Programación Web

TEMA 6. SERVICIOS WEB

Curso 2019-2020

Último cambio: 18/5/2020

Bibliografía

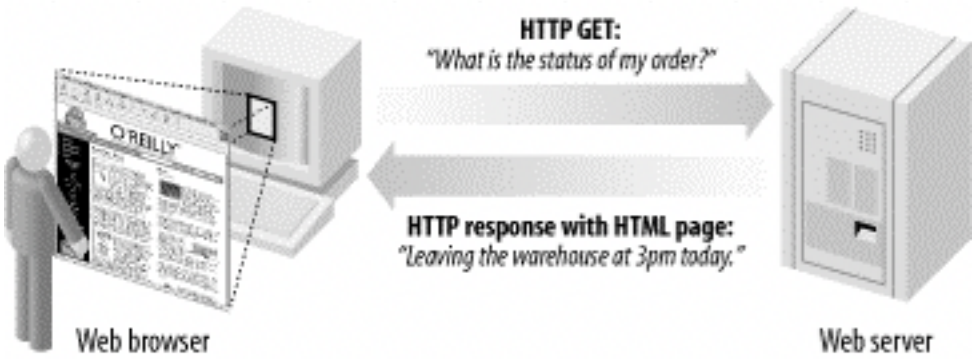
- E. Cerami, «Web Services Essentials», O'Reilly, 2002
- D. Chappell y T. Jewell, «Java Web Services», O'Reilly, 2002
- L.J. Mitchel, "PHP Web Services", O'Reilly, 2013.
- E. Newcomer, «Understanding Web Services», Addison-Wesley Professional, 2002
- W3C, «Web Services Architecture», http://www.w3.org/TR/ws-arch

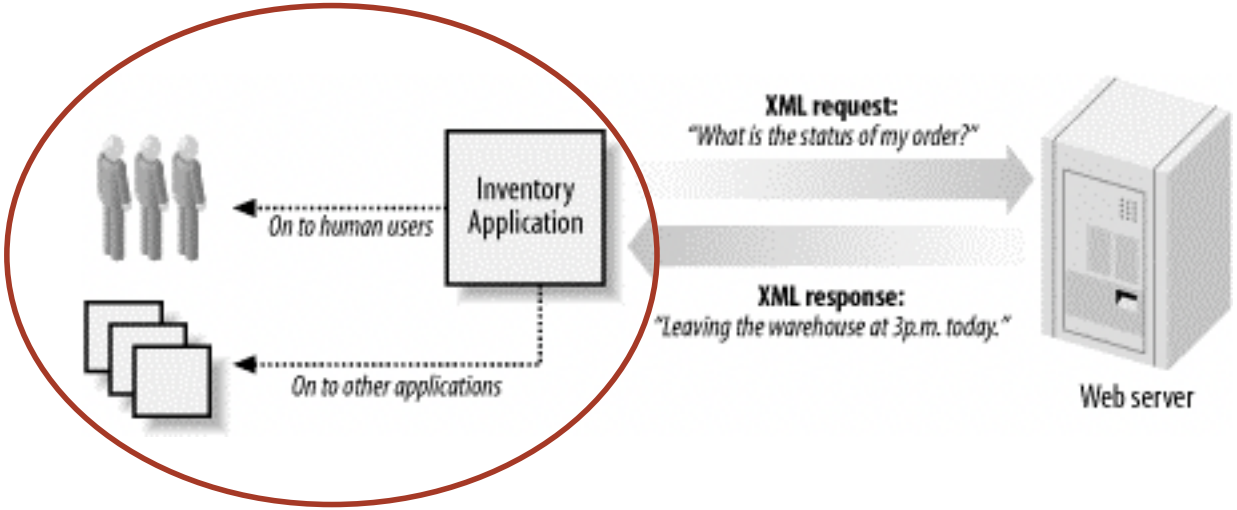
Contenido

- Definición
- Conceptos básicos
- Arquitectura

Ejemplos de código disponibles en: http://betatun.ugr.es/~jmbs/WebServices

DEFINICIONES


HTTP GET:
"What is the status of my order?"
HTTP response with HTML page:
"Leaving the warehouse at 3pm today."
Web browser
Web server


Inventory Application
On to human users
On to other applications
XML request:
"What is the status of my order?"
XML response:
"Leaving the warehouse at 3p.m. today."
Web server

Servicio web

- Tecnología que utiliza un conjunto de protocolos y estándares para intercambio de datos entre aplicaciones.
- Cualquier servicio ofrecido a través de Internet, mediante de un sistema de mensajes XML
- Sistemas de mensajes XML:
  - XML-RPC
  - SOAP
  - HTTP GET/POST

Definición del W3C

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 WSDL), a través de la que es posible interactuar con el mismo mediante el intercambio de mensajes SOAP, típicamente transmitidos usando serialización XML sobre HTP conjuntamente con otros estándares web.

Eric Newcomer's definition

Web services provide a ***layer of abstraction*** above existing software systems, such as application servers, CORBA, .NET servers, messaging, and packaged applications. Web services work at a level of abstraction similar to the Internet and *are* ***capable of bridging*** any operating system, hardware platform, or programming language, just as the Web is.

