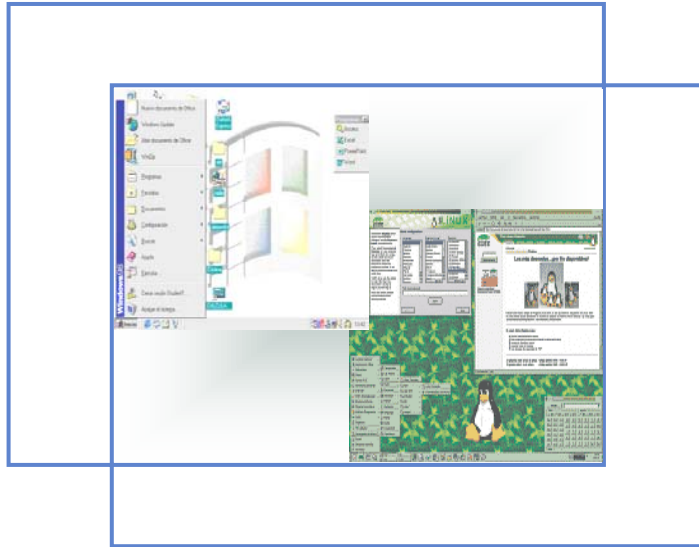


Capítulo 1



WINDOWS Y LINUX COMO PLATAFORMAS DE INTERNET

- ❖ Introducción
 - ❖ Windows como servidor de Internet
 - ❖ Requerimientos
 - ❖ Características
 - ❖ Arquitectura
 - ❖ Linux como servidor de Internet
 - ❖ Requerimientos
 - ❖ Características
 - ❖ Arquitectura
 - ❖ Windows y Linux como alternativas propietaria y de código abierto
 - ❖ Windows como alternativa propietaria
 - ❖ Linux como alternativa de código abierto
 - ❖ Comparación de Windows y Linux

INTRODUCCIÓN

Dentro del mundo de Internet, un elemento clave a nivel técnico es la utilización de tecnologías que soporten los servicios básicos de Internet.

Por eso la importancia de estudiar en este capítulo a las tecnologías de sistemas operativos. Así mismo, hacer un estudio que permita comparar las dos tendencias tecnológicas en Internet: Microsoft y Linux.

Cabe aclarar que en este capítulo nos enfocaremos a los sistemas operativos a utilizarse en los servidores Internet y no en las computadoras cliente con acceso a Internet.

Para el caso de los servidores sí es relevante la selección del sistema operativo debido a que dependiendo de éste, se escoge las herramientas y tecnologías para la programación y publicación de las aplicaciones web.

1.1 WINDOWS 2000 SERVER COMO SERVIDOR DE INTERNET

Este sistema operativo fue creado por la compañía Microsoft.

Microsoft realizó un cambio de nomenclatura para su sistema Windows NT. Así, Windows NT 5.0 pasó a llamarse Windows 2000.

Hasta la versión 4.0 Windows NT se comercializaba en tres versiones: Workstation, Server y Advanced Server. Desde Windows 2000, también se perdió la nomenclatura Workstation y Server, quedando de la siguiente manera[WWW-01]:

- **Windows 2000 Professional:** Windows 2000 Pro, sucesor de Windows NT Workstation, está destinado a ser un cliente de red seguro y una estación de trabajo corporativa. Soporta hasta 2 procesadores y es útil, como sistema operativo autónomo, para correr aplicaciones de alto rendimiento, especialmente en diseño gráfico, por ejemplo.
- **Windows 2000 Server:** Sucesor de Windows NT Server, soporta hasta 4 procesadores y está destinado a ser el servidor de impresión, archivos, aplicaciones, incluso Web, de una empresa pequeña a mediana.
- **Windows 2000 Advanced Server:** Sucesor de Windows NT Server Enterprise Edition, soporta hasta 8 procesadores y será el servidor departamental de aplicaciones en empresas medianas a grandes, con más de un dominio y tareas de misión crítica. Entre otras prestaciones, se incluye soporte para **RAID** y tolerancia a fallos.
- **Windows 2000 Data Center Server:** Soporta hasta 32 procesadores y sólo se entrega sobre pedido. Está destinado a grandes empresas que requieran *Data Warehousing*, análisis econométricos, simulaciones científicas y de ingeniería a gran escala, etc.

1.1.1 REQUERIMIENTOS DEL SISTEMA

Los requerimientos para la instalación de Windows 2000 Server se muestran en la **Tabla 1.1**. [WWW-02]:

Tabla 1.1 Requerimientos de Hardware para Windows 2000 Server

PROCESADOR	Pentium 133 MHz o Pentium de mayor capacidad
MEMORIA	256 MB de RAM mínimo recomendado [128 MB mínimo soportado; 4 GB máximo]
DISCO DURO	2 GB de disco duro con un mínimo de 1.0 GB de espacio libre. (Se necesita este espacio libre en el disco duro si el sistema se instala sobre una red). EIDE, SCSI
UNIDAD DE DISCO	Unidad de disco CD-ROM o DVD-ROM
TARJETAS	Tarjeta de interfaz de red Tarjeta de video VGA o superior

1.1.2 CARACTERÍSTICAS

Dentro de las características de Windows 2000 Server, se pueden citar [WWW-03] [WWW-04]:

- **Multitarea:** Los ciclos de la **CPU** se dividen y se ponen a la disposición de muchos subprocesos al mismo tiempo mediante un método conocido como “alternativa de contexto”. En este método, la **CPU** ejecutará un subproceso hasta que se complete o hasta que el sistema operativo lo indique.
- **Multiplataforma:** Las versiones del sistema operativo Windows NT se generaron para las arquitecturas X86 de Intel, Alpha de Digital Computer, PPC de Motorola y el procesador MIPS, aunque las últimas dos versiones fueron abortadas.
- **Multiprocesamiento:** Windows 2000 Server asigna niveles de prioridad a los diferentes procesos que ejecuta. Existen 32 niveles, los números grandes se ejecutan con mayor preferencia de acceso al procesador. Windows 2000 Server también brinda la capacidad de ejecutarse en 2 o más procesadores al mismo tiempo.
- **Fiabilidad:** La fiabilidad y la capacidad de gestión se han mejorado con herramientas que ayudan a los usuarios y administradores de red a gestionar de forma más sencilla sus sistemas. Por ejemplo: Windows 2000 Server permite que las **DLL** se instalen en los

directorios de sus aplicaciones específicas, y evita que se eliminen las *DLL* compartidas.

- **Escalabilidad:** Windows 2000 Server está diseñado para escalar desde pequeñas estaciones de trabajo a grandes instalaciones de centros de datos.
- **Rendimiento:** La plataforma Windows 2000 Server, junto con la familia de servidores Microsoft, dispone de registros de rendimiento de transacciones por minuto *TPC*.
- **Manejabilidad y Aceptación:** Windows 2000 Server es fácil de implementar, administrar y usar, y además tiene mayor grado de aceptación entre los usuarios finales.
- **Integración:** Windows 2000 Server se integra con miles de dispositivos hardware y sistemas software de otras empresas.
- **Gestión de este sistema:** La gestión global de un sistema se realiza a través de un módulo denominado Administración del equipo, que organiza los recursos, servicios, dispositivos de almacenamiento y seguridad que utilizan tanto en el sistema local como en ordenadores remotos. El panel es una herramienta muy valiosa para los administradores de red y se divide en tres módulos: Herramientas del Sistema, Almacenamiento y Servicios y Aplicaciones.
- **Infraestructura de seguridad:** Puesto que se trata de un sistema operativo orientado al trabajo en red y a compartir recursos, la infraestructura de seguridad funciona en tres niveles:
 1. **Local.** Se refiere a la protección de datos en el ordenador. El sistema está diseñado para evitar que usuarios no autorizados no omitan el sistema de arranque y, por tanto, también las funciones de seguridad. Algunos fabricantes de hardware integran sistemas de contraseña, una solución no muy adecuada para entornos de trabajo compartido. La encriptación de los datos en el disco *NTFS* es un servicio que se basa en la arquitectura CriptoAPI de Windows para implementar el sistema de llaves públicas. Cada archivo (incluyendo sus temporales de trabajo) se encripta a través de una llave generada aleatoriamente, utilizando algoritmos asimétricos.
 2. **Corporativo.** Se refiere a la protección de datos en una red local. Windows 2000 Server utiliza el protocolo de autenticación Kerberos versión 5, un estándar de seguridad en redes locales e Intranets que verifica y hace un seguimiento de la actividad de cada usuario dentro de la red.
 3. **Público.** Utiliza también sistemas de llaves públicas y protocolos de autenticación para mantener la seguridad de las comunicaciones que se realizan por Internet, de

forma que verifique la procedencia de mensajes de correo o garantice las fuentes de donde proceden las descargas. Por otra parte, incluye soporte para **VPN**, protocolos encapsulados que crean un canal de comunicación privado a través de redes públicas.

- **Active Directory:** Active Directory es el servicio de directorio incluido en Windows 2000 Server. Un servicio de directorios es un servicio de red que identifica todos los recursos en ella y los vuelve accesibles a los usuarios y a las aplicaciones que los organiza jerárquicamente en dominios.
- **Servicios de Windows 2000 Server**
 - a) Internet Information Services.
 - b) Servicios de Terminal.
 - c) Servicios de Componentes.
 - d) Servicios de Acceso Remoto.
- **Administración de Windows 2000 Server**
 - a) Manejo del Registro.
 - b) Tolerancia a Fallos.
 - c) Herramientas de seguridad y bitácoras del sistema.
- **Arquitectura de sistema más confiable:** Mejora de claves al sistema operativo, para que sea arquitectónicamente más confiable, como protección de memoria avanzada para el núcleo y firma de controlador.
- **Reducción de los reinicios requeridos:** Un gran número de tareas administrativas que requieren de reinicio en el sistema Windows NT Server 4.0 ya no requieren reinicio en Windows 2000 Server. Esto ayuda a incrementar la disponibilidad del sistema para los usuarios finales.
- Soporte Real para Plug and Play.
- Mayor integración con Internet e Intranet.
- Está destinado a ser el servidor de impresión, archivos, aplicaciones e, incluso Web de una empresa pequeña a mediana que incluye:
 - a) Servidor Web y **FTP** (integrados en Internet Information Server)
 - b) Protocolos estándares de Internet
 - c) Seguridad totalmente integrada (**IPSec**, Kerberos, etc)
 - d) Compartir la conexión de Internet

e) Red Privada Virtual

1.1.3 ARQUITECTURA

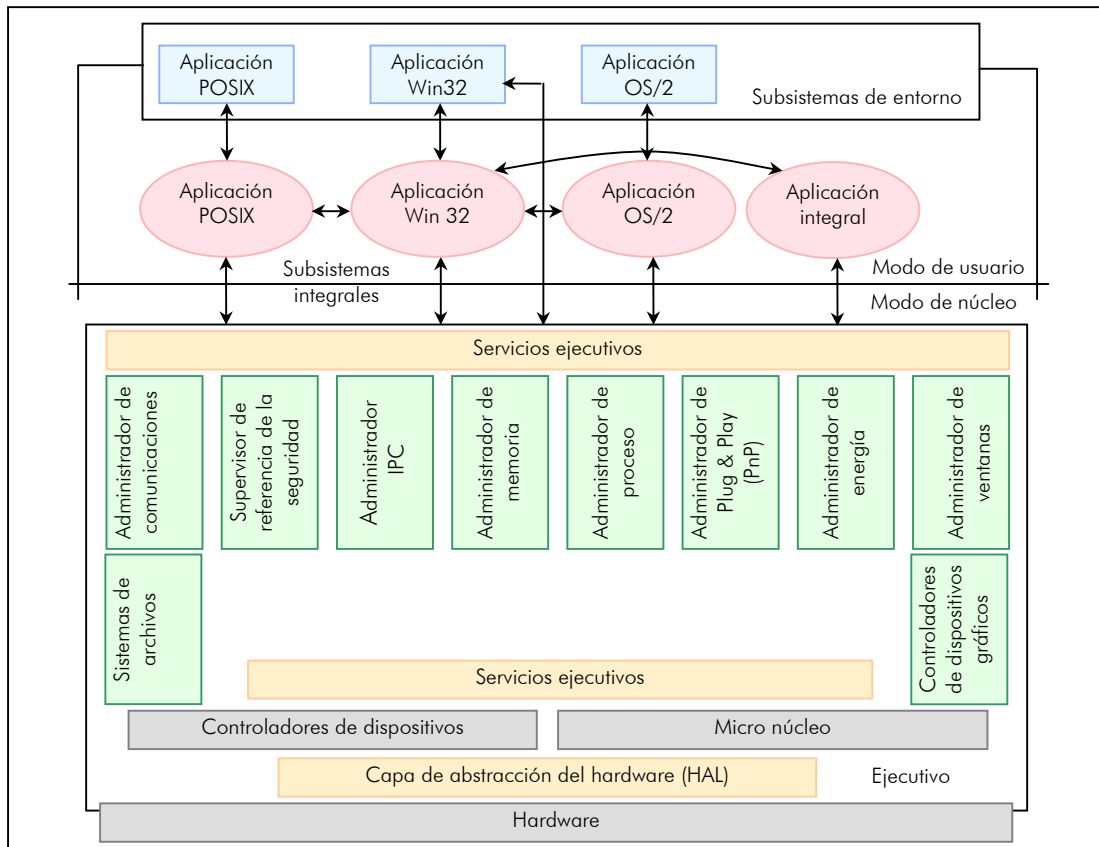


Fig. 1.1 Arquitectura de Windows 2000 Server

La arquitectura interna de Windows 2000 Server consta de un conjunto de módulos, cualquiera de los cuales puede ser actualizado y mejorado internamente sin requerir de un gran reacondicionamiento de todo el sistema operativo, como se puede observar en la figura Fig.1.1.

Al diseñar un pequeño núcleo (la parte del sistema que controla a la CPU) rodeado por un conjunto de módulos interactivos, el sistema operativo aísla los más importantes procesos de la CPU y de comunicación con el hardware que son iniciados por las aplicaciones. A su vez, cada aplicación se ejecuta en su propio espacio protegido de memoria, lo que significa que Windows 2000 Server tiene la capacidad de ejecutar diversas aplicaciones y resistir muchos errores generados por ellas sin colapsarse.

Los módulos de Windows 2000 Server son:

Módulo de Usuario: Es la parte que interactúa con el usuario y comprende un conjunto de subsistemas de entorno: *POSIX*, Win32, OS/2, entre otros. Los comandos que un usuario ejecuta dentro de una aplicación que requiere tener acceso al hardware se manejan a través de la capa de Servicios ejecutivos del núcleo. Y aquellos procesos que se ejecutan en este modo se les asigna a un bloque de memoria elegido por el sistema operativo.

Módulo de Núcleo: Es un conjunto de módulos que controlan al hardware y acceden a los datos del sistema. Este modo se ejecuta en un espacio protegido de la *CPU* al que no pueden acceder otros programas. Consta de las siguientes partes importantes: El Ejecutivo de Windows 2000 Server, Diversos módulos administrativos, Controladores de dispositivos, Núcleo y Capa de abstracción del hardware.

1.2 LINUX COMO SERVIDOR DE INTERNET

La palabra Linux está derivada de Linus Torvalds, nombre del inventor de este sistema operativo. Originalmente se refería solamente a su kernel o núcleo, aunque comúnmente se utiliza para referirse al sistema completo, incluyendo sus aplicaciones.

Linux es una reimplementación completamente gratuita de las especificaciones *POSIX*, con extensiones de *SYSV* y *BSD* (lo cual significa que parece *Unix* pero no proviene del mismo código fuente base), que está disponible tanto en su versión fuente como en la compilada.

Existen algunas distribuciones de Linux como: Debian, Red Hat, Suse, Slackware, etc. Normalmente éstas distribuciones traen las herramientas necesarias para la instalación del sistema. Básicamente, para la instalación se necesitan efectuar dos particiones en el disco duro, una para el sistema y otra más pequeña para utilizarla como memoria virtual (swap).

Como servidor de Internet, Linux puede utilizar Windows NT, Novell Netware y la mayoría de los sistemas de *Unix* en la misma dotación física. Linux ha sido elegido por miles y miles de *ISPs*, por muchos laboratorios de universidades y negocios.

1.2.1 REQUERIMIENTOS DEL SISTEMA

Los requerimientos mínimos para la instalación de Linux se muestran en la **Tabla 1.2**. [WWW-05]:

Tabla 1.2 Requerimientos de Hardware para Linux Red Hat

PROCESADOR	386-486 o CPU Pentium de mayor capacidad
MEMORIA	Desde 32 MB de RAM (mínimo recomendado), dependiendo de la distribución, versión del Sistema Operativo y procesador
DISCO DURO	Desde 540 MB, dependiendo de los paquetes a instalarse
UNIDAD DE DISCO	CD-ROM o diskette
TARJETAS	Tarjeta de interfaz de red Tarjeta de video VGA o superior

1.2.2 CARACTERÍSTICAS

Entre las principales características de Linux tenemos [WWW-06][WWW-07]:

- **Multitarea:** Linux utiliza la llamada multitarea preventiva, la cual asegura que todos los programas que se están utilizando en un momento dado serán ejecutados, siendo el sistema operativo el encargado de ceder tiempo de microprocesador a cada programa.
- **Multiusuario:** Linux debido a que es un sistema multitarea, permite que varios usuarios accedan a la computadora y ejecuten programas que compartirán la CPU. Además la ejecución de los procesos de cada usuario, su memoria, ficheros etc. estarán protegidos de modo que cada usuario pueda decidir quien accede a sus recursos.
- **Multiplataforma:** Las plataformas en las que en un principio se puede utilizar Linux son 386, 486. Pentium, Pentium Pro, Pentium II, AMD Kx, Amiga y Atari, también existen versiones para su utilización en otras plataformas, como Alpha, ARM, MIPS, PowerPC y SPARC.
- **Multiprocesador:** Soporte para sistemas con más de un procesador, está disponible para Intel y SPARC (hasta 16 CPUs).
- **Política de copia en escritura para la compartición de páginas entre ejecutables:** Esto significa que varios procesos pueden usar la misma zona de memoria para ejecutarse. Cuando alguno intenta escribir en esa memoria, la página se copia a otro lugar. Esta política de copia en escritura tiene dos beneficios: aumenta la velocidad y reduce el uso de memoria.
- **Sigue POSIX:** Desde sus inicios, Linus Torvalds toma los estándares POSIX para la escritura de su sistema operativo, de modo que Linux ha seguido y sigue unos estándares para su escritura y funcionamiento.
- **Sistemas de archivos:** Soporta varios formatos de sistemas de archivos como: ext2, FAT, OS/2, NFS, Coda, HFS, etc.
- **Seguridad:** Linux no sólo proporciona el sistema de protección entre procesos y entre ficheros, también provee métodos de protección de red, sistemas de encriptación de información entre otros. La arquitectura de seguridad de Linux consta de:
 - a) Cuentas de usuario
 - b) Control de acceso discrecional
 - c) Control de acceso a la red

- d) Cifrado
- e) Conexión
- f) Detección de intrusos

- **Estabilidad:** Linux es un sistema operativo en el que no se dan habitualmente cuelgues generales del sistema por causa del mismo. Aísla los procesos y su ejecución de manera que no alteren a la ejecución y funcionamiento del sistema operativo. Linux es indicado para ser usado como servidor porque es un sistema con escasas caídas.
- **Memoria virtual:** Usando paginación (sin intercambio de procesos completos) a disco: una partición o un archivo en el sistema de archivos, o ambos, con la posibilidad de añadir más áreas de intercambio sobre la marcha (se sigue denominando intercambio, pero es en realidad un intercambio de páginas). Un total de 16 zonas de intercambio de 128MB de tamaño máximo pueden ser usadas en un momento dado con un límite teórico de 2GB para intercambio.

La memoria se gestiona como un recurso unificado para los programas de usuario y para el caché de disco, de tal forma que toda la memoria libre puede ser usada para caché y éste puede a su vez ser reducido cuando se ejecuten grandes programas.

- **Consolas virtuales múltiples:** Varias sesiones de login a través de la consola entre las que se puede cambiar con las combinaciones adecuadas de teclas (totalmente independiente del hardware de video).
- **Librerías compartidas de carga dinámica** y también librerías estáticas.
- Linux se ejecuta completamente en **modo protegido** y las ayudas pueden tener tanto un tratamiento de 32 o bien de 64 bits.
- Todo el código fuente está disponible, incluyendo el núcleo completo y todos los controladores, las herramientas de desarrollo y los programas de usuario; además esto se puede distribuir libremente.
- Acceso transparente a particiones MS-DOS (o a particiones OS/2, FAT) mediante un sistema de archivos especial: no es necesario ningún comando especial para usar la partición MS-DOS, ésta parece un sistema de archivos normal de Unix (excepto por algunas restricciones en los nombres de archivo o permisos). Las particiones comprimidas de MS-DOS 6 no son accesibles en este momento.
- Soporte para **VFAT** (Windows NT, Windows 95 y Windows 98), y también para FAT32.
- Lan Manager / Windows Native (SMB), software cliente y servidor.

- Diversos protocolos de red incluidos en el kernel: **TCP**, **IPv4**, **IPv6**, **AX.25**, **X.25**, **IPX**, **DDP**, Netrom, etc. **AX.25** es el protocolo de red de las transmisiones de Radio Paquete, habitualmente conocido como Packet.

1.2.3 ARQUITECTURA

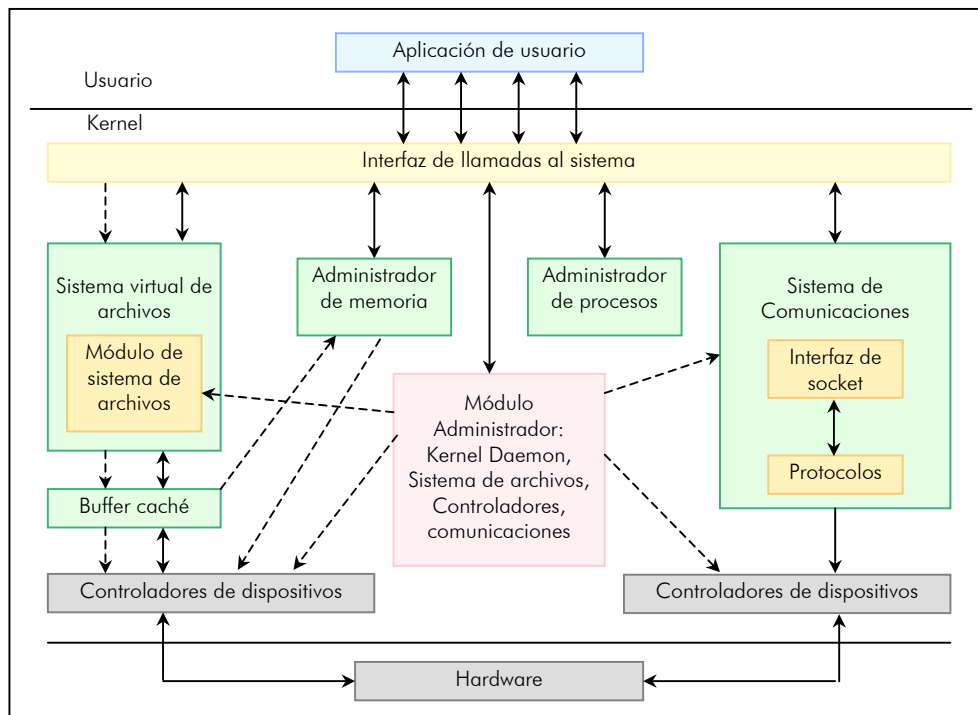


Fig. 1.2 Arquitectura de Linux

En la figura Fig.1.2 se puede apreciar la arquitectura micro-kernel de Linux que hace una división entre los servicios del sistema operativo dividiéndolos en dos niveles: uno implementado para el usuario y otro implementado para supervisar o administrar.

El administrador de procesos suministra soporte para la carga dinámica de módulos (un módulo es un fichero objeto, obtenido a partir de un fuente en "C" compilado pero no enlazado), cada módulo representa funcionalidad al kernel, que puede ser cargado después de que el kernel inicie su ejecución.

Esta funcionalidad se utiliza para:

- Implementar manejadores de dispositivos.

- La abstracción de interfaces que utilizan el sistema virtual de archivos y la interfaz de red.
- Para añadir una nueva funcionalidad.

1.3 WINDOWS Y LINUX COMO ALTERNATIVAS PROPIETARIA Y DE CÓDIGO ABIERTO

1.3.1 WINDOWS COMO ALTERNATIVA PROPIETARIA

SOFTWARE PROPIETARIO

El software propietario es software cerrado, donde el dueño del software controla su desarrollo y no divulga sus especificaciones.

El software propietario es el producido principalmente por las grandes empresas, tales como Microsoft y muchas otras. Antes de poder utilizar este tipo de software se debe pagar por él. Cuando se adquiere una licencia de uso de software propietario, normalmente se tiene derecho a utilizarlo en un solo computador y a realizar una copia de respaldo. En este caso la redistribución o copia para otros propósitos no es permitida.

En el software propietario, los propietarios o autores ponen a sus programas bajo un Copyright, con el objetivo de que éstos sean capaces legalmente de copiar y físicamente de modificar el código fuente para de esta manera ser los únicos que se beneficien económicamente del mismo. [WWW-08]

Microsoft concede derechos específicos de uso con respecto a cada uno de los productos para los que adquiera una licencia, siempre que cumpla todos los términos y condiciones que sean de aplicación para dicho producto. Dichos productos están protegidos por leyes y por tratados internacionales en materia de derechos de autor, así como por otras leyes y tratados sobre propiedad intelectual. El software se otorga bajo licencia y no es objeto de venta.

Estos derechos de uso de producto también rigen a cualquier servicio de soporte del producto relacionados con el software y los servicios en línea, salvo que se indique lo contrario en otro contrato con Microsoft.

Microsoft tiene, según el tipo de software, ciertas especificaciones para su derecho de uso y licenciamiento, en este caso, algunas especificaciones como Servidor para Windows 2000 Server son:

DERECHOS DE USO DEL SERVIDOR

Instalación: Software de Servidor

Por cada licencia adquirida, se puede instalar una copia del Software de Servidor en un único servidor. No puede separar los componentes del Software de Servidor para su uso en más de un Servidor.

Software de Cliente

Se puede instalar el Software de cliente en cualquier equipo.

Límite de procesadores

Se puede utilizar el Software de Servidor de Microsoft Windows 2000 Server con hasta cuatro CPUs del servidor en un momento dado. El Software de Servidor para Microsoft Windows 2000 Advanced Server se puede utilizar con un máximo de ocho CPUs del servidor al mismo tiempo.

Necesidad de Licencias CAL para Windows 2000 Server

Se debe adquirir una Licencia **CAL** independiente por cada equipo que sea utilizado por un "Usuario autenticado" o que use los "Servicios de Servidor de Windows 2000 Server" independientemente del software que se utilice.

Modo de Implantación

Se puede utilizar Licencias **CAL** con el Software de Servidor tanto en modo "Por Puesto" como en modo "Por Servidor". Cuando el producto sea licenciado dentro de la suite BSB o se esté utilizando Servicios de Terminal, se deberá utilizar Licencias **CAL** con el Software de Servidor en modo "Por Puesto".

Servicios de Terminal

Además de la Licencia **CAL** para Windows 2000 Server, es obligatorio adquirir una Licencia **CAL** para Servicios de Terminal para cada equipo que utilice los "Servicios de Terminal". Sin embargo, no se necesitará una Licencia **CAL** para Servicios de Terminal si el equipo utiliza una copia licenciada de Windows 2000 Professional o de Windows XP Professional.

Servicios de Windows 2000 Server

Incluye Servicios de Archivo, Servicios de Impresión, Servicio de Acceso Remoto y Servicios de Terminal.

Servicios de Terminal

Consisten en:

- Utilizar la prestación de servicios de terminal que incorpora el Software de Servidor para que los equipos puedan utilizar el software residente en el servidor.
- Utilizar otro software conjuntamente con el Software de Servidor a fin de obtener servicios similares.

Modo de Implantación

Cuando utilice los Servicios de Terminal, no podrá utilizar el Producto en el modo "Por Servidor".

Funciones de Actualización

Si se utiliza las características de actualización dentro del producto, es necesario utilizar cierta información del sistema, hardware y software para implementarlas. Al utilizar estas características, se autoriza explícitamente a Microsoft o a su agente designado a tener acceso y a utilizar la información necesaria para la actualización.

Componentes de Servicios Basados en Internet

El producto contiene componentes que permiten y facilitan el uso de ciertos servicios basados en Internet.

Microsoft puede comprobar automáticamente la versión del Producto y/o los componentes que está utilizando, y puede proporcionar actualizaciones o revisiones al Producto, las que se descargarán automáticamente en un equipo.[WWW-09]

1.3.2 LINUX COMO ALTERNATIVA DE CÓDIGO ABIERTO

EL CÓDIGO ABIERTO

Podría traducirse como Código Abierto (**Open Source**) un programa que ofrece al usuario la posibilidad de entrar en su código para poder estudiarlo o modificarlo. Pero no sólo hace referencia al libre acceso del código fuente. La intención de la definición de open source es establecer que esos criterios contengan la esencia de lo que los programadores quieren que signifique: que aseguren que los programas distribuidos con licencia open source estarán disponibles para su continua revisión y mejora para que alcancen niveles de fiabilidad. [WWW-08]

CONDICIONES DE DISTRIBUCIÓN

Las condiciones de distribución de un programa open source deben cumplir una serie de criterios [WWW-10]:

- 1. Redistribución Libre:** La licencia no deberá impedir la venta o el ofrecimiento del software como un componente de una distribución de software agregado conteniendo programas de muchas fuentes distintas. La licencia no deberá requerir el pago de los derechos de autor u otra tasa por dicha venta.
- 2. Código Fuente:** El programa tiene que incluir el código fuente, y tiene que permitir la distribución tanto en código fuente como en forma compilada. Si alguna forma de un producto no es distribuida con el código fuente, tiene que haber un medio publicado para obtener el código fuente por no más que un costo razonable de reproducción, preferentemente una descarga a través de Internet sin costo.
- 3. Trabajos Derivados:** La licencia tiene que permitir modificaciones y trabajos derivados, y ser distribuidos bajo los mismos términos de la licencia del software original.
- 4. Integridad del Código Fuente del Autor:** La licencia tiene que permitir explícitamente la distribución del software construido a partir del código fuente modificado. La licencia puede requerir que los trabajos derivados tengan un nombre distinto o un número de versión distinto al del software original.
- 5. Discriminación de Personas o Grupos:** La licencia no tiene que discriminar a ninguna persona o grupo de personas.
- 6. Discriminación de Campos Laborales:** La licencia no tiene que restringir a nadie que

haga uso del programa en un campo laboral específico.

7. Distribución de la Licencia: Los derechos adjuntos al programa tienen que aplicarse a todos aquellos que reciben el programa sin la necesidad de ejecutar una licencia adicional.

8. La Licencia no tiene que ser Específica de un Producto: Los derechos adjuntos al programa no tienen que depender de que el programa forme parte de una distribución particular de software. Si el programa es extraído de esa distribución y es usado o distribuido de acuerdo a los términos de la licencia del programa, todas las partes a las que el programa sea redistribuido deben tener los mismos derechos que son garantizados en conjunto con la distribución original del software.

9. La Licencia no tiene que restringir a otro Software: La licencia no tiene que colocar restricciones en otro software que es distribuido junto con el software licenciado.

Nota: El modelo Open Source se basa en que los propietarios de los derechos de autor compartan su trabajo para una mayor y mejor colaboración. Los desarrolladores permiten que sus programas sean distribuidos y modificados libremente, pidiendo solamente a cambio el tener los mismos privilegios.

La Licencia de Software Libre es la *GNU*, o *GNU GPL*, licencia que equivale al reparto equitativo.

GNU GPL fue creada para mantener y defender las libertades que definen el software libre; establece como derechos intransferibles de los programas publicados bajo la *GPL*. La *GPL* garantiza tener la libertad para estudiar, cambiar, y redistribuir el programa, diciendo que nadie está autorizado para quitarle estas libertades redistribuyendo el programa bajo una licencia restrictiva.

LO QUE OFRECE EL CÓDIGO ABIERTO

- **Adaptabilidad:** El software libre es más adaptable ya que no hay ningún obstáculo ni técnico ni legal para su modificación.
- **Economía:** La mayoría de los sistemas operativos abiertos, como Linux, FreeBSD y la mayoría de los programas que funcionan en esos sistemas, como todos los generados bajo la licencia *GNU*, son gratuitos o el costo es reducido.
- **Acceso Universal:** Cuando un sistema es gratuito o el costo disminuye, implica que está

al alcance de muchas más personas.

- **Fiabilidad:** En el software de código abierto, se puede saber cómo funciona el programa y qué tipo de operaciones realiza en el equipo, de forma que los errores se pueden detectar y modificar.
- **Seguridad:** Con el software de código abierto, se puede saber qué hace el programa, y por tanto qué agujeros de seguridad puede presentar.

1.4 COMPARACIÓN DE WINDOWS Y LINUX

En una era de cambios tecnológicos y de una amplia oferta en sistemas operativos es necesario estudiar dos de las alternativas existentes más utilizadas como plataformas para Internet: Windows como alternativa propietaria y Linux como alternativa de código abierto; para determinar su comportamiento y funcionalidad en la gran red, Internet.

LINUX(Red Hat 7.2) vs. WINDOWS 2000 SERVER

En la **Tabla 1.3** se muestra una pequeña comparación entre los dos sistemas operativos, en base a los siguientes parámetros:

Tabla 1.3 Comparación de Windows 2000 Server y Linux Red Hat 7.2

PARÁMETROS	SISTEMA OPERATIVO	
	LINUX Red Hat 7.2	WINDOWS 2000 SERVER
SISTEMA OPERATIVO	Gratis \$49.95 US (CD-ROM)	5-Usuarios \$809 10-Usuarios \$1129 Enterprise Ed. 25-Usuarios \$3999
SOPORTE TÉCNICO EN LÍNEA	Si	Si
FUENTES DEL KERNEL	Si	Sólo disponible para algunos gobiernos
SERVIDOR WEB	Apache Web Server	Internet Information Server
SERVIDOR FTP	Si	Si
SERVIDOR TELNET	Si	Si
SERVIDOR SMTP/POP3	Si	Si(SMTP soportado en Internet Information Server)
DNS	Si	Si
SISTEMA DE ARCHIVOS DE RED	NFS y SMB	SMB
SERVIDOR DE VENTANAS X (CORRER APLICACIONES REMOTAS BASADAS EN GUI)	Si	Si
HERRAMIENTAS DE ADMINISTRACIÓN REMOTAS	Si	Si
SERVIDOR DE NEWS	Si	Si(incluido en el Option Pack no en el Sistema Operativo)
SISTEMAS DE ARCHIVO POSIBLES	32	3
CUOTAS DE DISCO	Si	Si
NÚMERO DE GUIs A ESCOGER	4	1

Una comparación más amplia se encuentra en el **Anexo 4, Tabla 1**, en el CD adjunto a este

documento en: CD_TESIS:\Tesis\Anexos_Tesis\Anexos.pdf.

Del estudio de este capítulo se podría decir que el sistema operativo Linux sería la elección para sitios con presupuesto limitado o en corporaciones que demanden servidores de varios procesadores y que requieran de sistemas operativos escalables.

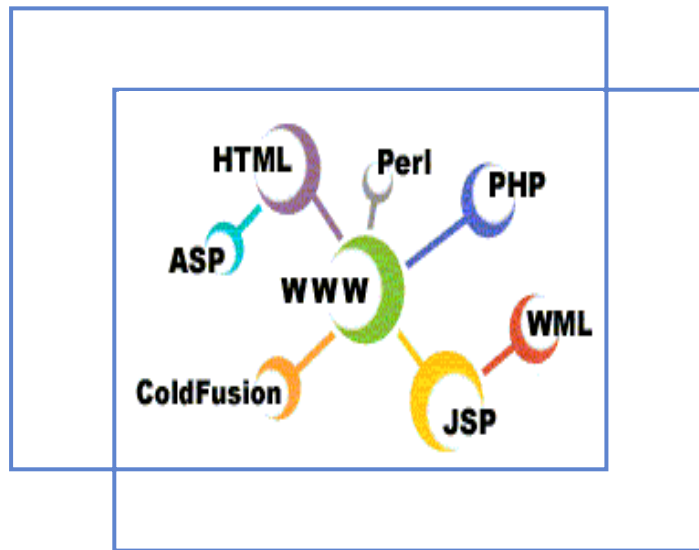
En resumen:

Para la utilización de Windows se debe contar con una licencia, la misma que autoriza su legal y libre utilización dentro de ciertas especificaciones. A pesar de esto, Windows se encuentra instalado en la gran mayoría de las máquinas, lo que ha permitido que tanto el sistema operativo como otros productos de Microsoft, permitan que el tamaño de la curva de aprendizaje de la gente que ya lo utiliza, disminuya, al no tener que aprender el uso de un sistema operativo y herramientas nuevas.

Por otro lado Linux se obtiene mediante una serie de distribuciones como Red Hat, Slackware, Debian, Suse, etc., las que se diferencian por su método de instalación y por los paquetes que vienen incluidos. Todo el software de Linux está regido por la licencia de GNU, con la que cualquier persona o empresa capacitada puede modificar un programa y venderlo según desee, con la condición de que la persona o empresa que compra ese producto puede realizar la misma acción o simplemente hacer copias para todos aquellos que lo quieran. Esta licencia es la garantía que afirma la absoluta libertad de este sistema operativo. Linux ofrece software de código abierto, independencia de vendedores de software y la posibilidad de participar en una gran comunidad de desarrollo. Las desventajas de la adopción de Linux tiene que ver con la inestabilidad de los vendedores, la problemática de encontrar personal para soporte técnico y la falta de adaptación a todo el hardware existente.

En fin, tanto un sistema operativo como el otro tiene ventajas y desventajas, pero la última palabra tienen aquellos quienes deciden en qué y por qué plataforma optan, dadas las necesidades y requerimientos tanto de la empresa como de las aplicaciones que se deseen desarrollar o ejecutar. [WWW-11]

Capítulo 2



HERRAMIENTAS O TECNOLOGÍAS DE GENERACIÓN DINÁMICA

- ❖ Introducción
- ❖ Internet como medio de comunicación e intercambio de información
- ❖ Evolución del contenido estático a dinámico
 - ❖ Interactividad y Generación de Contenido Dinámico en Internet
 - ❖ Generación de Páginas Dinámicas o Contenido Dinámico
 - ❖ Aplicaciones Web
 - ❖ Areas de aplicación del contenido dinámico
 - ❖ Herramientas para la generación de contenido dinámico
 - ❖ Herramientas Dinámicas del lado del cliente
 - ❖ Herramientas Dinámicas del lado del servidor
 - ❖ Comparación de las herramientas: PHP, ASP y JSP

INTRODUCCIÓN

A través de los años, Internet ha evolucionado aceleradamente. Al principio era una gran biblioteca virtual, donde se podía encontrar todo tipo de información publicada en páginas web. Estas páginas web eran sencillas y estáticas, actualmente son páginas totalmente dinámicas, es decir le permiten al usuario interactuar con ellas, por ejemplo, se pueden hacer reservaciones de vuelos, pagos de servicios, compras, etc.

Para hacer posible esta evolución han surgido todo tipo de herramientas o tecnologías que permiten desarrollar páginas web dinámicas. Desde aquellas que permiten realizar sencillas animaciones, grandes cálculos, hasta las más usadas actualmente: las que permiten recuperar información de una base de datos.

Si se desea que un sitio web o portal permanezca y prevalezca en Internet, debe contar con páginas que permitan un alto grado de interactividad con el usuario visitante, por esto es importante conocer las herramientas que existen para lograrlo.

En este capítulo se hace un estudio de cómo han evolucionado las herramientas de generación de contenido dinámico, para comprender su clasificación, ventajas, desventajas, funcionamiento y de esta forma llegar a determinar en qué casos utilizar cada herramienta.

Cabe destacar que este capítulo se centrará en las herramientas: PHP, ASP y JSP, de las que se hará un análisis comparativo para entender el comportamiento de una herramienta gratuita(código abierto), una comercial y una compartida.

2.1 INTERNET COMO MEDIO DE COMUNICACIÓN E INTERCAMBIO DE INFORMACIÓN

El origen de la “gran red de redes” data a partir de 1960, cuando el Departamento de Defensa de los Estados Unidos decidió formar una red de computadoras que funcionara bajo condiciones extremas(incluso bajo un ataque nuclear), ya que se necesitaba establecer un medio de comunicación para transmitir información confidencial.

Para este proyecto, el gobierno de los Estados Unidos invitó a participar a las universidades con mayor desarrollo tecnológico en el área. En 1969 se unieron al proyecto varias universidades, formando así una red llamada **ARPANET**.

Al analizar los buenos resultados del proyecto, se unieron otras universidades y **ARPANET** se separó en dos redes, una red militar llamada **MILnet** y otra académica llamada **NSFnet** que evolucionó hasta lo que hoy conocemos como Internet.

Inicialmente, Internet se utilizó para el intercambio de información científica y académica entre profesores y alumnos de programas avanzados de las universidades por medio del correo electrónico y algunos programas de clasificación y búsqueda de información.

A partir de 1993, Internet se convirtió en el sistema más innovador de publicación e intercambio de información, gracias al sistema **WWW** y al concepto de páginas web, ambos basados en la técnica del hipertexto(texto, imágenes, sonido, video, animaciones e hipervínculos) y creados por el científico Tim Berners Lee en 1989.

En 1994, las empresas comerciales, industriales y todo tipo de organizaciones se interesan por el gran potencial de Internet y ejercen gran presión para utilizar la red. El principal objetivo, publicar información para darse a conocer.

Hasta este punto, Internet era un contenedor de millones de páginas web sencillas y estáticas, es decir páginas con información que no cambia a menos que se modifique el código **HTML** y el hipertexto en sí. [REV-01]

Con el paso del tiempo y la necesidad de hacer negocios y brindar servicios a través de Internet, el enfoque de las páginas web cambió. Era fundamental que las páginas cumplieran con dos requisitos:

- Capacidad de cambiar automáticamente su contenido.
- Interactuar con el usuario para capturar y procesar información proveída por el mismo.

Para conseguirlo, se cambió el concepto de páginas web estáticas a **páginas web dinámicas**¹. Las páginas web dinámicas son capaces de generar su propio contenido y de procesar información proveída por el usuario, debido a que además de estar formadas por código *HTML/DHTML/XML/WML* e hipertexto, están formadas también por porciones de código denominados **scripts**, que son los que realmente dotan de dinamismo a las páginas.

Debido a que el *HTML* es un lenguaje enfocado a la presentación de páginas web, se necesitaba un lenguaje especial para permitir la programación de esos scripts. A partir de esta necesidad surgió todo tipo de lenguajes que hoy en día forman parte de una completa tecnología de desarrollo de páginas web dinámicas.

Gracias al dinamismo de las páginas web y a las tecnologías para desarrollarlas, Internet, en un futuro muy cercano, se habrá confundido con la radio, la televisión, la telefonía, los sistemas de transacciones financieras y comerciales, convirtiéndose en un solo sistema de comunicación global sin barreras de tiempo y espacio.

LAS TECNOLOGÍAS WEB

Las Tecnologías Web son aquellas tecnologías que permiten el desarrollo y presentación de páginas web dinámicas. Una tecnología web básicamente está compuesta por tres herramientas: un Servidor Web, una Base de Datos y el lenguaje de programación. Estos lenguajes son más conocidos como "**herramientas de generación de contenido dinámico**" y permiten insertar scripts en una página web para hacerla dinámica.

Los scripts pueden ser pequeños o grandes dependiendo de lo que se necesite realizar, por ejemplo, si se necesita que una página web cambie su fecha diariamente, bastará con insertar un script con unas cuantas líneas de código, pero si se desea procesar información almacenada en una base de datos, se necesitará de una completa aplicación. En Internet las aplicaciones se basan en el modelo Cliente/Servidor tradicional y se denominan

¹ Páginas web que se generan al instante de la petición con datos proveídos por el usuario.

aplicaciones web².

Entonces, al hablar de tecnologías web se habla principalmente de tecnologías para el desarrollo de aplicaciones web y por lo tanto de páginas y sitios web dinámicos. Estas tecnologías se encuentran en constante desarrollo, principalmente las herramientas de generación de contenido dinámico, que han evolucionado conjuntamente con la forma de generar y presentar la información en Internet. [WWW-12]

2.2 EVOLUCIÓN DEL CONTENIDO ESTÁTICO A DINÁMICO

Como es evidente, Internet ha evolucionando a gran velocidad. El sector de diseño de sitios y páginas web está teniendo más y más desarrollo en la parte tecnológica. Los sitios web y **portales**³ que son los que contienen las páginas web dinámicas, se están tornando cada día más interactivos, estéticos y sofisticados. Hoy en día se puede distinguir claramente la existencia de cuatro generaciones de sitios web:

- **Sitios web de primera generación:** Las páginas se alojaban en el servidor y éste se encargaba de enviarlas al navegador, es un modelo basado en páginas estáticas. Si bien no eran visualmente atractivos, estos sitios estaban enfocados a funcionar en forma veloz y entregar al usuario gran cantidad de información relacionada.
- **Sitios web de segunda generación:** Los sitios se volvían más comerciales, y el hecho de “capturar” usuarios y ofrecer servicios(chats, correo, etc.) se tornó una prioridad, dando gran importancia al aspecto visual. En esta generación aparecieron las primeras herramientas para programar los servicios ofrecidos y para dominar la forma de presentar la información.
- **Sitios web de tercera generación:** Esta generación sigue basada en lo visual, el gran cambio se da en la forma en que se genera la información. Las páginas estáticas son reemplazadas por páginas dinámicas generadas en tiempo real a gran velocidad. Se desarrollan las aplicaciones web usando herramientas de programación más eficientes que las de la segunda generación.
- **Sitios web de cuarta generación:** Surgen como una evolución mejorada de los sitios

² Son aplicaciones Cliente/Servidor que trabajan en Internet y son utilizadas por las páginas web dinámicas.

³ Son portales los sitios web que ofrecen personalización, es decir configuración de los contenidos para cada usuario.

web de tercera generación y se denominan portales. Los portales le permiten al usuario encontrar una gran cantidad de servicios (información, chats, foros, motores de búsqueda, correo, etc.) en un solo lugar, también le permiten personalizar el contenido del portal según sus preferencias. Al igual que los sitios web de tercera generación, los portales utilizan potentes aplicaciones web para programar los servicios que ofrecen.

[WWW-13]

Los sitios web que hoy en día se destacan son los de tercera y cuarta generación, por esta razón, actualmente ningún web-master consideraría diseñar sitios o portales y aplicaciones web usando sólo un editor de archivos *HTML*. Un sitio o portal moderno, actualizado y atractivo tiene que ser dinámico e interactivo si quiere prevalecer en la red. De igual manera, las aplicaciones web deben tener las mismas características.

Un web-master moderno y capacitado debe estar informado sobre las herramientas más utilizadas para el diseño de sitios o portales y aplicaciones web. Debe conocer el uso, áreas de aplicación, alcance y evolución de las herramientas existentes, empezando desde lo más básico hasta lo más avanzado en el sector del diseño y la programación web.

Para ello es importante, primeramente, conocer como ha evolucionado el contenido de estático a dinámico, para luego conocer la evolución de las herramientas.

Nota: En la sección 2.2.2 Herramientas para la generación de contenido dinámico, se puede comprender la evolución de dichas herramientas por medio de la clasificación de las mismas.

EVOLUCIÓN DEL CONTENIDO ESTÁTICO: DE HTML A DHTML

HTML

El lenguaje *HTML*, aunque es llamado “lenguaje”, no se considera propiamente como lenguaje de programación. Ya que su única finalidad es dar formato a la información desplegada en una página web.

Las páginas *HTML* son almacenadas en forma de “texto puro”, es decir se pueden abrir con cualquier editor de textos. El lenguaje *HTML* está conformado por **TAGS** (etiquetas) que

están comprendidas entre los signos "<" y ">". Ver **Fig. 2.1**.

A partir de este lenguaje se tiene la capacidad de introducir enlaces o hipervínculos, seleccionar el tamaño de letra, intercalar imágenes, insertar audio y video, marquesinas, etc.; todo esto de una manera prefijada y en ningún caso automática e inteligente. En efecto, el *HTML* no permite realizar un simple cálculo matemático u obtener datos de una base de datos.

<html>	//Inicio de página HTML
<head> <title>Página en HTML</title></head>	//Cabecera y título de la página
<body bgcolor="#FFFF33" background="fondo.gif">	//Cuerpo de la página
<div align="center">	//Alineación
<p></p>	//Insertar imagen
</div>	
<h1>Texto HTML</h1>	//Formatear texto
</body>	//Fin de cuerpo de la página
</html>	//Fin de página HTML

Fig. 2.1 Ejemplo de Página HTML

En realidad *HTML*, aunque es muy útil a pequeña escala, resulta bastante limitado a la hora de construir grandes sitios o portales en Internet.

Esta deficiencia del *HTML*, es la razón por la que se ha hecho necesario el empleo de otros lenguajes, herramientas o tecnologías mucho más versátiles y que implican un aprendizaje relativamente más complicado, pero a la vez son capaces de responder de manera inteligente a las demandas del navegador y permiten la automatización de determinadas tareas rutinarias tales como: ingreso, actualización, eliminación de información, tratamiento de pedidos de una tienda virtual, procesamiento de una transacción bancaria, etc.

HTML trabaja de una forma simple, atendiendo la petición del visitante. Las páginas *HTML* se encuentran almacenadas en un servidor web, cuando un usuario solicita determinada página, el servidor la busca entre sus archivos y le envía como respuesta al navegador del visitante. Dicha página web solicitada, es almacenada con la extensión: **.htm** o **.html** que el navegador o explorador interpreta(en otras palabras traduce) y es mostrada en forma de hipertexto. Pero en realidad, no existe ningún proceso más que atender la petición y enviar el documento tal y como lo formatean las etiquetas o *TAGS*. Ver **Fig. 2.2**.

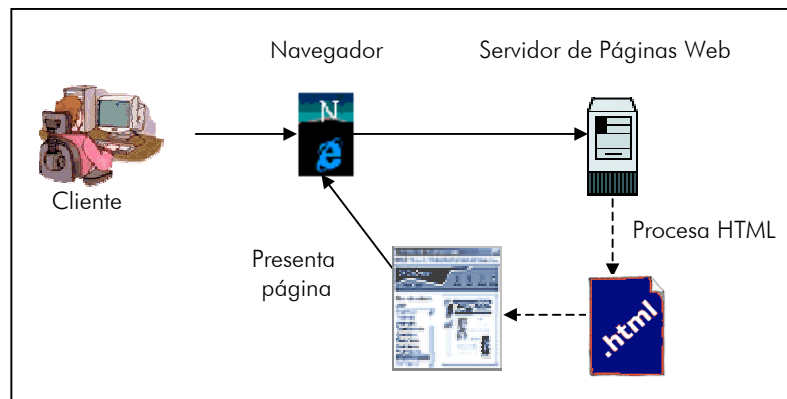


Fig. 2.2 Funcionamiento de HTML

El *HTML* tiene sus propias normas que han sido impuestas por una asociación internacional denominada **W3C**. Hay diferentes versiones de *HTML*, la actual es 4.0, dichas versiones sólo son soportadas por navegadores tales como Internet Explorer 4.0 en adelante y Netscape Navigator 4.0 en adelante.

Cabe mencionar además que las versiones nuevas de *HTML* sólo agregan funciones y *TAGS* nuevas a la base ya establecida por las versiones anteriores, por lo cual a veces se puede ver una página web hecha en *HTML* 4.0 con un navegador antiguo que no soporta las *TAGS* de la nueva versión, pero en este caso el navegador lo único que hace es ignorar las *TAGS* nuevas. [WWW-14]

DHTML

El *DHTML* es en realidad la versión 4.0 de *HTML* que incorpora nuevas herramientas o lenguajes como: *JavaScript* y *CSS*, de los que se hablará más adelante.

En *DHTML*, el *HTML* original se usa para la presentación de información, pero como las presentaciones tienen limitación en la flexibilidad del diseño visual, se utilizan las facilidades de *CSS* haciendo más ágil el manejo del diseño y formateo visual. En cambio *JavaScript* introduce el dinamismo y un alto grado de interacción entre la página y el usuario.

DHTML es el lenguaje que introdujo el término "sitios multimedia" o "sitios hipermedia" en Internet. Pero uno de sus más grandes inconvenientes es su compatibilidad. Sólo versiones más nuevas de los navegadores pueden ejecutar páginas *DHTML* sin problemas.

Nota: Sobre DHTML, ver detallado en la sección 2.2.2.1 Herramientas dinámicas del lado del cliente.

2.2.1 INTERACTIVIDAD Y GENERACIÓN DE CONTENIDO DINÁMICO EN INTERNET

Uno de los ejes fundamentales que diferencian a Internet de otros medios de comunicación es la interacción con el usuario y personalización de la información de acuerdo al mismo. La interactividad, que proporcionan las páginas web generadas dinámicamente, permite desarrollar contenido personalizado dependiendo de, por ejemplo, el perfil del visitante o los datos que éste completó en un formulario.

2.2.1.1 GENERACIÓN DE PÁGINAS WEB DINÁMICAS O CONTENIDO DINÁMICO

La generación de páginas web dinámicas o generación de contenido dinámico se refiere a que la información proveída por el usuario es procesada por medio del cliente o del servidor y presentada al cliente de forma estructurada y organizada, basándose en criterios. La interactividad que se da a través del usuario parte desde la introducción de información, la solicitud de un proceso y/o acción y el requerimiento de obtener los resultados de dicho proceso. La información puede ser proveída por el cliente o el servidor y los resultados son siempre generados hacia el cliente.

Para llevar a cabo la recopilación y procesamiento de información, así como la generación de resultados, se utilizan páginas web dinámicas personalizables y capaces de manejar enormes cantidades de contenidos propios.

DEFINICIÓN DE PÁGINAS WEB DINÁMICAS

Las páginas web dinámicas son aquellas que se generan al instante de la petición del usuario. Su contenido cambia automáticamente. Las principales características de las páginas web dinámicas son:

- Pueden ser procesadas en el cliente(navegador) o en el servidor. Cabe destacar que el

mayor dinamismo se encuentra en el servidor.

- Se presentan de forma distinta para cada usuario, son personalizables.
- Además de lenguajes de marcado como: *HTML*, *DHTML*, *XML*, *WML* e hipertexto, pueden contener pequeños scripts o completas aplicaciones web que les permiten realizar desde las operaciones más sencillas hasta las más complejas. Pueden desplegar la hora actual, la fecha diariamente, realizar grandes cálculos o procesar datos de extensas transacciones bancarias, etc. Pueden también estar compuestas únicamente de scripts y ser utilizadas por otras páginas.
- Pueden acceder a bases de datos para almacenar y recuperar gran cantidad de información.
- Son capaces de generar su propio contenido, así como también de generar contenido para otras páginas web.
- Proporcionan un alto grado de interacción con el usuario.
- En la parte visual están formadas principalmente por formularios, cuadros de diálogo, o cualquier medio que sea fuente para capturar datos.
- Los resultados de la ejecución de estas páginas se presentan como páginas estáticas o semi-dinámicas(pueden volver a pedir datos o contener scripts, controles **ActiveX**, **Applets**, etc.).
- Estas páginas web se encuentran en sitios web o portales que no son estáticos, es decir que cambian constantemente. Por ejemplo, en un sitio web que procesa una compra – venta de libros. En este caso un grupo de páginas web dinámicas serán las que se encargarán de procesar los datos ingresados por el usuario, los scripts o aplicaciones web contenidos en ellas o asociados a las mismas realizarán los cálculos y presentarán los resultados de la transacción en una página estática o semi-dinámica. [WWW-15]

En conclusión, los lenguajes o herramientas de generación de contenido dinámico son capaces de recrear a partir de ciertos scripts un sinnúmero de páginas web automatizadas, convirtiéndose en los componentes principales del significado de las páginas web dinámicas y de las aplicaciones web.

2.2.1.2 APLICACIONES WEB

Las aplicaciones web se basan en el modelo Cliente/Servidor, trabajan con un servidor web, gestionan información almacenada en una base de datos y utilizan como interfaz páginas web estáticas o dinámicas. Ver **Fig. 2.3**. Si no existiese tratamiento de los datos en el lado del servidor, difícilmente se hablaría de una aplicación web.

Las aplicaciones web suponen un importante cambio de enfoque con respecto al desarrollo de aplicaciones tradicionales. El usuario se comunica con la aplicación web desde cualquier cliente conectado a la red, en la mayoría de casos, sin importar la plataforma.

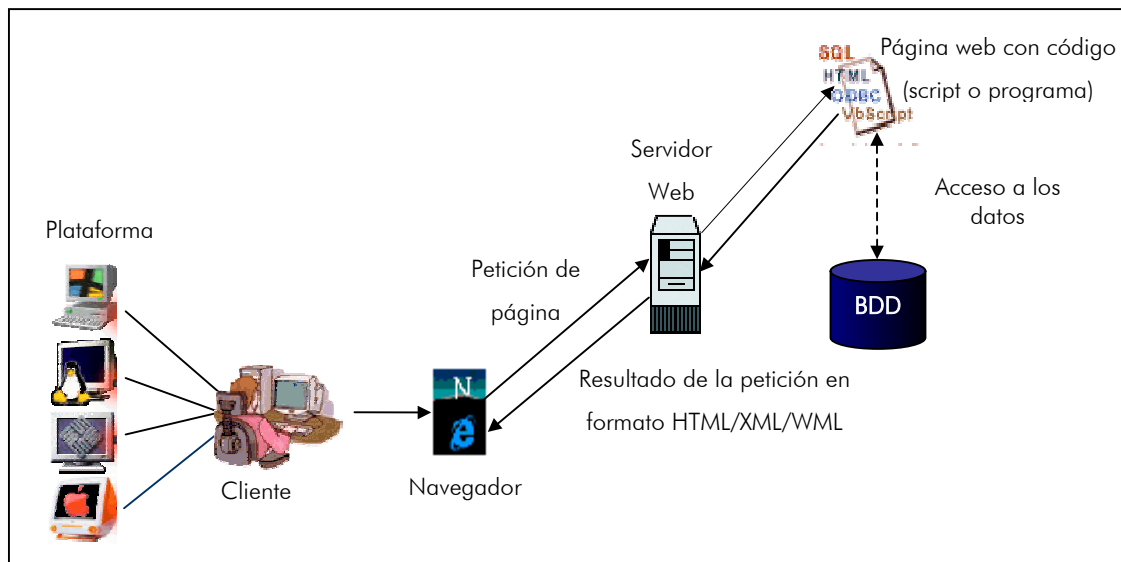


Fig. 2.3 Componentes de una Aplicación Web

Una aplicación web es más que un conjunto de páginas web entrelazadas entre sí, supone la existencia de código que se encarga de realizar algún proceso. Por eso, la principal cuestión que se tuvo que resolver para la construcción de aplicaciones web, fue el hecho de que el lenguaje *HTML* no tiene capacidad de programación: no cuenta con sentencias condicionales, ni bucles, no puede hacer cálculos y peor aún acceder a bases de datos, ya que fue pensado sólo como un formato de transmisión y presentación de documentos.

