



KEVIN BLANCO

Tech Manager en Wondersauce
Previamente Tech Lead en
Cognitiva/GBM.

Background:

- Drupal Developer (Certified Specialist)
- Front-End Developer en varias agencias.
- Arquitectura y DevOps
- Opensource Evangelist

Piloto Privado en Formación

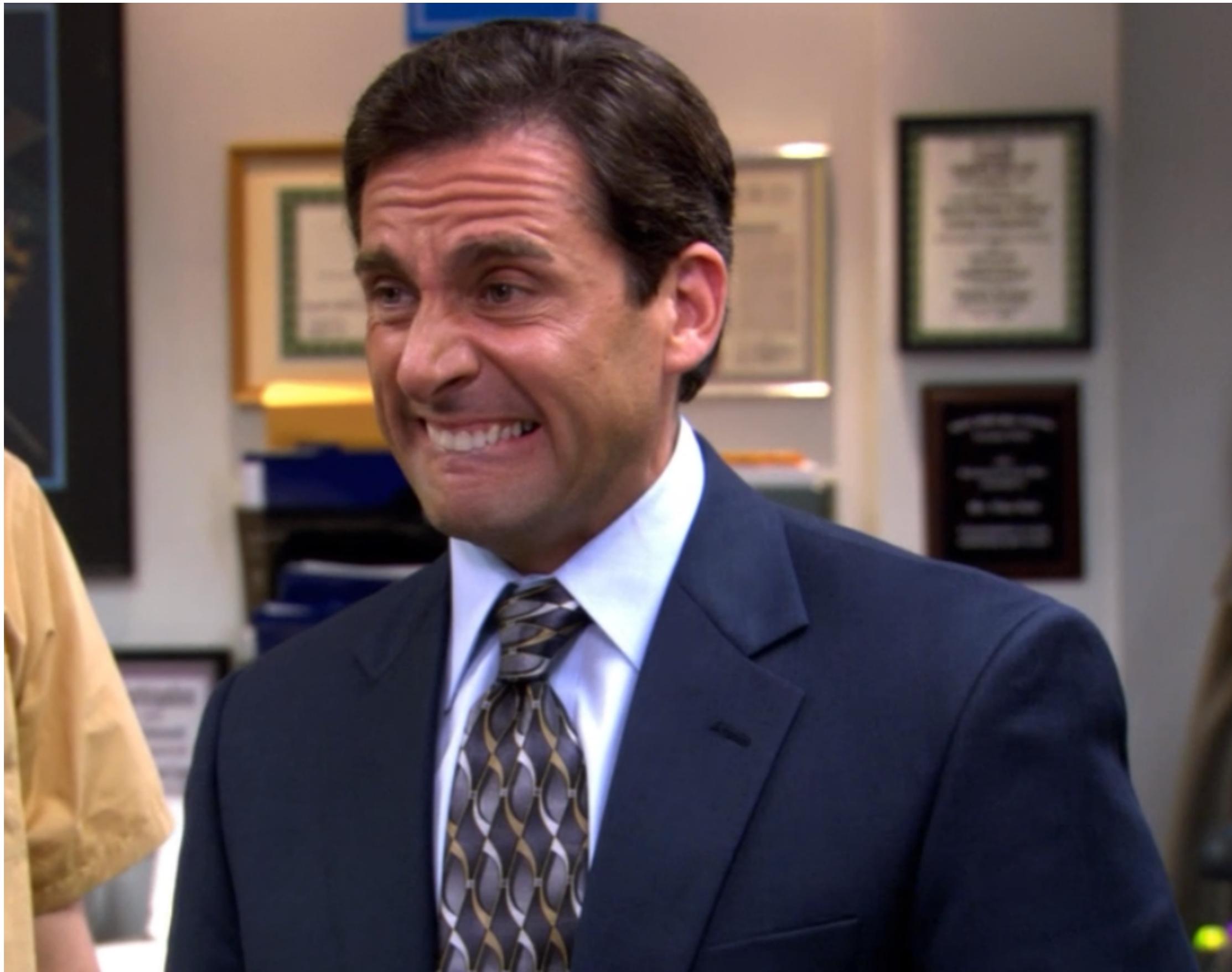
“

Primero Qué?, luego
Porque?, y al final
Cómo?

“

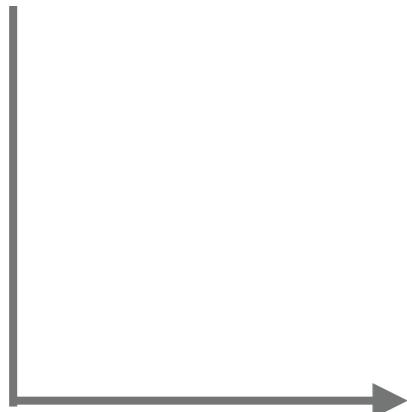
Qué?

HABLANDO DE CMS EN 2019-2020



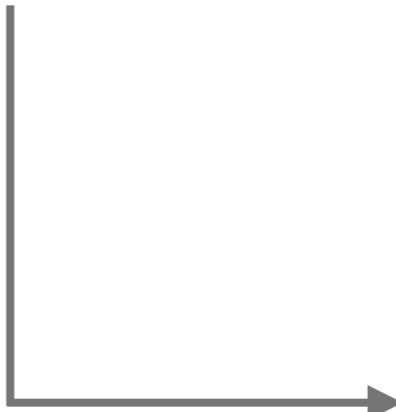
Los 90s

1. Archivos HTML y CSS planos.
2. Bugs afectan cada pagina.
3. El desarrollo de las paginas y su despliegue no era complejo.
4. No existía uso de dispositivos móviles.



Los 2000s

1. Sitios dinámicos en stacks básicos
2. Servidos clásicos
3. Desarrollo y despliegue WAMP / MAMP
4. Uso limitado de dispositivos móviles.

