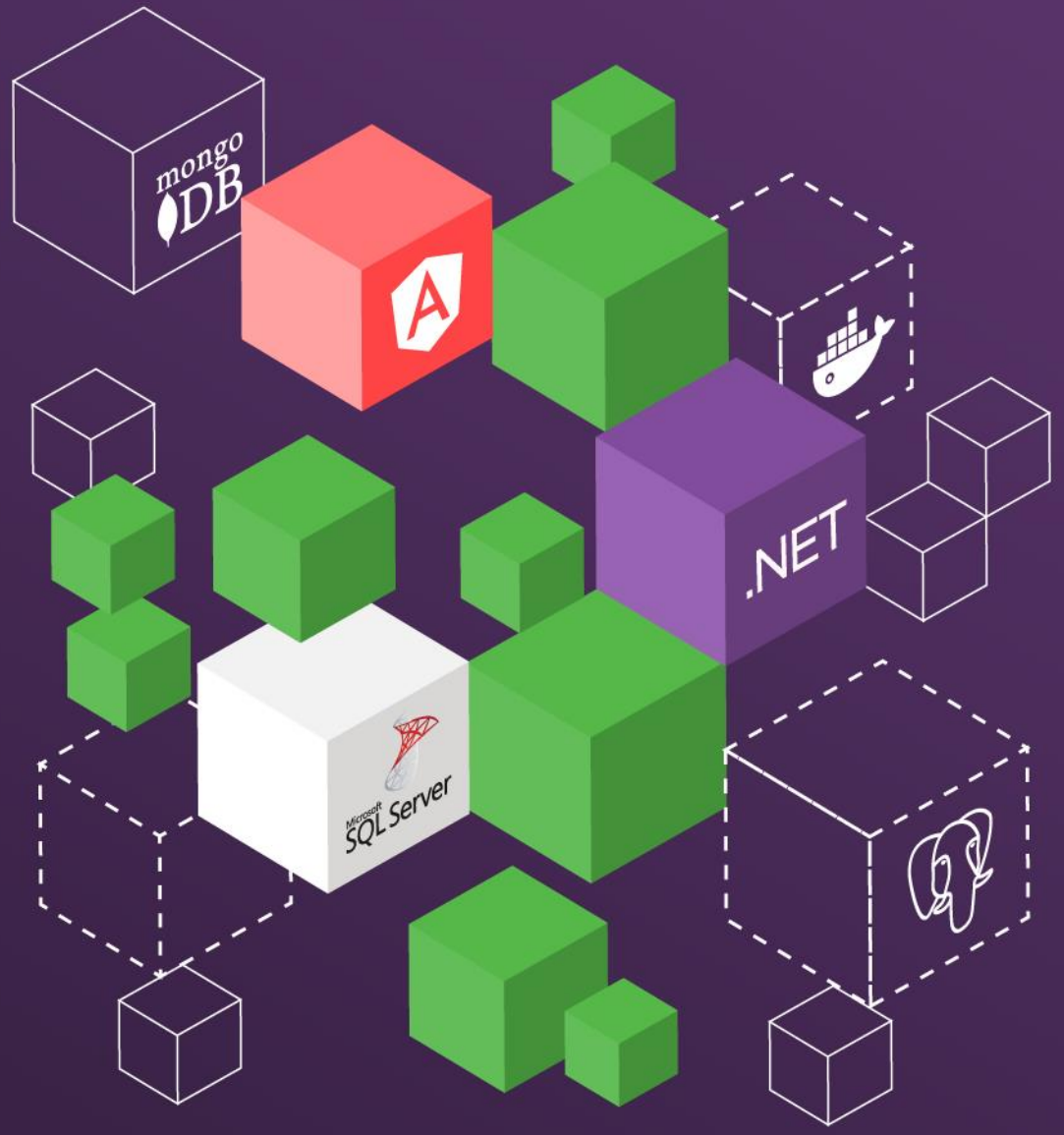


BIENVENIDOS
AL CURSO:

Arquitectura de Microservicios en Net

SESIÓN 01

GALAXY
TRAINING

ÍNDICE

ÍNDICE

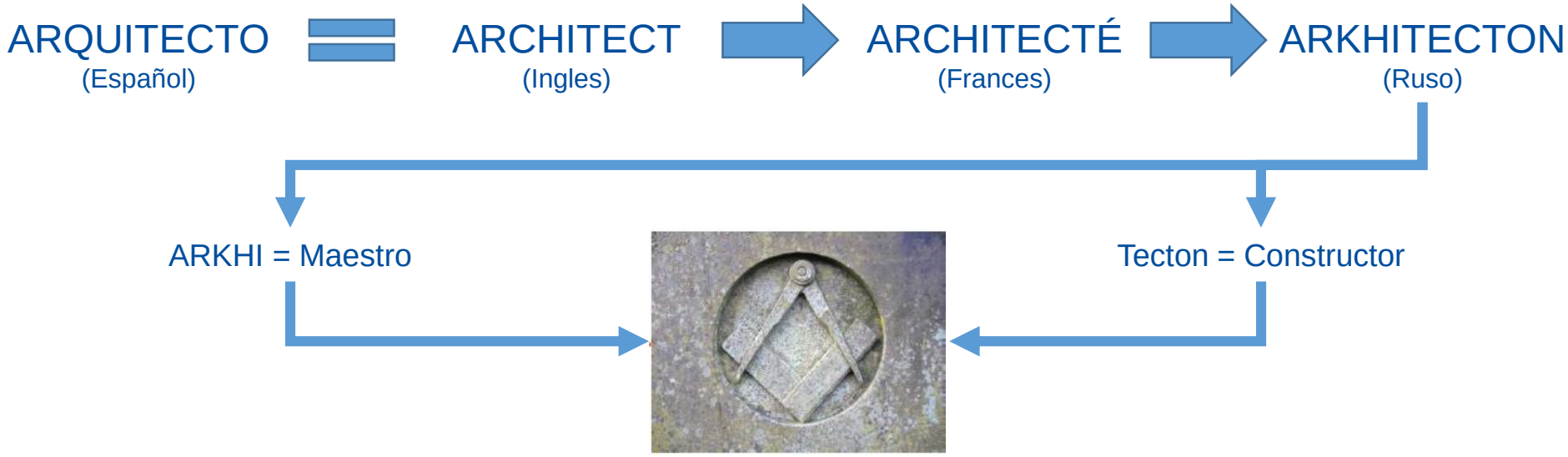
¿Qué es arquitectura de aplicación?

¿Qué es una empresa?

- Una sola organización
- Partes de una gran organización (como una unidad de negocios)
- Una colección de organizaciones que colaboran en una cadena de valor.
- La palabra "Empresa" cubre un amplio espectro de entidades organizacionales

¿Qué es arquitectura de aplicación?

Análisis etimológico de la arquitectura



La arquitectura es el oficio de los maestros constructores

¿Qué es arquitectura de aplicación?

Análisis etimológico de la arquitectura

La arquitectura empresarial se puede interpretar como el arte de crear un plan de ejecución para la empresa

¿Qué es arquitectura de aplicación?

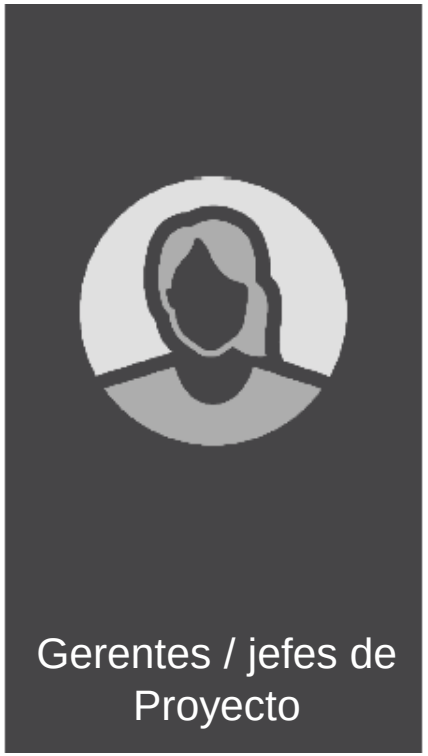
¿Por qué necesitamos arquitectura empresarial?

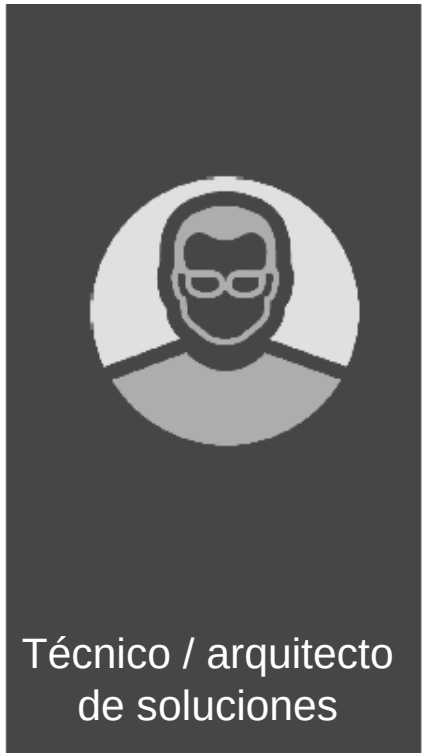
Históricamente, surgió como un mecanismo para manejar la complejidad de la implementación de sistemas de TI

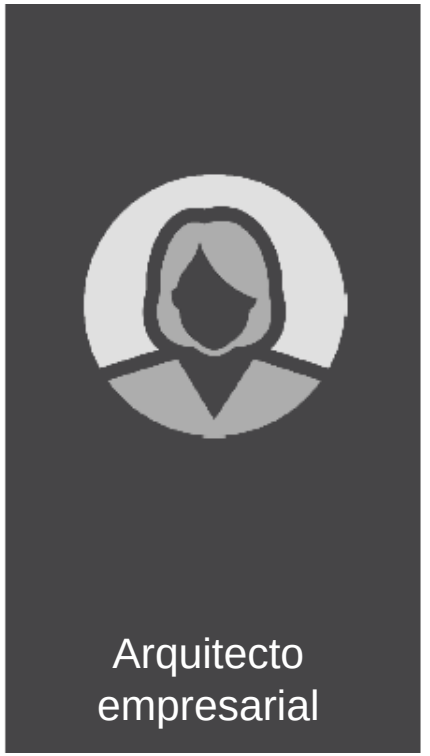
En el camino, el papel de EA se transformó para abordar la arquitectura de toda la empresa en lugar de solo los componentes de TI

¿Qué es arquitectura de aplicación?

Interesados

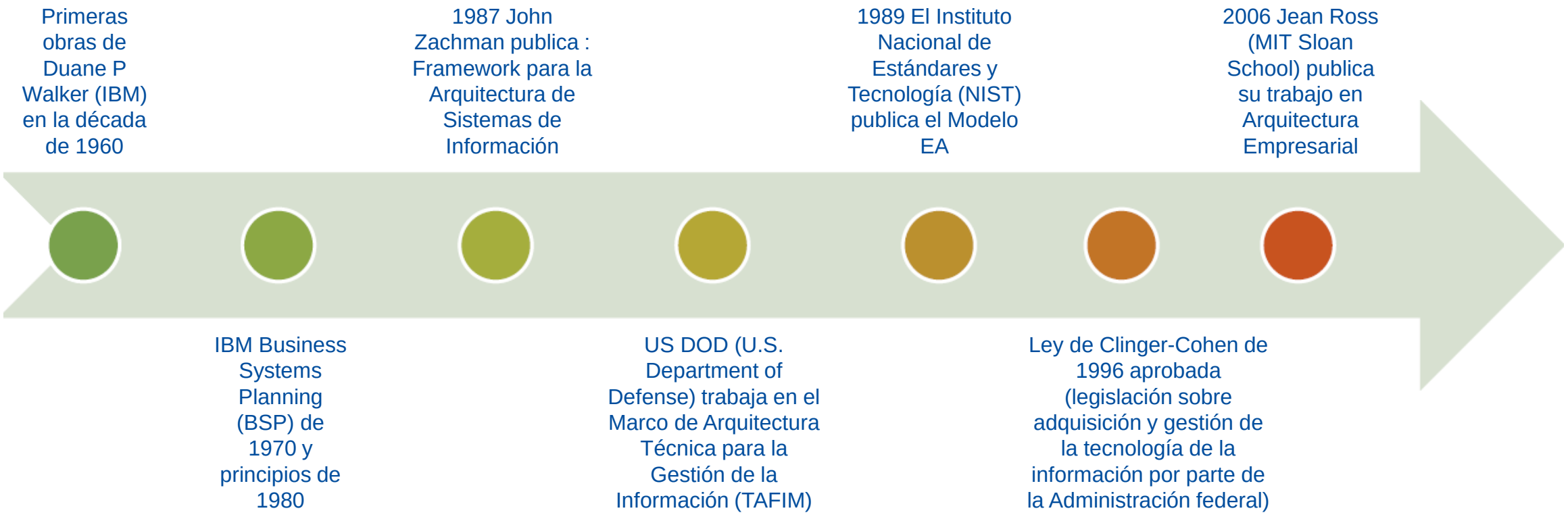






¿Qué es arquitectura de aplicación?

Arquitectura Empresarial línea de tiempo



¿Qué es arquitectura de aplicación?

Realidad actual

La realidad empresarial actual es significativamente diferente de la de los años sesenta y setenta
La naturaleza de los problemas, las oportunidades y el entorno empresarial que enfrentan las empresas hoy en día son significativamente diferentes a las décadas anteriores.

¿Qué es arquitectura de aplicación?

Realidad actual

Las empresas se han vuelto cada vez más dinámicas y complejas.
El papel de la TI dentro de las empresas tomó muchos giros inesperados en el camino.
La mayoría de las empresas ven la TI como una competencia básica fundamental
El ritmo del cambio solo se está acelerando

¿Qué es arquitectura de aplicación?

Arquitectura empresarial

La Arquitectura empresarial es una disciplina que permite diseñar la empresa de manera consciente y deliberada, en lugar de dejar que ocurra al azar.

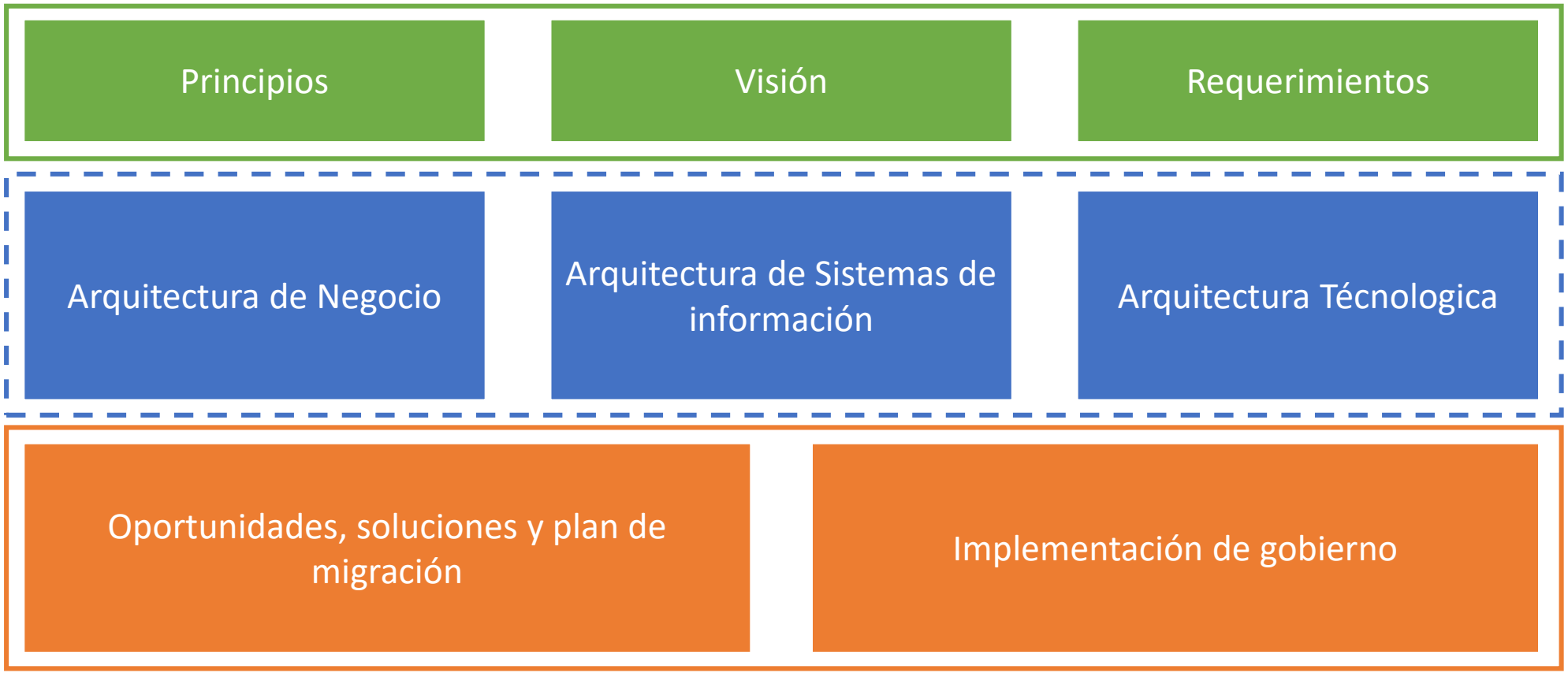
El diseño se basa en la visión empresarial, la intención estratégica y los conocimientos sobre el funcionamiento de la empresa.

