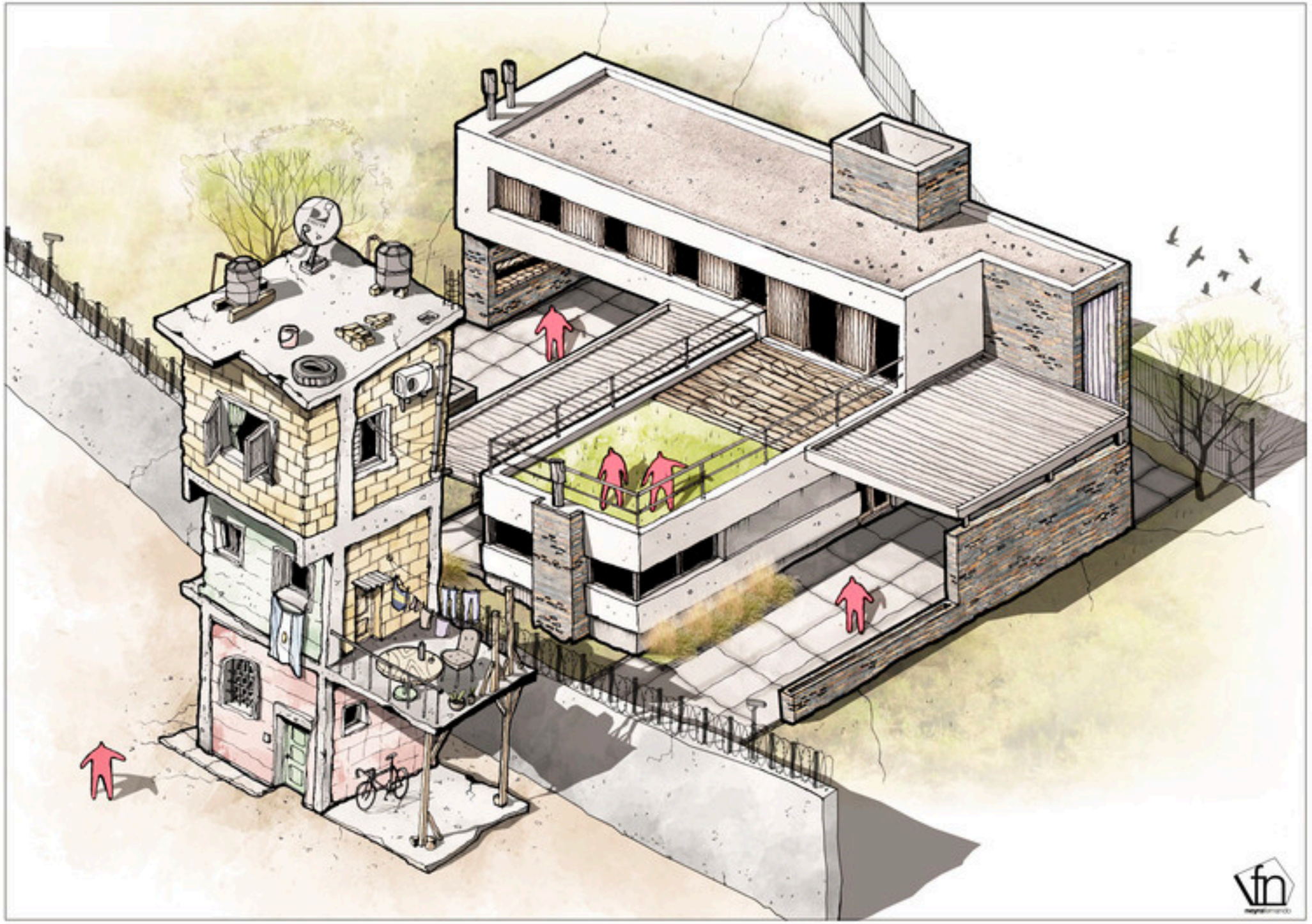
Arquitecturas Comunes

Juan Pablo Sandoval



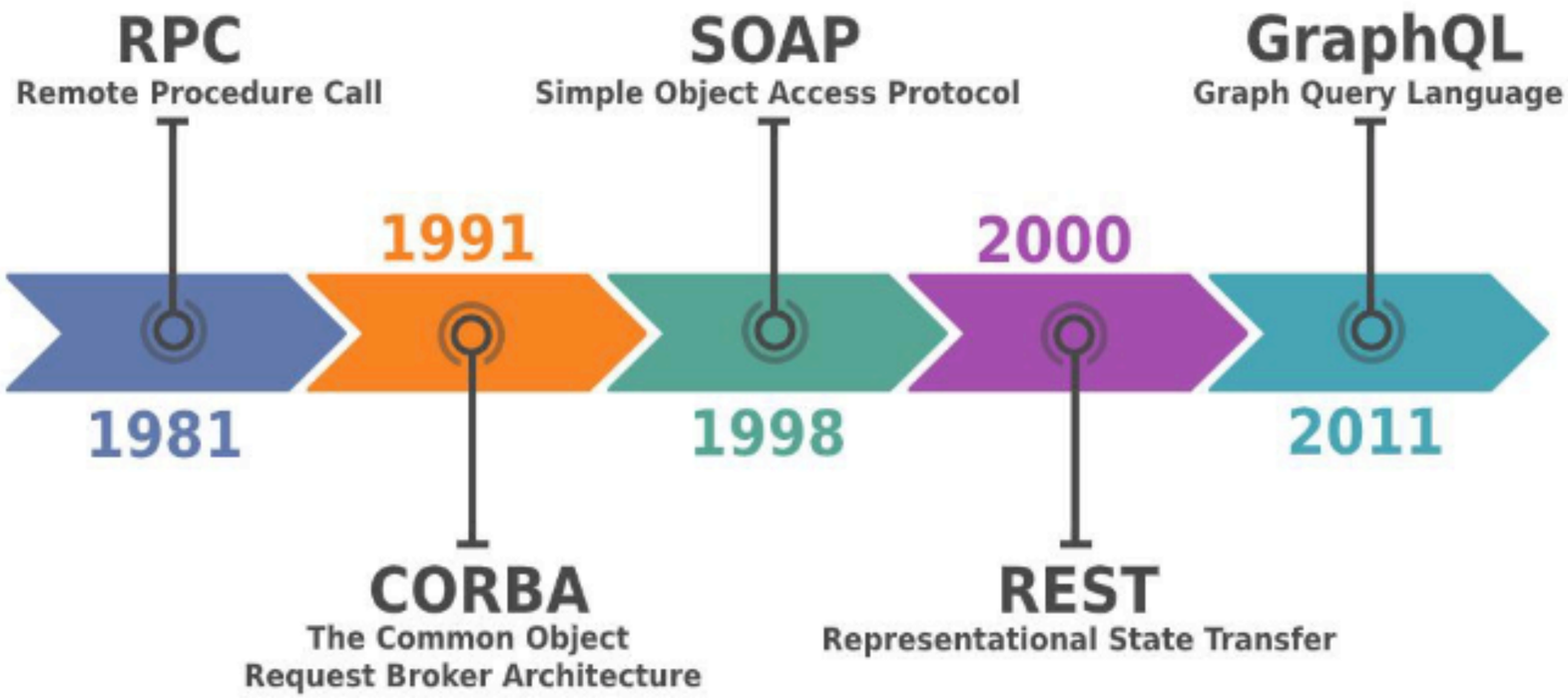
Que es arquitectura de Software?

- *Centrarse en la estructura*
- *Anticiparse a decisiones costosas.*
- *Hacer explícitas las decisiones para tener una buena calidad*

Requisitos funcionales siglo 2021

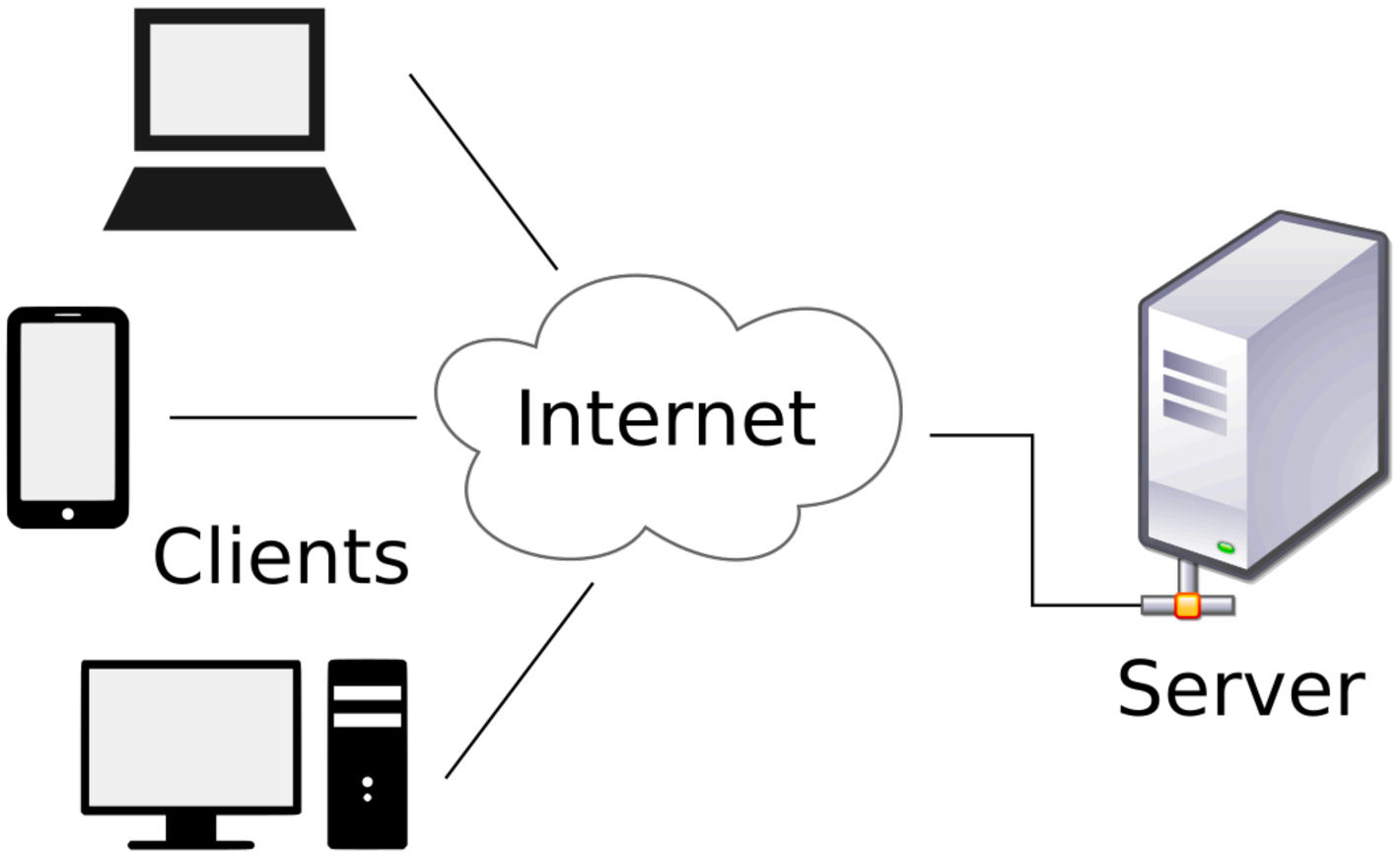
- *Sea desarrollado y mantenido por muchos años (**Maintainability**)*
- *Soporte millones de usuarios (**Scalability**)*
- *Disponible 24/7 (**Reliability**)*
- *Queremos que tenga buena latencia (**Efficiency**)*