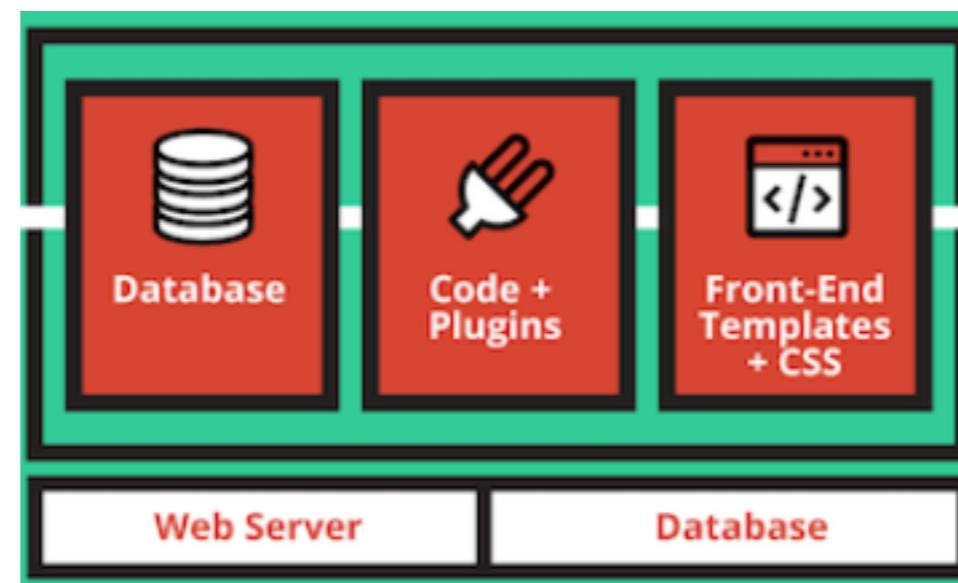


Actualidad

1. Stacks avanzados y complejos
2. Servicios en la nube
3. Desarrollo y despliegue más complejo.
4. Uso común de dispositivos móviles.

ARQUITECTURA MONOLÍTICA

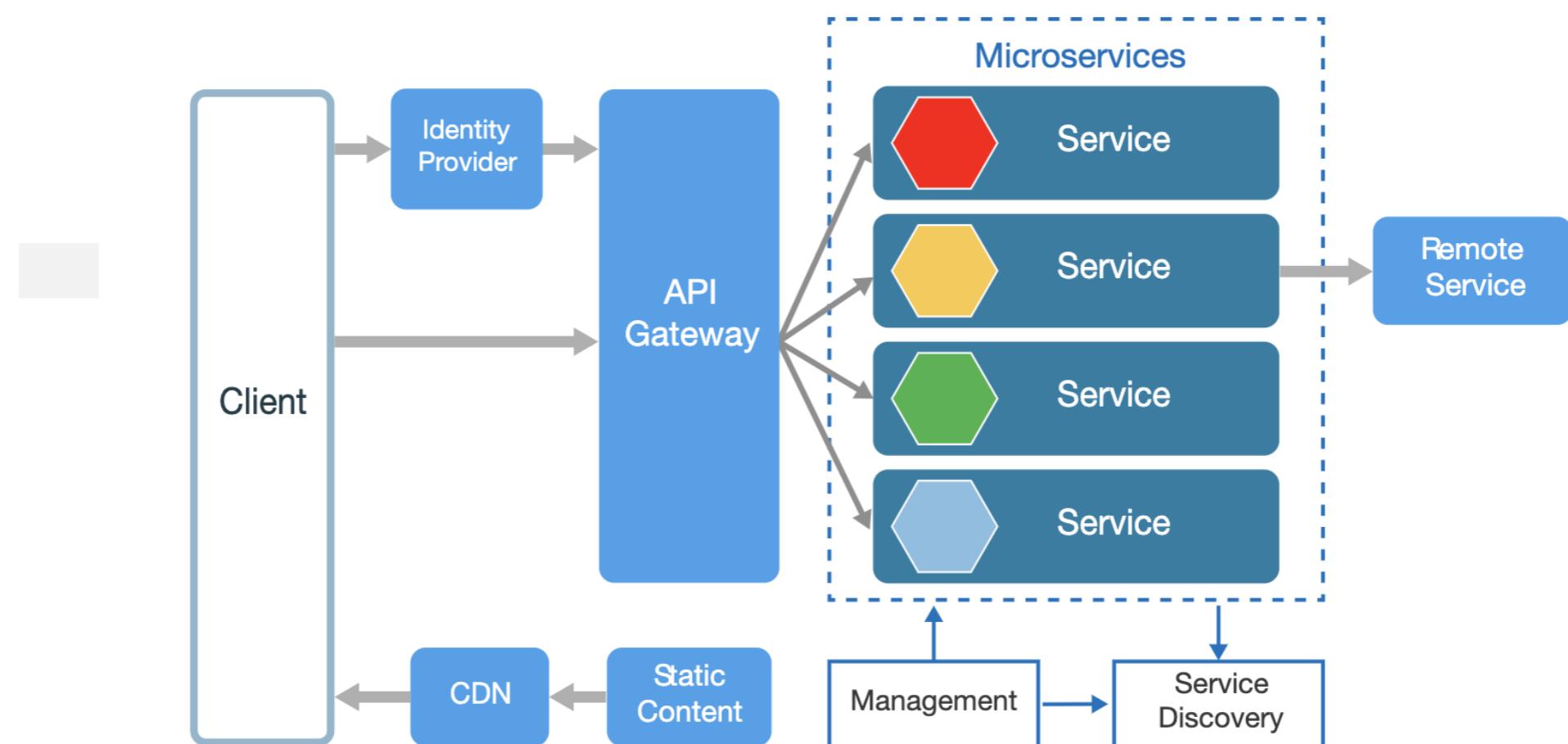
En el contexto de arquitectura, significa compuesto todo en una sola pieza. El software monolítico está diseñado para ser autónomo; Los componentes del programa están interconectados y son interdependientes en lugar de estar acoplados libremente, como es el caso de los programas de software modulares.



