M&GT Consulting

INICIO

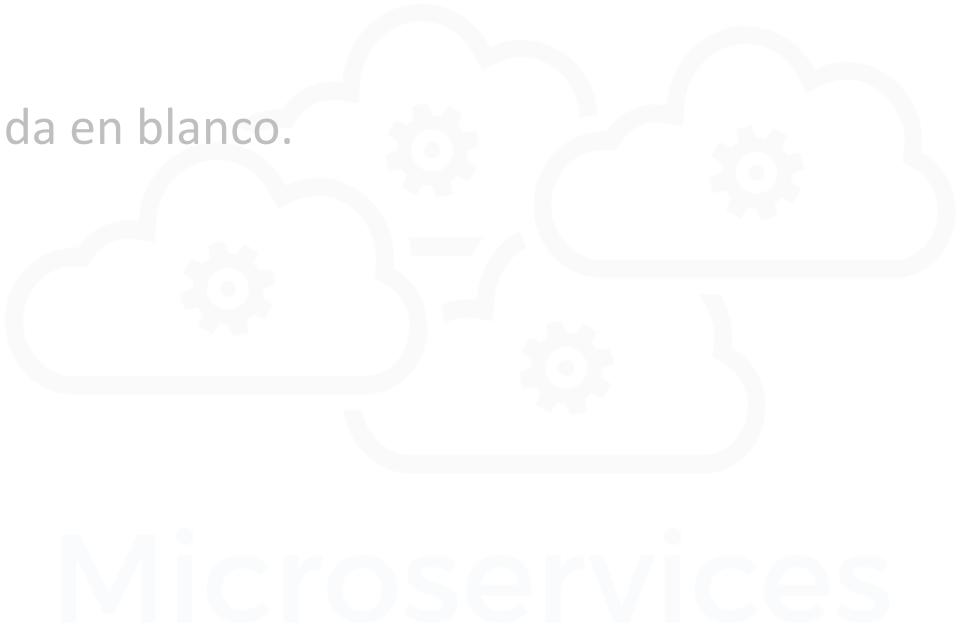
Desarrollo de Microservicios con Spring Cloud Netflix OSS

ISC. Ivan Venor García Baños

Agenda

1. Presentación
2. Objetivos
3. Contenido
4. Despedida

Esta página fue intencionalmente dejada en blanco.



3. Contenido

i. Arquitectura de sistemas monolíticos
ii. Introducción a la Arquitectura Orientada a Servicios
iii. Fundamentos Spring Boot 2.x
iv. Arquitectura de Microservicios
v. Microservicios con Spring Cloud y Spring Cloud Netflix OSS

3. Contenido

i. Arquitectura de sistemas monolíticos
ii. Introducción a la Arquitectura Orientada a Servicios
iii. Fundamentos Spring Boot 2.x
iv. Arquitectura de Microservicios
v. Microservicios con Spring Cloud y Spring Cloud Netflix OSS

i. Arquitectura de sistemas monolíticos

i. Arquitectura de sistemas monolíticos

i.i ¿Qué son los sistemas monolíticos?

i.ii Escalabilidad de sistemas monolíticos

i.iii Protocolos de integración

i.i ¿Qué son los sistemas monolíticos?

Objetivos de la lección

i.i ¿Qué son los sistemas monolíticos?

- Comprender que son los sistemas monolíticos.
- Analizar las ventajas y desventajas de la arquitectura de sistemas monolíticos.

i.i ¿Qué son los sistemas monolíticos? (a)

- El término monolito o monolítico (monolith) se refiere a un estilo de arquitectura o a un patrón (o anti-patrón) de desarrollo de software.

- Diferentes estilos de arquitectura o patrones de desarrollo de software se clasifican en diferentes tipos de vista o “viewtypes” (un “viewtype” es un conjunto o categoria de cómo se visualiza una arquitectura de software).

i.i ¿Qué son los sistemas monolíticos? (b)

- El estilo de arquitectura monolítica se visualiza principalmente en tres tipos de vista (viewtypes) los cuales son:
  - Módulos (modules)
  - Asignación (allocation) y,
  - Ejecución (runtime)

i.i ¿Qué son los sistemas monolíticos? (c)

- Módulos monolíticos.

- Se refiere a que todo el código de un sistema se encuentra en un único código base o código fuente que es compilado y produce un único artefacto o pieza de software.

- El código fuente puede estar bien estructurado, mediante clases y paquetes, y puede estar escrito implementando las mejores prácticas de desarrollo sin embargo, no se encuentra dividido en distintos módulos para su mantenimiento, compilación, despliegue y ejecución.

i.i ¿Qué son los sistemas monolíticos? (d)

- Módulos monolíticos.

- En el estricto sentido, el diseño de un módulo no-monolítico mantiene el código fuente del sistema dividido o separado en múltiples módulos o librerías los cuales pueden ser compilados, mantenidos, desplegados y ejecutados de manera separada.

- A su vez, el diseño de un módulo no-monolítico puede estar almacenado en diferentes codigos base y diferentes repositorios, lo que permite que puedan ser referenciados o modificados de forma independiente cuando sea necesario.

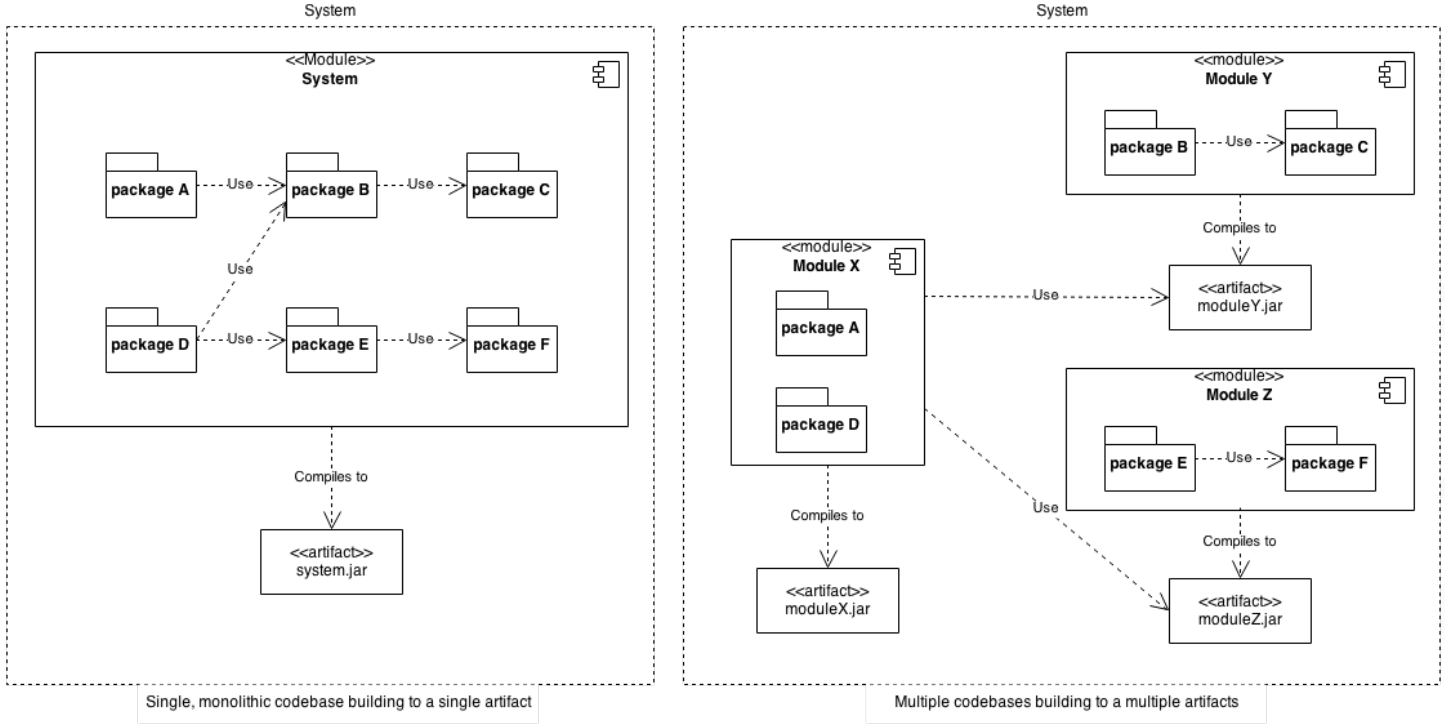
i.i ¿Qué son los sistemas monolíticos? (e)

