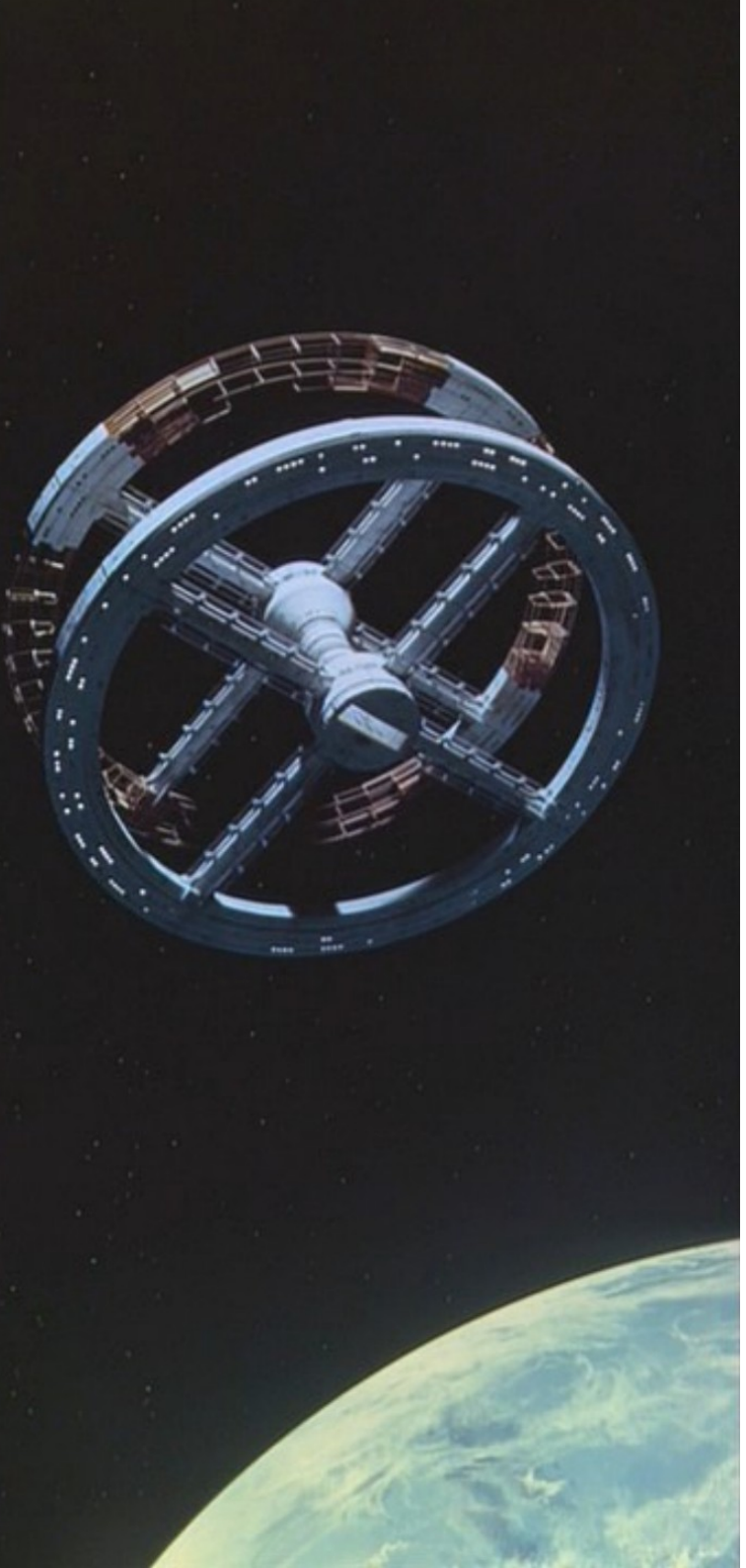


Agenda

- El infierno monolítico
- Arquitectura monolítica
- Microservicios al rescate
- Arquitectura de microservicios
- API Gateway
- Estrategias de despliegue
- De monolítico a microservicios
- Estoy interesado, y ahora?

