

A close-up, shallow depth-of-field photograph of a computer keyboard. The central focus is on a single black key with a white 'X' character. Other keys are visible in the foreground and background, but they are blurred. The lighting is soft, creating a professional and technical atmosphere.

INTRODUCCIÓN A LOS MICROSERVICIOS

INTEGRACIÓN DE SOLUCIONES

Arquitectura de software: Definición

Arquitectura de software

La arquitectura de software define guías generales que indican la estructura, funcionamiento e interacción, entre las partes del software.

La arquitectura de un sistema de software puede basarse en uno o en varios modelos o estilos arquitectónicos bien conocidos.

Sommerville

Estilo arquitectónico

Arquitectura de software

- Un **estilo arquitectónico** define una familia de sistemas (cierto tipo de sistemas) en términos de patrones estructurales, de control, de comunicación.
- Los estilos de arquitectura no requieren el uso de tecnologías concretas, pero algunas tecnologías son adecuadas para ciertas arquitecturas.

Sommerville

Un estilo arquitectónico describe:

Arquitectura de software

Un conjunto de componentes.

Un conjunto de conectores entre componentes (comunicación, coordinación, cooperación, etc.).

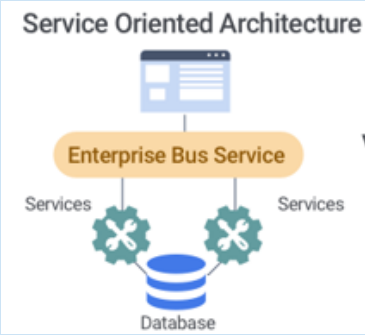
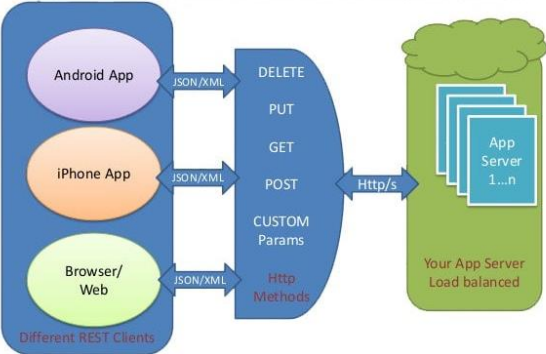
Restricciones que definen cómo se integran los componentes para formar el sistema.

Modelos que permiten comprender las propiedades de un sistema general.

Pressman

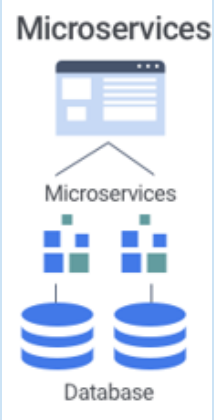
Estilos de Arquitectura

Arquitectura de software

Estilo de arquitectura	Descripción	Diagrama
Orientada al Servicio (SOA)	Se forma la estructura de una aplicación descomponiéndola en varios servicios (normalmente como servicios HTTP) que se comunican por medio de un servicio de bus de mensajes.	
Representational State Transfer (REST)	Se refiere a una interfaz, que consiste en una red de recursos Web relacionada por links y operaciones como GET, DELETE (transiciones de estado).	

Estilos de Arquitectura

Arquitectura de software

Estilo de arquitectura	Descripción	Diagrama
Microservicios	<p>Consta de una colección de servicios autónomos y pequeños. Los servicios son independientes entre sí y cada uno debe implementar una funcionalidad de negocio individual.</p> <p>Los servicios están acoplados de forma flexible y se comunican a través de contratos de API.</p>	 <p>The diagram, titled 'Microservices', illustrates a distributed architecture. At the top is a browser window icon representing a user interface. Below it, two lines connect to two separate server icons, each labeled 'Microservices'. At the bottom, two database icons are labeled 'Database', with lines indicating their connection to the microservices above.</p>

Arquitectura Orientada a Servicios (SOA)

Arquitectura de software

SOA es un modelo de componentes que interrelaciona las diferentes unidades funcionales de las aplicaciones, denominadas servicios, a través de interfaces y contratos bien definidos entre esos servicios. La interfaz se define de forma neutral, y debería ser independiente de la plataforma hardware, del sistema operativo y del lenguaje de programación utilizado. Esto permite a los servicios, construidos sobre sistemas heterogéneos, interactuar entre ellos de una manera uniforme y universal.

IBM

Servicio Web

Un servicio web es un sistema software diseñado para soportar la interacción máquina-a-máquina, a través de una red, de forma interoperable. Cuenta con una interfaz descrita en un formato procesable por un equipo informático (específicamente en WSDL), a través de la que es posible interactuar con él mismo, mediante el intercambio de mensajes SOAP, típicamente transmitidos usando serialización XML sobre HTTP conjuntamente con otros estándares web.

W3C Working Group

Arquitectura de microservicios

Arquitectura de software

- Los **microservicios** son una evolución de los conceptos SOA.
- Aumentan la agilidad y velocidad para hacer cambios, y la habilidad de hacerlos con un costo total e impacto menores en la infraestructura existente.
- Aunque los microservicios se derivan de SOA, no es lo mismo que la arquitectura de microservicios.

“La arquitectura de microservicios es SOA bien hecho”.

Microsoft.

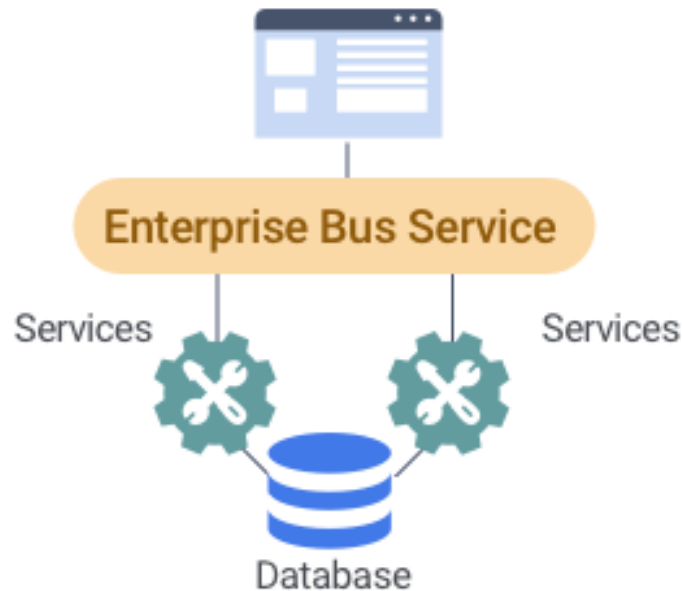
Arquitectura SOA Vs Microservicios

Arquitectura de software

SOA vs Microservices



Service Oriented Architecture



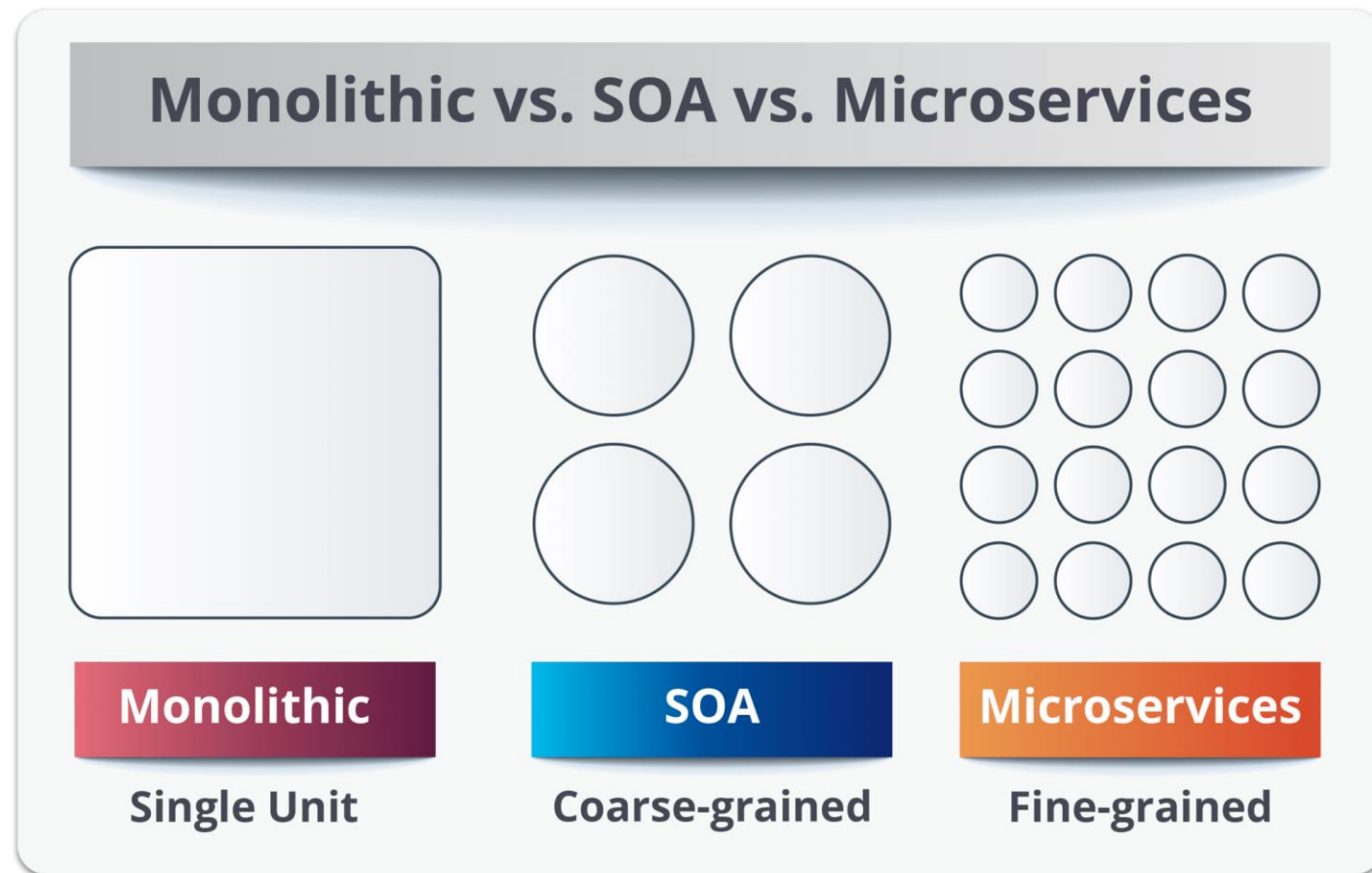
Vs

Microservices



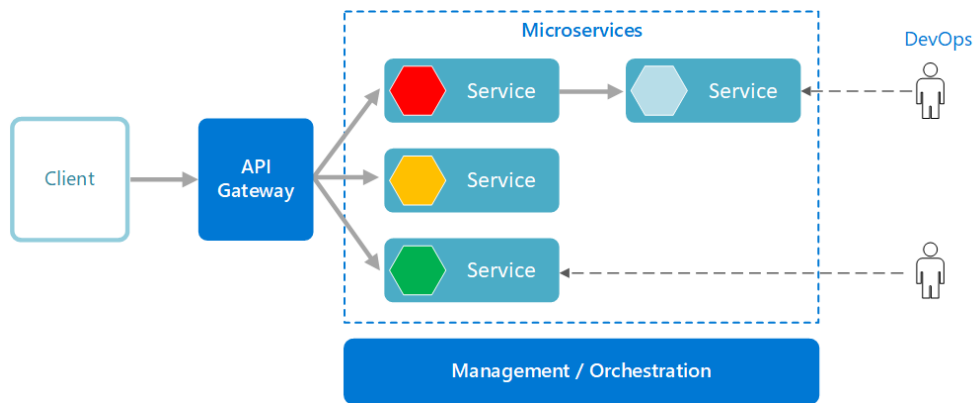
Monolítica Vs SOA Vs Microservicios

Arquitectura de software



Arquitectura de microservicios

Arquitectura de microservicios



- Utilizada para crear aplicaciones usando un conjunto de pequeños servicios, los cuales se comunican entre sí, pero se ejecutan de forma individual.
- Dr. Peter Rodgers (investigador en HP) en 2005 usó el término “Micro-Web-Services”.
- En 2011 se usó por primera vez el concepto “microservicios”.

Características de los microservicios

Arquitectura de microservicios

Son pequeños e independientes.

Cada servicio puede administrarse por un equipo de desarrollo pequeño.

Los servicios pueden actualizarse sin tener que volver a implementar toda la aplicación.

Los servicios son los responsables de conservar sus propios datos o estado externo.

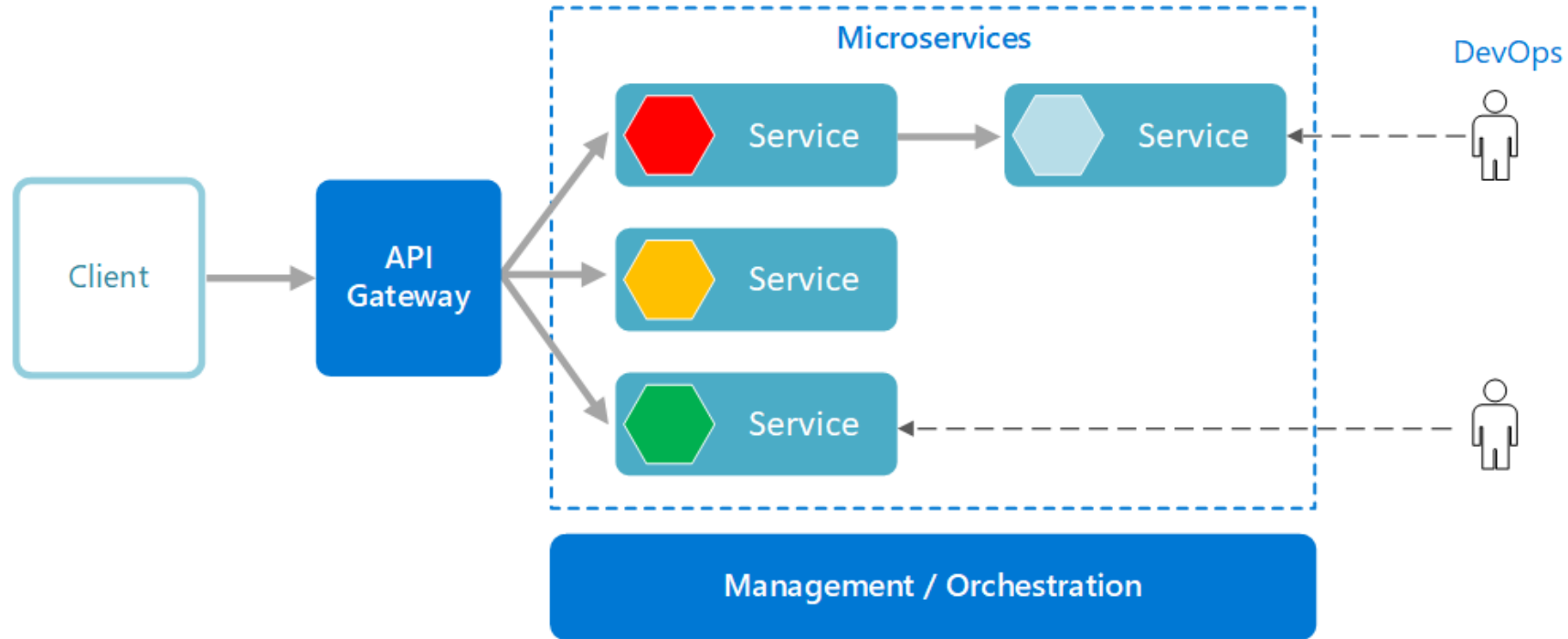
Los servicios se comunican mediante API bien definidas.