Un poco de historia



Cliente Servidor

Juan Pablo Sandoval


Internet
Clients
Server

Clientes

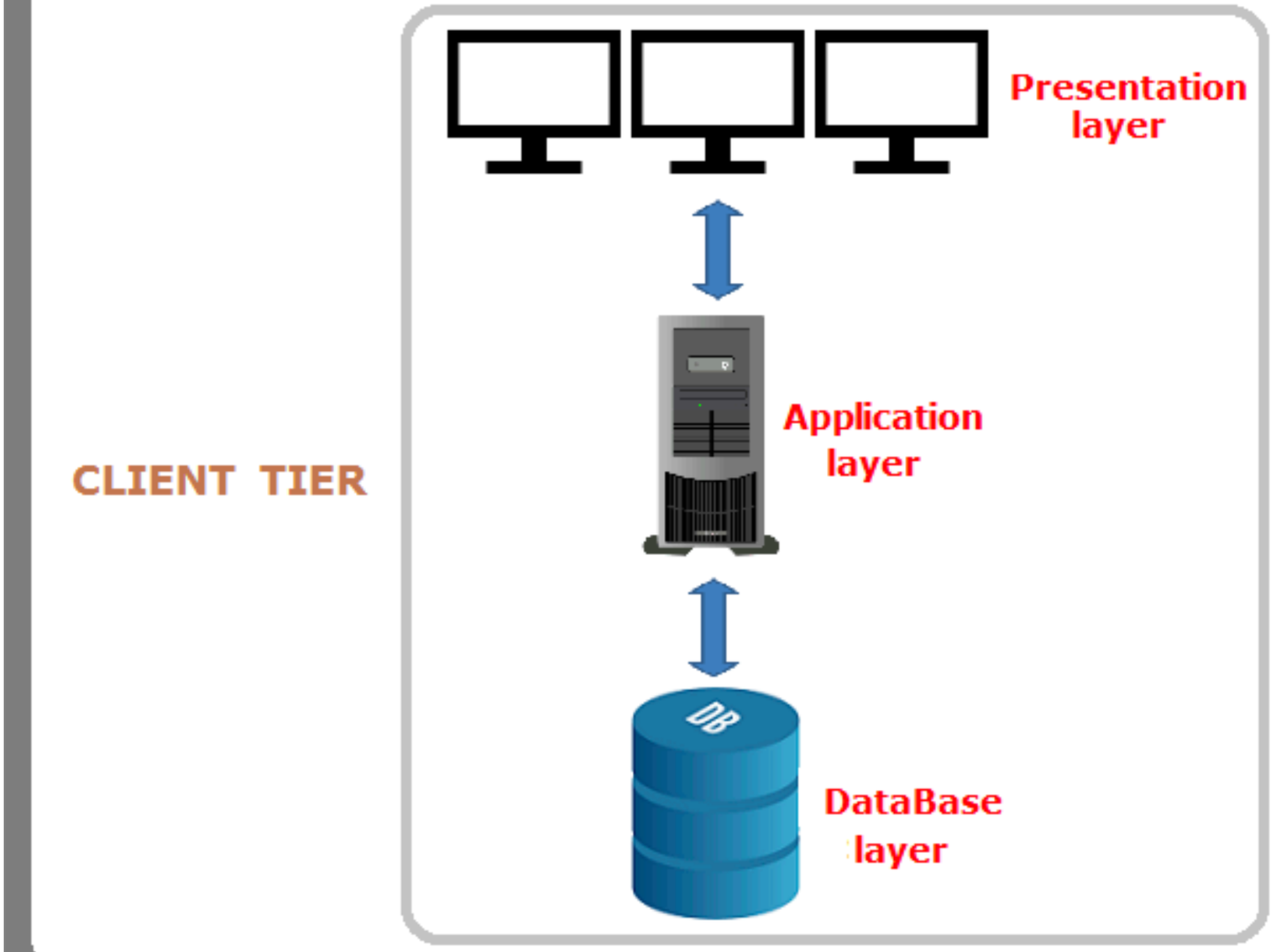
- *Una máquina o un programa que tiene la capacidad y una forma de enviar solicitudes (request) a través de internet.*
- *No necesariamente un browser.*
- *Un computador puede tener varios clientes.*

POSTMAN

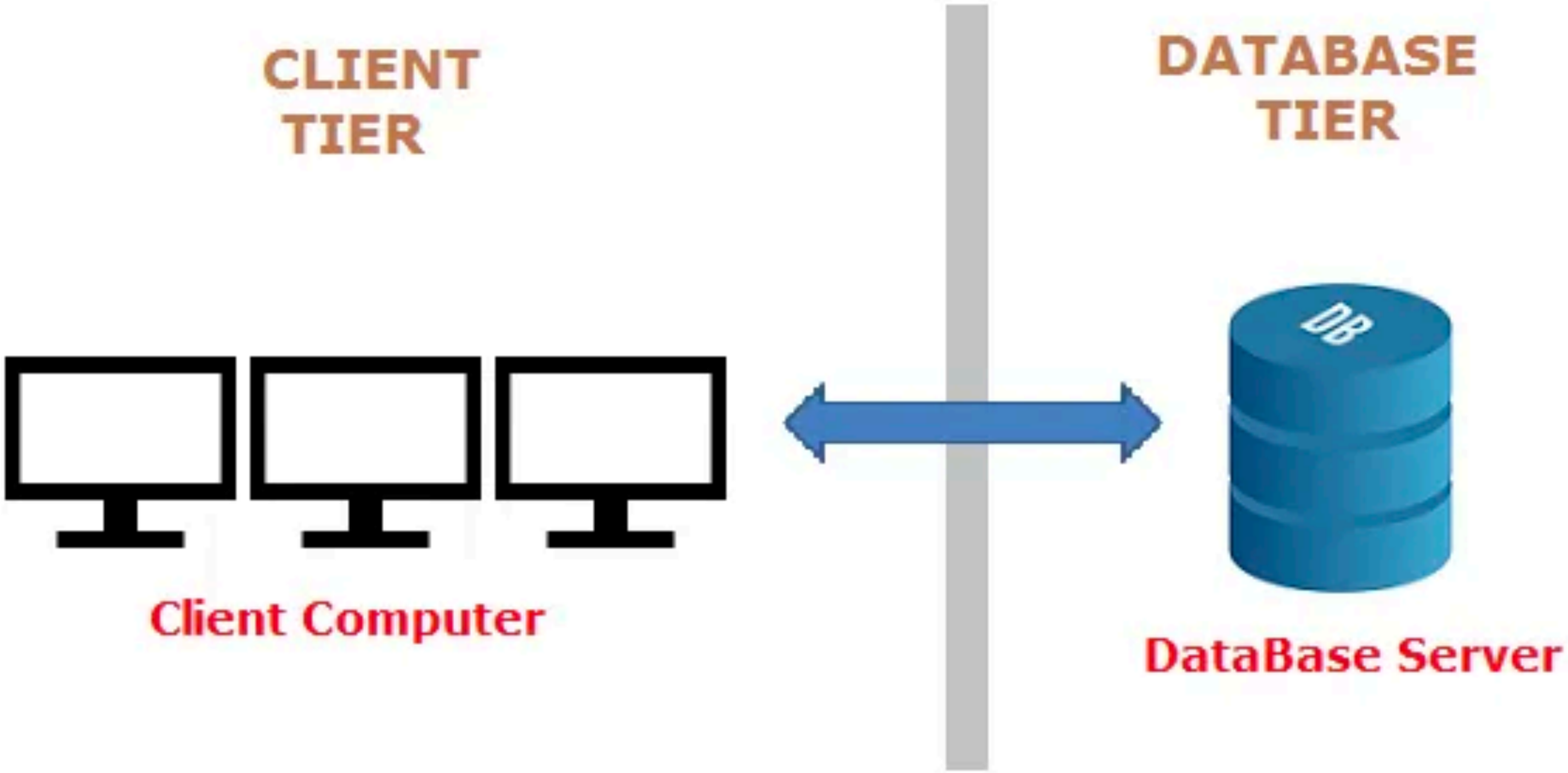
Servidor

- *No es necesariamente un dispositivo (i.e. computadora)*
- *Las computadoras de alto rendimiento son llamadas servidores porque ejecutan programas que dan servicios.*
- *Un servidor puede atender múltiples clientes al mismo tiempo.*

