Servidores de aplicaciones web

DESARROLLO WEB EN ENTORNO SERVIDOR. UNIDAD 1

2º DESARROLLO DE APLICACIONES WEB. ARCIPRESTE DE HITA. 2025/2026

AARÓN MONTALVO

1. Servidores de aplicaciones web. Criterios de evaluación

a) Se han caracterizado y diferenciado los modelos de ejecución de código en el servidor y en el cliente web.

b) Se han reconocido las ventajas que proporciona la generación dinámica de páginas.

c) Se han identificado los mecanismos de ejecución de código en los servidores web.

d) Se han reconocido las funcionalidades que aportan los servidores de aplicaciones y su integración con los servidores web.

e) Se han identificado y caracterizado los principales lenguajes y tecnologías relacionados con la programación web en entorno servidor.

f) Se han verificado los mecanismos de integración de los lenguajes de marcas con los lenguajes de programación en entorno servidor.

g) Se han reconocido y evaluado las herramientas y frameworks de programación en entorno servidor.

1. Servidores de aplicaciones web. Contenidos

1.1. Aplicaciones web

1.2. Páginas web estáticas y dinámicas

1.3. Tecnologías web de lado del servidor

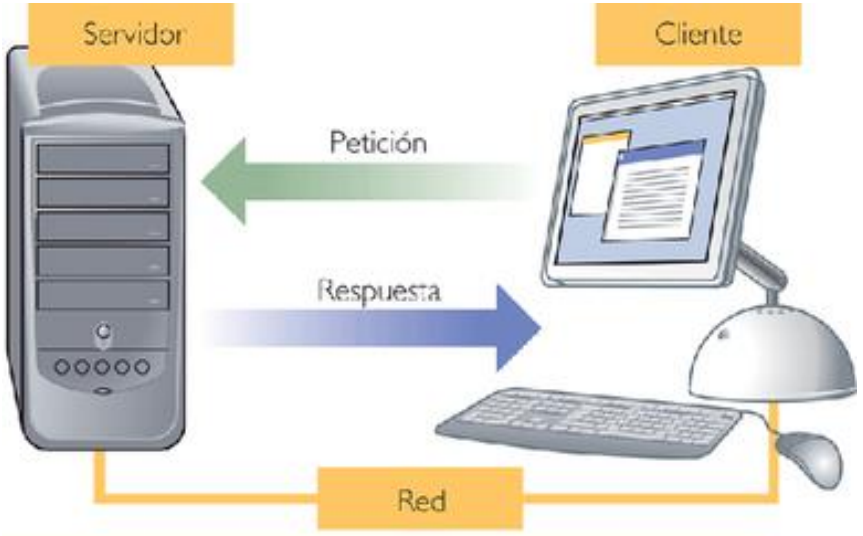
1.4. Lenguajes en entorno servidor

1.5. Herramientas y aplicaciones

1.1. Aplicaciones web. Definición

Herramientas software que los usuarios pueden utilizar accediendo a un servidor web (Internet o intranet) utilizando un navegador.

- No es sólo una página web que proporciona información.



1.1. Aplicaciones web. Ventajas

Reducción de costes. No requiere instalar *software* en los clientes. Además, los clientes no requieren *hardware* potente, solo un navegador.

Información centralizada. Facilita la consistencia de la información y su mantenimiento: seguridad, copias, etc.

Actualizaciones. El cliente siempre se conectará a la última versión instalada en el servidor.

Movilidad. Se puede acceder desde cualquier lugar con conexión a internet y un navegador.

1.1. Aplicaciones web. Inconvenientes

Interfaz web. Las características y opciones de la interfaz están limitadas por lo que pueda mostrar el navegador.

Conectividad continua. No se puede trabajar si falla la conexión.

Límites de tamaño y tiempo. Hay ciertas operaciones que no se pueden implementar correctamente en web: tiempo real, edición de multimedia, etc.