[WWW-16]

Nota: En el capítulo 5, se hace un estudio más profundizado de las aplicaciones web, se detalla sus características, funcionamiento, ventajas y desventajas.

2.2.1.3 ÁREAS DE APLICACIÓN DEL CONTENIDO DINÁMICO

Como conclusión de lo visto anteriormente, se puede decir que el contenido dinámico o las páginas web dinámicas son el sinónimo de las Aplicaciones Web; entonces al hablar de áreas de aplicación del contenido dinámico, en realidad se refiere a las áreas de aplicación de las aplicaciones web.

En Internet, podemos encontrar aplicaciones web de todo tipo, empezando desde las más simples como: el despliegue de la fecha y hora actual, el tipo de cambio, el estado del tiempo, el marcador de un partido de fútbol al instante que se está jugando, entre otras. Esta información aunque sencilla que parezca ya se encuentra automatizada y es generada por medio de pequeñas aplicaciones web.

Al ir navegando por Internet, además de estas simples aplicaciones podemos encontrarnos con avanzadas y sofisticadas páginas web, que hacen uso de grandes aplicaciones web, ejemplos de éstas pueden ser: los buscadores siendo los más comunes, las salas de chat, el servicio de correo, noticias, agendas, sitios de subastas, consultas del estado de una cuenta bancaria, pago de servicios, reservaciones de vuelos, por nombrar algunas.

También están las aplicaciones web que proporcionan una gran interacción con el usuario y quizá las más utilizadas hoy en día tanto por las grandes empresas y los usuarios: las aplicaciones web que se encuentran en los portales dedicados al comercio electrónico o también conocidos como tiendas virtuales, donde el objetivo común es la compra – venta de bienes y servicios, y el pago se hace a través de la transferencia de fondos por Internet con las tarjetas de crédito.

Con el simple acceso a Internet, tenemos un mundo lleno de servicios a nuestro alcance, como consecuencia nuestra vida se ha facilitado y todo gracias a las aplicaciones web que se encuentran presentes en todas las áreas, formando parte de todo tipo de sitios o portales, desde los más simples, pasando por los de entretenimiento, educación, medicina, hasta los más sofisticados como son los de comercio electrónico. Ver **Fig. 2.4**.

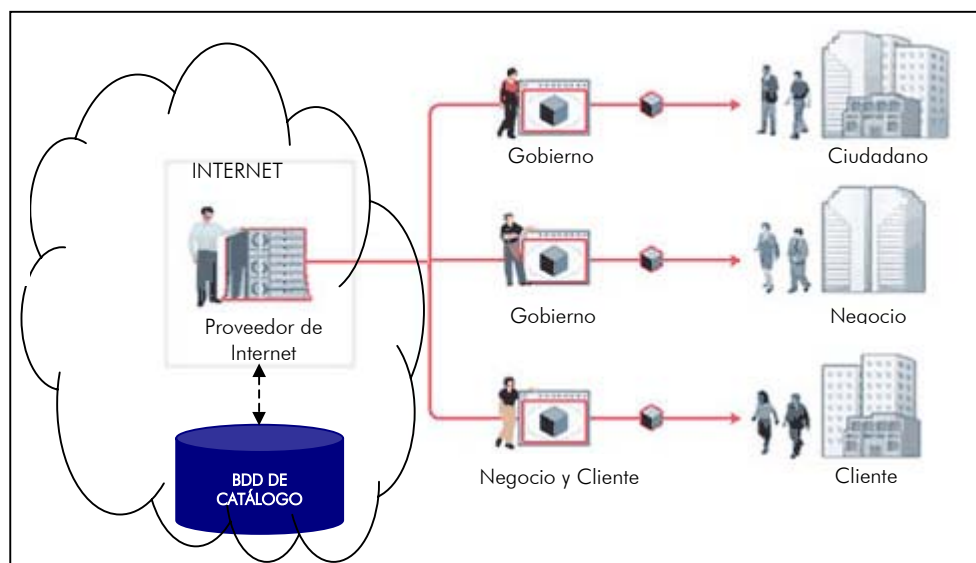


Fig. 2.4 Aplicación Web de comercio electrónico

2.2.2 HERRAMIENTAS PARA LA GENERACIÓN DE CONTENIDO DINÁMICO

Como se ha manifestado a través del tema, actualmente existe gran variedad de lenguajes o herramientas que se pueden utilizar para la generación de contenido dinámico y desarrollo de aplicaciones web. Por cuestión de términos, de aquí en adelante se hablará de herramientas, cuando se refiera al lenguaje de generación de: contenido dinámico o páginas dinámicas; y se hablará de tecnología web cuando se refiera a la herramienta en conjunto con su respectivo servidor web y base de datos que son los componentes complementarios y fundamentales de una aplicación web y de los que se hablará más adelante.

Nota: Del Servidor Web y Base de Datos se hablará en los capítulos 3 y 4 respectivamente.

Una de las principales diferencias de las herramientas de generación de contenido dinámico y la razón de su clasificación viene en función de donde se ejecutan; en el servidor, en el cliente o en ambos:

- Las herramientas que trabajan del lado del cliente se pueden clasificar también como las que trabajan del lado del navegador del usuario, y son conocidas como **CSS**(Client Side Scripts).

- Las herramientas que trabajan en el servidor se pueden clasificar como las que se procesan en el lugar donde está alojada la página web, y son conocidas como **SSS**(Server Side Scripts).
- También existen herramientas que pueden trabajar tanto en el cliente como en el servidor, aunque no cuentan con las mismas características en ambos lados.

Cada una de las herramientas, ya sean las que trabajan del lado del cliente o del lado del servidor, poseen ventajas y desventajas. Y a la hora de programar hay que tener en cuenta esta clasificación por algunas situaciones. Por ejemplo, si se desea mostrar la hora actual en una página web e incluimos un script o programa del lado del servidor, lo que veremos será la hora local en la ubicación del servidor que seguramente no es la misma hora del usuario (que generalmente está en otra ubicación). Una de las grandes ventajas de la programación en el servidor es que es transparente a los ojos del usuario.

Luego de conocer la clasificación de las herramientas de generación de contenido dinámico es importante comprender su funcionamiento, ya que se pueden combinar herramientas del lado del cliente y del lado del servidor y así explotar el potencial de cada una para obtener la funcionalidad necesaria para el desarrollo de aplicaciones web. Las posibles combinaciones se listan en el **Anexo 4, Tabla 2**, en el CD adjunto a este documento en: CD_TESIS:\Tesis\Anexos_Tesis\Anexos.pdf.

A continuación se detalla como funcionan las herramientas del lado del cliente, más adelante se detallará cómo funcionan las herramientas del lado del servidor.

2.2.2.1 HERRAMIENTAS DINÁMICAS DEL LADO DEL CLIENTE

Una aplicación web puede ser utilizada por múltiples usuarios simultáneamente, lo que implica que puede suponer una carga importante de trabajo para el servidor a medida que aumentan el número de usuarios y de tareas demandadas. La programación del lado del cliente tiene como principal ventaja que la ejecución de la aplicación se delega al cliente, con lo cual se evita recargar el trabajo al servidor. El servidor sólo envía el código, y es tarea del navegador interpretarlo.

Una de las cuestiones básicas para utilizar herramientas que corran en el cliente es la

descarga de trabajo que supone, así como la reducción en el **ancho de banda** empleado en las comunicaciones, ya que el número de accesos entre el cliente y el servidor disminuyen. Además, facilitan la validación interactiva de los datos y permiten utilizar recursos locales.

Dentro de las desventajas de este tipo de herramientas hay que destacar el tema de la seguridad, ya que debido a que se ejecuta código que proviene de una fuente externa, hay que tomar todas las medidas posibles para que no suponga el desencadenamiento de acciones no deseadas.

Otra cuestión a considerar es el hecho de que existe una dependencia del navegador utilizado. Puede suceder que cierta herramienta no funcione adecuadamente (el lenguaje no es interpretado por el navegador), por ejemplo, Netscape Navigator no interpreta **VBScript**; o que distintos navegadores interpreten el código de forma distinta.

Las principales funciones para las que se utilizan las herramientas de generación dinámica del lado del cliente son, por ejemplo:

- Poder acceder a los recursos del cliente y tomar el control del navegador.
- Manipular todos los objetos que haya en una página web.
- Validación dinámica de formularios, por medio de eventos.
- Desarrollo de páginas web interactivas con: sonidos, videos, menús expandibles.
- Proveer al cliente páginas web con controles avanzados, por ejemplo: *ActiveX*.
- Cambiar el contenido de las páginas web, poniendo valores por defecto.
- Reaccionar y realizar gestiones ante los eventos proporcionados por el usuario, por ejemplo ante un clic, movimientos del mouse, presión de una tecla, etc.

[WWW-17]

En general, el funcionamiento de las herramientas del lado del cliente se puede resumir en tres pasos:

- El navegador envía una petición al servidor.
- El servidor envía una respuesta que contiene código que el navegador entiende.
- El navegador interpreta el código enviado por el servidor y realiza una determinada acción.

conseguidos con una imagen **GIF** o una rutina en lenguaje **Java**. Sumados estos movimientos a efectos en tiempo real (una vez que la página web ha sido cargada) que reaccionan a movimientos del mouse realizados por el usuario, por ejemplo, cambio de colores del texto y/o tamaño del mismo, o desplazamientos de imágenes, conforman una página **HTML** con un nuevo dinamismo y más interactiva con el usuario.

Una animación de tipo **GIF**, **Java** o **AVI** se desarrolla en una pequeña porción de la pantalla, obligada por los recursos de memoria y velocidad de carga. Una animación realizada con **CSS/JavaScript** puede recorrer toda la pantalla, consumiendo la misma memoria (o poco más) y a la misma velocidad de carga (o poco más) que una imagen **GIF**.

CARACTERÍSTICAS DE DHTML

- Utiliza **CSS** para posicionar objetos en la pantalla.
- Utiliza **JavaScript** para mover, mostrar, ocultar o cambiar de aspecto los objetos.
- Los objetos se mueven, se ocultan, aparecen o cambian de aspecto al ser utilizados por el usuario, luego de haber sido cargada la página. Se puede decir, entonces, que **DHTML** es la combinación de **CSS** y **JavaScript** en un documento **HTML** para lograr cambios en su aspecto gráfico y lograr la interacción con el usuario.
- **DHTML** es la herramienta que introdujo el término "sitios multimedia" o "sitios hipermedia" en Internet.
- **DHTML** versión 4.0, es reconocido por Internet Explorer y Netscape Navigator.

VENTAJAS Y DESVENTAJAS DE DHTML

- Proporciona la técnica para realizar un sitio web interactivo, funcional y atractivo.
- Convierte las etiquetas de **HTML** en objetos programables lo que nos permite manipularlos a nuestro gusto.
- Todos los cambios de propiedades se producen sin que tenga que volverse a cargar la página web desde el servidor, y éste no requiere una configuración especial.
- Mejora la presentación de un sitio web con facilidad y flexibilidad pues permite separar el contenido de las propiedades de cada elemento.

- Tiene ciertos problemas de estandarización, sin embargo pueden ser corregidos con *JavaScript*.
- *DHTML* no requiere ningún tipo de **plug-in**⁴ para poder utilizarlo.
- Uno de sus más grandes inconvenientes es su compatibilidad, ya que sólo versiones más nuevas de los navegadores pueden ejecutar *DHTML* sin problemas.

FUNCIONAMIENTO DE DHTML

Lo primero que se debe hacer para crear una página con *DHTML* es definir las capas que contendrá dicha página. Esta definición se realiza de forma diferente en cada navegador, por tanto es importante conocer que etiqueta se utiliza en cada uno. Para evitar el problema de compatibilidad que esto implica puede utilizarse *JavaScript* para crear una página web que funcione en ambos navegadores.

Dentro de la capa se definen los atributos y características de la misma con *CSS*, después se puede manipular las capas u objetos de cada página, cambiando sus atributos o su posición(al nivel de pixeles) con *CSS-P*. Este cambio de atributos o posiciones, así como la interacción con el usuario se realiza con ayuda de *JavaScript*.

[WWW-18]

⁴ Módulo externo que al ser instalado junto al navegador permite visualizar formatos que no eran posibles. Cada empresa elabora su propio plug-in si desea que su formato sea reconocido por los navegadores.

MÓDULOS COMPILADOS:

CONTROLES ACTIVE X

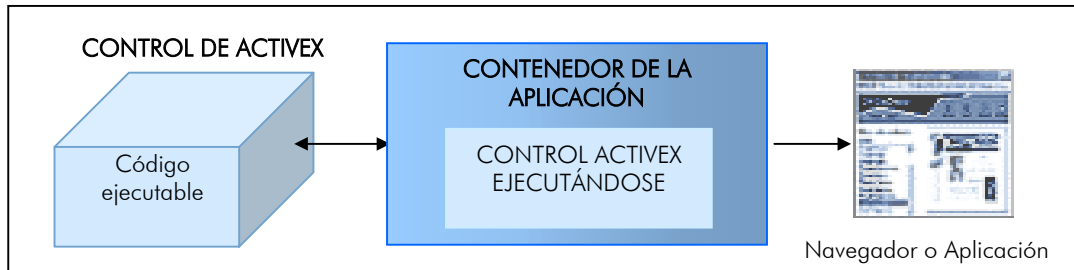


Fig. 2.6 Funcionamiento de los controles ActiveX

Aunque *HTML* continúa extendiéndose día a día, todavía no satisface todas las necesidades de los usuarios de Internet. Por ejemplo, si se desea publicar en una página web una presentación generada en Power Point, se podría convertirla a *HTML*, pero se perderían gran parte de los formatos. Lo ideal sería que al visitar las páginas, los usuarios abriesen la presentación; una opción para ello sería crear un enlace al archivo en Power Point. Pero si algún usuario no tiene instalado Power Point en su sistema o si se confunde al trabajar con dos aplicaciones en lugar de mantenerse usando el navegador, ¿entonces qué hacer?. Aquí es donde entran los controles *ActiveX*.

Desarrollados por Microsoft y prácticamente compatibles sólo con la plataforma Windows e Internet Explorer. Los controles *ActiveX* son como una versión reducida de los controles *OLE*, llamados controles *COM*. Estos controles actúan como una capa alrededor del código. Son la interfaz entre el código y el contenedor en que se ejecutan (por ejemplo el navegador). Ver Fig. 2.6.

En términos generales, consisten en un estándar que permite la ejecución de programas en el cliente, es decir, un programa se puede vincular como objeto en una página web (por ejemplo la presentación de Power Point) que siga el estándar *ActiveX* y ejecutarlo en cualquier equipo navegador que cargue la página web. [LIB-02]

Siguiendo con el ejemplo de la presentación de Power Point, se puede ejecutar dicha aplicación dentro del navegador. No es necesario instalar Power Point en la computadora cliente, sólo es necesario que se encuentre en el servidor. Incluso se pueden ver las barras

de herramientas, así los usuarios no tienen que abrir ninguna otra aplicación.

CARACTERÍSTICAS DE LOS CONTROLES ACTIVEX

- Los controles *ActiveX* tienen la extensión **.ocx**; y se basan en la idea principal de: *OLE* y *COM*, que es diseñar aplicaciones que puedan intercambiar datos y compartir código, de forma que sean accesibles unas desde otras. En concreto estos controles actúan en forma de pequeños módulos de aplicaciones, listos para ser incluidos por los programadores en aplicaciones finales, de los cuales se sabe como utilizarlos pero no como realizan su trabajo internamente.
- Son reutilizables desde otras páginas web e incluso desde otras aplicaciones, debido a que utilizan un modelo orientado a objetos para definir una jerarquía de componentes, entre los que pueden existir relaciones de dependencia. Cada componente (equivalente a un objeto, desde el punto de vista de la *POO*) se caracteriza por un comportamiento, que se puede alterar a través de sus métodos, propiedades o la interacción con otros objetos.
- Son capaces de interactuar con el usuario presentándose en pantalla (videos, animaciones), o pueden trabajar en segundo plano (realizar cálculos, procesar datos).
- Se insertan dentro de una página con las etiquetas **<object>** y **<param>**, que corresponden a las operaciones de inserción del componente y paso de sus parámetros de ejecución. Ver **Fig. 2.7**. Hay una gran similitud con los plug-in de Netscape.
- Aunque se trata de una herramienta de programación general, fue desarrollada con visión a implementar páginas más interactivas y en las que se pudiera emplear diversos lenguajes de programación perfectamente establecidos (*Java*, Visual Basic, Visual C++, Borland C++, Delphi, etc.).
- *ActiveX* consta de dos partes diferenciadas: el servidor y los clientes. La plataforma servidora debe contener los controles *ActiveX* o la referencia de donde se encuentran y en caso de no estar ya en la plataforma cliente, transferirlos a ésta, registrarlos en el sistema y ejecutar el código asociado.
- Los controles *ActiveX* son dependientes de las plataformas Windows de 32 bits, pero está desarrollándose su soporte para Macintosh (MacOS) y se espera que llegue a múltiples plataformas UNIX.
- Los controles *ActiveX*, desde la perspectiva del programador, se pueden utilizar en

lenguajes habituales en Internet, incluyendo *Java*, *JavaScript* y, por supuesto *VBScript*, además de todos los tradicionales que sean capaces de manejar controles tipo *OLE* (Compiladores de C/C++, Symantec o Watcom, Sybase Optima++, PowerBuilder, Visual Basic, Borland Delphi, etc.).

- Cada control *ActiveX* contiene una firma digital que permite comprobar la procedencia del código recibido, para reducir la posibilidad de ser afectado por un virus informático.

```
<object id="activemovie1" width=267 height=273>           //Inserción de un control ActiveX
<param name="_extentx" value="7038">                   //Establcer parámetros del control
<param name="moviewindowsize" value="2">
<param name="moviewindowwidth" value="262">
<param name="autostart" value="-1">
<param name="autorewind" value="0">
<param name="appearance" value="0">
<param name="filename" value="/videos/video.mpg">       //Establecer ruta del control
</object>                                               //Marca fin de inserción
```

Fig. 2.7 Código de inserción de un control ActiveMovie

VENTAJAS Y DESVENTAJAS DE LOS CONTROLES ACTIVEX

- La principal ventaja es que el código puede ser escrito en lenguajes como Visual Basic o C++ y puede hacer uso de los recursos locales del cliente. Esto supone que se pueden crear aplicaciones potentes y de gran utilidad para el usuario, que además son reutilizables.
- Una de sus grandes limitaciones es el tema de la seguridad, ya que se abre el camino para que los usuarios puedan realizar operaciones no deseadas.
- Otro problema que se puede plantear, es que los controles *ActiveX* se generan para una determinada plataforma, que puede no coincidir con la del cliente. Microsoft está tratando de liberar los *ActiveX* como estándar abierto para ser portado a otras plataformas.

FUNCIONAMIENTO DE LOS CONTROLES ACTIVEX

Cuando se encuentra un control *ActiveX* en una página web, se descargan automáticamente ciertos archivos al disco duro. Ello significa que hay que soportar el tiempo de descarga, que puede ser considerable si la conexión a Internet es lenta. Cuando estén los archivos en la computadora, la siguiente vez que se quiera ver la misma página

web, los controles *ActiveX* se ejecutan inmediatamente, ya que no se tiene que volver a descargarlos.

Al visitar una página web puede que no haya necesidad de descargar el control *ActiveX*, ya que pudo haber sido descargado anteriormente desde otra página. Como los archivos se copian en el disco duro, se debe estar seguro sí el sitio visitado es confiable. Se debería configurar el navegador para que alerte cuando encuentra un control *ActiveX*.

Una herramienta para insertar controles *ActiveX* en una página web es *ActiveX-Control Pad*.

APPLETS JAVA

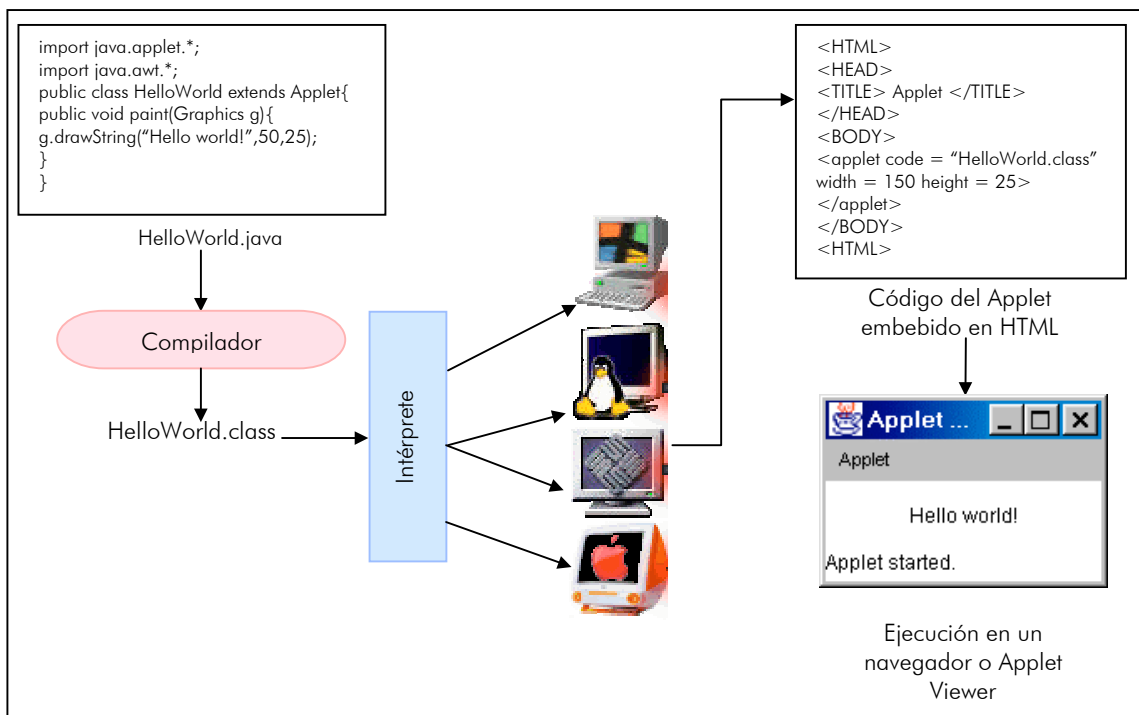


Fig. 2.8 Funcionamiento de un Applet

Un *Applet* es una mini-aplicación programada en lenguaje *Java*. Emplea la tecnología Cliente/Servidor para realizar animaciones interactivas, cálculos inmediatos, juegos, gráficos en 3D, texto en desplazamiento, etc. En general un *Applet* es una herramienta de generación de contenido dinámico del lado del cliente diseñada por la empresa Sun Microsystems, con la cual se pueden realizar diversas tareas sin la necesidad de una comunicación continua entre el servidor (donde reside el código) y los clientes (aquellos que

lo ejecutan a través del navegador).

Nota: La definición más extendida y resumida de Applet indica que es "una pequeña aplicación accesible en un servidor Internet, que se transporta por la red, se instala automáticamente y se ejecuta como parte de una página web en un navegador que soporte Java". En realidad, un Applet es una aplicación relativamente pequeña, basada en un formato gráfico sin representación independiente, es decir, se trata de un elemento a embeber en otras aplicaciones; es un componente en su sentido estricto. [LIB-03]

Como se puede ver en las definiciones anteriores, los Applets son similares a los controles ActiveX, debido a que son componentes insertados en una página web, y por lo tanto se ejecutan cuando son cargados en un navegador. Pero se requiere que el navegador cuente con una *JVM* capaz de ejecutar el código Java, dicha *JVM* está disponible con mayor o menor éxito en casi cualquier plataforma, ya que viene integrada con el navegador. Ver Fig. 2.9.

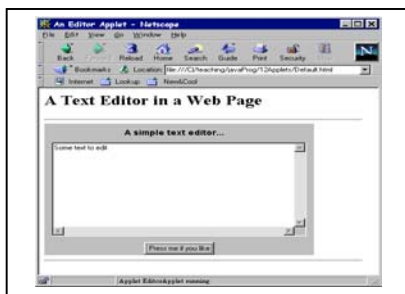


Fig. 2.9 Applet ejecutándose en Netscape Navigator

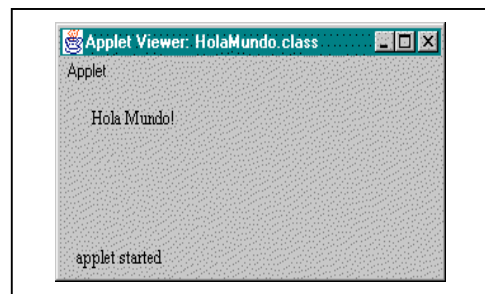


Fig. 2.10 Applet ejecutándose en el Applet Viewer

Hasta este punto se puede establecer a Internet como el entorno de ejecución de un Applet, pero también pueden ser ejecutados en el **Applet Viewer**⁵, que es una aplicación que permite ver Applets, sin necesidad de utilizar un navegador. Cabe destacar que el Applet Viewer trabaja sólo en el cliente. Ver Fig. 2.10.

⁵ Es una aplicación para visualizar los resultados de los Applets, viene con el **JDK** (Kit de Desarrollo Java).

CARACTERÍSTICAS DE LOS APPLETS

- Son archivos programados en *Java* que al ser compilados, en lugar de obtener un ejecutable, se obtiene un código de bytes que no necesita ser compilado para cada plataforma. Es decir se compila sólo una vez, o cada vez que el código cambia.
- Son almacenados con la extensión: **.class** (significa clases de *Java*).
- Los *Applets* son llamados por las páginas web de forma embebida en *HTML*, por medio de las etiquetas **<applet></applet>**, dentro de las cuales van los parámetros de ejecución y carga del *Applet*.
- No corren solos, siempre dentro de páginas web, embebidos en éstas.
- Se descargan a través de la red desde un servidor de Internet, hasta el navegador en cuya *JVM* se ejecutan.
- No tienen ventana propia: se ejecutan en paneles o contenedores, siendo estos los navegadores o el *Applet Viewer*.
- Pueden ser aplicaciones tan simples como: texto en movimiento, despliegue de la hora; hasta aplicaciones más complejas como: cálculos sofisticados, editores de texto(ver **Fig. 2.9**), graficadores de imágenes 3D, reproductores de sonido, etc.
- Tienen importantes restricciones de seguridad que se comprueban al llegar al navegador: sólo pueden leer y escribir archivos en el servidor del que han venido, sólo pueden acceder a una limitada información sobre el ordenador en el que se están ejecutando, etc. Con ciertas condiciones, los *Applets* “de confianza” pueden pasar por encima de estas restricciones.
- Los *Applets* se pueden convertir en aplicaciones *Java* y viceversa, por medio de algunas modificaciones en su código. Pero al convertirse en aplicaciones ya no pueden ser insertados en una página web, ya que funcionan como aplicaciones de consola.

VENTAJAS Y DESVENTAJAS DE LOS APPLETS

- Debido a que los *Applets* son programados en *Java*, la orientación a objetos les permite ser componentes reutilizables e independientes de la plataforma.
- Además de contar con varios navegadores(Netscape, Internet Explorer) que soportan su ejecución, tienen su propio navegador: *HotJava* y su propio visor de consola: *Applet Viewer*.

- Los *Applets* no trabajan con el sistema de archivos: leer, escribir, borrar, renombrar, listar, etc. Tampoco permiten el acceso al sistema.
- No se pueden establecer conexiones de red a máquinas distintas de la que envió el *Applet*.
- Los *Applets* no pueden hacer uso de todas las características de *Java*, por ejemplo, un *Applet* no puede contener menús, ni ventanas.
- Si el navegador no cuenta con la versión correcta de la *JVM*, el plug-in o el *JRE* correctos, el usuario no visualizará el *Applet* correctamente.
- Se pueden crear *Applets* seguros, pero al hacerlo las restricciones aumentan.

FUNCIONAMIENTO DE UN APPLET

Un *Applet* es programado en lenguaje *Java*, el fichero es almacenado con la extensión: **.java**, cuando se compila se genera un código de bytes con la extensión: **.class**. Una vez que se ha compilado se puede utilizar en cualquier plataforma, no es necesario compilar otra vez, salvo si se hacen cambios al código original.

Cuando se cuenta con el fichero **.class**, significa que ha sido compilado y el *Applet* está listo para ser interpretado por el navegador o por el Applet Viewer. Para ello se debe insertar de forma embebida en *HTML* y dentro de las etiquetas **<applet>** **</applet>** los parámetros del *Applet* a ser ejecutado. Ver **Fig. 2.8**.

Al igual que un control *ActiveX*, una vez que se ha cargado el *Applet* en la página web, la próxima vez que se abra dicha página se descargará más rápido, ya que el *Applet* se encuentra residente en memoria.

FLASH

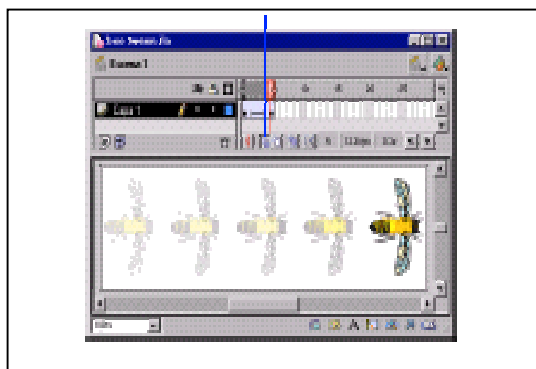


Fig. 2.11 Animación creada en Flash

De la empresa Macromedia, *Flash* es un lenguaje estándar en todo el mundo y es un excelente ejemplo de lo que quiere decir del lado del cliente. Aunque no se debe confundir el dinamismo con la capacidad de dar animación y movimiento a la página; si se requiere verdadero dinamismo se debe recurrir a la integración de *Flash* con las herramientas del lado del servidor.

La clave del éxito de *Flash* es su capacidad para crear sitios web vectoriales en los que se aplica: sonido, interactividad, gráficos y animaciones que se pueden visualizar en cualquier navegador en menos de 10 segundos. Las animaciones o películas *Flash* pueden tomar diferentes formas: desde un cómic animado, un cortometraje de ficción, hasta un interactivo juego de aventuras o un formulario de compra en cualquier tienda virtual, permitiendo de esta manera la interacción del usuario con la página web. Otras capacidades que presenta *Flash* son: crear controles de navegación, menús, logos animados, e incluso sitios web con capacidad sensorial. [WWW-19]

CARACTERÍSTICAS DE FLASH

- Trabaja con interpretación informática vectorial, los elementos se construyen a través de fórmulas matemáticas.
- *Flash* es independiente del navegador y el plug-in es universal, por lo que las animaciones se verán casi idénticamente en cualquier plataforma y navegador.
- Los archivos *Flash* rompen con la barrera del ancho de banda gracias a su rapidez en la descarga, en comparación con formatos similares.

- La forma en que *Flash* dota de interactividad y dinamismo a las páginas web es mediante animación de películas(desde barras de navegación hasta animaciones complejas con sonido).
- En *Flash* se programa en un lenguaje denominado **ActionScript**.
- Las películas *Flash* se pueden reproducir de varias formas:
 - En el **Flash Player**⁶(ver Fig. 2.11) o como proyector independiente sin disponer de Flash Player.
 - En navegadores que estén equipados con Flash Player.
 - Con el control *ActiveX* de *Flash*.

VENTAJAS Y DESVENTAJAS DE FLASH

- Se puede incluir sonidos con formato **.mp3** en las animaciones.
- *Flash* es una tecnología con mucho futuro por su funcionamiento y facilidad de uso.
- Soporta gran cantidad de formatos de imágenes, audio, video, etc.
- La única desventaja de *Flash*, es que para poder visualizar las animaciones, es necesario tener el plug-in.

FUNCIONAMIENTO DE FLASH

Para que un usuario pueda visualizar en Internet una página web que contenga animaciones o películas *Flash*, es necesario que cuente con el reproductor Flash Player en la máquina y que el servidor web del sitio esté configurado para reconocer este tipo de películas; entonces el navegador estará listo para reproducir una página web generada con *Flash*. Las páginas de este tipo se descargan rápidamente (a pesar de que cuentan con varias imágenes o sonidos), debido a que *Flash* comprime todos sus archivos y el resultado final es una película de unos pocos KB.

Además las películas *Flash* son gráficos vectoriales compactos que se descargan y se adaptan de inmediato al tamaño de la pantalla.

A continuación las herramientas de generación de contenido dinámico clasificadas como

⁶ Programa para crear animaciones Flash, cuenta con un editor de texto y un editor de películas.

lenguajes de scripts o guiones.

LENGUAJES DE SCRIPTS O GUIONES:

El uso de este tipo de herramientas está muy extendido actualmente. Los lenguajes de script o guiones ofrecen una forma de extender los documentos *HTML* en formas altamente activas e interactivas:

- Permiten modificar el contenido de las páginas web de manera dinámica, en el cliente.
- Se ejecutan al dispararse eventos específicos como la carga y descarga de la página, los movimientos del mouse, al pulsar un botón o seleccionar una casilla de verificación.
- Pueden procesar y validar los datos de un formulario.
- Actúan sobre el navegador a través de objetos integrados que representan a la página, a la ventana activa, a cada uno de los elementos de un formulario, etc.
- Pueden combinarse con otras herramientas, por ejemplo con scripts *CGI* o con *DHTML*.
- Con respecto a la seguridad, los lenguajes de scripts o guiones suelen presentar fuertes restricciones de acceso a los recursos de la máquina del cliente.

Nota: Sobre *CGI* ver en sección 2.2.2.2 *Herramientas dinámicas del lado del servidor*

A continuación la descripción de los lenguajes de scripts o guiones: *JavaScript* y *VBScript*.

JAVASCRIPT

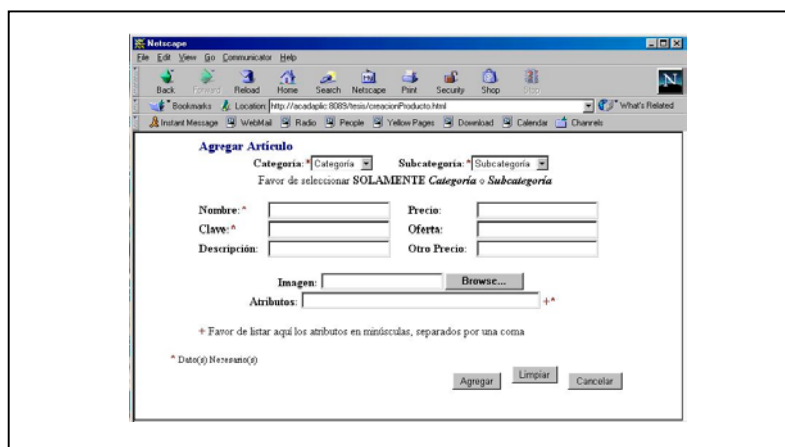


Fig. 2.12 Página web que responde a eventos programados mediante JavaScript

JavaScript es un lenguaje de scripts compacto basado en objetos, no orientado a objetos. Originalmente era denominado *LiveScript*, y fue desarrollado por Netscape para su navegador Netscape Navigator 2.0. Fue éste el primer cliente en incorporarlo. Actualmente se ejecuta sobre 16 plataformas diferentes, incluyendo los entornos de Microsoft e incluso el Internet Explorer 3.0.

JavaScript se puede considerar como la herramienta de generación de contenido dinámico del lado del cliente más soportada y popular. Aunque su nombre se derivó luego de adoptar algunas características del lenguaje *Java*, no se debe confundir con éste.

JavaScript permite la realización de aplicaciones de propósito general a través de Internet y aunque no está diseñado para el desarrollo de grandes aplicaciones web, es suficiente para la implementación de aplicaciones un poco complejas. Por ejemplo, una aplicación escrita en *JavaScript* puede ser incrustada en una página *HTML* proporcionando un mecanismo para la detección y tratamiento de eventos, como clics del ratón o validación de entradas realizadas en formularios, dotando de gran dinamismo a la página web. Ver **Fig. 2.12**.

Sin existir comunicación a través de la red una página *HTML* con *JavaScript* incrustado puede interpretar y alertar al usuario con una ventana de diálogo, de que las entradas de los formularios no son válidas. O bien realizar algún tipo de acción como ejecutar un fichero de sonido, un *Applet* de *Java*, etc.

JavaScript además de ser utilizado del lado del cliente, también puede ser utilizado del lado del servidor. Aunque *JavaScript* de cliente y de servidor comparten el mismo conjunto base de funciones y características, en algunos casos se utilizan de distinta forma. Los componentes de *JavaScript* son los siguientes:

- Núcleo de *JavaScript*
- *JavaScript* para Cliente(código y funciones para el cliente).
- *JavaScript* para Servidor(código y funciones para el servidor).

El código *JavaScript* para cliente se integra directamente en páginas *HTML* y es interpretado en su totalidad por el cliente web en tiempo de ejecución; las aplicaciones *JavaScript* desarrolladas para servidores se pueden compilar antes de instalarlas en dichos servidores.

CARACTERÍSTICAS DE JAVASCRIPT

- Es un lenguaje interpretado, no se compila. El navegador interpreta el código que recibe, tal y como *JavaScript* lo envía.
- La manera más convencional en que aparece código *JavaScript* en una página web es en forma de programa o como manejador de eventos del mouse, teclado.
- Presenta la posibilidad de ser utilizado del lado del cliente y del lado del servidor. Del lado del servidor se compila a código binario y se conoce como **SSJS**.
- Se incrusta en una página web por medio de las etiquetas:

```
<script language="JavaScript">  
</script>
```
- El código *JavaScript* también puede ser almacenado en un archivo **.js** y ser llamado dentro de una página web por medio de las etiquetas:

```
<script language="JavaScript" src="ruta_archivo/archivo_javascript.js">  
</script>
```
- No es obligatoria la declaración de tipos de variables.

VENTAJAS Y DESVENTAJAS DE JAVASCRIPT

- La principal ventaja de *JavaScript*, es que es soportado por más navegadores que su oponente *VBScript* de Microsoft.
- Entre muchas cosas se pueden crear botones dinámicos, ventanas de diálogo, animar imágenes, generar sonidos, incluso implementar algunos cálculos y manejo de datos sin conexión a la red.
- *JavaScript* es capaz de controlar los *Applets Java*.
- No permite escribir aplicaciones independientes sin la presencia de un intérprete.
- Tiene una capacidad muy limitada para leer y escribir archivos.
- Depende de que el navegador cuente con la versión adecuada del intérprete y la *JVM*.

FUNCIONAMIENTO DE JAVASCRIPT

Como *JavaScript* funciona del lado del cliente y del servidor, es necesario detallar el funcionamiento de cada tipo.

Los navegadores que soportan *JavaScript* pueden interpretar sentencias *JavaScript* incrustadas en un documento *HTML*. Cuando el navegador solicita una página que contenga *JavaScript*, el servidor le envía por la red el contenido completo de la página, incluyendo todos los códigos *HTML* y las sentencias *JavaScript* que pudieran existir. El navegador lee entonces la página de forma secuencial desde el principio hasta el final, representando visualmente los códigos *HTML* y ejecutando las sentencias *JavaScript* conforme avanza el proceso de lectura e interpretación.

El código *JavaScript* incrustado en una página web puede dar respuesta a eventos de usuario, tales como la pulsación de un botón del ratón, la entrada de datos en un formulario y la navegación por una página.

Por ejemplo, se puede crear una función *JavaScript* que permita verificar que los datos introducidos por el usuario en un formulario (número de teléfono, código postal, etc.) tengan el formato correcto. En este caso, lo importante es que, sin necesidad de realizar ninguna transmisión por la red, se puede validar dichos datos, mostrando al usuario un cuadro de diálogo en caso de que sean incorrectos.

En el servidor, *JavaScript* también está integrado en páginas web. Las aplicaciones se compilan generando archivos binarios. Las sentencias de *JavaScript* del servidor pueden realizar tareas como:

- Conectarse a bases de datos relacionales de varios fabricantes a través de **LiveWire**.
- Compartir información entre usuarios de una aplicación.
- Acceder al sistema de archivos del servidor.
- Comunicarse con otras aplicaciones a través de **LiveConnect** y *Java*.
- Accesar a aplicaciones basadas en objetos distribuidos **CORBA**.
- Gestionar Sesiones.

[WWW-20]

VBSCRIPT

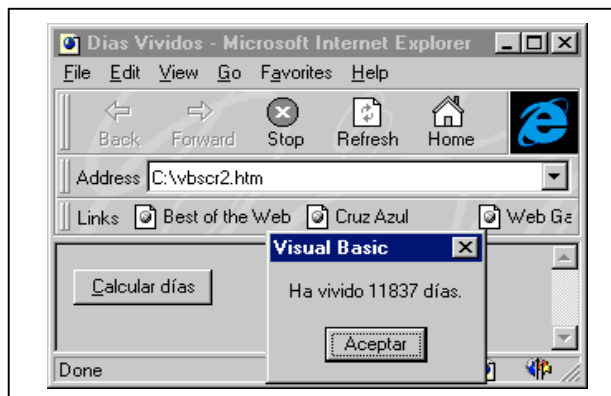


Fig. 2.13 Página web que responde a eventos programados con VBScript

Es la respuesta de Microsoft a la popularidad de *JavaScript*, pero compatible sólo con Internet Explorer. *VBScript*, prácticamente tiene las mismas características que *JavaScript*. Pero su desventaja es que no es aceptado por los navegadores Netscape. Así como *JavaScript* no se debe confundir con *Java*, *VBScript* no se debe confundir con Visual Basic.

VBScript es un subconjunto de **VBA** que, a su vez es un subconjunto del lenguaje Visual Basic. *VBScript* fundamentalmente se integra dentro de *HTML* para ampliar las características de una página web.

Tiene la misma ideología orientada a eventos de Visual Basic, con la adición de estrategias de seguridad para crear scripts confiables, eficientes y compactos.

Al igual que *JavaScript*, *VBScript* puede ser utilizado en el lado del cliente y en el lado del servidor (su sintaxis es idéntica). El modo de funcionamiento de *VBScript* para construir efectos especiales en páginas web es muy similar al utilizado en *JavaScript* y los recursos a los que se puede acceder también son los mismos: el navegador.

VBScript en el lado del cliente no puede utilizar instrucciones para representar elementos de interfaz de usuario.

CARACTERÍSTICAS DE VBSCRIPT

- Es interpretado principalmente por el navegador Internet Explorer.
- Es el lenguaje de scripts favorito para ser usado en la creación de las páginas

dinámicas del lado del servidor: **ASP**.

- Es un lenguaje interpretado, no se compila. El navegador interpreta el código que recibe, tal y como *VBScript* lo envía.
- Se incrusta en una página web, por medio de las etiquetas:

```
<script language="VBScript">  
</script>
```
- No tiene tipos de variables, ni se declaran constantes.
- Cuenta con una cantidad suficiente de instrucciones para crear el contenido dinámico de una página web, hacer más eficiente la validación de formularios, procesar información, manipular los objetos básicos de *HTML* y los de tipo *ActiveX*, crear juegos, etc.

VENTAJAS Y DESVENTAJAS DE VBSCRIPT

- Presenta la posibilidad de ser utilizado del lado del cliente y del lado del servidor.
- Es soportado íntegramente sólo por Internet Explorer **Fig. 2.13**, siendo una desventaja frente a *JavaScript*.
- Con *VBScript* no hay acceso a la Entrada/Salida de archivos.

FUNCIONAMIENTO DE VBSCRIPT

Para que un script de *VBScript* funcione en el navegador, éste debe tener las posibilidades de interpretar el código relacionado. Todos los navegadores pueden interpretar *HTML* en cualquiera de sus versiones, sin embargo no todos pueden interpretar un script. Para ello se necesita que el navegador incluya el archivo en tiempo de ejecución (Run-Time) de *VBScript*. Por el momento solamente el Internet Explorer de Microsoft (y con agregados, el Netscape) tiene este archivo en tiempo de ejecución incluido, lo que hace que el *VBScript* sólo funcione en Windows y en Macintosh, pero se espera que otros navegadores incluyan estas posibilidades.

VBScript, al igual que *JavaScript*, es un lenguaje 100% interpretado. Sin embargo su velocidad de ejecución es lo suficientemente buena como para hacer más eficiente un proceso de validación de información directamente en la máquina cliente, y evitar con esto el tráfico en Internet sólo para hacer la misma validación y devolver un mensaje de error al

encontrar algún dato no válido. Este tipo de cuestiones se realiza con mayor eficiencia en el cliente.

El código de *VBScript* se transmite a la par de la página web en modo de texto, no está compilado (debido a que es interpretado) y por lo tanto, sobrecarga un poco la página que el usuario está descargando a su máquina. Por esto el código de *VBScript* debe ser compacto para que, a la vez que se ejecute con mayor eficiencia, se reduzca el tiempo de transmisión.

2.2.2.2 HERRAMIENTAS DINÁMICAS DEL LADO DEL SERVIDOR

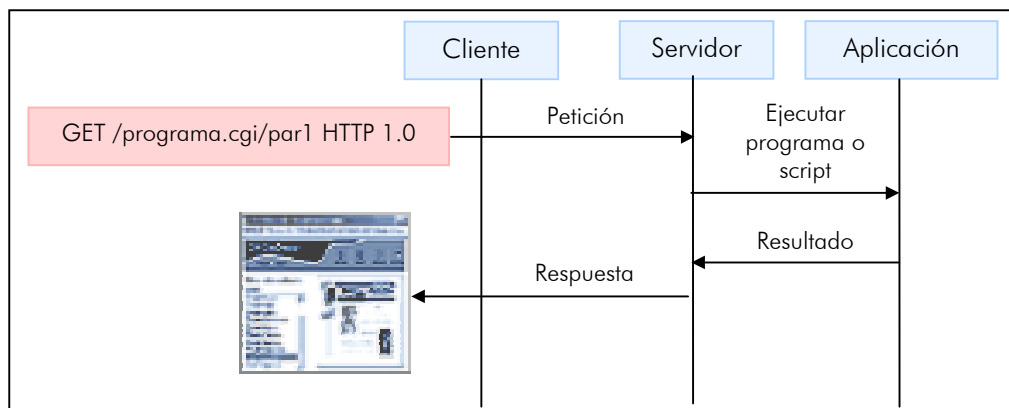


Fig. 2.14 Funcionamiento de las herramientas del lado del servidor

En el caso de las herramientas del lado del servidor, el código de la aplicación web se ejecuta en el servidor. Cuando una página es solicitada por parte de un cliente, el servidor ejecuta una aplicación (scripts o programas en el servidor, que pueden conectarse con bases de datos, entre otras cosas) y genera una página de resultado, que envía al cliente. La página web de resultado contiene únicamente código que entiende el navegador, por lo que puede ser interpretada independientemente por cualquier navegador sin lugar a errores, de este modo la compatibilidad está garantizada. Ver **Fig. 2.14**.

La independencia del navegador y contar con todos los recursos del servidor son ventajas significativas con relación a las herramientas del lado del cliente. Dichos recursos que podrían ser gestores de bases de datos, servidores de correo o el propio sistema de archivos del servidor, son los que permiten construir todo tipo de aplicaciones.

Como ventaja adicional se puede destacar que el código de las aplicaciones web nunca llega al cliente, ya que al navegador sólo le llega *HTML*, *DHTML*, *WML*, *JavaScript*, etc., y esto implica que los visitantes nunca van a poder acceder al núcleo de las aplicaciones, es decir, a los scripts o programas del lado del servidor. Además no es imprescindible la presencia de un formulario para que un script o programa del lado del servidor se ejecute.

La desventaja está en que el servidor se sobrecarga de trabajo ya que además de servir páginas es responsable de ejecutar aplicaciones. Mientras existe más demanda, se consumen más recursos.

Las herramientas del lado del servidor se deben utilizar cuando:

- El sitio web debe funcionar en cualquier navegador.
- Se desea proteger el código fuente de las aplicaciones web.
- Se necesita mayor velocidad de procesamiento; los scripts del servidor son más rápidos.
- Se requiere interacción con Bases de Datos en el servidor.
- Es necesario utilizar **variables de entorno** (variables *HTTP*) del servidor.
- Se requiere controlar la permanencia del usuario en un sitio web mediante Sesiones.
- Se necesita crear páginas con contenido dinámico, a partir de las entradas del usuario.
- Se necesita comprobar las capacidades del navegador del usuario.
- Es necesario una potencia de cálculo y procesamiento que no se puede obtener en el cliente.
- Se necesita establecer comunicación entre el servidor web y el navegador. Para este propósito, existen numerosas variables *HTTP* que van desde la dirección **IP** del usuario hasta el tipo de navegador que está utilizando. Esta información puede ser utilizada por los scripts del servidor.

[WWW-21]

En general, el funcionamiento de las herramientas del lado del servidor se puede resumir en cuatro pasos:

- El navegador envía una petición.
- El servidor ejecuta una aplicación web(programas o scripts) que realiza una determinada acción.

- El servidor envía el resultado de dicha aplicación al navegador.
- El navegador muestra al cliente el resultado recibido del servidor.

Para el desarrollo de aplicaciones web del lado del servidor, las herramientas que se pueden utilizar se dividen en tres categorías:

- El protocolo estándar *CGI* y su variante *FastCGI*.
- *APIs* provistas por el servidor web: *ISAPI*, *NSAPI*.
- Módulos del servidor, para interpretar:
 - *PERL*, directivas *SSI*.
 - Scripts incrustados en páginas web, para crear páginas de tipo: *PHP*(.php), *ASP*(.asp), *JSP*(.jsp), *CFML*(.cfm).

Cabe destacar que con las herramientas del lado del servidor se pueden crear programas, aplicaciones o scripts que pueden ser invocados desde cualquier elemento de un formulario (botones, enlaces) o desde la *URL* directamente. Por lo cual hay que tener mucho cuidado en programarlos.

Se puede comprender de mejor manera la clasificación y funcionamiento de las herramientas tanto del lado del cliente y del lado del servidor, por medio de la siguiente figura. Ver **Fig. 2.15**.

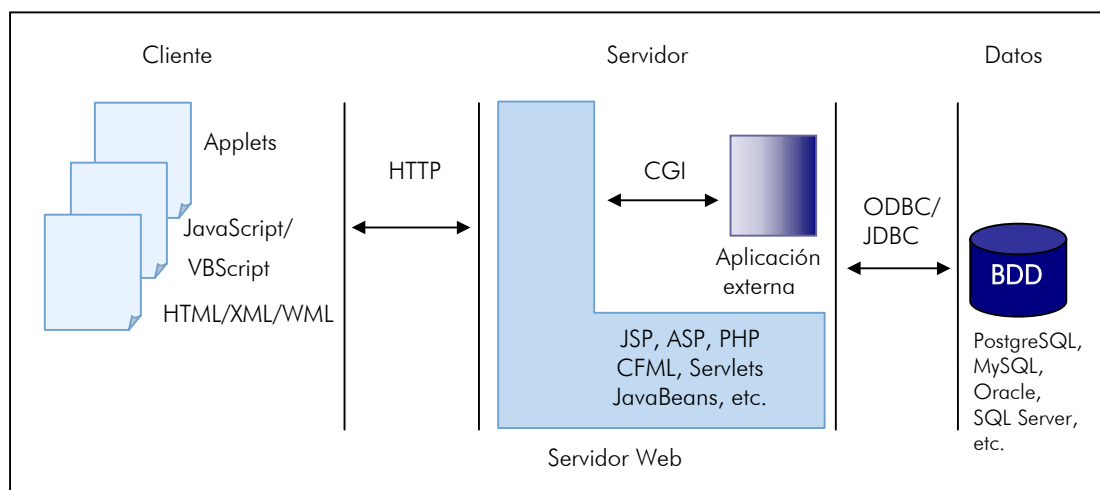


Fig. 2.15 Clasificación de herramientas de generación de contenido dinámico

PROCOLO ESTÁNDAR CGI

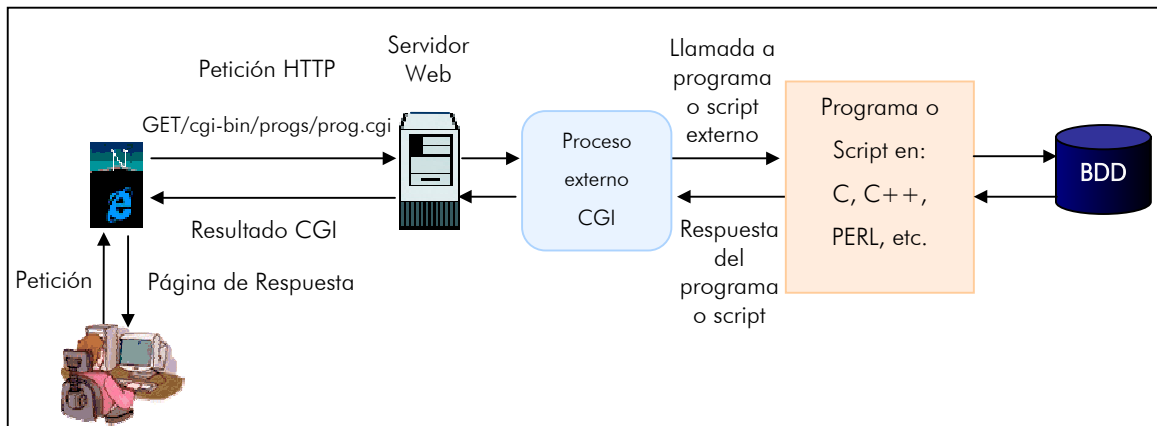


Fig. 2.16 Funcionamiento de CGI

Las aplicaciones *CGI* hasta no hace mucho eran la única alternativa real para crear páginas dinámicas. En general *CGI* es un protocolo que fue creado por *NCSA* para establecer un mecanismo estándar de comunicación entre un servidor web y un lenguaje de programación, de tal forma que desde aplicaciones programadas en dicho lenguaje el servidor web pueda procesar peticiones de usuario.

Con la ayuda de *CGI*, un servidor web pasa una petición de usuario a una aplicación (programa o script que se encuentra y se ejecuta en tiempo real en dicho servidor), recibe datos que el usuario envía, los procesa y retransmite como respuesta a la solicitud del navegador. La respuesta es generada como resultado de la ejecución de la aplicación.

Una manera común de utilizar *CGI* es al enviar formularios, donde el navegador envía los datos a la aplicación *CGI* que reside en el servidor y éste se encarga de procesarlos para generar una página de respuesta. Los datos que el usuario envía se almacenan en *variables de entorno* protegidas.

El mecanismo de las aplicaciones *CGI*, se puede utilizar comúnmente para programar contadores, motores de búsqueda, generadores de e-mail automático, foros de discusión, chats, comercio electrónico, mapas de imágenes, juegos en línea e interactuar con bases de datos y formularios, entre otros.

CARACTERÍSTICAS DE CGI

- Una aplicación *CGI* puede ser escrita como programa(lenguajes compilados) o como script(lenguajes interpretados). Los lenguajes compilados deben permitir crear archivos ejecutables(C, C++, Pascal, Java, TCL, Fortran, Delphi, **Python**, etc.). Cabe destacar que *PERL* que es un lenguaje interpretado es el más utilizado. Sin embargo los lenguajes compilados son más seguros, ya que impiden el acceso al código.
- Un lenguaje para crear *CGIs* debe ser capaz de acceder a las *variables de entorno*.
- La extensión para un programa *CGI* depende del lenguaje o como se configure el servidor web. Generalmente la extensión es **.cgi**. Si es programado en *PERL* es **.pl**.
- Requiere obligatoriamente de un servidor web, ya que ahí reside la aplicación.
- Permite desarrollo de aplicaciones distribuidas, acceso concurrente y/o compartido.
- Se crea un proceso cada vez que se debe ejecutar la aplicación *CGI*, significando un proceso por cada petición de usuario.
- La ejecución de un programa en el servidor es relativamente peligrosa, por lo tanto un *CGI* debe estar en un directorio determinado(/cgi-bin/) y ser programado con cuidado. Ya que puede ser invocado desde cualquier elemento del formulario(botón, enlace) o desde la *URL*.

VENTAJAS Y DESVENTAJAS DE CGI

- Esta interfaz rápidamente se volvió un estándar, siendo implementada en todo tipo de servidores web, ya que permite la elección del lenguaje.
- Es portable, trabaja sobre cualquier plataforma que soporte *CGI* y el lenguaje en que esté programada la aplicación(programa o scripts). Los *CGI* para Windows se denominan **WinCGI** y deben ser ejecutables de modo consola.
- El problema con los programas en *CGI* radica cuando las solicitudes por parte de los clientes son demasiadas para el servidor, entonces el tiempo de respuesta aumenta en la creación de la página de resultado.
- Otro problema es que un *CGI* no conoce, ni maneja, ni implementa el concepto de sesiones.
- Un gran problema es que la lógica de desarrollo está incluida en el programa o script que es el encargado de la creación de la página web de respuesta, de esta manera es

casi imposible cambiar el diseño de la página, debido a que la información necesita concordar con la presentación.

- Una aplicación *CGI* mal diseñada podría permitir el acceso total o parcial al servidor.

FUNCIONAMIENTO DE CGI

Un cliente, desde un navegador, llena los datos en un formulario, dichos datos son enviados al servidor web por medio de las *variables de entorno* en forma de petición, dicha petición es atendida por una aplicación *CGI*. La aplicación *CGI* procesa la petición por medio de sus programas o scripts, accesa a la base de datos y a otros recursos del servidor y genera una página web de respuesta hacia el cliente. Dicha página es enviada línea a línea por la salida estándar. Ver **Fig. 2.16**.