No es necesario que los servicios compartan la misma pila de tecnología, las bibliotecas o los marcos de trabajo.

Elementos de la arquitectura de microservicios

Arquitectura de microservicios

- Puerta de enlace de API
- Microservicio
- Consumidores o clientes
- Administración u orquestación



Puerta de enlace de API

Elementos de la arquitectura de microservicios

- Es el punto de entrada para los clientes. En lugar de llamar a los servicios directamente.
- La mayor ventaja de usar una puerta de enlace de API es que desacopla los clientes de los servicios.
- Los servicios pueden cambiar de versión o refactorizarse sin necesidad de actualizar todos los clientes.



La puerta de enlace de API es una de las principales características de los microservicios.

Microsoft.

Microservicio

Elementos de la arquitectura de microservicios

Es una función bien definida, pequeña, autocontenida e independiente del contexto o estado de otros servicios cuyas características son la reutilización, estandarización, abstracción y autonomía.

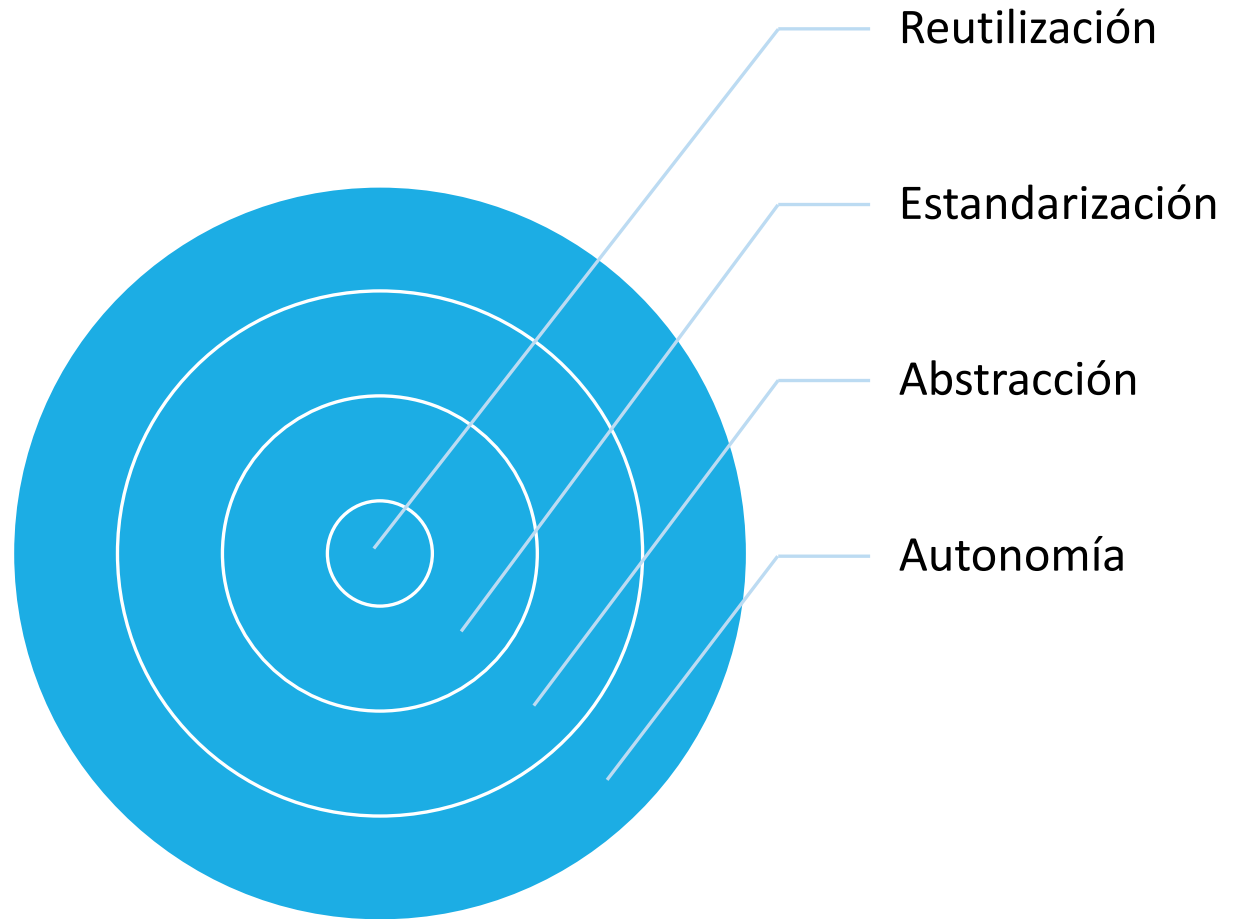
Microsoft.

- Cada servicio es independiente y debe implementar una funcionalidad de negocio individual dentro de un contexto delimitado.
- Un contexto delimitado es una división natural de una empresa y proporciona un límite explícito dentro del cual existe un modelo de dominio.
- Se basan en un modelo de recursos donde un servicio es publicado como un recurso.
- No requieren establecer un contrato previo como en SOA.
- Solo se requiere conocer la dirección URI del recurso.



Arquitectura Orientada a Servicios (SOA)

- **Servicio:** Es una función bien definida, auto-contenida e independiente del contexto o estado de otros servicios.



Consumidores o clientes

Elementos de la arquitectura de microservicios

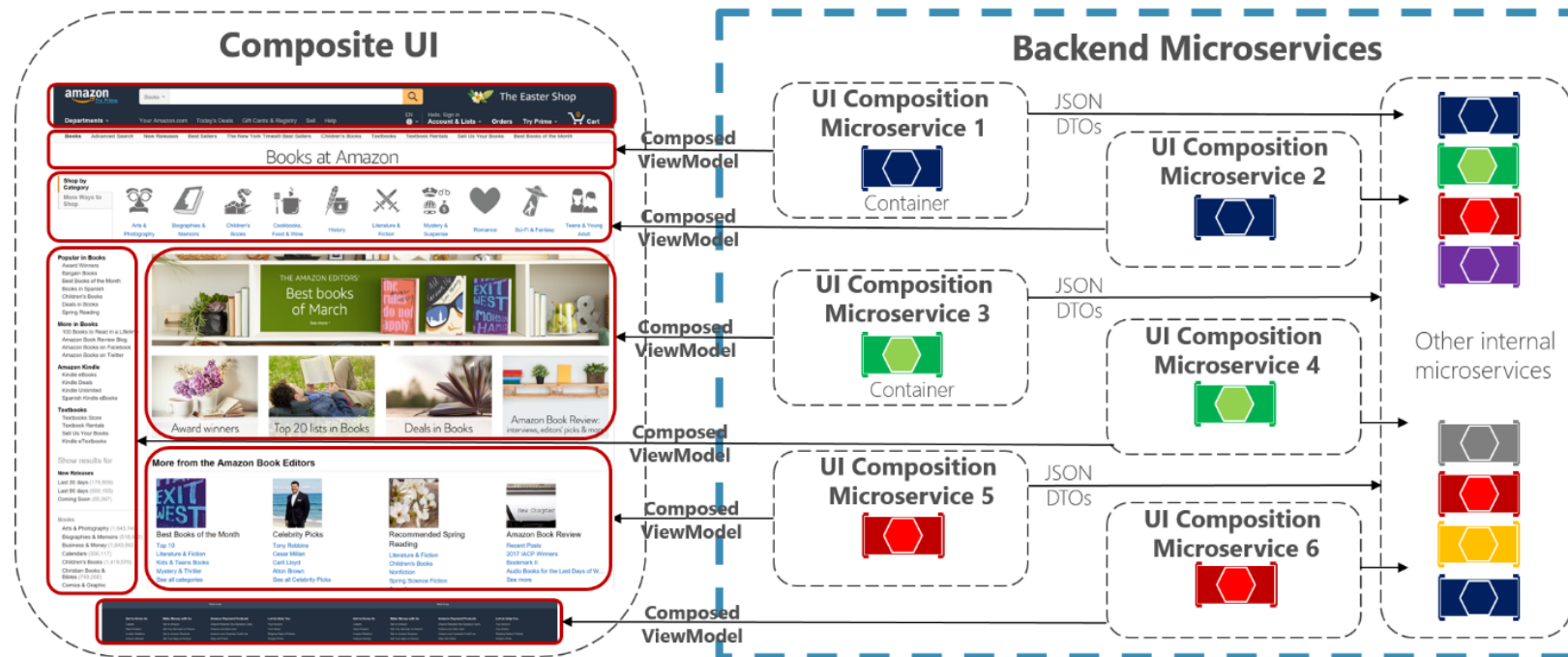
- Puede ser una aplicación móvil, sistema, módulo de software u otro servicio
- Demanda la funcionalidad específica que el servicio proporciona.
- Se ejecuta en una interfaz definida llamando a los microservicios como recursos.



Interfaces diseñadas para microservicios

Implementación de una arquitectura de microservicios

Composite UI *generated* by microservices



Administración u orquestación

Elementos de la arquitectura de microservicios

- Responsable de la colocación de servicios en los nodos, la identificación de errores, el reequilibrio de servicios entre nodos de manera automatizada.
- Normalmente, este componente es una tecnología estándar automatizada, como Kubernetes, Docker Swarm, Azure Service Fabric o DC/OS, aunque puede ser realizado de forma manual o personalizada.



Microsoft Azure Service Fabric

Ventajas de los microservicios

Agilidad.

Equipos pequeños y centrados.

Base de código pequeña.

Mezcla de tecnologías.

Aislamiento de errores.

Escalabilidad.

Aislamiento de los datos.

Desventajas de los microservicios

Complejidad.

Desarrollo y pruebas.

Falta de gobernanza.

Congestión y latencia de red.

Dificultad para la integridad de datos.

Administración compleja.

Control de versiones delicada.

Evaluar conjunto de habilidades del equipo.

Proceso para el desarrollo de una arquitectura de microservicios

DISEÑO E IMPLEMENTACIÓN DE UNA ARQUITECTURA DE MICROSERVICIOS

Ciclo de vida de los microservicios

Proceso para el desarrollo de una arquitectura de microservicios



Ciclo de vida de los microservicios desde el punto de vista del proveedor



Ciclo de vida de los microservicios desde el punto de vista del consumidor

Ejemplos de estrategias para el desarrollo de microservicios

Proceso para el desarrollo de una arquitectura de microservicios

- SDDM: Service Oriented Design and Development Methodology (Papazoglou & Janvan den Heuvel, 2006)
- SOMF: Service Oriented Modeling Framework (Bell, 2008)
- SOMA: Service Oriented Modeling and Architecture (Arsanjani, 2004)
- DDD: Domain Driven Design-oriented microservice (Evans, DDD & Microservices: At last, some boundaries!, 2016)

DDD: Domain Driven Design-oriented microservice

Proceso para el desarrollo de una arquitectura de microservicios

El diseño basado por dominios es un enfoque para el desarrollo de software que centra el desarrollo en la programación de un modelo de dominio que tiene una amplia comprensión de los procesos y las reglas de un dominio. En el contexto de la creación de aplicaciones, DDD habla de problemas como dominios. Describe áreas de problemas independientes como contextos limitados (cada contexto limitado se correlaciona con un microservicio) y enfatiza un lenguaje común para hablar sobre estos problemas.

Eric Evans, DDD & Microservices (2016)

DDD: Domain Driven Design-oriented microservice

Proceso para el desarrollo de una arquitectura de microservicios

- Propuesto por Eric Evans (Evans, Domain-Driven Design: Tackling Complexity in the Heart of Software, 2003)
- Desde entonces, una comunidad de practicantes ha desarrollado aún más las ideas, generando otros libros y cursos de capacitación.
- El enfoque es particularmente adecuado para microservicios, es decir, dominios complejos, donde es necesario organizar una gran cantidad de lógica desordenada.
- Eric Evans publicó una actualización especialmente dirigida a microservicios (Evans, DDD and Microservices: At Last, Some Boundaries!, 2016)

Proceso para crear una arquitectura de microservicios

DDD: Domain Driven Design-oriented microservice

- 1. Análisis de dominios.** En esta etapa se definen los límites de los servicios individuales alrededor de las funcionalidades de la empresa.
- 2. Diseño de microservicios.** En esta etapa se construye la arquitectura considerando la comunicación, la tecnología, la API de entrada y se elige el patrón de diseño.
- 3. Implementación.** Esta etapa es la encargada de la distribución con operaciones sólidas para su implementación y supervisión.