- Módulos monolíticos.

- Existen ventajas y desventajas referentes a módulos monolíticos y no-monolíticos sin embargo, únicamente estamos referenciando cómo el código fuente, de ambas estratégias, es utilizado.

i.i ¿Qué son los sistemas monolíticos? (f)

- Módulos monolíticos vs no-monolíticos.



i.i ¿Qué son los sistemas monolíticos? (g)

- Asignación (allocation) monolítica.

- Se refiere a que todo el código fuente del módulo o sistema monolítico ha sido empaquetado y desplegado en el mismo momento, al mismo tiempo.

- La asignación monolítica, se refiere a que, una vez que el código fuente ha sido compilado y esta listo para ser liberado, existe una única versión del empaquetado el cuál contiene todos los componentes necesarios para su ejecución.

i.i ¿Qué son los sistemas monolíticos? (h)

- Asignación (allocation) monolítica.

- Todos los componentes en ejecución tienen la misma versión del software que se ejecuta en un momento determinado.

- La asignación (allocation) monolítica es independiente de si la estructura (a nivel código) del módulo es un monolito o no, debido a una de las dos posibles asignaciones (allocations) siguientes:
  - Despliegue de un único compilado, de un único código fuente o,
  - Despliegue simultaneo de múltiples compilados, de distintos códigos fuente.

i.i ¿Qué son los sistemas monolíticos? (i)

- Asignación (allocation) monolítica.

- Despliegue de un único compilado, de un único código fuente.
  - Es posible que se haya compilado todo el código base a la vez antes de su despliegue.

i.i ¿Qué son los sistemas monolíticos? (j)

- Asignación (allocation) monolítica.

- Despliegue simultaneo de múltiples compilados de distintos códigos fuente.
  - Es posible que se hayan compilado un conjunto de artefactos diferentes, desde repositorios de código diferentes lo cual origino diferentes empaquetados si, todos sus componentes han sido desplegados a la vez, al mismo tiempo, en el mismo despliegue y ello a deteniendo la operación de todo el sistema debido al despliegue del sistema el cuál ha sido, después, reiniciado.

i.i ¿Qué son los sistemas monolíticos? (k)

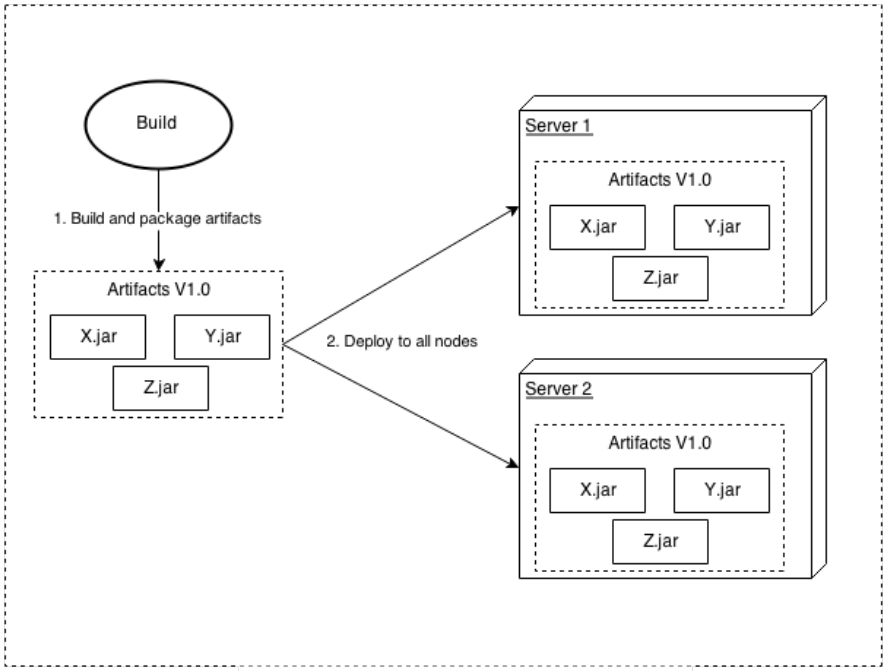
- Asignación (allocation) monolítica.

- La asignación (allocation) no-monolítica implicaría desplegar diferentes compilados (con diferentes versiones de código o no), en diferentes nodos o equipos, en diferentes momentos y que la afectación del reinicio del sistema, sólo afecta al nodo donde se está realizando el despliegue.

