

AZURE LIVE

PREPARADO POR: JOSE YAPUR

COMENZAMOS 8 PM EN PUNTO



CLASE 2

CONTENEDORES, MICROSERVICIOS Y KUBERNETES

¿POR QUÉ SON IMPORTANTES LOS CONTENEDORES?

01

Evitar casarme con un proveedor

02

Mi aplicación funcione en cualquier nube o incluso en mi DC

03

No hay especialistas en el mercado

CONCEPTOS BÁSICOS DE DOCKER



Image

The basis of a Docker container. The content at rest.



Container

The image when it is 'running.' The standard unit for app service



Engine

The software that executes commands for containers. Networking and volumes are part of Engine. Can be clustered together.



Registry

Stores, distributes and manages Docker images

AZURE CONTAINERS

Container
Instance

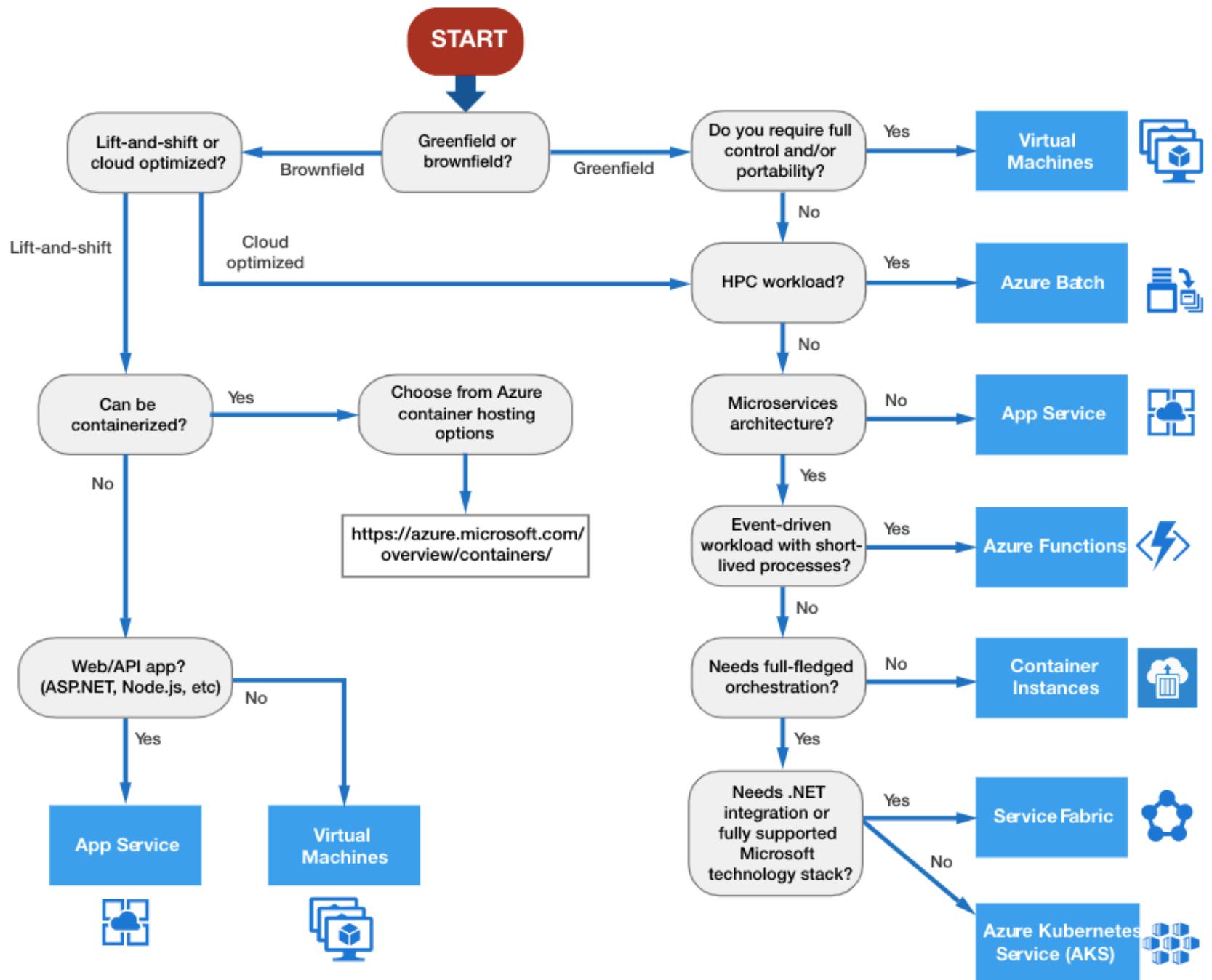
Web App for
Containers

Function
App

Kubernetes
Service

Leyenda	
	APaaS
	KaaS
	Serverless

Azure Container Registry



Virtual

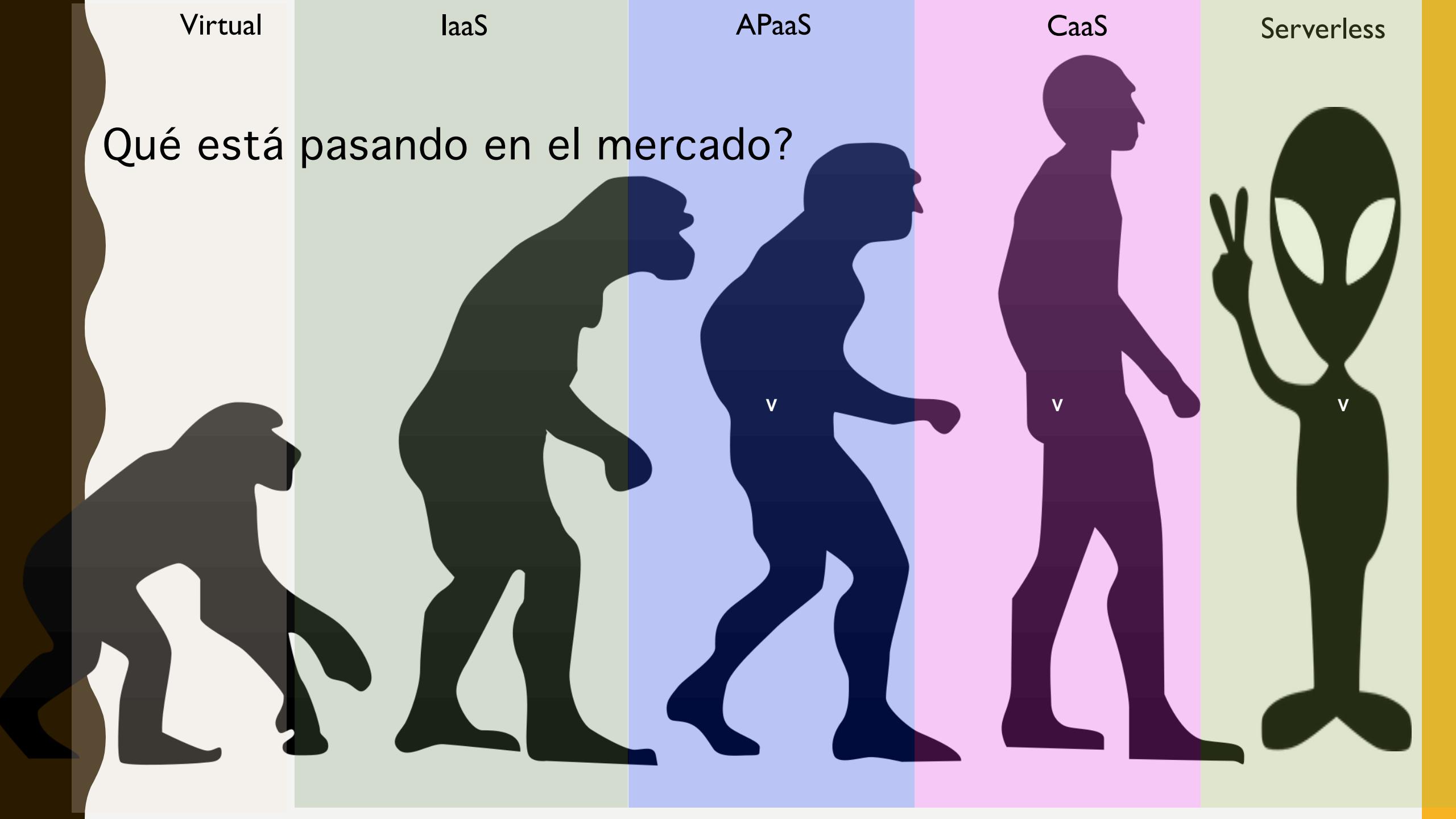
IaaS

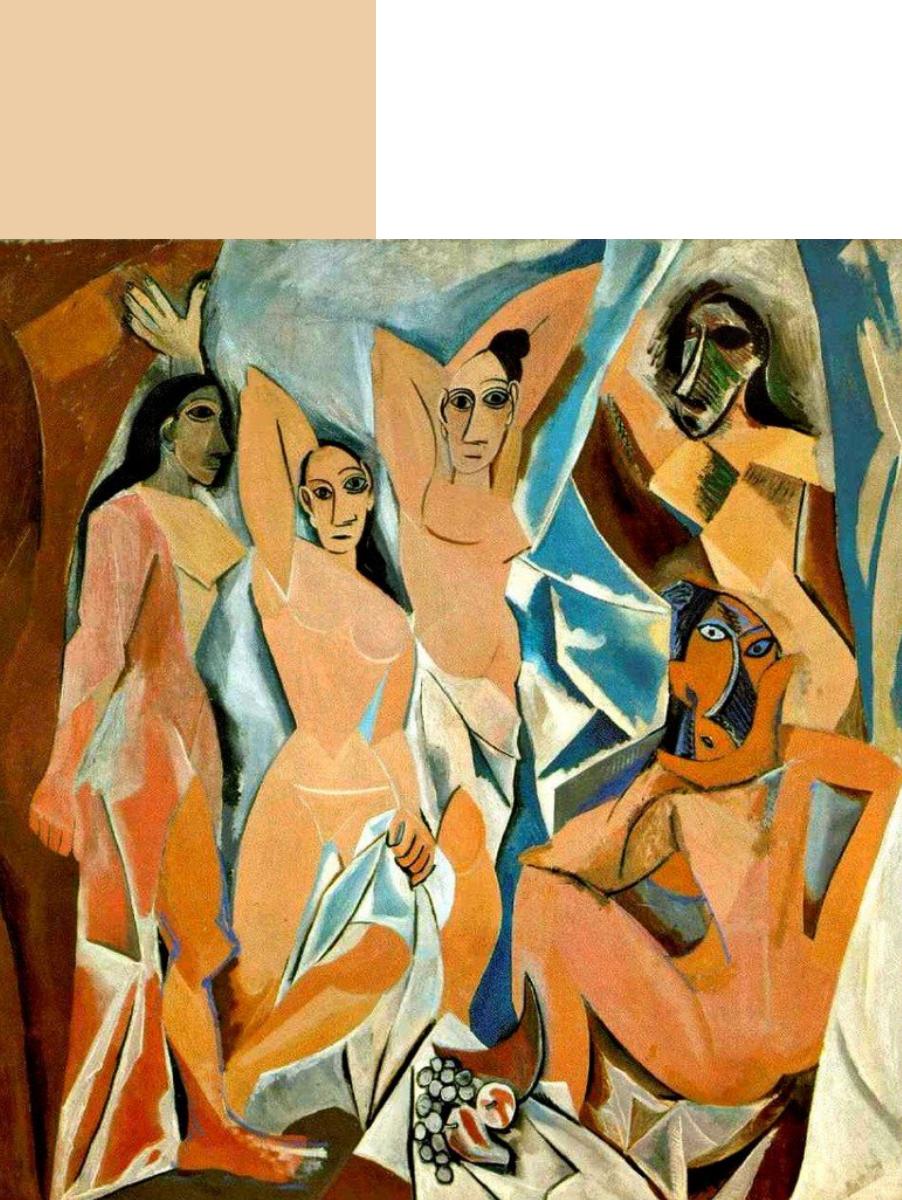
APaaS

CaaS

Serverless

Qué está pasando en el mercado?





Microservicio != Contenedor

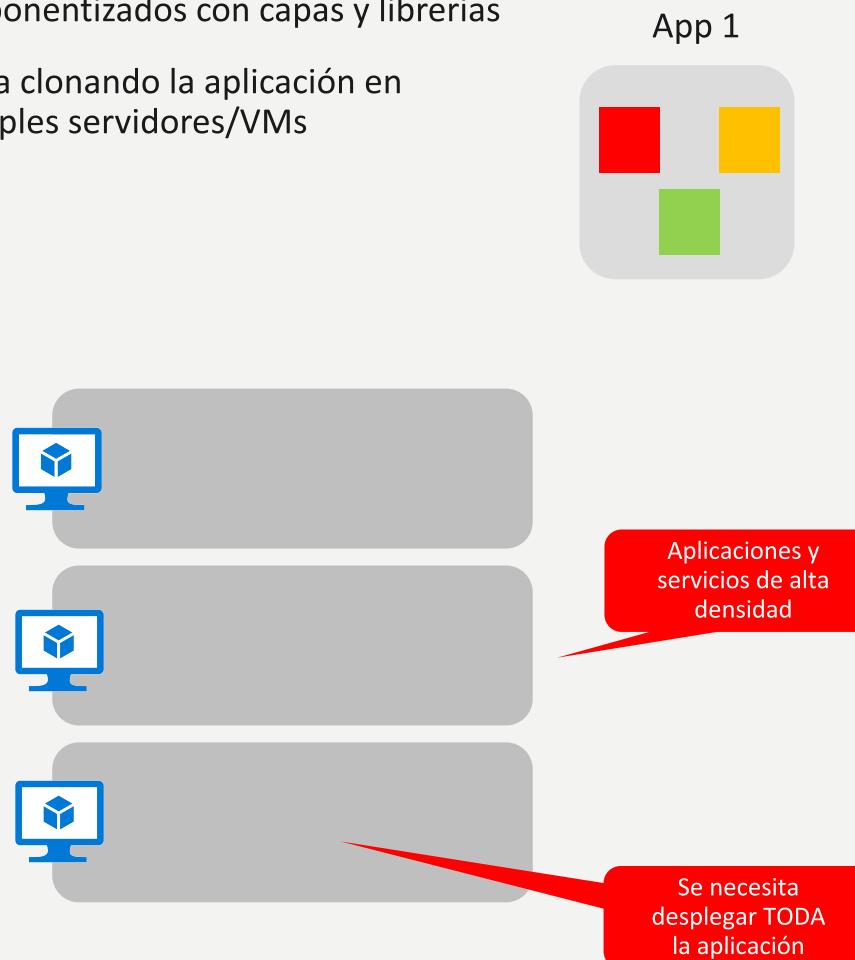
Es un *estilo arquitectural* que estructura una aplicación como un *conjunto de servicios* que son:

- Altamente mantenibles y comprobables
- Independientemente desplegables
- Débilmente acoplado
- Organizadas alrededor de capacidades de negocio

- Chris Richardson

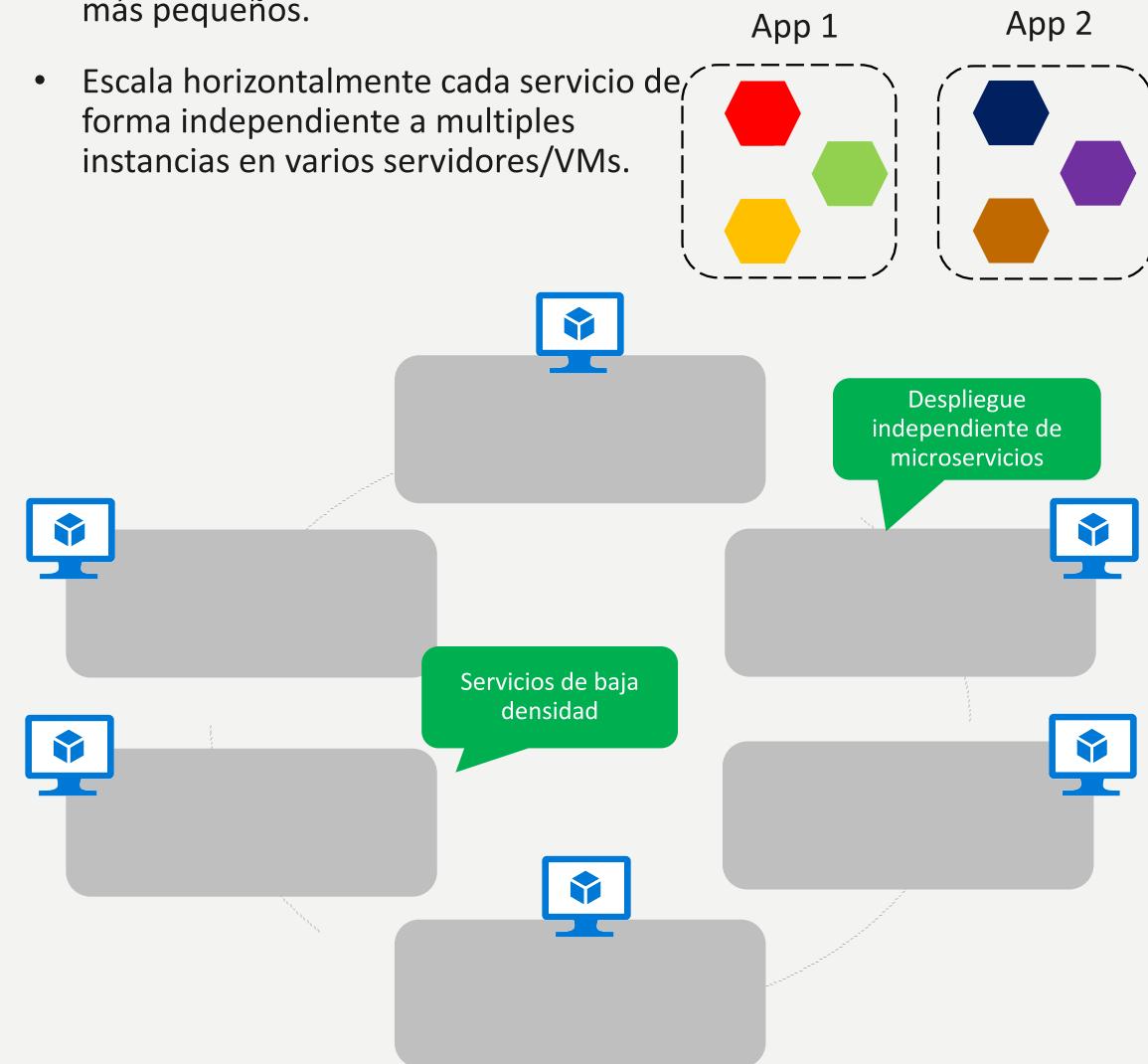
Enfoque de aplicaciones tradicionales

- Una aplicación tradicional tiene su funcionalidad entre pocos procesos componentizados con capas y librerías
- Escala clonando la aplicación en múltiples servidores/VMs



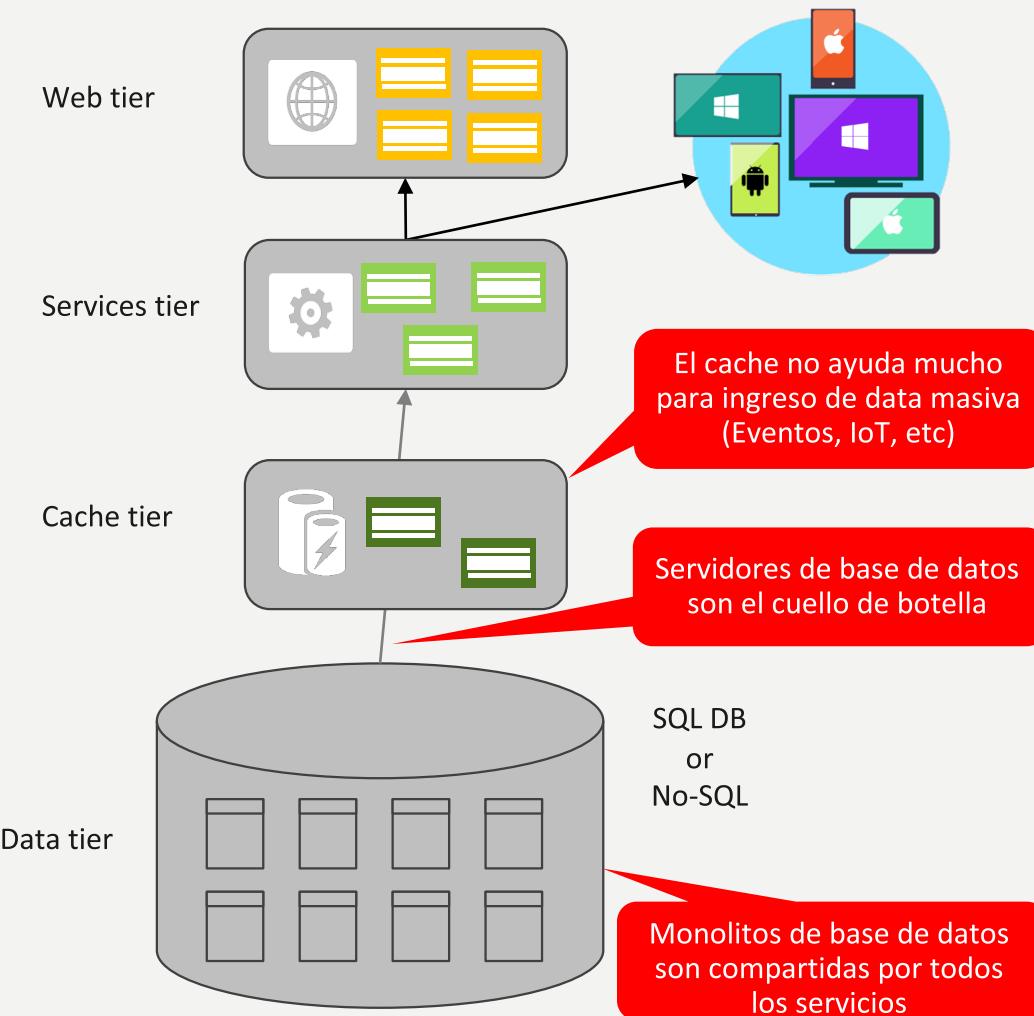
Nuevo paradigma de Microservicios

- Una aplicación de microservicios separa la funcionalidad en servicios separados más pequeños.
- Escala horizontalmente cada servicio de forma independiente a múltiples instancias en varios servidores/VMs.