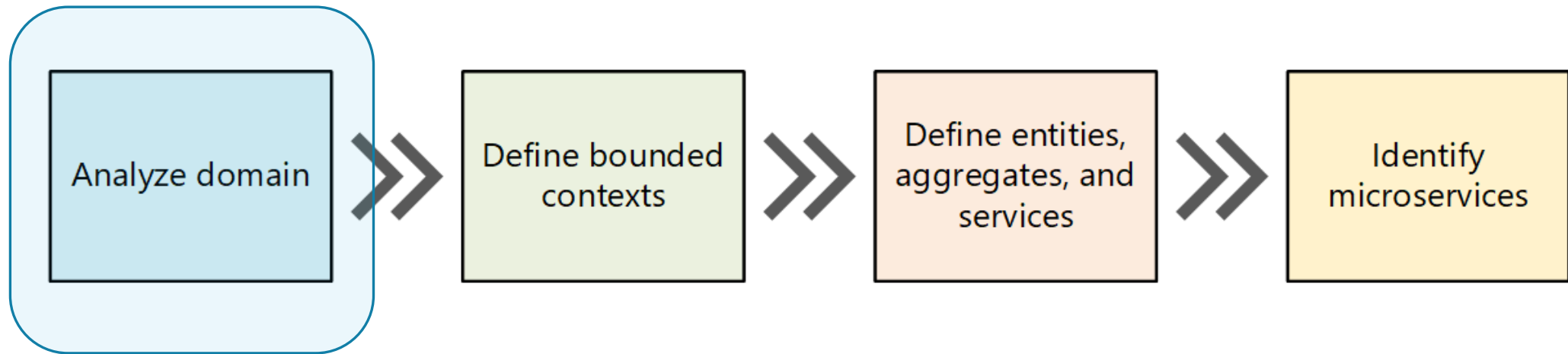
Análisis de dominios de una arquitectura de microservicios

Proceso para el desarrollo de una arquitectura de microservicios

DISEÑO E IMPLEMENTACIÓN DE UNA ARQUITECTURA DE MICROSERVICIOS

Fases del análisis de dominios

Análisis de dominios de una arquitectura de microservicios



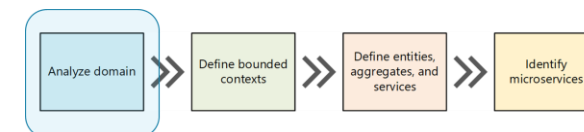
Eric Evans

Escenario

Análisis de dominios de una arquitectura de microservicios

En la biblioteca de la Facultad de Estadística e Informática de la Universidad Veracruzana, se requiere realizar la captura de un listado de varios libros nuevos junto al nombre de su autor que se van a agregar a su colección. Al mismo tiempo se requiere registrar los préstamos de los libros a los estudiantes.

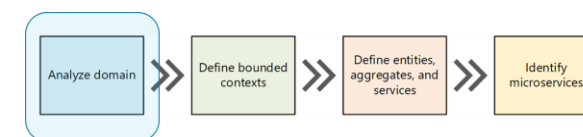
La aplicación necesita operar en la nube, con un objetivo de nivel de servicio alto, pues varias aplicaciones clientes en distintas plataformas, tanto de dispositivos móviles como de escritorio, desean conectarse y consultar ese listado en Internet. Además, se desea publicarlo en tiempo real en la página web de la Facultad. Esta consideración hace que la biblioteca se decante por una arquitectura de microservicios.



Análisis de dominio

Análisis de dominios de una arquitectura de microservicios

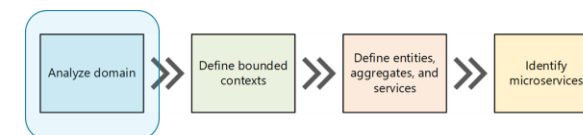
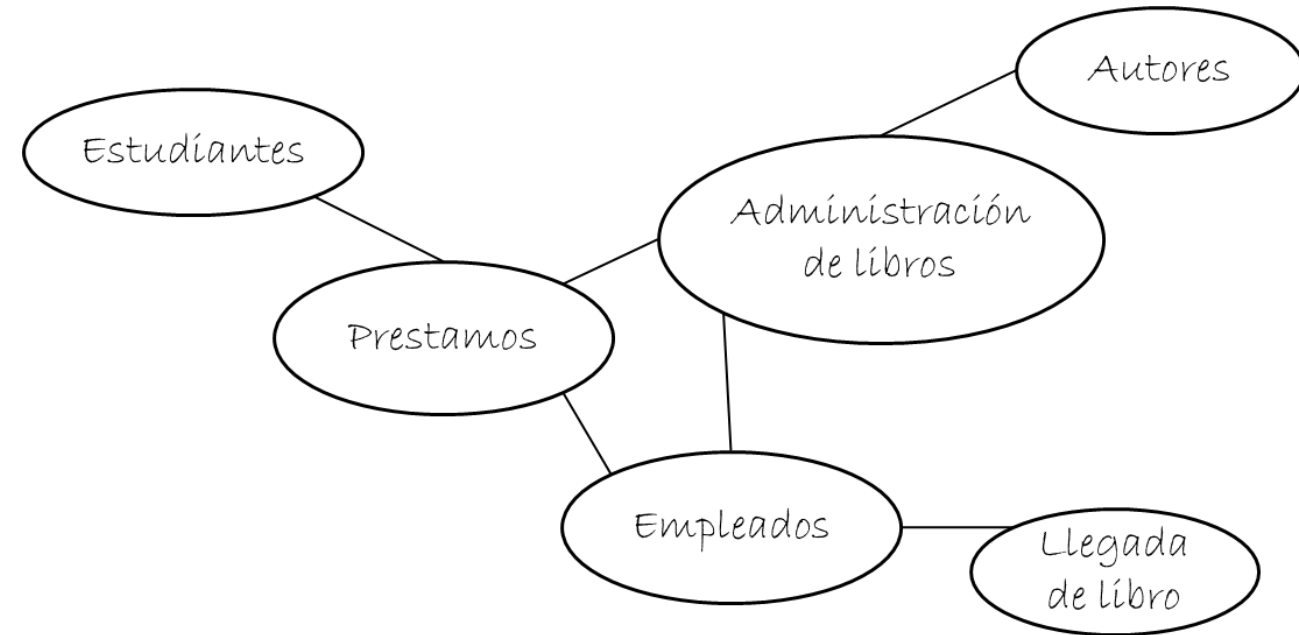
- Un libro llega a la biblioteca y debe ser registrado.
- Las características del libro que deben ser identificadas son el título, autor, año, precio y género.
- Varios libros pueden ser prestados a un estudiante con fecha de entrega. Una vez entregados los libros, el estudiante puede solicitar otro libro a préstamo.
- Se desea que la lista de libros pueda ser consultada en Internet por una variedad de consumidores como aplicaciones móviles, páginas web y sistemas de escritorio.
- Al igual que su consulta, la lista debe poder actualizarse desde Internet.



Análisis de dominio

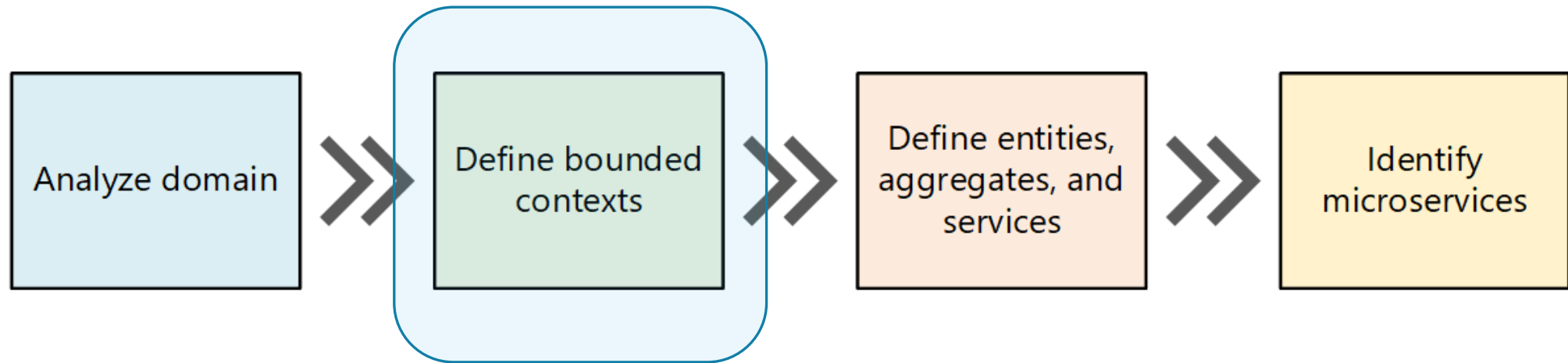
Análisis de dominios de una arquitectura de microservicios

- Se necesita una visión general del sistema que se va a crear.
- Se empieza por modelar el dominio empresarial y se crea un modelo de dominio
- El modelo de dominio es un modelo abstracto del ámbito empresarial



Fases del análisis de dominios

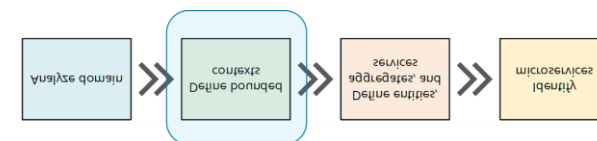
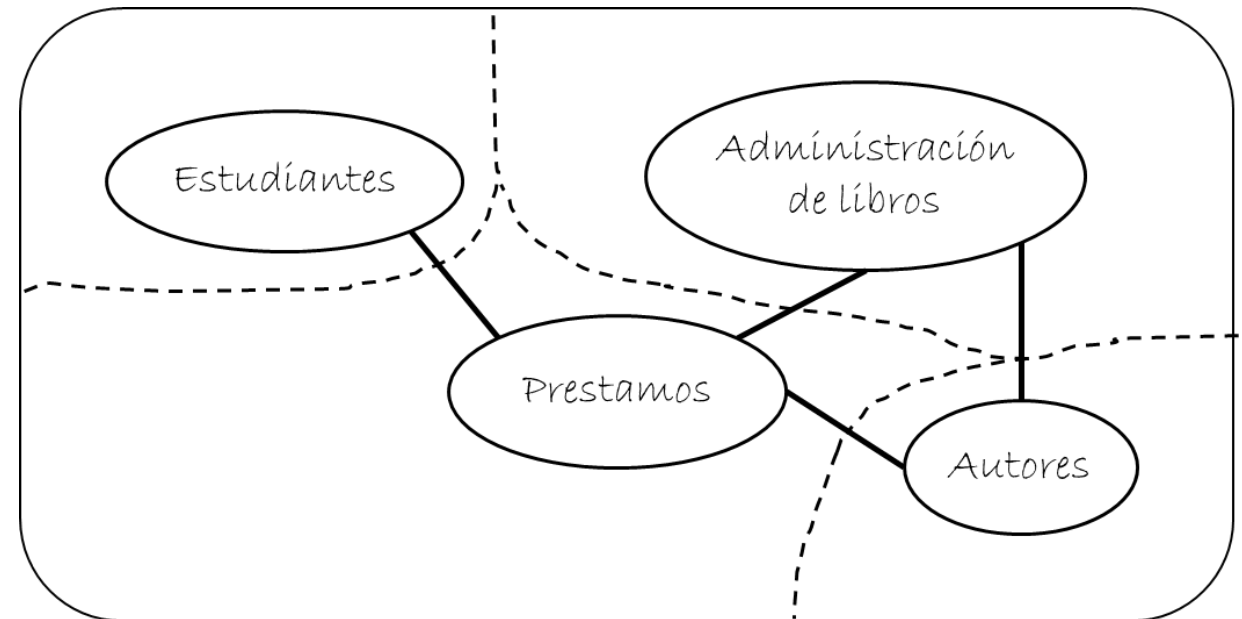
Análisis de dominios de una arquitectura de microservicios



Definición de contextos delimitados

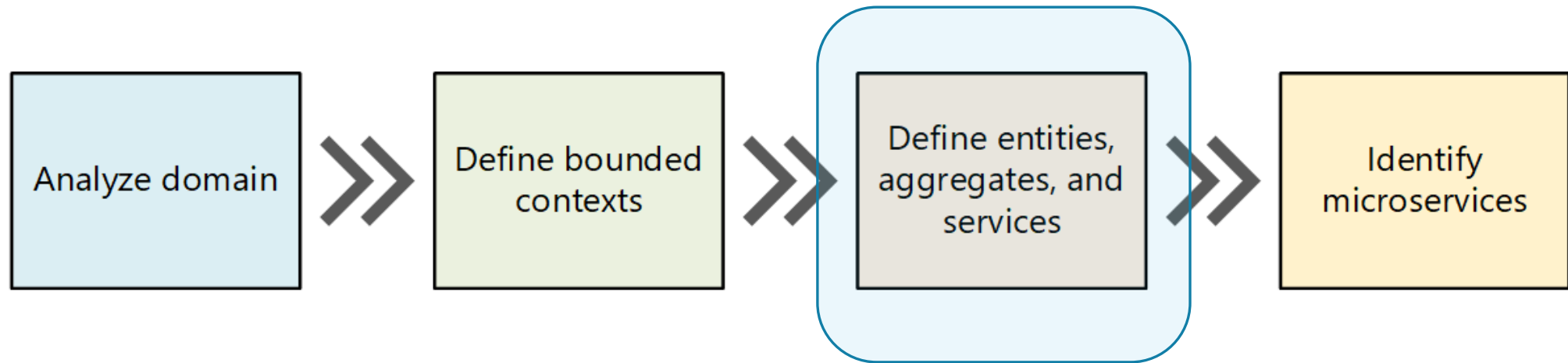
Análisis de dominios de una arquitectura de microservicios

- El modelo de dominio incluye representaciones de cosas reales del mundo, pero eso no significa que todas las partes del sistema deban representarse.
- Remover lo que no se va a representar.
- Se deben definir claramente límites entre los dominios.



Fases del análisis de dominios

Análisis de dominios de una arquitectura de microservicios

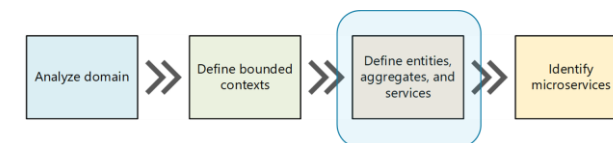


Definir entidades, agregados y servicios

Análisis de dominios de una arquitectura de microservicios

- Esta etapa también llamada, *diseño basado en dominios táctico*, consiste en definir los modelos de dominio con más precisión.

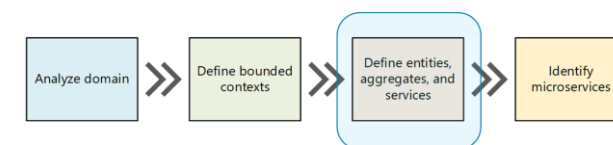
Como principio general, un microservicio no debe ser menor que un agregado ni mayor que un contexto delimitado.