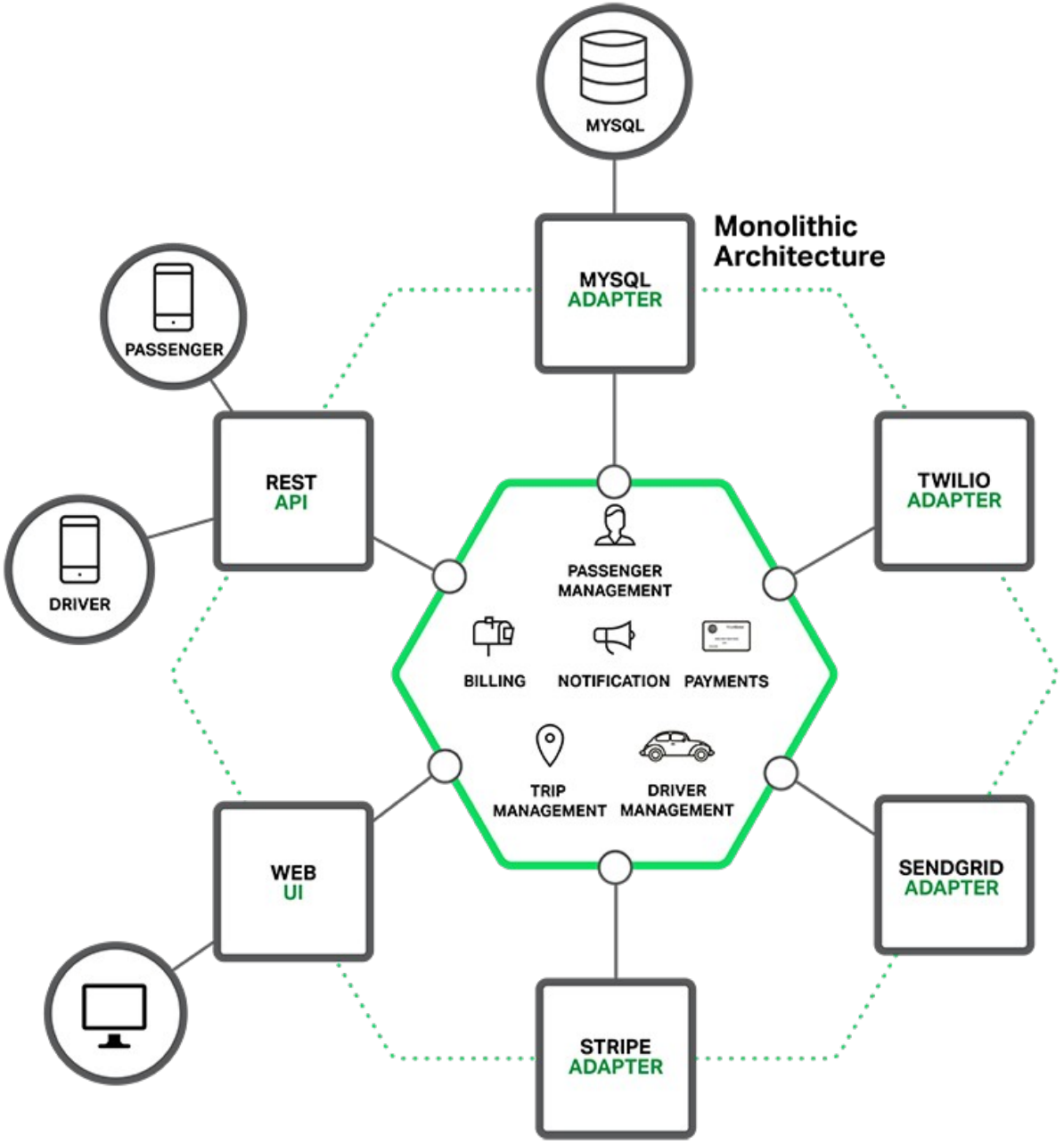
El infierno monolítico

- Complejidad abrumadora.
- Demasiado grande para que un desarrollador individual lo entienda completamente.
- Correción de errores e implementación de nuevas funciones correctamente se vuelve difícil y requiere mucho tiempo.
- Dificultad para escalar.

El infierno monolítico

“Un mundo de dolor en donde cualquier intento de desarrollo y entrega ágil será difícil”.

Arquitectura monolítica



- Lógica empresarial implementada en módulos.
- Módulos definen servicios, objetos de dominio y eventos, con adaptadores que interactúan con el mundo externo.
- A pesar de ser modular, la aplicación se empaqueta y se implementa como un monolito.
- Fáciles (en un principio) de desarrollar, probar, implementar y escalar.

Microservicios al rescate

- Muchas organizaciones, como Amazon, eBay y Netflix, han resuelto este problema mediante la adopción de microservicios.
- En lugar de crear una única aplicación monstruosa y monolítica, la idea es dividir la aplicación en un conjunto de servicios más pequeños e interconectados.
- Cada microservicio implementa un conjunto de características o funcionalidades distintas.

