



DWES

Aplicaciones Web

2º DAW


Índice


1. Arquitectura cliente-servidor
2. Servicio y aplicación Web
3. Creación de aplicaciones Web

Arquitectura cliente-servidor

La arquitectura cliente-servidor es un modelo de aplicación distribuida en el que **las tareas se reparten** entre los proveedores de recursos o servicios, llamados **servidores**, y los demandantes, llamados **clientes**.

Un **cliente realiza peticiones** a otro programa, el servidor, que le da respuesta. Esta idea también se puede aplicar a programas que se ejecutan sobre una sola computadora, aunque es más ventajosa en un sistema operativo multiusuario distribuido a través de una red de computadoras.

- ✓ Centralización del control.
 - ✓ Escalabilidad.
 - ✓ Fácil mantenimiento.
 - ✓ Existen tecnologías que aseguran la seguridad en las transacciones.
- 

- + Congestión del tráfico.
 - + Cuando un servidor está caído las peticiones de los clientes no pueden ser satisfechas.
 - + El software y el hardware de un servidor son generalmente muy determinantes.
 - + El cliente no dispone de los recursos que puedan existir en el servidor.
- 

Arquitectura cliente-servidor

Características del cliente

- Es el que inicia solicitudes o peticiones. Tiene, por tanto, un papel activo en la comunicación.
- Espera y recibe las respuestas del servidor.
- Por lo general, puede conectarse a varios servidores a la vez.
- Normalmente, interactúa directamente con los usuarios finales mediante una interfaz gráfica de usuario.



Arquitectura cliente-servidor

Características del servidor

- Al iniciarse espera a que le lleguen las solicitudes de los clientes. Desempeñan entonces un papel pasivo en la comunicación.
- Tras la recepción de una solicitud, la procesan y luego envían la respuesta al cliente.
- Por lo general, aceptan conexiones desde un gran número de clientes (en ciertos casos el número máximo de peticiones puede estar limitado).



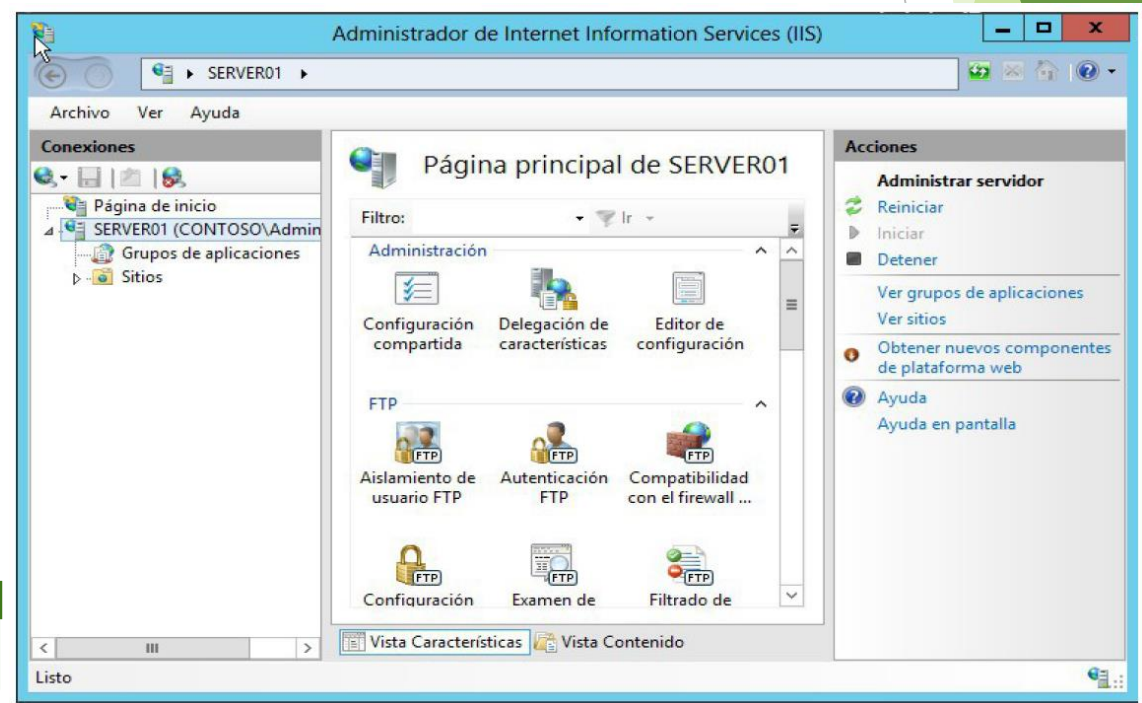
Servidor y aplicación Web

Los **servicios** son procesos, programas en ejecución, que suelen ejecutarse de forma transparente al usuario. Muchos se activan de forma automática al inicio del sistema operativo, o tras una petición del usuario, en función del rendimiento del equipo, del tráfico de la red, etc.

