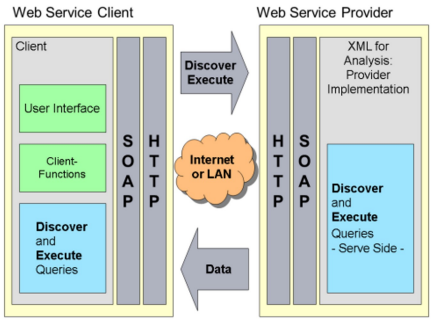
T1: Intro

lunes, 12 de mayo de 2025 18:13

Una **API** es una ***Application programming interface*** que puede ser local o remota, actuando como puente de comunicación/interacción entre 2 piezas de software, un *proveedor* -la entidad que ofrece el servicio- y un *consumidor* -la entidad que lo solicita-.

Por otro lado un **Servicio Web** puede definirse como una API remota que permite la comunicación entre aplicaciones a través de la red, utilizando estándares como XML, SOAP, UDDI, WSDL… Caracterizado por ser interoperable -es decir, que no depende de SSOO ni de lenguajes-. Permite la estandarización, el encapsulamiento, la integración y composición de servicios. Descritos/publicados/localizados/utilizados a través de la red

Estructura



Las **ventajas** que ofrecen los servicios web:
- Interoperabilidad.
- Estandarización.
- Encapsulamiento -información oculta-.
- Integración y composición de servicios -reutilización de servicios complejos-.
- Ubicuidad -ya que se comunican con cualquiera de: HTTP, XML, JSON-.

Y como **ventajas frente a Aplicaciones Web**:
- No hay problemas de compatibilidad.
- Son multiplataforma -portables-.
- Ahorran tiempo: instalación, despliegue, actualización…
- Alta disponibilidad

Sin embargo, también tienen **desventajas**:
- No tienen soporte a transacciones
- No muy buen rendimiento
- Problemas derivados de HTTP
- Interfaces inmutables
- Las respuestas no están garantizadas

Las desventajas **frente a Apps Web**:
- Menos funcionalidades
- Alta dependencia del servidor, de la red y de HTTP
- Necesidad de cifrado de la información

App Web	Diseñadas para humanos (HTML)
Servicio Web	• Exponen APIs • Machine to Machine

Tipos de servicios web

SOAP
Es un ***protocolo de mensajería*** estandarizado, con estado. Es un protocolo que define ***cómo*** se envía un mensaje, NO qué contiene, para lo cual se basa en XML. Está **orientado a operaciones**, es decir, **invoca funciones remotas RPC** (descritas con WSDL) que representan lógica del negocio. Es poco legible por humanos, es complejo, técnico y difícil de implementar.

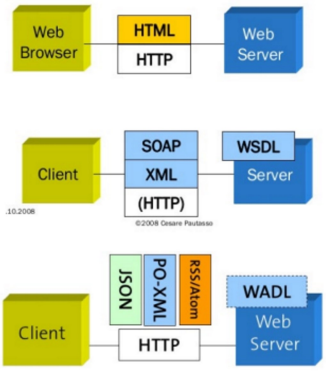
RESTful
Orientadas al recurso (basadas en la arquitectura REST)

RPC-Style
Orientadas al proceso (toda la información va encapsulada)

REST-RPC Hybrid
Es un mix

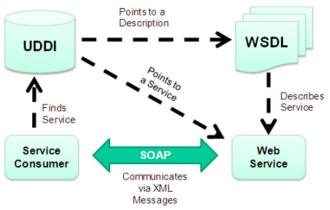
Un breve repaso histórico:

70
| *Remote Procedure Calls* (RPC)
80
|
90 --> Intento de estandarización -CORBA, COM, DCOM- … resultaron demasiado complicados
|
1998 --> XML & RPC, implementan HTTP para el transporte
1999 --> **SOAP** (*Simple Object Access Protocol*) (evolución de lo anterior)
2000 --> **REST** (*Representational State Transfer*)
|
|
2010's
| --> Webhooks
| --> GraphQL , provee solo los recursos estrictamente definidos
| --> gRPC
| --> MQTT , estándar ligero para IoT



Con respecto a la ***Arquitectura*** de la comunicación de estos protocolos tenemos:

Descubrimiento de servicios --> UDDI
Descripción de servicios --> WSLD
Mensajes XML --> XML-RPC y SOAP
Capa de transporte --> HTTP, FTP, SMTP



Un archivo "***.wsdl***" incluye, las operaciones que ofrece el servicio, que datos recibe y devuelve cada operación, dónde están alojados los servicios (URL) y qué protocolo y formato usan.

UDDI (Universal Descripción Discovery and Integration)
"Es como las páginas amarillas de los servicios web"
Basado en XML, funciona como repositorio central, que permite a los proveedores, publicar las descripciones y ubicaciones de sus servicios web. En este espacio, los clientes buscan al servicio, por su nombre, categoría, proveedor…

WSLD (Web Services Description Language)
"Es el manual de instrucciones de un servicio web SOAP"
Basado en XML, describe las operaciones de un servicio web -tanto la implementación de este (puertos), como su definición en sí (mensajes, tipos, parámetros…)-.

WADL (Web Application Description Language)
"Es un WSDL más sencillo, para servicios REST"
Basado en XML, describe la interfaz pública del servicio web, sin embargo, no se usa mucho, porque esto, ya lo hace por si mismo REST.