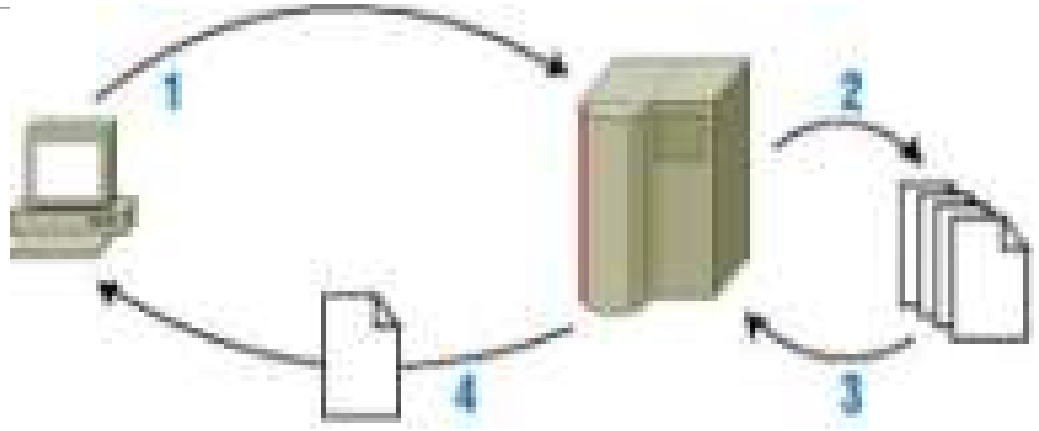
1.2. Páginas web estáticas y dinámicas. Páginas web estáticas

El usuario recibe (descarga) una página web desde el servidor sin ningún tipo de interacción, ni en la propia página web ni generando ninguna respuesta del servidor.

Utilizan lenguajes de marcado (HTML o XHTML) con estilos (CSS) para la organización visual de la información.

Usan una **arquitectura cliente-servidor** clásica.

1.2. Páginas web estáticas y dinámicas. Páginas web estáticas



1. El usuario solicita una página web a un servidor.
2. El servidor busca esa página en su almacén
3. El servidor recupera la página solicitada, si esta es encontrada.
4. El servidor envía la página al navegador del usuario para que este la pueda mostrar

1.2. Páginas web estáticas y dinámicas. Páginas web estáticas. Ventajas

Portabilidad: funcionan en cualquier servidor.

Tiempo de acceso: se cargan directamente.

Tamaño: tiene menores costes de alojamiento.

Facilidad: no hace falta saber programar, solo marcas

Instalación: no hay que instalar ni configurar ningún *software* en el servidor.

1.2. Páginas web estáticas y dinámicas. Páginas web estáticas. Inconvenientes

Funcionalidades: ausencia de movimiento y adaptaciones al usuario.

Extensiones: no pueden usar bases de datos, ni tecnologías relacionadas como foros, galerías, etc.

Mantenimiento: toda actualización ha de ser manual, y el administrador ha de acceder al servidor para cambiar cualquier contenido.

1.2. Páginas web estáticas y dinámicas. Páginas web dinámicas

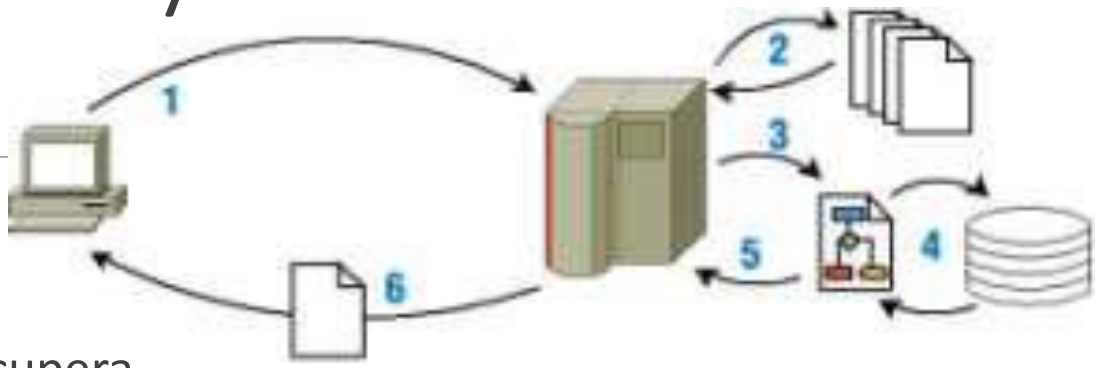
El usuario recibe una página web desde el servidor con interacción, que puede ser de dos tipos:

Interacción en la propia página web. La propia página incluye código ejecutable, normalmente en JavaScript, que añade funcionalidades de apariencia, visualización, etc.

