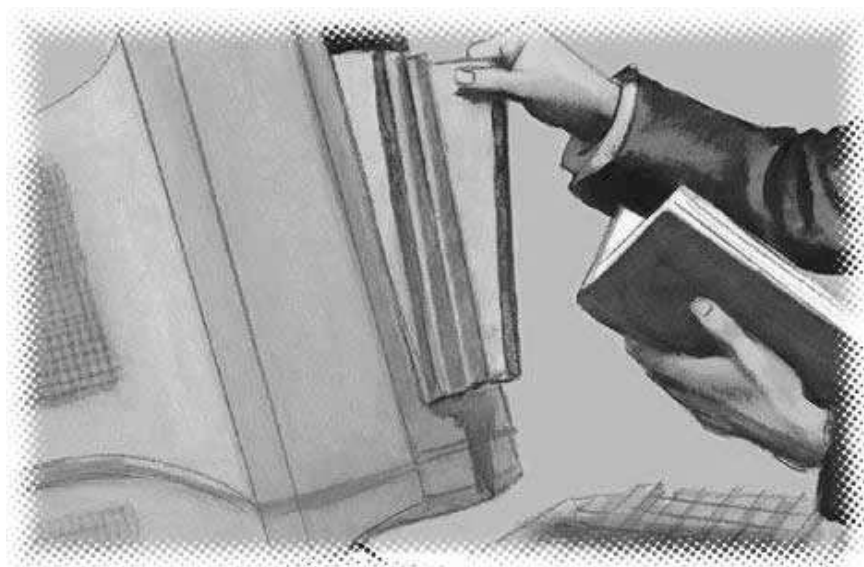




CAPÍTULO 2

Marco teórico





CAPÍTULO 2 MARCO TEÓRICO

2.1 Comercio electrónico y aplicaciones Web

2.1.1 Comercio electrónico

Como se mencionaba en la introducción de la presente tesis, gracias al desarrollo de las nuevas Tecnologías de la Información y Comunicación (TICs), las distancias y el tiempo entre las transacciones de proveedores y consumidores, así como otros factores involucrados en el comercio, han ido mejoradas, siendo esta actividad una de las más importantes en el tráfico de Internet. De esta manera la Web ofrece un nuevo mercado que define lo que ahora es llamado como la "economía digital".

De acuerdo a la OECD (*Organisation for Economic Co-Operation and Development, Organización para la Cooperación y el Desarrollo Económicos*) el Comercio Electrónico es definido como: el proceso de compra, venta o intercambio de bienes, servicios e información a través de las redes de comunicación.

La anterior definición aparentemente es llana sin embargo define lo realmente necesario para comprender el comercio electrónico, es decir, el comercio electrónico no es una nueva tecnología sino una metodología moderna para hacer negocios en la que se sule esa necesidad de las empresas, comerciantes y consumidores por reducir costos, mejorar la calidad de los bienes y servicios, además de mejorar el tiempo de entrega de los bienes o servicios con el uso de la tecnología. Implica además cualquier transacción comercial en la cual las partes involucradas interactúan de manera electrónica en lugar de hacerlo de la manera tradicional con intercambios físicos o trato físico directo.

Es así como el comercio electrónico es el medio para llevar a cabo los cambios tanto en organización como en operaciones en los negocios, dentro de una escala global, logrando así que las compañías sean más eficientes y flexibles en sus operaciones



internas, para así trabajar de una manera más cercana con sus proveedores y estar más pendiente de las necesidades y expectativas de sus clientes.

En la actualidad las empresas ya están comenzando a usar Internet como un nuevo canal de ventas, sustituyendo las visitas personales, correo y teléfono por pedidos electrónicos, ya que gestionar un pedido por Internet cuesta 5% menos que hacerlo por vías tradicionales. Es así como surge entonces el comercio electrónico, como una alternativa de reducción de costos y una herramienta fundamental en el desempeño empresarial.

Asimismo surgen también algunos nuevos inconvenientes relacionados con el comercio electrónico que van desde la validez legal de las transacciones y contratos sin papel, la necesidad de acuerdos internacionales que armonicen las legislaciones sobre comercio, el control de las transacciones internacionales, incluido el cobro de impuestos; la protección de los derechos de propiedad intelectual, la protección de los consumidores en cuanto a publicidad engañosa o no deseada, fraude, contenidos ilegales y uso abusivo de datos personales, hasta otros provocados por la dificultad de encontrar información en Internet, comparar ofertas y evaluar la fiabilidad del vendedor y del comprador en una relación electrónica, la falta de seguridad de las transacciones y medios de pago electrónicos, la falta de estándares consolidados, la proliferación de aplicaciones y protocolos de comercio electrónico incompatibles y la congestión de Internet.

En México se necesita de una normativa jurídica en relación al comercio electrónico y el formato digital. En un contexto mundial, nuestro país se encuentra desactualizado, implicando esto una toma de medidas apremiantes respecto a esto ya que en el país a pesar de esto el comercio electrónico se ha incrementado en forma considerable.

De acuerdo a los estudios hechos por la AMIPCI (***Asociación Mexicana de la Industria Publicitaria y Comercial en Internet***) mejor conocida como Asociación mexicana de Internet, tan solo para el año 2009 las ventas por Internet representaron el



16% del total de ventas (Figura 2.1.1.1). Estos estudios fueron basados en las empresas que operan en México de gran diversidad de giros que van desde las ventas de boletos de avión, artículos de cómputo, espectáculos, cine, accesorios para celulares, electrónica, hospedajes, viajes, etc., y cuyos movimientos mercantiles se encuentran tanto en el sector presencia física como on-line.

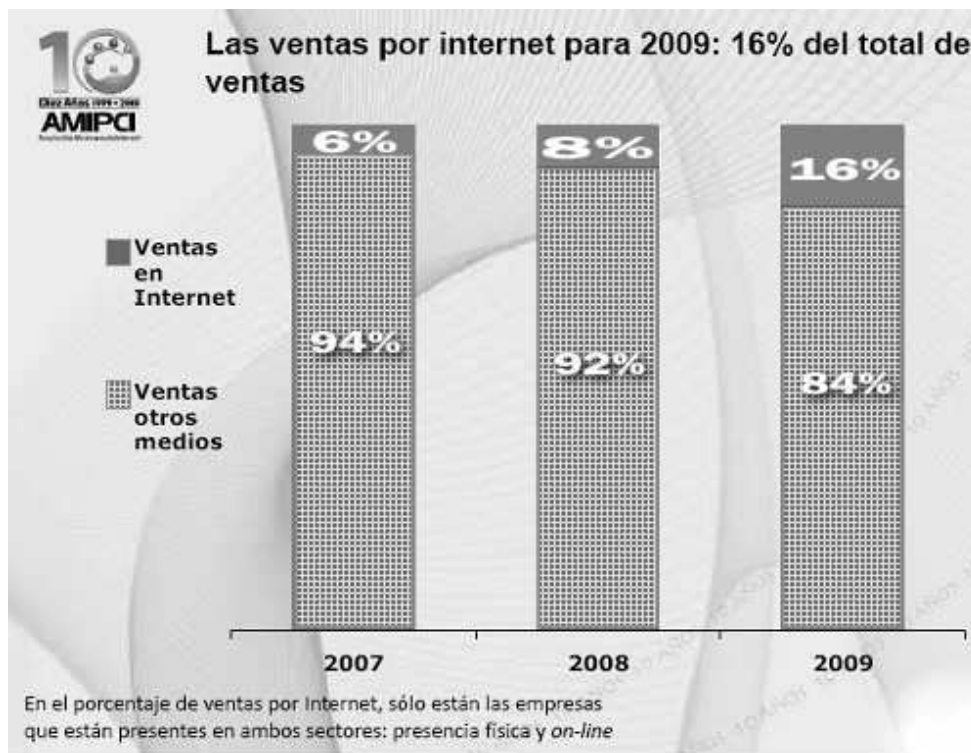


Figura 2.1.1.1 Porcentaje de ventas por Internet en 2009

En México el comercio electrónico se enfoca en primer lugar a los portales de cómputo (44%), en segundo lugar las aerolíneas (37%) y en tercer lugar los portales de hogar y electrodomésticos (33%).

De esta manera el crecimiento esperado del comercio online se ha superado (ver figura Figura 2.1.1.2)



Figura 2.1.1.2 Se superan expectativas de crecimiento

Como también se menciona en la introducción de la Tesis con el surgimiento de la tarjeta de crédito la venta por catálogo en línea tomó un mayor impulso y hoy en día según los estudios de la AMIPCI la tarjeta de crédito (Fig. 2.1.1.3) continúa siendo el medio de pago preferido.



Figura 2.1.1.3 La tarjeta de crédito sigue siendo el medio de pago preferido.

(www.amipci.org.mx)



Con el incremento del ancho de banda y el uso de los dispositivos móviles se ha impulsado el comercio electrónico. Con la combinación del crecimiento en el acceso de banda ancha y el uso de dispositivos móviles cada vez más inteligentes como los teléfonos 3G, se espera que las ventas en el comercio electrónico aumenten significativamente pronto.

Finalmente al mismo tiempo surgen también los derechos de los consumidores que cada vez demandan más confianza por parte de los sitios Web y que al momento de efectuar una compra solicitan los que posean algún sello de confianza por alguna entidad reconocida, los consumidores se inclinan por aquellos sitios recomendados o que gocen de buena reputación.

A continuación se muestra el ciclo del comercio electrónico.

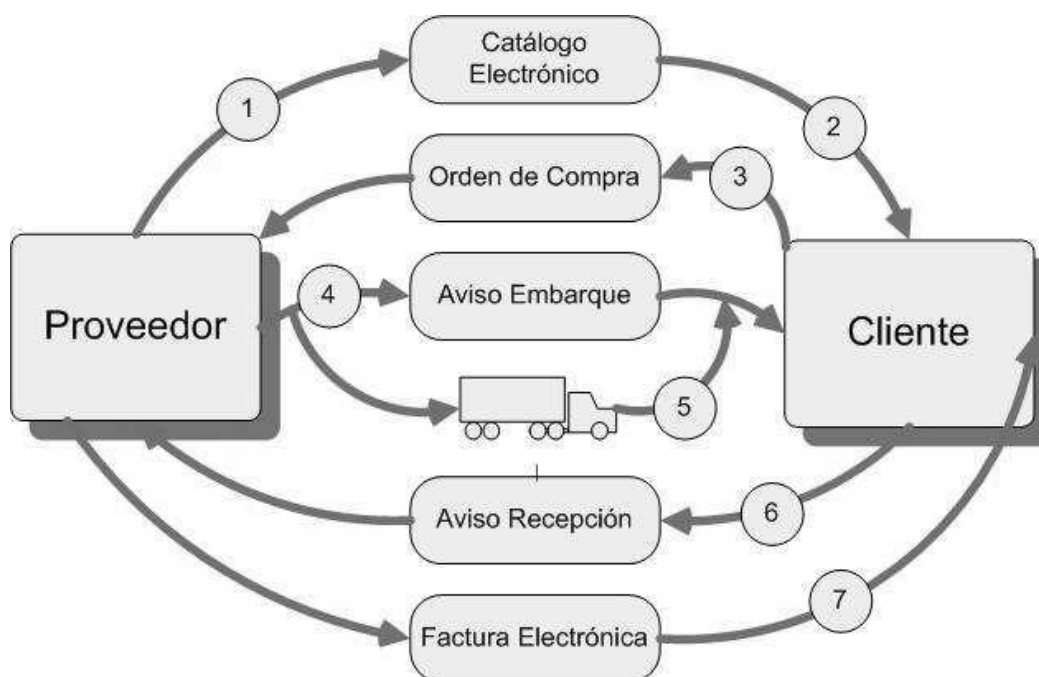


Figura 2.1.1.4 Ciclo del comercio electrónico



Considerando la figura 2.1.1.4, se muestra como primera instancia del ciclo del comercio electrónico la información que ofrece el proveedor al cliente mediante un sistema de información bien estructurado como un catálogo electrónico que ofrezca ayuda para la decisión de compra “remota”, después de haber tomado la decisión el cliente efectúa su orden de compra así como el establecimiento de las cláusulas de compra como envío, empaquetado, donde y como pagar, así como las tecnologías inherentes. Posteriormente el proveedor envía el producto y a la recepción de este por el cliente se notifica al proveedor con un aviso de recepción y con la factura electrónica es como la venta es finalizada.

El éxito del negocio en esta modalidad sigue estando en conocer el mercado y los clientes, varias prácticas y esquemas de personalización basados en el comercio inteligente han evolucionado para ayudar a las compañías de comercio electrónico para mejorar las relaciones con sus clientes y poder suplir sus necesidades de manera óptima.

Ventajas

Para los clientes:

- Permite el acceso a más información. La naturaleza interactiva del Web y su entorno hipertexto permiten búsquedas profundas no lineales que son iniciadas y controladas por los clientes, por lo tanto las actividades de mercadeo mediante el Web están más impulsadas por los clientes que aquellas proporcionadas por los medios tradicionales.
- Facilita la investigación y comparación de mercados. La capacidad del Web para acumular, analizar y controlar grandes cantidades de datos especializados permite la compra por comparación y acelera el proceso de encontrar los artículos.
- Abarata los costos y precios. Conforme aumenta la capacidad de los proveedores para competir en un mercado electrónico abierto se produce una



baja en los costos y precios, de hecho tal incremento en la competencia mejora la calidad y variedad de los productos y servicios.

Para las empresas:

- se puede disminuir el tiempo que se tardan en realizar las transacciones comerciales, incrementando la eficiencia de las empresas.
- Comunicaciones de mercadeo. Actualmente, la mayoría de las empresas utiliza el Web para informar a los clientes sobre la compañía, a parte de sus productos o servicios, tanto mediante comunicaciones internas como con otras empresas y clientes. Esta capacidad ofrece oportunidades sin precedentes para ajustar con precisión las comunicaciones a los clientes individuales, facilitando que éstos soliciten tanta información como deseen.
- Beneficios operacionales. El uso empresarial del Web reduce errores, tiempo y sobre costos en el tratamiento de la información.

Desventajas

- Entorno empresarial y tecnológico cambiante. Empresas y clientes desean tener flexibilidad para cambiar, según su voluntad, de socios comerciales, plataformas y redes. No es posible evaluar el costo de esto, pues depende del nivel tecnológico de cada empresa, así como del grado deseado de participación en el comercio electrónico. Como mínimo una empresa necesitará una computadora personal con sistema operativo Windows o Mac, un módem, una suscripción a un proveedor de servicios de Internet, una línea telefónica. Una compañía que desee involucrarse más, deberá prepararse para introducir el comercio electrónico en sus sistemas de compras, financieros y contables, lo cual implicará el uso de un sistema para el intercambio electrónico de datos (EDI) con sus proveedores y/o una intranet con sus diversas sedes.



- Privacidad y seguridad. La mayoría de los usuarios no confía en el Web como canal de pago. En la actualidad, las compras se realizan utilizando el número de la tarjeta de crédito, pero aún no es seguro introducirlo en Internet sin conocimiento alguno. Cualquiera que transfiera datos de una tarjeta de crédito mediante el Web, no puede estar seguro de la identidad del vendedor. Análogamente, éste no lo está sobre la del comprador. Quien paga no puede asegurarse de que su número de tarjeta de crédito no sea recogido y sea utilizado para algún propósito malicioso; por otra parte, el vendedor no puede asegurar que el dueño de la tarjeta de crédito rechace la adquisición. Resulta irónico que ya existan y funcionen correctamente los sistemas de pago electrónico para las grandes operaciones comerciales, mientras que los problemas se centren en las operaciones pequeñas, que son mucho más frecuentes.