NETFLIX

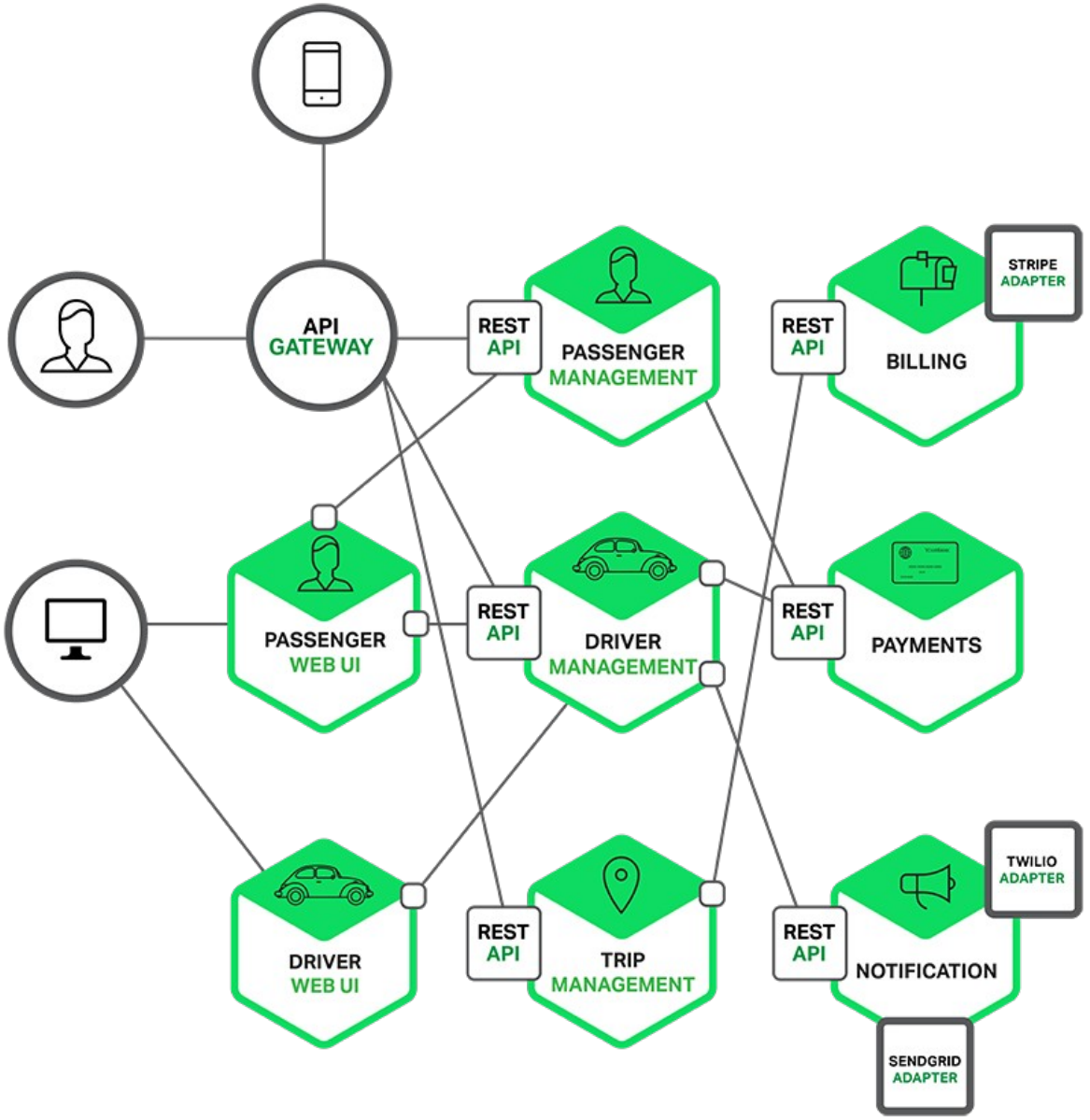
amazon

ebay

Micoservicios al rescate

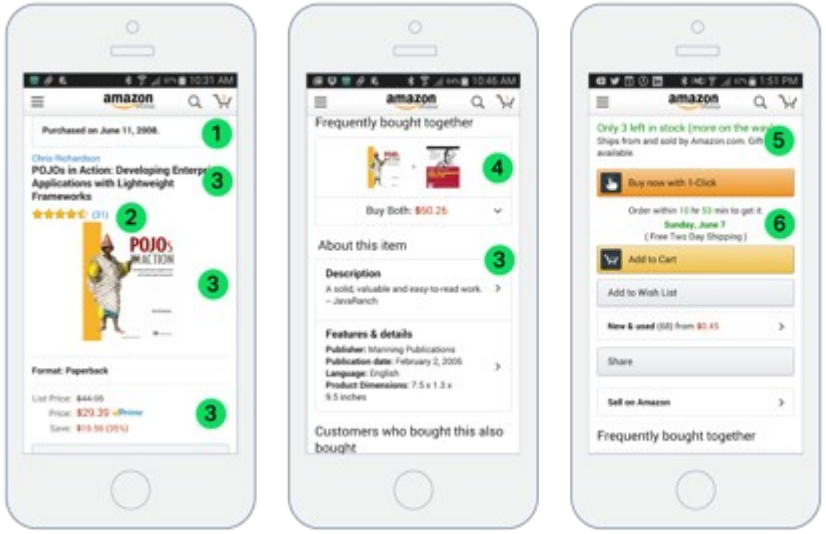
- Cada servicio es altamente mantenible y testeable. permitiendo desarrollos y despliegues rápidos y frecuentes.
- Débilmente acoplado con los otros servicios.
- Es posible implementar independientemente un servicio sin tener que coordinar con otros equipos
- Es capaz de ser desarrollado por un equipo pequeño.

Arquitectura de microservicios



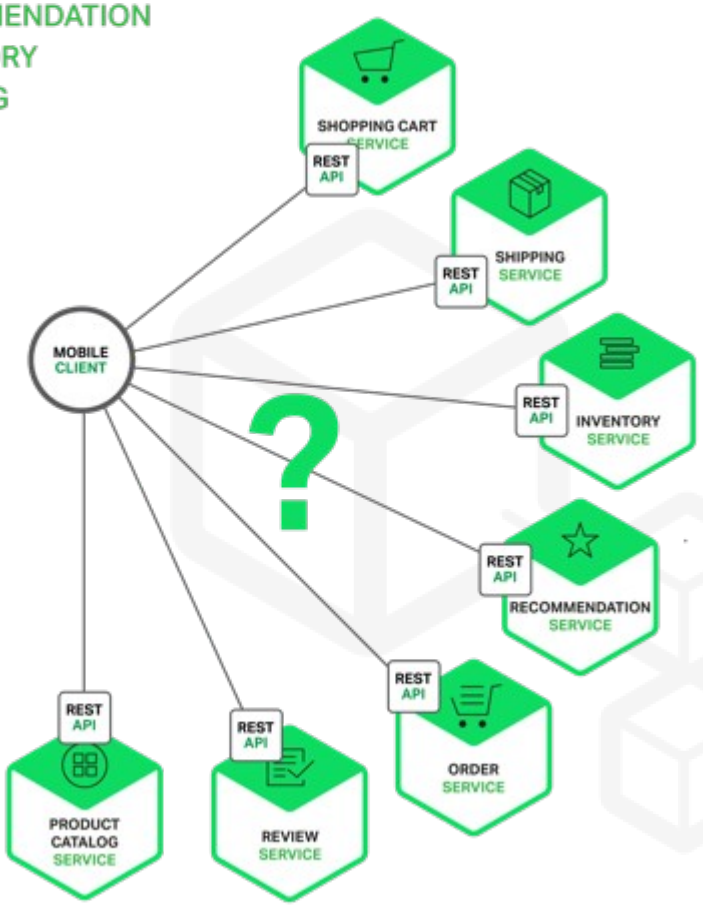
- Pueden ser desarrollados en tecnologías diferentes.
- Se comunican a través de API’s, y las llamadas pueden ser externas o entre servicios.
- Cada microservicio puede tener su propia base de datos.
- Usando herramientas como Docker/Kubernetes el despliegue y escalamiento es muy rápido .

