Servicios Web

Sistemas Distribuidos

Franco Bocalon Luis Guanuco Santiago Nolasco

Especialización en Sistemas Embebidos
Instituto Universitario Aeronáutico

29 de Abril del 2016

Contenidos

Arquitectura Orientada a Servicios (SOA)

Plantea una organización de recurso de forma que una colección desordenada de sistemas distribuidos y aplicaciones complejas se transformen en una red de recursos integrados, simplificados y sumamente flexible.

Arquitectura tradicional	**Arquitectura orientada a servicios**
Forzada a la funcionalidad	Orientado al proceso
Diseñado para durar	Diseñado para el cambio
Desarrollo con largos ciclos de tiempo	Desarrollo iterativo
Fuerte acoplamiento	Bajo acoplamiento
Aplicación especifica	Heterogéneo
Orientado a datos	Orientado al servicio empresarial

Contenidos

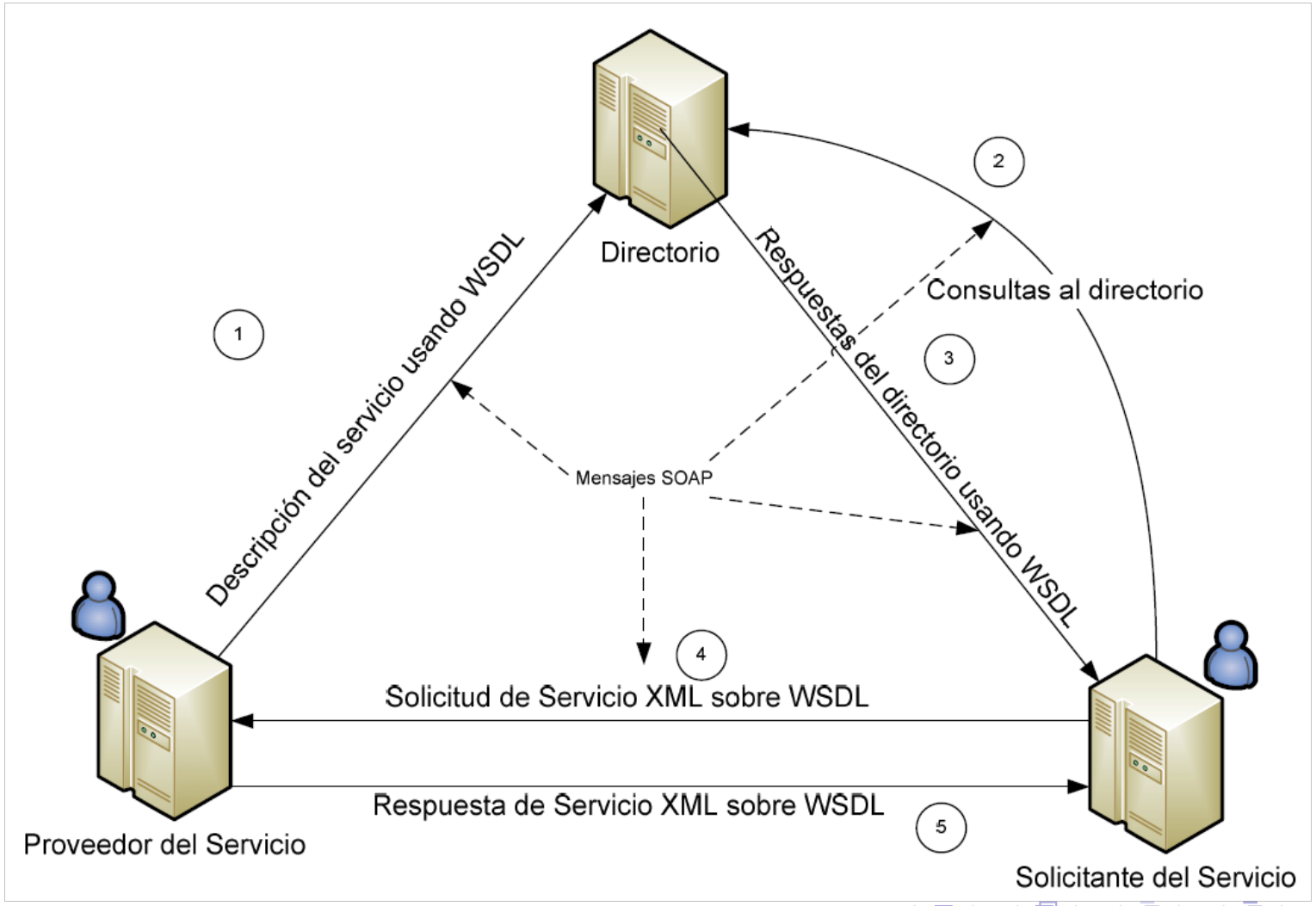
Estándares

XML: Permite definir la gramática de lenguajes específicos para estructurar documentos grandes.

WSDL: Basado en XML, describe la forma de comunicación (protocolos, formatos, etc.) con los servicios ofrecidos.