- Cuestiones legales, políticas y sociales. Existen algunos aspectos abiertos en torno al comercio electrónico: validez de la firma electrónica, no repudio, legalidad de un contrato electrónico, violaciones de marcas y derechos de autor, pérdida de derechos sobre las marcas, pérdida de derechos sobre secretos comerciales y responsabilidades. Por otra parte, deben considerarse las leyes, políticas económicas y censura gubernamentales.

2.1.2 Aplicaciones Web

Con el surgimiento de las aplicaciones Web el crecimiento de comercio electrónico ha superado las expectativas.

Después de revisar algunos conceptos implicados en el comercio electrónico es momento de introducir los conceptos que serán requeridos para poder desarrollar la aplicación que satisfaga los requerimientos.



Por tanto se tratará en esta sección los conceptos importantes de las aplicaciones Web, pero antes de definirlos será necesario tener una comprensión básica de las redes y de World Wide Web.

Transacciones http

HTTP define un conjunto de métodos y encabezados que permiten a los clientes y servidores interactuar e intercambiar información de una manera uniforme y confiable.

En su forma más simple, una página Web es un documento XHTML o HTML, es decir, un archivo de texto simple que contiene marcado, es decir, etiquetas, para describir a un navegador Web cómo mostrar y dar formato a la información del documento. Los documentos XHTML o HTML también pueden contener datos de hipertexto (lo que se conoce como hipervínculos) que vinculan a distintas páginas, o a otras partes de la misma página.

Cuando el usuario activa un hipervínculo, la página Web solicitada se carga en el navegador Web del usuario.

HTTP utiliza URI's que al especificar las ubicaciones de los documentos se llaman URL's (Localizadores uniformes de recursos).

Los URL's comunes se refieren a archivos, directorios u objetos que realizan tareas complejas, como búsquedas en bases de datos y en Internet.

Un URL contiene la información que dirige a un navegador al recurso que el usuario desea utilizar. Cuando el navegador Web recibe un URL, realiza una transacción HTTP simple para obtener y mostrar la página Web que se encuentra en esa dirección.

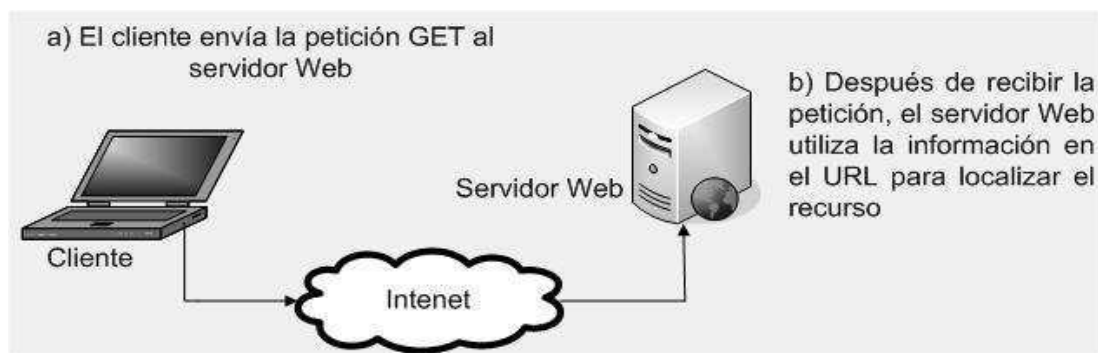


Figura 2.1.2.1 Interacción entre el navegador Web y servidor Web

El navegador Web envía una petición HTTP al servidor de la forma: *GET /ruta/recurso.html HTTP/1.1*, donde GET es un método HTTP, el cual indica que el cliente desea obtener un recurso del servidor. El resto de la petición proporciona el nombre de la ruta del recurso, junto con el nombre del protocolo y el número de versión (HTTP/1.1). Ver figura 2.1.2.1. Cualquier servidor que entienda HTTP (versión 1.1) puede traducir esta petición y responder en forma apropiada.

En la figura 2.1.2.2 se muestran los resultados de una petición exitosa. El servidor responde enviando una línea de texto que indica la versión de HTTP, seguida de un código numérico y una frase que describe el estado de la transacción. Estos códigos de estado HTTP están especificados por el RFC 2616 (ver Tabla 2.1.2.1)

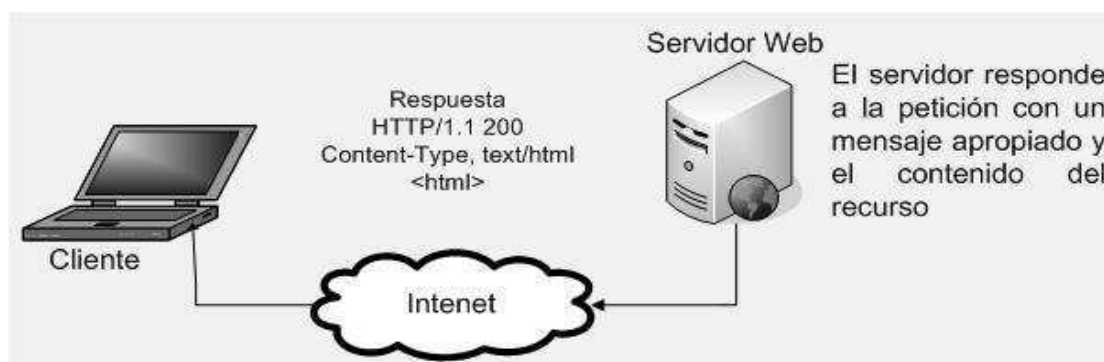


Figura 2.1.2.2 Interacción entre el cliente y el servidor Web: La respuesta HTTP.



Código de status	Descripción
1XX	Respuestas informativas
100	Continúa
101	Conmutando protocolos
102	Procesando (WebDav)
⋮	
2xx	Peticiones correctas
200	OK
201	Creado
202	Aceptada
⋮	
3xx	Redirecciones
301	Movido permanentemente
302	Movido temporalmente
304	No modificado
⋮	
4xx	Errores del cliente
401	No autorizado
403	Prohibido
404	No encontrado
⋮	
5xx	Errores de servidor
500	Error interno
503	Servicio no disponible
507	Almacenamiento insuficiente
⋮	

Tabla 2.1.2.1 Códigos de status HTTP

Después, el servidor envía uno o más encabezados HTTP, los cuales proporcionan información adicional sobre los datos que se van a enviar.

Por ejemplo:

- ✦ Content-type: text/HTML: El servidor está enviando un documento de texto XHTML.
- ✦ Content-type: text/plain: La información enviada es texto que puede mostrarse directamente, sin interpretar el contenido como marcado de XHTML.
- ✦ Content-type : image/jpeg: Indica que el contenido es una imagen JPEG



El encabezado, o conjunto de encabezados, va seguido por una línea en blanco, la cual indica al cliente que el servidor terminó de enviar encabezados HTTP. Después el servidor envía el contenido del documento HTML solicitado. El servidor termina la conexión cuando se completa la transferencia del recurso. El navegador del lado del cliente analiza el marcado de HTML que recibe y despliega (o visualiza los resultados).

Aplicación Web

Con base en lo anterior, una aplicación Web es aquella aplicación que los usuarios pueden utilizar accediendo a un servidor Web a través de Internet o de una intranet mediante un navegador o browser, de forma dinámica.

Es una aplicación de software que se codifica en un lenguaje soportado por los navegadores Web como HTML, JavaScript, Java, PHP, etc., en la que se confía la ejecución al navegador.

De esta manera una aplicación Web cuenta con un conjunto de páginas dinámicas y estáticas.

Páginas estáticas. Las páginas estáticas son archivos HTML alojados en un equipo que ejecuta un servidor Web, el servidor Web, que se verá más adelante en detalle, es quien suministra las páginas Web en respuesta a las peticiones de los navegadores Web.

El servidor sencillamente al recibir la petición de una página estática lee la solicitud, localiza la página y la envía al navegador que lo solicitó.

Cuando el servidor Web recibe una petición para mostrar una página Web estática, el servidor la envía directamente al navegador que la solicita (Ver Figura 2.1.2.3).

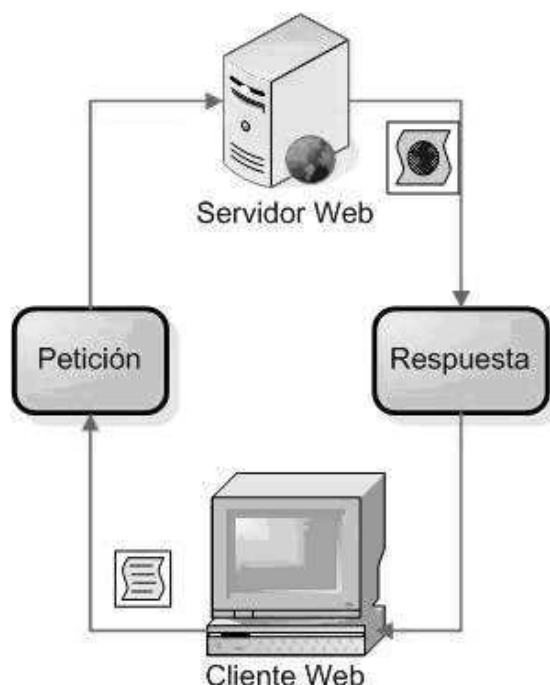


Figura 2.1.2.3 Procesamiento de una página Web estática

Página dinámica: Una página dinámica utiliza lenguajes que a su vez serán interpretados por el navegador, como JavaScript, para añadir elementos dinámicos a la interfaz de usuario. Permite una comunicación activa entre usuario y la información.

Cuando el servidor Web recibe una petición para mostrar una página dinámica, transfiere la página a un software especial encargado de finalizar la página. Este software especial se denomina servidor de aplicaciones. El servidor de aplicaciones lee el código de la página, finaliza la página en función de las instrucciones del código y elimina el código de la página. El resultado es una página dinámica que el servidor de aplicaciones devuelve al servidor Web, que a su vez la envía al navegador que la solicita, así lo único que el navegador recibe cuando llega la página es código HTML puro. (ver figura. 2.1.2.4)

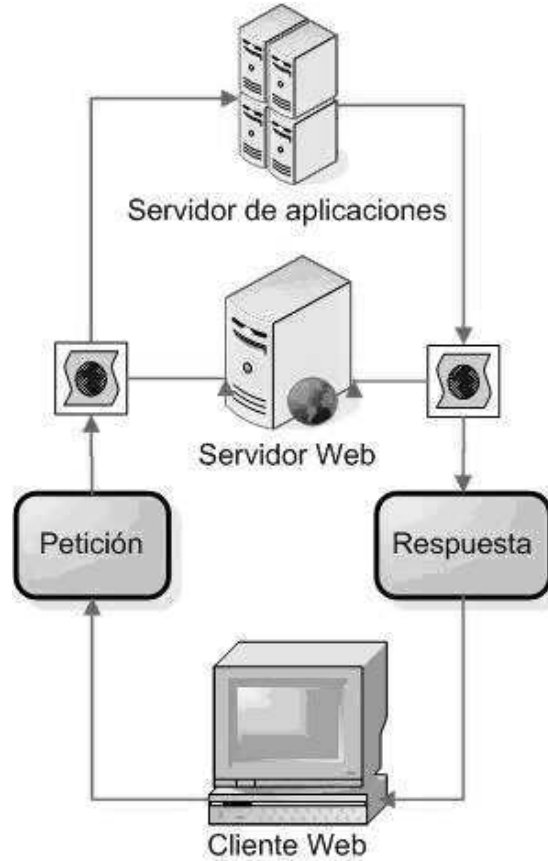


Figura 2.1.2.4 Procesamiento de una página Web dinámica

Funcionamiento de una aplicación Web

Las aplicaciones Web son aplicaciones multinivel (comúnmente conocidas como aplicaciones de n niveles), que dividen su funcionalidad en niveles separados, es decir, agrupaciones lógicas de funcionalidad.

Aunque los niveles pueden localizarse en la misma computadora, por lo general, los niveles de las aplicaciones Web residen en computadoras separadas.

En la figura 2.1.2.5 se presenta la estructura básica de una aplicación Web en tres niveles. Aunque existan muchas variaciones posibles en su forma más común una aplicación Web está estructurada como una aplicación en tres capas.

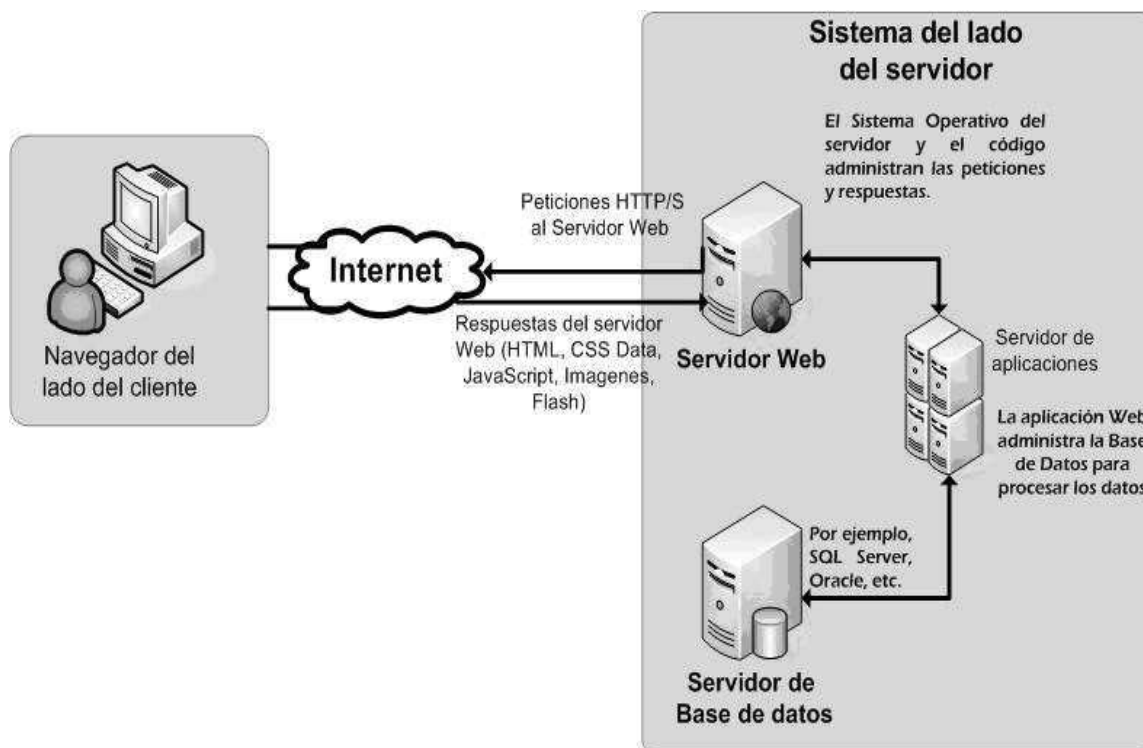


Figura 2.1.2.5 Arquitectura de tres niveles o capas.

Primer Nivel o nivel superior: Es la interfaz de usuario de la aplicación, la cual recopila los datos de entrada y de salida.

Segundo nivel o nivel intermedio: Implementa la lógica de negocios, de controlador y de presentación para controlar las interacciones entre los datos en el nivel de información y los clientes de la aplicación. La lógica de control de este nivel procesa las peticiones de los clientes y obtiene datos de la base de datos. La lógica comercial en el nivel intermedio hace valer las reglas comerciales y asegura que los datos sean confiables antes de que la aplicación servidor actualice la base de datos, o presente los datos a los usuarios. Las reglas comerciales dictan la forma en que los clientes pueden o no acceder a los datos de la aplicación, y la forma en que ésta procesa los datos. Es el motor capaz de usar alguna tecnología Web dinámica (PHP, JSP, ASP, Java, etc.).