- => ***DWEC***

Interacción con el servidor. El contenido almacenado en servidor no se muestra directamente a cada cliente, sino que se realizan diversas operaciones y finalmente se envía un resultado distinto.

1.2. Páginas web estáticas y dinámicas. Páginas web dinámicas



1. El usuario solicita una página web al servidor.
2. El servidor busca esa página en su almacén y la recupera.
3. Si es una web dinámica, el servidor contacta con el módulo de ejecución y le envía el contenido ejecutable.
4. Si el módulo detecta que hay que obtener información de algún repositorio (como una base de datos), el módulo la recupera.
5. El módulo de ejecución devuelve una página en HTML al servidor
6. El servidor envía la página al navegador del usuario.

1.2. Páginas web estáticas y dinámicas. Páginas web interactivas o asíncronas

Ejecución de código en servidor y en cliente

Las tecnologías de servidor y de cliente se complementan unas con otras.

Por ejemplo, el código en el servidor web se encarga de recuperar los correos, mientras que el del cliente (el navegador) avisa cuando se quiere borrar uno.

Tradicionalmente, el código que se ejecuta en el cliente no tenía acceso a los datos que están almacenados en el servidor.

1.2. Páginas web estáticas y dinámicas. Páginas web interactivas

La técnica de desarrollo web (no lenguaje) conocida como AJAX (Asynchronous JavaScript And XML) permite la interacción del usuario con la página web, produciendo un diálogo entre cliente y servidor de manera transparente al usuario y sin necesidad de recargar la página.

En el lado del cliente encontramos HTML y JavaScript.

En el lado del servidor hay lenguajes de script y/o ejecutables.

En el correo web, JavaScript detecta que se ha pulsado un correo y automáticamente le pide al servidor la información para mostrarlo en el navegador.

Actividad 1.1

1. Busca ejemplos reales de las desventajas de una aplicación web frente a una aplicación de escritorio.

2. Busca ejemplos reales de cada uno de los tipos de aplicación Web:
   - estática
   - dinámica
   - interactiva o asíncrona.

3. Enumera 2 ventajas y 2 inconvenientes de cada tipo de aplicación web.

1.3. Tecnologías web de lado del servidor. Componentes principales

Servidor web. Recibe las peticiones y envía las página. También se comunica con los módulos de ejecución.

Módulo(s) de ejecución de código. Programa que genera la página (HTML) resultante. Se integra con el servidor y depende del lenguaje usado.

Lenguaje de programación de las aplicaciones. Utilizado para el desarrollo y depuración.

SGBD. No es necesario, pero todas las aplicaciones web suelen contar con uno.

1.3. Tecnologías web de lado del servidor. Arquitecturas web

Modelo cliente-servidor

Aplicación distribuida con tareas repartidas entre los proveedores de recursos o servicios (**servidores**) y los consumidores de los mismos (**clientes**). La relación entre ambos se basa en el intercambio de mensajes de control y datos.

El concepto *servidor* se refiere tanto a distintas máquinas como distintos procesos.

1.3. Tecnologías web de lado del servidor. Arquitecturas web

Modelo cliente-servidor

El **cliente** tiene un papel activo:

- Inicia las solicitudes y espera y recibe las respuestas del servidor.
- Interactúa directamente con los usuarios a través del GUI.
- Puede conectarse con varios servidores a la vez.

El **servidor** tiene un papel pasivo:

- Espera solicitudes, las procesa y envía la respuesta.
- No suele interactuar directamente con los usuarios.
- Puede aceptar varias peticiones a la vez.

1.3. Tecnologías web de lado del servidor. Arquitecturas web

Modelo cliente-servidor

Ventajas

Centralización del control: los accesos se controlan por el servidor.

Escalabilidad: se puede aumentar el número de clientes a los que da servicio.

Mantenimiento: las actualizaciones son en el servidor.

Inconvenientes

Punto único de entrada: si falla el servidor falla todo el sistema.

Picos de funcionamiento: puede haber momentos en que el sistema no pueda atender todas las peticiones pero no merezca la pena escalar.

1.3. Tecnologías web de lado del servidor. Arquitecturas web

Otras arquitecturas: Arquitecturas físicas

Hacen referencia a la forma en la que se distribuye la infraestructura (elementos *hardware*).