API Gateway



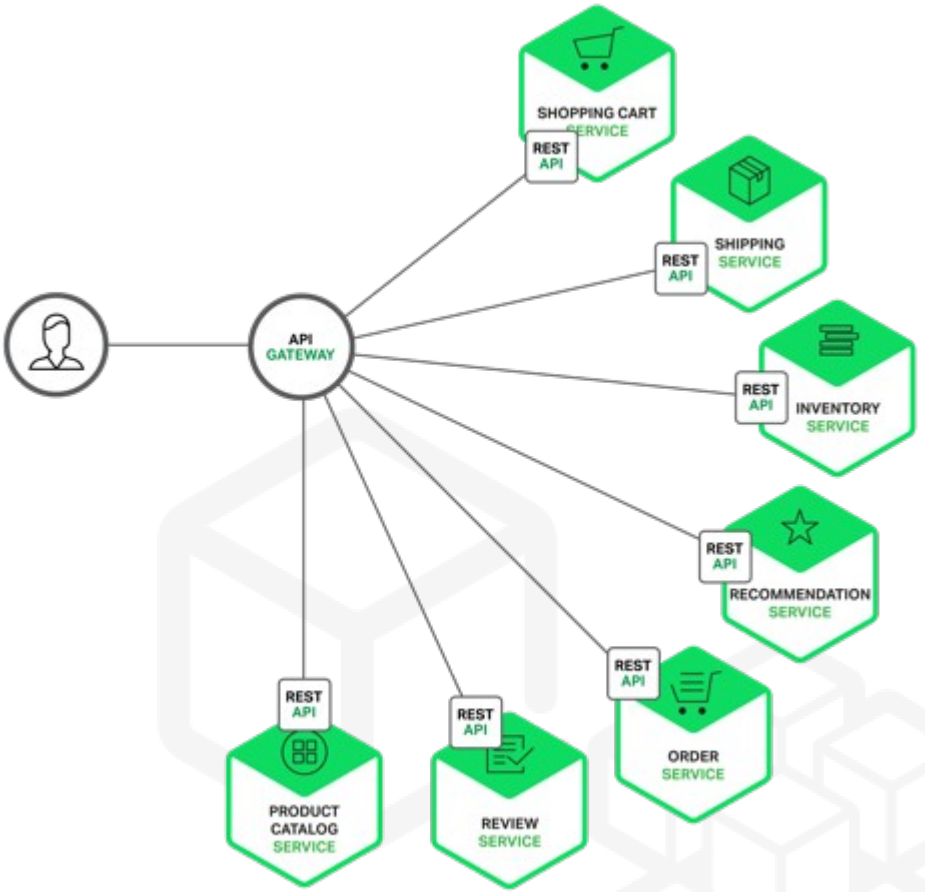
1. ORDER HISTORY
2. REVIEWS
3. BASIC PRODUCT INFO
4. RECOMMENDATION
5. INVENTORY
6. SHIPPING

¿Debe el cliente comunicarse directamente con cada microservicio?



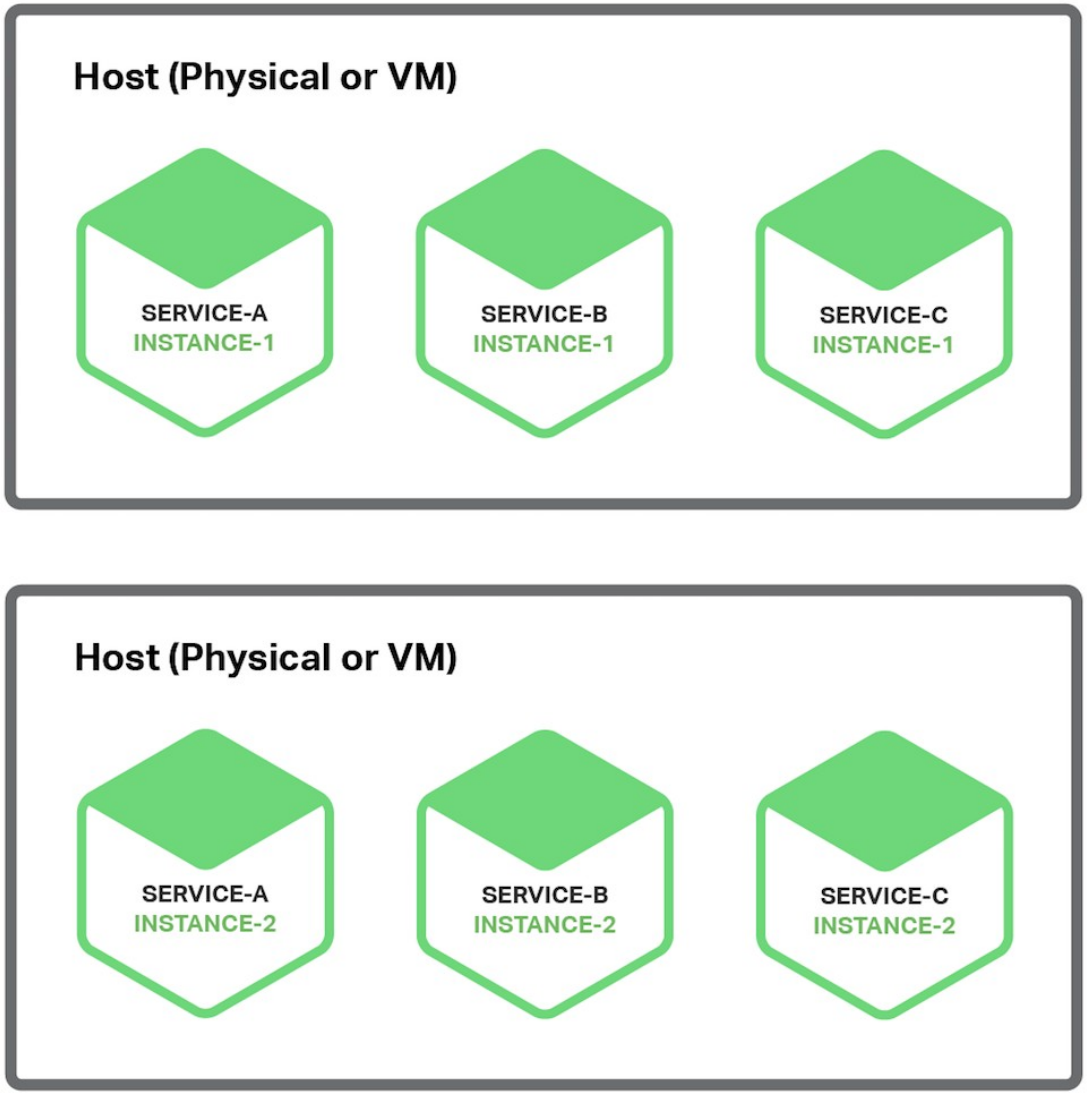
API Gateway

- Punto de entrada único.
- Es responsable del enrutamiento de solicitudes, la composición y la traducción de protocolos.
- Proporciona a cada uno de los clientes de la aplicación una API personalizada.
- Puede enmascarar fallas de los microservicios devolviendo datos en caché o predeterminados.