Definir entidades, agregados y servicios

Análisis de dominios de una arquitectura de microservicios

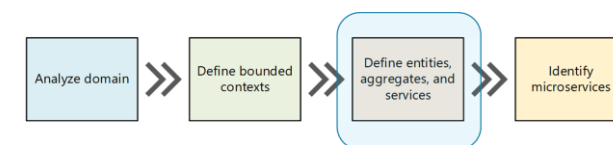
- ENTIDADES
- Las entidades representan objetos del dominio y se definen principalmente por su identidad, continuidad y persistencia en el tiempo y no solo por los atributos que las componen.
- Son la base para un modelo.
- Por ejemplo, en una aplicación bancaria, las cuentas y los clientes serían entidades



Definir entidades, agregados y servicios

Análisis de dominios de una arquitectura de microservicios

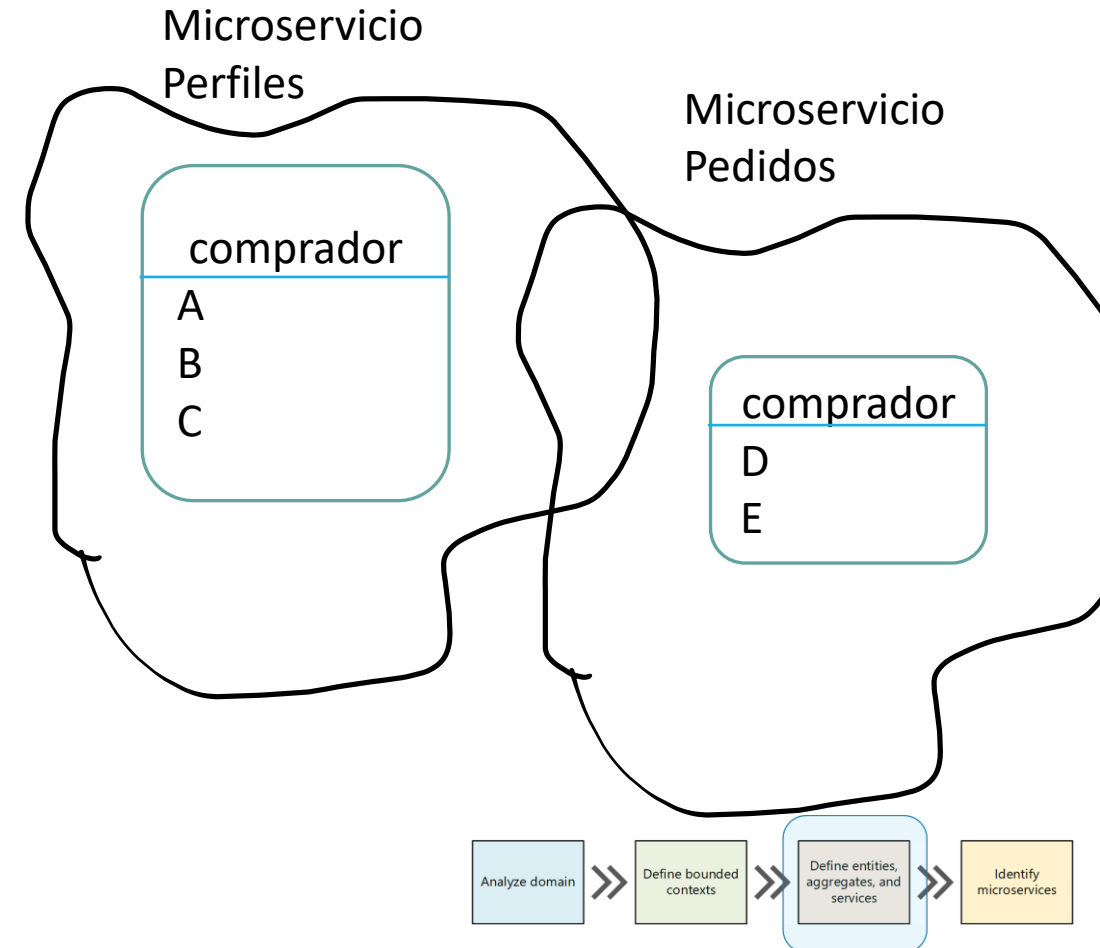
- ENTIDADES
- La misma identidad se puede modelar en varios contextos delimitados o microservicios.
- Las entidades de cada contexto delimitado limitan sus atributos y comportamientos a los requeridos en el dominio de ese contexto delimitado.



Definir entidades, agregados y servicios

Análisis de dominios de una arquitectura de microservicios

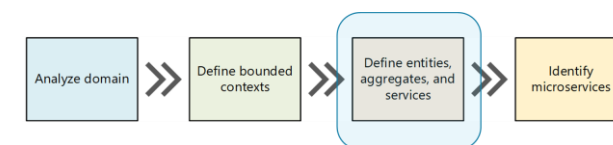
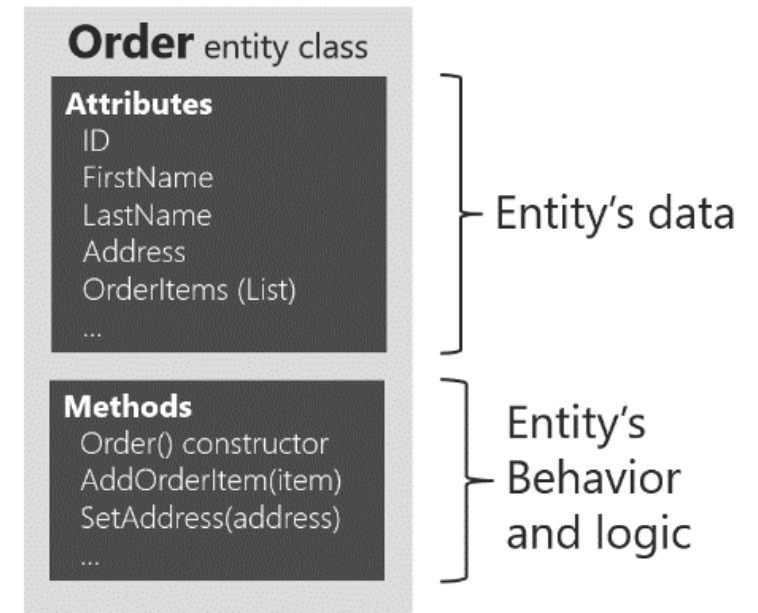
- ENTIDADES
- Por ejemplo, es posible que la **entidad de comprador** tenga la mayoría de los atributos de una persona que estén definidos en la entidad de usuario en el **microservicio de perfiles** o identidades.
- Pero la **entidad de comprador** en el **microservicio de pedidos** podría tener menos atributos, porque solo determinados datos del comprador están relacionados con el proceso de pedido.
- El contexto de cada microservicio o contexto delimitado afecta a su modelo de dominio.



Definir entidades, agregados y servicios

Análisis de dominios de una arquitectura de microservicios

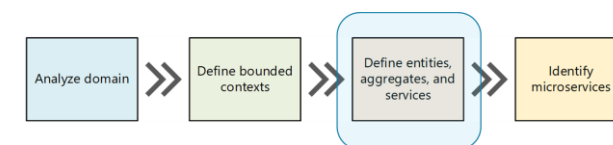
- ENTIDADES
- Una entidad de dominio en DDD debe implementar la lógica del dominio o el comportamiento relacionado con los datos de entidad.
- Por ejemplo, la entidad **pedido** debería implementar la lógica de negocios y las operaciones como métodos, para tareas como agregar un elemento de pedido, la validación de datos y el cálculo total.
- Los métodos de la entidad se encargan de los elementos invariables y las reglas de la entidad en lugar de tener esas reglas distribuidas por el nivel de aplicación.



Definir entidades, agregados y servicios

Análisis de dominios de una arquitectura de microservicios

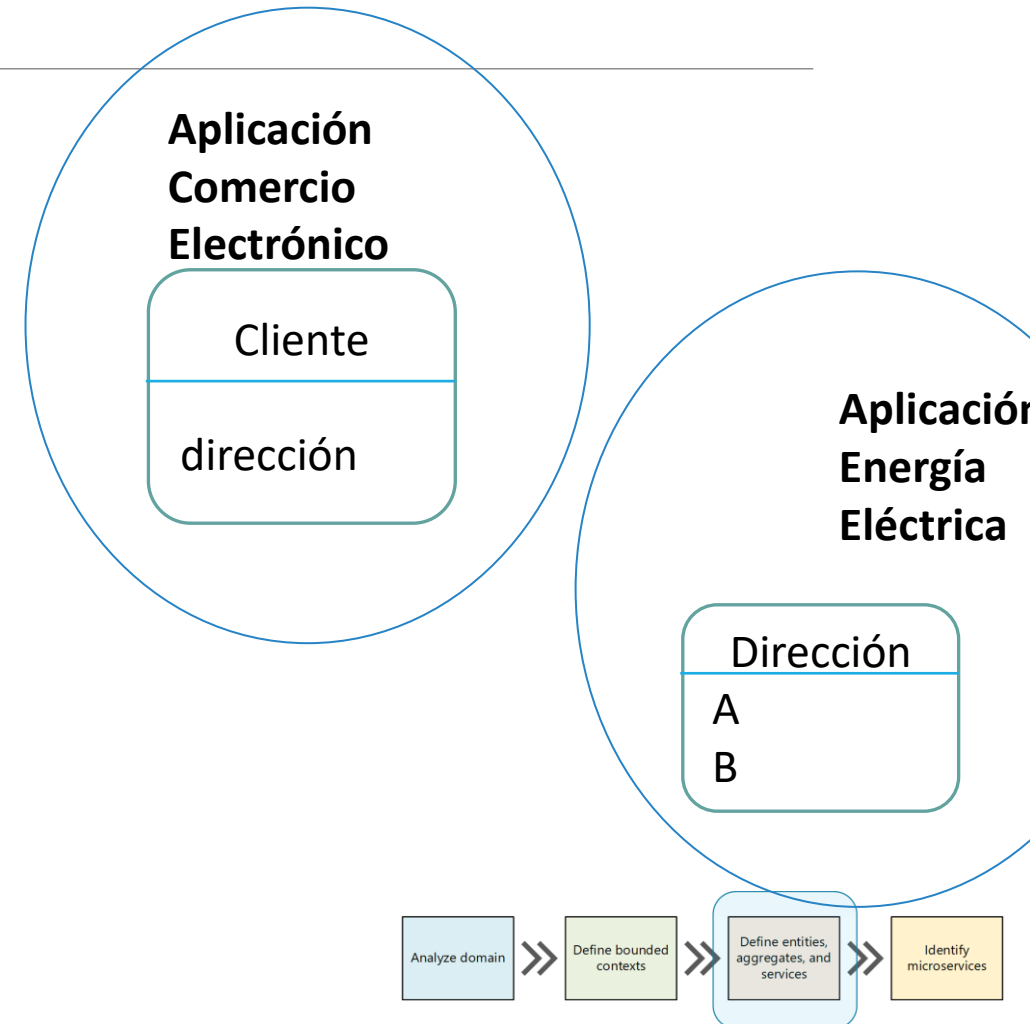
- OBJETOS DE VALOR
- Un objeto de valor es un objeto sin identidad conceptual que describe un aspecto de dominio.
- Se trata de objetos de los que se crea una instancia para representar elementos de diseño que solo interesan temporalmente. Interesa lo que son, no quiénes son.



Definir entidades, agregados y servicios

Análisis de dominios de una arquitectura de microservicios

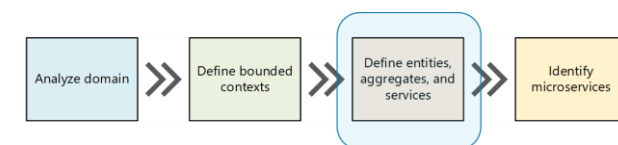
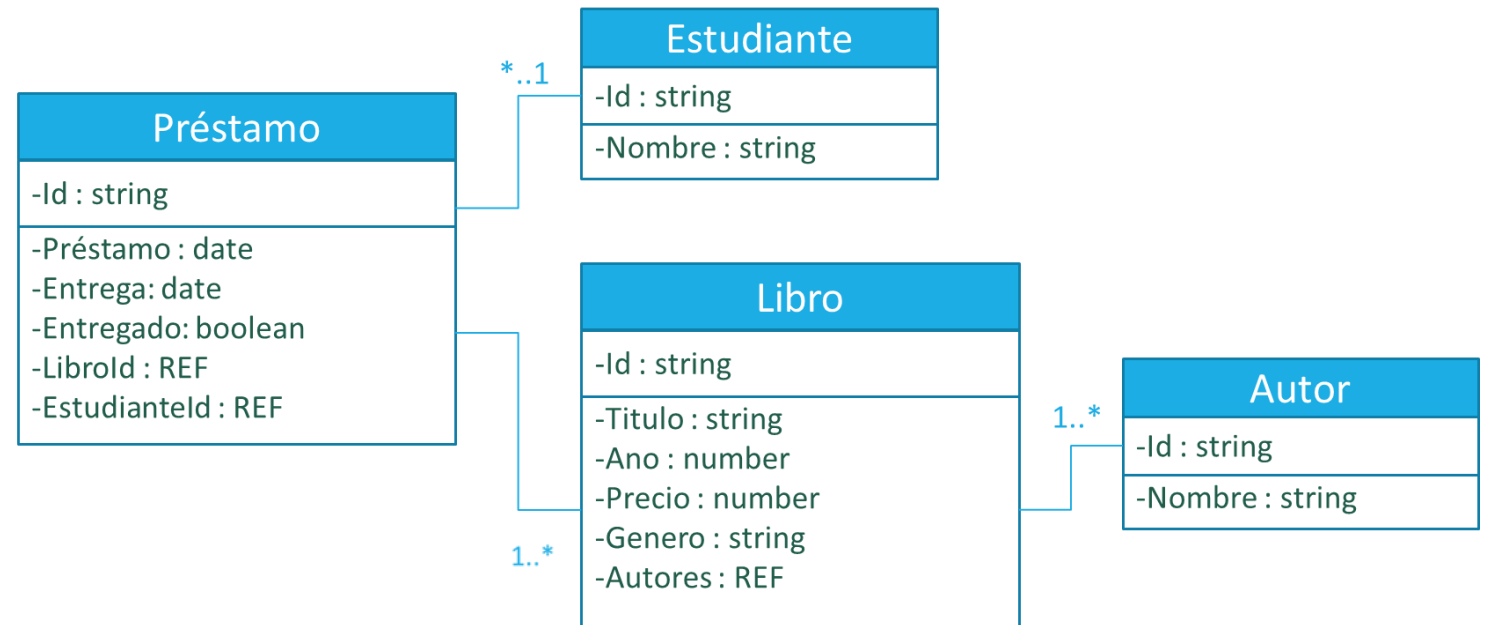
- OBJETOS DE VALOR
- Es posible que algo que sea una entidad en un microservicio no lo sea en otro, porque en el segundo caso, es posible que el contexto delimitado tenga un significado diferente.
- Por ejemplo, una **dirección** en una **aplicación de comercio electrónico** podría solo representar un grupo de atributos del perfil de **cliente**. La dirección es un **objeto de valor**.
- Pero en una aplicación para una empresa de energía eléctrica, la **dirección del cliente** podría ser importante para el dominio de negocio.
- La dirección debe tener una identidad para poder vincular el sistema de facturación directamente con la dirección. En ese caso, una dirección debería clasificarse como una **entidad de dominio**.