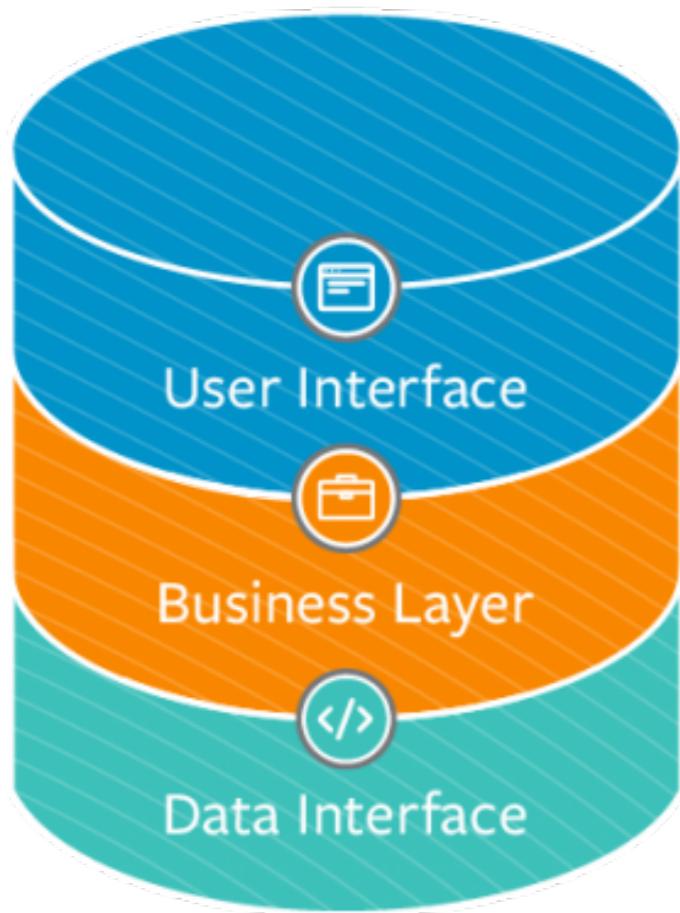


ARQUITECTURA DESACOPLADA

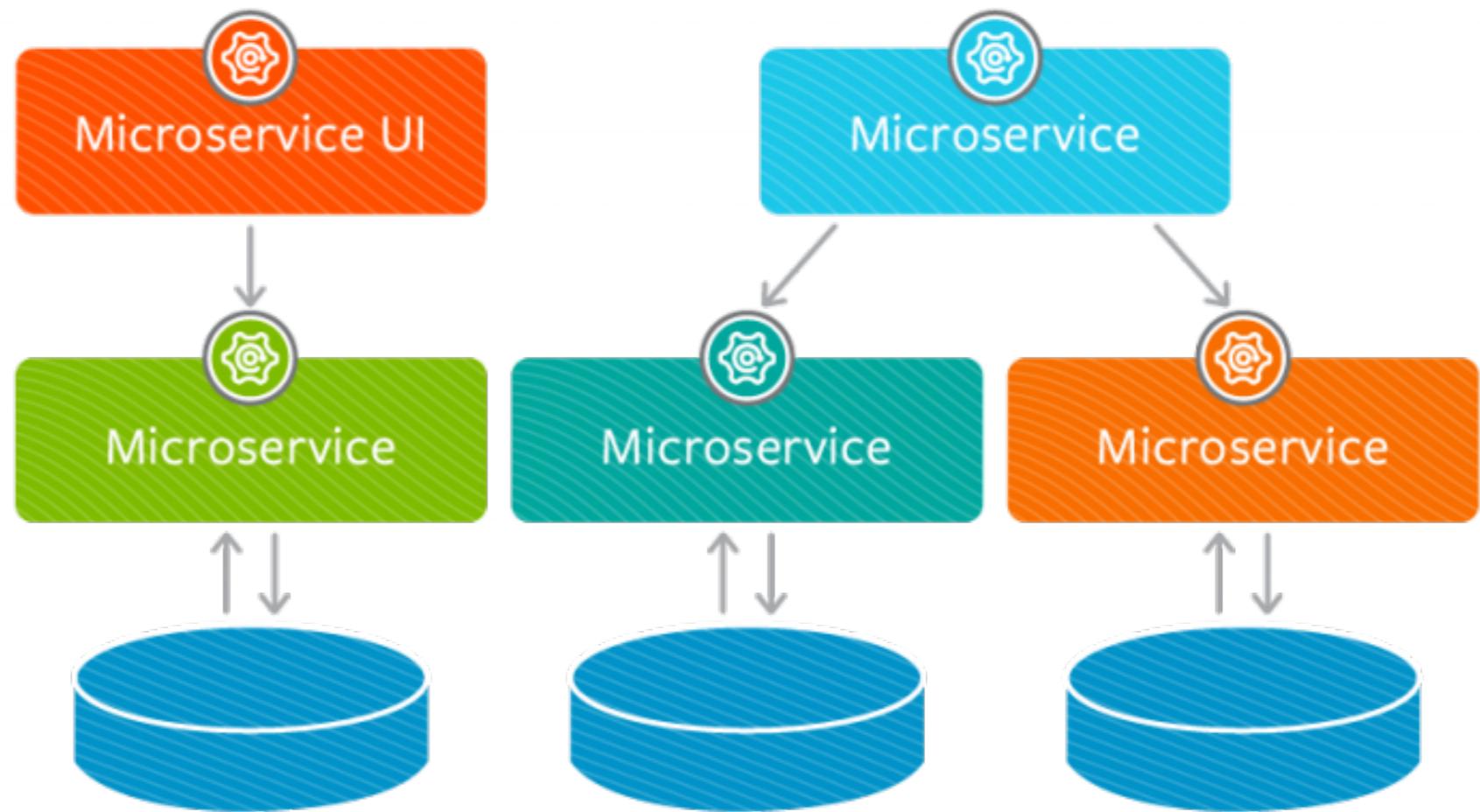
Una arquitectura desacoplada es donde los diferentes componentes / capas que componen el sistema interactúan entre sí mediante interfaces bien definidas en lugar de depender estrechamente el uno del otro. Con dicha arquitectura, los componentes/capas se pueden desarrollar de forma independiente sin tener que esperar a que se completen sus dependencias.