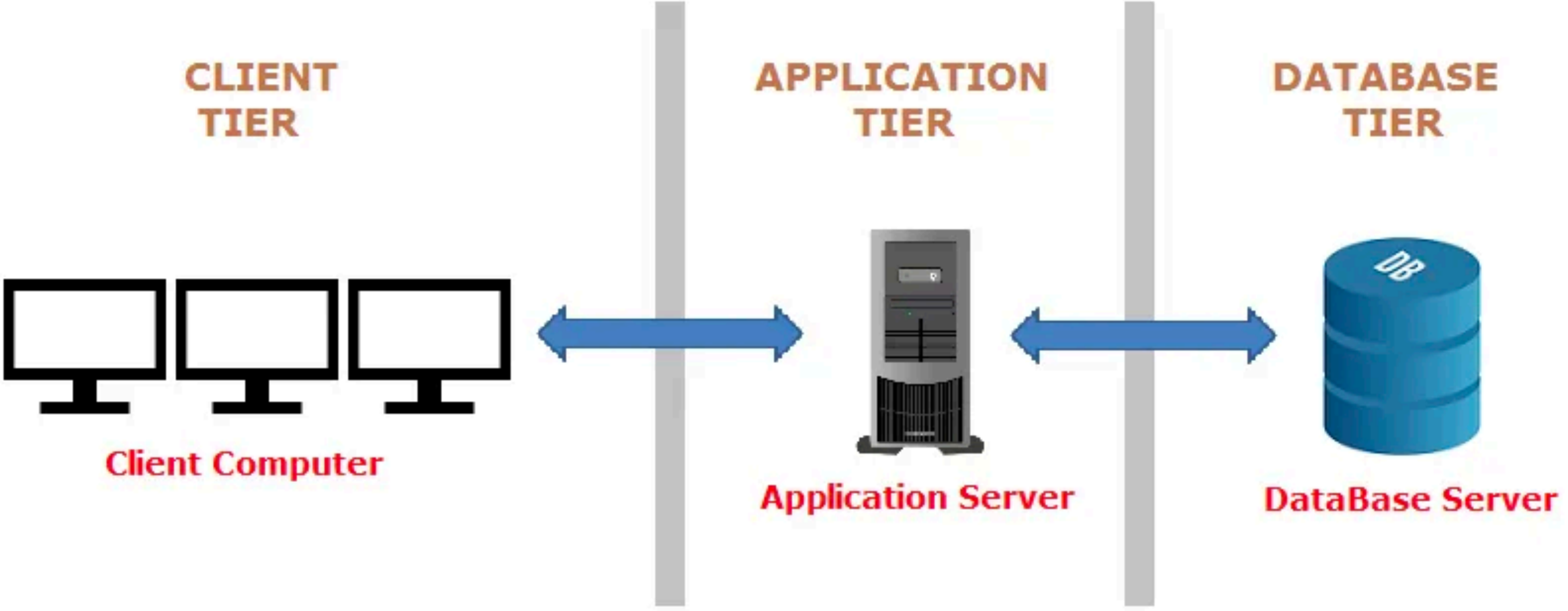
ONE-TIER ARCHITECTURE



TWO-TIER ARCHITECUTRE

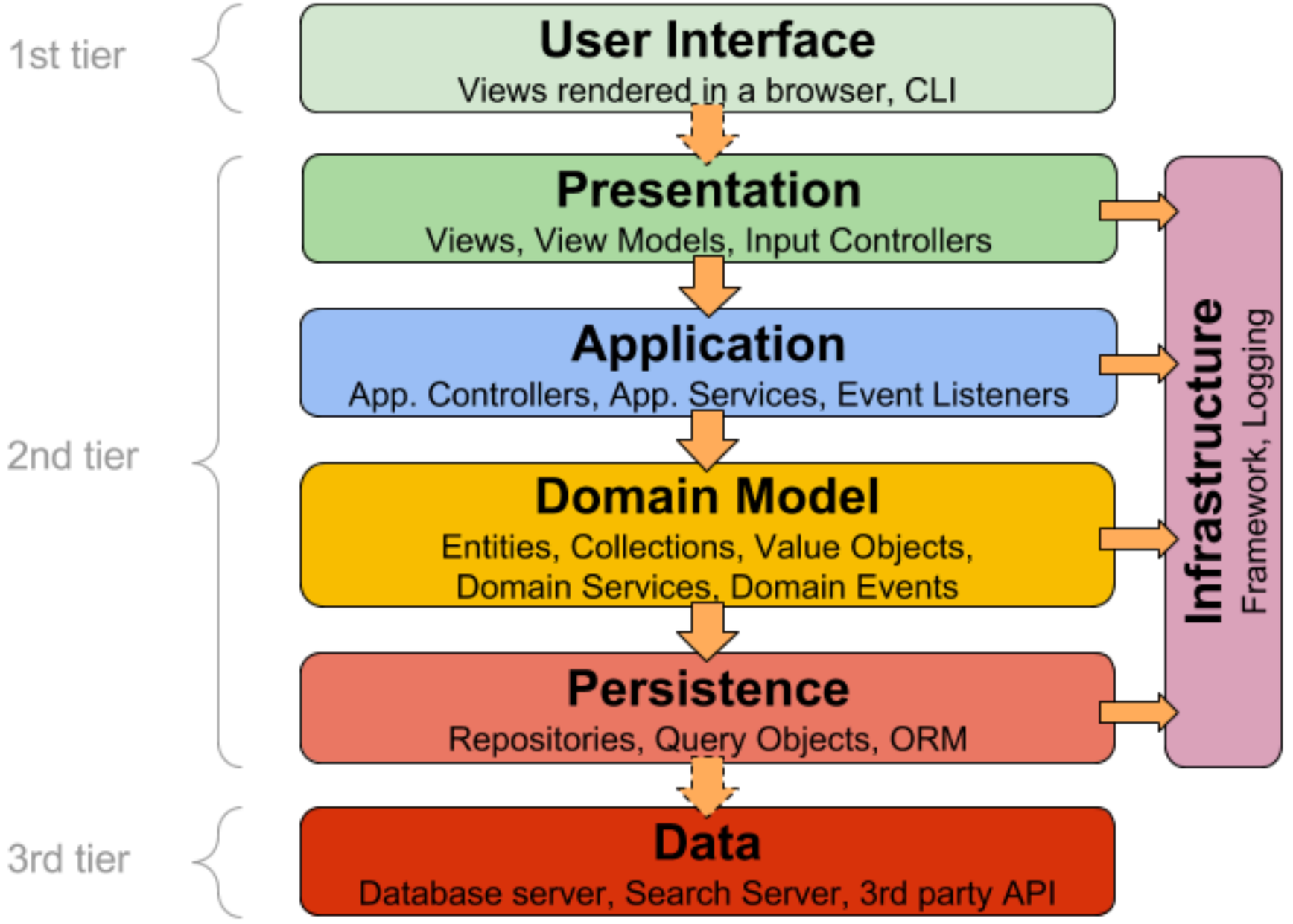


THREE-TIER ARCHITECTURE



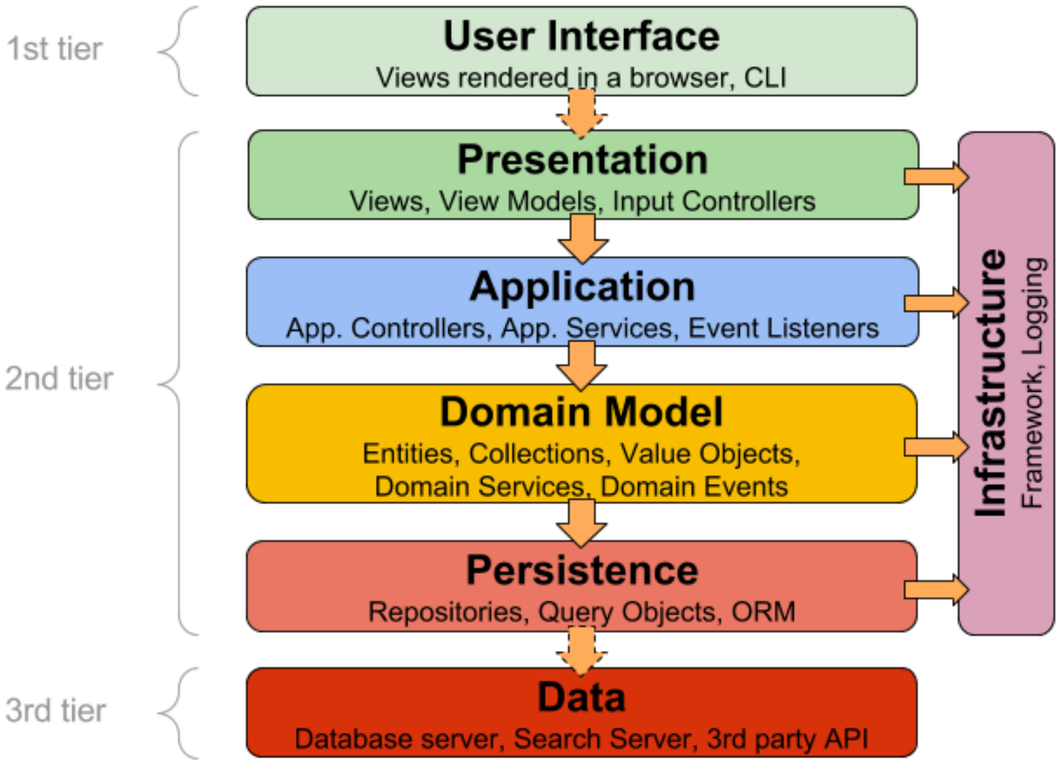
Arquitectura de Capas

Juan Pablo Sandoval



www.herbertograca.com

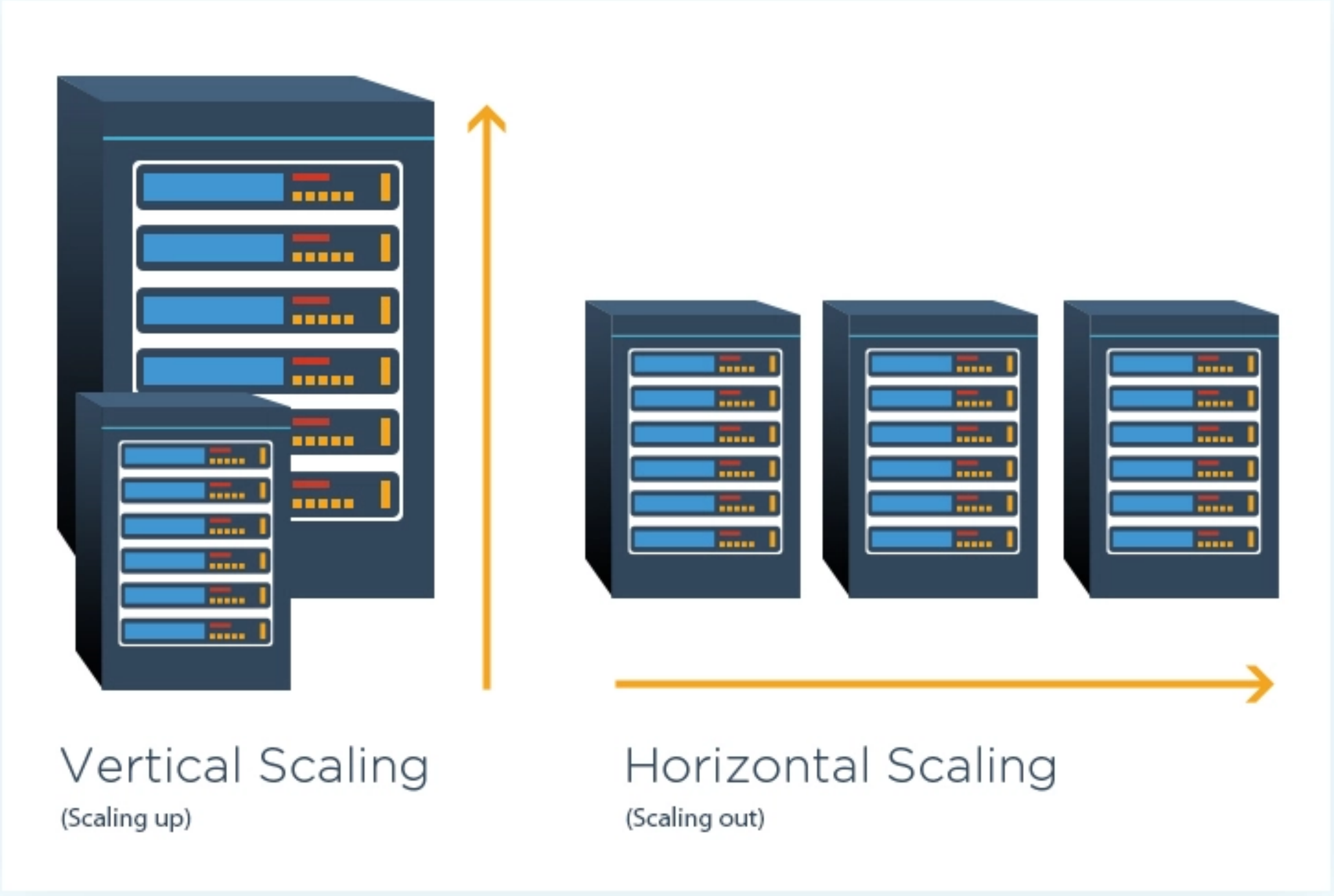
Pero podemos escalar?