Cada vez que el servidor recibe una solicitud que accesa a un programa *CGI*, debe crear un nuevo proceso para la ejecución de dicho programa, pasándole a través de *variables de entorno* o datos, desde la entrada estándar, la información que pudiera ser necesaria para generar la respuesta. Crear un nuevo proceso para cada solicitud requiere de tiempo y de significantes recursos del servidor, limitando el número de solicitudes que pueden ser procesadas de manera concurrente. [WWW-22]

FastCGI

FastCGI es una alternativa a *CGI* estándar, cuya diferencia radica principalmente en el hecho de que el servidor crea un único proceso persistente por cada programa *FastCGI* en lugar de por cada solicitud del cliente. Este proceso no muere, tras atender a un usuario, sino que se queda residente en memoria esperando por una nueva petición.

Considerando que cada uno de estos procesos puede estar por ejemplo ejecutando el intérprete de *PERL*, este modelo no parece tan distinto del *CGI* estándar. Una solución que *FastCGI* ofrece para este problema es su habilidad para distribuir todos estos procesos entre múltiples servidores, e incluso correr en una máquina distinta al servidor web. [WWW-23]

Existen implementaciones disponibles para los principales servidores web del mercado: Apache, Netscape, Internet Information Server, Zeus Web Server y bibliotecas para su

utilización con los lenguajes: C, C++, PERL, Java, Python, etc.

En resumen, *FastCGI* intenta mejorar al estándar *CGI* y a las *APIs*, de las que se habla a continuación.

APIs PROVISTAS POR EL SERVIDOR WEB:

Una *API* es una interfaz de programación de aplicaciones que describe en detalle los métodos de un sistema operativo (imprimir en pantalla, leer del teclado, escribir en el disco) o de una aplicación, los mismos que pueden ser usados por un programador mediante llamadas a ellos. Las *APIs* son provistas por el servidor web (para extender su funcionalidad), es decir que el servidor provee un lenguaje en el cual se pueden desarrollar aplicaciones.

Este esquema es mucho más eficiente que *CGI*, ya que el servidor es el encargado de ejecutar las aplicaciones, en forma directa, sin necesidad de crear un nuevo proceso cada vez. En la actualidad existen las siguientes *APIs*:

- *ISAPI* (Internet Server API – API del Servidor Internet Information Server).
- *NSAPI* (Netscape API – API de los servidores web de Netscape).

ISAPI

ISAPI es una extensión de *API* creada por Microsoft y Process Software para el desarrollo de aplicaciones web. *ISAPI* es similar a los *CGIs* en cuanto a la funcionalidad, pero posee varias ventajas con relación a éstos. La característica más destacada de las aplicaciones *ISAPI* es su rendimiento, ya que al ser *DLLs* (no programas, ni scripts) se ejecutan en el espacio de direcciones del servidor Internet Information Server, se cargan en memoria y se mantienen permanentemente desde que arranca el servidor, por lo que no es necesario crear un nuevo proceso, ahorrando un tiempo considerable en posteriores llamadas.

Las *DLLs* de *ISAPI* al funcionar como parte del servidor web siempre están listas para ejecutarse. Si no hay peticiones en un determinado tiempo se descargan de memoria. Por lo tanto, *ISAPI* consigue menor sobrecarga que los *CGI*, siendo más eficiente desde el punto de vista de uso de memoria y velocidad.

El usuario también puede desarrollar sus propias extensiones *ISAPI* con cualquier lenguaje que permita crear *DLLs*, por ejemplo: C, C++, Pascal, Visual Basic.

NSAPI

NSAPI es una interfaz para desarrollo de aplicaciones web, desarrollada por Netscape Communications Corporation. *NSAPI* permite extender el núcleo del servidor Netscape, para ello utiliza la interfaz *SAF*, que es un conjunto de funciones que el servidor llama durante el proceso petición/respuesta.

Cada *SAF* retorna un código de resultado, el cual le indica al servidor que es lo que sucede, que no hizo o lo que ha fallado. Basándose en el código de resultado, el servidor decide que hacer: continuar el proceso, saltar a la próxima *SAF* o abortar el proceso.

Con *NSAPI* el programador también tiene la oportunidad de escribir sus propias funciones para extender el núcleo de la funcionalidad de los servidores Netscape. Al igual que *ISAPI*, una aplicación escrita en *NSAPI*, se ejecuta en el servidor web en lugar de crear otro proceso por cada petición.

Sin embargo, las desventajas de las *APIs* anteriores son importantes:

- Las aplicaciones creadas en este marco no son portables ya que sólo pueden ejecutarse en un servidor web específico:
 - *ISAPI* en Internet Information Server y servidores web de Microsoft.
 - *NSAPI* en Netscape Server y servidores web derivados de éste.
- Frecuentemente un error de programación de *ISAPI/NSAPI* podría ocasionar que el servidor deje de funcionar, genere un error, se cuelgue, pierda memoria u otros problemas, ocasionando que este tipo de aplicación no sea confiable.
- Es más complicado escribir extensiones *ISAPI* y funciones *NSAPI* que programas *CGI*.
- En realidad *ISAPI/NSAPI* son *CGI* mejorados y de alto rendimiento.
- Este tipo de *APIs* está actualmente en desuso.

[WWW-23]

MÓDULOS DEL SERVIDOR WEB PARA INTERPRETAR SCRIPTS:

```
<html>                                     //Inicio de página HTML
<body>                                     //Cuerpo de la página
<?php for $i = 3; $i<7;$i++; $?>           //Código PHP incrustado en HTML
  <font size = <?php echo $i; $?>> Hello World </font> //Ciclo for para imprimir en HTML
<?php $i++; $?>
<br>
</body>                                    //Fin de cuerpo de página
<html>                                     //Fin de HTML
```

Fig. 2.17 Ejemplo de Script(código PHP) incrustado en HTML, interpretado por el servidor web

En la actualidad, los servidores web aceptan ampliaciones de funcionalidad mediante “módulos”, así como se puede añadir plug-ins para que los navegadores interpreten nuevos formatos, los servidores web pueden ser mejorados añadiéndoles módulos que interpreten scripts en determinados lenguajes. Entre los lenguajes de scripts que pueden interpretar los servidores web están: *PERL*, directivas *SSI* y lenguajes para crear páginas de tipo: *PHP*, *ASP*, *JSP* y *CFML*. Dichos módulos se ejecutan dentro del servidor web y los scripts a ser interpretados se incrustan en *HTML/XML/WML* o se llaman desde ahí. Ver **Fig. 2.17**.

De esta forma se logra eficiencia ya que el servidor web no necesita crear un nuevo proceso por cada aplicación que ejecuta. Las aplicaciones generadas en la mayoría de casos son portables y confiables. En el caso de los módulos, si se producen errores y el módulo es sólido, no afecta al servidor web.

Como conclusión, las aplicaciones web desarrolladas con este tipo de herramientas (scripts en lenguajes interpretados), brindan más ventajas en comparación con *CGI* y *API*. Actualmente, las páginas de tipo: *PHP*(.php), *ASP*(.asp) y *JSP*(.jsp), son las más utilizadas para generar aplicaciones web. Por esta razón se pondrá más énfasis en el estudio de la generación de éstas páginas. Más adelante se realizará un análisis comparativo de dichas herramientas, para determinar el comportamiento de una herramienta gratuita, una comercial y una compartida respectivamente. Ver **Tabla 2.1, 2.2, 2.3 y 2.4** al final del presente capítulo.

A continuación las herramientas de generación de contenido dinámico que pertenecen al tipo de scripts en lenguajes interpretados mediante módulos del servidor web.

PERL

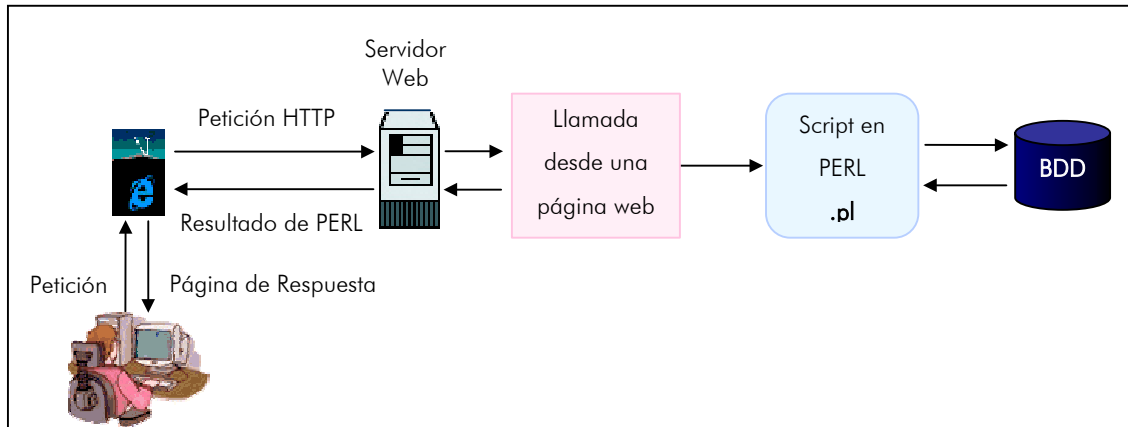


Fig. 2.18 Funcionamiento de PERL

Por mucho tiempo fue *PERL* el lenguaje elegido para desarrollar aplicaciones (de tipo CGI) para la web, como: carritos de compras, buscadores, mails, sitios de subastas, etc. *PERL* fue creado originalmente para ser un lenguaje de navegación optimizado, que permita buscar de una manera eficaz archivos de texto y generar reportes basados en la información extraída, de ahí se deriva su nombre: **P**ractical **E**xtraction and **R**eport Language. [WWW-25]

Nota: A pesar de su enorme popularidad, *PERL* no fue desarrollado desde el principio con la web en mente, debido a esto, se hicieron evidentes sus puntos débiles siendo la causa principal de la aparición de nuevas herramientas del lado del servidor.

CARACTERÍSTICAS DE PERL

- No se compila, se interpreta, comportándose en un estado intermedio. Sin embargo, ya existen compiladores de *PERL* con la versión 5, pero para Windows.
- Es bastante utilizado para muchos sistemas manipuladores de tareas como lenguaje de contenido dinámico. Principalmente para hacer scripts CGI.
- La extensión de un script *PERL* es **.pl**.
- Tiene una orientación más práctica (facilidad de uso y eficiencia) que estética (no establece ninguna filosofía de programación).
- Es un lenguaje intermedio entre los shell scripts y la programación en C (parecido en su sintaxis y en varias funciones para manipulación de cadenas y archivos).
- *PERL* es generalmente portable. Cabe destacar que hay algunas funciones que son

dependientes de la plataforma.

- Es una utilería que pretende facilitar el procesamiento de grandes volúmenes de información sin sacrificar el rendimiento.
- Provee una gran cantidad de módulos tanto para la generación de aplicaciones particulares así como para la administración de servidores web.
- Los servidores web que soportan *PERL* son las versiones de Apache.
- *PERL* se inserta en páginas web solamente por medio de ActivePerl, mason y embperl.

VENTAJAS Y DESVENTAJAS DE PERL

- *PERL* es software libre, siendo fácil de obtener e instalar.
- Es sumamente compatible entre versiones, aunque la versión 5.0 es orientada a objetos.
- Existe una versión mejorada de *PERL*, denominada *mod_perl*, que funciona como módulo en el servidor web Apache.
- *PERL* no tiene limitaciones en: tamaño de arreglos, recursión, nombres de variables.
- Puede determinar datos derivados de fuentes inseguras para prevenir operaciones peligrosas.
- Tiene la habilidad de procesar múltiples operaciones con un único comando.
- Trabaja con diferentes herramientas de texto como: *WAIS*; bases de datos relacionales como: Sybase, Oracle; bases de datos de objetos relacionales como: *Illustra*. [WWW-26]
- *PERL* provee herramientas muy sencillas de uso para la generación de páginas dinámicas vía procesamiento de información de entrada/salida a través del Internet.
- La gran desventaja de *PERL* es que no genera código binario.
- Otra desventaja está en la lentitud de ejecución de los scripts, al igual que *CGI* crea un nuevo proceso por cada petición.

FUNCIONAMIENTO DE PERL

PERL no es ni un compilador ni un intérprete completamente, está en un punto intermedio. Cuando se ejecuta un programa en *PERL*, se transforma el código fuente a un código intermedio en memoria, se optimiza como si fuese a elaborar un programa ejecutable, pero

es ejecutado por un motor como si se tratase de un intérprete. El resultado final, es algo que se comporta como un intérprete pero que tiene un rendimiento comparativo al de programas compilados. Cuando es usado para programar aplicaciones CGI, lo único que hay que hacer es correr el script. Ver Fig. 2.18.

SSI

<pre><html> <body > Ejemplo de inserción de datos <!--# exec cmd="/bin/date" --> //Inserción de comando </body> </html></pre>	<pre><html> <body> Ejemplo de inserción de datos Sun sep 2 16:20:12 gmt 2001 //Resultado </body> </html></pre>
---	--

Fig. 2.19 Funcionamiento de SSI. (Inserción de la fecha del sistema)

SSI puede considerarse como un lenguaje sencillo de directivas(definidas externamente) para inserción de datos en páginas web. El propósito principal es incrustar directivas dentro del HTML, que permitan desplegar el contenido de archivos de texto, información dinámica sobre archivos (como por ejemplo su tamaño) o la salida resultante de la ejecución de ciertos comandos del sistema.

CARACTERÍSTICAS DE SSI

- Permite construir documentos HTML a partir de otros documentos, insertar fechas u horas, fecha de última modificación, realizar acciones condicionales, ejecutar comandos del sistema.
- Las directivas SSI se incrustan en HTML exactamente en el mismo lugar donde se generará el resultado. En otras palabras, se reemplazan a sí mismas en el instante en que se sirven las páginas que las contienen. Ver Fig. 2.19.
- La extensión de una página que contiene directivas SSI es .shtml.
- La sintaxis general de una directiva SSI es: <!--# exec cmd="programa/comando"-->.
- SSI es un estándar que está orientado a servidores web que funcionen sobre UNIX.
- Los comandos más utilizados dentro de las directivas SSI son: **include**, **echo** y **exec**.
- Se compila por defecto y puede ser habilitado en los servidores web, salvo que explícitamente se indique lo contrario.

VENTAJAS Y DESVENTAJAS DE SSI

- Proporcionan una forma sencilla de crear páginas web dinámicas.
- Las directivas son sencillas de incrustar en *HTML*.
- Los SSI pueden contener a su vez otros SSI, de forma que se puede llegar a introducir una cantidad enorme de texto en una página con la inclusión de un único comando.
- SSI realmente está diseñado para inclusiones sencillas, no para programas "reales" que usen formularios, hagan conexiones a bases de datos, etc.
- Pueden impedir la portabilidad de las páginas web.
- Las directivas SSI pueden ejecutar varios comandos produciendo daños irreparables.
- Como gran desventaja está la dependencia de la plataforma: UNIX.
- Otra desventaja es también la dependencia del servidor web, son pocos los servidores web que soportan SSI.

FUNCIONAMIENTO DE SSI

En la figura 2.19, se muestra el hipertexto de una página web que contiene una directiva SSI que obtiene la fecha del sistema. Esta página es almacenada con la extensión *.shtml*. Cuando un cliente web solicita la página *.shtml*, el servidor web sabe por la extensión que tiene que explorar el hipertexto por la presencia de directivas SSI. El identificador de una directiva SSI es `<!--# -->`. Cuando encuentra una de estas directivas, el servidor ejecuta el programa o comando especificado entre `<!-- # -->` y el resultado se inserta en lugar de la directiva.

Las directivas son ejecutadas a medida que el servidor comprueba la sintaxis del *HTML*. El resultado de la ejecución es incrustado en *HTML* y luego será enviado al navegador para que éste se encargue de interpretarlo.

[WWW-27]

Nota: Cabe destacar que las herramientas que se describen a continuación: PHP, ASP, JSP y ColdFusion no son lenguajes de scripts en sí, mas bien son entornos para que se ejecuten scripts del servidor. Por ejemplo: PHP ejecuta scripts en PHP/PERL, ASP ejecuta scripts en VBScript/JScript/PerlScript, JSP ejecuta scripts en Java/SSJS, ColdFusion ejecuta scripts en CFML.

PHP

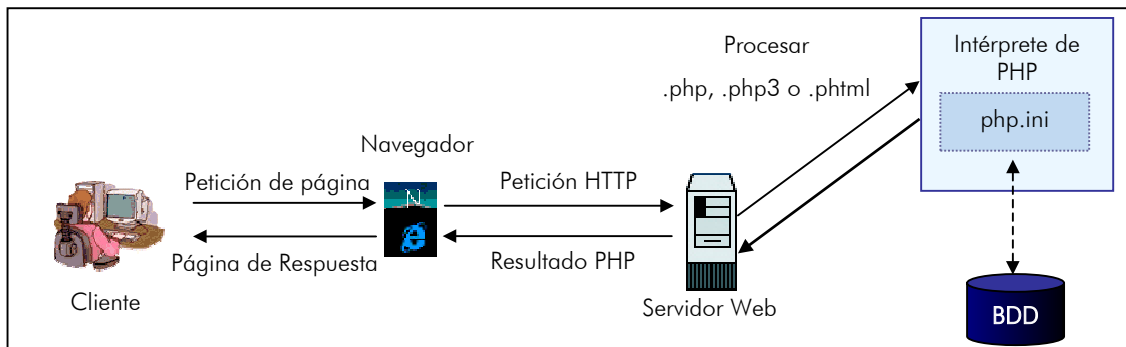


Fig. 2.20 Funcionamiento de PHP

PHP es un lenguaje de script interpretado e incrustado en páginas web, que es ejecutado del lado del servidor. Diseñado para facilitar el rápido desarrollo de sitios y aplicaciones web dinámicas. *PHP* es el lenguaje utilizado para crear páginas dinámicas que llevan la extensión **.php**, **.phtml**, **.php3**, dependiendo de la versión de *PHP* y la configuración del servidor web.

PHP fue concebido en 1994 por Rasmus Lerdorf. Las primeras versiones no distribuidas fueron usadas en sus páginas personales para mantener un control de visitas. Originalmente, *PHP* significaba **Personal Home Pages**. Consistía en un analizador sintáctico muy simple, con el que Lerdorf realizó un libro de visitas, un contador y otras pequeñas aplicaciones. El analizador sintáctico fue reescrito y nombrado *PHP/FI*, ya que se le añadió la característica de procesar datos de formularios.

PHP/FI creció a gran velocidad y la gente empezó a contribuir en el código. A finales de 1996 *PHP/FI* se estaba usando al menos en 15.000 páginas web en el mundo. En 1997 en más de 50.000; pero en el mismo año, el desarrollo de *PHP/FI* dio un profundo cambio, dejó de ser un proyecto personal y se convirtió en un proyecto de grupo más organizado, llamado **Professional Home Pages**. El analizador sintáctico fue reescrito desde el principio por Zeev Suraski y Andi Gutmans y nació *PHP* versión 3, siendo llamado finalmente **Preprocesador de Hipertexto PHP**(de ahí sus siglas).

Actualmente *PHP/FI*, *PHP 3*, *PHP 4* y *PHP 5*(beta) se distribuyen en un gran número de productos tales como el servidor web "C2's StrongHold" y Red Hat Linux. Estadísticas

realizadas por **NetCraft**⁷ estiman que más de 1´000.000 de servidores web y 1.4 millones de sitios usan páginas de tipo *PHP*.

CARACTERÍSTICAS DE PHP

- Es un lenguaje de programación del lado del servidor, gratuito y de código abierto.
- *PHP* puede ser instalado como *CGI* o como módulo del servidor web. Instalado como *CGI* puede correr sobre cualquier servidor que soporte *CGIs*. Instalado como módulo debe ser compilado para cada servidor web.
- La mayoría de su sintaxis se ha tomado de *C*, *C++*, *Java* y *PERL*. Los scripts son fáciles de programar, mantener y actualizar. *PHP* puede ser incrustado fácilmente en las páginas web, permitiendo la interacción entre diseñadores y programadores.
- Una página *PHP* puede contener código *PHP* y *HTML/XML/WML* o sólo código *PHP*.
- La extensión de una página *PHP* puede ser: **.php**, **.php3** o **.phtml** (de acuerdo a cómo esté configurado en el servidor web).
- Un script *PHP* puede ser incrustado en una página web, por medio de cualquiera de las *TAGS*(dependiendo de la configuración del archivo **php.ini**):
<? ?>
<?php ?>
<script language = "php"> </script>
<% %>
- Con *PHP* es muy sencillo tratar cadenas de texto o procesar archivos, crear documentos *PDF*, *XML*, *HTML*, generar imágenes *GIF*, *Flash*.
- Al nivel más básico, *PHP* puede hacer cualquier cosa que se pueda hacer con un script *CGI*, como procesar la información de formularios, generar páginas con contenidos dinámicos o mandar y recibir **cookies**.
- Quizá la característica más potente y destacable de *PHP* es su soporte para más de 16 motores de bases de datos: Adabas DB, Ingres, Oracle, dBase, InterBase, PostgreSQL, Empress, FrontBase, MS SQL Server, Solid, FilePro, mSQL, Sybase, IBM DB2, MySQL, Velocis, Informix, **ODBC**, Unix dbm.
- *PHP* también utiliza las sesiones de *HTTP*, conectividad de *Java*, expresiones regulares.

⁷ Netcraft.com, sitio del organismo dedicado a realizar estudios estadísticos sobre Servidores Web en Internet.

- Soporta protocolos como: *LDAP, SNMP, IMAP, CORBA, Calc, FTP, POP3, NNTP, HTTP* y derivados, protocolos de *COM* (sólo bajo Windows).
- Tiene funciones para comercio electrónico: *Cybercash, Verisign*.
- Maneja encriptación y compresión de datos con: *MD5, Gzip, Bzip2, OpenSSL*.
- A diferencia de *ASP* y *JSP*, las páginas *PHP* son generadas mediante el lenguaje de script del mismo nombre: *PHP*. Mientras que *ASP* utiliza *VBScript/JScript/PerlScript* y *JSP* utiliza *Java/SSJS*. Cabe destacar que en *PHP* también se puede utilizar *PERL*.

VENTAJAS Y DESVENTAJAS DE PHP

- Es un lenguaje creado específicamente para la web.
- *PHP* es multiplataforma, por ser código abierto, corre en “casi” cualquier plataforma, utilizando el mismo código fuente, puede ser instalado y ejecutado en algo así como 25 plataformas, incluyendo diferentes versiones de: *UNIX, Windows (9x,NT,ME,200x,XP), Mac, Solaris, Novell Netware*.
- Es también multiservidor, puesto que existe un módulo de *PHP* para casi cualquier servidor web, ha sido probado con los siguientes: *Internet Information Server 4.0, 5.0, 6.0, Personal Web Server 4.0, Apache Web Server⁸(en Linux y Windows), Omni HTTPD, Xitami Web Server, AOLServer, Zeus Web Server*, entre otros. También puede correr con cualquier servidor que soporte *CGI* e *ISAPI*. Pronto existirá un módulo para Enterprise de Netscape.
- Hoy *PHP* es uno de los más exitosos proyectos de código abierto existentes, con gran cantidad de programadores y colaboradores guiando su desarrollo, y la empresa Zend brindando el soporte que requiere su aplicación comercial y corporativa.
- Es rápido, sencillo, seguro y estable; cuenta con una gran librería de funciones y mucha documentación.
- Se puede realizar programación simple (scripts **.php**) o estructurada con clases (archivos **.inc**).
- Tiene soporte para conectarse a bases de datos con *ADO*, mediante *adodb.inc*.
- Una ventaja importante es que se puede portar un sitio o aplicación desarrollada en *PHP* de un sistema a otro casi sin ningún trabajo. Como en todos los sistemas se utiliza

⁸ Servidor web de preferencia para almacenar y servir páginas web de tipo *PHP*.

el mismo código base, los scripts pueden ser ejecutados de manera independiente al sistema operativo.

- *PHP* es completamente expandible y modular. Está compuesto de un sistema principal (escrito por Zend), un conjunto de módulos y una variedad de extensiones de código.
- *PHP* es utilizado en aplicaciones web por organizaciones importantes tales como: Mitsubishi, Red Hat, Der Spiegel, MP3-Lycos, Ericsson y NASA.
- Una desventaja de *PHP* es el manejo de errores, que no es tan sofisticado. Tampoco tiene un depurador incorporado por defecto.
- No cuenta con un entorno visual propio para programación de scripts, existen editores como el PHPEd sólo para Windows. En Linux, los scripts se desarrollan en editores de texto. Un editor recomendado es Dreamweaver, aunque también es sólo para Windows.

FUNCIONAMIENTO DE PHP

El navegador envía un pedido de un archivo con extensión **.php**(o sus variantes), el servidor web analiza que la extensión es **.php**, obtiene el archivo y lo envía al intérprete *PHP*. El intérprete *PHP* del servidor analiza el archivo en busca de *TAGS* que pueden aparecer varias veces y procesa todo lo que se encuentre entre dichas *TAGS*, accesa a bases de datos, realiza cálculos, etc.

Todo aquello que está fuera de las *TAGS* se envía al navegador sin interpretar. El resultado combinado de aquello que no debe interpretarse y el resultado del código interpretado por el servidor web se envía al navegador, dando como resultado una página web que interpretará por completo el navegador. Ver **Fig. 2.20**. De esta manera el cliente no puede tener acceso al código *PHP*, sólo al hipertexto.

Cabe destacar que para el adecuado funcionamiento de los scripts, el intérprete de *PHP* cuenta con un archivo denominado **php.ini**, la ejecución de los scripts, la comunicación con el servidor web y la conexión con la base de datos dependen de la correcta configuración de dicho archivo.

[WWW-28]

ASP

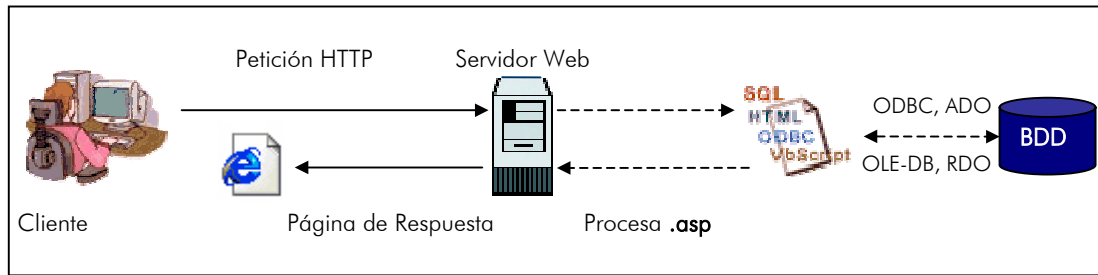


Fig. 2.21 Funcionamiento de ASP

ASP es un ambiente de aplicación, que presenta un entorno para crear páginas web dinámicas de tipo: **.asp**. Una página ASP puede contener scripts en *VBScript* o *JScript* y componentes *ActiveX* incrustados dentro de una página web.

ASP es una herramienta de generación de contenido dinámico del lado del servidor desarrollada por Microsoft e introducida en diciembre de 1996. Es distribuida en conjunto y como parte del Internet Information Server, desde la versión 3.0, que es el principal servidor web que soporta las páginas de tipo ASP. [WWW-29]

Los scripts de las páginas ASP se escriben dentro de la misma página web o aparte, utilizando *VBScript* o *JScript*, que son los lenguajes que vienen de forma nativa con ASP. La elección del lenguaje depende de la configuración en el servidor web. Si se desea utilizar otro lenguaje, por ejemplo *PerlScript*, se debe instalar el intérprete correspondiente.

CARACTERÍSTICAS DE ASP

- No debemos olvidar que ASP sólo nos ofrece un entorno para que se ejecute el script de servidor, no es un lenguaje de script propiamente dicho, porque hace uso de *VBScript*, *JScript* o *PerlScript* (sólo en entornos Unix/Linux, con el software APACHE::ASP).
- Tanto *VBScript* y *JScript* pueden ser utilizados del lado del servidor y del cliente, aunque del lado del cliente la sintaxis presenta algunos cambios.
- Todo el entorno ASP no es más que una *DLL* del tipo *ISAPI*, que es invocada automáticamente para los archivos cuya extensión sea **.asp**.
- Debido a que las páginas ASP son programadas mayormente con *VBScript* y *JScript* de Microsoft, son dependientes del servidor web. Actualmente los servidores web que

soportan páginas ASP son: Internet Information Server para Windows NT, 200x, XP; y Personal Web Server para Windows 9x. Mediante el software **ChiliSoft ASP** puede ser soportado por Apache Web Server(en Linux y Windows), se están haciendo adaptaciones para ser soportado por Zeus Web Server.

- Las plataformas que permiten el desarrollo y ejecución de estas páginas son obviamente todas las de Microsoft, aunque actualmente Linux y Novell Netware también las soportan por medio de la instalación de un software adicional llamado **ChiliSoft ASP** e **InstantASP** respectivamente. También existe APACHE::ASP para Linux.
- Con ASP se puede tener acceso a bases de datos por medio de ODBC, al sistema de archivos del servidor y en general a todos los recursos que tenga el propio servidor.
- Los scripts para programar páginas ASP son incrustados en las páginas web por medio de las TAGS:

```
<%@ language="VBScript" RUNAT=Server %>
```

```
<%@ language="JScript" RUNAT=Server %>
```

```
<% %>
```

- Se puede programar scripts ASP en un archivo aparte y llamarlos dentro de una página web por medio de:

```
<!--#Include File="archivo.asp" -->
```

- También existe la posibilidad de utilizar componentes ActiveX fabricados por distintas empresas para envío de correo, generación de gráficos dinámicos, etc.
- Una página ASP puede consistir sólo en código script. La mezcla de HTML/XML/WML y código script es opcional.
- Actualmente se ha presentado ya la segunda versión variante de ASP, el ASP.NET, que comprende algunas mejoras, sintaxis distinta y mayor orientación a Internet. Los lenguajes para programar en ASP.NET son: C#, VB.NET.

VENTAJAS Y DESVENTAJAS DE ASP

- Al ser una herramienta dependiente del servidor web, la principal ventaja radica en la seguridad que tiene el programador sobre su código.
- Permite elegir el lenguaje de script de entre los tres posibles, y puede combinar *VBScript* y *JScript* a lo largo de la página, aunque es recomendable usar un solo lenguaje.
- Para la conexión a bases de datos, se conecta a cualquiera cuyo controlador sea

ODBC. Entre éstas bases de datos están: Access, Microsoft SQL Server, Oracle, Sybase, Informix, DB2, entre otras. Además permite conexiones con: **ADO**, **OLE-DB**, **RDO**, *ActiveX*.

- Existen en el mercado tres herramientas para desarrollar profesionalmente páginas *ASP* en modo visual: Drumbeat 2000, Visual Interdev de Microsoft y Dreamweaver.
- Se pueden utilizar componentes de software reutilizables: *COM* y *ActiveX*, que se incluyen directamente en la instalación.
- La diferencia entre *ASP* y *ASP.NET* está bien marcada, haciéndose complicado pasar los scripts de una versión a otra.
- No soporta protocolos directamente, sino por medio de *COM*.
- Una gran desventaja está en la limitación de plataformas, necesita de software de terceros para trabajar en Linux y Novell Netware; para trabajar con otros servidores web, por ejemplo para el servidor Netscape necesita un plug-in denominado **ScriptActive**.
- Al usar *Chili!Soft ASP* en Linux o *InstantASP* en Novell Netware el costo es muy elevado, sin embargo existe una versión gratuita de *Chili!Soft ASP*, pero con características reducidas.

FUNCIONAMIENTO DE ASP

Desde el navegador del usuario se solicita el archivo **.asp** al servidor web, el script (escrito en *VBScript*, *JScript* o *PerlScript*) del lado del servidor comienza a ejecutarse y produce el archivo de salida en formato de página web, el cual es enviado al navegador por medio del servidor web. En realidad, no existe gran diferencia entre el funcionamiento de las páginas *ASP* y sus similares: *PHP*, *JSP* y *CFML*. Ver **Fig. 2.21**.

Cabe destacar que *ASP* no interpreta directamente el código, sino que en función del lenguaje en el que está escrito, invoca a otra *DLL* que se encarga de ejecutar el script. El código *VBScript*, *JScript* o *PerlScript* nunca llega al navegador, así se protege el código de cambios o accesos no deseados.

JSP Y JAVA SERVLETS

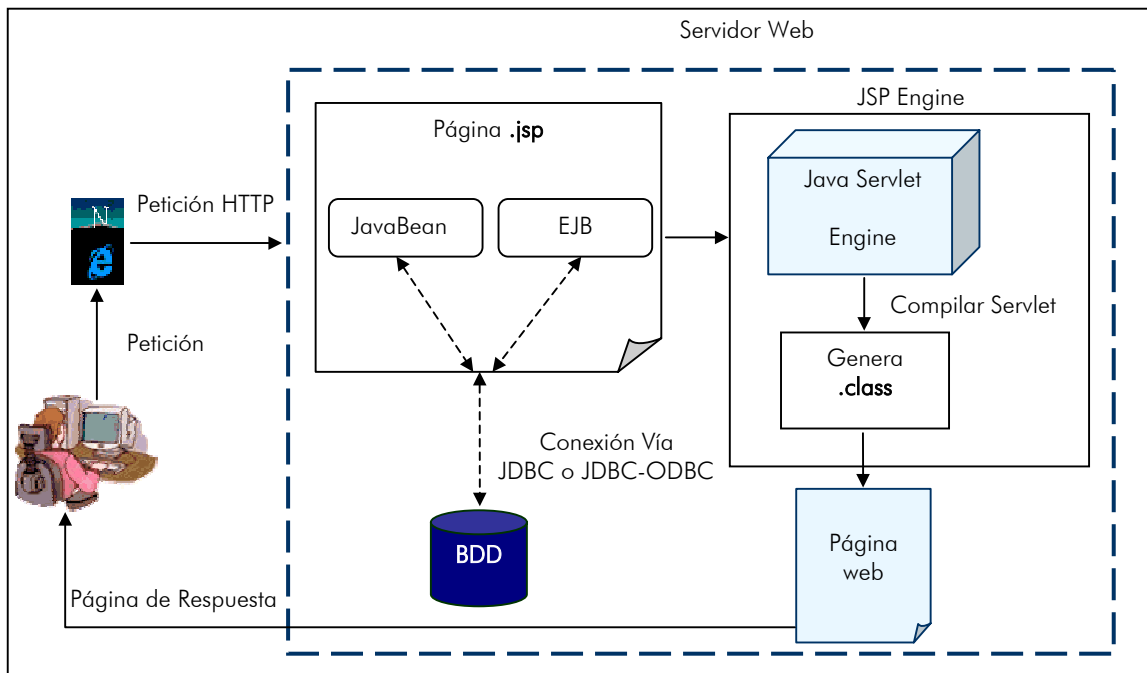


Fig. 2.22 Funcionamiento de JSP

Las *JSP* están basadas en la tecnología *Java Servlets*. Por lo tanto es necesario hacer una breve revisión de los *Servlets*, antes de pasar al estudio de *JSP*.

Los *Servlets* son la respuesta de la tecnología *Java* a la programación *CGI*. Son programas que se ejecutan en un servidor web y construyen páginas web dinámicamente. [WWW-30]

Los *Servlets* son un reemplazo efectivo para los scripts *CGI*, sirviendo como mecanismo para invocar a un programa *Java*. Proporcionan una forma de generar páginas web dinámicas que son rápidas en ejecutarse. Los *Servlets* también solucionan el problema de hacer la programación del lado del servidor con *APIs* específicos de la plataforma. Están desarrollados con el *API Java Servlet*, una extensión estándar y multiplataforma de *Java*.

En sí, los *Servlets* son clases (archivos *.class*) desarrolladas 100% en *Java*, que pueden ser dinámicamente cargadas en un servidor para expandir la funcionalidad de éste.

Un *Servlet* es creado e inicializado, luego se procesan las peticiones recibidas y por último se destruye. Este diseño explica por que un *Servlet* reemplaza perfectamente a un *CGI*, ya

que el *Servlet* es cargado una sola vez y está residente en memoria mientras se procesan las peticiones recibidas y se generan las respuestas a los usuarios. Además, con los *Servlets* se pueden realizar operaciones que no son posibles con *CGI*. Se puede traducir rutas de archivos, realizar accesos, verificar autorizaciones, realizar mapeo de tipos *MIME* y hasta agregar usuarios a la base de datos del servidor de una manera sencilla, elegante y segura. [WWW-31]

Sin embargo, los *Servlets* poseen una desventaja importante que consiste en que complican la generación del código *HTML/XML/WML* para la página de respuesta. El *HTML/XML/WML* es escrito dentro de sentencias *println* de *Java*, presentando gran dificultad a los diseñadores y programadores. Ver **Fig. 2.23**.

```
import java.io.*; //Importar paquete de entrada/salida
import java.servlet.*; //Importar paquetes de Servlets
import java.servlet.http.*; //Importar paquetes HTTP
public class HelloWWW extends HttpServlet { //Iniciar clase para Servlet
public void doGet(HttpServletRequest request, HttpServletResponse response) //Iniciar variables GET
throws ServletException, Exception { //Capturar excepciones
response.setContentType("text/HTML"); //Definir tipo de contenido: HTML
PrintWriter out = response.getWriter(); //Imprimir HTML
out.println("<!DOCTYPE HTML PUBLIC(\n- // W3C // DTD HTML 4.0 ">" + \n" +
"<HTML>\n" + "<HEAD><TITLE>Ejemplo de Servlet</TITLE></HEAD>\n" + "<BODY>\n" +
"<H1>Hola Mundo</H1>\n" + "</BODY></HTML>");
} //Fin de capturar excepciones
} //Fin de clase Servlet
```

Fig. 2.23 Servlet para generar una página HTML de respuesta

Es mucho más conveniente escribir y modificar *HTML/XML/WML* normal que tener que hacer miles de sentencias *println* que generen *HTML/XML/WML*. Con la idea de mejorar este inconveniente surgió *JSP*.

JSP es una herramienta que fue creada como una extensión de los *Servlets* de *Java* para solucionar las desventajas de éstos últimos. Las páginas *JSP* están ideadas para facilitar la creación de aplicaciones web gracias a sus características mejoradas. *JSP* proporciona una capacidad tan poderosa como los *Servlets* y potencialmente mucho más fácil de desarrollar y usar, ya que en una página *JSP* se puede separar la parte de la presentación y la parte de la programación de forma similar a como lo hacen *PHP* y *ASP*. En el caso de las páginas *.jsp*, el código que se incrusta en la página web es *Java* o *SSJS*.

Las *JSP* y los *Servlets* son alternativas semejantes entre sí, sus características y

funcionamiento son similares. Cabe destacar que *JSP* cuenta con ventajas respecto a los *Servlets*, por esta razón las ventajas y desventajas de *JSP* se detallarán, más adelante, respecto a otras herramientas(*PHP, ASP, CFML*), no respecto a *Servlets*.

CARACTERÍSTICAS DE JSP Y SERVLETS

- *JSP* es una herramienta que surgió tras los inconvenientes de los *Servlets*, por lo tanto *JSP* se puede definir como un tipo especial de *Servlet*. Sin embargo los *Servlets* no tienen restricciones de ejecución.
- Tanto *JSP* como *Servlets*, son herramientas creadas por Sun Microsystems con la filosofía propia de *Java*: “Escríballo una vez, Ejecútelo donde quiera”. Esto significa que las *JSP* y los *Servlets* son traducidos a bytecodes, y esto les hace independientes de la plataforma: Windows, Mac, Linux, Unix.
- Debido a que ambas herramientas utilizan código *Java*, los *Servlets* son programas 100% *Java* que contienen plantillas *HTML/XML/WML* dentro de las sentencias *println*, ver **Fig. 2.23**; las *JSP* son páginas que contienen scripts *Java* o *SSJS* incrustados en *HTML/XML/WML*, ver **Fig. 2.24**. Cabe destacar que los *Servlets* también pueden ser escritos en *Python*.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0">
<HTML>
<HEAD><TITLE>Ejemplo de Página JSP</TITLE>
</HEAD>
<BODY>
  <% @ page language = "java" %>
  <% System.out.println("<H1>Hola Mundo</H1>"); %>
</BODY>
</HTML>
```

//Iniciar página HTML
//Cabecera de la página
//Cuerpo de la página
//Incrustar Java en HTML
//Imprimir HTML
//Fin del cuerpo de la página
//Fin de HTML

Fig. 2.24 Script JSP incrustado en HTML

- Las *JSP* y los *Servlets* son soportados por una gran variedad de servidores web, mediante el uso de módulos o plug-ins, que cuenten con las especificaciones *JSP 1.0* ó *1.1* y *Servlets 2.1* ó *2.2* respectivamente: **Tomcat Web Server**⁹, **Java Web Server**¹⁰, Netscape Enterprise, Internet Information Server, Personal Web Server, viejas versiones de Apache Web Server, New Atlanta's ServletExec, Gefion's LiteWebServer, Mort Bay

⁹ Módulo de Apache Web Server para soportar JSP y Servlets. Tomcat es parte del proyecto Jakarta-Apache.

¹⁰ Primer servidor web desarrollado por Sun para soportar JSP y Servlets. Totalmente desarrollado en Java.

Jetty, entre otros.

- En *JSP* es posible la separación entre generación de contenido y presentación, facilitando el trabajo a diseñadores y programadores, al contrario de *Servlets*.
- *JSP* permite reutilización de componentes de software, llamados **JavaBeans** y **EJB**, *Servlets* y *Applets*. Los *Servlets* no cuentan con esta característica.
- Ambas herramientas se conectan con cualquier base de datos que soporte **JDBC** o el puente *JDBC-ODBC*.
- A las *JSP* les falta un poco de estructura, mientras que los *Servlets* tienen la estructura de un programa en *Java*.
- Los scripts *JSP* son incrustados mediante las *TAGS*(etiquetas):

```
<%@page language = "java" %>  
<% %>
```
- También los scripts *JSP* se pueden programar en archivos aparte y ser llamados desde una página web por medio de:

```
<%@ include file = "archivo.jsp" %>
```
- Las *JSP* además de generar *HTML*, pueden generar cualquier salida con formato de texto, como por ejemplo *XML*, incluso *WML*.
- En las *JSP* se puede utilizar *TAGS* creadas por el usuario y almacenarlas en archivos.
- Los archivos *JSP* tienen extensión **.jsp**, mientras que los *Servlets* **.class**.
- A diferencia de las *JSP*, los *Servlets* sirven mucho mejor para manejar dinámicamente datos binarios(modificar archivos, crear imágenes dinámicas.)
- *JSP* y *Servlets* se pueden editar en un Bloc de Notas o en editores más avanzados como Dreamweaver y Forte.

VENTAJAS Y DESVENTAJAS DE JSP

- Debido a que *JSP* tiene la característica de traducirse a *Servlets*, es una gran ventaja respecto a otras herramientas, ya que así aumenta su desempeño al combinarse con el poder propio de los *Servlets*. Así las ventajas de *Servlets*, pueden ser tomadas como ventajas de *JSP*.
- Cuenta con el amplio respaldo de la empresa Sun Microsystems, que ha suministrado bibliotecas de *Servlets* que pueden ser utilizados en las *JSP* y que facilitan el desarrollo de aplicaciones web. Estas bibliotecas permiten extraer valores de encabezados *HTTP*,

analizar los certificados que se utilizan para garantizar la seguridad de las conexiones por capas de sockets y procesar inclusiones del lado del servidor.

- Es conveniente resaltar, que la herramienta *JSP* es un componente clave de *J2EE*.
- Además de poseer sus propias características, cuenta con las características de seguridad, robustez y portabilidad propias del lenguaje *Java*.
- En una página *JSP* se puede incluir clases *Java*, *Applets* y *Servlets* para comunicarse con éstos.
- También se puede utilizar *SSJS(JavaScript)* del lado del servidor) para programar los scripts de una página *JSP*.
- Es fácil modificar la apariencia de una página *JSP* sin tener que modificar ningún código *Java* ni en la página ni en los componentes(si se usan).
- Como desventaja se puede anotar que las *JSP* son soportadas sólo por servidores web que cuenten con las especificaciones de *JSP* y el motor de *Servlets*.
- Hay que conocer a fondo la estructura de las clases de *Java* para utilizarlas en las páginas *JSP*.

FUNCIONAMIENTO DE JSP Y SERVLETS

En realidad, *JSP* no es más que una herramienta para escribir *Servlets* de una manera más sencilla, de modo que existe un momento en que su funcionamiento es igual, ya que las *JSP* son traducidas a *Servlets* por el motor(llamado Java Servlet Engine) del servidor web antes de ser ejecutadas.

Cuando se produce una solicitud de una página **.jsp** por parte de un navegador, se comienza comprobando si se trata de la primera solicitud de dicha página. Luego, básicamente, el motor del servidor web compila la página **.jsp** que hayamos solicitado y la convierte en un *Servlet(.class)* que es cargado a memoria, dicho *Servlet* es el que generará la respuesta al cliente. El proceso de compilación se realiza la primera vez y siempre que el *Servlet* asociado a la página no exista o cuando el fichero **.jsp** ha sido modificado. Ver **Fig. 2.25**.

El proceso de compilación de las páginas consiste en analizar su contenido buscando etiquetas *JSP* y traduciendo éstas a código *Java* equivalente. El contenido estático de las

páginas web es traducido a cadenas de caracteres en lenguaje *Java*. Las etiquetas de componentes *JavaBeans* son traducidas a su correspondiente objeto, mientras que los elementos script son transferidos tal cual. En caso de existir accesos a bases de datos, éstos se realizan, ya sea traduciendo los scripts, *JavaBeans* o *EJB*, en caso de que los haya.

Las siguientes solicitudes son más eficientes al no necesitar que dicha página sea de nuevo compilada, ya que sólo se invoca al *Servlet* que se generó en la petición inicial. En este sentido ésta herramienta es más ventajosa que otras como *ASP*, que requieren cada vez una nueva compilación.

Así, la implementación de *JSP* involucra, necesariamente, una fase de traducción (ejecutada una sola vez) y una fase de procesamiento de solicitud (ejecutada por cada solicitud).

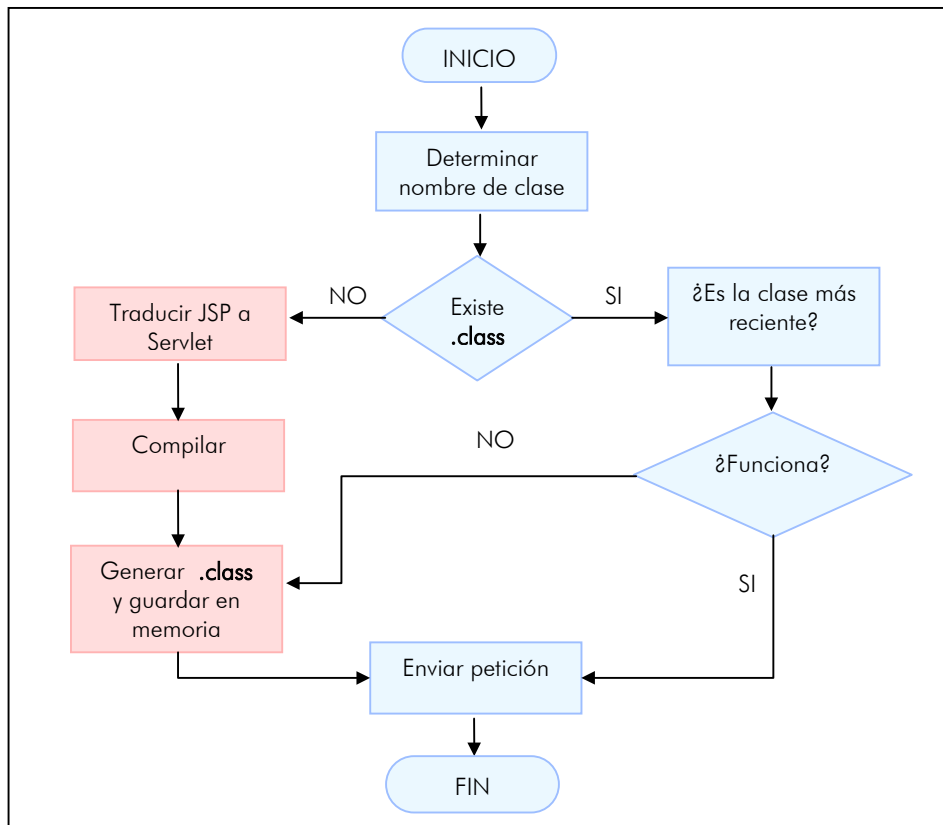


Fig. 2.25 Proceso de compilación de una página JSP

COLDFUSION (CFML)

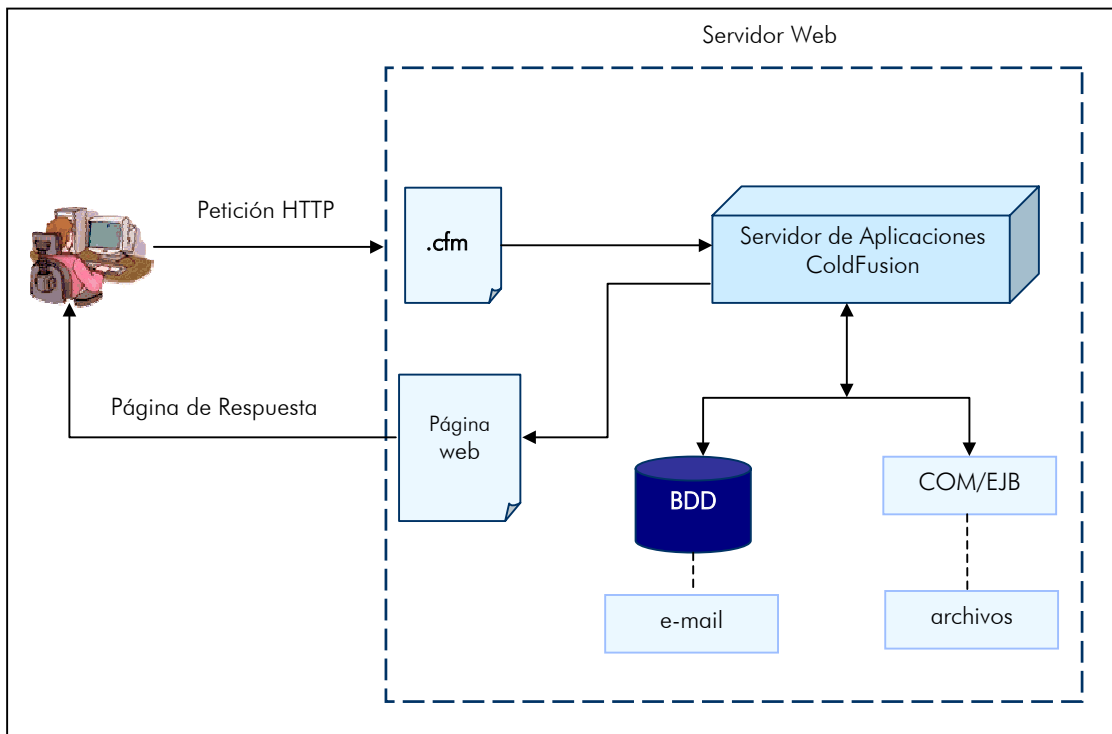


Fig. 2.26 Funcionamiento de ColdFusion

ColdFusion es una extraordinaria herramienta del lado del servidor para desarrollo de sitios y aplicaciones web, diseñada por la Corporación Allaire e introducida al mercado desde 1995.

ColdFusion difiere de otras herramientas similares debido a que integra el diseño de páginas, la administración del sitio, la seguridad, procedimientos de integración con bases de datos, y todos los recursos necesarios en un único sistema de desarrollo de aplicaciones web. Aplica las tecnologías más nuevas como son: *DHTML*, *XML*, Bases de Datos Relacionales, *e-mail*, *ActiveX*, *Java*, etc.

Además soporta un poderoso lenguaje de scripts del lado del servidor: **CFML**. *CFML* es un lenguaje 100% basado en *TAGS*, similar en simplicidad al *HTML*, razón por la cual es muy fácil de aprender y usar. *CFML* es muy flexible ya que se integra de manera fácil y transparente con todos los lenguajes y tecnologías web populares, para crear páginas de tipo ColdFusion (*.cfm*).

CARACTERÍSTICAS DE COLDFUSION

- ColdFusion es más que una herramienta de generación de contenido dinámico del lado del servidor, es un completo **servidor de aplicaciones web** que permite administración, monitoreo, diseño, desarrollo, depuración, manejo de seguridad, etc. Este servidor de aplicaciones se denomina ColdFusion Server y además cuenta con un ambiente visual de desarrollo llamado UltraDev Studio.
- Es una herramienta multiplataforma y multiservidor web. Tanto la edición profesional y empresarial de ColdFusion 4.0 son soportadas por Apache y los servidores web que utilizan *NSAPI*, *ISAPI* y *CGI*. ColdFusion es soportado en plataformas Windows NT, 200x, Linux, Solaris.
- Las *TAGS CFML* encapsulan complejos procedimientos de servidor como interacción con bases de datos o envío de e-mail, así que no se tiene necesariamente que escribir código complicado, basta con abrir y cerrar una etiqueta sencilla.
- Un script *CFML* puede ser incrustado en una página web a través de las *TAGS* similares a las de *HTML*, por ejemplo, si se desea declarar una variable o imprimir un valor, se usa respectivamente:

```
<cfset mi_variable = expresión>           //Declarar una variable  
<cfoutput>#mi_variable#</cfoutput>      //Imprimir el valor de una variable
```
- Las páginas de tipo ColdFusion se almacenan con la extensión **.cfm**.
- *CFML* soporta más de 70 *TAGS*, 200 funciones y 800 componentes de terceras partes, haciendo que sea uno de los mejores lenguajes disponibles para crear aplicaciones web avanzadas. [WWW-32]
- Al utilizar cualquier *ODBC* comercial, se puede acceder a prácticamente cualquier base de datos: Sybase, Oracle, MySQL, PostgreSQL, SQL Server, Access. Usando **SQL** estándar, las páginas y aplicaciones web pueden fácilmente recuperar, guardar, formatear y presentar información dinámicamente.
- Permite la utilización de formularios dinámicos *Java*. Se pueden crear fácilmente formularios que usan *JavaScript* y componentes *Java* utilizando la sintaxis familiar de los formularios *HTML*. Estos formularios proporcionan la mejor manera de crear avanzadas interfaces con una amplia gama de nuevos controles *Java* que superan a los formularios *HTML* comunes.
- ColdFusion puede utilizar cualquier componente disponible para *ASP*, incluyendo

ActiveX, ADO; componentes COM, CORBA, JavaBeans, escritos en Visual Basic, Delphi, C/C++ o Java, respectivamente.

- Soporta indexación transparente de datos de texto almacenados en bases de datos relacionales. Permite indexar fácilmente y buscar muchos tipos de contenido, incluyendo páginas web y documentos Microsoft Office 2000.
- Permite encriptar todas las páginas de una aplicación, para proteger el código.
- Soporta todos los servidores *POP*, deja correo en el servidor, recupera y entrega mensajes y anexos.
- Soporta autenticación estándar de servidores web, encriptación *SSL* y certificados.
- Tiene soporte para *LDAP, SMTP, POP 3, FTP, HTTP, Telnet, NNTP*.
- Se pueden crear *TAGS* personalizadas para encapsular complejas rutinas creadas con *HTML* y otras herramientas del lado del cliente. También tiene *TAGS* para *COM, CORBA, Applets* y *Servlets*.

VENTAJAS Y DESVENTAJAS DE COLDFUSION

- ColdFusion usa menos líneas de código y son más intuitivas, el código se encapsula en etiquetas similares a las de *HTML*.
- ColdFusion provee un entorno de desarrollo muy fácil de usar.
- Las aplicaciones en ColdFusion pueden migrar a otras plataformas y motores de bases de datos, con pequeños cambios y plug-ins externos, y se puede incluso cambiar los motores de bases de datos con un poco más de esfuerzo, pero sin tener que empezar de nuevo.
- Los componentes ColdFusion pueden ser desarrollados por programadores que no tengan gran experiencia, si han programado en *JavaScript, PHP* y otros lenguajes similares, estarán listos para hacerlo en *CFML*.
- En el corazón de cada aplicación de ColdFusion hay un servidor ColdFusion, el cual combina una arquitectura abierta y extensible que se integra fácilmente con sistemas existentes.
- Se puede reutilizar las bibliotecas de etiquetas *JSP* como etiquetas personalizadas de ColdFusion.
- ColdFusion está diseñado para correr en máquinas multi-procesador, y permite

construir sitios que pueden correr en un **cluster**¹¹ de servidores.

- Permite continuar usando scripts *CGI* existentes.
- Permitir convertir datos desde y hacia *XML*.
- ColdFusion se combina muy bien con las herramientas de la empresa Macromedia, tales como Flash Player y Dreamweaver.
- La gran desventaja de ColdFusion es su costo, que además va en aumento según la edición que se elija, al momento la edición Profesional cuesta \$1295, mientras que la Empresarial cuesta \$3495.

FUNCIONAMIENTO DE COLDFUSION

Cada vez que se solicita una página de tipo *.cfm*, el servidor web pasa la petición de la página al servidor de aplicaciones ColdFusion, a través de una *API*. El servidor de aplicaciones ejecuta el script o programa contenido en dicha página. Ver **Fig. 2.26**.

Las páginas ColdFusion se compilan automáticamente en el servidor con un optimizador llamado **p-coder just-in-time**. Esta nueva característica incrementa significativamente la ejecución y escalabilidad para páginas complejas y aplicaciones a través de múltiples peticiones. [WWW-33]

2.2.2.2.1 COMPARACIÓN DE LAS HERRAMIENTAS: PHP, ASP Y JSP

Una vez realizado el estudio de las diferentes herramientas de generación de contenido dinámico del lado del servidor, se han determinado algunos puntos que permitan realizar el análisis comparativo de tres herramientas de este tipo.

Las herramientas a ser comparadas son: *PHP*, *ASP* y *JSP*, debido a que éstas tienen características, comportamiento y funcionalidad similares. Se ha dejado de lado a otras herramientas como *CGI*, *PERL*, *SSI* y ColdFusion, ya que no existe punto de comparación entre éstas y las escogidas. *CGI* y *PERL* tienden a ser herramientas casi ya sin uso, mientras que *SSI* no es una herramienta para aplicaciones grandes y ColdFusion es un entorno más sofisticado y costoso.

¹¹ Un cluster está formado por varias computadoras que al unirse para formar un servidor se tratan como una.

Al comparar *PHP*, *ASP* y *JSP*, se comparará el comportamiento, ventajas y desventajas de una herramienta gratuita, una comercial y una compartida respectivamente.

Cabe destacar que en esta comparación también se toma en cuenta las herramientas complementarias que son el Servidor Web y la Base de Datos que trabajan en conjunto para formar la tecnología completa para el desarrollo de aplicaciones web.

El principal objetivo de esta comparación es llegar a determinar cual herramienta de generación de contenido dinámico del lado del servidor(*PHP*, *ASP*, *JSP*) junto con su respectivo Servidor Web y Base de Datos son más convenientes escoger para ser utilizados a la hora de programar aplicaciones web y por lo tanto sitios y portales en Internet.

Más adelante, en el presente capítulo, en las tablas **2.1**, **2.2**, **2.3** y **2.4**, se compara en detalle: aspectos generales, sintaxis del lenguaje, conexión a la base de datos y principales funciones de: *PHP*, *ASP* y *JSP*.

Como complemento, en el **Anexo 4, Tabla 4** que se encuentra en el CD adjunto a este documento en: CD_TESIS:\Tesis\Anexos_Tesis\Anexos.pdf, se realiza la comparación y evaluación de servidores web que trabajan con: *PHP*, *ASP* y *JSP*.

Luego de realizar dichas comparaciones se plantea alternativas que se clasifican en herramientas de código abierto y propietario, bajo ciertos parámetros de comparación. En el capítulo 5, tablas **5.2** y **5.3** se presentan las alternativas mencionadas para el desarrollo de aplicaciones web.

Tabla 2.1 Análisis Comparativo de Herramientas de generación de contenido dinámico del lado del servidor (PHP, ASP y JSP)

	PHP	ASP	JSP	RESULTADO		
				PHP	ASP	JSP
HERRAMIENTA CREADA POR	<ul style="list-style-type: none"> ♦ Rasmus Lerdorf ♦ Comunidad Internet ♦ Empresa Zend 	<ul style="list-style-type: none"> ♦ Microsoft Corporation 	<ul style="list-style-type: none"> ♦ Sun Microsystems 	3	1	2
TIPO DE LICENCIA	<ul style="list-style-type: none"> ♦ Código abierto y Gratuito ♦ Comercial 	<ul style="list-style-type: none"> ♦ Comercial 	<ul style="list-style-type: none"> ♦ Código protegido y Gratuito ♦ Comercial 	3	1	2
EMPRESA DISTRIBUIDORA	Zend	Microsoft Corporation	Sun Microsystems	3	3	3
FORMAS DE DISTRIBUCIÓN	<ul style="list-style-type: none"> ♦ Gratuita(Código abierto) ♦ Comercial 	<ul style="list-style-type: none"> ♦ Comercial 	<ul style="list-style-type: none"> ♦ Gratuita(Código protegido) ♦ Comercial 	3	1	2
MEDIOS DE DISTRIBUCIÓN	<ul style="list-style-type: none"> ♦ Internet ♦ Con RPM (para Linux) ♦ Con StrongHold Server ♦ Con herramientas derivadas 	<ul style="list-style-type: none"> ♦ Con Internet Information Server ♦ Con Visual Studio ♦ Con Windows 200x 	<ul style="list-style-type: none"> ♦ Internet ♦ Con JSWDK ♦ Con J2EE ♦ Con JDK 	3	2	3
LA DISTRIBUCIÓN CUENTA CON	<ul style="list-style-type: none"> ♦ Código Fuente abierto ♦ CD de RPMs o herramientas derivadas 	<ul style="list-style-type: none"> ♦ CD 	<ul style="list-style-type: none"> ♦ Código Fuente protegido ♦ CD de JSWDK/J2EE/JDK 	3	2	2
DISTRIBUCIÓN DOCUMENTADA	Si	Si	Si	3	3	3
CÓDIGO FUENTE	Abierto (Open Source)	Protegido	Protegido, pero gratuito (Shared o Compartido)	3	1	2
SITIO WEB OFICIAL	www.php.net	www.microsoft.com	java.sun.com	3	3	3

EL SITIO WEB SE RENUEVA	Siempre mantiene las últimas versiones del software	Si	Si	3	3	3
VERSIONES	<ul style="list-style-type: none"> ♦ PHP ♦ PHP/FI 1.# ♦ PHP/FI 2.# ♦ PHP3.#.# ♦ PHP4.#.# ♦ PHP5.#.# (Beta) 	<ul style="list-style-type: none"> ♦ ASP 1.0 ♦ ASP 2.0 ♦ ASP 3.0 ♦ ASP.NET 	<ul style="list-style-type: none"> ♦ JSP 1.0 ♦ JSP 1.1 ♦ JSP 1.2 	3	2	2
VERSIONES ESTABLES Y COMPATIBLES	Si	No	No	3	1	1
SITIOS WEB QUE COLABORAN CON LA HERRAMIENTA	<ul style="list-style-type: none"> ♦ www.phpes.com ♦ www.phpnuke.org ♦ www.phpbuilder.com ♦ www.zend.com ♦ www.mundophp.com 	<ul style="list-style-type: none"> ♦ www.asp.net ♦ www.apache-asp.org ♦ www.activescripting.org ♦ www.chilisoft.com ♦ www.halcyonsoft.com 	<ul style="list-style-type: none"> ♦ www.jspin.com ♦ www.javahispano.com 	3	3	2
AÑO DE LANZAMIENTO	1994	1996	1995	3	1	2
SIGLAS	PHP PHP Hypertext Preprocesor Preprocesador de Hipertexto PHP	ASP Active Server Page Página Activa de Servidor	JSP Java Server Page Página Java de Servidor	3	3	3
EXTENSIÓN DE PÁGINA DE SERVIDOR	.phtml, php, php3, php4	.asp	.jsp	3	1	1

<p>HERRAMIENTAS DERIVADAS O COMPLEMENTARIAS</p>	<ul style="list-style-type: none"> ♦ AbriaSoft ♦ NuSphere SQL ♦ GreatBridge SQL ♦ LAMPARA, WAMP ♦ phpMyAdmin ♦ PHP-Nuke, Post Nuke ♦ PHPWebLog ♦ PHPLiB ♦ PHP-Easy-Installer (Windows) ♦ PHPTriad (Windows) 	<ul style="list-style-type: none"> ♦ InstantASP (Novell Netware) ♦ ChiliSoft ASP (Windows y Linux) ♦ APACHE::ASP (Linux) ♦ OpenASP 	<ul style="list-style-type: none"> ♦ J2EE ♦ Java Server Faces ♦ Java Web Services 	<p>3</p>	<p>2</p>	<p>2</p>
<p>PLATAFORMAS SOPORTADAS</p>	<ul style="list-style-type: none"> ♦ Windows 9x, NT, 200x, ME, XP. ♦ Linux ♦ Unix/HP-UX ♦ Solaris ♦ Mac ♦ Novell Netware ♦ OpenBSD 	<ul style="list-style-type: none"> ♦ Windows 9x, NT, 200x, ME, XP. ♦ Linux (con ChiliSoft ASP) ♦ Novell Netware (con InstantASP) 	<ul style="list-style-type: none"> ♦ Windows 9x, NT, 200x, ME, XP. ♦ Linux ♦ Unix ♦ Mac ♦ Solaris 	<p>3</p>	<p>2</p>	<p>3</p>
<p>SERVIDORES WEB SOPORTADOS</p>	<ul style="list-style-type: none"> ♦ Apache Web Server ♦ IIS 4.0, 5.0,6.0 ♦ Personal Web Server ♦ Omni HTTPD, ♦ Xitami Web Server ♦ AOLServer ♦ Zeus Web Server ♦ Roxen Web Server ♦ Caudium ♦ Sambar ♦ Nuevas versiones de Netscape ♦ Servidores con soporte: <ul style="list-style-type: none"> ♦ CGI ♦ ISAPI, NSAPI 	<ul style="list-style-type: none"> ♦ IIS 4.0, 5.0, 6.0 ♦ Personal Web Server ♦ O'Reilly WebSite Pro ♦ Apache Web Server (con ChiliSoft ASP en Linux) ♦ iPlanet (con ChiliSoft ASP en Windows) 	<ul style="list-style-type: none"> ♦ Tomcat Web Server ♦ Apache Web Server ♦ Sun Java Web Server ♦ Netscape Enterprise ♦ IIS 4.0, 5.0, 6.0 ♦ Personal Web Server ♦ New Atlanta Servlet Exec ♦ Gefion LiteWebServer ♦ Resin ♦ BEA Weblogic ♦ Macromedia JRun 	<p>3</p>	<p>1</p>	<p>3</p>

PROTOCOLOS SOPORTADOS	<ul style="list-style-type: none"> ♦ LDAP ♦ SNMP ♦ IMAP ♦ CORBA ♦ Calc ♦ FTP ♦ POP3 ♦ NNTP ♦ NIS ♦ HTTP y derivados ♦ COM (sólo bajo Win) 	<ul style="list-style-type: none"> ♦ Protocolos soportados a través de COM 	<ul style="list-style-type: none"> ♦ SMTP ♦ POP ♦ IMAP 	3	1	1
CONECTIVIDAD A BASES DE DATOS	<ul style="list-style-type: none"> ♦ Conexión ODBC <ul style="list-style-type: none"> ♦ Adabas DB ♦ IBM DB2 ♦ Solid ♦ Sybase SQL Anywhere ♦ MS SQL Server ♦ Ingres ♦ Oracle ♦ dBase ♦ InterBase ♦ PostgreSQL ♦ Empress ♦ FrontBase ♦ FilePro ♦ mSQL ♦ Sybase SQL Anywhere ♦ MySQL ♦ Velocis ♦ Informix ♦ Unix dbm 	<ul style="list-style-type: none"> ♦ Conexión ODBC <ul style="list-style-type: none"> ♦ Access ♦ MS SQL Server ♦ Oracle ♦ Sybase SQL Anywhere ♦ Informix ♦ DB2 ♦ PostgreSQL (con Chili!Soft ASP) ♦ Conexión ADO ♦ Conexión ActiveX ♦ Conexión RDO ♦ Conexión OLE-DB. 	<ul style="list-style-type: none"> ♦ Conexión JDBC <ul style="list-style-type: none"> ♦ MS SQL Server ♦ Oracle ♦ PostgreSQL ♦ Progress ♦ MySQL ♦ mSQL ♦ Informix ♦ Ingres ♦ DB2 ♦ Puente JDBC-ODBC <ul style="list-style-type: none"> ♦ MS Access ♦ PostgreSQL ♦ MS SQL Server ♦ Oracle ♦ MySQL ♦ DB2 ♦ Sybase SQL Anywhere ♦ Informix 	3	2	2
LENGUAJES DE SCRIPT SOPORTADOS	<ul style="list-style-type: none"> ♦ PHP ♦ PERL 	<ul style="list-style-type: none"> ♦ VBScript ♦ JScript ♦ PerlScript (sólo en Unix/Linux) 	<ul style="list-style-type: none"> ♦ Java ♦ SSJS (JavaScript del lado del servidor) 	2	3	1
LENGUAJES DE MARCADO SOPORTADOS	<ul style="list-style-type: none"> ♦ HTML/DHTML/WML ♦ XML a través de funciones 	<ul style="list-style-type: none"> ♦ HTML/DHTML/WML ♦ XML 	<ul style="list-style-type: none"> ♦ HTML/DHTML/WML ♦ XML 	3	3	3

TIPO DE PÁGINA DE RESPUESTA QUE PUEDE GENERAR	<ul style="list-style-type: none"> HTML/DHTML XML WML 	<ul style="list-style-type: none"> HTML/DHTML XML WML 	<ul style="list-style-type: none"> HTML/DHTML XML WML 	3	3	3
TAGS PARA INCRUSTAR SCRIPTS	<ul style="list-style-type: none"> <? ?> <?php ?> <script language="php"> </script> <% %> 	<ul style="list-style-type: none"> <%@language="VBScript" RUNAT=Server %> <%@language="JScript" RUNAT=Server % > <% %> 	<ul style="list-style-type: none"> <%@page language ="java" %> <% %> TAGS personalizados creados por el usuario 	3	2	3
EDITORES DE SCRIPTS	<ul style="list-style-type: none"> Editores de texto PhpEd (sólo Win32) Dreamweaver (sólo Win32) 	<ul style="list-style-type: none"> Editores de texto Drumbeat 2000 Visual Interdev ASPEXpress Editor Dreamweaver (sólo Win32) 	<ul style="list-style-type: none"> Editores de texto Powerfull JSP Editor Dreamweaver (sólo Win32) Forte Visual Age for Java. 	1	3	3
COMBINA HERRAMIENTAS CLIENTE	<ul style="list-style-type: none"> HTML/DHTML/WML JavaScript (opcional) XML (opcional) 	<ul style="list-style-type: none"> HTML/DHTML/WML VBScript 	<ul style="list-style-type: none"> HTML/DHTML/WML JavaScript XML (opcional) 	3	2	3
USO DE COMPONENTES REUTILIZABLES	<ul style="list-style-type: none"> Interacción con: <ul style="list-style-type: none"> COM/DCOM(Windows) Componentes Java PHP Component Model PCOM (Próximamente con PHP 5) 	<ul style="list-style-type: none"> COM (VB, C++, J++) ActiveX 	<ul style="list-style-type: none"> JavaBeans EJB TAGS personalizados 	1	3	3
PORTABILIDAD	Si (Independiente de plataforma)	No (Dependiente de plataforma)	Si (Independiente de plataforma)	3	0	3
FORMAS DE EJECUCIÓN	<ul style="list-style-type: none"> Como CGI Como módulo del servidor 	<ul style="list-style-type: none"> Como API del servidor 	<ul style="list-style-type: none"> Como módulo del servidor 	3	2	2

TIPO DE INSTALACIÓN	<ul style="list-style-type: none"> • Vía RPM en Linux • Binaria mediante compilación • Ejecutable (Windows) • Copiando archivos (Windows) 	<ul style="list-style-type: none"> • Ejecutable con Internet Information Server • Mediante Chili!Soft ASP: <ul style="list-style-type: none"> • Ejecutable en Windows • Binario ejecutable en Linux 	<ul style="list-style-type: none"> • Copiando archivos en Windows • Copiando archivos en Linux 	3	3	2
SEPARA PROGRAMACIÓN DE PRESENTACIÓN	Si	Si	Si	3	3	3
SITIOS QUE USAN LA HERRAMIENTA	<ul style="list-style-type: none"> • Mitsubishi • RedHat • Der Spiegel • MP3-Lycos • Ericsson • NASA 	<ul style="list-style-type: none"> • Berkley International • Estaentodo 	<ul style="list-style-type: none"> • Portal de Java • Portal de IEEE • Portal de Siemens(celulares) 	3	2	2
CARACTERÍSTICAS QUE DESTACAN	<ul style="list-style-type: none"> • Modular • Gran cantidad de funciones • Configurable mediante php.ini 	<ul style="list-style-type: none"> • Cuenta con el archivo golbal.asa, donde se puede especificar parámetros de inicio 	<ul style="list-style-type: none"> • Permite TAGS propias del usuario 	3	2	2
SOPORTE TÉCNICO	<ul style="list-style-type: none"> • Comunidad Internet • Empresa Zend • FAQs • Foros 	<ul style="list-style-type: none"> • Empresa MicroSoft • FAQs • Foros • Sun Chili!Soft para Chili!Soft ASP 	<ul style="list-style-type: none"> • Comunidad Internet • Empresa Sun Microsystems • FAQs • Foros 	3	3	3
MANEJO DE SESIONES	Si	Si	Si	3	3	3
MODO DE EJECUCIÓN	Intérprete	Intérprete	Intérprete	3	3	3
FACILIDAD DE USO	<ul style="list-style-type: none"> • Conocimientos de PHP, C++ o PERL • Conocimientos de P.O.O 	<ul style="list-style-type: none"> • Conocimientos VBScript, JScript o PerlScript 	<ul style="list-style-type: none"> • Conocimientos de Java o SSJS • Conocimientos de P.O.O 	3	2	3
TIPO DE DOCUMENTACIÓN	<ul style="list-style-type: none"> • Referencia del lenguaje • Ejemplos 	<ul style="list-style-type: none"> • Ayuda MSDN de Microsoft • Ayuda PDF de Chili!Soft 	<ul style="list-style-type: none"> • Páginas de ejemplos 	3	3	2

FORMATO DE DOCUMENTACIÓN	<ul style="list-style-type: none"> HTML PDF 	<ul style="list-style-type: none"> Ayuda típica de Windows PDF 	<ul style="list-style-type: none"> HTML PDF 	3	3	3
IDIOMA DE DOCUMENTACIÓN	<ul style="list-style-type: none"> Inglés Español Francés, Portugués, Italiano, etc. 	<ul style="list-style-type: none"> Inglés Español 	<ul style="list-style-type: none"> Inglés Español 	3	2	2
SEGURIDAD	Si (Configurando el archivo php.ini)	No existe archivo de configuración	No existe archivo de configuración	3	0	0
MANEJO DE ERRORES	Si	Si	Si	3	3	3
SOFTWARE REQUERIDO (Plataformas más utilizadas)	<ul style="list-style-type: none"> Windows o Linux Internet Information Server o Apache 	<ul style="list-style-type: none"> Windows Internet Information Server 3.0 o superior Service Pack 3 o 6 Controlador ODBC Con ChiliSoft ASP para Linux o Windows : Apache Web Server 	<ul style="list-style-type: none"> Windows o Linux Internet Information Server o Tomcat-Apache JDK 1.2 o superior Módulo mod_jk Controlador JDBC o JDBC - ODBC 	3	1	2
HARDWARE REQUERIDO (Plataformas más utilizadas)	<ul style="list-style-type: none"> Pentium II o superior 	<ul style="list-style-type: none"> Pentium II o superior 	<ul style="list-style-type: none"> Pentium II o superior 	3	3	3
TOTAL:				127	93	103

Tabla 2.2 Sintaxis de los lenguajes del lado del servidor: PHP, ASP, JSP

SINTAXIS PARA:	PHP (Usando PHP)	ASP (Usando VBScript)	JSP (Usando Java)
Incluir código en las páginas web	<ul style="list-style-type: none"> ♦ <? ?> ♦ <?php ?> ♦ <script language="php"> </script> ♦ <% %> ♦ <% =expresion; %> 	<ul style="list-style-type: none"> ♦ <%@ language="VBSCRIPT" RUNAT=Server %> ♦ <% %> ♦ <% =expresion %> 	<ul style="list-style-type: none"> ♦ <%@page language ="java" %> ♦ <% %> ♦ <% =expresion; %>
Sensible a mayúsculas/ minúsculas.	<ul style="list-style-type: none"> ♦ Sí para variables ♦ No para palabras reservadas 	<ul style="list-style-type: none"> ♦ No 	<ul style="list-style-type: none"> ♦ Si
Tipos de datos	<ul style="list-style-type: none"> ♦ int, integer ♦ real, double, float ♦ string ♦ array ♦ object 	<ul style="list-style-type: none"> ♦ Variant, que se subdivide en: <ul style="list-style-type: none"> ♦ Byte, Integer, Long, Currency, Array ♦ Single, Double ♦ Null, Empty ♦ Boolean ♦ Object ♦ Date ♦ String ♦ Error 	<ul style="list-style-type: none"> ♦ byte, short, int, long ♦ float, double ♦ true, false ♦ char, string ♦ array
	<ul style="list-style-type: none"> ♦ Aritméticos: +, -, *, /, % ♦ Asignación: =, +=, -=, *=, /=, %=, -= 	<ul style="list-style-type: none"> ♦ Aritméticos: +, -, *, /, mod ♦ Asignación: = 	<ul style="list-style-type: none"> ♦ Aritméticos: +, -, *, /, % ♦ Asignación: =, +=, *=, /=, %=, -=

Operadores básicos	<ul style="list-style-type: none"> Comparación: ==, ===, !=, <, >, <=, >=, ?: Incremento/Decremento: ++, -- Lógicos: and, or, xor, !, &&, 	<ul style="list-style-type: none"> Comparación: =, <>, <, >, <=, >= Incremento/Decremento: ++, -- Lógicos: AND, OR 	<ul style="list-style-type: none"> Comparación: ==, !=, <, >, <=, >=, ?: Incremento/Decremento: ++, -- Lógicos: &&,
Finalizar expresiones	<ul style="list-style-type: none"> Punto y coma (;) obligatorio 	<ul style="list-style-type: none"> Fin de línea 	<ul style="list-style-type: none"> Punto y coma (;) obligatorio
Incluir archivos	<ul style="list-style-type: none"> include("archivo.php"); include("archivo.inc"); require("archivo.php"); require("archivo.inc"); requireonce("archivo.php"); requireonce("archivo.inc"); 	<ul style="list-style-type: none"> <!-- #Include File="archivo.asp"--> 	<ul style="list-style-type: none"> <%@ include file= "archivo.jsp" %> <%@ page import="clase_java" %> (incluir o importar clases o paquetes Java)
Comentarios	<ul style="list-style-type: none"> //Aquí va el comentario de una línea /* Aquí va el comentario de varias líneas */ #Aquí va el comentario de una línea, tipo Unix 	<ul style="list-style-type: none"> 'Aquí va el comentario de una línea 	<ul style="list-style-type: none"> <!-- Aquí va el comentario de una o varias líneas -->
Necesita declarar variables y asignarles tipo	<ul style="list-style-type: none"> No se declara, se usa directamente, no ocurren errores No se define tipo, toma el valor de lo que se asigne 	<ul style="list-style-type: none"> Opcional, pero para evitar errores y obligar a declararlas se requiere la siguiente instrucción al inicio: <% Option Explicit %> Toman el tipo del valor que se asigne, por defecto son tipo Variant 	<ul style="list-style-type: none"> Si se declaran las variables obligatoriamente Si se declaran los tipos de datos obligatoriamente
Reglas para variables y constantes	<ul style="list-style-type: none"> Comenzar con carácter alfabético, no usar palabras reservadas. Las constantes siguen la misma regla que las variables pero se declaran en mayúsculas. 	<ul style="list-style-type: none"> Comenzar con carácter alfabético, no contener punto, no pasar de 255 caracteres, no usar palabras reservadas. Las constantes siguen la misma regla que las variables pero se declaran en mayúsculas. 	<ul style="list-style-type: none"> Comenzar con carácter alfabético, no tienen límite de longitud, no usar palabras reservadas. Las constantes siguen la misma regla que las variables pero se declaran en mayúsculas.
Declarar variables	<ul style="list-style-type: none"> <?php \$variable = valor; ?> (Se usa directamente sin declarar y sin tipo de dato) <?php \$variable = new clase(); ?> 	<ul style="list-style-type: none"> <% Dim variable %> <% Dim variable As tipo %> <% Public variable %> <% Public variable As tipo %> <% Private variable %> <% Private variable As tipo %> 	<ul style="list-style-type: none"> <%! tipo_variable variable; %> <%! variable = new tipo_variable(); %> <%! variable = new Clase(); %>
Declarar	<ul style="list-style-type: none"> define("MI_CONSTANTE", valor); 	<ul style="list-style-type: none"> Const MI_CONSTANTE = valor 	<ul style="list-style-type: none"> final tipo_dato MI_CONSTANTE = valor;

constantes			
Cambiar tipo de variables	<ul style="list-style-type: none"> Si, usando: <ul style="list-style-type: none"> \$var_nuevo_tipo = (tipo_datos) \$variable; expresion; typecasting settype(\$var expresion, tipo_datos) ejem: \$variable = 1; settype(\$variable, string); 	<ul style="list-style-type: none"> Si, usando las siguientes funciones: <ul style="list-style-type: none"> Cbool(variable expresion) Cbyte(variable expresion) Cint(variable expresion) Clng(variable expresion) Csng(variable expresion) Cdbl(variable expresion) Ccur(variable expresion) Cdate(variable expresion) Cstr(variable expresion) 	<ul style="list-style-type: none"> Si, usando: <ul style="list-style-type: none"> (tipodato)variable; expresion; ejem: (long)variable; tipodato.valueOf(variable expresion).tipodatoValue(); ejem: Float.valueOf(variable).floatValue(); tipodato.toTipodato(variable expresion); ejem: Integer.toString(variable); tipodato variable = new tipoFormat("####.00");
Asignar valores	<ul style="list-style-type: none"> \$variable = valor; 	<ul style="list-style-type: none"> variable = valor 	<ul style="list-style-type: none"> tipo_variable variable = valor; variable = new tipo_variable(valor);
Abreviar asignaciones	<ul style="list-style-type: none"> \$variable+= \$variable+1; \$variable.= 'cadena'; 	<ul style="list-style-type: none"> No 	<ul style="list-style-type: none"> variable+= variable+1; variable+= 'cadena';
Imprimir variables	<ul style="list-style-type: none"> El valor es : <?php echo \$variable; ?> El valor es: <?php print (\$variable); ?> El valor es: <?php print_r (\$variable); ?> 	<ul style="list-style-type: none"> El valor es: <%= variable %> 	<ul style="list-style-type: none"> El valor es: <% System.out.println(variable); %> El valor es: <% out_println(variable); %> El valor es: <%= variable; %>
Manejo de cadenas	<ul style="list-style-type: none"> \$cad = 'esta es una cadena'; \$cad = "esta es una cadena"; \$cad = <<<EOS esta es una cadena EOS; 	<ul style="list-style-type: none"> cad = "esta es una cadena" 	<ul style="list-style-type: none"> string cad ="esta es una cadena "; string cad = new string("esta es una cadena");
Imprimir cadenas	<ul style="list-style-type: none"> echo "Hola Mundo"; print("Hola Mundo"); 	<ul style="list-style-type: none"> Response.Write "Hola Mundo" 	<ul style="list-style-type: none"> System.out.println("Hola Mundo"); out_println("Hola Mundo");

<p>Concatenar cadenas</p>	<ul style="list-style-type: none"> ♦ \$cadena = 'A' . 'B'; \$cadena="A". "B"; ♦ \$cadena = \$cad1.\$cad2; 	<ul style="list-style-type: none"> ♦ cadena = "A" & "B" ♦ cadena = "cad1" & "cad2" 	<ul style="list-style-type: none"> ♦ string cadena = "A" + "B"; ♦ string cadena = cad1 + cad2;
<p>Declarar arrays</p>	<ul style="list-style-type: none"> ♦ \$variable_array = array(); ♦ \$variable_array [1] = valor; ♦ \$variable_array = array(valor_1, valor_2, valor_n); ♦ \$variable_array [1][0] = valor; ♦ list(\$var_1, \$var_2, \$var_3, ..., \$var_n) 	<ul style="list-style-type: none"> ♦ Dim variable_array(10) ♦ variable_array(1) = valor ♦ Dim variable_array(3, 5) ♦ variable_array(1, 2) = valor 	<ul style="list-style-type: none"> ♦ tipo_dato variable_array[]; ♦ string variable_array[1] = valor; ♦ tipo_dato variable_array[][] = new tipo_dato[4][5]; ♦ int variable_array[2][1] = valor; ♦ tipo_dato variable_array{valor_1, valor_2, valor_n}; ♦ string variable_array{"Uno","Dos","Tres"};
<p>Arrays asociativos</p>	<ul style="list-style-type: none"> ♦ \$variable_array ["nombre"] = valor; ♦ \$variable_array ["nombre"][2] = valor; ♦ \$variable_array [nombre]["color"] = valor; ♦ \$variable_array ["color"][4]["tamaño"][0] = valor; ♦ \$variable_array = array("color" => "rojo", "sabor" => "dulce", 3 => 4); 	<ul style="list-style-type: none"> ♦ No 	<ul style="list-style-type: none"> ♦ No
<p>Declarar fechas/horas</p>	<ul style="list-style-type: none"> ♦ \$fecha_actual= date("Y-m-d"); ♦ \$fecha_hora= date("Y-m-d H:i:s"); ♦ \$fecha = time(); ♦ \$fecha = mktime(0,0,0,1,30,2000); ♦ echo date('d-m-Y',\$fecha); 	<ul style="list-style-type: none"> ♦ fecha_actual = Now ♦ fecha = #30-01-2000# ♦ fecha = CDate('2000-1-30') 	<ul style="list-style-type: none"> ♦ Date fecha = new Date(); ♦ fecha_actual = new SimpleDateFormat("dd/MM/yyyy"); ♦ System.out.println(fecha_actual.format(fecha)); ♦ Calendar fecha = Calendar.getInstance(); ♦ fecha.set(1999, 9, 11); ♦ Date fecha1 = fecha.getTime(); ♦ fecha_ = new SimpleDateFormat("dd/MM/yyyy"); ♦ System.out.println(fecha_.format(fecha1));

Funciones básicas para fechas/horas	<ul style="list-style-type: none"> ♦ checkdate (valida fechas) ♦ getdate (obtiene fecha y hora actual) ♦ gettimeofday (obtiene hora actual) ♦ time (obtiene timestamp) 	<ul style="list-style-type: none"> ♦ FormatDateTime(fecha,2) (formato para fechas) 	<ul style="list-style-type: none"> ♦ fecha = new SimpleDateFormat("dd/MM/yyyy"); ♦ fecha = Calendar.getInstance(); ♦ fecha1 = fecha.getTime();
Recoger variables POST	<ul style="list-style-type: none"> ♦ \$variable_post = \$_POST['variable']; ♦ \$variable_post = \$HTTP_POST_VARS['variable']; ♦ \$variable_post = \$variable; 	<ul style="list-style-type: none"> ♦ variable_post = Request.Form("variable") 	<ul style="list-style-type: none"> ♦ String variable_post = request.getParameter("variable");
Recoger variables GET	<ul style="list-style-type: none"> ♦ \$variable_get = \$_GET['variable']; ♦ \$variable_get = \$HTTP_GET_VARS['variable']; ♦ \$variable_get = \$variable; 	<ul style="list-style-type: none"> ♦ variable_get = Request.QueryString("variable") 	<ul style="list-style-type: none"> ♦ String variable_get = request.getQueryString("variable");
Manejar Sesiones	<ul style="list-style-type: none"> ♦ session_start(); session_name("nombre_sesion"); session_register("variable_sesion"); \$variable_sesion=valor; ♦ session_start(); session_name("nombre_sesion"); \$_SESSION['variable_sesion']=valor; ♦ session_start(); session_name("nombre_sesion"); \$HTTP_SESSION_VARS['variable_sesion']=valor; 	<ul style="list-style-type: none"> ♦ Session("variable_sesion")= valor 	<ul style="list-style-type: none"> ♦ HttpSession session = request.getSession(true); session.setAttribute(variable_sesion, valor); ♦ HttpSession session = request.getSession(true); session.putValue("variable_sesion", valor);
Recoger variables de Sesión	<ul style="list-style-type: none"> ♦ \$var_sesion = \$variable_sesion; ♦ \$var_sesion = \$_SESSION['variable_sesion']; ♦ \$var_sesion = \$HTTP_SESSION_VARS['variable_sesion']; 	<ul style="list-style-type: none"> ♦ var_sesion = Session("variable_sesion") 	<ul style="list-style-type: none"> ♦ var_sesion = session.getValue("variable_sesion"); ♦ var_sesion = (String) session.getAttribute(variable_sesion);

Destruir Sesiones	<ul style="list-style-type: none"> ♦ session_start(); ♦ session_name("nombre_sesion"); ♦ session_destroy(); 	<ul style="list-style-type: none"> ♦ Session.Abandon 	<ul style="list-style-type: none"> ♦ HttpSession session = request.getSession(false);
Manejo de Cookies	<ul style="list-style-type: none"> ♦ setcookie ("nombre_cookie", \$var_cookie, valor); 	<ul style="list-style-type: none"> ♦ Response.cookies("var_cookie")= valor ♦ Response.cookies("var_cookie")("atributo")= valor 	<ul style="list-style-type: none"> ♦ Cookie var_cookie = new Cookie("nombre_cookie", valor); ♦ response.addCookie(var_cookie);
Recoger cookies	<ul style="list-style-type: none"> ♦ \$var_cookie_ = \$var_cookie; ♦ \$var_cookie_ = \$HTTP_COOKIE_VARS['var_cookie']; 	<ul style="list-style-type: none"> ♦ var_cookie_ = Request.cookies("var_cookie") ♦ var_cookie_ = Request.cookies("var_cookie")("atributo") 	<ul style="list-style-type: none"> ♦ Cookie[] var_cookies = request.getCookies(); ♦ var_cookie_ = var_cookies[posicion].getValue();
VARIABLES de Servidor (VAR_SERVER Depende del servidor que se use)	<ul style="list-style-type: none"> ♦ \$HTTP_SERVER_VARS[VAR_SERVER] como módulo ♦ \$var_servidor = \$HTTP_SERVER_VARS[SERVER_NAME] ♦ \$_SERVER[VAR_SERVER] como módulo ♦ \$var_servidor = \$_SERVER[SERVER_NAME] ♦ \$HTTP_ENV_VARS[VAR_SERVER] como CGI ♦ \$var_servidor = \$HTTP_ENV_VARS[SERVER_NAME] 	<ul style="list-style-type: none"> ♦ Request.ServerVariables("VAR_SERVER") ♦ var_servidor = Request.ServerVariables("SERVER_NAME") 	<ul style="list-style-type: none"> ♦ Existen métodos para acceder a cada variable, ejemplo: ♦ var_servidor = Request.getServerName(); ♦ var_servidor = Request.getHost(); ♦ var_servidor = Request.getRemoteUser();
Redireccionar páginas	<ul style="list-style-type: none"> ♦ header("Location: url/pagina.php"); ♦ header("Location: url/pagina.php?param_1=valor_1&param_2=valor_2&param_n=valor_n"); 	<ul style="list-style-type: none"> ♦ response.redirect "url/pagina.asp" ♦ response.redirect "url/pagina.asp?param_1=valor_1&param_2=valor_2&param_n=valor_n" ♦ Server.Transfer("url/pagina.asp") 	<ul style="list-style-type: none"> ♦ response.sendRedirect(response.encodeRedirectURL("../pagina.jsp")); ♦ response.sendRedirect(response.encodeRedirectURL("../pagina.jsp?param_1=valor_1&param_2=valor_2&param_n=valor_n"));

<p>Redireccionar páginas con enlaces</p>	<ul style="list-style-type: none"> ♦ <code><a href "url/pagina.php">Enlace</code> ♦ <code><a href "url/pagina.php?param_1=valor_1&param_2=valor_2&param_n=valor_n"> Enlace </code> 	<ul style="list-style-type: none"> ♦ <code><a href "url /pagina.asp">Enlace</code> ♦ <code><a href "url /pagina.asp?param_1=valor_1&param_2=valor_2&param_n=valor_n"> Enlace </code> 	<ul style="list-style-type: none"> ♦ <code><a href "url /pagina.asp">Enlace</code> ♦ <code><a href "url /pagina.asp?param_1=valor_1&param_2=valor_2&param_n=valor_n"> Enlace </code> ♦ <code><a href="<%= response.encodeURL("../pagina.jsp") %>">Enlace</code> ♦ <code><a href="<%= response.encodeURL("../pagina.jsp?param_1=valor_1&param_2=valor_2&param_n=valor_n")%>">Enlace </code>
<p>Escribir hipertexto</p>	<ul style="list-style-type: none"> ♦ <code>echo "<tr><td>".\$var. "</td><td>" . \$var1. "</td></tr>";</code> ♦ <code>echo "";</code> 	<ul style="list-style-type: none"> ♦ <code>Response.Write "<tr><td>"& var & "</td><td>" & var1 & "</td></tr>"</code> ♦ <code>Response.Write ""</code> 	<ul style="list-style-type: none"> ♦ <code>System.out.println("<tr><td>" + var + "</td><td>" + var1 + "</td></tr>");</code> ♦ <code>System.out.println("");</code>
<p>Declarar funciones</p>	<ul style="list-style-type: none"> ♦ <code>function nombre funcion(\$var 1, \$var n) {</code> ♦ <code>}</code> 	<ul style="list-style-type: none"> ♦ <code>Function nombre funcion(Dim var 1, Dim var n)</code> ♦ <code>End Function</code> 	<ul style="list-style-type: none"> ♦ <code>function nombre funcion(tipo var 1, tipo var n) {</code> ♦ <code>}</code>
<p>Declarar proced.</p>	<ul style="list-style-type: none"> ♦ Son funciones que no retornan valor 	<ul style="list-style-type: none"> ♦ <code>Sub nombre_procedimiento(Dim var_1, Dim var_n)</code> ♦ <code>End Sub</code> 	<ul style="list-style-type: none"> ♦ Son funciones que no retornan valor
<p>Manejo de clases y herencia</p>	<ul style="list-style-type: none"> ♦ Si maneja clases ♦ Si maneja herencia 	<ul style="list-style-type: none"> ♦ Si maneja clases, método primitivo sólo desde Visual Basic 5.5 ♦ No maneja herencia 	<ul style="list-style-type: none"> ♦ Si maneja clases ♦ Si maneja herencia

<p>Estructuras básicas de control</p>	<ul style="list-style-type: none"> ♦ Ifelse else if ♦ else if.....else : endif; ♦ while ♦ while.....endwhile; ♦ do.....while ♦ for..... ♦ foreach..... ♦ break; ♦ continue; ♦ switch.....case.....break; 	<ul style="list-style-type: none"> ♦ if.....then..... else elseifendif ♦ while.....endwhile ♦ while.....wend ♦ do...loop.....while.....endwhile ♦ for..... next ♦ for each.....next ♦ select..... case.....break;.....end select 	<ul style="list-style-type: none"> ♦ Ifelse else if ♦ while ♦ do.....while ♦ for..... ♦ break; ♦ continue; ♦ switch.....case.....break; ♦ try.....catch
<p>Manejo de Errores</p>	<ul style="list-style-type: none"> ♦ Por medio de constantes con determinado valor, para errores de ejecución, compilación y errores de usuarios. ♦ Con la función: die("Mensaje de Error"); ♦ Depende de la configuración del archivo php.ini. 	<ul style="list-style-type: none"> ♦ Se maneja por medio de OnError. 	<ul style="list-style-type: none"> ♦ Se maneja capturando errores por medio de excepciones, usando la estructura de control: try.....catch
<p>Objetos implícitos</p>	<ul style="list-style-type: none"> ♦ No cuenta con objetos implícitos, pero sí con variables equivalentes. 	<ul style="list-style-type: none"> ♦ Application ♦ Request ♦ Response ♦ Redirect ♦ Server ♦ Session ♦ Object 	<ul style="list-style-type: none"> ♦ Request ♦ Response ♦ Page ♦ Put ♦ Session ♦ PageContext ♦ Exception ♦ Application ♦ Config ♦ Out

Tabla 2.3 Sintaxis para conexión y operaciones con la base de datos PostgreSQL

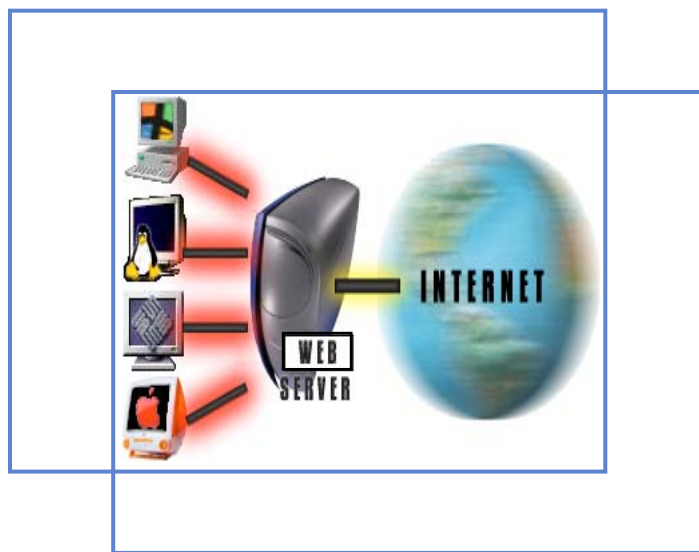
SINTAXIS PARA:	PHP (Usando PHP)	ASP (Usando VBScript)	JSP (Usando Java)
Conexión a PostgreSQL	<pre>\$conexion=pg_connect("host", "base_de_datos"); \$conexion=pg_connect("host", port="5432", "base_de_datos", "user", "pass"); \$conexión=pg_pconnect("host", "base_de_datos"); \$conexión=pg_pconnect("host", port="5432", "base_de_datos", "user", "pass");</pre>	<pre>cadena_conexion="DATABASE=prueba;UID=adminbdd;PWD=;DSN=postgresql_esp" Set conexion = Server.CreateObject ("ADODB.Connection") Set resultado = Server.CreateObject ("ADODB.RecordSet") conexion.Open cadena_conexion</pre>	<pre><%@page import="java.sql.*"%> String url = "jdbc:postgresql://localhost/"; Driver driver_bdd = (Driver) Class.forName("org.postgresql.Driver"); Connection conexion = DriverManager.getConnection(url, "usuario_bdd", "clave_bdd");</pre>
Consultar datos	<pre>\$sql="SELECT nombre, tipo_usuario FROM usuarios;"; \$resultado = pg_Exec (\$conexion, \$sql); \$filas = pg_NumRows(\$resultado); \$i = 0; while (\$i <= \$filas) { echo "NOMBRE: ".pg_result(\$resultado, \$i, 0)."
 . "TIPO_DE_USUARIO: ".pg_result(\$resultado, \$i, 1)."
"; }</pre>	<pre>Set resultado = Server.CreateObject ("ADODB.RecordSet") sql = "SELECT nombre, tipo_usuario FROM usuarios" resultado.Open sql, conexion Do While not resultado.EOF Response.Write "NOMBRE:" Response.Write resultado ("nombre") & " - "
" Response.Write "TIPO_DE_USUARIO:" Response.Write resultado ("tipo_usuario ") & " - "
" resultado.MoveNext Loop</pre>	<pre>ResultSet resultado = stmt.executeQuery("SELECT nombre, tipo_usuario FROM usuarios "); while (resultado.next()) { String nombre_usr = resultado.getString("nombre"); String tipo_usr = resultado.getString("tipo_usuario"); System.out.println("NOMBRE:" + nombre_usr + "
"); System.out.println("TIPO_DE_USUARIO:" + tipo_usr + "
"); }</pre>
Cerrar la conexión	<pre>pg_close(\$conexion); pg_pclose(\$conexion);</pre>	<pre>conexion.Close</pre>	<pre>conexion.close();</pre>

Tabla 2.4 Principales Funciones de PHP, ASP y JSP

FUNCIONES:	PHP (Usando PHP)	ASP (Usando VBScript)	JSP (Usando Java)
BÁSICAS DEL LENGUAJE	<ul style="list-style-type: none"> ♦ Funciones sobre variables ♦ Funciones sobre cadenas ♦ Funciones de matrices ♦ Funciones matemáticas de precisión arbitraria ♦ Funciones matemáticas ♦ Funciones para manejo de errores ♦ Funciones para manejo de calendario ♦ Funciones para manejo de fecha y hora ♦ Funciones para archivos y directorios ♦ Funciones para manejo de sesiones ♦ Funciones para manejo de cookies ♦ Funciones para Apache Web Server ♦ Funciones para clases/objetos ♦ Funciones para BDD: dBase, dbm, Informix, InterBase, Ingres, FrontBase, filePro, MS SQL Server, mSQL, MySQL, PostgreSQL, ODBC, Oracle , Oracle 8, Sybase SQL Anywhere 	<ul style="list-style-type: none"> ♦ Funciones para variables ♦ Funciones sobre cadenas ♦ Funciones matemáticas ♦ Funciones para fechas y hora ♦ Funciones para formato de fechas ♦ Funciones de conversión de datos ♦ Funciones para ADO ♦ Funciones para manejo de sesiones ♦ Funciones para manejo de cookies 	<ul style="list-style-type: none"> ♦ Funciones para tipo de variables ♦ Funciones sobre cadenas ♦ Funciones para fechas y hora ♦ Funciones matemáticas ♦ Funciones para formato de fechas ♦ Funciones para conversión de datos ♦ Funciones para de archivos y directorios ♦ Funciones para manejo de sesiones ♦ Funciones para manejo de cookies ♦ Funciones para cargar controladores JDBC
	<ul style="list-style-type: none"> ♦ Funciones COM para Windows ♦ Funciones para compresión ♦ Funciones para PDF ♦ Funciones para XML ♦ Funciones para formato de datos de formularios ♦ Funciones FTP 	<ul style="list-style-type: none"> ♦ Funciones para manejo de componentes 	<ul style="list-style-type: none"> ♦ Funciones para manejo de JavaBeans

<p>AVANZADAS DEL LENGUAJE</p>	<ul style="list-style-type: none">♦ Funciones HTTP♦ Funciones para Lotus Notes♦ Funciones ICAP♦ Funciones para imágenes♦ Funciones IMAP♦ Funciones LDAP♦ Funciones Java♦ Funciones de red♦ Funciones de correo♦ Funciones criptográficas♦ Funciones Hash♦ Funciones de información de PHP♦ Funciones POSIX♦ Funciones de ejecución de programas♦ Funciones de impresión♦ Funciones de expresiones regulares♦ Funciones para manejo de memoria♦ Funciones para Shockwave Flash♦ Funciones SNMP♦ Funciones para sockets♦ Funciones URL♦ Funciones para OpenSSL♦ Funciones MimeTypes♦ Funciones para mnoGoSearch♦ Funciones para manejo de funciones definidas por el usuario		
-------------------------------	--	--	--

Capítulo 3



SERVIDORES WEB

- ❖ Introducción
- ❖ Arquitectura Cliente/Servidor en Internet
- ❖ Concepto de Servidor Web
- ❖ Evolución de los Servidores Web
- ❖ Funcionamiento de un Servidor Web
- ❖ Características de los Servidores Web
- ❖ Importancia como componente principal de sitios y aplicaciones web
- ❖ Evaluación de los servidores web gratuitos y comerciales
 - ❖ Apache Web Server
 - ❖ Internet Information Server
 - ❖ Tomcat Web Server
- ❖ Comparación de los servidores web estudiados

INTRODUCCIÓN

Un Servidor Web es un componente fundamental en un sitio web o portal en Internet, ya que ahí es donde se almacenarán las páginas web y todos los componentes necesarios para que éstas funcionen correctamente, por esto es necesario conocer sus características, ventajas, desventajas y funcionamiento para saber como integrarlo con su respectiva herramienta de generación de contenido dinámico(PHP, ASP, JSP).

Es importante hacer un estudio comparativo generalizado de servidores web gratuitos y comerciales, para de esta forma determinar qué características buscar a la hora de elegir un servidor web.

En este capítulo se seleccionará y estudiará en detalle un servidor web para cada herramienta de generación de contenido dinámico(PHP, ASP, JSP).

3.1 ARQUITECTURA CLIENTE/SERVIDOR EN INTERNET

Evidentemente, Internet es la red de redes, y como en toda red, participan: la máquina local desde la que accedemos a la red, que funciona gracias a un programa cliente, y la máquina remota a la que accedemos, y nos facilita lo que necesitamos gracias a un programa servidor.

Los aspectos técnicos que hicieron posible el gran desarrollo y funcionamiento de Internet fueron: la creación de un eficiente protocolo de comunicaciones llamado *TCP/IP* y la popularización del sistema *WWW*.

Mientras el protocolo *TCP* realiza el control de los datos a transmitir, el chequeo de errores de la transmisión y reenvío de paquetes cuando se requiere; el protocolo *IP* define una **dirección IP** única para cada uno de los servidores existentes en Internet, como sistema de identificación.

De esta manera, Internet hace uso del protocolo *TCP/IP* para la transmisión de datos y para identificar una máquina y la red a la cual está conectada. Únicamente el *NIC* asigna las direcciones *IP* en Internet.

Una dirección *IP* es un número de 32 bits que suele representarse en forma decimal, separando con puntos cada uno de los 4 bytes que constituyen la dirección, por ejemplo: 123.12.1.23.

Debido a que es más fácil recordar e identificar un nombre simbólico, en lugar de una serie de números, los usuarios le asignan a los servidores una dirección alfabética como por ejemplo: *www.mi-servidor.com*, denominada **nombre de dominio**. Este nombre de dominio debe ser único.

Para que exista compatibilidad entre la dirección *IP* y el nombre de dominio, se creó el sistema *DNS* para poder convertir los nombres de los servidores en las direcciones *IP* respectivas y viceversa. El objetivo es conciliar el mundo de las personas y el de las máquinas. Nosotros usamos nombres y las máquinas usan números. El *DNS* es fundamental para que Internet funcione.

Cabe destacar que toda máquina servidor proporciona sus servicios en Internet utilizando puertos numerados, para cada servicio que dispone. Para el servicio WWW se utiliza el puerto 80 por defecto. En Internet se puede acceder a un sitio de dos maneras: especificando el número de puerto **www.mi-servidor.com:80** o sin especificarlo **www.mi-servidor.com**; en caso de no especificar, se admitirá el puerto 80 por defecto. En el **Anexo 4, Tabla 3**, que se encuentra en el CD adjunto a este documento en: **CD_TESIS:\Tesis\Anexos_Tesis\Anexos.pdf**, se listan los servicios que puede ofrecer un servidor en Internet y el puerto correspondiente a cada uno de ellos.

Hasta este punto, la comunicación entre clientes y servidores en Internet estaba solucionada, pero hacía falta el software que se encargara de establecer la interfaz de comunicación Cliente/Servidor, más aún cuando se difundió masivamente el uso del sistema WWW, cuyo principal objetivo es el intercambio de documentos en formato hipertexto(más conocido como páginas web).

Pero para hacer posible el correcto funcionamiento del WWW, se implementó: el protocolo **HTTP**, el concepto de páginas web, (siendo éstas la unidad mínima de información del WWW), navegador(software del cliente) y **Servidor Web**(software del servidor).

Nota: El protocolo **HTTP**, básicamente, especifica el modo de comunicación entre un cliente y un servidor. El cliente solicita un documento, imagen o fichero del espacio de direcciones del servidor, y éste se lo sirve.

Todo usuario que ingresa a un sitio web, lo conozca o no, hace uso del protocolo **HTTP** y de los tres conceptos: páginas web, navegador y servidor web, aunque cabe destacar que la gran mayoría de usuarios sólo están familiarizados con el concepto de páginas web y navegadores, debido a que el uso del servidor web y el protocolo **HTTP** son de cierta forma transparentes al usuario.

Los servidores web cumplen un papel fundamental en los sitios web, ya que cada vez que se accesa a una página web hay comunicación con éstos. De ahí la importancia de conocer con exactitud que es un servidor web, más aún a la hora de desarrollar aplicaciones web.

3.2 CONCEPTO DE SERVIDOR WEB

Un **Servidor Web** o **Servidor de Páginas** es un software que se encuentra instalado en una máquina servidor conectada de forma permanente a Internet. Este software escucha en un puerto determinado, 80 por defecto, se ejecuta continuamente en espera de conexiones de clientes web(navegadores) y es el responsable de atender las peticiones(localizando los recursos solicitados) que hacen los usuarios de Internet, las 24 horas del día todos los días del año. [REV-02]

Un servidor web también puede ser visto como una estructura de directorios que almacenan todos los archivos necesarios(imágenes, audio, video, animaciones) para que las páginas web sean visualizadas correctamente. De la misma forma almacena los scripts o programas ,si los hay, necesarios para que las aplicaciones web sean ejecutadas.

Los servidores web técnicamente son conocidos como servidores *HTTP* o servidores *HTTPD*, por hacer uso del protocolo *HTTP*, al comunicarse con los navegadores.

Nota: La "D" de *HTTPD* es la inicial de *daemon*. Un *daemon* se refiere a un programa ejecutado en segundo plano y que espera solicitudes.

EVOLUCIÓN DE LOS SERVIDORES WEB

Tradicionalmente los servidores web se limitaban a comportarse como servidores de archivos *HTML*. Sin embargo la transición de páginas estáticas a dinámicas, exigió nuevas capacidades a los servidores web. Inicialmente, estos servidores incorporaban capacidades para interactuar con otros programas externos que se encargaban de generar el contenido a servir. Los ejemplos más conocidos son los programas *CGI*. La tendencia actual de los servidores web, gracias a la aparición de nuevas herramientas como *PHP*, *ASP*, *JSP*, es la de integrar dentro de su propio software, mediante módulos, el proceso que realizaban los programas externos; convirtiéndose así en la segunda generación de servidores web.

Actualmente, como una evolución más de los servidores web, existen los **Servidores de Aplicaciones**, estrictamente no son lo mismo que los servidores web de la segunda generación, pero se prestan para generar confusiones.

Los servidores de aplicaciones pueden ser vistos como la tercera generación de los servidores web, ya que además de cooperar con éstos, se encargan de ejecutar aplicaciones web, debido a que cuentan con características añadidas como:

- Generación dinámica de contenido incorporada, es decir, cuentan con su propio lenguaje, no se necesita añadir módulos.
- Acceso a bases de datos
- Multiprocesos
- Manejo de Sesiones
- *Lógica del Negocio*
- Balanceo de carga
- Administración del sitio web que sirven

Hoy en día, se puede decir que la mayoría de servidores web se comportan como servidores de aplicaciones ya que se les ha añadido alguna característica especial que les permite ejecutar aplicaciones web que generan contenido dinámico. Sin embargo, se ha generalizado el término y comúnmente se habla de servidores web aunque sean estrictamente servidores de aplicaciones. Algunos servidores de aplicaciones son: Bea WebLogic Server, ColdFusion, Oracle Application Server, WebSphere, Zope, JBoss, Sybase Enterprise App Server. Algunos servidores web son: Apache, Internet Information Server, etc.

FUNCIONAMIENTO DE UN SERVIDOR WEB

Cada vez que nos conectamos a Internet, a través de un navegador, para solicitar los servicios(principalmente páginas web) de un servidor web, se ejecutan los siguientes pasos:

1. Cuando el usuario digita, en la barra de direcciones, por ejemplo: **<http://www.microsoft.com/support/support.html>**, en el campo dirección del navegador o al hacer clic en un enlace de una página web, está haciendo referencia a un *URL*, el cual es una dirección de un servidor web en Internet. Dicho *URL*, está compuesto de tres partes:
 - El protocolo por medio del cual se solicita el servicio, ejemplo: **http://**
 - El nombre de dominio o dirección *IP* del servidor, ejemplo: **www.microsoft.com**
 - La ruta del recurso que solicitamos, ejemplo: **/support/support.html**

2. Seguidamente, el navegador decodifica el *URL*, separando sus diferentes partes: protocolo, nombre de dominio o dirección *IP* del servidor y la ruta del recurso requerido del servidor.
3. Dicho *URL* es enviado a un servidor *DNS* para ser buscado en todo Internet, de no encontrarse ésta dirección se devolverá un mensaje de error al navegador.
4. Caso contrario, se abre una conexión *TCP/IP* con el servidor.
5. Para realizar la petición, el navegador traduce el *URL* en una petición **GET**, **POST**, **HEAD**,..¹², según sea el caso, quedando de la siguiente manera, por ejemplo: **GET /support/support.html HTTP /1.1**. Donde se especifica el método para la petición, la ruta y la versión del protocolo.
6. Al enviar la petición, el navegador puede proporcionar información opcional al servidor, para que éste genere la respuesta. Esta información(denominada mensaje *MIME*) en forma de cabeceras, es enviada línea a línea y puede contener: el tipo de navegador, el tipo de contenidos que entiende(imágenes gif, jpeg), lenguaje, etc. Un ejemplo de cabeceras enviadas por el Internet Explorer 5.0 se muestra en la **Fig. 3.1**:

Accept: image/gif, image/x-bitmap, image/jpeg, application/vnd.msexcel, application/msword, application/vnd.ms-powerpoint, */*	//Formatos <i>MIME</i> aceptados
Accept-Language: es	//Lenguaje aceptado
Accept-Encoding: gzip, deflate	//Formato de codificación
User-Agent: Mozilla/4.0 (compatible; MSIE 5.0; Windows 98; DigExt)	//Nombre del navegador
Host: localhost	//Nombre del host
Connection: Keep-Alive	//Tipo de conexión

Fig. 3.1 Cabeceras enviadas por el navegador Internet Explorer 5.0 al servidor web

7. Una vez que se han enviado todos los datos de la petición, el servidor la procesa y devuelve una respuesta. Las respuestas del servidor consisten en tres partes: línea de estado, cabeceras y cuerpo. La línea de estado especifica la versión del protocolo *HTTP* que utiliza el servidor, un código de estado(error o éxito) y una descripción del código de estado(error o éxito). Por ejemplo: **HTTP /1.1 200 OK**.
8. A continuación, el servidor envía información(mensaje *MIME*) de igual manera que el navegador, en forma de cabeceras y línea a línea. Por ejemplo, el servidor Tomcat envía las cabeceras que se muestran en la **Fig. 3.2**:

¹² Métodos pertenecientes al protocolo *HTTP*, sirven para especificar la forma en que se realiza la petición al servidor web, son métodos transparentes al usuario.


