Seminario de Computador Universitario

CV-DGR
Carga y Búsqueda de Currículos

Alumna: Cristina Valeria Badino
Directores: MG. Daniel Arias Figueroa, Sr. Sergio Ponce
Seminario de Computación – Lic. en Análisis de Sistemas
UNSa - **Fac. de Ciencias Exactas**

INDICE

Agradecimientos

Es difícil diferenciar, entre las intenciones de estas líneas, la gratitud de la dedicación.

Por un lado quiero agradecer a todas las personas que, intencionadamente o no, sabiendo o no que colaboraban con mi trabajo, tuvieron la generosidad de guiarme para completar este gran pendiente que era escribir el trabajo final de CU. Rodrigo, Alejandra, Martín, Sandra y Pablo W.; mis compañeros Laura, Ricardo y Pablo.

Por otro lado a los profesores, no por los conocimientos impartidos porque ese es su compromiso como docentes, sino por la calidad humana y ese mundo tan especial de intercambio y generosidad del cual nosotros, los estudiantes, formamos parte.

Y por último quiero dedicar, no porque no esté eternamente agradecida sino porque creo que la compañía y el amor no se retribuyen con agradecimiento, sino que no hay tributo mejor que la satisfacción de los logros compartidos, a los tres cuartos de mi alma que residen fuera de mí, Val, Mamá, Vane y a Facu, aunque las cuentas no den, mi otra mitad.

CAPÍTULO I
INTRODUCCIÓN

Motivación

Este primer apartado temático es para exponer las circunstancias que propiciaron la elección del tema objeto de este trabajo de investigación y comentar el móvil para tal inclinación. Éste podría resumirse en “obtener el título de Computador Universitario” si la intención fuese ahorrar palabras a costa del ridículo, pero la referencia buscada tiene que ver más con una cuestión subjetiva que práctica.

La realidad es que lo que me llevó a proyectar el desarrollo de una página de carga de CVs de postulantes para las vacantes producidas en la Dirección General de Rentas del Gobierno de la Provincia, fue el espacio que me brindó la Administración de RRHH de esta dirección para proponer una solución a la falta de una herramienta de asistencia en la organización y administración de los currículos que recibía tanto de postulantes espontáneos como de los convocados para la cobertura de puestos específicos.

El área de RRHH no contaba con ningún software de asistencia para incorporar o almacenar los CV de los postulantes en sus archivos.

Desde mi puesto de pasante de la DGR mi responsabilidad, supuestamente, sólo se limita a “colaborar” con las tareas que definen mi función dentro del organismo, pero en la realidad el asesoramiento que brindamos a los clientes internos, sobre cuestiones técnicas específicas del área de sistemas, muchas veces se traduce en acciones como la presentación de proyectos que importan una solución global a una problemática puntual o una mejora marginal en la gestión de algún área de la administración central.

Dado que el Portal Web de la DGR es un servicio tercerizado, la administración del espacio y diseño del mismo está limitado al Alcance presentado por la empresa desarrolladora del proyecto, el cual excluye un módulo del tipo propuesto para este seminario. Además la DGR no cuenta con un espacio público de Internet, por lo que la posibilidad de carga de CVs debía atender tanto a las necesidades de los postulantes ajenos a la DGR de cargar su CV a través de Internet como a las de los empleados de RRHH de la Dirección de cargar los que les llegan por mostrador, en un ambiente de intranet.

Antecedentes

La DGR cuenta aún hoy con un sistema desarrollado en el año 2001 por la entonces Jefa de Subprograma Informática, CU Margarita Baucis. Este sistema representaba una solución, en el contexto en el que había sido concebido, de alcance limitado por las herramientas de desarrollo de que se disponía y la informalidad en la definición de procesos específicos del área de RRHH.

Por ese tiempo, los empleados marcaban tarjeta en relojes mecánicos; la organización consideraba el proceso de los datos de pasantes, que representaban aproximadamente el 40% de su volumen de empleados, como un caso especial carente de respaldo normativo alguno que estandarizase la gestión de los mismos; y finalmente la liquidación de los sueldos se realizaba en una dependencia de gobierno externa a la DGR.

Así las cosas, el sistema DGR Personal, sólo ofrecía la posibilidad de cargar manualmente la información contenida en las tarjetas de registro de los empleados, las novedades que representaban las solicitudes de licencia y demás, y la confección de reportes y notificaciones

de generación dinámica, que no quedaban almacenados en ningún otro soporte que no fuese papel. Ese único sistema con que contaba la entonces “Supervisión de RRHH” no integraba ninguna otra funcionalidad y mucho menos la opción de registrar datos de los postulantes a algún puesto de la Dirección.

Objetivos

El objetivo de este seminario es desarrollar un módulo de carga y administración de currículos, para publicarlo en la Web.

Particularmente esta aplicación debe cumplir las siguientes funciones:
1. Registrar los datos del CV de los postulantes a un puesto en la DGR y almacenarlo en una BD.
2. Procesar consultas parametrizadas según los requisitos fijados por la organización para el puesto en cuestión.
3. Emitir reportes en base a las consultas definidas por los usuarios autorizados.

Amén de los objetivos particulares del sistema, con el desarrollo del mismo se desea lograr:
- Una investigación sobre el diseño de aplicaciones Web y las herramientas más conocidas para implementar las mismas.
- Adquirir la habilidad de programar en PHP y leer código PHP.
- Establecer una relación de cooperación con otros actores en Internet en el área en cuestión, utilizando el inglés como lenguaje de interacción.
- Adquirir conocimiento preciso y la habilidad necesaria para diseñar y analizar estructuras creadas a partir de Sistemas de Administración de BD Relacionales.
- Instrumentar un servicio más, en el plan de “mejora continua” que lleva a cabo la DGR, incluido en el nuevo Portal Informático Tributario de la DGR.

Alcance

El sistema abarca la carga de datos propios de un Currículum Vitae vía Internet e intranet y la consulta de los mismos para la Supervisión de RRHH de la DGR, también vía intranet.

Esta aplicación es parte integrante del SIARH, Sistema Integral de Administración de Recursos Humanos, cuyo proyecto fue creado para ser implementado en la DGR y el cual se encuentra funcionando en ambiente de producción.

Posterior a la presentación del proyecto del SIARH, fue planteada por la supervisión de RRHH la necesidad de integrar esta aplicación al sistema para ofrecer a los empleados de la dirección la misma opción de carga de CV accesible desde la Web. Para ello el módulo de carga en particular, fue rediseñado de modo que funcionase tanto dentro del SIARH, incorporando un control de acceso por sesiones al mismo, como independientemente vía Internet.

El módulo de administración de los currículos ingresados, sería únicamente accesible para los usuarios autorizados por la Supervisión. Éste incluiría también la impresión de informes basados en consultas parametrizadas a la BD en función de criterios de selección simples definidos por la supervisión. Cabe destacar que este módulo está pensado para asistir al Supervisor de Recursos Humanos en la preselección de postulantes y no para implementar la administración remota de la BD. Este seminario no incorpora tal funcionalidad.

CAPÍTULO II
LAS HERRAMIENTAS

Elección de las Herramientas

Para este tipo de aplicaciones las soluciones más difundidas y probadas se conocen como soluciones LAMP o WAMP, dependiendo de si se implementan sobre sistemas GNU/Linux o Windows. Las siglas refieren a cada uno de los componentes necesarios para implementar una aplicación típicamente Web:
L: Linux o **W**: Windows según corresponda.
A: Apache como servidor Web.
M: MySQL como Sistema Gestor de Bases de Datos Relacionales (DBMS en inglés).
P: PHP como lenguaje de desarrollo.

A continuación de estas líneas expondré los motivos técnicos, por diferenciarlos de alguna manera de premisas relacionadas con las políticas administrativas de las organizaciones y el estado en particular, que me condujeron a una solución WAMP. En estos fundamentos no se encuentra el de la elección de MySQL ya que, según se explica en secciones posteriores la implementación de la BD en MS SQL Server se impuso casi como una condición.

A modo de "justificación filosófica" y título introductorio, extiendo una breve reflexión personal:

En un país en vías de desarrollo como Argentina incorporar TICs en la Administración Pública es una de las tareas prioritarias si lo que se busca es hacer el mejor aprovechamiento de recursos posible. Esto permite no sólo optimizar el uso de las nuevas tecnologías disponibles sino organizar de manera eficiente el trabajo de los agentes de las distintas dependencias, contribuyendo a la comunicación interna de las mismas y a la minimización de los tiempos de respuesta internos.

Las organizaciones estatales, es sabido, se desenvuelven en la escasez de recursos o, peor aún, con recursos distribuidos sin planificación y pobremente organizados. La situación se agrava si pensamos en los recursos tecnológicos informáticos. En este sentido ni siquiera es valorada la ventaja que significa, en particular, utilizar software debidamente licenciado o libre.

La elección de software libre por sobre la oferta "propietaria", no es una cuestión estrictamente ligada al criterio personal de los responsables informáticos de las dependencias estatales. A nivel nacional existen oficinas que funcionan como órganos rectores en materia de empleo de tecnologías informáticas de la Administración Pública Nacional (APN), este es el caso de la ONTI (Oficina Nacional de Tecnologías de Información) dependiente de la Subsecretaría de la Gestión Pública de la Jefatura de Gabinete de Ministros de la Nación, que promueven el uso de Software Libre en las dependencias estatales de gobierno a través de proyectos tales como ETAP's (Estándares Tecnológicos para la Administración Pública) y el Foro de RRII (Responsables Informáticos).

De manera que mi elección de Software Libre para el desarrollo del sistema de este seminario no sólo responde a una visión de la conveniencia económica que esto importa, sino también a un espíritu de colaboración con un proyecto común que es el de estandarización del uso de tecnologías de la información en la APN.

Por qué Apache?
Apache es un servidor de red para el protocolo http que se distribuye como free software, por explícito deseo del grupo que lo desarrolla, el Apache Group, conformado por voluntarios de todo el mundo. Este software cuenta con versiones para ser utilizado tanto sobre sistemas operativos GNU/Linux como MS Windows y otros menos difundidos.

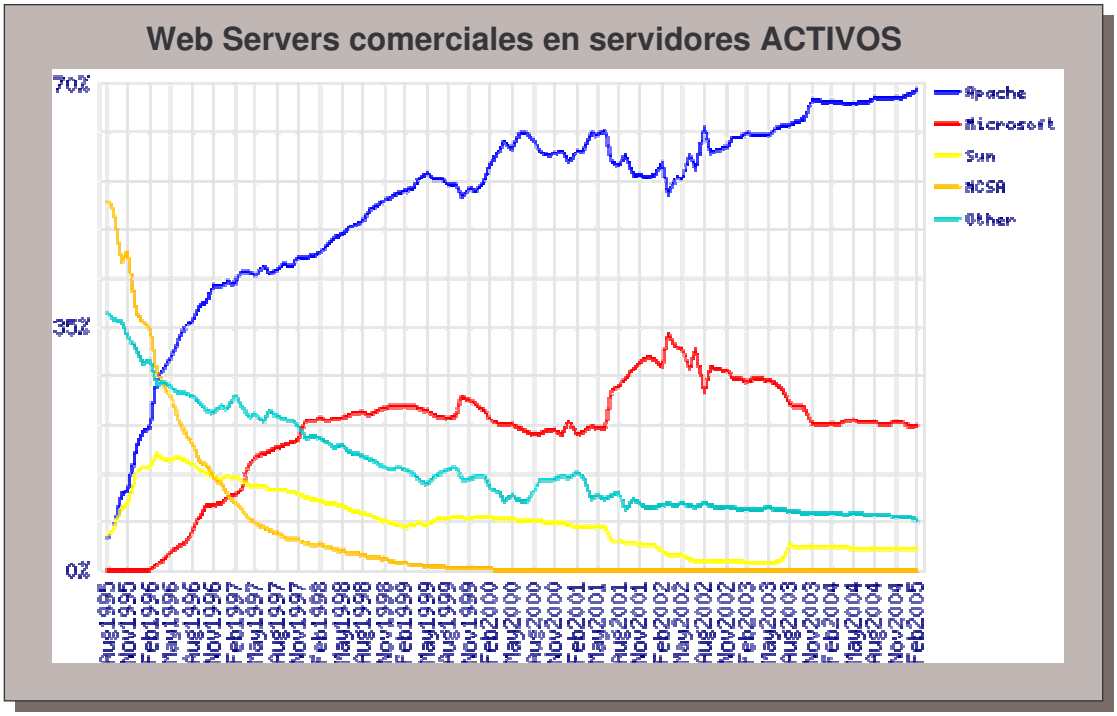


Figura 1 – Comparación del uso de servidores comerciales ACTIVOS

La explicación técnica podría ser una enumeración como:
Es multiplataforma. Existen binarios precompilados para casi todos los sistemas operativos, entre los que se encuentran Win32(9x, NT, XP, etc.), los tipo Unix(Linux, BSD, Solaris, etc.), OS/2, MacOS, etc.

- Es software libre y el código fuente está disponible para cualquiera que desee investigarlo, mejorarlo y/o modificarlo.
- Admite el protocolo HTTP/1.1 y es compatible con HTTP/1.0
- Relativa simplicidad de configuración. La configuración se encuentra en un único archivo de texto y existen herramientas gráficas que permiten configurar el servidor de manera sencilla
- Admite CGI(Common Gateway Interface)
- Soporta servidores virtuales. Permite tener dentro de un mismo servidor direcciones IP y servidores virtuales
- Cuenta con un servidor Proxy .
- Registra toda la actividad producida en el servidor, de forma que es posible monitorearlo en caso de problemas
- Soporta SSL.

Pero entonces no sería sensible la diferencia con otros Web Servers. La realidad, en mi caso, es que al ser tan difundido fue el que primero me incliné por analizar y por otro lado es el Web Server LIBRE que está mejor integrado con las demás herramientas listo para ser utilizado en un entorno Windows. Por lo tanto la elección era clara, dadas las variables condiciones que se presentaban a la hora de presentar el proyecto en cuanto a la plataforma que debería usar.

Por qué PHP?
De entre los lenguajes de desarrollo de aplicaciones Web LIBRES, el de uso más simple en combinación con Apache es PHP. Apache incluye módulos de interpretación de código PHP, Perl, Pitón y varios otros. El avance de PHP sobre el manejo de sesiones y la simplicidad de las funciones que debía implementar el sistema a desarrollar, hicieron inclinar considerablemente la balanza hacia este lenguaje en el momento de la elección.

Además, al usar un lenguaje de scripting se deben tener presentes cuatro grandes características: Velocidad, estabilidad, seguridad y simplicidad.

Velocidad: Se deben considerar la velocidad de ejecución y además el no crear demoras en la máquina. Por esta razón no debe requerir demasiados recursos de sistema. PHP se integra junto a otro software y cuando se configura como módulo de Apache, supera notablemente a otros lenguajes.

Estabilidad: La velocidad es desestimada si el sistema se cae cada cierta cantidad de ejecuciones. Ninguna aplicación es 100% libre de bugs, pero con el respaldo de la comunidad de programadores y usuarios es mucho mas difícil para lo bugs sobrevivir. Con su sofisticado método de manejo de variables y su propio sistema de administración de recursos PHP conforma un sistema robusto y estable.

Seguridad: El sistema debe poseer protecciones contra ataques. PHP provee diferentes niveles de seguridad, estos pueden ser configurados desde el archivo .ini

Conectividad: PHP dispone de una amplia gama de librerías, por lo que es fácilmente extensible. Esto le permite al PHP ser utilizado en muchas áreas diferentes, tales como encriptado, gráficos, XML y otras.

Además como ventajas adicionales de PHP cabe destacar:

- ✓ PHP corre en (casi) cualquier plataforma utilizando el mismo código fuente, pudiendo ser compilado y ejecutado en aproximadamente 25 plataformas, incluyendo diferentes versiones de Unix, Windows (95,98,NT,ME,2000,XP) y Macs. Como en todos los sistemas se utiliza el mismo código base, los scripts pueden ser ejecutados de manera independiente al OS.
- ✓ La sintaxis de PHP es similar a la del C, por esto cualquiera con experiencia en lenguajes del estilo C podrá entender rápidamente PHP. Entre los lenguajes del tipo C incluimos al Java y JavaScript, de hecho mucha de la funcionalidad del PHP se la debe al C en funciones como fread() o srtlen().
- ✓ PHP es completamente expandible. Está compuesto de un sistema principal (escrito por Zend), un conjunto de módulos y una variedad de extensiones de código.
- ✓ Muchas interfaces distintas para cada tipo de servidor. PHP actualmente se puede ejecutar bajo Apache, IIS, AOLServer, Roxen y THTTPD. Otra alternativa es configurarlo como modulo CGI.
- ✓ Puede interactuar con muchos motores de bases de datos tales como MySQL, MS SQL, Oracle, Informix, PostgreSQL, y otros. Siempre se dispone de ODBC en caso de ser necesario.
- ✓ Una gran variedad de módulos cuando un programador PHP necesite una interfase para una librería en particular, fácilmente podrá crear una API para esta. Algunas de las que ya vienen implementadas permiten manejo de gráficos, archivos PDF, Flash, Cybercash, calendarios, XML, IMAP, POP, etc.
- ✓ Rapidez. PHP generalmente es utilizado como módulo de Apache, lo que lo hace extremadamente veloz. Esta completamente escrito en C, así que se ejecuta rápidamente utilizando poca memoria.
- ✓ PHP es Open Source, lo cual significa no se está forzado a pagar actualizaciones anuales para tener una versión que funcione.

No todas son rosas. Aunque muy pocas, las desventajas a considerar son el manejo de errores que no es tan sofisticado como Cold Fusion o ASP y la no existencia de un IDE o un Debugger. Una IDE puede no ser importante para la mayoría de los programadores y un debugger ha sido prometido por Zend Tech para un futuro muy cercano. El gráfico de la Figura 2 muestra la creciente popularidad de PHP como lenguaje de scripting desde casi sus inicios hasta Enero de 2005. Esta tendencia continúa en la actualidad.

Estas consideraciones y la incertidumbre acerca de cuál sería el DBMS a utilizar, dado el condicionamiento que sabía representaría el gestor utilizado en el sistema de control de los relojes digitales de proximidad que reemplazarían a los viejos relojes, dieron lugar a la elección de una herramienta que libera la aplicación para ser utilizada sobre sistemas operativos GNU/Linux o MS Windows y con casi cualquiera de los motores de base de datos conocidos: las librerías ADODB. Aunque en un principio se hubiese definido para este trabajo a MySQL

como el gestor de BD, las variaciones surgidas durante la marcha del proyecto condujeron a la no elección de un solo DBMS en particular.

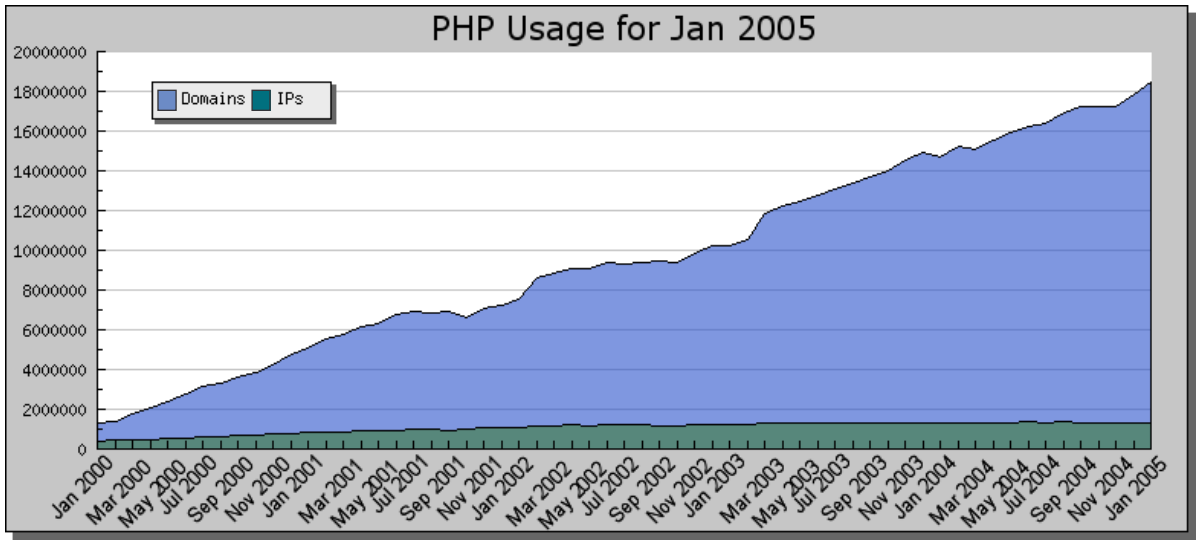


Figura 2 – Uso creciente de PHP

Por qué ADOdb?

En las secciones que continúan el desarrollo de este capítulo se desglosan funcionamiento, características principales y demás aspectos de cada una de las herramientas enunciadas, pero a modo de ilustración sobre la diferencia entre el uso de las librerías ADOdb que implementan una capa de abstracción de base de datos y el uso directo de una API de BD, este gráfico muestra a las claras la ventaja de la elección de ADOdb en aplicaciones Web:

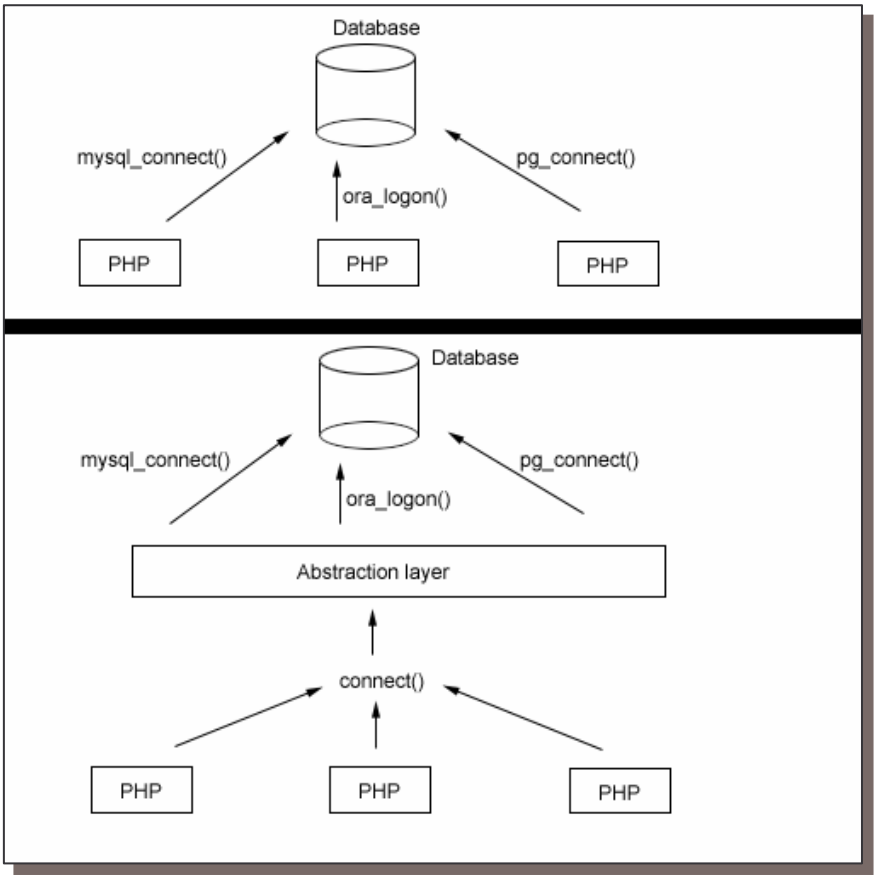


Figura 3 – Capa de Abstracción con ADOdb - Comparación

Típicamente una capa de abstracción de BD funciona como un wrapper, que significa envoltorio en inglés, alrededor del código, exponiendo sólo un conjunto de métodos genéricos

de interacción con el servidor de BD. Internamente estos métodos están mapeados a la API nativa de cada base de datos y la capa de abstracción asegura que al hacer alguna petición a la BD el método utilizado sea el correcto.

En síntesis la capa de abstracción de BD "traduce" una llamada genérica a la base de datos en una llamada nativa de la BD elegida para la implementación del sistema, cualquiera sea esta mientras esté soportada por ADOdb, la librería creada por John Lim. De esta manera es que, gracias a la capa de abstracción de BD, la aplicación queda liberada y es independiente de SGBD que se decida utilizar para su implementación.

Las demás.
El resto de las herramientas no fueron elegidas. Las circunstancias a las que debían adaptarse las propuestas del proyecto dictaban que era más sencillo utilizar el mismo MS SQL Server que ya estaba corriendo que revisar la correspondencia de datos comunes, o replicar la configuración del mismo en otro motor. En este mismo sentido, siendo PHPLib la librería que extiende las funcionalidades de PHP para el manejo de sesiones no había motivos para buscar otras opciones.

Una necesidad no considerada en el anteproyecto de este seminario fue la utilización de algún lenguaje que permitiera agregar valor a la aplicación en cuanto a la interactividad del usuario con la interfaz del mismo. Siguiendo la línea impuesta por las premisas que se imponían JavaScript era la opción casi obligada.

En los párrafos que siguen extiendo una breve reseña descriptiva de las características fundamentales de las herramientas elegidas:

I. Apache,
II. PHP,
III. ADOdb,
IV. MS SQL Server y
V. JavaScript.

Apache2Triad

La instalación de Apache en MS Windows se simplifica notablemente si se cuenta con un paquete que integre en su instalación no sólo el Web Server sino también el intérprete PHP, un servidor FTP, un servidor de mail y demás prestaciones deseables en un servidor que brinde servicios de red dentro de una organización o a través de Internet.

Apache2triad es una distribución de algunos de los servidores e intérpretes *open source* más populares conjugados para desarrollar y publicar contenidos Web, con Windows como plataforma. Este paquete software fue creado por un ingeniero en sistemas rumano llamado Vlad Alexa Mancini cuya intención primaria fue liberarse de reconfigurar cada una de las herramientas que utilizaba en su servidor con cada cambio de sistema operativo. Apache2Triad es distribuido bajo la licencia GNU GPL como la mayoría del software que incluye.

Ya que para plataformas Windows contamos con este paquete, la instalación de las herramientas elegidas para mi seminario, sobre Windows, se resumirá a la referencia de instalación de Apache2triad que extiendo a continuación, por lo que en los apartados referidos a cada herramienta en particular, sólo se describirá su instalación sobre distribuciones GNU/Linux.

La instalación de Apache2triad en Windows es tan simple como ejecutar un único instalador, decidir en qué directorio se desea guardar todos los archivos y carpetas del software incluido y eso es todo. Los archivos de configuración de las herramientas están predefinidos y pueden modificarse accediendo a ellos desde un panel de control que se ejecuta desde el explorador con la dirección:
http://localhost/apache2triadcp
una vez terminada la instalación y corriendo Apache.

Servidor Apache

Introducción.
Apache nació como una sustitución para el servidor de red httpd 1.3 desarrollado por el NCSA (National Center for Supercomputing Applications), incluyendo nuevas características y solucionando los problemas reportados. Sus desarrolladores son un grupo formado por voluntarios conocidos como Apache Group.

"El Proyecto Apache es un esfuerzo comunitario de desarrollo de software, dedicado a implementar un servidor HTTP (Web) de código fuente libremente disponible."

El grupo de desarrollo de Apache considera que instrumentos de este tipo tienen que ser accesibles a todos y que las compañías de software deben buscar réditos en la producción de addons[1], o en personalizaciones de algunas categorías de usuarios. Además, permitiendo acceso completo a los fuentes, es posible que los propios usuarios puedan mejorar el producto final.

Instalación y configuración de Apache.
Hasta el momento las versiones más difundidas de apache son la 1.3.X y la 2.0.X, considerando que si se quería instalar este servidor sobre Windows 98 la versión 1.3.X se volvía algo inestable. Las dos versiones se usan bastante y varían ligeramente la configuración de una a otra. Pero en este apartado nos referiremos a la instalación de la versión 1.3.20 sobre un sistema GNU/Linux y si se desea se puede reemplazar la cadena '1.3.20' por el identificador de la versión que corresponda.

- Instalación de Apache en GNU/Linux (Desde las fuentes):

► Descargue apache_1.3.20.tar.gz desde www.php-es.com en el directorio /usr/local/usr
► En el directorio donde bajamos apache descomprimimos el programa con la siguiente instrucción :

tar –xzf apache_1.3.20.tar.gz

Esto creará el directorio apache_1.3.20 el cual contendrá todos los archivos necesarios para compilar el programa.

► Ingrese al directorio apache_1.3.20 y escriba el siguiente comando
./configure –prefix=/usr/local/apache
► # make (Este comando compila el programa)
► # make install
► Inicie Apache desde la consola como root:
[root@ford root] # /usr/local/apache/bin/apachectl Stara
► Detenga Apache desde la consola usando el siguiente comando:
[root@ford root] # /usr/local/apache/bin/apachectl stop

En el apartado sobre PHP en este mismo capítulo veremos la instalación de Apache seguida de PHP como módulo estático que es la forma más sencilla de instalación en sistemas GNU/Linux si ya contamos con todos los paquetes necesarios.

Una vez instalado el Apache, en la raíz de la instalación, se encontrarán los siguientes directorios:

- bin: ficheros ejecutables del Apache.
- conf: ficheros de configuración del servidor.
- error: ficheros con los mensajes de error del servidor, en varios lenguajes.
- htdocs: directorio raíz por defecto del servidor (Se guardan las páginas Web).
- icons: directorio donde se encuentran los iconos que utiliza el servidor (entre otras cosas

[1] Incrementos de versión, parches con nuevas funcionalidades, etc.

para mostrar estructuras de directorios).
- logs: directorio donde se almacenan los registros de acceso y errores del servidor.
- manual: directorio donde se encuentra el manual del Apache.
- proxy: Directorio con los ficheros de la cache del servidor.

Las configuraciones del servidor residen dentro de dos ficheros, el de configuración principal que se encuentra dentro de la carpeta conf, con el nombre httpd.conf, o dentro de un fichero con el nombre .htaccess que se puede encontrar dentro de cualquier directorio que se encuentre mapeado dentro del servidor. En cada uno de estos archivos se ubican las directivas de configuración. Muchas de ellas se pueden encontrar tanto dentro del fichero de configuración principal como dentro de un fichero .htaccess. Es muy importante tener en cuenta lo siguiente:

Las directivas que se encuentran dentro .htaccess, prevalecen frente a los valores de configuración especificados dentro del fichero httpd.conf.

El archivo httpd.conf
Este archivo es el fichero principal de configuración de Apache. Se ubica dentro del directorio Conf, en la raíz de la instalación de Apache. El mismo está dividido en tres secciones, que son:

1º. Parámetros globales
2º. Directivas de Funcionamiento
3º. Hosts Virtuales

En el httpd.conf se encuentran todos los parámetros de funcionamiento del Apache. Algunos parámetros son generales para la instalación y funcionamiento del Apache. Muchos otros de los parámetros se pueden configurar independientes para un conjunto de directorios y/o ficheros. En estos casos los parámetros se encuentran ubicados dentro de secciones donde se indica el ámbito de aplicación del parámetro.

Las secciones más importantes son:

<Directory> : Los parámetros que se encuentran dentro de esta sección, sólo se aplicarán al directorio especificado y a sus subdirectorios.

<DirectoryMatch>: Igual que Directory, pero acepta en el nombre del directorio expresiones regulares.

<Files>: Los parámetros de configuración proporcionan control de acceso de los ficheros por su nombre.

<FilesMatch>: Igual que Files, pero acepta expresiones regulares en el nombre del fichero.

<Location>: Proporciona un control de acceso de los ficheros por medio de la URL

<LocationMatch>: Igual que Location, pero acepta expresiones regulares en el nombre del fichero.

Algunas veces las directivas de funcionamiento de las secciones anteriores se pueden cruzar en cuyo caso tienen el siguiente orden de preferencia:

1. <Directory> y .htaccess
   (.htaccess prevalece frente a <Directory>)
2. <DirectoryMatch> y <Directory>
3. <Files> y <FilesMatch>
4. <Location> y <LocationMatch>

Es importante saber que el fichero contiene un montón de comentarios para su correcta utilización, las líneas comentados aparecen con el símbolo #.

Arquitectura del servidor

Apache es un servidor de red para el protocolo http el cual está diseñado para poder funcionar como un proceso standalone[2], para lo cual crea subprocesos llamados "children processes" y así puede gestionar las solicitudes. Estos procesos no podrán nunca interferir con el proceso padre, sin embargo puede pasar lo contrario: que ellos envíen una señal de stop a éste, lo que ocasionará que los children se terminen también.

Este software está estructurado en módulos. Cada módulo debe ser configurado mediante la especificación de las directivas que están contenidas dentro del mismo. Los módulos del Apache se pueden clasificar en tres categorías:

- Módulos Base: Módulo con las funciones básicas de Apache.
- Módulos Multiproceso: son los responsables de la unión con los puertos de la máquina, acepando las peticiones y enviando a los hijos a atender a las peticiones.
- Módulos Adicionales: Cualquier otro módulo que le añada una funcionalidad al servidor.

Las funcionalidades elementales conforman el módulo base, mientras que las que administran peticiones se encuentran en el módulo multiproceso. Por cada sistema operativo sobre el que se pueda ejecutar el Apache hay varios módulos multiproceso diseñados. De esta manera se optimiza el rendimiento y la rapidez del código.

El servidor puede sumar funcionalidades por medio de módulos adicionales que se pueden cargar o no según sean las necesidades de cada caso. Para añadir un conjunto de utilidades, simplemente hay que añadir un módulo, de forma que no es necesario volver a instalar el software.

Módulos Base y Módulos Multiproceso

- ✓ core: Funciones básicas del Apache que están siempre disponibles.
- ✓ mpm_common: Colección de directivas que se implementan en más de un módulo multiproceso.
- ✓ beos: Módulo de multiproceso optimizado para BeOS.
- ✓ leader: Variable experimental de MPM.
- ✓ mpm_netware: Módulo de multiproceso que implementa un servidor web optimizado para Novell NetWare.
- ✓ mpmt_os2: MPM híbrido, multiproceso y multihilo para OS/2 .
- ✓ perchild: Módulo multiproceso que permite a los procesos demonio servir las peticiones que se asignan a distintos id de usuario.
- ✓ prefork: Implementa un servidor sin hilos.
- ✓ threadpool: Variante experimental del módulo estándar de MPM .
- ✓ mpm_winnt: Módulo multiproceso optimizado para Windows NT.
- ✓ worker: Módulo multiproceso que implementa un híbrido multihilos y multiprocesos de servidor Web.

Módulos adicionales

- ✓ mod_access: proporciona control de acceso basándose en el nombre del host del cliente, su dirección IP u otras características de la petición del cliente.
- ✓ mod_actions: este módulo se utiliza para ejecutar *Scripts* CGI, basándose en el tipo de medio o el método de petición.
- ✓ mod_alias: proporcionado para mapear diferentes partes del sistema de ficheros del servidor en el árbol de documentos del servidor, y para redirección de URL's.
- ✓ mod_asis: envío de ficheros que tienen sus propias cabeceras http.
- ✓ mod_auth: autentificación de usuario utilizando ficheros de texto.
- ✓ mod_auth_anon: permite a usuarios anónimos acceder a áreas autentificadas.
- ✓ mod_auth_dbm: proporciona autentificación utilizando ficheros DBM.
- ✓ mod_auth_digest: autentificación de usuario utilizando MD5.
- ✓ mod_auth_ldap: permite la utilización un directorio LDAP para almacenar la base de datos

[2] Un proceso que no necesita de el apoyo de otras aplicaciones, no importa dependencias.

de autentificación.
- ✓ mod_autoindex: muestra los contenidos de un directorio automáticamente, parecido al comando ls de Unix.
- ✓ mod_cache: Cache de contenidos indexados por URI's.
- ✓ mod_cern_meta: Semántica de etiquetas meta del CERN.
- ✓ mod_cgi: Ejecución de Scritps CGI.
- ✓ mod_cgid: ejecución de *Scripts* CGI utilizando un demonio CGI externo.
- ✓ mod_charset_lite: para la especificación del juego de caracteres de las traducciones.
- ✓ mod_deflate: comprime el contenido antes de ser enviado al cliente.
- ✓ mod_dir: Proporcionado para redirecciones y para servir los ficheros de listado de directorios.
- ✓ mod_disk_cache: Cache para almacenar contenidos identificados por URI.
- ✓ mod_echo: Un servidor simple de echo para ilustrar los módulos del protocolo.
- ✓ mod_env: modificación del entorno que se envia a los *scripts* CGI y las páginas SSI.
- ✓ mod_expires: Generación de las cabeceras http Expires, de acuerdo de los criterios especificados por el usuario.
- ✓ mod_ext_filter: pasa el cuerpo de la respuesta a través de un programa antes de enviársela al cliente.
- ✓ mod_file_cache: cachea una lista estática de ficheros en memoria.
- ✓ mod_headers: personalización de las peticiones HTTP y las cabeceras de las respuestas.
- ✓ mod_imap: proceso de imágenes en el lado del servidor.
- ✓ mod_include: Documentos HTML generados por el servidor (Server Side Includes).
- ✓ mod_info: proporciona una visión comprensiva de la configuración del servidor.
- ✓ mod_isapi: Extensiones ISAPI en Apache para Windows.
- ✓ mod_ldap: pool de conexiones LDAP y cacheo de resultados para la utilización de otros módulos LDAP.
- ✓ mod_log_config: registro de las peticiones hechas al servidor.
- ✓ mod_logio: registro del número de bytes recibidos y enviados en cada respuesta.
- ✓ mod_mem_cache: Cache de contenidos identificados por URI.
- ✓ mod_mime: asocia las extensiones de peticiones de los ficheros con el comportamiento del fichero (manejadores y filtros) y contenido (tipos mime, idioma, juego de caracteres y codificación).
- ✓ mod_mime_magic: determina el tipo MIME de un fichero mirando unos pocos bytes del contenido.
- ✓ mod_negotiation: se proporciona para la negociación del contenido.
- ✓ mod_proxy: servidor HTTP/1.1 proxy/gateway.
- ✓ mod_proxy_connect: extensión de mod_proxy para la gestión de las peticiones CONNECT.
- ✓ mod_proxy_ftp: soporte FTP para mod_proxy.
- ✓ mod_proxy_http: soporte HTTP para el módulo mod_proxy.
- ✓ mod_rewrite: proporciona un motor de reescritura basado en reglas que rescribe las peticiones de URL's al vuelo.
- ✓ mod_setenvif: permite la configuración de las variables de entorno basándose en las características de la petición.
- ✓ mod_so: carga del código ejecutable y los módulos en al iniciar o reiniciar el servidor.
- ✓ mod_speling: intenta corregir las URL mal puestas por los usuarios, ignorando las mayúsculas y permitiendo hasta una falta.
- ✓ mod_ssl: criptografía avanzada utilizando los protocolos Secure Sockets Layer y Transport Layer Security.
- ✓ mod_status: proporciona información en la actividad y rendimiento del servidor.
- ✓ mod_suexec: permite a los *scripts* CGI ejecutarse con un nombre y grupo específico.
- ✓ mod_unique_id: proporciona variables de entorno y un identificador único para cada petición.
- ✓ mod_userdir: directorios específicos para usuarios.
- ✓ mod_usertrack: registro de actividad de un usuario en el sitio.
- ✓ mod_vhost_alias: Proporcionado para configurar muchos servidores virtuales dinámicamente.

El lenguaje PHP

Introducción.
PHP es el acrónimo de "PHP: Hypertext Preprocessor", es decir, un "preprocesador del hipertexto". Nació en 1994 como proyecto "personal" y la primera versión se utilizó públicamente en 1995 con el nombre "Personal Home Page". PHP fue primero escrito por Rasmus Lerdorf como un simple conjunto de scripts de Perl para guiar a los usuarios en sus páginas. Luego para satisfacer inquietudes del mismo tipo por parte de otra gente lo reescribe, pero esta vez como un lenguaje de script agregándole entre otras características soporte para formularios.

Este lenguaje se desarrolló como proyecto open-source hasta que en 1996, ya se estaba utilizando en 15.000 sitios web. En el momento de la release 3 (a mediados de 1999) el número de servidores que utilizaban PHP se había decuplicado. Al ver como la popularidad del lenguaje aumenta, un grupo de desarrolladores crea para él un API, convirtiéndose así en el PHP3. Fue en ese momento cuando el *parser* de scripts PHP es completamente reescrito (el Zend Engine) dando vida al PHP4 mucho mas rápido, tal y como lo conocemos en la actualidad. De acuerdo a las encuestas de NetCraft, PHP es ahora el módulo más popular para el servidor Apache, creciendo un 4% mensual sobre la totalidad de sitios de Internet.

PHP es un lenguaje de "código abierto" interpretado, de alto nivel, embebido en páginas HTML y ejecutado en el servidor, es decir, un lenguaje de scripting server-side. La principal diferencia entre JavaScript y PHP es que éste último se ejecuta en el servidor en lugar de en el cliente directamente, como el primero. El cliente solamente recibe el resultado de la ejecución del script PHP en el servidor, sin ninguna posibilidad de determinar qué código ha producido el resultado recibido. El servidor web puede ser incluso configurado para que procese todos los archivos HTML con PHP.

