**TEMA 1: IMPLANTACIÓN DE ARQUITECTURAS WEB**

**1. Introducción**

La World Wide Web (www) es un universo de información interconectada a través de internet propuesta por Tim Berners-Lee. Triunfó gracias a que estaba basado en HTML, un lenguaje de hipertexto e hipermedia.

**1.1 Componentes semánticos de la Web**

**URI (Uniform Resource Identifier):**

Identifica de forma única los recursos web para su acceso y manipulación. Tipos:

* URN (Uniform Resource Name): Identifica de forma única el recurso independientemente de donde resida. Está en fase experimental.
* URL (Uniforme Resource Locator): Identifica donde se encuentra el recurso. Está compuesto de: Esquema, Servidor y Nombre del recurso

Sintáxis de la URL:

**esquema://[usuario]:[password]@<maquina>:[puerto]/<camino>;[parametros]?[consulta]#[sección]**

* Esquema: Protocolo (http, ftp…).
* Usuario:password: Login.
* Máquina: Nombre del servidor.
* Puerto: Número del puerto que escucha el servidor.
* Camino: Directorio virtual y nombre del recurso.
* Parámetros: Parejas de nombre=valor utilizados en algunos esquemas.
* Consultas: Parejas de nombre=valor separadas por & cuando se realizan consultas.
* Sección: Nombre de parte del recurso.

La URL se ha diseñado para ser portable. El tratamiento de los caracteres de la URL es el siguiente:

* Los números y letras ASCII se dejan intactos.
* Los espacios en blanco se sustituyen por +.
* Los caracteres especiales y los que tienen un significado dentro de la url se sustituyen por su valor hexadecimal con el prefijo %.

**HTML (HyperText Markup Language):**

Lenguaje de marcas que provee una representación estándar de documentos hipertexto en formato ASCII. Permite darle formato al texto, añadir imágenes, enlazar otros documentos, etc…

**HTTP (Hypertext Transfer Protocol)**

Protocolo que define el formato y el significado de los mensajes intercambiados entre los componentes web.

**2. Componentes software de la web**

La arquitectura cliente servidor implica uno o más clientes solicitando servicios a uno o más servidores. Además, existirán una serie de elementos que interactuarán con ambos componentes: Proxies, cachés…

Este sistema permite mayor usabilidad, flexibilidad, interoperabilidad y escalabilidad

**2.1 Clientes**

Son los que originan el tráfico web, solicitando y recibiendo respuestas del servidor. Tipos:

**Navegadores**

Programas en los que los usuarios realizan peticiones cuando navegan por un site. Funciones:

* Construyen y mandan la petición por http.
* Reciben, analizan, y presentan las respuestas. El resultado final dependerá de la configuración del navegador.
* Proporcionan interfaz para utilizar servicios de otros protocolos: ftp, mail, etc…
* Sirve recursos guardados en la caché sin conectarse al servidor. La consistencia puede ser fuerte (revalida siempre) o débil (revalida según los parámetros http)
* Maneja las cookies

**Spiders**

Robots dedicados a la búsqueda automática de información, que es utilizada posteriormente en aplicaciones de búsqueda (Google).

No indexan los recursos dinámicos (CGI, PHP) ni tampoco aquellos webs que lo impidan (Puede hacerse a través de una etiqueta META).

**2.2 Servidores**

Los más populares son Apache e IIS. Son programas que generan la respuesta HTTP a la petición del cliente. Pasos:

* Conecta con el cliente
* Recibe el mensaje HTTP
* Procesa el mensaje HTTP
* Localiza y envía la respuesta por HTTP.

Los servidores de altas prestaciones tratan múltiples conexiones, gestionan la caché, y pueden generar contenido dinámicamente a través de ASP, PHP, JSP…

Los servidores proveen acceso a recursos estáticos y dinámicos. Pasos:

* Lee e interpreta el mensaje de petición.
* Localiza el recurso apuntado en la URL.
* Determina si el cliente está autenticado y autorizado.
* Genera la respuesta y la transmite.
* Registra la operación en un log.