¿Qué es arquitectura de aplicación?

La arquitectura empresarial adopta vistas tanto atómicas como holísticas

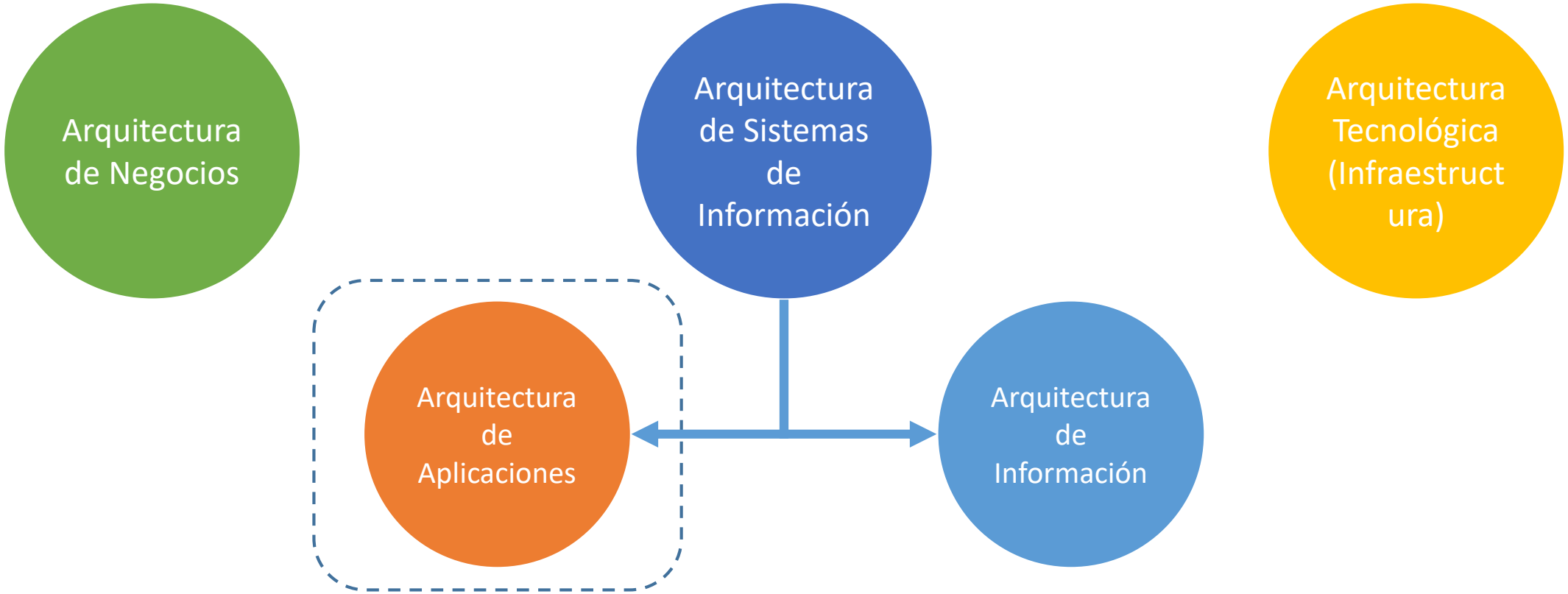
La arquitectura empresarial adopta vistas tanto atómicas como holísticas

Contenido de la arquitectura empresarial



¿Qué es arquitectura de aplicación?

Dominios de arquitectura empresarial



¿Qué es arquitectura de aplicación?

En las últimas décadas, las empresas han invertido mucho en aplicaciones comerciales en todos los sectores industriales.

¿Qué es arquitectura de aplicación?

El arquitecto de aplicaciones empresariales crea una hoja de ruta de cartera de aplicaciones de estado objetivo, teniendo en cuenta:

Costo total de cambio
Rendimiento de las inversiones
Riesgos y camino de menor resistencia.

¿Qué es arquitectura de aplicación?

El estado objetivo podría incluir

- Brechas identificadas en las capacidades de la aplicación
- Decisión de retirar el envejecimiento y las aplicaciones de bajo valor
- Modernización de aplicaciones heredadas pero de alto valor
- Eliminar la redundancia
- Estandarización en plataforma tecnológica común
- Consolidando aplicaciones

¿Qué es arquitectura de aplicación?

¿Qué es la arquitectura de software?

Cuando las personas en la industria del software hablan de "**arquitectura**", se refieren a una noción definida de los aspectos más importantes del diseño interno de un sistema de software.

Una buena arquitectura es importante, de lo contrario se vuelve más lento y más caro agregar nuevas capacidades en el futuro.

https://martinfowler.com/architecture/

Arquitectura de aplicaciones

Las decisiones importantes en el desarrollo de software varían con la escala del contexto en el que estamos pensando. Una escala común es la de una aplicación, por lo tanto, "**arquitectura de aplicación**".

El primer problema con la definición de la arquitectura de aplicación es que no hay una definición clara de lo que es una aplicación. Mi opinión es que las aplicaciones son una construcción social:

- Un cuerpo de código que los desarrolladores ven como una sola unidad
- Un grupo de funcionalidades que los clientes empresariales ven como una sola unidad
- Una iniciativa que los que tienen el dinero ven como un presupuesto único

Una definición tan suelta conduce a muchos tamaños potenciales de una aplicación, que varían de unas pocas a unos pocos cientos de personas en el equipo de desarrollo. (Te darás cuenta de que miro el tamaño como la cantidad de personas involucradas, que creo que es la forma más útil de medir esas cosas.) La diferencia clave entre esto y la arquitectura empresarial es que hay un grado significativo de propósito unificado en torno a la construcción social.

https://martinfowler.com/architecture/

¿Qué es arquitectura de aplicación?

Arquitectura de aplicaciones y arquitectura empresarial

Mientras que la arquitectura de aplicaciones se concentra en la arquitectura dentro de algún tipo de límite de aplicación, la arquitectura empresarial mira la arquitectura en una gran empresa.

Tal organización suele ser demasiado grande para agrupar todo su software en cualquier tipo de agrupación cohesiva, por lo que requiere coordinación entre equipos con muchas bases de código, que se han desarrollado de forma aislada entre sí, con financiación y usuarios que operan independientemente de Uno al otro.

Gran parte de la arquitectura empresarial consiste en comprender lo que vale la pena en los costos de la coordinación central y qué forma debe tomar esa coordinación.

https://martinfowler.com/architecture/

¿Qué es arquitectura de aplicación?

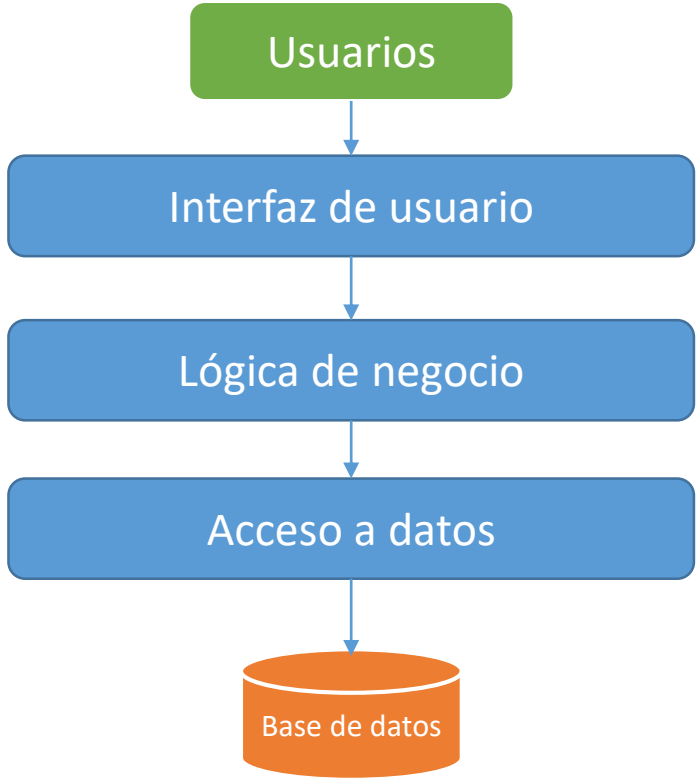
Considere la complejidad de :

- Codificación a una interfaz
- Servicios
- Pruebas automatizadas
- Domain Driven Design
- Acceso a datos
- Arquitectura en capas

¿Qué es arquitectura de aplicación?

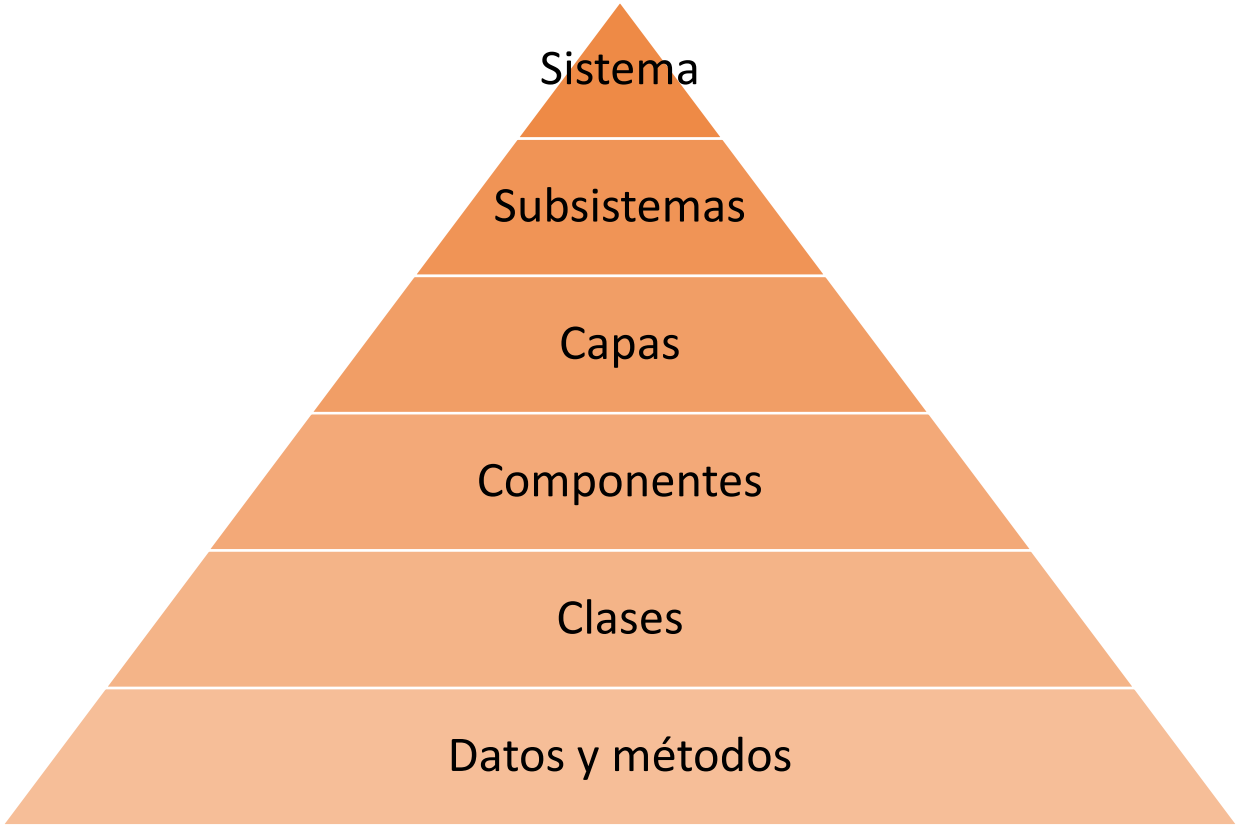
¿Qué es la arquitectura de software?

Estructura a alto-nivel de capas, componentes y la relación entre estas.



¿Qué es arquitectura de aplicación?

Niveles de abstracción arquitectónica



¿Qué es arquitectura de aplicación?

Arquitectura desordenada vs limpia

¿Qué es arquitectura de aplicación?

¿Cuándo una arquitectura es mala?

- Compleja
- Incoherente
- Rígido
- Frágil
- Inestable
- Insostenible

¿Qué es arquitectura de aplicación?

¿Cuándo una arquitectura es buena?

- Sencilla
- Comprensible
- Flexible
- Emergente
- Testeable
- **Mantenible**

¿Qué es arquitectura de aplicación?

¿Qué es la arquitectura limpia?

Habitantes (Usuarios)

Arquitecto

Maquina

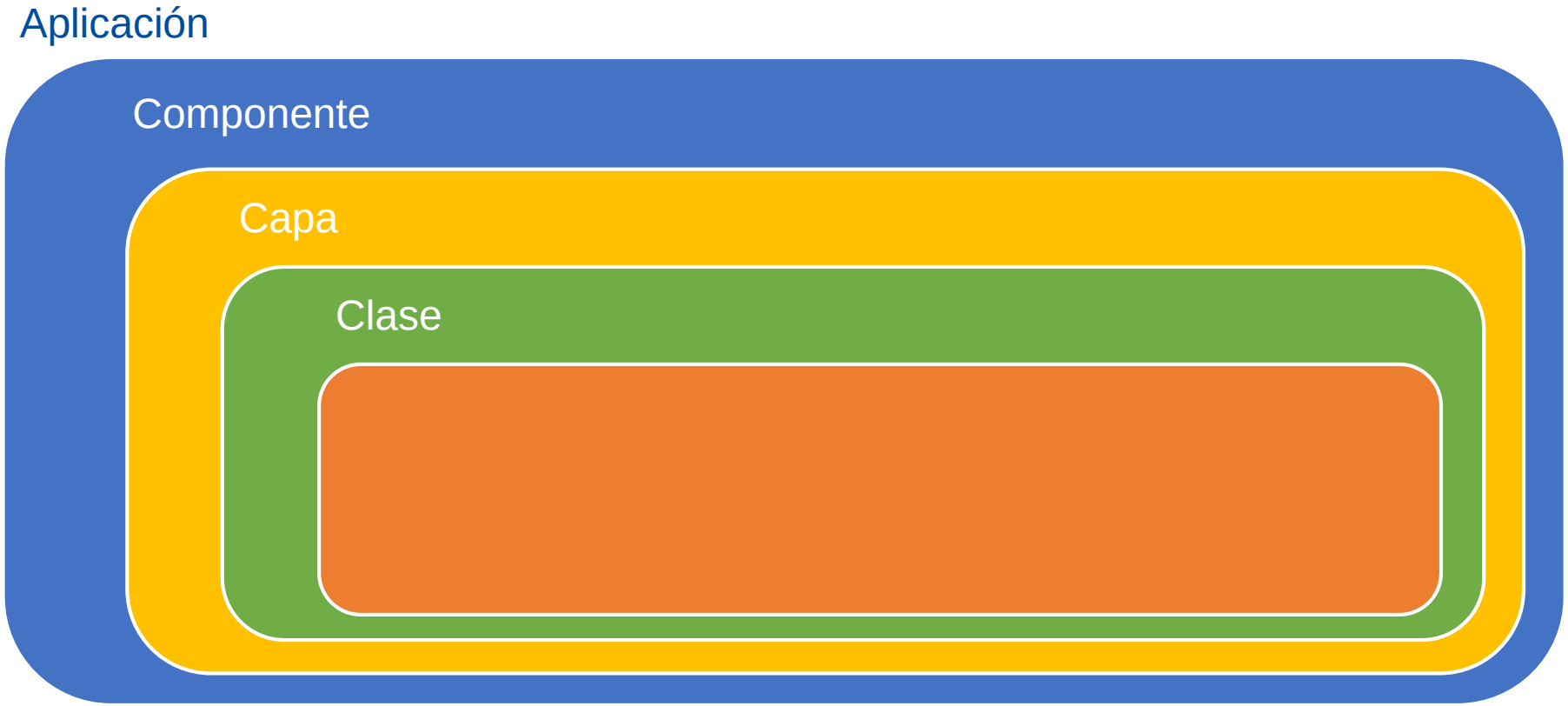
¿Qué es arquitectura de aplicación?