Un **servidor** es un ordenador o máquina informática que está al servicio de otras máquinas, ordenadores o personas llamadas clientes y que le suministran a estos todo tipo de información.

1. Servidor DHCP
2. Servidor DNS
3. Servidor FTP
4. Servidor Web
5. Servidor de correo
6. Servidor de BBDD
7. Servidor de aplicaciones
8. Servidor de archivos
9. Servidor proxy

Actividad: Instalación Servidor



Servidor y aplicación Web

El uso de **aplicaciones web** requiere:

- ❑ **Software servidor:** conjunto de herramientas software que responde a las peticiones de los usuarios que estará normalmente instalado en un servidor, ejemplo: Apache
- ❑ **Cliente:** normalmente un navegador, que puede estar ejecutándose en un PC, móvil u otra máquina
- ❑ Un **protocolo** de comunicaciones: el más usado HTTP o HTTPS
- ❑ Un **lenguaje de codificación** de la información: el más conocido HTML



Desde el navegador del PC se solicita una página web, para ello el PC envía un mensaje HTTP al servidor. El servidor busca la página en su almacenamiento y la envía en una respuesta HTTP. Esta respuesta contiene un código HTML y cuando el navegador recibe el código HTML lo interpreta y visualiza para que el usuario lo lea.

Servidor y aplicación Web

Servidor Web

Conocido con sus siglas **www (World Wide Web)** que aparecen en el nombre de prácticamente todos los servidores web, el servicio web es el servicio más utilizado de los que se ofrecen en Internet. Un servidor web se encarga de alojar y proporcionar las páginas web solicitadas por los clientes desde sus navegadores.

Un servidor web maneja el protocolo HTTP (Hypertext Transfer Protocol, Protocolo de transferencia de hipertexto). La variante segura de HTTP es el protocolo HTTPS donde la S significa Secure.

El protocolo **HTTPS protege la integridad y confidencialidad** de los datos entre el cliente y servidor.

- La integridad se refiere a que la página recibida en el cliente sea la realmente enviada por el servidor.
- La confidencialidad a que los datos enviados en la comunicación no puedan ser interceptados por viajar cifrados.

Los **servidores web más conocidos** son:

- ✓ Apache: software libre multiplataforma para Windows, Linux y MacOS.
- ✓ Nginx, también software libre multiplataforma
- ✓ IIS - Internet Information Server, servicio de Microsoft para los sistemas operativos Windows

Servidor y aplicación Web

Aplicación Web

Los clientes son los navegadores web como Mozilla Firefox, Google Chrome, Internet Explorer, Safari... Las principales diferencias entre los clientes web residen en el número e importancia de **vulnerabilidades** que presentan así como en diferentes matizaciones que existen en cuanto a la **interpretación del código HTML** y que puede impedir la correcta visualización de algunas páginas en determinados clientes. También incluyen la interpretación de scripts del lado cliente como Javascript.



Podemos decir que una **aplicación web** es una herramienta instalada en un servidor web que es utilizada a través de un navegador.

Actividad: Buscar información sobre la información de la web.
Web 1.0, Web 2.0, Web 3.0 ...

Servidor y aplicación Web

Existen muchísimos tipos de aplicaciones web, desde aplicaciones muy genéricas hasta aplicaciones muy específicas, algunos ejemplos:

- **Buscadores de internet**, ejemplo Google, el servicio nos devuelve un resultado dinámico con una relación de webs donde hay referencias a la búsqueda que hemos tecleado previamente.
- El uso masivo de los navegadores ha hecho que el **correo electrónico** se incorpore como una aplicación web que facilita la lectura del correo a través del navegador (Gmail, Outlook, Yahoo, etc.)
- Un sistema de **gestión de contenidos** es un programa que permite crear una estructura de soporte (framework) para la creación y administración de contenido, principalmente en páginas web, por parte de administradores, editores, participantes... Consiste en una interfaz que controla una o varias bases de datos donde se aloja el contenido del sitio web, manejando independiente el contenido y el diseño.
- **E-commerce**: Al igual que los gestores de contenidos permiten publicar contenidos con facilidad, estando más orientados al diseño de tiendas virtuales (**CommercePrestaShop, OpenCart**)
- **Enseñanza online**: (Learning Management System), programas desarrollados en PHP o similar que se utilizan para la formación online. El más famoso es Moodle, utilizado en el aula virtual.