Los elementos se dividen en niveles

- Cada elemento que tenga un rol distinto representará una capa diferente.

Esta distribución da lugar a arquitecturas multinivel.

1.3. Tecnologías web de lado del servidor. Arquitecturas web

Otras arquitecturas: Arquitecturas lógicas

Hace referencia a la forma en la que se divide el software.

La aplicación web se divide en **capas** para mejorar la organización por funcionalidad. Cada capa no tiene por qué estar en una máquina diferente.

- Facilita la reusabilidad, el mantenimiento y ciclos de desarrollo más cortos.

1.3. Tecnologías web de lado del servidor. Arquitecturas web lógicas

Modelo-Vista-Controlador (MVC): separa datos y la lógica de negocio.

Arquitectura en 3 capas: separa presentación, lógica de negocio y persistencia (datos).

Arquitectura multicapas: como la anterior, pero añade una capa de entidades.

Arquitectura orientada a servicios (SOA): usa estándares de comunicación (XML, WSDL, JSON).

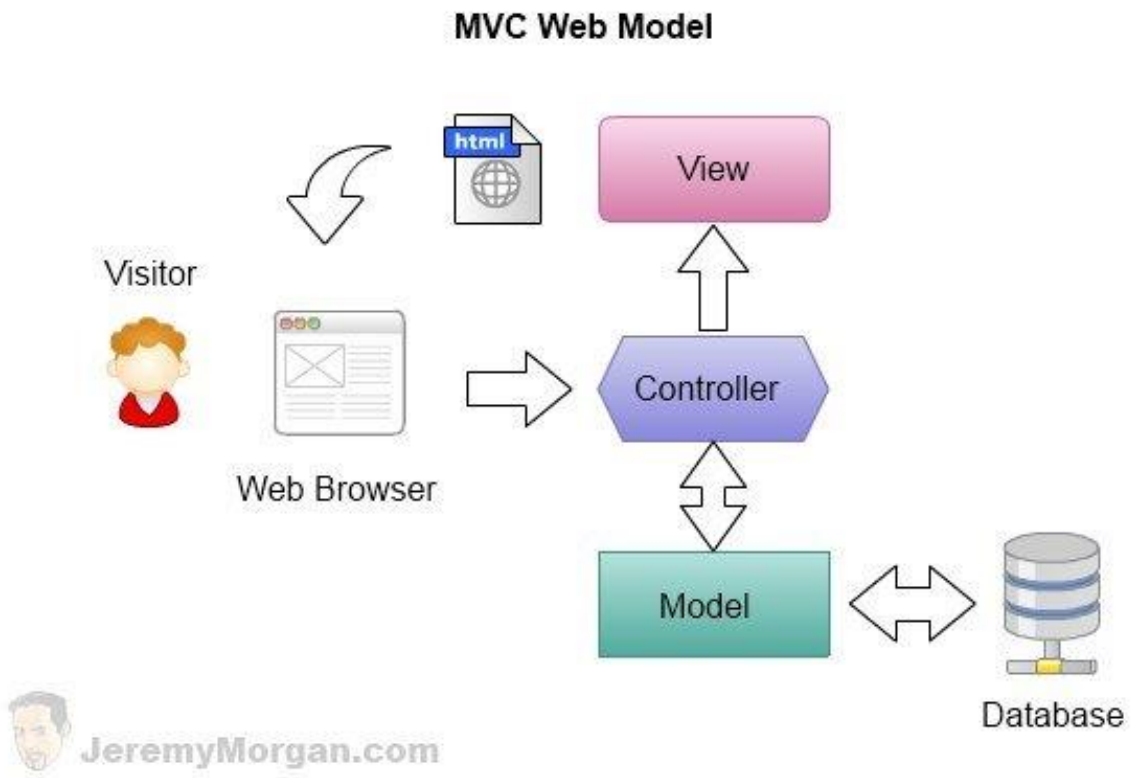
1.3. Tecnologías web de lado del servidor. Arquitecturas web lógicas

Modelo-Vista-Controlador (MVC)

Modelo: implementa la lógica de negocio (la funcionalidad).

Vista: despliega la interfaz de usuario.