Definir entidades, agregados y servicios

Análisis de dominios de una arquitectura de microservicios

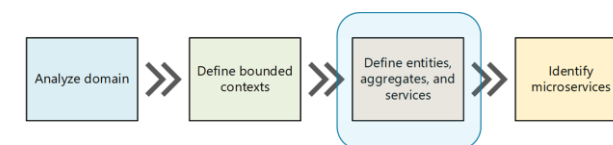
- Entidades: Libro, Préstamo, Autor, Estudiante.
- Objetos de valor: título, año, precio y género de libro; nombre del autor; fecha de préstamo y entrega.



Definir entidades, agregados y servicios

Análisis de dominios de una arquitectura de microservicios

- AGREGADOS
- La mejor manera de identificar agregados es pensar en las operaciones de transacción.
- Contiene grupos de entidades (de datos diferentes y procesos) que pueden controlar un área importante de funcionalidad, como el cumplimiento de pedidos o el inventario.
- Describe un clúster o grupo de entidades y comportamientos que se pueden tratar como una unidad coherente.



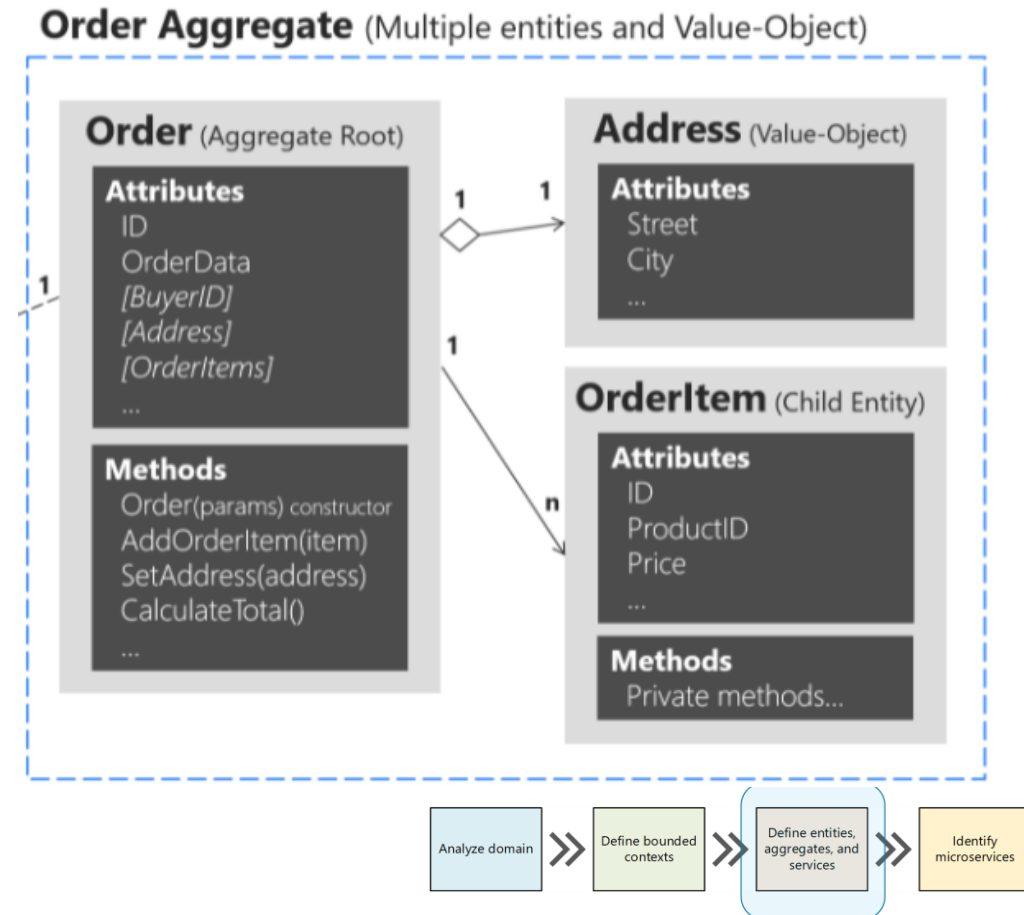
Definir entidades, agregados y servicios

Análisis de dominios de una arquitectura de microservicios

■ AGREGADOS

Por ejemplo:

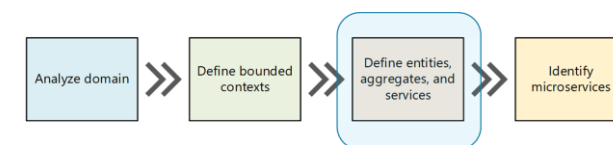
- Una orden de pedido
 - Contiene una lista de elementos de pedido.
 - Un elemento de pedido será una entidad.



Definir entidades, agregados y servicios

Análisis de dominios de una arquitectura de microservicios

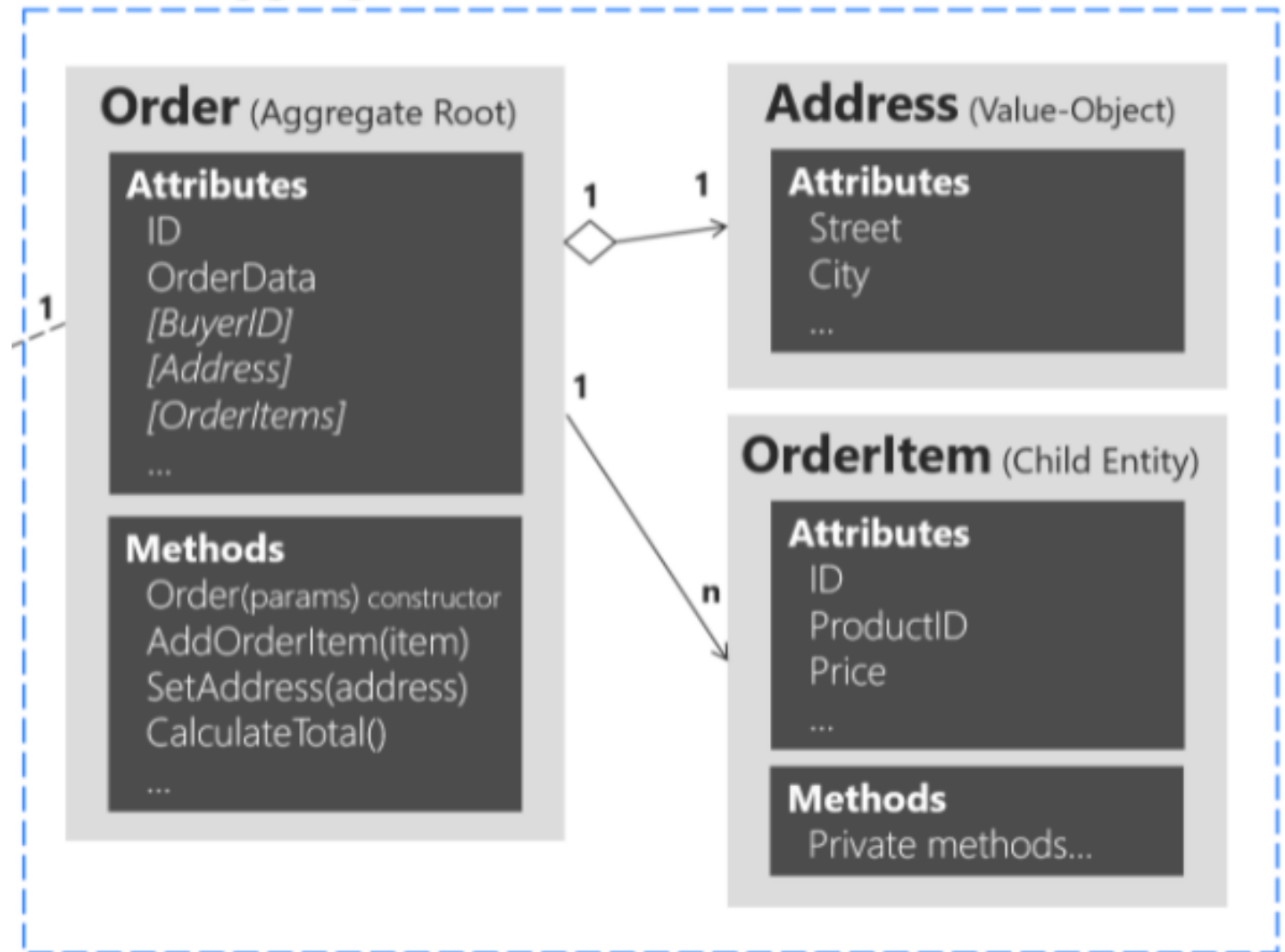
- AGREGADOS
- Un agregado es un grupo de objetos que deben ser coherentes entre sí, pero no se puede seleccionar simplemente un grupo de objetos y etiquetarlos como agregado.
- Debe empezar por un concepto de dominio y pensar en las entidades que se usan en las transacciones más comunes relacionadas con ese concepto.
- Esas entidades que deben ser transaccionalmente coherentes son las que constituyen un agregado.

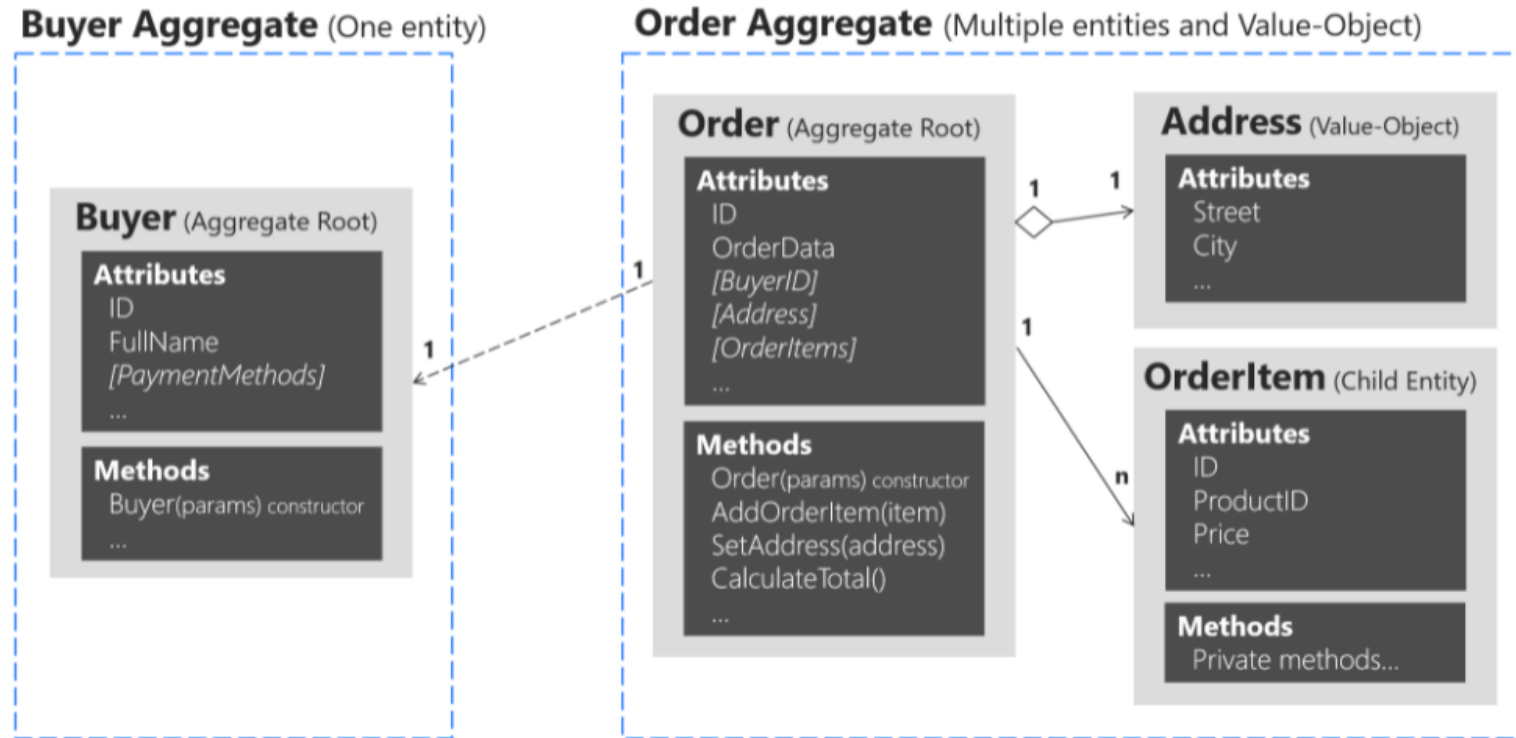


Definir entidades, agregados y servicios
Análisis de dominios de una arquitectura de microservicios

- § AGREGADOS. AGREGADO RAÍZ.
- § Un agregado se compone de al menos una entidad: la raíz agregada, que también se denomina entidad raíz o entidad principal.
- § Además, puede tener varios objetos de valor y entidades secundarias, con todas las entidades y objetos trabajando de forma conjunta para implementar las transacciones y el comportamiento necesarios.

Order Aggregate (Multiple entities and Value-Object)





Definir entidades, agregados y servicios
 Análisis de dominios de una arquitectura de microservicios

§ AGREGADOS. AGREGADO RAÍZ.

§ El propósito de una raíz agregada es asegurar la coherencia del agregado; debe ser el único punto de entrada para las actualizaciones del agregado a través de métodos u operaciones en la clase de raíz agregada.

