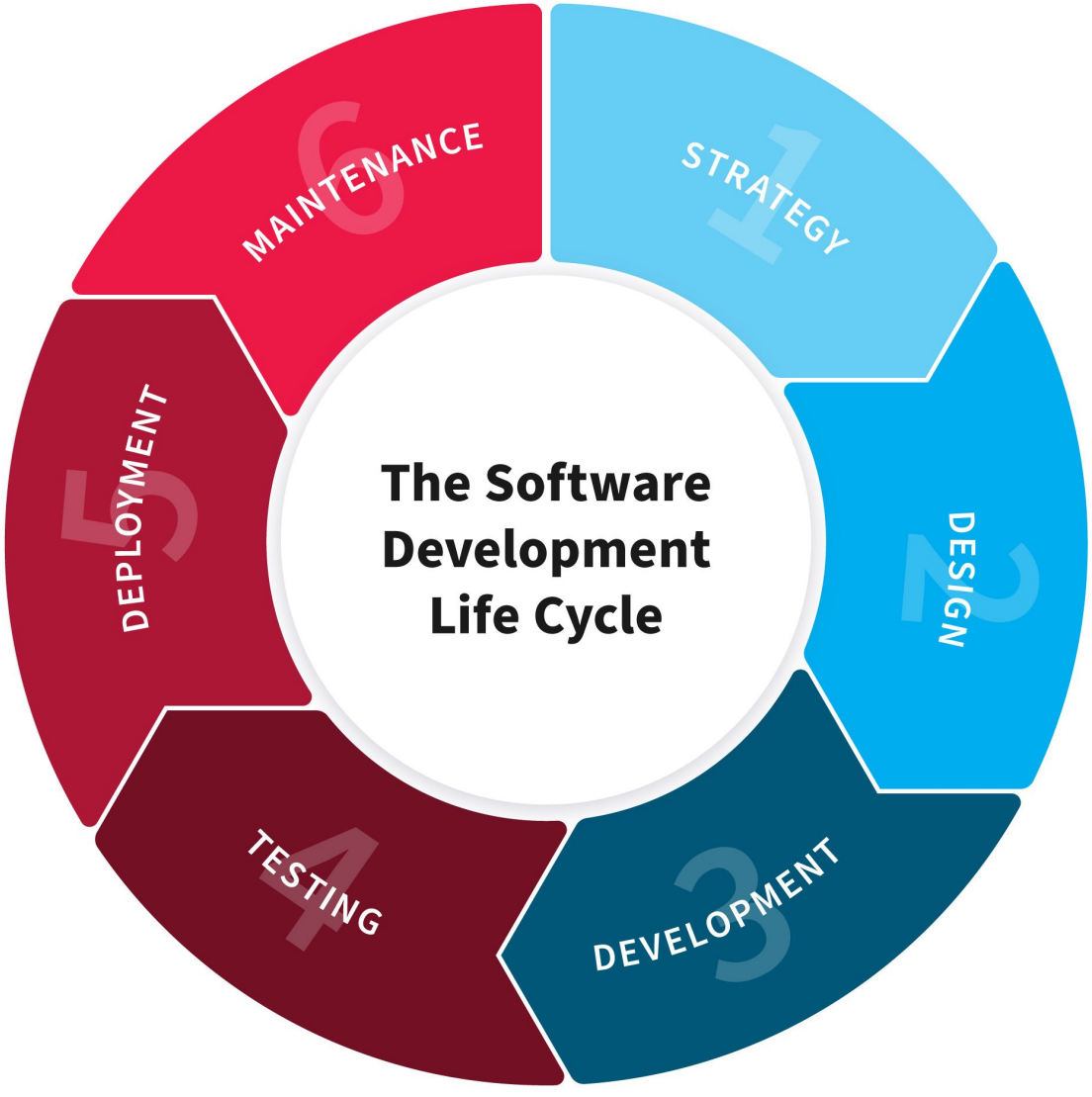
# ¿Qué debo considerar para que mi aplicación sea “apta nube”?