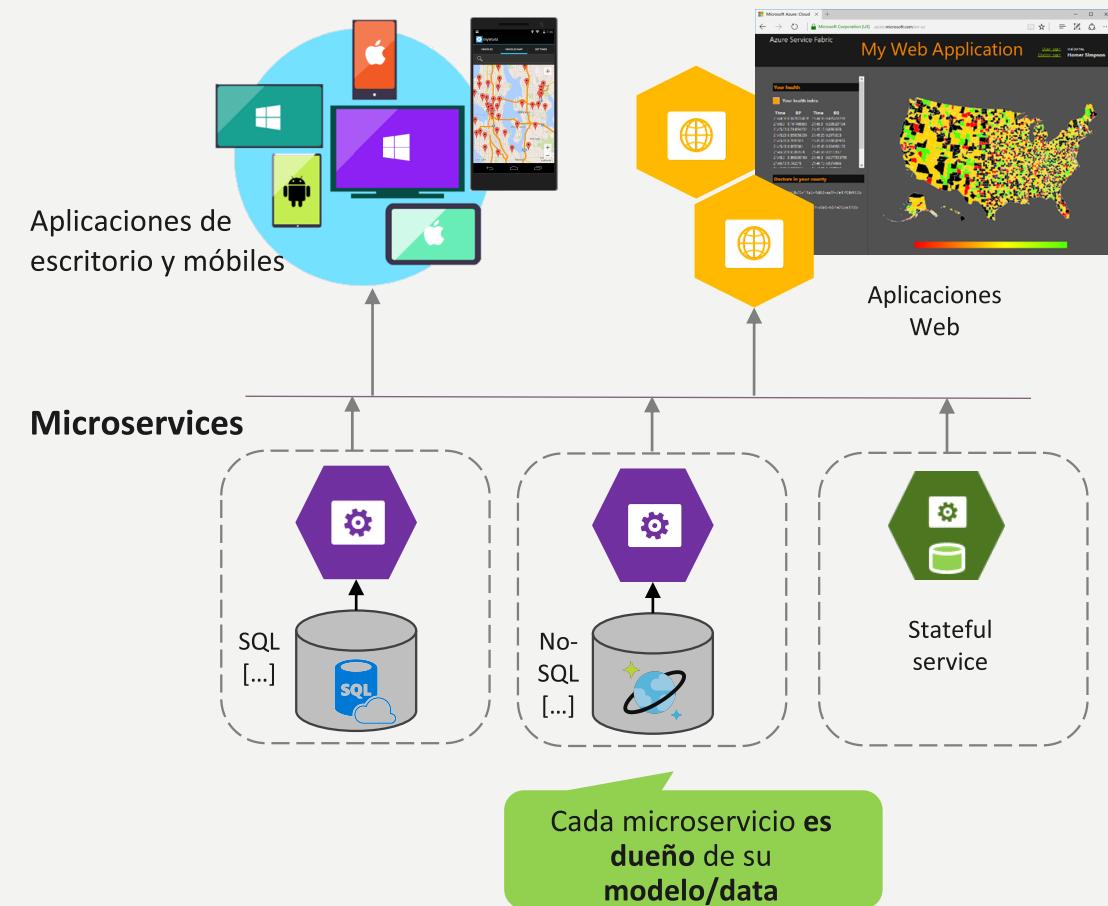
Enfoque de aplicaciones tradicionales

- Un solo Monolito de base de datos.
- Tiers de tecnologías específicas

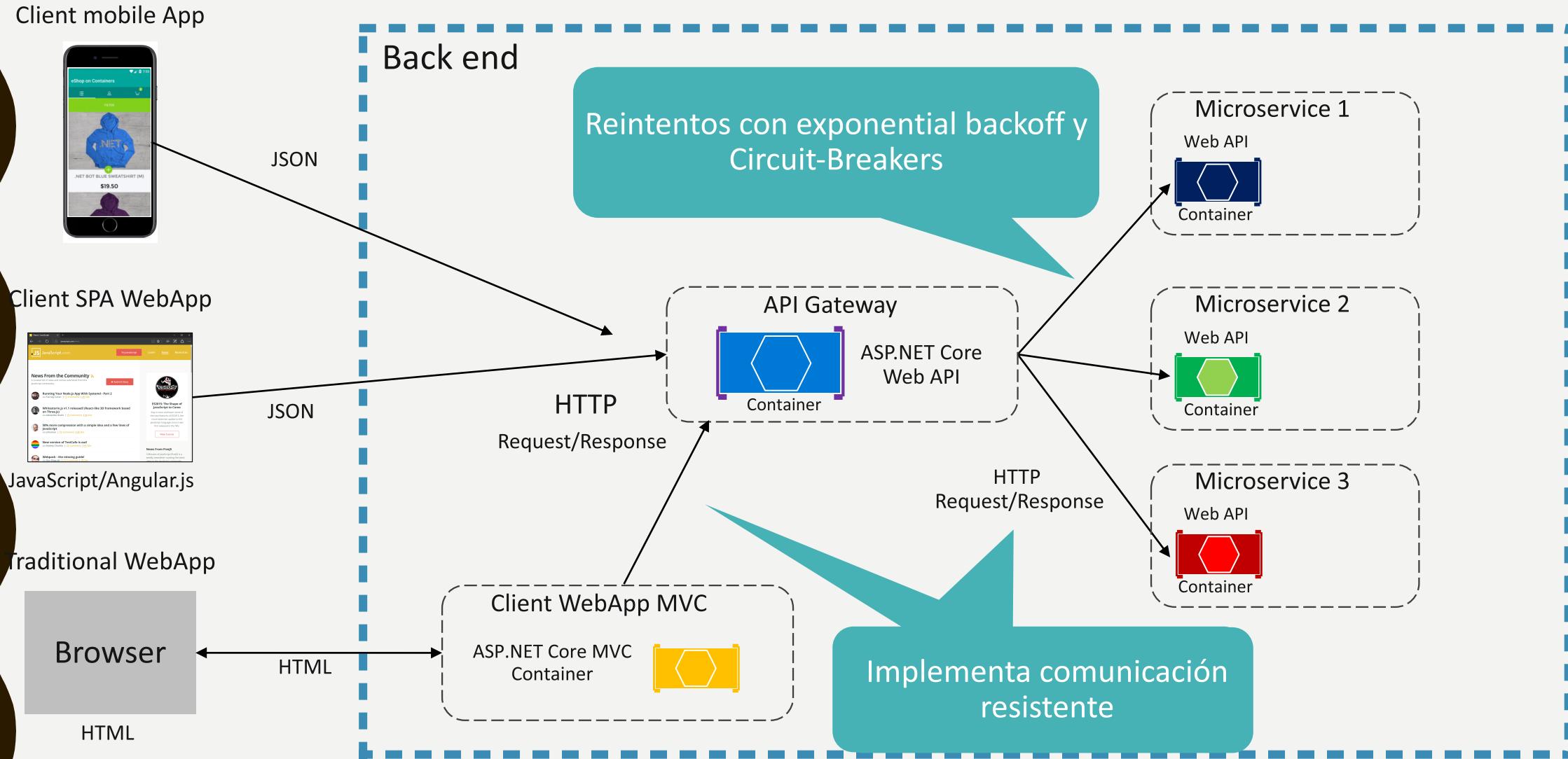


