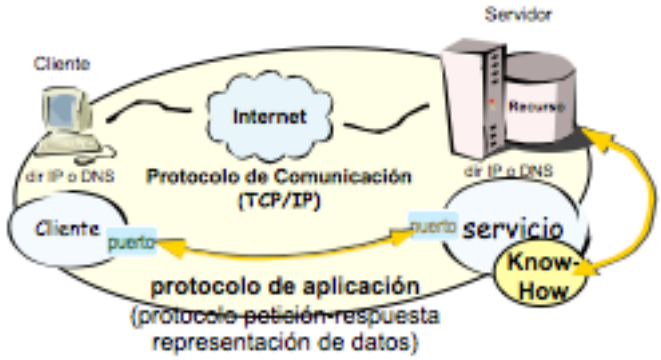
Tema 2. TECNOLOGÍAS WEB Y MIDDLEWARE

TECNOLOGÍAS WEB Y MIDDLEWARE

TECNOLOGÍAS WEB. MODELOS WEB BÁSICOS.

MODELO CLIENTE/SERVIDOR



Http

Protocolo de transferencia de hipertexto, usado para la comunicación entre un navegador y un servidor web.

Cada petición de cliente especifica un método que deberá ser aplicado en el servidor. Estos son los diferentes métodos:

- **Método GET**: pide el recurso cuyo URL le da como argumento. El servidor siempre devolverá el mismo tipo de datos enviados en la petición URL del cliente.
- **Método head**: petición idéntica a get, pero no devuelve datos, simplemente devuelve toda la información sobre los datos.
- **Método Post**: está diseñado para proporcionar un bloque de datos a un proceso de gestión de datos. Especifica el URL de un recurso que puede tratar los datos aportados con la petición.
- **Método Pat**: indica que los datos aportados en la petición deben ser almacenados con la URL aportada como su identificador.
- **Método Delete**: el servidor borrará el recurso identificado por el URL. No siempre se permite.
- **Método Options**: el servidor proporciona al cliente una lista de métodos aplicables a una URL y sus requisitos.
- **Método trace**: el servidor envía de vuelta el mensaje de petición. Se utiliza para depuración.

http devolverá unos códigos de respuesta:

- **1xx** : mensajes de información.
- **2xx** : operación exitosa.
- **3xx** : redirección hacia otra URL.
- **4xx** : error por parte del cliente.
- **5xx** : error por parte del servidor.

Para el mantenimiento de sesión en http podemos utilizar algunos mecanismos como por ejemplo:

- Del lado del servidor: ficheros , bases de datos y sistema centralizado (sobrecarga).
- Del lado del cliente: campos ocultos y cookies.

Cookies

Son pequeñas partes de información que se almacenan en el disco duro del visitante a través de su navegador, a petición del servidor de una página.
Tiene la ventaja de que se almacena en el usuario y que tienen fecha de caducidad para evitar la sobrecarga.

Una cookie no identifica a una persona sino a una combinación de pc y navegador.
Es importante para una cookie conseguir información de hábitos de navegación de usuario.

El funcionamiento de una cookie se basa en que cuando un cliente realiza una petición al servidor, éste le envía junto a la respuesta información en la cabecera http para la creación de cookies en el cliente, mediante set-cookie. Para acabar la sesión, el servidor debe establecer una cookie de duración 0.

Campos ocultos

Los valores de estos elementos deben ser transferidos mediante atributos value y normalmente contienen datos para transferir el estado de sesión.

MIME (Multipurpose Internet Mail Extension)

Es un estándar para enviar mensajes de correo electrónico compuestos por varias partes contendiendo a la vez texto, imágenes y sonido.

Los recursos considerados como datos se proporcionan en forma de estructuras de tipo MIME, tanto en los argumentos como en los resultados.
Los datos van precedidos por su tipo de MIME, para que el receptor pueda saber gestionarlos.

MODELO CGI (Common Gateway Interface)

CGI es un protocolo que proporciona una interfaz o pasarela entre un servidor de información y un proceso externo.

Un cliente web puede especificar un programa conocido como script CGI como objeto web de destino de una petición http.

El servidor web llama al script y lo activa como un proceso, pasándole los datos de entrada transmitidos por el cliente web.

El script web ejecuta y transmite su salida al servidor web, que devuelve los datos generados por dicho script.