```
Date: Tue, 03 Oct 2000 22:54:03 GMT //Fecha y hora
Status: 200 //Estado
Servlet-Engine: Tomcat Web Server/3.1(JSP 1.1; Servlet 2.2; Java 1.3; //Nombre del servidor
Windows 98 4.10 x86; java.vendor=Sun Microsystems Inc.)
Content-Type: text/HTML //Contenido MIME
Last-Modified: Thu, 14 Sep 2000 23:28:28 GMT //Última modificación
Content-Length: 2520 //Tamaño del contenido
Content-Language: en //Lenguaje aceptado
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN"> //Cuerpo de la respuesta
<HTML><HEAD>.....</HEAD>
<BODY>.....</BODY>
</HTML>
```

Fig. 3.2 Cabeceras enviadas por el servidor web Tomcat al navegador

9. Seguido de todo esto, el servidor envía la información solicitada en formato de hipertexto(*HTML/XML/WML*), aunque también puede enviar código *JavaScript*, *Java*, etc., para que sea interpretado por el navegador.
10. El navegador recibe la respuesta en formato de hipertexto, la interpreta y muestra la página web en pantalla.
11. Por último el servidor cierra la conexión *TCP/IP*.

Todos los pasos anteriores son seguidos de igual forma cada vez que solicitamos recursos al servidor web, ya que siempre la respuesta es enviada en formato de página web estática o semi-dinámica(puede volver a procesar datos). [WWW-34]

La única diferencia radica cuando solicitamos páginas web que son generadas por aplicaciones web. Como se mencionó en el capítulo anterior, la mayoría de páginas web, actualmente, son generadas por medio de aplicaciones web que se encargan de procesar datos, hacer cálculos, etc. En este caso el servidor web primero ejecuta los scripts o programas(*CGI*, *PHP*, *ASP*, *JSP*, etc.) que conforman la aplicación web que generará la página de respuesta al instante y luego se envía dicha página al navegador.

3.3 CARACTERÍSTICAS DE LOS SERVIDORES WEB

Algunos programas servidores web permiten configurar niveles de seguridad multicapa y administrarlos desde una PC. Se pueden fijar distintos niveles de acceso a diferentes departamentos y personas dentro y fuera de la organización.

El servidor web almacena y administra las páginas web. También recibe las solicitudes de

los clientes, las procesa y las contesta.

Según los estudios realizados por **ServerWatch**¹³, las características que se indican a continuación son las principales con las que debe contar un servidor web:

- Trabajar como Servidor Proxy
- Requerir autenticación de usuarios
- Escribir en múltiples logs
- Soportar *SNMP*
- Posibilidad de mantenimiento remoto
- Ser configurable
- Incluir sus propias *APIs*
- Contar con Servidor de Certificados
- Integrar motor de búsqueda
- Integrar lenguajes de script o módulos, tales como *PHP, ASP, JSP*, etc.
- Soportar *IPV6*
- Soportar *ISAPI, NSAPI*
- Soportar Servidores Virtuales
- Soportar *SSL, WebDAV*
- Trabajar como Servidor de Aplicaciones
- Personalizar páginas de error

[WWW-35]

En el **Anexo 4, Tabla 4**, que se encuentra en el CD adjunto a este documento en: CD_TESIS:\Tesis\Anexos_Tesis\Anexos.pdf, se realiza una evaluación que recolecta información sobre las características que cumplen algunos servidores web seleccionados.

3.4 IMPORTANCIA COMO COMPONENTE PRINCIPAL DE SITIOS Y APLICACIONES WEB

Como todos conocemos, Internet funciona las 24 horas del día, 365 días al año. Los sitios web deben brindar un servicio permanente el mismo tiempo, para asegurar su éxito y

¹³ ServerWatch es la empresa que se dedica a realizar estudios sobre servidores web. Ver www.serverwatch.com.

presencia en la gran red.

El componente fundamental para poder acceder a un sitio web todo el tiempo es el servidor web, ya que cualquier fallo o falta de disponibilidad de éste puede ocasionar que el sitio web deje de funcionar parcial o completamente. Los usuarios no podrán acceder ni solicitar los servicios de un sitio web hasta que el fallo sea arreglado.

Como podemos ver, el servidor web es fundamental para el correcto funcionamiento de un sitio web, razón por la cual la elección del software que hará el papel de servidor web es un aspecto de trascendental importancia y depende en gran medida de las necesidades concretas de cada caso.

En un principio los servidores web sólo se encargaban de almacenar y servir páginas web estáticas, a medida que las necesidades de Internet fueron cambiando, las páginas pasaron de ser estáticas a dinámicas. Hoy existe gran cantidad de servidores web que han incrementado sus funcionalidades y cubren esas necesidades.

Una de las funcionalidades y quizá la más importante es la capacidad de aceptar módulos agregados. Estos módulos sirven para que el servidor interprete un determinado lenguaje y por lo tanto ejecute aplicaciones web programadas en dicho lenguaje.

La tendencia de los sitios web es el dinamismo generado por las aplicaciones web, por eso es importante escoger un servidor web que cuente con la funcionalidad de interpretar lenguajes para servir páginas generadas mediante éstas aplicaciones.

A continuación una revisión de las alternativas de servidores web existentes en el mercado.

3.5 EVALUACIÓN DE LOS SERVIDORES WEB GRATUITOS Y COMERCIALES

El uso de software gratuito en Internet se incrementa cada vez más, éste tipo de software se presenta como una alternativa muy competitiva frente al software comercial. Los servidores web gratuitos no son la excepción, incluso las estadísticas muestran que están por encima de los servidores web comerciales.

Sin embargo, a la hora de elegir un servidor web, lo más importante no es sólo el costo,

más bien se deben tomar en cuenta otros aspectos como: la plataforma en la que va a operar, características, ventajas, desventajas, soporte técnico, implantación en el mercado y si cumple con las funcionalidades requeridas.

Para motivos de este estudio, se necesita que los servidores web a ser evaluados cumplan con ciertos requisitos, ya que si no cumplen, no tendría ningún objetivo la evaluación, los requisitos son:

- Trabajar sobre las plataformas: Windows o Linux.
- Servir páginas web de tipo: *PHP*, *ASP* o *JSP*.

A continuación, en la **Tabla 3.1**, se encuentran seis servidores web que cumplen con los requerimientos anteriores; se seleccionó dos servidores para cada tipo de página a servir (un gratuito y un comercial). De esta forma se evaluará si el aspecto del costo influye o no en el rendimiento del servidor.

Tabla 3.1 Servidores Web a evaluar

	TIPO DE PÁGINAS A SERVIR		
	PHP	ASP	JSP
GRATUITO	Apache Web Server	Personal Web Server	Tomcat Web Server
COMERCIAL	Zeus Web Server	Internet Information Server	iPlanet Web Server

En el **Anexo 4, Tabla 4**, que se encuentra en el CD adjunto a este documento en: CD_TESIS:\Tesis\Anexos_Tesis\Anexos.pdf, se realiza la evaluación de los seis servidores web. Luego de realizar la evaluación, se escogió tres servidores web, un servidor web para cada tipo de página (*PHP*, *ASP*, *JSP*) a servir. Cada servidor web escogido será estudiado en detalle más adelante. Los tres servidores web escogidos son:

- Apache Web Server para servir páginas *PHP*.
- Internet Information Server para servir páginas *ASP*.
- Tomcat Web Server para servir páginas *JSP*.

Las razones para escoger éstos servidores fueron:

- Según estudios realizados por www.serverwatch.com, ambos servidores, tanto Apache

Web Server como Zeus Web Server presentan igual rendimiento. Pero Apache tiene varias ventajas frente a Zeus, además de ser gratuito es más fácil de configurar, se adapta con mayor facilidad a *PHP*, es el servidor web más utilizado en Internet, tiene el respaldo de una comunidad de programadores y está en constante mejoramiento.

- Internet Information Server es la mejor alternativa para servir páginas *ASP*, debido a que son productos de una misma empresa. Es el segundo servidor web más usado en Internet. Su similar: Personal Web Server tiene funcionalidades reducidas y es mejor para sitios pequeños o personales. Según www.serverwatch.com, Internet Information Server tiene un excelente rendimiento.
- Tomcat Web Server es la implementación oficial para servir páginas *JSP* y *Servlets*, es gratuito, al igual que Apache, tiene el respaldo de una comunidad de programadores, está en constante desarrollo. Su oponente: iPlanet Web Server, aunque tiene un buen rendimiento no cuenta con una versión para Linux y es costoso.

3.5.1 APACHE WEB SERVER

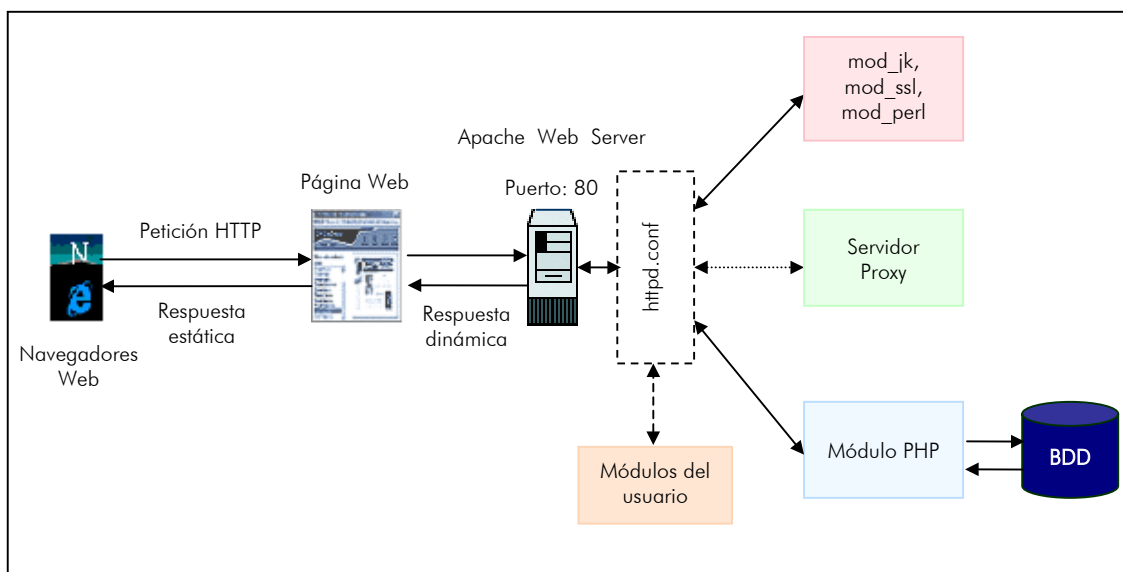


Fig. 3.3 Arquitectura de Apache Web Server

El servidor web Apache de Apache Software Foundation debe su nombre a su origen, ya que consiste en una versión parchada (**APAtCHy sErver**) del servidor del NCSA.

Es el producto más ampliamente conocido y de mayor éxito del Proyecto Apache, surgió de

la inquietud de un grupo de personas por desarrollar un servidor *HTTP* que incluyera y mejorara todas las características de los servidores web comerciales existentes hasta ese momento.

Con Unix en mente y dentro de la filosofía *Open Source*(la cual se caracteriza por la gratuidad de los productos), a principios de 1995, fue desarrollado un servidor web gratuito, robusto, potente, disponible para casi todas las plataformas y de código fuente abierto.

Al permitir el acceso al código fuente, quien tenga conocimientos puede ofrecerlos e integrarse al equipo de desarrollo para aportar con modificaciones que pueden distribuirse como módulos del servidor o como distribuciones comerciales. Dichas modificaciones contribuyen a la rápida corrección de errores, ampliación de funcionalidades y por lo tanto al desarrollo de nuevas y mejoradas versiones.

Este modelo de desarrollo constante es la causa principal del éxito de Apache, ya que se ha conseguido un software fiable, competitivo y de gran aceptación por parte de los profesionales.

CARACTERÍSTICAS DE APACHE WEB SERVER

- Es un servidor web potente, flexible y seguro, actualmente está en la versión 2.
- Proporciona todo el código fuente con licencia sin restricciones.
- Implementa los últimos protocolos, incluyendo el *HTTP/1.1*.
- Puede ser instalado en: Windows 9x/NT/ME/200x/XP, Novell Netware 5.x, OS/2, y muchas versiones de Unix, principalmente Linux.
- Se puede instalar mediante el código fuente, los archivos binarios precompilados o vía **RPM** en Linux; por medio de ejecutables en Windows.
- Es extensible gracias a módulos que agregan funcionalidad; dichos módulos se pueden cargar dinámicamente en el momento de la ejecución o compilarse estáticamente incorporándolos en los archivos binarios antes de instalarse.
- Puede trabajar con *APIs* que permiten a los usuarios avanzados construir y agregar nuevos módulos.
- Es actualizado y desarrollado constantemente por la comunidad de Internet.

- Las funciones de administración de Apache no van más allá de la edición de textos y de unas cuantas páginas web útiles que dan el estado del servidor. Para la administración se puede utilizar **Comanche**.
- La configuración y control del Apache se efectúa, casi exclusivamente, desde la línea de comandos o desde un editor de textos.
- Las versiones 1.3 y superiores utilizan un nuevo método de configuración llamado **APACI**, que reduce considerablemente el procedimiento de instalación y configuración.
- Cuenta con un amplio soporte que se puede obtener en línea.
- Puede trabajar con *CGI*, *FastCGI*, *SSI*, *PHP*, *PERL*, *JSP* y *Servlets*. Mediante el software *ChiliSoft ASP* se puede trabajar con *ASP* tanto en Linux como en Windows. Incorpora el lenguaje *PERL* y *PHP* como módulo.
- Escucha las peticiones de los clientes en el puerto 80.
- Admite configuración de servidores virtuales.
- Puede ser habilitado como servidor Proxy.
- Permite re-iniciación sin re-iniciar la máquina.
- Permite personalizar las páginas web de error.
- Proporciona seguridad mediante *SSL*, control de acceso y autenticación.

[LIB-04]

USO DE APACHE WEB SERVER

Según los estudios estadísticos, sobre servidores web, que realiza mensualmente **NetCraft**¹⁴, Apache es el servidor web más usado, se encarga de servir contenido a la mayor parte de los sitios web. Razón por la cual, Apache ha adquirido el prestigio de ser el servidor web más confiable que hay. Ver **Fig. 3.4**.

Actualmente está siendo utilizado por: Javasoft, The FBI, Financial Times, The Movies Database, W3(probablemente el sistema de hospedaje más grande del mundo, soporta 44,280 nombres de dominios con 39,905 direcciones *IP* diferentes y funciona con una versión personalizada de Apache).

¹⁴ Las estadísticas que realiza NetCraft sobre servidores web se encuentra en la dirección <http://www.netcraft.com/survey>.

De igual forma que NetCraft, Security Space también realiza estudios similares. Dichos estudios también ubican a Apache como el servidor web más usado. Ver Fig. 3.5.

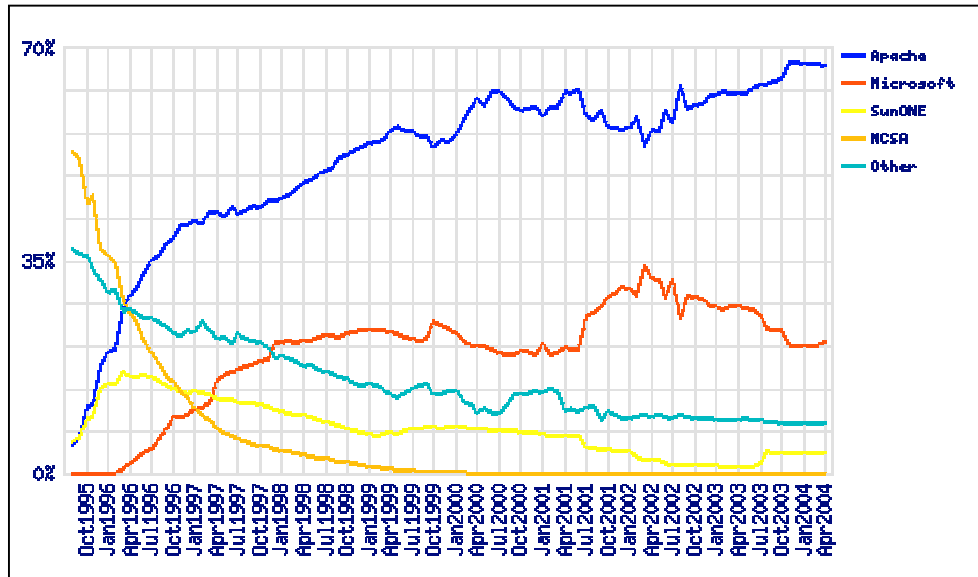


Fig. 3.4 Uso de servidores web desde Octubre-1995 hasta Abril-2004

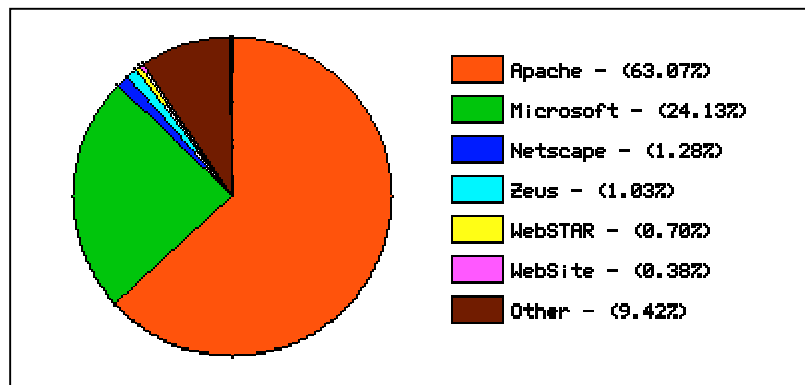


Fig. 3.5 Uso de servidores web en Oct-2003

INTEGRACIÓN CON PHP

Según un estudio realizado por www.php.net¹⁵, más de 9 millones de sitios web utilizan *PHP* para la generación dinámica de sus contenidos. Aunque *PHP* es multiplataforma y puede ser soportado por una gran variedad de sistemas operativos y servidores web, se integra de mejor manera con Linux y Apache.

¹⁵ En el sitio oficial de PHP: www.php.net/usage.php, se puede encontrar estadísticas sobre el uso de PHP.

Precisamente, la causa fundamental para que *PHP* se haya convertido en una herramienta tan popular, se debe a que es el complemento ideal para la combinación Linux – Apache. La agrupación de estas tres herramientas se ha vuelto tan exitosa que en la mayoría de versiones Linux, viene incluido *PHP* como parte de la instalación.

PHP también se puede obtener e instalar (como módulo o como *CGI* del servidor Apache) en forma independiente o como parte de distribuciones precompiladas que contienen Apache, *PHP* y una base de datos (MySQL o PostgreSQL). Estas distribuciones vienen listas para ser instaladas en Linux, Windows, Solaris, etc. y se utilizan cuando no se desea instalar y configurar cada herramienta por separado. Las distribuciones más conocidas son: AbriaSoft, NuSphere SQL, GreatBridge SQL.

Debido a que Apache y *PHP* son herramientas multiplataforma, se pueden integrar en casi cualquier sistema operativo. Elegir la forma de obtener e instalar *PHP*, depende entonces del sistema operativo seleccionado. En nuestro caso se ha seleccionado Linux y Windows para evaluar en cual de éstas plataformas se integran mejor Apache y *PHP*.

Los criterios para evaluar la integración de Apache y *PHP* son: instalación, configuración y facilidad de uso. En el **Anexo 1**, que se encuentra en el CD adjunto a este documento en: CD_TESIS:\Tesis\Anexos_Tesis\Anexos.pdf, se detalla la instalación, configuración e integración de *PHP* con Apache en Linux y en Windows.

3.5.2 INTERNET INFORMATION SERVER

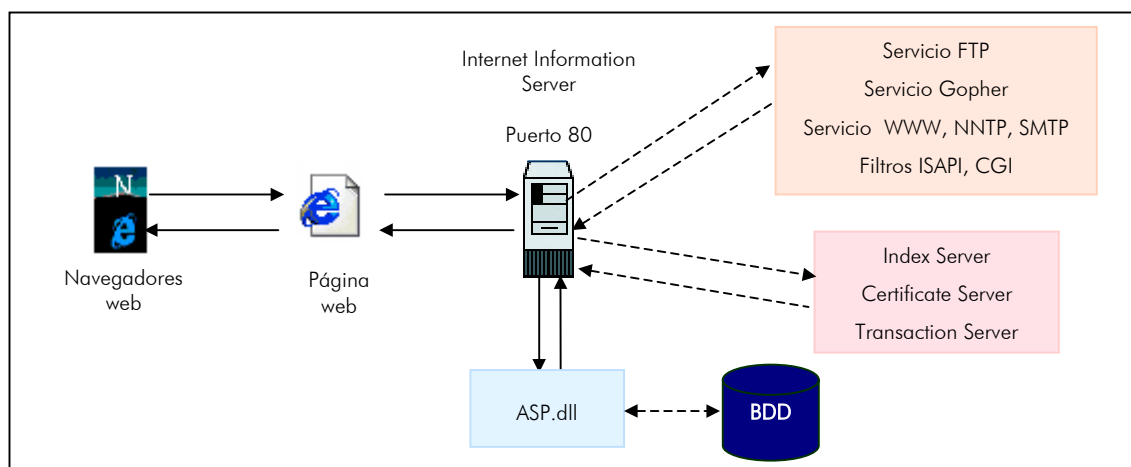


Fig. 3.6 Arquitectura de Internet Information Server

Internet Information Server es el servidor web de Microsoft Corporation. Es también uno de los mejores servidores web que existen. Es mucho más fácil de administrar y mantener que los servidores web para UNIX/Linux (tal como Apache) e igual de confiable y poderoso en desempeño.

Internet Information Server está orientado a la construcción de complicadas aplicaciones para Internet e Intranet y cuenta con una amplia variedad de herramientas para el desarrollo, monitoreo, análisis, mantenimiento, administración remota, etc.

Este servidor web viene incorporado en productos Microsoft como: Option Pack o en versiones Windows, por lo tanto su licencia es comercial y su costo depende del producto en el que se encuentre incorporado.

Internet Information Server fue diseñado especialmente para trabajar con tecnologías Microsoft, esta cualidad resulta beneficiosa y perjudicial a la vez. Beneficiosa puesto que al integrarse con productos de la misma empresa, pueden sacarse provecho mutuamente y perjudicial debido a que no puede integrarse con productos gratuitos, ni instalarse en Unix/Linux u otras plataformas, siendo una gran desventaja frente a otras alternativas.

CARACTERÍSTICAS DE INTERNET INFORMATION SERVER

- Es un servidor web de licencia comercial.
- Además de ser un servidor web, es servidor de *FTP* y *GOPHER*.
- Actualmente se encuentra en la versión 6.0.
- Soporta los protocolos: *HTTP*, *SMTP*, *NNTP*, *FTP* y *GOPHER*.
- Es fácil de instalar. La instalación y puesta en marcha es cuestión de minutos.
- Puede ser instalado sólo en plataformas Windows NT en adelante. Desde la versión 5.0 se instala como parte del sistema operativo Windows 2000 o como servicio de red.
- Trabaja como un servicio NT, por lo que se integra perfectamente con el resto de servicios.
- Utiliza una herramienta de configuración gráfica muy sencilla, que ayuda en las labores de administración del servidor. Dicha herramienta es la **MMC**.
- Permite la administración remota vía Internet, utilizando para ello un navegador.
- Está completamente integrado con el sistema de seguridad de Windows NT. Los

usuarios acceden a las páginas en función de sus permisos sobre ellas.

- Permite la publicación de páginas mediante herramientas web, como el FrontPage y similares, o mediante la simple copia de archivos sobre la red.
- Permite utilizar mecanismos de seguridad avanzados, como autenticación cifrada de NT.
- Proporciona seguridad mediante *SSL 3.0* y *TLS*.
- Permite integrar de un modo sencillo el servidor *SQL* de Microsoft, para crear consultas a las bases de datos.
- Escucha las peticiones de los clientes en el puerto 80.
- Soporta *ASP*, *PHP*, *JSP*, *CFML*, protocolos *CGI* e *ISAPI*, para añadir dinamismo a las páginas web.
- Se integra con herramientas complementarias tales como: *Index Server*, *Transaction Server*, *Certificate Server*, servicio *SMTP*, servicio *NNTP*.
- Tiene la capacidad de hospedar múltiples sitios web por medio de tres técnicas: asignando diferentes puertos, direcciones *IP* y nombres de dominio.
- Es posible la re-iniciación del servidor web sin re-iniciar la máquina.
- Permite personalizar los mensajes de error ocasionados por los clientes al solicitar páginas.

[WWW-36]

USO DE INTERNET INFORMATION SERVER

En las figuras 3.4 y 3.5 se puede apreciar que el uso de Internet Information Server se ubica en el segundo lugar. Esta ubicación responde a que la mayoría de sitios web en el mundo trabajan prácticamente bajo dos plataformas: Linux y Windows.

Los sitios que trabajan con Linux funcionan en su mayoría con Apache, mientras que los sitios que trabajan con Windows funcionan indiscutiblemente con Internet Information Server. El porcentaje de sitios que trabajan con Windows son evidentemente aquellos que prefieren utilizar software comercial y ser respaldados por una empresa reconocida como Microsoft.

INTEGRACIÓN CON ASP

ASP es la herramienta de generación de contenido dinámico diseñada especialmente para ser soportada por Internet Information Server. Al ser de la misma empresa, Internet Information Server es el servidor que mejor se integra con ASP.

ASP viene incluido en Microsoft Visual InterDev y en las últimas versiones de Windows 200x, XP, ME; en los dos casos se instala como módulo del servidor web, permitiendo una fácil integración.

En el **Anexo 2**, que se encuentra en el CD adjunto a este documento en: CD_TESIS:\Tesis\Anexos_Tesis\Anexos.pdf, se detalla la instalación, configuración y uso de ASP, para permitir la integración con Internet Information Server. Adicionalmente, en el mismo Anexo, se detalla la instalación de ASP en Linux, cabe destacar que en este caso el servidor web es Apache Web Server y se requiere del software *Chili!Soft ASP*.

3.5.3 TOMCAT WEB SERVER

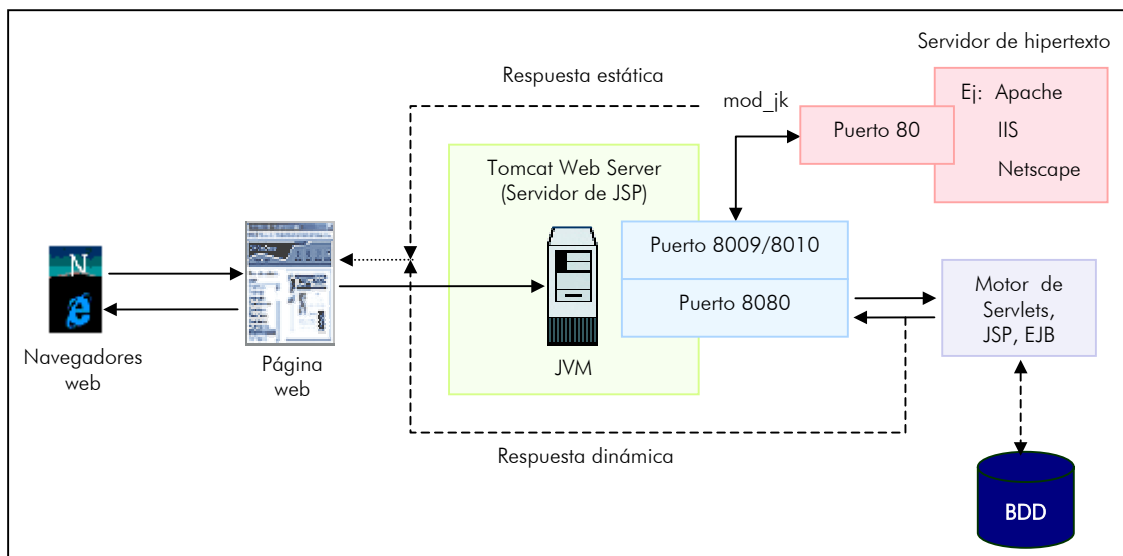


Fig. 3.7 Arquitectura de Tomcat Web Server

Apache Software Foundation es la compañía que lidera Jakarta, uno de los proyectos de código abierto más innovadores. Jakarta es un repositorio compuesto por 21 subproyectos que dan solución a problemas en particular. Entre esos subproyectos está el servidor web Tomcat, que es la herramienta más conocida del proyecto Jakarta. [WWW-37]

El servidor web Tomcat es en sí una aplicación web desarrollada en *Java*, creada para ejecutar *JSP* y *Servlets*, siendo así la implementación oficial de referencia de las especificaciones *JSP 1.2* y *Servlet 2.3*. Tomcat reemplaza al antiguo proyecto JServ que era un módulo para ser agregado a Apache Web Server, y que permitía ejecutar aplicaciones programadas con *JSP* y *Servlets*.

CARACTERÍSTICAS DE TOMCAT WEB SERVER

- Como todos los proyectos de Apache Software Foundation, Tomcat también es gratuito y de código abierto.
- Al ser parte del proyecto Jakarta, Tomcat se distribuye bajo la licencia: Apache Software Licence.
- Se puede instalar el servidor ya sea por medio del código fuente o mediante los archivos binarios precompilados.
- Actualmente se encuentra en la versión 5, desde la versión 4 también es conocido como Catalina.
- Tomcat puede trabajar como servidor web independiente o puede cooperar con otro servidor web, debido a que es lento para servir páginas web estáticas.
- Se ejecuta sobre una *JVM*, por defecto escucha los clientes en el puerto 8080, al cooperar con otro servidor web este puerto debe ser cambiado a 8009 u 8010.
- Se puede obtener de Internet el código fuente o los archivos binarios del servidor web para instalarlos.
- Puede ser instalado en: Linux, Windows NT, ME, 200x, XP y Solaris.
- La instalación se realiza mediante la línea de comandos en Linux o por medio de ejecutables en Windows.
- El servidor se configura por medio de archivos de texto.
- Al ser parte del proyecto Jakarta, se puede integrar con algunos de los otros subproyectos, para extender la funcionalidad. Se puede integrar por ejemplo con Xalan y James para procesar *XML* y enviar correo respectivamente.
- Soporta *JSP*, *EJB* y *Servlets* para programación de páginas dinámicas.

USO DE TOMCAT WEB SERVER

A diferencia de Apache Web Server e Internet Information Server, las estadísticas de uso de servidores web no posicionan a Tomcat en los primeros lugares. La principal razón se debe a que Tomcat no es tan utilizado como servidor independiente, ya que es lento para servir páginas web estáticas.

Para solucionar el problema de lentitud al servir páginas estáticas, Tomcat coopera, por medio del módulo denominado **mod_jk**, con otros servidores como: Apache Web Server, Internet Information Server y Netscape(actualmente iPlanet Web Server). Mientras que Tomcat se utiliza principalmente para desarrollo, depuración y ejecución de scripts *JSP* y *Servlets*, el servidor con el cual coopera se encarga de realizar el resto de funciones.

Motivo por el cual las estadísticas no reflejan el uso de Tomcat, sino más bien reflejan el uso del servidor con el que coopera. Apache Web Server es el servidor que mejor se adapta para trabajar con Tomcat.

Debido a que Tomcat coopera con los tres servidores web más utilizados en Internet: Apache Web Server, Internet Information Server y Netscape(ver **Fig. 3.4** y **Fig. 3.5**), su uso está incluido en un porcentaje del uso de cada uno de los tres servidores.

INTEGRACIÓN CON JSP

Al ser la implementación oficial para ejecutar y servir páginas *JSP* y *Servlets*, Tomcat es el servidor web más utilizado para soportar éstas herramientas de generación de contenido dinámico.

Tomcat, por ser escrito en *Java*, se adapta fácilmente a toda tecnología *Java*. Borland lo utiliza dentro de su ambiente de desarrollo JBuilder para permitir la programación de aplicaciones web utilizando ambas tecnologías.

La forma de instalar el servidor Tomcat, dependerá del tamaño y necesidades del sitio web. En sitios pequeños y personales, se puede utilizar Tomcat como servidor independiente, pero para sitios grandes y "reales", es recomendable utilizar Tomcat en cooperación con otro servidor web. Apache es el servidor más recomendado para trabajar con Tomcat. Al

ser *JSP*, *Servlets*, Tomcat y Apache, todas herramientas portables, se puede hablar de una combinación excelente para el desarrollo de aplicaciones web.

En el **Anexo 3**, que se encuentra en el CD adjunto a este documento en: CD_TESIS:\Tesis\Anexos_Tesis\Anexos.pdf, se detalla la instalación, configuración y uso de *JSP*, para permitir la integración con Tomcat en cooperación con Apache Web Server en Linux y Windows.

3.6 COMPARACIÓN DE LOS SERVIDORES WEB ESTUDIADOS

La comparación de los servidores: Apache Web Server, Internet Information Server y Tomcat, se centra en comparar características generales, ya que cada uno es eficiente en su entorno y fue desarrollado para servir páginas de un tipo específico. Apache es ideal para servir páginas *PHP*, Internet Information Server para servir páginas *ASP* y Tomcat para servir páginas *JSP*.

No tiene ningún objetivo, por ejemplo, comparar detalladamente Apache con Internet Information Server, debido a que Apache no soporta *ASP* directamente, hace falta instalar *Chili!Soft ASP* primero. Tampoco es posible comparar Internet Information Server con Tomcat porque éste último no soporta *ASP*. Aunque los tres servidores seleccionados soportan *JSP*, Tomcat es la implementación oficial.

Además escoger un servidor web depende en gran medida de la herramienta de generación de contenido dinámico y de la plataforma que se va a utilizar.

En los **Anexos 1, 2 y 3**, que se encuentran en el CD adjunto a este documento en: CD_TESIS:\Tesis\Anexos_Tesis\Anexos.pdf, se detalla información sobre las instalaciones, configuraciones y uso de cada servidor web; en el **Anexo 4, Tabla 4** del mismo CD, se realiza una evaluación general de los tres servidores; con la ayuda de dicha información se puede concluir que:

Apache Web Server es recomendable utilizarlo en los siguientes casos:

- Cuando se necesite trabajar en plataformas UNIX, especialmente en Linux ya que fue desarrollado especialmente para éste tipo de plataformas. Aunque se adapta sin

mayores inconvenientes en Windows, la mejor opción es instalarlo en Linux, ya sea por medio del administrador de paquetes *RPM* o mediante la compilación del código fuente. Además las versiones de Apache para Windows aunque son estables, no son seguras.

- Cuando se necesite añadir al servidor módulos específicos según las necesidades. Apache Web Server soporta la adición de módulos mediante la compilación del código fuente, o cargándoles dinámicamente después de la instalación de Apache.
- Cuando se vaya a trabajar e integrar con herramientas de código abierto tales como *PHP*, bases de datos PostgreSQL, MySQL, etc.
- Cuando se requiera abaratar costos de inversión.
- Cuando la herramienta de generación de contenido dinámico sea de preferencia *PHP*.

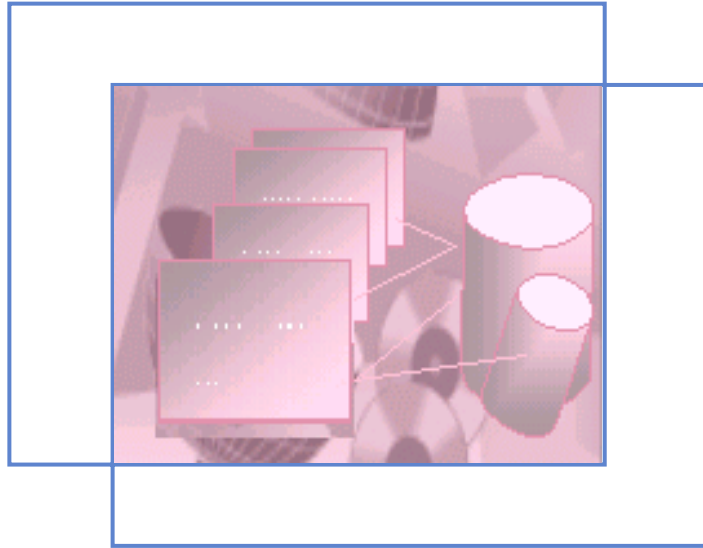
Internet Information Server es recomendable usar:

- Donde ya exista inversión en equipos, sistemas y aplicaciones desarrolladas en ambientes Microsoft.
- Cuando la herramienta de generación de contenido dinámico sea *ASP*, ya que es el servidor oficial para servir páginas de éste tipo.
- Cuando no se requiera migrar a otra plataforma. Aunque en la evaluación(**Anexo 4, Tabla 4** en: CD_TESIS:\Tesis\Anexos_Tesis\Anexos.pdf) se puede ver que Internet Information Server tiene buenas características, la dependencia de la plataforma lo limita mucho.

Tomcat Web Server es recomendable utilizar:

- Cuando el sitio no necesite gran capacidad de procesamiento, caso contrario deberá usarse conjuntamente con otro servidor(Apache Web Server más recomendado).
- Cuando las herramientas de generación de contenido dinámico sean *JSP*, *EJB* o *Servlets*, ya que Tomcat es el contenedor desarrollado específicamente para soportar éstas herramientas.
- En el caso de que existan aplicaciones desarrolladas anteriormente con herramientas como: *PHP*, Apache, *PERL*, en este caso no habrá problema al aumentar el servidor web Tomcat y otras herramientas de desarrollo tales como *JSP* o *Servlets*. Cada servidor se encargará de servir páginas de su propio entorno.

Capítulo 4



MOTOR DE BASE DE DATOS

- ❖ Introducción
- ❖ PostgreSQL
 - ❖ Características
 - ❖ Funcionamiento
 - ❖ Ventajas y desventajas
 - ❖ Conexión con PHP
 - ❖ Gestión de la base de datos PostgreSQL
- ❖ Introducción a ODBC
 - ❖ Características
 - ❖ Funcionamiento
 - ❖ Ventajas y desventajas
 - ❖ Conexión con ASP
- ❖ Introducción a JDBC
 - ❖ Características
 - ❖ Funcionamiento
 - ❖ Ventajas y desventajas
 - ❖ Conexión con JSP

INTRODUCCIÓN

El mundo de las Bases de Datos ha estado históricamente dominado por tres grandes compañías (Oracle, Informix e IBM) que ofrecen productos destinados a la gestión de grandes cantidades de información. Estos productos tienen unos precios normalmente altos para una empresa media y no hablemos para un usuario doméstico.

Dirigidos a un mercado medio encontramos varios de productos que pueden cubrir perfectamente las necesidades de cualquier empresa mediana sin hacer que el presupuesto se eleve.

Menos conocidos por la mayoría de usuarios, existen también varios Gestores de Bases de Datos gratuitos desarrollados generalmente por Universidades o por grupos de voluntarios. Algunos de estos productos tienen una gran calidad y en muchos casos superan las cualidades de sus competidores comerciales. Entre estos productos gratuitos y de gran calidad encontramos a PostgreSQL.

PostgreSQL comienza como un proyecto de investigación del grupo del Profesor Michael Stonebraker en Berkeley. Su proyecto original, Ingres, que se comercializa actualmente con el mismo nombre.

Su siguiente proyecto fue "Postgres" (post-ingres), que usaba un lenguaje de consulta más avanzado: **POSTQUEL**. En 1987, Postgres introdujo reglas, procedimientos, tipos definibles y conceptos de orientación a objetos. Postgres fue comercializado con el nombre de *Illustra* y forma parte actualmente del Universal Server de Informix.

Alumnos del laboratorio de Stonebraker, decidieron reescribir gran parte del código de Postgres y sustituir **POSTQUEL** por un subconjunto ampliado de *SQL*. Esta nueva versión fue lanzada en 1995 como Postgres95 y en ese momento el desarrollo pasó a manos de la comunidad de Internet. Postgres95 evolucionó hasta lo que es ahora PostgreSQL. [LIB-05]

En este capítulo se estudia PostgreSQL, debido a que es la base de datos que se conecta con las tres herramientas (PHP, ASP, JSP) comparadas en el capítulo 2. Además se hace una revisión de ODBC para ASP y JDBC para JSP, ya que por medio de los controladores ODBC y JDBC se conectan ASP y JSP, respectivamente, a la base de datos PostgreSQL.

4.1 POSTGRESQL

PostgreSQL se coloca en la categoría de las bases de datos conocidas como Objeto-Relacionales. Ha generado algunas características que son propias de las bases de datos orientadas a objetos y usa un subconjunto ampliado de SQL como lenguaje de consulta y está implementado siguiendo la arquitectura Cliente/Servidor. [WWW-38]

Ofrece gran variedad de herramientas y librerías para acceder a las Bases de Datos, y existen interfaces para los siguientes lenguajes:

- C, C++, C Embebido
- Java (a través de JDBC)
- PERL
- ODBC (existen controladores ODBC para PostgreSQL para Windows, Mac, UNIX)
- Python
- TCL
- PHP

4.1.1 CARACTERÍSTICAS DE POSTGRESQL

PostgreSQL proporciona un gran número de características que normalmente sólo se encontraban en las bases de datos comerciales tales como DB/2 u Oracle. Entre las principales características a partir de PostgreSQL 7.1.x. tenemos:

- **Directorio de almacenamiento:** PostgreSQL almacena los datos en el directorio /pgsql/data/base y a partir de ahí un directorio para cada base de datos.
- **Manejo de Tablas:** Cada tabla es un archivo, así como los índices.
- **Tablas del Sistema:** Los nombres de las tablas propias del sistema llevan el prefijo pg_.
- **Información de Creación:** El archivo PG_VERSION (presente en cada base) contiene la versión mayor con la que fue creada la base.
- **RespalDOS:** PostgreSQL posee las herramientas pg_dump y pg_dumpall que nos ayudan tanto como para respaldar las tablas, como para copiarlas de un sistema a otro en un formato transportable. En algunos casos, al actualizar la versión de PostgreSQL será necesario primero respaldar las tablas con estas herramientas, para posteriormente

volverlas a cargar.

- **Herramienta para trabajar en modo de línea de comandos:** La herramienta completa para poder manipular las bases de datos es **psql**.
- **Licencia BSD:** No impone condiciones sobre lo que el usuario puede hacer con el código fuente.
- **DBMS Objeto – Relacional:** PostgreSQL aproxima los datos a un modelo objeto - relacional, y es capaz de manejar rutinas, reglas, restricciones (constraints), disparadores (triggers). Ejemplos de su avanzada funcionalidad son consultas SQL declarativas, control de concurrencia multi – versión (el cual permite a los accesos de sólo lectura continuar leyendo datos consistentes durante la actualización de registros y permite copias de seguridad desde `pg_dump` mientras la base de datos permanece disponible para consultas), soporte multi – usuario, transacciones, claves foráneas, replicación, optimización de consultas, herencia y arrays.
- **Altamente Extensible:** PostgreSQL soporta operadores, funciones, métodos de acceso y tipos de datos definidos por el usuario.
- **Soporte SQL Comprensivo:** PostgreSQL soporta la especificación SQL99 e incluye características avanzadas tales como las uniones (joins) SQL92.
- **Integridad Referencial:** PostgreSQL soporta integridad referencial, la cual es utilizada para garantizar la validez de los datos de la base de datos.
- **API Flexible:** La flexibilidad del API de PostgreSQL ha permitido fácilmente proporcionar soporte al desarrollo para el **RDBMS** PostgreSQL. Estas interfaces incluyen Object Pascal, Python, PERL, PHP, ODBC, Java/JDBC, Ruby, TCL, C/C++ y Pike.
- **Lenguajes Procedurales:** PostgreSQL tiene soporte para lenguajes procedurales internos, incluyendo un lenguaje nativo denominado PL/pgSQL. Este lenguaje es comparable al lenguaje procedural de Oracle, PL/SQL.
- **MVCC:** MVCC o Control de Concurrencia Multi-Versión es la tecnología que PostgreSQL usa para evitar bloqueos innecesarios. En algunas DBMS con capacidades SQL, tal como MySQL o Access, hay ocasiones en las que una lectura tiene que esperar para acceder a información de la base de datos. La espera está provocada por usuarios que están escribiendo en la base de datos. Por lo tanto, el lector está bloqueado por los escritores que están actualizando registros.

Mediante el uso de MVCC, PostgreSQL evita este problema por completo. MVCC está

considerado mejor que el bloqueo a nivel de fila porque un lector nunca es bloqueado por un escritor. En su lugar, PostgreSQL mantiene una ruta a todas las transacciones realizadas por los usuarios de la base de datos. PostgreSQL es capaz entonces de manejar los registros sin necesidad de que los usuarios tengan que esperar a que los registros estén disponibles.

- **Cliente/Servidor:** PostgreSQL usa una arquitectura proceso-por-usuario Cliente/Servidor. Hay un proceso maestro que se ramifica para proporcionar conexiones adicionales para cada cliente que intente conectar a PostgreSQL.
- **Write Ahead Logging (WAL):** La característica de PostgreSQL conocida como Write Ahead Logging incrementa la dependencia de la base de datos al registro de cambios antes de que éstos sean escritos en la base de datos. Esto garantiza que en el hipotético caso de que la base de datos se caiga, existirá un registro de las transacciones a partir del cual podremos restaurar la base de datos. Esto puede ser enormemente beneficioso en el caso de caída, ya que cualquier cambio que no fue escrito en la base de datos puede ser recuperado usando el dato que fue previamente registrado. Una vez que el sistema ha quedado restaurado, un usuario puede continuar trabajando desde el punto en que lo dejó cuando cayó la base de datos.[WWW-39]
- **Multiplataforma:** Además puede ser integrada al ambiente Windows permitiendo mantener y desarrollar aplicaciones en esta plataforma, ya que permite desarrollar o migrar aplicaciones desde Access, Visual Basic, Foxpro, Visual Foxpro, C/C++, Delphi, etc, para que utilicen PostgreSQL como servidor de base de datos. [WWW-40]
- **Tipos de Datos:** Los tipos de datos que soporta PostgreSQL se muestran en la **Tabla 4.1.**

Tabla 4.1 Tipos de Datos de PostgreSQL

CATEGORÍA	TIPO	DESCRIPCIÓN
Character string	TEXT VARCHAR(<i>length</i>) CHAR(<i>length</i>)	Variable storage length Variable storage length with maximum length Fixed storage length, blank-padded to length
Number	INTEGER INT2 INT8 OID NUMERIC(<i>precision</i> , <i>decimal</i>) FLOAT FLOAT4	Integer, +- 2 billion range, internally INT4 Integer, +- 32 thousand range Integer, +-4 x 10 ¹⁸ range Object identifier Number, user-defined precision and decimal location Floating-point number, 15-digit precision, internally FLOAT8 Floating-point number, 6-digit precision
Temporal	DATE TIME TIMESTAMP	Date Time Date and Time

	INTERVAL	Interval of Time
Logical	BOOLEAN	Boolean, <i>True</i> or <i>False</i>
Geometric	POINT LSEG PATH BOX CIRCLE POLYGON	Point Line segment List of points Rectangle Circle Polygon
Network	INET CIDR MACADDR	IP address with optional netmask IP network address Ethernet MAC address

4.1.2 FUNCIONAMIENTO DE POSTGRESQL

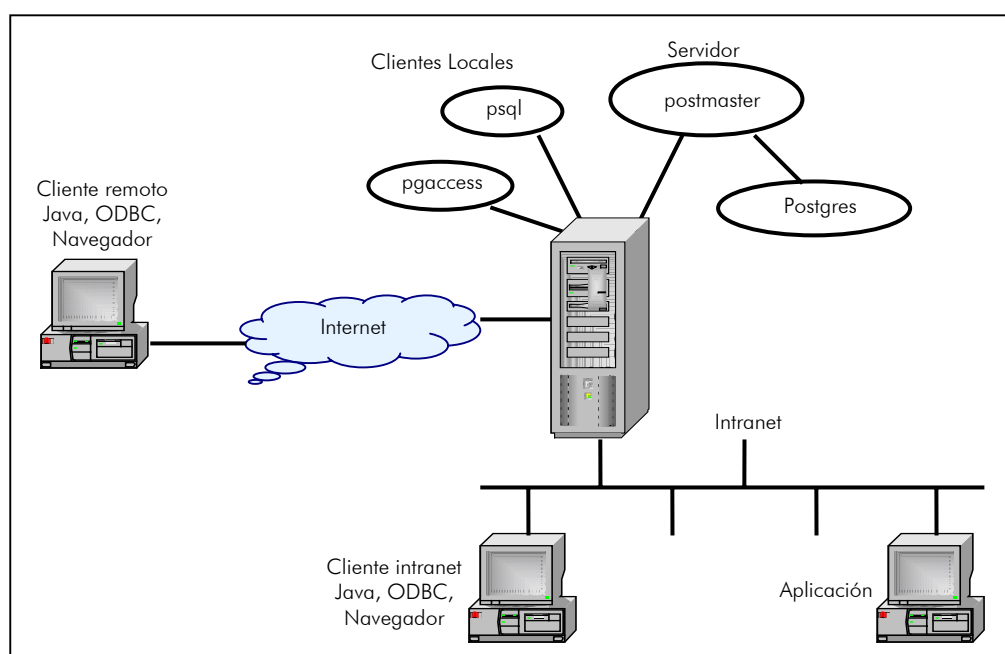


Fig. 4.1 Funcionamiento de PostgreSQL

PostgreSQL usa un modelo Cliente/Servidor conocido como "proceso por usuario". Una sesión PostgreSQL consiste en los siguientes procesos cooperativos de Unix (programas), ver Fig. 4.1:

- Un proceso demonio supervisor (**postmaster**¹⁶).

¹⁶ Postmaster es el proceso supervisor que actúa como una puerta de control para las peticiones al sistema PostgreSQL.

- Una aplicación sobre la que trabaja el usuario (**frontend**¹⁷: el programa psql).
- Uno o más servidores de bases de datos en segundo plano (el mismo proceso postgres).

Un único postmaster controla una colección de bases de datos dadas en un único host. Debido a esto una colección de bases de datos se suele llamar una instalación o un sitio. Las aplicaciones que quieren acceder a una determinada base de datos dentro de una instalación hacen llamadas a la librería **libpq**, la librería libpq envía peticiones de usuario a través de la red al postmaster, éste en respuesta inicia un nuevo proceso en el servidor (**backend**¹⁸) y conecta el proceso de frontend al nuevo servidor. A partir de este punto, el proceso de frontend y el servidor en backend se comunican sin la intervención del postmaster. Aunque, el postmaster siempre se está ejecutando, esperando peticiones, tanto los procesos de frontend como los de backend vienen y se van.

La librería libpq permite a un único proceso en frontend realizar múltiples conexiones a procesos en backend. Aunque, la aplicación frontend todavía es un proceso en un único hilo. Conexiones multihilo entre el frontend y el backend no están soportadas de momento en libpq. Una implicación de esta arquitectura es que el postmaster y el proceso backend siempre se ejecutan en la misma máquina (el servidor de base de datos), mientras que la aplicación en frontend puede ejecutarse desde cualquier sitio. Se debe tener esto en cuenta, porque los archivos que pueden ser accedidos en la máquina del cliente pueden no ser accesibles (o sólo pueden ser accedidos usando un nombre de archivo diferente) en la máquina del servidor de base de datos.

Los servicios postmaster y postgres se ejecutan con el identificador del "**superusuario postgres**"¹⁹ el mismo que no necesita ser un usuario especial, tampoco el superusuario del sistema. En cualquier caso, todos los archivos relacionados con la base de datos deben pertenecer al superusuario postgres. [WWW-41]

¹⁷ Aplicación que se conecta al postmaster y mantiene registros de los errores del sistema y de la comunicación entre los procesos backend.

¹⁸ Backend es el programa ejecutable Postgres real.

¹⁹ Un usuario que tiene todos los permisos, en todas las bases de datos, incluyendo la capacidad de crear a otros usuarios.

PSQL

Es una aplicación que sirve como cliente, monitor y frontend hacia el motor de PostgreSQL. Permite teclear comandos SQL interactivamente, enviarlos a PostgreSQL y visualizar los resultados de consultas, inserciones, actualizaciones, eliminaciones, etc., de datos.

Este programa cuenta con ayuda en línea, por medio de la instrucción `\?`, para los comandos de psql (comandos internos) y `\h` para examinar la sintaxis de las instrucciones SQL.

Para acceder a una base de datos a través de psql, en el prontuario de una consola se debe digitar, ver **Fig. 4.2**:

```
$psql nombre_base_de_datos
```

Fig. 4.2 Comando psql para modo texto

La siguiente pantalla, ver **Fig. 4.3**, es el resultado de la ejecución de psql.

```
$ psql test
Welcome to psql, the PostgreSQL interactive terminal.