Monolithic Architecture



Microservices Architecture

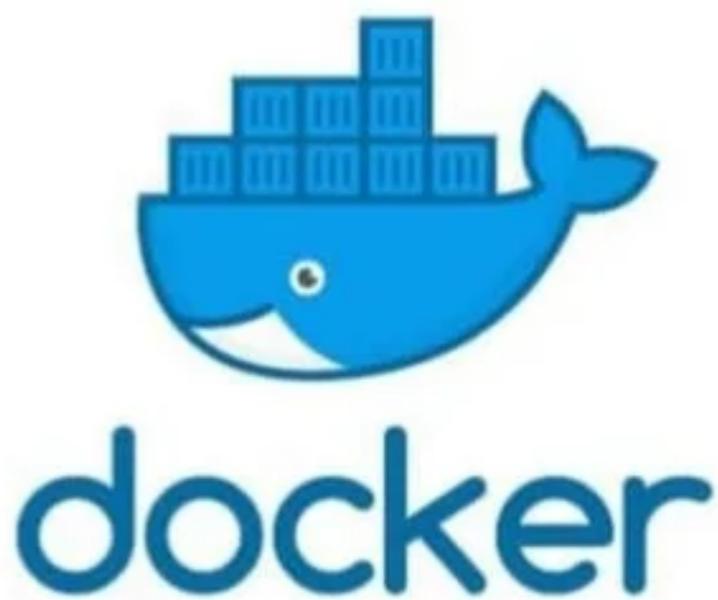


This is docker!

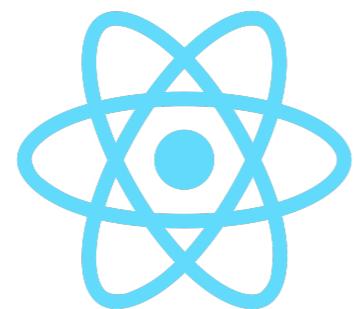
Docker emphasises on **isolation** of applications inside containers, so that different applications have no effect on each other.

Docker is smart.

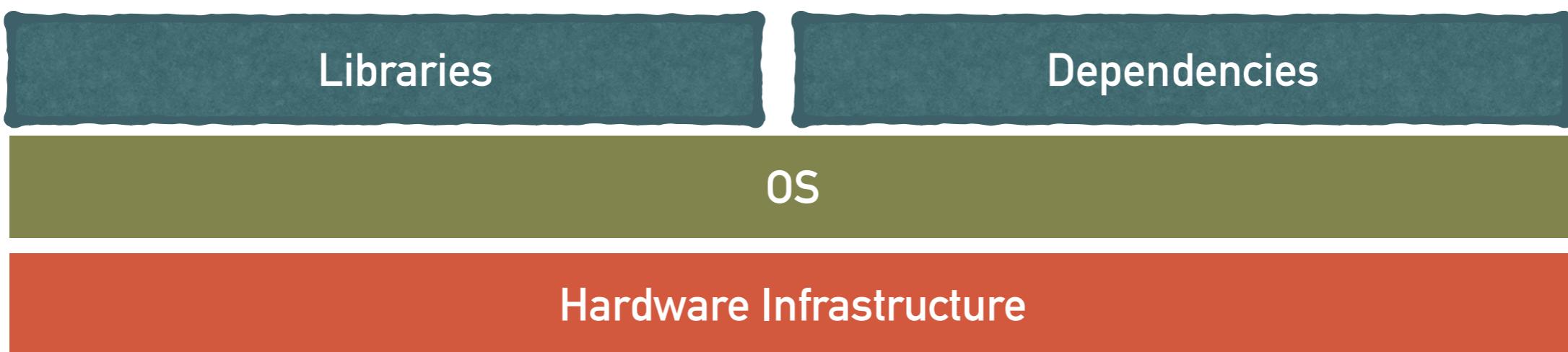
Be like docker.