Ventajas del uso de CGI:

- capacidad de respuesta dinámica.
- Libre elección del lenguaje de programación.

Desventajas del uso de CGI:

- no hay relación entre programa CGI y servidor web.
- No hay control sobre la ejecución.
- Nueva instanciación por cada solicitud.
- Sobrecarga de recursos.

TECNOLOGÍAS WEB. AMPLIACIONES

Los conceptos para que aparezca el término ampliación son:

- Ampliación del concepto WEB como escaparate o catálogo estático.
- Mejor gestión de recursos y mayor organización.
- Aparición de numerosas tecnologías.

APLICACIÓN WEB

Aquella que está basada en arquitectura C/S son cliente->navegador, servidor->servidor web , utilizando protocolos de Internet (TCP/IP) para su entendimiento.

Ampliaciones en el cliente.

Scripts en el cliente: técnica utilizada para dotar de dinamismo las páginas web.

- Ventajas: máxima fiabilidad, control total sobre el aspecto final y funcionalidad asegurada.
- Desventajas: pérdida de compatibilidad, imposibilidad de obtener Know-how debido a la plataforma o versión no valida.

Ampliaciones en el servidor

Servlets de Java.

Programas que se pueden invocar desde el cliente de forma similar a los CGI. El servidor web actúa como contenedor de contenedores. Realizan una tarea que se recoge en el servidor web y luego es enviada al navegador. Características:

- Portabilidad: sólo precisa motor JVM.
- Rendimiento: única instanciación.

- Sesión: mantiene información entre diferentes conexiones al servlet.
- Software distribuido: recuperación de disco/servidor de servlets. Comunicación entre sí.
- Multithread: múltiples peticiones concurrentemente.

Modelo de extensiones (ISAPI/NSAPI)

Ventajas:

- librerías de acceso dinámico: sólo se instancian una vez, ganando rendimiento y aprovechamiento de los recursos.

Desventajas:

- tecnología cerrada a fabricante y servidor web: ISAPI/Microsoft/IIS o NSAPI/Netscape/NES.

Modelo de páginas activas

Es un mecanismo muy extendido. Se basa en fragmentos de código insertados en páginas html que el servidor se encarga de interpretar antes de servir dichas páginas. Se realiza con el fin de dinamizar las páginas.

Esta es la forma de desarrollar más común actualmente, en parte por la amplia alternativa de lenguajes dependientes para utilizar con la plataforma.

Ventajas:

- habilita el concepto de sesión de usuario.
- Buena gestión de recursos externos.
- Control sobre protocolo petición/respuesta.
- Operaciones ejecutadas en servidor.

Desventajas:

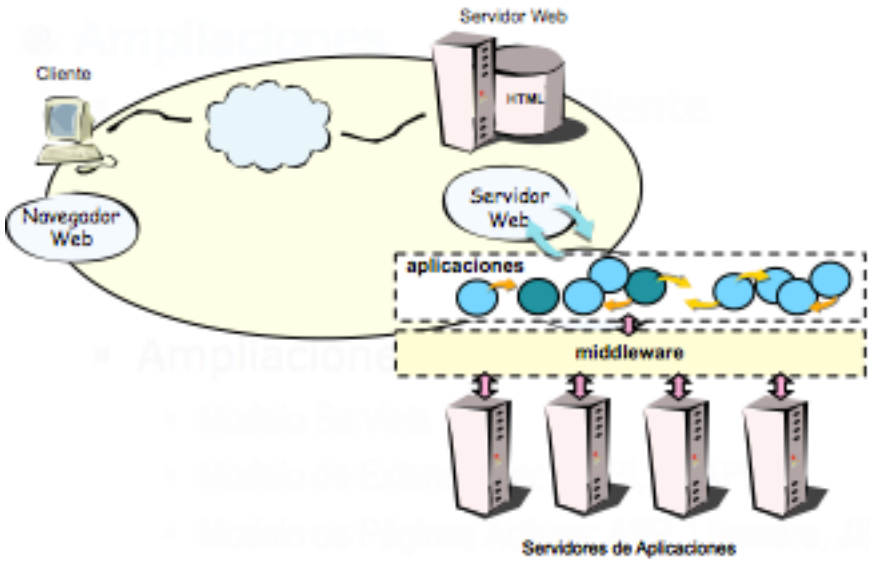
- necesidad de conocer lenguaje script.
- Microsoft/ASP/VBScript.
- NETscape/Liveware/Javascript.
- Sun Microsystems/JSP/Java Script.