Servidor y aplicación Web

- Los servicios de almacenamiento en la nube han solucionado los problemas de almacenamiento en dispositivos (coste alto y disponibilidad). Desde cualquier dispositivo con conexión a internet podemos consultar nuestros documentos originales en la nube, editarlos y tenerlos siempre actualizados en cualquier ubicación que nos encontremos. Algunas de las principales aplicaciones **Dropbox**, a **Google Drive**, **One Drive**, etc.

En general los servicios de almacenamiento suelen ofrecer:

- Varios Gigas de almacenamiento en la nube.
- Compartición de carpetas y archivos.
- Sincronización de archivos con nuestro PC, Tablet o móvil.
- Apps para acceder desde móvil y Tablet.
- Algunos admiten edición de archivos directamente desde el navegador.



**Ampliación: DWES_UT1_Cloud
Computing**

Servidor y aplicación Web

Protocolo HTTPs

HTTP (Protocolo de Transferencia de Hipertexto) es el lenguaje de comunicación que se utiliza en la Web para que los clientes y los servidores puedan entenderse entre sí.

- Escucha por el puerto TCP 80.
- Es un protocolo orientado a transacciones y sigue el esquema petición-respuesta entre un cliente y un servidor. No es un protocolo seguro, cargan información al servidor que puede ser interceptado y leído
- Para una comunicación segura a través de Internet, se utiliza el protocolo HTTP seguro (HTTPS) para acceder o subir información al servidor Web. Este protocolo puede utilizar autenticación y encriptación para asegurar los datos cuando viajan entre el cliente y el servidor. Utiliza un cifrado basado en SSL/TLS. Escucha por el puerto 443.



Ampliación: DWES_UT1_Introduccion al protocolo HTTP

Creación de aplicaciones Web

Principales elementos necesarios para poner aplicación web operativa:



- ❖ **SERVIDOR WEB:** Máquina o software capaz de **interpretar peticiones web**: hay servidores PHP, servidores .NET, servidores Java...
- ❖ **SERVIDORES DE APLICACIONES WEB:** Tomcat, Node.js
- ❖ **SISTEMAS GESTORES DE BASES DE DATOS:** MYSQL, MongoDB