MIKRO WAYS

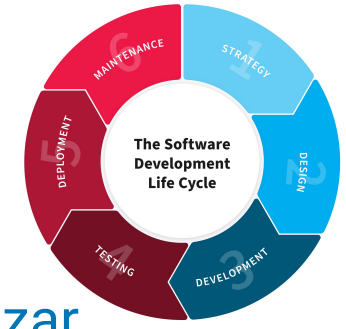
Lic. Christian Adrián Rodríguez

# Ciclo de vida del desarrollo


The Software Development Life Cycle
1
STRATEGY
2
DESIGN
3
DEVELOPMENT
4
TESTING
5
DEPLOYMENT
6
MAINTENANCE

MIKRO WAYS

# Ciclo de vida del desarrollo




- **Estrategia:** es la concepción de una app. Antes de empezar se debe analizar la audiencia, demografía, patrones de comportamiento: ***ingeniería social***. ¿Hay competidores? Si es viable, comienza el relevamiento y análisis funcional.
- **Diseño:** definiciones sobre la arquitectura, dispositivos. Decisiones de diseño. Prototipos visuales. Alcance de MVP
- **Desarrollo:** en base a la arquitectura elegida, se definen repositorios, lenguajes y frameworks. Siempre es aconsejable utilizar ***metodologías ágiles*** y ***TDD***.
- **Testing:** no confundir con correr tests que deberían correrse en desarrollo. Estos tests son de aceptación, seguridad, integración, stress.

MIKRO WAYS

# Ciclo de vida del desarrollo

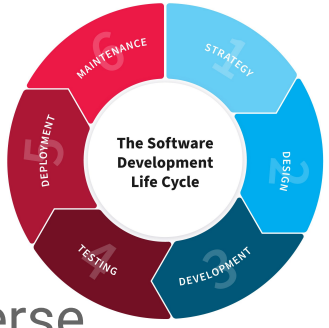

- **Deployment:** despliegue de la aplicación y su actualización entre releases. Estos despliegues pueden realizarse en diferentes ambientes.
- **Mantenimiento:** asegurar que la aplicación se comporta como se espera ante el uso. Surgen errores, nuevas funcionalidades que motivan nuevos releases y despliegues. Conocer el comportamiento amerita relevar datos sobre el estado y compararlo en diferentes momentos con gráficas.

MIKRO WAYS

# ¿Con cuáles hitos estamos menos cómodos?

MIKRO WAYS

# Ciclo de vida del desarrollo




- **Testing:** no confundir con correr tests que deberían correrse en desarrollo. Estos tests son de aceptación, seguridad, integración, stress.
- **Deployment:** despliegue de la aplicación y su actualización entre releases. Estos despliegues pueden realizarse en diferentes ambientes.
- **Mantenimiento:** asegurar que la aplicación se comporta como se espera ante el uso. Surgen errores, nuevas funcionalidades que motivan nuevos releases y despliegues. Conocer el comportamiento amerita relevar datos sobre el estado y compararlo en diferentes momentos con gráficas.