Aplicaciones monolíticas

Aplicación monolítica

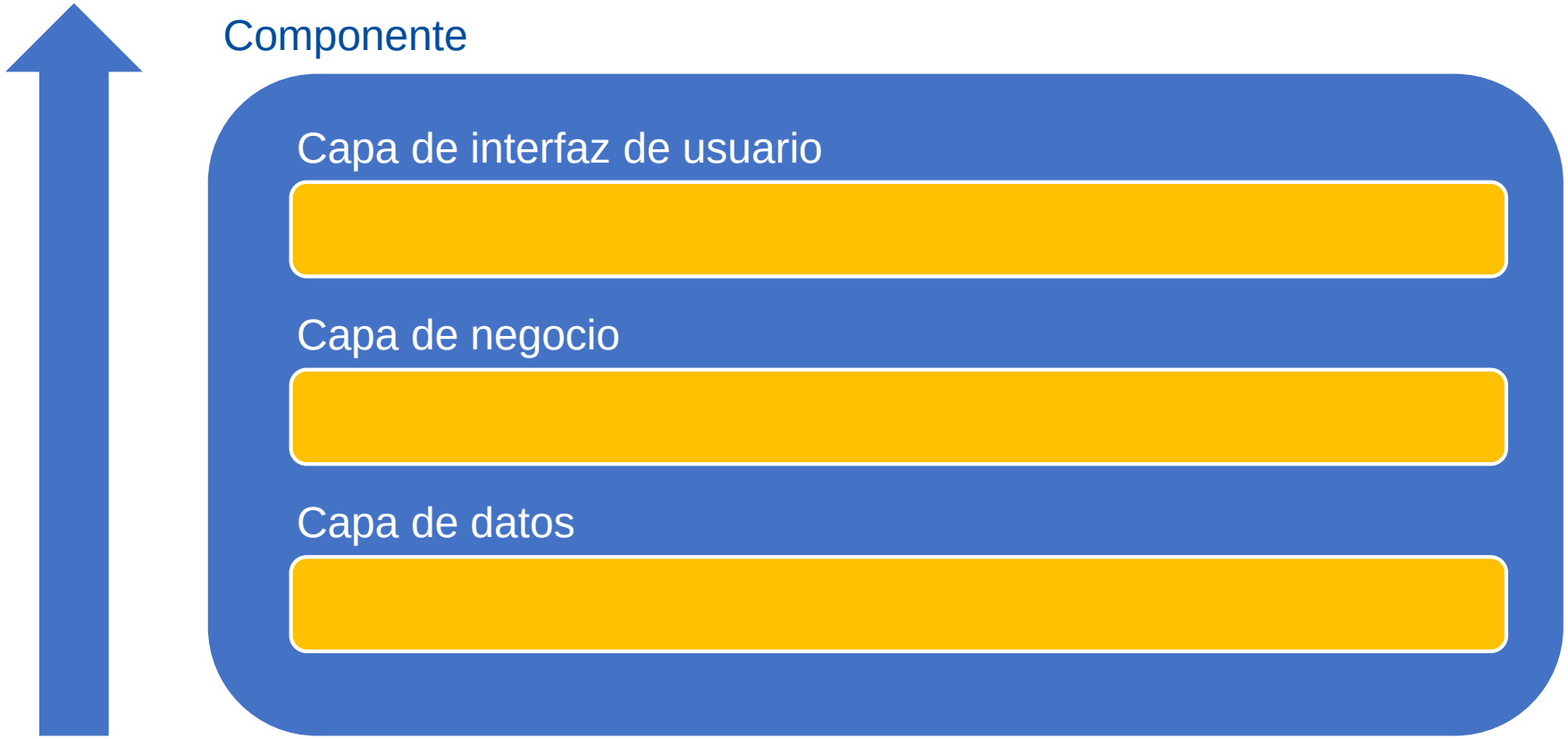
Una aplicación de software de un solo nivel en el que la interfaz de usuario y el código de acceso a datos se combinan en un solo programa desde una sola plataforma

Arquitectura general de una aplicación

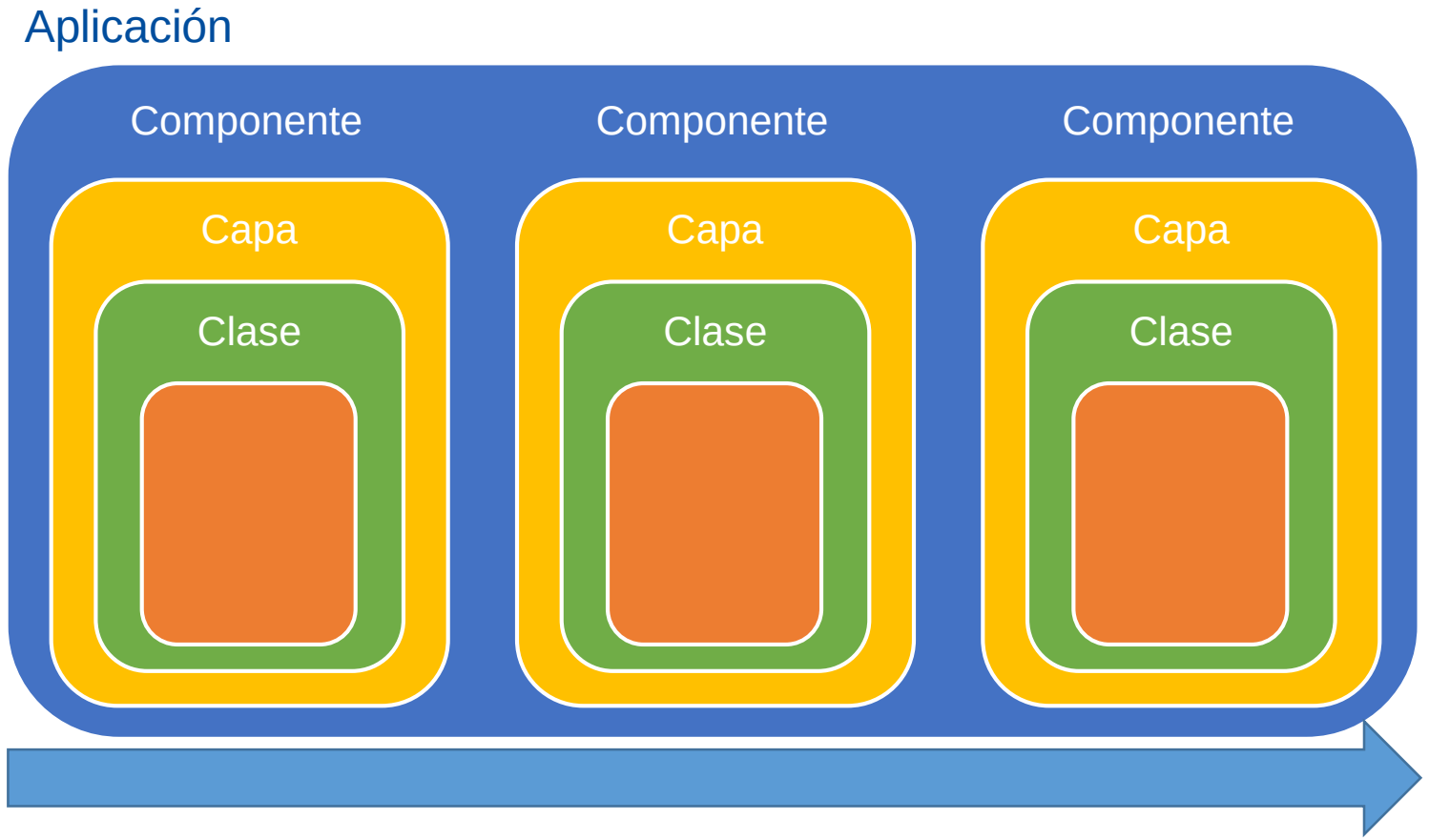


Aplicaciones monolíticas

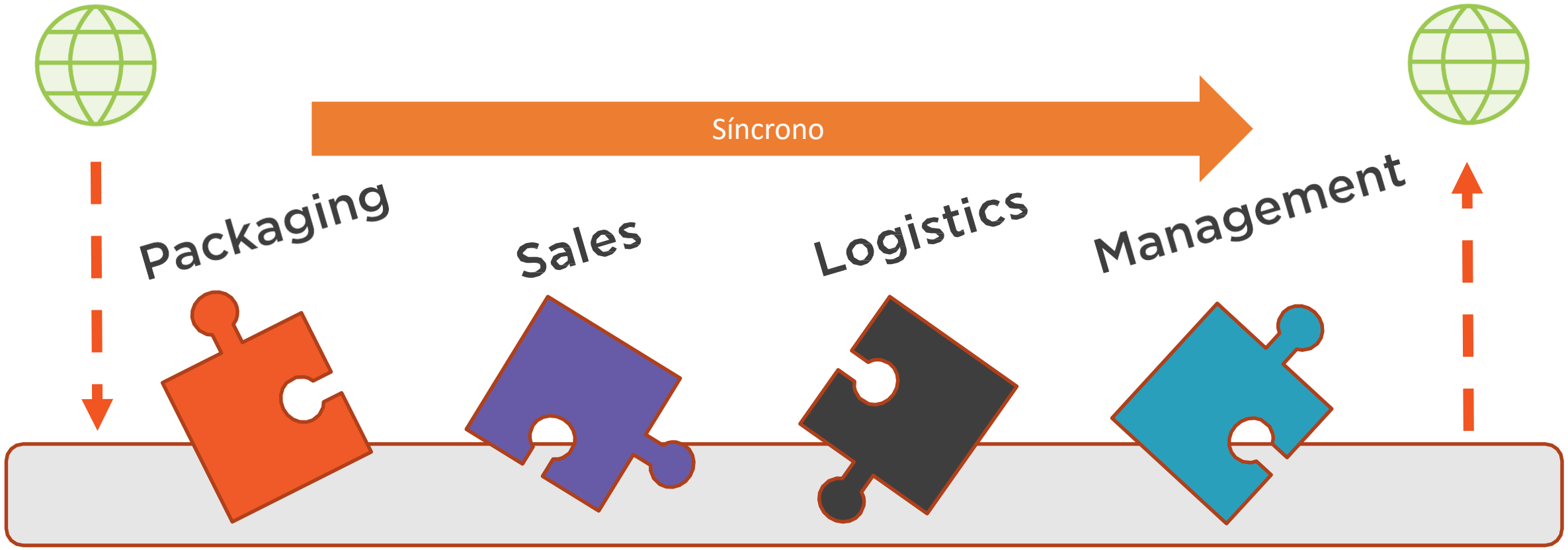
Acoplamiento vertical



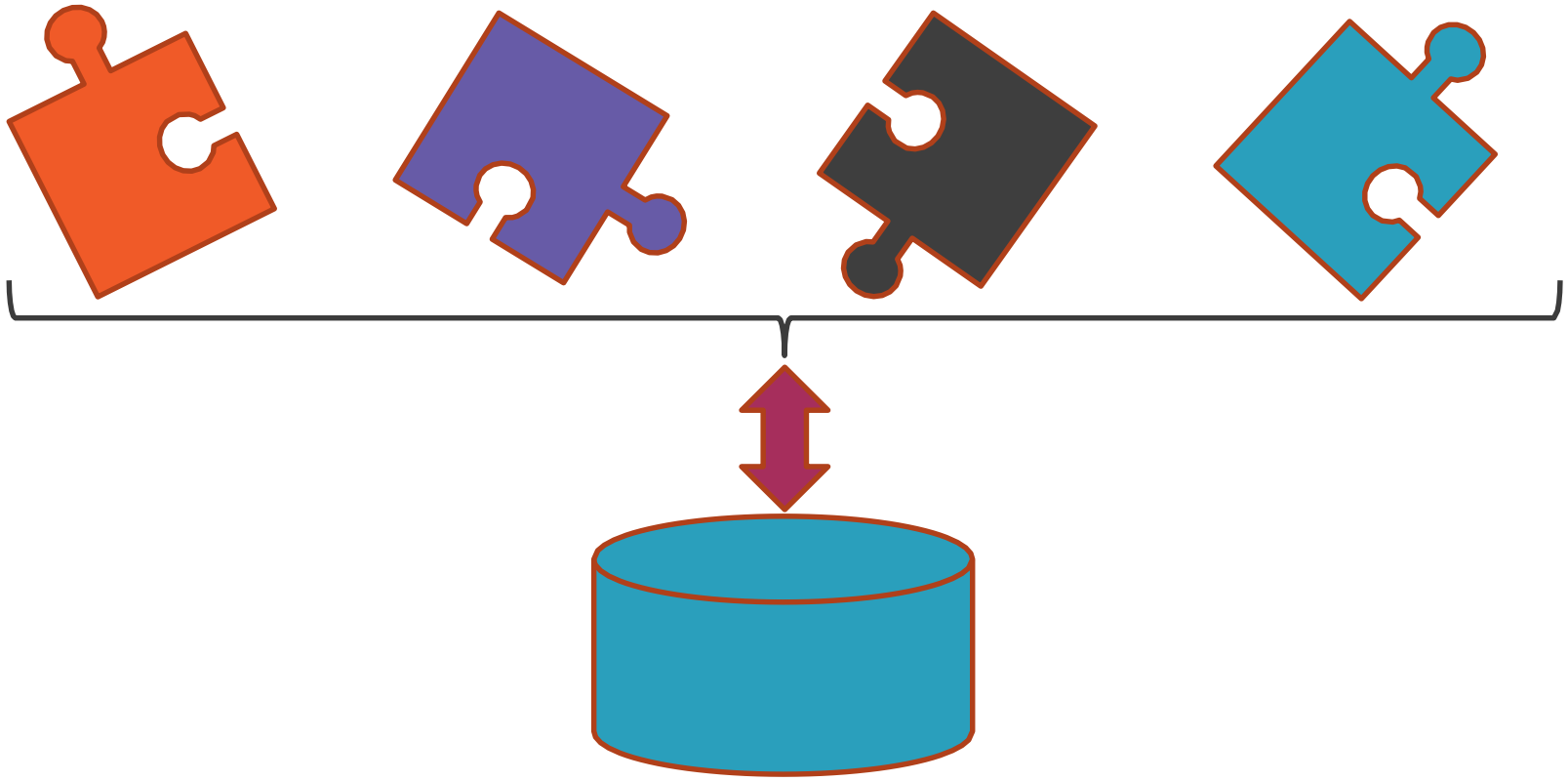
Acoplamiento horizontal



Manejo de solicitudes



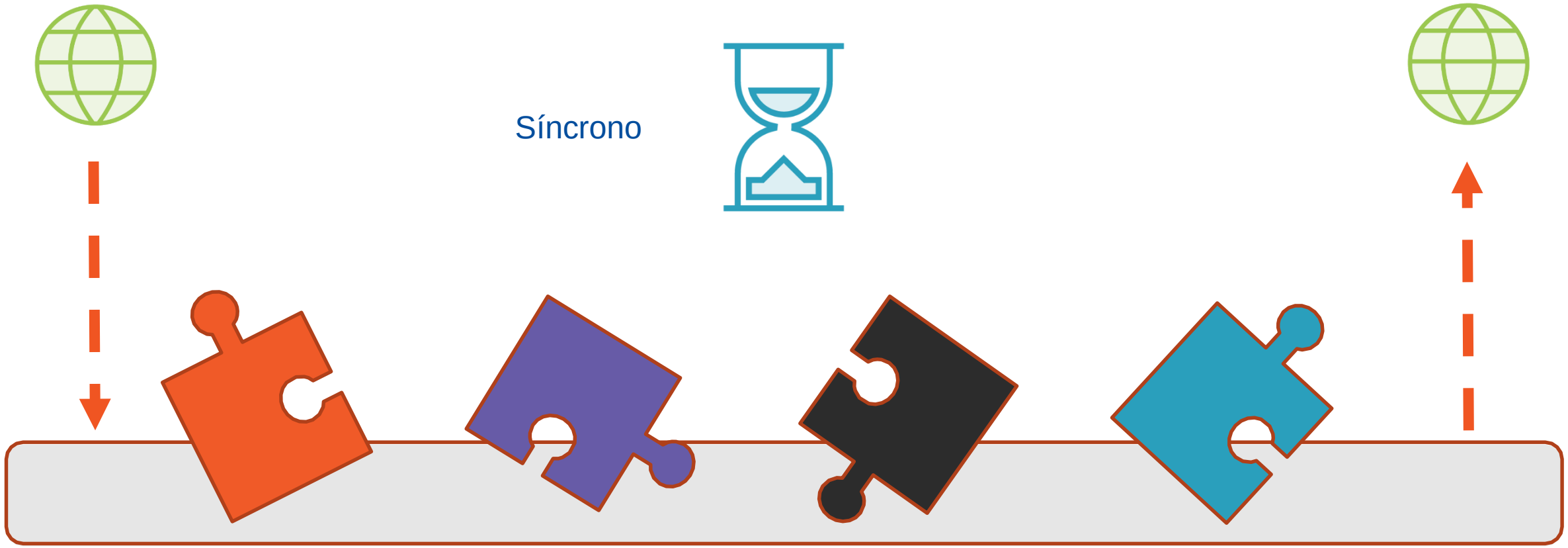
Manejo de solicitudes



Beneficios de las aplicaciones monolíticas

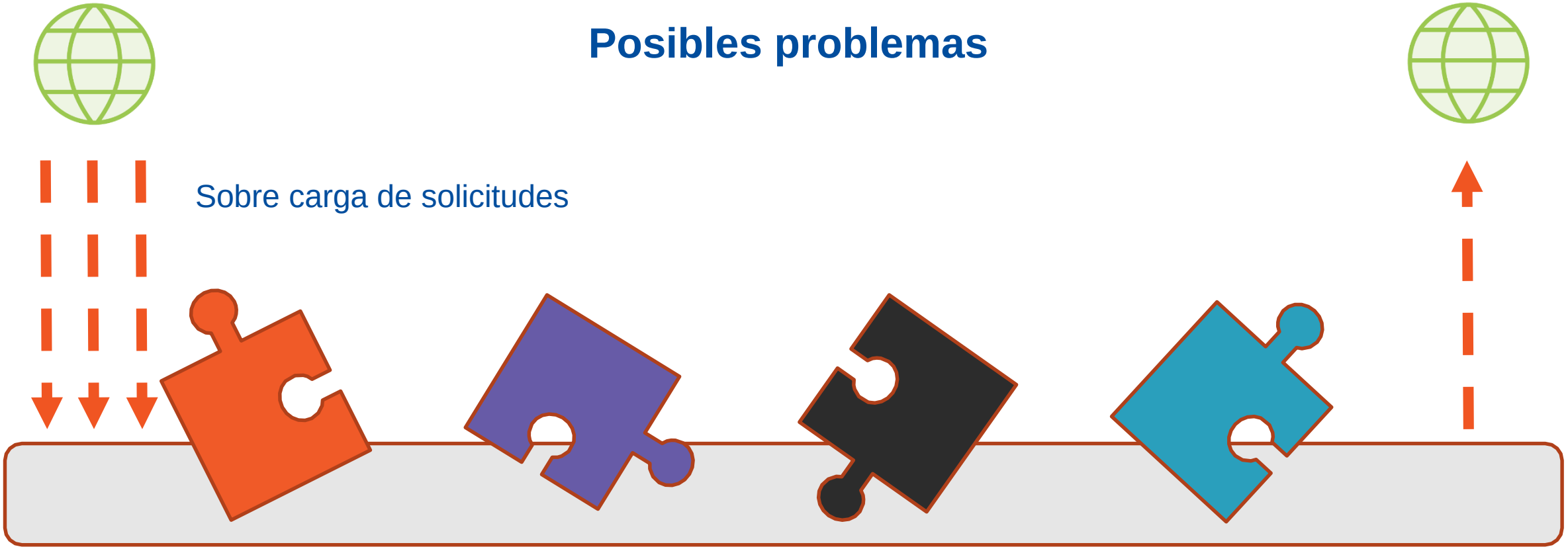
- Fácil de desplegar
- Bien conocidas
- Sin dependencias externas
- Amigable con los IDE’s