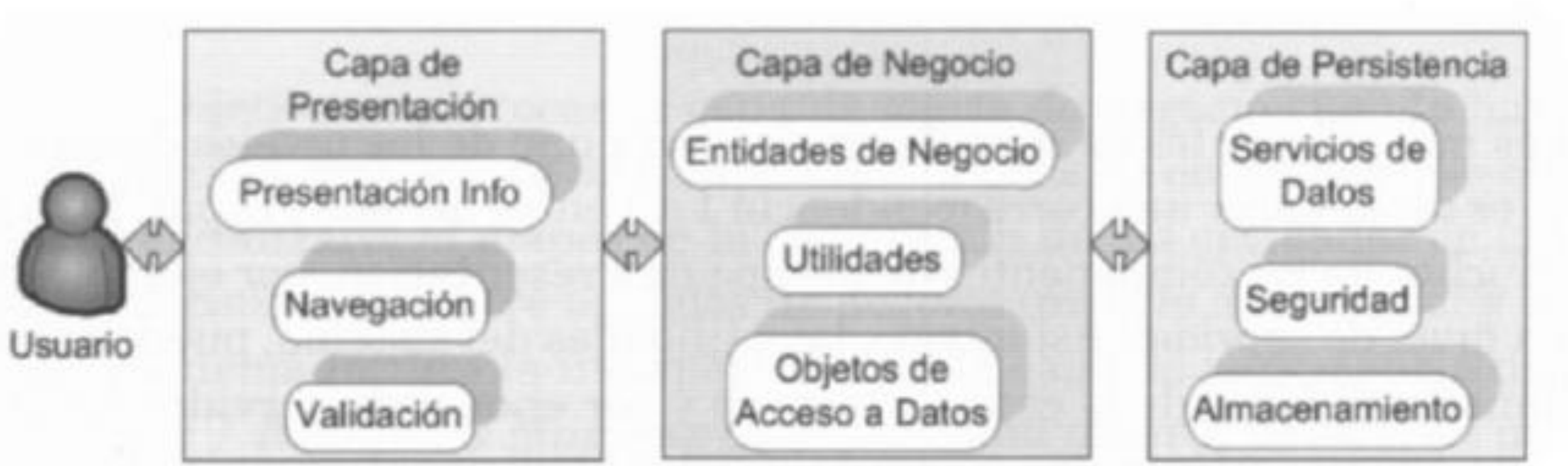
Controlador: maneja la interfaz de usuario, manipulando el modelo y seleccionando la vista a desplegar.



1.3. Tecnologías web de lado del . Arquitecturas web lógicas

Arquitectura en 3 capas

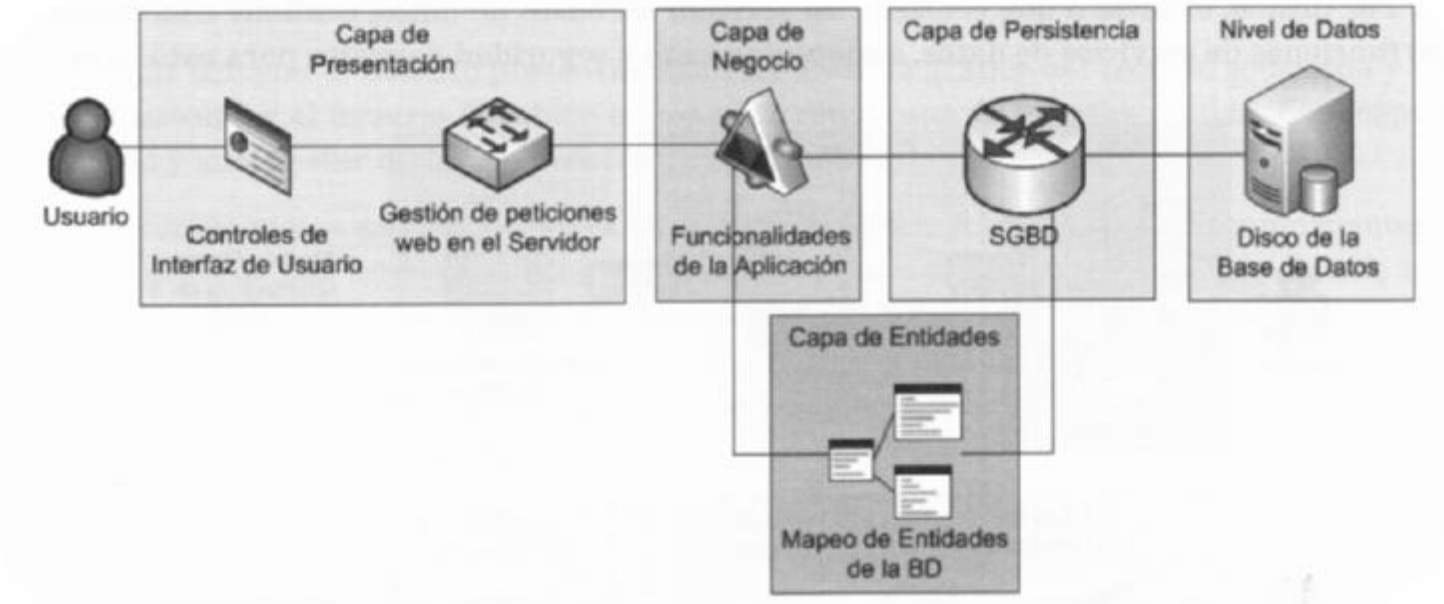
Cada capa está formada por componentes que proporcionan servicios