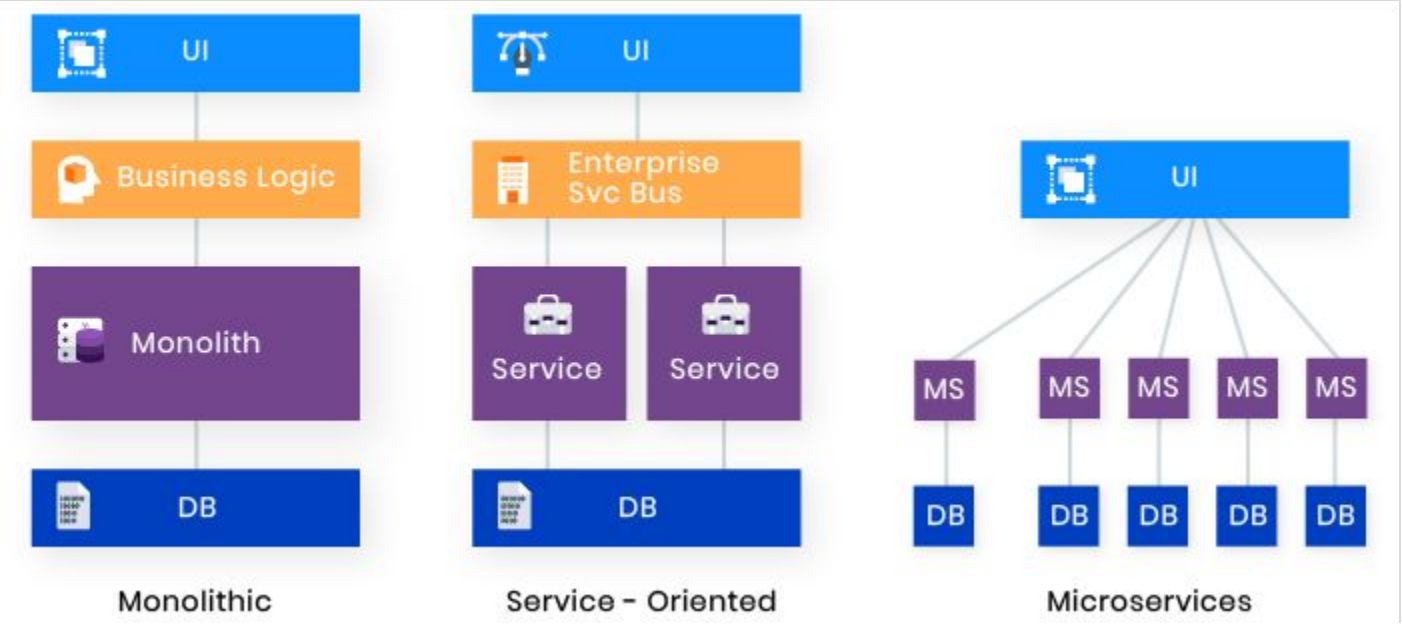
MIKRO WAYS

# ¿Arquitectura de aplicaciones?

# ¿Arquitectura de aplicaciones?

- Conocemos paradigmas, lenguajes y frameworks de programación.
- Aplicamos buenas prácticas en cada uno de ellos:
  - Patrones de objetos
  - Estándares de codificación
  - Seguimos los lineamientos de un framework
  - Aplicamos TDD
- ¿Pero qué tipos de arquitectura de aplicaciones conocemos?

# Algunas arquitecturas




MIKRO WAYS

# Algunas arquitecturas

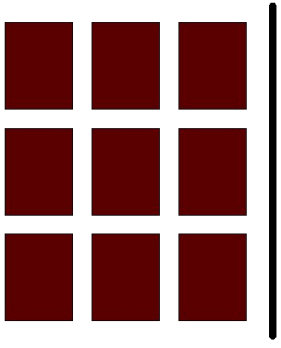
- Diseño en capas
- MVC (Model View Controller)
- SPA (Single Page Applications)
- PWA (Progressive Web Applications)
- SOA
  - Service bus
  - API gateway
  - Restful
  - SOAP
- Microservicios

# ¿Despliegues?

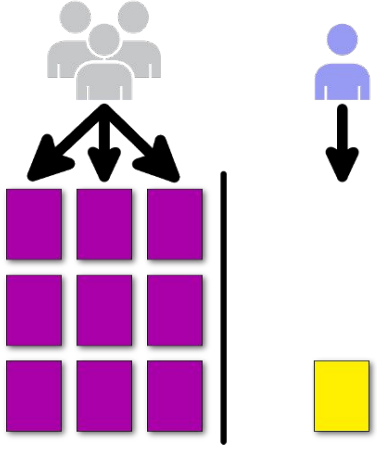
# Tipos de despliegue

- ¿Puedo **parar** una aplicación **para actualizarla**?
  - ¿Qué pasa si un proveedor de streaming decide actualizar mientras miramos un video?
  - ¿Y si un ente gubernamental decide actualizar una aplicación? ¿Debería hacerlo **fuera de horario**? ¿Notificar a sus usuarios?
- ¿Cómo podría evitar ***downtimes*** ocasionados por un despliegue?
  - Usando Tests, ambientes donde pueda probar los despliegues y algunas estrategias de despliegue convenientes.