Type: \copyright for distribution terms
      \h for help with SQL commands
      \? for help on internal slash commands
      \g or terminate with semicolon to execute query
      \q to quit

test->
```

Fig. 4.3 Consola de psql

Este prontuario indica que el monitor de psql está listo y se puede escribir SQL dentro de este espacio de trabajo.[WWW-42][WWW-43]

PGACCESS

Pgaccess es la herramienta gráfica de PostgreSQL usada para acceder a tablas, consultas, vistas, secuencias, funciones, reportes.

Para acceder a una base de datos a través de Pgaccess, en el prontuario de una consola se

debe digitar, ver **Fig. 4.4**:

```
$pgaccess nombre_base_de_datos
```

Fig. 4.4 Comando pgaccess para modo gráfico

Se obtiene una ventana similar a la de Microsoft Access. Ver **Fig. 4.5**:

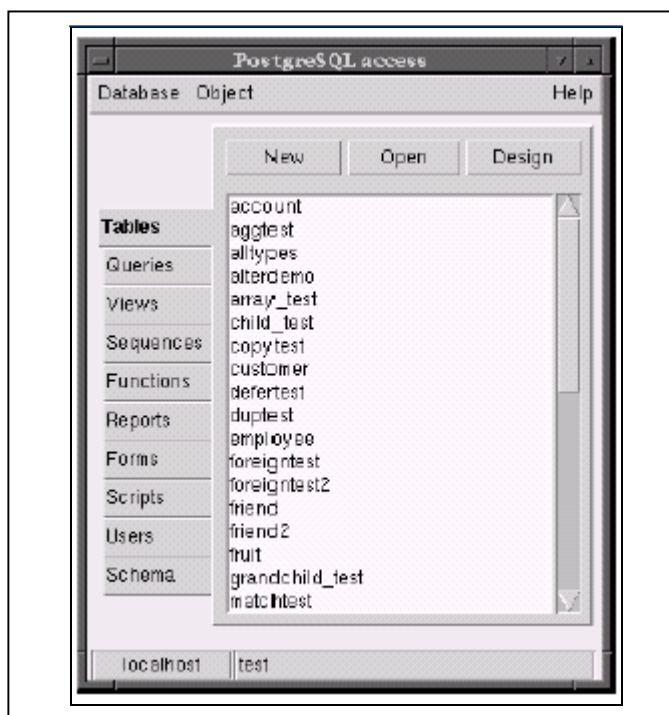


Fig. 4.5 Pantalla de pgaccess

Con Pgaccess se puede [WWW-44]:

- Abrir cualquier base de datos en un host, especificando: el puerto, nombre de usuario y contraseña. El puerto por defecto para conectarse a PostgreSQL es 5432.
- Ejecutar **VACUUM**²⁰.
- Guardar las preferencias en el archivo ~/.pgaccessrc.
- Abrir tablas múltiples para ver.
- Redimensionar columnas.
- Seleccionar el texto en celdas.

²⁰ VACUUM es el comando que limpia y analiza una base de datos.

- Ajustar la altura de las filas cuando se edita.
- Guardar el diseño de las tablas.
- Editar texto que se quiere cambiar.
- Borrar registros presionando la tecla Delete o Suprimir.

4.1.3 VENTAJAS DE POSTGRESQL

- PostgreSQL es un proyecto de código abierto, se puede obtener el código fuente, usar el programa y modificarlo libremente para adaptarlo según las necesidades particulares.
- Usa un lenguaje relacional que se basa en el modelo de datos relacional.
- Por su arquitectura de diseño, escala muy bien al aumentar el número de CPUs y la cantidad de RAM.
- Soporta transacciones y desde la versión 7.0 claves foráneas (con comprobaciones de integridad referencial).
- Tiene mejor soporte para triggers y procedimientos en el servidor. Soporta un subconjunto de SQL92 mayor que el que soportan otros motores como MySQL.
- Tiene ciertas características orientadas a objetos.
- Es multiplataforma: Linux, Solaris, HP-UX, AIX, IRIX, FreeBSD, OpenBSD, NetBSD, MacOs, SCO OpenServer, SCO Unixware, BeOS, BSDI, Compaq Tru64, QNX, Windows 9x/NT/ME/200x/XP, permitiendo de esta manera a los desarrolladores, generar nuevas aplicaciones o mantener las ya existentes. Cabe destacar que en sistemas Windows se necesita del software *Cygwin* y *CygIPC*.
- PostgreSQL puede usar *PERL*, *PHP*, *Python* o *TCL* como lenguaje procedural embebido.
- Posee herramientas para sacar respaldos.
- Permite desarrollar o migrar aplicaciones desde Access, Visual Basic, Foxpro, Visual Foxpro, C/C++ Visual C/C++, Delphi, etc., para que utilicen a PostgreSQL como servidor de Base de Datos.

4.1.4 DESVENTAJAS DE POSTGRESQL

- Consume más recursos y carga más el sistema que otras bases de datos similares.
- Límite del tamaño de cada fila de las tablas a 8KB (se puede ampliar a 32KB)

recompilando, pero con un coste añadido en el rendimiento).

- Menos funciones en *PHP*, que es uno de los lenguajes con el que más trabaja.
- Uno de los principios del modelo relacional es que los atributos de una relación son atómicos. PostgreSQL no tiene esta restricción. [WWW-45]

4.1.5 CONEXIÓN CON PHP

Para conectarse a una base de datos en PostgreSQL con *PHP* se requiere de la siguiente función, ver **Fig. 4.6**:

```
pg_connect (string host, string port, string options, string tty, string bdd) //Conectarse de forma normal o
pg_pconnect (string host, string port, string options, string tty, string bdd) //Conectarse persistentemente
```

Fig. 4.6 Función para conectarse a la Base de Datos

A esta función se le puede dar el siguiente uso, asignado a una variable *PHP*, ver **Fig. 4.7** y **Fig. 4.8**:

```
<?php //Abrir PHP
$conexion = pg_connect("localhost", "", "", "nombre_bdd"); //Abrir la conexión

( conjunto de operaciones sobre la conexión ) //Operaciones

pg_close($conexion); //Cerrar la conexión
?> //Cerrar PHP
```

Fig. 4.7 Conectarse a una Base de Datos

```
<?php //Abrir PHP
$conexion = pg_connect("localhost", "", "", "nombre_bdd"); //Abrir la conexión
$sql = "SELECT * FROM comunidades;"; //Ejecutar SQL
$resultado = pg_exec($conexion,$sql_select); //Recordset
$filas_resultado = pg_num_rows($resultado); //Operaciones
$filas=0;
while($filas<$filas_resultado){
    list($comunidad, $mision)=pg_fetch_row($resultado, $filas); //Recorrer Recordset
    echo "Comunidad:". $comunidad." " . "Misión:". $mision;
}
pg_close($conexion); //Cerrar la conexión
?> //Cerrar PHP
```

Fig. 4.8 Consultar los datos de una Tabla

4.1.6 GESTIÓN DE UNA BASE DE DATOS POSTGRESQL

Si el programa postmaster de PostgreSQL está cargado y en ejecución, podemos realizar las siguientes tareas:

Crear bases de datos

```
$createdb nombre_base_de_datos
```

Fig. 4.9 Crear una Base de Datos

Ya creada la base de datos (Fig. 4.9) se puede optimizar el tiempo con la ejecución de scripts que contengan el código SQL de creación de tablas, ver Fig. 4.10:

```
$psql nombre_base_de_datos < script_crear_tablas.sql
```

Fig. 4.10 Ejecutar script para crear Tablas

De igual forma se puede hacer respaldos de la base de datos existente con, ver Fig. 4.11:

```
$pg_dump nombre_base_de_datos > script_respaldo.sql
```

Fig. 4.11 Respaldo de una Base de Datos en script

Eliminar bases de datos

Para eliminar una base de datos se utiliza el siguiente comando, ver Fig. 4.12:

```
$dropdb nombre_base_de_datos
```

Fig. 4.12 Eliminación de una Base de Datos

Comandos PSQL

Los comandos de psql de mayor uso para la gestión de bases de datos se muestran en la Tabla 4.2:

Tabla 4.2 Comandos de psql

COMANDO	DESCRIPCIÓN
\h	Ayuda de comandos SQL

\?	Ayuda de comandos internos de psql
\dS	Lista las tablas del sistema
\d nombre_tabla	Muestra la estructura de la tabla especificada
\d	Lista tablas, vistas, secuencias, índices de la base de datos actual
\ds	Lista sólo las secuencias de la base de datos actual
\df	Lista funciones de la base de datos actual
\dt	Lista todas las tablas de la base de datos actual
\di	Lista sólo los índices de la base de datos actual
\dv	Lista sólo las vistas de la base de datos actual
\do	Lista los operadores
\g	Ejecuta una instrucción SQL, equivale al punto y coma(;)
\l	Lista todas las bases de datos existentes
\lo_list	Lista los large objects de la base de datos actual
\r	Resetea el buffer de la consulta actual
\q	Salir de psql

Comandos SQL

Los comandos *SQL* para gestionar bases de datos son: *select*, *insert*, *update*, *delete*, *inner join*, etc. cuya sintaxis se puede encontrar con facilidad en cualquier manual *SQL* o de base de datos. Dentro de *psql* se encuentra la ayuda de la sintaxis de los comandos *SQL* con: **\h comando_sql;**

4.2 INTRODUCCIÓN A ODBC

ODBC es un intermediario entre bases de datos y aplicaciones, cuya tarea es sostener una conversación de preguntas y respuestas entre dos aplicaciones que no hablan el mismo idioma y que gestionan sus recursos de forma diferente.

ODBC está basado en *SQL* como un estándar de acceso a datos. Esta interfaz provee máxima interoperabilidad: una aplicación sencilla puede acceder a diferentes sistemas manejadores de bases de datos *SQL* a través de un conjunto de código común. Estos habilitan un diseño para construir y distribuir una aplicación Cliente/Servidor sin designar un *DBMS* específico.

Los controladores de la base de datos son entonces quienes crean el enlace de la aplicación a la base de datos.

4.2.1 CARACTERÍSTICAS DE ODBC

- Crea punteros de conexión, especificando un origen de datos y un controlador.
- Efectúa conexiones con la base de datos (abriendo un socket).
- Crea punteros de peticiones *SQL*.
- Ejecuta peticiones sobre la base de datos.
- Recoge resultados de las peticiones.
- Libera y cierra todos los recursos previamente asignados.
- Se comunica casi con cualquier base de datos, mediante una interfaz común que se conoce como controlador *ODBC*, este controlador contiene código que entiende las características específicas de una base de datos en particular y proporciona acceso a ella mediante un conjunto estándar de llamadas *API*.
- Es una interfaz que permite la flexibilidad y compatibilidad total entre casi cualquier base de datos y una aplicación, generalmente con sólo cambiar el controlador *ODBC*.
- Cada base de datos tiene su controlador, un controlador *ODBC* es una biblioteca que entiende como comunicarse con la base de datos. Generalmente, los controladores *ODBC* son ofrecidos por el proveedor de la base de datos o compañía de software independiente. [WWW-46][WWW-47]

4.2.2 FUNCIONAMIENTO DE ODBC

ODBC es un API de interfaz entre clientes de bases de datos y servidores de bases de datos. La figura 4.13 ilustra este esquema:

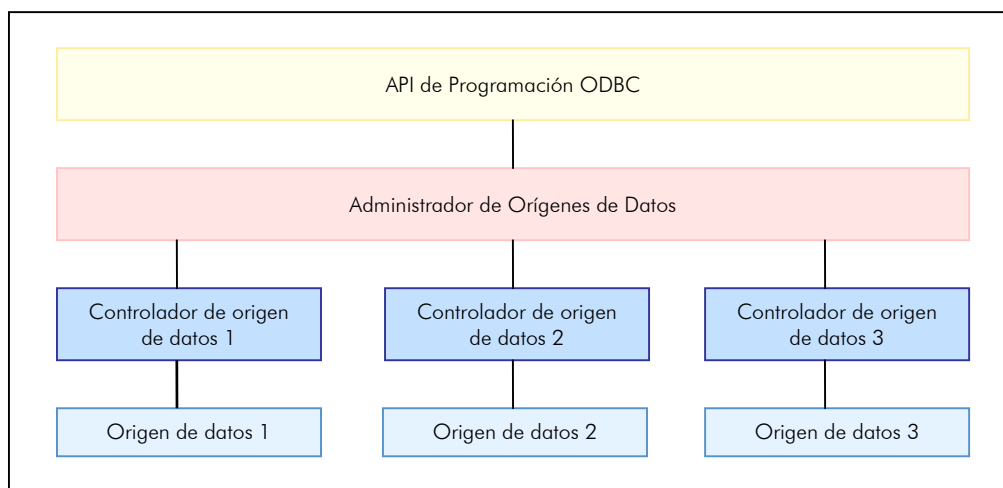


Fig. 4.13 Funcionamiento de ODBC

La primera capa constituye la librería del API que utilizan las diversas aplicaciones que "hablan" ODBC.

Nota: Microsoft proporciona para sus sistemas el fichero ODBC32.DLL, que contiene el API y la interfaz con el sistema operativo, permitiendo a los desarrolladores de controladores ODBC incluir dicha librería en sus distribuciones. Para sistemas UNIX, se ha desarrollado su propia librería GPL libodbc.so.x.x que es totalmente compatible con las especificaciones descritas por Microsoft.

El administrador de orígenes de datos es el responsable de dar las rutas de peticiones de ODBC desde la librería hasta los controladores. Para ello se diferencian tres tipos de orígenes de datos: de usuario, de archivo y de sistema.

Los orígenes de datos de usuario, se refieren a las operaciones que realiza el usuario con su base de datos desde la aplicación nativa para la que han sido desarrollados, y sin realizar ningún tipo de compartición con otros usuarios. Por ejemplo: cuando se trabaja con MS-Access y no se comparte la base de datos.

Cuando se comparte la base de datos mediante un servidor de ficheros, compartiendo físicamente los datos almacenados en un fichero determinado, se habla de orígenes de

datos de archivo. Este método permite, por ejemplo a un usuario de dBase manejar una base de datos de MS-Access, o bien que varios usuarios puedan compartir una misma base de datos.

Cuando no se comparte un fichero, sino que se trabaja con la base de datos a través de un sistema Cliente/Servidor, se habla de un origen de datos de sistema.

Por último, cada origen de datos tiene asociado un controlador, que actúa de "pasarela" entre la *API* y el acceso físico a los datos.

4.2.3 VENTAJAS DE ODBC

- Se podría decir que existe un "acuerdo" entre Microsoft y los fabricantes de software para Windows: "Los que hacen software no específico para bases de datos, enseñen a sus aplicaciones a comunicarse con el *ODBC*; y los fabricantes de bases de datos, hagan controladores de sus sistemas para ponerlos en el *ODBC*, si quieren que otras aplicaciones puedan acceder a su información".
- Soluciona el problema que existe al conectar sistemas en diferente plataforma, aplicación y base de datos.
- Es flexible para aceptar configuraciones diferentes de conectividad. Es decir, que la misma aplicación del desarrollador pudiese acceder a varios *DBMS*.

4.2.4 DESVENTAJAS DE ODBC

- Mientras se trabaje en Microsoft se debe usar por ciertas incompatibilidades entre aplicaciones y bases de datos.
- Si no hubiera el controlador, se dificultaría trabajar con la base de datos.
- *ODBC* se instala manualmente en cada máquina.
- La *API ODBC* no es apropiada utilizarla directamente con *Java* porque utiliza una interfaz desarrollada en C. Las llamadas desde *Java* a código nativo C tienen una serie de inconvenientes desde el punto de vista de la seguridad, implementación, robustez y portabilidad.

4.2.5 CONEXIÓN CON ASP

Creación de Recursos

Para conectarse a un servidor de base de datos, se necesita crear un Recurso de Sistema que establezca la comunicación Cliente/Servidor, este recurso se denomina **DSN**.

***Nota:** DSN es el nombre lógico usado por el ODBC para referirse a la unidad y otra información necesaria para tener acceso a los datos.*

En Windows el **DSN** es creado en el servidor mediante el programa de administración de **ODBC**, ubicado en el Panel de Control.

La información que se guarda en el **DSN**, se describe a continuación:

- **DSN:** Nombre del Recurso.
- **Description:** Una breve descripción del recurso.
- **Server:** Nombre del Servidor de Bases de Datos en la Red.
- **Network Address:** Dirección en la cual se encuentra la base de datos desde la que los controladores van a recuperar información, usualmente se debe a su valor por defecto.
- **Network Library:** Bibliotecas dinámicas *DLL* que utilizará el servidor de base de datos para comunicarse con los clientes, se configura automáticamente.
- **Use Trusted Connection:** Establece el tipo de conexión, si es de confianza (Trusted), el servidor utiliza como login el nombre de usuario y password autenticados por Windows 2000 Server.
- **Database Name:** Nombre de la base de datos a la cual se conectará por defecto.
- **Language Name:** Lenguaje nacional utilizado por el servidor de base de datos.

Todas estas propiedades deben ser establecidas al crear un recurso en el servidor. Los recursos están establecidos por el proveedor de software.

Conexión a un recurso ODBC para Servidores Windows

Para la conexión se trabaja con objetos, los cuales permiten establecer la comunicación entre el cliente y el servidor de bases de datos.

El Objeto Connection es el que permite conectarse a la base de datos, este objeto tiene una

serie de colecciones, propiedades y métodos propios que permiten manipular las operaciones con la base de datos.

Otro objeto importante es el Objeto Recordset, mediante el cual se realiza las operaciones sobre las tablas de la base de datos.

Los recursos *ADO*, son útiles para manejar la conexión, los objetos mencionados anteriormente pertenecen a este modelo, y la manera de utilizar este modelo en el código ASP es, ver **Fig. 4.14** y **Fig. 4.15**:

```
<%
Set conn=Server.CreateObject("ADODB.Connection") //Abrir ASP
conn.Open "Personal" //Crear el objeto conexión
// Abrir conexión ODBC al DSN

( conjunto de operaciones sobre la conexión ) //Operaciones

conn.close //Cerrar la conexión
%> //Cerrar ASP
```

Fig. 4.14 Conectarse a un DSN

```
<%
Set conn = Server.CreateObject("ADODB.Connection") //Abrir ASP
Set RS = Server.CreateObject("ADODB.Recordset") //Crear el objeto recordset
conn.Open "Personal" //Crear el objeto recordset
// Abrir conexión ODBC al DSN
sql = " SELECT * FROM comunidad" //Ejecutar SQL
RS.Open sql, conn //Almacenar el resultado

( conjunto de operaciones con el recordset ) //Operaciones

RS.close //Cerrar el recordset
conn.close //Cerrar la conexión
%> //Cerrar ASP
```

Fig. 4.15 Consultar los datos de una Tabla

En contraparte, para Linux, el proyecto FreeODBC ha desarrollado una librería denominada *iODBC*, que cumple con las especificaciones del *API ODBC 2.0* de Microsoft y que integra las funciones de *API* y de administrador de orígenes de datos.

Existe un fichero `/${HOME}/.iodbc.ini`, que indica a la librería `libodbc.so.x.x` los controladores con los que dispone cada sistema y cómo se accede a ellos. Toda la interfaz está implementada mediante librerías dinámicas. El resultado de todo esto, es que el

programador se encuentra con una *API* virtualmente idéntica a la que se encontraría si estuviera trabajando en una máquina con Windows.

4.3 INTRODUCCIÓN A JDBC

JDBC es una *API*, esto quiere decir que es un conjunto de objetos y funciones para que los programadores puedan integrar servicios de *DBMS* dentro de sus aplicaciones: programas modo consola, *JSP*, *Servlets* y *Applets* creados en *Java*.

JDBC existe desde 1996, y fue incluido desde la versión 1.1 del *JDK*, aunque también se hizo disponible para usarse con el *JDK* 1.02.

Sun Microsystems optó por crear una nueva *API*, en lugar de utilizar *APIs* ya existentes como *ODBC*, con la intención de obviar los problemas que presenta el uso desde *Java* de éstas *APIs*, que suelen ser de muy bajo nivel y utilizar características no soportadas directamente por *Java*, como punteros.

4.3.1 CARACTERÍSTICAS DE JDBC

- Fácil acceso al conjunto de la información, incluso en entornos heterogéneos.
- Aumenta la facilidad de programación.
- Permite utilizar órdenes *Java* para actualizar bases de datos.
- Nuevos tipos de datos.
- *JDBC* está implementado como un conjunto de clases e interfaces, donde sólo se definen los nombres y cada fabricante del controlador *JDBC* se encarga de implementar el cuerpo de dichas clases e interfaces.
- Mejora del rendimiento (manipulación de grandes objetos, como *BLOB* y *CLOB*, sin traerlos de la base de datos).
- Actualizaciones en grupo.
- Soporte *JNDI*. Independencia del controlador *JDBC*, que hace a las aplicaciones *JDBC* más fácil mantenerlas.
- Transacciones distribuidas.
- Envía datos a través de la red a clientes ligeros, tales como navegadores web, laptops, PDAs, etc.
- Acceso a cualquier fuente de datos tabular, incluso hojas de cálculo y ficheros planos.
- Encapsula un controlador *JDBC* como un componente *JavaBean* que se puede usar con una aplicación *GUI*.

4.3.2 FUNCIONAMIENTO DE JDBC

El *API* consiste en una serie de clases e interfaces *Java* implementadas por un controlador. Este programa de gestión se encarga de la traducción a las llamadas estándar que requiere la base de datos compatible con él. De esta manera el programador puede abstraerse de la programación específica de la base de datos creando código que funcionará para todas las bases de datos que cuenten con un controlador *JDBC* con sólo cambiar tal controlador.

Conceptualmente es similar a *ODBC*, pero éste no es apropiado para usar directamente desde *Java* porque usa una interfaz en *C* y una traducción literal de *C* a *Java* no es deseable.

JDBC soporta dos modelos de acceso a bases de datos: modelo de dos capas (**Fig. 4.16**) y modelo de tres capas (**Fig. 4.17**). En el primer caso, la aplicación o *Applet* en *Java* se comunica directamente con la base de datos mediante un controlador *JDBC* específico para cada *DBMS* que se desee manipular. En el segundo caso, los comandos son enviados a una capa intermedia de servicios, encargada de reenviar las sentencias *SQL* a la base de datos.

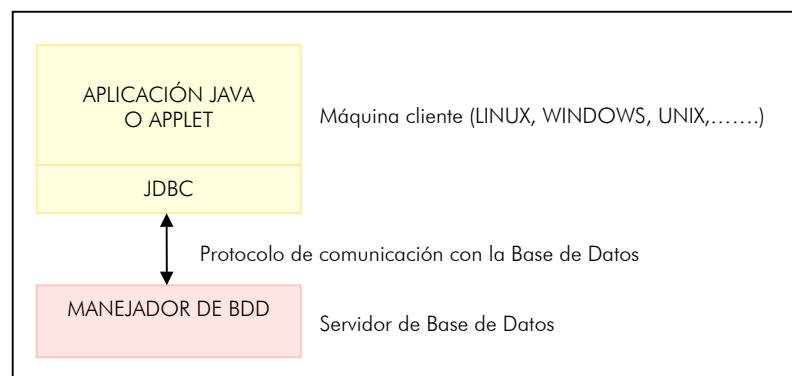


Fig. 4.16 Modelo de Dos Capas

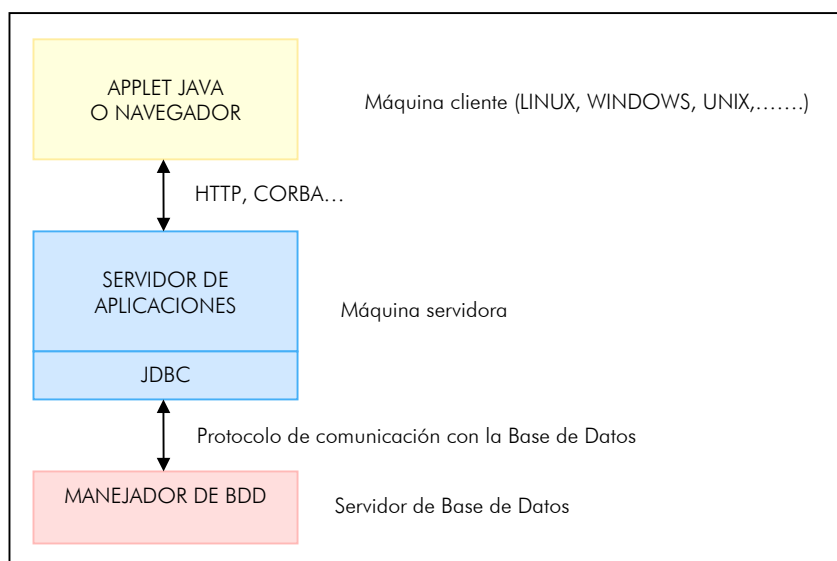


Fig. 4.17 Modelo de Tres Capas

Para acceder a una base de datos, se debe obtener un controlador *JDBC*. El controlador se encarga de manejar las cosas específicas del motor, y coopera con el *API* del *JDBC*, para lograr la conectividad. Los controladores se pueden escribir totalmente en *Java* o mezclarse con métodos nativos escritos en *C++* o *Delphi*.

Un controlador *JDBC* suele ser un fichero *.jar* que contiene una implementación de todas las interfaces *API* de *JDBC*.

TIPOS DE CONTROLADORES JDBC

Los cuatro tipos de controladores *JDBC* existentes son (Fig. 4.18):

- **Nivel 1: Puente *JDBC-ODBC*.**

Esta categoría de controladores se remite al controlador de puente *JDBC-ODBC* original. Este último utiliza el controlador *ODBC* de Microsoft para comunicarse con los servidores de base de datos. Se implementa tanto en código binario como en *Java* y debe ser instalado previamente en la computadora cliente antes de poder usarlo.

Ventajas: Dada la gran popularidad de *ODBC*, casi todas las bases de datos cuentan actualmente con un controlador *ODBC*. Por esta razón, la conectividad con *Java* se puede hacer inmediatamente sin hacer ningún desarrollo adicional.

Desventajas: El uso de métodos nativos implica que esta conectividad no se puede usar

en *Applets*. Además la conexión se vuelve más lenta por usar *ODBC*.

- **Nivel 2: Controlador JDBC de protocolo nativo en Java y binario.**

Esta categoría está compuesta por controladores que hablan con los servidores de bases de datos en el protocolo nativo de la base de datos. Se implementan como una combinación de código binario y *Java* y deben ser instalados en el cliente antes de poder ser usados.

Ventajas: Casi todos los motores tienen un cliente ya probado y depurado, que hace que el desarrollo de estos controladores pueda ser implementado fácilmente por todos los proveedores.

Desventajas: Tampoco se puede usar en *Applets* debido al llamado a métodos nativos.

- **Nivel 3: Controlador JDBC de protocolo no nativo en Java puro.**

Esta categoría está formada por controladores de *Java* puro (no hay código binario) que hablan un protocolo de red estándar (como *HTTP*) con un servidor de acceso a bases de datos. Este servidor traduce el protocolo de red a uno de base de datos específico de la marca. No necesita instalación en el cliente.

Ventajas: Esta es la alternativa más flexible para conectarse con diferentes motores de bases de datos simultáneamente. Además, como el controlador está totalmente desarrollado en *Java*, puede funcionar en *Applets* y aplicaciones.

Desventajas: El uso de una máquina intermedia hace necesariamente más lenta la transferencia de datos que usando controladores de tipo 2 ó 4.

Nota: La versión Cliente/Servidor de *JBuilder* trae un controlador de este tipo llamado "Borland DataGateway", que brinda conexión nativa de muy alto desempeño con los siguientes motores de bases de datos: Oracle, InterBase, Sybase SQL Anywhere, Informix, DB2, MS SQL, Access, FoxPro, dBase y Páradox. Este controlador también soporta conexiones *ODBC*.

- **Nivel 4: Controlador JDBC de protocolo nativo en Java puro.**

Está formado por controladores de *Java* puro que hablan el protocolo de la base de datos específico de la marca del servidor de bases de datos que se haya designado. No necesita instalación en el cliente.

Ventajas: La eliminación de intermediarios, hace que el rendimiento accedando la base de datos sea óptimo. Al ser un controlador sólo *Java*, puede funcionar tanto en *Applets* como en aplicaciones.

Desventajas: Es el tipo de controlador más difícil de implementar por los proveedores de bases de datos, porque involucra desarrollo en el cliente y en el servidor. Muchos proveedores no ofrecen aún esta forma de conectividad.

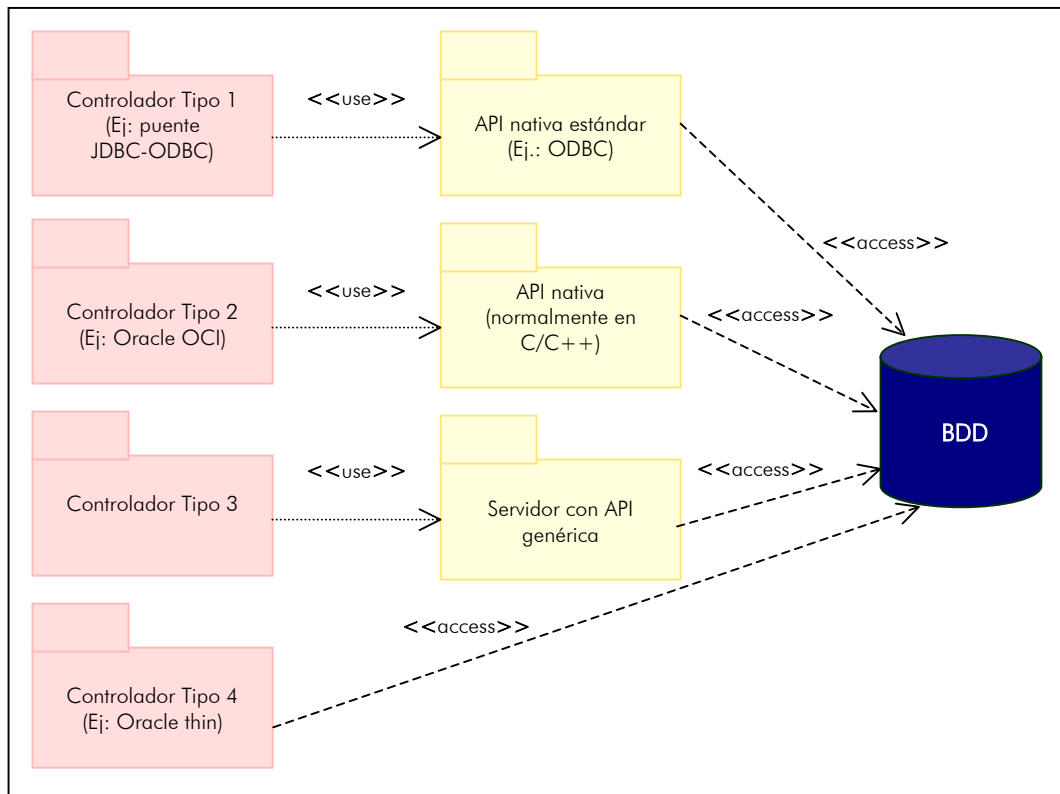


Fig. 4.18 Tipos de Controladores JDBC

Nota: Los controladores de tipo 1 y 2 llaman a APIs nativas mediante JNI (Java Native Interface), requieren que la máquina en la que corre la aplicación tenga instaladas las librerías de las APIs nativas (.DLL, .so). Los controladores de tipo 3 y 4 son 100% Java. La ventaja de un controlador de tipo 3 es que una aplicación puede usarla para conectarse a varias bases de datos. Y en general, los controladores más eficientes son los de tipo 2.

4.3.3 VENTAJAS DE JDBC

- Es útil para el desarrollo de aplicaciones de servidor en donde se puede controlar mucho mejor el ambiente.
- Los controladores de JDBC como están escritos en Java son automáticamente instalables, portables y seguros. No hay necesidad de instalar el controlador en cada

máquina.

- El programador tiene contacto directo con el *DBMS*, no con un socket o un proceso en cualquier máquina.
- La comunicación al manejador de bases de datos es fácil de programar.
- Es flexible para aceptar configuraciones diferentes de conectividad. Es decir, que la misma aplicación del desarrollador pudiese acceder a varios *DBMS* a la vez.
- Mejora del rendimiento en la manipulación de grandes objetos, como BLOB y CLOB, sin traerlos de la base de datos y en el envío de múltiples UPDATEs que se procesan “por lotes”.

4.3.4 DESVENTAJAS DE JDBC

- Es un poco lento e incómodo para usar, si en Internet no se puede controlar los navegadores, sistemas operativos o especificaciones de hardware del usuario.
- El problema de usarse con el *JDK 1.02* es que *JDBC* es una añadidura al mismo. Esto quiere decir que los usuarios comunes tendrían que agregar a su máquina virtual de *Java* las clases que implementan *JDBC*, lo cual sería una carga para ellos.

4.3.5 CONEXIÓN CON JSP

La conexión con el controlador de base de datos que se quiere utilizar, implica dos pasos:

1. Cargar el controlador, ver **Fig. 4.19**.

```
Class.forName([UBICACIÓN DEL CONTROLADOR]);
```

Fig. 4.19 Cargar un controlador JDBC

Una vez cargado el controlador *JDBC*, es posible hacer una conexión con un servidor de base de datos.

2. Hacer la Conexión, ver **Fig. 4.20** y **Fig. 4.21**:

```
<% //Abrir JSP
Connection conexion; //Objeto conexión
conexion=DriverManager.getConnection //Abrir la conexión
("jdbc:odbc:Nombre_ODBC", "usuario", "password");

( conjunto de operaciones sobre la conexión) //Operaciones
conexion.close(); //Cerrar la conexión
%> //Cerrar JSP
```

Fig. 4.20 Conectarse a una Base de Datos

```
<% //Abrir JSP
Connection conexion; //Objeto conexión
conexion=DriverManager.getConnection //Abrir la conexión
("jdbc:odbc:Nombre_ODBC", "usuario", "password");

Statement Estamento = conexion.createStatement(); //Crear cadena y recordset

ResultSet rs = Estamento.executeQuery("SELECT * FROM comunidad ");