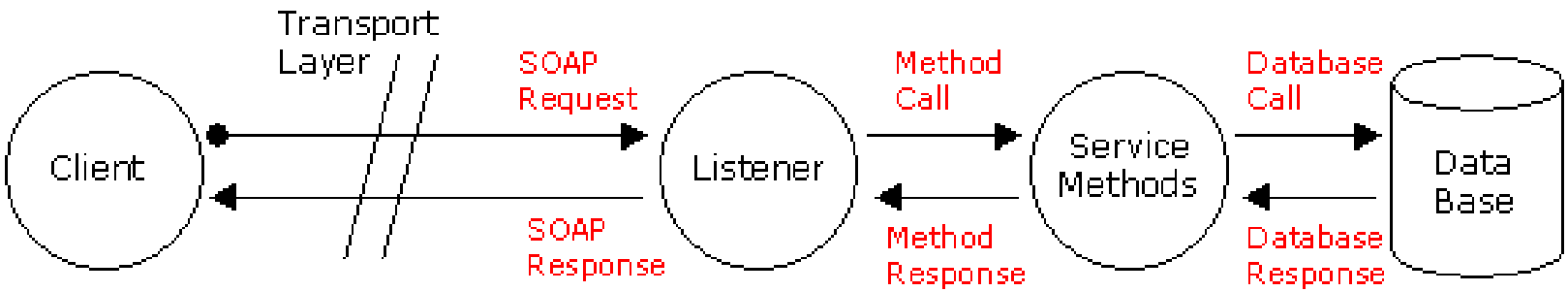
SOAP: Utilizado para el intercambio de datos. La base de este protocolo son: *Extensibilidad*, *Neutralidad* y *Independencia*.

Principal uso



SOAP

El enfoque de SOAP es encapsular la lógica como solicitud de la base de datos para el servicio.



Contenidos

Orígenes

En el 2000 Roy Fielding[1] presenta el estilo de arquitectura REST. Lo define como un estilo híbrido derivado de otras arquitecturas con la *aplicación de restricciones a cumplir*.

Las restricciones planteadas por Fielding buscan mejorar las características deseadas en las arquitecturas Web modernas.

[1] `https://es.wikipedia.org/wiki/Roy_Fielding`

Restricciones de REST

Client-Server: La independencia del cliente hacia el servidor (y recíprocamente) permite la implementación de *múltiples plataformas* desde el lado del cliente, y la *escalabilidad* por el lado del servidor.

Figure 5-2. Client-Server

Restricciones de REST

Stateless: El servidor no debe almacenar ninguna información sobre el contexto del cliente en sus llamadas.

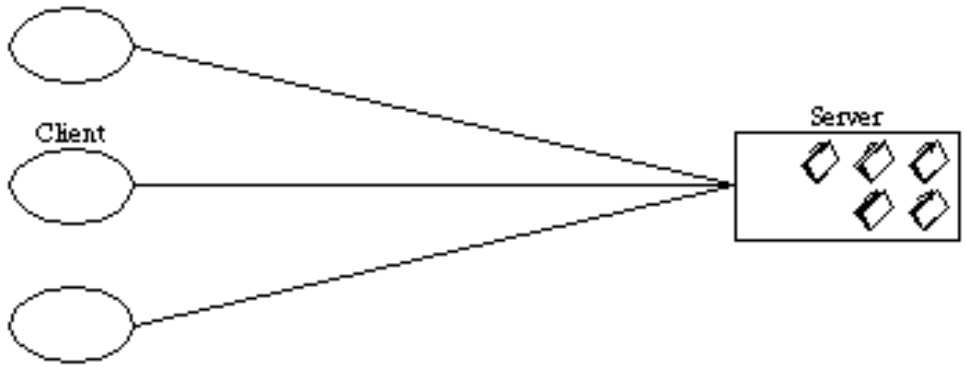


Figure 5-3. Client-Stateless-Server

Restricciones de REST

Cache: Mejorar la eficiencia de la red. Lo importante en esta restricción es proporcionar información suficiente para saber *qué* almacenar para ser reutilizado.

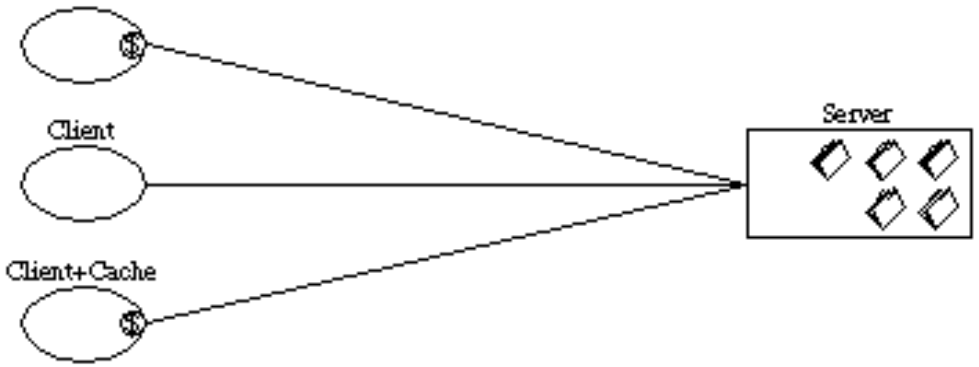


Figure 5-4. Client-Cache-Stateless-Server

Restricciones de REST

Uniform Interface: Una de las características centrales de REST, que lo distingue de otras arquitecturas, es el énfasis en una *interfaz uniforme* entre componentes.