Por ejemplo, si una página contiene código similar al siguiente:

```
<html><body>
<? phpinfo(); ?>
</body></html>
```

y se invoca la página desde el browser, no se ve la línea del código entre <? y ?>, las etiquetas del PHP, sino el resultado que devuelve la instrucción comprendida entre ellas. En este caso particular *phpinfo()* devuelve información acerca de PHP, junto a una lista de todas las variables que puede usar. Las etiquetas <? y ?> nos permiten “embeber” el código PHP dentro del HTML entrando y saliendo del modo PHP.

El servidor HTTP debidamente configurado, una vez que se le solicita una página parecida, la reconoce y pone en marcha la parte "dinámica". Al browser llega el resultado de la ejecución de las instrucciones comprendidas entre las etiquetas PHP, y se puede conocer que se trata de código PHP sólo por su extensión, típicamente .php.

En qué puede usarse PHP?
Las posibilidades del lenguaje PHP son muy extensas, incluso es posible crear en PHP todas las aplicaciones que se podrían crear con unos scripts CGI. La diferencia principal entre los dos es que el primero hace mucho más simple la conexión y las preguntas con las bases de datos.

Existen tres campos en los que se usan scripts escritos en PHP.

- **Scripts del lado del servidor.**
  Es el campo más tradicional y el principal foco de trabajo. Se necesita un intérprete PHP (CGI ó módulo), un servidor web y un navegador. Es necesario correr el servidor web con PHP instalado. El resultado del programa PHP se puede obtener a través del navegador,

conectándose con el servidor web.

- **Scripts en la línea de comandos.**
  Un script PHP puede ejecutarse sin ningún servidor web o navegador. Solamente necesita el intérprete PHP para usarlo de esta manera. Estos scripts también pueden ser usados para tareas simples de procesamiento de texto.
- **Escribir aplicaciones de interfaz gráfica.**
  Probablemente PHP no sea el lenguaje más apropiado para escribir aplicaciones gráficas pero con él se pueden usar algunas características avanzadas en programas clientes echando mano de PHP-GTK (una extensión de PHP, no disponible en la distribución principal) para escribir dichos programas.

PHP puede utilizarse en cualquiera de los principales sistemas operativos del mercado:

1. Linux,
2. muchas variantes Unix (incluyendo HP-UX, Solaris y OpenBSD),
3. Microsoft Windows,
4. Mac OS X,
5. RISC OS y alguno más.

PHP soporta la mayoría de servidores web de hoy en día, incluyendo Apache, Microsoft Internet Information Server, Personal Web Server, Netscape e iPlanet, Oreilly Website Pro server, Caudium, Xitami, OmniHTTPd y muchos otros. PHP tiene módulos disponibles para la mayoría de los servidores, para aquellos otros que soporten el estándar CGI, PHP puede usarse como procesador CGI.

Su estructuración lógica permite tanto la programación procedimental como la programación orientada a objetos. Aunque no todas las características estándar de la programación orientada a objetos están implementadas en la versión actual de PHP, muchas bibliotecas y aplicaciones grandes (incluyendo la biblioteca PEAR) están íntegramente escritas usando POO.

Entre las habilidades de PHP se incluyen: creación de imágenes, películas Flash (usando libswf y Ming) y archivos PDF, esta última aprovechada en el módulo de Administración de CVs de este seminario. También puede presentar resultados en formato XHTM y archivos XML. Pero quizás la característica más potente y destacable de PHP es su soporte para una gran cantidad de bases de datos que simplifica la tarea de desarrollar interfaces con BD para aplicaciones Web. Las bases de datos soportadas son:

- ✓ Adabas D
- ✓ dBase
- ✓ Empress
- ✓ FilePro (read-only)
- ✓ Hyperwave
- ✓ IBM DB2
- ✓ Informix
- ✓ Ingres
- ✓ InterBase
- ✓ FrontBase
- ✓ mSQL
- ✓ Direct MS-SQL
- ✓ MySQL
- ✓ ODBC
- ✓ Oracle (OCI7 and OCI8)
- ✓ Ovrimos
- ✓ PostgreSQL
- ✓ Solid
- ✓ Sybase
- ✓ Veloces
- ✓ Unix dbm

Además el paquete del lenguaje cuenta con una extensión DBX de abstracción de base de datos que permite usar de forma transparente cualquier base de datos soportada por ella. Adicionalmente, PHP soporta ODBC (el Estándar Abierto de Conexión con Bases de Datos), así que puede conectarse a cualquier base de datos que soporte tal estándar.

PHP también cuenta con soporte para comunicarse con otros servicios usando protocolos tales como LDAP, IMAP, SNMP, NNTP, POP3, HTTP y COM (en Windows). PHP soporta WDDX para el intercambio de datos entre lenguajes de programación en web y puede utilizar objetos Java de forma transparente como objetos PHP. La extensión de CORBA puede ser utilizada para acceder a objetos remotos.

PHP cuenta con características de procesador de texto tales como intérprete de expresiones regulares POSIX extendidas o tipo Perl y procesador de documentos XML. Para procesar y

acceder a documentos XML, soporta los estándares SAX y DOM, y para transformar documentos XML utiliza la extensión XSLT.

Finalmente, a título informativo, PHP cuenta con extensiones muy interesantes, las funciones del motor de búsquedas mnoGoSearch, funciones para pasarelas de IRC, utilidades de compresión (gzip, bz2), conversión de calendarios, traducción, etc.

En el próximo apartado de este capítulo ampliaré acerca de una librería muy útil que es, podría decir, un “recomendado” a la hora de implementar sistemas que necesitan gestionar accesos por sesiones, PHPLib.

Instalación y configuración.
Dependiendo del uso que se le vaya a dar a PHP hay que considerar si se necesita de un servidor Web o no. La instalación varía según sean las intenciones del programador y del soporte que el intérprete necesitará, es decir si se instalará como módulo de apache o no.

Por otro lado hay que considerar cuál será la plataforma sobre la que se va a trabajar. Pero teniendo presente que la opción de un paquete como Apache2Triad es más que suficiente para las necesidades del módulo de carga y administración de CV, dejaré de lado la instalación de PHP sobre Windows ya que, con este software, un par de clicks bastan para que el Web Server, PHP, el sistema de estadísticas AWStats y tantas otras aplicaciones más queden instaladas y configuradas correctamente de manera que estemos listos para trabajar.

SI bien ya he descrito en forma resumida la instalación de Apache sobre GNU/Linux, en mi experiencia personal siempre resultó menos conflictivo instalar Apache y PHP seguidamente para poder configurar ambos al mismo tiempo y no perder tiempo en la verificación de correspondencia de versiones, dependencias y demás.

Instalación de APACHE y PHP4 desde los fuentes.
Antes que todo, lo primero que debemos hacer es bajar los paquetes correspondientes de dichos programas. Se recomienda bajar los paquetes fuentes y 'NO' los paquetes en RPM, ya que la instalación de estos programas bajo RPM trae una serie de complicaciones para llevarla a cabo.

Lo primero que debemos hacer es saber si tenemos corriendo el servidor de web Apache. Si es así, debemos 'matar el proceso' con la siguiente instrucción:

(como root)
killall -9 httpd

Luego, debemos saber qué tipo de fuente fue instalado. Si se ocupa la distribución de RedHat, lo más probable es que haya sido instalado desde RPM. Para desinstalar dichos paquetes debemos hacer una búsqueda para ver los paquetes que corresponden a Apache, con el siguiente comando :

rpm -qa | grep apache

Si Apache fue instalado desde RPM, el comando anterior nos listará los paquetes de este, como por ejemplo:
rpm -qa | grep apache
Apache-1.3.12
Ya sabiendo los paquetes que corresponden, los desinstalaremos corriendo el comando siguiente con el nombre correspondiente de cada paquete listado:
rpm -e --nodeps Apache-1.3.12

Si Apache no fue instalado desde RPM, debemos dirigirnos al programa controlador de paquetes de nuestra distribución.

Lo siguiente que debemos hacer ahora es descargar los paquetes indicados desde las siguientes direcciones :
Estas versiones son actualmente estables.
Apache 1.3.12 :
http://www.apache.org/dist/apache_1.3.12.tar.gz
PHP 4.0.0:
http://www.php.net/do_download.php?download_file=php-4.0.0.tar.gz&source_site=www.php.net,
o desde el servidor FTP.

Copiamos los paquetes a una carpeta llamada, por ejemplo, 'dirphp':

```
# mkdir ~/dirphp
# cp apache_1.3.12.tar.gz ~/ dirphp /
# cp php-4.0.0.tar.gz ~/ dirphp /
```

Y los descomprimimos:

```
# cd ~/ dirphp
# tar zxfv apache_1.3.12.tar.gz
# tar zxfv php-4.0.0.tar.gz
```

Ahora estamos en condiciones de proceder con la instalación propiamente dicha, con la intención de obtener una configuración estática del modulo PHP en Apache:

```
# cd ~/dirphp/apache_1.3.12
# ./configure
# cd ../php-4.0.0
# ./configure --with-apache=../apache_1.3.12 --enable-track-vars
# make
# make install
```

(por comodidad de configuración y manejo dejaremos Apache instalado bajo /www)

```
# cd ../apache-1.3.12
# ./configure --prefix=/www --activate-module=src/modules/php4/libphp4.a
# make
# make install
```

luego debemos copiar el archivo de inicio de PHP4 :

```
# cd ../php-4.0.0
# cp *.ini-dist /usr/local/lib/php.ini
```

A continuación configuraremos el servidor web activándole el soporte PHP4:

```
# cd /www/conf
```

Dentro del directorio /www/conf debemos editar el archivo httpd.conf y agregar las siguientes líneas o modificarlas en caso de que existan:

```
AddType application/x-httpd-php .php .php3 .php4
AddType application/x-httpd-php-source .phps
```

Si estas líneas comienzan con un '#' debemos sacar el '#', ya que esto para el programa es un comentario del usuario y no una configuración.

Algo opcional que se puede hacer es editar una sección del httpd.conf el cual indica el archivo de inicio de la carpeta para un pagina web. Por defecto esta activado el index.htm, esta sección es la siguiente:

```
<IfModule mod_dir.c>
        DirectoryIndex index.htm
</IfModule>
```

algo mas general que se podría hacer es dejarlo como:

```
<IfModule mod_dir.c>
        DirectoryIndex index.htm index.html index.php index.php3 index.php4
</IfModule>
```

así cualquiera de estos index.x podría ser el archivo de inicio para una carpeta que contiene una pagina web.

- Iniciando el servidor web Apache

Con Apache ya instalado, para correr el servidor debemos ir a la carpeta /www/bin y ejecutar el siguiente comando como root :

```
# cd /www/bin
# ./apachectl start
```

con lo que deberíamos recibir de respuesta si el servidor pudo ser ejecutado sin problemas:

./apachectl start: httpd started

Si es así, nuestro servidor Apache esta corriendo sin problemas, de lo contrario deberemos revisar todos los pasos anteriores.
Para hacer una comprobación final y visual de nuestro servidor web lo que debemos hacer es abrir nuestro explorador favorito y como dirección colocaremos lo siguiente:

127.0.0.1

con lo cual se debería desplayar la pagina de inicio del servidor apache.

Para que nuestro servidor se inicie cada vez que reiniciemos nuestra maquina, debemos agregar la siguiente línea al final del archivo
/etc/rc.d/rc.local :
/www/bin/./apachectl start

- Comprobación del modulo PHP4

Para una comprobación de este modulo lo mas fácil es hacer una pagina con cualquiera de las extensiones php anteriormente habilitadas (.php , .php3, .php4).
Creemos y editemos por ejemplo un archivo llamado test.php, el cual deberá tener el siguiente código:

```
<HTML>
<BODY bgcolor="#ffffff">
<?php printf("PHP Funciona correctamente"); ?>
</BODY>
</HTML>
```

Otra manera es agregar en el archivo test.php el texto siguiente:
<?php phpinfo(); ?>

A continuación debemos copiar el archivo a la carpeta de inicio del servidor Web:
cp test.php /www/htdocs/

Para verificar la instalación del modulo con éxito, deberemos abrir nuestro explorador y colocar la siguiente dirección:
127.0.0.1/test.php

Si PHP fue instalado sin problemas, deberíamos recibir como pagina web el siguiente texto:
"PHP Funciona correctamente"

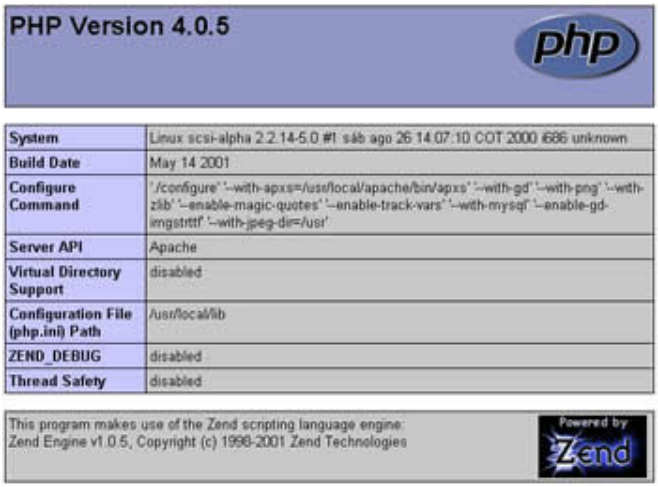


Figura 4 – Página informativa de PHP

De lo contrario puede ser que se esté cometiendo algún pequeño error.

Funcionamiento de PHP.
El gráfico que se observa a continuación es figura repetida en casi todos los trabajos que se publican con el objeto de explicar la ejecución de un script PHP en una aplicación Web o en la creación de páginas dinámicas, pero vale la pena incluirlo cuantas veces haga falta por ser la manera más sencilla de ilustrar el concepto.

Figura 5 – Ejecución de PHP

.

El esquema muestra el proceso que se dispara con la solicitud de una página desde el browser del cliente. El servidor Web recibe la solicitud, ubica la página solicitada dentro de su directorio público, el intérprete PHP busca secciones de código embebidas dentro de etiquetas PHP, las ejecuta y devuelve el resultado en HTML al servidor Web. Éste resuelve la página solicitada y la completa para devolverla al cliente que hizo la solicitud.

El cliente nunca verá el código del programa PHP, sólo le llegarán las páginas HTML que genere el programa. A diferencia de Java Script, que se ejecuta en las máquinas clientes, un programa PHP se ejecuta en el servidor web.

Internándonos más en la arquitectura que interviene en la ejecución de un sistema de BD desarrollado en PHP la Figura 6 resulta explicativa.

PHP es una extensión para servidores web. Lo que hace es ponerse "entre" el servidor y el cliente. Este “preprocesador” toma el código dentro de las páginas, lo ejecuta en el servidor y envía el resultado al cliente. Por ser un lenguaje de scripting, los programas no se compilan, sino sólo se interpretan; esto significa que es más lento en ejecutarse que, por ejemplo, un programa en C, pero al mismo tiempo los cambios en el código PHP tienen efecto de inmediato.

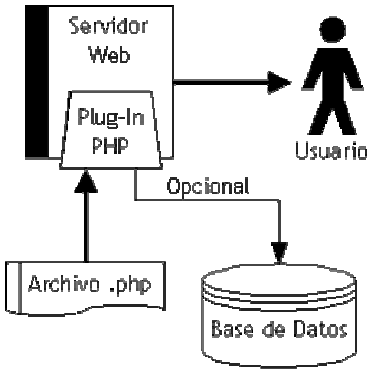


Figura 6 – PHP extensión de Web Server

No todos los archivos son interpretados por el plug-in de PHP, sólo aquellos que hayan sido definidos en la configuración del servidor como tales, puesto que la interpretación de código es un proceso que toma un tiempo mayor al que se emplea en enviar una página tal como está. Es común utilizar una o varias de las siguientes extensiones para informar al servidor que debe interpretarlas: .php, .phtml, .php3 y .php4.

Un aspecto importante a la hora de desarrollar aplicaciones en PHP es la manipulación de errores. Hay 4 tipos de errores y avisos, esto son:

1 - Errores Normales de Funciones (Normal Function Errors)
2 - Avisos Normales (Normal Warnings)
4 - Errores del Analizador de código (Parser Errors)
8 - Avisos (Notices, advertencia que puedes ignorar, pero que puede implicar un error en tu código).

Los 4 números de arriba son sumados para definir un nivel de aviso de error. El nivel de aviso de error por defecto es el nivel 7, el cual es la suma de 1+2+4, es decir todo excepto los avisos. Este nivel puede ser cambiado en el fichero php.ini (o php3.ini) con la directiva error_reporting. También se puede definir en el fichero de configuración del servidor de páginas Apache httpd.conf, con la directiva php3_error_reporting o también se puede cambiar en tiempo de ejecución usando la función error_reporting ().

Todas las expresiones PHP pueden también ser llamadas con el prefijo "@", el cual desactiva el aviso de errores para esa expresión en particular. Si ocurre un error en una expresión en tal situación y la característica track_errors está habilitada, se puede encontrar el mensaje de error en la variable global $php_errormsg. Es muy importante tener en cuenta estos aspectos a la hora de desarrollar una aplicación con PHP.

Variables y Sintaxis PHP.

- **Variables**

Una variable es un espacio para contener datos. El contenido de las variables se puede leer y se puede cambiar durante la ejecución de una página PHP.

Todas las variables PHP comienzan con el símbolo del dólar $ y pueden usarse sin necesidad de definición y/o tipificación previas. En PHP una misma variable puede contener un número y luego puede contener caracteres. Vemos un ejemplo para ilustrar conceptos antes de continuar.


```
<html>
<head><title>Ejemplo_1</title>
</head>
<body>
<?php
```

```
  $a = 1;
  $b = 3.34;
  $c = "Hola Mundo";
  echo $a,"<br>",$b,"<br>",$c;
?>
</body>
</html>
```

Las variables en PHP son diferentes a las de los lenguajes compilados como C y Java. Esto es así por su naturaleza débilmente tipificada, lo cual quiere decir que no es necesario declararlas antes de usarlas, ni a ellas ni a su tipo y como resultado las variables pueden cambiar su tipo cuantas veces cambie su contenido.

PHP no soporta variables globales como otros lenguajes de programación, excepto por algunas variables especiales pre-definidas. Las variables son de alcance local, si fueron creadas dentro de una función están disponibles únicamente durante la vida de esa función. Las que son creadas en el script principal no son globales, no se pueden ver dentro de funciones pero sí pueden accederse usando un arreglo especial $GLOBALS[], usando el nombre de la variable como identificador de posición dentro del mismo.

Una propiedad muy útil de PHP es que se pueden acceder variables usando “referencias indirectas”, es decir se pueden crear y acceder variables por el nombre en tiempo de ejecución. Otro ejemplo para simplificar:

```
<?
$name = "John";
$$name = "Registered user";
print $John;
?>
```

El resultado de esto en pantalla será la frase: *"Registered user"*. Y así sse pueden crear tantos niveles de referencias indirectas como sea necesario sólo con agregar signos $ al comienzo del nombre de variable.

Profundizando un poco más en el manejo de variables podemos destacar tres construcciones disponibles para verificar la existencia de una variable, removerla y/o verificar su valor de verdad.

- ✓ isset() -> Determina si una variable ha sido declarada por PHP. Devuelve TRUE si fue definida y FALSE si no o si lleva asignado el valor NULL.
- ✓ unset() -> Deshace la declaración de una variable y libera espacio en la memoria si ninguna otra hace referencia a su valor. Una llamada a *isstet()* sobre una variable que ha sido *unset()* devuelve FALSE.
- ✓ empty() -> Se usa para verificar si una variable no fue declarada o si su valor es FALSO. Su utilidad radica en comprobaciones sobre el envío de una variable de un formulario o si ésta contiene datos por ejemplo.

Las variables especiales pre-definidas a las que hice referencia anteriormente son las que tienen un comportamiento global, similar al de las variables globales de otros lenguajes. Estas son llamadas en PHP “súper globales” entre las cuales se destacan:

$_GET[]. Un array que incluye todas las variables GET que PHP recibió del browser del cliente.
$_POST[].Un array que incluye todas las variables POST que PHP recibió del browser del cliente.
$_*COOKIE*[].Un array que incluye todas las *cookies* que PHP recibió del browser del cliente.
$_ENV[].Un array que incluye todas las variables de entorno.
$_SERVER[].Un array que contiene los valores de las variables del Web server. Esta es accesible desde cualquier lugar del script ya sea dentro de una función, desde un método o en forma global y no se debe usar $GLOBALS[] para ello.

- **Tipos de datos básicos**

Hay 8 tipos de datos diferentes en PHP, de los cuales 5 son escalares y los 3 restantes de distintiva unicidad. A continuación se extiende una breve referencia de cada uno de ellos.

Enteros - Integers
Los enteros son equivalentes a un tipo *long* de C, por ejemplo. En la mayoría de las PC comunes significa un entero de 32 bits con signo, lo cual se traduce en un rango de valores que va de –2.147.483.648 a +2.147.483.647. Los datos de este tipo pueden ser escritos en notación decimal, hexadecimal (con prefijo 0x), y octal (con prefijo 0), e incluir signo +/-. Algunos ejemplos:
- 240000
- 0xABCD
- 007
- -100

Numéricos de Punto flotante - Floating-Point Numbers
Representan números reales y son equivalentes a un tipo *double* del compilador para la plataforma C. En plataformas comunes el tamaño de este tipo es de 8 bytes y puede tomar valores de 2.2E–308 a 1.8E+308 aproximadamente. Los números de punto flotante incluyen un punto decimal y pueden tener un signo +/- y un exponente. Ejemplos de este tipo son:
- 3.14
- +0.9e-2
- -170000.5
- 54.6E42

Cadenas - Strings
Las strings son secuencias de caracteres que internamente mantienen el cálculo de su longitud, lo cual distingue a PHP de otros lenguajes. Esto permite un sencillo manejo de datos binarios (como por ejemplo crear una imagen on-the-fly y enviar su salida al browser.
La longitud máxima de un string varía de acuerdo a la plataforma y el compilador C pero se puede esperar que soporte aproximadamente 2GB.

Al escribir valores tipo string en el código de un script, para delimitarlos se pueden usa double quotes ("), single quotes (') o here-docs. Los manuales y tutoriales están llenos de consejos sobre *cuándo* usar una u otra forma pero no dicen nada del *por qué*. A continuación se explica brevemente la diferencia.

Comillas Dobles - Double Quotes

Para estar claros de a qué nos referimos veamos algunos ejemplos:
- "PHP: Hypertext Pre-processor"
- "GET / HTTP/1.0\n"
- "1234567890"

Una cadena puede contener prácticamente cualquier caracter. Algunos caracteres no pueden escribirse tal cual se haría comúnmente, estos requieren notación especial:

NOTACIÓN	DESCRIPCIÓN
\n	Newline.
\t	Tab.
\"	Double quote.
\\	Backslash.
\0	ASCII 0 (null).
\r	Line feed.

\$	Escape $ sign so that it is not treated as a variable but as the character $.
\{Octal #}	The character represented by the specified octal #—for example, \70 represents the letter 8.
\x{Hexadecimal #}	The character represented by the specified hexadecimal #—for example, \0x32 represents the letter 2.

Otra particularidad especial de las string acotadas por double-quote es que la notación de algunas variables y expresiones pueden ser usadas de manera embebida. La referencia a variables es reemplazada automáticamente por los valores de las mismas en el código resultante, esto se conoce en otros lenguajes como *interpolación*. Suponiendo que:
$result = 5, $i = 2 y $arr[2] = 'Hola' el resultado de:

a. "The result is $result"
b. "The array offset $i contains $arr[$i]"

Sería:

a. The result is 5
b. The array offset 2 contains Hola
Donde 5 y 2 son los strings "5" y "2".

En casos donde sea necesario concatenar cadenas con valores o expresiones y la sintaxis no es suficiente, se utiliza el . (punto) como operador de concatenación.

Comilla simple - Single Quotes

Las *single quotes* también delimitan cadenas. Sin embargo estas no soportan todos los escapes de las comillas dobles ni la sustitución de variables.

NOTACIÓN	**DESCRIPCIÓN**
\'	Single quote.
\\	Backslash, used when wanting to represent a backslash followed

Dos ejemplos:

- 'Hello, World'
- 'Today\'s the day'

String Offsets

Los caracteres de una cadena pueden accederse individualmente usando el offset o "desplazamiento" que indica la distancia al primer caracter dentro de una cadena siendo 0 (cero) la del primer caracter. La notación es $str{offset}. Hay dos maneras de usarla, en modo lectura y en modo escritura. Para leer un caracter, sólo puede emplearse para acceder índices válidos. Para modificar un caracter, se pueden acceder offsets inexistentes, siendo que si el offset es mayor que el fin de la cadena original por más de una unidad, PHP rellena la cadena con el caracter de espacio (' ') hasta llegar al offset indicado.

Este ejemplo crea e imprime la cadena "Andi":

```
<?
$str = "A";
$str{2} = "d";
$str{1} = "n";
$str = $str . "i";
```

```
print $str;
?>
```

Manejo de cadenas

Dado el uso del lenguaje PHP el tratamiento de cadenas es muy importante, existen bastantes funciones para el manejo de cadenas, a continuación explicaremos las más usadas.

- ✓ **strlen(***cadena***)** -> Nos devuelve el número de caracteres de una cadena.
- ✓ **split(***separador,cadena***)** ->. Divide una cadena en varias usando un carácter separador.
- ✓ **sprintf(***cadena de formato, var1, var2...***)** ->. Formatea una cadena de texto al igual que printf pero el resultado es devuelto como una cadena.
- ✓ **substr(***cadena, inicio, longitud***)** -> Devuelve una subcadena de otra, empezando por inicio y de longitud *longitud*.
- ✓ **chop(***cadena***)** -> Elimina los saltos de línea y los espacios finales de una cadena.
- ✓ **strpos(***cadena1, cadena2***)** -> Busca la cadena2 dentro de cadena1 indicándonos la posición en la que se encuentra.
- ✓ **str_replace(***cadena1, cadena2, texto***)** -> Reemplaza la cadena1 por la cadena2 en el texto.

```
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<?php
  echo strlen("12345"),"<br>";

  $palabras=split(" ","Esto es una prueba");
  for($i=0;$palabras[$i];$i++)
    echo $palabras[$i],"<br>";

  $resultado=sprintf("8x5 = %d <br>",8*5);
  echo $resultado,"<br>";

  echo substr("Devuelve una subcadena de otra",9,3),"<br><br>";

  if (chop("Cadena \n\n ") == "Cadena")
    echo "Iguales<br><br>";

  echo strpos("Busca la palabra dentro de la frase", "palabra"),"<br><br>";

  echo str_replace("verde","rojo","Un pez de color verde, como verde es la hierba."),"<br>";

?>
</body>
</html>
```

Nulo - Null
Este es un tipo de dato que tiene un único valor posible: NULO. Marca variables al quedar vacías y es muy útil para diferenciar entre valores nulos de las BD y la cadena nula. El operador isset($variable) de PHP devuelve FALSE si $variable es NULL mientras exista la variable.

Booleanos
Un booleano puede tomar uno de dos valores, VERDADERO o FALSO. PHP tiene la potencialidad de convertir automáticamente tipos cuando lo necesita, y los boléanos son probablemente el tipo más usado para esas conversiones. Un ejemplo:

```
<?
$numerator = 1;
$denominator = 5;
if ($denominator == 0) {
print "El denominador debe ser un valor distinto de cero \n";
}
?>
```

El resultado del operador es un Booleano, en este caso FALSO y en la ejecución del script la sentencia dentro del if{} no sería ejecutada. Veamos el ejemplo con una variación:

```
<?
$numerator = 1;
$denominator = 5;
if ($denominator) {
/* Perform calculation */
} else {
print "El denominador debe ser un valor distinto de cero \n ";
}
?>
```

Aquí no se usó ningún operador de comparación, de todos modos PHP convirtió $denominator, o para ser más estrictos el valor 5 a su equivalente Booleano, TRUE, para entrar al if{} y luego ejecutar el bloque /* Perform calculation */.

Recursos - Resources
Recursos representan una extensión de los recursos PHP tales como una consulta o una conexión a una BD, un archivo abierto y varios otros tipos externos.

Arreglos - Arrays
Un arreglo es una colección de pares clave/valor, lo cual significa que mapea las claves a los valores. Las claves o índices de un arreglo pueden ser enteros o cadena, y los valores de cualquier tipo (incluso otros arrays).

Dependiendo del estilo del programador y de las necesidades surgidas del diseño de sistemas el tratamiento de los arrays puede hacerse de muchas maneras. Hay varias funciones, métodos y formas de notación a disposición en PHP para crear, acceder y modificar datos de una arreglo o array. A continuación sólo veremos algunos fundamentales.

Array() construct

La forma más clara de declaración de arrays es utilizando una construcción declarativa del lenguaje como la siguiente:

```
array(
        [key =>] value,
        [key =>] value,
        ...);
```

Los elementos dentro de los corchetes son opcionales y si no son especificados, las claves toman valores enteros incrementados de uno en uno automáticamente. En un mismo código se puede utilizar indistintamente la declaración explícita o tácita de las claves, incluso en la declaración de un mismo array. Por ejemplo:

- ✓ array(1, 2, 3) es igual que (0 => 1, 1 => 2, 2=> 3)
- ✓ array("name" => "John", "age" => 28)
- ✓ array(1 => "ONE", "TWO", "THREE") es equivalente a

array(1 => "ONE", 2 => "TWO", 3 => "THREE")
- ✓ array()es un arreglo vacío.

Acceder, crear y modificar elementos de un array

La forma más común de acceder elementos de una array es mediante la notación $arr[key] para cualquier tipo de *key* o $arr[“key”] si la cadena de *key* es una constante. Para crear elementos dentro de un array podemos utilizar una sentencia de asignación con la misma notación, de la siguiente forma $arr[2] =2; por ejemplo.

A continuación analizaremos un ejemplo más complejo para entender no sólo el funcionamiento de las sentencias de asignación y lectura de valores sino ver las formas más usadas de recorrido e impresión de un array, la construcción *foreach()* y la función *print_r()*.

La sintaxis genérica para *foreach()* es:

```
foreach($array as [$key =>] [&] $value) {
/* Bloque de código a ejecutar*/
}
```

[$key =>] y [&] son opcionales. Cuando se especifica [$key =>] éste contiene la clave del valor iterado. [&] se utiliza para modificar el valor de $value y propagarlo en el array. Veamos el ejemplo:

```
<?
$people = array(1 => array("name" => "John", "age" => 28),array("name" => "Barbara", "age" => 67));
foreach ($people as $key => $person) {
if ($person["name"] == “John”) {
$person["age"] ++;
}
}
foreach ($people as &$person) {
if ($person["age"] >= 35) {
$person["age group"] = "Old";
} else {
$person["age group"] = "Young";
}
}
print_r($people);
?>
```

Aquí utilizamos la función *print_r()* para visualizar el resultado de manera legible, y lo que se hace es agregar a cada arreglo correspondiente a una “persona” una clasificación dentro de un grupo de edad en una nueva entrada identificada como “age_group”. La salida de este código será:

```
Array
(
[1] => Array
(
[name] => John
[age] => 29
[age group] => Young
)
[2] => Array
(
[name] => Barbara
```

```
[age] => 67
[age group] => Old
)
)
```

Cabe destacar que la utilidad de la función print_r() es muy valorable a la hora de depurar un script.

Constantes
En PHP, se pueden definir nombres, llamados constantes, para valores simples. Una vez que un valor es asignado a un nombre no se puede cambiar el valor del mismo. Los nombres de las constantes siguen las mismas reglas que las variables en PHP excepto por la ausencia del signo $ al comienzo de los mismos. El uso común en la mayoría de los lenguajes de programación es nombrar las constantes con letras mayúsculas, sin embargo esto no es obligatorio, sino más bien una convención.

A diferencia de las variables, las constantes son accesibles globalmente una vez definidas. La sintaxis general para su declaración es mediante la función:

```
define("CONSTANT_NAME", value [, case_sensitivity]);
```

Donde:
- ✓ "CONSTANT_NAME" es un string.
- ✓ value es cualquier expresión PHP válida, excepto arrays y objetos.
- ✓ case_sensitivity es un Boolean (true/false) y es opcional. El valor por defecto es TRUE.

Un ejemplo sencillo de una constante es la definición de un booleano:

```
<?
define("MI_OK", 0);
define("MI_ERROR", 1);
...
if ($error_code == MY_ERROR) {
print("Se encontró un error\n");
}
?>
```

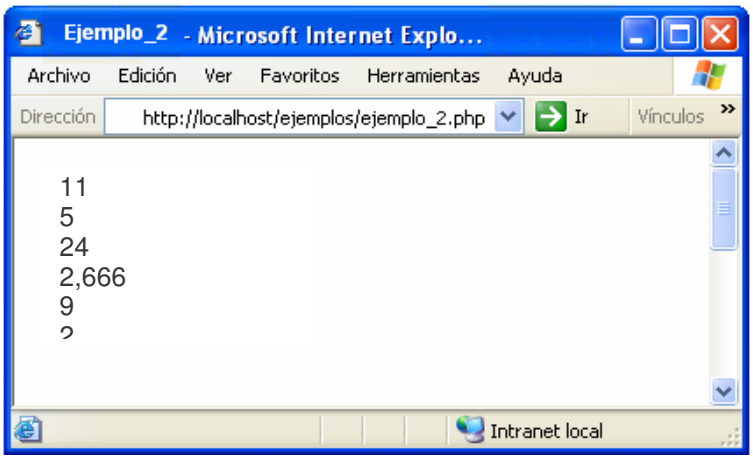
- **Operadores**

Los operadores de PHP son muy parecidos a los de C y JavaScript, si usted conoce estos lenguajes le resultaran familiares y fáciles de reconocer.

Los operadores ARITMÉTICOS se pueden aplicar a las variables y constantes numéricas.

Operador	Nombre	Ejemplo	Descripción
+	Suma	5 + 6	Suma dos números
-	Resta	7 – 9	Resta dos números
*	Multiplicación	6 * 3	Multiplica dos números
/	División	4 / 8	Divide dos números
%	Módulo	7 % 2	Devuelve el resto de dividir ambos números, en este ejemplo el resultado es 1
++	Suma 1	$a++	Suma 1 al contenido de una variable.
--	Resta 1	$a--	Resta 1 al contenido de una variable

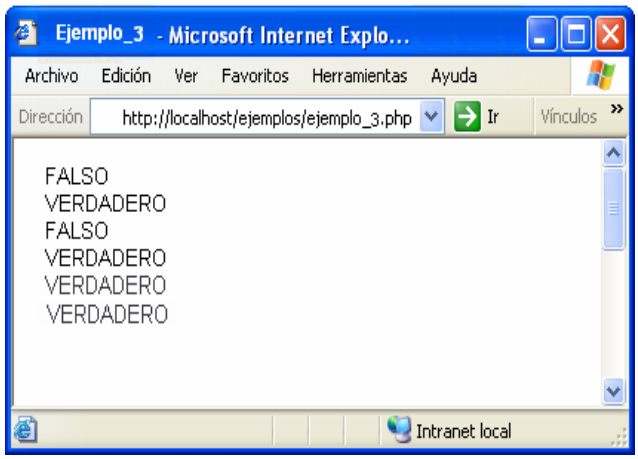
```
<html>
<head>
  <title>Ejemplo_2</title>
</head>
<body>
<?php
  $a = 8;
  $b = 3;
  echo $a + $b,"<br>";
  echo $a - $b,"<br>";
  echo $a * $b,"<br>";
  echo $a / $b,"<br>";
  $a++;
  echo $a,"<br>";
  $b--;
  echo $b,"<br>";
?>
</body>
</html>
```



Los operadores de COMPARACIÓN son usados para comparar valores y así poder tomar decisiones.

Operador	Nombre	Ejemplo	Devuelve cierto cuando:
==	Igual	$a == $b	$a es igual $b
!=	Distinto	$a != $b	$a es distinto $b
<	Menor que	$a < $b	$a es menor que $b
>	Mayor que	$a > $b	$a es mayor que $b
<=	Menor o igual	$a <= $b	$a es menor o igual que $b
>=	Mayor o igual	$a >= $b	$a es mayor o igual que $b

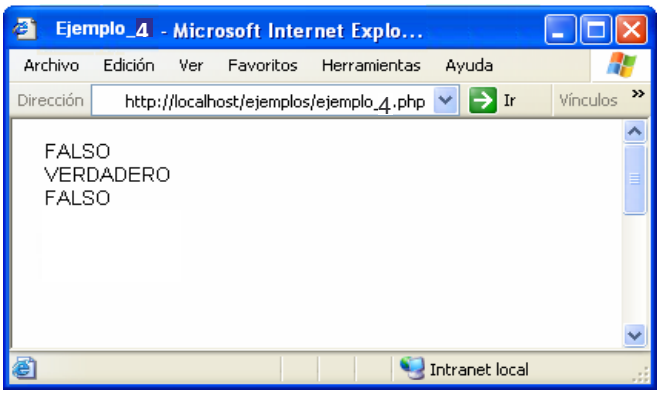
```
<html>
<head>
  <title>Ejemplo_3</title>
</head>
<body>
<?php
  $a = 8;
  $b = 3;
  $c = 3;
  echo $a == $b,"<br>";
  echo $a != $b,"<br>";
  echo $a < $b,"<br>";
  echo $a > $b,"<br>";
  echo $a >= $c,"<br>";
  echo $b <= $c,"<br>";
?>
</body>
</html>
```



Los operadores LÓGICOS son usados para evaluar varias comparaciones, combinando los posibles valores de estas.

Operador	Nombre	Ejemplo	Devuelve cierto cuando:
&&	Y	(7>2) && (2<4)	Ambas condiciones son verdaderas.
and	Y	(7>2) and (2<4)	Ambas condiciones son verdaderas
||	O	(7>2) || (2<4)	Al menos una de las dos es verdadera.
or	O	(7>2) or (2<4)	Al menos una de las dos es verdadera.
!	No	! (7>2)	Niega el valor de la expresión.

```
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<?php
  $a = 8;
  $b = 3;
  $c = 3;
  echo ($a == $b) && ($c > $b),"<br>";
  echo ($a == $b) || ($b == $c),"<br>";
  echo !($b <= $c),"<br>";
?>
</body>
</html>
```



Las SENTENCIAS CONDICIONALES permiten ejecutar o no unas ciertas instrucciones dependiendo del resultado de evaluar una condición. Las más frecuentes son la instrucción if y la instrucción switch.

- **Sentencia if ... else**

```
<?php
  if (condición)
  {
    Sentencias a ejecutar cuando la
    condición es cierta.
  }
  else
  {
    Sentencias a ejecutar cuando la
    condición es falsa.
  }
?>
```

La sentencia if ejecuta una serie de instrucciones u otras dependiendo de la condición que le pongamos. Probablemente sea la instrucción más importante en cualquier lenguaje de programación.

```
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<?php
  $p = 8;
  $q = 3;
  if ($p < $q)
  {
    echo "p es menor que q";
  }
  else
  {
    echo "p es mayor que q";
  }
?>
</body>
</html>
```

En este ejemplo la condición no es verdadera por lo que se ejecuta la parte de código correspondiente al else.

- **Sentencia switch ... case**

```
<html>
<head>
<title>Ejemplo de PHP</title>
</head>
<body>
<?php
  $posicion = "arriba";
    switch($posicion) {
  case "arriba":   // Bloque 1
       echo "La variable contiene";
       echo " el valor arriba";
       break;
    case "abajo":   // Bloque 2
       echo "La variable contiene";
       echo " el valor abajo";
       break;
   default:   // Bloque 3
       echo "La variable contiene otro valor";
       echo " distinto de arriba y abajo";
  }
?>
</body>
</html>
```

Con la sentencia switch podemos ejecutar unas u otras instrucciones dependiendo del valor de una variable, en el ejemplo anterior, dependiendo del valor de la variable $posicion se ejecuta el bloque 1 cuando el valor es "arriba", el bloque 2 cuando el valor es "abajo" y el bloque 3 si no es ninguno de los valores anteriores.

Los BUCLES nos permiten iterar conjuntos de instrucciones, es decir repetir la ejecución de un conjunto de instrucciones mientras se cumpla una condición.

- **Sentencia while**

```
<?php
  while (condición)
  {
    instrucciones a ejecutar.
  }
?>
```

Mientras la condición sea cierta se reiterará la ejecución de las instrucciones que están dentro del while.

```
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
Inicio<BR>
<?php
  $i=0;
  while ($i<10)
  {
```

```
    echo "El valor de i es ", $i,"<br>";
    $i++;
  }
?>
Final<BR>
</body>
</html>
```

En el siguiente ejemplo, el valor de $i al comienzo es 0, durante la ejecución del bucle, se va sumando 1 al valor de $i de manera que cuando $i vale 10 ya no se cumple la condición y se termina la ejecución del bucle.

- **Sentencia for**

```
<?php
  for (incial ; condición ; ejecutar en iteración)
  {
    instrucciones a ejecutar.
  }
?>
<!-- Ejemplo del ciclo for -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
Inicio<BR>
<?php
  for($i=0 ; $i<10 ; $i++)
  {
    echo "El valor de i es ", $i,"<br>";
  }
?>
Final<BR>
</body>
</html>
```

La instrucción for es la instrucción de bucles más completa. En una sola instrucción nos permite controlar todo el funcionamiento del bucle.