100 user … okay

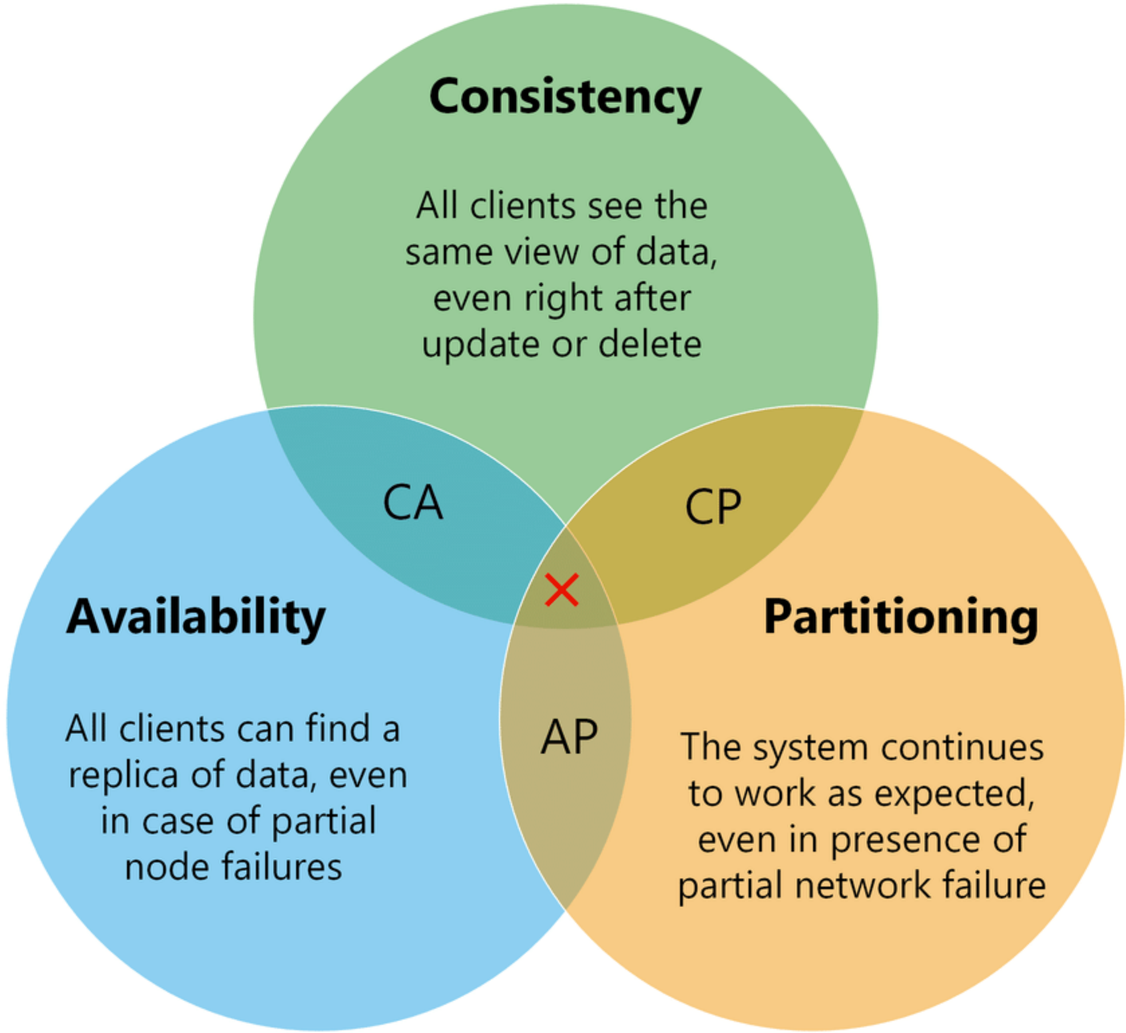
1000 users … okay

1000000 users … mmm no lo creo


Vertical Scaling
(Scaling up)
Horizontal Scaling
(Scaling out)

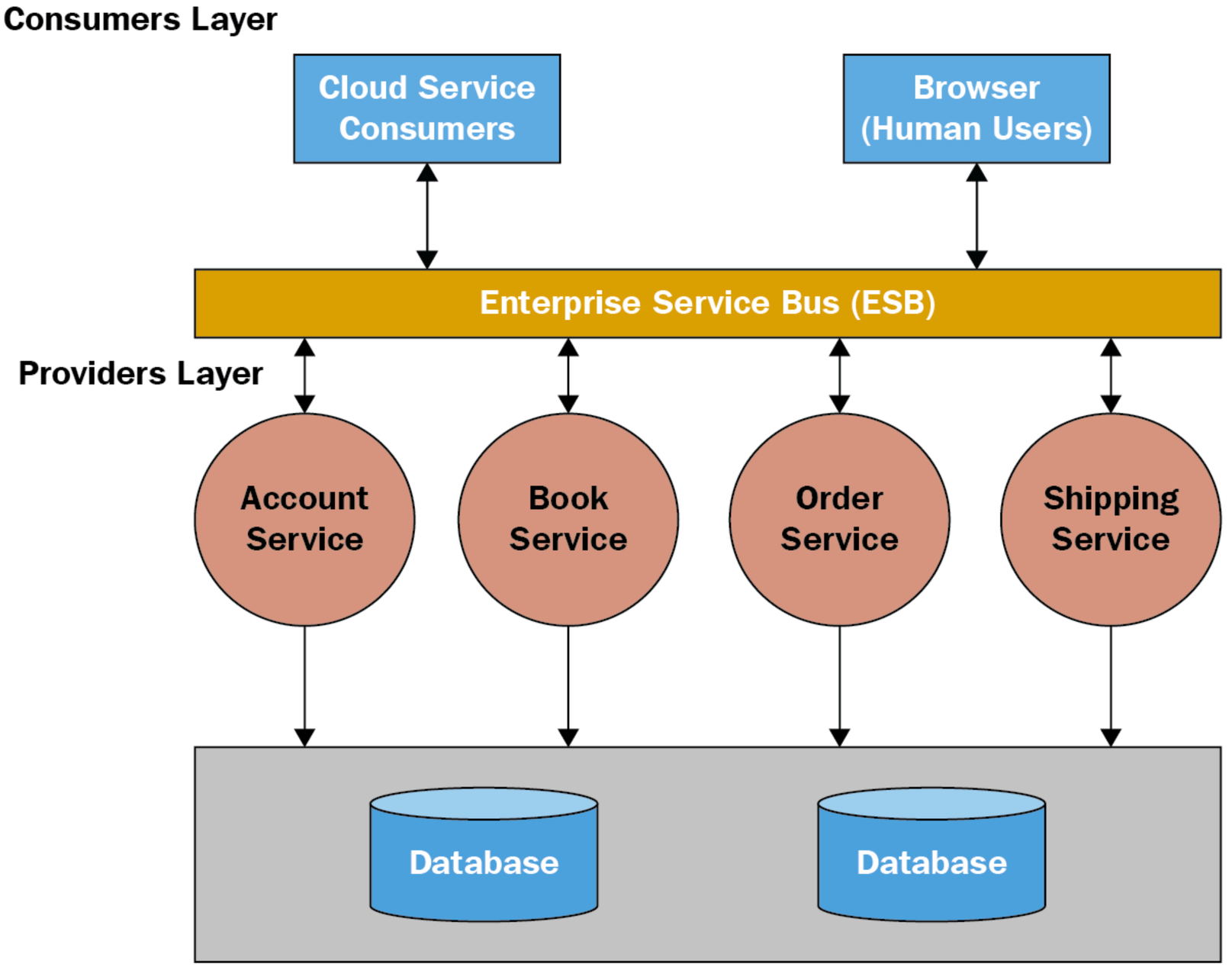
Escalar horizontalmente es fácil?