Ampliación : <http://nilclass.com/courses/how-websites-work/#2>

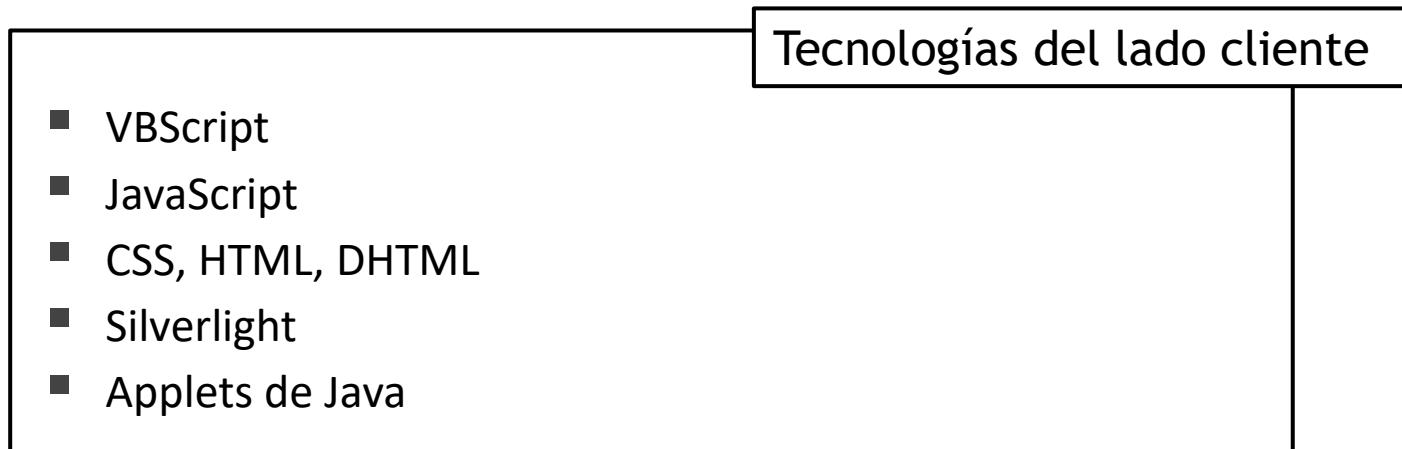
Creación de aplicaciones Web

Lenguaje de lado cliente

El intérprete que ha de ejecutar los scripts es accesible directamente desde el cliente, sin que sea necesario hacer ninguna petición al servidor.

Los usuarios tienen la posibilidad de visualizar el código fuente de los scripts de dos formas:

- ✓ De manera directa: mirando el código fuente de la página recibida
- ✓ Leyendo el contenido de fichero externos, que se encuentran generalmente en la caché del navegador.



Creación de aplicaciones Web

Lenguaje de lado servidor

La ejecución de los script se efectúa en el propio servidor antes de dar respuesta a la petición, de manera que el cliente no recibe el documento original sino el resultante de esta interpretación previa.

Cuando se utilizan este tipo de lenguajes, el cliente recibe un documento en el que cada script contenido en el original habrá sido sustituido por los resultados de su ejecución.

- ✓ Los usuarios no podrán visualizar el código fuente
- ✓ La utilización de este tipo de scripts requiere que el interprete del lenguaje sea accesible desde el propio servidor.

Tecnologías del lado servidor

- Lenguajes: Perl, Python, Ruby, Java, C#, JavaScript
- Lenguajes de script: PHP, ASP, JSP, ColdFusion
Silverlight
- Applets de Java

Creación de aplicaciones Web

Los **contenidos estáticos** son aquellos cuyo contenido no cambia, es decir, el servidor se limita a recuperar el fichero y enviarlo tal cual al cliente, y éste siempre visualiza la misma página.

Los **contenidos dinámicos** son aquellos que si pueden ser modificados (de forma automática o mediante intervención del usuario) bien sea desde el cliente y/o el servidor.

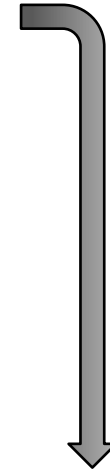
- ✓ Para que esas modificaciones puedan producirse se necesita que *algo* o *alguien* especifique **qué, cómo, cuándo, dónde y de qué forma** realizarse y que exista otro *algo* o *alguien* capaz de acceder, interpretar y ejecutar las instrucciones en el momento preciso.
- ✓ Las especificaciones y las instrucciones requieren de un **lenguaje** para definirlas, un **soporte** para almacenarlas y un **intérprete** capaz de ejecutarla.

Las aplicaciones web son páginas dinámicas, el contenido cambia, puesto que lo normal es que el usuario interactúe con el servidor

Creación de aplicaciones Web

¿cómo *sabe* el servidor si son páginas estáticas o dinámicas?

- Si en la *petición* se alude a un documento con extensión **.htm** o **.html** el servidor entenderá que esa página no requiere la intervención previa de ningún *intérprete* de *su lado* y entregará la página *tal cual*.
- Si en esa petición se aludiera a una extensión distinta –**.php**, por ejemplo– el servidor entendería que *antes de servir la página* debe *leerla* y requerir al intérprete de PHP que ejecute los scripts desarrollados en ese lenguaje (en caso de que los tuviera) y devolvería al cliente el **documento que resultara** de las eventuales ejecuciones de tales scripts.