El primer parámetro del for, es ejecutado la primera vez y sirve para inicializar la variable del bucle, el segundo parámetro indica la condición que se debe cumplir para que el bucle siga ejecutándose y el tercer parámetro es una instrucción que se ejecuta al final de cada iteración y sirve para modificar el valor de la variable de iteración.

La que sigue es una sentencia que maneja la salida por pantalla de ciertas cadenas. Hasta ahora hemos usado la instrucción echo para realizar salida a pantalla, esta instrucción es bastante limitada ya que no nos permite formatear la salida. En esta página veremos la instrucción printf que nos da mucha más potencia.

- **Sentencia printf**

```
<?php
  printf(cadena formato, variable1, variable2...);
?>
```

La cadena de formateo indica cómo se han de representar las valores que posteriormente le indicaremos. La principal ventaja es que además de poder formatear los valores de salida, nos permite intercalar texto entre ellos.

```
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<?php
  printf("El numero dos con diferentes formatos: %d %f %.2f",2,2,2);
?>
</body>
</html>
```

La cadena de formato puede incluir una serie de caracteres especiales que indican como formatear las variables que se incluyen en la instrucción.

Elemento	Tipo de variable
%s	Cadena de caracteres.
%d	Número sin decimales.
%f	Número con decimales.
%c	Carácter ASCII.

Aunque existen otros tipos, estos son los más importantes.

```
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<?php
  $var="texto";
  $num=3;
  printf("Puede fácilmente intercalar <b>%s</b> con números <b>%d</b> <br>",$var,$num);

  printf("<TABLE BORDER=1 CELLPADDING=20>");
  for ($i=0;$i<10;$i++)
  {
    printf("<tr><td>%10.d</td></tr>",$i);
  }
  printf("</table>");
?>
</body>
</html>
```

- **Funciones**

El uso de funciones aporta la capacidad de agrupar varias instrucciones bajo un solo nombre y la posibilidad de llamarlas varias veces desde diferentes sitios, ahorrando la reiteración de código dentro del script.

```
<?php
  function Nombre(parametro1, parametro2...)
  {
    instrucción1;
    instrucción2;
    instrucción3;
    instrucción4;

    return valor_de_retorno;
```

```
  }
?>
```

Opcionalmente podemos pasarle parámetros a las funciones que se trataran como variable locales y así mismo podemos devolver un resultado con la instrucción return valor; lo que produce la terminación de la función retornando un valor.

```
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<?php

  function media_aritmetica($a, $b)
  {
    $media=($a+$b)/2;
    return $media;
  }

  echo media_aritmetica(4,6),"<br>";
  echo media_aritmetica(3242,524543),"<br>";

?>
</body>
</html>
```

- **Librerías**

A la vez que las funciones ahorran código reiterativo, las librerías permiten agrupar varias funciones y variables en un mismo fichero, de manera que luego podemos incluir esta librería en distintas páginas y disponer de esas funciones fácilmente. Con el código del ejemplo creamos una librería *‘´libreria01.phtml’* que contenga 2 funciones:

```
<?php
  function CabeceraPagina()
  {
?>
  <FONT SIZE="+1">Esta cabecera estará en todas sus páginas.</FONT><BR>
  <hr>
<?
  }

  function PiePagina()
  {
?>
  <hr>
  <FONT SIZE="-1">Este es el pie de página.</FONT><BR>
  Manual de php
<?
  }
?>
```

La librería puede guardarse con extensión .phtml, .inc.php, etc. La instrucción para incluir una librería en una página creada por nosotros es *include("nombre de librería");*

✓ Página 1

```
<html>
<head>
  <title>Página 1</title>
</head>
<body>
<?php include("libreria01.phtml") ?>
<?php CabeceraPagina(); ?>

Página 1
<BR><BR><BR><BR><BR>

Contenido curso de php <BR><BR>
Teórico - práctico...<BR><BR>

fin<BR><BR>

<?php PiePagina(); ?>
</body>
</html>
```

✓ Página 2

```
<html>
<head>
  <title> Página 2</title>
</head>
<body>
<?php include("libreria01.phtml") ?>
<?php CabeceraPagina(); ?>

Esta es otra página<BR><BR>
completamente distinta<BR><BR>
pero comparte el pie y la cabecera con la otra.<BR><BR>

<?php PiePagina(); ?>
</body>
</html>
```

Otras instrucciones para incluir librerías que difieren conceptualmente de include son include_once, require y require_once. En la siguiente tabla se resumen sus respectivas particularidades.

Sentencia	**Acción**	**Ante un ERROR**	**Mensaje de Error**
include()	Incluye y evalúa el archivo especificado.	El script seguirá funcionando de todas maneras.	*Warning*: Indica un problema que debía ser atrapado por el script antes de que el llamado fuera hecho.
require()	Incluye y evalúa el archivo especificado.	Si el fichero especificado no se encuentra detiene la ejecución del script.	*Fatal_error*: Indica un problema del que no puede realizarse una recuperación.
include_once()	Ídem a *include* pero si el fichero ya fue incluido antes no lo vuelve a incluir.	El script seguirá funcionando de todas maneras.	*Warning*: Indica un problema que debía ser atrapado por el script antes de que el llamado fuera hecho.

require_once()	Ídem a *require* pero si el fichero ya fue incluido antes no lo vuelve a incluir.	Si el fichero especificado no se encuentra detiene la ejecución del script.	*Fatal_error*: Indica un problema del que no puede realizarse una recuperación.

Nota: include_once() y require_once() fueron incorporado recién en la versión 4.0.1pl2 de PHP. Estas instrucciones deben usarse cuando se quiere estar seguro de evitar redefiniciones de funciones o reemplazo de valores de variables y no se puede asegurar que una misma librería, por ejemplo, no fue llamada antes.

La librería PHPLib

PHPLIB es una librería con licencia GPL (Licencia General Pública) y permite de forma sencilla trabajar con diferentes bases de datos de forma transparente, gestionar sesiones de usuarios y llevar un control de acceso basado en bases de datos. Todo esto es posible sin necesidad de "ensuciar" el código del script con extensas líneas de conexión a BD y demás.

Dado que en el sistema CV-DGR sólo se empleó PHPLib para el manejo de sesiones, esta reseña estará limitada exclusivamente a describir los aspectos básicos y necesarios que hacen al funcionamiento de la librería con ese fin.

Las características principales de PHPLib se podrían resumir describiendo las siguientes clases:

- ✓ **DB_Sql** Esta clase es una abstracción de las bases de datos. Hasta el momento hay 8 implementaciones: MySQL, MSSQL (MS SQL Server), PostgreSQL, ODBC, Sybase, MSQL, Oracle y OCI8. Básicamente esta clase realiza todas las tareas de acceso a BD que se harían mediante las funciones de PHP, con la diferencia que PHPLib lo hace de forma abstracta. Esto permite cambiar de BD sin la necesidad re-escribir el código fuente.
- ✓ **Session** Se encarga del manejo de sesiones. El manejo de sesiones lo hace a través de *cookies*.
- ✓ **Auth** Se encarga de la "autenticación" de las sesiones. Esto se hace a través de un usuario/password y se puede setear un tiempo de expiración.
- ✓ **Perm** Se encarga del manejo de permisos. Se puede proteger una página entera o sólo cierta parte de ella. De no tener los permisos necesarios se muestra una pantalla de login. Esta clase facilita enormemente la creación de un sistema de privilegios basado en perfiles de usuario, donde cada perfil de usuario permitirá el acceso a sólo una cierta parte de la Base de Datos, o por ejemplo, a sólo ciertos módulos/opciones de la aplicación web.
- ✓ **Cart** El típico Carro de Compras para aplicaciones web de comercio electrónico.
- ✓ **Template** Una implementación de Templates, o plantillas HTML. Uno de los problemas fundamentales con lenguajes como PHP, donde se mezcla el código PHP con el código HTML, es que no está bien delimitado el campo del diseñador HTML y del programador y que el código resultante puede ser complejo de entender y por lo tanto, muy complicado de mantener. Esta clase permite separar el diseño de la programación de aplicaciones web.

Por último los HTML widgets que son, básicamente, formas de dibujar tablas y formularios dentro de una estructura OO y más automatizada. A una tabla, se le puede pasar como argumento un objeto DB_Sql y ésta generará automáticamente la tabla con los resultados del query.

Las clases de los HTML widgets son:

- ✓ **Table** Crea una tabla HTML desde un array bidimensional o desde el resultado de un query (de un objeto DB_Sql). Se puede elegir que columnas mostrar y se puede mostrar una cabecera con el nombre de cada columna. Soporta StyleSheets para poder darle la

apariencia deseada a la tabla (especificando que class usar). También se puede agregar un checkbox a cada fila para poder realizar operaciones con ellas a través de un formulario. (Clase independiente);
- ✓ **CSV_Table** Muestra un dump de un array bidimensional o un resultado de un query en formato CSV (Necesita Table, subclase de Table);
- ✓ **Sql_Query** Es una interfaz para permitir a un usuario hacer un query de una forma simple a través de un formulario. Con esos datos nos genera automáticamente una cláusula "WHERE" para incluir en el query. (Necesita Session);
- ✓ **Form** Crea un formulario HTML. Esta clase es bastante compleja, como características principales se destacan la validación, ya sea desde el cliente (con JavaScript) o desde el server, que permite "congelar" algunos elementos del formulario para que no puedan ser modificados y cada elemento del formulario (checkbox, list, text, etc) esta vinculado a una clase que puede ser fácilmente extendida y personalizable (Clase independiente).

Pero la característica fundamental de PHPLib es la abstracción de la Base de Datos. Esto significa que, una vez instalada la librería, el trabajo con una BD se hará mediante clases, y ya no será necesario ocupar las funciones del PHP específicas para cada BD.

Instalación
El primer paso es, obviamente, descargar la versión de PHPLib, que mejor se ajuste a la plataforma sobre la que se está trabajando, desde la página oficial: http://phplib.sourceforge.net.
Su instalación es algo compleja ya que toda su funcionalidad principal utiliza base de datos. Por lo tanto es requisito tener una base de datos funcionando en el sistema. Para resumir y concentrar la atención en la configuración de la librería, se supone ya instalado algún SGBD como MySQL, PostgreSQL o MS SQL Server.

- En Linux

Se debe crear un directorio llamado php fuera del DocumentRoot del servidor web. Por ejemplo, si el DocumentRoot es /var/www/html, se debe crear el directorio /var/www/php. Luego se descomprime el archivo tar.gz del PHPLib, y se copia el contenido del directorio php/ en /var/www/php.

```
# tar xzvf phplib-7.4-pre2.tar.gz
# cd phplib-7.4-pre2
# cd php
# cp * /var/www/html
```

Si se desea usar PHPLib para un sitio que está en un webhosting, se podría solicitar la creación de un directorio fuera del DocumentRoot, y luego subir ahí todos los archivos del PHPLib.

El archivo de configuración donde se encuentra toda la información acerca de “qué debe cargarse junto con PHP” es el php.ini como se había indicado anteriormente. Si se tiene acceso a los archivos de configuración de la máquina, se podría modificar el archivo /etc/php.ini para que la inclusión de los archivos del PHPLib sea más automática, aunque no es necesario. Además, se debe actualizar el include_path para que contenga la ruta donde se encuentran los archivos del PHPLib:
► include_path = ".:/usr/share/php:/var/www/php";

En auto_prepend_file se puede agregar el archivo prepend.php, que contiene las llamadas a las distintas funcionalidades del PHPLib. Al hacer este cambio, el archivo prepend.php será "incluido" automáticamente en todas las páginas .php que se tengan en la máquina.
► auto_prepend_file = "/var/www/php/prepend.php";

Si se pretende usar PHPLib para el manejo de sesiones se deberán habilitar las siguientes variables:
► track_vars = On
► magic_quotes_gpc = On

Se debe reiniciar el Apache para que estos cambios se hagan efectivos
service httpd restart

En caso de un sitio que está en un webhosting, la solución sería poner la siguiente línea en cada archivo .php que haga uso de PHPLib:
include ("ruta_al_phplib/prepend.php");

Con esto, tenemos lista la instalación de PHPLib.

- En Windows

La única variación para la instalación de estas librerías en un entorno Windows es la de las rutas que no siguen el mismo esquema que en Linux ya que la instalación de PHP se define en un directorio diferente. Por todo lo demás los pasos y recursos se mantienen:
1. Descargar los archivos para la instalación en Windows.
2. Descomprimir los archivos en una carpeta creada para tal fin que puede encontrarse dentro del directorio DocumentRoot.
3. Actualizar las rutas del sistema.

Los archivos prepend.php y local.inc
En la configuración de estos archivos es donde todo cobra sentido. En el archivo prepend.php están las variables que serán tomadas como directivas para que el intérprete PHP sepa con cuál BD debe trabajar. Se de be corroborar la existencia de la siguiente línea según sea el caso:

Directiva	**Motor de BD**
require($_PHPLIB["libdir"] . "db_mysql.inc");	MySQL
require($_PHPLIB["libdir"] . "db_oracle.inc");	Oracle
require($_PHPLIB["libdir"] . "db_pgsql.inc");	PostgreSQL

En el archivo local.inc se definen las clases necesarias para poder operar con una BD. Este mismo archivo incluye a modo de ejemplo una clase llamada DB_Example. Esta se podría usar como base:

```
class DB_prueba extends DB_Sql {

  var $Host     = "nombrehost";      # nombre del host donde se encuentra la BD
  var $Database = "nombrebd";        # nombre de la BD
  var $User     = "usuario";         # nombre de usuario
  var $Password = "aabbcc";          # password
  var $Remote   = 1;                 # 0 si la BD está en la misma máquina, 1 de lo contrario
 }
```

Nota: En esta sección el caracter # en los párrafos que contengan porciones de código marca el inicio de líneas de comentario en verde, las cuales se pueden usar para auto documentar el código de un script.

Específicamente, para gestionar sesiones, en stuff/ dentro del directorio que se generó al descomprimir el archivo phplib-7.x-tar.gz, se tienen los scripts create_database. Se puede usar el script respectivo para generar las tablas ad hoc active_sessions y auth_user.

En esta versión del CV-DGR no se han incluido el manejo de errores, pero de considerarse se debería agregar una línea en la definición de clase como:

```
var $Halt_On_Error="no";       # En este caso no se reportarán los errores
```

que funciona según indica el cuadro:

$Halt_On_Error=	Al encontrarse un error de operación	Asigna valor Errno (nro) y Error (mensaje)?
“yes”	Se mostrará cualquier error de BD y se detendrá la ejecución de la aplicación.	Si
“no”	Se mostrarán los mensajes de error de la BD pero continuará la ejecución de la aplicación.	Si
“report”	No se reportarán errores de BD y no se detendrá la ejecución de la aplicación.	Si

Sintaxis básica
Una vez definida una clase de BD, la conexión a la BD se hace creando el objeto correspondiente:

```
$db=new DB_prueba();                                    # nueva instancia de la BD
```

Luego, para realizar una consulta sólo se debe escribir la consulta e invocar el método correspondiente que la ejecuta:

```
$sel="SELECT campo1,campo2 FROM tabla";      # consulta SQL
$db->query($sel);                                       # se hace la consulta
```

Y por último pueden leerse los registros resultantes y presentarse en forma de tabla:

```
echo "<table border='1'>\n";
echo "<tr><td>Campo1</td><td>Campo2</td></tr>\n";

 while ($db->next_record()) {
     echo "<tr><td>".$db->f('campo1')."</td><td>".$db->f('campo2')."</td></tr>\n";
  }
 echo "</table>\n";
```

El método next_record() que recorre los registros obtenidos mediante un select, al llegar al último registro + 1 cierra la conexión a la BD. Para recuperar el valor de una propiedad de la clase se utiliza la siguiente forma:
$q->variable # representa una variable de la clase.

Dentro de una clase la llamada a las funciones y variables se hace con la palabra reservada this. Por lo tanto cuando un $this->algo, es una llamada a un método de la clase o alguna operación con variables, basta observar si algo va seguido de paréntesis o no para diferenciar a uno de otro.

La principal ventaja del uso de clases no es visible para el observador que pretenda encontrar grandes diferencias en cuanto a cantidad de líneas empleadas u otros aspectos similares. La misma radica en que si se quiere usar este código para ejecutar la misma consulta sobre una BD PosgtreSQL u Oracle, basta con modificar el archivo prepend.php indicando db_pgsql.php o db_oracle.php.

Dado que para este sistema sólo empleé PHPLib para el manejo de sesiones no me extenderé sobre la utilidad de métodos como query(), num_rows() y affected_rows(), vitales en cualquier script que implemente formularios de consulta y presentación de datos, por ejemplo. En los párrafos que siguen describiré las clases que intervienen en el manejo de sesiones, autenticación y privilegios de usuarios en el CV-DGR, con sus respectivos métodos y variables.

Manejo de sesiones
Una sesión es, básicamente, la secuencia de páginas que un usuario visita en un sitio web

desde que entra, hasta que lo abandona. Pero profundizando más sobre este concepto, se denomina sesión al método por el cual un usuario accede a un sitio o una aplicación Web con permisos de acceso específicos o la posibilidad de garantizar la autenticidad de su identidad.
El término sesión en PHP, session en inglés, se aplica a esta secuencia de navegación. Para distinguirla se crea un identificador único que se utiliza también para consultar sus valores particulares y sus métodos. A este identificador de sesión se le denomina comúnmente *sesión*.

Las instrucciones para el manejo de sesiones pueden resumirse bajo la siguiente secuencia lógica:

1. Existe una sesión?
2. Si existe la retomamos.
3. Si no existe creamos una nueva.
4. Generar un identificador único.

Para no perder el hilo de la navegación del usuario se debe asociar esta sesión a todas las URLs y acciones de formulario. Se podría crear una *cookie* que incluya el identificador de sesión, pero es bueno recordar que la disponibilidad o no de las *cookies* depende del navegador del usuario, y no es conveniente confiarse de lo que un usuario pueda o no tener habilitado.

- La clase Session

Esta clase mantiene una lista de propiedades (variables de una clase) y provee de un conjunto de funciones (métodos) para traerlas y guardarlas desde y hacia un contenedor de datos (storage container).

Variables	**Descripción**
classname	El nombre de la clase.
magic	String secreto usado en la creación del ID.
mode	Modo de propagación del ID de Session. *cookie* o get.
fallback_mode	Modo de propagación del Session ID en caso de que $mode no funcione. Dejar $mode en *cookie* y $fallback_mode en get.
lifetime	Duración de la *cookie* en minutos. Si se deja en 0 quedan como *cookies* de sesión.
gc_time	Manejo de garbage collection.
gc_probability	Manejo de garbage collection.
allowcache	Control del cache de sesión de las páginas, si se deja en no (por defecto), la página no queda en cache bajo HTTP/1.1 o HTTP/1.0; si se deja en public, la página queda cache público bajo HTTP/1.1 y HTTP/1.0; si se deja en private, la página queda en cache privado bajo HTTP/1.1 y no es dejada en cache bajo HTTP/1.0.
allowcache_expires	Cantidad de minutos a permanecer en cache, cuando esta habilitado el caching.
that_class	Nombre de la clase que especifica nuestro contenedor de datos. Session usa esta clase para almacenar y obtener datos.
auto_init	Nombre de archivo a ser cargado al establecerse la sesión.
secure_auto_init	Se deja en 0, si todas las páginas siempre llaman a page_close().

Ejemplo de una sub-clase de Session

```
class Prueba_Session extends Session {
  var $classname = "Prueba_Session";
   var $cookiename     = "Prueba";            ## por defecto mismo nombre que clase
  var $magic          = "cambiaresto";       ## semilla del ID
  var $mode           = "cookie";            ## los session ID se propagan con cookies
  var $fallback_mode  = "";
  var $lifetime       = 0;                   ## 0 = crear session cookies, sino minutos
  var $that_class     = "Prueba_CT_Sql";     ## nombre del contenedor de datos
```

```
  var $gc_probability = 5;
  var $allowcache     = "no";
}
```

- La clase del contenedor de datos

La clase Session necesita de un "contenedor" donde serán almacenadas las variables de sesión. Este "contenedor" se indica en la variable $that_class. En el ejemplo:
var $that_class   = "Prueba_CT_Sql";
y luego se debe definir la sub-clase del contenedor:

```
class Prueba_CT_Sql extends CT_Sql {
  var $database_class = "DB_prueba";        ## A que base de datos conectarse...
  var $database_table = "active_sessions"; ## obtener información sobre sesiones en esta tabla
}
```

- La clase Auth para Autenticación

La siguente clase, a su vez sub-clase de Auth, se llamará Prueba_Auth. En ella basta definir el nombre de la clase *$classname*, duración de la sesión autentificada en minutos *$lifetime*, la clase de conexión a la BD *$database_class* y la tabla con la información de los usuarios *$database_table*. Así la clase Prueba_Authresultante es:

```
class Prueba_Auth extends Auth {
  var $classname = "Prueba_Auth";
  var $lifetime = 15;
  var $database_class = "DB_Prueba";
  var $database_table = "auth_user";

  function auth_loginform() {
    global $sess;
    global $_PHPLIB;

# nombre de archivo html que mostrará el formulario usuario/contraseña cuando no exista una
# sesión autentificada o esta haya expirado. Por defecto este archivo se encuentra en
# $_PHPLIB["libdir"]

    include($_PHPLIB["libdir"] . "loginform.ihtml");

  }

  function auth_validatelogin() {
    global $HTTP_POST_VARS;

    if(isset($HTTP_POST_VARS["username"])) {
      $this->auth["uname"] = $HTTP_POST_VARS["username"];
    }

# uid se deja en falso.
    $uid = false;

# se consulta si la combinación usuario/contraseña existe en tabla de usuarios.
# Aunque extenso, el argumento de la función Quero es un SELECT, pero armado con la
# función sprintf que inserta las cadenas de “username” y “password” con slashes en los
# lugares especificados por %s.
    $this->db->query(sprintf("select user_id, perms ".
                             "        from %s ".
                             "       where username = '%s' ".
                             "         and password = '%s'",
                         $this->database_table,
```

```
                addslashes($HTTP_POST_VARS["username"]),
                addslashes($HTTP_POST_VARS["password"])));


# se obtiene el uid y los permisos del usuario si existe.
   while($this->db->next_record()) {
     $uid = $this->db->f("user_id");
     $this->auth["perm"] = $this->db->f("perms");
   }

# retorna el uid, si usuario no existe retornará uid = false.
   return $uid;
  }
}
```

Una vez configuradas correctamente las clases de sesión y autentificación, se puede crear una página restringida sólo a usuarios válidos, más específicamente aquellos que se encuentran en la tabla auth_user. Basta utilizar las funciones page_open() y page_close(), al inicio y final de la página respectivamente para que todo lo que se encuentre entre estas funciones sólo pueda ser ejecutado por usuarios autentificados.

```
<?php
# page_open recibe como parametros las clases sess, auth y perm
# previamente configuradas, validando al usuario/contraseña ingresado
# vía el formulario que se muestra la primera vez.

page_open(array("sess" => "Prueba_Session",
          "auth" => "Prueba_Auth",
          "perm" => "Example_Perm"));
// Bloque de código restringido a usuarios válidos

# la funcion page_close() se debe ejecutar siempre al final de la
# página para que se guarde/mantenga la info de la sesión.

page_close();
 ?>
```

La idea es que cada página restringida llame a las funciones page_open() y page_close(), y cada vez que se acceda a las páginas se verificará si existe una sesión autentificada, si no es el caso, se mostrará el formulario que solicita el ingreso de usuario y contraseña.

Para terminar se puede crear una página salir.php que elimine una sesión ejecutando el método logout() de la clase Auth: $auth->logout();
Para esto, que quedaría como:

```
<?php
# verificamos si existe una sesión
page_open(array("sess" => "Prueba_Session", "auth" => "Prueba_Auth","perm" =>
"Example_Perm"));

# se termina la sesión
$auth->logout();

# volvemos al index.php sin mostar nada en pantalla
header("location: index.php");
?>
```

- La clase Perm, manejo de privilegios.

El manejo de privilegios depende de una sesión autentificada. Se asocia un conjunto de

permisos requeridos por una página. El contenido de la página es sólo visible para usuarios con TODOS los privilegios correspondientes; el resto de los usuarios verán una página personalizada o un mensaje de error.

Variables de Perm	Descripción
classname	Nombre de la clase.
permissions	Un hash de (nombre, bit de permiso) pares.

Métodos de Perm	Descripción
check($required)	Verifica si el usuario autentificado tiene todos los privilegios que se especifican en $required. Si no, se llama al método perm_invalid(). Si uno o más de los permisos requeridos no son válidos, también se llama a perm_invalid().
have_perm($required)	Similar a check(), sólo que no detiene la ejecución si el usuarios no tiene los permisos correspondientes. Se puede usar para restringir solo ciertas partes de una página.

Se debe definir una sub-clase de Perm llamada Prueba_Perm, donde otorgamos los privilegios a los usuarios.

```
class Prueba_Perm extends Perm {

# nombre de la clase
  var $classname = "Prueba_Perm";

# arreglo con los privilegios a ser utilizados/controlados
  var $permissions = array(
                          "usuario"    => 1,
                          "autor"      => 2,
                          "editor"     => 4,
                          "supervisor" => 8,
                          "admin"      => 16
                         );


  function perm_invalid($does_have, $must_have) {
    global $perm, $auth, $sess;
    global $_PHPLIB;

# si el usuario no tiene los privilegios requeridos, se mostrará la siguiente página
    include($_PHPLIB["libdir"] . "perminvalid.ihtml");
  }
}
```

Para restringir una página se deben usar las funciones page_open() y page_close(). Se debe indicar el nombre de nuestra subclase Perm como parámetro de la función page_open(). Perm requiere de sess y auth.

```
<?php
 page_open(array("sess" => "Prueba_Session", "auth" => "Prueba_Auth","perm" => "Prueba_Perm"));

# se llama al método check()
$perm->check("admin");

// código entre estas líneas será sólo ejecutado por usuarios con perfil "admin"
```

```
 page_close();
?>
```

Las librerías ADOdb

Introducción.
PHP requiere una librería que esconda las diferencias entre las interfaces de administración de los distintos sistemas de gestión por no contar con funciones de acceso a base de datos estandarizadas. Para ser libres de cambiar de base de datos manteniendo la portabilidad de una aplicación desarrollada es necesario encapsular las diferencias.
ADOdb es un conjunto de librerías de bases de datos para PHP y Python que usa SQL y permite implementar fácilmente una capa de abstracción de base de datos, para realizar las páginas o escribir los *scripts* una única vez y utilizar cualquier sistema gestor de base de datos sin cambiar el código.

Instalación y Conexión a la BD.
A grandes rasgos estos son los pasos que se deben tener en cuenta para la instalación de las librerías ADOdb:

1. Verificar que la versión del lenguaje que se está utilizando es PHP 4.0.4 o superior. ADOdb requiere la versión de PHP 4.0.4 o posterior ya que usa la función str_replace con arreglos.
2. Descargar el archivo comprimido que contiene todas las carpetas con librerías y archivos de configuración de http://adodb.sourceforge.net/#download o alguno de sus espejos.
3. Descomprimir todos los archivos en un directorio accesible por el servidor Web.
4. Modificar el archivo 'directorio_accesible'/adodb/adodb.inc.php para configurar los parámetros de conexión según las necesidades propias. Es decir, se debe asignar a la variable '$dbdriver' el valor que corresponda, por ejemplo 'mssql' o 'mysql'.
5. Por último se deben asignar los valores de las variables '$servidor', '$database', '$usuario' y 'contraseña' según correspondan.

Al ejecutar ADOdb, al menos se cargan dos archivos. El archivo de inicio adodb/adodb.inc.php, que contienen todas las funciones usadas por todas las clases de bases de datos y el código especifico de acceso a una base de datos en particular que se encuentra en adodb/driver/adodb-????.inc.php, donde ???? corresponde a la cadena de identificación del DBMS especificada en 'adodb.inc.php'.
Para conectarse a una base de datos mysql, por ejemplo, en el *script* PHP deben incluirse unas líneas como las siguientes:

```
include('adodb/adodb.inc.php');
$conn = &ADONewConnection('mysql');
```

La función *ADONewConnection*($driver), o *NewADOConnection*($driver) que es un nombre alterno para la misma función, crea un objeto de conexión nuevo y debe invocarse cada vez que se necesite conectar a una BD.

Una conexión puede ser *persistente* o *no persistente*. La ventaja de las conexiones persistentes es que son mas rápidas, debido a que la conexión no es cerrada nunca (aun usando Close()). Las conexiones No persistentes consumen menos recursos reduciendo el riesgo de sobrecargar la base de datos o el servidor Web.
Para conexiones persistentes, se usa *$conn->PConnect()*, o *$conn->Connect()* para conexiones no persistentes. Algunas bases de datos también manejan *NConnect()*, la cual fuerza la creación de una nueva conexión.

Se pueden consultar más detalles acerca de la instalación y los links de descarga en http://php.weblogs.com/

Principales características.

- A diferencia de otras clases PHP de base de datos, que se enfocan únicamente en el enunciado SELECT, ADOdb soporta código para manejar INSERT y UPDATE que son

rápidamente adaptables a múltiples bases de datos.
- Tiene un sistema de metatipos (metatype) para poder determinar cuales tipos como CHAR, TEXT y STRING son equivalentes en diferentes bases de datos.
- Es fácil de portar debido a que todo el código que depende de la base de datos esta en funciones.
- Crea tablas e índices portables con las clases de diccionario de datos datadict.
- Monitorea el rendimiento de base de datos y ajuste de enunciados SQL con la clase de performance monitoring.
- Administra sesiones en base de datos con la clase session management. Maneja notificaciones por sesión vencida. Aunque en este seminario no la empleé porque el manejo de sesiones se hizo a través de PHPLib.

Variables de Entorno y Globales.

Conocer el aporte de estas variables es fundamental para poder asignarles valor de manera conciente para obtener el máximo provecho en la ejecución de nuestro código. Ellas son:
- *$ADODB_COUNTRECS*

Si el driver de la BD que usamos no soporta el conteo de filas devueltas tras la ejecución de una sentencia select, ADO emula la función RecordCount() si esta variable es 'true', el cual es su valor por defecto. Esta emulación se logra haciendo un buffering de las filas resultantes por lo que tiene como desventaja un rendimiento disminuido para recordsets muy grandes, por el mayor uso de memoria.
- *$ADODB_CACHE_DIR*

Si se desea usar recordset caché, esta variable contiene el nombre del directorio donde almacenar recordsets. Se debe definir antes de la llamada a cualquier función de Catching. Por seguridad es conveniente combinar con register_globals=off en php.ini.
- *$ADODB_LANG*

Es el lenguaje que se usará en MetaErrorMsg(). Su valor por defecto es 'en', inglés.
- *$ADODB_ANSI_PADDING_OFF*

Determina un trim derecho a los campos de tipo CHAR si su valor es 'true'.
- *$ADODB_FETCH_MODE*

Esta variable global define en qué manera se arman los arrays devueltos a partir de las filas resultantes, cualquier cambio subsiguiente a *$ADODB_FETCH_MODE* no tiene efecto en los recordsets existentes, sólo en lo que se creen en el futuro.
Las constantes usadas como patrón son:
- ✓ `define('ADODB_FETCH_DEFAULT',0);`
- ✓ `define('ADODB_FETCH_NUM',1);`
- ✓ `define('ADODB_FETCH_ASSOC',2);`
- ✓ `define('ADODB_FETCH_BOTH',3);`

Un ejemplo ilustrativo:

```
$ADODB_FETCH_MODE = ADODB_FETCH_NUM;
$rs1 = $db->Execute('select * from table');
$ADODB_FETCH_MODE = ADODB_FETCH_ASSOC;
$rs2 = $db->Execute('select * from table');
print_r($rs1->fields);# muestra array([0]=>'v0',[1] =>'v1')
print_r($rs2->fields);# muestra array(['col1']=>'v0',['col2'] =>'v1')
```

Si el modo fetch no es definido, el valor por defecto es ADODB_FETCH_DEFAULT y el comportamiento de este modo variará de uno a otro driver. En beneficio de la portabilidad es aconsejable decidirse por ADODB_FETCH_NUM o ADODB_FETCH_ASSOC. Muchos drivers no soportan ADODB_FETCH_BOTH.

Funciones más utilizadas.
Es muy pretencioso intentar listar una selección de las funciones más utilizadas de ADOdb, ya que su uso es extensamente difundido en la actualidad y no es posible conocer todo el volumen de aplicaciones que recuren a estas librerías. Mucho más factible resulta describir brevemente aquellas que forman parte del sistema desarrollado para este seminario y alguna

otra que se destaca por su practicidad en aplicaciones Web.

Clase ADOConnection

- *Connect($host,[$user],[$password],[$database])*

Conexión no persistente a la fuente de datos en el servidor *$host*, con usuario *$user* y clave *$password*. Si el servidor soporta BD múltiples, a la DB *$database*.
Devuelve ► true/false según si el intento de conexión es éxito o fracaso.

- *PConnect($host,[$user],[$password],[$database])*

Conexión persistente a la fuente de datos.

- *NConnect($host,[$user],[$password],[$database])*

Fuerza a una nueva conexión.

- *IsConnected()*

Devuelve ► true si está conectado.

- *Execute($sql,$inputarr=false)*

Ejecuta la sentencia *$sql*, El parámetro *$inputarr* puede ser usado para ligar variables a parámetros. Es de notar que siempre que la ejecución de la sentencia sea exitosa, se devuelve un arreglo, sea ésta un *insert* o un *update*.
Devuelve ► Una clase derivada de ADORecordSet. Por ejemplo: se devuelve ADORecordSet_mysql si la conexión se hizo vía mysql. Devuelve ► false si hubo un error.

- *SelectLimit($sql,$numrows=-1,$offset=-1,$inputarr=false)*

Ejecuta un select, simulando el SELECT de PostgreSQL, con las cláusulas LIMIT *$numrows* OFFSET *$offset*.
Devuelve ► un registro si tuvo éxito. Si no, devuelve ► false.

- *Close( )*

Cierra la conexión a la BD.

- *ErrorMsg()*

Devuelve el último estado o mensaje de error. El mensaje de error es reseteado cuando se invoca un Execute().

- *InParameter($stmt, $var, $name, $maxLen = 4000, $type = false)*

Enlaza una variable PHP como entrada a una variable del procedimiento almacenado. El parametro *$stmt* es el valor que regreso PrepareSP(), *$var* es la variable PHP a ligar, *$name* es el nombre de la variable del procedimiento almacenado. *$maxLen* es opcional y es la longitud máxima de los datos a ligar y *$type* depende de cada base de datos.
InParameter() es una función envolvente que llama Parameter() con $isOutput=false.

- *OutParameter($stmt, $var, $name, $maxLen = 4000, $type = false)*

Idem a la anterior pero referida a la salida y llamando a Parameter() con $isOutput=trae.

- *Replace($table, $arrFields, $keyCols,$autoQuote=false)*

Intenta hacer un UPDATE al registro, si no encuentra el registro, se genera y ejecuta un INSERT. Es diferente al replace de MySQL el cual borra el registro e inserta uno nuevo. Esto también significa que no se puede actualizar la llave primaria. La única excepción es con Interbase y sus derivados, que si usan delete e insert debido a algunas limitantes del API de Interbase.
Los parámetros son el nombre de la tabla ($table), $arrFields que es un arreglo asociativo donde las llaves son los nombres de los campos y $keyCols es el nombre del campo llave primaria o un arreglo de nombres de campos si es una llave compuesta. Si $autoQuote tiene el valor true, entonces Replace() encerrara entre comillas todos los valor que no son numéricos, los NULLs no se encomillan. Observe que este encomillado automático no funcionara si se usan funciones SQL u operadores.
Devuelve ► 0 si falla, 1 si efectuó el update y 2 si no se encontró el registro y el insert fue con éxito.

- *Affected_Rows( )*

Regresa el número de renglones afectados por un enunciado update o delete.
Devuelve ► falso si la función no esta soportada. Actualmente no esta soportada para interbase/firebird.

Clase ADORecordSet

Cuando se ejecuta satisfactoriamente un enunciado SQL con el método ADOConnection->Execute($sql), se obtiene un objeto ADORecordSet. Este objeto contiene: un cursor virtual

para podernos mover de renglón en renglón, funciones para obtener información acerca de las columnas y sus tipos de datos, y funciones auxiliares para el formateo de los resultados para ser mostrados al usuario.
Sus campos o propiedades son:

- ✓ fields: Arreglo que contiene el renglón actual. No es asociativo, sino un arreglo indexado del 0 al (columnas - 1). Vea también la función Fields, que se comprota como un arreglo asociativo.
- ✓ dataProvider: El mecanismo subyacente empleado para conectarse a la base de datos. Normalmente vale native, a menos que sea odbc o ado.
- ✓ blobSize: Tamaño máximo de un objeto char, string o varchar antes de que se considere como un Blob (Los Blobs se deberán mostrar en textareas). Vea la función MetaType function.
- ✓ sql: Contiene el enunciado sql empleado para generar este conjunto de datos.
- ✓ canSeek: Con valor verdadero si la función Move( ) funciona..
- ✓ EOF: Verdadero si se ha navegado el cursor después del ultimo registro.

- *GetAssoc([$force_array])*

Genera un arreglo asociativo del recordset. Observe que esta función también esta disponible en el objeto de conexión. Mas detalles se pueden encontrar ahí.

- *GetArray([$number_of_rows])*

Genera un arreglo bidimensional de registros desde la posición actual del cursor, indexado desde 0 a ($number_of_rows - 1). Si no se define $number_of_rows, se indexa hasta el final de recordset (EOF).

- *Fields($colname)*

Regresa el valor de la columna *$colname*. Al nombre de la columna no le afectan las mayúsculas. Esta función solo esta por comodidad. Para un mejor rendimiento use $ADODB_FETCH_MODE.

- *FetchRow()*

Regresa un arreglo conteniendo el renglón actual, o falso si es EOF. Internamente FetchRow() se mueve al siguiente renglón después de regresar el renglón actual.

- *FetchField($column_number)*

Regresa un objeto conteniendo name, type y max_length del campo solicitado. Si max_length no se pueden determinar con confianza, tendrá el valor de -1. El numero de columna esta en base a cero (la primer columna es 0).

- *FieldCount( )*

Regresa el número de campos (columnas) en el recordset.

- *RecordCount( )*

Regresa el número de renglones en el recordset. Si el numero de registros obtenidos no se puede determinar del API de la base de datos, se leerán todos los renglones para poderse contar. Esta lectura puede ser deshabilitada (por rendimiento) asignándole la variable $ADODB_COUNTRECS = false. Cuando esta deshabilitada, RecordCount( ) regresara -1 en algunas bases de datos. Vea arriba la lista de bases de datos soportadas para más detalle.

Bases de Datos soportadas.
En la tabla que se presenta ahora se resume claramente la información sobre las BD soportadas más conocidas. Para su mejor interpretación deben tenerse en cuenta las referencias que siguen:

- La columna "sirve RecordCount()" indica si RecordCount() regresa la cantidad de renglones o regresa -1 cuando se ejecuta un enunciado SELECT. Si la columna muestra S/N entonces RecordCount() es simulado cuando la variable global $ADODB_COUNTRECS=true (esto es el valor por omisión). Nota que para recordsets muy grandes, puede ser mejor deshabilitar la emulación de RecordCount()debido a la gran cantidad de memoria para leer el recordset para contarlo.