Tercer Nivel o nivel inferior: También conocido como nivel de datos o nivel de información, mantiene los datos de la aplicación. Generalmente este nivel almacena los datos en un sistema de administración de bases de datos relacionales (RDBMS). Es donde se encuentra la base de datos que almacena el contenido y datos del cliente como nombres de usuario, contraseñas, números de seguridad social y detalles de tarjetas de crédito.

Proceso del acceso a una base de datos.

Puede utilizar prácticamente cualquier base de datos con la aplicación Web, siempre y cuando se haya instalado el controlador de base de datos correcto en el servidor. Los usuarios interactúan en forma directa con la aplicación a través de la interfaz de usuario, que por lo general es el navegador Web, el teclado y el ratón.

En respuesta a las acciones del usuario, el nivel cliente interactúa con el nivel intermedio para hacer peticiones y obtener datos del nivel de información. Después, el nivel cliente muestra los datos obtenidos para el usuario. El nivel cliente nunca interactúa directamente con el nivel de información. El navegador Web envía peticiones a la capa de en medio que ofrece servicios valiéndose de consultas y actualizaciones a la base de datos y a su vez proporciona una interfaz de usuario.

En la figura de 2.1.2.6 se muestra el flujo del proceso de acceso a la base de datos en el que se aprecia como la petición inicial es activada por el usuario a través del navegador por Internet al servidor de aplicación Web.

La aplicación Web accede a los servidores de bases de datos para realizar la tarea pedida actualizando y recuperando la información que queda dentro de la base de datos. La aplicación Web entonces presenta la información al usuario a través del navegador.

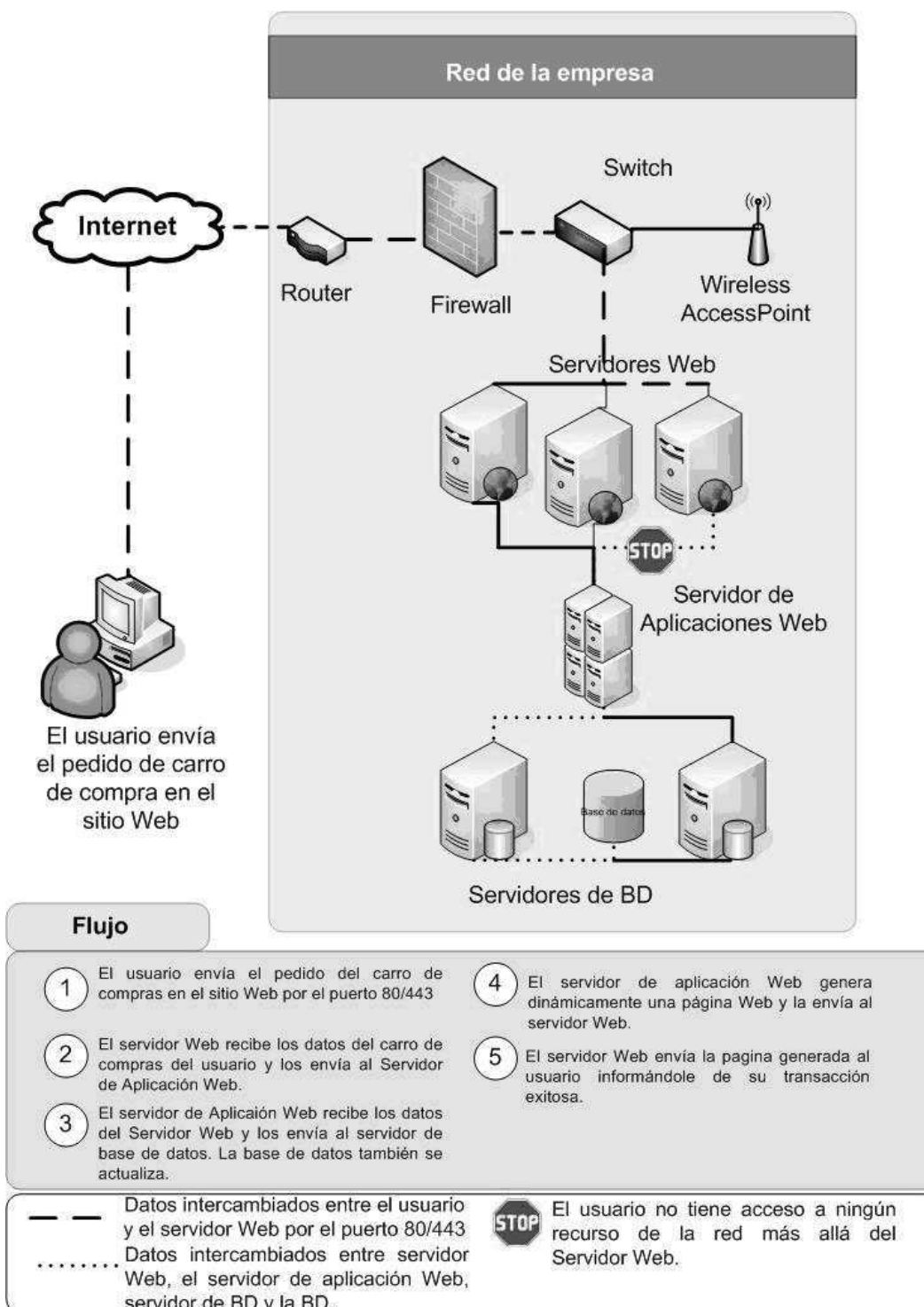


Figura 2.1.2.6 Proceso de acceso a la base de datos



Las aplicaciones Web solicitan al servidor de contenido (esencialmente una base de datos) y dinámicamente genera documentos Web del servidor al cliente. Los documentos son generados en un formato estándar que permite ser soportado por todos los navegadores (ej. HTML o XHTML).

El navegador es una pieza clave ya que es quien interpreta y ejecuta los scripts mientras muestra las páginas solicitadas y contenido, es el cliente universal para cualquier aplicación Web.

A continuación se detallan algunas ventajas y desventajas de las aplicaciones Web que podemos encontrar cuando se desea desarrollar aplicaciones de estas arquitecturas.

VENTAJAS

- Ligeras. Al ser ejecutadas en un navegador Web es práctico y ligero al cliente.
- Portables. Es independiente de la computadora donde se utilice debido a que accede a través de una página Web, solo es necesario contar con una conexión a Internet. Cuando el acceso se realiza mediante teléfonos móviles es necesario un diseño específico de los archivos CSS para no dificultar el acceso a estos usuarios.
- Para actualizarla y mantenerla es más fácil, ya que solo basta con actualizarla en el servidor y no es necesario distribuir e instalar la aplicación en todos los equipos de la red.
- Multiplataforma. Funcionan igual, independiente de la versión del sistema operativo instalado en el cliente. La aplicación Web se escribe una vez y se ejecuta igual en todas partes. Sólo es necesario contar con un navegador compatible y mínimamente actualizado.



- Ahorro de tiempo. Se pueden realizar tareas sencillas sin necesidad de descargar ni instalar ningún programa, eso mejora la rapidez en su funcionamiento.
- No requiere de “grandes instalaciones”.
- Actualizaciones inmediatas. Gracias a que el software lo gestiona el propio desarrollador, cuando se acceda a la aplicación Web se está usando la última versión que haya lanzado.
- No ocupan espacio en el disco duro local.
- Bajo consumo de recursos. Gracias a que la aplicación no se encuentra instalada en la computadora local muchas de las tareas que realiza la aplicación no consumen recursos de la computadora local ya que se realizan desde otra computadora.
- Alta disponibilidad. Esto se logra debido a que el servicio se ofrece desde múltiples localizaciones para asegurar la continuidad del mismo.
- Existe cierto “blindaje” contra los virus ya que los datos están almacenados en el servidor de la aplicación.
- Colaboración. El acceder y compartir los datos por parte de varios usuarios es más fácil ya que el acceso al servicio se realiza desde una única ubicación.

DESVENTAJAS

- Se ofrecen menos funcionalidades que las aplicaciones de escritorio, o más limitadas.



- Es necesario contar con una conexión a Internet permanente (generalmente) y una velocidad y ancho de banda promedio para una óptima navegación.
- La disponibilidad depende de un tercero, el proveedor de la conexión a Internet, o el que provee el enlace entre el servidor de la aplicación y el cliente.
- La seguridad de los datos, depende de la seguridad de la aplicación Web y del servidor donde este alojado.
- Está limitado al espacio donde se guarda la información.
- Poco soporte a varios idiomas.
- En ocasiones depende bastante de la configuración de la privacidad del navegador para trabajar eficientemente (ej. JavaScript, cookies, Flash).
- La estabilidad de la aplicación está sujeta al número de visitas en un mismo lapso de tiempo (sobrecarga del servidor).

2.2 Características, ventajas y desventajas de Apache, PHP, JavaScript y XML

2.2.1 Apache

Uno de los elementos que serán imprescindibles para una aplicación Web es contar con un servidor Web, para poder elegir cual es el que es conveniente para los propósitos del presente trabajo es necesario comprender algunas cuestiones relacionadas y que se muestran a continuación.



Servidor. Un servidor técnicamente es un sistema que proporciona recursos, este sistema es conocido como una computadora que pertenece a una red y que suministra información a otras computadoras, las cuales ejecutan los programas clientes.

Dependiendo el tipo de recursos que suministra es la clasificación del tipo de servidor, por ejemplo, servidor de archivos, de impresión, de correo, de telefonía, de acceso remoto, de bases de datos, o el que por ahora es de interés es el servidor Web.

Servidor Web. El servidor Web es un programa cuya función es atender y responder a las diferentes peticiones de los navegadores, proveyendo de los recursos que se soliciten usando el protocolo http (Hipertext Transfer Protocol,) o el https (Hipertext Transfer Protocol Secure), que es la versión cifrada y autenticada. Este protocolo pertenece a la capa de aplicación del modelo OSI (Open System Interconnection, Modelo de referencia de interconexión de sistemas abiertos) (Figura 2.2.1.1) y que está diseñado para transferir lo que llamamos hipertextos, páginas Web o páginas HTML (Hypertext Markup Language):

Textos complejos con enlaces, imágenes, formularios, botones y objetos incrustados como animaciones o reproductores de música.

Este programa llamado servidor generalmente se instala en una computadora conectada en forma permanente en Internet, de esta manera podrá atender las solicitudes de páginas Web y tareas de procesos de datos que los clientes pudieran solicitar en cualquier momento.

Con esto existe la ventaja de la disponibilidad de la información desde cualquier punto de la red y además de que es posible almacenar aplicaciones en el servidor, permitiendo que el cliente no requiera de elementos adicionales para ejecutar la aplicación.

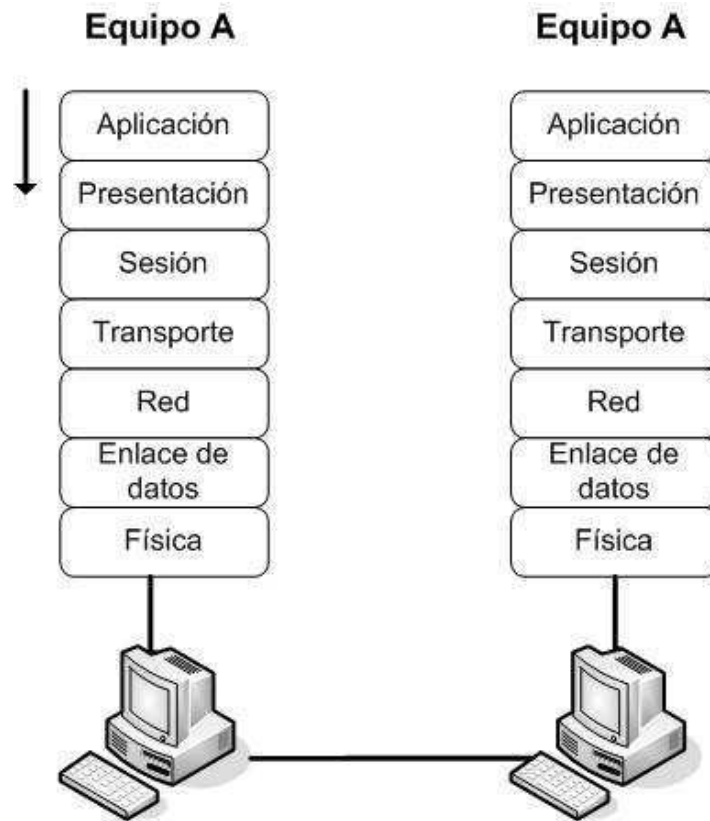


Figura 2.2.1.1 El protocolo HTTP opera en la capa de aplicación del modelo OSI

Gracias a los avances en conectividad y la gran disponibilidad de banda ancha, hoy en día es muy común establecer los servidores Web dentro de la propia empresa, sin tener que recurrir a caros alojamientos en proveedores externos.

Funcionamiento

Los servidores se conectan a la red mediante una interfaz que puede ser una red verdadera o mediante conexión vía telefónica o digital.

El ciclo que ejecuta un servidor Web básico es el siguiente:



- ✦ El servidor se ejecuta en una computadora manteniéndose a la espera de peticiones por parte de un cliente (navegador), esta espera de peticiones es en el puerto TCP (por defecto es el 80).
- ✦ Recibe petición.
- ✦ Busca recurso.
- ✦ Responde a estas peticiones enviando el recurso y utilizando la misma conexión por la que recibió la petición mostrándose en el navegador o señalándose el respectivo mensaje si se detectó algún error. El servidor tan sólo se limita a transferir el código de la página sin llevar a cabo ninguna interpretación de la misma.
- ✦ El cliente es el encargado de interpretar el código HTML, es decir, de mostrar las fuentes, colores y disposición de los textos y objetos de la página.
- ✦ Regresa al paso en que se recibe la petición.

Como se vio en la sección 2.1.2 (aplicaciones Web) si existió algún problema de conexión se envía un mensaje de error de estatus http. (Figura 2.1.2.3).

El esquema anterior sólo es válido para páginas estáticas, por tanto no es suficiente actualmente ya que la tendencia es a desarrollar páginas dinámicas, de esta manera es importante que en la elección de un servidor Web se analice el soporte para contenido de tipo dinámico.

De esta manera además de transferir código HTML los servidores Web pueden entregar aplicaciones Web, estas son porciones de código que se ejecutan cuando se realizan ciertas peticiones o respuestas http.

- ✦ *Aplicaciones en el lado del cliente:* El cliente Web es el encargado de ejecutarlas en la máquina del usuario. Son las aplicaciones tipo Java “applets” o JavaScript: el servidor proporciona el código de las aplicaciones al cliente y éste, mediante el navegador, las ejecuta. Es necesario, por tanto,



que el cliente disponga de un navegador con capacidad para ejecutar aplicaciones (también llamadas scripts).

- ✦ Aplicaciones del lado del servidor: el servidor Web ejecuta la aplicación; ésta, una vez ejecutada, genera cierto código HTML; el servidor toma este código recién creado y lo envía al cliente por medio del protocolo http.

Cuando se contrata un plan de alojamiento Web con una compañía, esta proporciona un servidor al dueño del sitio para poder alojarlo; al respecto hay dos opciones, optar por un “servicio dedicado”, lo que se refiere a una computadora servidora dedicada exclusivamente al sitio del cliente (para aplicaciones de alta demanda), o un “servidor compartido”, lo que significa que un mismo servidor (computadora + programas servidor) se usará para varios clientes compartiendo los recursos.

Para seleccionar el servidor Web que satisfaga nuestras necesidades se deberán tomar en cuenta los siguientes aspectos:

- ✦ Número de conexiones concurrentes.
- ✦ Número de transacciones por segundo.
- ✦ Soporte para el desarrollo.
- ✦ Seguridad. Que pueda garantizar la integridad (al viajar la información el contenido y el significado no se altere), confiabilidad (el servicio debe estar disponible en todo momento) y confidencialidad (que impida que personas no autorizadas lean y conozcan la información que se transmite) de la información.