ejemplo001.html

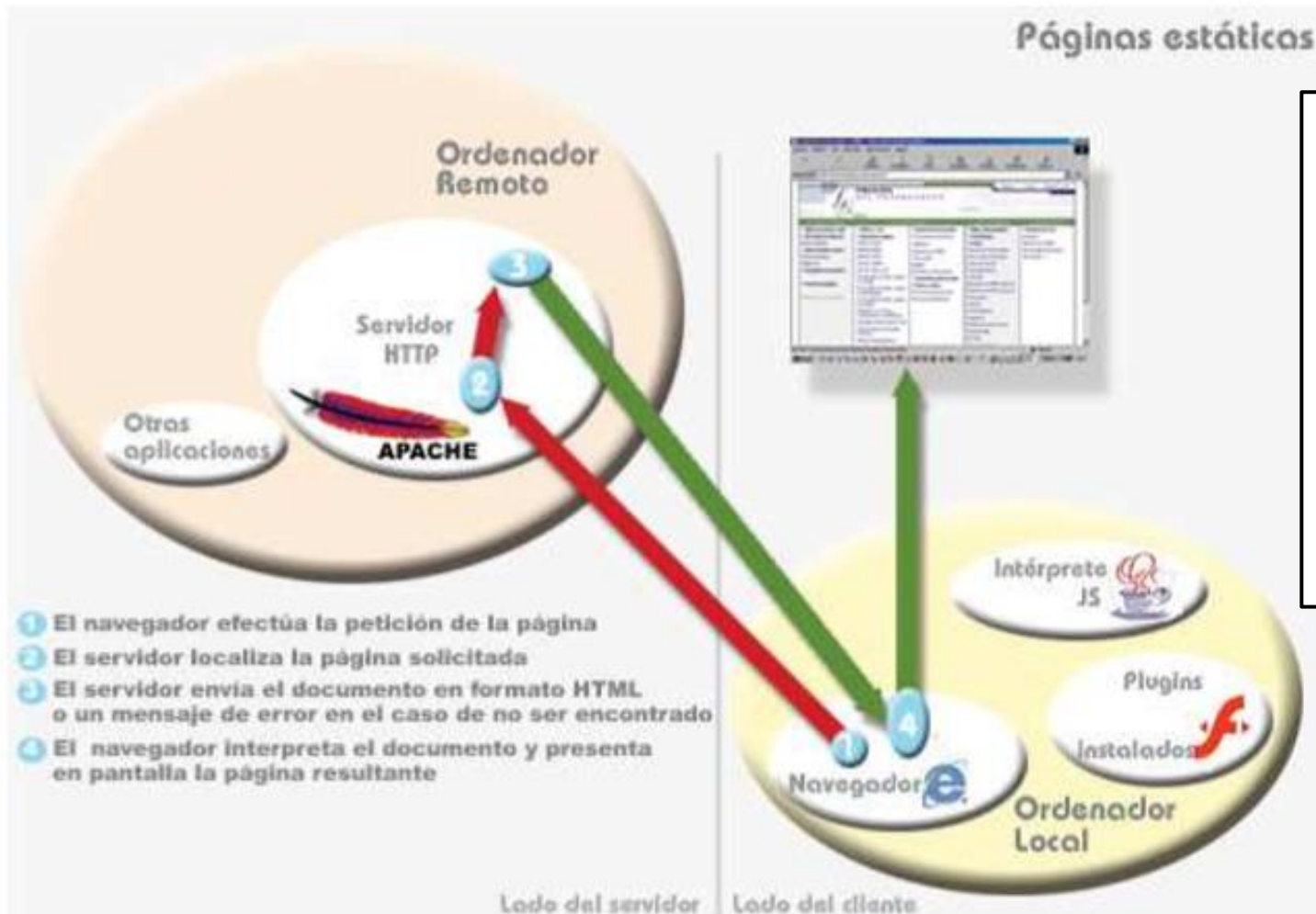
```
<HTML>
  <HEAD>
  </HEAD>
  <BODY>
    Hoy es 11-7-2005 y son las 14:23:57 horas
  </BODY>
</HTML>
```

PÁGINA ESTÁTICA

Cualquier usuario que acceda a esta página -ya sea en *modo local*, o a través de un *servidor remoto*– visualizará siempre la misma fecha: *11 de julio de 2005*.

Creación de aplicaciones Web

Funcionamiento PÁGINA ESTÁTICA



1. El navegador efectúa la petición de la página.
2. El servidor localiza la página solicitada
3. El servidor envía el documento en formato HTML o un mensaje de error en el caso de no ser encontrado
4. El navegador interpreta el documento y presenta en pantalla la página resultante

Creación de aplicaciones Web

En estos dos ejemplos verás que la fecha que aparece en la página es la *fecha actual* de tu sistema, y además, cada vez que pulses el botón *Actualizar* de tu navegador podrás comprobar que *se actualiza* la hora.

Una *intervención* del usuario *modifica los contenidos*.