Estrategias de despliegue

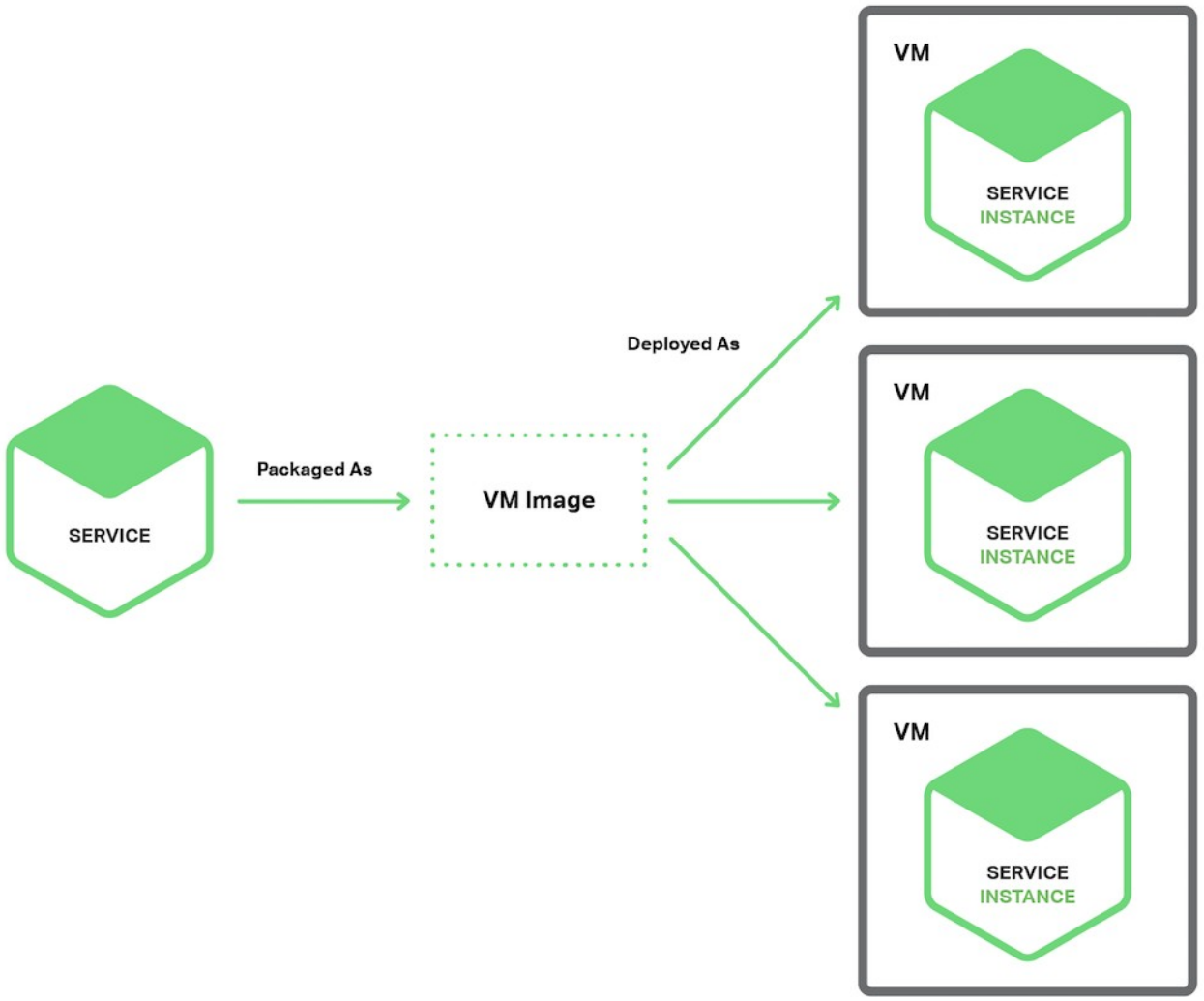
Múltiples instancias de servicio por Host



- Este es el enfoque tradicional para la implementación de aplicaciones.
- Cada instancia de servicio se ejecuta en un puerto conocido en uno o más hosts.
- *Los hosts son comúnmente tratadas como mascotas.*