Web services are Extensible Markup Language (XML) **applications mapped** to programs, objects, or databases or to comprehensive business functions.

A program sends a request to a Web service (message as an XML document) across the network, and, optionally, receives a reply, also in the form of an XML document.

Interfaces

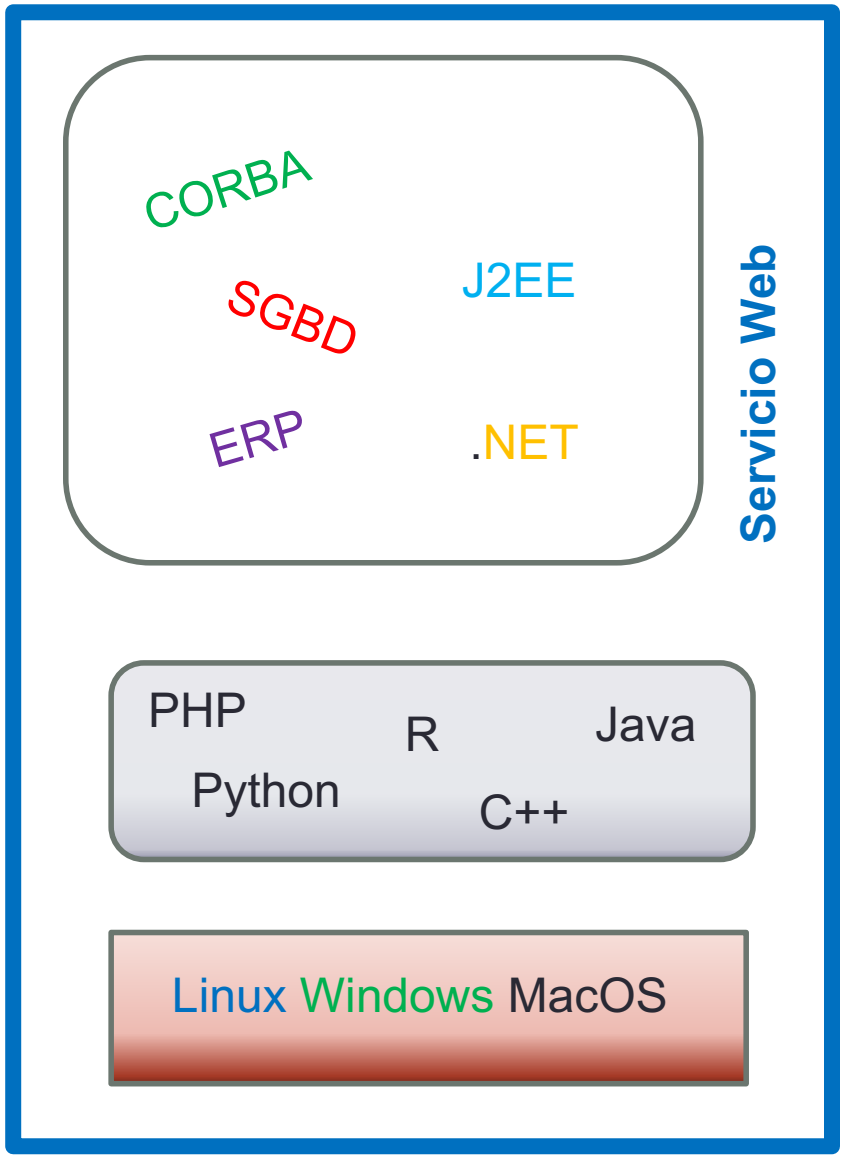
Recibe un mensaje en XML de la red

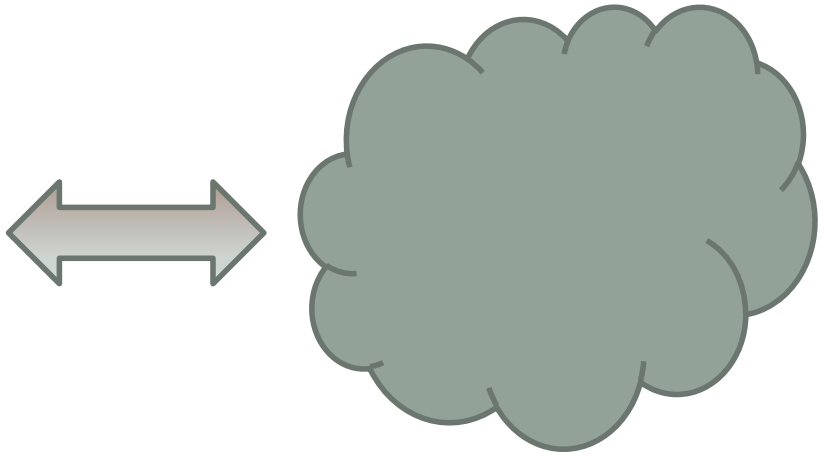
Transforma los datos XML en un formato procesable por el back-end