1.3. Tecnologías web de lado del . Arquitecturas web lógicas

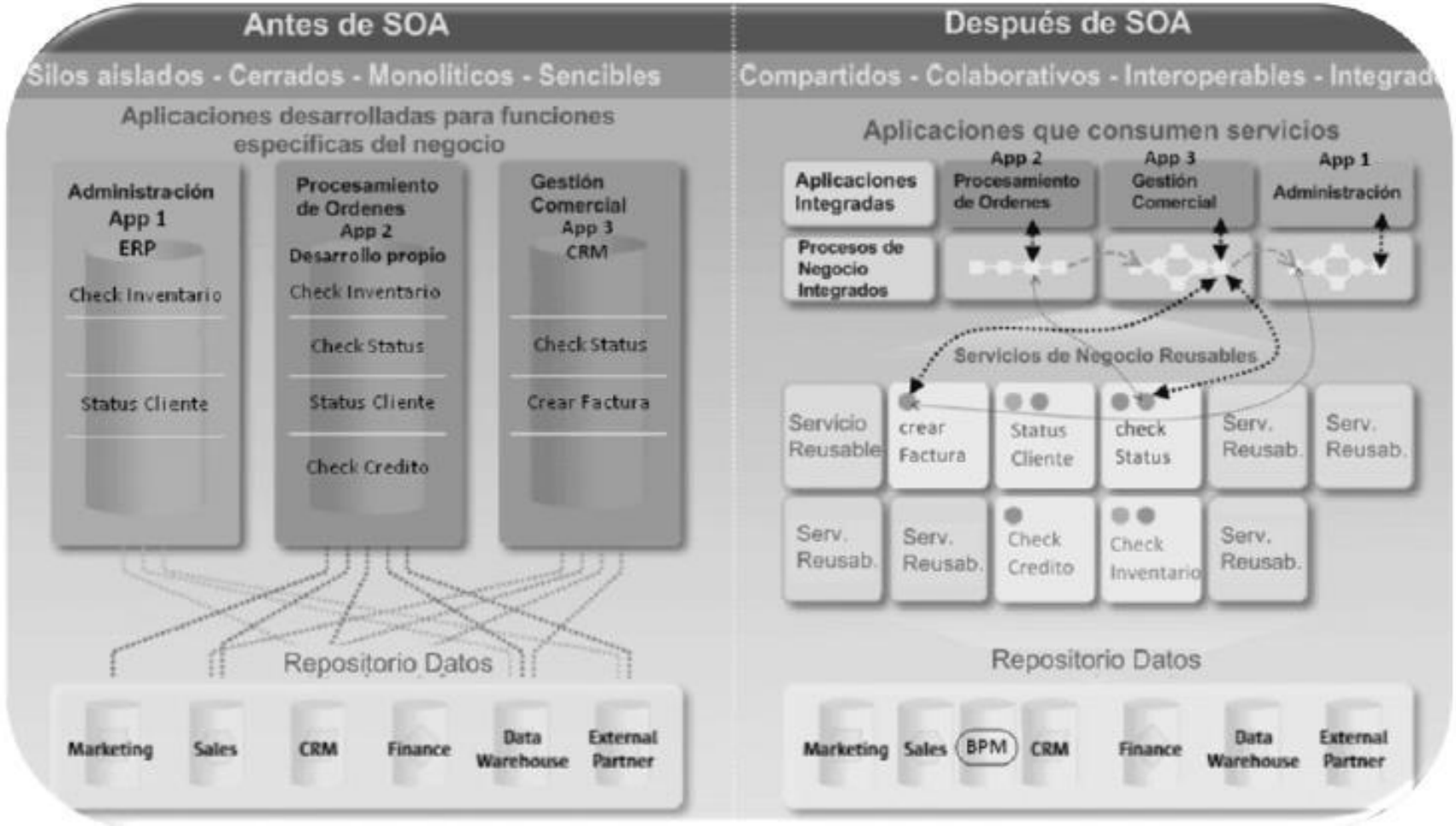
Arquitectura multicapas

Relaciona las capas de negocio y de persistencia a través de nuevas capas de entidades



1.3. Tecnologías web de lado del servidor. Arquitecturas web lógicas

Arquitectura orientada a servicios (SOA):



1.3. Tecnologías web de lado del servidor. Servidores

El **servidor web** provee el contenido solicitado al navegador del cliente

La solicitud debe tener una estructura, la URL

- Protocolo (HTTP), dirección del servidor, puerto y ruta

Devuelve tanto webs estáticas como dinámicas en HTML.

Ejemplo: **Apache**

El **servidor de aplicaciones** se encarga de ejecutar el código de las aplicaciones

Genera el resultado (HTML) que se va a enviar al cliente.

Muchas veces se instala como extensión del servidor web.

Ejemplo: **Apache Tomcat** (contenedor de *servlets*)

1.3. Tecnologías web de lado del servidor. *Front-end* vs. *back-end*

También hay que tener en cuanta dónde se sitúan y para qué se utilizan las tecnologías, teniendo 3 perfiles diferenciados.

Front-end

Es la parte del desarrollo que se encarga del diseño y la maquetación de la aplicación web.

Utiliza lenguajes de marcas como HTML con CSS, pero también Javascript.

El desarrollador tiene que asegurar que la presentación es correcta en cualquier dispositivo.

1.3. Tecnologías web de lado del servidor. *Front-end* vs. *back-end*

Back-end

Es la parte del desarrollo que se encarga del lado del servidor web.

Incluye el servidor de aplicaciones y las bases de datos.

Utiliza lenguajes como PHP, ASP o Python.

Full stack

Es un perfil que se compone de la suma los dos anteriores.

El desarrollador no es experto en ninguna tecnología, pero tiene una visión de conjunto y da apoyo al resto.