Posibles problemas



Posibles problemas

Posibles problemas



Posibles problemas

Desventajas de las aplicaciones monolíticas

- Complejo y difícil de mantener
- Tiende a complicarse con el tiempo
- Despliegue trabajoso (Coordinación)
- Problemas de rendimiento
- Baja confiabilidad (Degradación)
- One Stack (Una sola tecnología)

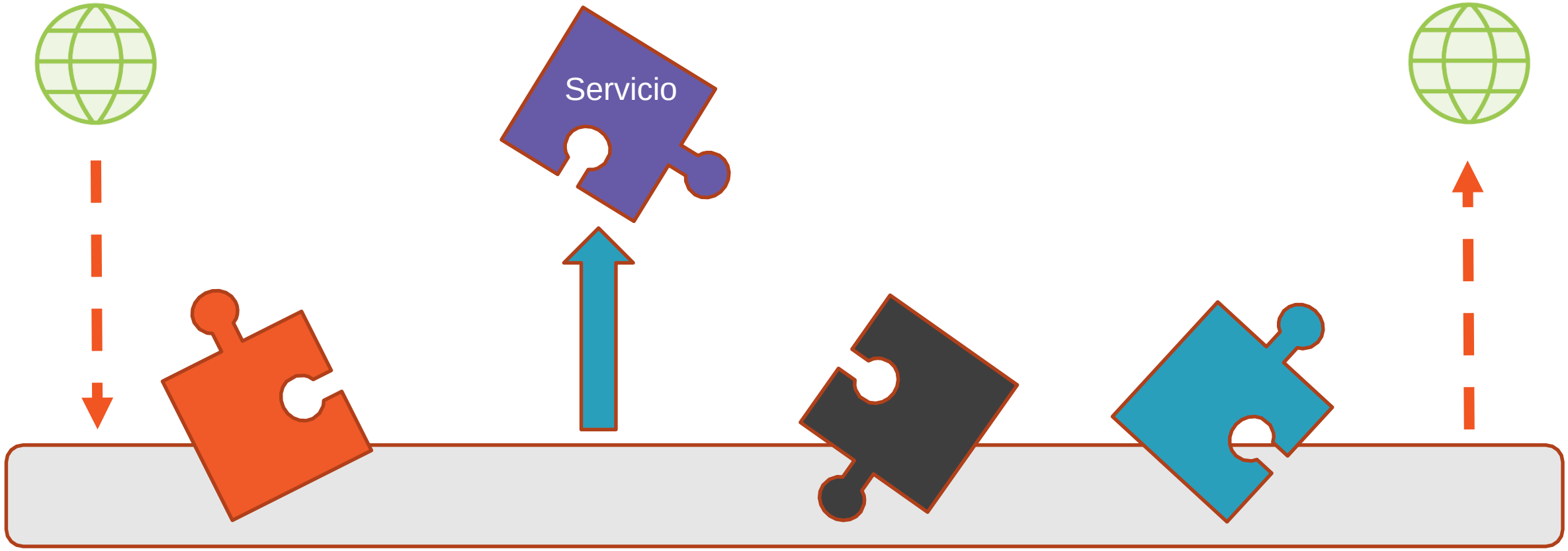
Demo simplificada

Aplicaciones SOA

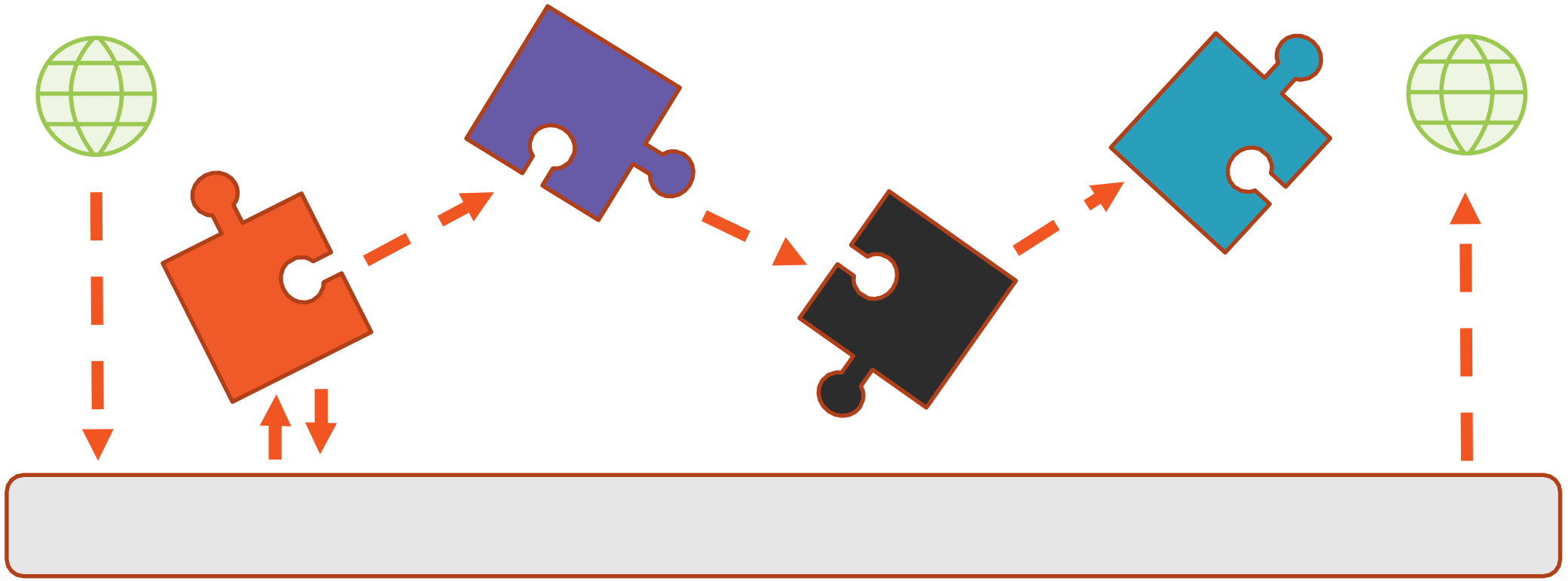
Sistema Distribuido

Un sistema distribuido es una aplicación de software en la que los componentes están ubicados en computadoras en red y se comunican y coordinan sus acciones emitiendo llamadas o pasando mensajes

Componentes distribuidos



Service-oriented Architecture



¿Cuándo una arquitectura es buena?

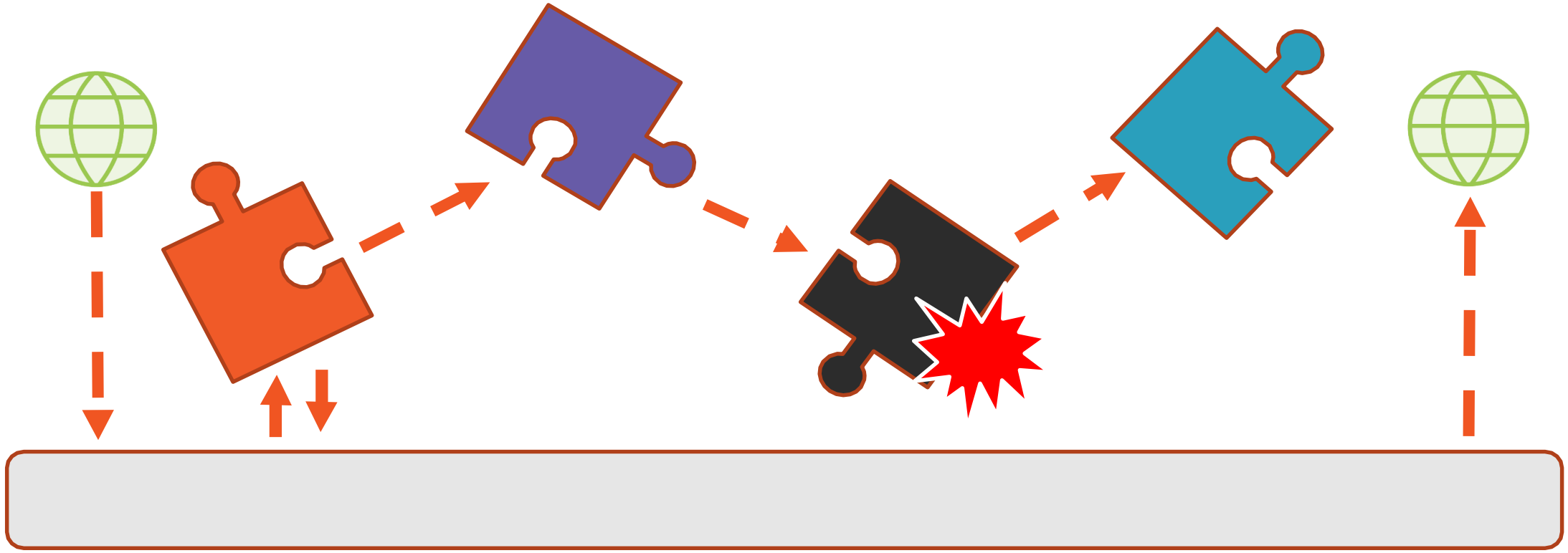
Falacias de la computación distribuida

- La red es confiable
- La latencia es cero
- El ancho de banda es infinito
- La red es segura
- La topología no cambiará
- Hay un administrador
- El costo de transporte es cero
- La red es homogénea.

Acoplamiento

Plataforma	Comportamiento	Temporal
(Asociado a una misma tecnología)	(Conocimiento de la firma)	(Dependencia entre servicios)

Remote Procedure Call



¿Qué es Arquitectura SOA?

Modelo/Estilo de arquitectura fundamentado en el paradigma de diseño de la **orientación a servicios**.

Evolución de la industria en metodologías, tecnología y estándares.

Nueva generación de la **computación distribuida**.

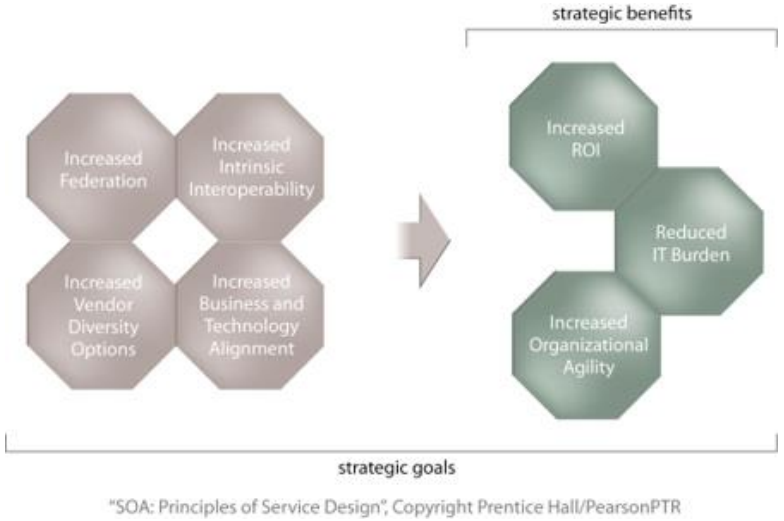
Brinda beneficios estratégicos a las organizaciones.

Define los siguientes elementos :

- Servicios
- Contratos
- Mensajes

Aplicaciones SOA

¿Qué es Arquitectura SOA?



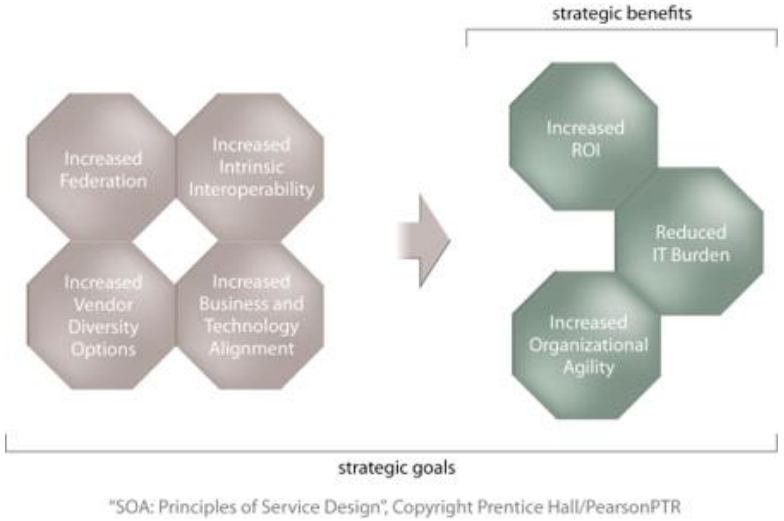
"SOA: Principles of Service Design", Copyright Prentice Hall/PearsonPTR

Objetivos y beneficios estratégicos.

- Incrementar la interoperabilidad intrínseca.
- Incrementar la alineación de TI con el Negocio.
- Agilidad Organizacional.
- Incrementar el ROI

Aplicaciones SOA

¿Qué es Arquitectura SOA?