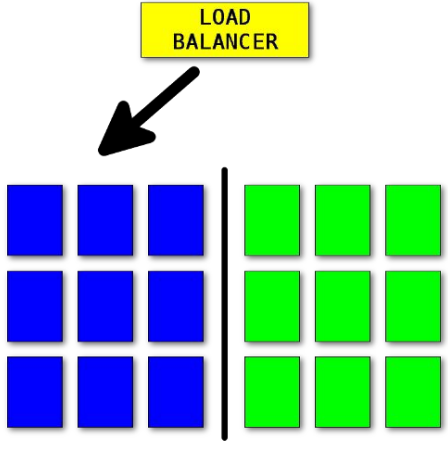
# Estrategias de despliegue



rolling updates



canary




blue green

# Las estrategias consideran varias instancias

- Esto habla sobre la **escalabilidad de una aplicación.**
- ***Escalar una aplicación ¿es fácil?***
  - ¿Cómo balanceo el tráfico entre las instancias?
  - ¿Cómo aplico la misma configuración a todas las instancias?
  - Aparecen problemas:
    - Patrones que son antipatrones
    - Logs
    - Sesiones

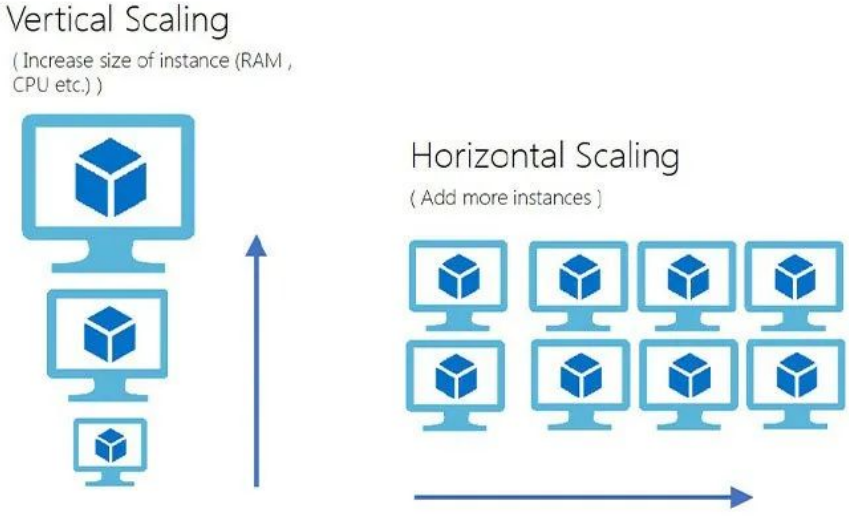
# ¿Escalabilidad?

La escalabilidad o escalamiento nos permite **ajustar la capacidad de un recurso cambiando su escala**.

Se escala para **crecer** o **decrecer**, según sea la necesidad del recurso en cuestión.

# Tipos de escalamiento

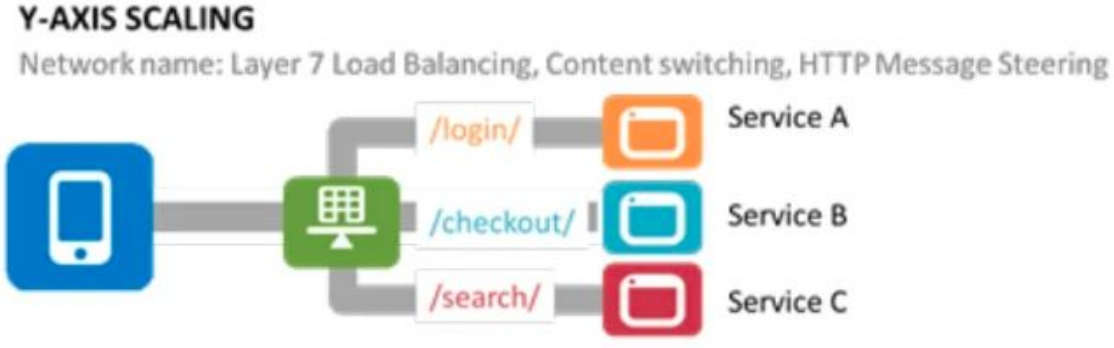
- **Vertical:** agrega hardware. ***Paradójicamente no escala*.**
- **Horizontal o en el eje X:** agrega instancias similares de una aplicación que distribuyen la carga