Servidores Web más populares:

- ✦ Apache
- ✦ IIS
- ✦ Cherokee

Otros servidores, más simples pero más rápidos, son:

- ✦ lighttpd



✦ tthttpd

El más usado es el Apache del ambiente Unix, seguido por el IIS del sistema Windows Server. Se explica el predominio del servidor Apache porque es gratis, muy estable y confiable, tiene una gran capacidad y existen versiones para todas las plataformas de computación o se lo puede adaptar con facilidad

Servidor Web apache

Cuando comenzó su desarrollo en 1995 se basó inicialmente en código del popular servidor Web NCSA HTTPd 1.3 cuyo desarrollo se suspendió en 1998, pero más tarde todo el código de Apache fue reescrito por completo. Apache consistía solamente en un conjunto de parches a aplicar al servidor de NCSA, era en inglés, “a *patchy Server*” (un servidor “parcheado”).

Posteriormente todo el soporte de Apache fue provisto por “The Apache Group” o “La Fundación Apache”, logrando la primera aparición de Apache en Abril de 1995. Este servidor se sigue desarrollando como un proyecto de Software libre.

La licencia de software bajo la cual el software de la fundación Apache es distribuido es una parte distintiva de la historia de Apache HTTP Server y de la comunidad de código abierto. La licencia Apache permite la distribución de derivados de código abierto y cerrado a partir de su código fuente original.

Apache tiene amplia aceptación en la red: desde 1996, Apache, es el servidor HTTP (conforme al protocolo HTTP/1.1) más usado. Alcanzó su máxima porcentaje de mercado en 2005 siendo el servidor empleado en el 72% de los sitios Web en el mundo para esa fecha.

Aunque el porcentaje de su uso ha descendido un poco, para Octubre de 2010, de acuerdo a la empresa Netcraft, empresa británica de servicios de Internet, el servidor



Web Apache sigue siendo el servidor más usado de Internet ocupando el 67% del mercado. Ver Figura 2.2.1.2

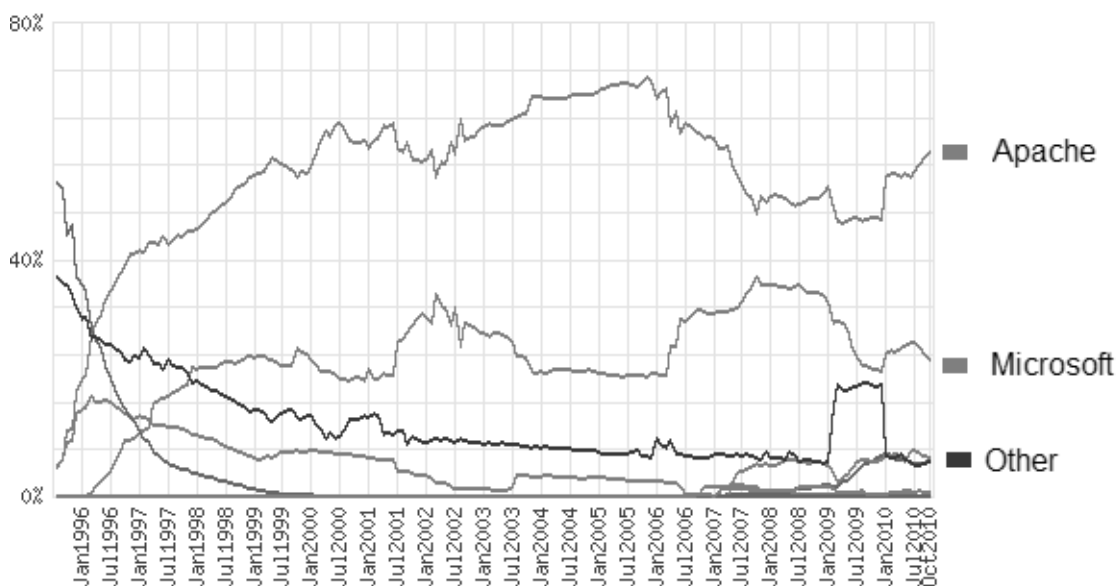


Figura 2.2.1.2 El servidor Apache es el más usado en los sitios Web a nivel mundial

Apache es usado para muchas otras tareas donde el contenido necesita ser puesto a disposición en una forma segura y confiable. Es principalmente usado para servir páginas Web estáticas y dinámicas en la WWW. Apache es el servidor Web del popular sistema XAMP, junto con MySQL y los lenguajes de programación PHP/Perl/Python. La "X" puede ser la inicial de cualquier sistema operativo, si es Windows: WAMP, si es el Linux: LAMP, etc.

Características

Código abierto. Todo el código fuente de Apache está escrito en C, Apache es una tecnología gratuita de código fuente abierto (*open source*) es el término con el que se conoce al software distribuido y desarrollado libremente, es decir, existe transparencia en la forma de su desarrollo. El API de Apache es de dominio público por lo que pueden



realizarse mejoras al código además de agregarse nuevos módulos al servidor, así como una capacidad de adaptación a necesidades particulares. Existe una gran comunidad de desarrolladores a nivel mundial que desarrollan y lo mejoran gracias a esta realimentación de los usuarios, obteniendo nuevas ideas, informes de fallos y parches para la solución de los mismos, logrando actualizaciones constantes sin necesidad de esperar a que un grupo reducido de desarrolladores resuelva un fallo sino que puede ser resuelta por un grupo más completo. Esto permite producir binarios para plataformas en las que no hay un binario oficial o compilarlo para un mejor rendimiento en una plataforma específica.

Es un servidor basado en procesos, utilizando la técnica pre-fork (consiste en la creación previa de un grupo de procesos y su mantenimiento hasta que sea necesaria su utilización).

Arquitectura modular. Arquitectura en la que existe una sección core o base y diversos módulos multiproceso y módulos adicionales que aportan mucha de la funcionalidad que podría considerarse básica para un servidor Web, es decir, gracias a esto se puede adaptar a diferentes entornos y necesidades con los diferentes módulos de apoyo que proporciona y con la API de programación de módulos para el desarrollo de módulos específicos. Cualquiera que posea una experiencia decente en la programación de C o Perl puede escribir un módulo para realizar una función determinada.

Administración. Es uno de los elementos principales de cualquier servidor. Apache cuenta con un conjunto de archivos de configuración ampliamente documentados, estos se encuentran en formato ASCII y pueden ser editados con cualquier editor de texto. Si se necesita realizar alguna actualización resulta sencillo respaldar la configuración del servidor. Se puede administrar de forma remota ya que puede operarse y configurarse desde la línea de comandos, así como por medio de interfaces gráficas y un conjunto de herramientas desarrolladas con este propósito.



Extensible: Gracias a que es modular se han desarrollado diversas extensiones entre las que destaca PHP, un lenguaje de programación del lado del servidor.

Personalizable, la arquitectura modular de Apache permite construir un servidor hecho a la medida. Además permite la implementación de los últimos y más nuevos protocolos.

Configuración de errores. Cuenta con archivos de configuración específicos que hacen posible configurar el servidor para que ejecute un determinado script cuando ocurra un error en concreto, es decir, Apache permite la creación de *logs* de errores configurados por el administrador. Tiene una alta configuración en la creación y gestión de *logs* que contienen enormes cantidades de información, necesitando una herramienta que realice resúmenes de los datos para facilitar el análisis del contenido, de este modo se puede tener un mayor control sobre lo que sucede en el servidor.

VENTAJAS

- ✦ *Eficiencia y estabilidad.* La eficiencia de Apache es lograda gracias a que se encuentra optimizado para consumir la menor cantidad de recursos del sistema en comparación a otros servidores, gracias a esto puede lograr un gran desempeño convirtiéndolo en una aplicación estable y robusta, soportando un mayor número de transacciones.
- ✦ *Multiplataforma.* Apache es soportado por la gran mayoría de plataformas Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh y otras, que implementan el protocolo HTTP/1.11 y la noción de sitio virtual, es por tanto una aplicación sumamente portable y configurable en diferentes ambientes lo que lo hace prácticamente universal.
- ✦ *Extensibilidad.* Apache es un servidor altamente configurable de diseño modular. Es muy sencillo ampliar las capacidades del servidor Web Apache. Actualmente



existen muchos módulos para Apache que son adaptables a éste. Desde módulos que permiten generar contenido dinámico (con PHP, Java, Perl, Python), monitoreo del rendimiento, comunicaciones seguras por SSL, creación de servidores virtuales por IP. Otra cuestión importante es que cualquiera que posea una experiencia decente en la programación de C o Perl puede escribir un módulo para realizar una función determinada.

- ✦ *Seguridad.* La mayoría de las vulnerabilidades de la seguridad descubiertas y resueltas puede en la mayoría de los casos ser abusada solamente por los usuarios locales y no puede ser accionada remotamente. Cuenta con Módulos de autenticación: mod_access, mod_auth y mod_digest, así como Soporte para SSL y TLS. También permite autenticación de base de datos basada en SGBD.
- ✦ *Soporte para los lenguajes.* Apache trabaja con gran cantidad de lenguajes como Perl, Python, tcl, PHP y otros lenguajes de script. Perl destaca en el mundo del script y Apache utiliza su parte de Perl tanto con soporte CGI como con soporte mod perl. También trabaja con Java y páginas jsp. Teniendo todo el soporte que se necesita para tener páginas dinámicas.
- ✦ *Personalización de los mensajes de error.* Apache permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor. Es posible configurar Apache para que ejecute un determinado script cuando ocurra un error en concreto y negociación de contenido de esta manera se puede tener un mayor control sobre lo que sucede en el servidor.

Desventajas

- ✦ *Administración compleja.* Como la mayoría de los programas open-source, uno depende de configurar los archivos a mano o tener que instalarse herramientas adicionales para las tareas de administración. Fue criticado por la falta de una interfaz gráfica que ayude en su configuración ya que resultar difícil de



configurar incluso para tareas sencillas y por ello algunos usuarios con poca experiencia tratan de evadir su uso ya que una configuración errónea puede hacer que no funcione adecuadamente el servidor, es necesario contar con la documentación necesaria respecto a los archivos de configuración, esto para poder ajustar correctamente algunos parámetros del servidor.

- ✦ *Falta de integración.* Al ser un producto multiplataforma, el servidor no aprovecha al máximo las posibilidades que ofrece el sistema operativo.
- ✦ *Formatos de configuración no estándar.* Esto dificulta un poco la automatización y el procesamiento de la configuración al no estar basada en formatos más soportados como el XML.
- ✦ *Lentitud de arranque y ejecución.* El servidor es aproximadamente un 20% más lento en su arranque debido a la sobrecarga que la resolución representa para el cargador (loader) y es aproximadamente un 5% más lento en su ejecución bajo algunas plataformas.

2.2.2 PHP

Fue originalmente diseñado en Perl como un complemento para ese lenguaje, con base en la escritura de un grupo de CGI binarios escritos en el lenguaje C por el programador danés-canadiense Rasmus Lerdorf en el año 1994. El 8 de junio de 1995 fue publicado "*Personal Home Page Tools*" después de que Lerdorf lo combinara con su propio *Form Interpreter* para crear PHP/FI. En 1997 las aportaciones de Zeev Suraski y Andy Gutsman llevaron al desarrollo del PHP versión 3 y fue cuando éste se popularizó de manera definitiva, sobre todo al combinarse con la base de datos MySQL y el servidor Apache. Dado que estos tres productos son Open Source, o código abierto, puede usarse sin limitación alguna por cualquier usuario o empresa que así lo desee. La implementación principal de PHP ahora es producida por *The PHP Group*.

Actualmente PHP se usa como acrónimo recursivo de **H**ypertext **P**re-processor (Pre-procesador de hipertexto) y se encuentra instalado en más de 20 millones de sitios Web y en un millón de servidores. De acuerdo a la empresa *tiobe.com*, (dedicada a las



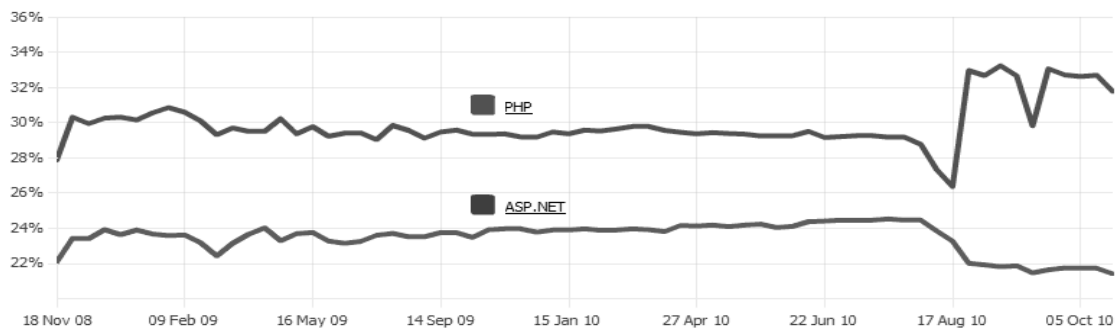
estadísticas de uso de los lenguajes de programación más populares), (ver tabla 2.2.2.1) PHP es ubicado en 2010 en la cuarta posición ocupando el 8.32% del mercado.

Position Oct-10	Position Oct-10	Delta in Position	Programming Language	Rantings Oct-10	Delta Oct-09
1	1	=	Java	18.17%	-0.48%
2	2	=	C	17.18%	0.33%
3	3	↑	C++	9.80%	-0.08%
4	4	↓	PHP	8.32%	-2.03%
5	5	=	(Visual) Basic	5.65%	-3.04%
6	6	=	C#	4.96%	0.55%
7	7	=	Python	4.86%	0.96%

Tabla 2.2.2.1 Estadísticas de uso de lenguajes de programación (www.tiobe.com).

Incluso con su competidor más cercano de su categoría por la parte de software propietario se encuentra muy por encima. (Ver figura 2.2.2.1)

Comparison Results
PHP compared to ASP.NET



This chart represents **PHP** penetration compared with **ASP.NET** penetration over a historical time period on a large selection of website homepages queried by BuiltWith.

Figura 2.2.2.1 Estadística PHP vs ASP.NET (<http://trends.builtwith.com>)



Definición del lenguaje PHP

Es un lenguaje de programación interpretado, diseñado para la creación de páginas Web dinámicas y que puede ser incrustado dentro de código HTML, usado principalmente en interpretación del lado del servidor tomando el código en PHP como su entrada y creando páginas Web como salida. Actualmente puede ser utilizado desde una interfaz de línea de comandos.

Combinando con la base de datos MySQL, PHP es el lenguaje estándar a la hora de crear sitios de comercio electrónico o páginas Web dinámicas. Posee un gran número de funciones predefinidas. A diferencia de otros lenguajes de programación, este lenguaje fue diseñado especialmente para el desarrollo de páginas Web dinámicas. Por ello, está dotado de un gran número de funciones que simplificarán enormemente tareas habituales como descargar documentos, enviar correos, trabajar con *cookies* y sesiones, etc.

Modo de operación

Cuando el cliente hace una petición al servidor para que le envíe una página Web, el servidor ejecuta el intérprete de PHP. Éste procesa el script solicitado que generará el contenido de manera dinámica (por ejemplo obteniendo información de una base de datos). El resultado es enviado por el intérprete al servidor, quien a su vez se lo envía al cliente. Mediante extensiones es también posible la generación de archivos PDF, Flash, así como imágenes en diferentes formatos. Podemos resumir la manera en que opera PHP en la siguiente manera:

- ✦ El Navegador realiza una petición al servidor (se escribe la URL).
- ✦ Después el servidor ejecuta el código PHP solicitado y retorna el código HTML generado al Navegador.
- ✦ Por último el Navegador muestra la respuesta del servidor.