Estrategias de despliegue

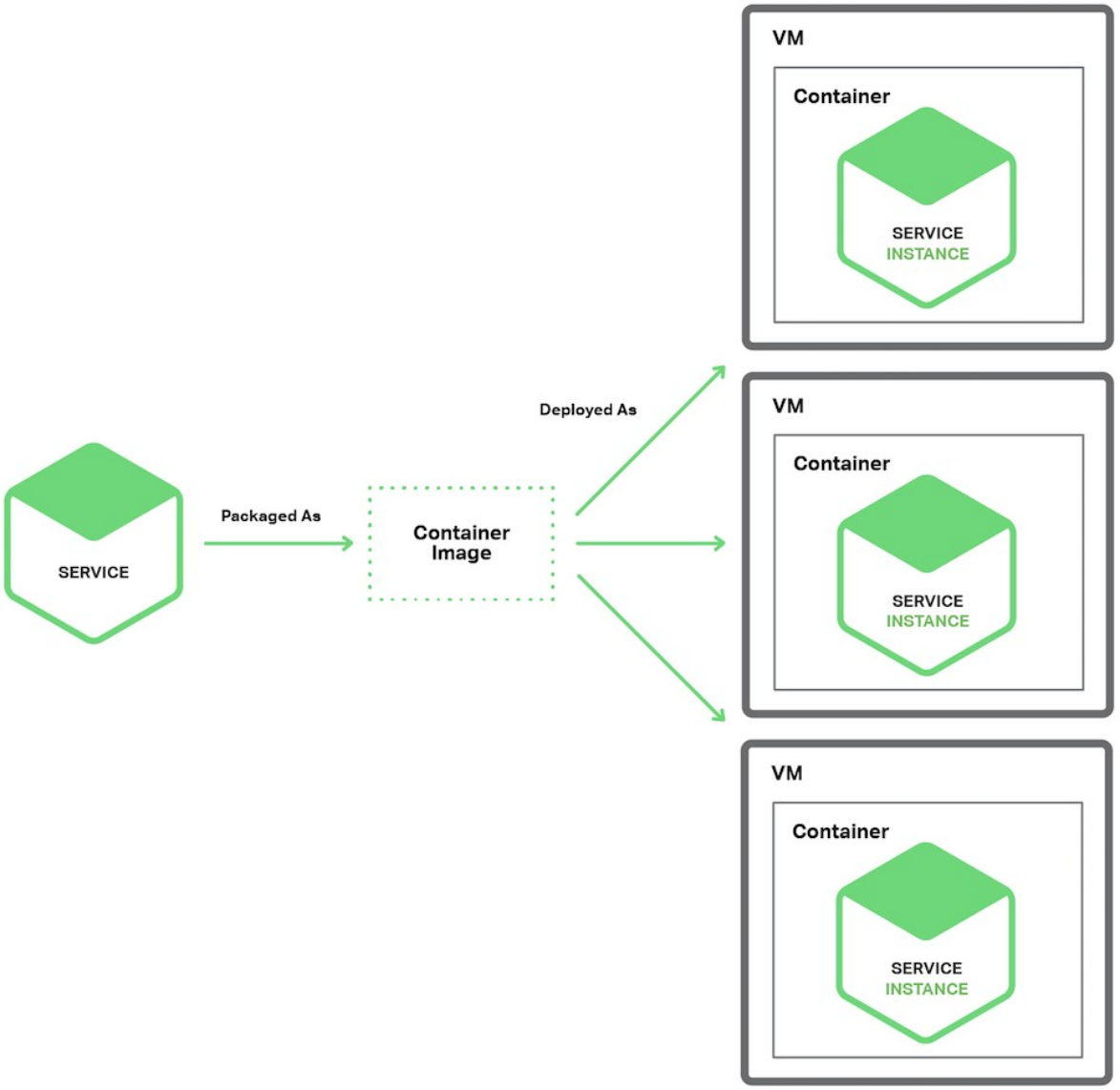
Una instancia de servicio por máquina virtual



- Cada instancia de servicio es una VM que se inicia utilizando esa imagen de VM.
- Este es el enfoque principal utilizado por Netflix para implementar su servicio de transmisión de video.

Estrategias de despliegue

Una instancia de servicio por contenedor



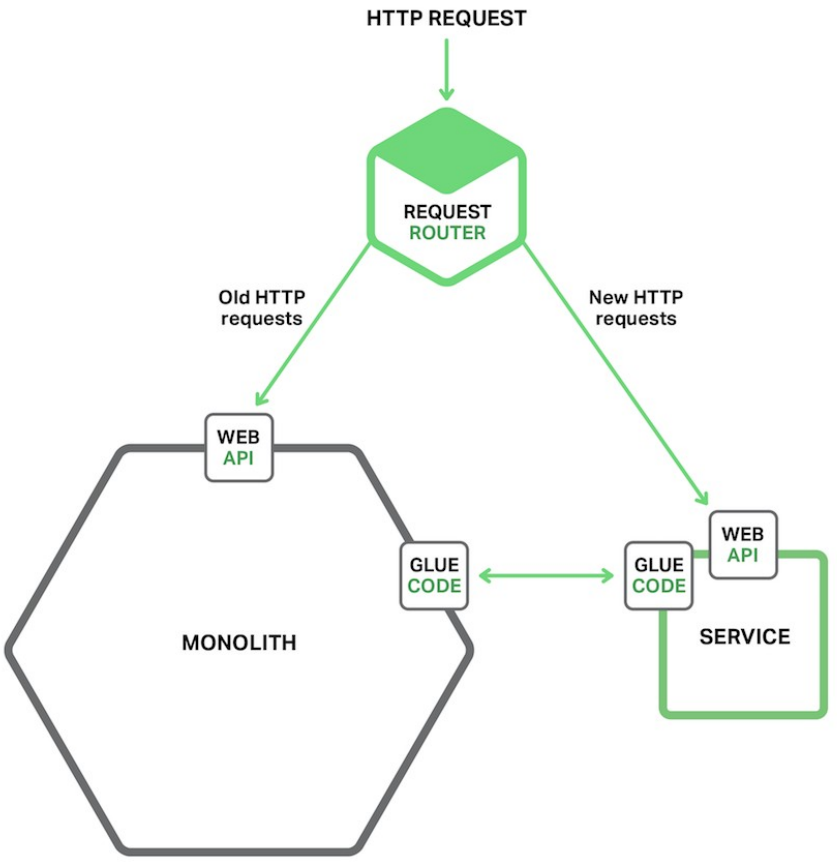
- Cada instancia de servicio se ejecuta en su propio contenedor.
- Un contenedor consta de uno o más procesos que se ejecutan en una sandbox.

De monolítico a microservicios

Implementar nuevas funcionalidades como microservicios

Estrategias de acceso a datos:

- Invoca una API remota proporcionada por el monolito.
- Accede directamente a la base de datos del monolito.
- Mantener su propia copia de los datos, que se sincroniza con la base de datos del monolito.



