¿Qué es un **servicio WEB**?

Un *webservice* (o servicio web) es una forma de integrar aplicaciones web.

Básicamente se trata de un servidor que expone parte de su funcionalidad para que sus clientes (que generalmente son otros servidores de diferentes aplicaciones) puedan utilizarlas.

**Las principales ventajas de usar servicios WEB son:**

La posibilidad de valernos de la capacidad instalada (¡y mantenida!) por terceros

La facilidad para extender la funcionalidad de nuestra aplicación (Basta con realizar las llamadas al servicio web que deseamos)

**La principal desventaja** de usar este enfoque es que dependemos de servicios de terceros que, salvo en casos muy puntuales, no podemos controlar.

**Ejemplos de servicios WEB:**

Autenticación de usuarios vía Facebook, Google, Twitter, etc…

Generación de mapas en tiempo real

Consulta de cotizaciones de acciones

**Tipos de servicios WEB.**

**SOA (Service-Oriented Architecture)** es un tipo de arquitectura de software, la cual se basa en la integración de aplicaciones mediante servicios.

Sobre estos servicios se construyen: composiciones, BPM, proxys e incluso APIs. Pero… ¿REST y SOAP no son también servicios? Sí, son tecnologías que aparentemente ofrecen las mismas funcionalidades, pero no hacen lo mismo, aunque tanto REST como SOAP, siguen la misma arquitectura SOA, por lo que las dos siguen los mismos “principios”.

**SOAP Web Services.**

**SOAP (Simple Object Access Protocol) services.** Se trata de un protocolo para el intercambio de mensajes sobre redes de computadoras, generalmente usando HTTP. Este protocolo está basado en XML, facilitando la lectura, aunque los mensajes resultan más largos y por lo tanto considerablemente más lentos de transferir. Google, Facebook o entidades bancarias son algunos de los grandes sitios que actualmente utilizan aplicaciones que usan sistemas *web services.*

**REST (Representational State Transfer),** es un estilo de arquitectura de software dirigido a sistemas hipermedias distribuidos como lo es la web.

Este término se refiere específicamente a una colección de principios para el diseño de arquitecturas en red.

Existen varios proyectos que pueden verse beneficiados de una arquitectura REST. Concretamente aquellos en los que la idea principal está en la manera en la que se hacen las peticiones al servidor desde el cliente, basados en el recurso de interés.



**WSDL (Web Services Description Language)** es un protocolo basado en XML que describe los accesos al Web Service. Podríamos decir que es el manual de operación del mismo, porque nos indica cuáles son las interfaces que provee el Servicio web y los tipos de datos necesarios para su utilización. Utilizado para describir servicios SOAP.

**WADL (Web Application Description Language)** es una descripción XML legible por máquina de aplicaciones web basadas en HTTP (normalmente servicios web REST).​ WADL modela los recursos proporcionados por un servicio y las relaciones entre ellos. WADL está diseñado para simplificar la reutilización de servicios web basados en la arquitectura HTTP existente de la web. Es independiente de la plataforma y del lenguaje y tiene como objetivo promover la reutilización de aplicaciones más allá del uso básico en un navegador web.

WADL fue presentado al World Wide Web Consortium por Sun Microsystems el 31 de agosto de 2009,​ pero el consorcio no tiene planes actualmente para estandarizarlo. WADL es el equivalente de REST al Web Services Description Language (WSDL) de SOAP, que también puede ser usado para describir servicios web REST.

En nuestros ejemplos usaremos **JAX-RS**: ***Java API for RESTful Web Services*** es una API del lenguaje de programación Java que proporciona soporte en la creación de servicios web de acuerdo con el estilo arquitectónico Representational State Transfer (REST).1​ JAX-RS usa anotaciones, introducidas en Java SE 5, para simplificar el desarrollo y despliegue de los clientes y puntos finales de los servicios web.

A partir de la versión 1.1 en adelante, JAX-RS es una parte oficial de Java EE 6. Una característica notable de ser parte oficial de Java EE es que no se requiere configuración para comenzar a usar JAX-RS. Para los entornos que no son Java EE 6 se requiere una (pequeña) entrada en el descriptor de despliegue web.xml.

**Especificación:**

JAX-RS proporciona algunas anotaciones para ayudar a mapear una clase recurso (un POJO: el acrónimo de **Plain Old Java Object**, es una sigla utilizada por programadores Java para enfatizar el uso de clases simples y que no dependen de un framework en especial) como un recurso web. Entre estas anotaciones se incluyen:

**@ApplicationPath** especifica una ruta general del servicio.

**@Path** especifica la ruta de acceso relativa para una clase recurso o método.

**@GET**, **@PUT**, **@POST**, **@DELETE** y **@HEAD** especifican el tipo de petición HTTP de un recurso.

**@Produces** especifica los tipos de medios MIME de respuesta.

**@Consumes** especifica los tipos de medios de petición aceptados.

Además, proporciona anotaciones adicionales para los parámetros de método para extraer información de la solicitud. Todas las anotaciones @\*Param toman una clave de alguna forma que se utiliza para buscar el valor requerido.

**@PathParam** enlaza el parámetro a un segmento de ruta.

**@QueryParam** enlaza el parámetro al valor de un parámetro de consulta HTTP.

**@MatrixParam** enlaza el parámetro al valor de un parámetro de matriz de HTTP.

**@HeaderParam** enlaza el parámetro a un valor de cabecera HTTP.

**@CookieParam** enlaza el parámetro a un valor de cookie.

**@FormParam** enlaza el parámetro a un valor de formulario.

**@DefaultValue** especifica un valor por defecto para los enlaces anteriores cuando la clave no es encontrada.

**@Context** devuelve todo el contexto del objeto. (Por ejemplo: @Context HttpServletRequest request)

Para generar la documentación **WADL** de nuestro servicio WEB se puede usar **Jersey** <https://riptutorial.com/es/ebook/jersey>.

Ejemplo de formato JSON:

{  
    "personas": [  
        {  
            "nombre": "Carmen Tirosa",  
            "altura": 1.72,  
            "peso": 75,  
            "pasatiempos": [  
                "Comics",  
                "Baloncesto",  
                "Gaming"  
            ],  
            "soltero": true,  
            "direccion": {  
                "calle": "Roberto Rijas",  
                "numero": "44",  
                "ciudad": "Gotham"  
            }  
        },  
        {  
            "nombre": "Elena Politana",  
            "altura": 1.69,  
            "peso": 72,  
            "pasatiempos": [  
                "Programación",  
                "Senderismo"  
            ],  
            "soltero": true,  
            "direccion": {  
                "calle": "Eva Yatela",  
                "numero": "78",  
                "ciudad": "Hobbiton"  
            }  
        }  
    ]  
}