( conjunto de operaciones con el recordset ) //Operaciones

rs.close(); //Cerrar el recordset
Estamento.close();
conexion.close(); //Cerrar la conexión
%> //Cerrar JSP
```

Fig. 4.21 Consultar datos de una Tabla

Capítulo 5



SOLUCIONES WEB

- ❖ Aplicaciones Web
 - ❖ Características
 - ❖ Funcionamiento
 - ❖ Ventajas
 - ❖ Desventajas
- ❖ Aplicaciones web con herramientas de código abierto y propietario

INTRODUCCIÓN

Luego de haber estudiado las plataformas Windows y Linux, las herramientas de generación de contenido dinámico del lado del servidor: *PHP*, *ASP* y *JSP*, los servidores web Apache Web Server, Internet Information Server y Tomcat Web Server y la base de datos PostgreSQL en los capítulos anteriores, este capítulo se centra en su integración para el desarrollo de aplicaciones web.

Para desarrollar aplicaciones web es importante conocer sus características, funcionamiento, ventajas y desventajas e indispensable elegir las herramientas adecuadas. En el mercado existen muchas y muy variadas, tanto en los aspectos técnicos como en el costo, por esto se plantean algunas soluciones para facilitar la elección el momento de desarrollar aplicaciones web tomando en cuenta la plataforma y los componentes necesarios.

Las soluciones web planteadas se clasifican en: Alternativas de Código Abierto y Alternativas de Código Propietario.

5.1 APLICACIONES WEB

No existe una definición exacta de aplicación web, ya que en muchos casos depende del punto de vista de quien la defina, y en otros casos se ve como sinónimo de sitio web o portal. A continuación se citan algunas definiciones:

"Una Aplicación Web es un sitio web donde la navegación y el ingreso de datos por parte de un usuario, afectan el estado de la **lógica del negocio**²¹ del servidor. En esencia, una aplicación web usa un sitio web como entrada(front-end) a una aplicación típica. Si no existe lógica del negocio en el servidor, la aplicación no puede ser llamada aplicación web". [WWW-48]

"Una Aplicación Web es aquella que corre en al menos un servidor y a la que el usuario accede desde un cliente de propósito general, por ejemplo un navegador, un teléfono WAP". [WWW-49]

"Una Aplicación Web es un conjunto de páginas web que se manejan desde un servidor web el cual está en capacidad de interpretar datos suministrados por el cliente y enviar una respuesta creando así páginas dinámicas y permitiendo el flujo de información a través del navegador". [WWW-50]

"Una Aplicación Web es aquella aplicación accesible mediante Internet, que modifica el estado del servidor que la aloja". [WWW-51]

"Una Aplicación Web es aquella aplicación inteligente que asesora en Internet a los usuarios/visitantes, e incluso les sule en la realización de determinadas tareas". [WWW-52]

En conclusión, una Aplicación Web puede ser definida como: "Aquel software Cliente/Servidor que interactúa con el usuario a través de Internet, extendiendo la funcionalidad de los sitios web y los portales de forma transparente".

²¹ Conjunto de reglas, validaciones y restricciones que maneja una aplicación web para evitar que se lleven a cabo acciones no válidas.

5.1.1 CARACTERÍSTICAS DE LAS APLICACIONES WEB

- Son aplicaciones que se basan en el modelo Cliente/Servidor y se comportan en Internet de igual forma que las aplicaciones de escritorio tradicionales.
- Su interfaz gráfica está formada por páginas web, forman parte de sitios web y portales a los que dotan de gran dinamismo. Pueden ser vistas como la evolución de los sitios web.
- La arquitectura de una aplicación web consta de tres componentes principales: un navegador, un servidor web y una base de datos.
- Frente a las aplicaciones tradicionales con un único código fuente (“monolíticas”), las aplicaciones web se caracterizan porque están formadas por diferentes módulos o componentes (páginas estáticas, scripts, componentes, imágenes, etc.) independientes entre sí.
- En el desarrollo de aplicaciones web hay una gran tendencia a separar la presentación y la *lógica de negocios* de la aplicación. Esto se hace con el fin de delegar roles de trabajo, dentro del desarrollo del sistema: los diseñadores gráficos tienen la tarea de elaborar la capa de presentación de la aplicación utilizando *HTML/XML/WML*. Los programadores y administradores de las bases de datos escribirán la lógica de la aplicación y las rutinas de acceso a los datos respectivamente.
- Se ejecutan en el servidor o pueden ser ejecutadas en el cliente y en el servidor, dividiendo parte de la ejecución en cada uno. Cuando se ejecutan en el cliente y en el servidor, el cliente cuenta con objetos que permiten realizar acciones complejas, y el servidor cuenta con las aplicaciones que se encargan de manejar esas acciones. No pueden ser ejecutadas sólo en el cliente, las herramientas para programar en el cliente no cumplen con las características suficientes.
- Accesan a bases de datos, mediante formularios capturan los datos de entrada, los procesan y formatean la correspondiente información de salida. Si no se gestionara datos, difícilmente se podría hablar de una aplicación web.
- Son capaces de manejar sesiones para controlar las acciones y navegación del usuario.
- Se presentan de igual forma que las páginas web, por lo tanto hay que tomar en cuenta que no existan vínculos rotos, no sobrecargar de imágenes ni textos, evitar la inconsistencia entre páginas.

[WWW-53]

5.1.2 FUNCIONAMIENTO DE UNA APLICACIÓN WEB

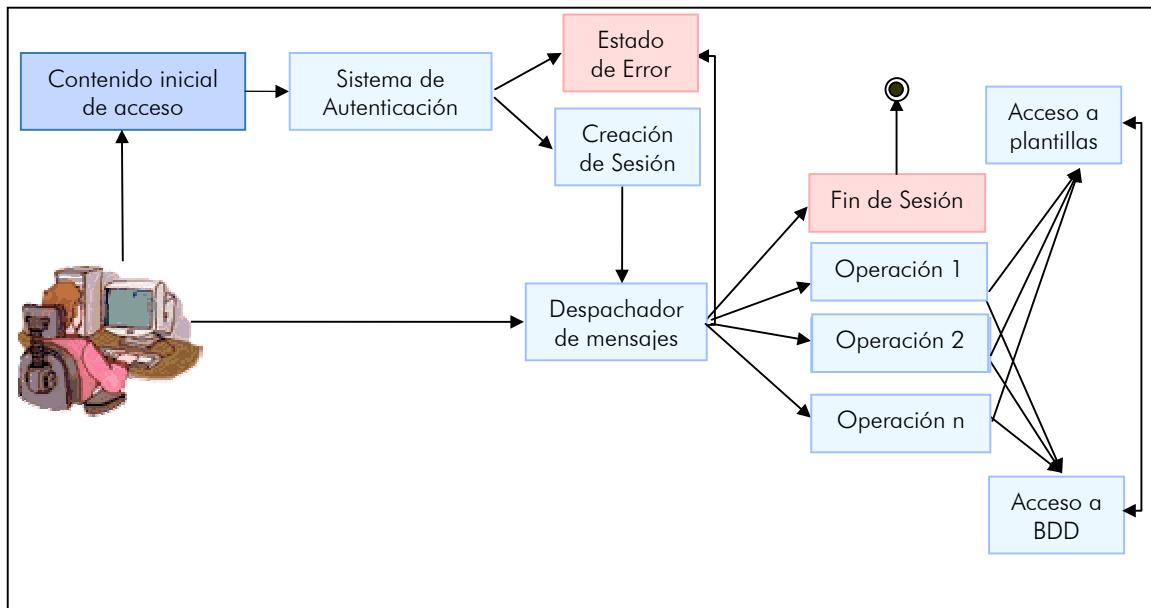


Fig. 5.1 Funcionamiento de una Aplicación Web

En primera instancia, lo que ve un usuario al ingresar a un sitio que alberga una aplicación web es la página de contenido inicial, ésta es la página de bienvenida, generalmente en ella existe un espacio dedicado a la **autenticación**²² del usuario para su acceso a áreas restringidas del sistema.

Una vez que el usuario es autenticado, la aplicación web debe de poner en marcha un mecanismo que le permita determinar, en todo momento, cual es el estado del usuario, es decir, en qué parte de la aplicación se encuentra, qué variables se han establecido, etc. Lo anterior se logra con el manejo de sesión.

A partir de este punto, todas las acciones o peticiones del usuario son enviadas, un despachador de mensajes se encarga de ejecutar operaciones de validación y autenticación y posteriormente manda a ejecutar el procedimiento de acuerdo al mensaje solicitado y al estado encontrado. En caso de que haya errores en una petición, el despachador de mensajes se encarga de enviar el correspondiente mensaje de error.

Cada dato que el usuario envía a la aplicación debe ser previamente evaluado por un

²² Proceso para determinar si algo o alguien es realmente quien declara ser. Este proceso es comúnmente implementado con la validación de un nombre de usuario y una contraseña.

mecanismo de validación; es decir, se debe evitar que la aplicación procese datos no autorizados.

Toda aplicación web procesará información proveída por el usuario, para ello accederá a una base de datos para leer y escribir información.

Un aspecto importante en la estructura de una aplicación web es el uso de plantillas. Las plantillas son una serie de archivos .html, .htm, .xml, etc., que definen la apariencia de la aplicación, de esta manera, si se desea cambiar la presentación de la aplicación, se tiene que modificar las plantillas sin necesidad de modificar el código que le da funcionalidad a la aplicación.

Las aplicaciones web pueden ser estructuradas mediante el concepto de capas, lo más común es que se apliquen tres capas:

Tabla 5.1 Descripción de las Capas de una Aplicación Web

CAPA	DESCRIPCIÓN	HERRAMIENTAS
LÓGICA DE PRESENTACIÓN	<p>Esta capa establece la interfaz de usuario, y gestiona la entrada de datos del usuario. Se tratan aspectos como:</p> <ul style="list-style-type: none"> • Login (validación de usuarios y contraseñas) • Perfil y sesión de usuario • Manejo de eventos y validaciones 	<ul style="list-style-type: none"> • HTML/DHTML/WML • XML/XSL • CSS • Flash • JavaScript/VBScript • Applets • JavaBeans
LÓGICA DEL NEGOCIO	<p>Esta capa se encarga de manejar y controlar la lógica de la aplicación. Son los scripts o programas que procesan los datos, realizan los cálculos, se conectan a la base de datos y generan dinámicamente las páginas web de respuesta.</p>	<ul style="list-style-type: none"> • CGI, ISAPI, NSAPI • PHP/ASP/JSP/CFML, etc. • Servlets • JavaBeans • Conexión ODBC, JDBC
MANEJO DE DATOS	<p>Esta capa se encarga de mantener la integridad de los datos de la base de datos y gestiona el acceso a éstos.</p>	<ul style="list-style-type: none"> • SQL • XML • ODBC, JDBC, DBMS

[WWW-54]

5.1.3 VENTAJAS DE LAS APLICACIONES WEB

La creciente popularidad de las aplicaciones web se debe a sus múltiples ventajas, entre las cuales podemos citar:

- **Multiplataforma:** Con un solo programa o conjunto de scripts, una aplicación web puede ser utilizada a través de múltiples plataformas, tanto de hardware como de software. Los navegadores web se han desarrollado para todo tipo de plataforma, por lo que cualquier usuario de Internet, dispone de la herramienta básica para acceder a la aplicación. Las aplicaciones web no deben ser programadas para un navegador específico.
- **Actualización instantánea:** Debido a que todos los usuarios hacen uso de una sola aplicación que radica en el servidor, los usuarios siempre utilizarán la versión más actualizada.
- **Curva de aprendizaje fácil:** Los usuarios utilizan la aplicación como si estuvieran navegando, la interacción con el usuario se establece basándose en elementos a los que se está cada vez más acostumbrado: páginas web (que vienen a ser el front-end), por lo cual su acceso es más intuitivo. La aplicación puede ser utilizada por un usuario avanzado o por un inexperto.
- **Facilidad de integrarse con otros sistemas:** Debido a que se basa en protocolos estándares, la información manejada por la aplicación puede ser accedida con mayor facilidad por otras aplicaciones o se pueden añadir más módulos sin mucha dificultad.
- **Fácil acceso:** El usuario puede acceder a la aplicación con la única restricción de que cuente con un acceso a la red privada de una organización o a Internet.
- **Facilidad de desarrollo:** Hay dos aspectos a destacar que determinan el desarrollo de este tipo de aplicaciones. En primer lugar, la organización en módulos, así como el hecho de que en cierta medida, algunos aspectos que tienen que ver con el carácter distribuido de la aplicación están resueltos de antemano (por ejemplo el protocolo HTTP controla el acceso a datos en el servidor) y no hay que preocuparse mucho por ellos.
- **Integración de herramientas de desarrollo:** Pueden ser implementadas utilizando lenguajes, herramientas y plataformas diferentes tanto en el lado del cliente como en el lado del servidor.
- **Menor consumo de recursos:** Las aplicaciones web en el lado del cliente requieren de

menos recursos que una aplicación tradicional, ya que utilizan solamente un navegador, el cual a través de la red se conecta al servidor que es quien procesa las peticiones del cliente.

- **Mayor oportunidad de validación y procesamiento:** Existe la oportunidad de validar y procesar tanto en el lado del servidor como en el lado del cliente o en ambos.

Además de las ventajas anteriores debemos recalcar dos aspectos muy importantes que deben tomarse en cuenta a la hora de programar una aplicación web:

- **Rapidez de respuesta:** Debido a que el acceso a Internet suele ser en muchos casos lento, hay que diseñar las páginas de tal forma que su carga no demore más de lo normal. Aunque la aplicación web funcione sólo en una Intranet también es importante la rapidez de respuesta.
- **Igual funcionalidad que las aplicaciones tradicionales:** Al acceder a una aplicación web se debe encontrar la misma funcionalidad que se encuentra en una aplicación tradicional, el estar formada por un conjunto de páginas web no debe marcar la diferencia entre una y otra, ya que una aplicación web puede hacer lo mismo que una aplicación tradicional.

5.1.4 DESVENTAJAS DE LAS APLICACIONES WEB

- **Insuficiente validación de los datos de entrada:** Cada dato que el usuario envía a la aplicación debe ser analizado antes de ser procesado. La validación insuficientemente de los datos puede generar errores graves.
- **Falta de seguridad:** Es necesaria la adopción de medidas que garanticen la utilización de la aplicación únicamente por usuarios autorizados, sobre todo si las operaciones que se pueden realizar son críticas. Por esta razón, se pone de manifiesto la necesidad de asegurar mediante algún mecanismo la intimidad y la integridad en las sesiones con el servidor web.
- **Riesgo en el envío de datos:** Al enviar datos con el método *GET* se corre el riesgo de que dichos datos, al ser pasados por la *URL*, sean mal utilizados y modificados pudiendo ocasionar su pérdida, conflictos con el servidor o la base de datos.
- **Nuevos retos que enfrentar:** El desarrollo de aplicaciones web, implica retos no

enfrentados previamente por los programadores, administradores de proyectos, analistas de sistemas, etc. Aunque las aplicaciones web se basan en el modelo Cliente/Servidor, cambian profundamente los conceptos preestablecidos acerca de la programación de aplicaciones para escritorio. Cuestiones de seguridad, multiprocesamiento, presentación de datos, etc., toman un giro diferente, y pueden tomar desprevenidos a los desarrolladores.

[WWW-54]

5.2 APLICACIONES WEB CON HERRAMIENTAS DE CÓDIGO ABIERTO Y PROPIETARIO

El desarrollo de aplicaciones web crece día con día, existe gran cantidad de herramientas, éstas se dividen en dos tendencias bien marcadas: el software de código abierto y el software propietario. Lógicamente los costos de las licencias son muy variados, pero más allá del costo, lo más importante a la hora de elegir el software adecuado para desarrollar aplicaciones web es analizar las alternativas para determinar cual es más conveniente.

Cabe destacar que dichas alternativas se presentan como opciones para Windows y Linux.

5.2.1 HERRAMIENTAS DE CÓDIGO ABIERTO

Del estudio realizado en los capítulos 2, 3 y 4, y las tablas comparativas del capítulo 2, se pueden establecer alternativas que resultan de la combinación: lenguaje – servidor web – base de datos de código abierto para desarrollar aplicaciones web, las mismas que se comparan bajo los siguientes aspectos: a)plataforma, b)instalación, c)herramientas adicionales requeridas, d)editor de lenguaje, e)prestación de las herramientas, f)conocimientos básicos, g)portabilidad, h)facilidad de integración entre sí; y se muestran en la **Tabla 5.2**.

Tabla 5.2 Alternativas de Código Abierto para desarrollar Aplicaciones Web

LENGUAJE	SERVIDOR WEB	BASE DE DATOS	OBSERVACIONES
PHP	Apache	PostgreSQL	<p>a) Disponibles en Windows y Linux</p> <p>b) En Windows: PHP se instala manualmente como API, Apache y PostgreSQL tienen asistentes. En Linux: PHP, Apache y PostgreSQL se encuentran como opciones en la instalación del S.O., o se pueden instalar vía RPM o Binarios.</p> <p>c) En Windows: PostgreSQL requiere de Cygwin y CygIPC</p> <p>d) En Windows: PHP puede ser editado en un bloc de notas o en Dreamweaver. En Linux: en editores avanzados de texto. Ejm: Kate</p> <p>e) Con la instalación de los módulos requeridos, PHP cuenta un gran número de funciones.</p> <p>f) PERL, C, C++</p> <p>g) Esta combinación puede ser portada entre las dos plataformas, con el cuidado que implica la diferencia de versiones.</p> <p>h) En las dos plataformas se integran muy satisfactoriamente.</p>
PHP	Apache	MySQL	<p>a) Disponibles en Windows y Linux</p> <p>b) En Windows: PHP se instala manualmente como API, Apache y MySQL tienen asistentes. En Linux: PHP, Apache y MySQL se encuentran como opciones en la instalación del S.O., o se pueden instalar vía RPM o Binarios.</p> <p>c) Ninguna</p> <p>d) En Windows: PHP puede ser editado en un bloc de notas o en Dreamweaver. En Linux: en editores avanzados de texto. Ejm: Kate</p> <p>e) Con la instalación de los módulos requeridos, PHP cuenta un gran número de funciones. MySQL no es relacional, no soporta triggers ni integridad referencial.</p> <p>f) PERL, C, C++</p> <p>g) Esta combinación puede ser portada entre las dos plataformas, con el cuidado que implica la diferencia de versiones.</p> <p>h) En las dos plataformas se integran muy satisfactoriamente.</p>
JSP	Apache - Tomcat	PostgreSQL	<p>a) Disponibles en Windows y Linux</p> <p>b) En Windows: JSP se instala con el JDK, Tomcat manualmente, Apache y PostgreSQL tienen asistentes. En Linux: JSP se instala con el JDK, Tomcat manualmente, Apache y PostgreSQL se encuentran como opciones en la instalación del S.O., o se pueden instalar vía RPM o Binarios. En los dos S.O. requiere de configuraciones avanzadas.</p> <p>c) Para Apache y Tomcat se requiere el módulo mod_jk en las dos plataformas.</p>

			<p>Para conectarse a PostgreSQL requiere del controlador JDBC.</p> <p>En Windows: PostgreSQL requiere de Cygwin y CyglPC</p> <p>d) En Windows: JSP puede ser editado en un bloc de notas o en Dreamweaver.</p> <p>En Linux: en editores avanzados de texto. Ejm: Kate</p> <p>e) JSP puede integrarse con Servlets y JavaBeans.</p> <p>f) Java y Servlets.</p> <p>g) Esta combinación puede ser portada entre las dos plataformas, con el cuidado que implica la diferencia de versiones.</p> <p>h) En las dos plataformas se integran satisfactoriamente.</p>
JSP	Apache - Tomcat	MySQL	<p>a) Disponibles en Windows y Linux</p> <p>b) En Windows: JSP se instala con el JDK, Tomcat manualmente, Apache y MySQL tienen asistentes.</p> <p>En Linux: JSP se instala con el JDK, Tomcat manualmente, Apache y MySQL se encuentran como opciones en la instalación del S.O., o se pueden instalar vía RPM o Binarios.</p> <p>En los dos S.O. requiere de configuraciones avanzadas.</p> <p>c) Para Apache y Tomcat se requiere el módulo mod_jk en las dos plataformas.</p> <p>Para conectarse a MySQL requiere del controlador JDBC.</p> <p>d) En Windows: JSP puede ser editado en un bloc de notas o en Dreamweaver.</p> <p>En Linux: en editores avanzados de texto. Ejm: Kate</p> <p>e) JSP puede integrarse con Servlets y JavaBeans.</p> <p>MySQL no es relacional, no soporta triggers ni integridad referencial.</p> <p>f) Java y Servlets.</p> <p>g) Esta combinación puede ser portada entre las dos plataformas, con el cuidado que implica la diferencia de versiones.</p> <p>h) En las dos plataformas se integran satisfactoriamente.</p>
ASP (Chili!Soft ASP)	Apache	PostgreSQL	<p>a) Disponible en Linux con la versión gratuita de Chili!Soft ASP. También está disponible en Windows, pero sólo se integra con el servidor iPlanet que no es gratuito.</p> <p>b) ASP se instala con el asistente de Chili!Soft ASP.</p> <p>c) ASP requiere de Chili!Soft ASP. Recomendadas las versiones 1.3.19 o 1.3.22 de Apache. Para conectarse a PostgreSQL utiliza los controladores ODBC que se instalan con Chili!Soft ASP.</p> <p>d) En el propio editor de Chili!Soft ASP o en editores avanzados de texto. Ejm: Kate.</p> <p>e) El VBScript de Chili!Soft ASP posee menos funciones que el VBScript original de Microsoft.</p> <p>f) VBScript, JScript</p> <p>g) Esta combinación es sólo para Linux.</p>

ASP (Chili!Soft ASP)	Apache	MySQL	<p>h) Se integra satisfactoriamente.</p> <p>a) Disponible en Linux con la versión gratuita de Chili!Soft ASP. También está disponible en Windows, pero sólo se integra con el servidor iPlanet que no es gratuito.</p> <p>b) ASP se instala con el asistente de Chili!Soft ASP.</p> <p>c) ASP requiere de Chili!Soft ASP. Recomendadas las versiones 1.3.19 o 1.3.22 de Apache. Para conectarse a MySQL utiliza los controladores ODBC que se instalan con Chili!Soft ASP.</p> <p>d) En el propio editor de Chili!Soft ASP o en editores avanzados de texto. Ejm: Kate.</p> <p>e) El VBScript de Chili!Soft ASP posee menos funciones que el VBScript original de Microsoft.</p> <p>f) VBScript, JScript</p> <p>g) Esta combinación es sólo para Linux.</p> <p>h) Se integra satisfactoriamente.</p>
----------------------------	--------	-------	--

5.2.2 HERRAMIENTAS DE CÓDIGO PROPIETARIO

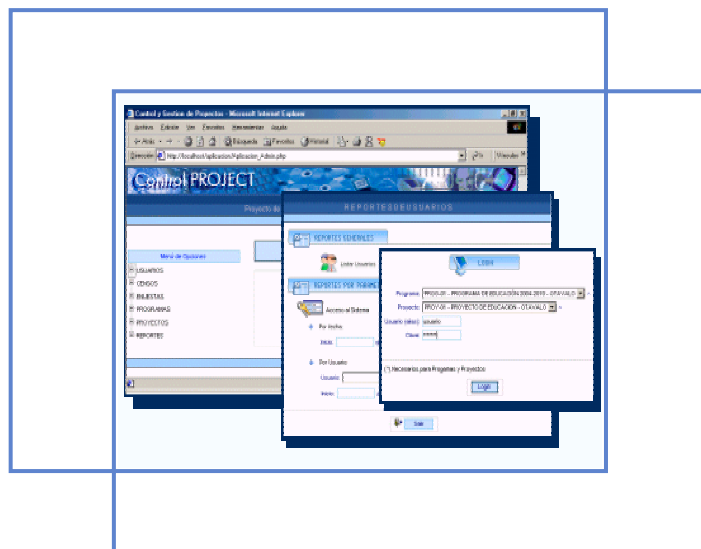
Del estudio realizado en los capítulos 2, 3 y 4, y las tablas comparativas del capítulo 2, se pueden establecer alternativas que resultan de la combinación: lenguaje – servidor web – base de datos de código propietario para desarrollar aplicaciones web, las mismas que se comparan bajo los siguientes aspectos: a)plataforma, b)instalación, c)herramientas adicionales requeridas, d)editor de lenguaje, e)prestación de las herramientas, f)conocimientos básicos, g)portabilidad, h)facilidad de integración entre sí; y se muestran en la **Tabla 5.3**.

Cabe aclarar que en las alternativas de código propietario se toma en cuenta sólo *ASP* porque *PHP* y *JSP* son de código abierto y se presentan en la **Tabla 5.2**. Y para tener otra herramienta con qué comparar *ASP* también se establecen alternativas con *CFML*(ColdFusion), a pesar de que *CFML* no consta en el Análisis Comparativo del capítulo 2.

Tabla 5.3 Alternativas de Código Propietario para desarrollar Aplicaciones Web

LENGUAJE	SERVIDOR WEB	BASE DE DATOS	OBSERVACIONES
ASP	Internet Information Server/Personal Web Server	PostgreSQL	<ul style="list-style-type: none"> a) Disponible en Windows con Internet Information Server o con Visual Interdev. b) ASP se instala con Internet Information Server o con Visual Interdev de Visual Studio. c) Para conectarse a PostgreSQL requiere de Cygwin, CygIPC, ODBC y MS Access. d) En Visual Interdev o en el Bloc de Notas. e) En comparación con PHP, ASP posee menos funciones. f) Visual Basic., VBScript, JScript. g) Esta combinación es sólo para Windows. h) Se integra satisfactoriamente.
ASP	Internet Information Server/Personal Web Server	MySQL	<ul style="list-style-type: none"> a) Disponible en Windows con Internet Information Server o con Visual Interdev. b) ASP se instala con Internet Information Server o con Visual Interdev de Visual Studio. c) Para conectarse a MySQL requiere de ODBC. d) En Visual Interdev o en el Bloc de Notas. e) En comparación con PHP, ASP posee menos funciones. f) Visual Basic., VBScript, JScript. g) Esta combinación es sólo para Windows. h) Se integra satisfactoriamente.
CFML (ColdFusion)	Apache/ Internet Information Server	PostgreSQL	<ul style="list-style-type: none"> a) Disponible en Windows y Linux. b) ColdFusion se instala con asistente. c) Para conectarse a PostgreSQL en Windows requiere de Cygwin, CygIPC y ODBC. d) En UltraDev Studio. e) CFML soporta más de 70 TAGS, 200 funciones y 800 componentes de terceras partes. f) CFML g) Las aplicaciones en CFML pueden migrar a otras plataformas y motores de bases de datos, con pequeños cambios y plug-ins externos. h) Se integra muy satisfactoriamente, pero es costoso.
CFML (ColdFusion)	Apache/ Internet Information Server	MySQL	<ul style="list-style-type: none"> a) Disponible en Windows y Linux. b) ColdFusion se instala con asistente. c) Ninguno. d) En UltraDev Studio. e) CFML soporta más de 70 TAGS, 200 funciones y 800 componentes de terceras partes. f) CFML g) Las aplicaciones en CFML pueden migrar a otras plataformas y motores de bases de datos, con pequeños cambios y plug-ins externos. h) Se integra muy satisfactoriamente, pero es costoso.

Capítulo 6



DISEÑO DE LA APLICACIÓN

- ❖ Investigación preliminar
- ❖ Análisis FODA de aplicaciones de Gestión y Control de Proyectos de Inversión
- ❖ Estudio de Factibilidad
- ❖ Diseño de la aplicación
 - ❖ Diseño de la Base de Datos
 - ❖ Diseño de páginas web dinámicas
- ❖ Desarrollo de la aplicación
- ❖ Implementación del prototipo de la aplicación
- ❖ Documentación

INTRODUCCIÓN

Para diseñar la aplicación web, en este capítulo se realiza una investigación previa acerca del funcionamiento de Programas y Proyectos y los procesos que conllevan. Y como complemento, con la información disponible acerca de otros sistemas de programas y proyectos de inversión se marca los lineamientos bajo los cuales se desarrollará el Sistema Prototipo de Gestión y Control de Proyectos de Inversión.

Una vez marcado los lineamientos a seguir se realiza el estudio de factibilidad que permite establecer las herramientas a utilizar en el desarrollo e implementación de la aplicación.

El siguiente paso a seguir es diseñar la estructura general del sistema y la base de datos, para esto utilizamos UML (Unified Modeling Language), el cual cuenta con las herramientas apropiadas para modelar diagramas de Casos de Uso, Secuencia y Actividades del sistema. Y en el caso de la base de datos, permite diseñar los modelos Conceptual y Físico de la misma.

6.1 INVESTIGACIÓN PRELIMINAR

6.1.1 PROGRAMAS

Un Programa es una estrategia de acción cuyas directrices determinan los medios que gerencialmente permiten dar una solución integral a los problemas. Como estrategia, un programa contiene los objetivos y metas, el conjunto de Proyectos y el Plan de Inversiones. Los programas se definen en términos de objetivos de carácter multisectorial.

6.1.2 PROYECTOS

Un Proyecto es una unidad operacional que vincula recursos, actividades, responsables y componentes o resultados durante un período determinado y con una ubicación definida, para resolver problemas o necesidades de la población.

CICLO DEL PROYECTO

El Ciclo de un Proyecto es el proceso de decisiones por el que atraviesa un proyecto durante su vida útil. Las tres etapas en el Ciclo del Proyecto se muestran en la **Fig. 6.1**:

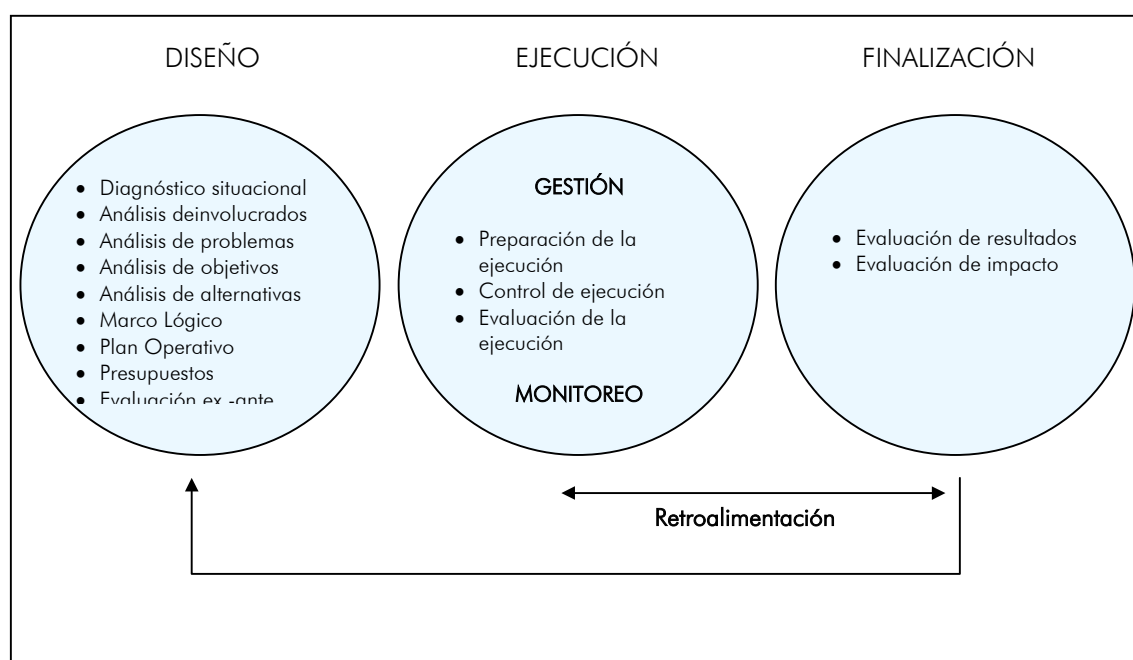


Fig. 6.1 Etapas del Ciclo de los Proyectos

1. DISEÑO

Para realizar la formulación o diseño del proyecto se requiere efectuar lo siguiente:

- a) Diagnóstico situacional, el cual recopila información relevante para diseñar el proyecto.
- b) Análisis de involucrados, permite conocer las necesidades más sentidas e intereses de los grupos participantes del proyecto.
- c) Análisis de problemas, para lo cual se utiliza el árbol de problemas y el diagnóstico situacional.
- d) Análisis de objetivos, se realiza una jerarquización de objetivos para resolver los problemas encontrados. Proporciona una visión clara de la situación que se busca alcanzar mediante la implementación del proyecto.
- e) Análisis de alternativas, permite identificar diferentes opciones alternativas y escoger la mejor estrategia para resolver la problemática.
- f) Marco Lógico, permite resumir en un solo cuadro los elementos más importantes de un proyecto, objetivo de desarrollo, propósito, componentes o resultados, actividades, indicadores, medios de verificación y factores externos.
- g) Plan Operativo, es un cuadro que permite mostrar los resultados, las actividades y tareas secuenciales con sus responsables, duración de las mismas, costos, recursos(humanos, financieros, materiales), y las fuentes de financiamiento para la ejecución de cada proyecto.
- h) Evaluación ex – ante, permite conocer si el diseño del proyecto es viable desde diferentes puntos de vista. Esta evaluación se realiza de acuerdo al carácter del proyecto, económico o social.

2. EJECUCIÓN

La etapa de ejecución o implementación del proyecto considera los siguientes aspectos:

- a) Preparación de la ejecución, analiza la organización de la entidad ejecutora y la programación de tiempos y costos.
- b) Control de la ejecución, es decir, control de tiempo, costo, calidad, producción del bien o servicio y del financiamiento.
- c) Evaluación de la ejecución, se realiza una evaluación de procesos, siendo las más conocidas: monitoreo, evaluación diagnóstica y evaluación desde la perspectiva de los

beneficiarios.

3. FINALIZACIÓN

Esta etapa tiene dos niveles: el primero se refiere a la evaluación de resultados y el segundo es una evaluación de impacto.

- a) Evaluación de resultados, intenta conocer el grado de cumplimiento del propósito y los componentes o resultados del proyecto.
- b) Evaluación de impacto, mide el cambio generado en la población beneficiaria.

ENFOQUE DEL MARCO LÓGICO

El Marco Lógico es una herramienta para facilitar el proceso de conceptualización, diseño, ejecución y evaluación de proyectos. Su propósito es brindar estructura al proceso de planificación y comunicar información esencial relativa al proyecto. Puede utilizarse en todas las etapas del ciclo del proyecto.

El Marco Lógico se presenta como una matriz de cuatro filas por cuatro columnas. Las columnas suministran la siguiente información:

- Un resumen narrativo de los objetivos y las actividades.
- Indicadores, son especificaciones cuantitativas y cualitativas utilizadas para medir el logro de un objetivo.
- Medios de verificación, especifican dónde y cómo se van a obtener los datos para verificar los indicadores.
- Supuestos, son factores que están fuera del control de la entidad ejecutora, pero es necesario conocerlos para que el proyecto logre sus objetivos.

Las filas de la matriz presentan información acerca de los objetivos, indicadores, medios de verificación y supuestos en cuatro momentos diferentes del proyecto:

- Finalidad
- Propósito(s)
- Componentes o Resultados

- Actividades

6.1.3 LA GESTIÓN DE PROYECTOS

La Gestión en un Proyecto, es el conjunto de procedimientos que permiten hacer realidad las ideas. Funciona como el mecanismo de toma de decisiones durante el ciclo del proyecto, que utilizan los ejecutores para movilizar recursos, abrir oportunidades y moverse en el contexto local con el fin de lograr los objetivos propuestos de acuerdo al curso de acción adoptado por las organizaciones.

INSTRUMENTOS PARA LA GESTIÓN DEL PROYECTO

- Marco Lógico
- Redes de actividades y cronogramas
- Presupuestos e informes de gastos
- Indicadores de avance (con encuestas y/o censos)
- Informes
- Documentación Fundamental
- Herramientas Informáticas

6.1.4 SEGUIMIENTO Y MONITOREO DE PROYECTOS

El Seguimiento y Monitoreo son los procesos que garantizan el cumplimiento de los objetivos del proyecto. El Seguimiento se lo realiza a nivel de Actividades mediante su ejecución, control de recursos previstos y tiempos, detectando avances, deficiencias y ajustes. Y el Monitoreo se realiza a nivel de Componentes o Resultados, evaluando los indicadores planificados con los obtenidos.

El Seguimiento se inicia simultáneamente a la ejecución de los proyectos y tiene como objeto brindar información oportuna y adecuada para la toma de decisiones, con la elaboración de reportes que se realizan periódicamente.

El Monitoreo se realiza al cumplimiento de todas las actividades planificadas, en este proceso se establecen nuevos indicadores para su evaluación.

6.1.5 SISTEMA DE GESTIÓN Y CONTROL DE PROYECTOS DE INVERSIÓN

Es un instrumento para registrar los programas y proyectos viables técnica, ambiental y socioeconómicamente, susceptibles de ser financiados. Este constituye una herramienta fundamental para los proyectos, apoyando de manera directa el logro de la misión y visión institucional.

Permite organizar la información para ejecutar los programas, proyectos, planes de desarrollo y presupuestos, además realizar el análisis, seguimiento y monitoreo de los resultados obtenidos en la ejecución de los proyectos.

Sólo se deben registrar en un sistema los programas y proyectos que hayan sido debidamente formulados para realizar su seguimiento.

El Sistema Prototipo de Gestión y Control de Proyectos de Inversión de este estudio se enfocará en la etapa de Ejecución, permitiendo seguir y monitorear las actividades con sus respectivos recursos asignados; y parte de la etapa de Finalización, al permitir elaborar censos y encuestas para la evaluación de resultados.

[LIB-06] [LIB-07] [WWW-55]

6.2 ANÁLISIS FODA DE APLICACIONES DE GESTIÓN Y CONTROL DE PROYECTOS DE INVERSIÓN

Para el diseño del prototipo, se realizó un análisis FODA de algunas aplicaciones que brindan la Gestión y Control de Proyectos de Inversión.

Con este análisis tratamos de obtener las mejores características con las que cuenta cada una de estas aplicaciones, para que la herramienta prototipo desarrollada en el presente trabajo de Tesis se enmarque dentro de lo que debe realizar un Sistema de Gestión y Control de Proyectos de Inversión.

Tabla 6.1 Análisis FODA de otras Aplicaciones de Gestión y Control de Proyectos de Inversión

APLICACIÓN	FORTALEZAS	OPORTUNIDADES	DEBILIDADES	AMENAZAS
SSEPI Sistema de Seguimiento y Evaluación a los Proyectos de Inversión	<ul style="list-style-type: none"> • Inicia con la formulación • Maneja Programas y Proyectos • Permite la reasignación de recursos 	<ul style="list-style-type: none"> • Mejorar en la integración de subsistemas para hacerlo en un solo sistema. • Implementar módulo Web 	<ul style="list-style-type: none"> • Permite evaluar Indicadores a través de la utilización de subsistemas 	<ul style="list-style-type: none"> • La competencia tiene integrado módulos en un solo sistema • La competencia tiene Aplicaciones Web
SIGAP Sistema Integral Gerencial de Administración de Proyectos	<ul style="list-style-type: none"> • Inicia con la formulación • Permite medir Indicadores y administrar recursos • Guía en los pasos de la metodología del Marco Lógico • Maneja claves de acceso • Adoptado por varias instituciones gubernamentales financiadas por el Banco Mundial 	<ul style="list-style-type: none"> • Implementar módulo Web 	<ul style="list-style-type: none"> • Maneja sólo Proyectos 	<ul style="list-style-type: none"> • La competencia tiene Aplicaciones Web
SIPI Sistema de Información de Proyectos de Inversión	<ul style="list-style-type: none"> • Clasifica Proyectos de Inversión públicos y privados, por ubicación geográfica y estado de avance • Mantiene información histórica 	<ul style="list-style-type: none"> • Superar restricción de montos • Implementar módulo Web 	<ul style="list-style-type: none"> • Sólo maneja montos iguales o superiores a los \$100000 	<ul style="list-style-type: none"> • Proyectos de menor inversión no utilizarían el sistema • La competencia tiene Aplicaciones Web
GESPRO Sistema de Gestión y Control de Proyectos	<ul style="list-style-type: none"> • Aplicación Web • Control del Proyecto en fases • Fácil acceso desde cualquier PC • Vistas en función del rol de cada usuario • Permite modificaciones en la ejecución del Proyecto 	<ul style="list-style-type: none"> • Util para empresas distribuidas geográficamente • Curva de aprendizaje fácil en la navegación de los usuarios 		<ul style="list-style-type: none"> • Posible acceso de hackers y virus
ISOSYSTEM Project Control de Proyectos y Actividades	<ul style="list-style-type: none"> • Aplicación Web • Análisis de Indicadores • Fácil acceso desde cualquier PC • Control de Proyectos en normas 	<ul style="list-style-type: none"> • Util para empresas distribuidas geográficamente • Curva de aprendizaje fácil en la navegación de los usuarios 		<ul style="list-style-type: none"> • Posible acceso de hackers y virus

	<p>internacionales de calidad</p> <ul style="list-style-type: none"> • Comunicación entre empresa, proveedores y clientes • Establece prioridades • Almacenamiento de Anexos 			
<p>SGP Sistema de Gestión de Proyectos</p>	<ul style="list-style-type: none"> • Aplicación Web • Registra impacto de Proyectos • Asignación de fondos por Proyecto y fuentes de financiamiento • Manejo de claves de acceso • Difusión de la información 	<ul style="list-style-type: none"> • Util para empresas distribuidas geográficamente • Curva de aprendizaje fácil en la navegación de los usuarios 		<ul style="list-style-type: none"> • Posible acceso de hackers y virus

6.3 ESTUDIO DE FACTIBILIDAD

FACTIBILIDAD TÉCNICA

El Sistema Prototipo de Gestión y Control de Proyectos de Inversión es factible técnicamente considerando lo siguiente:

- *PHP*, Apache y PostgreSQL se integran de mejor manera y son multiplataforma, tal como se puede comparar en las **Tablas 5.2** y **5.3** del capítulo 5. Esta integración es adecuada para el desarrollo e implantación del sistema prototipo.
- Sólo se necesita un servidor y uno o varios clientes. Clientes que no requieren de características muy avanzadas, ya que la aplicación corre en un navegador que interprete *JavaScript*, *HTML* y *CSS*.

FACTIBILIDAD OPERATIVA

El Sistema Prototipo de Gestión y Control de Proyectos de Inversión es factible operativamente considerando lo siguiente:

- El sistema optimizará los procesos en el manejo de la información, permitiendo generar reportes automáticamente, disminuyendo el tiempo de desarrollo de ciertas tareas que antes se realizaban manualmente en la institución donde será implantado.
- El acceso al sistema para registrar datos o generar reportes se lo puede realizar desde cualquier cliente en la red de la institución.

FACTIBILIDAD ECONÓMICA

El Sistema Prototipo de Gestión y Control de Proyectos de Inversión es factible económicamente considerando lo siguiente:

- Las herramientas de desarrollo del sistema son de código abierto, lo que implica una gran diferencia de costos con relación a herramientas propietarias que se utilizan para el mismo fin.
- El sistema al ser multiplataforma puede ser implantado en Windows o en Linux, brindando la posibilidad de escoger la opción que represente menor costo.

6.4 DISEÑO DE LA APLICACIÓN

6.4.1 DISEÑO DE LA BASE DE DATOS

Para el Sistema Prototipo de Gestión y Control de Proyectos de Inversión se diseñó dos bases de datos:

1. **Censos y Encuestas:** Se utiliza para registrar información de encuestas y censos para luego establecer datos que se pueden utilizar como Indicadores en los Programas y Proyectos.

Esta base de datos almacena información referente a:

- a) Datos Informativos de Población, con las siguientes entidades:

- provincias
- cantones
- comunidades
- familias
- población

- b) Datos de Encuestas, con las siguientes entidades:

- encuestas
- preguntas
- respuestas
- tipos de respuestas
- población encuesta

- c) Datos de Censos, con las siguientes entidades:

- censos
- servicios
- población censo

- d) Información Auxiliar, con las siguientes entidades:

- ocupación
- estado civil
- parentesco
- religión
- nivel de educación

- años de censos

2. **Programas y Proyectos:** Se utiliza para almacenar, gestionar y controlar Programas y Proyectos de Inversión.

Esta base de datos almacena información referente a:

a) Datos de Programas y Proyectos, con las siguientes entidades:

- programas
- proyectos
- finalidad
- propósito
- componentes
- actividades
- subactividades
- recursos económicos
- estado de actividades
- años de proyectos
- documentos adjuntados

b) Datos de Usuarios, con las siguientes entidades:

- cargos
- usuarios
- historial de accesos

c) Información Auxiliar, con las siguientes entidades:

- tipos de gastos
- tipos de proyectos
- datos máximos permitidos
- gastos entre fechas
- presupuesto y gasto
- responsables
- costo de actividades

Diagrama de Casos de Uso

Ver archivo: Diagrama_Casos_Uso.pdf, ubicado en CD_TESIS:\Tesis\Documento_Tesis\
Diagrama_Casos_Uso.pdf.

Diagrama de Secuencia

Ver archivo: Diagrama_Secuencia.pdf, ubicado en CD_TESIS:\Tesis\Documento_Tesis\
Diagrama_Secuencia.pdf.

Diagrama de Actividades

Ver archivo: Diagrama_Actividades.pdf, ubicado en CD_TESIS:\Tesis\Documento_Tesis\
Diagrama_Actividades.pdf.

Modelo Conceptual de Datos

Base de Datos: encuesta_censo

Ver archivo: Diagrama_Conceptual_Encuesta_Censo.pdf, ubicado en
CD_TESIS:\Tesis\Documento_Tesis\Diagrama_Conceptual_Encuesta_Censo.pdf.

Modelo Físico de Datos

Base de Datos: encuesta_censo

Ver archivo: Diagrama_Fisico_Encuesta_Censo.pdf, ubicado en
CD_TESIS:\Tesis\Documento_Tesis\Diagrama_Fisico_Encuesta_Censo.pdf.

Modelo Conceptual de Datos

Base de Datos: proyectos

Ver archivo: Diagrama_Conceptual_Programas_Proyectos.pdf, ubicado en
CD_TESIS:\Tesis\Documento_Tesis\Diagrama_Conceptual_Programas_Proyectos.pdf.

Modelo Físico de Datos

Base de Datos: proyectos

Ver archivo: Diagrama_Fisico_Programas_Proyectos.pdf, ubicado en
CD_TESIS:\Tesis\Documento_Tesis\Diagrama_Fisico_Programas_Proyectos.pdf.

NOMENCLATURA DE LOS MODELOS CONCEPTUAL Y FÍSICO

A continuación, en las tablas 6.2 y 6.3 se presenta la nomenclatura y los tipos de datos utilizados en los Modelos Conceptual y Físico de las bases de datos: encuesta_censo y proyectos.

Tabla 6.2 Nomenclatura de claves e índices

SÍMBOLO	SIGNIFICADO
fk	Foreign key
pk	Primary key
ak	Alternate key
pi	Primary index
ai	Alternate index

Tabla 6.3 Tipos de Datos

TIPO DE DATO	SIGNIFICADO
Integer	Entero
Float	Flotante
Date	Fecha
CHAR(%n)	Characters
SERIAL	Autonumérico
A(%n)	Characters
NO	Serial
D	Fecha
VA	Variable characters
M	Not Null

DICCIONARIO DE DATOS

En las tablas 6.4 y 6.5 se muestran las descripciones de las tablas de cada base de datos del Sistema Prototipo de Gestión y Control de Proyectos de Inversión.

El Diccionario de Datos también se encuentra ubicado en el CD adjunto a este documento en: CD_TESIS:\Tesis\Diccionario\.

Tabla 6.4 Tablas de la base de datos: encuesta_censo

NOMBRE DE TABLA	DESCRIPCIÓN
provincias	Contiene una lista de los nombres de las provincias del Ecuador.
canton	Contiene los nombres de los cantones de cada provincia.
comunidad	Contiene datos de las comunidades de cada cantón y provincia.
familias	Contiene datos de las familias de las comunidades.
poblacion	Contiene datos de cada miembro de las familias.
censo	Contiene datos principales de los censos.
poblacion_censo	Contiene datos de la población censada en un censo determinado.
servicios	Contiene datos sobre agua, luz, tierra las familias censadas en un censo determinado.
encuesta	Contiene datos principales para la creación de una encuesta.
poblacion_encuesta	Contiene datos de la población encuestada en una encuesta determinada.
preguntas	Contiene las preguntas de una encuesta determinada.
tipos_respuestas	Contiene una lista de tipos de preguntas, y según el tipo se crea la pregunta.
respuestas	Contiene datos de la(s) etiqueta(s) o respuesta(s) a escoger de cada pregunta.
nivel_educacion	Contiene una lista de los niveles o años de educación que se utilizan para censar a la población.
estado_civil	Contiene una lista de los estados civiles que se utilizan para censar a la población.
parentezco	Contiene una lista de los parentescos posibles en la familia que se utilizan para censar a la población.
ocupacion	Contiene una lista de las ocupaciones posibles que utilizan para censar a la población.
religion	Contiene una lista de las religiones que se utilizan para censar a la población.
anios	Contiene una lista de años para seleccionar en la creación de censos.

Tabla: provincias

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonómico de la tabla
nombre	CHAR(30)	FALSE	FALSE	TRUE	Nombre de la provincia

Tabla: canton

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonómico de la tabla
nombre	CHAR(30)	FALSE	FALSE	TRUE	Nombre de cantón
parroquias	CHAR(500)	FALSE	FALSE	TRUE	Parroquias pertenecientes al cantón
id_provincia	Integer	FALSE	TRUE	TRUE	Provincia a la que pertenece el cantón

Tabla: comunidad

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonumérico de la tabla
nombre	CHAR(30)	FALSE	FALSE	TRUE	Nombre de la comunidad
sectores	CHAR(255)	FALSE	FALSE	TRUE	Sectores que contiene la comunidad
mision	CHAR(255)	FALSE	FALSE	TRUE	Misión de la comunidad
vision	CHAR(255)	FALSE	FALSE	TRUE	Visión de la comunidad
acuerdo_ministerial	CHAR(20)	FALSE	FALSE	FALSE	Código de acuerdo ministerial
altitud_piso	Integer	FALSE	FALSE	FALSE	Altitud de la comunidad en metros
id_canton	Integer	FALSE	TRUE	TRUE	Cantón al que pertenece la comunidad
parroquia	CHAR(20)	FALSE	FALSE	TRUE	Parroquia a la que pertenece la comunidad
zonal	CHAR(50)	FALSE	FALSE	TRUE	Organización Zonal a la que pertenece la comunidad
provincial	CHAR(50)	FALSE	FALSE	TRUE	Organización Provincial a la que pertenece la comunidad
regional	CHAR(50)	FALSE	FALSE	TRUE	Organización Regional a la que pertenece la comunidad
nacional	CHAR(50)	FALSE	FALSE	TRUE	Organización Nacional a la que pertenece la comunidad

Tabla: familias

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonumérico de la tabla
nombre	CHAR(50)	FALSE	FALSE	TRUE	Apellidos de la familia
num_personas	Integer	FALSE	FALSE	TRUE	Número de personas que conforman la familia
id_comunidad	Integer	FALSE	TRUE	TRUE	Comunidad a la que pertenece la familia
alias	CHAR(50)	FALSE	FALSE	FALSE	Alias(sobrenombre) que posee la familia
sectores	CHAR(50)	FALSE	FALSE	FALSE	Sector al que pertenece la familia

Tabla: poblacion

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonumérico de la tabla
cedula	CHAR(11)	FALSE	FALSE	TRUE	Número de cédula o patrocinio de la persona
nombres	CHAR(30)	FALSE	FALSE	TRUE	Nombres de la persona
apellidos	CHAR(30)	FALSE	FALSE	TRUE	Apellidos de la persona
fecha_nacimiento	Date	FALSE	FALSE	TRUE	Fecha de nacimiento de la persona
sexo	CHAR(1)	FALSE	FALSE	TRUE	Género de la persona
observaciones	CHAR(255)	FALSE	FALSE	FALSE	Observaciones opcionales
id_familias	Integer	FALSE	TRUE	TRUE	Familia a la que pertenece la persona

Tabla: censo

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonumérico de la tabla
nombre	CHAR(30)	FALSE	FALSE	TRUE	Nombre descriptivo del censo
fecha	Date	FALSE	FALSE	TRUE	Fecha de creación del censo
anio	Integer	FALSE	FALSE	TRUE	Año de creación del censo
descripcion	CHAR(255)	FALSE	FALSE	FALSE	Descripción general del censo

Tabla: poblacion_censo

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonumérico de la tabla
nombre_comunidad	CHAR(30)	FALSE	FALSE	TRUE	Comunidad a la que pertenece la persona censada
nombre_familia	CHAR(50)	FALSE	FALSE	TRUE	Familia a la que pertenece la persona censada
alias_familia	CHAR(50)	FALSE	FALSE	FALSE	Alias de la familia a la que pertenece la persona censada
cedula	CHAR(11)	FALSE	FALSE	TRUE	Número de cédula o patrocinio de la persona censada
nombres	CHAR(30)	FALSE	FALSE	TRUE	Nombres de la persona censada
apellidos	CHAR(30)	FALSE	FALSE	TRUE	Apellidos de la persona censada
sexo	CHAR(1)	FALSE	FALSE	TRUE	Género de la persona censada
patrocinado	CHAR(2)	FALSE	FALSE	FALSE	Indica si la persona censada está patrocinada o no
parentesco	CHAR(10)	FALSE	FALSE	TRUE	Posición familiar de la persona censada
ocupacion	CHAR(25)	FALSE	FALSE	TRUE	Ocupación de la persona censada
edad	Integer	FALSE	FALSE	TRUE	Edad de la persona censada
anio_educacion	CHAR(20)	FALSE	FALSE	TRUE	Año de educación de la persona censada
estado_civil	CHAR(12)	FALSE	FALSE	TRUE	Estado civil de la persona censada
dirigente	CHAR(2)	FALSE	FALSE	FALSE	Indica si la persona censada es dirigente o no
embarazada	CHAR(2)	FALSE	FALSE	FALSE	Indica si la mujer censada está embarazada o no
agente_salud	CHAR(2)	FALSE	FALSE	FALSE	Indica si la persona censada es agente de salud o no
id_poblacion	Integer	FALSE	FALSE	TRUE	Persona censada
id_censo	Integer	FALSE	TRUE	TRUE	Censo al que pertenece

Tabla: servicios

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonumérico de la tabla
agua	CHAR(2)	FALSE	FALSE	FALSE	Indica si la familia censada tiene agua
luz	CHAR(2)	FALSE	FALSE	FALSE	Indica si la familia censada tiene luz
letrina	CHAR(2)	FALSE	FALSE	FALSE	Indica si la familia censada tiene letrina
tierra	CHAR(2)	FALSE	FALSE	FALSE	Indica si la familia censada tiene propiedades
casa	CHAR(2)	FALSE	FALSE	FALSE	Indica si la familia censada tiene casa
religion	CHAR(15)	FALSE	FALSE	FALSE	Religión a la que pertenece la familia censada
observaciones	CHAR(255)	FALSE	FALSE	FALSE	Observaciones adicionales
nombre_comunidad	CHAR(30)	FALSE	FALSE	TRUE	Comunidad a la que pertenece la familia censada
nombre_familia	CHAR(50)	FALSE	FALSE	TRUE	Apellidos de la familia censada
alias_familia	CHAR(50)	FALSE	FALSE	FALSE	Alias de la familia censada
id_familias	Integer	FALSE	FALSE	TRUE	Familia censada
id_censo	Integer	FALSE	TRUE	TRUE	Censo al que pertenece

Tabla: encuesta

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonumérico de la tabla
cod_encuesta	CHAR(15)	FALSE	FALSE	TRUE	Código de encuesta
fecha	Date	FALSE	FALSE	TRUE	Fecha de creación de encuesta
problema	CHAR(255)	FALSE	FALSE	TRUE	Descripción del problema de la encuesta
objetivo	CHAR(255)	FALSE	FALSE	TRUE	Descripción del objetivo de la encuesta
num_preguntas	Integer	FALSE	FALSE	TRUE	Número de preguntas de la encuesta
num_encuestados	Integer	FALSE	FALSE	TRUE	Número de encuestados

Tabla: poblacion_encuesta

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonumérico de la tabla
respuesta1	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 1
respuesta2	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 2
respuesta3	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 3
respuesta4	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 4
respuesta5	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 5
respuesta6	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 6
respuesta7	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 7
respuesta8	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 8
respuesta9	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 9
respuesta10	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 10
respuesta11	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 11

respuesta12	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 12
respuesta13	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 13
respuesta14	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 14
respuesta15	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 15
respuesta16	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 16
respuesta17	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 17
respuesta18	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 18
respuesta19	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 19
respuesta20	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 20
respuesta21	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 21
respuesta22	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 22
respuesta23	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 23
respuesta24	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 24
respuesta25	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 25
respuesta26	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 26
respuesta27	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 27
respuesta28	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 28
respuesta29	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 29
respuesta30	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 30
respuesta31	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 31
respuesta32	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 32
respuesta33	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 33
respuesta34	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 34
respuesta35	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 35
respuesta36	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 36
respuesta37	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 37
respuesta38	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 38
respuesta39	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 39
respuesta40	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 40
respuesta41	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 41
respuesta42	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 42
respuesta43	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 43
respuesta44	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 44
respuesta45	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 45
respuesta46	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 46
respuesta47	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 47
respuesta48	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 48
respuesta49	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 49
respuesta50	CHAR(255)	FALSE	FALSE	FALSE	Respuesta a la pregunta 50
nombre_comunidad	CHAR(30)	FALSE	FALSE	FALSE	Comunidad a la que pertenece la persona encuestada
nombre_familia	CHAR(50)	FALSE	FALSE	FALSE	Familia a la que pertenece la persona encuestada
cedula	CHAR(11)	FALSE	FALSE	FALSE	Cédula de la persona encuestada
nombres	CHAR(30)	FALSE	FALSE	FALSE	Nombres de la persona encuestada
apellidos	CHAR(30)	FALSE	FALSE	FALSE	Apellidos de la persona encuestada
sexo	CHAR(1)	FALSE	FALSE	FALSE	Género de la persona encuestada
edad	Integer	FALSE	FALSE	FALSE	Edad de la persona encuestada
id_encuesta	Integer	FALSE	TRUE	TRUE	Encuesta respondida
contador	Integer	FALSE	FALSE	FALSE	Numeración de encuesta

Tabla: preguntas

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonómico de la tabla
num_pregunta	Integer	FALSE	FALSE	TRUE	Número de la pregunta
pregunta	CHAR(255)	FALSE	FALSE	TRUE	Descripción de la pregunta
id_encuesta	Integer	FALSE	TRUE	TRUE	Encuesta a la que pertenece la pregunta

Tabla: tipos_respuestas

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonómico de la tabla
nombre	CHAR(50)	FALSE	FALSE	TRUE	Nombre del tipo de respuesta
seleccion_multiple	CHAR(2)	FALSE	FALSE	FALSE	Indica si la respuesta permite selección múltiple
tipo_control	CHAR(2)	FALSE	FALSE	TRUE	Indica que tipo de control utiliza la pregunta
num_max	Integer	FALSE	FALSE	TRUE	Indica el número máximo de opciones, según el tipo de respuesta
num_min	Integer	FALSE	FALSE	TRUE	Indica el número mínimo de opciones, según el tipo de respuesta

Tabla: respuestas

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonómico de la tabla
num_opciones	Integer	FALSE	FALSE	FALSE	Indica número de respuestas escogido para la pregunta que corresponde
cv1	CHAR(50)	FALSE	FALSE	FALSE	Etiqueta para opción 1
cv2	CHAR(50)	FALSE	FALSE	FALSE	Etiqueta para opción 2
cv3	CHAR(50)	FALSE	FALSE	FALSE	Etiqueta para opción 3
cv4	CHAR(50)	FALSE	FALSE	FALSE	Etiqueta para opción 4
cv5	CHAR(50)	FALSE	FALSE	FALSE	Etiqueta para opción 5
cv6	CHAR(50)	FALSE	FALSE	FALSE	Etiqueta para opción 6
cv7	CHAR(50)	FALSE	FALSE	FALSE	Etiqueta para opción 7
cv8	CHAR(50)	FALSE	FALSE	FALSE	Etiqueta para opción 8
cv9	CHAR(50)	FALSE	FALSE	FALSE	Etiqueta para opción 9
cv10	CHAR(50)	FALSE	FALSE	FALSE	Etiqueta para opción 10
cv11	CHAR(50)	FALSE	FALSE	FALSE	Etiqueta para opción 11
cv12	CHAR(50)	FALSE	FALSE	FALSE	Etiqueta para opción 12
cv13	CHAR(50)	FALSE	FALSE	FALSE	Etiqueta para opción 13
cv14	CHAR(50)	FALSE	FALSE	FALSE	Etiqueta para opción 14
cv15	CHAR(50)	FALSE	FALSE	FALSE	Etiqueta para opción 15
id_preguntas	Integer	FALSE	TRUE	TRUE	Pregunta a la que pertenece
id_tipos_respuestas	Integer	FALSE	TRUE	TRUE	Tipo de respuesta a la que pertenece

Tabla: nivel_educacion

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	CHAR(3)	FALSE	FALSE	FALSE	Id autonumérico de la tabla
nombre	CHAR(10)	FALSE	FALSE	FALSE	Opción de nivel o año de educación

Tabla: estado_civil

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	CHAR(1)	FALSE	FALSE	FALSE	Id autonumérico de la tabla
nombre	CHAR(12)	FALSE	FALSE	FALSE	Opción de estado civil

Tabla: parentezco

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	CHAR(2)	FALSE	FALSE	FALSE	Id autonumérico de la tabla
nombre	CHAR(12)	FALSE	FALSE	FALSE	Opción de posición en el grupo familiar

Tabla: ocupacion

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	CHAR(2)	FALSE	FALSE	FALSE	Id autonumérico de la tabla
nombre	CHAR(12)	FALSE	FALSE	FALSE	Opción de ocupación

Tabla: religion

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	CHAR(2)	FALSE	FALSE	FALSE	Id autonumérico de la tabla
nombre	CHAR(20)	FALSE	FALSE	FALSE	Opción de religión

Tabla: anios

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
año_censo	Integer	FALSE	FALSE	FALSE	Años para los censos

Tabla 6.5 Tablas de la base de datos: proyectos

NOMBRE DE TABLA	DESCRIPCIÓN
tipo_proyecto	Contiene una lista de los posibles tipos de proyectos.
cargos	Contiene datos de los cargos en una institución.
usuarios	Contiene datos de los usuarios del sistema.
historial_accesos	Contiene datos de acceso de los usuarios al sistema.
programas	Contiene los datos principales de los programas que se registren.
fin_prog	Contiene la finalidad de un programa determinado.
proposito_prog	Contiene los propósitos de la finalidad de un programa.
resultados_prog	Contiene los resultados de un propósito y programa determinados.
actividades_prog	Contiene las actividades de un resultado y programa determinados.
subactividades_prog	Contiene las subactividades de una actividad y programa determinados.
maximos	Contiene una lista de datos que limita el número máximo de registros para el Marco Lógico.
proyectos	Contiene los datos principales de los proyectos que se registren.
fin	Contiene la finalidad de un proyecto determinado.
proposito	Contiene el propósito de la finalidad de un proyecto.
resultados	Contiene los resultados de un propósito y proyecto determinados.
anios_resultado	Contiene datos de los años de desarrollo de un proyecto.
actividades	Contiene las actividades de un resultado y proyecto determinados.
estado_actividades	Contiene el estado de las actividades de un proyecto determinado.
subactividades	Contiene las subactividades de una actividad y proyecto determinados
tipo_gasto	Contiene una lista de los tipos de gasto que se realiza para cada actividad.
recursos_economicos	Contiene datos de los diferentes gastos que se realizan para cada actividad de proyecto.
adjuntos	Contiene datos de la ubicación de los diferentes informes o documentos que tenga un programa o un proyecto.
responsables	Contiene datos de los responsables secundarios de las actividades de un proyecto.

Tabla: tipo_proyecto

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonumérico de la tabla
nombre	CHAR(50)	FALSE	FALSE	FALSE	Opción de tipo de proyecto

Tabla: cargos

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonumérico de la tabla
nombre	CHAR(50)	FALSE	FALSE	FALSE	Nombre de cargo en la institución
funciones	CHAR(500)	FALSE	FALSE	FALSE	Funciones para el cargo

Tabla: usuarios

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonumérico de la tabla
nombre	CHAR(50)	FALSE	FALSE	FALSE	Nombre de usuario
alias	CHAR(10)	FALSE	FALSE	FALSE	Alias de usuario
clave	CHAR(10)	FALSE	FALSE	FALSE	Clave de usuario
tipo	CHAR(3)	FALSE	FALSE	FALSE	Tipo de usuario
id_cargo	Integer	FALSE	TRUE	TRUE	Cargo al que pertenece
fecha_alta	Date	FALSE	FALSE	TRUE	Fecha de creación del usuario

Tabla: historial_accesos

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonumérico de la tabla
id_usuario	Integer	FALSE	TRUE	TRUE	Usuario que accesa al sistema
fecha_sesion	Date	FALSE	FALSE	TRUE	Fecha de acceso al sistema
hora_ini	Integer	FALSE	FALSE	TRUE	Hora de inicio de sesión
hora_fin	Integer	FALSE	FALSE	FALSE	Hora de fin de sesión
duracion_min	Integer	FALSE	FALSE	FALSE	Duración de la sesión

Tabla: programas

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonumérico de la tabla
codigo	CHAR(30)	FALSE	FALSE	FALSE	Código de programa
nombre	CHAR(255)	FALSE	FALSE	FALSE	Nombre de programa
pais	CHAR(10)	FALSE	FALSE	FALSE	País en donde se desarrolla el programa
provincia	CHAR(50)	FALSE	FALSE	FALSE	Provincia en donde se desarrolla el programa
canton	CHAR(50)	FALSE	FALSE	FALSE	Cantón(es) en donde se desarrolla el programa
parroquia	CHAR(255)	FALSE	FALSE	FALSE	Parroquia(s) en donde se desarrolla el programa
comunidad	CHAR(500)	FALSE	FALSE	FALSE	Comunidad(es) en donde se desarrolla el programa
num_beneficiarios_dir	Integer	FALSE	FALSE	TRUE	Número de beneficiarios directos del programa
num_beneficiarios_indir	Integer	FALSE	FALSE	TRUE	Número de beneficiarios indirectos del programa
fecha_inicio	Date	FALSE	FALSE	TRUE	Fecha de inicio del programa
fecha_fin_programada	Date	FALSE	FALSE	TRUE	Fecha de fin programada del programa
fecha_fin_ejecutada	Date	FALSE	FALSE	FALSE	Fecha de fin de ejecución del programa
duracion	Integer	FALSE	FALSE	TRUE	Duración del programa

mes_año	CHAR(1)	FALSE	FALSE	FALSE	Indica la duración en meses o años
inversion total programada	Float	FALSE	FALSE	TRUE	Inversión total programada
inversion_total_ejecutada	Float	FALSE	FALSE	FALSE	Inversión total ejecutada
inversion_solicitada	Float	FALSE	FALSE	TRUE	Inversión solicitada
inversion_aporte_com	Float	FALSE	FALSE	FALSE	Inversión de aporte comunitario
inversion_otras_org	Float	FALSE	FALSE	FALSE	Inversión de otras organizaciones
inversion_no_monetaria	Float	FALSE	FALSE	FALSE	Inversión no monetaria
obsr_no_monetaria	CHAR(255)	FALSE	FALSE	FALSE	Observaciones de inversión no monetaria
obsr_beneficiarios_dir	CHAR(255)	FALSE	FALSE	FALSE	Observaciones de beneficiarios directos
obsr_beneficiarios_indir	CHAR(255)	FALSE	FALSE	FALSE	Observaciones de beneficiarios indirectos
tipo_beneficiarios_dir	CHAR(1)	FALSE	FALSE	FALSE	Tipo de beneficiarios directos
tipo_beneficiarios_indir	CHAR(1)	FALSE	FALSE	FALSE	Tipo de beneficiarios indirectos
presupuesto_1	Float	FALSE	FALSE	FALSE	Presupuesto para mes o año 1
presupuesto_2	Float	FALSE	FALSE	FALSE	Presupuesto para mes o año 2
presupuesto_3	Float	FALSE	FALSE	FALSE	Presupuesto para mes o año 3
presupuesto_4	Float	FALSE	FALSE	FALSE	Presupuesto para mes o año 4
presupuesto_5	Float	FALSE	FALSE	FALSE	Presupuesto para mes o año 5
presupuesto_6	Float	FALSE	FALSE	FALSE	Presupuesto para mes o año 6
presupuesto_7	Float	FALSE	FALSE	FALSE	Presupuesto para mes o año 7
presupuesto_8	Float	FALSE	FALSE	FALSE	Presupuesto para mes o año 8
presupuesto_9	Float	FALSE	FALSE	FALSE	Presupuesto para mes o año 9
presupuesto_10	Float	FALSE	FALSE	FALSE	Presupuesto para mes o año 10
presupuesto_11	Float	FALSE	FALSE	FALSE	Presupuesto para mes o año 11
presupuesto_12	Float	FALSE	FALSE	FALSE	Presupuesto para mes o año 12
cumplido	CHAR(1)	FALSE	FALSE	FALSE	Indica si el programa está cumplido o no
responsable_tec_fin	CHAR(255)	FALSE	FALSE	FALSE	Nombre de responsable financiero
responsable_jur_fin	CHAR(255)	FALSE	FALSE	FALSE	Nombre de responsable jurídico

Tabla: fin_prog