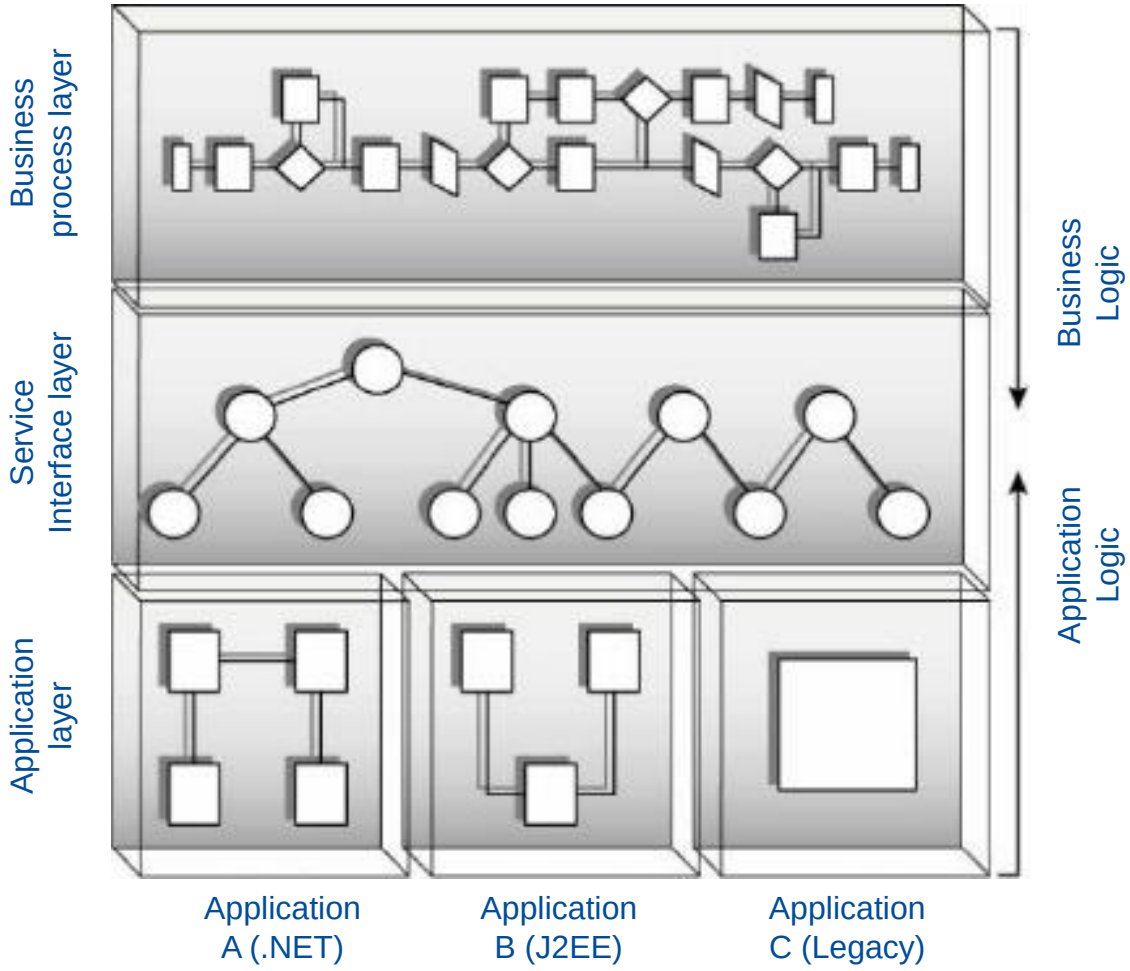


"SOA: Principles of Service Design", Copyright Prentice Hall/PearsonPTR

- Todos los objetivos son estratégicos en naturaleza, brindando beneficios a largo plazo.
- En comparación con objetivos tácticos, los cuales se basan en requerimientos inmediatos a corto plazo.
- Una característica distintiva con la computación orientada a servicios es su naturaleza estratégica.
- Contrario a la naturaleza táctica del desarrollo de aplicaciones basadas en silos.

Aplicaciones SOA

Elementos



Procesos

- Personas
- Reglas
- Lógicas
- Mensajes

Orchestration service layer
Task-Centric Business services
Entity-Centric Business services
Utily Application services

- Contratos
- Lógicas
- Mensajes

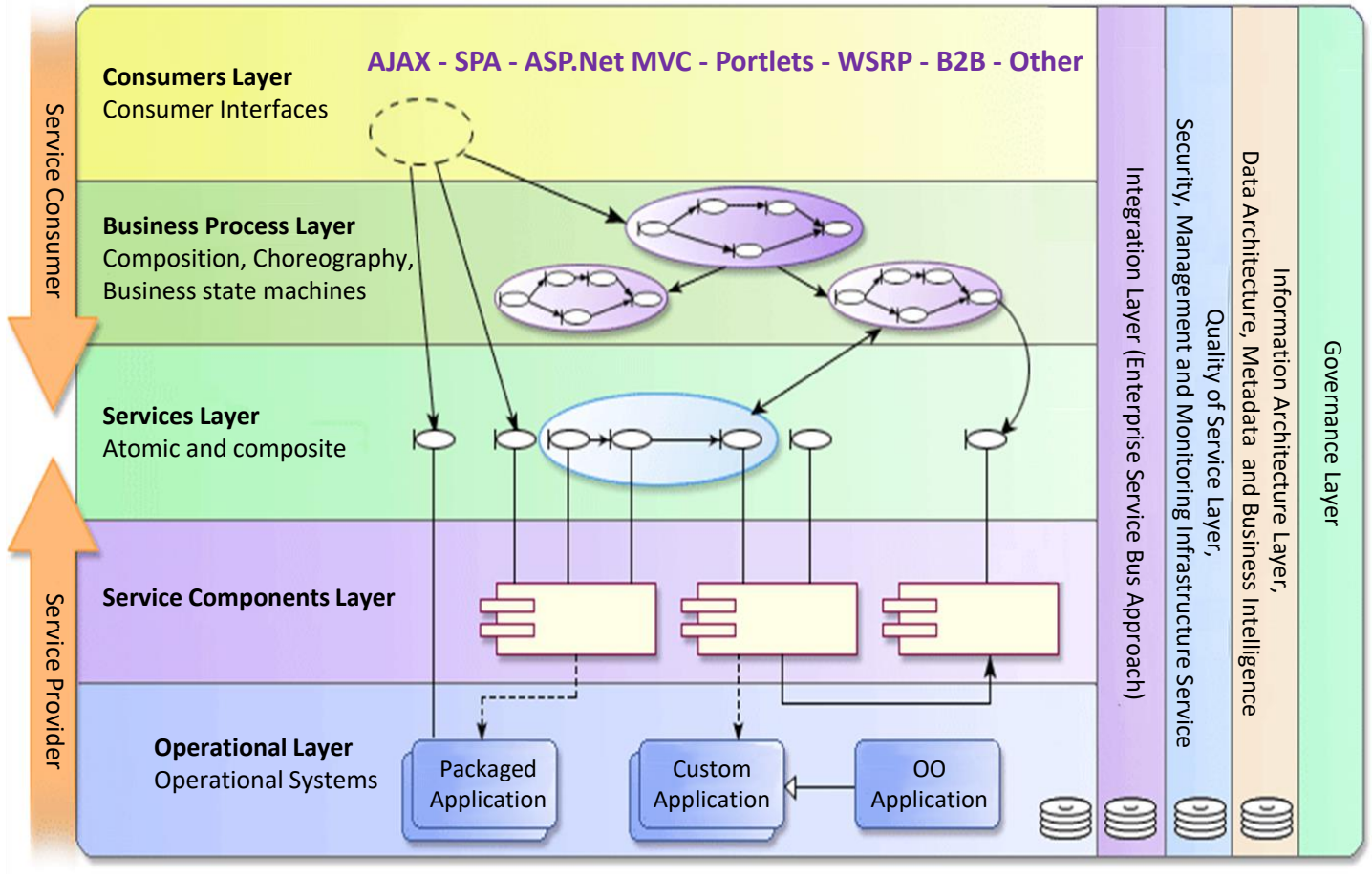
Recursos de TI

- Legados
- Base de datos
- Componentes
- Frameworks
- Redes
- Otros

Elementos

4 Capas relacionadas
al soporte transversal

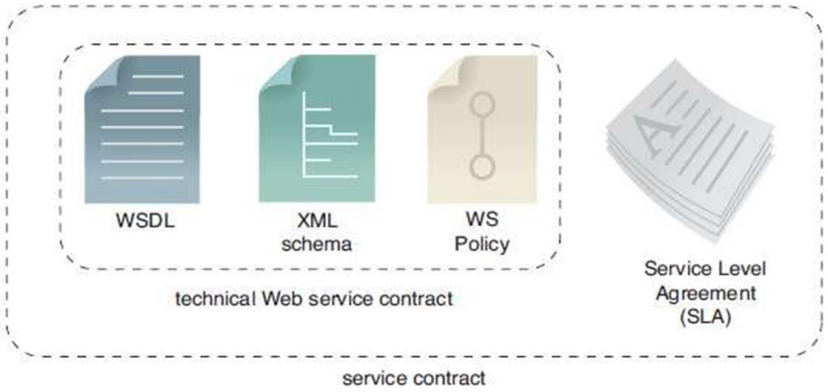
5 Capas relacionadas
a las capacidades
del negocio



Elementos - Servicio

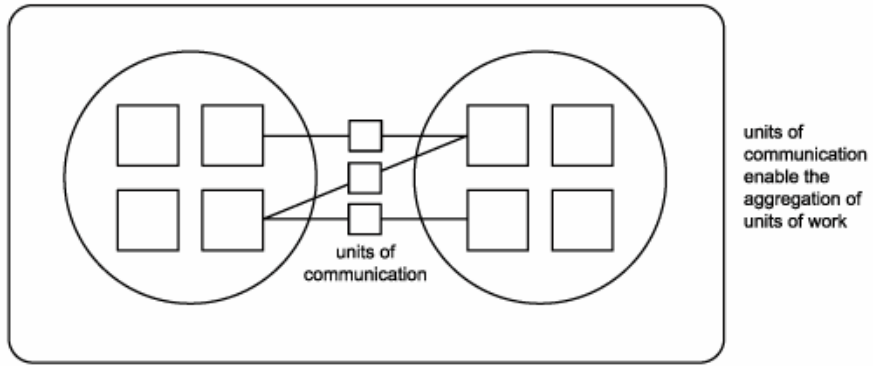
- Unidad fundamental de SOA
- Un contenedor de una o muchas capacidades
- Diseñado con capacidades reutilizables
- Servicios como: Web Services, Components, REST Services.

Elementos – Contratos



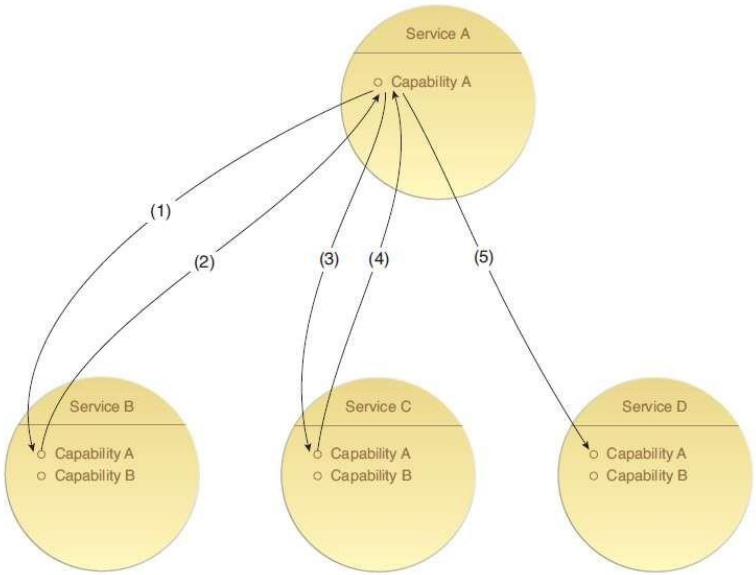
- El contrato expresa información acerca del servicio, sus capacidades y **tipos de datos**.
- Un servicio consumidor debe cumplir con los requerimientos expuestos en el contrato.
- La parte fundamental del contrato del servicio es su **interface técnica**.
- El contrato del servicio se puede complementar con un documento de SLA que describa información adicional de características y limitaciones.
- El como se diseñen y existan **físicamente** los servicios depende sobre la tecnología que se utiliza para crear el servicio.
- El principio de **estandarización del contrato** es dedicado a la definición del contrato, aspecto muy importante para lograr beneficios estratégicos.

Elementos – Mensajes



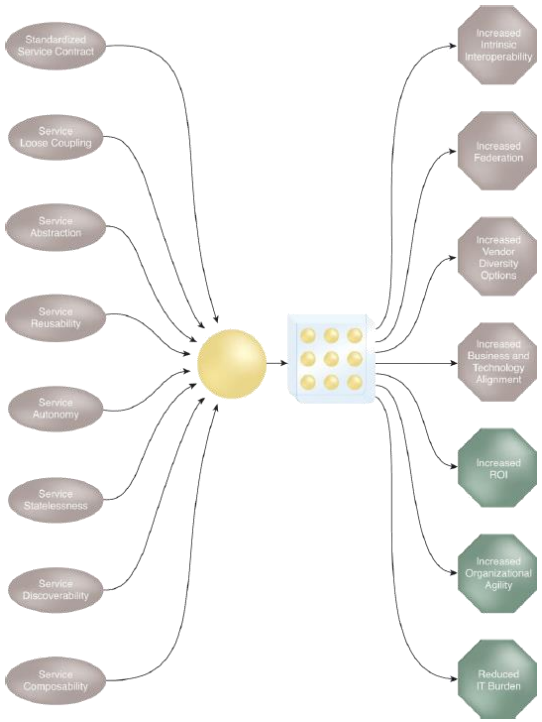
- Unidad de comunicación de los servicios.
- La estandarización de los mensajes permite:
  - La interoperabilidad intrínseca
  - Reducir costos de mantenimiento
  - Eliminar el uso de transformaciones que impactan el rendimiento

Composición



- Agregación coordinada de servicios
- Reutilización de capacidades para diferentes procesos de negocio

Principios de Diseño



- Estandarización del contrato
- Bajo acoplamiento del servicio
- Abstracción del Servicio
- Capacidad del Servicio de ser Reutilizado
- Autonomía del Servicio
- Servicio sin Estado
- Capacidad del Servicio de ser Descubierto e interpretado
- Capacidad del Servicio de ser Compuesto

Aplicaciones SOA

RPC, REST, GraphQL y gRPC.

Remote Procedure Call

Llamar a un metodo de manera remota
.NET Remoting/Java RMI

Remote Procedure Call

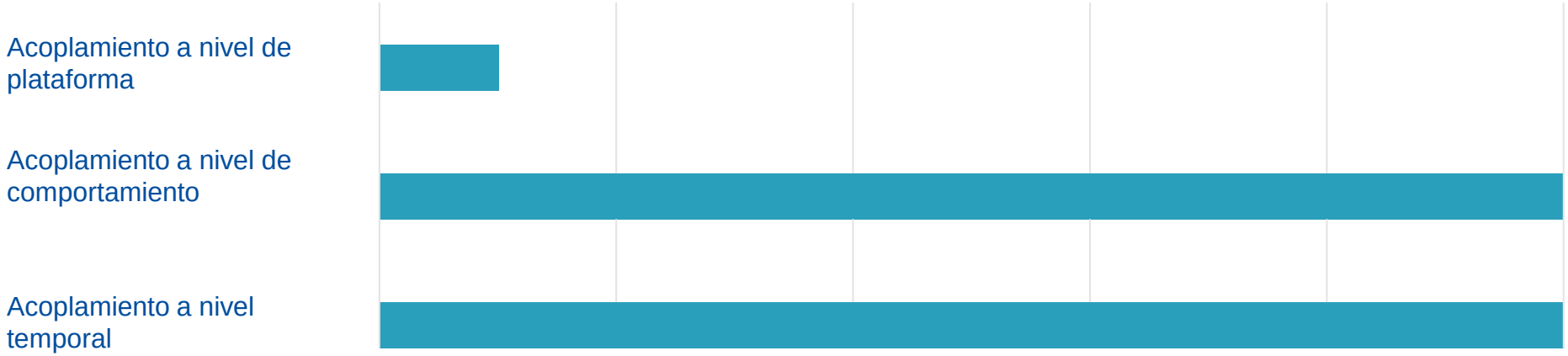
Remote Procedure Call (RPC)

- Clases de proxy
- ¡Cuidado con las falacias de la informática distribuida!