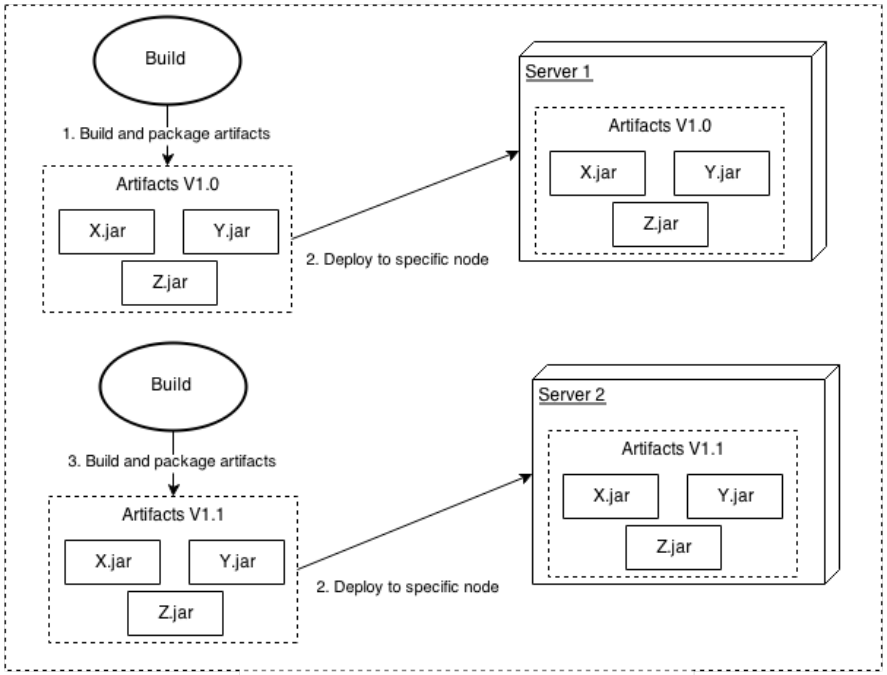
- De nueva cuenta, la asignación monolítica o no-monolítica es independiente de la estructura de código del módulo o sistema ya que, diferentes versiones de un módulo monolítico pueden desplegarse de forma independiente.

i.i ¿Qué son los sistemas monolíticos? (I)

- Asignación (allocation) monolítica vs no-monolítica.



Single, monolithic allocation/deploy



Non-monolithic allocation of monolithic modules

i.i ¿Qué son los sistemas monolíticos? (m)

- Ejecución (runtime) monolítica.
- Un monolito en tiempo de ejecución tiene un solo proceso que ejecuta el trabajo del sistema (task).
- Desde los inicios de la informática, muchos sistemas “tradicionales” se han escrito de forma monolítica y se ejecutan en un único proceso.
- La ejecución monolítica es independiente de si la estructura del código fuente del módulo (o sistema) es monolítica o no-monolítica.

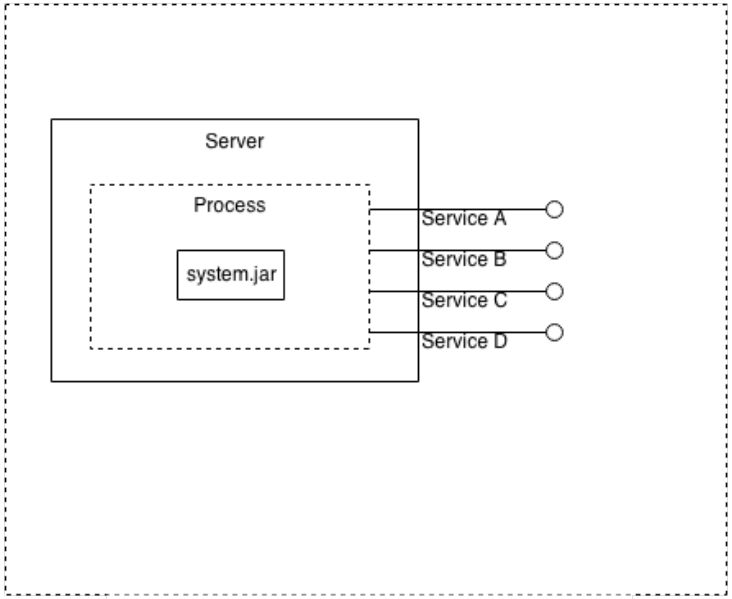
i.i ¿Qué son los sistemas monolíticos? (n)

- Ejecución (runtime) monolítica.

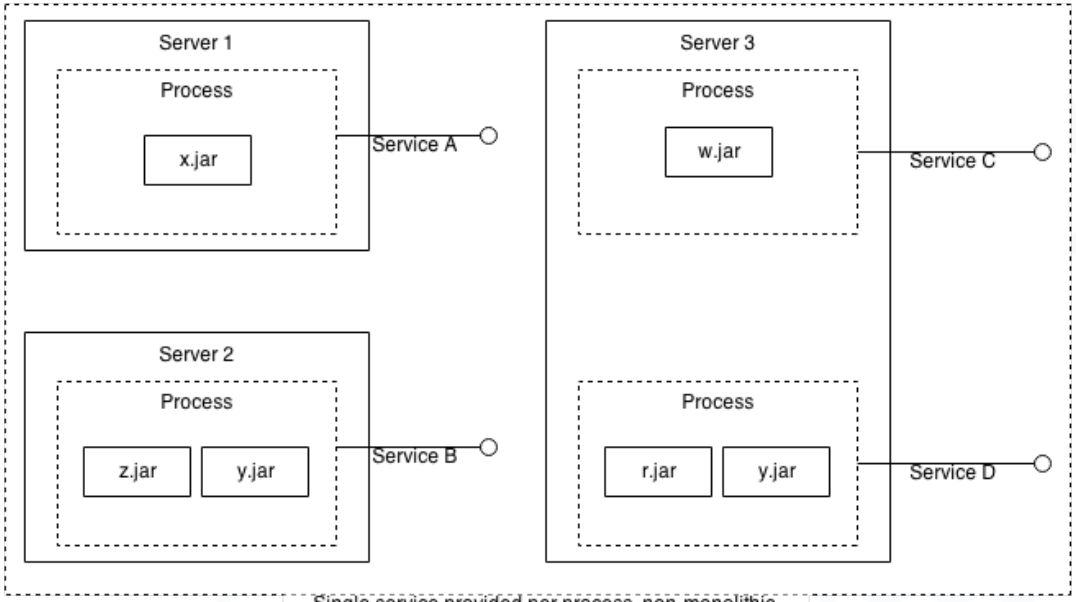
- Un monolito de tiempo de ejecución a menudo implica un monolito de asignación (asignación monolítica) si solo se despliega en un único nodo o equipo como componente principal.

i.i ¿Qué son los sistemas monolíticos? (ñ)

- Ejecución (runtime) monolítica.



Monolithic Runtime with monolithic module and allocation



Single service provided per process, non-monolithic modules and allocation

i.i ¿Qué son los sistemas monolíticos? (o)

- Ventajas de los sistemas monolíticos (a):

- Facilidad para desarrollar en entornos no colaborativos.
- Facilidad de “debug & test”.
- Performance, acceder a datos compartidos en memoria es más rápido que comunicación entre procesos (IPC) y aún más rápido que el ”overhead” que ocasiona comunicación entre procesos en diferentes nodos mediante protocolos de comunicación como HTTP.
- Sencillos de construir (compilar).
- Simples de desplegar.
- Facilidad de escalar horizontalmente.

i.i ¿Qué son los sistemas monolíticos? (p)

- Ventajas de los sistemas monolíticos (b):

- Más fáciles de implementar seguridad.
- Facilidad de monitoreo, operaciones.
- Facilidad de planificación agile.
- Fáciles de diseñar tradicionalmente (basado en capas).
- Facilidad para la implementación de AOP.

i.i ¿Qué son los sistemas monolíticos? (q)

- Desventajas de los sistemas monolíticos (a):

