Desarrollo web en entorno servidor

[Tema 1. Conceptos Generales 2](#_Toc146205156)

* [Desarrollar una aplicación 2](#_Toc146205157)
* [Aplicaciones Web 3](#_Toc146205158)
* [Entorno Servidor 4](#_Toc146205159)

[Tema 2. Arquitectura y tecnología web 6](#_Toc146205160)

* [Arquitectura web 6](#_Toc146205161)
* [Tecnologías web 7](#_Toc146205162)

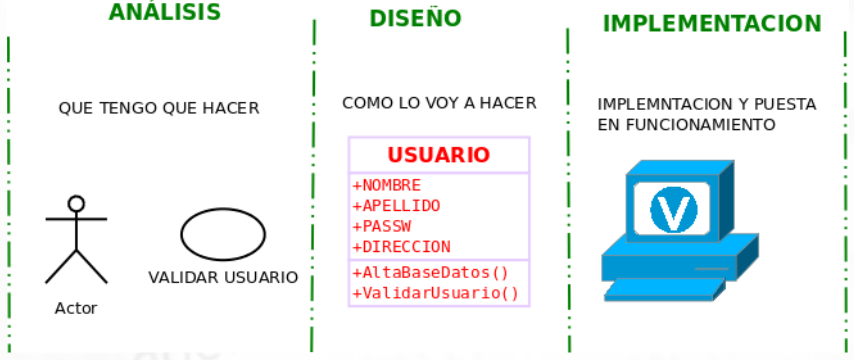
# Tema 1. Conceptos Generales

## Desarrollar una aplicación

Dado un problema de naturaleza lógica, consiste en

* *Implementar* o construir un programa
* Usando un determinado lenguaje de programación
* Dicho programa está formado por un conjunto de instrucciones
* Ejecutadas en un entorno computacional
* Y que solucionan de forma automatizada el problema planteado

Cuando decimos *implementar*, nos referimos a analizar el problema, diseñar una solución algorítmica válida y escribir el código de dicha solución utilizando uno o varios lenguajes de programación concretos (interpretados o compilados).



**¿Compilación o interpretación?**

Las instrucciones escritas han de pasar a código máquina para que se puedan ejecutar en la pantalla. Para ello, dichas instrucciones pueden compilarse o interpretarse.

*Java* es un lenguaje particular porque es compilado, pero es compilado a un lenguaje intermedio llamado bytecode, que después es interpretado. Se debe instalar el intérprete JRE (Java Runtime Environment) y el JDK (Java Development Kit) que incluye el compilador, entre otras herramientas de desarrollo.

En un entorno de ejecución web *debemos utilizar un modelo en entorno interpretado*, que se adapta a cualquier tipo de máquina.

## Aplicaciones Web

**¿Qué es una aplicación web?**

No todos los programas o aplicaciones son del mismo tipo. Algunos ejemplos de aplicaciones: de escritorio, de ingeniería, de tiempo real, de inteligencia artificial, de predicciones metereológicas, de cálculo científico, de juegos, web…

[Para elegir el lenguaje de programación que queremos emplear para desarrollar nuestra app web debemos tener en cuenta que se agrupan en “frameworks” o estructuras organizadas de código donde hay bibliotecas predefinidas, plantillas y escabilidad para no tener que empezar a escribir código desde 0. Algunos frameworks populares son Django (Python), Laravel (PHP) y Angular (JavaScript). Sirve como inicio de la organización y desarrollo del software y facilita la programación, minimizando errores y promoviendo el trabajo colaborativo para conseguir un software de calidad.]

World Wide Web es un sistema de documentos de hipertexto (html) enlazados entre sí (hipertexto e hipervínculos). A ellos se puede acceder por medio de Internet (red de tipo TCP/IP). Los usuarios navegan en ellos a través de *navegadores* o *buscadores*.