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonumérico de la tabla
fin	CHAR(500)	FALSE	FALSE	FALSE	Finalidad del programa
indicadores	CHAR(500)	FALSE	FALSE	FALSE	Indicadores de la finalidad de programa
	CHAR(500)	FALSE	FALSE	FALSE	Medios de Verificación de la

medios_verificacion					finalidad de programa
supuestos	CHAR(500)	FALSE	FALSE	FALSE	Supuestos de la finalidad de programa
cumplido	CHAR(1)	FALSE	FALSE	FALSE	Indica si la finalidad de programa está cumplida o no
id_programa	Integer	FALSE	TRUE	TRUE	Programa al que pertenece

Tabla: proposito_prog

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonumérico de la tabla
proposito	CHAR(500)	FALSE	FALSE	FALSE	Propósito de programa
indicadores	CHAR(500)	FALSE	FALSE	FALSE	Indicadores de propósito de programa
medios_verificacion	CHAR(500)	FALSE	FALSE	FALSE	Medios de Verificación de propósito de programa
supuestos	CHAR(500)	FALSE	FALSE	FALSE	Supuestos de propósito de programa
fecha_inicio	Date	FALSE	FALSE	TRUE	Fecha de inicio del propósito de programa
fecha_fin_programada	Date	FALSE	FALSE	TRUE	Fecha de fin programada del propósito de programa
fecha_fin_ejecutada	Date	FALSE	FALSE	FALSE	Fecha de fin ejecutada del propósito de programa
cumplido	CHAR(1)	FALSE	FALSE	FALSE	Indica si el propósito de programa está cumplido o no
id_fin_prog	Integer	FALSE	TRUE	TRUE	Finalidad de programa al que pertenece

Tabla: resultados_prog

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonumérico de la tabla
resultado	CHAR(500)	FALSE	FALSE	FALSE	Componente de programa
indicadores	CHAR(500)	FALSE	FALSE	FALSE	Indicadores de componente de programa
medios_verificacion	CHAR(500)	FALSE	FALSE	FALSE	Medios de verificación de componente de programa
supuestos	CHAR(500)	FALSE	FALSE	FALSE	Supuestos de componente de programa
fecha_inicio	Date	FALSE	FALSE	TRUE	Fecha de inicio de componente de programa
fecha_fin_programada	Date	FALSE	FALSE	TRUE	Fecha de fin programada de componente de programa
fecha_fin_ejecutada	Date	FALSE	FALSE	FALSE	Fecha de fin ejecutada de componente de programa
cumplido	CHAR(1)	FALSE	FALSE	FALSE	Indica si el componente de programa está cumplido o no
id_proposito_prog	Integer	FALSE	TRUE	TRUE	Propósito al que pertenece

Tabla: actividades_prog

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
-------	------	----------------	---------------	----------	-------------

id	SERIAL	TRUE	FALSE	TRUE	Id autonumérico de la tabla
actividad	CHAR(500)	FALSE	FALSE	FALSE	Actividad de programa
cumplido	CHAR(1)	FALSE	FALSE	FALSE	Indica si la actividad de programa está cumplida o no
id_resultados_prog	Integer	FALSE	TRUE	TRUE	Componente al que pertenece

Tabla: subactividades_prog

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonumérico de la tabla
indicadores	CHAR(500)	FALSE	FALSE	FALSE	Indicadores de actividad de programa
medios verificacion	CHAR(500)	FALSE	FALSE	FALSE	Medios de verificación de actividad de programa
supuestos	CHAR(500)	FALSE	FALSE	FALSE	Supuestos de actividad de programa
id_actividades_prog	Integer	FALSE	TRUE	TRUE	Actividad a la que pertenece

Tabla: maximos

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonumérico de la tabla
nombre	CHAR(10)	FALSE	FALSE	FALSE	Parámetro de Marco Lógico
maximo	Integer	FALSE	FALSE	TRUE	Número máximo de registros permitidos

Tabla: proyectos

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonumérico de la tabla
codigo	CHAR(30)	FALSE	FALSE	FALSE	Código de proyecto
nombre	CHAR(255)	FALSE	FALSE	FALSE	Nombre de proyecto
pais	CHAR(10)	FALSE	FALSE	FALSE	País en donde se desarrolla el proyecto
provincia	CHAR(50)	FALSE	FALSE	FALSE	Provincia en donde se desarrolla el proyecto
canton	CHAR(50)	FALSE	FALSE	FALSE	Cantón(es) en donde se desarrolla el proyecto
parroquia	CHAR(255)	FALSE	FALSE	FALSE	Parroquia(s) en donde se desarrolla el proyecto
comunidad	CHAR(500)	FALSE	FALSE	FALSE	Comunidad(es) en donde se desarrolla el proyecto
org_ejecutora	CHAR(255)	FALSE	FALSE	FALSE	Organización que ejecuta el proyecto
org_financiera	CHAR(255)	FALSE	FALSE	FALSE	Organización que financia el proyecto
num_beneficiarios_dir	Integer	FALSE	FALSE	TRUE	Número de beneficiarios directos del proyecto
num_beneficiarios_indir	Integer	FALSE	FALSE	TRUE	Número de beneficiarios indirectos del proyecto
fecha_inicio	Date	FALSE	FALSE	TRUE	Fecha de inicio del proyecto
fecha_fin_programada	Date	FALSE	FALSE	TRUE	Fecha de fin programada del

					proyecto
fecha_fin_ejecutada	Date	FALSE	FALSE	FALSE	Fecha de fin de ejecución del proyecto
duracion	Integer	FALSE	FALSE	TRUE	Duración del proyecto
mes_anio	CHAR(1)	FALSE	FALSE	FALSE	Indica la duración en meses o años
inversion_total_programada	Float	FALSE	FALSE	TRUE	Inversión total programada
inversion_total_ejecutada	Float	FALSE	FALSE	FALSE	Inversión total ejecutada
inversion_solicitada	Float	FALSE	FALSE	TRUE	Inversión solicitada
inversion_aporte_com	Float	FALSE	FALSE	FALSE	Inversión de aporte comunitario
inversion_otras_org	Float	FALSE	FALSE	FALSE	Inversión de otras organizaciones
inversion_no_monetaria	Float	FALSE	FALSE	FALSE	Inversión no monetaria
obsr_no_monetaria	CHAR(255)	FALSE	FALSE	FALSE	Observaciones de inversión no monetaria
obsr_beneficiarios_dir	CHAR(255)	FALSE	FALSE	FALSE	Observaciones de beneficiarios directos
obsr_beneficiarios_indir	CHAR(255)	FALSE	FALSE	FALSE	Observaciones de beneficiarios indirectos
tipo_beneficiarios_dir	CHAR(1)	FALSE	FALSE	FALSE	Tipo de beneficiarios directos
tipo_beneficiarios_indir	CHAR(1)	FALSE	FALSE	FALSE	Tipo de beneficiarios indirectos
presupuesto_1	Float	FALSE	FALSE	FALSE	Presupuesto para mes/año 1
presupuesto_2	Float	FALSE	FALSE	FALSE	Presupuesto para mes/año 2
presupuesto_3	Float	FALSE	FALSE	FALSE	Presupuesto para mes/año 3
presupuesto_4	Float	FALSE	FALSE	FALSE	Presupuesto para mes/año 4
presupuesto_5	Float	FALSE	FALSE	FALSE	Presupuesto para mes/año 5
presupuesto_6	Float	FALSE	FALSE	FALSE	Presupuesto para mes/año 6
presupuesto_7	Float	FALSE	FALSE	FALSE	Presupuesto para mes/año 7
presupuesto_8	Float	FALSE	FALSE	FALSE	Presupuesto para mes/año 8
presupuesto_9	Float	FALSE	FALSE	FALSE	Presupuesto para mes/año 9
presupuesto_10	Float	FALSE	FALSE	FALSE	Presupuesto para mes/año 10
presupuesto_11	Float	FALSE	FALSE	FALSE	Presupuesto para mes/año 11
presupuesto_12	Float	FALSE	FALSE	FALSE	Presupuesto para mes/año 12
cumplido	CHAR(1)	FALSE	FALSE	FALSE	Indica si el proyecto está cumplido o no
id_programas	Integer	FALSE	TRUE	TRUE	Programa al que pertenece
id_resultados_prog	Integer	FALSE	TRUE	TRUE	Componente al que pertenece
nombre_tipo_proyecto	CHAR(50)	FALSE	FALSE	FALSE	Tipo de proyecto

Tabla: fin

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonumérico de la tabla
fin	CHAR(500)	FALSE	FALSE	FALSE	Finalidad de proyecto
indicadores	CHAR(500)	FALSE	FALSE	FALSE	Indicadores de finalidad de proyecto
medios_verificacion	CHAR(500)	FALSE	FALSE	FALSE	Medios de verificación de finalidad de proyecto
supuestos	CHAR(500)	FALSE	FALSE	FALSE	Supuestos de finalidad de

					proyecto
cumplido	CHAR(1)	FALSE	FALSE	FALSE	Indica si el proyecto está cumplido o no
id_proyecto	Integer	FALSE	TRUE	TRUE	Proyecto al que pertenece

Tabla: proposito

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonumérico de la tabla
proposito	CHAR(500)	FALSE	FALSE	FALSE	Propósito de proyecto
indicadores	CHAR(500)	FALSE	FALSE	FALSE	Indicadores de propósito de proyecto
medios verificacion	CHAR(500)	FALSE	FALSE	FALSE	Medios de verificación de propósito de proyecto
supuestos	CHAR(500)	FALSE	FALSE	FALSE	Supuestos de propósito de proyecto
cumplido	CHAR(1)	FALSE	FALSE	FALSE	Indica si el propósito está cumplido o no
id_fin	Integer	FALSE	TRUE	TRUE	Finalidad a la que pertenece

Tabla: resultados

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonumérico de la tabla
resultado	CHAR(500)	FALSE	FALSE	FALSE	Componente de proyecto
indicadores	CHAR(500)	FALSE	FALSE	FALSE	Indicadores de componente de proyecto
medios_verificacion	CHAR(500)	FALSE	FALSE	FALSE	Medios de verificación de componente de proyecto
supuestos	CHAR(500)	FALSE	FALSE	FALSE	Supuestos de componente de proyecto
cumplido	CHAR(1)	FALSE	FALSE	FALSE	Indica si el componente de proyecto está cumplido o no
id_proposito	Integer	FALSE	TRUE	TRUE	Propósito al que pertenece

Tabla: anios_resultado

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonumérico de la tabla
id_resultado	Integer	FALSE	TRUE	TRUE	Componente al que pertenece
anio_cont	Integer	FALSE	FALSE	TRUE	Contador de los años que dura el proyecto
fecha_inicio	Date	FALSE	FALSE	TRUE	Fecha inicio de año de proyecto
fecha_fin	Date	FALSE	FALSE	TRUE	Fecha fin de año de proyecto

Tabla: actividades

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonumérico de la tabla
actividad	CHAR(500)	FALSE	FALSE	FALSE	Actividad de proyecto
fecha_inicio	Date	FALSE	FALSE	TRUE	Fecha inicio de actividad de proyecto
fecha_fin_programada	Date	FALSE	FALSE	TRUE	Fecha fin programada de

					actividad de proyecto
fecha_fin_ejecutada	Date	FALSE	FALSE	FALSE	Fecha fin ejecutada de actividad de proyecto
duracion	Integer	FALSE	FALSE	FALSE	Duración de la actividad de proyecto
dia_mes_anio	CHAR(1)	FALSE	FALSE	FALSE	Duración de proyecto en meses o años
id_responsable_primario_1	Integer	FALSE	TRUE	TRUE	Responsable primario de actividad de proyecto
responsable_primario_2	CHAR(50)	FALSE	FALSE	FALSE	Segundo Responsable primario de actividad de proyecto
responsable_secundario_1	CHAR(300)	FALSE	FALSE	FALSE	Responsable secundario de actividad de proyecto
presupuesto_total_programado	Float	FALSE	FALSE	TRUE	Presupuesto total programado de actividad de proyecto
gasto_total_ejecutado	Float	FALSE	FALSE	FALSE	Gasto total ejecutado de actividad de proyecto
observaciones	CHAR(255)	FALSE	FALSE	FALSE	Observaciones de actividad de proyecto
cumplido	CHAR(1)	FALSE	FALSE	FALSE	Indica si la actividad de proyecto está cumplida o no
id_anio_resultado	Integer	FALSE	TRUE	TRUE	Año al que pertenece la actividad

Tabla: estado_actividades

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonumérico de la tabla
id_actividad	Integer	FALSE	TRUE	TRUE	Actividad a la que pertenece
estado	CHAR(1)	FALSE	FALSE	FALSE	Estado de la actividad de proyecto
fecha	Date	FALSE	FALSE	TRUE	Fecha de estado de la actividad de proyecto
observacion	CHAR(500)	FALSE	FALSE	FALSE	Observaciones adicionales

Tabla: subactividades

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonumérico de la tabla
indicadores	CHAR(500)	FALSE	FALSE	FALSE	Indicadores de la actividad de proyecto
medios_verificacion	CHAR(500)	FALSE	FALSE	FALSE	Medios de verificación de la actividad de proyecto
supuestos	CHAR(500)	FALSE	FALSE	FALSE	Supuestos de la actividad de proyecto
presupuesto	Float	FALSE	FALSE	TRUE	Presupuesto de la actividad de proyecto
gasto	Float	FALSE	FALSE	FALSE	Gasto de la actividad de proyecto
recursos materiales	CHAR(255)	FALSE	FALSE	FALSE	Recursos materiales de la actividad de proyecto
producto entregado	CHAR(50)	FALSE	FALSE	FALSE	Producto entregado de la actividad de proyecto
cantidad programada	Float	FALSE	FALSE	FALSE	Cantidad programada del producto entregado de la actividad de proyecto
cantidad ejecutada	Float	FALSE	FALSE	FALSE	Cantidad ejecutada del producto entregado de la

					actividad de proyecto
unidad_de_medida	CHAR(70)	FALSE	FALSE	FALSE	Unidad de medida del producto entregado
observaciones	CHAR(255)	FALSE	FALSE	FALSE	Observaciones adicionales
id actividades	Integer	FALSE	TRUE	TRUE	Actividad de proyecto a la que pertenece

Tabla: tipo_gasto

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonumérico de la tabla
nombre	CHAR(20)	FALSE	FALSE	FALSE	Opción de origen de gasto

Tabla: recursos_economicos

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	FALSE	FALSE	FALSE	Id autonumérico de la tabla
descripcion	CHAR(255)	FALSE	FALSE	FALSE	Descripción de gasto de actividad de proyecto
cantidad	Float	FALSE	FALSE	TRUE	Cantidad de gasto de actividad de proyecto
referencia_doc_cont	CHAR(20)	FALSE	FALSE	FALSE	Documento de referencia de gasto de actividad de proyecto
fecha_gasto	Date	FALSE	FALSE	TRUE	Fecha de gasto de actividad de proyecto
tipo_gasto	CHAR(20)	FALSE	FALSE	FALSE	Origen de gasto de actividad de proyecto
id_subactividades	Integer	FALSE	TRUE	TRUE	Subactividad de proyecto a la que pertenece

Tabla: adjuntos

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonumérico de la tabla
pertenencia_pri	CHAR(3)	FALSE	FALSE	FALSE	Indica si el archivo adjunto pertenece a programa o proyecto
pertenencia_sec	CHAR(3)	FALSE	FALSE	FALSE	Indica si el archivo adjunto pertenece al marco lógico de programa o proyecto
titulo	CHAR(300)	FALSE	FALSE	FALSE	Título del archivo adjunto
ruta_anexo	CHAR(500)	FALSE	FALSE	FALSE	Ruta del archivo adjunto
id_pertenencia_prog	Integer	FALSE	TRUE	FALSE	Programa al que pertenece
id_pertenencia_proy	Integer	FALSE	TRUE	FALSE	Proyecto al que pertenece

Tabla: responsables

Campo	Tipo	Clave Primaria	Clave Foránea	Not Null	Descripción
id	SERIAL	TRUE	FALSE	TRUE	Id autonumérico de la tabla
nombre	CHAR(50)	FALSE	FALSE	FALSE	Nombres de responsables secundarios para actividades de proyecto

6.4.2 DISEÑO DE PÁGINAS WEB DINÁMICAS

Para el diseño de las páginas web se utilizó Dreamweaver MX de Macromedia. Con esta herramienta fue posible realizar paralelamente el diseño y la programación de las páginas. Dreamweaver permite incrustar cualquier tipo de elemento *HTML* para el diseño físico de la página web (tablas, formularios, botones, enlaces, imágenes, etc.), además facilita la inserción de código de lado del servidor, en nuestro caso: *PHP* y permite la interacción con *JavaScript* y Hojas de Estilo.

Los siguientes aspectos fueron tomados en cuenta para el diseño de las páginas:

- Todos los elementos están insertados y alineados por medio de tablas.
- Las proporciones de todos los elementos están de acuerdo al tamaño de la página.
- La longitud de los campos de los formularios está de acuerdo a la longitud máxima establecida en los campos de la base de datos.
- Las páginas están basadas en la misma hoja de estilo.
- Para la programación del lado del cliente se usó *JavaScript*.
- Para la programación del lado del servidor se usó *PHP*.
- Las imágenes son de tipo GIF.
- Se ha tomado en cuenta que las páginas no tengan muchas imágenes para que el acceso sea más rápido.
- Para obtener una fácil interacción con el usuario, se despliegan errores, avisos, mensajes y cuadros de diálogo de confirmación.
- Se cuenta con un menú principal, todas las páginas tienen enlaces hacia dicho menú. También existen enlaces en cada una de las páginas, esos enlaces están relacionados con la página en la que se encuentran.

6.5 DESARROLLO DE LA APLICACIÓN

Los detalles sobre el desarrollo de la aplicación se encuentran descritos en el Manual Técnico, ubicado en el CD adjunto a este documento en: CD_TESIS:\Tesis\Manuales\Manual Técnico.pdf.

6.6 IMPLEMENTACIÓN DEL PROTOTIPO DE LA APLICACIÓN

El Sistema Prototipo de Gestión y Control de Proyectos de Inversión fue diseñado para trabajar en Internet o en una Intranet, por lo tanto cumple con las características de toda aplicación web. La aplicación web se desarrolló para Linux y Windows utilizando las siguientes herramientas:

Tabla 6.2 Herramientas utilizadas en la implementación del Sistema Prototipo

	LINUX	WINDOWS
Sistema Operativo:	Linux Red Hat 7.2	Windows 2000 Server
Servidor Web:	Apache Web Server 1.3.20	Apache Web Server 1.3.12
Servidor de Base de Datos:	PostgreSQL 7.1.3	PostgreSQL 7.3.3 bajo Cygwin y CygIPC.
Herramienta del lado del servidor:	PHP 4.0.6 instalado como módulo API del servidor	PHP 4.0.4 instalado como API CGI del servidor
Herramienta de lado del cliente:	JavaScript	JavaScript
Front End:	Navegador web Mozilla 0.9.2.1.	Navegador web Internet Explorer 4.0 en adelante.
Diseñador de páginas web:	Editor Kate y Dreamweaver MX de Macromedia.	Dreamweaver MX de Macromedia.
Lenguaje de marcas y hojas de estilo:	HTML 4.0, CSS.	HTML 4.0, CSS.

Como resultado de la utilización de estas herramientas, el sistema prototipo se encuentra estructurado como se indica en la figura **Fig. 6.2**.

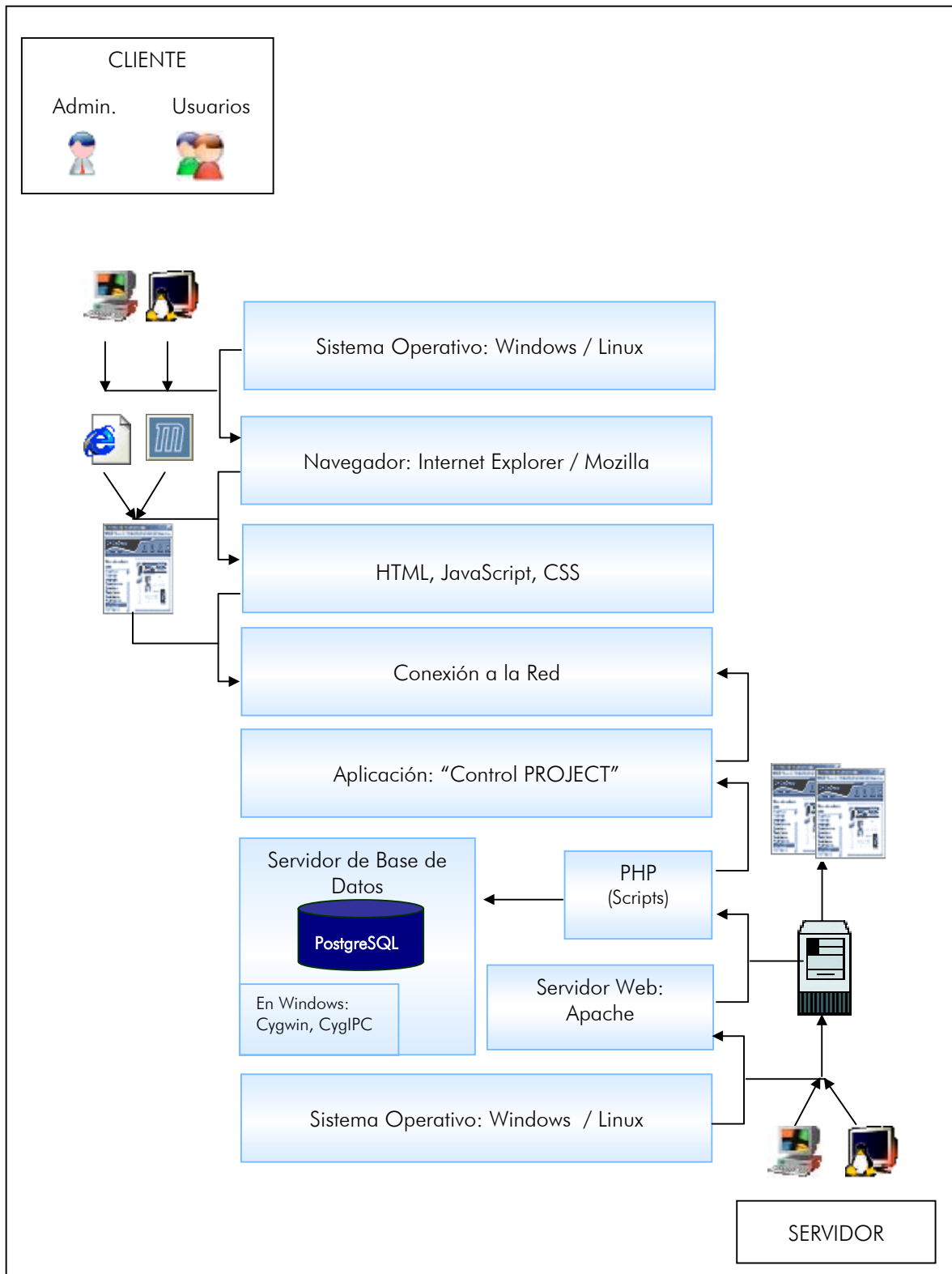
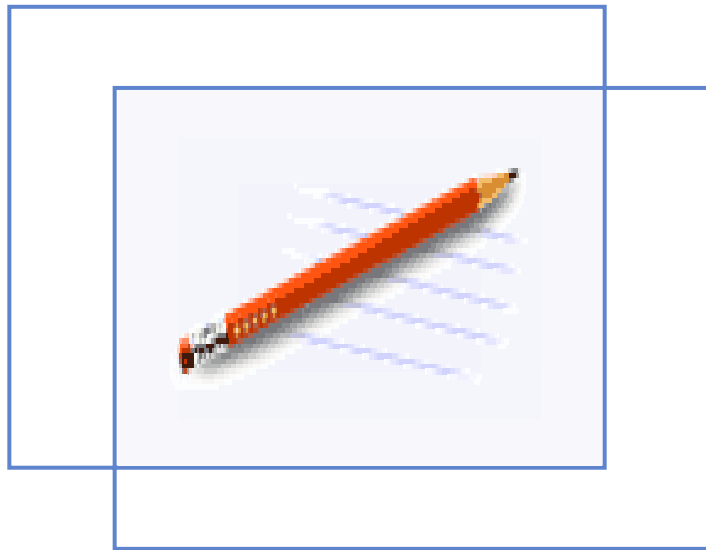


Fig. 6.2 Arquitectura del Sistema Prototipo de Gestión y Control de Proyectos de Inversión "Control PROJECT"

6.7 DOCUMENTACIÓN

La documentación de la aplicación web: Sistema de Gestión y Control de Proyectos de Inversión se encuentra detallada en los Manuales Técnico y de Usuario, ubicados en el CD adjunto a este documento en: CD_TESIS:\Tesis\Manuales\Manual Técnico.pdf y Manual de Usuario.pdf respectivamente.

Capítulo 7



CONCLUSIONES Y RECOMENDACIONES

- ❖ Verificación de hipótesis
- ❖ Conclusiones
- ❖ Recomendaciones

7.1 VERIFICACIÓN DE HIPÓTESIS

La Hipótesis planteada al inicio de la Tesis fue la siguiente:

“El estudio y análisis comparativo de herramientas para desarrollo de aplicaciones web dinámicas en conjunto con su respectivo Servidor Web y Base de Datos, permitirán establecer dichas herramientas como tecnologías alternativas y adaptables de acuerdo a las necesidades de cada medio y plataforma de desarrollo.”

Al finalizar este trabajo se ha logrado establecer alternativas formadas por: herramientas de generación de contenido dinámico(*PHP, ASP, JSP*), servidores web (Apache Web Server, Internet Information Server, Tomcat Web Server) y la base de datos (PostgreSQL) que se adaptan a las plataformas Windows y Linux.

Dichas alternativas clasificamos por Código Abierto y Propietario, permitiendo decidir y utilizar la integración más adecuada según las necesidades, infraestructura tecnológica y disponibilidad económica de la empresa o institución.

Por lo expuesto anteriormente se ha comprobado la hipótesis planteada.

7.2 CONCLUSIONES

En el presente trabajo de Tesis se concluye lo siguiente:

RESPECTO DEL ESTUDIO

- Del estudio realizado en este tema de Tesis, se logró realizar tablas comparativas de las herramientas de generación de contenido dinámico (*PHP, ASP y JSP*) y servidores web (Apache, Zeus, Internet Information Server, Personal Web Server, Tomcat e iPlanet) sobre aspectos técnicos que deben ser tomados en cuenta para desarrollar aplicaciones web.
- Luego de la comparación realizada en el capítulo 2, se concluye que *PHP* es la única herramienta que nació para Internet, ya que las otras herramientas comparadas se derivaron de otros lenguajes existentes, así por ejemplo: *ASP* se deriva de Visual Basic y *VBScript* y *JSP* se deriva de *Java* y *Servlets*.
- Se encontró aspectos favorables y desfavorables de las herramientas de generación de

contenido dinámico y de los servidores web. En las herramientas de generación de contenido dinámico: *PHP* tiene mayor número de funciones que *ASP* y *JSP*, pero no cuenta con opciones para navegar entre registros fácilmente; *ASP* cuenta con las características de programación de Visual Basic, pero no es multiplataforma, a menos que se instale el software *Chili!Soft ASP*; *JSP* al utilizar *Java* posee las mismas ventajas de éste, pero contiene menos funciones que *PHP*. Y respecto a los servidores web: Apache es el servidor web más utilizado y es multiplataforma; Internet Information Server no es multiplataforma y Tomcat es un servidor web que en la mayoría de los casos necesita de la cooperación de otro servidor web.

- Se conoció el manejo y funcionamiento de la base de datos PostgreSQL en Windows y en Linux. El manejo de la base de datos en las dos plataformas es igual, excepto en la ejecución de scripts que contengan registros en las tablas de una base de datos que se transporte de Windows a Linux, ya que la diferencia de versiones provoca inconvenientes al ejecutar el comando `pg_dump`. El problema surge al insertar los registros que contengan campos nulos, por la diferencia de representación.
- PostgreSQL se puede conectar con las tres herramientas de generación de contenido dinámico: *PHP*, *ASP* y *JSP*, tanto en Windows como en Linux y está al nivel de las bases de datos comerciales como Oracle y MS SQL Server. Cabe destacar que PostgreSQL para ser instalado en Windows requiere de los instaladores *Cygwin* y *CyglIPC*, que son *DLLs* que emulan el entorno Linux en Windows.
- Con las tablas comparativas de las herramientas de generación de contenido dinámico, servidores web y la base de datos, se plantearon alternativas de código abierto y propietario para el desarrollo de aplicaciones web en Windows y Linux.
- Se realizó la instalación y configuración de *PHP*, Apache Web Server y PostgreSQL tanto en Windows como en Linux, lo que permitió conocer acerca de la funcionalidad e integración de las tres herramientas para el desarrollo del aplicativo.
- Se aprendió que la programación web permite tener procesamiento y presentación separados o juntos. En ambos casos se debe tomar en cuenta las características que deben cumplir las aplicaciones web para que su carga sea óptima y liviana. De lo que podemos concluir, es más conveniente separar la programación de la presentación, ya que si se requiere de cambios en cualquiera de los dos aspectos, se facilita su implementación.

- Las aplicaciones web cambian los conceptos preestablecidos acerca de la programación de aplicaciones tradicionales. Cuestiones de seguridad, sesiones, presentación e implantación toman un giro diferente.
- Se aprendió los pasos que se deben seguir para la Gestión y Control de Proyectos de Inversión, dentro de los cuales el Marco Lógico es la estructura principal que permite resumir los elementos importantes de un proyecto, tales como: finalidad, propósito(s), componentes o resultados, actividades, indicadores, medios de verificación y supuestos.

RESPECTO DEL APLICATIVO

- Al usar las alternativas de código abierto para el desarrollo de aplicaciones web, se obtienen sistemas de información robustos, de gran desempeño, estabilidad, escalabilidad y a un costo menor al que se tendría usando alternativas propietarias. Teniendo como característica principal de ser multiplataforma, permitiendo portar las aplicaciones de una plataforma a otra sin mayores cambios.
- Con la investigación preliminar sobre Gestión y Control de Proyectos de Inversión y el análisis FODA de sistemas y aplicaciones destinados para este fin, se logró establecer los lineamientos para el diseño y desarrollo de nuestro sistema prototipo que se enfoca en la etapa de ejecución y parte de la etapa de finalización.
- Se desarrolló un sistema prototipo de Gestión y Control de Proyectos de Inversión que registra y controla la información del Marco Lógico, permite seguir y monitorear las actividades, por cada año de desarrollo de los proyectos, con sus respectivos recursos asignados y tiempos establecidos, para generar reportes con información oportuna y adecuada para la toma de decisiones.
- En *PHP* el manejo de sesiones depende de la versión del lenguaje y del navegador que interprete las páginas web. Y no cuenta con funciones para navegar entre registros, ni existen elementos que simulen grids, ni barras de registros(recordset). Al implementar navegadores de registros con *PHP* y *JavaScript* se obtiene resultados lentos, ya que la página se recarga con cada clic en el botón del navegador de registros.
- La configuración del archivo *httpd.conf* de Apache Web Server y *php.ini* de *PHP* en Windows y Linux fue muy similar, las directivas de configuración son las mismas, excepto por ciertos parámetros que tienen que ver con las rutas de acceso e instalación de Apache y *PHP*, que obviamente son diferentes en cada sistema operativo.

7.3 RECOMENDACIONES

En el presente trabajo de Tesis se concluye lo siguiente:

RESPECTO DEL ESTUDIO

- Se recomienda el estudio a fondo de *CFML*(ColdFusion) para utilizarlo como herramienta de desarrollo en futuros proyectos de tesis, ya que según nuestro estudio es la mejor opción, comparada con *ASP* en la categoría de alternativas propietarias, si se cuenta con el presupuesto necesario. Al utilizar esta herramienta se aprovecharía la funcionalidad que presta: administración, monitoreo, diseño, desarrollo, depuración y manejo de seguridad de una aplicación web.
- Estudiar y evaluar servidores de aplicaciones para desarrollo de aplicaciones web en las plataformas: Windows y Linux, ya que son auténticos multiservidores capaces de contener en una sola herramienta los elementos que se han estudiado en este proyecto de tesis.
- Hacer un análisis comparativo de la herramienta de generación de contenido dinámico *ASP* para plataformas Windows y Linux mediante el software *Chili!Soft ASP* de Sun, para establecer ventajas y desventajas ante las versiones de *ASP* y *ASP.NET* de Microsoft.
- Debido a que Internet crece día a día, es importante investigar las nuevas herramientas que salen al mercado.

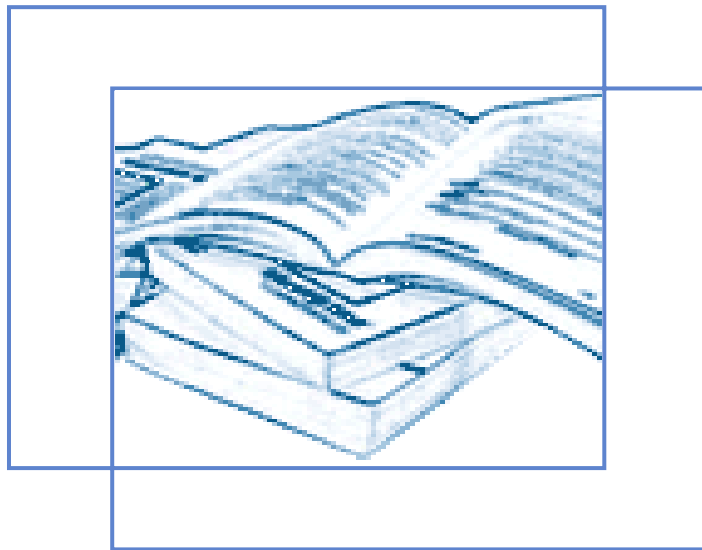
RESPECTO DEL APLICATIVO

- Utilizar herramientas de código abierto para el desarrollo de aplicaciones web, ya que muestran ventajas económicas y técnicas con respecto a las de código propietario.
- Aprovechar de mejor manera las funcionalidades de Dreamweaver MX para el diseño y desarrollo de sitios y aplicaciones web.
- Implementar un módulo adicional para el Sistema Prototipo de Gestión y Control de Proyectos de Inversión, que permita ver los resultados de las encuestas y censos en forma gráfica(barras, pies estadísticos, etc.).
- Ampliar el alcance del Sistema Prototipo de Gestión y Control de Proyectos de Inversión, para que abarque desde la etapa de Diseño de programas y proyectos.

POSIBLES TEMAS DE TESIS

- Estudio del lenguaje *CFML* y su servidor ColdFusion para desarrollo de aplicaciones web.
- Estudio y análisis de *ASP* para plataformas Windows y Linux mediante el software *Chili!Soft ASP* de Sun, para establecer ventajas y desventajas ante las versiones de *ASP* y *ASP.NET* de Microsoft.
- Estudio de *J2EE* para la implementación de portales y aplicaciones web.
- Estudio e implementación de redes ópticas e integración de IP sobre WDM (Wavelength Division Multiplexing - Multiplexación en Longitud de Onda).
- Estudio, diseño y construcción de una WLAN de bajo costo, tomando en cuenta los aspectos de seguridad de este tipo de redes.

Anexos



Los Anexos a este documento de Tesis se encuentran distribuidos en el CD (CD_TESIS) adjunto, en la carpeta **Tesis**, que contiene las siguientes carpetas:

- \Anexos_Tesis
- \Código_Fuente
- \Diccionario
- \Documento_Tesis
- \Manuales
- \Manuales_Varios
- \Marco_Lógico
- \Software_Aplicativo
- \Software_Varios

A continuación se describe lo que contiene cada una de las carpetas anteriores.

- **\Anexos_Tesis:** Contiene un documento en el que se encuentran los manuales de Instalación, Configuración y Uso de *PHP*, *ASP* y *JSP*, dicho documento se encuentra estructurado de la siguiente manera:

Anexo 1: MANUALES DE INSTALACIÓN, CONFIGURACIÓN Y USO DE *PHP* EN LINUX Y EN WINDOWS.

Anexo 2: MANUALES DE INSTALACIÓN, CONFIGURACIÓN Y USO DE *ASP* EN LINUX Y EN WINDOWS.

Anexo 3: MANUALES DE INSTALACIÓN, CONFIGURACIÓN Y USO DE *JSP* EN LINUX Y EN WINDOWS.

También contiene Tablas complementarias a la investigación, dichas tablas corresponden al **Anexo 4:** TABLAS.

- **\Código_Fuente:** Contiene el código fuente del Sistema Prototipo de Gestión y Control de Proyectos de Inversión. Dicho código se encuentra distribuido en la carpeta de cada sistema operativo (Linux o Windows) en las siguientes subcarpetas:

\aplicacion: Páginas web de la aplicación, hojas de estilo en cascada, scripts de control de cada página, scripts para el cliente, scripts para el servidor.

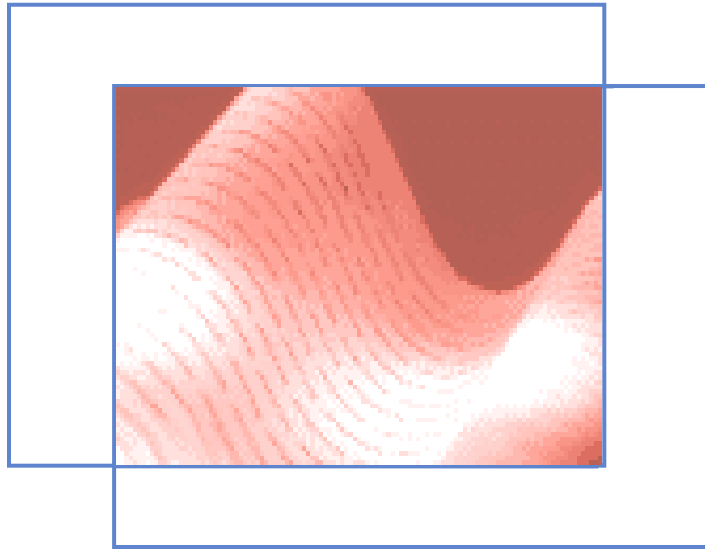
\aplicacion\Menu: Menús para la aplicación.

\aplicacion\Scripts: Scripts para las bases de datos de encuestas y censos, programas y proyectos.

\aplicacion\Imágenes_Sitio: Imágenes para las páginas web.

- **\Diccionario:** Contiene un documento con el Diccionario de Datos de las dos bases de datos utilizadas en el sistema.
- **\Documento_Tesis:** Contiene el Documento de Tesis.
- **\Manuales:** Contiene los Manuales Técnico y de Usuario del Sistema Prototipo de Gestión y Control de Proyectos de Inversión.
- **\Manuales_Varios:** Contiene manuales que se utilizaron en la investigación y documentación de la Tesis, en el desarrollo del aplicativo y en la documentación de los Anexos (manuales de instalación, configuración y uso de *PHP*, *ASP* y *JSP*).
- **\Marco_Lógico:** Contiene un documento con información sobre la estructura del Marco Lógico, dicha información es utilizada por los Programas y Proyectos.
- **\Software_Aplicativo:** Contiene los instaladores del software utilizado en el desarrollo del aplicativo: *PHP*, PostgreSQL, Apache Web Server, Cygwin, CygIPC.
- **\Software_Varios:** Contiene los instaladores del software utilizado en la documentación de los manuales de Instalación, Configuración y Uso de *PHP*, *ASP* y *JSP* (estos manuales se encuentran en la carpeta **\Anexos_Tesis**).

Glosario



TÉRMINOS Y SIGLAS

ActiveX	Lenguaje desarrollado por Microsoft, permite la ejecución de programas en el cliente, es decir, un programa se puede vincular como objeto en una página web para ser ejecutado por el navegador que cargue dicha página.
ADO	(ActiveX Data Object – Objeto de Datos ActiveX). Conjunto de interfaces de acceso a datos basadas en objetos optimizados para las aplicaciones basadas en Internet y centradas en datos. ADO se incluye con Internet Information Server y con Microsoft Visual InterDev.
APACI	(Apache AutoConf Interface – Interfaz Apache al estilo Autoconf). Interfaz de Apache que simplifica considerablemente el procedimiento de instalación y configuración.
API	(Application Programming Interface – Interfaz de Programación de Aplicaciones). Conjunto de rutinas, protocolos y herramientas que definen como se invoca un método o servicio desde un programa.
Aplicaciones Web	Aquel software que interactúa con el usuario a través de Internet, extendiendo la funcionalidad de los sitios web y portales de forma transparente al usuario.
Applets	Son pequeñas aplicaciones escritas en Java, difundidas a través de la red para ejecutarse en un navegador o en el Applet Viewer.
ARPANET	(Advanced Research Projects Agency Network – Red Avanzada de Agencias para Proyectos de Investigación). Es la red pionera en la tecnología de conmutación de paquetes y la base de lo que es ahora Internet.
ASP	(Active Server Page – Página Activa de Servidor). Es una herramienta de generación de contenido dinámico del lado del servidor desarrollada por Microsoft, permite la creación de páginas .asp que son ejecutadas por Internet Information Server o servidores compatibles. Los scripts ASP pueden ser escritos en VBScript/JScript/PerlScript e incrustarse en HTML/XML/WML.
Autenticación	Proceso para determinar si algo o alguien es realmente que o quien declara ser. Se realiza validando un usuario y una clave. Si el usuario y la clave son válidos, se crea una Sesión.
AVI	(Audio Video Interleaved – Audio Vídeo Intercalado). Formato de vídeo estándar de Windows, conforme a las especificaciones Microsoft Windows RIFF (Resource Interchange File Format).
BSD	(Berkeley System Distribution – Distribución del Sistema Berkeley). Licencia académica que no impone condiciones sobre lo que el usuario puede hacer con el software, sólo pide reconocimiento de autores.
Bytecodes	Conjunto de códigos de instrucciones binarias simples.
CAL	Licencia de Acceso de Clientes
CFML	(ColdFusion Markup Language – Lenguaje de Marcas ColdFusion). Es un lenguaje 100% basado en etiquetas, similar al HTML, es muy flexible ya que se integra de manera fácil y transparente con todos los lenguajes y tecnologías web populares, especialmente con HTML. Se incrusta dentro de HTML/XML/WML para crear páginas web dinámicas de lado del servidor. La extensión de este tipo de páginas es .cfm.
CGI	(Common Gateway Interface – Interfaz de Compuerta Común). Es un protocolo creado por el NSCA para establecer un mecanismo estándar de comunicación entre un servidor web y un lenguaje de programación, de tal forma que desde aplicaciones programadas en dicho lenguaje, el servidor web pueda procesar peticiones de usuario.
ChiliSoft ASP	Es el software que permite integrar Apache y ASP en Windows y en Linux. Cuenta con

licencias: gratuita y comercial.

CLI	(Command Line Interface – Interfaz por Línea de Comandos).
Cluster	Es un conjunto de computadoras que se unen para formar un servidor y ser tratadas como una sola.
COM	(Component Object Model – Modelo de Componentes de Objetos). Modelo de componentes de objetos desarrollado por Microsoft.
Cookie	Es una pieza de información que el servidor web envía al cliente para que se guarde en su disco duro. Cuando el cliente vuelve al sitio web reenvía el cookie al servidor y de esa forma éste puede distinguir a los usuarios y hacerles un seguimiento individual.
CORBA	(Common Object Request Broker Architecture – Arquitectura de Corredor Común para la Solicitud de Objetos). Es una arquitectura y especificación estándar para crear, distribuir y administrar objetos de programas distribuidos en una red.
CPU	(Central Process Unit – Unidad Central de Proceso).
CSS	(Cascade Style Sheet – Hoja de Estilo en Cascada). Es un conjunto de especificaciones que definen o redefinen el comportamiento de una etiqueta en un documento HTML.
CSS-P	(CSS Positioning – CSS con Posicionamiento). Variante de Cascade Style Sheet que permite definir posiciones y otros atributos.
CSS	(Client Side Scripts – Scripts del Lado del Cliente). Con este nombre se clasifica a los lenguajes de scripts o módulos que se ejecutan en el lado del cliente, tales como JavaScript, Applets, Flash, etc.
CyglPC	Es una utilidad requerida para correr PostgreSQL en Windows, coopera con Cygwin.
Cygwin	Es una "capa de emulación que expone suficiente funcionalidad del API de Unix". Es una API compuesta de una DLL(cygwin1.dll) y varias herramientas que traen la mayoría de las funciones de Unix para correr programas de Unix en Windows mediante una consola DOS.
DDP	(Datagram Delivery Protocol – Protocolo de Entrega de Datagramas). Protocolo de Apple Talk para el transporte de datos.
DHTML	(Dynamic HTML – HTML Dinámico). Es la versión 4.0 de HTML que incorpora nuevas herramientas como: JavaScript y Hojas de Estilo (CSS) para mejorar las limitaciones de HTML.
DLL	(Data Link Library – Librería de Enlace Dinámico).
DNS	(Domain Name System – Sistema de Nombres de Dominio). Sistema para convertir los nombres de los servidores en las direcciones IP respectivas y viceversa.
DSN	(Data Source Name – Nombre de Origen de Datos). Es el nombre lógico usado por ODBC para referirse a la unidad y otra información necesaria para tener acceso a los datos.
EJB	(Enterprise JavaBeans – Componentes Java Empresariales). Es un modelo de componentes específico de una arquitectura basada en Java para construir componentes transaccionales del lado del servidor. Los EJB encapsulan la lógica del negocio, tal como el acceso a bases de datos, seguridad e integridad de las transacciones.
FastCGI	Es la alternativa mejorada a CGI.
Flash	Es un lenguaje estándar e independiente del navegador para crear animaciones de gran calidad para ser usadas en las páginas web. Para la visualización de animaciones Flash es

- necesario tener instalado el plug-in correspondiente.
- FTP** (File Transfer Protocol – Protocolo de Transferencia de Archivos). Protocolo que permite a un usuario transferir archivos de una ubicación a otra a través de Internet.
- GET** Método de envío del protocolo HTTP. Las variables con sus respectivos valores son pasados al servidor web a través de la URL. Tiene límite en la cantidad de datos a enviar.
- GNU – GPL** (General Public License of GNU – Licencia Pública General de GNU). GNU es un proyecto que promueve el software libre con la condición de mantener disponible el código fuente.
- GOPHER** Es un protocolo que permite a los clientes acceder a un servidor que contiene información almacenada de forma jerárquica similar a un árbol de directorios con subdirectorios y archivos, todos desplegados como hipervínculos.
- HEAD** Método de envío del protocolo HTTP. El servidor responde de forma idéntica a GET, sólo que no se devuelve el cuerpo del documento, sino las cabeceras. Se suele utilizar para comprobar características del documento, como su extensión o la fecha de última actualización.
- HTML** (HyperText Markup Language – Lenguaje de Marcas de HiperTexto). Es el lenguaje estándar para crear documentos estructurados con formato de páginas web. Está basado en marcas o etiquetas que definen la presentación de las páginas.
- HTTP** (HyperText Transfer Protocol – Protocolo de Transferencia de HiperTexto). Protocolo que entrega información a través de WWW. Por medio de una URL permite recuperar hipertexto (texto, gráficos, sonido, etc.) desde un servidor web.
- HTTPD** (HTTP Daemon – Demonio HTTP). Así también se les conoce a los servidores web. Un daemon se refiere a un programa ejecutado en segundo plano y que espera solicitudes.
- Ilustra** Nombre comercial con el que se conoce a la base de datos PostgreSQL.
- IMAP** (Internet Message Access Protocol – Protocolo de Acceso de Mensajes en Internet). Protocolo para manejo de correo electrónico. Es más sofisticado que POP, ya que permite que el usuario vea los encabezados de los e-mails directamente en el servidor remoto (sin "bajarlos") y pueda decidir cuál bajar, borrar, etc. IMAP necesita que el usuario esté permanentemente conectado al servidor cada vez que lee su correo.
- InstantASP** Software para soportar páginas de tipo ASP en Novell Netware.
- IP** (Internet Protocol – Protocolo Internet). Componente del protocolo TCP/IP, encargado de efectuar el traslado de los datos a través de Internet.
- IPSec** (IP Security Protocol – IP segura sobre Internet).
- IPV6** Protocolo IP versión 6.
- IPX** Protocolo que se utiliza para encaminar mensajes de un nodo a otro. Los paquetes IPX incluyen direcciones de redes y pueden enviarse de una red a otra.
- ISAPI** (Internet Server API – API del Servidor Internet Information Server). Interfaz de programación de aplicaciones del Internet Information Server.
- ISP** (Internet Services Provider – Proveedor de Servicios de Internet).
- JavaBeans** Componentes desarrollados en Java que pueden ser insertados en páginas JSP.
- JavaScript** Lenguaje de programación del lado del cliente desarrollado por Netscape. Perteneció a la familia del lenguaje Java, pero a diferencia de éste último se incrusta en HTML para ser

	interpretado por los navegadores.
JDK	(Java Development Kit – Kit de Desarrollo Java). Conjunto de herramientas Java que permiten el desarrollo de: aplicaciones, applets, servlets, páginas JSP, JavaBeans, etc.
JDBC	(Java Data Base Connectivity – Conectividad de Bases de Datos con Java). Es una API Java incluida en el JDK que permite la conexión con bases de datos mediante Java.
JNDI	(Java Nomination Directory Interface – Interfaz Java de Denominación y Directorios). Es una API que permite acceder de forma transparente a los directorios y se utiliza para asignar nombres lógicos a los archivos, directorios y objetos Java.
JScript	Lenguaje similar a JavaScript, desarrollado por Microsoft y utilizado por las páginas ASP.
JSP	(Java Server Pages – Páginas Java de Servidor). Es una herramienta de generación de contenido dinámico del lado del servidor desarrollada por Sun, permite la creación de páginas .jsp para ser ejecutadas por Servidores web compatibles. Los scripts JSP se escriben en Java y se incrustan en HTML/XML/WML o se almacenan en archivos independientes.
JVM	(Java Virtual Machine – Máquina Virtual de Java). Es el componente que se encarga de verificar los bytecodes generados al compilar programas desarrollados en Java. Por lo general la JVM viene con los navegadores.
J2EE	(Java 2 Enterprise Edition – Java 2 Edición Empresarial). Kit de desarrollo de Java edición empresarial. Cuenta con herramientas como JDK, EJB, etc.
JRE	(Java Runtime Environment – Ambiente de Ejecución Java). El JRE es similar al intérprete de Java, es utilizado por usuarios que no requieren usar todo el Kit JDK.
LDAP	(Lightweight Directory Access Protocol – Protocolo Ligero de Acceso a Directorios). Protocolo que permite ubicar organizaciones, individuos y otros recursos tales como archivos y dispositivos en una red, ya sea en Internet o en una Intranet.
LiveConnect	Es una tecnología que permite a JavaScript, Java y a los plug-in comunicarse unos con otros.
LiveWire	Permite a JavaScript del lado del servidor comunicarse a bases de datos.
Lógica del negocio	Conjunto de reglas, validaciones y restricciones que maneja una aplicación web para evitar que se lleven a cabo acciones no válidas.
MILnet	Red militar que fue parte de ARPANET.
MIME	(Multipurpose Internet Mail Extensions – Extensiones Multipropósito para el Correo de Internet). Es un estándar oficial de Internet que especifica como se deben formatear los mensajes para que puedan ser intercambiados entre diferentes sistemas de correo electrónico. MIME es un formato muy flexible, que permite incluir cualquier tipo de dato en el correo.
MMC	(Microsoft Management Console – Consola de Administración Microsoft). La Consola de Administración de Microsoft es donde se puede realizar algunas tareas de administración del Sistema Operativo y de otros programas como el Internet Information Server.
mod_perl	Es una mejora del lenguaje PERL, funciona como módulo de Apache y elimina los problemas de lentitud y creación de un nuevo proceso por cada petición.
MVCC	(Multi-Version Concurrency Control – Control de Conurrencia Multi-Versión).
NIC	(Net Information Center – Centro de Información de Red). Es el organismo encargado de asignar las direcciones IP en Internet.

NNTP	(Network News Transport Protocol – Protocolo de Transporte de Noticias en Red). Los servidores de noticias hacen uso de este protocolo.
NSAPI	(Netscape API – API de Netscape). Interfaz de programación de aplicaciones del Netscape Server.
NCSA	(National Center for Supercomputing Applications – Centro Nacional para Aplicaciones de Supercomputadoras).
NSFnet	Red académica que formó parte de ARPANET y evolucionó hasta llegar a ser Internet.
ODBC	(Open Data Base Connectivity – Conectividad Abierta a Bases de Datos). Interfaz que permite a las aplicaciones acceder a los datos de un Sistema de Base de Datos usando SQL.
OLE-DB	(Object Linking And Embedding Data Base – Objetos Vinculados e Incrustados a Base de Datos). Es una API escrita en C++ que utiliza interfaces de manera eficiente para acceder, no sólo a datos, sino a funciones y objetos, se trate o no de una fuente de datos tradicional.
Open Source	Código Fuente Abierto. Es una modalidad en el desarrollo de software en la cual los autores hacen disponible el código fuente de dicho software a terceras personas.
Páginas estáticas	Páginas web que contienen hipertexto(texto, animaciones, sonido, imágenes), su contenido cambia sólo cuando la página es modificada. Son generadas por medio de editores de páginas como: FrontPage, Dreamweaver, etc.
Páginas dinámicas	Páginas web que son generadas dinámicamente al momento de la petición del cliente, accesan a bases de datos y su contenido se presenta de acuerdo al usuario. Son generadas por medio de herramientas del lado del servidor, tales como: PHP, ASP, JSP, ColdFusion, etc.
PERL	(Practical Extraction and Report Language – Lenguaje Práctico de Extracción y Reportes). Lenguaje de programación (interpretado para Unix, compilado para Windows) utilizado principalmente para programar aplicaciones de tipo CGI.
PerlScript	Lenguaje similar a PERL, usado por ASP en ambientes Unix, disponible con APACHE::ASP.
PHP	(PHP Hypertext Preprocessor – Preprocesador de Hipertexto PHP). Lenguaje de script interpretado e incrustado en HTML/XML/WML, se ejecuta del lado del servidor. Diseñado para facilitar el rápido desarrollo de sitios y aplicaciones web dinámicas. Su funcionamiento es similar a ASP, JSP y CFML. La extensión más común de los scripts en PHP es .php.
PHP/FI	(PHP/Form Interpreter – PHP/Intérprete de Formularios). Versión que precede al PHP actual.
POP	(Postal Office Protocol – Protocolo de Oficina Postal). Protocolo de correo electrónico.
Portal	Es un sitio web con valor añadido, ya que ofrece una variedad de servicios(buscador, correo, información, hospedaje de páginas, procesamiento de datos, actualización dinámica de contenido, etc.) a una gran cantidad de usuarios en tiempo real.
POO	(Programming Oriented Objects – Programación Orientada a Objetos).
POSIX	(Portable Operating System Interface UNIX – Interfaz del Sistema Operativo Portable UNIX). POSIX es el nombre para la familia de normas basadas en UNIX.
POST	Método de envío del protocolo HTTP. Las variables con sus respectivos valores son pasados al servidor web por medio del método Submit. Se usa para enviar al servidor un bloque de datos en el cuerpo de la petición, por ejemplo en un formulario HTML.
POSTQUEL	Lenguaje similar a SQL, utilizado anteriormente por Postgres(predecesora de PostgreSQL).

Python	Lenguaje de scripts interpretado, generalmente usado para programar aplicaciones CGI en ambientes Unix.
RAID	(Redundant Array Disk – Array Redundante de Discos).
RDBMS	(Relational Data Base Management System – Sistema de Gestión de Bases de Datos Relacional).
RDO	(Remote Data Object – Objeto de Datos Remotos). Conjunto de interfaces de acceso a datos similar a ADO.
RPM	(Red Hat Package Manager – Administrador de Paquetes Red Hat). Es el administrador de paquetes del sistema Red Hat Linux, es análogo a Agregar o quitar programas en Windows.
SAF	(Server Applications Functions – Funciones de Aplicaciones de Servidor). Conjunto de funciones utilizado por la interfaz de aplicaciones de NSAPI.
Sesión	Mecanismo con el que se controla y se da seguimiento a la navegación del usuario. Se controla a través de la autenticación en el servidor web.
Servidor de aplicaciones	Es aquel servidor dedicado para el desarrollo y gestión de aplicaciones web que hagan uso transaccional de componentes.
Servlets	Son la respuesta de la tecnología Java a la programación CGI. Son programas Java que se ejecutan en un servidor web y construyen páginas web dinámicas. Son los antecesores de JSP.
ScriptActive	Plug-in, de la empresa NetCompass, que permite interpretar VBScript en el servidor Netscape.
SMB	(Server Message Block – Bloque de Mensajes de Servidor). Es un protocolo de red usado por las redes de Microsoft para acceder a sistemas de archivos de otros sistemas operativos.
SMP	Sistema de Multiprocesamiento Simétrico.
SMTP	(Server Mail Transfer Protocol – Protocolo de Transferencia de Correo). Permite establecer conexiones para la transmisión de correo, facilita la inclusión de correo en formularios de páginas web, dichos formularios son procesados por medio del servidor de SMTP en lugar de ser procesados directamente con un CGI o script del servidor web.
SNMP	(Simple Network Management Protocol – Protocolo Simple de Administración de Red).
SQL	(Structured Query Language – Lenguaje Estructurado de Consulta).
SSI	(Server Side Includes – Inclusiones del Lado del Servidor). Lenguaje sencillo de directivas que se incrustan en HTML y permiten desplegar el contenido de archivos, información dinámica o el resultado de la ejecución de ciertos comandos del sistema. Sólo en ambientes Unix/Linux.
SSJS	(Server Side JavaScript – JavaScript del Lado del Servidor). Tiene características adicionales al JavaScript del lado del cliente, por ejemplo se puede conectar a bases de datos, manejar sesiones, comunicarse con Java y CORBA.
SSL	(Secure Socket Layer – Capa de Socket Seguro). Protocolo posterior al TCP/IP, creado por Netscape, posibilita la transmisión cifrada y segura de información a través de la red.
SSS	(Server Side Scripts – Scripts del Lado del Servidor). Con este nombre se conoce y clasifica a los scripts que son procesados en el lado del servidor, tales como: PHP, ASP, JSP, PERL, etc.
SYSV	Segmento de memoria compartida.
TAGS	Conjunto de etiquetas HTML o similares al formato: <etiqueta> </etiqueta>.

TCL	(Tool Command Language). Es un lenguaje de programación interpretado y multiplataforma.
TCP/IP	(Transmission Control Protocol/IP – Protocolo de Control de Transmisión/Protocolo Internet).
Telnet	Protocolo estándar de Internet para iniciar sesiones en ubicaciones remotas.
TLS	(Transport Layer Security – Capa de Transporte Seguro). Protocolo basado en SSL, es empleado para encriptaciones a nivel más bajo que el de aplicación, es decir, por debajo del protocolo HTTP utilizado para la visualización de las páginas web.
TPC	(Transaction Processing Council – Proceso de Transacción Conciliada).
Unix	Sistema operativo multi-usuario con TCP/IP incorporado.
URL	(Uniform Resource Locator – Localizador Uniforme de Recursos). Es la dirección que se ingresa en el navegador, por medio de la cual se solicita algún recurso de Internet.
Variables de entorno	Son una colección de variables HTTP que contienen información sobre la petición, el cliente, el servidor web, la instalación del servidor web, su existencia depende de la configuración de éste. Otras se derivan de la información del host, del lenguaje del lado del servidor, etc.. Las variables de entorno están disponibles para ser utilizadas por los scripts o programas del lado del servidor y dependen tanto del sistema operativo como del servidor web que las albergan.
VBA	(Visual Basic for Application – Visual Basic para Aplicación). Subconjunto del lenguaje Visual Basic de Microsoft.
VBScript	Lenguaje similar a JavaScript usado sólo por plataformas Microsoft. Se deriva de Visual Basic.
VFAT	(Virtual File Allocation Table – Tabla Virtual de Asignación de Archivos). Soporta nombres de archivos de hasta 255 caracteres de longitud.
VPN	(Network Private Virtual – Red Privada Virtual).
WAIS	(Wide Area Information Servers – Servidores de Información de Área Distribuida). Sistema de recuperación de información distribuida que se utiliza para recuperar documentos en función de claves suministradas.
WAL	(Write Ahead Logging). Mecanismo que permite restaurar una base de datos por medio de un registro de transacciones.
WAP	(Wireless Application Protocol – Protocolo de Aplicaciones Inalámbricas).
WebDAV	(Web-based Distributed Authoring and Versioning). Es un estándar para autorizar a múltiples usuarios construir documentos y publicarlos en una Intranet o en Internet.
WinCGI	Son los CGI de Windows, los datos de respuesta se envían en forma de archivo.
WML	(Wireless Markup Language – Lenguaje de Marcas Inalámbrico). Lenguaje similar a HTML, utilizado para generar páginas web para dispositivos inalámbricos.
WWW	(World Wide Web – Gran Telaraña Mundial). También conocida como Web. Es el universo de información accesible por Internet, funciona como un sistema basado en la técnica del hipertexto creada por el científico Tim Berners Lee en 1989, para el laboratorio CERN.
W3C	(WWW Consortium – Consorcio WWW). Es el organismo encargado de regular todos los estándares que hacen posible el funcionamiento de WWW.
XML	(eXtended Markup Language – Lenguaje de Marcas Extendido).

Bibliografía



LIBROS

- [LIB-01] PEREZ-SERRANO, A. Perspectiva general de la protección de los derechos de autor en los Estados Unidos y México; Marzo, 1999.
- [LIB-02] MANSFIELD, R. Manual Fundamental de ActiveX; Anaya Multimedia, 1998.
- [LIB-03] NAUGHTON, P. Manual de Java; McGraw Hill, 1996.
- [LIB-04] KABIR, M. La Biblia del servidor Apache; Anaya Multimedia D.L., 1998.
- [LIB-05] BRUCE, M. PostgreSQL Introduction and Concepts; ADDISON-WESLEY, Segunda Edición, 2001.
- [LIB-06] TORRES, V. Sistema de desarrollo local SISDEL; Abya-Yala, Primera Edición.
- [LIB-07] BID. Evaluación: Una herramienta de gestión para mejorar el desempeño de los proyectos; Marzo, 1997.

INTERNET

- [WWW-01] <http://www.monografias.com/trabajos5/windows/windows.shtml>
- [WWW-02] <http://pubtool.mslicense.com/preview/latam/503254.asp>
- [WWW-03] <http://www.microsoft.com/spain/servidores/windows2000/default.asp>
- [WWW-04] <http://www.lared.com.ve/archivo/tenden43.html>
- [WWW-05] <http://cybercursos.net>
- [WWW-06] <http://cluv.univalle.edu.co/caract.html>
- [WWW-07] <http://ftp.gul.uc3m.es/gul/raxi/curso/node15.html>
- [WWW-08] http://www.geocities.com/sl_edu_colombia/soluciones/pablo/licenc1.htm
- [WWW-09] <http://www.msemealca.com/POR/April-04/MicrosoftProductUseRights.pdf>
- [WWW-10] <http://securityportal.com/cover/coverstory20000117.html>
- [WWW-11] <http://www.fisica.uson.mx/carlos/Unix-NT/unix-nt-notes.html>
- [WWW-12] http://www.ei.uvigo.es/Departamento_Sistemas/TecWeb.pdf
- [WWW-13] http://www.expresateweb.com.sv/cursos/php4/curso_php_cap_01.pdf
- [WWW-14] http://www.ei.uvigo.es/Departamento_Sistemas/TecWeb.pdf
- [WWW-15] <http://postgrado.ei.uvigo.es/Titulacion.pdf>
- [WWW-16] <http://postgrado.ei.uvigo.es/Titulacion.pdf>

- [WWW-17] <http://www.ayudaasp.com/index.asp?page=asp01>
- [WWW-18] <http://www.desarrolloweb.com/manuales/15/Qué es DHTML.htm>
- [WWW-19] <http://www.elsabio.com/secc/Qué es Flash.htm>
- [WWW-20] <http://www.unav.es/javascript/leccion1.html>
- [WWW-21] <http://www.ayudaasp.com/index.asp?page=asp02>
- [WWW-22] <http://www.cgi.cs.cmu.edu/cgi-bin/perl-man/CGI.pdf>
- [WWW-23] <http://mailweb.udlap.mx/~is101711/tesis/tesis.html>
- [WWW-24] http://www.lcc.uma.es/~eat/services/apl_intra/apl_intra.html#2_2_2
http://www.lcc.uma.es/~eat/services/apl_intra/apl_intra.html#2_2_3
- [WWW-25] <http://www.perldoc.com/perl5.6/pod/perl.html>
- [WWW-26] <http://www.perldoc.com/perl5.6/pod/perl01.html>
- [WWW-27] <http://www.iec.csic.es/criptonomicon/cgi/ssi.html>
- [WWW-28] http://www.php.net/documentacion/es/php_manual_es.pdf
- [WWW-29] [http://www.maestrosdelweb.com/Introduccion al ASP \(Active Server Pages\) de Microsoft - Maestrosdelweb_com.htm](http://www.maestrosdelweb.com/Introduccion al ASP (Active Server Pages) de Microsoft - Maestrosdelweb_com.htm)
- [WWW-30] <http://www.sun.com>
- [WWW-31] http://www.programacion.com/java/tutorial.servlets.html#3_ejemplos
<http://www.javahispano.org/tutoriales/jsp/parte4.htm>
<http://java.sun.com/products/servlet/index.html>
- [WWW-32] <http://www.visualpr.net/Pagina de Productos.htm>
- [WWW-33] <http://www.talltech.com/student/imos99/studentweb/Heather/RESEARCH/process.htm>
- [WWW-34] <http://www.ei.uvigo.es/Http/Tema1/HTTP.pdf>
- [WWW-35] <http://www.serverwatch.com>
- [WWW-36] http://www.lcc.uma.es/~eat/services/apl_intra/apl_intra.html#2_4_2
- [WWW-37] <http://www.tejedoresdelweb.com/307/article-5541.html>
- [WWW-38] <http://aubit4gl.sourceforge.net/hy4gl/doc/SPA/01-Introduccion.html>
- [WWW-39] <http://www.sobl.org/traduccion/practical-postgres/node19.html>
- [WWW-40] <http://www.nextec.com.ar/postgres/principal.html>
- [WWW-41] <http://aubit4gl.sourceforge.net/hy4gl/doc/SPA/01-Introduccion.html>
- [WWW-42] <http://lucas.hispalinux.es/Tutoriales/NOTAS-CURSO-BBDD/notas-curso-BD/node198.html>
- [WWW-43] <http://www.postgres.org.mx/pgsql70.html>
- [WWW-44] <http://www.postgresql.org/docs/admin>
- [WWW-45] <http://www.mmlabx.ua.es/mysql-postgres.html>
- [WWW-46] <http://skyscraper.fortunecity.com/rofl/434/articulos/odbc/odbc.htm>
- [WWW-47] <http://es.tldp.org/Articulos-periodisticos/jantonio/odbc/odbc2.html>
- [WWW-48] <http://www.udechile.ch/web-app.ppt>

-
- [WWW-49] <http://www.tic.udc.es/~fbellas/teaching/is/#Ejemplos>
- [WWW-50] <http://www.htmlweb.net/HTML Web.html>
- [WWW-51] <http://www.owasp.org/Aplicaciones Web.pdf>
- [WWW-52] <http://postgrado.ei.uvigo.es/Aplicaciones inteligentes en internet.htm>
- [WWW-53] <http://www.monografias.com/Tecnologias web.pdf>
- [WWW-54] <http://www.uvigo.es/documentos/titulacion.pdf>
- [WWW-55] http://latined.com/web-ct/courses/Proyecto-Disenio5_B/DisenioS1/AsesoríaD.htm

REVISTAS

- [REV-01] Cekit S.A. "Qué es Internet";
Compañía Editorial Electrónica, Colombia, 1997.
- [REV-02] Cekit S.A. "Internet Práctico";
Compañía Editorial Electrónica, Colombia, 1997.

OTROS

- <http://www.infosecuritymag.com/>
- <http://aubit4gl.sourceforge.net/hy4gl/doc/SPA/01-Introduccion.html>
- <http://lucas.linux.org.mx/Postgresql-es/web/navegable/tutorial/arch.htm>
- <http://www.mmlabx.ua.es/mysql-postgres.html>

Anónimo. Linux Máxima Seguridad;
Pearson Educación S.A., Madrid, 2000, Pág. 808.

Jackett J, Burnett S. Linux Edición Especial;
Pentice Hall, Madrid, 2000, 4ta Edición, Pág. 110.

Índice

CAPÍTULO 1: WINDOWS Y LINUX COMO PLATAFORMAS DE INTERNET	1
INTRODUCCIÓN.....	2
1.1 WINDOWS 2000 SERVER COMO SERVIDOR DE INTERNET	3
1.1.1 REQUERIMIENTOS DEL SISTEMA.....	4
1.1.2 CARACTERÍSTICAS.....	4
1.1.3 ARQUITECTURA.....	7
1.2 LINUX COMO SERVIDOR DE INTERNET	9
1.2.1 REQUERIMIENTOS DEL SISTEMA.....	9
1.2.2 CARACTERÍSTICAS.....	10
1.2.3 ARQUITECTURA.....	12
1.3 WINDOWS Y LINUX COMO ALTERNATIVAS PROPIETARIA Y DE CÓDIGO ABIERTO	14
1.3.1 WINDOWS COMO ALTERNATIVA PROPIETARIA	14
SOFTWARE PROPIETARIO	14
DERECHOS DE USO DEL SERVIDOR.....	15
Instalación: Software de Servidor	15
Software de Cliente	15
Límite de procesadores	15
Necesidad de Licencias CAL para Windows 2000 Server.....	15
Modo de Implantación	15
Servicios de Terminal.....	15
Servicios de Windows 2000 Server	16
Servicios de Terminal.....	16
Modo de Implantación	16
1.3.2 LINUX COMO ALTERNATIVA DE CÓDIGO ABIERTO	17
EL CÓDIGO ABIERTO.....	17
CONDICIONES DE DISTRIBUCIÓN	17
LO QUE OFRECE EL CÓDIGO ABIERTO	18
1.4 COMPARACIÓN DE WINDOWS Y LINUX.....	20
LINUX(RED HAT 7.2) vs. WINDOWS 2000 SERVER	20
CAPÍTULO 2: HERRAMIENTAS DE GENERACIÓN DE CONTENIDO DINÁMICO	23
INTRODUCCIÓN.....	24
2.1 INTERNET COMO MEDIO DE COMUNICACIÓN E INTERCAMBIO DE INFORMACIÓN.....	25
LAS TECNOLOGÍAS WEB	26
2.2 EVOLUCIÓN DEL CONTENIDO ESTÁTICO A DINÁMICO.....	27
EVOLUCIÓN DEL CONTENIDO ESTÁTICO: DE HTML A DHTML.....	28
HTML.....	28
DHTML.....	30
2.2.1 INTERACTIVIDAD Y GENERACIÓN DE CONTENIDO DINÁMICO EN INTERNET	31
2.2.1.1 GENERACIÓN DE PÁGINAS WEB DINÁMICAS O CONTENIDO DINÁMICO.....	31
DEFINICIÓN DE PÁGINAS WEB DINÁMICAS	31
2.2.1.2 APLICACIONES WEB	33
2.2.1.3 ÁREAS DE APLICACIÓN DEL CONTENIDO DINÁMICO	34
2.2.2 HERRAMIENTAS PARA LA GENERACIÓN DE CONTENIDO DINÁMICO	35
2.2.2.1 HERRAMIENTAS DINÁMICAS DEL LADO DEL CLIENTE	36
DHTML.....	38
CONTROLES ACTIVEX	41
APPLETS JAVA.....	44
FLASH	48
LENGUAJES DE SCRIPTS O GUIONES:.....	50
JAVASCRIPT.....	50
VBSCRIPT	54
2.2.2.2 HERRAMIENTAS DINÁMICAS DEL LADO DEL SERVIDOR.....	56
PROTOCOLO ESTÁNDAR CGI.....	59

FastCGI.....	61
APIs PROVISTAS POR EL SERVIDOR WEB:.....	62
ISAPI	62
NSAPI.....	63
MÓDULOS DEL SERVIDOR WEB PARA INTERPRETAR SCRIPTS:.....	64
PERL.....	65
SSI	67
PHP.....	69
ASP.....	73
JSP Y JAVA SERVLETS	76
COLDFUSION (CFML)	82
2.2.2.2.1 COMPARACIÓN DE LAS HERRAMIENTAS: PHP, ASP Y JSP	85
CAPÍTULO 3: SERVIDORES WEB.....	106
INTRODUCCIÓN.....	107
3.1 ARQUITECTURA CLIENTE/SERVIDOR EN INTERNET	108
3.2 CONCEPTO DE SERVIDOR WEB.....	110
EVOLUCIÓN DE LOS SERVIDORES WEB	110
FUNCIONAMIENTO DE UN SERVIDOR WEB	111
3.3 CARACTERÍSTICAS DE LOS SERVIDORES WEB	113
3.4 IMPORTANCIA COMO COMPONENTE PRINCIPAL DE SITIOS Y APLICACIONES WEB	114
3.5 EVALUACIÓN DE LOS SERVIDORES WEB GRATUITOS Y COMERCIALES.....	115
3.5.1 APACHE WEB SERVER.....	117
3.5.2 INTERNET INFORMATION SERVER.....	121
3.5.3 TOMCAT WEB SERVER.....	124
3.6 COMPARACIÓN DE LOS SERVIDORES WEB ESTUDIADOS.....	127
CAPÍTULO 4: MOTOR DE BASE DE DATOS.....	129
INTRODUCCIÓN.....	130
4.1 POSTGRESQL	131
4.1.1 CARACTERÍSTICAS DE POSTGRESQL	131
4.1.2 FUNCIONAMIENTO DE POSTGRESQL.....	134
PSQL.....	136
PGACCESS	136
4.1.3 VENTAJAS DE POSTGRESQL.....	138
4.1.4 DESVENTAJAS DE POSTGRESQL.....	138
4.1.5 CONEXIÓN CON PHP	139
4.1.6 GESTIÓN DE UNA BASE DE DATOS POSTGRESQL.....	140
4.2 INTRODUCCIÓN A ODBC.....	142
4.2.1 CARACTERÍSTICAS DE ODBC.....	142
4.2.2 FUNCIONAMIENTO DE ODBC.....	143
4.2.3 VENTAJAS DE ODBC	144
4.2.4 DESVENTAJAS DE ODBC	144
4.2.5 CONEXIÓN CON ASP.....	145
4.3 INTRODUCCIÓN A JDBC.....	148
4.3.1 CARACTERÍSTICAS DE JDBC	148
4.3.2 FUNCIONAMIENTO DE JDBC	149
TIPOS DE CONTROLADORES JDBC	150
4.3.3 VENTAJAS DE JDBC.....	152
4.3.4 DESVENTAJAS DE JDBC.....	153
4.3.5 CONEXIÓN CON JSP.....	153
CAPÍTULO 5: SOLUCIONES WEB.....	155
INTRODUCCIÓN.....	156
5.1 APLICACIONES WEB	157
5.1.1 CARACTERÍSTICAS DE LAS APLICACIONES WEB	158
5.1.2 FUNCIONAMIENTO DE UNA APLICACIÓN WEB.....	159
5.1.3 VENTAJAS DE LAS APLICACIONES WEB.....	161

5.1.4	DESVENTAJAS DE LAS APLICACIONES WEB.....	162
5.2	APLICACIONES WEB CON HERRAMIENTAS DE CÓDIGO ABIERTO Y PROPIETARIO.....	163
5.2.1	HERRAMIENTAS DE CÓDIGO ABIERTO.....	163
5.2.2	HERRAMIENTAS DE CÓDIGO PROPIETARIO.....	166
CAPÍTULO 6: DESARROLLO DE LA APLICACIÓN.....		168
INTRODUCCIÓN.....		169
6.1	INVESTIGACIÓN PRELIMINAR.....	170
6.1.1	PROGRAMAS.....	170
6.1.2	PROYECTOS.....	170
	CICLO DEL PROYECTO.....	170
	ENFOQUE DEL MARCO LÓGICO.....	172
6.1.3	LA GESTIÓN DE PROYECTOS.....	173
	INSTRUMENTOS PARA LA GESTIÓN DEL PROYECTO.....	173
6.1.4	SEGUIMIENTO Y MONITOREO DE PROYECTOS.....	173
6.1.5	SISTEMA DE GESTIÓN Y CONTROL DE PROYECTOS DE INVERSIÓN.....	174
6.2	ANÁLISIS FODA DE APLICACIONES DE GESTIÓN Y CONTROL DE PROYECTOS DE INVERSIÓN.....	174
6.3	ESTUDIO DE FACTIBILIDAD.....	177
	FACTIBILIDAD TÉCNICA.....	177
	FACTIBILIDAD OPERATIVA.....	177
	FACTIBILIDAD ECONÓMICA.....	177
6.4	DISEÑO DE LA APLICACIÓN.....	178
6.4.1	DISEÑO DE LA BASE DE DATOS.....	178
6.4.2	DISEÑO DE PÁGINAS WEB DINÁMICAS.....	204
6.5	DESARROLLO DE LA APLICACIÓN.....	205
6.6	IMPLEMENTACIÓN DEL PROTOTIPO DE LA APLICACIÓN.....	205
6.7	DOCUMENTACIÓN.....	207
CAPÍTULO 7: CONCLUSIONES Y RECOMENDACIONES.....		208
7.1	VERIFICACIÓN DE HIPÓTESIS.....	209
7.2	CONCLUSIONES.....	209
	RESPECTO DEL ESTUDIO.....	209
	RESPECTO DEL APLICATIVO.....	211
7.3	RECOMENDACIONES.....	212
	RESPECTO DEL ESTUDIO.....	212
	RESPECTO DEL APLICATIVO.....	212
ANEXOS.....		214
GLOSARIO.....		217
	TÉRMINOS Y SIGLAS.....	218
BIBLIOGRAFÍA.....		225
	LIBROS.....	226
	INTERNET.....	226
	REVISTAS.....	228
	OTROS.....	228