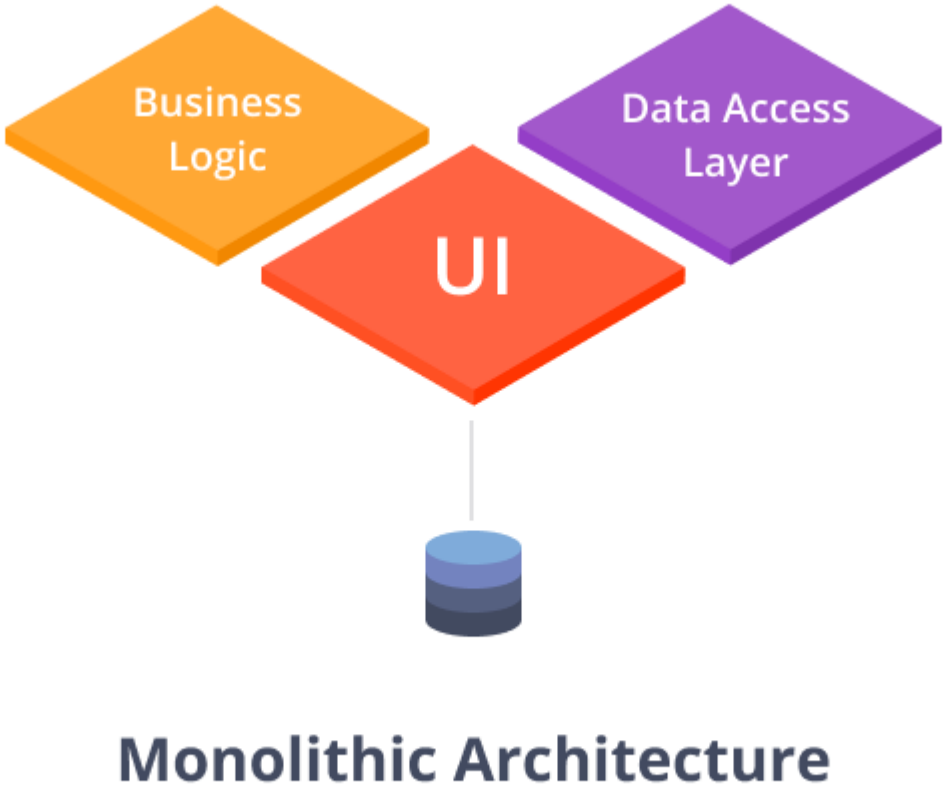
- Difíciles de mantener, difíciles de entender.
- Difíciles de evolucionar, difíciles de entender.
- Difíciles de desarrollar en entornos colaborativos.
- Detienen la operativa total del sistema dado un error en el mismo.
- Mantienen un alto acoplamiento entre sus componentes.
- Componentes difíciles de reutilizar.
- Costoso de escalar horizontal y verticalmente.
- Existe un punto en el que no podrá escalar horizontal ni verticalmente debido a limitaciones técnicas.

i.i ¿Qué son los sistemas monolíticos? (r)

- Desventajas de los sistemas monolíticos (b):

- Al crecer ampliamente el código base de un aplicativo:
  - Mayor sobrecarga para IDEs de desarrollo.
  - Mayor sobre carga de memoria RAM para el entorno de desarrollo.
  - Mayor sobrecarga de contenedores de aplicaciones (contenedores web).
  - Difíciles de desplegar.
  - Dificultad para implementar CI / CD.
- Requiere un fuerte compromiso con el stack tecnológico utilizado para su desarrollo.
- entre otros …

i.i ¿Qué son los sistemas monolíticos? (s)

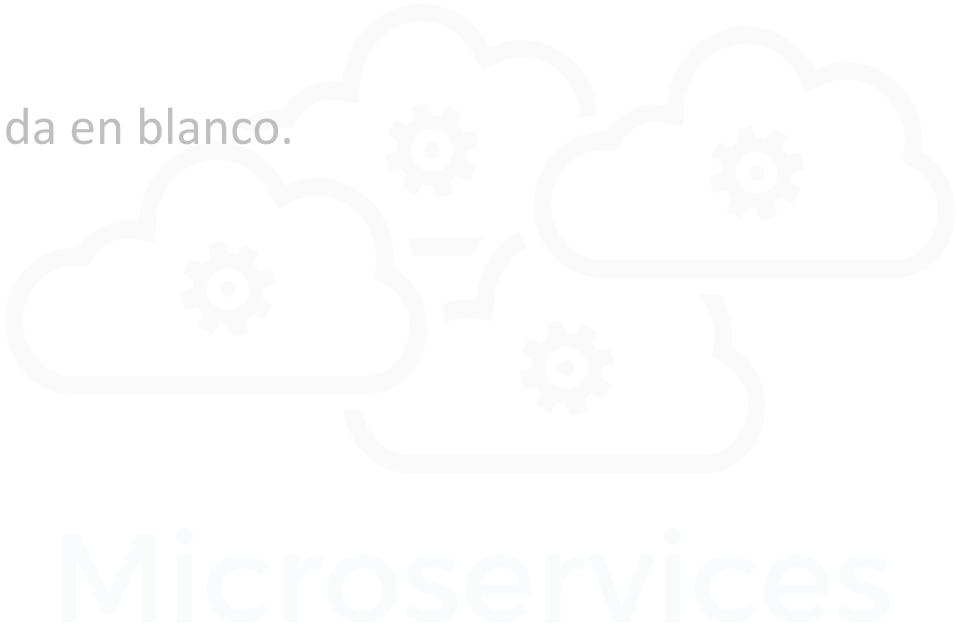


Resumen de la lección

i.i ¿Qué son los sistemas monolíticos?

- Comprendimos las características a nivel de código, despliegue y ejecución de sistemas monolíticos y no-monolíticos.
- Analizamos que un sistema monolítico implica el despliegue de un único compilado, independientemente de si es modular o no la estructura de su código.
- Comprendimos que un sistema monolítico se ejecuta en un único proceso.

Esta página fue intencionalmente dejada en blanco.



i. Arquitectura de sistemas monolíticos

i.i ¿Qué son los sistemas monolíticos?

i.ii Escalabilidad de sistemas monolíticos

i.iii Protocolos de integración

i.ii Escalabilidad de sistemas monolíticos

Objetivos de la lección

i.ii Escalabilidad de sistemas monolíticos

- Revisar los diferentes tipos de escalabilidad de los sistemas monolíticos.
- Analizar las ventajas y desventajas de los diferentes tipos de escalabilidad de los sistemas monolíticos.