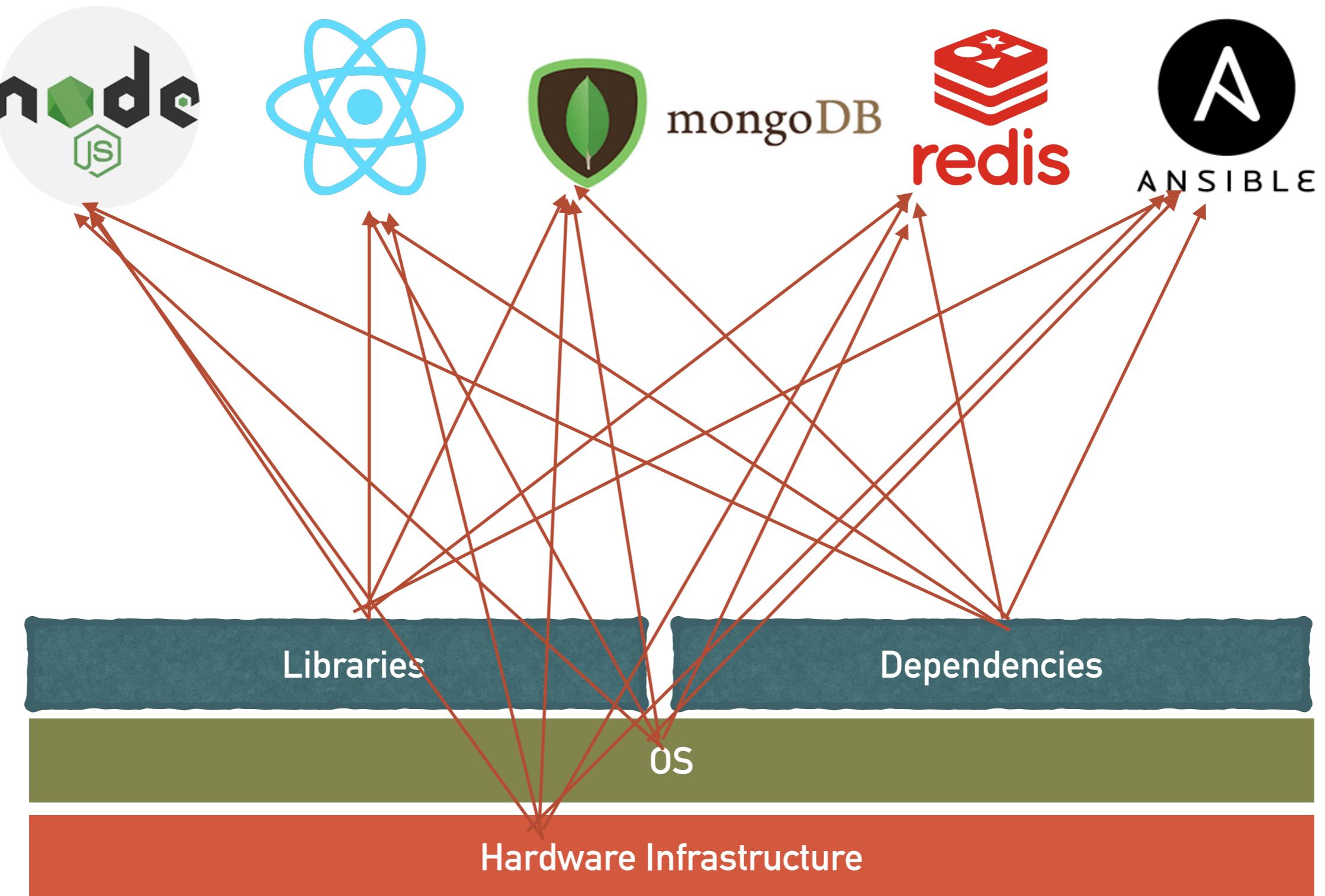


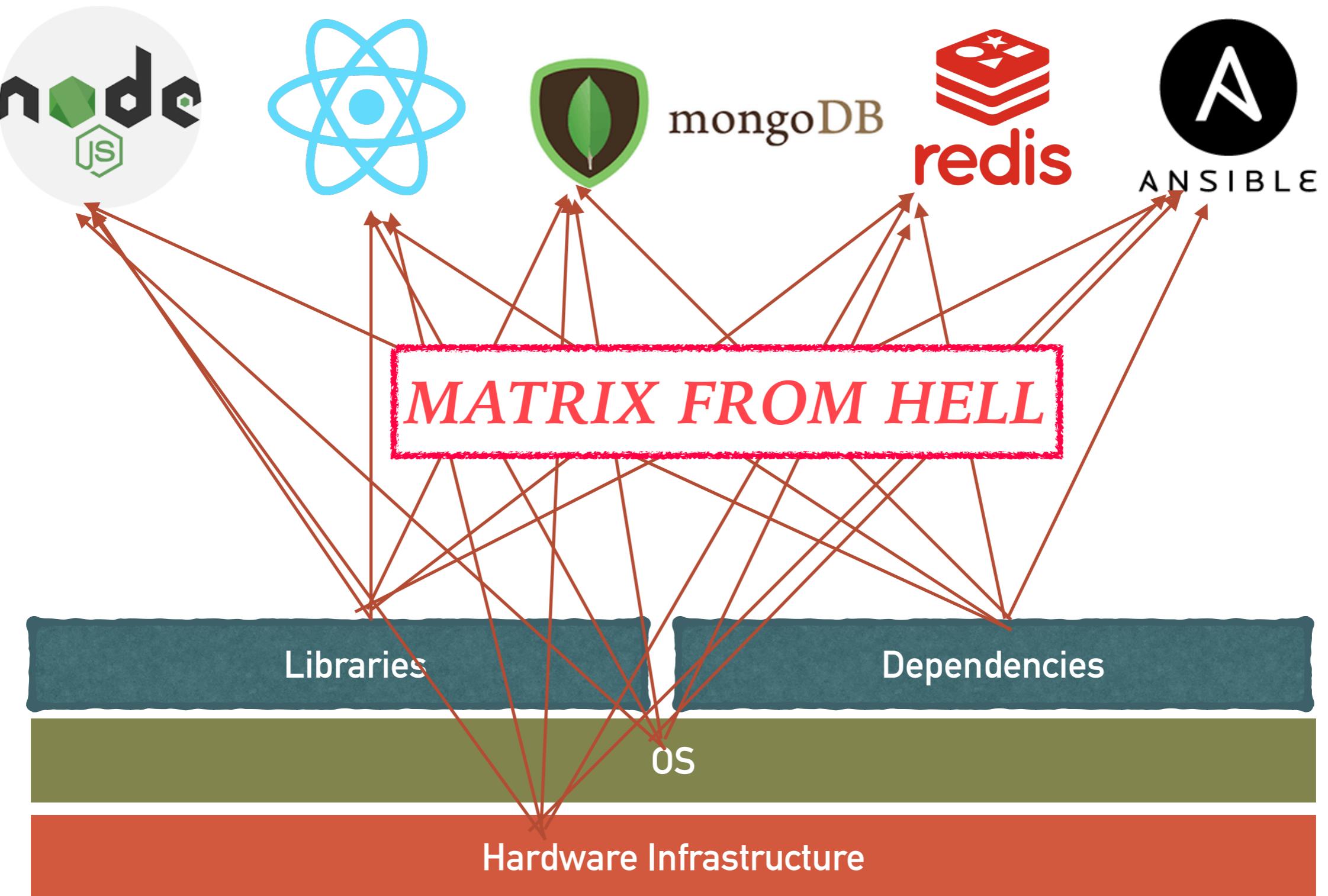
#staysafe #fightcorona

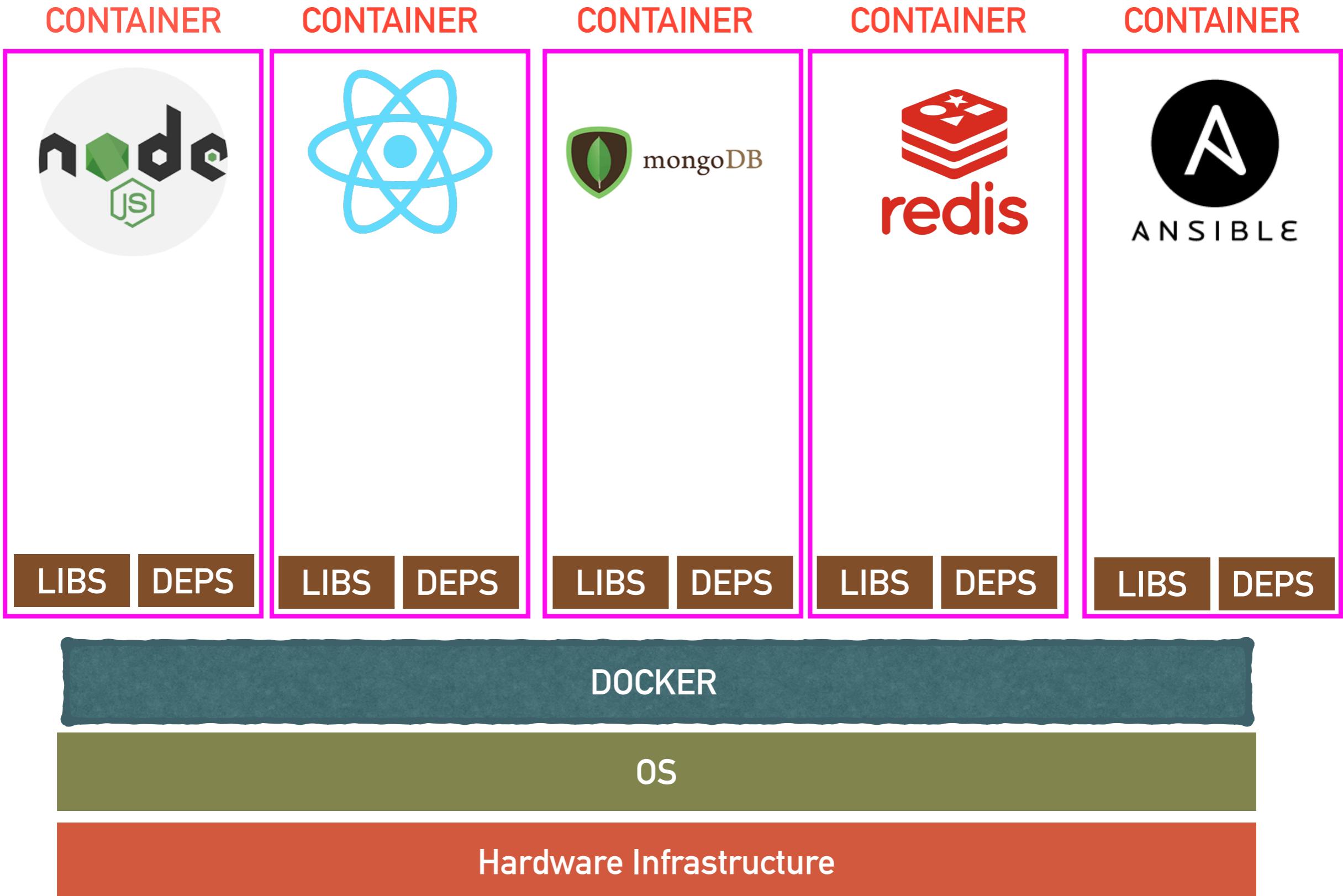


mongoDB









“

Porqué?

LOS CMS NO SE HAN QUEDADO ATRAS....

A pesar que los CMS sigan siendo en su concepción monolíticos, no quiere decir que se queden atrás en modernización. Siguen siendo la elección de muchas empresas importantes.



EL DESARROLLO LOCAL Y DELIVERY PIPELINE SE VUELVE UN DOLOR DE CABEZA

Base de Datos Local PHPMyAdmin —> Service DB.