- *Según el teorema no se puede asegurar mas de estas dos características simultáneamente.*



Arquitectura Orientada a Servicios

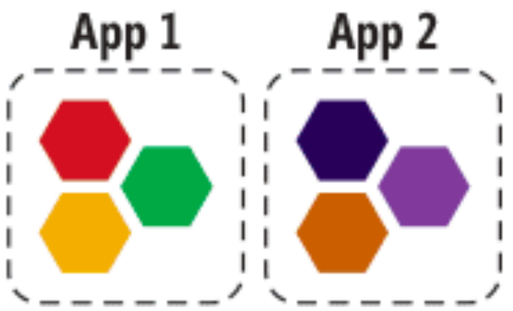
Juan Pablo Sandoval


Consumers Layer
Cloud Service Consumers
Browser (Human Users)
Enterprise Service Bus (ESB)
Providers Layer
Account Service
Book Service
Order Service
Shipping Service
Database
Database

Arquitectura Orientada a Micro-Servicios

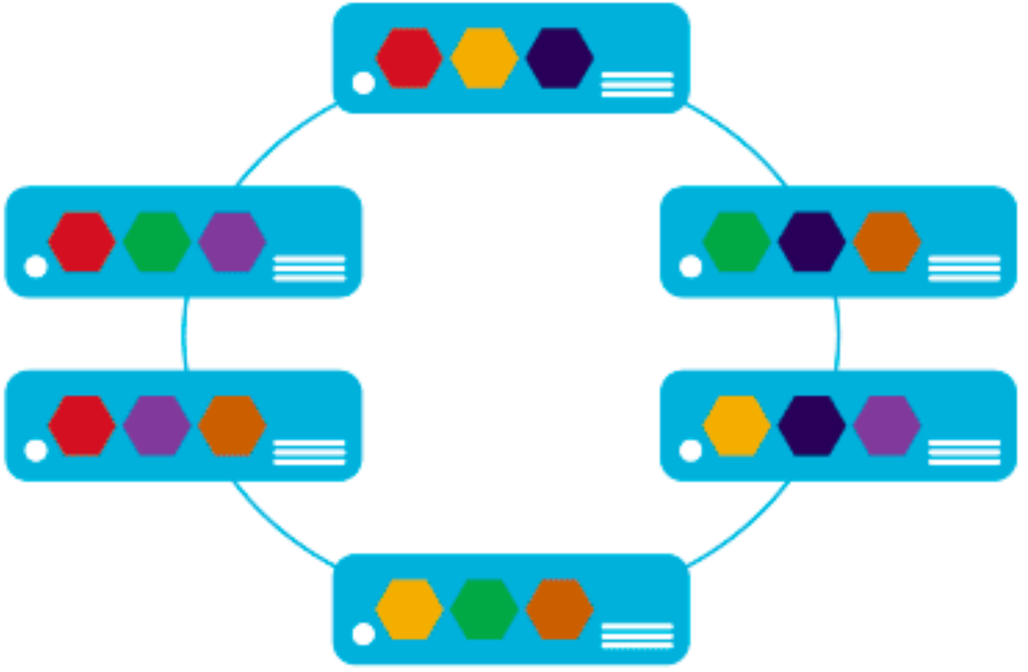
Juan Pablo Sandoval

Microservices Approach



A microservice approach segregates functionality into small autonomous services.

And scales out by **deploying independently** and replicating these services across servers/VMs/containers.

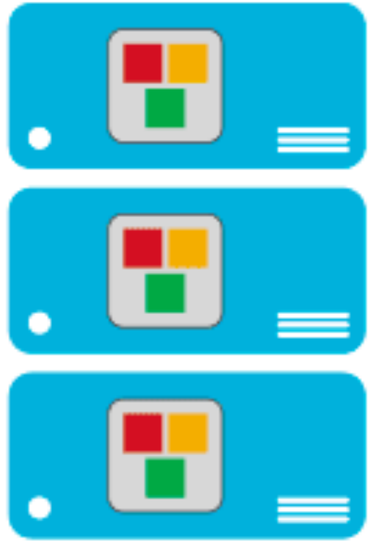


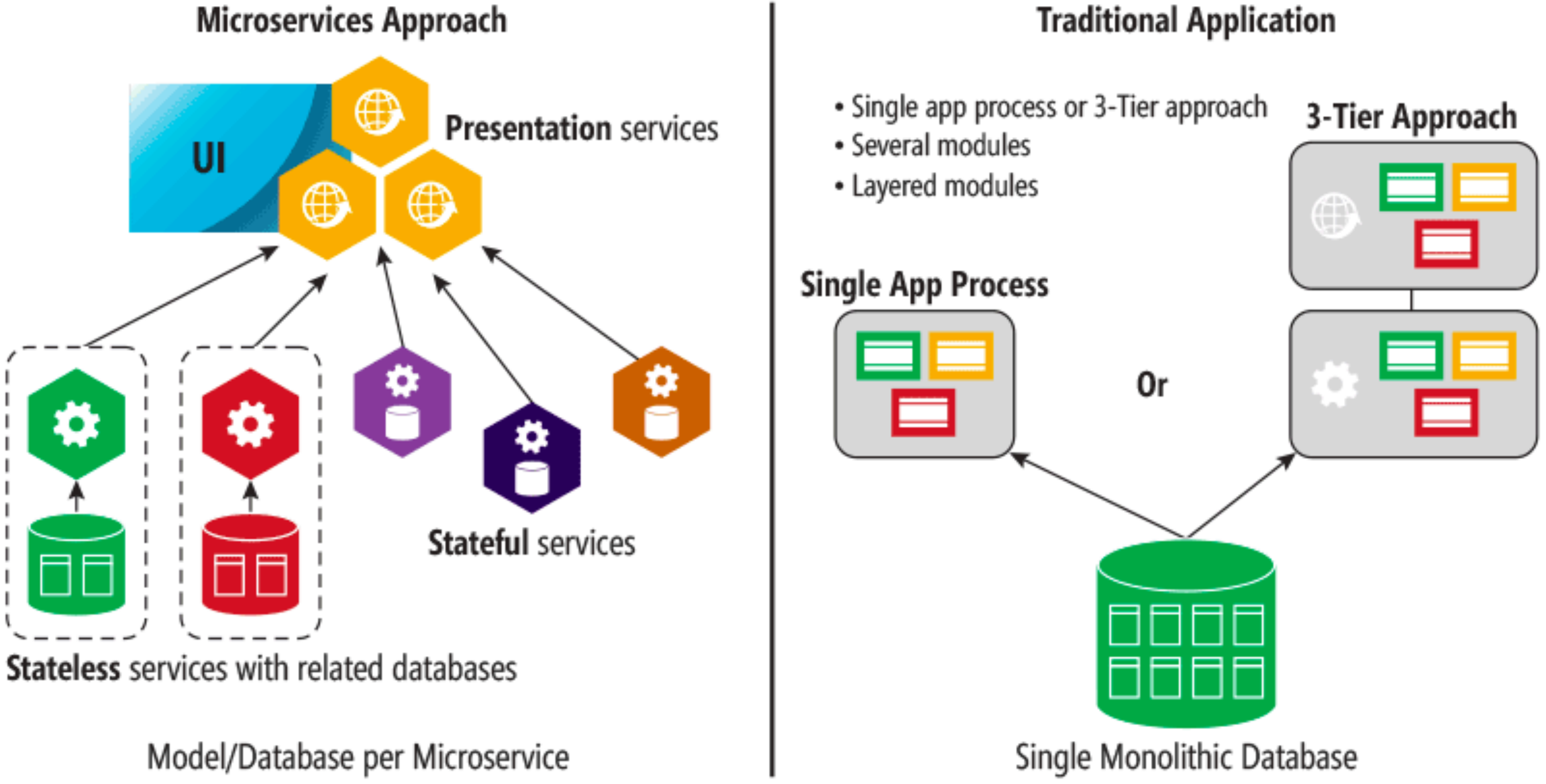
VS.

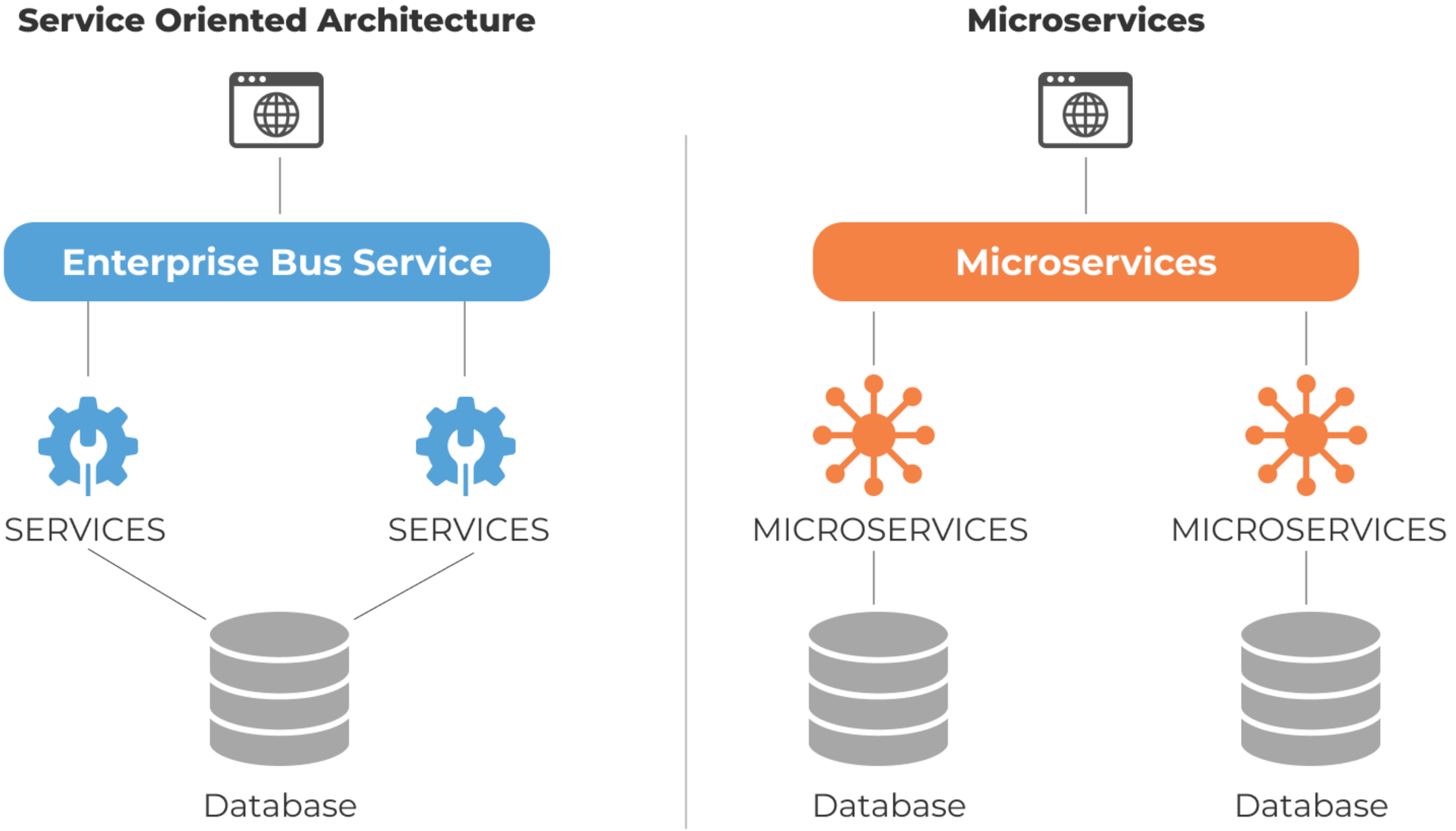
Traditional Approach

A traditional application (Web app or large service) usually has most of its functionality within a single process (usually internally layered, though).

And scales by cloning the whole app on multiple servers/VMs/containers.