Nuevo paradigma de Microservicios

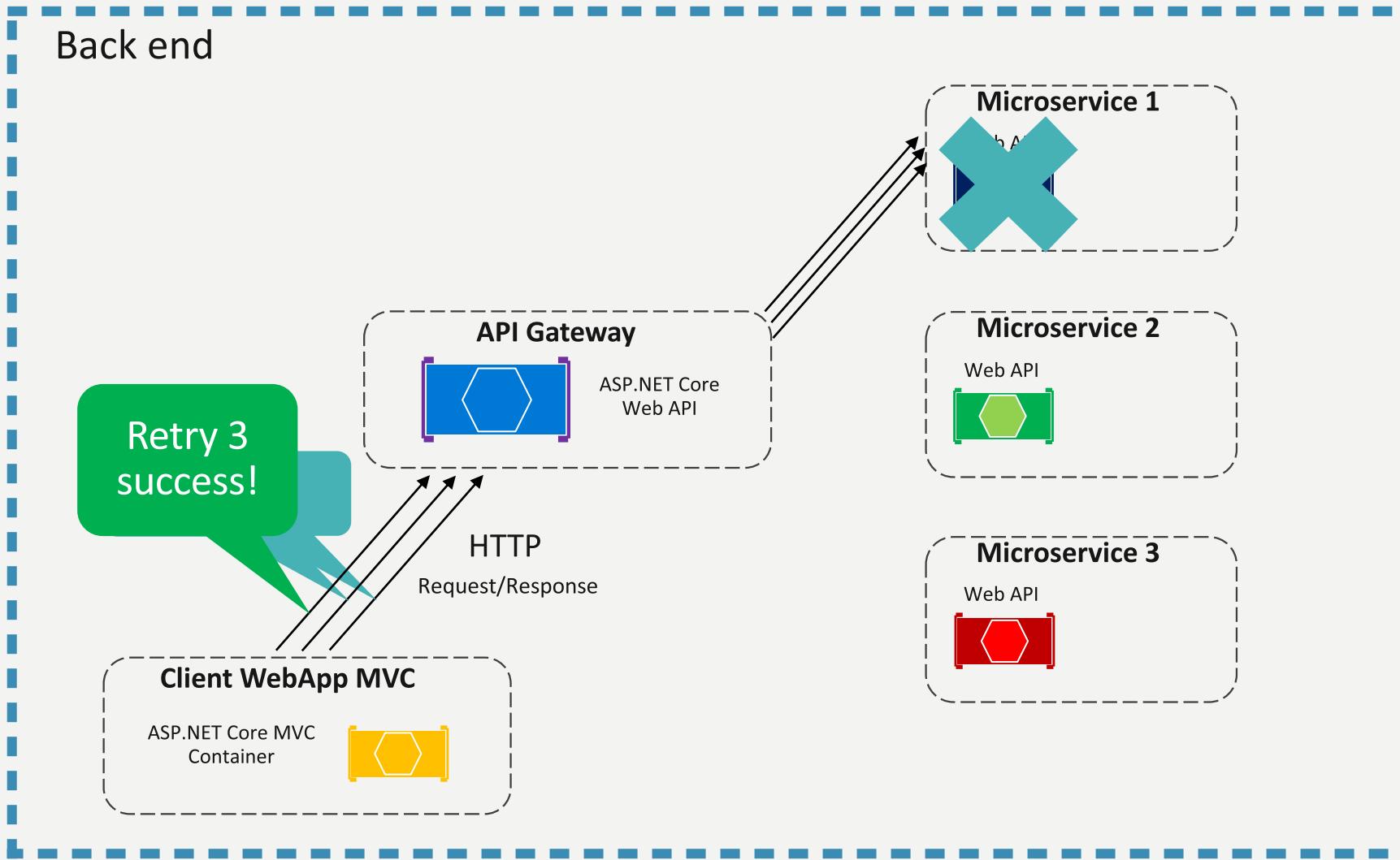
- Grafo de microservicios interconectados
- Estado típicamente acotado al microservicio
- Almacenamiento remote de data fria.