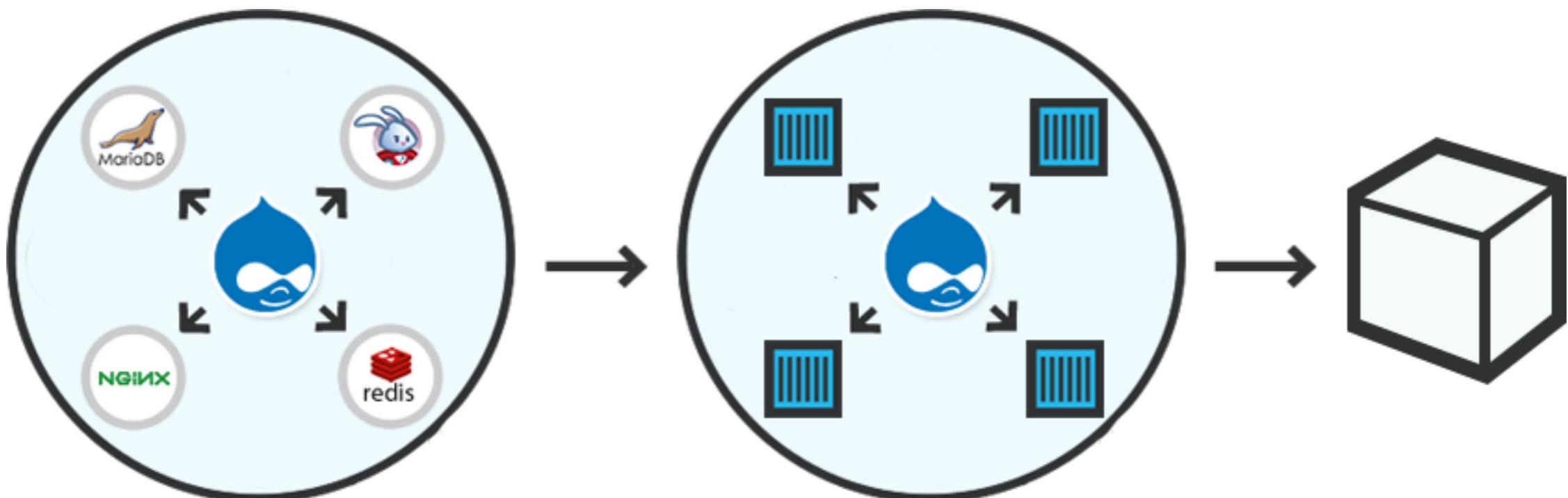
Local FileSystem —> ClusterFS

No cache —> Varnish / Redis / Akamai

Búsquedas locales —> ApacheSolR / ElasticCache.

MUCHAS HERRAMIENTAS Y TOOLS

Todo este conjunto de herramientas y plataformas alrededor del desarrollo, despliegue y operaciones de un CMS moderno necesita una manera mas simple de controlar su rendimiento (performance) y escalabilidad (scalability) principios muy importantes en arquitectura de soluciones.