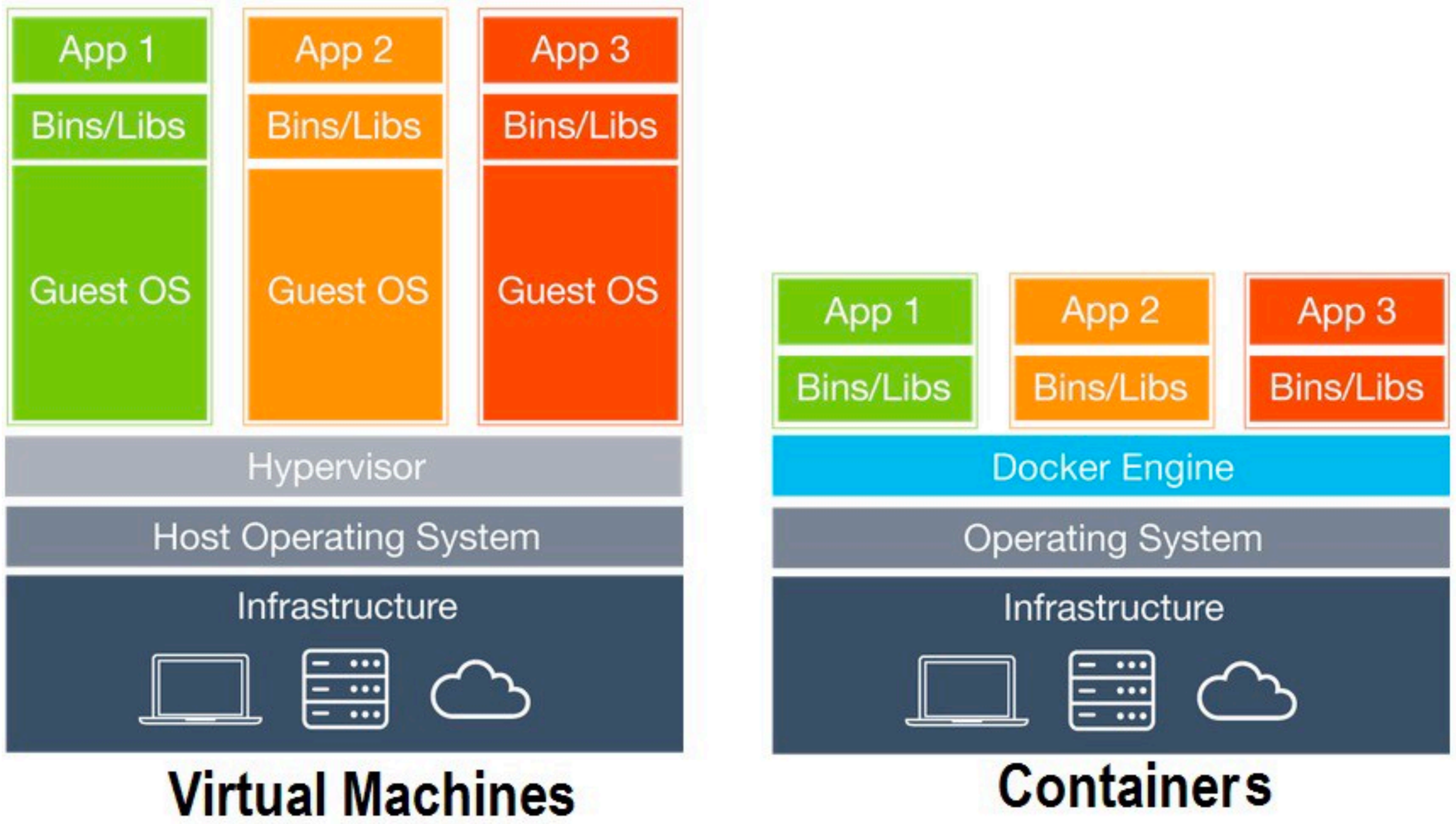

Microservices Approach
UI
Presentation services
Stateful services
Stateless services with related databases
Model/Database per Microservice
Traditional Application
• Single app process or 3-Tier approach
• Several modules
• Layered modules
3-Tier Approach
Single App Process
Or
Single Monolithic Database

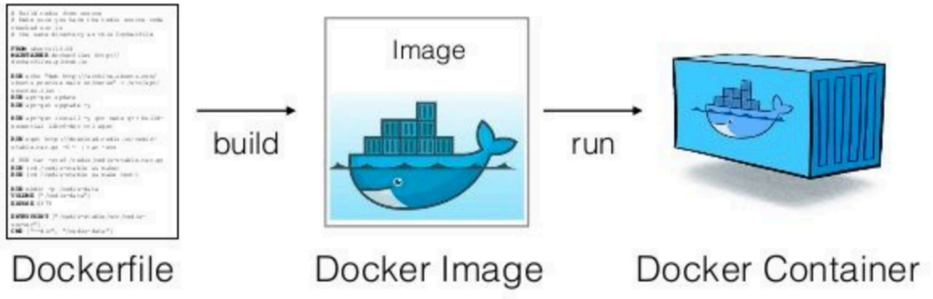

Service Oriented Architecture
Enterprise Bus Service
SERVICES
SERVICES
Database
Microservices
Microservices
MICROSERVICES
MICROSERVICES
Database
Database

¿Como implementar una arquitectura con micro-servicios?