CONSTRUYENDO APLICACIONES CLOUD RESISTENTES

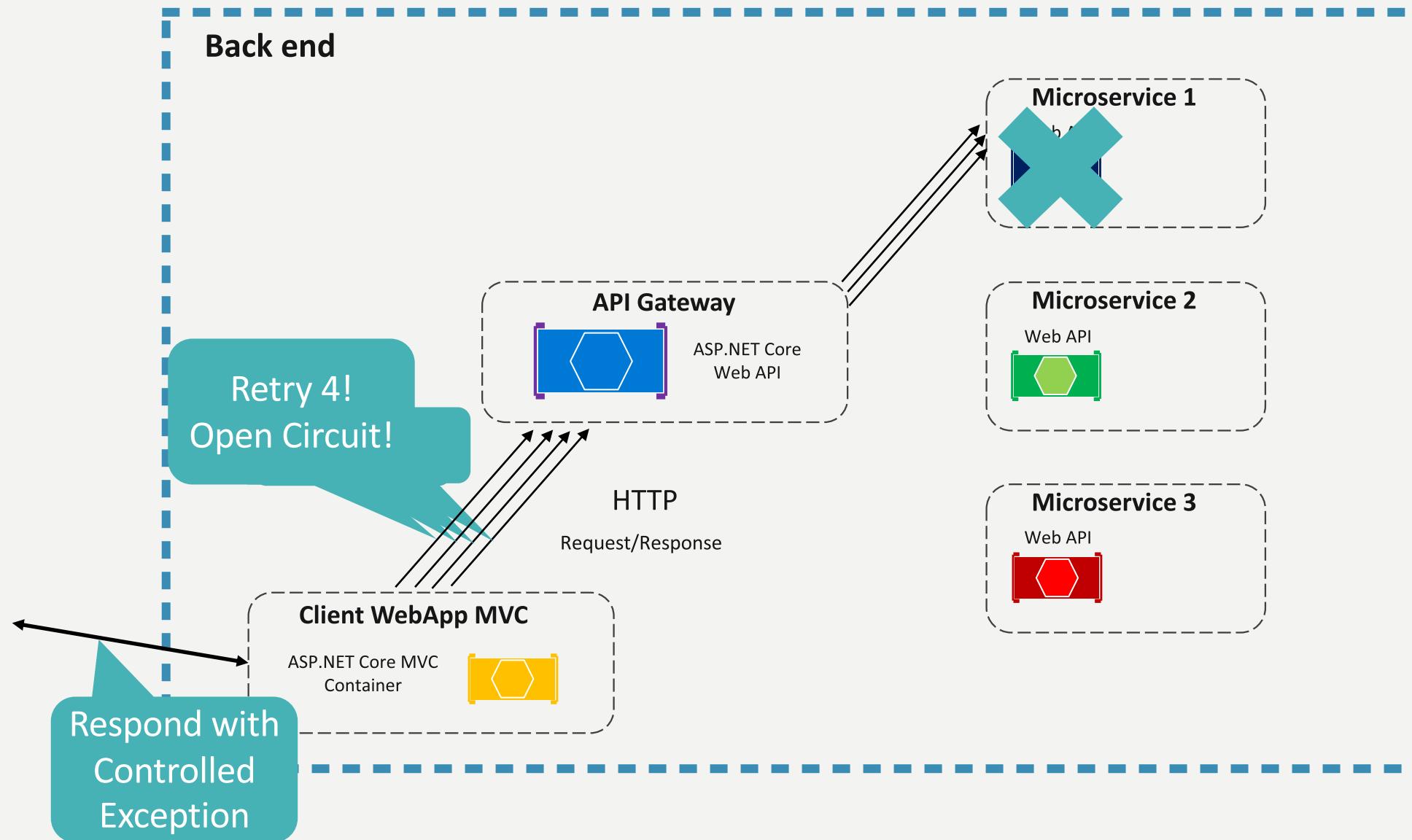


REINTENTOS CON EXPONENTIAL BACKOFF



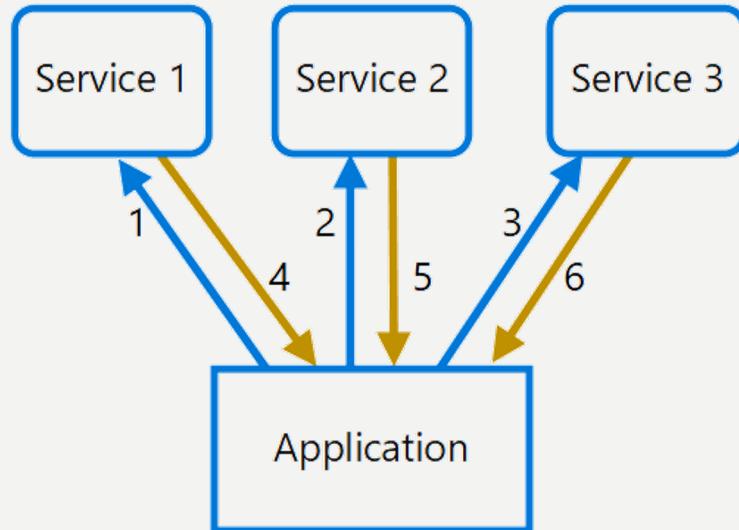
REINTENTOS CON EXPONENTIAL BACKOFF

+ circuit breaker

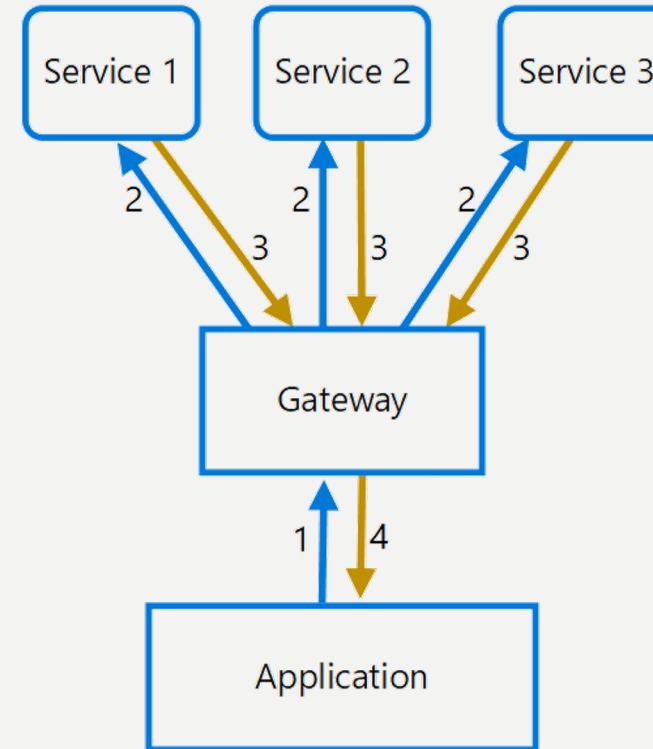


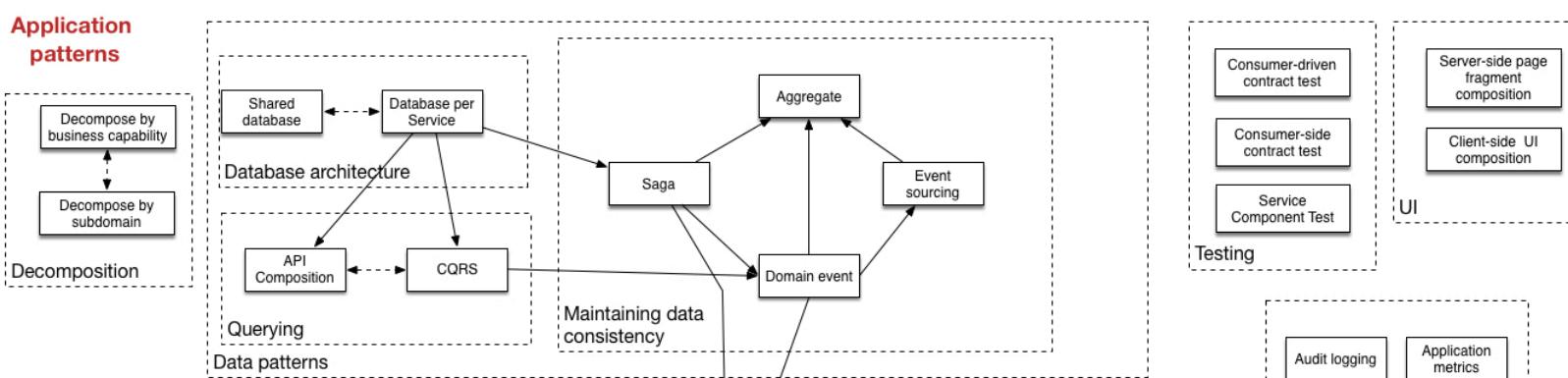
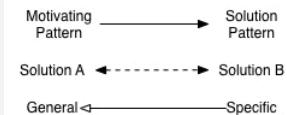
PATRÓN DE DISEÑO API GATEWAY

Comunicación directa al microservicio
("Sin uso de API Gateway")