(*Opcional*) Devuelve un mensaje

Casos de uso

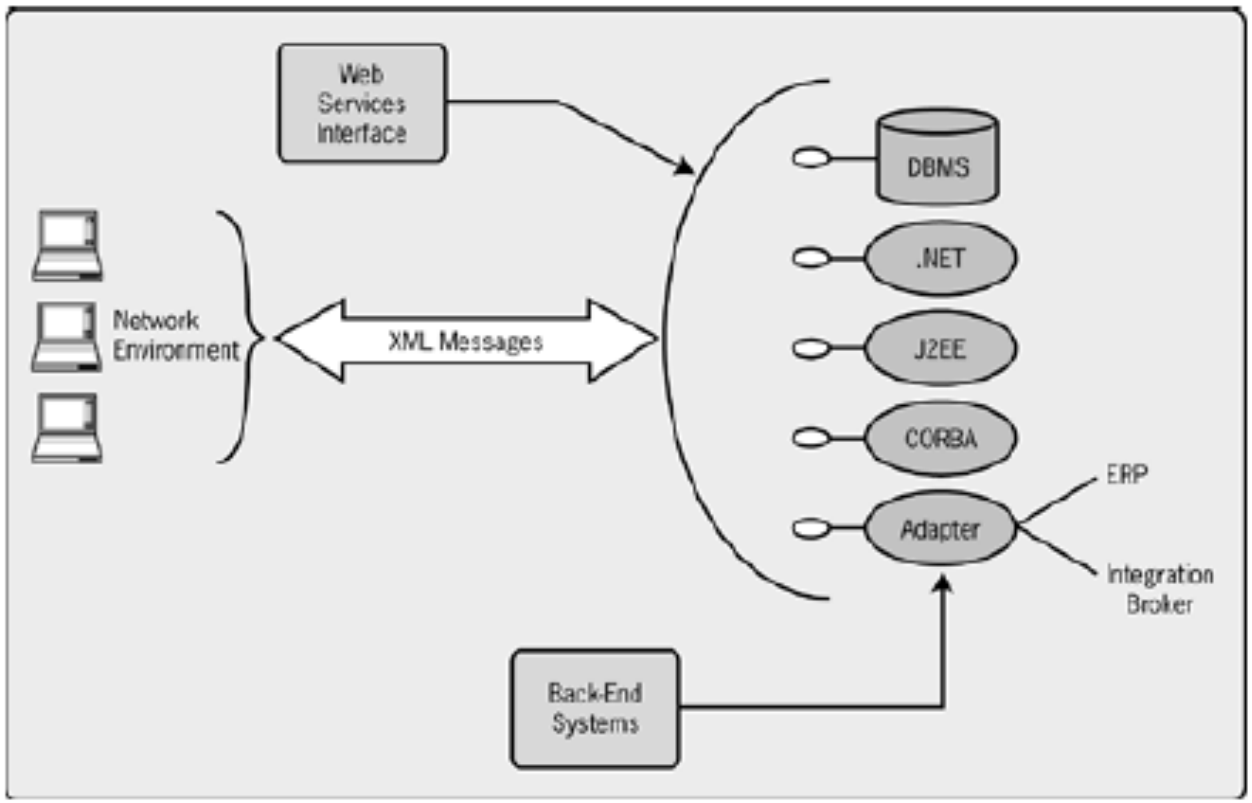
- Verificación de tarjetas de crédito
- Robots de compra
- Conversión de divisas
- Traducción de textos
- Gestión de carteras





Construcción de aplicaciones web

Esquema


Web
Services
Interface
DBMS
.NET
J2EE
CORBA
ERP
Adapter
Integration
Broker
Network
Environment
XML Messages
Back-End
Systems

Propiedades

- Son componentes de aplicaciones
- Se comunican a través de protocolos abiertos
- Son autocontenidos y autodescriptivos
- Descubribles a través de UDDI
- Utilizables por otras aplicaciones

Ejemplo: búsqueda de información

Mecanismo habitual: insertar datos en HTML

http://www.google.com/search?q=Skate+boots&btnG=Google+Search

Búsqueda de “Skate boots” en el motor de búsqueda de Google:

- search: servicio requerido
- “Skate+boots”: cadena buscada, enviada en HTML

Alternativa XML

```
<SOAP-ENV:Body>
  <s:SearchRequest
    xmlns:s="www.xmlbus.com/SearchService">
      <p1>Skate</p1>
      <p2>boots</p2>
</s:SearchRequest>
</SOAP-ENV:Body>
```

Ventajas del envío en XML

- Mejor control de los tipos de datos y la estructura de la información
- Más flexibilidad
- Más extensible
- …