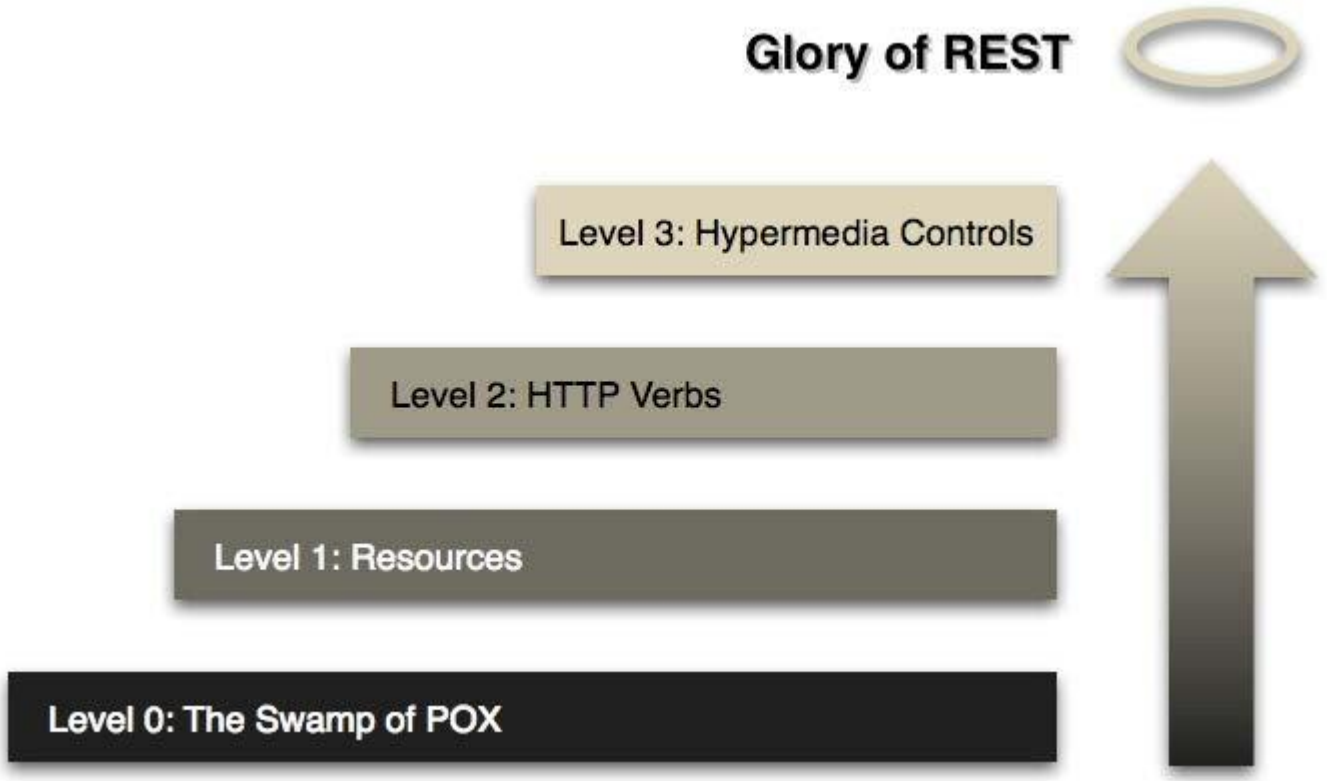
Remote Procedure Call: SOAP

Simple Object Access Protocol
(SOAP)

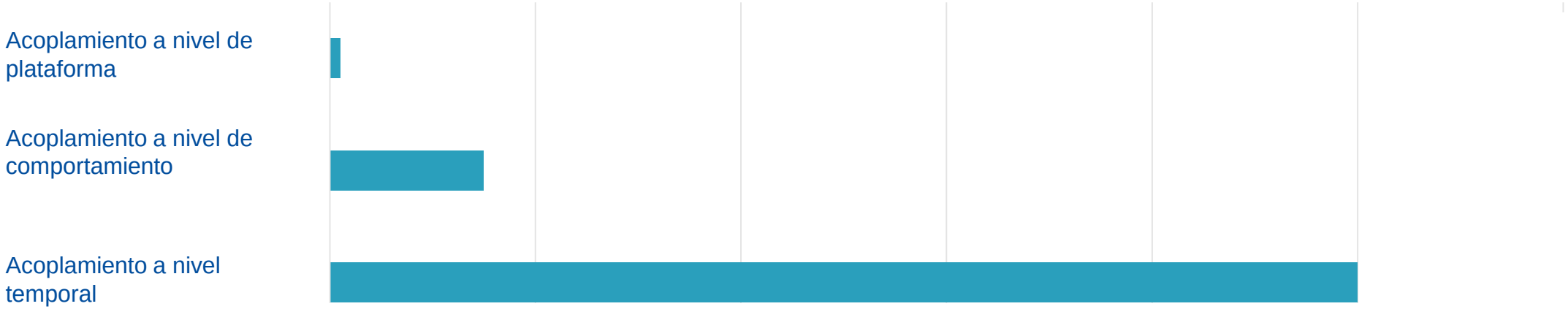
Call method using standarized XML
(Web Services Description Language WSDL)

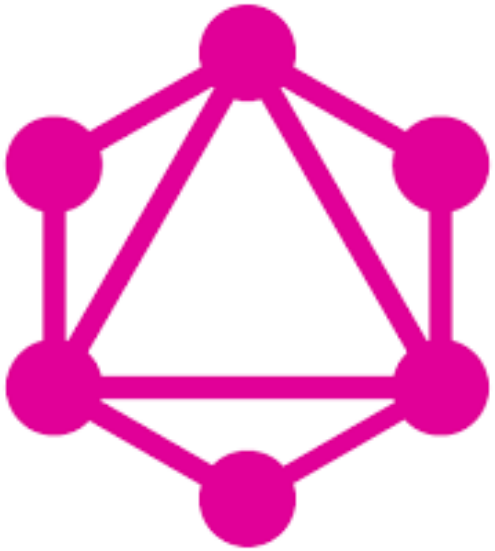


Representational State Transfer (REST)



Representational State Transfer (REST)



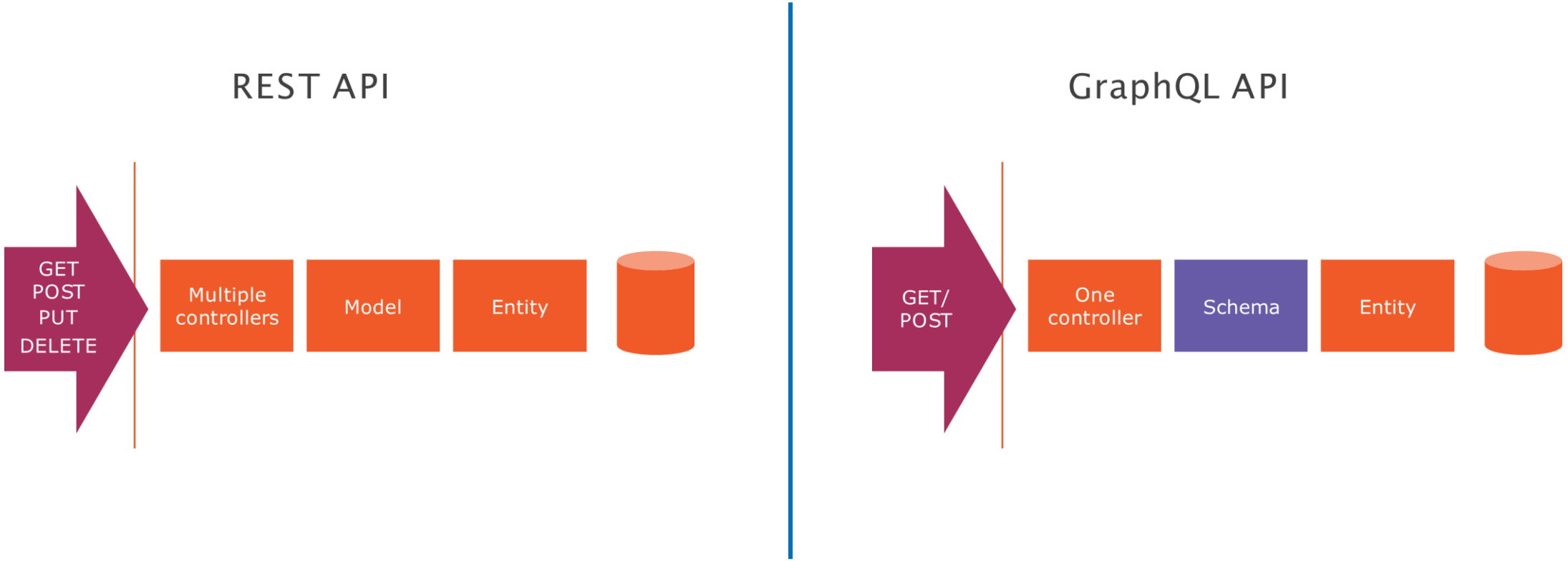


GraphQL

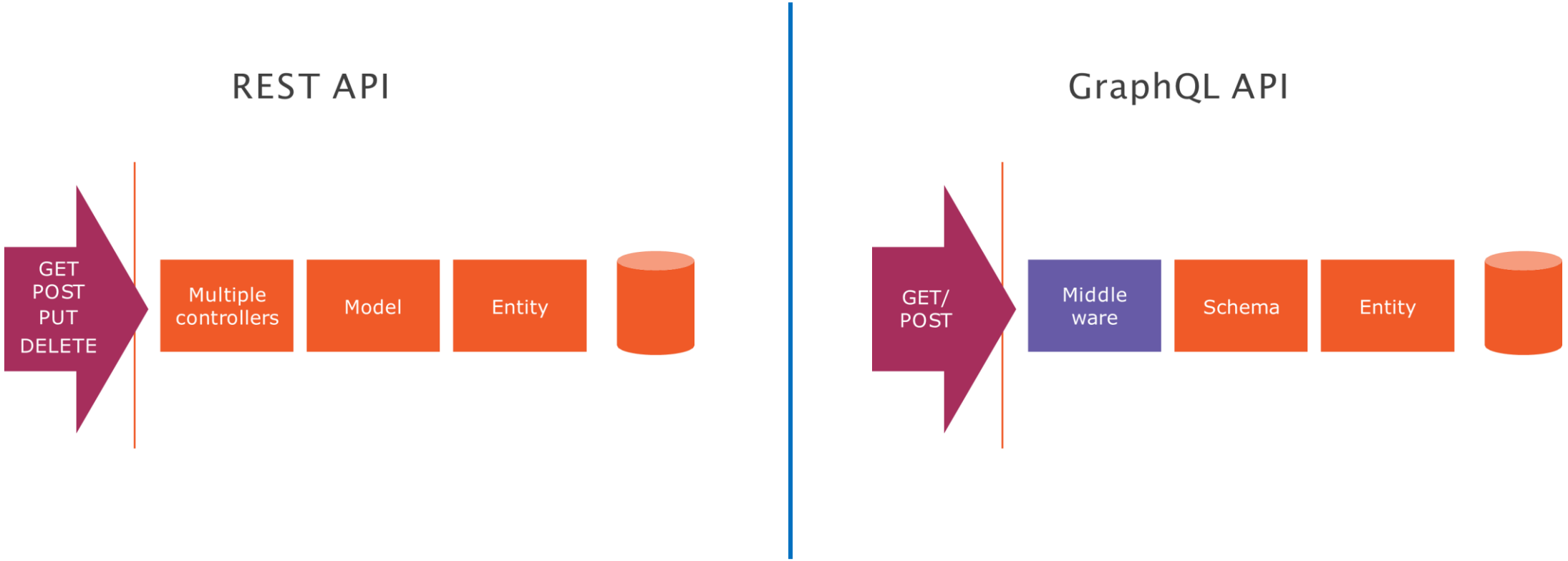
GraphQL es un lenguaje de consulta para LAS API y un tiempo de ejecución para satisfacer esas consultas con los datos existentes. GraphQL proporciona una descripción completa y comprensible de los datos de la API, ofrece a los clientes el poder de pedir exactamente lo que necesitan y nada más, facilita la evolución de las API con el tiempo y permite potentes herramientas de desarrollo.

https://graphql.org/

GraphQL


REST API
GET
POST
PUT
DELETE
Multiple controllers
Model
Entity
GraphQL API
GET/
POST
One controller
Schema
Entity

GraphQL


REST API
GET
POST
PUT
DELETE
Multiple controllers
Model
Entity
GraphQL API
GET/
POST
Middle ware
Schema
Entity

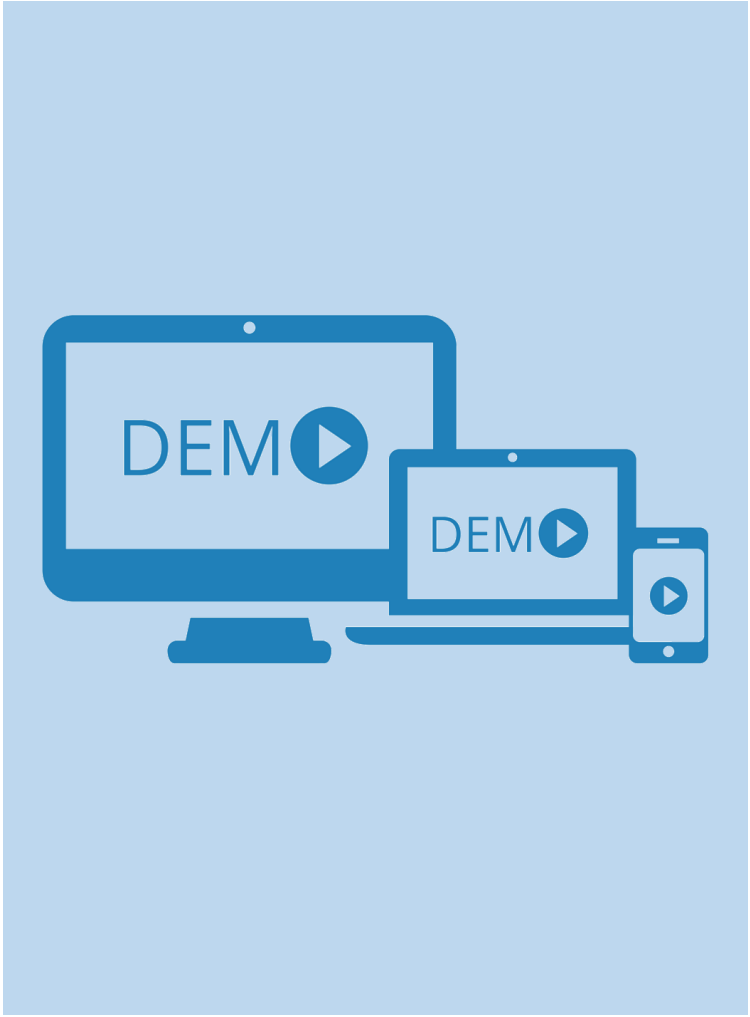
Diferencias entre GraphQL y REST

REST

- **Es solo una convención**: Es una manera de comunicarse entre el servidor y cliente, cada uno tiene sus reglas.
- **El servidor expone recursos**: Los clientes se tienen que adecuarse a como están expuestos.
- **Hace overfetching**: Envía más información de la que se necesita.
- **Múltiples request por vista**: Muy costoso en performance, básicamente es una aplicación en blanco que aún no ha cargado datos o tiene custom endpoints.
- **Documentación ajena al desarrollo**: No hay un estándar por lo que depende mucho del desarrollador para mantenerla.

GraphQL

- **Lenguaje tipado y validable**: Le damos una forma de lo que recibe y lo que devolvemos, Además de agregarle seguridad.
- **El Cliente define que recibe**: Haciendo una consulta, de la estructura que se define como respuesta.
- **Envía lo necesario**: Se tiene control total de las respuestas que se esperan del servidor.
- **Hace un solo request por vista**: Se maneja un solo row, prácticamente en solo request puedes mandar todo lo que necesitas.



Demo GraphQL

- Protocol Buffers

gRPC

Al igual que muchos sistemas RPC, gRPC se basa en la idea de definir un servicio, especificando los métodos que se pueden llamar de forma remota con sus parámetros y tipos de valor devuelto.

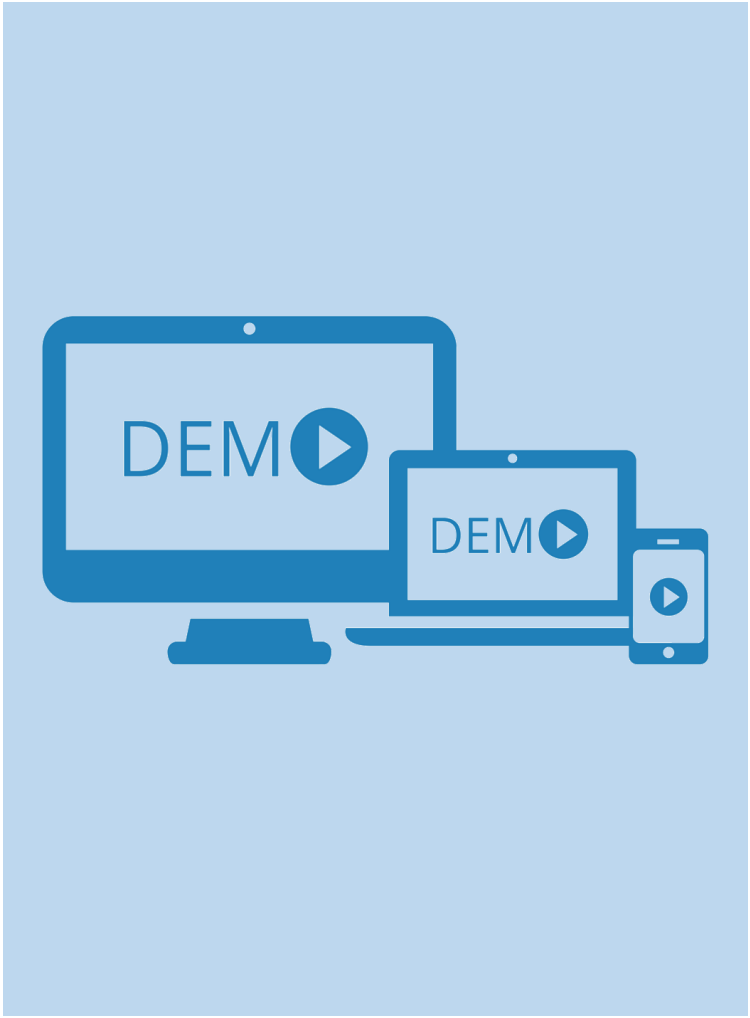
De forma predeterminada, gRPC utiliza Protocol Buffers como el lenguaje de definición de interfaz (IDL) para describir tanto la interfaz de servicio como la estructura de los mensajes de carga útil. Es posible utilizar otras alternativas si se desea.

https://grpc.io/docs/what-is-grpc/core-concepts/
https://developers.google.com/protocol-buffers