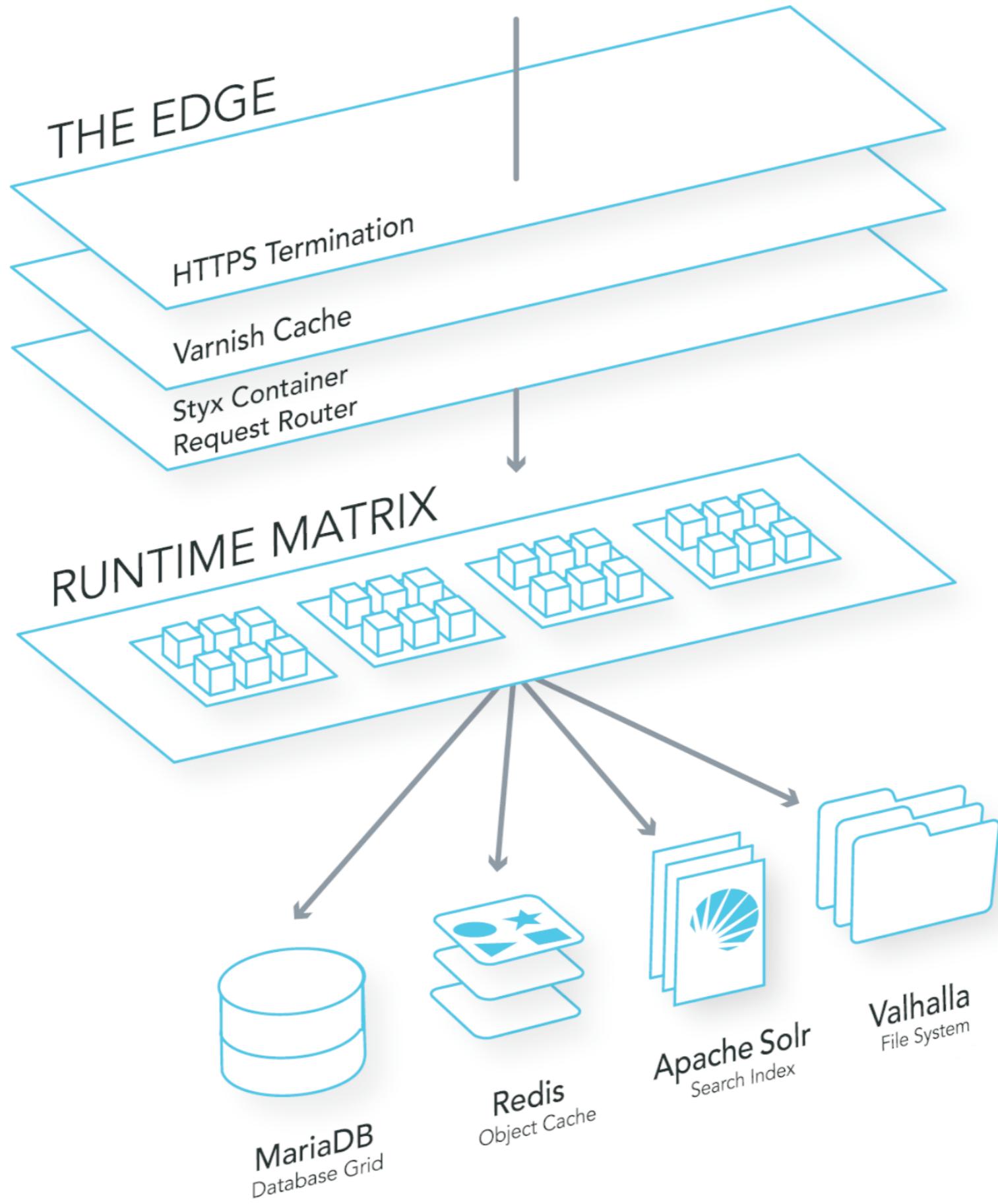


VENTAJAS QUE NOS BRINDA LOS CONTAINERS

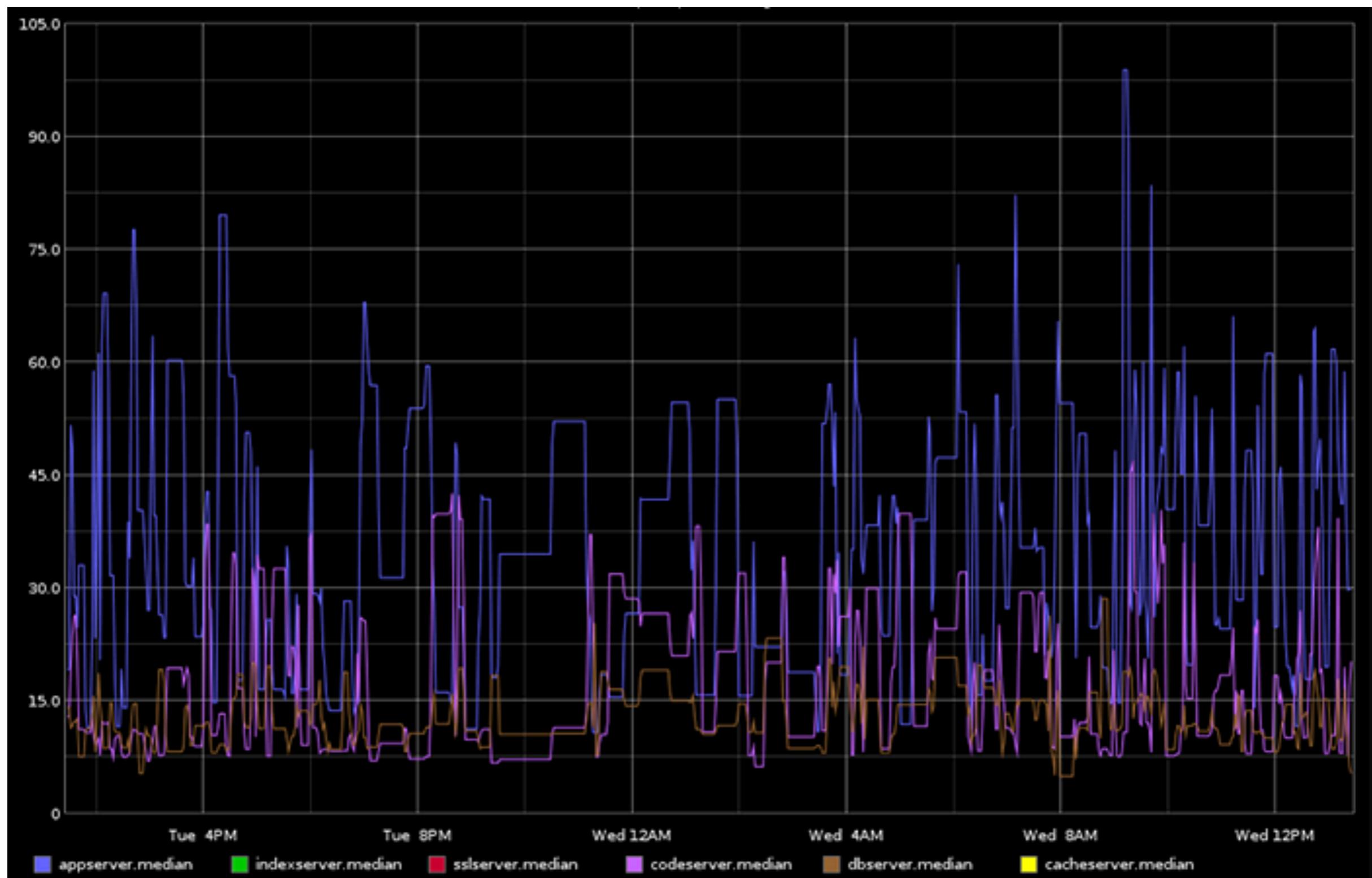
- Entornos virtuales ligeros y aislamiento de procesos y recursos como memoria/CPU.
- Más rápido, fácil de administrar.
- Experiencia consistente.
- Agregar, eliminar y redistribuir contenedores en segundos.
- Escalado por proceso, aprovisionado automáticamente.
- La configuración siempre está sincronizada.
- Alta disponibilidad.
- Prácticamente cero tiempo de transferencia dev-to-ops
- Seguridad gracias al aislamiento del contenedor.
- Reproducibilidad.

“

Cómo?



Levantar un nuevo contenedor dura entre 10 y 40 segundos



Appserver endpoint CPU usage (chios)



MUCHAS GRACIAS POR SU TIEMPO!

Web: <https://kevinblanco.com>

Correo: info@kevinblanco.com

Twitter: [@KevinBlancoZ](https://twitter.com/KevinBlancoZ)

LinkedIn: [kevinblanco](https://www.linkedin.com/in/kevinblanco/)

Github: [kevinblanco](https://github.com/kevinblanco)

@InformaticosEnCasa

@EstudianteDeAviación