📘 Paradigma del desarrollo de aplicaciones orientadas a servicios

El **paradigma de desarrollo de aplicaciones orientadas a servicios** se basa en la creación de software compuesto por **servicios independientes**, reutilizables y accesibles a través de la red. Cada servicio ofrece una funcionalidad específica y puede ser consumido por diferentes aplicaciones, sin importar el lenguaje o plataforma en la que estén desarrolladas.

Este paradigma permite construir sistemas **flexibles, escalables y fácilmente integrables**, especialmente en entornos distribuidos y en la computación en la nube.

1️⃣ Servicios que se ofrecen en la nube

Los servicios en la nube se clasifican principalmente en los siguientes modelos:

🔹 Infraestructura como Servicio (IaaS)

Proporciona recursos básicos de cómputo:

- Servidores virtuales
- Almacenamiento
- Redes
- Sistemas operativos

📌 Ejemplos: Amazon EC2, Google Compute Engine, Microsoft Azure VM.

🔹 Plataforma como Servicio (PaaS)

Ofrece un entorno completo para desarrollar, probar y desplegar aplicaciones:

- Lenguajes de programación
- Bases de datos
- Frameworks
- Herramientas de desarrollo

📌 Ejemplos: Heroku, Google App Engine, Azure App Service.

🔹 Software como Servicio (SaaS)

Permite utilizar aplicaciones completas a través de Internet:

- No requiere instalación local
- Acceso desde navegador

📌 Ejemplos: Gmail, Google Drive, Microsoft 365, Dropbox.

2️⃣ Características de las aplicaciones orientadas a servicios

Las aplicaciones orientadas a servicios presentan las siguientes características principales:

- **Desacoplamiento**: los servicios no dependen directamente entre sí.
- **Reutilización**: un mismo servicio puede ser usado por múltiples aplicaciones.
- **Interoperabilidad**: funcionan entre diferentes plataformas y lenguajes.
- **Escalabilidad**: pueden crecer o reducirse según la demanda.
- **Autonomía**: cada servicio controla su propia lógica y datos.
- **Acceso mediante red**: generalmente a través de protocolos web (HTTP).

3️⃣ Aplicaciones Web híbridas (Mashup)

🔹 Concepto

Una **aplicación Web híbrida o Mashup** es una aplicación que **combina datos o servicios de múltiples fuentes** para crear una nueva funcionalidad o servicio.

Por ejemplo, una aplicación que utiliza:

- Google Maps
- Datos del clima
- Información de tráfico

Todo integrado en una sola interfaz.

🔹 Características de las aplicaciones Mashup

- Integran **APIs y servicios externos**
- Uso intensivo de **servicios web**
- Contenido dinámico
- Alta interacción con el usuario
- Aprovechan datos en tiempo real
- Desarrollo rápido y flexible

🏗️ Arquitectura Orientada a Servicios (SOA)

1️⃣ Definición de la Arquitectura Orientada a Servicios

La **Arquitectura Orientada a Servicios (SOA)** es un estilo arquitectónico que organiza el software como un conjunto de **servicios independientes**, bien definidos y accesibles a través de interfaces estándar.

Estos servicios pueden ser:

- Descubiertos
- Publicados
- Consumidos

sin que el consumidor conozca los detalles internos de su implementación.

2️⃣ Principios de diseño aplicados a los servicios en SOA

Los principales principios de diseño de SOA son:

1. **Contrato estandarizado**
   Cada servicio expone una interfaz claramente definida.
2. **Bajo acoplamiento**
   Los servicios minimizan dependencias entre sí.
3. **Alta cohesión**
   Cada servicio cumple una función específica.
4. **Reutilización**
   Diseñados para ser utilizados por múltiples consumidores.

5. **Autonomía**
   Controlan su propia lógica y datos.
6. **Descubrimiento**
   Los servicios pueden ser localizados fácilmente.
7. **Composición**
   Pueden combinarse para formar procesos más complejos.

---

3 Estándares relacionados con los servicios

XML (eXtensible Markup Language)

- Lenguaje de marcado para estructurar datos.
- Independiente de plataforma y lenguaje.
- Base para el intercambio de información entre servicios.

SOAP (Simple Object Access Protocol)

- Protocolo para el intercambio de mensajes estructurados.
- Usa XML para enviar solicitudes y respuestas.
- Funciona sobre HTTP, SMTP u otros protocolos.

WSDL (Web Services Description Language)

- Describe cómo funciona un servicio web.
- Define operaciones, mensajes, tipos de datos y ubicación del servicio.
- Permite a los clientes saber cómo consumir el servicio.

UDDI (Universal Description, Discovery and Integration)

- Registro de servicios web.
- Permite publicar y descubrir servicios.
- Facilita la integración entre empresas.

- **REST (Representational State Transfer)**

  - Estilo arquitectónico basado en HTTP.
  - Usa métodos HTTP: GET, POST, PUT, DELETE.
  - Ligero, rápido y ampliamente utilizado en APIs modernas.
  - Utiliza formatos como JSON o XML.

☑ Conclusión

El paradigma de desarrollo orientado a servicios y la arquitectura SOA permiten construir sistemas modernos, escalables y flexibles, facilitando la integración entre aplicaciones y el aprovechamiento de la computación en la nube. El uso de estándares como XML, SOAP y REST garantiza la interoperabilidad y la comunicación eficiente entre servicios.