MIKRO WAYS

# Tipos de escalamiento

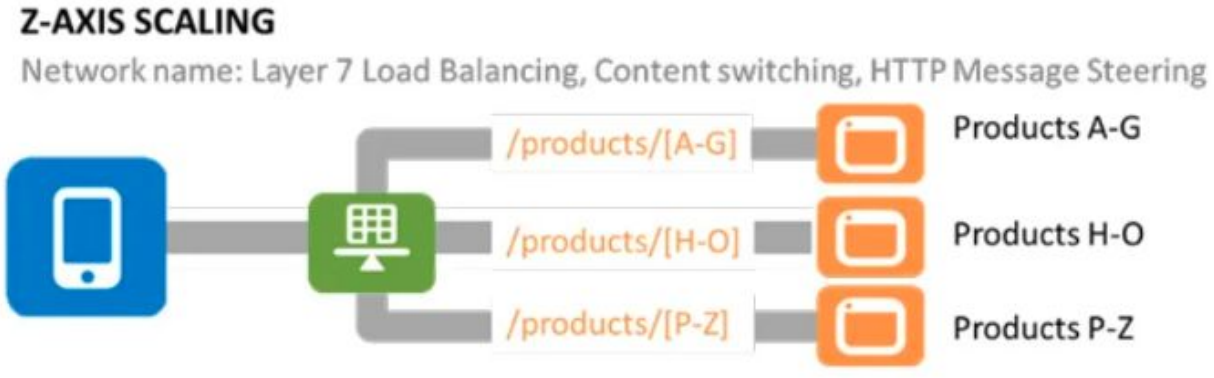
- **En el eje Y:** una aplicación de microservicios donde cada uno puede escalar en X




MIKRO WAYS

# Tipos de escalamiento

- **En el eje Z:** sharding. Por ejemplo en bases de datos.




MIKRO WAYS

# ¿Mantenimiento?

MIKRO WAYS

# No sólo es cuestión de desarrollo

- Las aplicaciones no solamente responden a cambios en el desarrollo por usabilidad.
- También hay cuestiones de **carga** que pueden hacer que una aplicación **colapse**.
- Tenemos que conocer la salud de cada instancia para poder **establecer su capacidad**.
- Una aplicación que alcance su capacidad y se verá **saturada**.

# ¿Qué debemos observar?

- **Logs:** ver los logs de una instancia es fácil.
  - De 10 es más complejo. De cientos, imposible
  - Tenemos que **centralizar los logs** y apoyarnos en herramientas que nos ayuden a identificar problemas.
- **Métricas:** medir parámetros de la aplicación nos ayudará a saber cómo funciona.
  - Se cree que lo importante es observar los recursos básicos de computación: CPU, memoria, disco y red.
  - Sin embargo, en las aplicaciones web se utilizan **métricas http específicas.**

# Métricas que importan

- Cantidad de peticiones
- Cantidad de errores sobre el total
- Tiempo de respuesta de cada petición (latencia)

# ¿Cómo atiendo todo esto?

MIKRO WAYS

# Resumen de nuestros problemas

- Mucho esfuerzo del ciclo de vida se destina en analizar requerimientos, desarrollar y generar entregables.
- Cuánto esfuerzo dedicamos a pensar:
  - ¿Cómo **entregamos**?
  - ¿Cómo **desplegamos**?
  - ¿Cómo **mantenemos**?

# 12 factor apps

https://12factor.net/

MIKRO WAYS

# 12 factor apps background

El primer commit del documento data de **Noviembre del 2011**

Describen la experiencia de personas que han estado involucradas directamente en el **desarrollo** y **despliegue** de **cientos de aplicaciones**, y han sido testigos indirectos del desarrollo, las operaciones y el escalado de cientos de miles de aplicaciones en la plataforma **Heroku**.

MIKRO WAYS

# 12 factor apps

Una aplicación que aplica los 12 factores garantiza:

- Que la aplicación pueda usarse como **SaaS**
- **Minimizar** las diferencias entre desarrollo y producción
- **Escalamiento** sin cambios significativos
- **Despliegues** en la nube

MIKRO WAYS

# Resumen de 12 factor apps

1. Código base
2. Dependencias
3. Configuraciones
4. Backing services
5. Construir, distribuir, ejecutar
6. Procesos sin estado
7. Puertos independientes
8. Concurrencia
9. Desechabilidad
10. Mantener desarrollo y producción similares
11. Logs
12. Administración

# Resumen de 12 factor apps

1. **Código base**
2. **Dependencias**
3. **Configuraciones**
4. **Backing services**
5. **Construir, distribuir, ejecutar**
6. **Procesos sin estado**
7. Puertos independientes
8. Concurrencia
9. Desechabilidad
10. **Mantener desarrollo y producción similares**
11. **Logs**
12. Administración