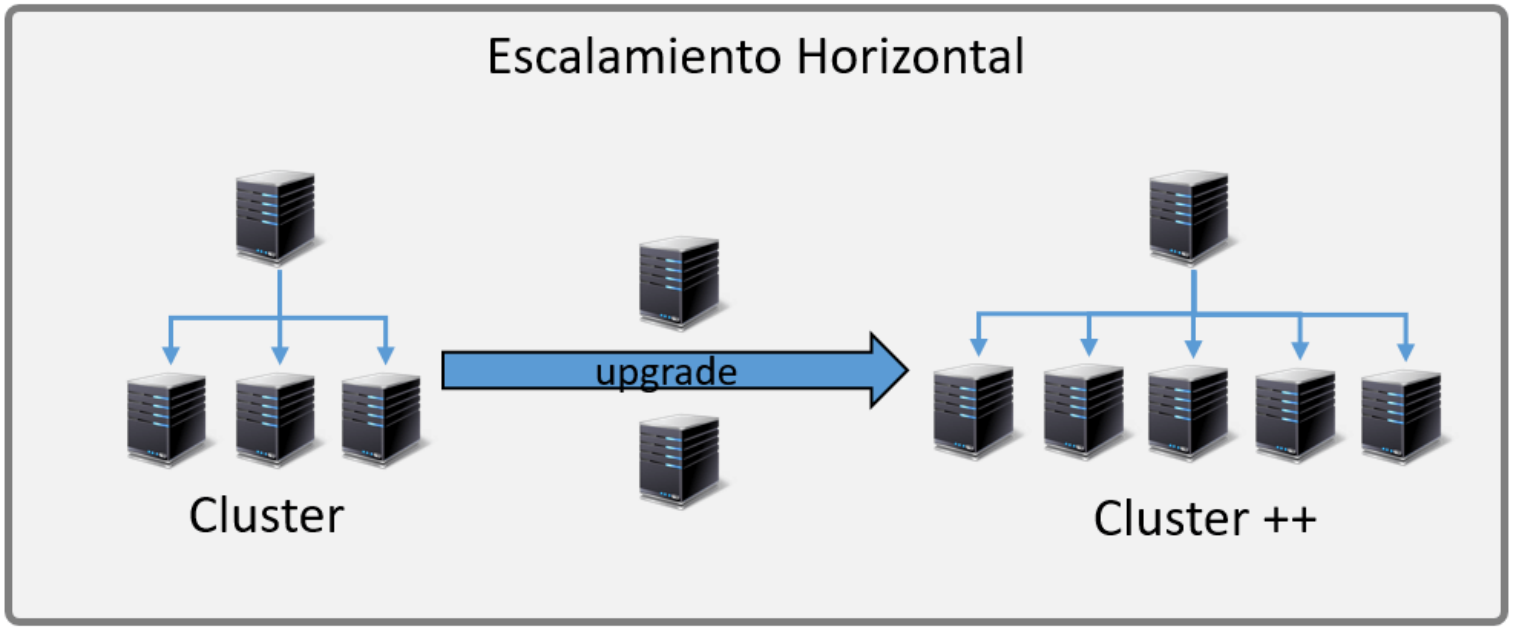
i.ii Escalabilidad de sistemas monolíticos (a)

- Los sistemas monolíticos pueden escalar en dos dimensiones:
  - Escalamiento horizontal.
  - Escalamiento vertical.

- Sin embargo, el sistema monolítico, escala por completo, es decir, todos sus componentes escalan en la misma proporción y no sólo los componentes o servicios que se requiera.

i.ii Escalabilidad de sistemas monolíticos (b)

- Escalamiento horizontal.



i.ii Escalabilidad de sistemas monolíticos (c)

- Ventajas escalamiento horizontal.

- Amplio espectro de escalabilidad, debido a que se podrían agregar tantos servidores como sean necesarios.
- Se puede combinar con el escalamiento vertical.
- Permite la alta disponibilidad del servicio.
- Soporta balanceo de carga.
- Facilidad de escalamiento si se cuenta con el conocimiento requerido.

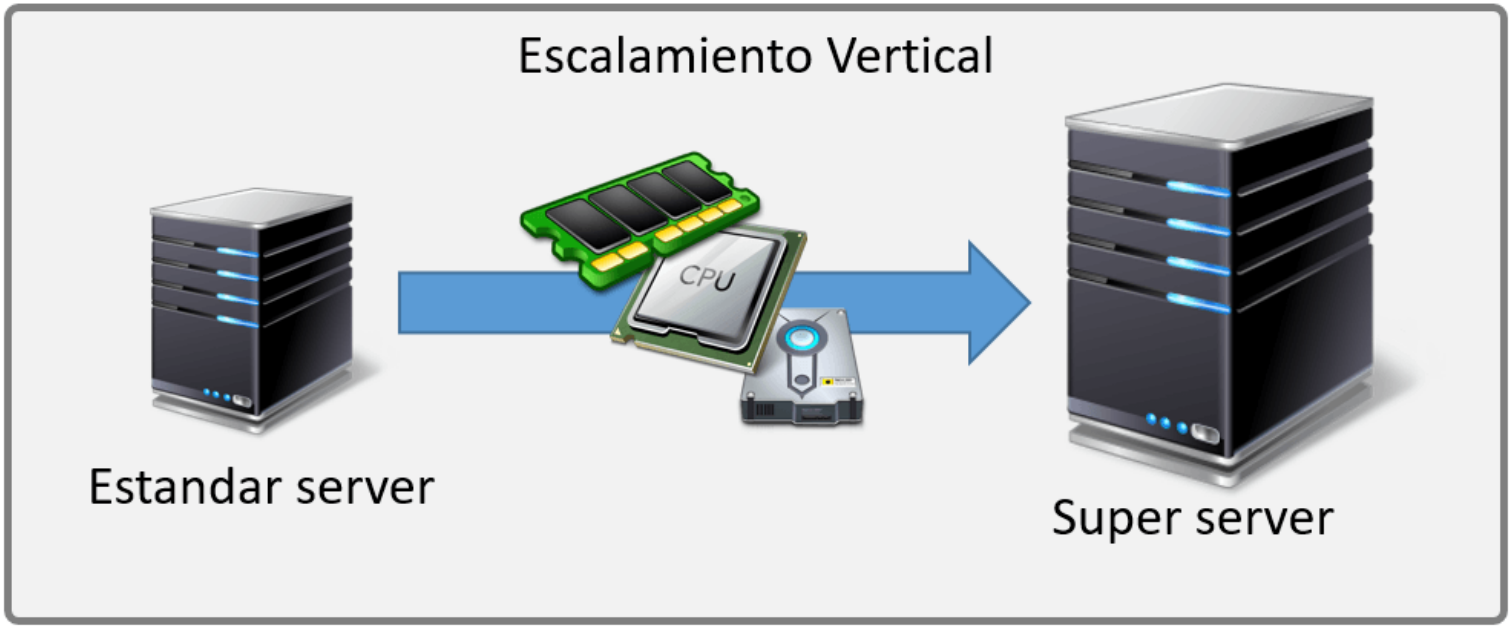
i.ii Escalabilidad de sistemas monolíticos (d)

- Desventajas escalamiento horizontal.