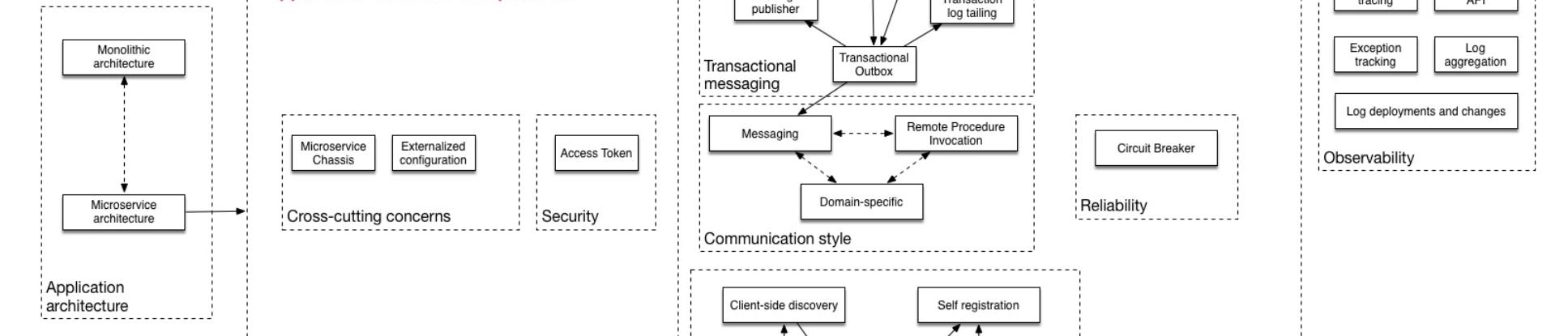


Usando el patron API Gateway

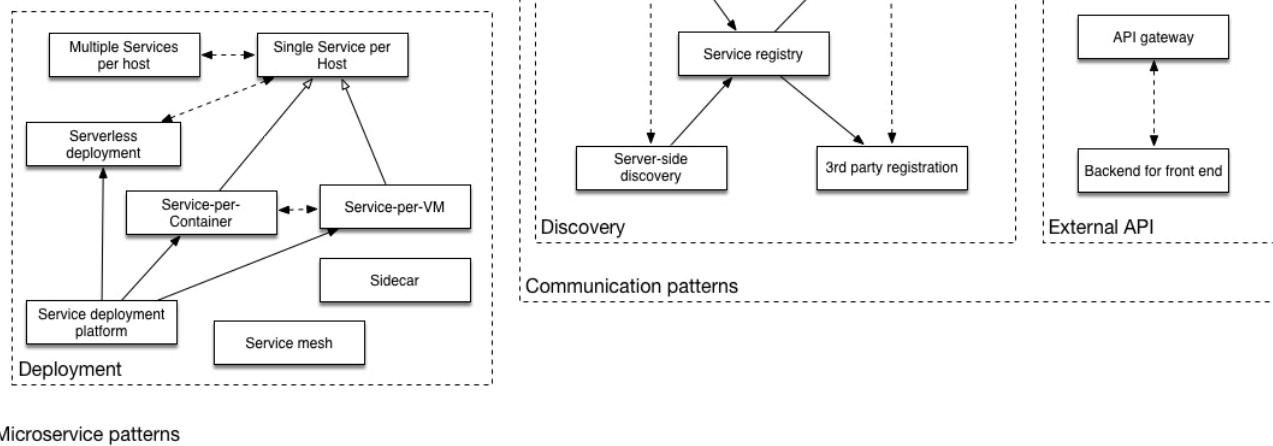




Application Infrastructure patterns



Infrastructure patterns

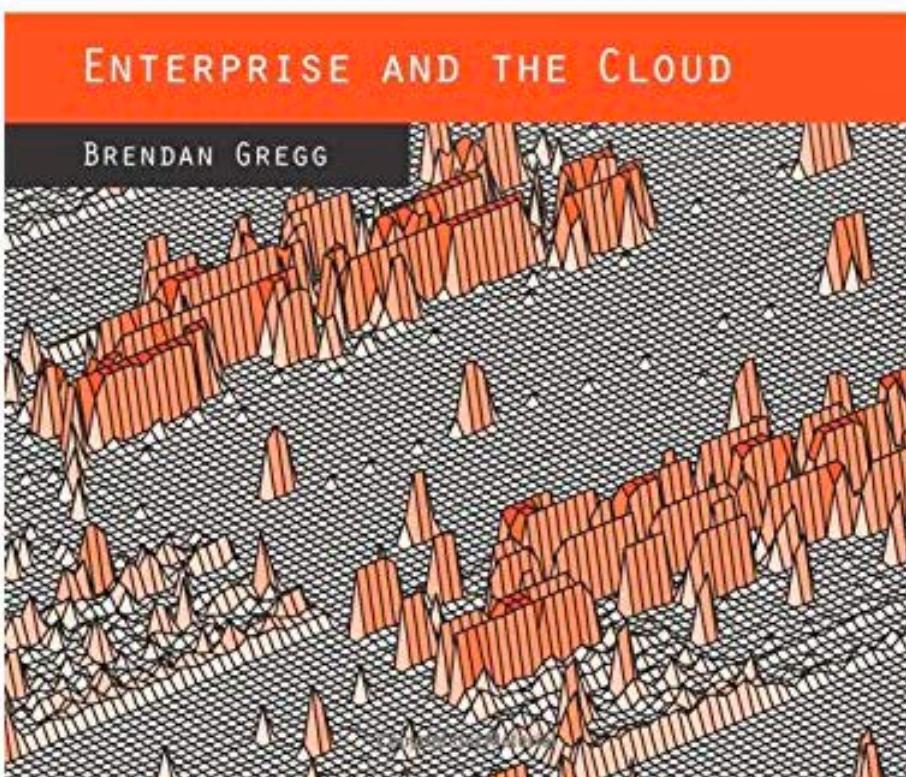


El desempeño de un **sistema distribuido** es el desempeño combinado de sus **servicios colaborativos** y de sus **enlaces de comunicación**

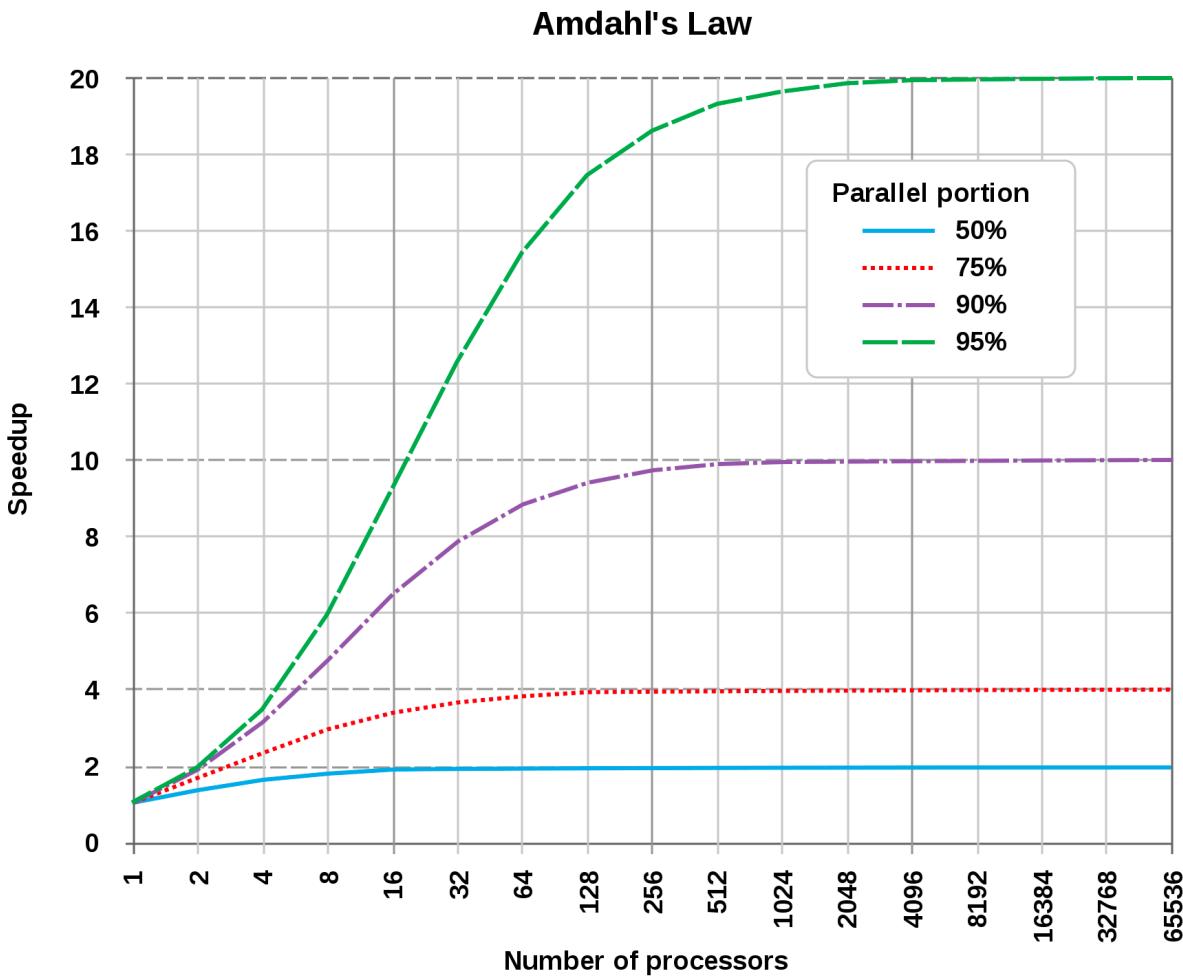
Copyrighted Material



Systems Performance



LEY DE AMDAHL



$$S_{\text{latency}}(s) = \frac{1}{(1-p) + \frac{p}{s}}$$

$$\begin{cases} S_{\text{latency}}(s) \leq \frac{1}{1-p} \\ \lim_{s \rightarrow \infty} S_{\text{latency}}(s) = \frac{1}{1-p}. \end{cases}$$