Este tipo de iteración permite algunas operaciones complejas como conexiones a bases de datos o ejecución de complejos programas. PHP además de soportar un número masivo de bases de datos, también nos ofrece una gran variedad de opciones que nos permiten desarrollar múltiples funcionalidades que van desde enviar un e-mail, subir un archivo (upload), crear una imagen en tiempo de ejecución, interactuar con diversos protocolos de comunicación, interactuar con documentos XML, autenticación, creación dinámica de documentos PDF, entre muchas otras cosas.

Características

El código se pone al día continuamente con mejoras y extensiones de lenguaje para ampliar las capacidades de PHP.

Lo que distingue a PHP del lado cliente como Javascript, es que el código es ejecutado en el servidor, generando HTML y enviándolo al cliente. El cliente recibirá los resultados de ejecutar el script, sin ninguna posibilidad de determinar qué código ha producido el resultado recibido. El servidor Web puede ser incluso configurado para que procese todos los archivos HTML con PHP y entonces no haya manera que los usuarios puedan saber el código fuente completamente.

Integración con varias bibliotecas externas, permite generar documentos en PDF (documentos de Acrobat Reader) hasta analizar código XML.

Diseñado especialmente para aplicaciones Web PHP, similar al ASP de Microsoft o el JSP de Sun, puede ser incrustado dentro de código HTML el cual se ejecuta en un servidor Web, tomando como entrada el código hecho en PHP y dando como salida la creación de páginas Web. La mayor parte de su sintaxis ha sido tomada de C, Java y Perl con algunas características específicas de sí mismo. La meta del lenguaje es permitir rápidamente a los desarrolladores la generación dinámica de páginas. No es un lenguaje de marcas como podría ser HTML, XML o WML. Está más cercano a JavaScript o a C.



A diferencia de Java o JavaScript que se ejecutan en el navegador, PHP se ejecuta del lado del servidor por eso nos permite acceder a los recursos que tenga el servidor como por ejemplo podría ser una base de datos. El programa PHP es ejecutado en el servidor y el resultado es enviado al navegador. El resultado es normalmente una página HTML pero también podría ser una página WML (Wap). Al ser un lenguaje libre dispone de una gran cantidad de características que lo convierten en la herramienta ideal para la creación de páginas Web dinámicas.

Ventajas

- ✦ Es libre y es uno de los productos más representativos y utilizados del Open Source, por lo que se presenta como una alternativa de fácil acceso para todos.
- ✦ Gran popularidad. Existe una gran comunidad de desarrolladores y programadores que continuamente implementan mejoras en su código.
- ✦ Gratuito. Al tratarse de software libre, puede descargarse y utilizarse en cualquier aplicación, personal o profesional, de manera completamente libre.
- ✦ Multiplataforma. PHP también tiene la capacidad de ser ejecutado en la mayoría de los sistemas operativos, tales como UNIX (y de ese tipo, como Linux o Mac OS X) y Windows, y puede interactuar con los servidores de Web más populares ya que existe en versión CGI, módulo para Apache, e ISAPI. PHP también soporta la mayoría de los servidores Web de hoy en día, esto incluye: Apache, Microsoft Internet Information Server, Personal Web Server, los servidores Netscape e iPlanet, Oreilly Website Pro Server, Caudium, Xitami, OmniHTTPd, y muchos otros. Para la mayoría de los servidores, PHP tiene un módulo, para los otros que soporten el estándar CGI, PHP puede funcionar como un procesador CGI.
- ✦ Completamente orientado al desarrollo de aplicaciones Web dinámicas con acceso a información almacenada en una Base de Datos.
- ✦ Capacidad de conexión con la mayoría de los motores de base de datos que se utilizan en la actualidad, destaca su conectividad con MySQL y PostgreSQL.



- ✦ Segura y confiable. El código fuente escrito en PHP es invisible al navegador y al cliente ya que es el servidor el que se encarga de ejecutar el código y enviar su resultado HTML al navegador. Esto hace que la programación en PHP sea segura y confiable. PHP provee diferentes niveles de seguridad, estos pueden ser configurados desde el archivo *.ini
- ✦ Permite aplicar técnicas de programación orientada a objetos. No requiere definición de tipos de variables aunque sus variables se pueden evaluar también por el tipo que estén manejando en tiempo de ejecución.
- ✦ Permite la conexión a diferentes tipos de servidores de bases de datos tales como MySQL, Postgres, Oracle, ODBC, DB2, Microsoft SQL Server, Firebird y SQLite. Es importante para una página Web verdaderamente dinámica, una correcta integración con base de datos. Aunque My SQL es la base de datos que mejor trabaja con PHP, puede conectarse también a PostgreSQL, Oracle, dbm, filePro, interbasem o cualquier otra base de datos compatible con ODBC (*Open Database Connectivity Standard*).
- ✦ Enorme eficiencia. Con escaso mantenimiento y un servidor gratuito (en nuestro caso, Apache), puede soportar sin problema millones de visitas diarias.
- ✦ Velocidad. No solo la velocidad de ejecución, la cual es importante, sino además no crea demoras en la máquina. Por esta razón no debe requerir demasiados recursos de sistema. PHP se integra muy bien junto a otro software, especialmente bajo ambientes Unix, cuando se configura como módulo de Apache, esta listo para ser utilizado.
- ✦ Estabilidad. La velocidad no sirve de mucho si el sistema se cae cada cierta cantidad de ejecuciones. Ninguna aplicación es 100% libre de bugs, pero teniendo de respaldo una increíble comunidad de programadores y usuarios es mucho mas difícil para lo bugs sobrevivir. PHP utiliza su propio sistema de administración de recursos y dispone de un sofisticado método de manejo de variables, conformando un sistema robusto y estable.
- ✦ Simplicidad y expansión. Es fácil de aprender, usuarios con experiencia en C y C++ podrán utilizar PHP rápidamente asimismo se cuenta con la capacidad de



expandir su potencial utilizando la enorme cantidad de módulos (llamados ext's o extensiones).

- ✦ Posee una amplia documentación en su página oficial, entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda. Biblioteca nativa de funciones sumamente amplia e incluida.

Desventajas

- ✦ El manejo de errores no es tan sofisticado como Cold Fusion o ASP. No existe IDE o Debugger, aunque una IDE puede no ser importante para la mayoría de los programadores.
- ✦ La legibilidad del código puede verse afectada al mezclar sentencias HTML y PHP.
- ✦ Al llamar y ejecutar PHP una función, consume más recursos (se vuelve ligeramente más lento) que ejecutando código que encuentra en línea, embebido en el script. Por eso no tiene sentido usar funciones salvo que efectivamente se vaya a utilizar esa porción de código varias veces.

2.2.3 JavaScript

Otro de los elementos importantes de una aplicación AJAX y del cual debe parte de su nombre es la “J” de JavaScript.

JavaScript fue creado originalmente por Brendan Eich de Netscape con el nombre de Mocha, el cuál fue renombrado posteriormente a LiveScript en 1995 para la versión 2.0 de su navegador, poco tiempo después Microsoft incorporaba a su navegador una implementación de ese mismo lenguaje, al que denominó JScript.



Cada nueva versión de Netscape y de Internet Explorer introducía novedades en sus particulares implementaciones de JavaScript, cada vez más incompatibles entre sí, tendencia que llegó a su fin cuando ambas empresas acordaron en Noviembre de 1996 que un organismo independiente, concretamente la institución europea de estandarizaciones ECMA, crease una especificación común del lenguaje que sería la tomada como referencia para futuras versiones de sus respectivos navegadores. Dicha especificación de referencia se denominó EcmaScript. En Junio de 1997 es adoptado este estándar y luego también por la *"Internacional Organization for Standardization"* (ISO).

Por tanto JScript no es lo mismo que JavaScript. Este último es el estándar (también se le llamó ECMAScript), mientras que el primero es propiedad de Microsoft.

La primera versión de EcmaScript surgió en 1998, seguida por actualizaciones en años posteriores, que en menor o mayor medida, han sido incorporadas en los distintos navegadores en uso actualmente. La especificación de la tercera edición de EcmaScript, fue en diciembre de 1999 y es la empleada como guía por los distintos fabricantes.

El DOM por sus siglas en inglés "Modelo de Objetos del Documento" fue diseñado para evitar incompatibilidades.

Actualmente es la fundación Mozilla la que sigue trabajando en la evolución de JavaScript con un mayor empeño.

Aunque JavaScript no es nuevo, pues existe hace más de una década, ha sido precisamente su uso en el desarrollo de aplicaciones AJAX lo que lo ha llevado a resurgir.



De acuerdo a la empresa *tiobe.com* dedicada a las estadísticas de uso de los lenguajes de programación JavaScript actualmente se encuentra dentro de los 10 lenguajes de programación más usados.

Funcionamiento

El lenguaje JavaScript es usado para realizar pequeñas aplicaciones que interactúan dentro de una página HTML. La forma en la que funciona principalmente JavaScript es de modo dinámico y para realizar interacciones con el usuario. La única forma donde puede ser ejecutado JavaScript es mediante un navegador Web, el cual interpreta y realiza según esté programado mediante los llamados eventos o instancias.

Las acciones que generalmente se suele utilizar en JavaScript:

- ✦ Efectos visuales: Donde se puede cambiar el estilo, crear movimientos a objetos y elementos HTML, etc.
- ✦ Interacción con el usuario: Donde podemos realizar instancias a consecuencia de una interacción con el usuario e incluso cambiar el contenido de la página Web.

Características:

- ✦ JavaScript es un lenguaje que lo mismo puede ser utilizado para pequeñas comprobaciones o alteraciones del contenido HTML, como también ser usado con funciones y demás estructuras complejas según requiera el sitio.
- ✦ Algunos consideran que JavaScript no es un lenguaje orientado a objetos por lo que no es necesario crear ni clases, ni métodos, etc., como en lenguajes de programación estructurada como C, Perl, etc.



- ✦ Sintaxis. Es similar a la usada en Java y C, al ser un lenguaje del lado del cliente este es interpretado por el navegador, no se necesita tener instalado ningún compilador adicional.
- ✦ *Lenguaje no extensible.* JavaScript en comparación con Java, no es totalmente extensible. El modelo de JavaScript está formado por un número limitado de objetos básicos, propiedades, métodos y tipos de datos, que son suficientes para desarrollar aplicaciones cliente-servidor.
- ✦ *Basado en acciones.* Es un lenguaje basado en acciones que posee menos restricciones.
- ✦ Lenguaje interpretado. Esto significa que los archivos con el código se ejecutan directamente. Ese tipo de lenguajes son más sencillos y fáciles de aprender. Además, normalmente es más fácil desarrollar, modificar y depurar programas para lenguajes interpretados porque no es necesario recompilar el código cada vez que se realiza un cambio. Los programas para lenguajes interpretados suelen requerir un mayor tiempo de ejecución que los programas compilados. En el caso de JavaScript esto no supone un problema, ya que, con los anchos de bandas actuales, la velocidad de ejecución es despreciable frente a la velocidad de transmisión. Los scripts escritos en JavaScript se compilan en formato Bytecode (como los programas Java) y evalúan mientras se está transmitiendo el script.

Actualmente existen dos tipos de JavaScript:

- ✦ El que se ejecuta en el cliente, este es el Javascript propiamente dicho, aunque técnicamente se denomina Navigator JavaScript.
- ✦ Javascript que se ejecuta en el servidor, es más reciente y se denomina LiveWire Javascript.



Frameworks y librerías

Las aplicaciones Web son cada vez más complejas, ya que incluyen efectos e interacciones que hasta hace poco tiempo eran exclusivas de las aplicaciones de escritorio. Al mismo tiempo, la programación de estas aplicaciones avanzadas se complica por varios motivos.

En primer lugar, las aplicaciones comerciales deben funcionar correctamente e igual de bien en al menos cinco navegadores diferentes: Internet Explorer 6 y 7, Firefox, Opera y Safari. En segundo lugar, el tiempo disponible para el desarrollo de cada nueva característica se reduce cada vez más por la necesidad continua de incluir novedades en las aplicaciones antes de que las incluya la competencia.

Por todo lo anterior, han surgido librerías y frameworks específicos para el desarrollo de aplicaciones con JavaScript. Utilizando estas librerías, se reduce el tiempo de desarrollo y se tiene la seguridad de que las aplicaciones funcionan igual de bien en cualquiera de los navegadores más populares.

La librería en JavaScript jQuery (fig 2.2.3.1) es una de las más sencillas y útiles que se puedan necesitar a la hora de desarrollar un sitio Web. Posee características potentes (selectores de CSS, XPath, con el poder de javascript de crear variables y funciones que interactúen con el documento) y es muy fácil de usar, a su vez es muy liviana. Además trae incorporadas funciones adicionales, como AJAX, manipulación del DOM y detección de navegadores.

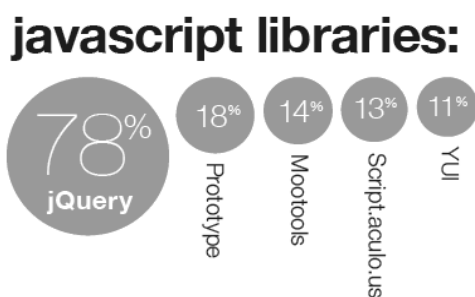


Figura 2.2.3.1 Estadísticas de uso para jQuery (<http://webdirections.org>)



Ventajas

- ✦ El código Javascript se ejecuta en el cliente por lo que el servidor no es solicitado más de lo debido.
- ✦ El lenguaje de scripting es seguro y fiable porque está en claro y hay que interpretarlo, por lo que puede ser filtrado; para el mismo Javascript, la seguridad es casi total y sólo en su primera versión el CIAC (Computer Incident Advisory Committee) señaló problemas de leve entidad, entre ellos la lectura de la caché y de los sitios visitados, de la dirección e-mail y de los archivos presentes en el disco. Sin embargo, estos fallos se corrigieron ya en las versiones de Netscape sucesivas a la 2.0;
- ✦ No requiere tiempo de compilación ya que el lenguaje funciona del lado del cliente, los navegadores son los encargados de interpretar estos códigos.
- ✦ Javascript tiene la ventaja de ser incorporado en cualquier página Web, puede ser ejecutado sin la necesidad de instalar otro programa para ser visualizado.
- ✦ Es un lenguaje independiente de la plataforma.
- ✦ Es simple, no hace falta tener conocimientos de programación para poder hacer un programa en JavaScript y para su desarrollo se necesita básicamente un editor de textos y un navegador Web compatible con JavaScript para sus pruebas.
- ✦ Maneja objetos dentro de nuestra página Web y sobre ese objeto se pueden definir diferentes eventos. Dichos objetos facilitan la programación de páginas interactivas, a la vez que se evita la posibilidad de ejecutar comandos que puedan ser peligrosos para la máquina del usuario, tales como formateo de unidades, modificar archivos, etc.



- ✦ Es dinámico, responde a eventos en tiempo real. Eventos como presionar un botón, pasar el puntero del ratón sobre un determinado texto o el simple hecho de cargar la página o caducar un tiempo. Con esto podemos cambiar totalmente el aspecto de nuestra página al gusto del usuario, evitándonos tener en el servidor un página para cada gusto, hacer cálculos en base a variables cuyo valor es determinado por el usuario, etc.
- ✦ Los programas JavaScript tienden a ser pequeños y compactos no requieren mucha memoria ni tiempo adicional de transmisión.

Desventajas

- ✦ Los script tienen capacidades limitadas, por razones de seguridad, por lo cual no es posible hacer todo con Javascript, sino que es necesario usarlo conjuntamente con otros lenguajes evolucionados, posiblemente más seguros, como Java. Se puede poner en riesgo la seguridad del sitio, con el actual problema llamado XSS (significa en inglés *Cross Site Scripting* renombrado a XSS por su similitud con las hojas de estilo CSS).
- ✦ Una desventaja considerable también es que el código es visible y puede ser leído por cualquiera, incluso si está protegido con las leyes del copyright.
- ✦ El código del script debe descargarse completamente antes de poderse ejecutar, si los datos que un script utiliza son muchos, el tiempo que tardará en descargarse será muy largo.