- Costos medianamente aceptables aunque puede crecer exponencialmente.
- Requiere demasiado mantenimiento.
- El exceso en el mantenimiento requerido aumenta los costos operativos.
- Requiere una infraestructura más compleja.
- La aplicación requiere estar diseñada para trabajar en cluster.
- Dificultad de implementación y configuración si no se cuenta con el conocimiento requerido.

i.ii Escalabilidad de sistemas monolíticos (e)

- Escalamiento vertical.



i.ii Escalabilidad de sistemas monolíticos (f)

- Ventajas escalamiento vertical.

- No implica cambios en el sistema a nivel código dado que el escalamiento es a nivel de hardware.
- Fácilidad de implementación.
- Rapidez.

i.ii Escalabilidad de sistemas monolíticos (g)

- Desventajas escalamiento vertical.

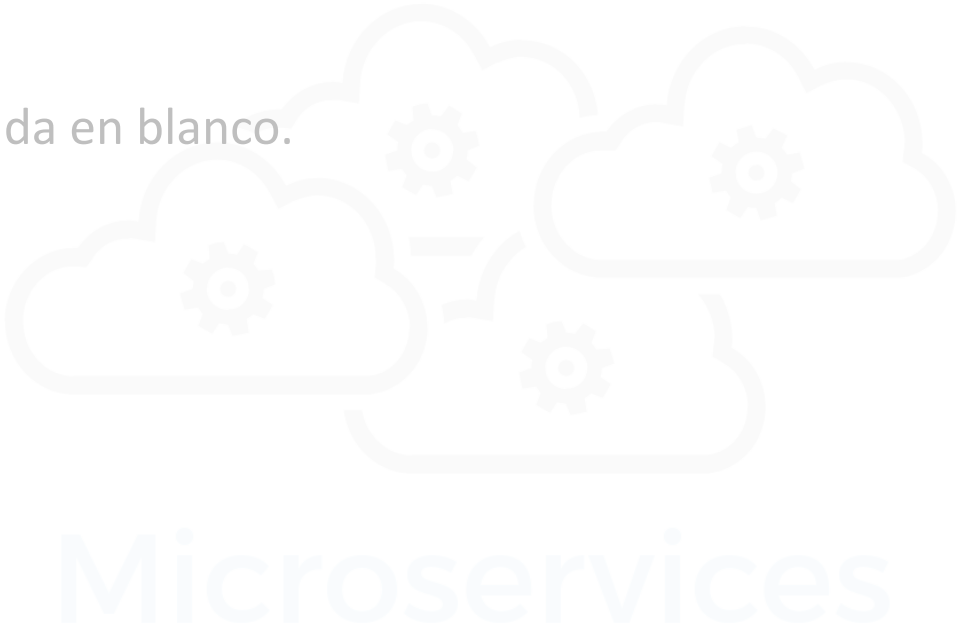
- El crecimiento está limitado por el hardware.
- Fallas a nivel hardware debido a su actualización pueden resultar catastróficas.
- El escalamiento vertical no proporciona alta disponibilidad del servicio.
- Elevados costos para la compra de hardware (disco, RAM y procesadores) más reciente y potente.
- No todos los equipos pueden escalar verticalmente, existe un límite.
- Posiblemente se llegue a un punto donde el hardware ya no puede escalar y será requerido adquirir otro equipo.

Resumen de la lección

i.ii Escalabilidad de sistemas monolíticos

- Comprendimos los diferentes tipos de escalabilidad aplicables a sistemas monolíticos.
- Analizamos las diferencias, ventajas y desventajas del escalamiento horizontal y vertical.
- Comprendimos que para aplicar escalabilidad horizontal, la aplicación debe estar preparada para ello a nivel implementación de código.
- Analizamos que escalar verticalmente un producto de software no proporciona alta disponibilidad del servicio.

Esta página fue intencionalmente dejada en blanco.



i. Arquitectura de sistemas monolíticos

i.i ¿Qué son los sistemas monolíticos?

i.ii Escalabilidad de sistemas monolíticos

i.iii Protocolos de integración

i.iii Protocolos de integración

Objetivos de la lección

i.iii Protocolos de integración

- Comentar brevemente los diferentes protocolos de integración utilizados más populares para sistemas monolíticos y/o distribuidos.

i.iii Protocolos de integración (a)

- CORBA

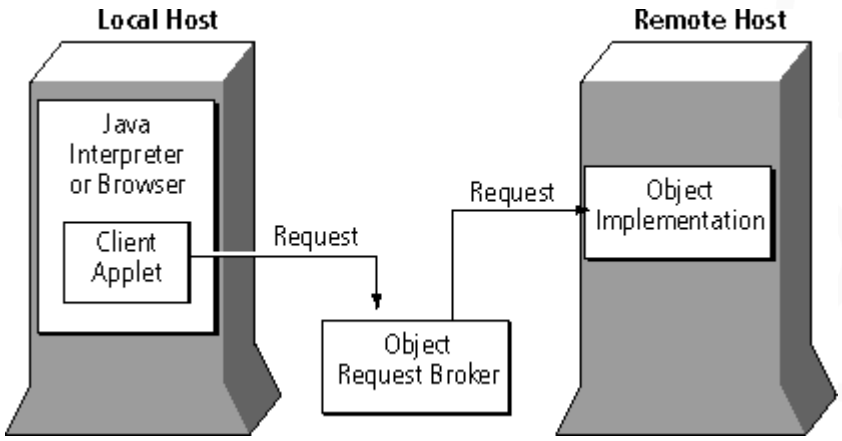
- Common Object Request Broker.

- Es un estandar, definido por la OMG (Object Management Group) que permite a los objetos realizar solicitudes y recibir respuestas entre sistemas distribuidos heterogeneos de forma transparente.

- Permite la interoperabilidad entre sistemas:
  - En diferentes maquinas, nodos o servidores.
  - Heterogeneos en ambientes distribuidos a través de la red.

i.iii Protocolos de integración (b)

- CORBA



i.iii Protocolos de integración (c)