Donde:

- S_{latency} es la velocidad teórica de ejecución de toda la tarea.
- s es la aceleración de la parte de la tarea que se beneficia del Sistema de recursos mejorado
- p es la proporción de tiempo de ejecución que se beneficia de los recursos originalmente ocupados.

$$S_{latencia}(s) = \frac{1}{(1 - p) + \frac{p}{s}}$$

Aceleración de parte
bajo optimización

Aceleración teórica
máxima del sistema

Porcentaje de tiempo
de ejecución de
parte bajo
optimización

$$\lim_{s \rightarrow \infty} S_{latencia}(s) = \frac{1}{(1 - p)}$$

Aceleración teórica
máxima del sistema

Aceleración de parte
bajo optimización

Porcentaje de tiempo
de ejecución de
parte bajo
optimización

Los microservicios suelen vivir dentro de contenedores



KUBERNETES

- Kubernetes no es para todo.
 - Kubernetes no arregla los problemas de tu infraestructura o aplicación
 - Kubernetes no agrega valor sin CI/CD
-
- <https://github.com/jyapurv/azure-workshop>

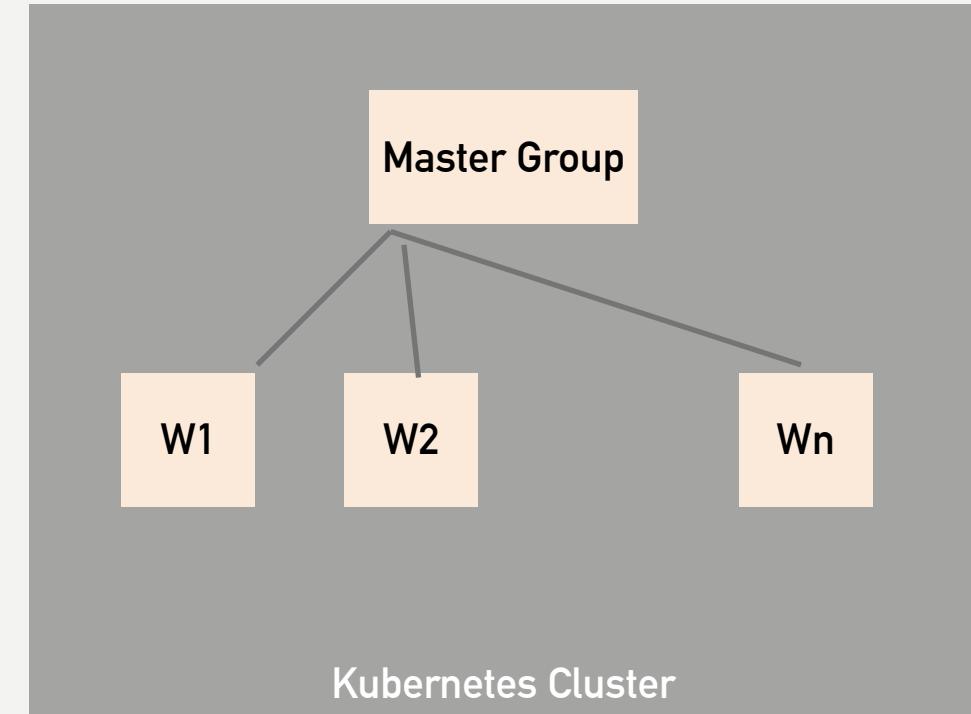
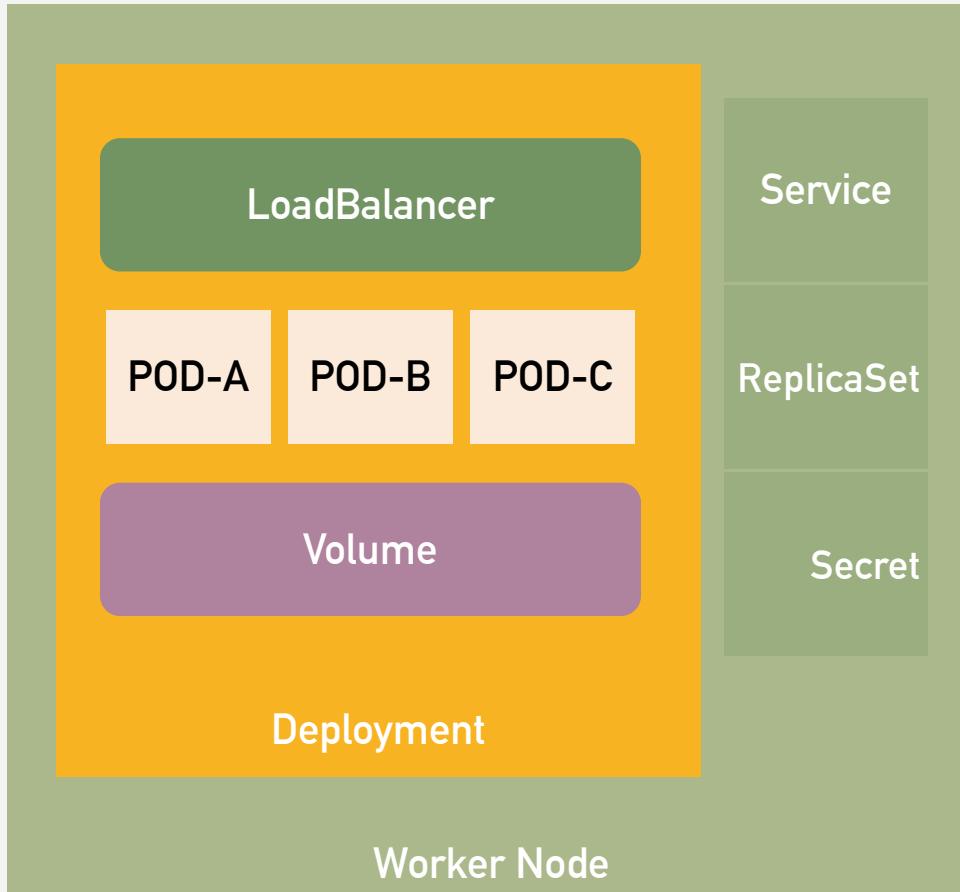
Dex
@dexhorty

Follow

Deployed my blog on Kubernetes

Retweets 1,873 Likes 3,456

ESTRUCTURA



EL KUBERNETES MANIFEST (AKA. EL YAML)

- Es un archivo YAML que contiene todas las instrucciones que necesita el orquestador para realizar acciones.
- Cuando se trata de aplicaciones, estas instrucciones se llaman “Deployments”.
 - Escalamiento: ReplicaSet
 - Servicios: Services

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels:
    app: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx
          image: nginx:1.7.9
          ports:
            - containerPort: 80
```