MIKRO WAYS

# 1. Código base

# 1. Código base

- Las aplicaciones se versionan.
- Si hay múltiples repositorios se tratará cada uno de una aplicación de 12 factores.
  - Por ejemplo microservicios, o servicios SOA.
- No se comparte código entre 2 aplicaciones de 12 factores.
  - Si fuese necesario, utilizar librerías **(punto 2).**
- Una versión de una aplicación con el mismo código base puede desplegarse en diferentes ambientes usando diferentes configuraciones **(punto 3).**

# 2. Dependencias

MIKRO WAYS

# 2. Dependencias

- Usar un gestor de dependencias que permita trazar qué versión de cada librería se usa en un release:
  - **PHP:** composer
  - **JS:** npm / yarn
  - **Java:** maven / gradle
  - **Python:** pip
  - **Ruby:** gem/bundler
- **Versionar la especificación de qué versiones se usaron**, y de ser posible el **lockfile** con *versiones exactas* de paquetes.

# 3. Configuraciones

MIKRO WAYS

# 3. Configuraciones

- La configuración de una aplicación es lo ***único que puede variar entre despliegues de una misma versión***.
- La configuración (junto con los datos), son el alma de una aplicación.
- Consideramos configurable cualquier dato que modifique cómo la aplicación se comporta o articula con algún backing service: *base de datos, storage, API externa, credenciales, etc.*
- Las configuraciones, en especial las credenciales o datos sensibles **no deben versionarse**.
    - Pueden versionarse configuraciones por defecto.

# 4. Backing services

# 4. Backing services

- Es cualquier servicio que puede consumirse por red: *base de datos, gestor de colas, storage, mail, cachés, colectores de métricas, etc.*
- Tratarlos como **recursos conectables.**
- Usar **variables de entorno** para configurarlos, de ser posible en forma de URL:

```
DATABASE_URL=mysql://user:pass@db.host:3306/db_name?encoding=utf8
```

- En caso que falle un backing service, el administrador podrá cambiarlo rápidamente sin mayores esfuerzos. **Son fácilmente conectables.**

MIKRO WAYS

# 5. Construir, distribuir, ejecutar

# 5. Construir, distribuir, ejecutar

- Separar las etapas de construcción (compilación o transpilación) de la de ejecución.
- La construcción genera un artefacto listo para ser desplegado posteriormente.
  - **Se entregan artefactos listos para usarse, no entornos de desarrollo para construir artefactos**
- La distribución consiste en mover el artefacto a algún repositorio.
- La ejecución es el runtime de la aplicación en un ambiente. Para ejecutar una ***nueva versión*** se debe realizar un **despliegue**.
  - Se automatizan con herramientas o estrategias

# 6. Procesos sin estado

# 6. Procesos sin estado

- El estado de una aplicación debe mantenerse en algún backing service (sesiones por ejemplo).
- La **persistencia** de datos puede estar en una base de datos o filesystems compartidos.
- El uso de **sticky sessions** es una **mala práctica**. ***Evitarlo*** y en su lugar llevar sesiones a memcached, redis o similares.

*Imaginar un Wordpress que se escale a 4. Pensar* ***cuál es el alma de esa aplicación***