Servidor de aplicaciones: Middleware

El middleware aparece ante la necesidad de gestionar grandes aplicaciones sobre entornos distribuidos altamente heterogéneos. Introducción de un nivel de abstracción que aporte independencia con respecto a la plataforma sobre la que se ejecutan las aplicaciones. ---Definición de Middleware

El objetivo es eliminar la carga del servidor web, colocándose detrás del mismo. Permite utilizar tecnologías vistas, con un mayor control para la asignación de recursos. El funcionamiento es el siguiente:

- Peticiones sobre contenidos estáticos: resueltas por el propio servidor Web.
- Solicitudes de contenidos dinámicos: delega en el servidor de aplicaciones (Middleware).



Modelo mixto

AJAX: tecnología de desarrollo web en conjunto con otras tecnologías existentes. Aplicaciones ricas por parte del cliente, con un buen aspecto.

Se basan en la ejecución en el cliente (JavaScript). Conexión asíncrona con el servidor (XML).

MIDDLEWARE

Se define como el nivel lógico del sistema, que proporciona una abstracción en términos de servicios cuya finalidad es proporcionar una visión única del sistema, independiente de la infraestructura que lo forme. Abstrae de la heterogeneidad y complejidad de las redes de comunicaciones, SO y lenguajes de programación.

Punto de vista de la arquitectura

Aplicaciones en términos de componentes. Los elementos fundamentales para este punto de vista son:

Modelo de representación de datos.

Las estructuras de datos y los atributos de los objetos deben ser convertidos en una secuencia de bytes antes de su transmisión y reconstruirse después en el destino. Para que dos computadores intercambien información deben acordar previamente un esquema común, al que se le denomina representación externa de datos.

Modelo de comunicación.

El protocolo petición-respuesta define el nivel adecuado para establecer una comunicación típica según la arquitectura cliente/servidor. El propio mensaje es la petición, para evitar sobrecargas. Sobre esta capa se realiza la llamada a procedimientos y la invocación remota de métodos.

Punto de vista del programador

Modelo de componentes

Este modelo ofrece reutilización de código. Transparencia con respecto a la plataforma sobre la que se ejecutan y con respecto al lenguaje de programación.
Capacidad de personalización a través de las propiedades.
Comunicación transparente entre ellos y con el contexto mediante eventos.
Los tipos de componentes son:

- Cliente
- Servidor: encapsulado de servicios o almacén de datos.

J2ee

Es una plataforma middleware basada en java que ofrece 3 tipos de tecnologías para el desarrollo de componentes:

- Componentes WEB: responden a una solicitud http.
    - servlets: programas java que componen páginas web.
    - JSP: páginas web que contienen código en java.
- Componentes Enterprise JavaBean(EJB): modelo de componentes distribuidos. Unidades de software reusables que contienen lógica de empresa. Tipos:
    - Beans de sesión: ejecuta solicitudes del cliente y es destruida cuando se completan. Se mantiene el estado entre solicitud y solicitud, aunque es opcional.

- Beans de entidad: objeto persistente que modela los datos de un almacén.
- Beans dirigidos por mensaje: no son invocados por el cliente y procesan mensajes asíncronos.

J2ee establece 4 tipos de contenedores:
- Contenedor Web: para servlets y JSP.
- Contenedor EJB: Components Enterprise JavaBean.
- Contenedor Applet: para aplicaciones java ejecutadas en navegadores.
- Contenedor de aplicaciones cliente: para aplicaciones java estándar.

SERVICIOS WEB

Componentes que ejecutan procesos o funciones de negocio, con una gran interfaz accesible a través de Internet, basada en el intercambio de documentos electrónicos en formato XML y que pueden ser combinados entre sí.

El paradigma utilizado para el desarrollo de sistemas distribuidos es B2B. Proporciona sistemas débilmente acoplados. Permite reutilización y composición. Interoperabilidad basada en contratos bien definidos.

La comunicación entre los negocios a nivel de aplicaciones mediante la utilización de estándares abiertos como XML, SOAP, UDDI y WSDL.