- Protocol Buffers

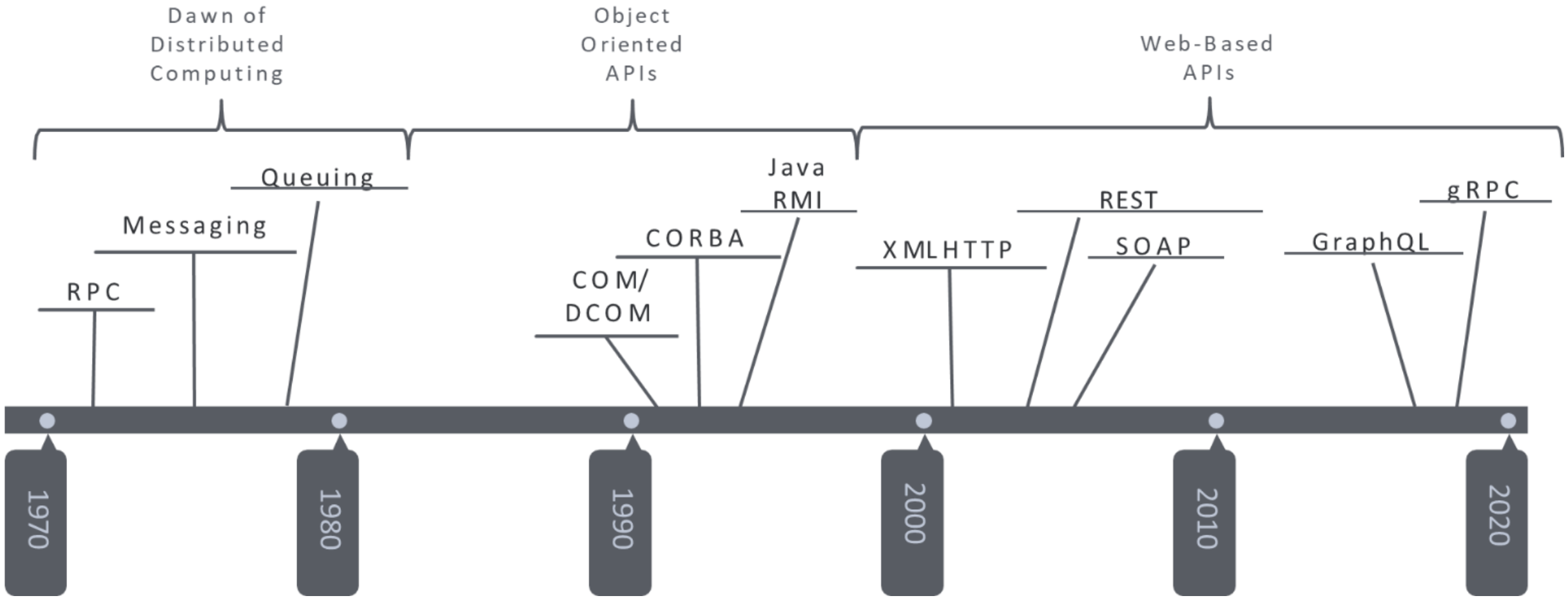
gRPC - Ventajas

- Marco de RPC moderno, ligero y de alto rendimiento.
- Desarrollo de la API de primer contrato utilizando búferes de protocolo de forma predeterminada, lo que permite realizar implementaciones independientes del idioma.
- Dispone de herramientas para muchos idioma con la finalidad de generar clientes y servidores fuertemente tipados.
- Admite llamadas de transmisión en secuencias bidireccionales, de servidor y de cliente.
- Uso reducido de red con serialización binaria Protobuf.

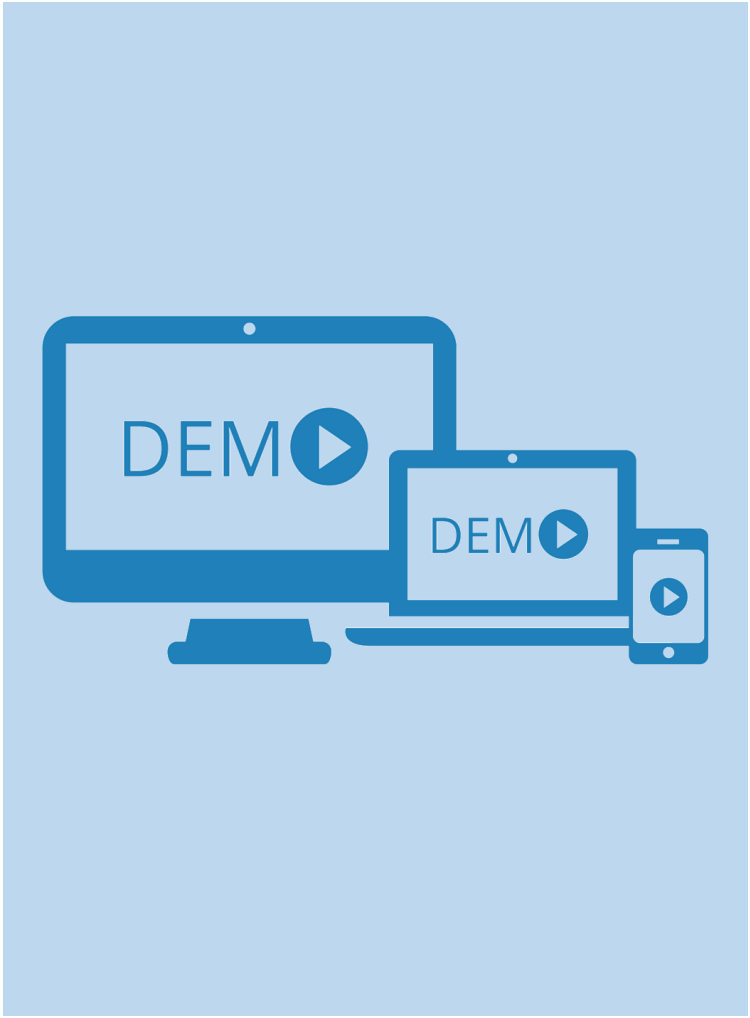


Demo gRPC

Historia



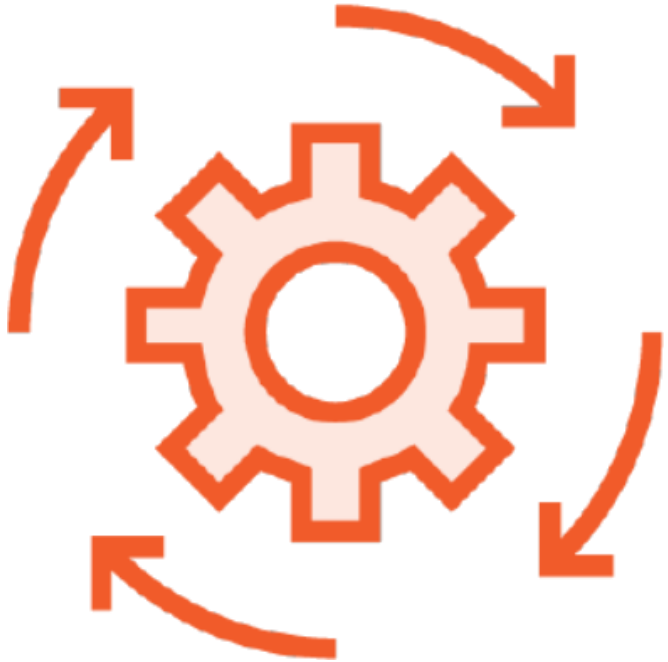
RPC, REST, GraphQL y gRPC.



Demo simplificada

¿Qué son microservicios?

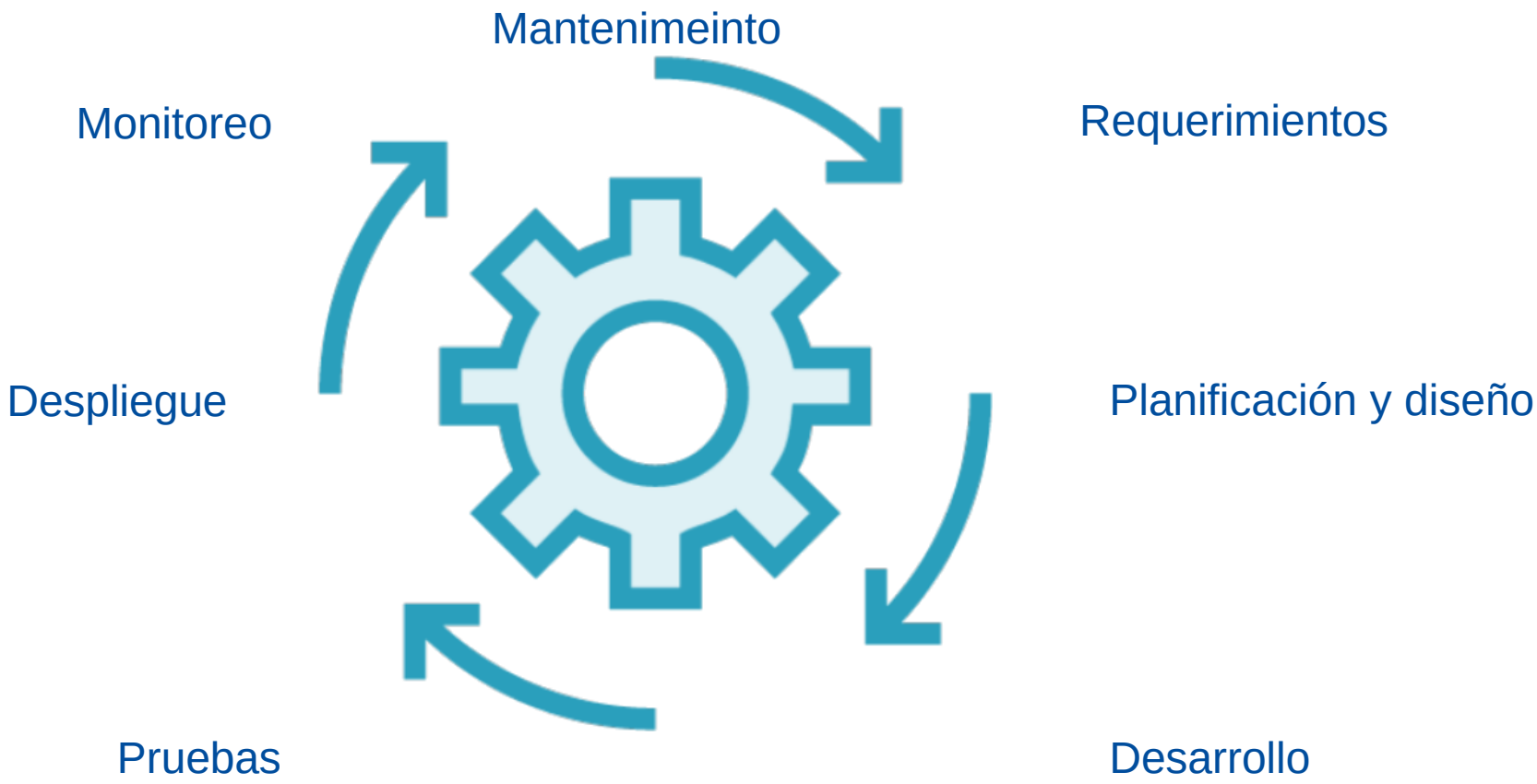
Ciclo de vida de desarrollo de software



- Ciclo de vida del desarrollo de software
- Del proyecto al producto
- Organización y gestión.
- Desplegado en producción
- Los microservicios tienen un impacto en este ciclo de vida

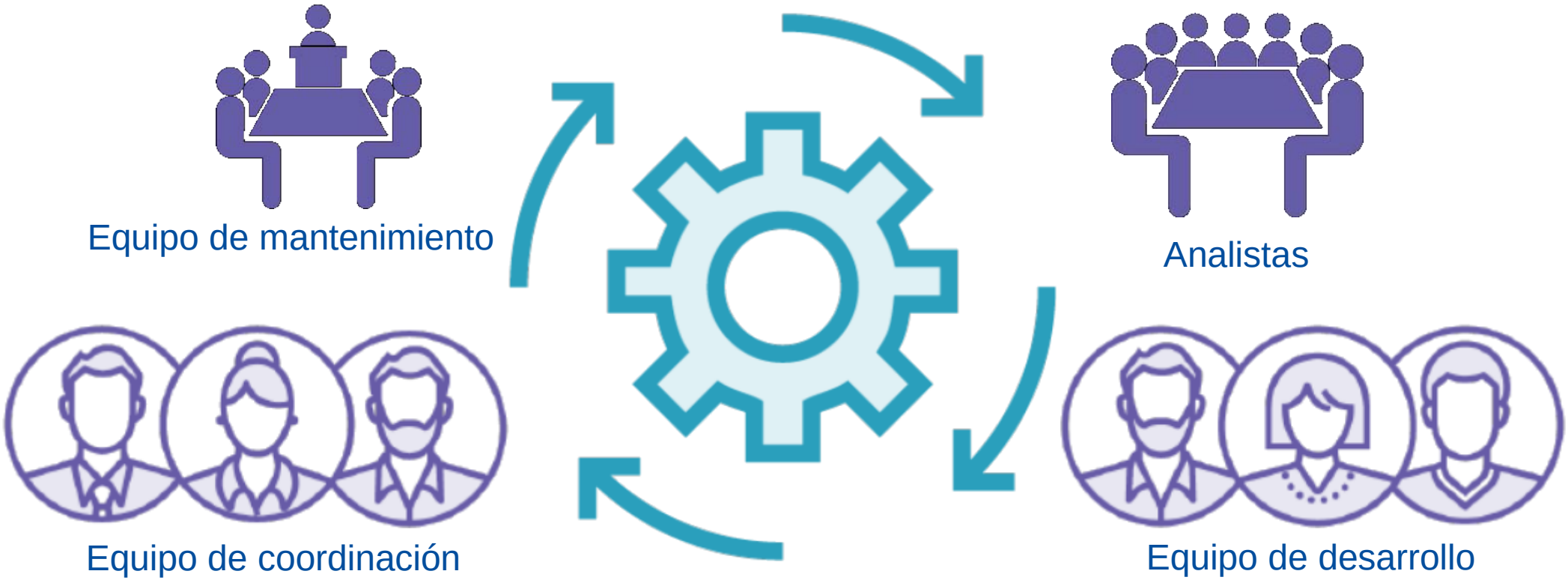
¿Qué son microservicios?

Ciclo de vida de desarrollo de software



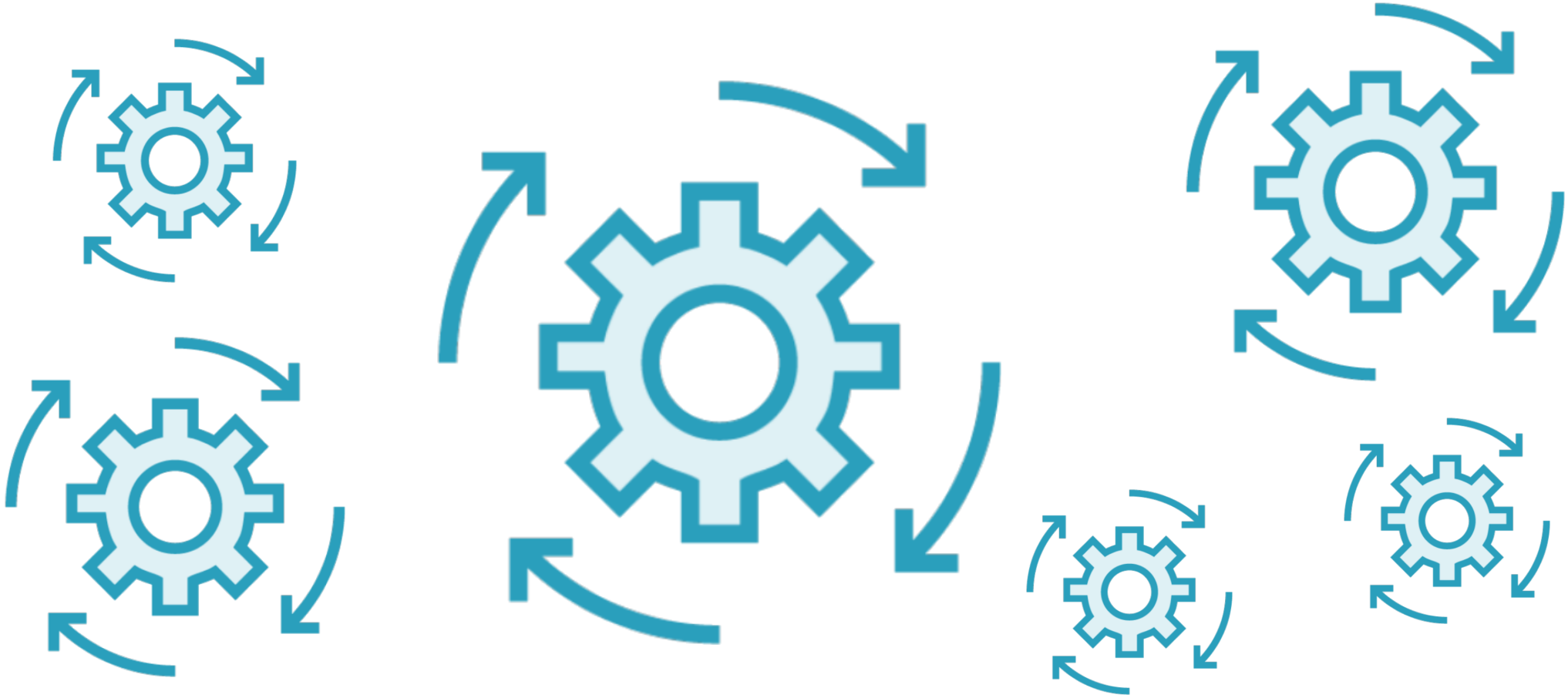
¿Qué son microservicios?