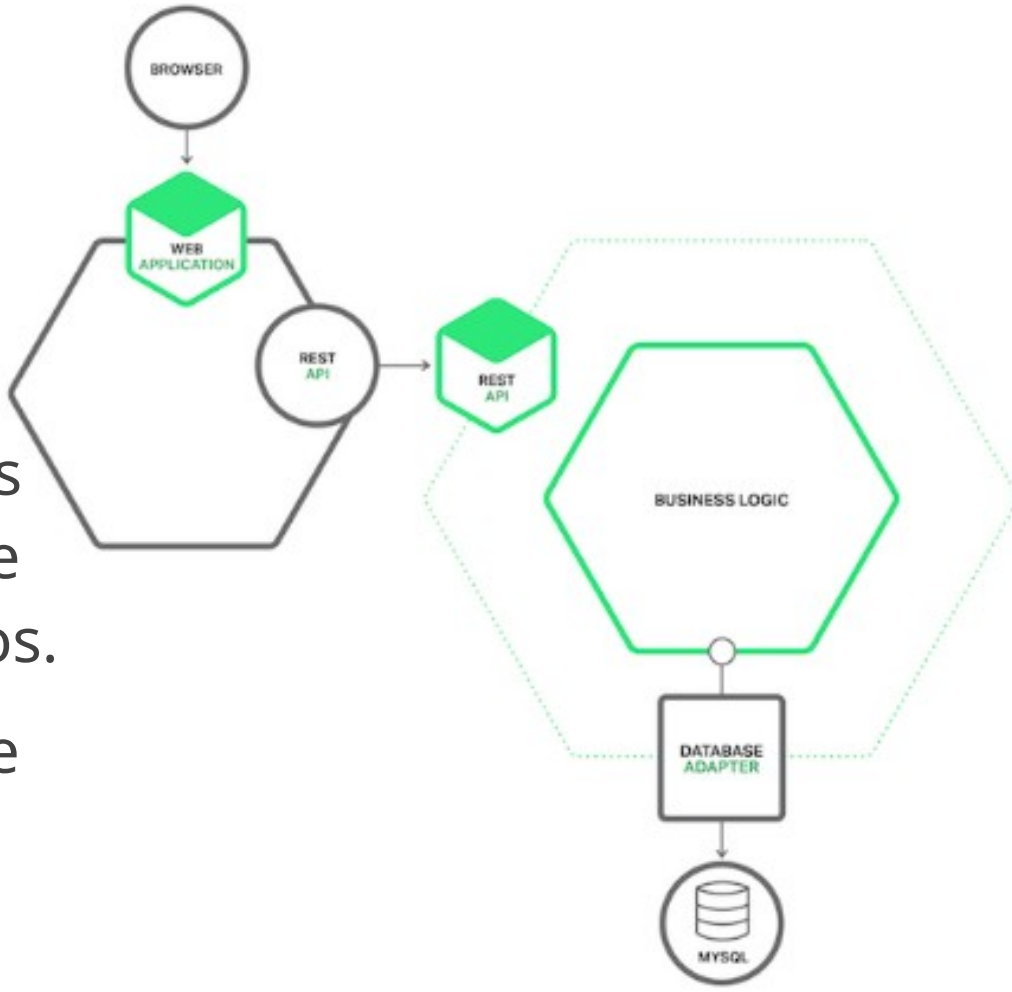
De monolítico a microservicios

Dividir presentación de lógica de negocio y acceso a datos

Presentación: Componentes que manejan solicitudes HTTP e implementan una API (REST) o una interfaz de usuario web basada en HTML.

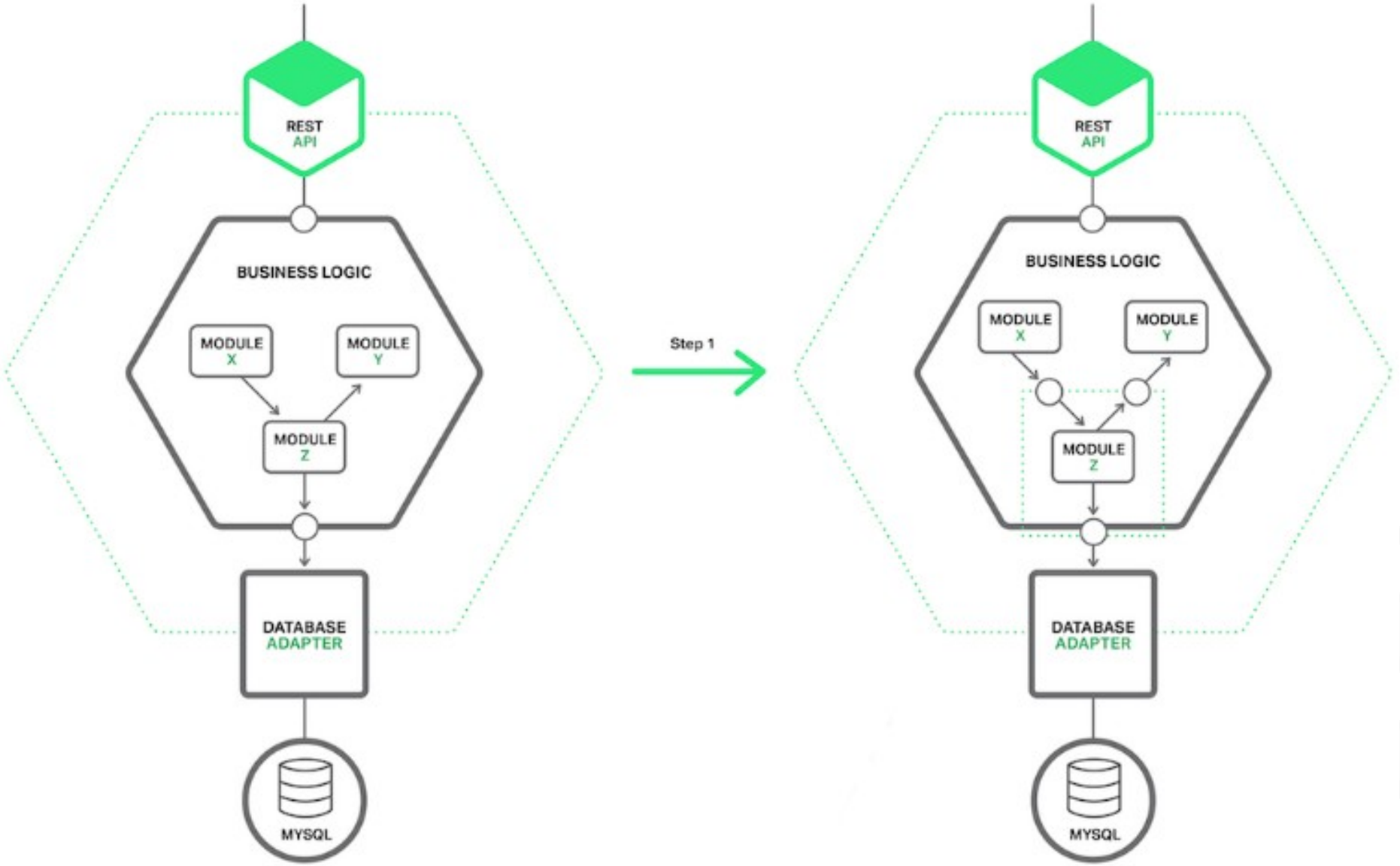
Lógica de negocios: Componentes que son el núcleo de la aplicación e implementan las reglas de negocios.