NOTAS:

- En *JavaScript*, igual que **PHP**, se diferencia entre **mayúsculas** y **minúsculas**.
- `getMonth ()` devuelve el mes (de 0 a 11) en la fecha indicada, de acuerdo con la hora local.
Nota: enero es 0, febrero es uno, y así sucesivamente. Sumamos 1 para que Enero sea 1.

ejemplo002.html

<HTML>

<HEAD>

```
<script language="JavaScript">  
    var son= new Date();  
    var fecha=son.getDate()+" - "+(son.getMonth()+1)+" - "+son.getFullYear();  
    var hora=son.getHours()+":"+son.getMinutes()+":"+son.getSeconds();  
    document.write('Hoy es '+fecha+' y son las '+hora+' horas');
```

</script>

</HEAD>

<BODY>

</BODY>

</HTML>

Creación de aplicaciones Web

En este otro ejemplo la modificación de los contenidos *no requiere intervención alguna* por parte del usuario. Cada *5 segundos* (fíjate donde dice `var frecuencia=5000`. *Cinco mil* es el período de actualización, expresado en *milisegundos*) se reescribirán de forma automática *la fecha y la hora*. Tenemos un *cronómetro* automático.

[ejemplo003.html](#)

```
<HTML>
<HEAD>
<script language="JavaScript">
    var reloj=0;
    var frecuencia=5000;
    function actualiza(){
        var son= new Date();
        var fecha=son.getDate()+" - "+(son.getMonth()+1)+" - "+son.getFullYear();
        var hora=son.getHours()+":"+son.getMinutes()+":"+son.getSeconds();
        var escribe='Hoy es '+fecha+' y son las '+hora+' horas'; → 2.
        var situa=document.getElementById('capa0'); → 3.
        situa.innerHTML=escribe; → 4.
        reloj=setTimeout("actualiza()",frecuencia); } → 5.
    }
</script>
</HEAD>
<BODY onLoad="actualiza()"; >
    <div id="capa0"> </div> → 1.
</BODY>
</HTML>
```


Creación de aplicaciones Web

En este otro ejemplo la modificación de la parte del usuario. Cada 5 segundos, período de actualización, expresado en *fecha y la hora*. Tenemos un *cronómetro* [ejemplo003.html](#)

```
<HTML>
<HEAD>
<script language="JavaScript">
    var reloj=0;
    var frecuencia=5000;
    function actualiza(){
        var son= new Date();
        var fecha=son.getDate()+" - "+(son.getMonth()+1)+" - "+son.getFullYear();
        var hora=son.getHours()+":"+son.getMinutes()+":"+son.getSeconds();
        var escribe='Hoy es '+fecha+' y son las '+hora+' horas'; → 2.
        var situa=document.getElementById('capa0'); → 3.
        situa.innerHTML=escribe; → 4.
        reloj=setTimeout("actualiza()",frecuencia); } → 5.
    }
</script>
</HEAD>
<BODY onLoad="actualiza()">
    <div id="capa0"> </div> → 1.
</BODY>
</HTML>
```

→ 2. Recogida de la fecha y la hora en la variable escribe

```
var escribe='Hoy es '+fecha+' y son las '+hora+' horas';
```

→ 3. Guarda una referencia del elemento HTML identificado por 'capa0' en situa

Al elemento o método getElementById se accede a través del objeto document.

```
var situa=document.getElementById('capa0');
```

→ 4. innerHTML es una función capaz de cambiar el texto de un contenido HTML

```
situa.innerHTML=escribe;
```

→ 5. Llamo de forma recursiva a la función actualiza ()