Nombre	Base de datos	RecordCount()	Pre-requisitos	SO
access	Microsoft Access/Jet. Hay que crear el DSN en el ODBC.	S/N	ODBC	Windows

ado	ADO genérico, no esta optimizado para ninguna base de datos especifica. Permite conexiones DSN-less. Para un mejor rendimiento, utilizar un proveedor de OLEDB. Esta la clase basa para todos los drivers ado. Hay que configurar $db->codePage antes de conectarse.	? depende de la base de datos	ADO o proveedor OLEDB	Windows
ado_access	Microsoft Access/Jet usando ADO. Permite conexiones DSN-less. Para mejor rendimiento usar un proveedor OLEDB.	S/N	ADO o proveedor OLEDB	Windows
ado_mssql	Microsoft SQL Server usando ADO. Permite conexiones DSN-less. Para mejor rendimiento usar un proveedor OLEDB.	S/N	ADO o proveedor OLEDB	Windows
db2	DB2. Debe de funcionar satisfactoriamente porque se base en el driver ODBC based on ODBC.	S/N	Interfase CLI/ODBC de DB2	Unix y Windows
vfp	Microsoft Visual FoxPro. Hay que crear un DSN.	S/N	ODBC	Windows
fbsql	FrontBase.	S	?	Unix y Windows
ibase	Interbase 6 o anterior. Algunos usuarios dicen que puedes necesitar esta sintaxis para conectarte $db->PConnect('localhost:c:/ibase/employee.gdb', "sysdba", "masterkey") . Actualmente le falta Affected_Rows. Antes de conectarte puedes modificarle $db->dialect, $db->buffers y $db->charSet.	S/N	cliente Interbase	Unix y Windows
firebird	Versión Firebird de interbase.	S/N	cliente Interbase	Unix y Windows
borland_ibase	Versión Borland de Interbase 6.5 o posterior. Desafortunadamente las version son diferentes.	S/N	cliente Interbase	Unix y Windows
informix72	Versiones Informix anteriores a 7.3 que no soportan SELECT FIRST.	S/N	Cliente Informix	Unix y Windows
informix	Driver genérico para informix.	S/N	Cliente Informix	Unix y Windows
ldap	Driver LDAP driver. Ver el ejemplo para información de uso.		Extensión LDAP	?
mssql	Microsoft SQL Server 7 y posterior. También funciona con Microsoft SQL Server 2000. Toma en cuenta que el formato de las fechas es problemático con este driver. Por ejemplo, la extensión mssql de PHP no regresa los segundos de los campos datetime!	S/N	cliente Mssql	Unix y Windows. Guía de instalación de Unix y otra mas.
mssqlpo	Driver portable de mssql. Es idéntico al driver de mssql anterior excepto que el operador de concatenación '||' se convirtió a '+'. Esto es útil para migrar código desde otras versiones de SQL.	S/N	Cliente Mssql	Unix y Windows.
mysql	MySQL sin manejo de transacciones. Puedes usar $db->clientFlags antes de conectarte.	S/N	Cliente MySQL	Unix y Windows
mysqlt or **maxsql**	MySQL con soporte de transacciones. Recomendamos usar || como el operador de concatenación para una mejor portabilidad. Esto se logra ejecutando MySQL con: *mysqld --ansi* o *mysqld --sql-mode=PIPES_AS_CONCAT*	S/N	Cliente MySQL	Unix y Windows

oci8	Oracle 8/9. Tiene mas funcionalidad que el driver *oracle* (ej. Affected_Rows). Puedes tener que hacer un putenv('ORACLE_HOME=...') antes del Connect/PConnect. Hay dos maneras de conectarse - Con la dirección IP del servidor y el nombre del servicio: PConnect('serverip:1521','scott','tiger','service') o en base a un renglón en TNSNAMES.ORA o ONAMES o HOSTNAMES: PConnect(false, 'scott', 'tiger', $oraname).	S/N	Cliente Oracle	Unix y Windows
oci805	Maneja una funcionalidad reducida para la version 8.0.5 de Oracle. SelectLimit no es tan eficiente como en los drivers oci8 u oci8po.	S/N	Cliente Oracle	Unix y Windows
oci8po	Driver portable de Oracle 8/9. Esta es casi idéntica al driver oci8 excepto que (a) Las variables 'bind' en los Prepare() usan ? en lugar de :bindvar, (b) los nombres de campos están en minúsculas (la manera mas usual en PHP). Usa este driver si la portabilidad de tu código a otras bases de datos es importante. En caso contrario usa el driver oci8 ya que da mejor rendimiento.	S/N	Cliente Oracle	Unix y Windows
odbc	Driver genérico para ODBC, no esta optimizado para ninguna base de datos especifica. Para conectarse usa PConnect('DSN','user','pwd'). Esta es la clase base para todos los drivers basados en ODBC .	? depende de la base de datos	ODBC	Unix y Windows.
odbc_mssql	Usa ODBC para conectarse a MSSQL	S/N	ODBC	Unix y Windows.
odbc_oracle	Usa ODBC para conectarse a Oracle	S/N	ODBC	Unix y Windows.
odbtp	Driver odbtp genérico. Odbtp es un programa para poder usar los DSN en el ODBC de Windows desde OTROS sistemas operativos (ej. Linux).	S/N	odbtp	Unix y Windows
odbtp_unicode	Odtbp con soporte unicode	S/N	odbtp	Unix y Windows
oracle	Implementa el viejo API de Oracle 7. Si te es posible usa el driver oci8 para un mejor rendimiento.	S/N	Cliente Oracle	Unix y Windows
netezza	Driver para Netezza. Netezza esta basado en el código base de postgres.	S	?	?
postgres	Driver genérico PostgreSQL. Actualmente es idéntico al driver postgres7.	S	Cliente PostgreSQL	Unix y Windows.
postgres64	Para PostgreSQL 6.4 y anteriores que no manejan LIMIT internamente.	S	Cliente PostgreSQL	Unix y Windows.
postgres7	PostgreSQL que soporta LIMIT y características de la version 7.	S	Cliente PostgreSQL	Unix y Windows.
sapdb	SAP DB. Debe de funcionar bien ya que esta basado en el driver ODBC.	S/N	Cliente ODBC de SAPdb	?
sqlite	SQLite. Verificada en PHP5.	S	-	Unix y Windows.
sybase	Sybase.	S/N	Cliente Sybase	Unix y Windows.

El motor MS Sql Server

Introducción.
SQL Server 2000 es un sistema de gestión de bases de datos relacionales (SGDBR o RDBMS: Relational Database Management System) diseñado para trabajar con grandes cantidades de información y la capacidad de cumplir con los requerimientos de proceso de información para aplicaciones comerciales y sitios Web.

SQL Server 2000 ofrece el soporte de información para aplicaciones tradicionales Cliente/Servidor, las cuales están conformadas por una interfaz a través de la cual los clientes acceden a los datos por medio de una LAN. La emergente plataforma NET exige un gran porcentaje de distribución de recursos, desconexión a los servidores de datos y un entorno descentralizado, para ello sus clientes deben ser livianos, tales como los navegadores de Internet los cuales accederán a los datos por medio de servicios como el Internet Information Services(IIS).

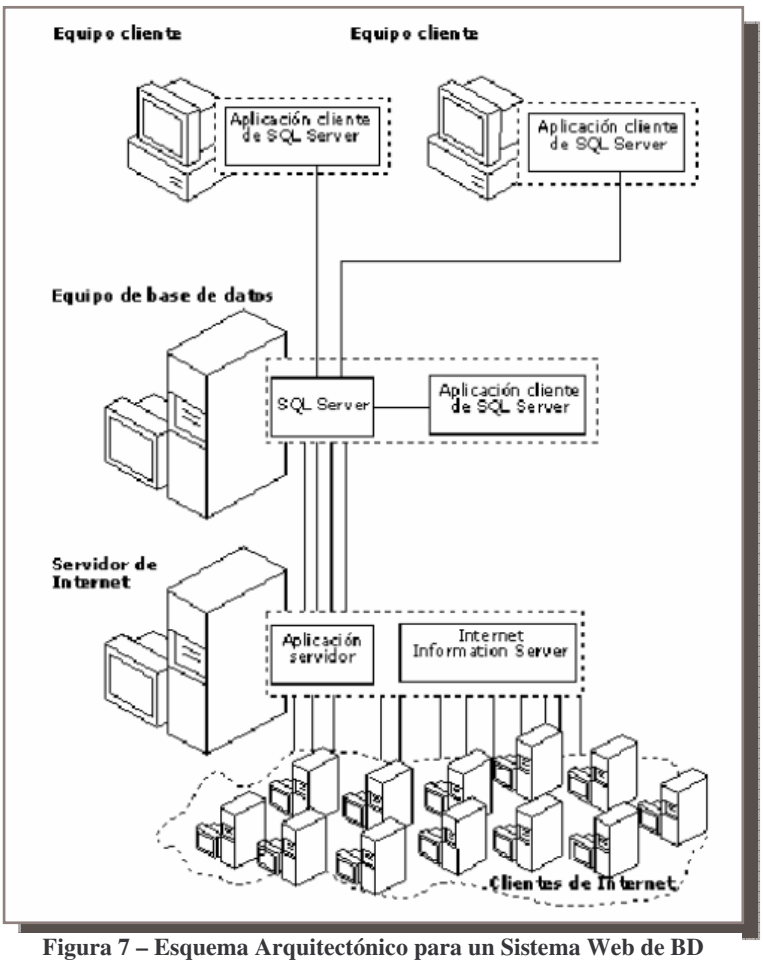


Figura 7 – Esquema Arquitectónico para un Sistema Web de BD

SQL Server 2000 está diseñado para trabajar con dos tipos de bases de datos:
• OLTP (OnLine Transaction Processing) Son bases de datos caracterizadas por mantener una gran cantidad de usuarios conectados concurrentemente realizando ingreso y/o modificación de datos. Por ejemplo : entrada de pedidos en línea, inventario, contabilidad o facturación.
• OLAP (OnLine Analytical Processing) Son bases de datos que almacenan grandes cantidades de datos que sirven para la toma de decisiones, como por ejemplo las aplicaciones de análisis de ventas.

SQL Server puede ejecutarse sobre redes basadas en Windows Server así como sistema de base de datos de escritorio en máquinas Windows NT Workstation, Windows Millenium y Windows 98. Los entornos Cliente/Servidor, están implementados de tal forma que la información se guarde de forma centralizada en un computador central (servidor), siendo el

servidor responsable del mantenimiento de la relación entre los datos, asegurarse del correcto almacenamiento de los datos, establecer restricciones que controlen la integridad de datos, etc.

Del lado cliente, este corre típicamente en distintas computadoras las cuales acceden al servidor a través de una aplicación. Para realizar la solicitud de datos los clientes emplean el Structured Query Language (SQL), este lenguaje tiene un conjunto de comandos que permiten especificar la información que se desea recuperar o modificar.

Existen muchas formas de organizar la información pero una de las formas más efectivas de hacerlo está representada por las bases de datos relacionales, en las que los datos están organizados en tablas (llamadas relaciones en la teoría relacional).

Al organizar los datos en tablas, se pueden encontrar varias formas de definirlas. La teoría de las bases de datos relacionales define un proceso, la normalización, que asegura que el conjunto de tablas definido organizará los datos de manera eficaz.

SQL Server incluye un conjunto de herramientas que facilitan la instalación y administración del servidor así como un conjunto de herramientas que facilitan el diseño e implementación de base de datos, entre ellos podemos mencionar:
• SQL Server 2000 Database Engine, diseñado para almacenar detalladamente los registros de las operaciones transaccionales (OLTP), este motor es responsable de mantener la seguridad de los datos, proveer un adecuado nivel de tolerancia a fallos, optimizar las consultas, emplear adecuadamente los bloqueos de recursos para optimizar la concurrencia, etc.
• SQL Server 2000 Analysis Services, provee herramientas para consultar información almacenada en data warehouses y data marts, como por ejemplo cuando se desea obtener información totalizada acerca de los niveles de ventas mensuales por regiones de ventas, etc.
Soporte para aplicaciones, SQL Server brinda a las aplicaciones clientes la posibilidad de acceder a los datos a través de un lenguaje denominado Transact-SQL, asimismo es importante mencionar que ahora existe un soporte para devolver la información en formato XML.

Como soporte para las aplicaciones clientes tenemos:
1. *SQL Distributed Management Objects* (SQL-DMO) API que brinda un conjunto de objetos COM que encapsulan toda la funcionalidad administrativa del motor de datos.
2. *Decision Support Objects* (DSO) API que brinda un conjunto de objetos COM que encapsulan las funcionalidades de los SQL Server 2000 Analysis Services.
3. *Windows Management Instrumentation* (WMI), es una API orientada a objetos que permite administrar aplicaciones scripts para monitorear, configurar y controlar los servicios, recursos y aplicaciones de Windows. SQL Server ofrece una API que devuelve la información del motor de datos y de todas sus instancias, esta API se denomina SQL Server 2000 WMI.

Entre los componentes adicionales de SQL Server 2000, podemos mencionar:
• *Data Transformation Services*, permite recuperar información de un origen de datos, realizar transformaciones sencillas o complejas (como totalización de datos) y almacenarlos en otro origen de datos, como una base de datos SQL o un cubo multidimensional.
Replicación, se puede distribuir la información a través de un mecanismo de replicación con la finalidad de optimizar el rendimiento o de mantener autonomía, mientras una copia de la información almacenada en diferentes computadoras mantengan sincronización.
• *English Query*, provee de un sistema que permite a los usuarios plantear una pregunta en lenguaje natural en lugar de emplear un formato Transact-SQL. Por ejemplo: “List all customers”, “How many blue dress were sold in 2001?”, etc.
• *Meta Data Services*, son un conjunto de servicios que permiten almacenar información acerca de las bases de datos y aplicaciones clientes que las emplean, esta información es aprovechada cuando se requiere intercambiar con otras aplicaciones. Los Meta Data Services proveen tres estándares: Meta Data Coalition Open Information Model (MDC OIM), Interfaces COM y XML Encoding.

Además de ello cuenta con la documentación apropiada para poder obtener información detallada de cada uno de los tópicos de SQL Server.

Componentes de SQL Server 2000.

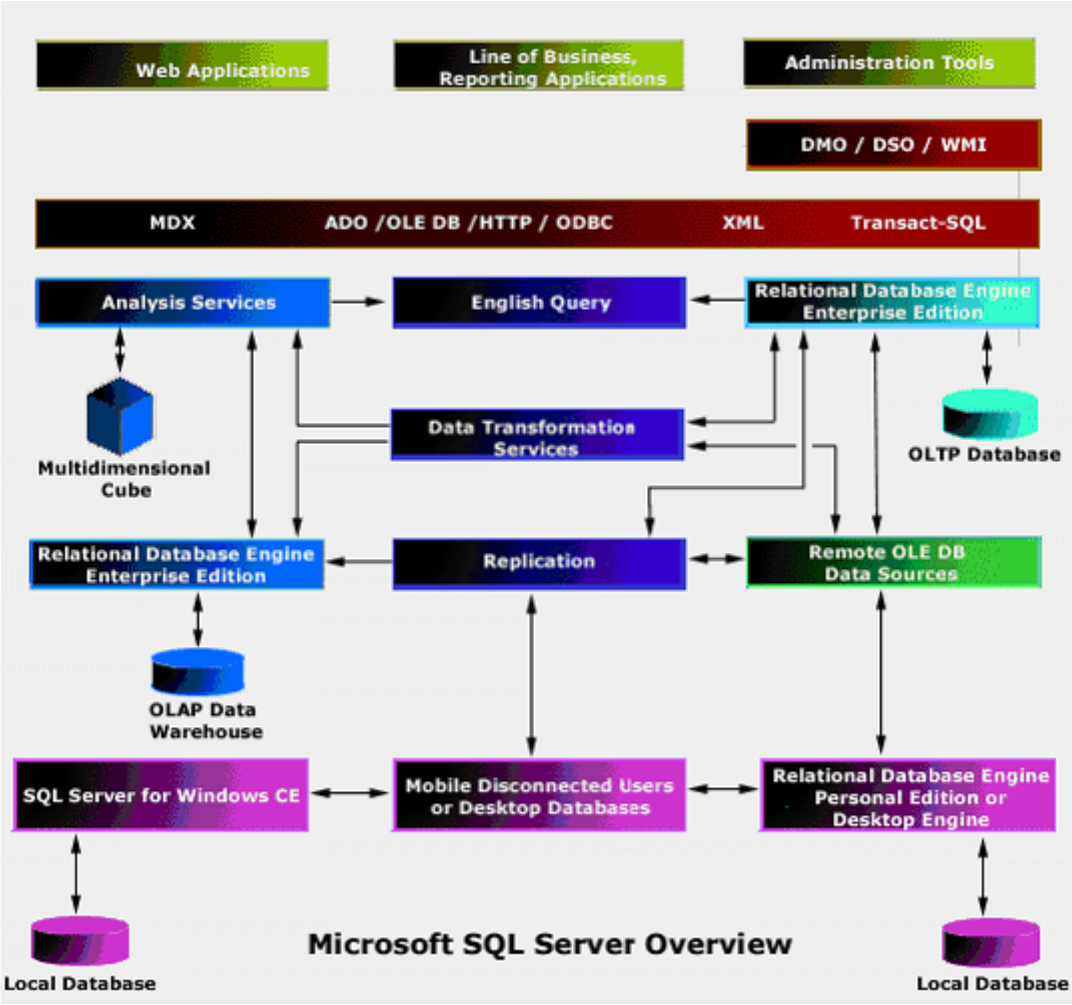


Figura 8 – Componentes de SQL Server

Ediciones SQL Server 2000.

SQL Server 2000 está disponible en seis diferentes versiones además en cualquier edición se incluye el SQL Server 2000 Desktop Engine:

• Enterprise, soporta todas las características de SQL Server 2000. Esta edición es para empresas que implementan medianas y grandes bases de datos, las cuales brindan recursos a soluciones Web, organizaciones con un alto índice de trabajo transaccional, y soporte para *data warehouse*[3].

• Estándar, ideal para aplicaciones que necesiten brindar información a grupos de trabajos o departamentos dentro de una organización.

Entre las características más salientes que no se encuentran disponibles para el motor relacional, podemos mencionar:

- ✓ Clustering
- ✓ Log Shipping
- ✓ Vistas indexadas

Entre las características más destacadas que no se encuentran disponibles para los servicios de análisis:

- ✓ Definición de cubos particionados

[3] Data Warehouse o Almacén de Datos, es una colección de datos orientadas a un dominio, integrado, no volátil y que varía en el tiempo que ayuda a la toma de decisiones de la empresa u organización.

- ✓ Cubos OLAP enlazados
- ✓ Soporte para dimensiones ROLAP
- ✓ Celdas calculadas

Personal, soporta todas las características del SQL Server 2000 Standard Edition, excepto la replicación transaccional, para lo cual sólo puede ser definido como un suscriptor, además de esto tampoco se encuentra disponible el full text search cuando se instala sobre Windows Me y Windows 98. Esta edición puede ser empleada para aplicaciones standalone y usuarios móviles que requieran un almacenamiento local de información.

• Windows CE Edition, es empleado para almacenar información en dispositivos Windows CE. SQL Server 2000 CE es implementado como un conjunto de librerías (DLLs) que operan como un OLE DB CE Provider. Está implementación permite que SQL Server 2000 CE soportar ActiveX Data Objects for Windows CE (ADOCE) y OLE DB CE APIs en Windows CE versiones disponibles para Visual Basic y Visual C++. Además también es posible que múltiples aplicaciones puedan compartir al mismo tiempo un conjunto de DLLs.

Los dispositivos Windows CE pueden conectarse a la red empleando Remote Data Access (RDA) característica de SQL Server CE para:

- ✓ Conectarse a instancias de SQL Server de diferentes plataformas
- ✓ Ejecutar sentencias SQL y colocarlas en un recordset
- ✓ Modificar la información de un recordset y enviarlas a una instancia de SQL Server inclusive de diferentes plataformas.
- ✓ Ser suscriptor en una replicación de tipo merge.

• Developer Edition, soporta todas las características de SQL Server 2000, además de un conjunto de herramientas gráficas para la configuración de idiomas, esta es una edición sólo para desarrolladores que emplean SQL Server como su origen de datos. Esta edición sólo esta licenciada para desarrollo y prueba de los sistemas.

• Enterprise Evaluation Edition, soporta todas las características de SQL Server 2000, a excepción de las herramientas gráficas para configuración del lenguaje.

Esta edición es libre y se puede descargar desde el Web aunque sólo podrá ejecutarla por 120 días.

• SQL Server 2000 Desktop Engine, es una versión distribuible del motor de base de datos relacional de SQL Server 2000. Esta edición es empleada para aquellas aplicaciones que no requieran la implementación de tareas administrativas para el cliente. Debe recordar que las bases de datos no deben exceder los 2 Gb. de tamaño.

Instalación de SQL Server.

Antes de instalar SQL Server 2000 es necesario conocer cuales son los requisitos mínimos para instalar este producto, el siguiente cuadro muestra los requerimientos para instalar SQL Server de acuerdo a la edición que Ud. emplee:

Recurso	Requerimiento
Computador	Intel o compatible
Procesador	Pentium 166
Monitor	800*600
Dispositivo puntero	Mouse
Tarjeta de red	Opcional (requerido para acceso a los recursos de la red)
CD-ROM	Requerido para la instalación

Para determinar correctamente el requerimiento de memoria, emplear la siguiente tabla:

	Enterprise	Estándar	Evaluation	Developer	Personal y Desktop Engine
Alguna edición de Windows 2000 Server	256 MB (128 MB soportado)	256 MB (128 MB soportado)	256 MB (128 MB soportado)	256 MB (128 MB soportado)	256 MB (128 MB soportado)
Alguna edición de Windows NT 4.0 Server con SP5 o posterior	128 MB (64 MB soportado)	64 MB	128 MB recomendado (64 MB soportado)	64 MB	32 MB
Windows 2000 Professional	N/A	N/A	128 MB recomendado (64 MB soportado)	64 MB	64 MB
Windows NT 4.0 Workstation, con SP5 o posterior	N/A	N/A	128 MB recomendado (64 MB soportado)	64 MB	32 MB
Windows ME	N/A	N/A	N/A	N/A	32 MB
Windows 98	N/A	N/A	N/A	N/A	32 MB

Hay que tener en cuenta que para instalar SQL Server 2000 se requiere de
Internet Explorer 5.0 o posterior, si desea instalar SQL Server 2000 sobre Windows NT en cualquiera de sus ediciones debe instalar previamente el Service Pack 5.0 o posterior.
Asimismo, para poder determinar el espacio en disco requerido para su instalación es conveniente analizar la siguiente tabla:

Opción de Instalación seleccionada	Espacio en disco requerido
Server y client tools	95-270 MB dependiendo de las opciones seleccionadas
Instalación Typical	250 MB (178 MB para el sistema, más 72 MB para programas y archivos de datos)
Instalación mínima	110 MB (73 MB para el sistema, más 37 MB para programas y archivos de datos)
Herramientas administrativas	113 MB (sistema solamente)
BoAceptars Online	30 MB (sistema solamente)
Analysis Services	47 MB mínimo 120 MB typical
English Query	80 MB
Sólo Desktop Engine	44 MB

A título informativo, estas son las consideraciones previas que deben realizarse antes de instalar el paquete del motor de base de datos. No enumeraré los pasos a seguir en la instalación de manera detallada porque es bastante intuitivo y con especificar unos cuantos parámetros se consigue una buena instalación de este software.
Una vez finalizada la instalación debe revisarla para cerciorarse que el producto se ha instalado correctamente para ello puede mostrar el Administrador de Servicios (Service Manager) que le permitirá mostrar el estado de los servicios, este utilitario tiene el siguiente aspecto:

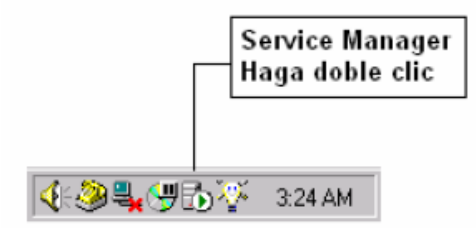


Figura 9 – Monitor del Servicio de SQL Server

Seguidamente observará una caja de diálogo con el siguiente aspecto:

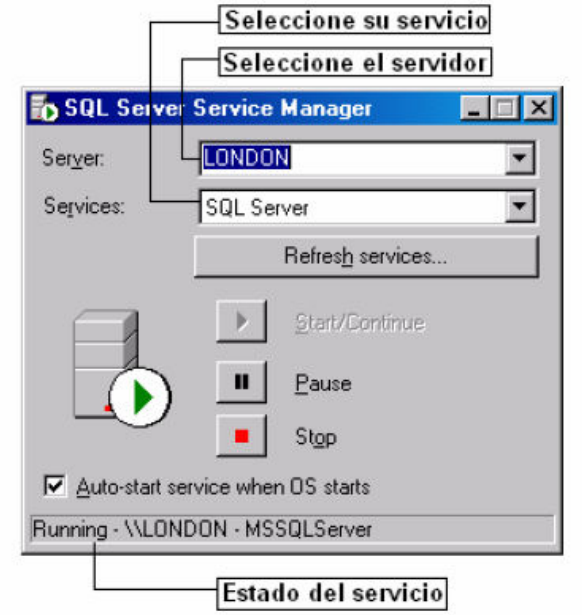


Figura 10 – Administrador del Servicio

Bases de Datos de SQL Server.

SQL Server soporta bases de datos del sistema y bases de datos del usuario.

Las bases de datos del sistema, almacenan información que permite operar y administrar el sistema, mientras que las de usuario almacenan los datos requeridos por las operaciones del cliente.

Las bases de datos del sistema son:

• master

La base de datos master se compone de las tablas de sistema que realizan el seguimiento de la instalación del servidor y de todas las bases de datos que se creen posteriormente. Asimismo controla las asignaciones de archivos, los parámetros de configuración que afectan al sistema, las cuentas de inicio de sesión. Esta base de datos es crítica para el sistema, así que es bueno tener siempre una copia de seguridad actualizada.

• tempdb

Es una base de datos temporal, fundamentalmente un espacio de trabajo, es diferente a las demás bases de datos, puesto que se regenera cada vez que arranca SQL Server. Se emplea para las tablas temporales creadas explícitamente por los usuarios, para las tablas de trabajo intermedias de SQL Server durante el procesamiento y la ordenación de las consultas.

• model

Se utiliza como plantilla para todas las bases de datos creadas en un sistema. Cuando se emite una instrucción CREATE DATABASE, la primera parte de la base de datos se crea copiando el contenido de la base de datos model, el resto de la nueva base de datos se llena con páginas vacías.

• msdb

Es empleada por el servicio SQL Server Agent para guardar información con respecto a tareas de automatización como por ejemplo copias de seguridad y tareas de duplicación, asimismo solución a problemas. La información contenida en las tablas que contiene esta base de datos,

es fácilmente accedida desde el Administrador Empresarial, así que se debe tener cuidado de modificar esta información directamente a menos que se conozca muy bien lo que se esta haciendo.
• Distribution
Almacena toda la información referente a la distribución de datos basada en un proceso de replicación.

Objetos de una Base de Datos

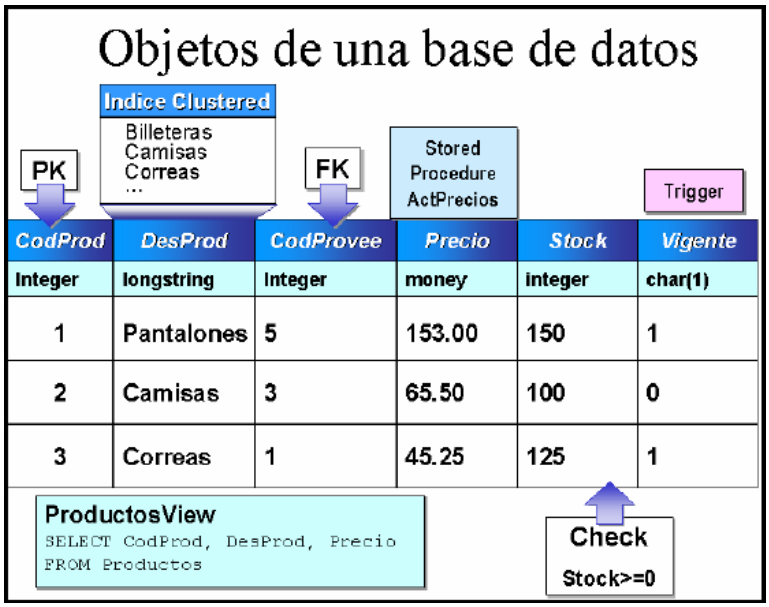


Figura 11 – Objetos de una BD

Las **Tablas** son objetos de la base de datos que contienen la información de los usuarios, estos datos están organizados en filas y columnas, similar al de una hoja de cálculo. Cada columna representa un dato aislado y en bruto que por sí solo no brinda información, por lo tanto estas columnas se deben agrupar y formar una fila para obtener conocimiento acerca del objeto tratado en la tabla. Por ejemplo, puede definir una tabla que contenga los datos de los productos ofertados por una tienda, cada producto estaría representado por una fila mientras que las columnas podrían identificar los detalles como el código del producto, la descripción, el precio, las unidades en stock, etc.
Una **Vista** es un objeto definido por una consulta. Similar a tabla, la vista muestra un conjunto de columnas y filas de datos con un nombre, sin embargo, en la vista no existen datos, estos son obtenidos desde las tablas subyacentes a la consulta. De esta forma si la información cambia en las tablas, estos cambios también serán observados desde la vista. Fundamental emplean para mostrar la información relevante para el usuario y ocultar la complejidad de las consultas.
Los **tipos de datos** especifican que tipo de valores son permitidos en cada una de las columnas que conforman la estructura de la fila. Por ejemplo, si desea almacenar precios de productos en una columna debería especificar que el tipo de datos sea money, si desea almacenar nombres debe escoger un tipo de dato que permita almacenar información de tipo carácter.
SQL Server nos ofrece un conjunto de tipos de datos predefinidos, pero también existe la posibilidad de definir tipos de datos de usuario. Un Procedimiento Almacenado es una serie de instrucciones SQL precompiladas las cuales organizadas lógicamente permiten llevar a cabo una operación transaccional o de control. Un Procedimiento almacenado siempre se ejecuta en el lado del Servidor y no en la máquina Cliente desde la cual se hace el requerimiento. Para ejecutarlos deben ser invocados explícitamente por los usuarios.
Un **Desencadenador** es un Procedimiento Almacenado especial el cual se invoca automáticamente ante una operación de Insert, Update o Delete sobre una tabla. Un Desencadenador puede consultar otras tablas y puede incluir complejas instrucciones

SQL, se emplean para mantener la integridad referencial, preservando las relaciones definidas entre las tablas cuando se ingresa o borra registros de aquellas tablas.
Los **Valores Predeterminados** especifican el valor que SQL Server insertará en una columna cuando el usuario no ingresa un dato específico. Por ejemplo, si se desconoce el apellido materno de un empleado SQL Server podría incluir automáticamente la cadena NN para identificar este campo.
Las **Reglas** son objetos que especifican los valores aceptables que pueden ser ingresados dentro de una columna particular. Las Reglas son asociadas a una columna o a un tipo de dato definido por el usuario. Una columna o un Tipo de dato puede tener solamente una Regla asociada con el.
Las **Restricciones** son restricciones que se asignan a las columnas de una tabla y son controladas automáticamente por SQL Server.
Esto nos provee las siguientes ventajas:
• Se puede asociar múltiples constantes a una columna, y también se puede asociar un constraints a múltiples columnas.
• Se pueden crear los Restricciones al momento de crear la tabla CREATE TABLE.
Los Restricciones conforman el standars ANSI para la creación y alteración de tablas, estos no son extensiones del Transact SQL.
Se puede usar un constraints para forzar la integridad referencial, el cual es el proceso de mantener relaciones definidas entre tablas cuando se ingresa o elimina registros en aquellas tablas.
Los **índices** de SQL Server son similares a los índices de un libro que nos permiten llegar rápidamente a las páginas deseadas sin necesidad de pasar hoja por hoja, de forma similar los índices de una tabla nos permitirán buscar información rápidamente sin necesidad de recorrer registro por registro por toda la tabla. Un índice contiene valores y punteros a las filas donde estos valores se encuentran.

Cómo se almacenan los datos en SQL Server.

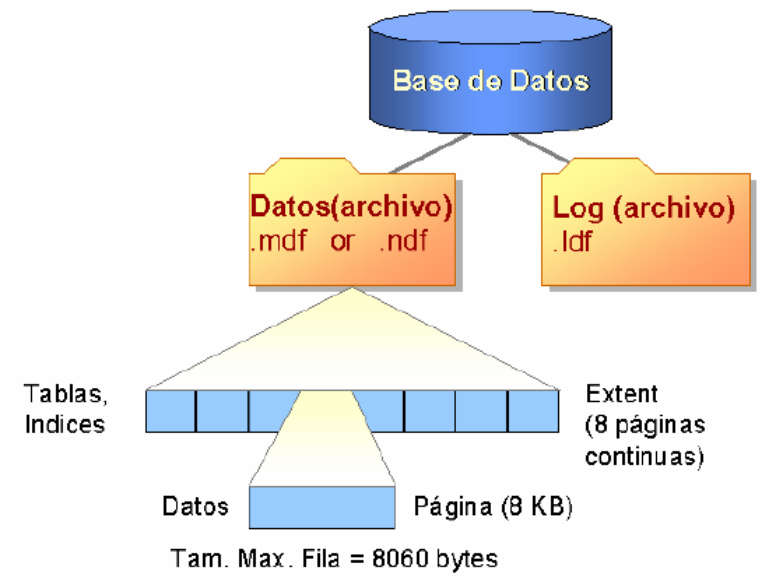


Figura 12 – Almacenamiento de los datos en SQL Server

- **Páginas y extensiones**

Antes de crear una base de datos con SQL Server 2000, debemos tomar en cuenta que la unidad básica de almacenamiento en SQL Server es la página(data page), el tamaño de cada pade es de 8 KB, lo cual representa un total de 128 páginas por cada megabyte.
El comienzo de cada página es una cabecera de 96 bytes que se utiliza para almacenar información de cabecera tal como el tipo de página, la cantidad de espacio libre de la página y el Id. del objeto propietario de la página. Existen ocho tipos de páginas en los archivos de datos de una base de datos SQL Server 2000.

Tipo de página	Contenido
Datos	Filas con todos los datos excepto los de tipo **text**, **ntext** e **image**.
Índice	Entradas de índices
Texto o imagen	Datos de tipo **text**, **ntext** e **image**.
Mapa de asignación global/ Mapa de asignación global Secundario	Información acerca de las extensiones asignadas.
Espacio libre en la página	Información acerca del espacio libre disponible en las páginas.
Mapa de asignación de índices.	Información acerca de las extensiones utilizadas por una tabla o un índice
Bulk Changed Map	Información de los extends modificados por operación bulk desde el último backup del log.
Differential Changed Map	Información de los extends modificados desde el último full database backup.

Los archivos de registro (LOG) no contienen páginas, contienen series de registros.
Las páginas de datos contienen todos los datos de las filas de datos excepto los datos text, ntext e image, que están almacenados en páginas separadas. Las filas de datos se colocan en las páginas una a continuación de otra, empezando inmediatamente después de la cabecera, al final de cada página se encuentra una tabla de posiciones de filas que contiene una entrada por cada fila de la página y cada entrada registra la posición, desde el principio de la página, del primer byte de la fila. Las entradas de la tabla de posiciones de filas están en orden inverso a la secuencia de las filas de la página.

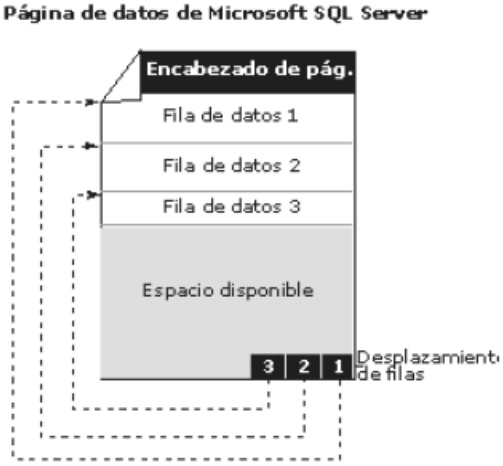
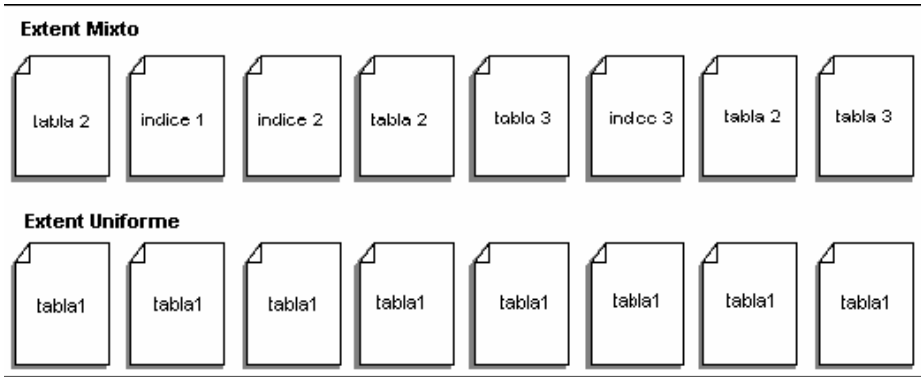


Figura 13 – Página de almacenamiento de datos

En SQL Server, las filas no pueden continuar en otras páginas.
Las extensiones son la unidad básica de asignación de espacio a las tablas e índices.
Consta de 8 páginas contiguas, es decir 64 KB. Lo cual representa 16 extensiones por MB.
Para hacer que la asignación de espacio sea eficiente, SQL Server 2000 no asigna extensiones enteras a tablas con poca cantidad de datos. SQL Server 2000 tiene dos tipos de extensiones:
• Las extensiones uniformes son propiedad de un único objeto; sólo el objeto propietario puede utilizar las ocho páginas de la extensión.
• Extensiones mixtas, pueden estar compartidas por hasta ocho objetos.



Las tablas o índices nuevos son asignados a páginas de extensiones mixtas. Cuando la tabla o el índice crecen hasta el punto de ocupar ocho páginas, se pasan a extensiones uniformes.

Archivos y grupos de archivos físicos de la base de datos

Un archivo de base de datos no es más que un archivo del sistema operativo. Una base de datos se distribuye en por lo menos dos archivos, aunque es muy probable que sean varios los archivos de base de datos que se especifican al crear o al modificar una base de datos.

Principalmente SQL Server divide su trabajo en un archivo para datos y otro para el registro de las transacciones (log).

SQL Server 2000 permite los tres siguientes tipos de archivos:

- Archivos de datos primarios

Toda base de datos tiene un archivo de datos primario que realiza el seguimiento de todos los demás archivos, además de almacenar datos. Por convenio este archivo tiene la extensión MDF.

- Archivos de datos secundarios

Una base de datos puede tener cero o varios archivos de datos secundarios. Por convenio la extensión recomendada para los archivos de datos secundarios es NDF.

- Archivos de registro (LOG)

1. Todas las bases de datos por lo menos tendrán un archivo de registro que contiene la información necesaria para recuperar todas las transacciones que suceden sobre la misma. Por convenio la extensión de este archivo es LDF.
2. Por lo tanto al crear una base de datos, debemos considerar los siguientes premisas y reglas para el almacenamiento de los datos:
3. Todas las Bases de Datos tienen un archivo de base de datos primario (.mdf) y uno para el Log de Transacciones (.ldf). Además puede tener archivos de datos secundarios (.ndf).
4. Cuando se crea una Base de Datos, una copia de la Base de Datos Model, la cual incluye tablas del sistema, es copiada en la Nueva Base de Datos.
5. La Data es almacenada en bloques de 8-kilobytes (KB) de espacio de disco contiguo llamado páginas.
6. Las filas o registros no pueden atravesar páginas. Esto, es, que la máxima cantidad de datos en una fila de datos simple es de 8060 bytes.
7. Las tablas y los índices son almacenados en Extents. Un Extents consta de ocho páginas contiguas, o sea 64 KB.
8. El Log de Transacciones lleva toda la información necesaria para la recuperación de la Base de Datos en una eventual caída del sistema. Por default, el tamaño del Log de Transacciones es del 25% del tamaño de los archivos de datos. Use esta configuración como punto de partida y ajuste de acuerdo a las necesidades de su aplicación.

- Archivos de Registro (LOG de Transacciones)

El LOG de transacciones archiva todas las modificaciones de los datos tal cual son ejecutados. El proceso es como sigue:

1. Una modificación de datos es enviada por la aplicación cliente.
2. Cuando una modificación es ejecutada, las páginas afectadas son leídas del disco a memoria (Buffer Cache), provista de las páginas que no están todavía en la Data Cache del query previo.
3. Cada comando de modificación de datos es archivado en el LOG. El cambio

siempre es archivado en el LOG y es escrito en el disco antes que el cambio sea hecho en la Base de Datos. Este tipo de LOG es llamado LOG de tipo write-ahead.
4. Una vez que las páginas de datos residen en el Buffer Cache, y las páginas de LOG son archivadas sobre el disco en el archivo del LOG, el proceso de CHECKPOINT, escribe todas las transacciones completas a la Base de Datos en el disco.
Si el sistema falla, automáticamente el proceso de recuperación usa el LOG de Transacciones para llevar hacia delante todas las transacciones comprometidas (COMMIT) y llevar hacia atrás alguna transacción incompleta (ROLLBACK).

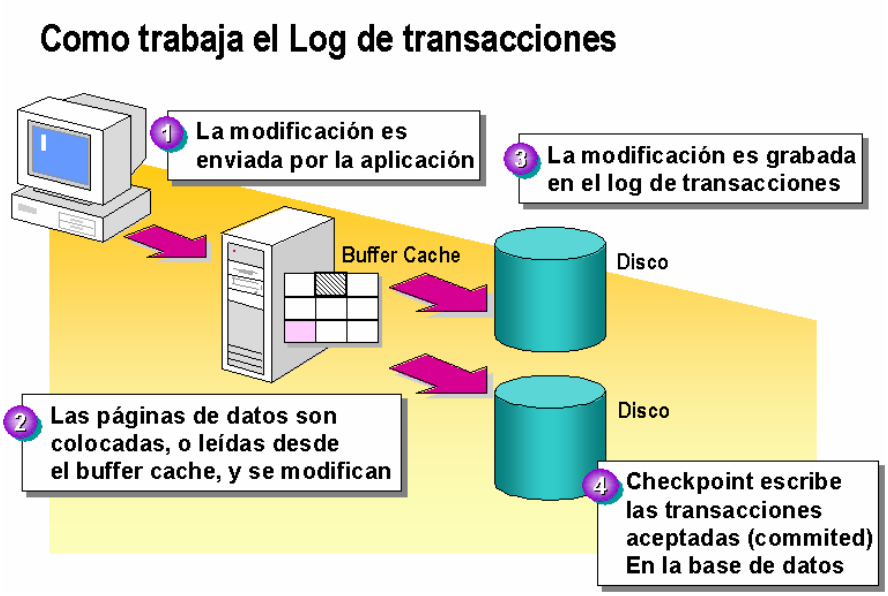


Figura 14 – Log de Transacciones

Los marcadores de transacción en el LOG son usados durante la recuperación automática para determinar los puntos de inicio y el fin de una transacción. Una transacción es considerada completa cuando el marcador BEGIN TRANSACTION tiene un marcador asociado COMMIT TRANSACTION. Las páginas de datos son escritas al disco cuando ocurre el CHECKPOINT.