1.4. Lenguajes en entorno servidor

Son aquellos cuyo código es reconocido, interpretado y ejecutado por el propio servidor antes de enviar el resultado en un formato comprensible para el navegador.

Existen múltiples alternativas:

- Lenguajes de *scripting*: PHP, ASP, JSP, Perl.
- Componentes ejecutables: CGI, EJB, *servlets*.
- Aplicaciones híbridas: ASP.NET.

1.4. Lenguajes en entorno servidor. Lenguajes de *scripting*

El código de la aplicación (*script*) se intercala dentro del código HTML.

PHP (PHP: Hypertext Processor): uno de los lenguajes más extendidos

- Código abierto, portable y gratuito.
- Soportado por la mayoría de servidores web actuales.

ASP (Active Server Pages): tecnología propietaria y de código cerrado

- Diseñado por Microsoft para IIS, puede ser utilizado con otros servidores.

JSP (Java Server Pages): La primera vez que se invoca es transformado en un *servlet* que permanece en memoria para sucesivas llamadas.

1.4. Lenguajes en entorno servidor. Componentes ejecutables

Un programa externo que recibe los parámetros de entrada y devuelve un resultado, la web que se mostrará en el cliente

CGI (Common Gateway Interface): estándar que permite al servidor ejecutar programas genéricos, escritos en cualquier lenguaje (C, C++, Perl, etc).

- establece los vínculos que hay que establecer con la aplicación independiente
- el principal inconveniente es el rendimiento, cada petición crea un nuevo proceso
- para solucionarlo hay varias opciones: extensiones **NSAPI** e **ISAPI**, CGI integradas en el propio servidor web y **FastCGI**, que crea un solo proceso para todas las peticiones.

servlets, **JavaBeans** y **EJB** (Enterprise Java Beans): plataformas independientes de ejecución de código basadas en Java (JVM).