SOAP (simple object Access protocol)

Protocolo que extiende http para permitir acceso a objetos distribuidos que representan servicios web. Incorpora la arquitectura de los objetos distribuidos y los protocolos de Internet.

El modelo para el protocolo simple de acceso a objetos es el siguiente:
Un cliente web realiza una petición http, cuyo cuerpo del mensaje es una llamada a un método de un objeto de servicio SOAP. La petición se transmite a un servidor web, que la reenvía junto con los parámetros de la llamada al método. Se ejecuta el método. Una vez completado se devuelve el valor del método al servidor web, el cual le transmite al cliente web en el cuerpo del mensaje una respuesta http.

El mensaje se codificará en XML y se transporta en peticiones o respuestas http.

Mensajes SOAP (XML (eXtensible Markup Language))

Solución para el intercambio de datos de una forma transparente.

Sobre o envoltura: define un marco de referencia general para expresar qué hay en el mensaje, quién debe atenderlo y si es opcional u obligatorio. Identifica un mensaje XML como SOAP.

Reglas de codificación: definen un mecanismo de serialización que se puede utilizar para intercambiar instancias de tipos de datos definidos para la aplicación.

Representación RPC/Document: define una convención que puede ser utilizada para representar las llamadas y respuestas a procedimientos remotos.

Hay dos partes en el cuerpo de un mensaje SOAP:

- **Recubrimiento SOAP**: definido por el elemento <SOAP-ENV: envelope>, con un conjunto de atributos requeridos que especifican el esquema de codificación y el estilo del recubrimiento.
- **Cuerpo SOAP**: definido por la etiqueta <SOAP-ENV: body>. El cuerpo sólo contiene un elemento que es la llamada del método con sus parámetros.

UDDI (Universal Description, Discovery and Integration)

Nos permite conocer con quién comunicarse y dónde.
Pasos para consumir un recurso UDDI:

1. Publicación del servicio web en el registro UDDI.
2. Búsqueda del cliente en el registro UDDI del servicio web.
3. Descubrimiento por parte del cliente desde el registro UDDI del servicio web.
4. Consumo del servicio web por parte de la aplicación web.

Un registro UDDI contiene 3 secciones conceptuales:

- **Blanca**: similar a la información que aparece en el directorio telefónico, que incluye nombre, teléfono y dirección.
- **Amarilla**: similar a su equivalente telefónico. Incluyen categorías de catalogación industrial tradicionales, ubicación geográfica, etc.
- **Verde**: información técnica acerca de los servicios ofrecidos por los negocios.

La estructura central de un registro UDDI:

- **businessEntity**: información sobre un negocio o entidad. Utilizada por el negocio para publicar información descriptiva sobre si mismo y los servicios que ofrece.
- **businessService**: servicios o procesos de negocios que provee la estructura businessEntity.
- **bindingTemplate**: datos importantes que describen las características técnicas de la implementación del servicio ofrecido.
- **tModel**: especificación y categorización técnica.

Entre las características de UDDI encontraremos dos categorías de API:

- **De publicación**: mecanismo para que los proveedores de servicios se registren en el registro UDDI.
- **De consulta:** permite a los suscriptores buscar los servicios disponibles y obtener el servicio una vez localizado.

WSDL (Web Service Description Language)

Documento XML para describir servicios Web. Descripción abstracta del servicio (interfaz pública). Descripción del protocolo a utilizar, con implementación concreta.

WSDL describe los mensajes SOAP que definen un servicio web en particular. Utiliza un IDL(Interface Definition Language) de servicios.

Los registros UDDI apuntan a una hoja WSDL.

Anatomía de un documento WSDL

<definitions> : contiene la definición de uno o más servicios.
<message> y <portType> : qué operaciones provee el servicio.
<binding> : cómo se invocan las operaciones.
<service> : dónde se ubica el servicio.
<documentation> : puede contener información del servicio para el usuario.

CONCLUSIONES

Hay una gran diversidad de tecnologías alrededor de la web, cada una con características idóneas para cada modelo de componentes y plataforma middleware.
Idoneidad del modelo web para gestionar la capa de presentación (B2C).
Necesidad de conocer tecnologías y herramientas para seleccionar adecuadamente.
Idoneidad del modelo de servicios Web para gestionar la interacción B2B.