Ciclo de vida de desarrollo de software



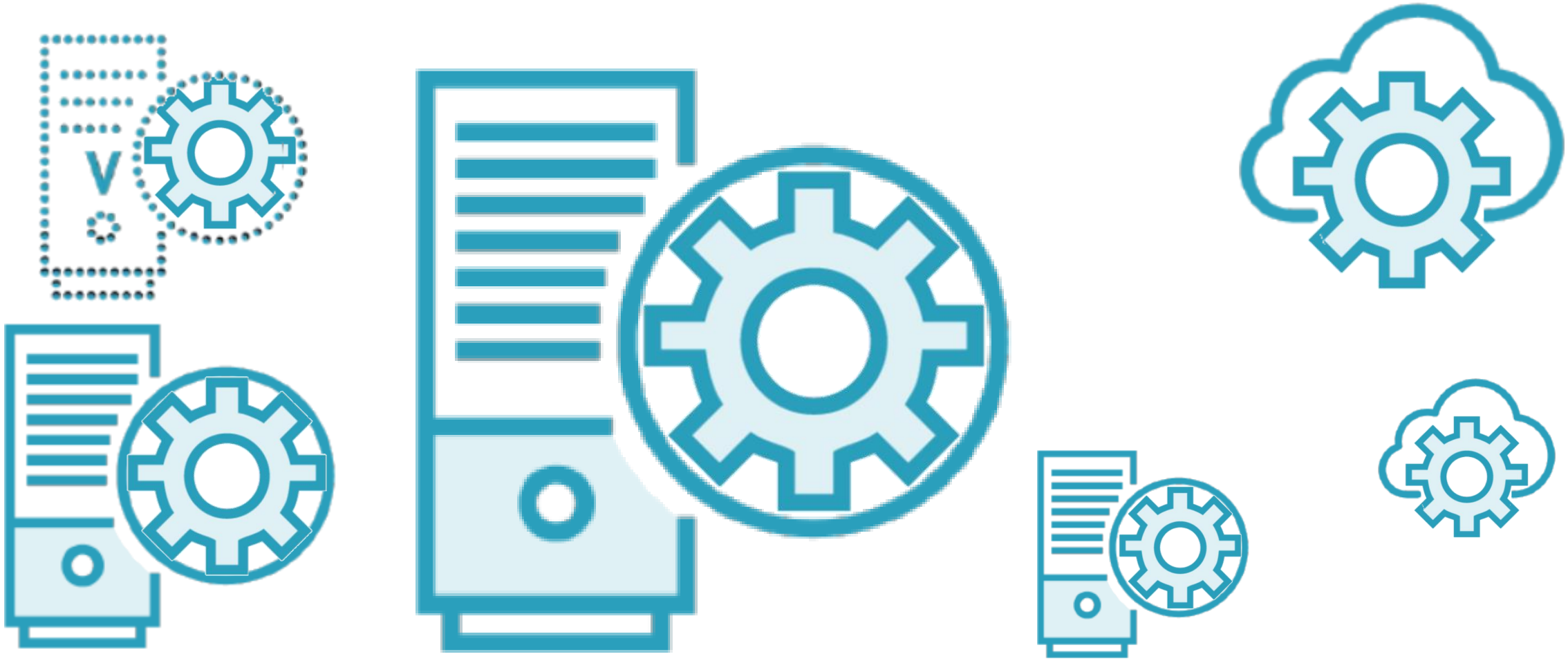
¿Qué son microservicios?

Ciclo de vida de desarrollo de software



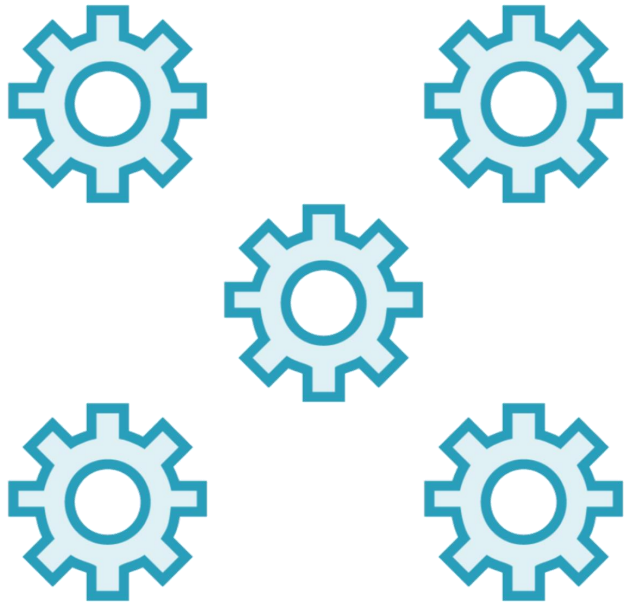
¿Qué son microservicios?

Ciclo de vida de desarrollo de software



¿Qué son microservicios?

Microservicios

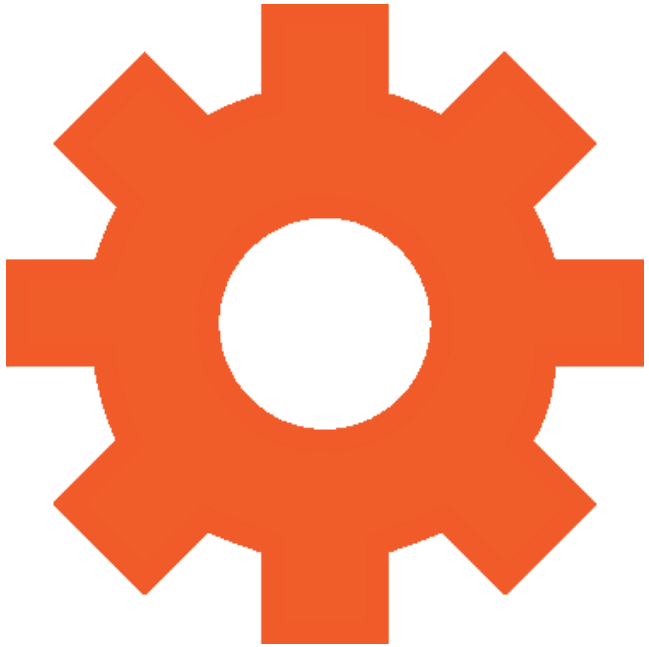


- Conjunto de practicas
- Aumentar la velocidad
- Escala
- Tecnología agnóstica
- Principios y patrones arquitectónicos.

¿Qué son microservicios?

Micro

- Grande o pequeño
- Ninguna medida universal
- "Hace una cosa"
- Alcance de funcionalidades
- Contexto limitado
- Identificar subdominios



Servicio

- Componente desplegable independientemente
- Interoperabilidad
- Comunicación basada en mensajes
- Arquitectura orientada a servicios (SOA)

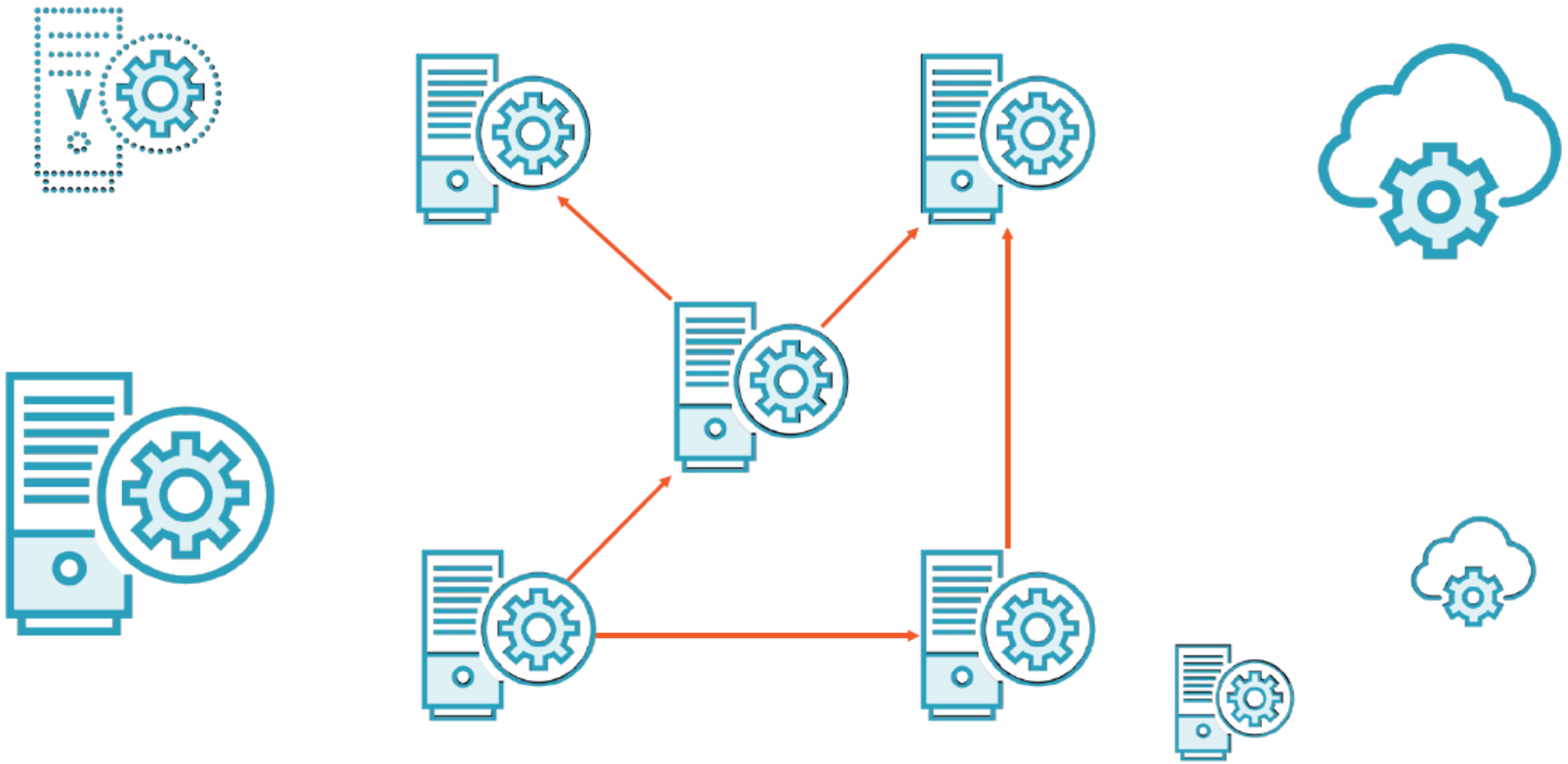
Microservicio (I)

El estilo arquitectónico de microservicios es un enfoque para desarrollar una sola aplicación como un conjunto de pequeños servicios, cada uno de los cuales se ejecuta en su propio proceso y se comunica con mecanismos ligeros.

James Lewis and Martin Fowler, Thoughtworks

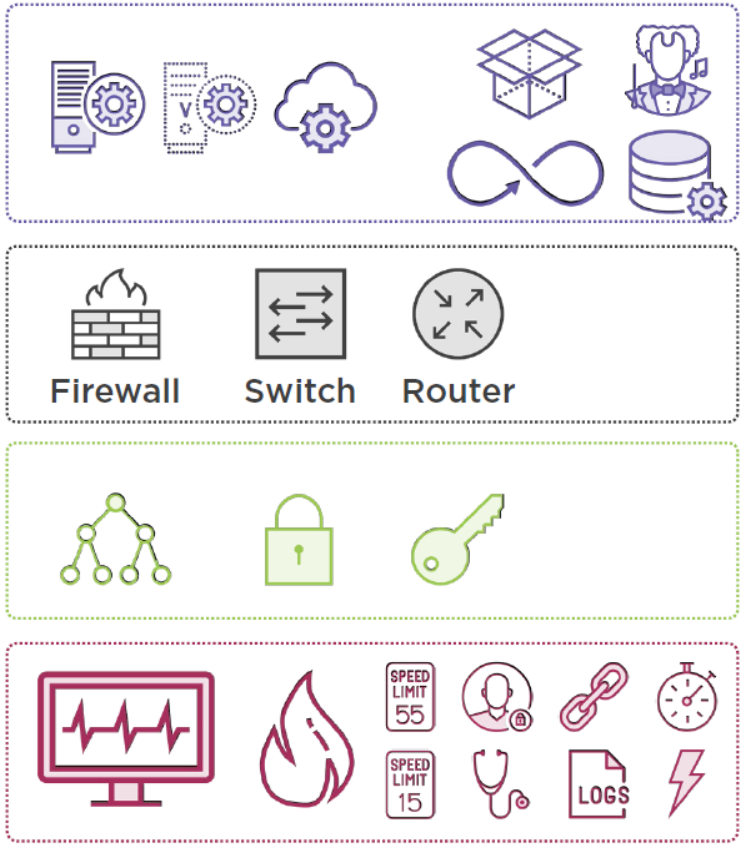
Microservicio (II)

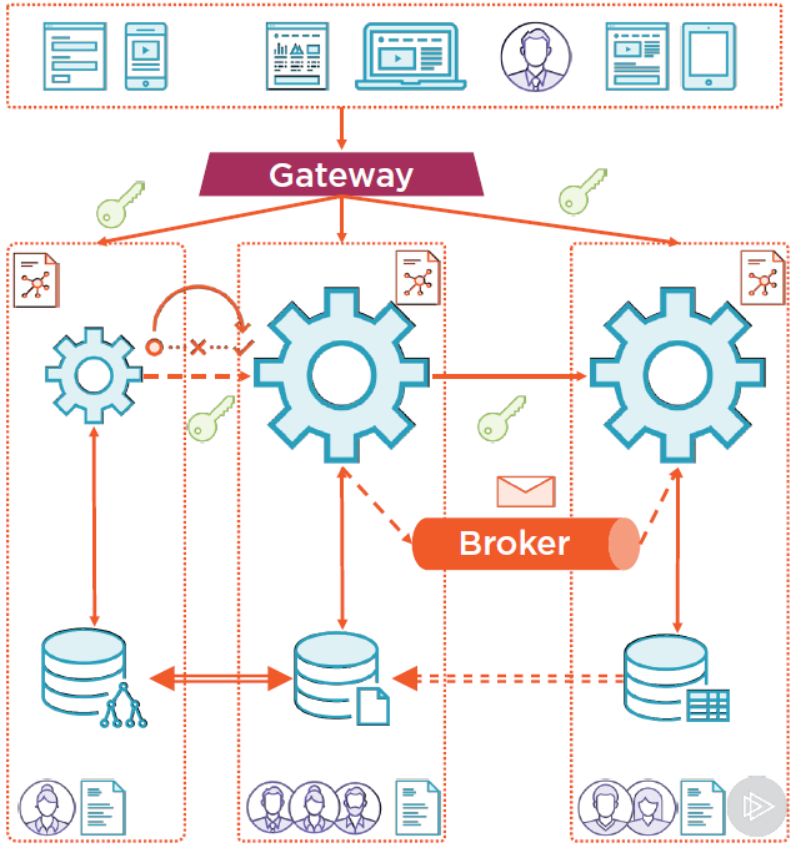
Un estilo de arquitectura de software, en el que las aplicaciones complejas se componen de pequeños procesos autónomos que se comunican entre sí mediante API independientes del lenguaje.



¿Qué son microservicios?

Elementos de un microservicio





¿Qué son microservicios?

¿Son los microservicios adecuados para mi organización?

Beneficios de los microservicios

Pequeños servicios

Puede ser propiedad de un equipo
Más fácil de entender
Puede ser reescrito

Elección de tecnología

Adopta nueva tecnología
Usa la herramienta adecuada
Estandarizar donde
tiene sentido

Despliegue individual

Menor riesgo
Minimiza el tiempo de inactividad
Actualizaciones frecuentes

Escalabilidad

Escalar servicios individualmente
Económico

Agilidad

Adaptarse rápidamente
Reutilización más fácil

¿Son los microservicios adecuados para mi organización?

Desafíos de los microservicios

Productividad del desarrollador

¿Cómo podemos facilitar que los desarrolladores sean productivos trabajando en el sistema?

Interacciones complejas

Tenga cuidado para evitando comunicaciones ineficientes e innecesarias entre microservicios

Despliegue

Necesitaras automatizar el proceso

Monitoreo

Necesitamos un lugar centralizado para verificar los registros y monitorear los problemas

¿Son los microservicios adecuados para mi organización?

GRACIAS

POR SU PREFERENCIA

GALAXY
TRAINING