1.4. Lenguajes en entorno servidor. Aplicaciones híbridas

Tecnología intermedia entre los lenguajes de *scripting* y las aplicaciones independientes

ASP.Net: Utiliza *webforms* que integran contenido estático (HTML) y *controles web* que se procesan en el servidor.

- Las aplicaciones pueden ser escritas en cualquier lenguaje del *framework* .NET (VB.Net, C#, Jscript.Net).
- La aplicación se precompila una sóla vez a lenguaje máquina, y cada nueva petición tendrá una compilación en tiempo de ejecución.

Actividad 1.2

Elabora un informe tabla comparativa de las características e inconvenientes de 5 lenguajes del lado servidor: PHP, C#, Python y otros dos lenguajes que tú elijas.

	PHP	C#	Python	Java	Ruby
Propósito	Específico: Aplicaciones web con bases de datos	General. Aplicaciones web, de escritorio, lado del cliente, juegos	General	Aplicaciones y servicios digitales generales	Scripting y aplicaciones generales
Licencia	Libre	Libre	Libre	Libre	Libre
Tipo de lenguaje	Orientado a objetos	Orientado a objetos	Multiparadigma	Orientado a objetos	Orientado a objetos
Ventajas	Facilidad de aprendizaje	Seguridad	Versatilidad	Portabilidad	Fuerte en frameworks
Inconvenientes	Rendimiento	Limitaciones en S.O	Tiempo de respuesta	Alta necesidad de recutrsos	Baja popularidad

1.5. Herramientas y aplicaciones. Herramientas de programación

Editor de texto.

Permite escribir código pero sin ningún tipo de ayuda.

- Bloc de notas

Marcador de texto.

Es un editor de texto que además ayuda con la sintaxis del lenguaje, con funciones avanzadas como añadir tabulaciones, cambiar de color las etiquetas en función de su función, etc.

- Notepad++, UltraEdit, Atom, Sublime.

1.5. Herramientas y aplicaciones. Herramientas de programación

Entorno de programación (IDE):

Permite editar, compilar y ejecutar aplicaciones.

Además de reconocer la sintaxis, ofrecen funcionalidades avanzadas orientadas a la programación.

- Genéricos: Eclipse, Dreamweaver, NetBeans IDE, Visual Studio Code / VSCodium
- Específicos para PHP: PhpStorm.

1.5. Herramientas y aplicaciones. Servidores web y plataformas

Para poder desarrollar aplicaciones web se necesita instalar un servidor web que sea capaz de interpretar los scripts del lenguaje de programación escogido.

Existen servidores web libres y propietarios, que a su vez pueden contener distintos servidores de aplicaciones.

Los servidores más usados son **nginx** y **Apache**, ambos de código abierto y compatibles con muchos otros componentes.

1.5. Herramientas y aplicaciones. Servidores web y plataformas

Cada servidor puede a su vez integrarse con otras tecnologías distintas formando una plataforma o *stack* de *software* completo para el desarrollo.

Una de las soluciones más versátiles es la llamada **AMP**, compuesta por Apache, MySQL/MariaDB y PHP/Perl/Python.

- Está disponible para distintos sistemas operativos: BAMP, LAMP, MAMP, WAMP.

Actividad 1.3 (1/2)

Realiza una comparativa de la plataforma XAMPP con al menos 4 soluciones distintas de servidor web completas que **no** sean de tipo AMP.

Deberás hablar de

- sistemas operativos soportados
- lenguajes de programación soportados
- sistemas gestores de bases de datos soportados
- tipo de licencia

Actividad 1.3 (2/2)

	XAMPP	Dockers	Vagrant	Laragon
SSOO soportados	Windows, Linux, MacOS	Windows, Linux, MacOS	Windows, Linux, MacOS	Windows
Lenguajes de programación	PHP, Perl	Cualquier lenguaje	Cualquier lenguaje	PHP, Node.js, Python y mas lenguajes modernos
SGBD soportados	MySQL / MariaDB	No tiene limitaciones	No tiene limitaciones	MySQL / MariaDB
Tipo de licencia	Libre	Libre	Libre	Gratuita cerrada