Acceso a datos: Componentes que acceden a componentes de infraestructura, como bases de datos y agentes de mensajes.



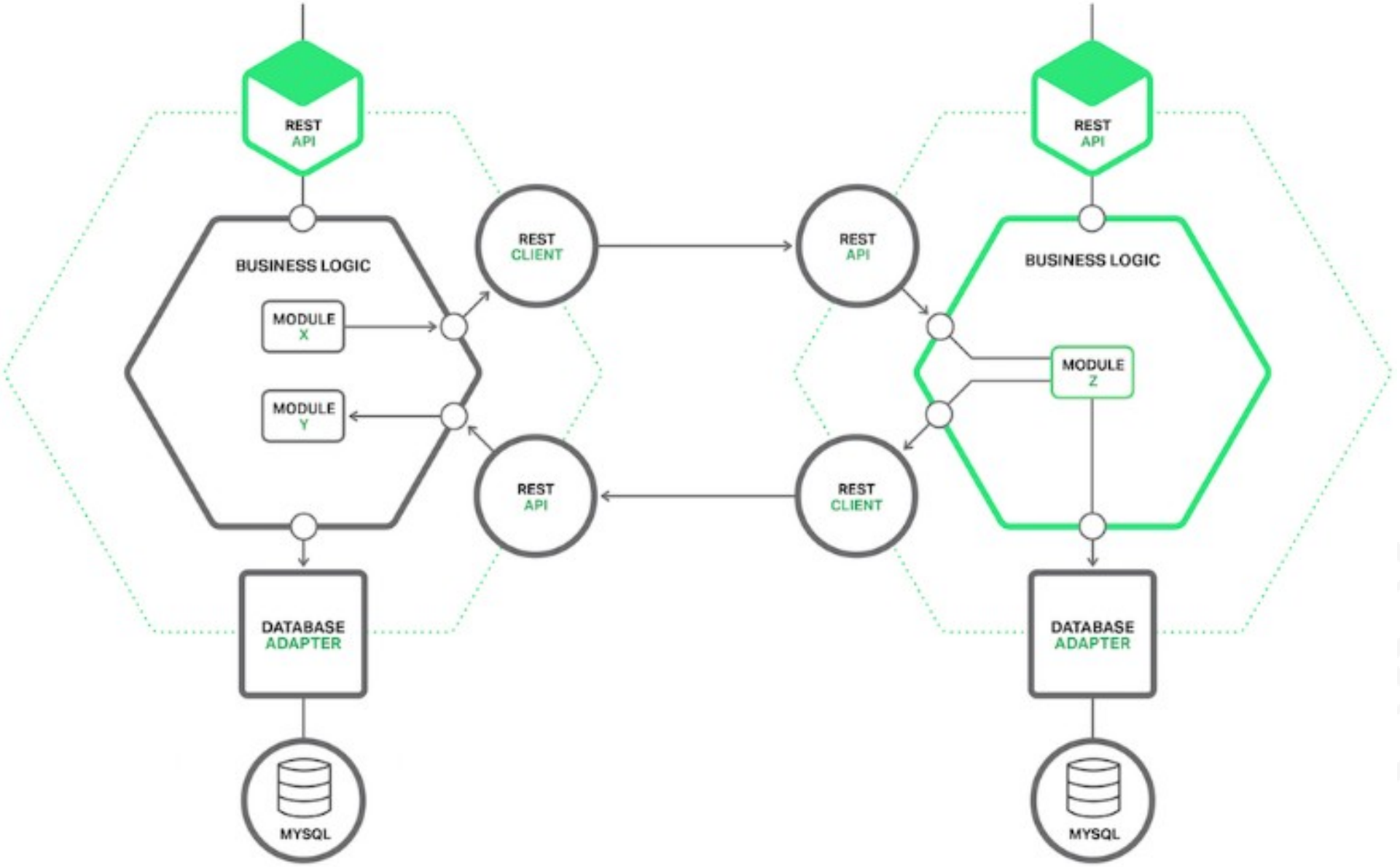
De monolítico a microservicios

Convertir módulos existentes del monolito en servicios



De monolítico a microservicios

Convertir módulos existentes del monolito en servicios



Estoy interesado, y ahora?

Material de Lectura:

- https://microservices.io/
- https://dzone.com/microservices-news-tutorials-tools
- https://netflix.github.io/

Cosas por aprender:

- DDD – Domain Driven Design
- Patrones de diseño: Event Sourcing, SAGA,
- IPC – Comunicación Inter Proceso: Asyn/Sync, Publish/Subscribe, REST, MBC (Bus de Mensajes → Apache Kafka)
- Laravel Lumen (PHP), SpringBoot(Java)
- Contenedores: Docker/Kubernetes
- Git