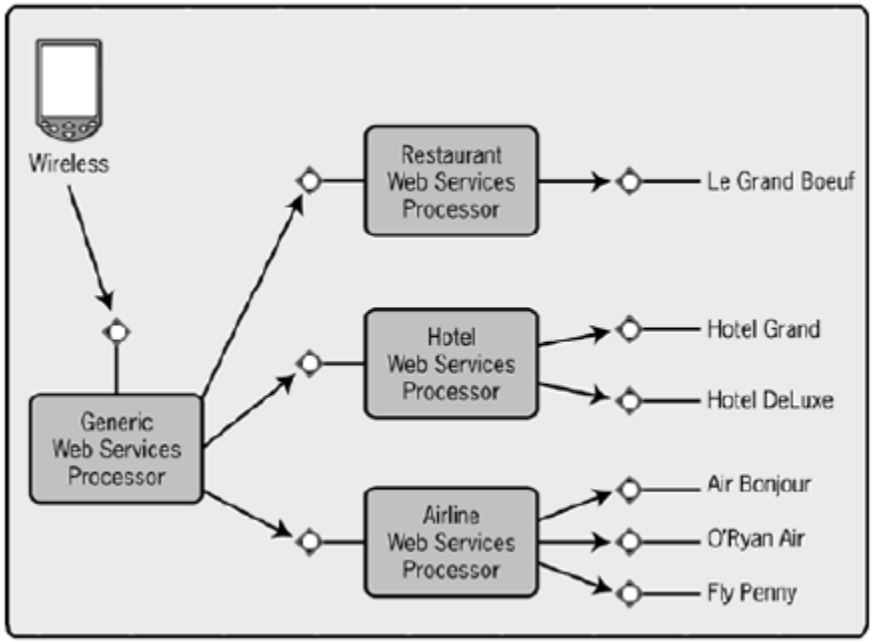
Desventajas

- Para transacciones, el desarrollo es mucho menos maduro que el de estándares de computación distribuida como CORBA
- El rendimiento es muy bajo, comparado con otros modelos de computación distribuida. RMI, CORBA
  - Entre los objetivos de XML no están la concisión, ni la eficiencia en el procesamiento

Ejemplos de aplicaciones

- Buscar y comprar bienes y servicios al mejor precio
  - Comprar billetes de avión; seguros de coche; hoteles, …
- Coordinar billetes de viajes y eventos
- Gestión de procesos de negocio: consecución, facturación y envío
- …

Coordinación de vuelo, alojamiento y restaurante



CONCEPTOS BÁSICOS

Agentes y servicios

El **agente** es el componente software que gestiona la comunicación (mensajes).

El **servicio** es el recurso, caracterizado por la funcionalidad que se provee

Ej.: el agente se puede implementar con distintos lenguajes de programación, mientras que el servicio es el mismo

Clientes y proveedores

- El proveedor es la entidad que ofrece un agente que implementa un servicio particular

- El cliente es la entidad que desea utilizar el servicio de un proveedor. Implementa un agente que dialoga con el agente del proveedor