- EAI

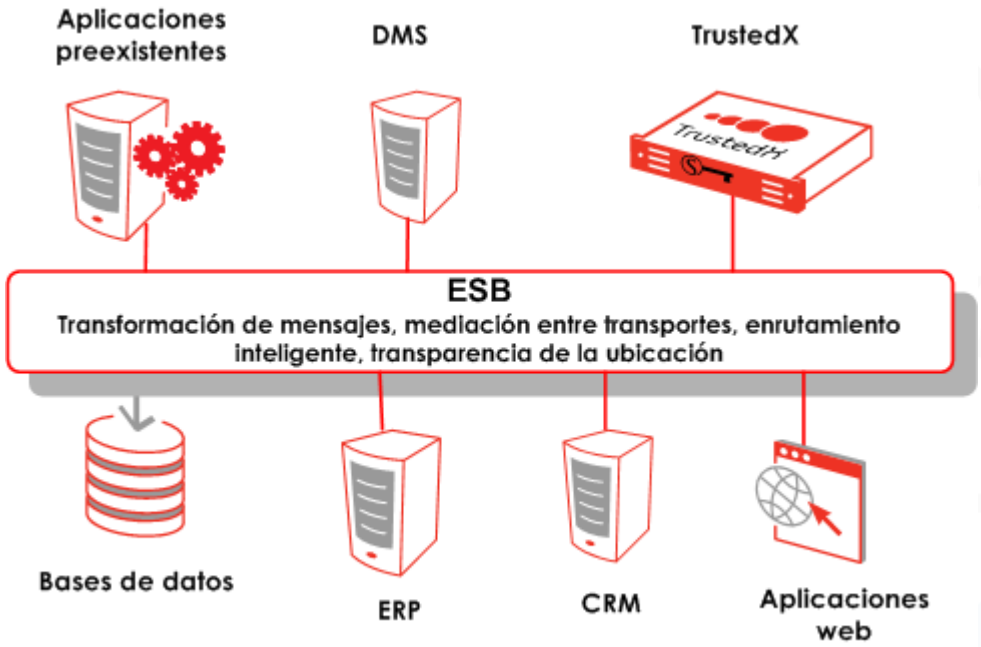
- Enterprise Application Integration.

- Es un framework compuesto por una colección de tecnologías y servicios en forma de middleware que permite la integración de sistemas y aplicaciones heterogeneas en las organizaciones.

- Colección de patrones, para interconectar sistemas de forma eficiente, para evitar los problemas comunes de integración.

i.iii Protocolos de integración (d)

- EAI



i.iii Protocolos de integración (e)

- EAI

i.iii Protocolos de integración (f)

- Web-services

- Los Web-services, o servicios web, son un método de comunicación entre dos componentes de software a través de una red.

- La implementación de un web-service utiliza una colección de protocolos abiertos y estándares, requeridos para intercambiar datos entre aplicaciones o sistemas heterogeneas, las cuales pueden estar escritas en diversos lenguajes de programación.

- La interoperatividad entre sistemas heterogeneos, por ejemplo entre Java y Python o Windows y Linux se debe al uso de estándares abiertos.

i.iii Protocolos de integración (g)

- Web-services

- Los Web-services utilizan XML como formato de intecambio de mensajes de forma estandarizada.

- XML-RPC es el protocolo más sencillo de implementar para el intercambio de datos entre sistemas utilizado para llevar a cabo llamadas a procedimientos remotos, RPCs.

- Los “Remote Procedure Call” son un protocolo de red que permite a un programa a ejecutar código en una máquina remota.

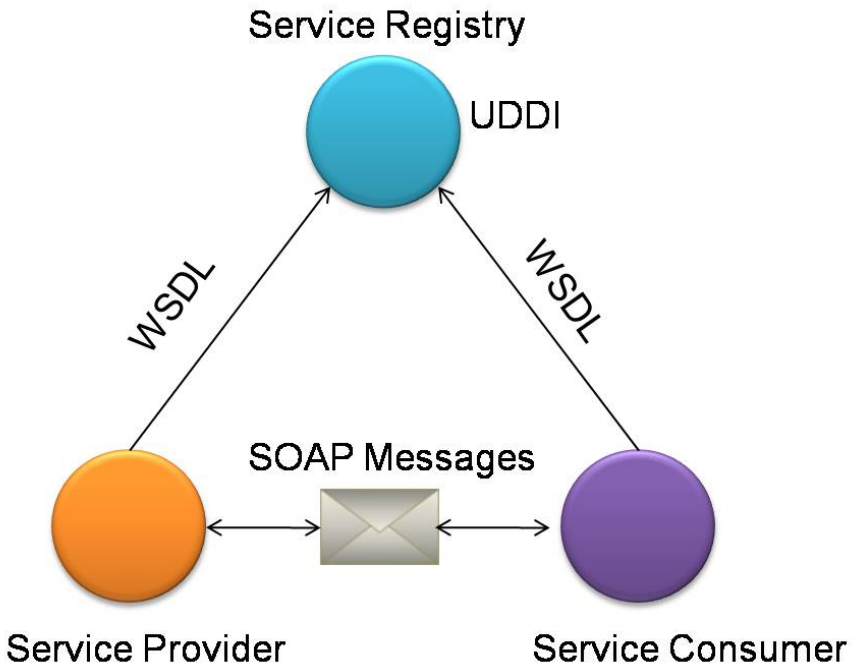
i.iii Protocolos de integración (h)

- Web-services

- Las solicitudes “XML-RPC” son una combinación entre contenido XML y encabezados HTTP; su simpleza hizo que el estándar evolucionara a SOAP, siendo éste el protocolo básico en la implementación de Web-services.

i.iii Protocolos de integración (i)

- Web-services



i.iii Protocolos de integración (j)

- REST

- REpresentational State Transfer, o transferencia de estado representacional, es un conjunto de restricciones con las que se define un estilo de arquitectura de software para la implementación de web-services respetando el protocolo HTTP.

- REST permite la interoperabilidad de sistemas distribuidos mediante hipermedia.

i.iii Protocolos de integración (k)

- REST

- Las restricciones que definen a un sistema basado en REST son (a):
  - **Cliente-servidor**: Esta restricción mantiene al cliente y a el servidor débilmente acoplados.
  - **Stateless**: El servidor no mantiene ninguna información con respecto a las peticiones del cliente, es decir, no guarda estado. No implementa sesiones.

i.iii Protocolos de integración (I)

- REST

- Las restricciones que definen a un sistema basado en REST son (b):
  - **Cacheable**: Debe admitir un sistema de almacenamiento en cache que permita evitar repetir una misma ejecución de una solicitud de un cliente al servidor para recuperar un mismo recurso.
  - **Interfaz uniforme**: Define una interfaz uniforme para administrar cada interacción que se produzca entre el cliente y el servidor. Esta restricción indica que cada recurso que manipula el sistema basado en REST es representado por una única URI, manteniendo un único identificador.

i.iii Protocolos de integración (m)

- REST

- Las restricciones que definen a un sistema basado en REST son (c):
  - **Sistema en capas**: El servidor puede disponer de diversas capas en su implementación lo cual ayuda a mejorar su escalabilidad, rendimiento y seguridad.
  - **Codigo bajo demanda (opcional)**: Esta restricción permite que el cliente pueda solicitar código bajo demanda para que el servidor lo retorne y el cliente pueda ejecutarlo.

i.iii Protocolos de integración (n)

- REST

- REST esta dirigido por recursos los cuales son cualquier tipo de objeto a los que un cliente puede acceder.

- En el diseño de sistemas basados en REST predomina la correcta implementación del protocolo HTTP para interactuar con los servicios que exponen los recursos.

- El mayor grado de madurez en una arquitectura de servicios basado en REST sugiere la implementación de HATEOAS.

i.iii Protocolos de integración (ñ)

- REST

- HATEOAS (Hypermedia As The Engine Of Application State) o Hipermedia como motor del estado de la aplicación, define que cada que un cliente solicita una operación sobre un recurso, el servidor devolverá, en formato de hipervinculo, la navegación asociada a los recursos u operaciones disponibles para el recurso solicitado.