§ Los cambios en las entidades dentro del agregado solo se deben realizar a través de la raíz agregada

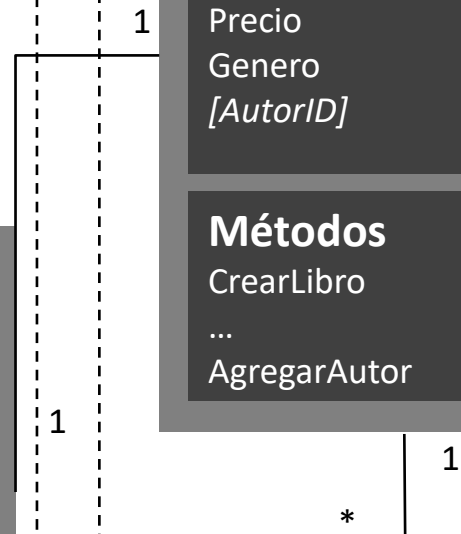
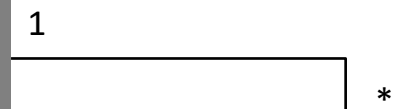
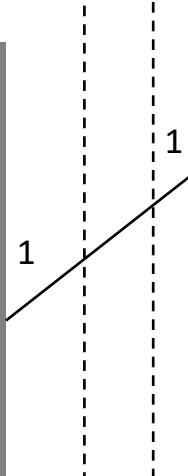
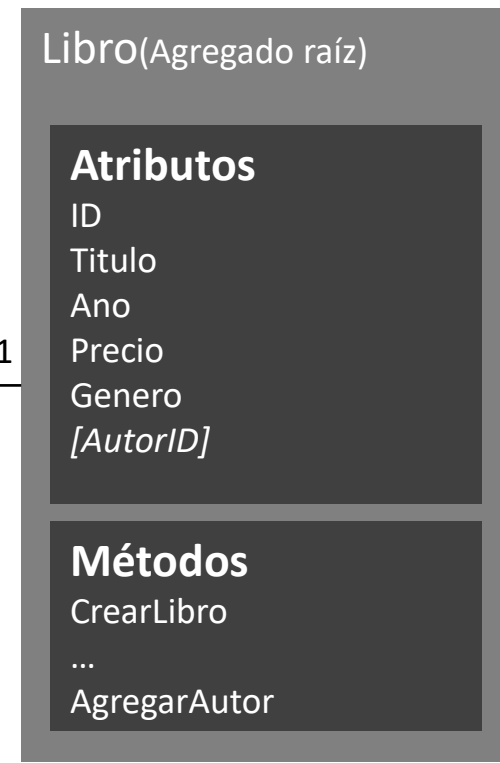
Agregado Estudiante



Agregado Prestamo



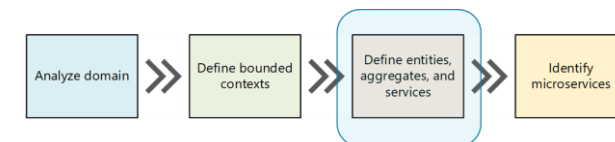
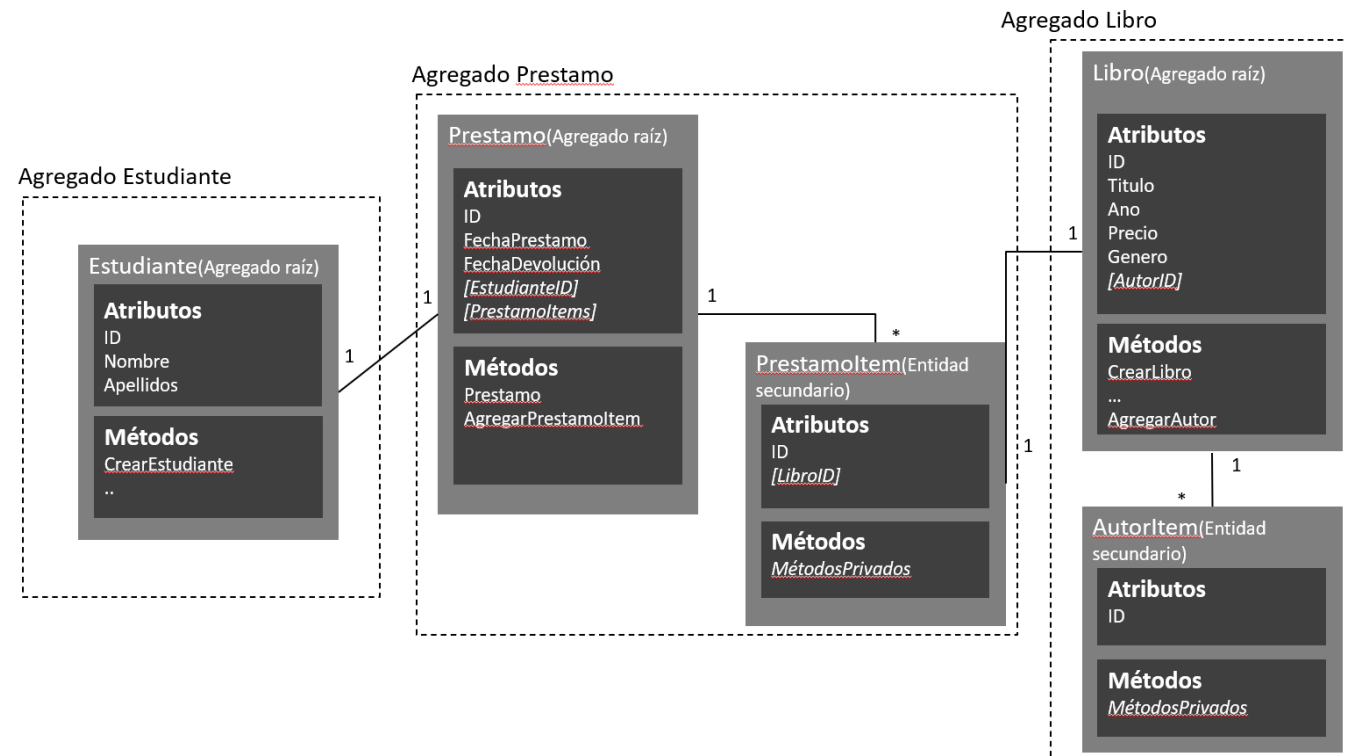
Agregado Libro



Definir entidades, agregados y servicios

Análisis de dominios de una arquitectura de microservicios

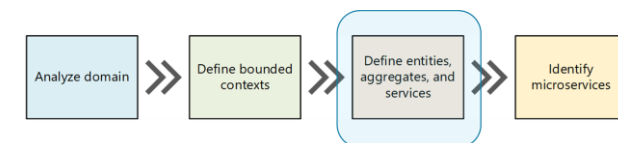
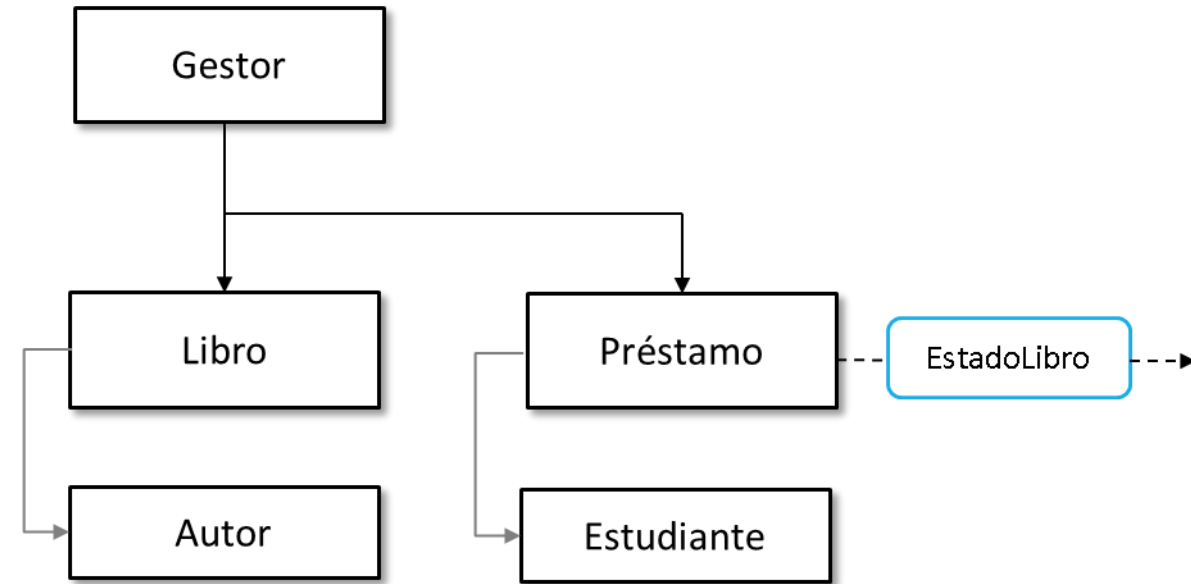
- Entidades: Libro, Préstamo, Estudiante, Prestamoltem, AutorItem.
- Agregados: Libro, Préstamo, Estudiante.
 - Entidades raíz: Libro, Préstamo, Estudiante.
 - Entidades secundarias: AutorItem y Prestamoltem.
- Objetos de valor: título, año, precio y género de libro; nombre del autor; fecha de préstamo y entrega, ID, Nombre, Apellidos.



Definir entidades, agregados y servicios

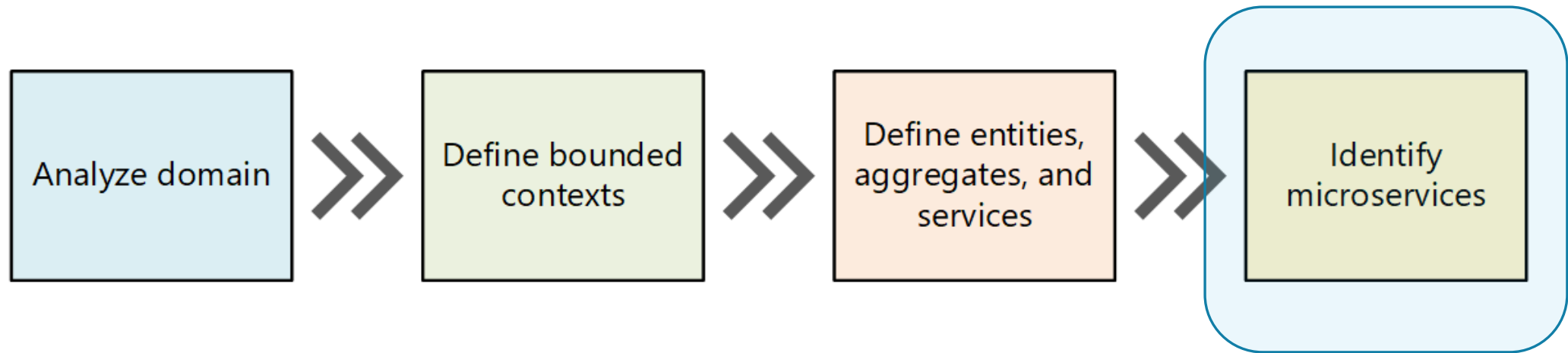
Análisis de dominios de una arquitectura de microservicios

- Evento de dominio: Préstamo que envía los eventos *EstadoLibro* que describiría el estado y fecha de entrega de un libro.
- Servicio de dominio: *Gestor* que coordina las peticiones de los Estudiantes.



Fases del análisis de dominios

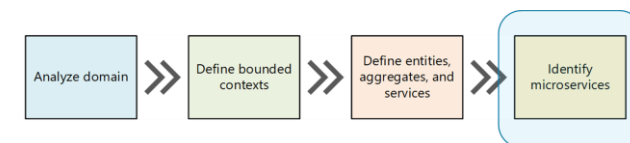
Análisis de dominios de una arquitectura de microservicios



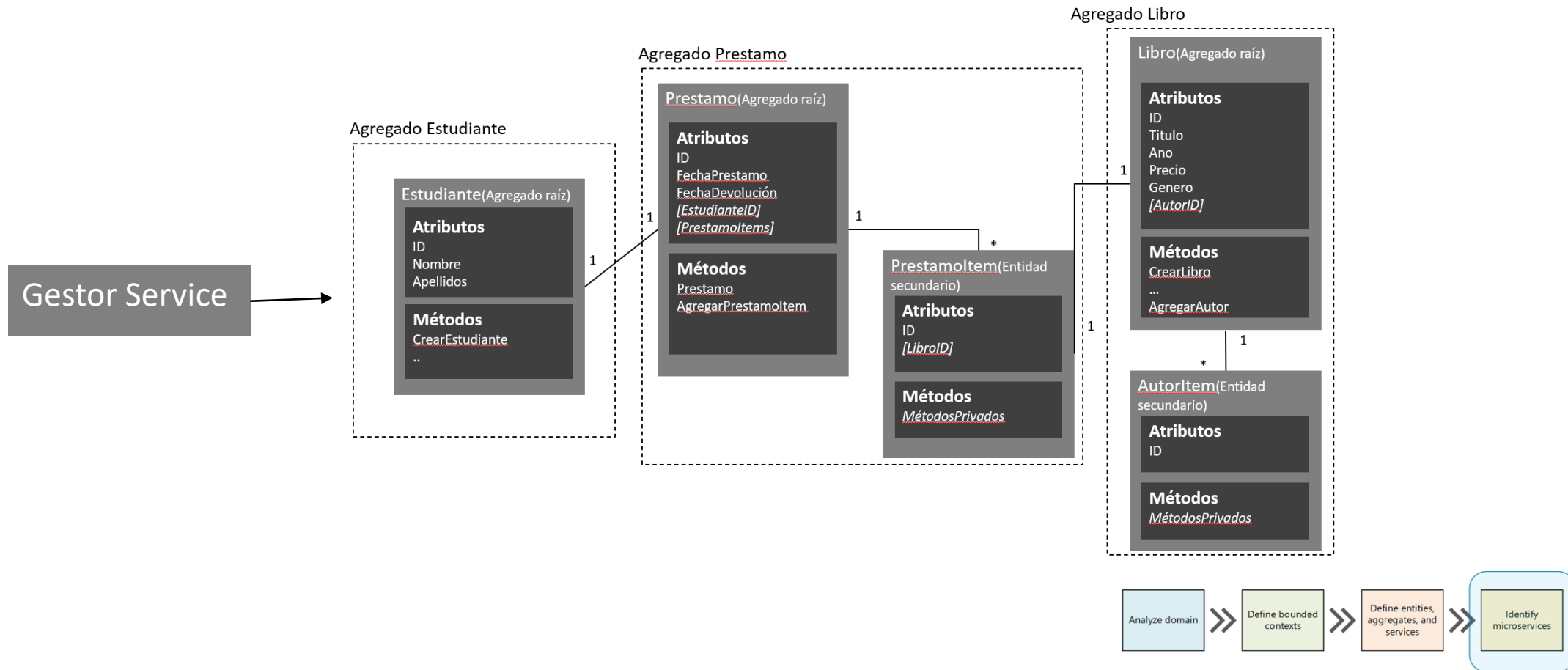
Identificar microservicios

Análisis de dominios de una arquitectura de microservicios

1. Comenzar con un contexto delimitado. En general, la funcionalidad en un microservicio no debe abarcar más de un contexto delimitado.
2. A continuación, analizar los agregados en el modelo de dominio. Los agregados suelen ser buenos candidatos para microservicios.
3. Los servicios de dominio también son buenos candidatos para microservicios. Los servicios de dominio son operaciones sin estado a través de varios agregados.
4. Por último, tener en cuenta los requisitos no funcionales: tamaño del equipo, los tipos de datos, tecnologías, requisitos de escalabilidad, requisitos de disponibilidad y requisitos de seguridad.