Pila de protocolos

Interacción para un servicio web

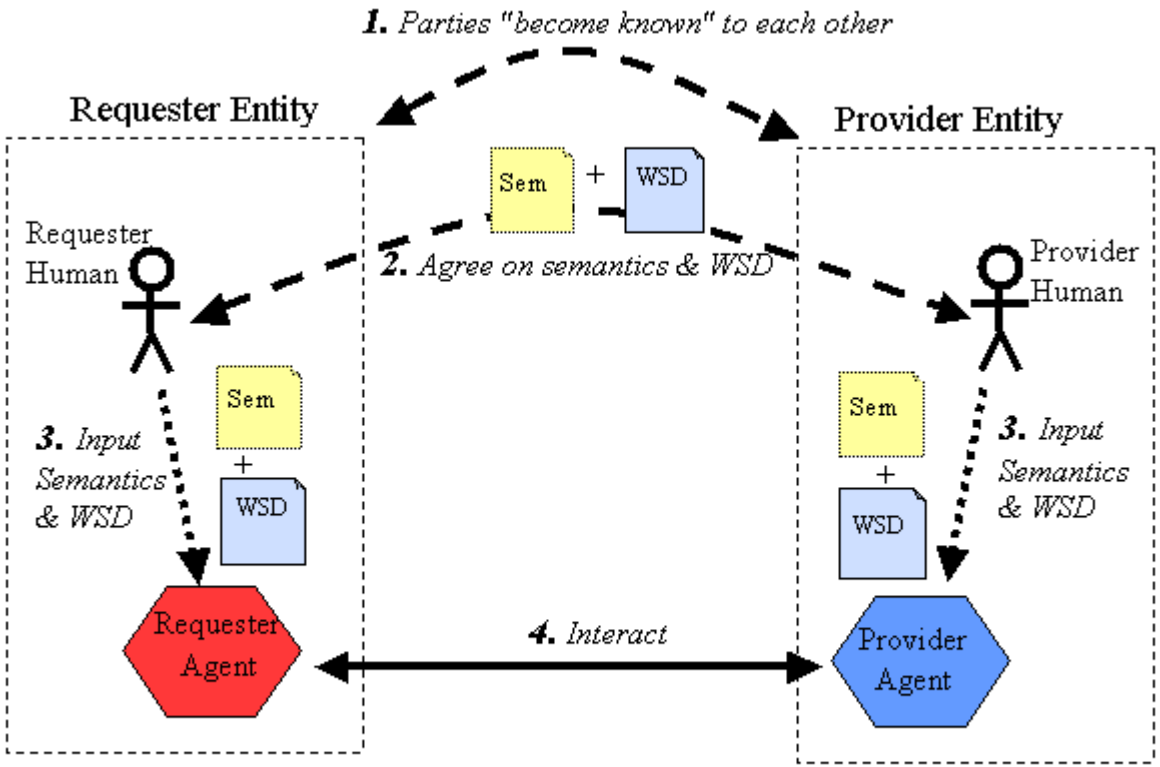


Figure 1-1. The General Process of Engaging a Web Service

ARQUITECTURA

Habilidades para interacción de programas a través de la web

Localizarse y descubrir información que les permita interaccionar

Averiguar los patrones de interacción: solicitud-respuesta o más elaborado

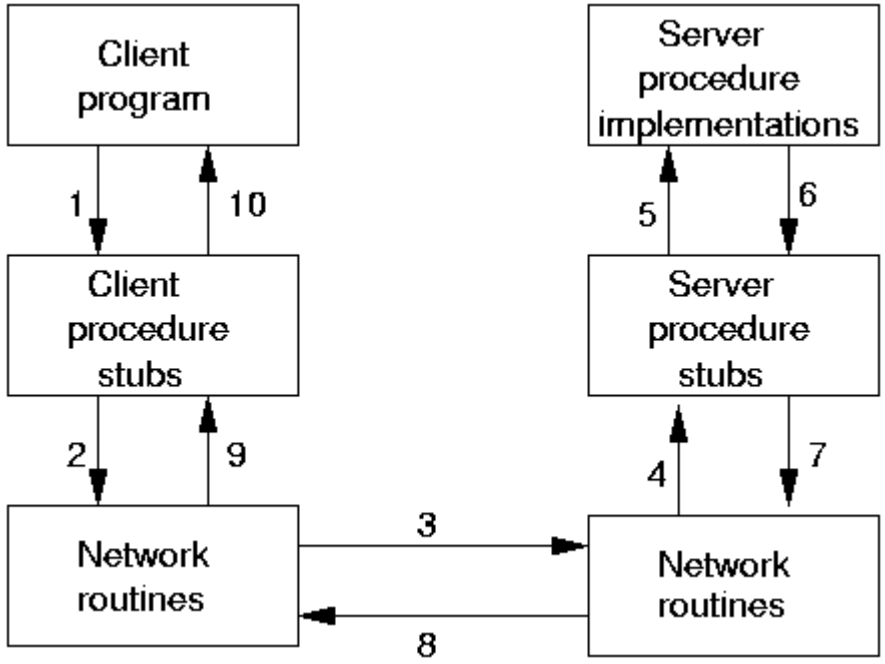
Negociar condiciones: calidad del servicio, fiabilidad en la comunicaciíon, transacciones

Estándares relacionados

- UDDI
- WSDL
- XML-RPC; SOAP
- HTTP

Remote Procedure Call (RPC)

- RPC implementa un proceso de llamada a un procedimiento, pero en una ubicación remota (no local)
- Existe una complejidad adicional en el proceso de comunicación entre el invocador y el llamado
- Protocolos existentes para computación distribuida



XML en servicios web

- XML se creó para superar las limitaciones de HTML. Especialmente para propocionar mejor soporte en la creación y gestión de contenido dinámico
- Con XML puedes crear cualquier elemento que asocie significado a los datos: mucha flexibilidad
- Los esquemas restringen dicha flexibilidad

XML-RPC

Extensible Markup Language-Remote Procedure Call

- Protocolo que usa XML para ejecutar RPC
- Solicitudes codificadas en XML y enviadas vía POST
- Las respuestas XML se incrustan en el cuerpo de la respuesta HTTP
- Es independiente de plataforma

Similar a las cabeceras de funciones

Ejemplo: weather service

getWeather-request.xml

```
<methodCall>
  <methodName>weather.getWeather </methodName>
     <params>
        <param><value>10016</value></param>
      </params>
</methodCall>
```

getWeather-reply.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<methodResponse>
   <params>
       <param>
    <value><int>65</int></value>
    </param>
  </params>
</methodResponse>
```

SOAP

Simple Object Access Protocol

- Protocolo de comunicaciones que define un formato de *serialización* para la transmisión de documentos XML sobre una red y para representar interacciones RPC
- Es más complejo que XML-RPC.
- Usa espacios de nombres y esquemas de XML.

Ejemplo: petición

getWeather-SOAP-request.xml