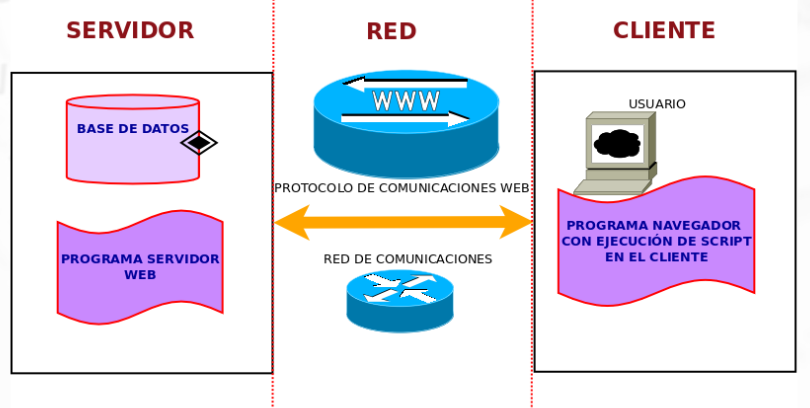
Una aplicación web nunca se va a detener esperando que le demos un valor: el servidor está trabajando de forma continua porque tiene que atender a múltiples usuarios. Es lo que le diferencia de las apps de escritorio. En la ejecución web, el servidor recibe la solicitud de ejecución junto con los parámetros o valores de variables para la ejecución.

**Características de una aplicación web:**

* Acceso a través de un navegador web (sin instalación)
* Interactividad
* Dinamismo (no hay necesidad de recargar toda la página)
* Acceso remoto a datos y servicios
* Arquitectura cliente-servidor
* Independencia de la plataforma
* Capacidad de respuesta (responsive)
* Seguridad (encriptación y autenticación)
* Mantenimiento centralizado

## Entorno Servidor

**Funcionamos con un cliente que solicita y un servidor que entrega**



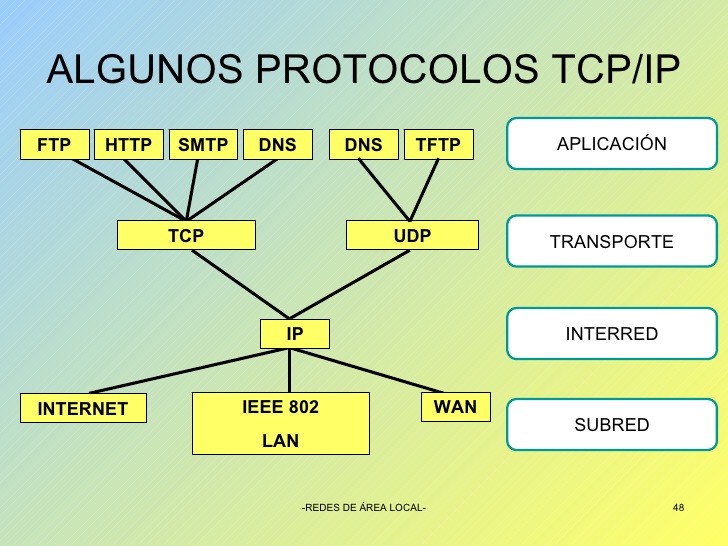
* El cliente es el navegador (una aplicación ejecutándose en una máquina con una IP) que solicita un recurso (generalmente una página web).
* El servidor es una aplicación (Apache) ejecutándose en una máquina que atiende a las solicitudes que entran por un puerto y entregan el recurso solicitado.
* Las aplicaciones Web usan una arquitectura Cliente/Servidor. Está basada en la idea del servicio.

El *servidor* es un software ejecutándose en una máquina escuchando a que le soliciten un servicio. El *cliente* (que también se conoce como *agente*) inicia el proceso solicitando un servicio al servidor. El servidor responde enviando uno o más flujos de datos al cliente.

Cuando el cliente hace una petición de una página web, lo hace escribiendo la URI en el navegador, casi siempre así: *protocolo://maquina/ruta/recurso/solicitud*

Para que la red funcione, hay varios programas funcionando simultáneamente en distintos niveles. Dentro del nivel de red tenemos el protocolo IP, cuya función es localizar la red donde está el equipo destino. En el nivel de transporte, tenemos los protocolos UDP y TCP, que localizan la aplicación a la que se destinan los datos a través de un número de puerto.