2.2.4 XML

Un elemento importante más de una aplicación AJAX y del cual es la “X” de su nombre es XML.

XML (*Extensible Markup Language*, lenguaje de marcas generalizado). Es un lenguaje que se utiliza para estructurar información de manera jerárquica en un documento o en



general en cualquier archivo que contenga texto, como puede ser una tabla de datos, configuración de programas o resultado de base de datos ordenados para ser mostrado en aplicaciones AJAX.

Se ha usado en los últimos años debido a ser algo estándar, abierto y libre. Creado por el Consorcio World Wide Web, W3C (los creadores de la WWW), en colaboración de las principales compañías productoras de software. La primera propuesta XML fue en 1996 y la primera especificación apareció en 1998. Desde entonces su uso ha tenido un crecimiento acelerado, uno de las principales razones de su crecimiento ha sido la aparición de AJAX.

Ventajas de su uso

Previamente, ya existían otros lenguajes de marcas, como por ejemplo el HTML, basado en el lenguaje generalizado de marcas (SGML). El problema con el lenguaje generalizado de marcas es que, por ser muy flexible y muy general, se complica el análisis sintáctico de un documento y la especificación de la estructura. XML es más exigente que SGML en la sintaxis, lo que facilita la construcción de librerías para procesarlo (como pueden ser lectores de RSS, script de proceso de feeds, etc.).

Comparado con otros sistemas usados para crear documentos, el XML tiene la ventaja de poder ser más exigente en cuanto a la organización del documento, con lo que se obtienen documentos mejor estructurados.

Por ser posible exigir la estructura que deben tener un tipo determinado de documentos, se vuelve posible extraer información de varios documentos automáticamente, por ejemplo para crear bases de datos o listados con información sobre todos los documentos.



Conceptos básicos de XML

Los archivos XML son archivos de texto, que generalmente utiliza una codificación Unicode, pero se pueden usar otros alfabetos según convenga para la aplicación.

Existen cinco caracteres especiales en XML:

- ✦ Los símbolos menor que (<) y mayor que (>).
- ✦ Las comillas dobles ("").
- ✦ El apóstrofe (').
- ✦ El carácter (&)

Los símbolos mayor que y menor que se usan para delimitar las marcas que dan la estructura al documento. Hay una nomenclatura especial cuando dentro del archivo XML se necesita alguno de los cinco caracteres especiales. Para este tipo de situaciones se hace uso de las diferentes entidades estándar HTML:

- ✦ Para el símbolo menor que (<), se utiliza <
- ✦ Para el símbolo mayor que (>), se utiliza >
- ✦ Para el símbolo comillas dobles (""), se utiliza "
- ✦ Para el símbolo apóstrofe ('), se utiliza '
- ✦ Para el símbolo ampersand (&), se utiliza &

De esta manera, se observa claramente por qué "&" es un carácter especial, se utiliza para representar entidades; una entidad es un carácter adicional que no forma parte del alfabeto que se emplea por defecto en el texto (los caracteres especiales quedan excluidos del alfabeto usado para el texto) comienza por &, seguido del nombre de la entidad e inmediatamente un punto y coma.

Una diferencia importante con SGML, y en particular HTML, es que los nombres de las marcas y de sus atributos distinguen entre mayúsculas y minúsculas; <a> y <A> serían dos marcas diferentes. Generalmente se utilizan únicamente minúsculas para los nombres de las marcas y de sus atributos. Otra diferencia sobresaliente con SGML es que en XML ninguna marca se puede dejar abierta; es decir, por cada marca, por



ejemplo <td>, deberá existir una marca correspondiente </td> que indica dónde termina el contenido de la marca.

Definición del tipo de documento (DTD)

Las posibles marcas que pueden aparecer en un documento XML y los atributos que éstas pueden tener, son definidos en un archivo de definición del tipo de documento (DTD, Document Type Definition).

Cada documento XML debe indicar al comienzo el DTD usado por medio de una marca <!DOCTYPE>, por ejemplo:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

Esta marca indica que lo que viene a continuación en el archivo es una marca “HTML” (con todas sus posibles sub-marcas), que ha sido definida en un DTD que se llama -//W3C//DTD XHTML 1.0 Strict//EN. La palabra clave PUBLIC procede al nombre oficial que se le ha dado al DTD respectivo; en este caso nos da alguna indicación adicional al nombre del DTD: el símbolo “-” indica que no es un DTD reconocido por alguna entidad oficial, en este caso w3.org, el lenguaje usado en el DTD es el inglés (EN) y la sintaxis usada es sintaxis XML. El nombre que viene entre comillas después de PUBLIC es algo arbitrario, pero como en cada sistema existe un catálogo SGML que identifica los DTD disponibles en el sistema, lo importante es usar exactamente el nombre que aparezca en el catálogo. Y para que documentos que usen el mismo DTD puedan ser transportables entre sistemas conviene usar la identificación exacta sugerida por el autor del DTD.

Después del identificador público (lo que esta entre comillas después de PUBLIC) puede venir un identificador del sistema que indica el camino y nombre del archivo



donde se encuentra el DTD; en el ejemplo anterior el identificador del sistema es una URL que indica donde se puede encontrar el DTD usado.

Un archivo DTD define siempre una o más estructuras jerárquicas, con una marca principal, o padre, compuesta por otras marcas o hijos.

Un documento XML que especifique el DTD usado y siga las reglas en él definidas, se dice que es un documento XML válido.

Archivos XML. Se les suele dar un nombre determinado en .xml para identificarlos como XML. Esto es simplemente una convención para los usuarios; el estándar XML 1.0 indica que para identificar un archivo como XML es necesario que la primera línea tenga el siguiente contenido:

```
<?xml version="1.0" ?>
```

Dentro de esta marca puede ir otra información adicional. El alfabeto usado por defecto en los archivos XML es el Unicode; para documentos en español será más conveniente usar el alfabeto latin-1, lo cual se logra usando el atributo *encoding* de la etiqueta xml:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

2.3 Características, ventajas y desventajas de MySQL

Una de las funciones de la parte servidor de una aplicación AJAX es generalmente almacenar y recuperar datos transferidos desde o hacia el cliente. Cuando la información a manipular es de tamaño considerable no es una buena opción almacenarla en archivos de XML y mantenerlos manualmente, por tanto es necesario una base de datos y su correspondiente aplicación para administrarla.

Para que pueda existir una comunicación con el servidor de bases de datos no se usará el protocolo HTTP con los métodos GET y POST sino con el lenguaje que



entiende este tipo de aplicaciones que no es más que SQL (*Structured Query Language*, Lenguaje de consultas estructurado).

Sistema de Gestión Bases de Datos

Por tanto para el desarrollo de la aplicación Web será necesario contar con un elemento dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan, este elemento es el RDBMS (***Relational Database Management System***) o Sistemas de Gestión bases de datos relacionales que (SGBDR en español) se instalan en un equipo y a través de una red los clientes se comunican para trabajar sobre las bases de datos.

El RDBMS se encarga de la privacidad, la integridad, la seguridad de los datos y la interacción con el sistema operativo. Proporciona una interfaz entre los datos, los programas que los manejan y los usuarios finales.

MySQL es el RDBMS más popular entre los producidos bajo los términos del Open Source, debido a su gran estabilidad y que fue diseñado para soportar sistemas de producción de misión crítica con altas cargas de procesamiento. Además existe una versión comercial distribuida y soportada por la compañía MySQL AB.

Características

Veloz. MySQL se caracteriza por su velocidad y estabilidad, rápido, fácil de administrar y usar. Posee un sistema de direccionamiento de memoria rápido basado en *threads* (subprocesos o hilos). Tablas en memoria que son usadas como tablas temporales, logrando con esto gran velocidad de respuesta al realizar las operaciones, esto lo hace ser uno de los gestores con mayor rendimiento.

Open Source. MySQL posee licencia GNU/GPL o Comercial, es Open Source, es decir, software de fuente abierta. Fuente abierta significa que es posible para cualquier



persona usarlo y modificarlo. Cualquier persona puede bajar el código fuente de MySQL y usarlo sin pagar. Cualquier interesado puede estudiar el código fuente y ajustarlo a sus necesidades. MySQL usa el GPL (GNU General Public License) para definir que puede hacer y que no puede hacer con el software en diferentes situaciones. Si no se desea ajustarse al GPL o se requiere introducir código MySQL en aplicaciones comerciales, se puede comprar una versión comercial licenciada.

Tipos de datos. Los tipos de datos que puede haber en un campo, se pueden agrupar en tres grandes grupos: numéricos, fecha-hora y cadenas (caracteres).

Conectividad. Los clientes pueden conectarse al servidor MySQL mediante TCP/IP o sockets de UNIX; utilizando aplicaciones hechas con C, C++, Java, Perl o PHP. Es importante mencionar que MySQL puede conectarse directamente a otros manejadores de bases de datos utilizando ODBC.

Multiplataforma. Funciona sobre múltiples plataformas incluyendo: AIX, BSD, FreeBSD, HP-UX, GNU/Linux, Mac OS X, NetBSD, Novell Netware, OpenBSD, OS/2 Warp, QNX, SGI IRIX, Solaris, SunOS, SCO OpenServer, SCO UnixWare, Tru64, eBD, Windows 95, Windows 98, Windows NT, Windows 2000, Windows XP, Windows Vista, Windows 7 y Windows Server (2000, 2003 y 2008).

Procesamiento. MySQL puede manipular el multiprocesamiento distribuido en uno o más CPU's.

Seguridad. La utilización de privilegios y contraseñas, hace de MySQL un gestor de bases de datos relacionales flexible y muy seguro, todas las claves viajan encriptadas a través de la Web. Mediante la dirección IP del cliente, MySQL puede restringir y administrar el acceso a los recursos del servidor.



Internacionalización. MySQL soporta UNICODE, admitiendo con esto el desarrollo de aplicaciones sin importar la plataforma o el lenguaje del país donde se genere. También permite el manejo interno de mensajes de error.

Almacenamiento dinámico de comandos SQL. Esto hace que MySQL tenga un mejor desempeño que los demás manejadores de bases de datos.

Asimismo soporta:

Operaciones de revisión del estado, las cuales son utilizadas para transferir una base de datos a un determinado estado después de una falla seria del sistema.

VENTAJAS

- **Escalabilidad.** Soporte a grandes bases de datos. Se usa MySQL Server con bases de datos que contienen hasta 50 millones de registros y más de 60,000 tablas. Soporta 64 índices por tabla y cada índice puede estar compuesto de 15 columnas o partes de ellas, con una longitud máxima de 1,000 bytes. Los índices pueden utilizar prefijos de columnas de tipo CHAR, VARCHAR, BLOB o TEXT.
- **Integridad.** Para revisar, optimizar y reparar tablas u otros objetos de una base de datos, MySQL posee comandos y mecanismos internos.
- **Conectividad.** Los clientes pueden conectarse con el servidor MySQL usando sockets TCP/IP en cualquier plataforma. En sistemas Windows de la familia NT (NT, 2000, XP, ó 2003). los clientes pueden usar *named pipes* (conexión de software temporal entre dos programas o comandos) para la conexión. En sistemas Unix, los clientes se pueden conectar usando ficheros socket Unix. En MySQL 5.0, los servidores Windows soportan conexiones con memoria compartida si se inicializan con la opción *shared-memory* (memoria compartida). Los clientes se pueden conectar a través de memoria compartida usando la opción *protocol=memory*. La interfaz para el conector ODBC (MyODBC) proporciona a MySQL soporte para programas clientes que usen conexiones



ODBC (Open Database Connectivity). Por ejemplo, puede usar MS Access para conectar al servidor MySQL. Los clientes pueden ejecutarse en Windows o Unix. La interfaz para el conector J MySQL proporciona soporte para clientes Java que usen conexiones JDBC. Estos clientes pueden ejecutarse en Windows o Unix. El código fuente para el conector J está disponible.

- **Ligero.** Para la elaboración de bases de datos no necesita de grandes requerimientos ya que puede ser ejecutado en una máquina con escasos recursos debido a su bajo consumo, sin problemas.
- **Portabilidad.** Escrito en C y en C++, Probado con un amplio rango de compiladores diferentes, Funciona en diferentes plataformas, Usa GNU Automake, Autoconf, y Libtool para portabilidad, APIs (Interfaz de Programación de Aplicaciones) disponibles para C, C++, Eiffel, Java, Perl, PHP, Python, Ruby, y Tcl, Uso completo de multi-hilo mediante hilos del kernel. Pueden usarse fácilmente múltiple CPU's si están disponibles, Relativamente sencillo de añadir otro sistema de almacenamiento. Esto es útil si se desea añadir una interfaz SQL para una base de datos propia, Las funciones SQL están implementadas usando una librería altamente optimizada y deben ser tan rápidas como sea posible. Normalmente no hay reserva de memoria tras toda la inicialización para consultas, El servidor está disponible como un programa separado para usar en un entorno de red cliente/servidor. También está disponible como biblioteca y puede ser incrustado (linkado) en aplicaciones autónomas. Dichas aplicaciones pueden usarse por sí mismas o en entornos donde no hay red disponible.
- **Diseño Multicapas.** El diseño de MySQL Server es multicapas, con módulos independientes. Algunos de los últimos módulos se listan a continuación con una indicación de lo bien probados que están:

- ✦ Replicación. Hay grandes grupos de servidores usando replicación en producción, con buenos resultados. Se trabaja para mejorar características de replicación en MySQL 5.x.
- ✦ InnoDB tablas. El motor de almacenamiento transaccional InnoDB es usado en grandes sistemas de producción con alta carga de trabajo.



- ✦ BDB tablas. El código Berkeley DB es muy estable, pero todavía se está trabajando para mejorarlo con el interfaz del motor de almacenamiento transaccional BDB en MySQL Server.
- Facilidad de configuración e instalación
- Baja probabilidad de corromper datos

DESVENTAJAS

Aunque MySQL se incluye en el grupo de sistemas de bases de datos relacionales, carece de algunas de sus principales características y aunque algunas de ellas ya se han incluido en las versiones más recientes no se puede asegurar la estabilidad y buen funcionamiento de estas:

- *Subconsultas*. Tal vez ésta sea una de las características que más hacía falta, aunque gran parte de las veces que se necesitan, es posible reescribirlas de manera que no sean necesarias.
- *Triggers*. Aunque éstos ya se incluyen en la última versión, no es posible hablar de una gran estabilidad ya que los *triggers* reducen de forma significativa el rendimiento de la base de datos, incluso en aquellas consultas que no los activan.
- *Procedimientos almacenados*. Los procedimientos almacenados fueron implementos hasta la versión 5, la cual aún presenta algunos problemas de estabilidad.
- *Transacciones*. A partir de las últimas versiones ya hay soporte para transacciones, aunque no por defecto (se ha de activar un modo especial).
- *Integridad referencial*. Aunque admite la declaración de claves foráneas en la creación de tablas, internamente no las trata de forma diferente al resto de los campos.
- *Índices*. MySQL utiliza a menudo índices para realizar operaciones de clasificación rápidamente sin examinar todas las filas. Los índices agilizan la recuperación pero hace más lentas las operaciones de insertar, borrar, así como las actualizaciones de valores en las columnas indexadas, es decir, la mayor



parte de las operaciones que implican escritura porque la escritura no afecta sólo a la fila de datos, sino a menudo también a los índices. Si tiene muchos índices, los archivos pueden alcanzar el tamaño máximo de archivo antes que el de datos.