```
<?xml version='1.0' encoding='UTF-8'?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://www.w3.org/2001/09/soap-
envelope/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <SOAP-ENV:Body>
    <ns1:getWeather
       xmlns:ns1="urn:examples:weatherservice"
        SOAP-ENV:encodingStyle=
            "http://www.w3.org/2001/09/soap-encoding/">
        <zipcode xsi:type="xsd:string">10016</zipcode>
     </ns1:getWeather>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Ejemplo: respuesta

getWeather-SOAP-reply.xml

```
<?xml version='1.0'  encoding='Utf-8'>
<SOAP-ENV:Envelope
   xmlns:SOAP-ENV=
          "http://www.w3.org/2001/09/soap-envelope/"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema">
   <SOAP-ENV:Body>
      <ns1:getWeatherResponse
         xmlns:ns1="urn:examples:weatherservice"
         SOAP-ENV:encodingStyle=
            "http://www.w3.org/2001/09/soap-encoding/">
         <return xsi:type="xsd:int">65</return>
      </ns1:getWeatherResponse>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Descripción de los servicios

- El protocolo de intercambio de mensajes para prestación de servicios web se describe en el lenguaje **WSDL**
- La descripción del protocolo (WSD) incluye formatos de mensajes, tipos de datos, protocolos de transporte y *serialización*
- También indicaciones en las que localizar el servicio

WSDL

Web Services Description Language

Lenguaje, basado en XML, para definir

- Tipos de datos incluidos en los mensajes
- Operaciones a realizar sobre los mensajes
- La traducción de mensajes en redes de transporte (interconexión entre servicios)

Ejemplo: servicio weather (I)

WeatherService.wsdl

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="WeatherService"

targetNamespace="http://www.ecerami.com/wsdl/WeatherSe
rvice.wsdl"
    xmlns="http://schemas.xmlsoap.org/wsdl/"
    xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"

xmlns:tns="http://www.ecerami.com/wsdl/WeatherService.
wsdl"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">

<message name="getWeatherRequest">
    <part name="zipcode" type="xsd:string"/>
</message>
```

Servicio weather (II)

```
<message name="getWeatherResponse">
<part name="temperature" type="xsd:int"/>
</message>

<portType name="Weather_PortType">
  <operation name="getWeather">
     <input message="tns:getWeatherRequest"/>
     <output message="tns:getWeatherResponse"/>
   </operation>
</portType>
```

Servicio weather (III)

```
<service name="Weather_Service">
<documentation>WSDL File for Weather
Service</documentation>
<port binding="tns:Weather_Binding"
name="Weather_Port">
<soap:address
location="http://localhost:8080/soap/servl
et/rpcrouter"/>
</port>
</service>
</definitions>
```

WSDL en acción

- Con WSDL un cliente puede localizar un servicio web e invocar funciones disponibles automáticamente.
- **Automatizable** con herramientas adecuadas

```
java clients.DynamicInvoker
http://localhost:8080/wsdl/WeatherService.wsdl getWeather
10016
```

```
Reading WSDL document from
'http://localhost:8080/wsdl/WeatherService.wsdl'
Preparing WSIF dynamic invocation
Executing operation getWeather
Result:
temperature=65
 Done!
```

UDDI

Universal Description, Discover, and Integration

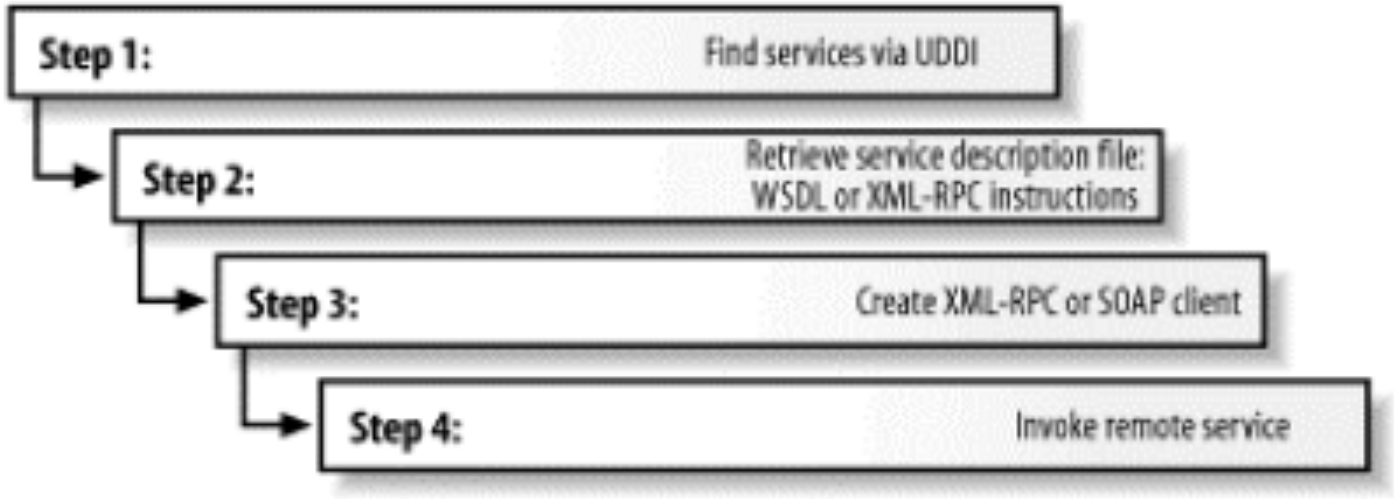
Mecanismos de registro y descubrimiento de servicios web utilizados para almacenar y categorizar información de negocio y para recuperar direcciones de acceso a servicios web

Directorio distribuido de negocios y servicios web.