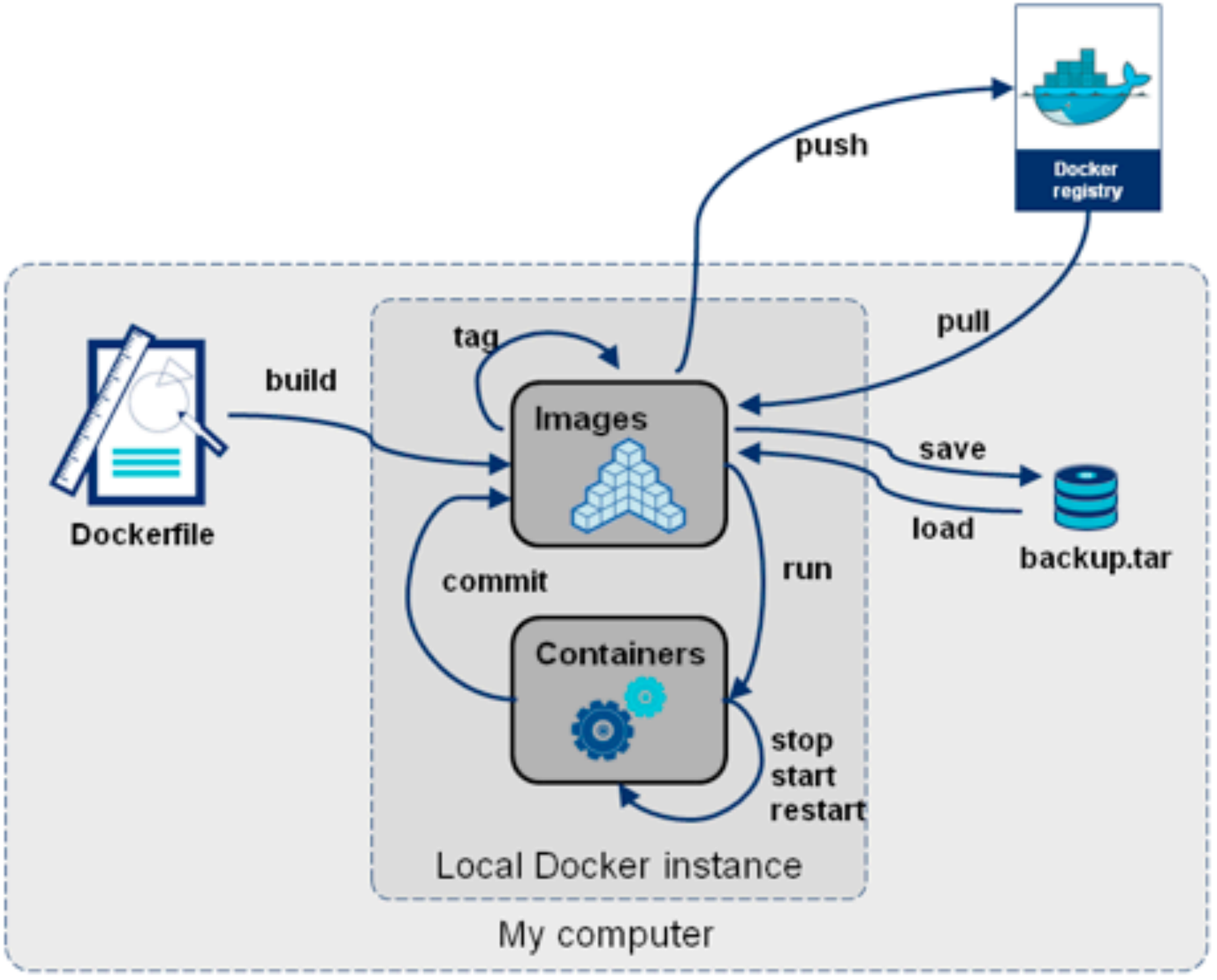
Juan Pablo Sandoval

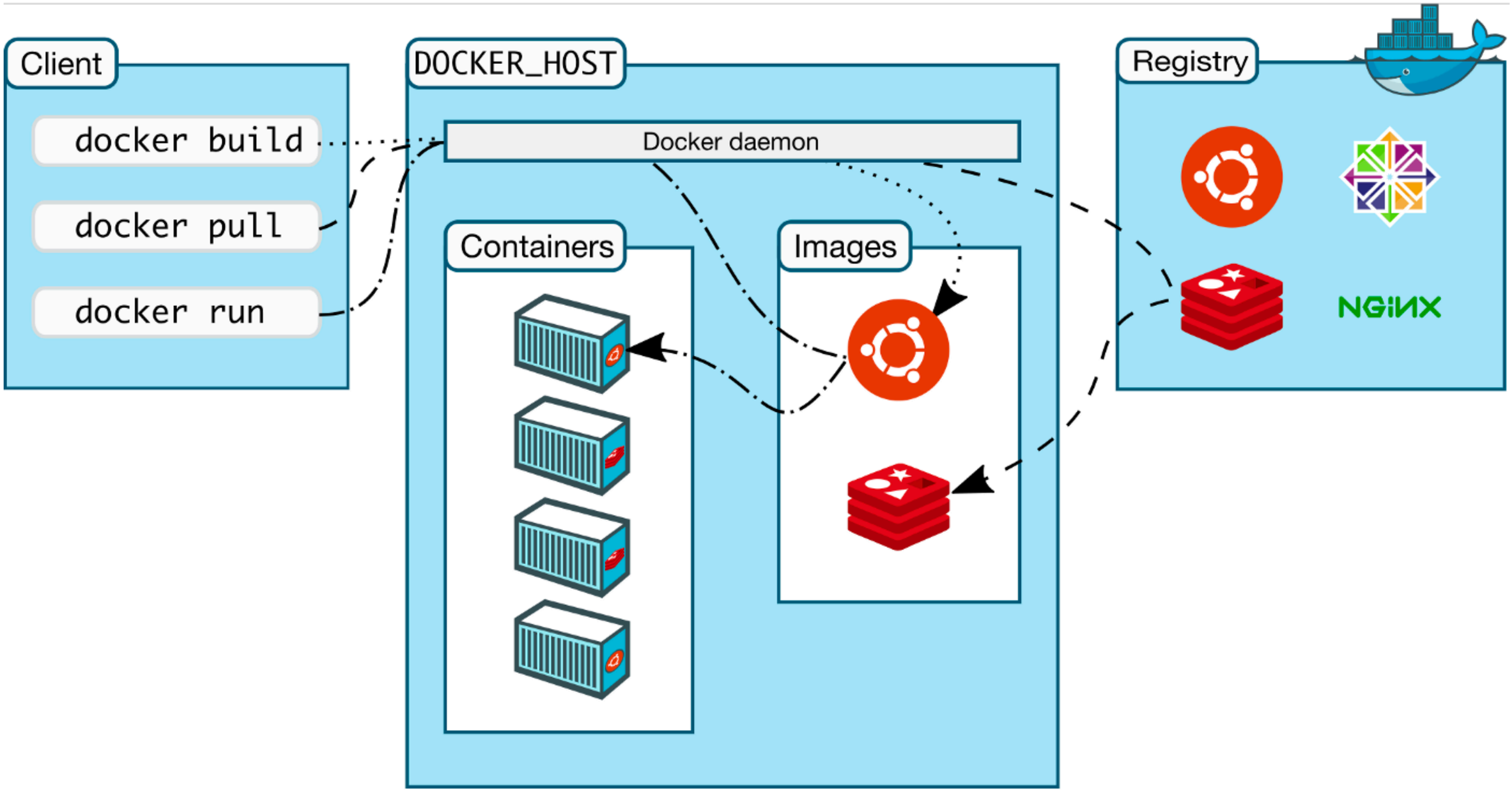
Containers (Docker)



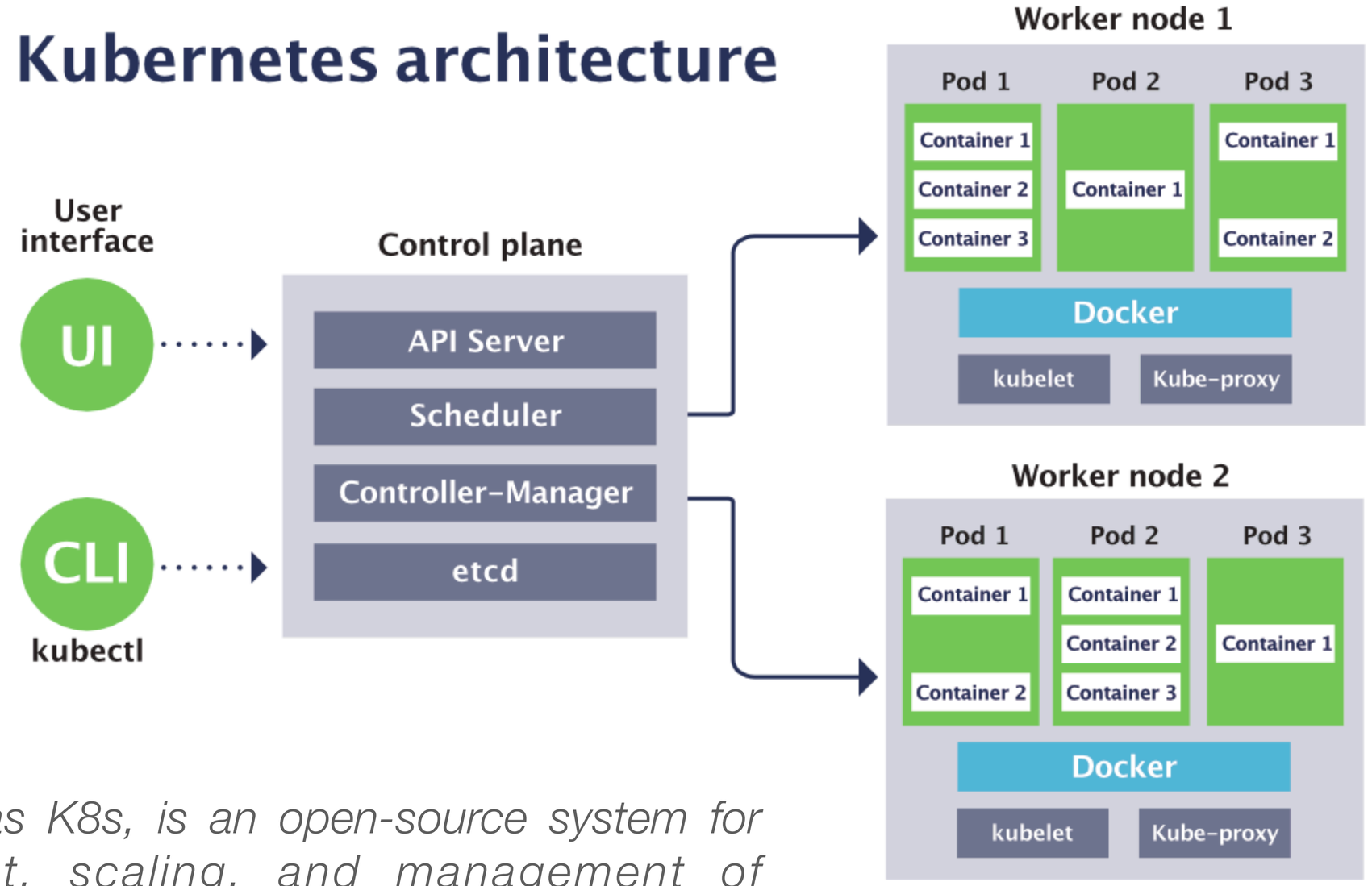



```
# Especificamos la imágen de la que se heredará

FROM python:3.6-alpine

# Definimos algunas variables de ambiente

ENV LIBRARY_PATH=/lib:/usr/lib

ENV PYTHONUNBUFFERED 1

# Ejecutamos algunos comandos para aprovisionar la imágen

RUN mkdir /code

WORKDIR /code

ADD requirements.txt /code/

RUN pip install -r requirements.txt

# Copiamos el código

ADD . /code/

# Exponemos el puerto por donde se interactuará con la aplicación

EXPOSE 8000

# Definimos el comando de entrada a la aplicación

CMD ["python", "manage.py", "runserver", "0.0.0.0:8000"]
```

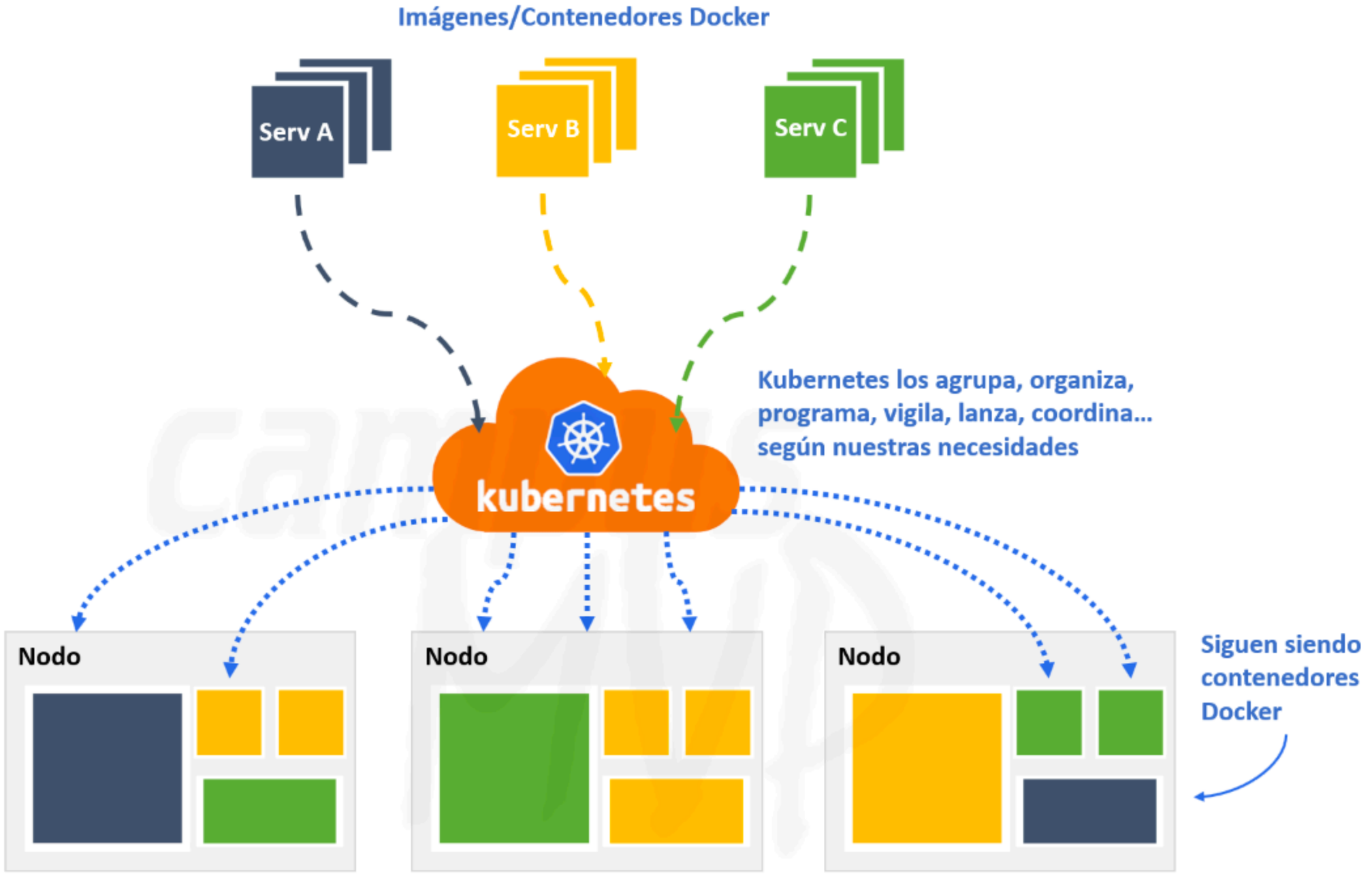

Docker registry
push
pull
tag
build
Images
save
load
backup.tar
Dockerfile
commit
run
Containers
stop
start
restart
Local Docker instance
My computer