- HATEOAS permite el descubrimiento de servicios mediante los hipervínculos que permiten la navegabilidad de un cliente REST a través de las URIs de recursos relacionados al recurso solicitado.

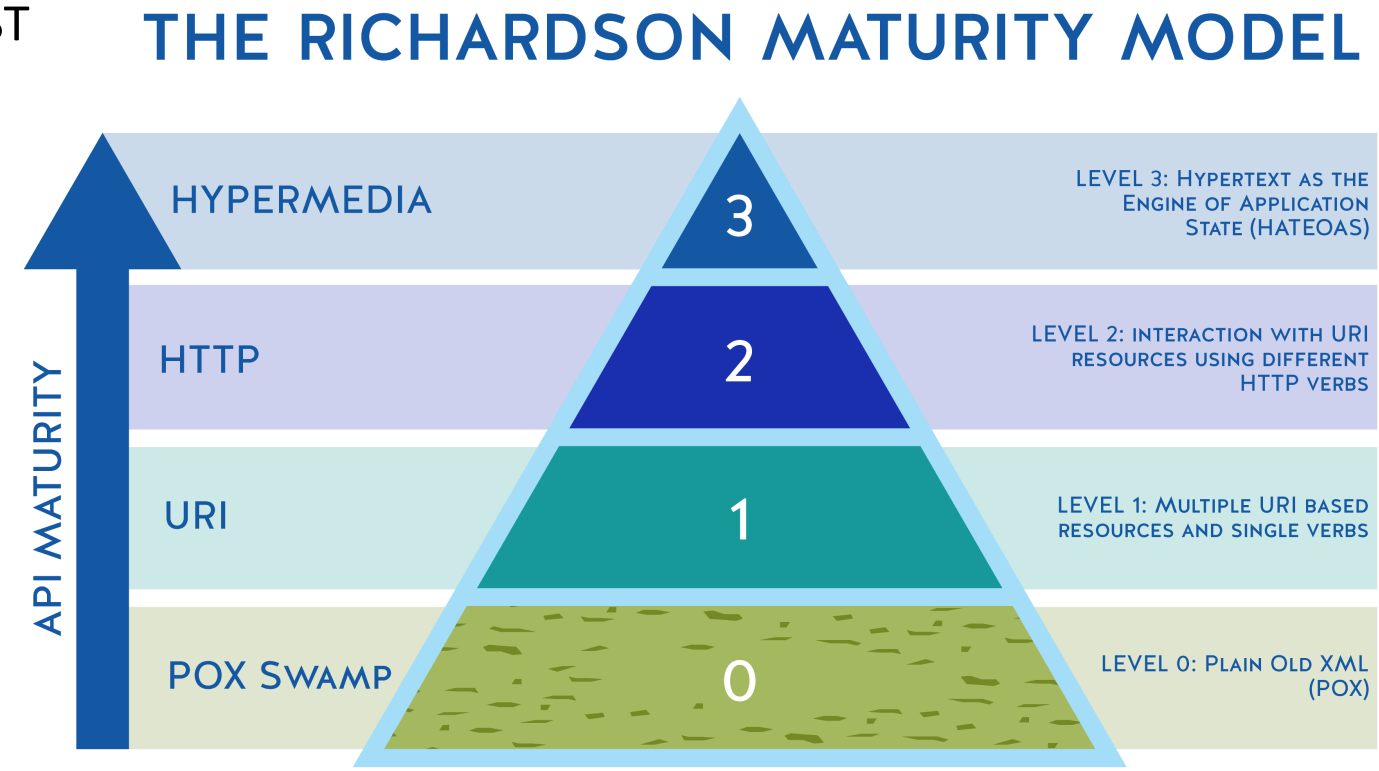
i.iii Protocolos de integración (o)

- REST

- HATEOAS desacopla el cliente del servidor permitiendo que el servidor evolucione de manera independiente y forza a que el cliente sea lo suficientemente inteligente para poder implementar la navegabilidad de recursos y descubrir nuevos recursos de manera eficaz.

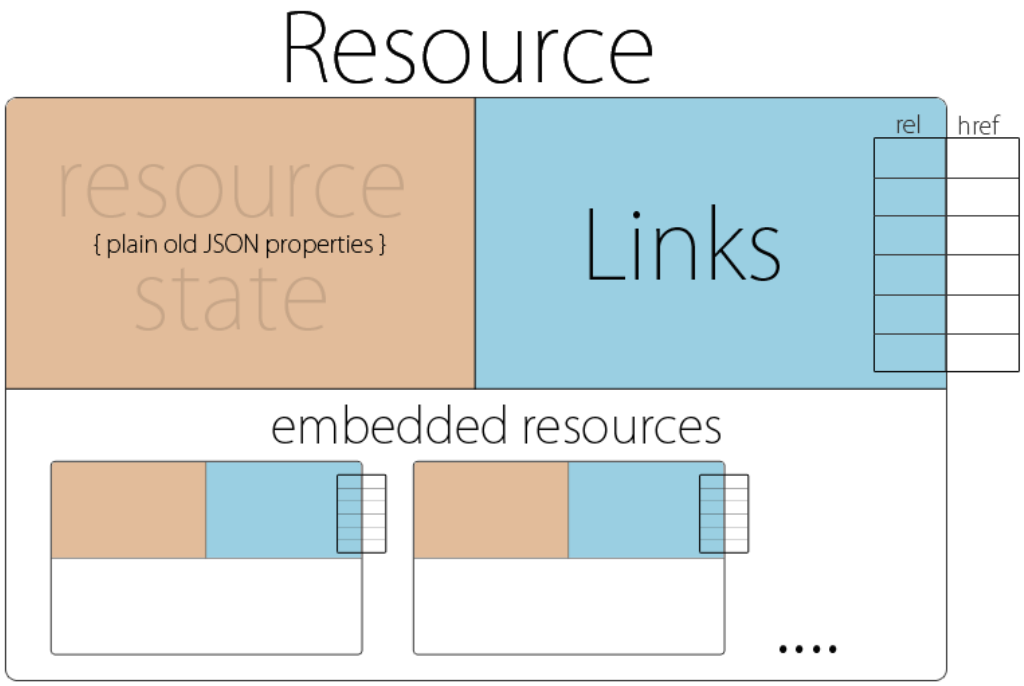
i.iii Protocolos de integración (p)

- REST



i.iii Protocolos de integración (q)

- REST



Resumen de la lección

i.iii Protocolos de integración

- Analizamos a grandes rasgos algunos de los estilos y protocolos que facilitan la interoperabilidad entre sistemas.
- Comprendimos la diferencia entre dichos protocolos y analizamos su aplicabilidad en sistemas distribuidos.
- Conocimos la principal diferencia entre web-services basados en SOAP y servicios REST o (web-services REST).

Esta página fue intencionalmente dejada en blanco.