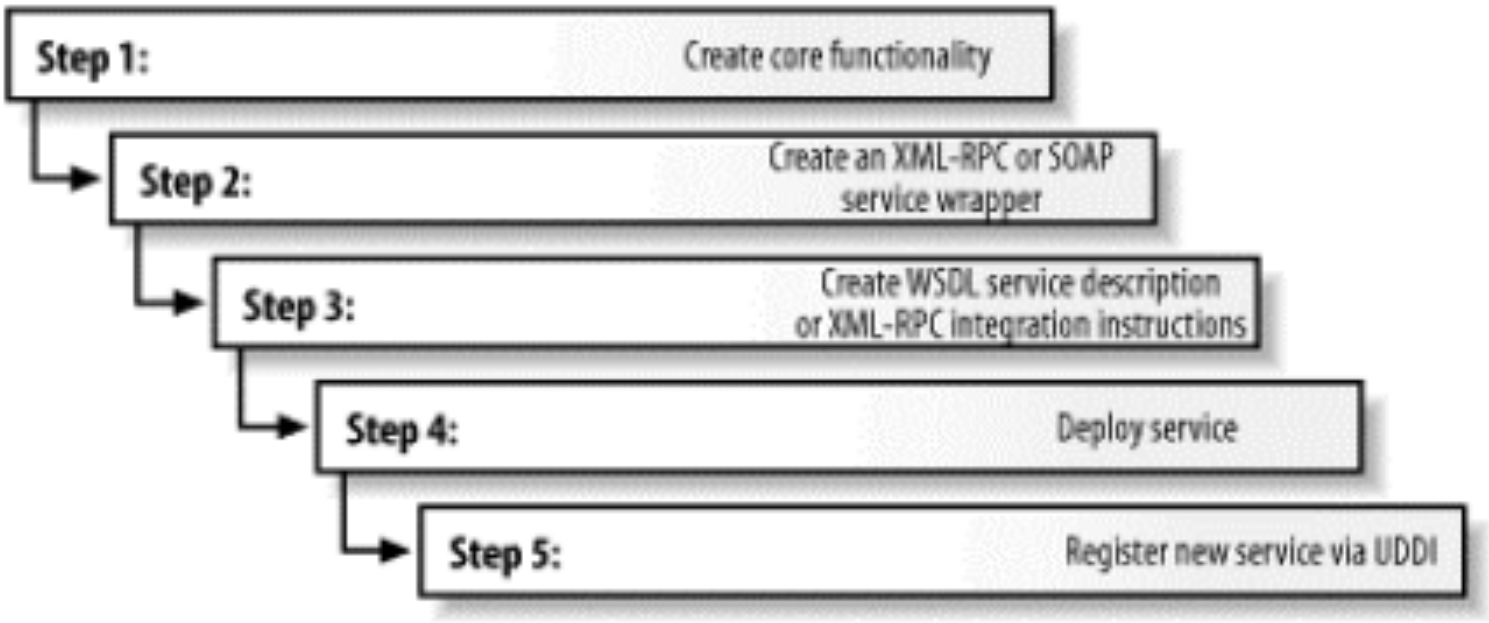
API para buscar datos y publicar nuevos datos

Implementación de la especificación UDDI

Perspectiva del cliente



Perspectiva del proveedor



Beneficios de los servicios web

- Exponer la funcionalidad a la red
- Conectar distintas aplicaciones (interoperatividad)
- Protocolo estandarizado
- Bajo coste de comunicación

ARQUITECTURAS PARA APLICACIONES WEB

Algunos tipos de arquitecturas

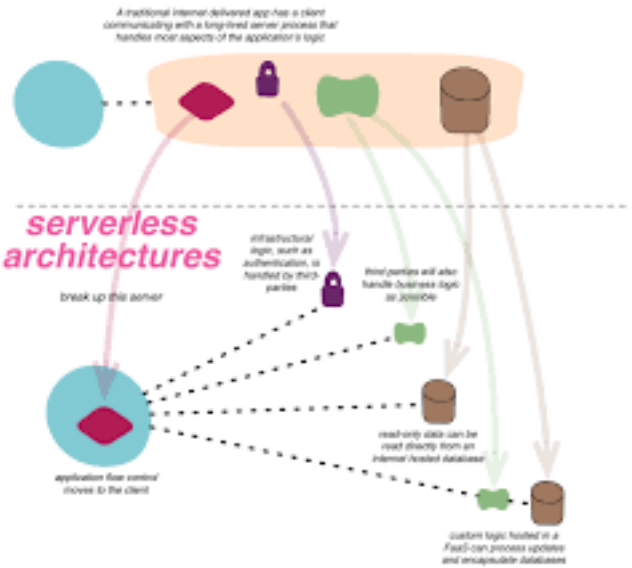
- Arquitectura en tres capas
- Service Oriented Architecture (SOA)
  - REST
- Microservicios
- Serverless:
  - Function as a Service (FaaS)