Client
docker build
docker pull
docker run
DOCKER_HOST
Docker daemon
Containers
Images
Registry
NGINX

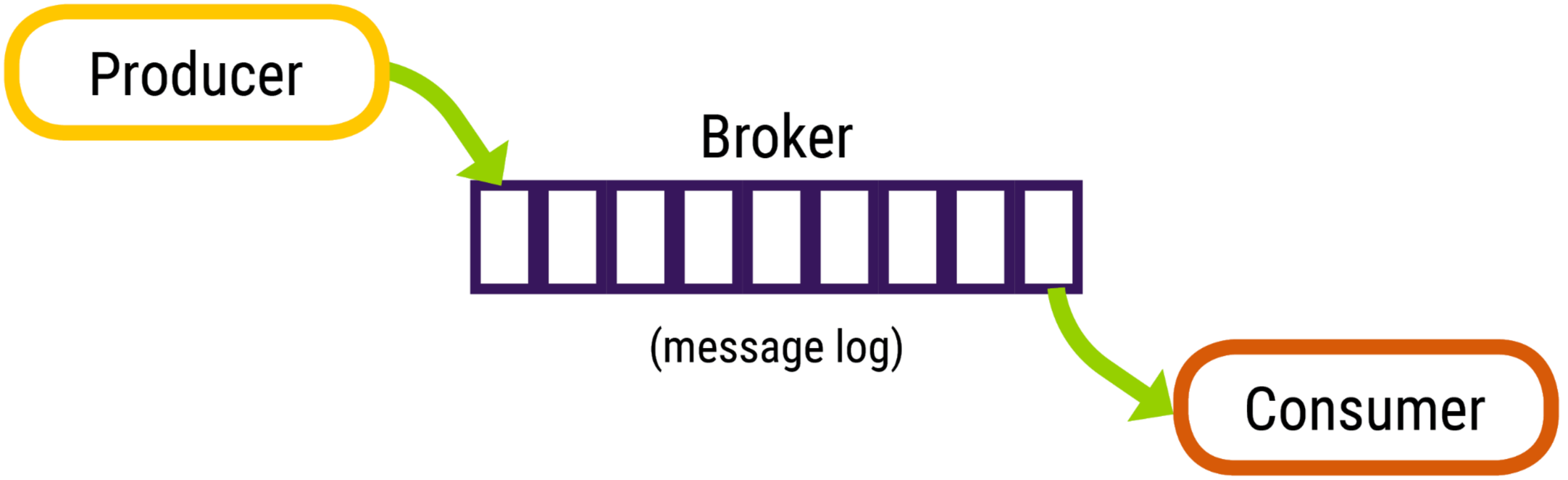
Kubernetes architecture

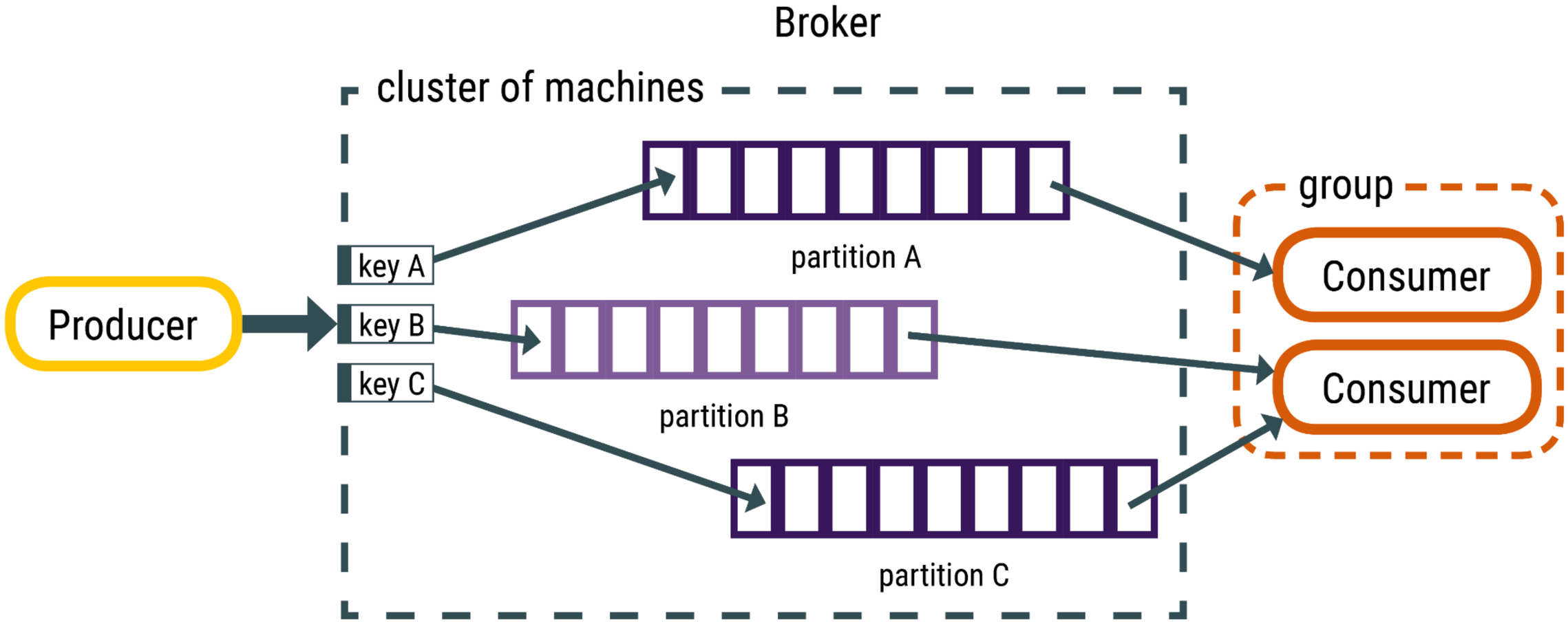


Kubernetes, also known as K8s, is an open-source system for automating deployment, scaling, and management of ***containerized*** *applications.*

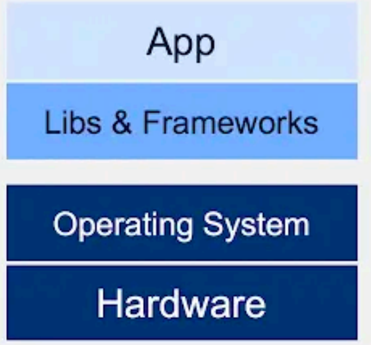

Imágenes/Contenedores Docker
Serv A
Serv B
Serv C
kubernetes
Kubernetes los agrupa, organiza,
programa, vigila, lanza, coordina...
según nuestras necesidades
Nodo
Nodo
Nodo
Siguen siendo
contenedores
Docker


ORACLE
salesforce
workday
kafka
SAP

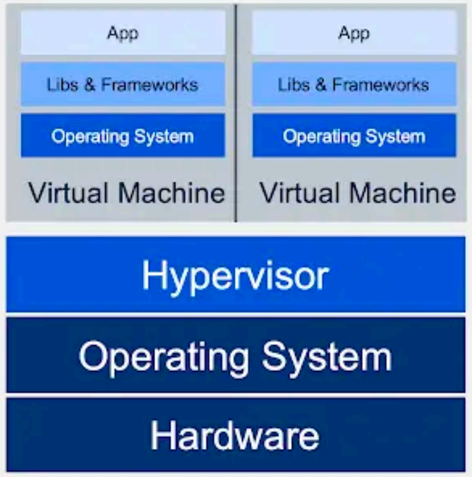

Producer
Broker
(message log)
Consumer


Broker
cluster of machines
Producer
key A
key B
key C
partition A
partition B
partition C
group
Consumer
Consumer

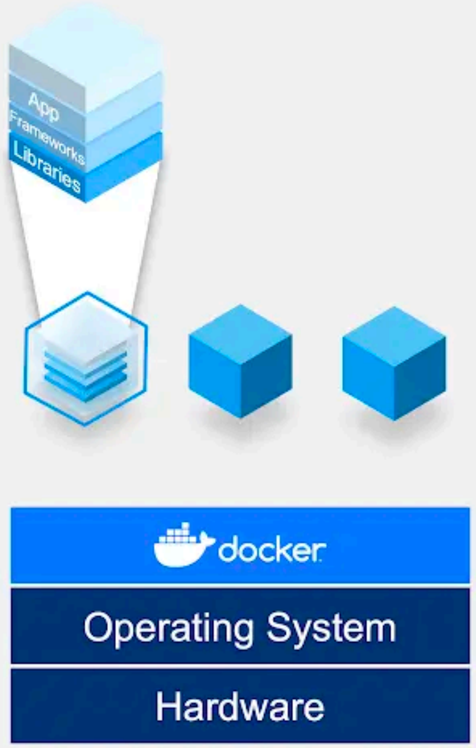
Kubernetes & Docker work together to build & run containerized applications



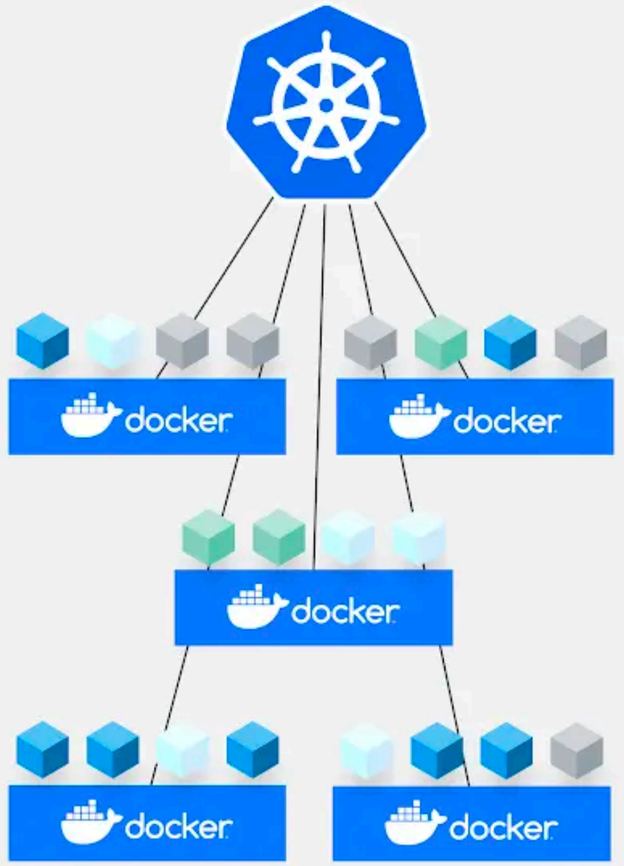
Traditional Deployment



Virtualized Deployment



Container Deployment



Kubernetes Deployment