2.4 Características, ventajas y desventajas de AJAX

Breve historia de AJAX

El término AJAX se presentó por primera vez en el artículo "AJAX: A New Approach to Web Applications (Un nuevo acercamiento a las aplicaciones Web)" publicado por el programador Jesse James Garrett el 18 de Febrero de 2005. Hasta ese momento, no existía un término normalizado que hiciera referencia a un nuevo tipo de aplicación Web que estaba apareciendo. El artículo define AJAX como un acrónimo de Asynchronous JavaScript + XML, que se puede traducir como "JavaScript asíncrono + XML". AJAX no es una tecnología en sí mismo. En realidad, se trata de varias tecnologías independientes que se unen.

Esta forma de programación describe el proceso de utilización del objeto *XMLHttpRequest* para obtener información de un servidor Web de un modo dinámico (asíncronamente). La posibilidad de utilizar *XMLHttpRequest* no es algo novedoso, puede hacerse desde 1998, e incluso algunos navegadores como Internet Explorer 4 ofrecían los medios para utilizar AJAX desde mucho antes. Incluso antes de que existiera la mayoría de los navegadores que se usan actualmente, ya se podía utilizar JavaScript para gestionar instantáneamente las peticiones al servidor. Sin embargo el uso generalizado del concepto AJAX es relativamente reciente.

Cómo trabaja AJAX

AJAX permite una nueva manera de navegar en una página Web. La manera clásica de navegación fue superada por otra forma más interactiva, más rápida y que permite una mejor experiencia de usuario.



Modelo tradicional de navegación Web.

El modelo tradicional o clásico de “experiencia de usuario” consiste en cargar una página, pulsar un enlace o realizar una acción que lleva a otra página, esperar la respuesta y así repetidamente. El inconveniente de este modelo es que se recarga la misma página una y otra vez donde un 90% de los elementos prácticamente permanecen sin cambios, siguiendo el siguiente proceso:

- ✦ El usuario realiza una acción (ej. Enviar formulario).
- ✦ La cabecera http de la página carga otra nueva que tendremos que cargar entera para ver un resultado (ej. Información correctamente enviada)
- ✦ Si estaba trabajando en la página anterior, se tendrá que volver al principio y cargar de nuevo toda la página.

Modelo clásico de aplicación Web (síncrono)

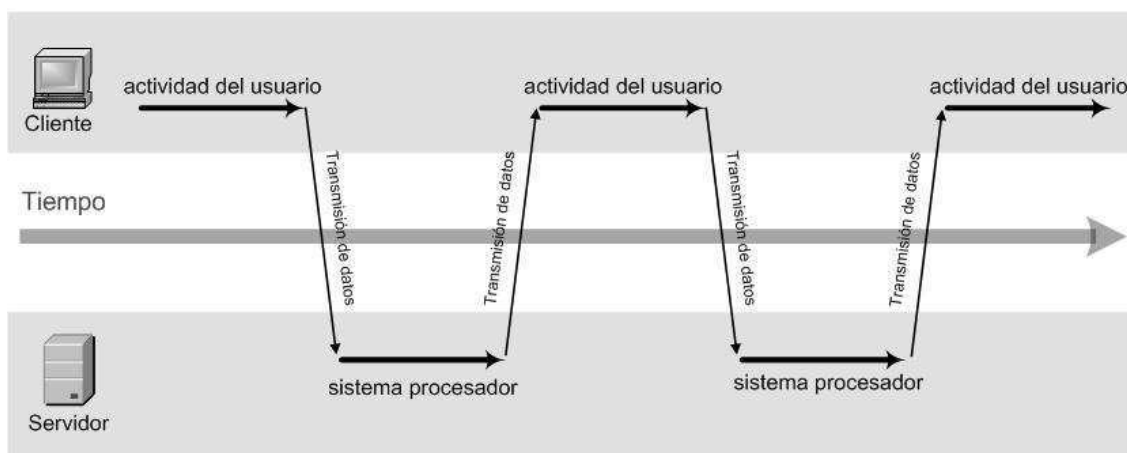


Figura 2.4.1 Modelo clásico de navegación por Internet

De la figura 2.4.1 se observa cómo la navegación del usuario se interrumpe cada vez que la página necesita conectar con el servidor para obtener nuevos datos.



Modelo AJAX. El modelo de navegación AJAX se basa en que las conexiones al servidor ocurren en segundo plano, y no necesitan cambiar de página, con AJAX se puede seguir trabajando en la página Web mientras la petición de información se gestiona en segundo plano.

Modelo Ajax de aplicaciones Web (asíncrono)

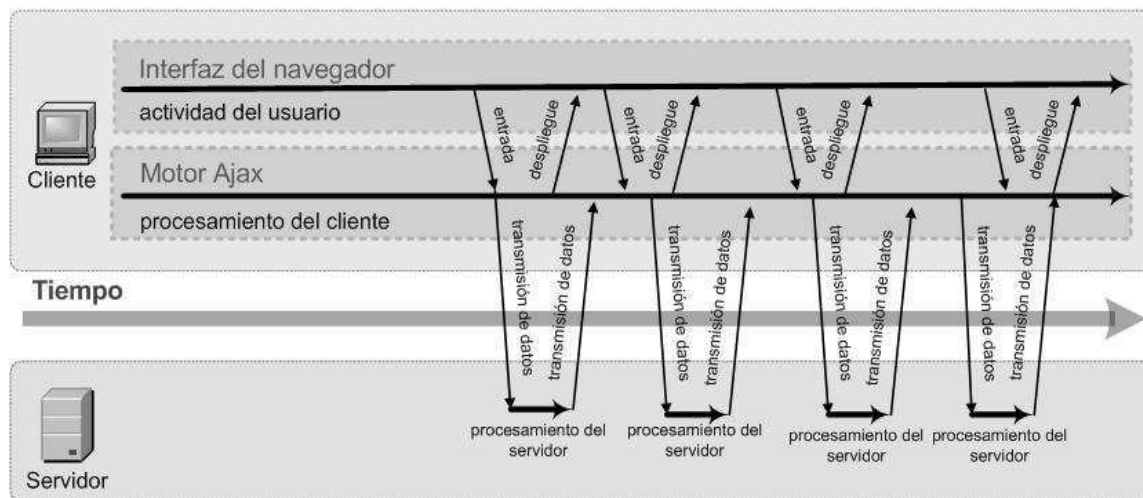


Figura 2.4.2 Modelo navegación AJAX

De la figura 2.4.2 se puede apreciar como las peticiones al servidor son paralelas al tiempo y a la experiencia del usuario, de aquí la cuestión asíncrona, es decir, no existe una espera en que una operación termine para que suceda la siguiente.

Se pueden hacer muchas llamadas y de acuerdo al tiempo de respuesta se irán ejecutando. De acuerdo al modelo clásico una página carga todas las peticiones, en cambio con el modelo AJAX cada petición es tratada de manera individual siguiendo su propio ciclo de vida.

Ciclo de vida de una petición AJAX:

- ✦ Acción: El usuario inserta un texto en un formulario o define unos parámetros de búsqueda.



- ✦ Evento: Es el disparador de la acción como pulsar un botón de formulario, pulsar una tecla, hacer clic en un enlace, etc.
- ✦ Captura: El evento llama al objeto *XMLHttpRequest* que realiza una petición al servidor.
- ✦ Servidor. El servidor ejecuta la acción con los parámetros que recibe del navegador y genera un archivo de respuesta (como lo hace en el modelo clásico de navegación).
- ✦ Filtrado. La respuesta es enviada al navegador cliente donde se procesa al llegar.
- ✦ Reacción. El navegador utiliza los datos recibidos para realizar alguna acción en la página.

Como se menciona al inicio AJAX no es una nueva tecnología sino la utilización de varias tecnologías ya existentes, de una manera coordinada. Se ejecuta en el cliente, es decir, en el navegador del usuario mientras este mantiene una comunicación asíncrona con el servidor en segundo plano.

Las tecnologías que forman AJAX son la mayoría de las tecnologías disponibles para páginas Web (Véase Figura 2.4.3):

- ✦ XHTML y CSS, para crear una presentación basada en estándares de todos los elementos de la aplicación y la información recibida por el servidor.
- ✦ DOM, para la interacción y manipulación dinámica de la presentación.
- ✦ XML, XSLT y JSON, para el intercambio y la manipulación de información, específicamente XML es el formato de los datos transmitidos del servidor al cliente (navegador) y que posteriormente serán mostrados.
- ✦ XMLHttpRequest, para el intercambio asíncrono de información.
- ✦ JavaScript, Para manejar el objeto XMLHttpRequest y para unir todas las demás tecnologías.
- ✦ Lenguaje de servidor, Genera la información útil en XML y la envía al navegador, puede ser PHP o ASP.

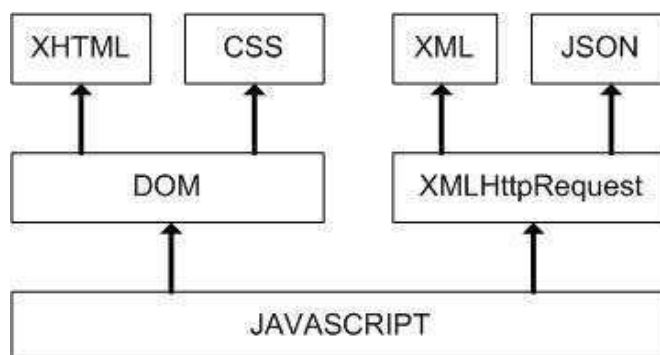


Figura 2.4.3 Tecnologías que forman AJAX

La comunicación asíncrona con el servidor, se logra mediante JavaScript al cargar las peticiones en segundo plano sin intervenir con la visualización de la página mediante el objeto *XMLHttpRequest* que está prácticamente disponible en la mayoría de los navegadores. Como AJAX está basado en estándares abiertos es una técnica válida para múltiples plataformas y sistemas operativos.

Para comprender bien el funcionamiento de AJAX es necesario primero conocer algunos conceptos básicos de las aplicaciones Web, como el procesamiento de las peticiones de los sitios Web y los tipos de respuesta que reciben del servidor.

El estándar que utilizan actualmente los navegadores para solicitar información a los navegadores Web se denomina HTTP (Hypertext Transfer Protocol). Este es el método que utilizan los navegadores Web para enviar una petición desde un sitio Web y a continuación recibir la respuesta del servidor en el que está alojado ese sitio. Al enviar una petición se incluye junto a ésta diversos encabezados que permiten al servidor saber exactamente que información debe servir y como gestionar la petición. Una vez recibida la petición, el servidor decide cómo debe contestar. Existen diversos códigos de respuesta (ver Figura 2.1.2.3). También existen varios métodos para realizar una petición, en la tabla 2.4.1 se muestran todos los métodos de petición disponibles (aunque generalmente se usan GET y POST).



Método	Descripción
GET	Obtener, Recuperar. Codifica el mensaje. Lo envía dentro de un string de consulta (query string) La información se añade a la URL del recurso del servidor simplemente solicita un recurso determinado al servidor. es la forma más frecuente de realizar una solicitud.
POST	Permite realizar la petición junto con los datos introducidos por el usuario (ideal para formularios Web). Sube datos al Servidor. Envía su petición en el cuerpo del mensaje (message body). Tampoco lo encripta.
HEAD	Es similar a GET, salvo porque la respuesta se enviará sin el cuerpo; es útil para obtener de forma rápida encabezados.
DELETE	Solicita la eliminación de un documento concreto.
TRACE	Devuelve una copia de la petición para poder controlar su progreso.
OPTIONS	Devuelve la lista completa de los métodos admitidos; es útil para averiguar qué métodos admite el servidor.
CONNECT	Es una petición de tipo Proxy que se utiliza en comunicaciones cifradas por túnel SSL.

Tabla 2.4.1 Métodos de peticiones HTTP

El funcionamiento del objeto XMLHttpRequest es el motor de AJAX para que pueda llevara cabo sus funciones en segundo plano.



El objeto XMLHttpRequest (XHR)

Antes de AJAX se usaba IFRAME para incrustar código de otra página o llamadas al servidor dentro del documento, pero con un IFRAME se carga una página Web entera.

El objeto XMLHttpRequest (Figura 2.4.4) que solucionaría este problema, al principio apareció como una creación de Microsoft.

Era un objeto ActiveX diseñado para poder usarse con Outlook. Entonces se llamó XMLHTTP, más adelante otros navegadores lo fueron incorporando, hasta que el consorcio W3C lo incluyó en sus estándares como XMLHttpRequest.

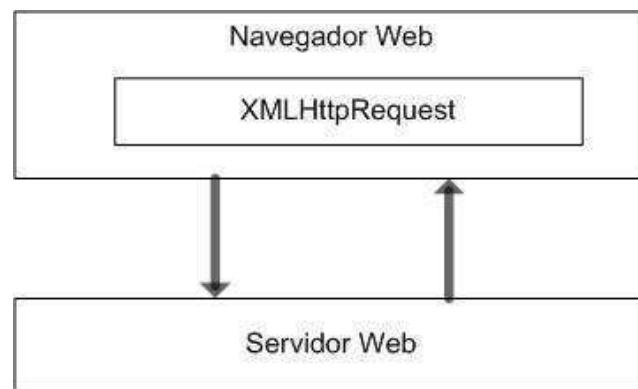


Figura 2.4.4 Objeto XMLHttpRequest

Esquema Cliente-Servidor, con instancia XMLHttpRequest

Los formatos que tienen los datos intercambiados entre el servidor Web puede ser XML o texto plano. Los datos devueltos por la llamada a XMLHttpRequest serán, por lo general obtenidos desde bases de datos en el servidor o un proceso de algún sistema Web (PHP, JSP, ASP, etc.)



Métodos del objeto XMLHttpRequest

- ✦ *setRequestHeader()*. Añade un par etiqueta/valor al encabezado en el momento de enviarlo.
- ✦ *send()*. Enviar la petición actual.
- ✦ *abort()*. Detiene la petición. Es útil cuando se quiere que una petición no dure más de un cierto periodo de tiempo, se puede aplicar para interrumpirla prematuramente.
- ✦ *getAllResponseHeader()*. Este método se puede usar para obtener la información completa sobre los encabezados http utilizados.
- ✦ *getResponseHeader* ("nombreEncabezado"). Puede utilizarse este método para obtener el contenido de un elemento concreto del encabezado. Útil para extraer los datos de la cadena, que contiene todos los encabezados. Por ejemplo, para conocer el tamaño del documento solicitado (*getResponseHeader*("Content-Length")).
- ✦ *Open* ("método", "URL", "async", "nombreUsuario", "clave"). Es el método más importante para utilizar el objeto XMLHttpRequest. Se utiliza para establecer una conexión con un archivo concreto del servidor. No todos los argumentos de esta función son obligatorios y se usan dependiendo de cada situación.
- ✦ *setRequestHeader* ("etiqueta", "valor"). Con este método se puede incluir en el encabezado de la petición un par etiqueta/valor. Se debe utilizar este método después de haber ejecutado el método *open()*, y siempre antes del método *send()*.
- ✦ *Send* ("contenido"). Es el método responsable de enviar realmente la petición al servidor. Si esta petición se envía de forma asíncrona, la respuesta se obtendrá de forma inmediata; en caso contrario no se obtendrá la respuesta hasta que haya sido completamente recibida.

Y en la Tabla 2.4.2 se muestra una tabla de las propiedades del objeto XMLHttpRequest.