Identificar microservicios



Diseño de una arquitectura de microservicios

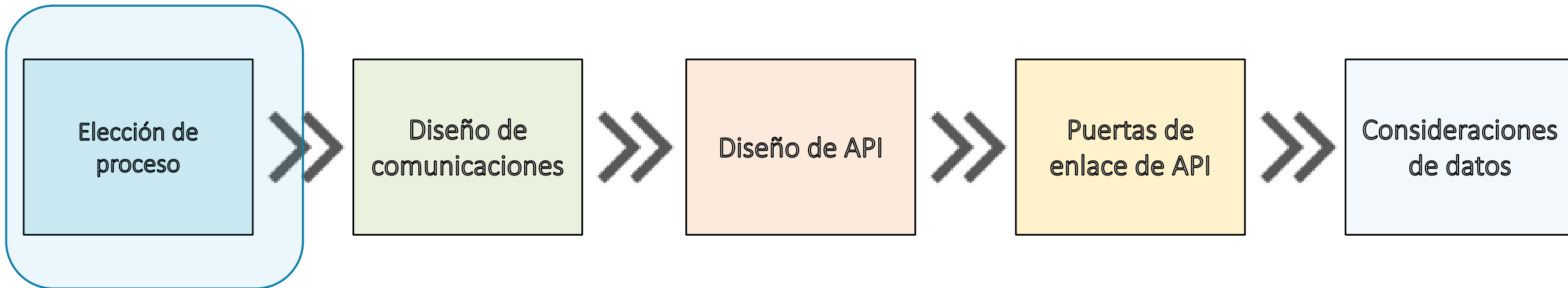
Proceso para el desarrollo de una arquitectura de microservicios

DISEÑO E IMPLEMENTACIÓN DE UNA ARQUITECTURA DE
MICROSERVICIOS

Fases del diseño

Diseño de una arquitectura de microservicios

Microsoft en su “guía para el diseño de una arquitectura de microservicios”, lista los pasos para diseñar una arquitectura de microservicios:



Eric Evans

Elección de una opción de proceso para los microservicios

Diseño de una arquitectura de microservicios

- El término *proceso* hace referencia al modelo de hospedaje para los recursos informáticos donde se ejecutan las aplicaciones.
- Para una arquitectura de microservicios, hay dos enfoques especialmente populares:
 - Una **arquitectura con orquestador de servicios** (Kubernetes, Docker Swarm) que administra los servicios que se ejecutan en nodos dedicados (contenedores).
 - Una **arquitectura sin orquestador**.



Orquestadores de servicios

Diseño de una arquitectura de microservicios

- Controla las tareas relacionadas con la implementación y administración de un conjunto de servicios.
 - Colocar servicios en los nodos.
 - Supervisar el estado de los servicios
 - Reiniciar los servicios en mal estado.
 - Equilibrar la carga del tráfico de red.
 - Escalar el número de instancias.
 - Aplicar actualizaciones de configuración.



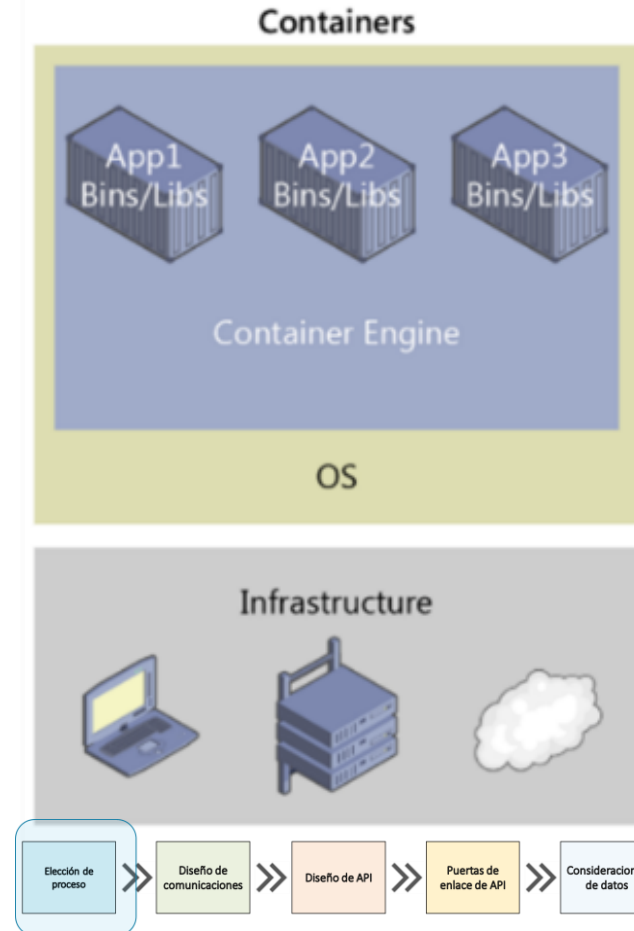
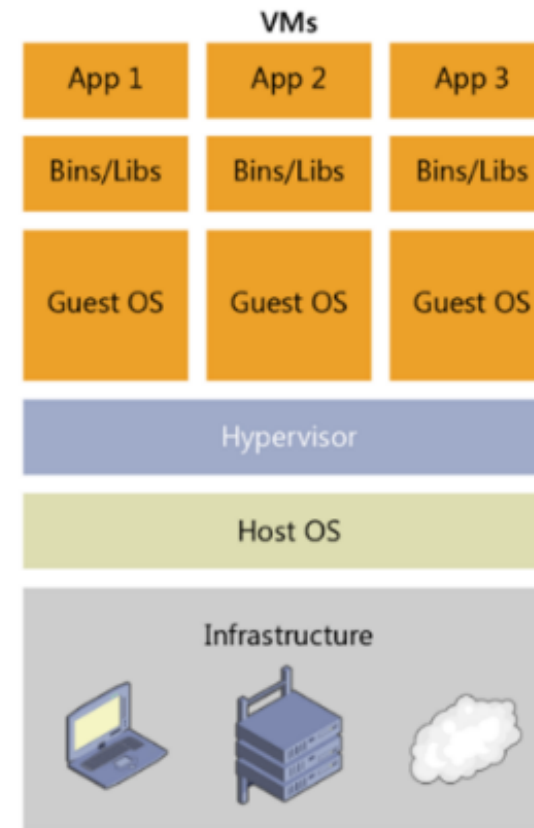
Microsoft Azure Service Fabric



Contenedores

Diseño de una arquitectura de microservicios

- Es una tecnología para empaquetar una aplicación o un servicio e implementarlo de forma confiable y reproducible.
- Similar a una máquina virtual pero más ligero.
- Funciona utilizando el sistema operativo que tiene la máquina en la que se ejecuta el contenedor.



Contenedores y microservicios

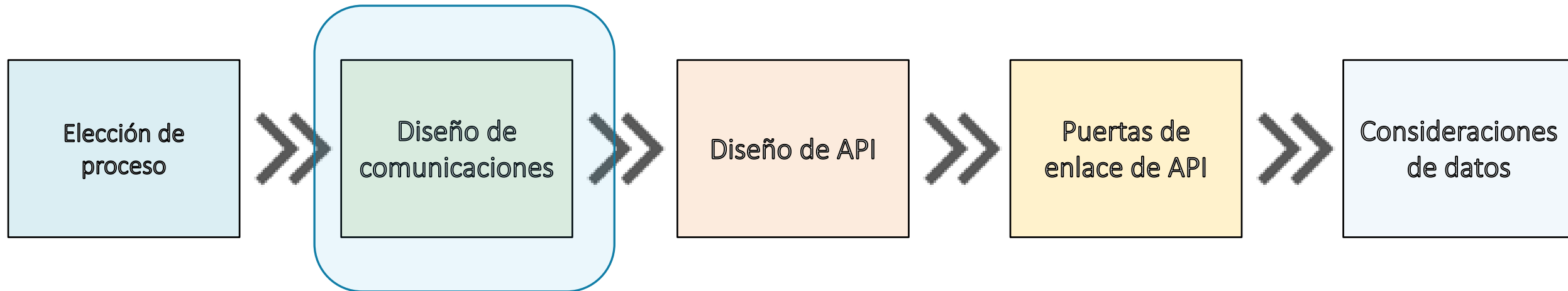
Diseño de una arquitectura de microservicios

- Los contenedores no son necesarios para generar microservicios.
- Los contenedores tienen algunas de las ventajas que son especialmente apropiadas para los microservicios:
 - Portabilidad
 - Agilidad
 - Aislamiento de recursos



Fases del diseño

Diseño de una arquitectura de microservicios



Eric Evans

Diseño de comunicaciones

Diseño de una arquitectura de microservicios

- Dos patrones de mensajería básicos (Microsoft, 2020):
 - **Comunicación sincrónica.** Un servicio llama a una API que otro servicio expone mediante un protocolo como HTTP o RPC.
 - El autor de la llamada espera a la respuesta del receptor.
 - **Paso de mensajes asíncronos.** Un servicio envía el mensaje sin esperar por la respuesta, y uno o más servicios procesan el mensaje de forma asíncrona.



Diseño de comunicaciones

Diseño de una arquitectura de microservicios

- Ventajas de la mensajería asíncrona:
 - El remitente del mensaje no tiene que conocer al consumidor
 - Si se produce un error en el consumidor, el remitente puede seguir enviando mensajes.
 - Las colas pueden usarse para administrar un flujo de trabajo

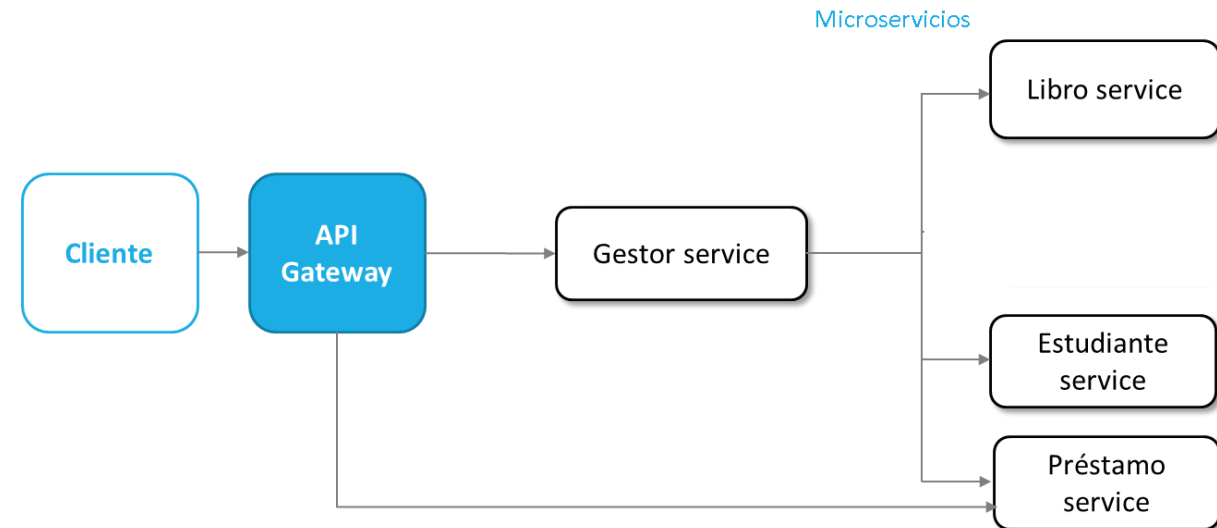
- Desventajas de la mensajería asíncrona:
 - Latencia
 - Costo
 - Complejidad



Diseño de comunicaciones: Ejemplo

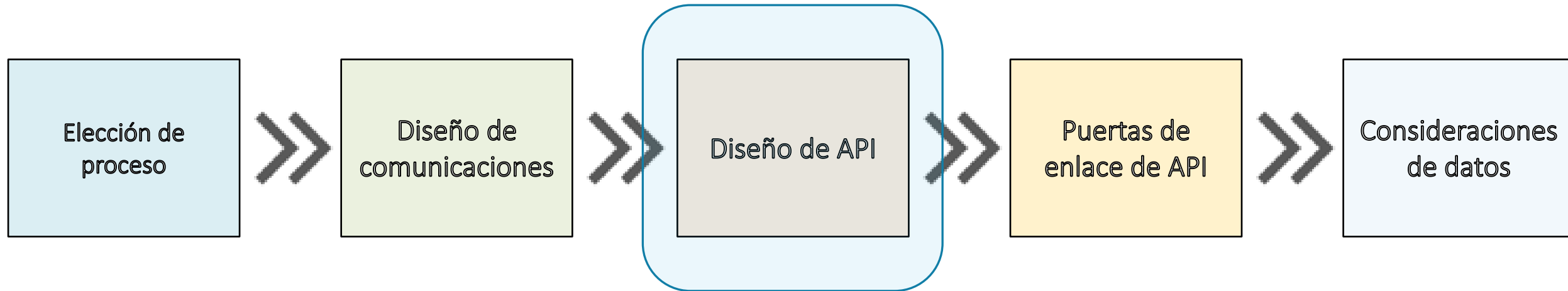
Diseño de una arquitectura de microservicios

- El servicio Gestor expone una API de REST pública.
- Todos los servicios Libro, Autor, Estudiante, Préstamo exponen las API de REST internas.
- El servicio Gestor llama a estas API para llevar a cabo una solicitud de usuario.
- Una razón para usar las API sincrónicas es que Gestor necesita obtener una respuesta de cada uno de los servicios de bajada.
- El servicio de Préstamo expone una API pública que los clientes pueden utilizar para obtener el estado de un libro.



Fases del diseño

Diseño de una arquitectura de microservicios



Eric Evans

Diseño de API

Diseño de una arquitectura de microservicios

- Todos los intercambios de datos entre servicios se producen mediante mensajes o llamadas API.
 - Las API públicas que llaman las aplicaciones cliente.
 - Las API de back-end que se usan para la comunicación entre servicios.

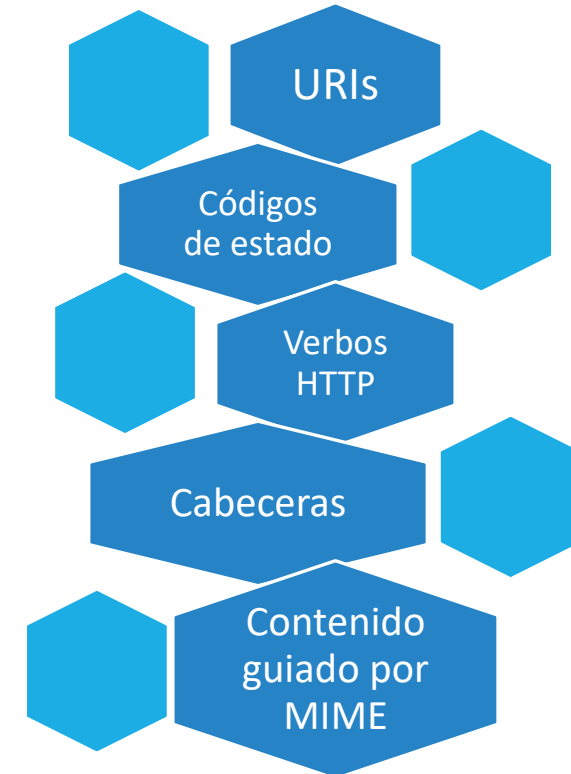


REST

Diseño de una arquitectura de microservicios

- REST (Representational State Transfer) es un estilo de arquitectura de software para sistemas distribuidos en la Web.
- Las API de REST se diseñan en torno a recursos (cualquier tipo de objeto, dato o servicio al que puede acceder el cliente.)
- Un recurso tiene un identificador, que es un URI que identifica de forma única ese recurso

<https://www.uv.mx/ordenes/1>



REST

Diseño de una arquitectura de microservicios

- Los clientes interactúan con un servicio mediante el intercambio de representaciones de recursos. Muchas API web usan JSON como formato de intercambio.
- El protocolo HTTP define una serie de métodos que asignan significado semántico a una solicitud:
 - GET
 - POST
 - PUT
 - DELETE

```
{"orderId":1,"orderValue":99.90  
,"productId":1,"quantity":1}
```



Asignación de REST a patrones de diseño basado en el dominio

Diseño de una arquitectura de microservicios

- Los agregados se asignan de forma natural a los recursos de REST.
- Las entidades tienen identidad única. En REST, los recursos tienen identificadores únicos en forma de direcciones URL.
- A las entidades secundarias de un agregado se accede desde la entidad raíz.
- Dado que los objetos de valor son inmutables, las actualizaciones se realizan mediante el reemplazo del objeto de valor completo mediante solicitudes PUT.