Creación de Bases de Datos.
Las dos formas más comunes de creación de BDs son por medio del asistente o desde el analizador de consultas. Si vamos por la segunda, emplearemos la sentencia CREATE DATABASE, cuya sintaxis reducida es:

```
CREATE DATABASE NombreBaseDatos
[ ON [PRIMARY
NAME = nombreArchivoLógico,
FILENAME = 'nombreArchivoSO',
SIZE = tamaño,
MAXSIZE = { tamañoMáximo | UNLIMITED } ,
FILEGROWTH = incrementoCrecimiento) [,…n]
]
[ LOG ON
NAME = nombreArchivoLógico,
FILENAME = 'nombreArchivoSO',
SIZE = tamaño,
MAXSIZE = { tamañoMáximo | UNLIMITED } ,
FILEGROWTH = incrementoCrecimiento) [,…n]
[COLLATE nombre_collation] [ FOR LOAD | FOR ATTACH ]
```

- Argumentos

nombreBaseDatos
Es el nombre de la nueva base de datos, deben ser únicos en un servidor y pueden tener hasta

128 caracteres, a menos que no se especifique ningún nombre lógico para el registro. Si no se especifica ningún nombre lógico de archivo de registro, SQL Server genera un nombre lógico al anexar un sufijo a nombreBaseDatos.

ON

Especifica que los archivos de disco utilizados para almacenar la parte de datos (archivos de datos) se han definido explícitamente. La palabra clave va seguida de una lista delimitada por comas de elementos que definen los archivos de datos del grupo de archivos principal.

PRIMARY

Especifica que la lista de archivos está asociada al grupo principal. Este grupo contiene todas las tablas del sistema de base de datos. También contiene todos los objetos no asignados a los grupos de archivos de usuario. El primer archivo especificado pasa a ser el archivo principal, el cual contiene el inicio lógico de la base de datos y de las tablas del sistema. Una base de datos sólo puede tener un archivo principal. Si no se especifica PRIMARY, el primer archivo enumerado en la instrucción CREATE DATABASE se convierte en el archivo principal.

LOG ON

Especifica que los archivos de registro de la base de datos (archivos de registro) se han definido explícitamente. La palabra clave va seguida de una lista delimitada por comas la cual define las características de los archivos de registro. Si no se especifica LOG ON, se crea automáticamente un único archivo de registro con un nombre generado por el sistema y un tamaño que es el 25% de la suma de los tamaños de todos los archivos de datos de la base de datos.

FOR LOAD

Cláusula que se mantiene por compatibilidad con versiones anteriores de SQL Server.

La base de datos se crea con la opción de base de datos dbo use only activada y el estado se establece en "cargando". En realidad esto no es necesario en SQL Server 7.0 porque la instrucción RESTORE puede volver a crear la base de datos como parte de la operación de restauración.

FOR ATTACH

Crea la base de datos desde un conjunto existente de archivos del sistema operativo. Debe existir una entrada de archivos que determine cual es el archivo principal, las otras entradas son necesarias si existen archivos creados en una ruta de acceso distinta de cuando se creó la base de datos por primera vez o se adjuntó por última vez.

Utilice el procedimiento almacenado del sistema sp_attach_db en lugar de emplear CREATE DATABASE FOR ATTACH directamente, esto deberá emplearlo si debe especificar más de 16 archivos.

COLLATE

Especifica el conjunto de caracteres que se empleará para almacenar información en la base de datos, se puede emplear un conjunto de caracteres especificado por Windows o por SQL Server. De no especificarse se empleará el conjunto de caracteres seleccionado en el momento de la instalación

NAME

Especifica el nombre lógico del archivo.

No se requiere este parámetro cuando se especifica FOR ATTACH. Este nombre es el utilizado para referenciar al archivo en las sentencias del Transact-SQL que se ejecuten después.

FILENAME

Especifica el nombre de archivo del sistema (archivo físico).

Se debe especificar la ruta de acceso y nombre de archivo que el sistema operativo utiliza cuando crea la base de datos. La ruta de acceso debe especificar un directorio en el servidor sobre el que se instalo SQL Server. No se puede especificar un directorio en un sistema comprimido de archivos.

SIZE

Especifica el tamaño para el archivo. De no hacerlo SQL Server utiliza el tamaño del archivo principal de la base de datos model.

Cuando este parámetro no es especificado para un archivo secundario o de registro SQL Server automáticamente le asigna 1 MB.

El valor mínimo a asignar es de 512 KB. Si no se especifica tamaño, el valor predeterminado es 1 MB. El tamaño especificado para el archivo principal debe tener al menos el tamaño del archivo principal de la base de datos model.

MAXSIZE
Especifica el tamaño máximo de crecimiento del archivo. Se pueden utilizar los sufijos KB y MB, el valor predeterminado es MB. Especifique un número entero; no incluya decimales. Si no se especifica, el archivo aumenta hasta que el disco esté lleno.
UNLIMITED
Especifica que el archivo aumenta de tamaño hasta que el disco esté lleno.
FILEGROWTH
Especifica el incremento de crecimiento del archivo, este valor no puede exceder el valor MAXSIZE. Emplee un número entero. Un valor 0 indica que no hay crecimiento.
El valor se puede especificar en MB, KB o %, el valor predeterminado es MB. Cuando se especifica %, el tamaño de incremento de crecimiento es el porcentaje especificado del tamaño del archivo en el momento en que tiene lugar el incremento. De no emplear FILEGROWTH, el valor predeterminado es 10% y el valor mínimo es 64 KB. El tamaño especificado se redondea al múltiplo de 64 KB más cercano.

Una vez creada la BD se está en condiciones de utilizarla para realizar consultas, almacenar nuevos datos, etc.

Javascript

Introducción.
JavaScript es un lenguaje interpretado, desarrollado por Netscape inicialmente para sus navegadores. Actualmente es compatible con los navegadores más utilizados: Netscape Communicator, Microsoft Internet Explorer, Opera…

JavaScript permite crear páginas web interactivas (DHTML) con relativa facilidad. Se utiliza tanto para programación en el lado del cliente (navegadores fundamentalmente) como en el lado del servidor (ASP, Netscape Enterprise Server).

El núcleo de JavaScript incluye los elementos típicos de un lenguaje de programación: variables, sentencias, estructuras, operadores… y además incorpora un conjunto de objetos de utilización frecuente (Array, Math, etc…).

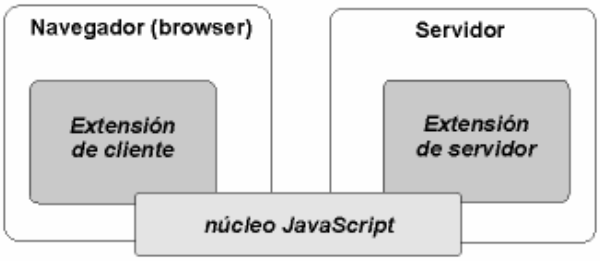


Figura 15 – Extensiones del Núcleo JavaScript

La extensión para cliente permite acceder a los objetos que utiliza el navegador y al DOM (Document Object Model), la jerarquía de objetos de un documento HTML. Añade soporte para control de eventos, de tal forma que el programa puede interactuar con el usuario.

La extensión para servidor aporta objetos que permiten un acceso sencillo a la interfaz CGI y la utilización de servicios del servidor web.

JavaScript es un lenguaje orientado a objetos (en el sentido de que permite construir objetos y utilizarlos), con control de tipos muy débil. Por ejemplo, el programador puede utilizar variables que no han sido declaradas y asignar valores de distintos tipos a la misma variable a lo largo del programa.

Aunque la similitud del nombre puede inducir a creer que JavaScript es equivalente a Java, la verdad es que no tienen prácticamente nada en común. Java ha sido desarrollado para construir grandes aplicaciones distribuidas, con el objetivo de máxima fiabilidad. JavaScript se creó con el objetivo de ser un lenguaje sencillo de aprender y utilizar, ideal para desarrollar pequeños programas que corren en un navegador.

En esta tabla se pueden apreciar algunas diferencias entre JavaScript y los applets de Java:

JAVASCRIPT	JAVA (Applets)
Programa interpretado por el cliente	Programa compilado en bytecodes. Es bajado del servidor y ejecutado en el cliente (Applets).
Orientado a objetos. No distingue entre tipos de objetos. Las propiedades y los métodos son añadidos dinámicamente a los objetos.	Los objetos son instancias de clases. La herencia se consigue a través de la jerarquía de clases. Las clases y los objetos se definen en tiempo de diseño y no pueden ser modificados en tiempo de ejecución.
El código JavaScript se funde con el código HTML	El código se integra en la página HTML, pero como una unidad llamada Applet
No hay definición de tipos para las variables	Definición estática de tipos. Control de tipos fuerte.
Limitaciones de seguridad	Limitaciones de seguridad

El núcleo de JavaScript.

- Variables

JavaScript define tres tipos básicos de valores: números, cadenas de texto y booleanos (true/false). Además incorpora los valores especiales: null , undefined y NaN.
Las variables no necesitan ser declaradas y pueden almacenar cualquier tipo de dato de forma dinámica:

```
cadena ="Esto es una cadena de texto";
cadena = 22; // ahora cadena vale 22
```

El tipo de dato que almacena una variable se resuelve según el contexto. Por ejemplo, si se utiliza el operador suma (+) mezclando cadenas de caracteres con valores numéricos, el resultado es una cadena:

```
cadena = "El número es ";
numero = 22;
resultado = cadena + numero; // >> El número es 22
```

Si la cadena de texto representa un número y el operador es distinto de (+), JavaScript intentará realizar la operación especificada:

```
cadena = "42";
numero = 10;
resultado = cadena - numero; // >> 32
```

Y devolverá error en caso de que la operación no tenga sentido:

```
cadena = "cadena";
numero = 10;
resultado = cadena - numero; // >> NaN
```

Es recomendable declarar las variables antes de utilizarlas:

```
var cadena; // contiene 'undefined'
```

Ya que una variable no declarada implícita o explícitamente provocará un error de ejecución si

se utiliza en una expresión:

```
// cadena no ha sido inicializada ni declarada explícitamente
resultado = cadena + 1; // error de ejecución (cadena no está definida)
resultado = cadena + "_1" // error de ejecución (cadena no está definida)
```

Sin embargo, si 'cadena' ha sido declarada, la expresión se evalúa y dará como resultado 'NaN' para el primer caso y 'undefined_1' para el segundo (concatena 'undefined' con '_1').

También tiene importancia la declaración de las variables a la hora de considerar su alcance y visibilidad. Las variables declaradas fuera de una función son consideradas globales al programa (documento). Para utilizar una variable local que utiliza el mismo nombre que una variable global es necesario declarar la primera de forma explícita con 'var'.

```
cadena = "hola";
function f ()
{
resultado = cadena + ", un saludo"; // >> "hola, un saludo"
cadena = "Hasta luego"; // cambia el valor de la variable global
}
function ff ()
{
var cadena;

cadena = 1; // cadena (local) vale 1
// cadena (global) vale "hola"
}
```

Las variables globales a nivel de documento pueden ser accesibles desde otros documentos (ventanas o frames del navegador) utilizando el nombre del documento y el operador de acceso:

```
resultado = ventana1.cadena + ventana2.cadena;
```

En el caso de un FRAMESET, los programas que se ejecutan en los frames pueden acceder a las variables globales de dicho FRAMESET utilizando el especificador 'parent':

```
resultado = ventana1.cadena + parent.cadena;
```

- Objetos en JavaScript

JavaScript ofrece una aproximación sencilla al modelo de programación orientada a objetos. En definitiva, un objeto es una estructura que encapsula tanto datos (variables) como la funcionalidad para manejarlos.

Los objetos pueden ser definidos utilizando un inicializador:

```
objeto = { propiedad1:valor1, propiedad2:valor2, ...};
objeto1 = { color:"rojo", forma:"círculo", tamano:20};
```

O a través de una función que haga de constructor. Esta es una opción más cercana a la filosofía de otros lenguajes orientados a objetos que utilizan clases para la declaración de

objetos.

```
function objeto( )
{
this.color = "rojo";
this.forma = "cuadrado";
this.tamano = 20;
}
```

Para crear nuevas instancias del objeto:

```
objeto2 = new objeto( );
```

Por supuesto, los constructores pueden recibir parámetros, utilizados habitualmente para inicializar el objeto:

```
function objeto (color, forma, tamano)
{
this.color = color;
this.forma = forma;
this.tamano = tamano;
}
```

```
objeto3 = new objeto ("rojo", "triangulo", 50);
```

Una vez creado el objeto, las propiedades se pueden definir (agregar o modificar) de forma dinámica en tiempo de ejecución:

```
objeto3.peso = 30;
objeto3.nombre = "ventana";
```

También se puede modificar el constructor en tiempo de ejecución, utilizando la palabra clave 'prototype':

```
objeto.prototype.estado = null;
```

```
objeto4 = new objeto ("rojo", "circulo", 30, "solido");
```

A partir de JavaScript 1.1 los parámetros definidos mediante índices en lugar de nombres deben ser referenciados con índices. Los parámetros definidos mediante nombres deben ser referenciados a través de dichos nombres. La excepción a esta regla son los objetos generados a partir de elementos HTML, que permiten el acceso a sus propiedades (por ejemplo, elementos de un formulario) a través de índices (que siguen el orden de creación de la página) o sus propios nombres.

```
objeto4[10] = "Otra propiedad";
cadena = objeto[10];
```

El acceso a las propiedades de los objetos se lleva a cabo a través del operador de alcance '.' (punto). Por ejemplo, si se crean objetos que contienen a otros objetos. El acceso a las propiedades más internas se haría:

```
objeto_nivel_superior.objeto_nivel_1.objeto_nivel2.propiedad_objeto_nivel_2
```

Los objetos también pueden definir y encapsular su propia funcionalidad a través de métodos (funciones internas). Estas funciones se definen como una función cualquiera y luego se añaden como las demás propiedades del objeto:

```
// declaración del método
function muestraCaracteristicas ( )
{
document.write ("color: "+ this.color + "forma: " + this.forma);
}

// constructor
function objeto (color, forma, tamano)
{
this.color = color;
this.forma = forma;
this.tamano = tamano;
this.muestraCaracteristicas = muestraCaracteristicas;
}

// instanciación y utilización del objeto

objeto5 = new objeto ("rosa", "cubo", 60);
objeto5.muestraCaracteristicas( );
```

A partir de JavaScript 1.1, los objetos pueden ser eliminados de memoria utilizando el operador 'delete':

```
delete objeto5; // objeto5 deja de estar disponible
```

Objetos predefinidos.

- El objeto Date (fecha)

Este objeto permite trabajar con fechas y horas en los programas. Por ejemplo, la sentencia:

hoy = new Date(); crea un nuevo objeto Date llamado 'hoy' y le asigna la fecha y la hora actuales.

Para trabajar con este objeto se suelen utilizar estos métodos:

Métodos del Objeto Date	
getDate()	Convierte la fecha a cadena de texto, utilizando como referencia la configuración horaria del sistema
getDay()	UTC Devuelve el número de milisegundos desde las 00 horas del 1 de enero de 1970 hasta la fecha almacenada por el objeto Date
getHours()	Convierte la fecha a cadena de texto, utilizando como referencia la hora del meridiano cero (GMT)
getMinutes()	Asigna el valor del objeto Date
getMonth()	Asigna el año
getSeconds()	Asigna los segundos
getTime()	Asigna el mes
GetTimezoneOffset()	Asigna los minutos
GetYear()	Asigna la hora
parse()	Asigna el día del mes a un objeto Date
SetDate()	Devuelve el número de milisegundos desde las 00 horas del 1 de enero de 1970 hasta el instante representado por la fecha que recibe como parámetro: milisegundos = Date.parse("Mon, 25 Dec 1995 13:30:00 GMT+0430")

	milisegundos2 = Date.parse ("Mon, 25 Dec 1995"); El formato de la fecha es el indicado en el ejemplo. Si no se incluye la zona horaria, se utiliza la zona horaria del sistema.
setHours()	Devuelve el año. El formato utiliza dos dígitos para años comprendidos entre 1900 y 1999. Para el resto utiliza cuatro dígitos.
setMinutes()	Devuelve el desfase horario (en minutos) del sistema local con respecto al meridiano cero.
setMonth()	Devuelve un valor numérico asociado al instante para la fecha especificada
setSeconds ()	Devuelve el día del mes
setTime()	Día de la semana
setYear()	Devuelve la hora
toGMTString()	Devuelve los minutos
toLocaleString()	Devuelve el mes

El objeto Date sólo representa un cierto instante de tiempo (no se actualiza automáticamente). Para consultar el instante actual se puede crear un nuevo objeto Date utilizando el operador 'new'.

Para crear un objeto Date con una fecha arbitraria se puede utilizar el constructor

aquel_dia = new Date (2000, 0, 1, 2, 30, 30)

En el ejemplo anterior se crea un objeto Date que representa el instante: 2 horas, 30 minutos, 30 segundos del 1 de Enero (enero = 0, febrero=1, …) de 2000.

- El objeto Array

La creación de un array se lleva a cabo con la sentencia:

mi_array = new Array();

Se pueden asignar a un array valores de cualquier tipo (incluyendo objetos)

```
mi_array[0] = 1;
mi_array[1] = "Cadena de texto";
mi_array[2] = Date();
```

Tanto el tamaño (longitud) del array como el tamaño de las celdas individuales se asignan de forma dinámica (automática) a medida que se asignan elementos.

Hay que tener en cuenta que el tamaño de los arrays en JavaScript puede crecer pero no decrecer, es decir, para el siguiente trozo de código:

var array_grande = Array();

array_grande[1000]=1;

Se construye un array de 1001 posiciones (aunque realmente sólo se utiliza una de ellas).

Se pueden utilizar cadenas de texto como identificadores de índices (arrays asociativos) en la versión de Navigator 3 y posteriores.

El constructor permite crear arrays de una determinada longitud inicial:

otro_array = new Array (10);

E inicializar valores a partir de la declaración:

este_array = new Array ("uno", "dos", "tres", "cuatro");

Propiedades de Objeto Array	
Length	Número de elementos del array

Métodos de Objeto Array	
concat()	Une dos arrays y devuelve como resultado un array con la unión
join()	Devuelve una cadena de texto que contiene la unión de los elementos del array
pop()	Borra y devuelve el último elemento de un array
push()	Añade un elemento a un array y devuelve ese elemento
reverse()	Refleja el contenido de un array (el primer elemento pasa a ser el último)
shift()	Borra y devuelve el primer elmento de un array

slice()	Extrae una sección de un array y la devuelve como un nuevo array:
	datos = new Array ()
	datos[1]="uno";
	datos[2]="dos";
	datos[3]="tres";
	dosprimeros = datos.slice(1,2);
	// dosprimeros[0]="uno"
	// dosprimeros[1]="dos"

splice()	Añade y/o elimina elementos de un array

sort()	Ordena los elementos de un array. Por defecto utiliza ordenación lexicográfica (alfabética). Se puede definir una función externa que implemente el criterio de ordenación:
	
	function compareNumbers(a, b) {
	return a – b
	}
	
	numberArray.sort(compareNumbers)

toString ( )	Devuelve una cadena que representa al array
unshift ( )	Añade uno o más elementos al comienzo de un array y devuelve el número actualizado de elementos

El objeto Array no existe en JavaScript 1.0 (Netscape 2.x y MIE 3.x). Sin embargo se puede utilizar una aproximación a través de funciones y arrays normales para conseguir resultados similares.

- El objeto String

Es un objeto predefinido de JavaScript para manejo de cadenas de caracteres.

Propiedades del Objeto String

length	longitud de la cadena de caracteres asociada al objeto

Se crea con el constructor String, pero cualquier variable que contiene una cadena de texto puede ser tratada como String (JavaScript se encarga de hacer las conversiones necesarias):

```
cadena1 = new String ("Un String");
longitud = cadena1.length;

cadena2 = "Una cadena de caracteres";
longitud = cadena2.length; // se convierte temporalmente a String para utilizar la propiedad length
```

Métodos del Objeto String	
anchor	Crea un anclaje HTML: referencia = new String ("Zona inferior"); document.write (referencia.anchor ("inferior")); Sería equivalente a: Zona inferior
big	devuelve una cadena que representa el texto en formato BIG de HTML
bold	devuelve una cadena que representa el texto en negrita (...) en HTML
charAt	devuelve el caracter correspondiente a la posición indicada (teniendo en cuenta que las cadenas comienzan con índice cero)
charCodeAt	devuelve el código (codificación ISO-Latin-1) correspondiente al carácter situado en la posición indicada (las cadenas comienzan con índice cero)
concat	concatena dos cadenas y devuelve como resultado esa unión
fixed	devuelve una cadena que representa el texto en formato TT de HTML
fromCharCode	devuelve una cadena con los códigos (ISO-Latin-1) de los caracteres

indexOf	devuelve el índice de la primera aparición del valor especificado dentro de la cadena representada por el objeto:
	
	cadena = "uno dos tres cuatro";
	posicionDeDos = cadena.indexOf ("dos");
	// posicionDeDos = 4

italics	devuelve una cadena que representa el texto en formato <I>...</I> (cursiva) de HTML

lastIndexOf	devuelve el índice de la última aparición del valor especificado dentro de la cadena representada por el objeto:
	
	cadena = "uno dos tres cuatro dos";
	posicionDeDos = cadena.lastIndexOf ("dos");
	// posicionDeDos = 20

link	construye una cadena que se corresponde con la marca de enlace (<a>) de HTML:
	
	enlace = new String ("Epsilon Eridani");
	marcaEnlace = enlace.link ("http://www.epsilon-eridani.com");
	// marcaEnlace = Epsilon Eridani

slice	extrae un trozo de una cadena: cadena = "dos trozos"; trozo = cadena.slice (4,9); // trozo='trozos' Si el segundo carácter es negativo, indica el la posición comenzando por la

	parte derecha de la cadena original: trozo = cadena.slice (4,-1); // trozo='trozo'
small	devuelve una cadena que se corresponde con el formato SMALL de HTML

split	divide una cadena en trozos y los devuelve en un array. Se pasa como parámetro la cadena utilizada como separador:
	
	cadena = "uno dos tres cuatro dos";
	trozos = cadena.split (" "); // el separador es el espacio
	document.write (trozos[0]); // trozos[0] = 'uno'
	
	Se puede especificar un segundo parámetro que indica el número máximo de trozos:
	
	trozos = cadena.split (" ",2);
	// se crea un array de 2 elementos

sub	devuelve una cadena con el formato SUB (subíndice) de HTML
substr	devuelve una subcadena a partir de los parámetros 'inicio' y 'longitud': cadena = "uno dos tres cuatro dos"; subcadena = cadena.substr (4, 3); // subcadena = 'dos' Si no se especifica la longitud, se considera hasta el final de la cadena. Si 'inicio' es un valor negativo se cuenta la posición inicial a partir del final de la cadena.
substring	devuelve una subcadena que está comprendida entre dos índices de la cadena actual: cadena = "uno dos tres cuatro dos"; subcadena = cadena.substring (4, 7); // subcadena = 'dos'
sup	devuelve una cadena con el formato SUP (superíndice) de HTML
toLowerCase	devuelve una cadena con los caracteres convertidos a minúsculas
toUpperCase	devuelve una cadena con los caracteres convertidos a mayúsculas

Los operadores y sentencias de control en JavaScript son muy similares a las de cualquier otro lenguaje de programación, por ello no profundizaré sobre estos aspectos del lenguaje en este apartado. Nuestro interés en esta sección es dar una descripción de las herramientas utilizadas haciendo foco en las características principales de las mismas que propiciaron su elección frente a otras y no una exponer un tutorial de las mismas.

- Funciones Globales

Estas son las que forman parte del propio núcleo de JavaScript, no pertenecen a ningún objeto en particular.

Otros atributos	

escape	codifica los caracteres no alfanuméricos de una cadena para que pueda formar parte de una URL. Por ejemplo, para especificar un valor de parámetro que contiene espacios:
	
	// url no válida (el parámetro contiene espacios)
	url = "cgi-bin/prog.pl?cadena=esta cadena";
	
	// ahora con 'escape':
	url = "cgi-bin/prog.pl?cadena=";
	url += escape ("esta cadena");

	// Resultado: cgi-bin/prog.pl?cadena=esta%20cadena No codifica los caracteres especiales: * @ - _ + . /

eval	evalúa una expresión JavaScript (como lo haría el analizador) sin necesidad de referenciar un objeto determinado ni una variable declarada.
	
	sentencias = new Array(2);
	sentencias[0]="document.write ('expresion 1')";
	sentencias[1]="suma = 2+2";
	
	if (condicion) eval (sentencias[0]);
	else eval (sentencias[1]);

isNaN	devuelve true si el valor pasado como argumento no es un número

parseFloat	evalúa una cadena de texto que expresa un número en notación científica y devuelve el número en coma flotante correspondiente o NaN en caso de no encontrar una expresión válida. Ejemplo:
	
	valor = parseFloat ("1.4142e-10");
	valor1 = parseFloat (".0012");
	valor2 = parseFloat ("a12"); // devuelve NaN
	
	Si la cadena no representa un número en coma flotante válido, la función devuelve NaN:
	
	valor3 = parseFloat ("12.8abc"); // valor3 = 12.8
	valor4 = parseFloat ("abc12.8"); // valor4 = NaN

parseInt	evalúa una cadena de texto y devuelve el entero representado por dicha cadena. Se puede especificar la base en la que se expresa el número:
	
	entero1 = parseInt ("FF", 16); // entero1 = 255
	entero2 = parseInt ("1010", 2); // entero2 = 10
	
	Si no se especifica la base, se sigue el siguiente criterio:
	· Los números que comienzan por 0 se considerarán en base octal
	· Los números que comienzan por 0x se consideran hexadecimales
	· El resto de los números se consideran enteros en base 10
	
	Si la cadena no representa un número válido, la función devuelve NaN
	
	Ejemplos:
	
	entero3 = parseInt ("123abcd"); // entero3 = 123
	entero4 = parseInt ("3.14"); // entero4 = 3
	entero5 = parseInt ("abc123") // entero5 = NaN

unescape	operación contraria a escape( ). Decodifica los caracteres especiales de la URL

- Extensión para el cliente

La extensión para cliente permite acceder a los objetos que utiliza el navegador y al DOM (Document Object Model), la jerarquía de objetos de un documento HTML:

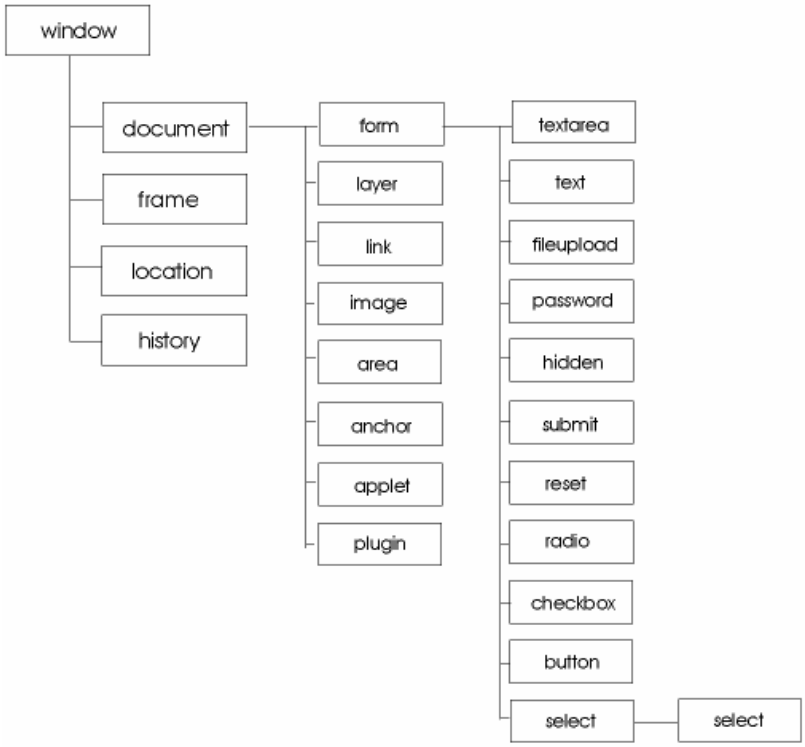


Figura 16 – Jerarquía de Objetos de DOM de JavaScript

Estos objetos los crea el motor JavaScript del navegador (o al menos están disponibles para dicho motor) a partir de la página HTML abierta por el propio navegador.

El proceso comienza cuando el navegador abre una página HTML (por ejemplo solicitándola a un servidor web o directamente del sistema de ficheros local). La información contenida en dicha página constituye una corriente de datos con un inicio y un final.

El analizador sintáctico (*parser*) del navegador lee esa corriente de datos de forma secuencial, extrayendo (y mostrando en pantalla si es necesario) los distintos elementos HTML. Estos elementos también son convertidos a objetos que formarán parte de la estructura del documento (DOM) y que estarán disponibles para la propia funcionalidad del navegador y para los scripts y objetos (plugins) incluidos en la página.

De hecho, los programas JavaScript del lado del cliente pueden interactuar de dos formas con las páginas de las que forman parte:

- ✓ Durante la carga del documento, es decir, durante el análisis de la corriente de datos, pueden incorporar a dicha corriente expresiones HTML que serán aceptadas por el *parser* HTML y

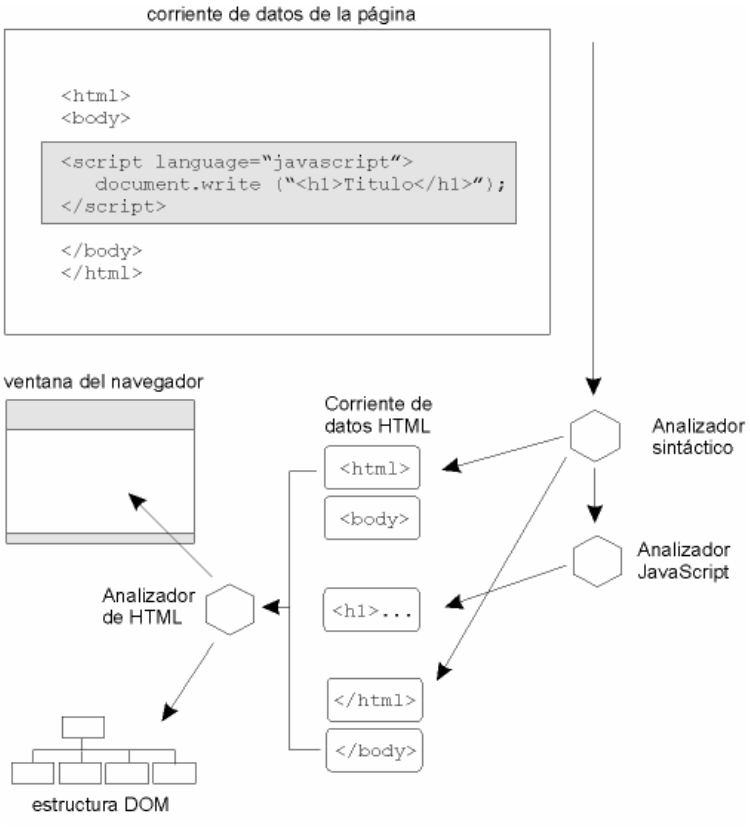


Figura 17 – Ejecución de un script JavaScript

convertidas en los elementos necesarios. Habitualmente se utiliza este método para añadir información dinámica (fecha, hora, etc...) o para mostrar información en función de alguna condición (tipo de navegador que utiliza el usuario, resolución, etc...). El proceso se muestra, de forma aproximada, en la siguiente figura:

- ✓ Una vez cargada la página, el documento HTML está reflejado en la estructura de objetos construida por el navegador. Los programas JavaScript pueden acceder a esta estructura para recoger información o incluso para modificarla de forma dinámica. La interactividad con el usuario se consigue a través del mecanismo de captura y tratamiento de eventos.

El acceso a los distintos objetos y sus propiedades se lleva a cabo recorriendo la jerarquía:

```
<html>
<body>
<form name='form1'>
<input type='text' name='nombre'>
</form>
...
window.document.form1.nombre.value = "Este es el nombre";
...
```

- Jerarquía de Objetos DOM

Esta organización de objetos que se muestra en la Fig. III.a, encapsula una serie de comportamientos básicos que no describiré individualmente, ya que puede consultarse una guía detallada en Internet[4].
Para comprender la función de los principales objetos utilizados en JavaScript, basta describir a grandes rasgos cuál es el objetivo de aquellos de más alto nivel:

WINDOW → El objeto WINDOW representa una ventana o marco del navegador. Es el elemento raíz de la estructura de objetos de cada página HTML.
Tiene definidos los eventos: *onBlur, onDragDrop, onError, onFocus, onLoad, onMove, onResize, onUnload.*

DOCUMENT → Este objeto se construye a partir de la marca BODY de HTML. Sobre él pueden actuar los eventos: *onClick, onDblClick, onKeyDown, onKeyPress, onKeyUp, onMouseDown, onMouseUp.*

LOCATION → Contiene información sobre la dirección de la página actual (URL actual). Sus métodos son *reload* y *replace*.

HISTORY → Contiene un array con las URLs de las páginas cargadas por la ventana actual. Posee los métodos: *back, forward* y *go*.

NAVIGATOR → Permite acceder (en modo sólo lectura) a algunas de las características del navegador.

- Tratamiento de eventos

Los eventos permiten construir aplicaciones interactivas en el lado del cliente. Los eventos están siempre asociados a una acción sobre un elemento determinado. Los más utilizados son:

[4] http://www.epsilon-eridani.com/p2D/ap/p2D.php/pub/eridani/accion/ver_doc/doc/Javasc986/pag_hoj/1/plantilla.htm?sesion=a5df2e264922820a8585019f4a444c11

Evento	Tiene efecto sobre	Ocurre cuando	Manejador
Abort	imágenes	El usuario cancela la carga de una imagen	*onAbort*
Blur	ventanas y elementos de formularios	El usuario abandona la ventana o elemento (pierde el foco)	*onBlur*
Change	text fields, textareas, select lists	El usuario cambia el valor de un elemento	*onChange*
Click	buttons, radio buttons, checkboxes, submit buttons, reset buttons, links	El usuario pulsa sobre el elemento o el enlace	*onClick*
Error	images, windows	La carga de la ventana o la imagen causa un error	*onError*
Focus	windows and all form elements	El usuario pasa el control (foco) a un elemento determinado	*onFocus*
KeyDown	documents, images, links, text areas	El usuario pulsa una tecla.	*onKeyDown*
KeyPress	documents, images, links, text areas	El usuario da una pulsación con una tecla o la mantiene pulsada.	*onKeyPress*
KeyUp	documents, images, links, text areas	El usuario deja de pulsar una tecla	*onKeyUp*
Load	document body	El usuario carga una página en el navegador	*onLoad*
MouseDown	documents, buttons, links	El usuario pulsa una tecla del ratón	*onMouseDown*
MouseMove		El usuario mueve el ratón	*onMouseMove*
MouseOut	areas, links	El usuario mueve el cursor fuera del elemento	*onMouseOut*
MouseOver	links	El usuario mueve el cursor dentro del elemento	*onMouseOver*
MouseUp	documents, buttons, links	El usuario deja de pulsar una tecla del ratón	*onMouseUp*
Move	windows	El usuario (o algún script) mueve una ventana	*onMove*
Reset	forms	El usuario pulsa el botón de reset	*onReset*
Resize	windows	El usuario (o algún script) redimensiona la ventana del navegador	*onResize*
Select	text fields, textareas	El usuario selecciona el contenido de los elementos del formulario	*onSelect*
Submit	forms	El usuario pulsa el botón para enviar un formulario	*onSubmit*
Unload	document body	El usuario sale de la página (carga otra página o cierra esa ventana del navegador)	*onUnload*

Su utilización es muy sencilla. Para indicar a JavaScript que se desea hacer el tratamiento de un determinado evento, se coloca el manejador correspondiente en la marca HTML del elemento:

```
<marca_HTML onEvento=expresión JavaScript >
```

Por ejemplo, podemos hacer que aparezca un mensaje cada vez que el usuario pulsa un botón:

...

```
<input type='button' name='boton1' value='pulsa' onClick = 'alert("Se ha pulsado"); return true;' >
...
```

Lo más habitual es colocar una llamada a una función que se utilizará como manejador propiamente dicho del evento:

```
function manejador ( )
{
// funcionalidad del manejador
....
}
...
<input type='button' name='boton1' value='pulsa' onClick = 'manejador()' >
...
```

CAPÍTULO III
SISTEMA DE BASE DE DATOS

Qué es una BD?

Según el propósito de cada autor una definición para 'base de datos' puede variar en complejidad, pero no en concepto. Consensuando una sola acepción, la que más clara se presenta es la siguiente:

"Una base de datos es una colección de datos organizados y estructurados según un determinado modelo de información que refleja no sólo los datos en sí mismos, sino también las relaciones que existen entre ellos."

Una base de datos se diseña con un propósito específico y debe ser organizada con una lógica coherente. Los datos pueden ser compartidos por distintos usuarios y aplicaciones, pero deben conservar su integridad y seguridad al margen de las interacciones de ambos. La independencia en el uso de esos datos y la minimización de la redundancia de los mismos dependen de la unicidad en la definición y descripción de los mismos.

Modelos de Bases de Datos

Los modelos clásicos de tratamiento de los datos son:

- Jerárquico: Puede representar dos tipos de relaciones entre los datos, relaciones de uno a uno y relaciones de uno a muchos. Este modelo tiene forma de árbol invertido, una rama puede tener varios hijos, pero cada hijo sólo puede tener un padre.
- En red: Permite la representación de muchos a muchos, de tal forma que un registro de la BD puede tener varias ocurrencias superiores a él. Este modelo evita redundancia en la información, a través de un tipo de registro denominado el conector. En el modelo en red la realidad se refleja por medio de registros lógicos que representan a una entidad y se relacionan entre sí por medio de flechas.
- Relacional: Desde los años 80 es el modelo más utilizado, ya que permite una mayor eficacia, flexibilidad y confianza en el tratamiento de los datos. Este modelo ofrece numerosas ventajas sobre los 2 anteriores, como ser el rápido aprendizaje por parte de usuarios que no tienen conocimientos profundos sobre sistemas de bases de dados. En el modelo relacional el mundo real es representado por tablas relacionadas entre sí por columnas comunes. Las BD de esta categoría se basan en una estructura principal, la relación, es decir una tabla bidimensional compuesta por líneas y columnas. Cada línea se llama tupla y representa una entidad que nosotros queremos memorizar en la base de datos. Las características de cada entidad están definidas por las columnas de las relaciones, que se llaman atributos. Entidades con características comunes, es decir descritas por el mismo conjunto de atributos, formarán parte de la misma relación.

Otros modelos que se destacan en la actualidad:

- Bases de datos distribuidas: Cada vez es más corriente el uso de arquitecturas de cliente-servidor y trabajo en grupo. Los principales problemas generados por esta tecnología se refieren a la duplicidad de datos y a su integridad al momento de realizar actualizaciones a los mismos. El control de la información también puede constituir una desventaja, debido a que se encuentra diseminada en diferentes localizaciones geográficas.
- Bases de datos orientadas a objetos: En estas, el esquema de la base de datos está representada por un conjunto de clases que definen las características y el comportamiento de los objetos que conformarán la base de datos. La diferencia principal respecto a los modelos anteriores es la *no positividad* de los datos. Esto es, con una base de datos tradicional, las operaciones que se tienen que efectuar en los datos se les piden a las aplicaciones que los usan, pero con una orientada a objetos sucede lo contrario. Los objetos

memorizados en la base de datos contienen tanto los datos como las operaciones posibles con tales datos. En cierto sentido, se podrá pensar en los objetos como en datos a los que se les ha dotado de "cierta inteligencia" que les permite saber cómo comportarse, sin tener que apoyarse en aplicaciones externas.

Arquitectura de un Sistema de BD

Uno de los objetivos fundamentales de un sistema de información es contar no sólo con recursos de información, sino también con los mecanismos necesarios para poder encontrarla y recuperarla. En este sentido las bases de datos son un elemento indispensable no sólo para el funcionamiento de los grandes motores de búsqueda y recuperación de información a lo largo y ancho de la Web, sino también para la creación de sitios Web, Intranets y otros sistemas de información en los que se precisa manejar grandes o pequeños volúmenes de información. Por ello, la creación de una BD a la que puedan acudir los usuarios para consultar y acceder a la información de su interés es imprescindible en cualquier sistema informativo en red o fuera de ella.