MIKRO WAYS

# 10. Mantener desarrollo y producción similares

MIKRO WAYS

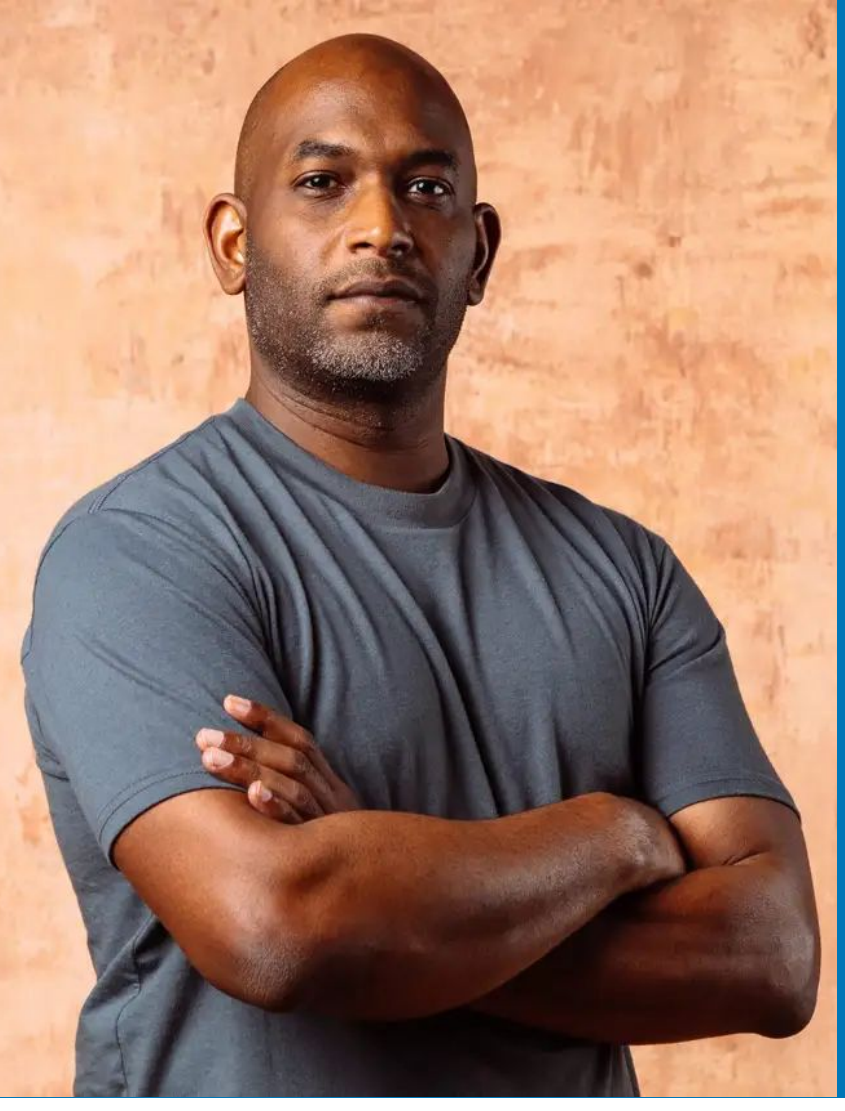
# 10. Mantener desarrollo y producción similares

- Reducir los tiempos de despliegue para que un desarrollador pueda ver sus cambios en cuestión de minutos.
- Quienes desarrollan deben involucrarse además en definiciones de despliegue.
- Quienes desarrollan observan el comportamiento de sus despliegues en producción.
- Las herramientas usadas en desarrollos y producción son tan parecidas como sea posible.

# 11. Logs

# 11. Logs

- Recomienda **no utilizar archivos** para los logs sino **stdout** y **stderr.**
  - *Tanto en desarrollo usando la consola como en producción.*
- Cada log debe estar en **una línea.**
  - *Salvo por las trazas de excepciones que podrán estar en varias líneas.*
- De esta forma se simplifica la captura de logs por el runtime y su centralización:
  - systemd
  - container engine
  - fluentd



# 12 fractured Apps

https://cutt.ly/GCmcCy3

Kelsey Hightower

MIKRO WAYS

# 12 factor apps y contenedores

Con la aparición de docker, 12 factor apps comenzó a brillar:

- Logs en stdout
- Configuraciones a través de variables de ambiente
- Se construyen, distribuyen y despliegan imágenes de contenedores

# Kubernetes

## ¿es la solución a todos mis problemas en la nube?

MIKRO WAYS

# ¿Qué es kubernetes?

- Su sitio lo define como
  - Una plataforma portable y extensible de código abierto para administrar cargas de trabajo y servicios.
  - Facilita la automatización y la configuración declarativa.
- Además, dicen que puede verse como una plataforma:
  - De contenedores
  - De microservicios
  - Portable de nube

# ¿Por qué nos ayuda?

- Porque podemos implementar k8s en nuestro datacenter on premise o cloud.
  - Esto nos permite portar de la nube al datacenter propio o al revés.
- Porque estandariza la forma de trabajo que nos ofrecen cloud providers, sin hablar de **un cloud provider.**

## ¿Entonces soluciona todos mis problemas con la nube?

- Por qué sí
- Por qué no

MIKRO WAYS

# ¿Preguntas?

MIKRO WAYS