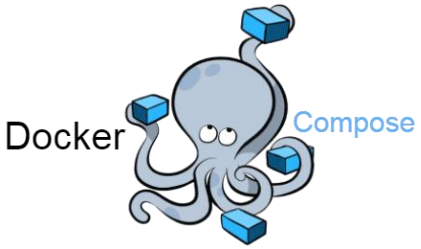


Contenedores – Workshop - Parte 2 (usar una app distribuida como contenedor)



Contenedores – Workshop - Prerequisitos

https://github.com/danielpenagos/reactjs-shopping-cart




```
apt install git

git --version

sudo apt-get install     apt-transport-https ca-certificates     curl software-
properties-common

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -

sudo add-apt-repository    "deb [arch=amd64]
https://download.docker.com/linux/ubuntu \
    $(lsb_release -cs) \
    stable"

sudo apt-get install docker-ce

docker --version

sudo curl -L "https://github.com/docker/compose/releases/download/1.22.0/docker-
compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose

sudo chmod +x /usr/local/bin/docker-compose

docker-compose --version

sudo usermod -a -G docker <usuario>
```

Contenedores – Workshop - Prerequisitos

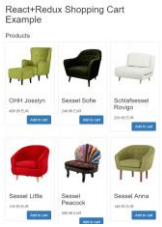
https://github.com/danielpenagos/reactjs-shopping-cart

```
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.34.0/install.sh | bash

source /home/dpenagbo/.bashrc

nvm install 11
```

1. Clonar el repo: `git clone https://github.com/danielpenagos/reactjs-shopping-cart.git`

2. Probar la aplicación: (`npm install , npm start`)

`http://dai-<usuario>.eastus.cloudapp.azure.com:3000/`



3. Revisar el Dockerfile en GitHub o el archivo clonado con vi o cat:

GitHub, Inc. [US] | https://github.com/danielpenagos/reactjs-shopping-cart/blob/master/Dockerfile

Branch: master ▾ reactjs-shopping-cart / Dockerfile

Ubuntu dockerfile

0 contributors

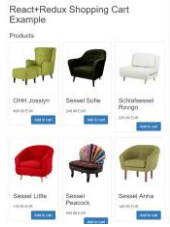
29 lines (19 sloc) | 520 Bytes

```
# use a node base image
FROM node:11

# Create app directory
WORKDIR /usr/src/app

# Install app dependencies
# A wildcard is used to ensure both package.json AND package-lock.json are copied
# where available (npm@5+)
COPY package*.json ./

RUN npm install
# If you are building your code for production
# RUN npm ci --only=production

# Bundle app source
COPY . .

# set maintainer
LABEL maintainer "dpenagbo@banrep.gov.co"

# set a health check

# tell docker what port to expose
EXPOSE 3000

CMD [ "npm", "start" ]
```

4. Construir la nueva imagen con la aplicación

```
docker build -t banrep/kart .

docker image ls

docker ps
```

Ojo con el conflicto de puertos



5. Probar la aplicación en contenedor:

```
docker run -d --name kart -p 3000:3000 banrep/kart
```

Comandos super útiles

```
docker ps
docker image ls
docker stop <container-id>
docker kill <container-id>
docker exec -it <container-id> bash
```

Contenedores – Workshop - Parte 2 (usar una app distribuida como contenedor)

https://github.com/danielpenagos/reactjs-shopping-cart

1. Revisar el archivo docker-compose.yaml

https://github.com/danielpenagos/reactjs-shopping-cart/blob/master/docker-compose.yaml

```yaml
version: '2'
services:
  jenkins:
    image: 'jenkins/jenkins:lts'
    ports:
      - '80:8080'
      - '443:8443'
      - '50000:50000'
    volumes:
      - 'jenkins_data:/jenkins_config'
volumes:
  jenkins_data:
    driver: local
```

2. Ejecutar el comando

```
docker-compose up -d
```

3. Revisar los logs e iniciar jenkins

```
docker ps

docker logs <container-id>
```

Comandos super útiles

```
docker ps
docker image ls
docker stop <container-id>
docker kill <container-id>
docker exec -it <container-id> bash
```


BANCO DE LA REPUBLICA