UD2. Introducción a las arquitecturas web y servidores

1. Introducción a las arquitecturas web

1.1 ¿Qué es una arquitectura web?

Una **arquitectura web** es la estructura **lógica** y **física** sobre la cual se construye y ejecuta una aplicación web. Se refiere a la organización de los diferentes componentes (clientes, servidores, bases de datos, servicios, etc.) y cómo se comunican entre ellos para ofrecer una experiencia coherente al usuario final.

Podemos entender la arquitectura web como el plano de una casa: define qué elementos existen, cómo se relacionan y qué papel juega cada uno en el conjunto. Si el código de una aplicación es el contenido, la arquitectura es el esqueleto que permite que ese contenido funcione de forma organizada, escalable y eficiente.

Las decisiones arquitectónicas tienen un gran impacto en aspectos clave del desarrollo y mantenimiento de aplicaciones web, como:

- El rendimiento y la velocidad de respuesta.
- La facilidad para escalar el sistema (atender a más usuarios).
- La seguridad de la información.
- La capacidad de mantenimiento y evolución del software.
- La tolerancia a fallos o caídas del sistema.

Por eso, antes de comenzar a desarrollar o desplegar una aplicación, es esencial entender qué arquitectura se va a utilizar y por qué.

1.2 La evolución del modelo cliente-servidor

Las arquitecturas web actuales son el resultado de décadas de evolución tecnológica. Uno de los pilares fundamentales sobre el que se construyen es el **modelo cliente-servidor**, una forma de dividir responsabilidades entre dos entidades:

- El **cliente** (por ejemplo, el navegador web del usuario) solicita recursos o servicios.
- El **servidor** (una máquina remota con software adecuado) responde a esas peticiones proporcionando la información solicitada, ejecutando procesos o sirviendo archivos.

Este modelo comenzó a popularizarse en los años 80 y 90 con el auge de las redes locales y luego con Internet. A partir de ahí, han surgido diferentes variantes y evoluciones del modelo, como las arquitecturas multicapa, distribuidas, en la nube o serverless, que veremos en profundidad en el siguiente apartado.

Entender cómo ha evolucionado este modelo ayuda a comprender por qué hoy en día usamos determinadas herramientas y enfoques en el desarrollo web moderno.

1.3 Diferencias entre arquitectura y tecnología

Es importante no confundir **arquitectura** con **tecnología**. Una arquitectura describe **cómo se organiza un sistema**; las tecnologías son **las herramientas concretas** que usamos para implementarla.

Por ejemplo:

- Una arquitectura puede ser multicapa (dividir la lógica en presentación, negocio y datos).
- Las tecnologías usadas pueden ser HTML para la capa de presentación, PHP o Java para la lógica de negocio, y MySQL como base de datos.

Dos equipos pueden construir aplicaciones con la **misma arquitectura**, pero utilizando **tecnologías diferentes**. De igual manera, una misma tecnología (como Node.js) puede formar parte de distintas arquitecturas.

Esta distinción es clave para poder analizar, diseñar o desplegar una aplicación de forma profesional.

1.4 Componentes básicos de una arquitectura web

A pesar de la variedad de arquitecturas posibles, la mayoría de aplicaciones web comparten una serie de **componentes fundamentales**:

- **Cliente**: normalmente un navegador web (Chrome, Firefox...) desde el que el usuario accede a la aplicación.
- **Servidor web**: recibe las peticiones HTTP y sirve archivos o respuestas generadas dinámicamente.
- **Servidor de aplicaciones**: ejecuta la lógica de negocio (por ejemplo, un servlet Java, un script PHP, una aplicación Node.js...).
- **Base de datos**: almacena y gestiona la información persistente de la aplicación.

- **DNS**: traduce nombres de dominio (como `midominio.com`) en direcciones IP para localizar los servidores.
- **Red y protocolos**: aseguran la comunicación entre los componentes.

En arquitecturas más complejas también pueden aparecer balanceadores de carga, sistemas de caché, servicios en la nube, colas de mensajes o contenedores, que iremos viendo a lo largo del módulo.

1.5 Ejemplos cotidianos de arquitecturas web

Muchas veces usamos aplicaciones web sin ser conscientes de la complejidad de su arquitectura. Veamos algunos ejemplos comunes:

- **Una web estática** (por ejemplo, una página personal o un sitio institucional sin formularios ni bases de datos) puede estar montada con una arquitectura muy simple: cliente + servidor web.

- **Una tienda online** como Amazon requiere una arquitectura mucho más elaborada, que incluye:

    - Balanceadores de carga.
    - Servidores de aplicaciones y bases de datos replicadas.
    - Sistemas de recomendación.
    - Microservicios especializados.
    - Servicios de autenticación.
    - Integración con pasarelas de pago.

- **Un servicio como Google Docs** es aún más complejo: permite colaboración en tiempo real, sincronización automática, almacenamiento en la nube, control de versiones y permisos, lo cual implica arquitecturas distribuidas, sistemas de mensajería, bases de datos NoSQL y muchos más componentes.

Entender la arquitectura que hay detrás de cada caso nos ayuda a visualizar la escala de los desafíos técnicos que plantea cada tipo de aplicación y a tomar decisiones acertadas en el diseño y despliegue.