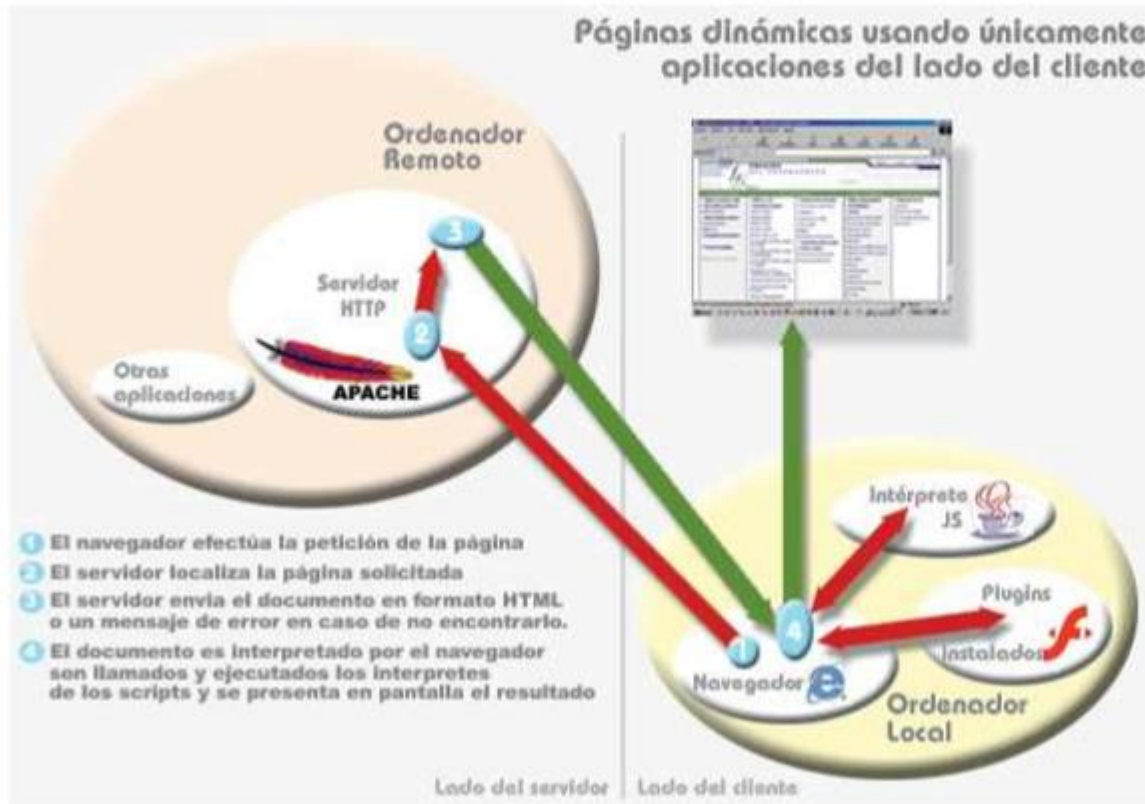
```
reloj=setTimeout("actualiza()",frecuencia); }
```

→ 1. Acceso desde HTML a la función a través de un id

```
<div id="capa0"> </div>
```

Creación de aplicaciones Web

Funcionamiento PÁGINA DINÁMICA usando únicamente aplicaciones del lado del CLIENTE



1. El navegador efectúa la petición de la página.
2. El servidor localiza la página solicitada
3. El servidor envía el documento en formato HTML o un mensaje de error en caso de no ser encontrado
4. El documento es interpretado por el navegador, son llamados y ejecutados los interpretes de los scripts y se presenta en pantalla el resultado.

Funcionamiento PÁGINA DINÁMICA usando aplicaciones del lado del CLIENTE y del lado del SERVIDOR



1. El navegador efectúa la petición de la página.
2. El servidor llama al intérprete de PHP si es necesario.
3. PHP ejecuta los scripts (interactuando con la BD si es preciso) y devuelve al servidor el documento generado.
4. El servidor envía el documento resultante en formato HTML.
5. El documento es interpretado por el navegador, se ejecutan los scripts del lado cliente y se presenta el resultado en pantalla.

Creación de aplicaciones Web

FECHA02.PHP

```
<?php
```

```
    setlocale(LC_TIME,'esp');
```

```
        //setlocale(LC_TIME,'Spanish'); //en Windows
```

```
$fecha = strftime("%A, %d de %B del %Y. A las %H:%M:%S");
```

```
echo 'Ahora: '. $fecha . "<br>";
```

```
$fecha2 = strftime("%A, %d de %B del %Y. A las %H:%M:%S",time() + (7* 24 * 60 * 60));
```

```
echo 'Próxima Semana: '. $fecha2 ;
```

```
?>
```

1ª EJECUCIÓN:

Ahora: martes, 27 de septiembre del 2011. A las 19:42:16

Próxima Semana: martes, 04 de octubre del 2011. A las 19:42:16

2ª EJECUCIÓN:

Ahora: martes, 27 de septiembre del 2011. A las 19:42:55

Próxima Semana: martes, 04 de octubre del 2011. A las 19:42:55

%A → nombre del día de la semana

%d → el día del mes con dos dígitos

%B → nombre del mes

%Y → año con 4 dígitos

Creación de aplicaciones Web

Funcionamiento PÁGINA DINÁMICA usando aplicaciones del lado del CLIENTE y del lado del SERVIDOR