Equivalencias entre el DDD y REST

Diseño de una arquitectura de microservicios

Concepto de DDD	Equivalente en REST	Ejemplo
Agregado	Recurso	{ "1":1234, "status":"pending"... }
Identidad	URL	https://delivery-service/deliveries/1
Entidades secundarias	Vínculos	{ "href": "/deliveries/1/confirmation" }
Actualización de objetos de valor	PUT o PATCH	PUT https://delivery-service/deliveries/1/dropoff



REST: Ejemplo

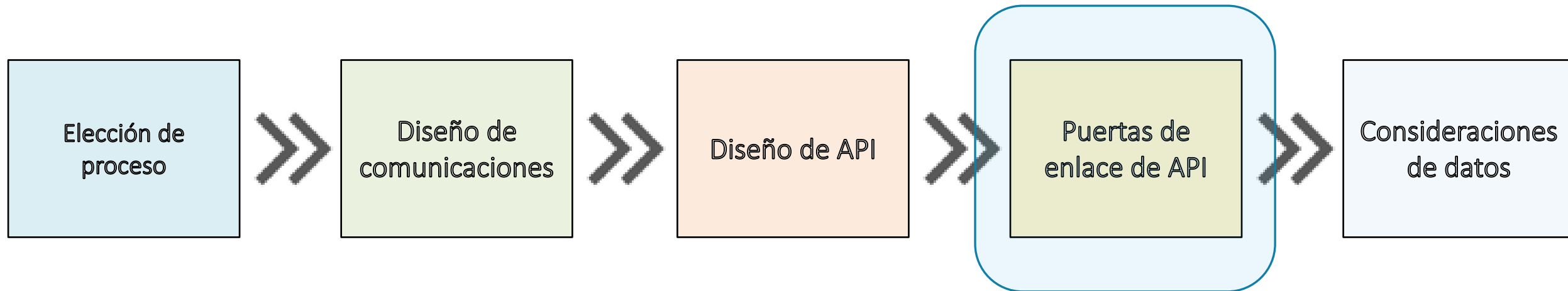
Diseño de una arquitectura de microservicios

Recurso	POST	GET	PUT	DELETE
/libros	Crear un nuevo libro	Recuperar todos los libros	Actualización masiva de libros	Eliminar todos los libros
/libros/1	Error	Recuperar los detalles del libro 1	Actualizar los detalles del libro 1 si existe	Quitar al libro 1
/libros/1/autor	Crear un nuevo autor para el libro 1	Recuperar el autor del libro 1	Actualizar el autor del libro 1	Quitar el autor del libro 1



Fases del diseño

Diseño de una arquitectura de microservicios



Eric Evans

Puertas de enlace API

Diseño de una arquitectura de microservicios

- Una puerta de enlace de API se ubica entre los clientes y los servicios. Actúa como un proxy inverso, enrutando las solicitudes de los clientes a los servicios.
- Si no se implementa una puerta de enlace, los clientes deben enviar las solicitudes directamente a los servicios front-end.



Puertas de enlace API

Diseño de una arquitectura de microservicios

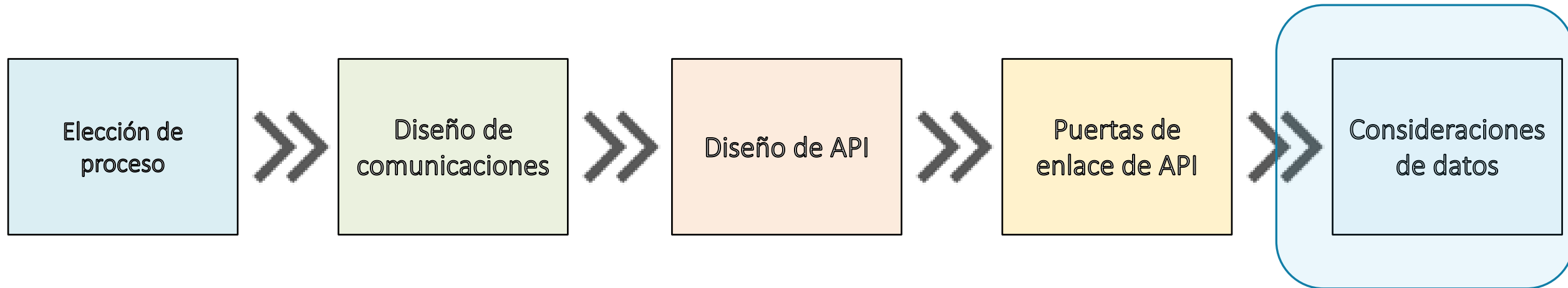
Existen muchas opciones para implementar una puerta de enlace de API en una aplicación, aquí se enlistan algunas:

- Servidor de proxy inverso como Nginx y HAProxy.
- Controlador de entrada del servicio de malla.
- Azure Application Gateway.
- Azure API Management.
- **API Gateway del framework de desarrollo empleado.**



Fases del diseño

Diseño de una arquitectura de microservicios

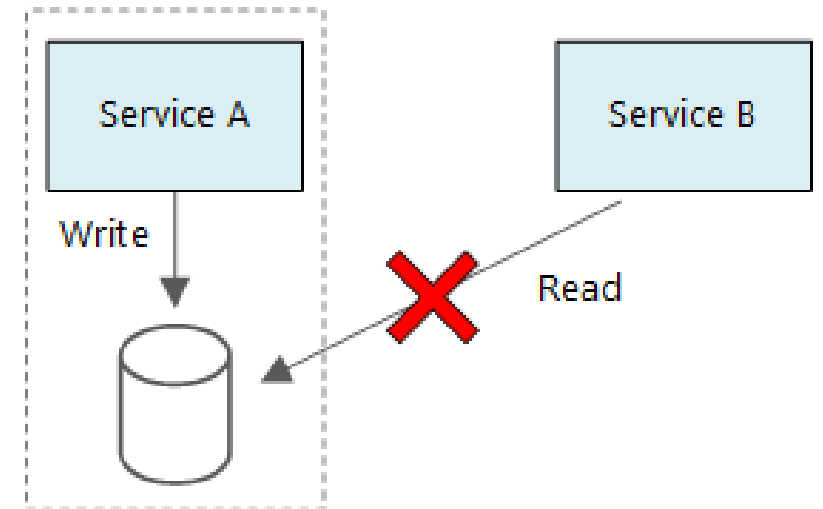


Eric Evans

Consideraciones de datos para microservicios

Diseño de una arquitectura de microservicios

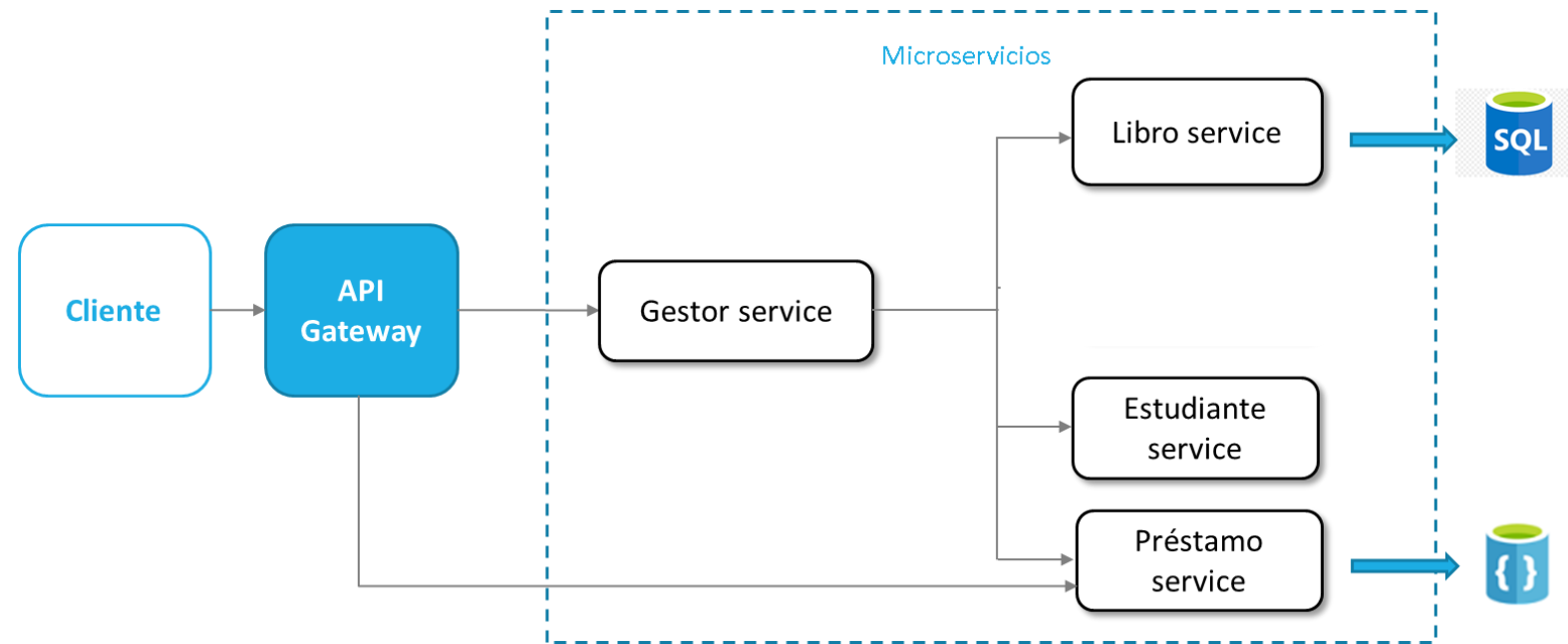
- Un principio básico de los microservicios es que cada servicio administra sus propios datos.
- Dos servicios no deben compartir un mismo almacén de datos.
- Almacenar únicamente los datos que necesita un servicio.



Consideraciones de datos para microservicios: Ejemplo

Diseño de una arquitectura de microservicios

- Microservicio Libro puede utilizar una base de datos SQL Relacional.
- Microservicio Préstamo, se requiere un almacenamiento a largo plazo, así como un gran volumen de elementos y compatibilidad.
- BD orientada a documentos como MongoDB o Cosmos DB.



Gracias por su atención

Referencias

- Arsanjani, A. (2004). Service-oriented modeling and architecture. IBM.
- Barcia , R., Brown, K., & Osowski, R. (2018). Guía de Microservicios, Punto de vista. Obtenido de IBM:
<https://www.ibm.com/downloads/cas/5O8YOPKN>
- Bell, M. (2008). Service-Oriented Modeling Service Analysis, Design, and Architecture (1era ed.). Wiley.
- Biggs, J., García, V., & Salanova, J. (2019). Building Intelligent Cloud Applications: Develop Scalable Models Using Serverless Architectures with Azure. O'Reilly Media.
- Evans, E. (2003). Domain-Driven Design: Tackling Complexity in the Heart of Software (1era ed.). Addison-Wesley Professional.



Referencias

- Evans, E. (16 de abr de 2016). DDD & Microservices: At last, some boundaries! Obtenido de QCon: <https://www.infoq.com/presentations/ddd-microservices-2016/>
- Hiberus. (2019). De una arquitectura tradicional a microservicios. Obtenido de Hiberus Tecnología: <https://www.hiberus.com/crecemos-contigo/de-una-arquitectura-tradicional-a-microservicios/>
- IBM Cloud Education. (2019). SOA (Service-Oriented Architecture). Obtenido de IBM: <https://www.ibm.com/cloud/learn/soa>
- Microsoft. (13 de 07 de 2016). Implementación de API web. Obtenido de Microsoft Azure Docs: <https://docs.microsoft.com/es-es/azure/architecture/best-practices/api-implementation#publishing-and-managing-a-web-api>



Referencias

- Microsoft. (20 de sep de 2018). Arquitectura orientada a servicios. Obtenido de Guía de la arquitectura de aplicaciones .NET: <https://docs.microsoft.com/es-es/dotnet/architecture/microservices/architect-microservice-container-applications/service-oriented-architecture>
- Microsoft. (20 de ago de 2018). Creación de interfaces de usuario compuestas basadas en microservicios. Obtenido de Libro electrónico sobre la arquitectura de microservicios de .NET: <https://docs.microsoft.com/es-es/dotnet/architecture/microservices/architect-microservice-container-applications/microservice-based-composite-ui-shape-layout>
- Microsoft. (10 de ago de 2018). Design a DDD-oriented microservice. Obtenido de .NET microservices - Architecture e-book: <https://docs.microsoft.com/en-us/dotnet/architecture/microservices/microservice-ddd-cqrs-patterns/ddd-oriented-microservice>

Referencias

- Microsoft. (30 de oct de 2019). Creación de microservicios en Azure. Obtenido de Azure Architecture Center: <https://docs.microsoft.com/es-es/azure/architecture/microservices/>
- Microsoft. (26 de feb de 2019). Diseño de una arquitectura de microservicios. Obtenido de <https://docs.microsoft.com/es-es/azure/architecture/microservices/design/>
- Microsoft. (14 de may de 2019). Estilo de arquitectura de microservicios. Obtenido de Microsoft Azure Docs: <https://docs.microsoft.com/es-es/azure/architecture/guide/architecture-styles/>
- Microsoft. (14 de may de 2019). Estilos de arquitectura. Obtenido de Guía de arquitectura de aplicaciones: <https://docs.microsoft.com/es-es/azure/architecture/guide/architecture-styles/>