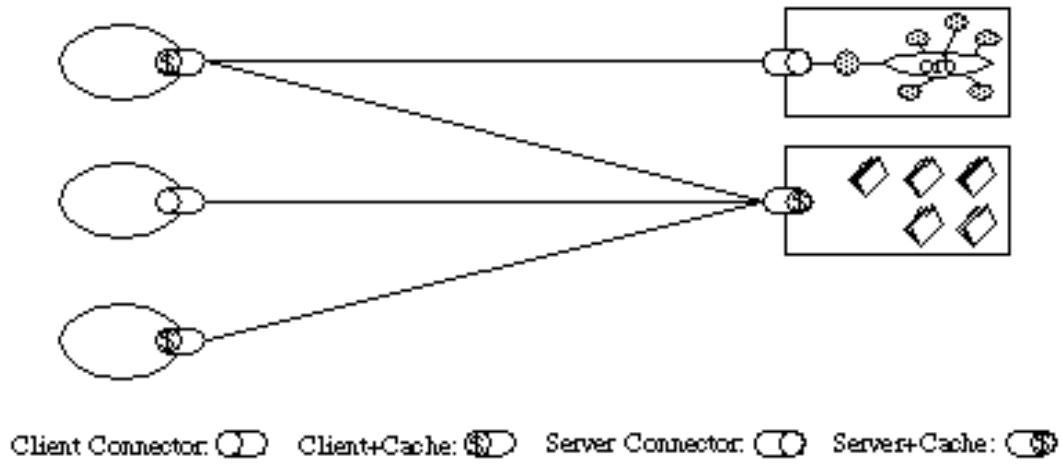


Figure 5-6. Uniform-Client-Cache-Stateless-Server

Un simple ejemplo – SOAP

Se pide obtener datos de un usuario a partir de su identificador. Usando la arquitectura *SOAP*, la petición mediante `HTTP POST` tendría la forma:

```
<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
<soap:body pb="http://www.acme.com/phonebook">
<pb:GetUserDetails>
<pb:UserID>12345</pb:UserID>
</pb:GetUserDetails>
</soap:Body>
</soap:Envelope>
```

La respuesta la almacena el cliente, podría ser un archivo XML, y la respuesta al pedido se encuentra embebido en el mensaje.

Un simple ejemplo – REST

Sí el servidor implementa RESTfull, la consulta anterior se hace con el URL de abajo. Aquí con el comando `HTTP GET` obtenemos la respuesta en bruto.

```
http://www.acme.com/phonebook/UserDetails/12345
```

Aquí el método REST es `UserDetails` en vez de `GetUserDetails`. Esto se debe a la convención del diseño de REST donde se usa *sustantivos* en vez de *verbos* para referirse a un *recurso*.

Principales diferencias

SOAP

Orientado a la manipulación de objetos remotos.

REST

Orientado a la aplicación de operaciones sobre los recursos.

Principales diferencias

SOAP

Orientado a la manipulación de objetos remotos.

Demanda mayor flujo de datos en comparación con REST.

REST

Orientado a la aplicación de operaciones sobre los recursos.

No requiere un mapeo total de los objetos.

Principales diferencias

SOAP	REST
Orientado a la manipulación de objetos remotos.	Orientado a la aplicación de operaciones sobre los recursos.
Demanda mayor flujo de datos en comparación con REST.	No requiere un mapeo total de los objetos.
Madurez y robustez en su estructura.	Flexible y fácil de implementar.

Principales diferencias

SOAP	REST
Orientado a la manipulación de objetos remotos.	Orientado a la aplicación de operaciones sobre los recursos.
Demanda mayor flujo de datos en comparación con REST.	No requiere un mapeo total de los objetos.
Madurez y robustez en su estructura.	Flexible y fácil de implementar.
Implementación de WS* para empresas	Implementado actualmente en todos los WS públicos.

Conclusiones desde nuestra perspectiva

- La *flexibilidad* de REST atrae a los desarrolladores.

Conclusiones desde nuestra perspectiva

- La *flexibilidad* de REST atrae a los desarrolladores.
- La propuesta de Fielding se alinea con la *escalabilidad* de los sistemas actuales.

Conclusiones desde nuestra perspectiva

- La *flexibilidad* de REST atrae a los desarrolladores.
- La propuesta de Fielding se alinea con la *escalabilidad* de los sistemas actuales.
- El *estado del arte* de los WS* apunta a las arquitecturas ROA (REST Oriented Architecture). La industria web mudo hace tiempo todos sus sistemas a *RESTful*.

Veamos un ejemplo

Se implementa un simple WS* con *RESTful*. Vamos a realizarle pedidos...

```
http://localhost:8080/SISTEMASDISTRIBUIDOS/RESTDEMO/...
```