La arquitectura de un sistema de base de datos se basa en 3 niveles distintos:

- Nivel físico: Es el nivel más bajo de abstracción, el nivel real de los datos almacenados. Este nivel define cómo se almacenan los datos en el soporte físico, ya sea en registros o de cualquier otra forma, así como los métodos de acceso. Este nivel lleva asociada una *representación de los datos*, que es lo que denominamos *Esquema Físico*.
- Nivel conceptual: Es el que se corresponde con la visión de la BD desde el punto de visto del mundo real. Se trata con la entidad u objeto representado, sin importar el cómo, ni su forma de almacenado. Es la representación que hace la organización, incluyendo la definición y las relaciones de los datos. Este nivel lleva asociado un *Esquema Conceptual*.
- Nivel de visión: Son partes del esquema conceptual. El nivel conceptual presenta toda la base de datos, mientras que los usuarios sólo tienen acceso a pequeñas parcelas de ésta. Este nivel es el encargado de dividir estas parcelas. Un ejemplo sería el caso del empleado de una organización que tiene acceso a la visión de su nómina, pero no a la de sus compañeros. Este nivel se asocia a un *Esquema de Visión*.

Otros autores utilizan la denominación de nivel interno, nivel conceptual y nivel externo, para referirse a estos mismos niveles:

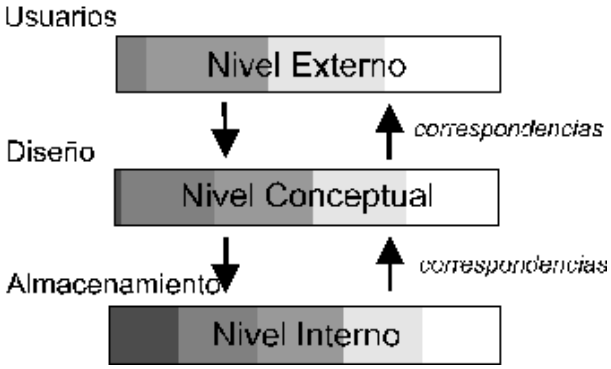


Figura 18 - Niveles de la arquitectura de un sistema de base de datos

Este modelo de arquitectura permite establecer el principio de independencia de los datos, sea esta independencia lógica o física. La independencia lógica significa que los cambios en el esquema lógico no deben afectar a los esquemas externos que no utilicen los datos modificados; la independencia física significa que el esquema lógico no se va a ver afectado por los cambios realizados en el esquema interno, correspondientes a modos de acceso, etc.

Diseño de la BD

El diseño de una base de datos debe contar con un buen análisis previo para definir

adecuadamente campos y relaciones necesarios entre los diferentes elementos. Si no se tiene en cuenta esto, si se necesita agregar un nuevo campo después de creada la base de datos, la misma tal vez deba ser creada nuevamente.

A la hora de diseñar una BD hay que distinguir por un lado el modelo de datos (instrumento) y por otro lado el esquema de datos (el resultado de aplicar ese modelo).

Modelo de datos.
Un modelo de datos es un conjunto de conceptos, reglas y convenciones que nos permiten describir los datos del universo del discurso. Un esquema es la estructura de datos obtenida tras aplicar dicho modelo.

El modelo de datos es fundamental a la hora de diseñar bases de datos. Jesús Tramullas en *Los sistemas de bases de datos* destaca 3 modelos fundamentales:

- Modelos lógicos **basados en objetos**: Los dos más extendidos son el modelo entidad-relación y el orientado a objetos. El modelo entidad-relación (E-R) se basa en una percepción del mundo compuesta por objetos, llamados entidades, y las relaciones entre ellos. Las entidades se diferencian unas de otras a través de atributos. El orientado a objetos también se basa en objetos pero que contienen valores y métodos, entendidos como órdenes que actúan sobre los valores, organizados en distintos niveles de anidamiento. Los objetos se agrupan en clases, relacionándose mediante el envío de mensajes. Algunos autores definen estos modelos como "modelos semánticos".

- Modelos lógicos **basados en registros**: El más extendido es el relacional, mientras que los otros dos, jerárquico y de red, se encuentran en retroceso. Estos modelos se usan para especificar la estructura lógica global de la BD, estructurada en registros de formato fijo de varios tipos. El modelo relacional representa los datos y sus relaciones mediante tablas bidimensionales, que contienen datos tomados de los dominios correspondientes. El modelo de red está formado por colecciones de registros, relacionados mediante punteros o ligas en grafos arbitrarios. El jerárquico es similar al de red, pero los registros se organizan como colecciones de árboles. Algunos autores definen estos modelos como "modelos de datos clásicos".

- Modelos **físicos** de datos: Son los menos usados. Algunos autores los definen como "modelos de datos primitivos".

Los objetivos del modelo de datos son por un lado formalizar y definir las estructuras permitidas para representar los datos, y por otro, diseñar la base de datos.

El Diseño propiamente dicho.
En el diseño de una base de datos, hay que tener en cuenta distintas consideraciones, entre las que destacan:
1. la velocidad de acceso,
2. el tamaño de la información,
3. el tipo de información,
4. la facilidad de acceso a la información,
5. la facilidad para extraer la información requerida y
6. el comportamiento del sistema de gestión de bases de datos con cada tipo de información.

Para plasmar los tres niveles en el enfoque o modelo de datos seleccionado, es necesario un programa o aplicación que actúe como interfaz entre el usuario, los modelos y el sistema físico. Estos son los **Sistemas de Gestión de Bases de Datos**. Un Sistema de Gestión de Bases de Datos, también llamado DBMS (Data Base Management System) no es más que un paquete de software, que se ejecuta en un ordenador anfitrión (host) que es quien centraliza los accesos a los datos y actúa de interfaz entre los datos físicos y los usuarios. Este sistema es capaz de llevar a cabo funciones como la creación y gestión de la base de datos misma, el control de accesos y la manipulación de datos de acuerdo a las necesidades de cada usuario.

Un sistema de gestión de base de datos **está formado por personas, máquinas, programas y datos**. Estos sistemas de gestión abarcan el conjunto de rutinas de software interrelacionadas cada una de las cuales es responsable de una determinada tarea.

Los componentes con los que debe contar un sistema de gestión de bases de datos ideal[5]:
1) Un lenguaje de definición de esquema conceptual.
2) Un sistema de diccionario de datos.
3) Un lenguaje de especificación de paquetes de entrada/salida.
4) Un lenguaje de definición de esquemas de base de datos.
5) Una estructura simétrica de almacenamiento de datos.
6) Un módulo de transformación lógica a física.
7) Un subsistema de privacidad de propósito general.
8) Un subsistema de integridad de propósito general
9) Un subsistema de reserva y recuperación de propósito general.
10) Un generador de programas de aplicación.
11) Un generador de programas de informes.
12) Un lenguaje de consulta de propósito general.

Los SGBD tienen dos funciones principales que son:
1º. La definición de las estructuras para almacenar los datos.
2º. La manipulación de los datos.

Además, los SGBD deben incorporar como herramienta fundamental dos tipos de lenguajes: uno para la definición de los datos, y otro para la manipulación de los mismos. El primero se denomina DDL (Data Definition Language) y es el que provee de los medios necesarios para definir los datos con precisión, especificando las distintas estructuras. El segundo se conoce como DML (Data Manipulation/Management Language) y es el facilita a los usuarios el acceso y manipulación de los datos.

La existencia de una sola copia de los datos para que todos los programas trabajen con ella, lo que se denomina “obtención de redundancia mínima”, para de esta manera poder eliminar la inconsistencia de los datos; o la capacidad de interactuar en un ambiente cliente/servidor donde los clientes o usuarios (ya sea en una intranet o desde Internet) puedan trabajar con un conjunto único de datos alojados en un servidor y donde varios clientes podrían estar trabajando al mismo tiempo son sólo algunas de las características con que cuenta el modelo de base de datos relacional. Existen diversos motores de base de datos que permiten trabajar ya sea con bases de datos existentes o creando nuevas con todas las capacidades de trabajo en red. Numerosas empresas se han volcado al desarrollo de sistemas de gestión de bases de datos como Oracle, Informix, PostgreSQL, Sybase, Microsoft, etc. y existen tanto soluciones comerciales de pago, como soluciones de acceso libre.

Tradicionalmente se ha hecho una distinción clara entre 2 tipos de bases de datos:

- Bases de datos referenciales: aquellas ofrecen registros que a su vez son representaciones de documentos primarios, entre ellas cabe distinguir:
  - ✓ bibliográficas: aquellas cuyo contenido son registros de tipo bibliográfico.
  - ✓ directorios: aquellas cuyo contenido está referido a la descripción de otros recursos de información, como por ejemplo un directorio de bases de datos.

- Bases de datos fuente: ofrecen el documento completo, no una representación del mismo, y entre las que cabe distinguir:
  - ✓ numéricas: contienen información de tipo numérico.
  - ✓ textuales: contienen el texto completo de un documento.
  - ✓ mixtas: combinan ambos tipos de información.

[5] TRAMULLAS, Jesús. "Sección 3: Los sistemas de bases de datos y los SGBD”. En Introducción a la Documática. http://tek.docunautica.com/

Sin embargo, el desarrollo de las aplicaciones multimedia ha dado un vuelco al concepto tradicional de base de datos, que sólo integraba elementos de información textual y numérica. Se han ido estandarizando poco a poco los formatos de archivo gráficos, de audio y de vídeo, y se han perfeccionado los métodos de compresión de este tipo de datos, ya que ocupan grandes cantidades de memoria. El desarrollo del hipertexto, al permitir la conectividad entre las referencias y los documentos fuente a través de los enlaces, ha roto también las fronteras entre documentos primarios y documentos secundarios, aunando en un mismo espacio datos referenciales y acceso directo al documento fuente.

Hasta no hace muchos años, las bases de datos eran productos comerciales desarrollados y mantenidos por ciertas empresas que las comercializaban ya fuese en formato CD-ROM o bien las distribuían para su consulta, previo pago, en línea vía telnet. Eran construidas sobre los sistemas de gestión más conocidos para crear y mantener bases de datos como FileMaker, Knosys, Access, etc. Hoy todos estos programas se han visto obligados a ser compatibles con la Web y a ofrecer la posibilidad de acceder, buscar y recuperar los datos en línea vía protocolo http.

El avance de esa tendencia condujo al desarrollo y comercialización una serie de herramientas y aplicaciones, comúnmente denominadas pasarelas Web, que permiten consultar las viejas -o nuevas- bases de datos creadas con estos sistemas de gestión mediante el navegador Web, pero también, la existencia de estas herramientas ha favorecido el hecho de que cualquier persona pueda hoy publicar su propia base de datos en su página Web, para que pueda ser consultada por cualquier usuario de la red. Estas pasarelas no son más que herramientas de software que permiten la comunicación entre el servidor Web y la base de datos.

Así pues, la World Wide Web se ha convertido en sí misma, en una interfaz de acceso a datos que puede ser utilizada por cualquier usuario. Los nodos de un hipertexto no se limitan a incluir texto, imagen o sonido, sino también *scripts* y otros elementos como APIs (Application Programming interface) o controladores para conectividad de bases de datos e intercambio de información tales como OLE (Open Database Connectivity), CGI (Common Gateway Interface), JDBC (Java Data Base Connectivity), SQL LINKS etc. Todos estos objetos son los que hacen posible la existencia de elementos y documentos dinámicos y los que aportan un verdadero dinamismo al hipertexto. Se trata de componentes que deben ser diseñados en la interfaz de programación para acceso a datos del hipertexto y que comprende tanto el diseño e interfaz de Objetos de Acceso a Datos, como la interfaz de programación de aplicaciones.

De esta forma, se pueden construir bases de datos utilizando aplicaciones y sistemas de gestión de bases de datos como Microsoft Access, Oracle, Sybase, MySQL, MSQL o SQL Server, etc. y, por medio de una serie de herramientas de acceso (CGI, DAO, ODBC, etc.) y desde entornos de desarrollo distintos, hacer que estos datos sean accesibles vía Web para cualquier usuario que quiera hacer una consulta en línea.

Acceso a los datos

Hay distintas tecnologías Web que hacen posible el acceso a los datos. Las más destacadas son:

- CGI: (Common Gateway Interface o Interfaz de pasarela común) http://hoohoo.ncsa.uiuc.edu/cgi/ es la especificación de un protocolo que permite al servidor Web (HTTP) comunicarse con programas o *scripts* externos. Los programas CGI trabajan en el servidor Web y pueden implementarse utilizando diferentes lenguajes de programación (COBOL, C, Perl, etc.). Para que el usuario recupere un documento dinámico HTML a través de CGI, generalmente se sigue la siguiente secuencia básica:
  1º. El usuario cumplimenta los campos de un formulario HTML y pulsa el botón de envío. Antes de proceder al mismo, el navegador determina el método HTTP para el envío, identifica los campos del formulario, construye el conjunto de datos como pares: nombre del control / valor asociado y codifica el conjunto de datos.

2º. El navegador realiza una solicitud HTTP al servidor Web, enviando el conjunto de datos del formulario para que sea procesado por el programa especificado en el atributo del formulario action.
3º. El servidor recibe la solicitud y a partir de ella determina que se le está pidiendo la activación de un programa CGI. Se lanza un nuevo proceso CGI que recibe la información necesaria para su ejecución.
4º. El programa CGI se ejecuta procesando la información y devolviendo el resultado al servidor Web.
5º. El servidor recibe el resultado de proceso CGI y prepara una respuesta HTTP válida (anexando alguna cabecera) que se le envía al cliente.
6º. El navegador muestra el resultado recibido que contendrá información dependiente de lo que el usuario introdujo en el formulario HTML.

- ASP: Las primeras soluciones desarrolladas por Microsoft se basaban en el servidor Web ISS (Internet Information Server), el lenguaje de *script* ASP (Active Server Pages) y la tecnología de objetos distribuidos COM (Componente Object Model). ASP proporciona acceso a datos apoyándose en los objetos ADO (ActiveX Data Objects) y ODBC. El uso de la interfaz ODBC le permite a ASP trabajar sobre cualquier sistema gestor de bases de datos que proporcione un controlador o driver (MySQL, SQL Server, Oracle, Informix, etc.). Los objetos ADO, basados en la tecnología COM (Component Object Model), ofrecen métodos que encapsulan el acceso a datos para su utilización en páginas ASP (Connection, RecordSet, Command, etc.). Se puede utilizar ASP sobre un IIS (Internet Information Server) ejecutándose en Windows NT Server 4.0. Se necesita dar de alta un DSN (Data Source Name) que asocia el SGBD (MySQL), el nombre de la fuente de datos y un driver ODBC para MySQL. http://www.asp.net/

- .NET es la última aplicación desarrollada por Microsoft e incluye ASP+, C#, mientras deja de lado las anteriores inversiones de Microsoft en Java (y programas relacionados como Microsoft Visual J++). Todas estas soluciones se basan en estándares propietarios, aunque en la plataforma .NET se incluye soporte a SOAP.

- JSP: El acceso a base de datos desde JSP (Java Server Pages), al igual que desde Servlets, se apoya en la tecnología JDBC de Java. Para ello se precisa un controlador o driver que proporcione el acceso a la base de datos subyacente (MySQL). JSP es un lenguaje muy potente de código abierto que permite crear de manera fácil aplicaciones Web. J2EE (Java 2 Enterprise Edition) es una tecnología de las más utilizadas. A veces se utiliza el término: servidores de aplicaciones Java para referirse a aquellos servidores de aplicaciones que implementan de forma adecuada las soluciones propuestas por J2EE. J2EE es una especificación que propone un estándar para servidores de aplicaciones. Define diferentes tecnologías e indica cómo deben trabajar juntas. Todos los servidores de aplicaciones J2EE deben pasar un test de compatibilidad, que garantiza la correcta implementación de las tecnologías Java. Muchos grandes fabricantes como IBM, Sun Microsystems, Hewlett-Packard, Oracle, Sybase, etc. utilizan J2EE.

- PHP: PHP o Hypertext Preprocessor ofrece interfaces propias de acceso a multitud de fuentes de datos: BBDDs (MySQL, mSQL, Oracle 8, etc.), servidores de directorio (LDAP), texto en XML, etc. Todas ellas están documentadas en la página Web de PHP: http://www.php.net/.

Una **aplicación Web** es entendida como una interfaz de software que permite una serie de funcionalidades como que el usuario pueda interrogar y consultar de forma directa a la base de datos y obtener las referencias o el acceso directo a los recursos o documentos buscados. Algunos SGBD incluyen herramientas de administración que permiten ajustar el rendimiento en función de las necesidades particulares.
Esta es la clasificación dentro de la cual se encuentra la aplicación de carga y administración de Currículos desarrollada para este seminario, CV-DGR.

Figura 19 – Aplicación Web como interfaz

Existen sistemas de gestión de bases de datos tanto de uso libre, como soluciones comerciales de licencia paga y/o restrictiva. Una de las tendencias más claras en la Web actual es integrar el acceso a datos en los servidores de aplicaciones y esto ha conducido a que casi todos los fabricantes de SGBDs comerciales ofrezcan sus propios servidores de aplicaciones que se integran a bajo nivel con los productos de bases de datos de la misma empresa. Como ejemplos, tenemos Sybase Enterprise Server y Oracle Application Server.

El servidor de aplicaciones

Un servidor de aplicaciones no es más que un cambio de nombre para algunos servidores Web de nueva generación que permiten construir aplicaciones. Suelen asociarse con servidores de alto rendimiento pensados para dar servicio a sitios Web con grandes necesidades para gestionar movimientos de datos, afluencia de visitas, atención de transacciones hacia bases de datos, etc.

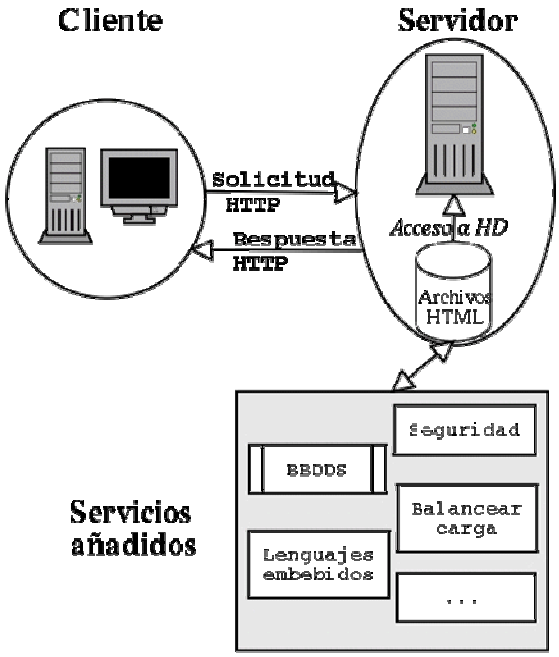


Figura 20 - Arquitectura de funcionamiento de un servidor de aplicaciones.

Un clásico servidor de aplicaciones se apoya en un modelo cliente/servidor de tres capas:

- **Presentación:** una interfaz, generalmente gráfica que reside en los clientes. El ejemplo típico es un navegador.
- **Lógica de negocio:** donde reside el servidor de aplicaciones y el conjunto de programas a los que da soporte.
- **Almacenamiento:** generalmente una base de datos.

Los servicios añadidos a estos servidores de aplicaciones suelen ser:

✓ generación de código HTML ó XML,

- ✓ trabajo con bases de datos y gestión de transacciones,
- ✓ funcionamiento multiproceso para atender a distintas peticiones,
- ✓ establecimiento de distintas sesiones para acceso de usuarios,
- ✓ mecanismos de seguridad y autentificación,
- ✓ monitorización para evitar fallos, etc.

Interfaz o presentación de los datos.
Amén de la atención en el cómo del almacenamiento de los datos, hay que tener en cuenta una cuestión importante, tal es la interfaz de presentación de esos datos. Las interfaces de una aplicación hacia el usuario han ido evolucionando y actualmente se utilizan muchos lenguajes visuales denominados de cuarta generación como son: Visual Fox Pro, Visual Basic, Delphi, etc. Los ambientes Web, se han vuelto una opción viable para las aplicaciones distribuidas en Internet y esto se ha logrado mediante el uso de herramientas como HTML, DHTML y JavaScripts.

Con tecnologías como el *script*ing y DHTML, un desarrollador de aplicaciones puede crear acciones con interfaces de Web funcionales, basadas en la entrada de datos o salida de resultados de búsqueda sin usar controles comunes o applets. Las interfaces de programación denotan el proceso de acceso a una BD y manipulación de los datos, partiendo de la aplicación. El siguiente esquema muestra 4 niveles o interfaces:

Figura 21 - Interfaces de Programación para el acceso a datos.

1. La primera interfaz corresponde a la de Aplicación, la cual abarca y/o corresponde a cada uno de los programas clientes.
2. La Interfaz de Objetos de Acceso a Datos, se encuentra como punto medio entre las aplicaciones y las API's que llegan a ser necesarias para el acceso a las bases de datos. Entre las tecnologías que pertenecen a la Interfaz de Objetos de Acceso de Datos encontramos: DAO (Data Access Objects), ADO (ActiveX Data Objects), RDO (Remote Data Object), RDS (Remote Data Service) y MIDAS (Middle-tier Distributed Application Service). Su función es encapsular los componentes que se encuentran en la interfaz que corresponde a la de API's, con la finalidad de reducir el desarrollo de la aplicación y los costos de mantenimiento y deben situarse en todos los equipos que ejecuten la aplicación, ya que se encuentran casi de manera conjunta con la aplicación.
3. Por su parte, la Interfaz de Programación de Aplicaciones (Application Programming Interface, API), se encarga de mantener el diálogo con la base de datos, para poder llevar a cabo el acceso y manipulación de los datos. Algunos de los componentes que forman parte de esta interfaz son los siguientes: OLE DB, ODBC (Open Database Connectivity), JDBC (Java Data Base Connectivity), ISAPI (Internet Server Application Programming Interface) y CGI (Common Gateway Interface).

   Las API's son una interfaz entre aplicaciones y BD. Ésta tarea se lleva a cabo a veces a través de los clientes y otras a través del servidor de BD.

   Es decir, puede darse que en el cliente residan las tres primeras interfaces o niveles, o que se encuentren las dos últimas en el servidor.

4. La interfaz correspondiente a la base de datos, es donde se encontrará el servidor y toda la información depositada en él.

Para lograr la conectividad necesaria a los datos, que permitirá accesar y manipular la información de una base de datos, es precisa la instalación de ciertas API's o controladores. Las que se describen a continuación, sirven como ejemplo y descripción del proceso de conectividad de datos.

- ODBC (Open Data Base Connectivity): Esta tecnología proporciona una interfaz común para tener acceso a bases de datos SQL heterogéneas. ODBC está basado en SQL (Structured Query Language) como un estándar para tener acceso a datos. ODBC permite la conexión fácil desde varios lenguajes de programación y se utiliza mucho en el entorno Windows.

- CGI (Common Gateway Interface): es una de las soluciones que se está utilizando más para la creación de interfaces Web/DBMS. Entre las ventajas de la programación CGI, destaca la sencillez, ya que es muy fácil de entender, además de ser un lenguaje de programación independiente, ya que los escritos CGI pueden elaborarse en varios lenguajes. También es un estándar para usarse en todos los servidores Web, y funcionar bajo una arquitectura independiente, ya que ha sido creado para trabajar con cualquier arquitectura de servidor Web. Como la aplicación CGI se encuentra funcionando de forma independiente, no pone en peligro al servidor, en cuanto al cumplimiento de todas las tareas que éste se encuentre realizando, o al acceso del estado interno del mismo. Pero el CGI presenta cierta desventaja en su eficiencia, debido al que el servidor Web tiene que cargar el programa CGI y conectar y desconectar con la base de datos cada vez que se recibe una requisición. Además, no existe un registro del estado del servidor, sino que todo hay que hacerlo manualmente.

- ISAPI (Internet Server Application Programming Interface): Es la interfaz propuesta por Microsoft como una alternativa más rápida que el CGI, y está incluida en el Servidor Microsoft Internet Information (IIS). Así como los escritos CGI, los programas escritos usando ISAPI habilitan un usuario remoto para ejecutar un programa, busca información dentro de una base de datos, o intercambia información como otro software localizado en el servidor. Los programas escritos usando la interfaz ISAPI son compilados como bibliotecas de enlace dinámico (DLL - Dinamic Link Library), ya que son cargados por el servidor Web cuando éste se inicia. Dichos programas se vuelven residentes en memoria, por lo que se ejecutan mucho más rápido que las aplicaciones CGI, debido a que requieren menos tiempo de uso de CPU al no iniciar procesos separados. Uno de los programas ISAPI más usados es el HTTPODBC.DLL que se usa para enviar y/o devolver información hacia y desde las bases de datos, a través de ODBC. Además, ISAPI permite realizar un procesamiento previo de la solicitud y uno posterior de la respuesta, con lo cual manipula la solicitud/respuesta HTTP. Los filtros ISAPI pueden utilizarse para aplicaciones tales como autenticación, acceso o apertura de sesión.

- DBI (PERL): Perl es uno de los lenguajes más utilizados para programación en la Web y proporciona su propia interfaz de acceso a datos, llamada DBI (DataBase Interface). Es especialmente utilizado bajo plataformas Linux/Unix, solucionando las complejidades de ODBC en estos sistemas. DBI actúa como una abstracción para un conjunto de módulos DBD (DataBase Driver). Cada módulo DBD actúa como manejador de un sistema gestor de base de datos distinto. Existen módulos para prácticamente cualquier SGBD (Oracle, Informix, MySQL, etc.) y puentes hacia otras tecnologías como ADO, JDBC ...

- JDBC (Java Data Base Connectivity): se trata del estándar para la conectividad entre el lenguaje Java y un amplio rango de sistemas gestores de bases de datos. Los JDBC pueden desenvolverse tanto en un nivel cliente, esto es, trabajando del lado de la aplicación, o en el servidor directamente relacionado con la base de datos. Cuando se encuentre a nivel cliente, trabajará con la tecnología ODBC para acceso a los datos. Hay diversos tipos de controladores JDBC, uno de los primeros fue el puente JDBC-OBDC que implementa un enlace para utilizar un controlador ODBC desde Java.
  Con el tiempo surgieron controladores JDBC específicos para cada base de datos que mejoran el rendimiento del puente JDBC-ODBC, pero no profundizaré sobre ese tema en este trabajo, ya que los mismos no constituyen herramientas del sistema desarrollado.

Conectividad a la DB

Las 2 tecnologías más importantes de conectividad a la base de datos son ADO y JDBC.

ADO
El siguiente esquema[6] muestra 2 de los principales niveles para lograr la comunicación o acceso a la base de datos a través de una aplicación. Dentro del más abstracto se encuentra ADO.

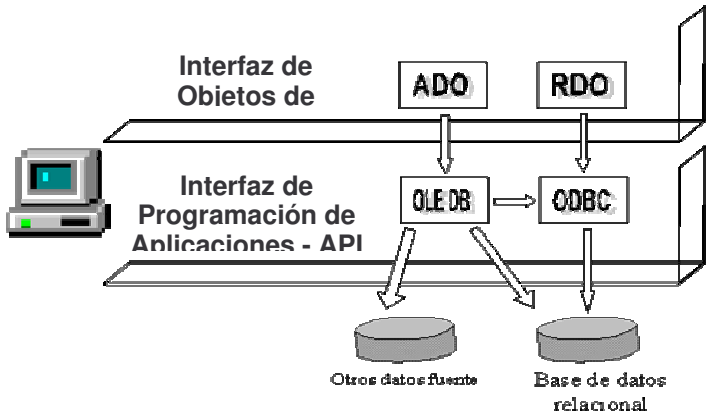


Figura 22 - Niveles de acceso a una BD

Por lo general, las interfaces de objetos de datos son más fáciles de usar que las APIS, aunque las APIs ofrecen más funcionalidades. ADO (ActiveX Data Objects) es la interfaz de objetos de datos para OLE DB, y RDO (Remote Data Objects) es la interfaz para el objeto ODBC.

ADO encapsula el API OLE DB en un modelo objeto simple que reduce el desarrollo, mantenimiento y costo de la aplicación. Es muy fácil de usar, utiliza lenguajes de programación como Visual Basic, Java, C++, VB*Script* y J*Script*, puede accesar datos desde cualquier recurso OLE DB y además, es extensible. Es la interfaz utilizada por Microsoft.

El modelo ADO, basado en el modelo de objetos, define una jerarquía de objetos programables que pueden ser usados por desarrolladores de páginas Web para acceder a la información almacenada en una base de datos. Una jerarquía es un grupo de objetos relacionados que trabajan juntos para un mismo propósito. Por ejemplo, en la Figura 23, cada caja representa un objeto, y cada línea representa una asociación directa entre ellos.

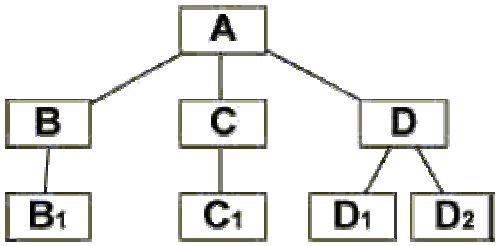


Figura 23 - Jerarquía de Objetos

ADO está compuesto de siete objetos, como se muestra en la *figura IV.g*, algunos de alto nivel como Connection, Command y Recordset, que pueden ser creados y eliminados por el usuario y otros con distintas funcionalidades como designar propiedades de conexión, definir sentencias y ejecutarlas, optimización de consultas, etc.

[6] Taller de Base de Datos.
http://www.itver.edu.mx/comunidad/material/tallerbd/apuntes/index.html

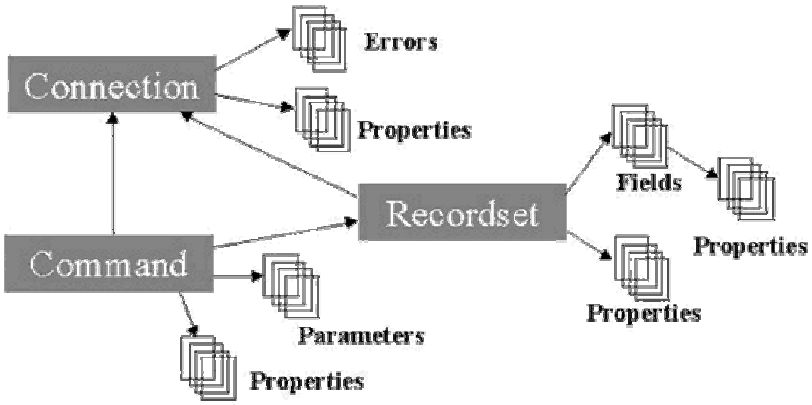


Figura 24 - Jerarquía de Objetos ADO

Cada uno de los objetos anteriores contiene una colección de objetos Property. El objeto Property permite a ADO mostrar dinámicamente las capacidades de un objeto específico.

ADO permite compartir conexiones y esto significa una menor carga de trabajo para el servidor de la base de datos, un tiempo de respuesta más rápida y más accesos a página con éxito.

Existe un componente llamado RDS (Remote Data Service) que ofrece el ambiente de Acceso Universal a Datos, ya sea desde Internet o la World Wide Web, creando un marco de trabajo que permite una interacción fácil y eficiente con los datos fuente OLE DB tanto en Intranets corporativas o en Internet. RDS ofrece la ventaja de obtener por el lado del cliente resultados de datos, actualización y soporte para controles ADO y ofrece el modelo de programación OLE DB/ADO para manipular datos de las aplicaciones del cliente.

JDBC

JDBC o Java Data Base Connectivity, creado por la empresa Sun, es la API estándar de acceso a bases de datos con Java. Sun optó por crear una nueva API en lugar de utilizar ODBC, porque esta última presentaba algunos problemas desde ciertas aplicaciones Java. ODBC es una interfaz escrita en lenguaje C, que al no ser un lenguaje portable, hacía que las aplicaciones Java también perdiesen la portabilidad. Además, ODBC ha de instalarse manualmente en cada máquina, mientras que los controladores (drivers) JDBC que están escritos en Java son automáticamente instalables y portables. Para trabajar con JDBC es necesario tener controladores que permitan acceder a las distintas bases de datos. Sin embargo, ODBC sigue siendo hoy en día la API más popular para acceso a Bases de Datos, por lo que: Sun se ha visto obligada a diseñar un puente que permite utilizar la API de JDBC en combinación con controladores ODBC.

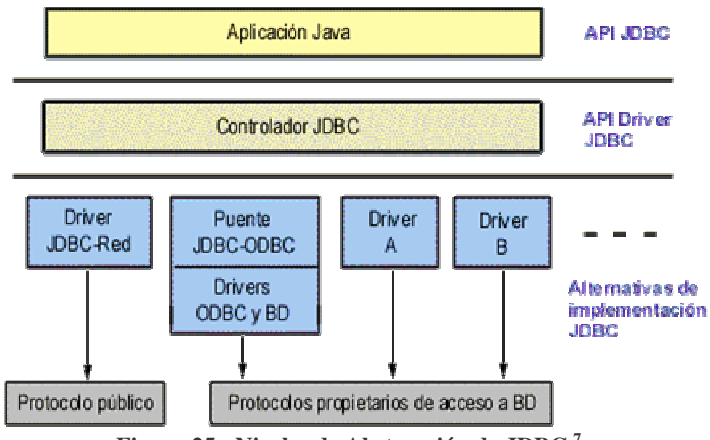


Figura 25 - Niveles de Abstracción de JDBC [7]

[7] Taller de Base de Datos.
http://www.itver.edu.mx/comunidad/material/tallerbd/apuntes/index.html

Las tecnologías que se emplea para la conectividad entre los datos y la aplicación, se ha convertido en un factor muy importante a la hora de desarrollar un proyecto Web que cuente con funcionalidad de acceso a datos. A continuación se muestra un cuadro comparativo de las dos tecnologías más importantes en este sentido: *ActiveX Data Objects (ADO) y Java Data Base Connectivity (JDBC)*[8].

ADO	JDBC
1. Tecnología elaborada por Microsoft 2. Tiene la principal función de realizar la solicitud de los datos a la base de datos. 3. Esta solicitud la realizará mediante la tecnología OLE DB, la cual estará en contacto de manera directa con la base de datos. 4. La tecnología OLE DB sólo se empleará cuando el DBMS pertenece de igual manera a Microsoft, como es SQL Server. 5. ADO encapsulará a ciertos objetos de OLE DB, para que de ésta manera se realice la conexión con la base de datos. 6. Para realizar la gestión de acceso a bases de datos heterogéneas por parte de ADO, éste hará uso de ciertos objetos de la tecnología RDO (Remote Data Objects). 7. RDO dependerá de los ODBC's para poder efectuar la conexión a la base de datos y con esto el acceso a la información. 8. ADO podrá encontrarse trabajando en una página Web en conjunto con código HTML; esto será posible mediante un mecanismo de introducción de instrucciones como es el VB*script*. 9. Los objetos que conforman al ADO, no son compatibles con otros lenguajes, solo por aquellos que pertenecen a la empresa Microsoft como son: Visual C++, Visual Basic, Visual Java, etc.	1. Tecnología hecha por Sun Microsistems. 2. Tiene la función de ser un gestor para la aplicación con respecto a la base de datos. 3. Por primera vez el JDBC fue empleado, tomando como intermediario entre él y la base de datos al ODBC. 4. Como modelo cliente/servidor, el JDBC se encontrará trabajando en el equipo cliente, conectándose directamente con la base de datos. 5. Como modelo de tres capas, el JDBC se encontrará en una capa intermedia, donde todos los usuarios pasarán por él para poder accesar a la base de datos. 6. Existen módulos JDBC que son propios de los fabricantes de DBMS, que son utilizados para el rápido acceso a la información de las bases de datos de los mismos. 7. JDBC no se encontrará ligado a trabajar con alguna tecnología en específica, ya que se elaboró con la finalidad de ser portable. 8. En aplicaciones Web, JDBC se encontrará laborando en conjunto con código HTML, mediante el mecanismo del Java *script*. 9. JDBC se elaboró con la finalidad de poder ser compatible y portable para poder ser empleado en aplicaciones y para la conexión con bases de datos.

Por último, hay que destacar también una tecnología llamada Web DB utilizada por algunos servidores de bases de datos, con la cual, un usuario puede solicitar la información que requiera y visualizarla a modo de respuesta en una página Web, que será creada y elaborada por el propio servidor de base de datos.

El proceso que comprende desde la solicitud a la visualización de la información, puede ser representado de la siguiente manera:

[8] Fuente: Taller de Base de Datos.
http://www.itver.edu.mx/comunidad/material/tallerbd/apuntes/index.html

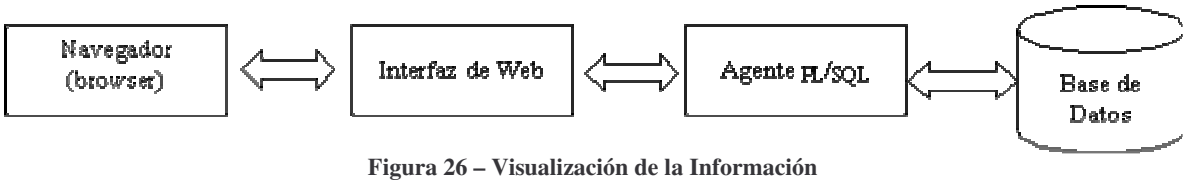


Figura 26 – Visualización de la Información

- Navegador (browser): es la aplicación mediante la cual, se tiene acceso libre a los servicios de Internet, y el medio que permite al usuario introducir la solicitud para visualizar la información, empleando el URL para especificar detalladamente el proceso que se desea ejecutar.
- Interfaz de Web: proporciona una interfaz para que un programa que se ejecute en el servidor genere como salida el código HTML, en lugar de leer simplemente un archivo estático de texto. Con ésta interfaz se podrán crear las páginas Web de forma dinámica y/o utilizar la implementación de formularios HTML. Esta interfaz permite tecnologías como los CGI's o aquellas otras que son propias del servidor de base de datos.
- Agente PL/SQL: es el eslabón final del proceso entre un navegador cliente y el servidor de base de datos. El agente ejecutará una llamada a un procedimiento almacenado en el servidor. Este procedimiento creará una página HTML dinámica como salida, y el agente devolverá dicha salida al cliente a través del navegador empleando de igual manera la Interfaz de Web.
- Base de Datos (BD). En ella se mantendrá almacenada la información; se encargará de proporcionar los datos que le hayan solicitado previamente, al momento de la ejecución de un procedimiento por parte del Agente PL/SQL.

CAPÍTULO IV
EL PROYECTO

Consideraciones Preliminares

Los requerimientos fueron una variable muy dinámica a lo largo del desarrollo de todo el SIARH y el módulo de Currículos no quedó ajeno a esa circunstancia.

Las instancias previas al desarrollo de este sistema en sí abarcaron tareas que iban desde la presentación del anteproyecto, hasta la capacitación en el uso de la herramienta de Carga de CV, y de manera un poco más informal la capacitación sobre la preselección de postulantes con CV registrado.

Más allá de lo que se haya planteado en la definición de un anteproyecto, a la hora de dar forma al proyecto hay consideraciones que de una u otra manera marcan el camino a seguir para poder ejecutar las tareas previstas. Incluso, esas mismas consideraciones, modifican las definiciones previas debido a situaciones que, aún cuando se hayan analizado, siempre se plantean de un modo 'ideal' y luego deben ser confrontadas con la realidad indefectiblemente.

Este es el caso del gestor de Base de Datos. En el anteproyecto presentado para este seminario se especificaba como SGBD elegido MySQL, pero luego fue planteada la necesidad de desarrollar el módulo de carga de CV también para uso de los agentes de la DGR dentro del Sistema de Administración de RRHH y como prioridad en el tiempo. Ese cambio de criterio sobre el entorno en el que debía funcionar el sistema ocasionó que el modelado de los datos se viera ineludiblemente condicionado por el DER del sistema de control de registraciones para los nuevos relojes de proximidad, LIKAY de Equiser SRL.

El SGBD utilizado para LIKAY es MSSql, por lo tanto el CV-DGR debía usar ese gestor también para poder definir sin inconvenientes las relaciones entre las tablas de la BD de LIKAY y la 'extensión' de la misma que serviría para el almacenamiento de los currículos.

Inicialmente el proyecto de un sistema que permitiese la carga de CVs fue concebido desde la necesidad de contar con una herramienta accesible desde la Web a través de un link publicado en el Portal de la Dirección General de Rentas, como si se tratase de una opción más del menú principal. El mismo representaría no sólo una sistematización de la recepción de currículos sino que además sería un punto de contacto más de la Dirección con la comunidad.

Otro de los aspectos atendidos en la gestión fue si se utilizaría un equipo diferente al servidor de BD para funcionar como servidor de aplicaciones o se instalaría todo en el mismo equipo. La sugerencia, por mi parte, era configurar un servidor de aplicaciones en otro equipo que también funcionara como servidor de Internet, de manera que en ese mismo equipo podría configurar un firewall que protegiera la aplicación y dejaría fuera del alcance de posibles ataques, por Internet, a la BD. Pero esto no se hizo así por falta de recursos (o mejor dicho criterios en la distribución de los mismos), y tanto el SGBD como la aplicación corren en el mismo equipo.

Así podría enumerar un sin fin de aspectos que se trataron y continúan definiendo en la gestión del proyecto, ya que la misma sigue su curso mientras el sistema se encuentre en uso, pero los más destacables ya fueron mencionados.

En resumidas cuentas podría decirse que la solución propuesta constituye un sistema dual. En sí el módulo de carga de CV es uno sólo, pero dada la disparidad de contextos en los que se deseaba implementar, éste tuvo que replicarse de forma tal que se obtuvieron dos versiones del mismo:

- uno que funcionase en un ambiente Web y
- otro que funcionase como parte integrante del Sistema concebido en el proyecto SIARH, mismo ambiente donde debería implementarse el módulo de Administración de Currículos.