En los recursos dinámicos se incluye código que debe ser ejecutado para resolver el contenido de la respuesta. Tipos:

* Scripts de servidor (PHP, ASP, JSP): Ficheros HTML que incluyen macros interpretadas por el servidor.
* Programas independientes del servidor (CGI, Servlets)

Algunos sitios están compuesto por varias máquinas servidoras que actúan como un único servidor:

* Servidor frontal: Direcciona las peticiones, la caché, balancea la carga…
* Servidor de contenido: Contenido diferenciado según el tipo de petición.
* Servidor de base de datos: Información utilizada para el contenido dinámico.

**2.3 Intermediarios web**

**Proxy**

Programa que actúa a la vez como cliente y servidor, recogiendo las peticiones del cliente y enviándolas al servidor.

Su objetivo es reducir la comunicación no deseada. Suelen actuar como parte del mecanismo de seguridad y como cachés. Funciones:

* Guarda las respuestas en caché y comparte accesos con otros clientes.
* Filtra las peticiones y respuestas
* Esconden la ip del cliente
* Ajustan las peticiones y las respuestas para ajustarlas a las capacidades del servidor y del cliente respectivamente.
* Actúan como pasarelas a otros sistemas no-http.

Caches

Cuando un proxy actúa como cache, gestiona las peticiones de los clientes. A diferencia de otros proxies, no todas las peticiones llegan al servidor. Proceso:

* El navegador envía la petición a la caché.
* La caché devuelve el recurso si está almacenado localmente y actualizado.
* Si no lo está, pasa la solicitud al servidor.
* Si está pasado de fecha, valida el recurso.

Las principales ventajas de usar la caché son:

* Reduce el uso de ancho de banda en la red
* Reduce los tiempos de espera
* Descarga al servidor, al resolver peticiones directamente.

Por lo tanto, aumentan un 50% la velocidad de respuesta y un 40% la cantidad de bytes transmitidos.

**Pasarela**

Servidor que actúa como intermediario de otro servidor. Recibe las peticiones como si fuese el servidor original, por lo que el cliente no tiene que preocuparse.

**Túnel**

Programa que permite tráfico no-http en conexiones http. Es activado por la petición http, aunque no forma parte de la comunicación http y se cierra cuando esta termina.

**3. Arquitectura de las aplicaciones web**

Una aplicación web es proporcionada por un servidor y utilizada por usuarios que se conectan desde un navegador. Características:

* Se basan en el modelo cliente/servidor y utilizan como interfaz páginas web.
* Son en gran parte dinámicas y están agrupadas para dar servicio al usuario.
* El acceso a las páginas está agrupado en el tiempo (sesión).

Componentes:

Lógica de negocio: Define los procesos de la aplicación Es la parte más importante.

Administración de los datos.

Interfaz: Es la funcionalidad accesible a través del dispositivo por el que accede el usuario: Navegadores, móviles, PDAs, etc…

**Modelo de capas**

El modelo de capas modeliza la estructura de una aplicación web. Una capa representa un elemento que procesa o trata información.

Modelo de dos capas:

Gran parte de la aplicación corre del lado de cliente, conocido como fat client.

El cliente soporta tanto la lógica de negocio como la interfaz, mientras que el servidor solo administra los datos. Limitaciones:

* Difícilmente escalable.
* Alta carga de la red.
* Flexibilidad restringida.
* Funcionalidad limitada.

Modelo de tres capas:

Es el modelo más habitual, y fue diseñado para superar las limitaciones del modelo de dos capas.

Introduce una capa intermedia, la capa de proceso, la cual centraliza la lógica de negocio. Puede integrar datos de múltiples fuentes, y las aplicaciones web actuales se ajustan a este modelo.

Capas:

* Capa de presentación: Recoge la información del usuario y la envía a la capa de proceso, para posteriormente recibir los resultados de la capa de proceso y visualizarlo en el cliente.
* Capa de proceso (Servidor web): Recibe los datos de la capa de presentación e interactúa con la capa de datos para realizar operaciones. Finalmente envía los resultados a la capa de presentación.
* Capa de datos (Servidor de datos): Almacena, recupera, mantiene y asegura la integridad de los datos.