Objeto XMLHttpRequest	
Propiedad	Descripción
onreadystatechange	Se utiliza como gestor de eventos para detectar cambios de estado.
readyState	Contiene el estado actual del objeto. 0: sin inicializar 1: cargándose 2: cargado 3: en interacción 4: completado
responseText	Contiene la respuesta en formato de cadena.
responseXML	Contiene la respuesta en formato XML.
Status	Contiene el estado de la petición en formato numérico.
statusText	Contiene el estado de la petición, pero en formato de cadena. (ej. Error 404 devolverá "Not found")
onreadystatechange	Es un gestor de eventos que nos permite ejecutar cierto bloque de código, o cierta función, cuando se produce algún cambio en el estado del avance del proceso.
readyState.	Esta propiedad indica con precisión en qué estado se encuentra actualmente la petición. Puede usarse cuando se desea ejecutar cierto bloque de código cuando la petición se encuentra en estado de carga y detenerlo cuando se alcanza el estado de completada.
responseText	Esta propiedad se devuelve una vez completada la petición.
responseXML	Es similar a la anterior pero más apropiada cuando se tiene la certeza de que la respuesta vendrá en formato XML, especialmente si se tiene pensado utilizar la funcionalidad del navegador para gestionar el formato XML.
status	Contiene el código de respuesta generado por el servidor ante la petición.
statusText	Es una variante textual de la propiedad status.

Tabla 2.4.2. Propiedades del objeto XMLHttpRequest.

La transferencia de datos entre cliente y servidor involucra una actualización de las páginas completas en cada solicitud, eso significa una gran cantidad de datos desde el



servidor y cliente y el refresco del navegador de todo el contenido que se muestra en pantalla en ese momento.

Para solucionar esto los *applets* de Java y programas Flash proporcionan una mayor interactividad sin comunicaciones continuas con el servidor y transferencias de páginas completas. La contraparte de esto es que se necesita la instalación de complementos en el navegador aparte de no ser basados en estándares y tener algunos inconvenientes relacionados con la accesibilidad.

Es aquí donde entra AJAX, es independiente del servidor, no es necesario instalar complementos en el lado del cliente, está basado en estándares y no es necesario transferir ni actualizar páginas completas.

Las interfaces que se pueden crear se comportan como elementos dinámicos que se comunican esporádicamente con el servidor para obtener porciones de datos específicos y no páginas completas.

Programación AJAX frente a los programas nativos

Ventajas

- ✦ No se necesita instalar una aplicación AJAX, solo es necesario introducir la URL en el navegador y puede comenzarse a trabajar.
- ✦ Puede accederse a la última versión de la aplicación con solo actualizarla en el servidor Web; ya no es necesario actualizaciones individuales en los clientes.
- ✦ Es independiente del sistema operativo, solo es necesario un navegador.
- ✦ La curva de aprendizaje del uso de una aplicación AJAX es bastante rápida.
- ✦ Una aplicación AJAX utiliza menos recursos, espacio en disco y memoria que un programa nativo.
- ✦ La migración de una aplicación AJAX a otras arquitecturas de hardware y software es más sencillo que el caso contrario.



Desventajas

- ✦ Las aplicaciones AJAX, en comparación con los programas nativos, tiene acceso limitado a los recursos locales de la computadora.
- ✦ Las aplicaciones AJAX no pueden acceder a bases de datos locales como lo hacen los programas nativos sino que debe transferir la información a través de redes generalmente más lentas.

Comparación frente a las aplicaciones Web clásicas

Ventajas

- ✦ Para las peticiones de los usuarios una aplicación AJAX no recarga la página Web completas sino solo las partes de la página que actúan como interfaz.
- ✦ En una petición al servidor solo se solicitarán los datos necesarios y no una nueva página completa, siendo más ágil que una aplicación Web clásica.
- ✦ No es necesario instalar complementos ni descargar programas pesados en el cliente como *applets* Java o Flash.
- ✦ La interacción del usuario con una aplicación AJAX es muy parecida a la que ocurre con un programa nativo.

Desventajas

- ✦ El navegador no muestra visual y automáticamente cuando la aplicación AJAX está esperando la recepción de una nueva página.
- ✦ No hay integración con los botones de avanzar y retroceder del navegador (aunque pueden solucionar en las aplicaciones AJAX preparando código adecuado para ello).
- ✦ Si el usuario desactiva el motor JavaScript del navegador las aplicaciones AJAX no funcionarán.
- ✦ Las aplicaciones AJAX son relativamente más complejas de desarrollar.



2.5 Metodologías y tecnologías similares a AJAX para el desarrollo de aplicaciones Web.

Aplicaciones RIA's

Las aplicaciones RIA (Rich Internet Applications, Aplicaciones de Internet enriquecidas) son aplicaciones Web de última generación con más ventajas que las tradicionales aplicaciones Web.

Estas surgen como una combinación de las ventajas que ofrecen las aplicaciones Web y las aplicaciones tradicionales ofreciendo una mejor experiencia de usuario.

Normalmente en las aplicaciones Web, hay una recarga continua de páginas cada vez que el usuario pulsa sobre un enlace. De esta forma se produce un tráfico muy alto entre el cliente y el servidor, llegando muchas veces, a recargar la misma página con un mínimo cambio.

Otra de las desventajas de las tradicionales aplicaciones Web es la poca capacidad multimedia que posee, por ejemplo, para ver un vídeo es necesario usar un programa externo para su reproducción.

Las capacidades multimedia son totales gracias a que estos entornos tienen reproductores internos y no hace falta ningún reproductor del Sistema Operativo del usuario.

Existen muchas soluciones para crear RIAs, una técnica muy conocida es AJAX, sin embargo existen hoy en día más herramientas para la creación de entornos RIA. Entre estas se puede mencionar las plataformas Adobe Flash, Adobe Flex y Adobe AIR de Adobe, AJAX, OpenLaszlo, Silverlight de Microsoft, JavaFX Script de Sun Microsystems, Bindows y Javascript.



Características:

A pesar de que el desarrollo de aplicaciones multimedia para navegadores Web está mucho más limitado y es más difícil que otro tipo de aplicaciones de escritorio, los esfuerzos se justifican por varios motivos:

- ✦ No necesitan instalación (solo es necesario mantener actualizado el navegador Web).
- ✦ Las actualizaciones hacia nuevas versiones son automáticas.
- ✦ Se pueden utilizar desde cualquier ordenador con una conexión a Internet sin depender del sistema operativo que este utilice.
- ✦ Generalmente es menos probable la infección por virus, que utilizando por ejemplo programas ejecutables.
- ✦ Más capacidad de respuesta, ya que el usuario interactúa directamente con el servidor, sin necesidad de recargar la página.
- ✦ Ofrecen aplicaciones interactivas que no se pueden obtener utilizando solo HTML, incluyendo arrastrar y pegar, cálculos en el lado del cliente sin la necesidad de enviar la información al servidor.
- ✦ Evita la problemática del uso de diferentes navegadores al abstraerse de ellos a través de un framework.
- ✦ Reducen costos de desarrollo y añaden valor a la empresa.
- ✦ Son más fáciles de mantener.
- ✦ Tienen un alcance mayor y se pueden ampliar más fácilmente
- ✦ El usuario interactúa en todo momento con la aplicación.
- ✦ Reduce los procesos y tiempos de ejecución.
- ✦ Rendimiento: puede ser mejorado dependiendo de la aplicación y de las características de la red. Aplicaciones que pueden procesar localmente en el cliente evitan viajes hacia el servidor lo que incrementa su rendimiento. Dándole más trabajo al cliente también se puede incrementar el rendimiento del servidor. Por el contrario los recursos necesarios hacen difícil que las aplicaciones puedan ejecutarse en dispositivos pequeños, móviles o embebidos



Debido al gran crecimiento de este tipo de aplicaciones, hoy en día existen, como se menciono anteriormente, una gran variedad de herramientas que disputan este mercado, entre ellas se mencionan las siguientes:

2.5.1 Silverlight

Silverlight es un producto desarrollado por Microsoft, que permite crear aplicaciones RIA.

Características

Desde un punto de vista técnico, no es más que un complemento que se instala en nuestro equipo (ocupa menos de 5MB) y hacen uso de él todos los navegadores. Actualmente, los sistemas operativos soportados oficialmente por parte de Microsoft son:

- ❖ Windows 2000 y posteriores
- ❖ Mac OS 10.4 y 10.5

Y para futuras versiones: Nokia S60, Windows Mobile, Mobin, etc.

Por otra parte la lista de navegadores soportados es:

- ❖ Internet Explorer 6 y posteriores
- ❖ Firefox
- ❖ SeaMonkey
- ❖ Safari
- ❖ Opera
- ❖ Google Chrome

El mayor potencial de Silverlight está en el entorno de desarrollo, Silverlight incluye un subconjunto de la plataforma .NET de Microsoft y con ello toda la potencia que ello implica. Como entorno preferido para el desarrollo, bajo Windows, tenemos Visual



Studio (desde la versión gratuita, la Express, hasta el entorno de gestión de ciclo de vida Team System). Además para otros entornos, Mac y Linux, tenemos MonoDevelop y Eclipse (instalando las recientes Eclipse Tools For Silverlight).

Microsoft ha intentado enfocarlo hacia aplicaciones empresariales, juegos casuales y contenido multimedia.

2.5.2 OpenLaszlo

OpenLaszlo es una plataforma open source de aplicaciones ricas de Internet (RIA), que fue fundado en el año 2000 como Laszlo Systems, pero a mediados del año 2004 fue Open Source.

La plataforma consta de dos piezas: una declarativa XML basada en un lenguaje llamado LZX que es usado para escribir aplicaciones clientes, y una aplicación servidor (y LZX Compilador) llamado el servidor presentador.

OpenLaszlo es una plataforma de código abierto para el desarrollo y distribución de RIA's. Ha sido publicado bajo la licencia Common Public License, certificada por la *Open Source Initiative*. La plataforma OpenLaszlo consiste en el lenguaje de programación LZX y el Servidor OpenLaszlo:

Ventajas

- ✦ En cuanto a Usabilidad, proporciona al desarrollador la oportunidad de implementar una aplicación que le sea muy sencilla de usar al usuario sin necesidad de formación y asistencia.
- ✦ No es necesario la carga completa de una página para hacer un pequeño cambio, se hace de manera inmediata y en el área que se desea se realiza el cambio.



Desventajas

- ✦ OpenLaszlo sólo se muestra a través de Flash, por ende, afecta al rendimiento al lado del cliente – si este no cuenta con una conexión de banda ancha-, debido a que Flash originalmente fue diseñado para presentar películas o animaciones, pero no fue pensado para realizar cálculos numéricos o manipulaciones de datos.
- ✦ Lo otro es por el lenguaje a usar “LZX”, es aventurarse a programar en este lenguaje ya que implica implementar librerías, debido a que existen pocas.

2.5.3 Mozilla Prism

Mozilla Prism, anteriormente conocido como WebRunner, es un producto de Mozilla actualmente en desarrollo que integra aplicaciones Web con el entorno de escritorio, lo que permite a las aplicaciones Web ejecutarse desde el escritorio y configurarse independientemente del navegador Web por defecto.

El software se encuentra actualmente en fase beta.

Mozilla Prism fue la primera de estas aplicaciones en llegar al gran público, es la apuesta de Mozilla. Estas son las principales cosas a tener en cuenta:

- ✦ Es multiplataforma: Windows, Linux y Mac funcionan perfectamente.
- ✦ Su motor interno es el de Firefox, por tanto la compatibilidad con casi todas las páginas está garantizada. Por lo contrario, las extensiones que se tengan que usar no funcionarán de manera directa, y aunque es posible hacerlas funcionar, es algo un tanto laborioso.
- ✦ Existe en dos versiones: como extensión para Firefox o como aplicación independiente. Las dos maneras funcionan de forma similar, con las mismas opciones.
- ✦ Posibilidad de ocultar o no la barra de navegación.
- ✦ Permite elegir cualquier imagen como icono de la aplicación que se creará.



- ✦ Tiene soporte para notificaciones, como por ejemplo cuando llegue un email; también nos puede avisar en la barra de tareas/Dock/bandeja de notificación cambiando el icono para mostrar el número de elementos sin leer.
- ✦ Integración con la bandeja de notificación, a la que podemos minimizar la aplicación.

2.5.4 Adobe AIR.

Adobe AIR es una plataforma que permite crear aplicaciones que funcionen a la vez en múltiples Sistemas Operativos. Se piensa en ello como una página Web: se verá en Windows, Mac, Linux, etc. Adobe AIR es una especie de plugin que se instala en la computadora, de una manera similar al plugin de Flash del mismo Adobe.

Una vez instalado se pueden descargar aplicaciones en formato .air que se instalarán de una manera sencilla y parecida a los típicos .exe.

Estas aplicaciones .air sirven para cualquier sistema operativo que tenga AIR instalado, de tal manera que creándolas una sola vez nos sirve para multitud de equipos.

Ventajas

- ✦ Además de ser multiplataforma “por defecto”, las aplicaciones en AIR se desarrollan muy rápidamente y de manera muy sencilla. Esto se debe principalmente a que está basado en HTML/Javascript/Flash, por tanto si se tiene experiencia en el desarrollo de páginas Web resultará muy simple crear una aplicación en AIR.
- ✦ Respecto a una página Web, al ser una aplicación “nativa” no se necesitará tener un navegador abierto, se integrará en el sistema como una aplicación más y se podrá trabajar en local sin tener una conexión a Internet.



Desventajas

- ✦ Como no es una aplicación nativa de verdad, no son muy eficientes y puede tener peor estabilidad y mayores gastos de recursos que una nativa. Además, la integración no es tan buena como las nativas 100%, y aunque el aspecto suele ser espectacular no está muy a la par con el sistema en general.

2.5.5 JavaFX

JavaFX es una familia de productos y tecnologías de Sun Microsystems, anunciados en la conferencia de desarrolladores JavaOne en Mayo de 2007.

Los productos JavaFX se han pensado para ser usados en la creación de Rich Internet Applications (RIAs). Actualmente JavaFX se compone de JavaFX Script y JavaFX Mobile, aunque hay más productos JavaFX planeados.

JavaFX se ha creado para competir en el espacio que ya ocupan Flash, de Adobe, y Silverlight, de Microsoft.

Ventajas

- ✦ JavaFX es una extensión a la plataforma Java que ofrece interactividad, animación y programación compatible con AJAX, Flash de Adobe y Silverlight de Microsoft. A diferencia de éstas anteriores, utiliza los runtimes Java ya instalados en el cliente local en lugar de JavaScript.
- ✦ JavaFX permite a los creadores de contenido crear contenido rico en medios, lo que incluye el arrastrar y soltar contenido de equipos de escritorio o móviles hacia el escritorio, algo que no es posible en ninguna otra RIA.
- ✦ JavaFX se trata de un sistema de desarrollo para la Web que pretende competir con el Flash de Adobe y el Silverlight de Microsoft, con el añadido de que es Open Source.



- ✦ Millones de desarrolladores, muchos equipos con Java instalado y a eso hay que sumar que Java es libre bajo GPL.
- ✦ Multiplataforma, desde el escritorio (corriendo sobre una máquina virtual Java SE) hasta los dispositivos móviles para los que habrá un JavaFX Mobile.
- ✦ Utiliza el mismo lenguaje para la Web, para el escritorio y para la telefonía móvil.
- ✦ Tiene como objetivo que el navegador opere mejor que con JavaScript, que es lento y se implementa en formas diferentes de un navegador a otro.

Desventajas

- ✦ Con JavaFx, se agrega un lenguaje de script complejo, que necesita mantener en memoria su propia estructura semántica y sus propios objetos de plataforma, simultáneamente con los objetos gráficos.
- ✦ Al ser un lenguaje de script agrega un factor de ralentización importantísimo, por eso el desempeño no es óptimo.
- ✦ Por otro lado, las APIs de tratamiento de XML ya vienen incorporadas al JDK, generalmente están implementadas a bajo nivel, por lo que un sistema de templates es mucho mas liviano que soportar JavaFX.

Finalmente después de analizar las tecnologías similares a AJAX podemos observar la gran ventaja sobre las anteriores descritas ya que AJAX en nativo en los “navegadores Web” es el único “RIA Framework” que puede ser encontrado por los diferentes motores de búsqueda.