**Medios físicos y lógicos del proceso web**



El servidor en el proceso de ejecución de una aplicación web:

* Recibe e interpreta la solicitud del cliente
* Accede a bases de datos según necesidades
* Ejecuta código en el servidor, generando página HTML

# Tema 2. Arquitectura y tecnología web

## Arquitectura web

Una **arquitectura web** es la estructura del producto final, y se va a ir perfilando durante todo su desarrollo. Primero tendremos que planificar, diseñar y construir el producto final para después usar las herramientas para llevarlo a cabo.

Los **componentes** de una arquitectura web son:

* El servidor web, para recibir las peticiones de los clientes.
* Un servidor web es un programa encargado de recibir una solicitud por la red y responder a dicha solicitud. En una red de tipo TCP/IP, que es la que usamos en internet, la solicitud se recibe usando el protocolo HTTP.
* Es posible que a la hora de verificar permisos y ejecutar script necesite hacer solicitudes a otros servidores.
* El servidor de bases de datos.
* Es un programa encargado de gestionar una base de datos.
* Dada la importancia de esta sección, por confidencialidad y posible cuello de botella, en una aplicación el servidor de bases de datos suele estar en un servidor dedicado específicamente a ese cometido.
* El módulo de ejecución de código en el servidor.
* Ante una solicitud del cliente el servidor no solo entrega una página web, sino que suele ejecutar código al que el cliente no tiene acceso. Para ello, tenemos que tener un módulo instalado en el servidor para ejecutar ese código. Nosotros usaremos PHP y posteriormente JavaScript en el servidor con NodeJS.
* Genera la página web final, y es un módulo que debe integrarse de alguna forma con el servidor web. Dependerá del lenguaje y tecnología que utilicemos para programar la aplicación.
* En el cliente, el módulo de ejecución de código:
  + Le muestra la página del servidor.
  + Le interpreta las etiquetas de HTML con sus atributos.
  + Aplica el estilo detallado en las páginas CSS.
  + Ejecuta el código Javascript y posibilita sus eventos.

Además de tener en cuenta los componentes, también es importante decidir cómo se va a organizar el código de la aplicación. Muchas de las arquitecturas que se usan en la programación de aplicaciones web te ayudan a estructurar el código de las aplicaciones en *capas* o niveles. El motivo de dividir en capas el diseño de una aplicación es que se puedan separar las funciones lógicas de la misma, de tal forma que sea posible ejecutar cada una en un servidor distinto (en caso necesario).

Internamente se puede dividir la funcionalidad de una aplicación web en tres capas:

1. **Capa de acceso a datos**. Se tendrá que encargar de almacenar la información de la aplicación en una base de datos y recuperarla cuando sea necesario.
2. **Capa intermedia**. Donde deberás programar la funcionalidad de tu aplicación.
3. **Capa cliente**. Donde programarás todo lo relacionado con el interface de usuario, la parte visible de la aplicación con la que interactuará.

## Tecnologías web

Al hablar de tecnologías, hablamos de *herramientas* *concretas que usamos para una arquitectura establecida*. En este aspecto, estamos hablando de lenguajes concretos de programación, de gestores de bases de datos y otros elementos concretos que usamos para crear nuestro sistema.

Las tecnologías son muy dinámicas, van cambiando, aparecen nuevas tecnologías, aparecen nuevas formas de utilizar una tecnología existente. No debe de asustarnos tanto dinamismo: Conocer bien una tecnología concreta te posibilita aprender o adaptarte fácilmente a otra situación (otra tecnología u otra forma de usar una tecnología, como usar un framework concreto).

Algunos ejemplos de tecnologías serían PHP, MySQL, MongoDb, NodeJS…

Un framework establece un modelo VC donde se establece una estructura de carpetas, un modo de realizar la programación y unas utilidades disponibles para el desarrollador. La tecnología no sería el framework sino el lenguaje que éste utiliza.