Microservicios

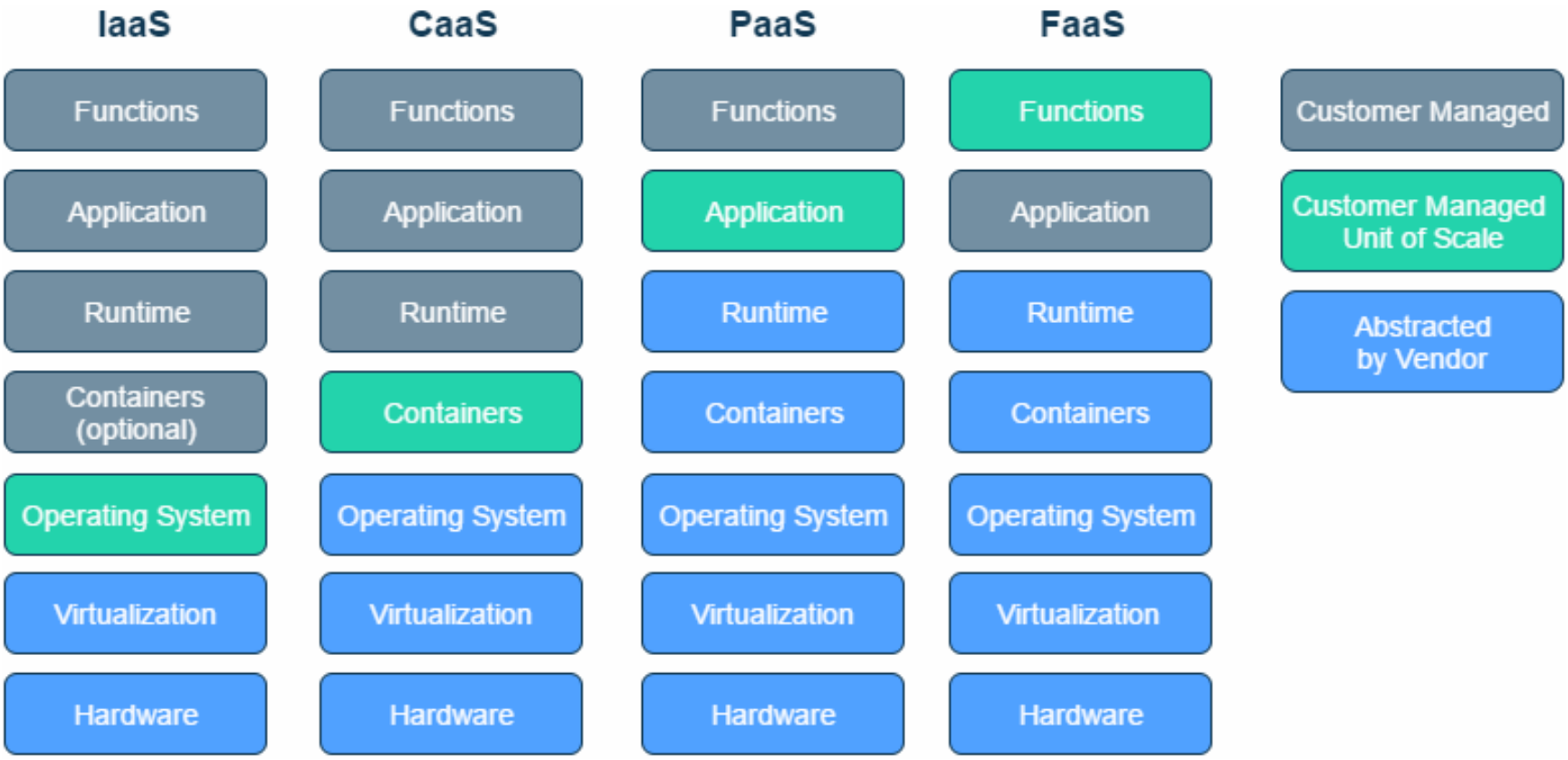
Estilo de desarrollo de aplicaciones software a partir de la composición de un conjunto de servicios pequeños, cada uno ejecutándose de forma autónoma y comunicándose entre sí.

- Despliegue automatizado
- Control descentralizado de módulos y datos

Serverless



Modelo de computación proveedor de IaaS **permite ejecutar durante un periodo de tiempo determinado porciones de código denominadas "funciones"** sin necesidad de hacernos cargo de la infraestructura subyacente que se provisiona para dar el servicio


IaaS
Functions
Application
Runtime
Containers (optional)
Operating System
Virtualization
Hardware
CaaS
Functions
Application
Runtime
Containers
Operating System
Virtualization
Hardware
PaaS
Functions
Application
Runtime
Containers
Operating System
Virtualization
Hardware
FaaS
Functions
Application
Runtime
Containers
Operating System
Virtualization
Hardware
Customer Managed
Customer Managed Unit of Scale
Abstracted by Vendor

ACTIVIDADES COMPLEMENTARIAS

Para profundizar

1. Estudiar con más profundidad XML:
   1. E.T. Ray, “Learning XML”, O’Reilly,
   2. E.R. Harold, W.S. Means, “XML in a Nutshell: A Desktop Quick Reference”, O’Reilly,
2. Estudiar servicios *serverless* y FaaS de proveedores cloud:
   1. Amazon Lambda
   2. Azure Functions
   3. Google Cloud Functions