En el capítulo siguiente describiré brevemente en qué consiste el SIARH – Sistema Ingral de Administración de Recursos Humanos, me extenderé acerca de los módulos de Carga y Administración de CV que se desarrollaron para ser implementados en el entorno de Intranet de la DGR, y finalmente comentaré las variaciones que se introdujeron en el módulo de Carga de CVs para la Web, ya que la base de éste módulo es igual a la del que funciona como parte del SIARH.

En esta sección mi intención es guiar a los lectores por una vista de lo que fue la gestión del proyecto, identificar las etapas en que se llevó a cabo su desarrollo y comentar el aprendizaje experimentado en lo que a organización de un trabajo de desarrollo de software de este tipo se refiere.

Etapas del Proceso de desarrollo

Para el desarrollo de este trabajo me guié por una metodología simple basada en 4 etapas, claramente influidas por el modelo del Proceso unificado de Desarrollo de Software aunque sin llegar a serlo completamente:

ETAPA I - Análisis.
En esta etapa se identificó el problema, se planteó una idea de solución y se realizó el análisis que posibilitaría la definición del proyecto. La mayor parte de la información se obtuvo mediante un proceso de investigación por diferentes medios: entrevistas personales a los responsables del área y los agentes ejecutores de los procesos que se deseaban automatizar, recopilación de todos los formularios y documentos tipo que administraba RRHH para el registro de la información relacionada con la selección de personal, y a modo de asistencia ilustrativa se hicieron breves análisis verbales sobre sistemas conocidos de este mismo tipo publicados en Internet.

Una tarea muy importante que se llevó a cabo en esta primera instancia fue el estudio de las estructuras de datos utilizadas en el sistema de control de las *registraciones* llamado LIKAY, desarrollado y provisto por la empresa tucumana Equiser S.R.L., ya que en base a esa estructura se debía definir el modelo de datos para el CV-DGR de modo tal que los agentes ya pertenecientes a la organización también cargasen su CV. Esta fue una de las modificaciones en los requerimientos que se impuso como guía para replantear la estructuración del proyecto.

ETAPA II – Diseño.
En general el diseño de todo el SIARH fue tomando forma casi paralelamente a su desarrollo. Es decir, el único modelo definido para comenzar el desarrollo estaba dado por la estructura a la que debían corresponder los *scripts* de las distintas páginas de carga o consulta de datos, la jerarquía de usuarios, y el DER.

Una aplicación como el CV-DGR, que no es más que una porción de otra de mayor envergadura y complejidad, debe ser planificada cuidando detalles cómo parámetros de entrada, formato de la salida, punto de acceso a la misma y visibilidad respecto de la jerarquía de usuarios definida.

De igual manera que en el SIARH, la tabla de datos de la entidad Empleados era el único punto en común entre la BD provista con LIKAY y la de nuestra aplicación. Esto se definió así no sólo por una cuestión de cuidado de aquellas entidades que forman parte del DER del sistema de control de las *registraciones*, sino por una decisión de “atacar el problema” por medio de una subdivisión conceptual que involucrara la menor cantidad posible de entidades en la

intersección entre subsistemas.

La subdivisión conceptual del SIARH facilitó el control sobre la integridad de los datos y el mantenimiento del sistema. A su vez de este modo, el tratamiento de los datos propios del módulo de currículos no afectaría en absoluto al contenido de la tabla de empleados. Dado que los mismos forman parte del legajo tampoco debían duplicarse y por ello su tratamiento se haría únicamente desde el LIKAY y la interfaz de carga del CV de un empleado debía mostrarse en campos o *inputs* no editables.

En el Capítulo V se muestra en detalle el diseño de la BD y el resultado de estas consideraciones que se traducen automáticamente en el esquema de funcionamiento que sigue el *script* de la página de carga de CV.

ETAPA III - Desarrollo.
En esta instancia se hizo el planteo del modelo de una solución tentativa, que resulta al conjuntar las tecnologías seleccionadas de la etapa anterior y de las necesidades detectadas en la definición de requisitos del sistema.

La mayor atención en esta etapa fue para el diseño de la BD, ya que la misma debía modelarse teniendo presentes dos premisas:
1º. La tabla que representaba la entidad 'Empleados' debía ser ampliada de manera tal que contuviese los campos necesarios para almacenar información requerida en un CV y a su vez debían mantenerse intactos tanto los campos ya incluidos como los procedimientos almacenados, que afectaban directamente el funcionamiento de LIKAY.
2º. El registro de empleados en LIKAY ya involucraba datos personales, fotografía y demás por lo que la tarea de diseño consistía más bien en una ampliación de la BD de este sistema volcando de la manera más independiente posible los datos de tipo académico, experiencia laboral previa y demás.

El diseño del sistema en sí se simplificó en la concepción del formulario de carga de datos como una sola página con subformularios desplegables para cada sección temática de llenado. Luego debía respetarse el esquema de autenticación de usuarios definido para todo el SIARH y por último traducir muy claramente los parámetros y las reglas que normalizarían la preselección de los CV. Se consideró suficiente la generación del reporte resultante de la consulta de preselección en una página dinámica, la cual podría imprimirse directamente desde el Web Browser, pero luego se hizo la oferta de incorporar reportes en formato PDF y la misma fue muy bien recibida.

El diseño gráfico se simplificó con el uso de una hoja de estilos guiada por la paleta de los colores institucionales de gobierno y el gris que se traducen en una imagen sobria, el uso de imágenes es escaso por no decir nulo por expreso pedido del Supervisor del área y por tratarse de un sistema en línea y no de una Web, donde la información es de carácter más general y debe apoyarse con imágenes.

ETAPA IV - Implementación.
Finalmente, llegamos a la aplicación de la solución propuesta. La implementación se hizo sólo para el módulo de carga de CV y el de administración de CVs del SIARH. El módulo que se desarrolló para uso general de los postulantes vía Internet no se está utilizando debido a que aún no se pudieron cerrar las negociaciones de un espacio público disponible en los servidores de Gobierno. En este aspecto se puso de manifiesto claramente cuán importante es hacer una buena gestión, aún cuando consuma tiempo considerable, desde los comienzos de un proyecto para que el producto del mismo llegue a "ver la luz" al final de su desarrollo.

La supervisión de RRHH consideró más provechosa la liberación del sistema por grupos dependiendo de los roles de los usuarios siguiendo la siguiente secuencia:
1º. Operadores administrativos del área de RRHH.
2º. Director y Jefes de Subprograma.

3º. Personal de la DGR en general.

En el mismo servidor de RRHH donde funcionan LIKAY y MSSQL Server con la base de datos en ambiente de producción se instalaron Apache, PHP y el SIARH, que incluye el módulo CV-DGR. A su vez las modificaciones e incrementos continuan aún desarrollándose y probándose sobre un servidor de prueba de idéntica configuración al de RRHH.

Para publicar el sistema de carga de CVs en la Web hubiésemos tenido que definir un nombre de dominio y demás cuestiones de acceso y seguridad. Pero en este caso, dado que este módulo no sería liberado de manera 'independiente' en la Web, este aspecto no era una definición que nos preocupase para su implementación. Sólo debíamos ajustar las configuraciones necesarias para plantear correctamente la relación de dependencia con la página desde la cual se accedería al mismo y nada más.

Como corolario de la implementación del sistema se hizo una presentación instructiva del mismo donde se entregó los manuales que confeccioné a los usuarios. Esta presentación funcionó también como una primera capacitación. La comunicación del sistema aún no terminó de completarse porque todavía se está observando el impacto que ejercen los usuarios de la segunda etapa en la secuencia de implementación.

CAPÍTULO V
LA APLICACIÓN CV-DGR

Breve descripción del SIARH

El SIARH es un proyecto desarrollado para brindar una solución integral a la administración de RRHH que no contaba con una herramienta que se ajuste no sólo a las necesidades de gestión del área sino que además fuese fácilmente adaptable a la dinámica y casi carente de reglas topología de la red de la DGR.

SIARH son las siglas para Sistema de Administración de Recursos Humanos. Este sistema es de tipo intracorporativo, es decir que funciona únicamente dentro de la red de la dirección.

La necesidad de un sistema que integre las tareas de carga de novedades diarias de personal, la emisión de reportes, solicitud de licencias y demás procesos específicos de la administración de RRHH fue planteada en un principio como un aspecto independiente a toda otra herramienta existente, justamente con un espíritu de renovación tecnológica. Tal necesidad se tornó imperativa justo inmediatamente después de presentado el anteproyecto del sistema por una avería en los relojes de registro de ingresos y egresos del personal.

La dirección adquirió nuevos relojes para esta función pero una generación más moderna de los mismos, se llaman relojes de proximidad porque toman la identificación del empleado de tarjetas magnéticas a través de sensores de corto alcance. Junto con estos equipos la empresa proveedora, Equiser S.R.L. de Tucumán, le facilitó a Rentas un sistema de control de los movimientos registrados por los relojes llamado LIKAY.

Este sistema refleja todos los datos y movimientos del personal en una BD generada en MS SQL Server. Por este motivo y dado que todavía no habíamos avanzado en el desarrollo del proyecto SIARH, reconsideré el SGBD elegido y diseñé el sistema para poder utilizar las entidades principales ya consideradas dentro del LIKAY como por ejemplo la de Empleados y la de Registraciones (donde se asienta cada una de las “marcas” de tarjeta del empleado).

El DER del SIARH acusa 74 tablas entre las que se incluyen la tabla de Empleados y Registraciones ambas de uso común al LIKAY y sin considerar tablas de sistema del SQL Server. No es oportuno describir en detalle las entidades que lo integran, si lo será en mi próximo trabajo de Seminario de Sistemas. Sólo especificaré que las tablas de Empleados y Registraciones almacenan los datos personales y de categorización dentro de la estructura del cuadro de Cargos de la DGR de la Provincia, y de “marcas” de entradas o salidas de empleados según un estado de “espera” particular para cada situación de los empleados.

El SIARH controla el acceso y la visión que tendrán los usuarios por medio de sesiones. Cada agente de la Dirección posee un usuario y contraseña que le permitirá ingresar al sistema y a su vez, por formar parte de una jerarquía, consultar datos sesgados por los permisos que posea. Para ingresar al SIARH es necesario especificar la dirección “http://ruta_del_server/siarh” y se abrirá la página de autenticación de usuarios ya que Apache fue configurado para abrir por defecto la página ‘login.php’.

Figura 27 – Pantalla de registro en SIARH

Dependiendo del permiso de su usuario el agente que ingresa al SIARH ve más o menos opciones en el menú principal ubicado en el panel izquierdo de portal. En ese menú del sistema, desarrollado en JavaScript, pueden visualizarse todas las opciones con que cuenta el usuario logueado, en este caso un 'Administrador':

Figura 28 – Vista de un formulario de ABM de Descripciones de Puestos

Como puede observarse en la figura de pantalla, las opciones están agrupadas dentro de submenús desplegables. A continuación se detallan las funcionalidades **propias del sistema CV-DGR** y roles que tienen acceso a ellas:

⊞ Personal		
Puestos	Agrega nombres de puestos que formarán parte de la estructura organizativa con aval de resolución normativa.	'operrh', 'jeferh' y 'superuser'

Descripciones	Consta de 3 secciones, altas, modificaciones y reportes. Define aspectos propios de un puesto independientemente del agente que lo ocupe. Guiado por la organización de un manual de calidad.	'operrh', 'jeferh' y 'superuser'
Cargos	Consta de 3 secciones, altas, modificaciones y vista del cuadro de cargos. Establece una relación inyectiva entre agentes y puestos que define al cuadro de cargos.	'operrh', 'jeferh' y 'superuser'
Solicitud Vacantes	Consta de 4 secciones, alta de solicitudes de búsquedas para cubrir vacantes, modificaciones y vista de solicitudes aprobadas y rechazadas.	'operrh', 'jeferh' y 'superuser'
Empleados	Consta de 3 secciones, altas, modificaciones y vista parametrizada de empleados activos o inactivos. No registra datos personales pero sí los de vinculación al cargo que ocupan.	'operrh', 'jeferh' y 'superuser'
Familiares	Consta de 3 secciones, altas, modificaciones y vista de los familiares registrados por empleado.	'operrh', 'jeferh' y 'superuser'

⊞ Licencias		
Solicitudes	Consta de 2 secciones, altas y modificaciones. Sólo se pueden modificar aquellas solicitudes que aún no fueron aprobadas.	'operrh', 'jeferh' y 'superuser'
Compensatorias	Consta de 2 secciones, altas y modificaciones. Registra solicitudes de autorización para realizar tareas fuera del horario normal de trabajo.	'operrh', 'jeferh' y 'superuser'
Consultas	Muestra un calendario con los días de licencia solicitados y aprobados resaltados en color. Los tipos de licencia están identificados por color.	'operrh', 'jeferh' y 'superuser'

⊞ Currículo		
Registrar CV	Carga la información curricular de un agente, organizada en secciones. Permite modificaciones en el mismo formulario.	TODOS
Reporte	Muestra una impresión del CV del agente logueado si es un usuario con permiso de 'user', o del agente seleccionado por un 'superuser', 'jeferh', 'dire' o 'jefedgr' y brinda la opción de generar un archivo PDF con la información del CV del mismo.	TODOS

⊞ Búsquedas		
Preselección	Filtra los CV de postulantes que se ajusten a los criterios especificados para la cobertura de una vacante. Muestra una vista del resultado y ofrece la opción de imprimir un reporte en formato PDF.	'operrh', 'jeferh' y 'superuser'

⊞ Consultas		
Cursos	Muestra el cronograma de cursos planificados por la DGR dentro del programa de capacitación aprobado.	TODOS
Legajo	Muestra una vista del legajo del empleado logueado o del seleccionado por un usuario de mayor jerarquía.	TODOS
Licencias	Muestra un calendario con los días de licencia solicitados dentro del año elegido que hayan sido aprobados. Los tipos de licencia están identificados por color.	TODOS
Asistencias	Muestra un reporte sobre las asistencias computadas discriminadas por edificio al que se encuentran asignadas las tareas de los agentes.	TODOS

Reportes		
Puestos	Informe sobre la nomenclatura y ubicación jerárquica de los puestos que conforman el organigrama de la DGR. El resultado se obtiene en un archivo de extensión PDF.	'operrh', 'jeferh' y 'superuser'
Desc. Puestos	Informe con la descripción de todos los ítems especificado para el puesto especificado. El resultado se obtiene en un archivo de extensión PDF.	'operrh', 'jeferh' y 'superuser'
Cupos-Vacantes	Informe sobre la cantidad de personal asignado a un área específica de la DGR y el número de puestos vacantes generados por desvinculación o aprobación de un cupo mayor.	'operrh', 'jeferh' y 'superuser'
Lic. Solicitadas	Informe sobre las licencias solicitadas y otorgadas por empleado o por sector en un período de tiempo especificado.	'operrh', 'jeferh' y 'superuser'
Lic. Médicas	Informe sobre las licencias por enfermedad, solicitadas y otorgadas por empleado o por sector en un período de tiempo especificado.	'operrh', 'jeferh' y 'superuser'
Compensatorias	Informe sobre la cantidad de horas compensatorias (extras) realizadas acumuladas por empleado o por sector, según se especifique.	'operrh', 'jeferh' y 'superuser'
Familiares	Informe con la referencia de todos los familiares de el/los empleados seleccionados. El resultado se obtiene en un archivo de extensión PDF.	'operrh', 'jeferh' y 'superuser'

Usuarios		
ABM Usuarios	Modifica, deshabilita o da de alta a los usuarios del sistema a partir de la tabla de empleados. Sólo se consideran aquellos que no hayan sido dados de baja.	'superuser'

Sesión		
Salir del sistema	Cierra la sesión activa. Es una regla que aporta seguridad en la operación del sistema.	TODOS
Cambiar Clave	Modifica la clave del usuario registrado, reemplazando la anterior y la almacena en la base de datos encriptada.	TODOS

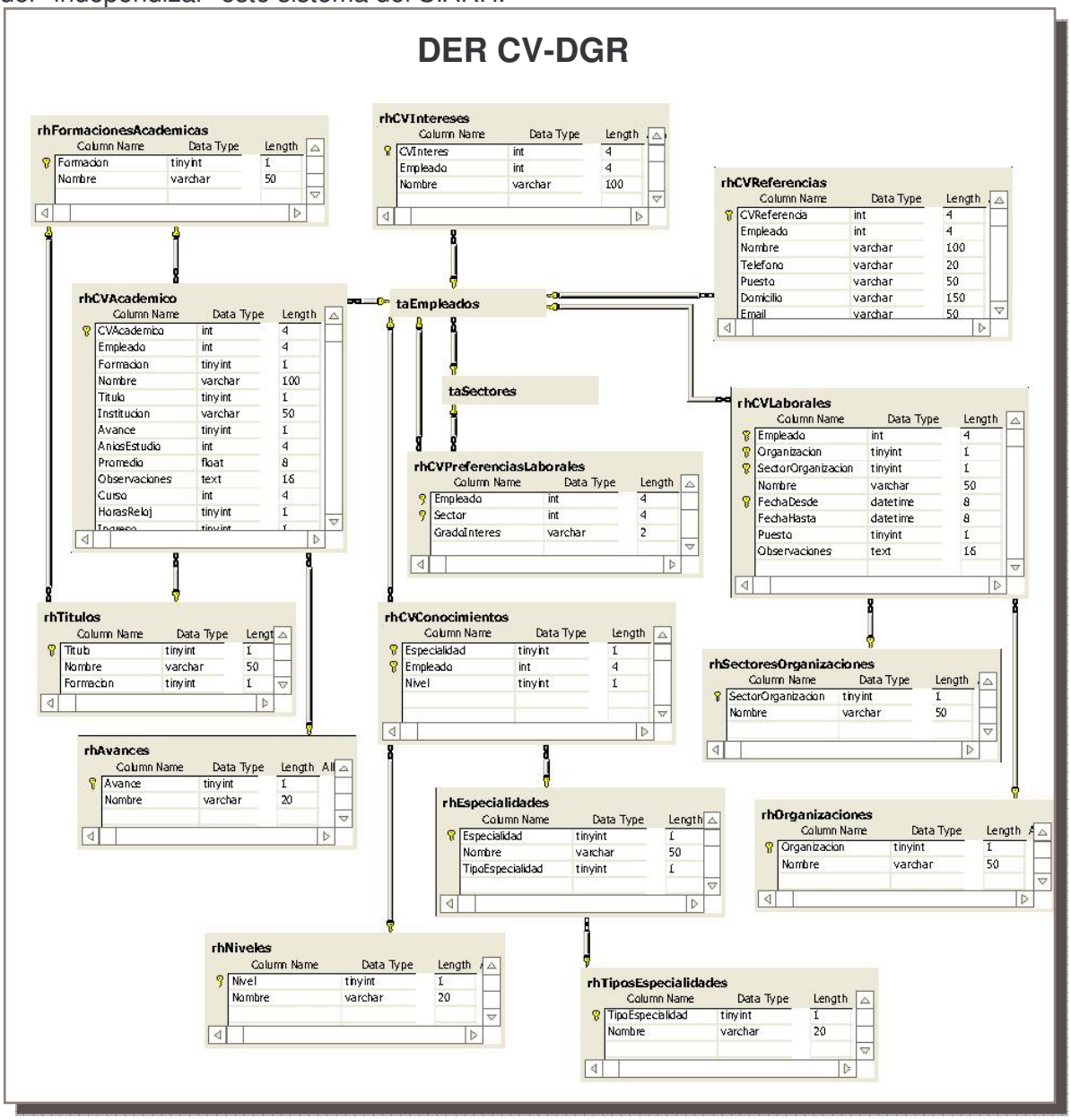
Sobre SIARH		
Acerca del SIARH	Muestra información sobre el entorno de ejecución y el desarrollo del sistema.	TODOS

Como puede verse en la figura el SIARH cuenta con muchas opciones más como la generación de reportes, todos de generación dinámica en formato PDF, la gestión de licencias, desde la solicitud hasta la autorización de las mismas, que impacta directamente en el estado de espera de "marca" de tarjetas en los relojes, etc. Todas ellas son ajenas al tema de este seminario por eso no ahondaré en ellas.

Sólo resta decir que SIARH es un sistema que integra un gran número de procesos que antes se encontraban dispersos en distintos sistemas o se realizaban manualmente con la asistencia de simples aplicaciones de oficina, es decir de manera casi artesanal. En la actualidad el SIARH está funcionando y ha acotado sensiblemente los tiempos de respuesta en procesos vitales como por ejemplo la liquidación de sueldos, por identificar alguno. Además están analizándose incorporaciones de nuevos módulo para cubrir funcionalidades específicas de otras áreas.

DER del CV-DGR

Como ya se introdujo anteriormente, el DER del sistema de carga de CV del SIARH no es otra cosa que una porción del DER de todo el SIARH. Esta vista reducida es completa y suficiente para soportar el funcionamiento del CV-DGR y basta reemplazar la tabla taEmpleados para poder “independizar” este sistema del SIARH.

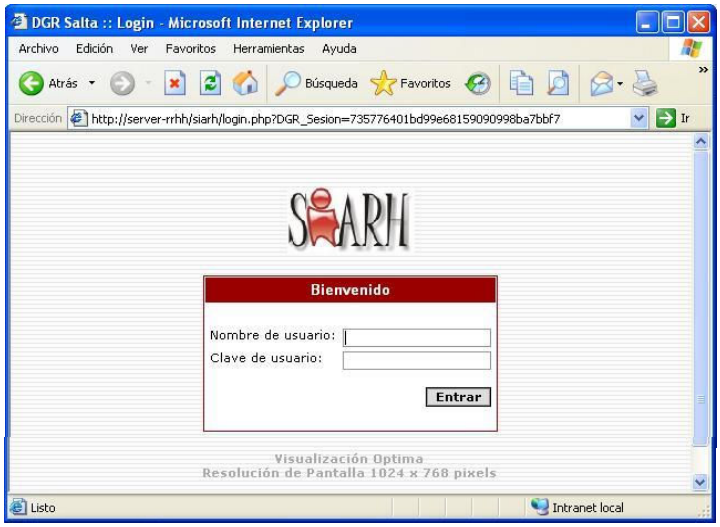


Módulo de Carga de CVs

Este es el módulo a través del cual un agente de la DGR puede ingresar su historia académica, experiencia laboral, habilidades adquiridas, etc. Al ser un módulo incluido dentro del SIARH, éste controla el acceso de un usuario registrado y los permisos que posee.

Como ya quedó especificado en el apartado anterior, el acceso a este módulo está dispuesto en la opción Registrar CV dentro de Currículo en el menú principal, tal como se remarca en la Fig. . A través del formulario de carga de CVs el usuario podrá registrar datos de un nuevo

currículo o modificar los existentes. El registro del usuario se realiza desde la pantalla de inicio cual es por defecto login.php:



Los usuarios en general, cualquiera sean los permisos que posean, verán sus datos personales en campos no editables. Se tomo la decisión de hacerlo así porque estos datos forman parte del *Legajo* del empelado y por ello sólo pueden modificarse vía notificación al área de RRHH.

- **Registrar CV**

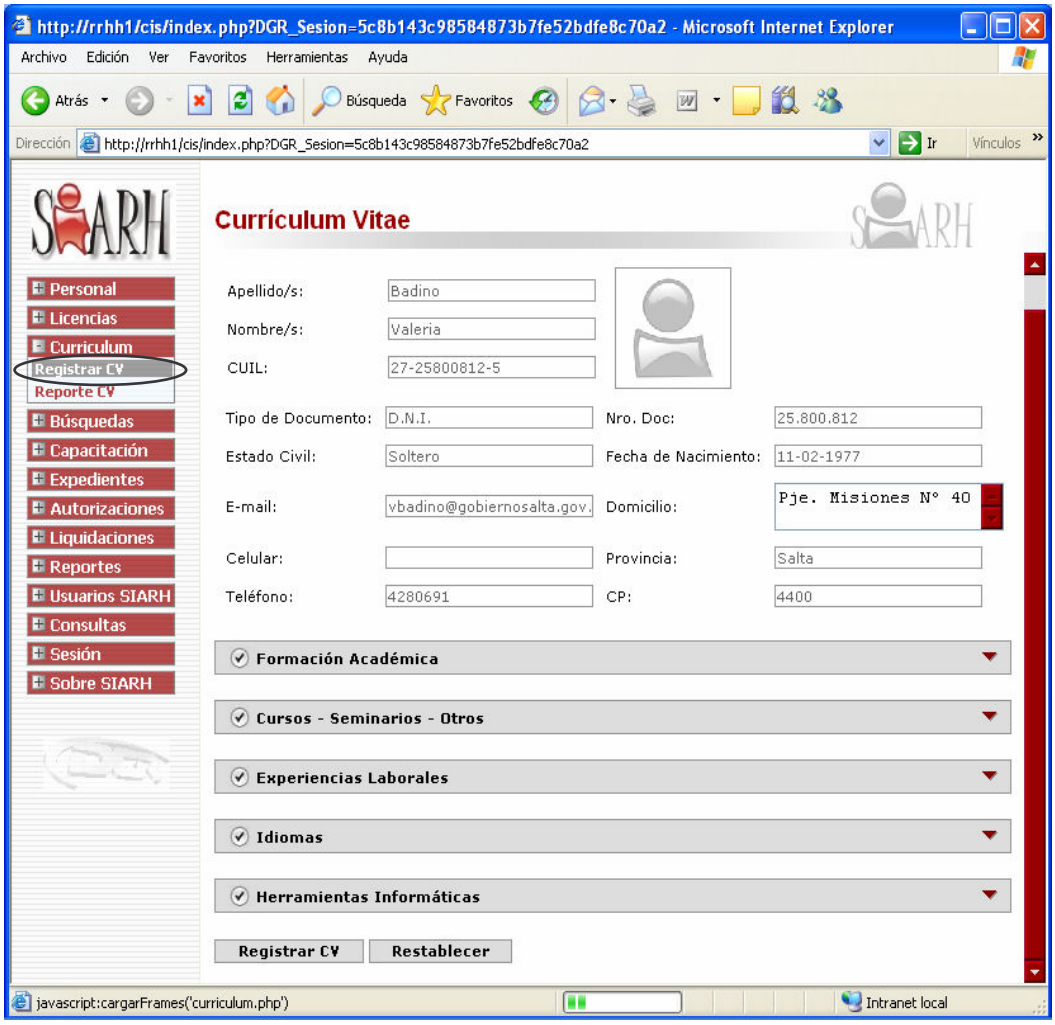
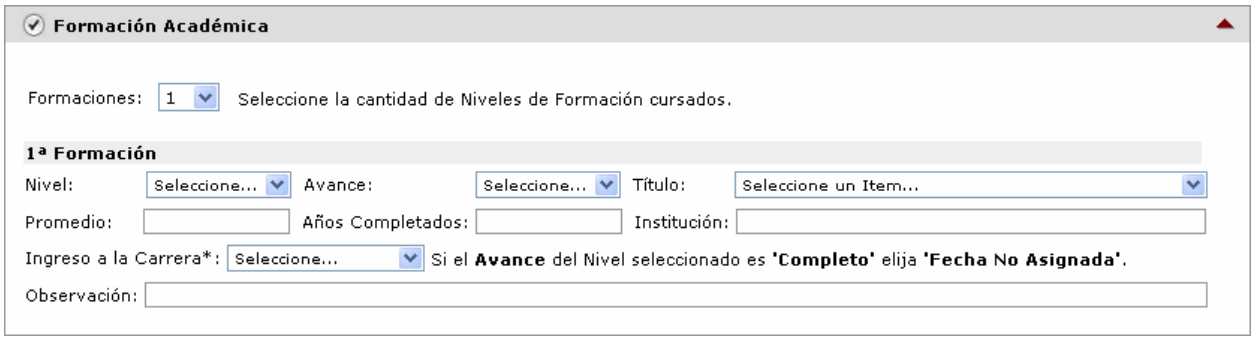


Figura 29 – Formulario de Carga de CV en SIARH

La información del CV está clasificada en 5 secciones aparte de los datos personales:
1. Formación Académica
2. Cursos – Seminarios – Otros
3. Experiencias Laborales
4. Idiomas
5. Herramientas Informáticas.

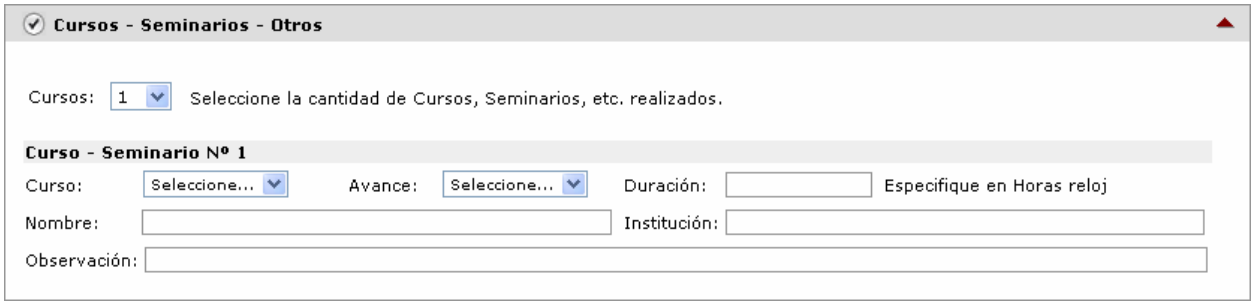
1. Formación Académica

Aquí se solicita al usuario que ingrese, tantas veces como niveles de formación desee cargar, el grado alcanzado, titulo, años de estudio, avance en el estudio, promedio y año de ingreso a la carrera por medio de controles que pueden ser entradas de texto o listas de selección.



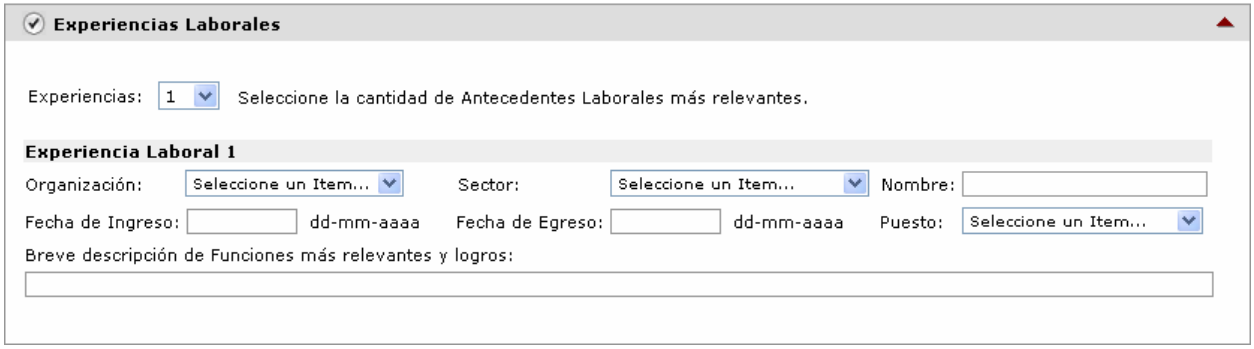
2. Cursos – Seminarios – Otros

En esta sección también debe especificarse la cantidad de cursos o seminarios, jornadas y demás capacitaciones que van a registrarse. Seguidamente se recarga el formulario y como primer paso se debe especificar a qué categoría extracurricular se referirá la capacitación a ingresar.



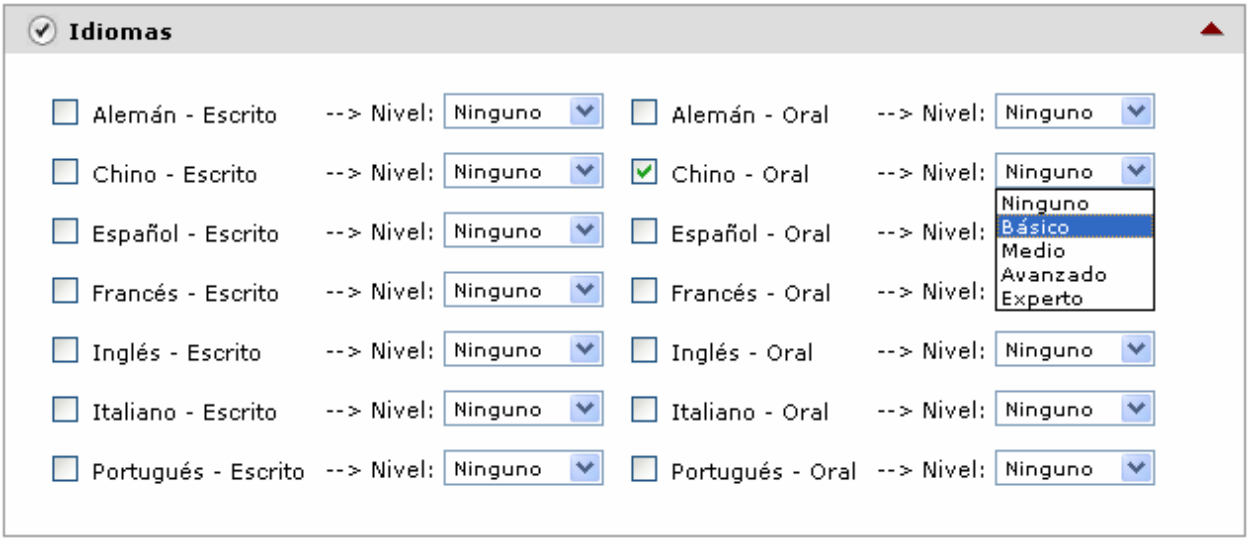
3. Experiencias Laborales

Los antecedentes laborales son importantes a la hora de, evaluar la experiencia que se define como necesaria para la cobertura de una vacante, y si el responsable de la búsqueda lo considera necesario solicitar referencias. Los datos a ingresar son organización, sector de la Industria, puesto de desempeño de funciones, fechas de ingreso y egreso al puesto, etc.



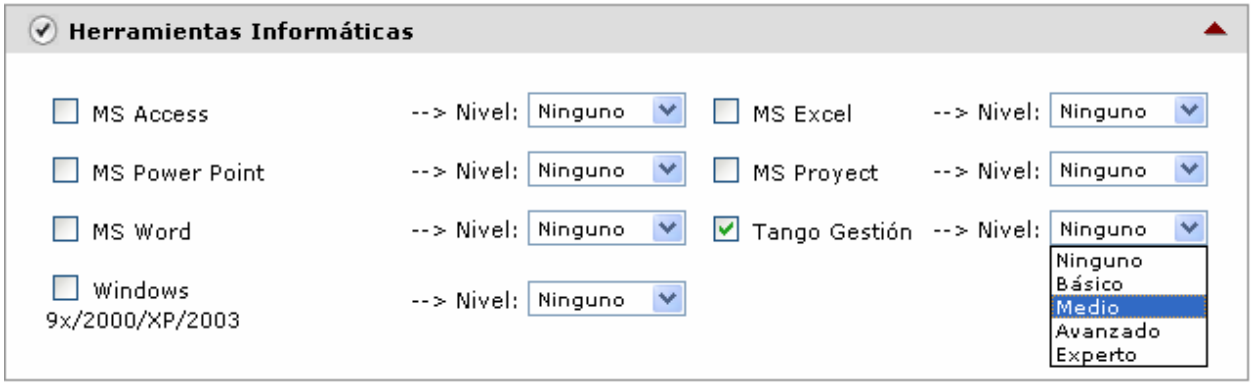
4. Idiomas

Esta es una especificación de las habilidades adquiridas en cuanto al manejo de lenguas extranjeras. La forma de ingreso de estos ítems es a través de casillas de selección y listas desplegables para el nivel adquirido.



5. Herramientas Informáticas.

La mecánica para el ingreso de estos datos es idéntica a la de la sección anterior. Las habilidades en este caso se refieren al manejo de procesadores de texto, planillas de cálculo y demás herramientas informáticas de uso imprescindible en las oficinas de cualquier organización.



Una vez completados todos los datos que se quieran incorporar al CV se debe accionar el botón 'Registrar CV' que activará una secuencia de controles y transacciones que desembocan en la inserción de los datos a las tablas correspondientes de la BD.

Teniendo presente que los datos personales no se contemplan en este formulario, para no generar una falsa sospecha de omisión, los pasos que sigue el script para registrar el CV en la BD se reducen a la porción de código que se extrae a continuación y que forma parte de la definición de los casos que se presentan para la variable *$accion* cuando un usuario acciona algún botón del formulario:

```
switch ($GLOBALS['accion']) {

 case 'Registrar '.$titulo:

// Verifica si se eligió al menos una Formación Académica para cargar
   if ($GLOBALS['tot'] > 0) {
     $nulos = 0;
```

```
     $c = 1;
     while ($c <= $tot && $nulos == 0) {
      if ($GLOBALS['formacion'.$c] == '-' || $GLOBALS['avance'.$c] == '-' ||
↪$GLOBALS['tit'.$c] == '-' || $GLOBALS['anio'.$c] == '') {
        $nulos = $c;
       } else {
        if ($GLOBALS['avance'.$c] == 3 && $GLOBALS['anio'.$c] <> 26) {
           $GLOBALS['msg'] = 'NO se grabó su CV. Verifique que el \nAño de Ingreso a
↪la Carrera respete la condición indicada.';
           $GLOBALS['secondPass'] = 0;
           break;
         }
           $nulos = 0;
       }
      $c++;
      }
     if ($nulos > 0) {
       $GLOBALS['msg'] = sprintf('Los campos Nivel, Avance, Título y Años Completados
↪\n\nde la %sª Formación son requeridos. Verifique.', $nulos);
       $GLOBALS['secondPass'] = 0;
       break;
     }
     $c = 1;
     $acad_ok = 1;
  while ($c <= $tot && $acad_ok == 1) {
      if (substr_count($prom, ',')) {
      $prom = str_replace (',', '.', $prom);
      }
     $qry = "SELECT * FROM rhCVAcademico
             WHERE Empleado = '$empleadoMod' AND Formacion =
↪".$GLOBALS['formacion'.$c]." AND Titulo = ".$GLOBALS['tit'.$c];
     $r = doQx($qry, $numRows, $a);
    if ($numRows == 0) {
     $q = "INSERT INTO rhCVAcademico
                  (Empleado, Formacion, Nombre, Titulo, Institucion,
                  Avance, AniosEstudio, Promedio, Observaciones, Curso, HorasReloj,
↪Ingreso)
           VALUES ($empleadoMod, ".$GLOBALS['formacion'.$c].", '',
↪".$GLOBALS['tit'.$c].", '".$GLOBALS['instit'.$c]."',
                   ".$GLOBALS['avance'.$c].", ".$GLOBALS['anios'.$c].",
↪'".$GLOBALS['prom'.$c]."', '".$GLOBALS['obs'.$c]."', '', '',
↪".$GLOBALS['anio'.$c].")";
     $r = &$db->Execute($q);
     if ($db->Affected_Rows() > 0) {
       $c++;
     } else {
        $acad_ok = 0;
     }
   } else {
     $q = "UPDATE rhCVAcademico SET
             Nombre = '',
             Titulo = ".$GLOBALS['tit'.$c].",
             Institucion = '".$GLOBALS['instit'.$c]."',
             Avance = ".$GLOBALS['avance'.$c].",
             AniosEstudio = ".$GLOBALS['anios'.$c].",
             Promedio = '".$GLOBALS['prom'.$c]."',
             Observaciones = '".$GLOBALS['obs'.$c]."',
             Curso = '',
             HorasReloj = '',
             Ingreso= ".$GLOBALS['anio'.$c]."
           WHERE Empleado = '$empleadoMod' AND Formacion =
↪".$GLOBALS['formacion'.$c]." AND Titulo = ".$GLOBALS['tit'.$c];
     $r = &$db->Execute($q);
     if ($db->Affected_Rows() > 0) {
       $c++;
     } else {
        $acad_ok = 0;
     }
   }
  }
     if ($acad_ok == 0) {
       $GLOBALS['msg'] = sprintf('Se encontró un error al registrar Formaciones
```

```
↪Académicas.\nEmpleado: %d',$motivo);
       $GLOBALS['secondPass'] = 0;
       break;
      }
    } else {
      $qry = "SELECT * FROM rhCVAcademico
              WHERE Empleado = '$empleadoMod'";
      $r = doQx($qry, $numRows, $a);
      if ($numRows > 0) {
// Si ya existen Datos Académicos cargados se puede continuar el proceso
        $acad_ok = 1;
      } else {
         $GLOBALS['msg'] = 'Debe cargar al menos una Formación Académica.';
         $GLOBALS['secondPass'] = 0;
         break;
      }
  }

// Verifica si se eligió al menos una Formación Extracurricular para cargar
   if ($nro > '' && $acad_ok == 1) {
   $nulos = 0;
   $c = 1;
   while ($c <= $nro && $nulos == 0) {
     if ($GLOBALS['formcur'.$c] == '-' || $GLOBALS['nomcur'.$c] == '' ||
↪$GLOBALS['durcur'.$c] == '' || $GLOBALS['insticur'.$c] == '') {
      $nulos++;
     } else {
        $nulos = 0;
     }
    $c++;
    }
   if ($nulos > 0) {
     $GLOBALS['msg'] = sprintf('Los campos Curso, Duración, Nombre e Institución
↪\n\ndel %sº Curso son requeridos. Verifique.', $nulos);
     $GLOBALS['secondPass'] = 0;
     break;
   }
   $c = 1;
   $GLOBALS['cur_ok'] = 1;
   while ($c <= $nro && $cur_ok == 1) {
     $q = "SELECT * FROM rhCVAcademico
           WHERE Empleado = $empleadoMod AND
                 Formacion = ".$GLOBALS['formcur'.$c]." AND
                 Nombre = '".$GLOBALS['nomcur'.$c]."' AND
                 Institucion = '".$GLOBALS['insticur'.$c]."'";
      $r = doQx($q, $numRows, $a);
      if ($numRows == 0) {
       $q = "INSERT INTO rhCVAcademico
                    (Empleado, Formacion, Nombre, Institucion,
                    Avance, Observaciones, HorasReloj)
             VALUES ($empleadoMod, ".$GLOBALS['formcur'.$c].",
↪'".$GLOBALS['nomcur'.$c]."', '".$GLOBALS['insticur'.$c]."',
                     ".$GLOBALS['avcur'.$c].", '".$GLOBALS['obs'.$c]."',
↪".$GLOBALS['durcur'.$c].")";
       $r = &$db->Execute($q);
       if ($db->Affected_Rows() > 0) {
         $c++;
       } else {
          $cur_ok = 0;
       }
   }
   if ($cur_ok == 0) {
     $GLOBALS['msg'] = 'Se encontró un error al registrar Cursos - Seminarios.';
     $GLOBALS['secondPass'] = 0;
     break;
   }
  }
}

// Para cargar el CV es condición necesaria tener al menos 1 Form. Académica
   if ($acad_ok == 1) {

// Se limpia cualquier registro anterior sobre Idiomas o Herramientas Informáticas
       $del = "DELETE FROM rhCVConocimientos
               WHERE Empleado='".$GLOBALS['empleadoMod']."'";
```

```
     $res = &$db->Execute($del);
     foreach ($arEspecialidad AS $te_actual => $arr1) {
        if (is_array($arr1) && sizeof($arr1)) {
           foreach ($arr1 AS $esp_actual) {
            if ($GLOBALS['nivel|'.$te_actual.'|'.$esp_actual] == '-') {
              $GLOBALS['nivel|'.$te_actual.'|'.$esp_actual] = 1;
            }
// Se cargan TODOS los Idiomas o Herramientas Informáticas marcados
           if ($esp_actual > 0) {
            $q = "INSERT INTO rhCVConocimientos (Especialidad, Empleado, Nivel)
                  VALUES (".$esp_actual.", '".$GLOBALS['empleadoMod']."',
↪".$GLOBALS['nivel|'.$te_actual.'|'.$esp_actual].")";
            $r = &$db->Execute($q);
            if ($db->Affected_Rows() > 0) {
              $esp_ok = 1;
            } else {
               $esp_ok = 0;
            }
           }
         }
       }
     }
     if (isset($esp_ok) && $esp_ok == 0) {
       $GLOBALS['msg'] = 'Se encontró un error al registrar Idiomas o Herramientas
↪Informáticas.';
       $GLOBALS['secondPass'] = 0;
       break;
     }
   }

   if ($cant > '' && $acad_ok == 1) {
     $nulos = 0;
     $c = 1;
     while ($c <= $tot && $nulos == 0) {
       if ($GLOBALS['organizacion'.$c] == '-' || $GLOBALS['nombre'.$c] == '' ||
↪$GLOBALS['sector'.$c] == '-' || $GLOBALS['puesto'.$c] == '-') {
        $nulos++;
       } else {
          $nulos = 0;
       }
      $c++;
      }
     if ($nulos > 0) {
       $GLOBALS['msg'] = sprintf('Los campos Organizacion, Nombre y Fechas Ingreso y
↪Egreso \n\nde la %sª Experiencia son requeridos. Verifique.', $nulos);
       $GLOBALS['secondPass'] = 0;
       break;
     }
     $c = 1;
     $GLOBALS['exp_ok'] = 1;
     while ($c <= $cant && $exp_ok == 1) {
       $GLOBALS['fechaIngreso'.$c] = formatear_fecha($GLOBALS['fechaIngreso'.$c]);
       $GLOBALS['fechaEgreso'.$c] = formatear_fecha($GLOBALS['fechaEgreso'.$c]);

// Control de no existencia de la exper a ingresar
       $qry = "SELECT * FROM rhCVLaborales
               WHERE Empleado = '$empleadoMod' AND
                                Organizacion = ".$GLOBALS['organizacion'.$c]." AND
                                SectorOrganizacion = ".$GLOBALS['sector'.$c]." AND
                                FechaDesde = '".$GLOBALS['fechaIngreso'.$c]."'";
       $res = &$db->Execute($qry);
       if ($res->RecordCount() == 0) {
                           $q = "INSERT INTO rhCVLaborales
                           (Empleado, Organizacion, SectorOrganizacion, Nombre,
↪FechaDesde, FechaHasta, Puesto, Observaciones)
                    VALUES ($empleadoMod, ".$GLOBALS['organizacion'.$c].",
↪".$GLOBALS['sector'.$c].",'".$GLOBALS['nombre'.$c]."','".$GLOBALS['fechaIngreso'.$c]
↪."', '".$GLOBALS['fechaEgreso'.$c]."', '".$GLOBALS['puesto'.$c]."',
↪'".$GLOBALS['observaciones'.$c]."')";
       $r = &$db->Execute($q);
       if ($db->Affected_Rows() > 0) {
        $c++;
        } else {
```

```
                    $exp_ok = 0;
                }
        } else {
            $GLOBALS['msg'] = sprintf('Ya registró anteriormente la Experiencia Laboral
↪%s.',$c);
            $GLOBALS['secondPass'] = 0;
            break;
        }
      }
      if ($exp_ok == 0) {
        $GLOBALS['msg'] = 'Se encontró un error al registrar Experiencias Laborales.';
        $GLOBALS['secondPass'] = 0;
        break;
      }
    }


    if ($db->Affected_Rows() >= 1) {
      $GLOBALS['msg'] = 'Se registró el '.$titulo.' con éxito.';
    } elseif (!$affRows) {
      $GLOBALS['msg'] = 'Ningún cambio detectado. No se registraron modificaciones.';
    } else {
      $GLOBALS['msg'] = 'Se encontró un error al actualizar el CV.';
    }
    $GLOBALS['secondPass'] = 1;
    break;
}
```

Finalmente el proceso de carga de CV termina con la creación de un nuevo registro en cada tabla de la BD correspondiente a cada una de las secciones del Currículo, tal como se muestra en el esquema. En la sección del Módulo Web se analiza cómo es que difiere la carga a solicitud de un postulante externo a la DGR quien debe primero registrarse como usuario del sistema.

- **Reporte CV**

Para generar el reporte de CV y todos los reportes del SAIRH en general se utilizó la clase FPDF.

FPDF es una clase escrita en PHP que permite generar documentos PDF directamente desde PHP, es decir, sin usar la biblioteca PDFlib. La ventaja es que, mientras PDFlib es de pago para usos comerciales, la F de FPDF significa Free (gratis y libre): puede usted usarla para cualquier propósito y modificarla a su gusto para satisfacer sus necesidades.

Según el permiso que posea el usuario que va a utilizar esta opción, el camino para obtener el reporte difiere apenas por un control inicial. Los usuarios con permiso ‘user’, el de menor jerarquía, al seleccionar la opción Reporte CV obtienen directamente el informe de preimpresión de su CV.

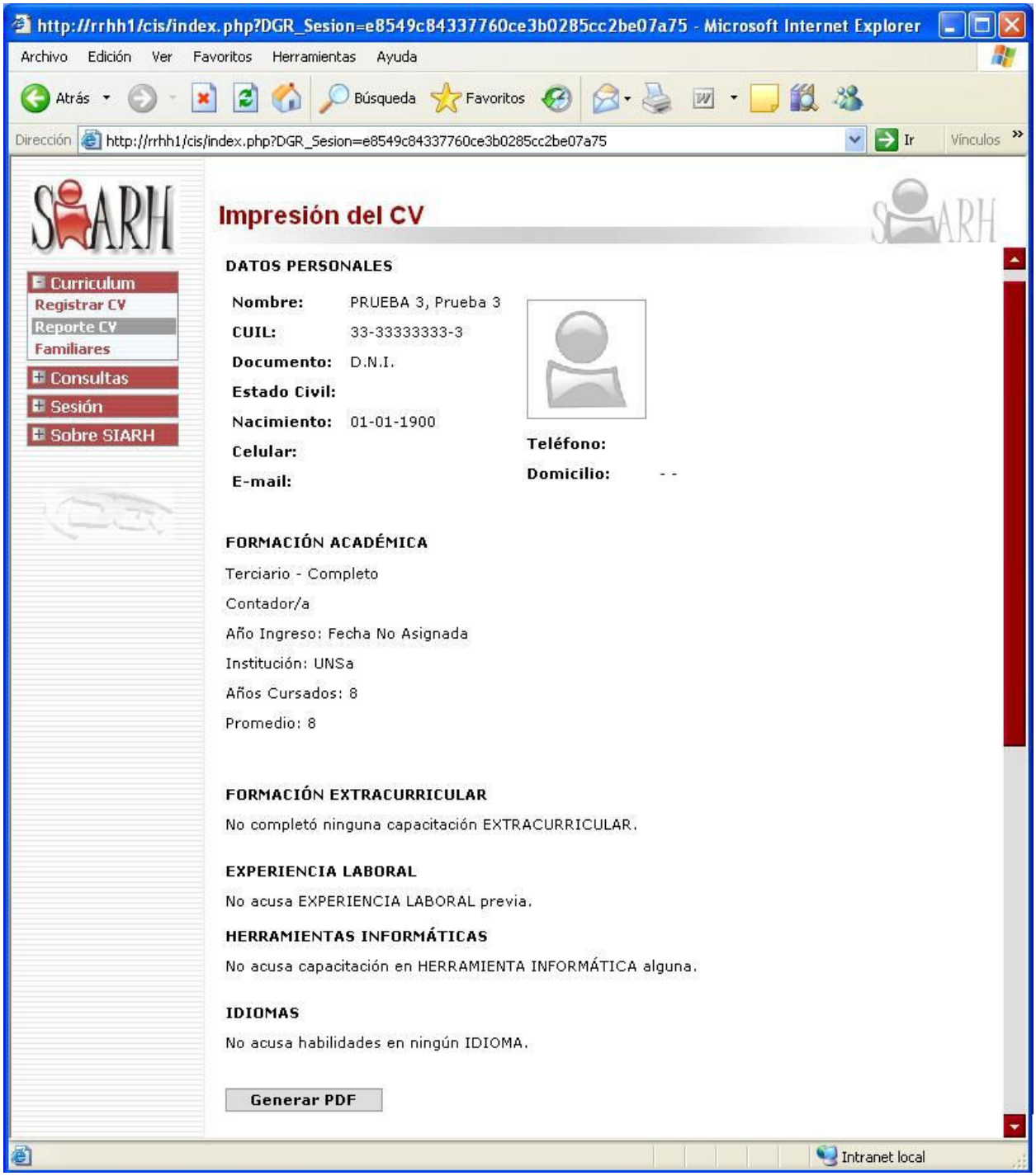


Figura 30 – Página con reporte de CV.

Si el usuario posee cualquier otro permiso además de 'user' al seleccionar la opción de Reporte CV, se desplegará en su pantalla una lista por medio de la cual podrá seleccionar el nombre del empleado de su área (si el permiso es 'jefedgr') o de cualquiera de la Dirección (si el permiso es cualquiera de los restantes) del cual desea obtener la impresión del CV.

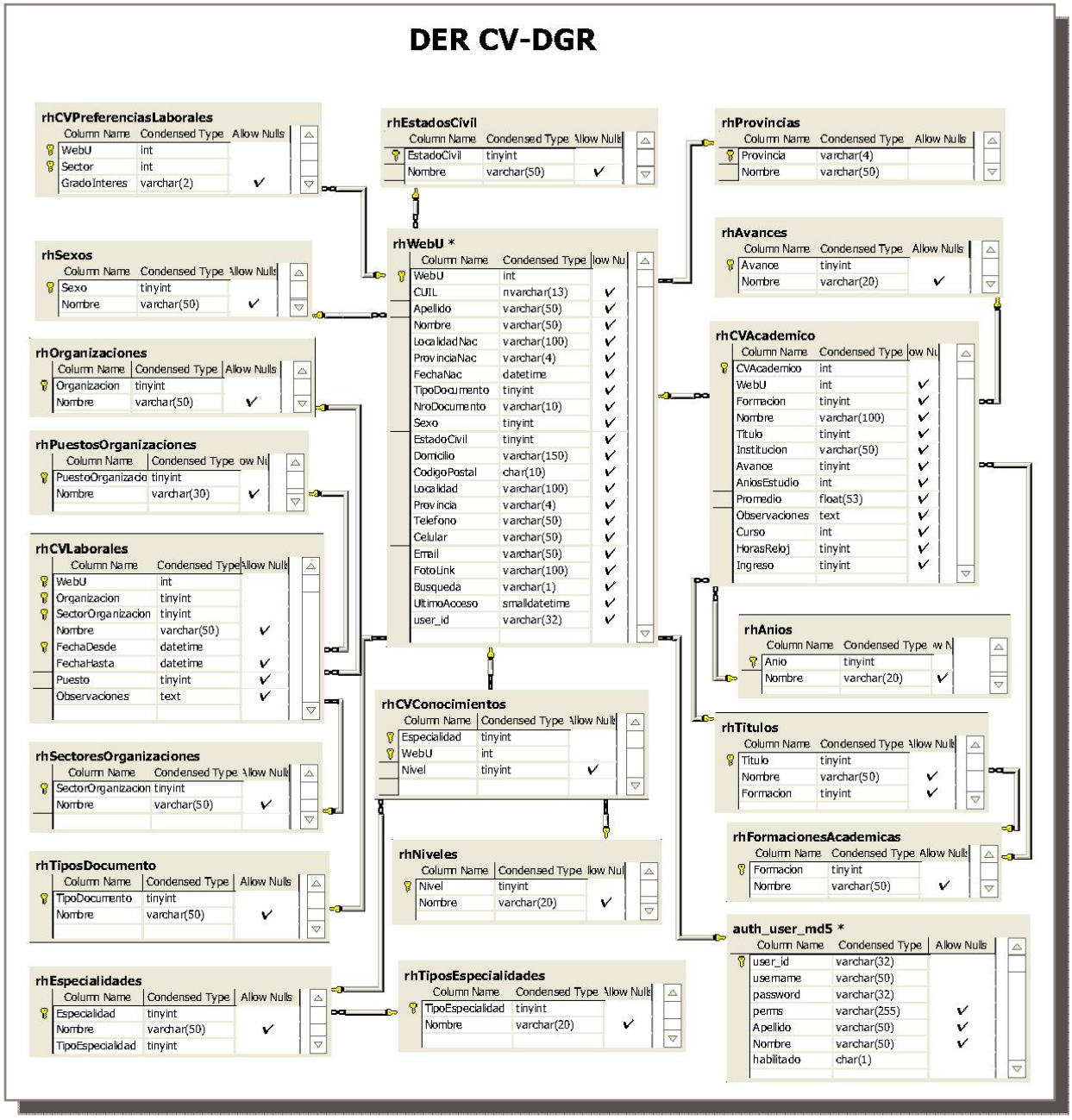
Módulo de Carga de CVs vía Web

La gran diferencia entre el CV-DGR imbuido en el SIARH y el CV-DGR Web es que para este segundo es preciso que el postulante se registre en la BD del sistema como usuario habilitado.

El script que implementa la carga de currículos desde Internet, dispuesta para el público interesado en general, funciona de la siguiente manera:

1. Antes de comenzar el proceso de carga se verifica que el usuario esté registrado en la DB de la aplicación, si no lo está no podrá ingresar su CV.
2. Los datos personales del postulante se completan en este mismo formulario y no se procede con la carga de datos de ninguna otra sección si éstos no fueron llenados.
3. El formulario de carga del CV actúa también como formulario de actualización del CV. Si el usuario hubiere registrado su CV 6 meses antes de la fecha actual sin realizar posteriormente ninguna actualización o modificación, los datos del mismo se eliminan de la Base de Currículos.

El DER de este módulo, que está diseñado para funcionar como un módulo independiente, muestra pequeñas variaciones respecto de aquel que esquematiza la BD del CV-DGR del SIARH. Las mismas son observables en la tabla auth_user_md5 y en rhWebU que es la tabla que “reemplazaría” a taEmpleados del SIARH.



Al ingresar al módulo vía Web, el postulante verá una pantalla como la de la figura . Si ya se hubiese registrado anteriormente sólo debe introducir su nombre y contraseña para acceder al Formulario de Carga de CV. Si no, debe remitirse al formulario de Altas de usuario siguiendo el link dispuesto en la pantalla de bienvenida.

A. Registro de Nuevos Usuarios

Para registrarse como usuario del CV-DGR sólo es necesario completar el siguiente formulario con nombre, apellido, nombre de usuario y contraseña. El nombre de usuario se solicita sea una dirección de mail, de modo tal que en un futuro pueda implementarse la funcionalidad de notificar a los usuarios cuando su CV esté a punto de quedar fuera de consideración por no haber sufrido modificaciones en un lapso de 6 meses. La contraseña debe ser una cadena de no menos de 5 caracteres y debe confirmarse en la segunda entrada dispuesta para tal fin.

Figura 31 – Formulario de Alta de Usuario Web

Al accionar el botón 'Registrarse' el script verifica que no exista ya un usuario registrado con el mismo nombre y graba los datos pertinentes en dos tablas casi simultáneamente. Crea un

nuevo registro en la tabla auth_user_md5 y a su vez copia nombre, apellido y mail (nombre de usuario) en la tabla rhWebU, que es la tabla que se actualizará luego con los datos personales del usuario cuando éste efectúe la carga de su CV. LA que sigue es la porción del código que se ocupa de tal registro:

```
switch($accion) {

// AGREGAR USUARIO
  case "Registrarme":
    $GLOBALS['usuario_alta'] = trim($GLOBALS['usuario_alta']);
    $GLOBALS['clave_alta'] =   trim($GLOBALS['clave_alta']);
    $GLOBALS['clave_alta2'] =  trim($GLOBALS['clave_alta2']);

// Verifica el formato de Nombre de Usuario y concordancia de las claves
    if (!chkUname($GLOBALS['usuario_alta'])) {
      $GLOBALS['msg'] = "El nombre del usuario debe ser una casilla de e-mail:
↪nombre_usuario@serv_mail.com";
      $isErr = 1;
    }
    if (!chkStr($GLOBALS['clave_alta'], 5)) {
      $GLOBALS['msg'] = "La clave debe tener un mínimo de 5 caracteres.";
      $isErr = 1;
    }
    if ($GLOBALS['clave_alta'] == $GLOBALS['usuario_alta']) {
      $GLOBALS['msg'] = "La clave no puede ser la misma del nombre de usuario.";
      $isErr = 1;
    }
    if ($GLOBALS['clave_alta2'] != $GLOBALS['clave_alta']) {
      $GLOBALS['msg'] = "Las claves no concuerdan.";
      $GLOBALS['clave_alta'] = $GLOBALS['clave_alta2'] = '';
      $isErr = 1;
    }
    if (!$isErr) {
// Verifica si el usuario no existe aún
      $q = "SELECT username FROM auth_user_md5 WHERE username =
↪'".$GLOBALS['usuario_alta']."'";
          $r = &$db->Execute($q);
          if ($r->RecordCount() > 0) {
           addMsg("El usuario ya existe");
          } else {
            $newHab = 1;
            $challenge = md5(uniqid($hash_secret));
            $pw = md5($GLOBALS['clave_alta']);

// Agrega el usuario en la tabla de usuarios
        $q = "INSERT INTO auth_user_md5 VALUES
↪('$challenge','".$GLOBALS['usuario_alta']."','$pw','wuser','".$GLOBALS['apellido']."
↪','".$GLOBALS['nombre']."', $newHab)";
        $r = &$db->Execute($q);
        if ($db->Affected_Rows() == 1) {

// Agrega datos del usuario en la tabla de Datos Personales de postulantes
        $q = "INSERT INTO rhWebU (user_id, Apellido, Nombre, Email)
             (SELECT user_id, Apellido, Nombre, username
             FROM auth_user_md5
             WHERE user_id = '$challenge')";
             $r = &$db->Execute($q);
             if ($db->Affected_Rows() == 1) {
             addMsg('Se agregó el usuario '.$GLOBALS['usuario_alta'].'
↪exitosamente.');
             } else {
                    $q = "DELETE
                          FROM auth_user_md5
                          WHERE user_id = '$challenge'";
                    $r = &$db->Execute($q);
                    addMsg('NO se agregó el usuario''.$GLOBALS['usuario_alta'].'\'');
             }
       } else {
         addMsg('NO se agregó el usuario \''.$GLOBALS['usuario_alta'].'\''); }
      }
    } break;  }
```

B. Formulario de Carga de CVs

Una vez registrado como usuario del sistema, para acceder al formulario de carga de CV basta con introducir el nombre de usuario y contraseña creados. La forma de operar con este formulario es exactamente igual que en SIARH, excepto por la carga de datos personales que aquí sí son requeridos y se verifica al comienzo del la declaración del *switch*.

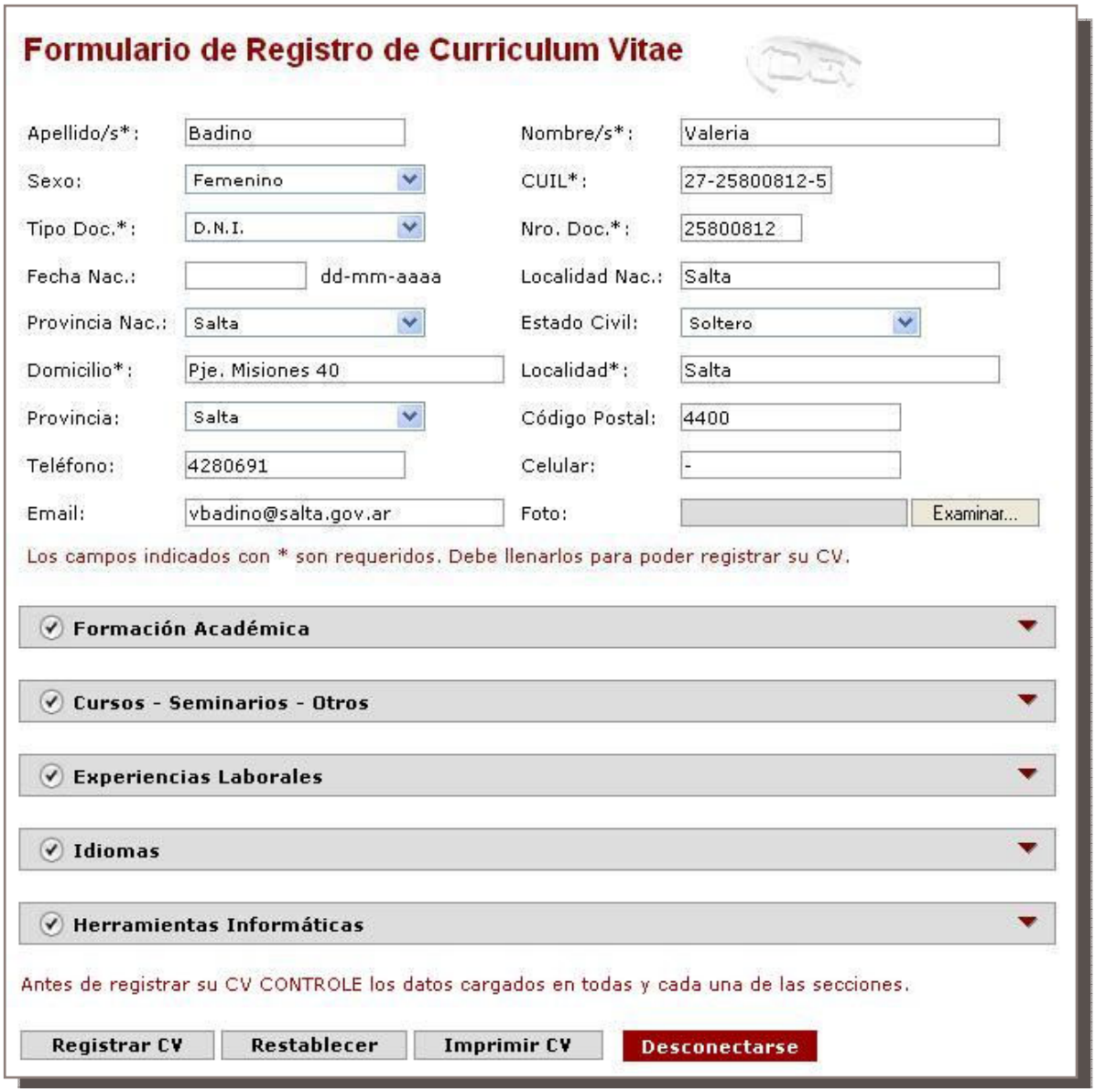


Figura 32 – Formulario de Alta y Modificación de datos de CV vía Web

En este mismo formulario se dispone de un botón que acciona la generación del reporte de CV a solicitud. El mismo es generado en formato PDF tal como lo es aquel que se genera en SIARH.

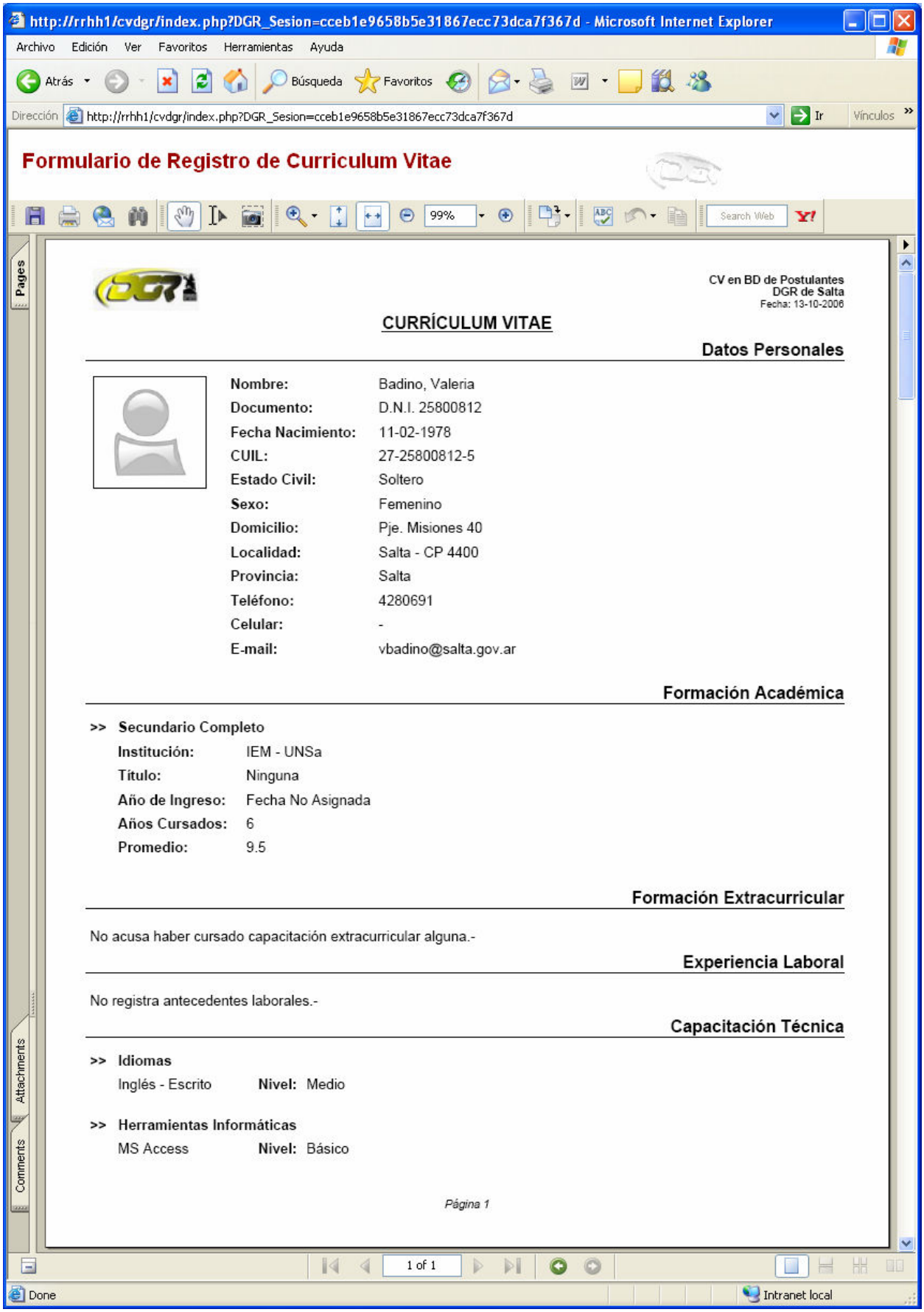


Figura 33 – Reporte del CV en formato PDF

Módulo de Búsquedas

El módulo de Búsquedas se tituló en un principio 'Administración de Currículos'. Dado que la principal funcionalidad del mismo es asistir al responsable del área de RRHH en la preselección de postulantes para la cobertura de vacantes, el rótulo adecuado para esta agrupación de funciones fue luego 'Búsquedas'.

Los usuarios con acceso a este submenú son aquellos con permiso 'jeferh', 'userrh' y obviamente 'superuser'. La opción de preselección despliega un formulario de selección de criterios de búsqueda, los cuales por indicación del Jefe de RRHH, son:

- ✓ Título y grado de avance, se pueden especificar hasta 3 posibilidades de títulos con sus respectivos avances.
- ✓ Edad máxima preferida de los postulantes, de acuerdo al puesto que se dessee cubrir las normas pueden restringir a distintos valores la edad máxima aceptada a los postulantes.
- ✓ Lugar de residencia, dado que algunas vacantes requieren una cobertura inmediata, existe la posibilidad de acotar el espectro de posibilidades a aquellos postulantes residentes en Salta, o Buenos Aires (donde la DGR posee una delegación) o alguna otra provincia.
- ✓ Experiencia, en algunos casos las incorporaciones a las que se desea llegar serán consideradas exitosas cuando sea posible encontrar una persona que haya realizado experiencias de trabajo en alguna organización, sector de una organización o puesto en particular.

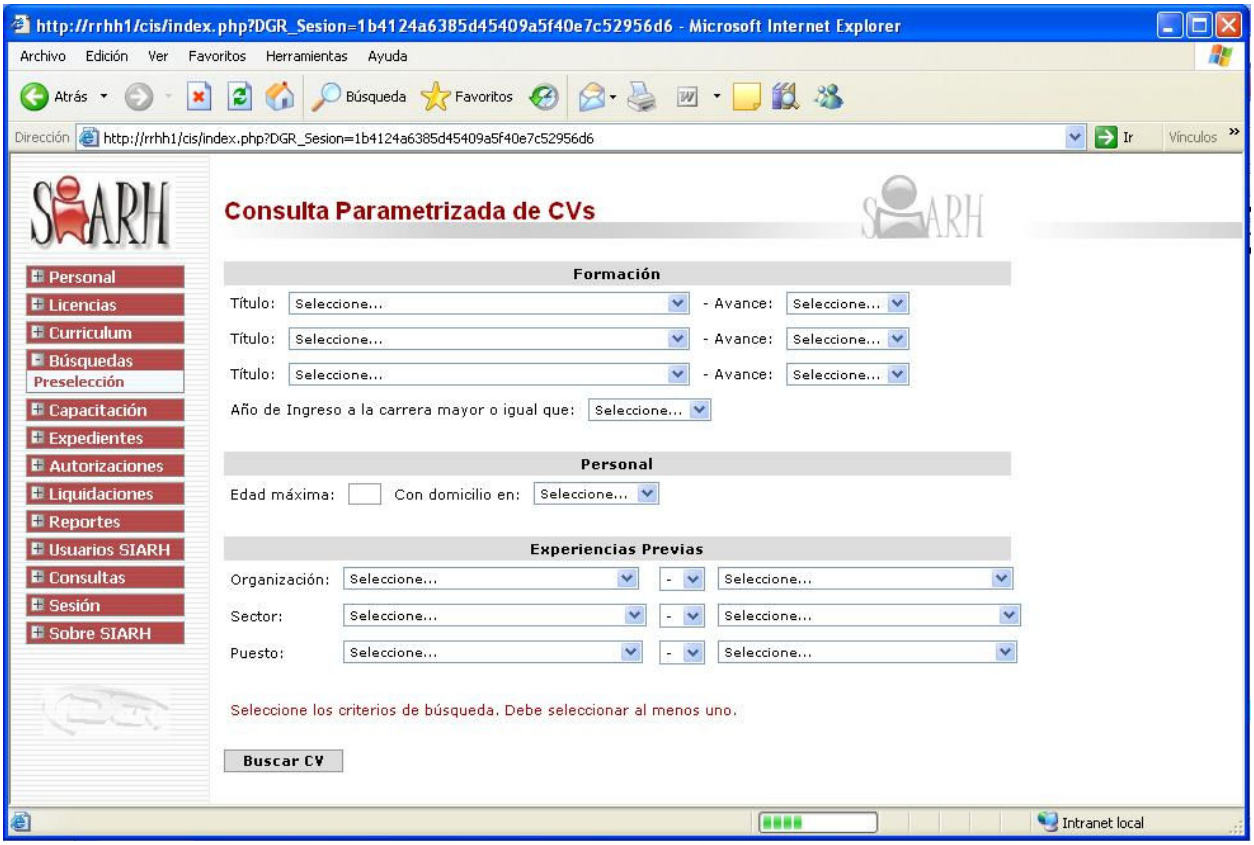


Figura 34 – Formulario de Definición de consulta de CVs

Una vez especificados los criterios de búsqueda se acciona el botón 'Buscar' del formulario para obtener un resumen curricular, de los postulantes que se ajusten a los criterios de búsqueda. El mismo es presentado en forma tabular.

Figura 35 – Vista del Resultado de la consulta

Al pie de la página de este resumen se encuentra el botón 'Generar PDF' que permite obtener el informe, que se visualiza en el marco principal de la ventana, impreso en un archivo PDF.

El código que implementa la consulta parametrizada de este módulo es el siguiente:

```
  $q = "SELECT w.WebU
        FROM rhWebU w
        INNER JOIN wCVAcademico a ON w.WebU = a.WebU
        INNER JOIN wCVLaborales l ON w.WebU = l.WebU";

  $q .= ($edad == '')? '': " AND CASE
          WHEN FechaNac < GetDate() THEN Year(GetDate()) - Year(FechaNac) -1
          WHEN FechaNac >= GetDate() THEN Year(GetDate()) - Year(FechaNac)
          END <= $edad";

  $q .= ($prov == '-')? '':((substr($q,-6) == "l.WebU")? " WHERE w.Provincia = $prov"
↪: " AND w.Provincia = $prov");

  $pas = 0;
  for ($i=1; $i<=3; $i++) {
    if ($GLOBALS['tit'.$i] == '-') {
      $GLOBALS['tit'.$i] == 0;
      $pas++;
    }
  }

  $q .= ($pas==3)? '': ((substr($q,-6) == "l.WebU")? " WHERE ((a.Titulo = '$tit1'
↪".(($avance1 == '')? '' : "AND a.Avance = '$avance1' ").") OR (a.Titulo = '$tit2'
↪".(($avance2 == '')? '' : "AND a.Avance = '$avance2' ").") OR (a.Titulo =
↪'$tit3'".(($avance3 == '')? '' : "AND a.Avance = '$avance3' ").."))" : " AND
↪((a.Titulo = '$tit1' ".(($avance1 == '')? '' : "AND a.Avance = '$avance1' ").") OR
↪(a.Titulo = '$tit2' ".(($avance2 == '')? '' : "AND a.Avance = '$avance2' ").") OR
↪(a.Titulo = '$tit3'".(($avance3 == '')? '' : "AND a.Avance = '$avance3' ").."))" );

  $q .= ($organizacion1 == '-' && $organizacion2 == '-')? '':
  ((substr($q,-6) == "l.WebU")? " WHERE (l.Organizacion = '$organizacion1' ".$cond1."
```

```
↪l.Organizacion = '$organizacion2')" : " AND (l.Organizacion = '$organizacion1'
↪".$cond1." l.Organizacion = '$organizacion2')");

  $q .= ($sector1 == '-' && $sector2 == '-')? '':
  ((substr($q,-6) == "l.WebU")? " WHERE (l.sector = '$sector1' ".(($cond2 == 'AND')?
↪$cond2 : "OR")." l.sector = '$sector2')" : " AND (l.sector = '$sector1' ".(($cond2
↪== 'AND')? $cond2 : "OR")." l.sector = '$sector2')");

  $q .= ($puesto1 == '-' && $puesto2 == '-')? '':
  ((substr($q,-6) == "l.WebU")? " WHERE (l.puesto = '$puesto1' ".(($cond3 == 'AND')?
↪$cond3 : "OR")." l.puesto = '$puesto2')" : " AND (l.puesto = '$puesto1' ".(($cond3
↪== 'AND')? $cond3 : "OR")." l.puesto = '$puesto2')");

$pre = "SELECT w.WebU, w.Apellido, w.Nombre, w.CUIL, w.FotoLink, td.Nombre+'
↪'+w.NroDocumento, ec.Nombre, w.FechaNac, w.Email, w.Domicilio, p.Nombre,
↪w.CodigoPostal, w.Telefono, w.Celular
          FROM rhWebU w
           INNER JOIN rhTiposDocumento td ON td.TipoDocumento = w.TipoDocumento
           INNER JOIN rhEstadosCivil ec ON ec.EstadoCivil = w.EstadoCivil
          INNER JOIN rhProvincias p ON p.Provincia = w.Provincia
          WHERE w.WebU in ($q)";
$res = &$db->Execute($pre);
```

Las opciones que completarán este módulo en un futuro son:

- Entrevistas, que consistirá de un formulario por medio del cual se cargarán los resultados de la instancia de entrevistas realizadas a los postulantes preseleccionados para una búsqueda en particular.
- Incorporaciones, por la cual se adjuntará, a la información ya recogida en las instancias previas, el informe final que determinará si la búsqueda fue positiva y cambiará el estado del postulante a Contratado, incorporando automáticamente su CV al banco de Currículos de agentes de la DGR accesible a través del SIARH.

CONCLUSIÓN

Si bien el sistema CV-DGR no es un sistema de alta complejidad, emprender el desarrollo de una aplicación según las especificaciones que aporta el especialista en una materia específica como es la Administración de RRHH, el intercambio y las tareas de gestión que ello requiere son para mí el mayor aporte de esta experiencia.

Durante el desarrollo las mayores dificultades que se presentaron fueron dos. Por un lado el tener que adaptar el diseño del sistema, que se había pensado en un principio para funcionar independientemente, al DER de una BD creada para otra aplicación. Por otro, el cambio del personal a cargo del área de RRHH para la que se desarrollaría este proyecto, que originó una serie de modificaciones en los requerimientos iniciales y lo más importante en la visión del alcance que tendría la aplicación. Tanto fue así que finalmente se asesoró al administrador para que aceptase la implementación de esta como la primer versión y se dejase para versiones posteriores el desarrollo de funcionalidades como la de incluir en los registros del sistema los resultados de las entrevistas a los postulantes preseleccionados en una búsqueda por ejemplo, que continúan desarrollándose en la actualidad.

Estas dificultades que se presentaron sólo generaron una dilación de aproximadamente un mes en la puesta en marcha de la primer versión, pero considerando todo el desarrollo del SIARH y no solamente el CV-DGR.

La tarea de revisión de los procesos que se deseaba automatizar fue tan importante como la de relevamiento de documentación y formularios en circulación, porque al “automatizarlos” surgieron prácticas o pasos dentro de la secuencia de los mismos que se volvían innecesarios u obsoletos. Si la comunicación o la relación entre las partes involucradas no son fluidas, obviamente se complica la resolución de casos de este tipo y el resultado puede llegar a ser incluso, el abandono del proyecto.

Una situación que me hizo reflexionar mucho respecto del lugar que ocupa el “informático” dentro de las organizaciones fue la de hacer comprender a los directivos y responsables de otras áreas la importancia de considerar, dentro del proyecto, todo el tiempo de análisis que sea necesario previo a la instancia de codificación de la aplicación y explicar que de ello depende luego el tiempo que se invertirá en la programación. La buena predisposición de las partes interesadas y la dirección facilitaron mucho esta comprensión. Es en este punto y otros a nivel operativo que valoricé enormemente la tarea de gestión de los informáticos, aquí es donde aprehendí un enunciado que dice:

El 50% del éxito del desarrollo de un proyecto software está en la GESTIÓN.

Amén del aprendizaje que significa llevar adelante el desarrollo de software desde su gestión hasta la implementación de un sistema, es más importante, desde mí punto de vista, el de adquirir la habilidad para poder establecer una comunicación clara y cordial con el responsable designado en el área que solicita el desarrollo, conducente a la cooperación de ambas partes para obtener el mejor resultado posible.

BIBLIOGRAFÍA

- MANUAL DE ESTILO WWW – © Universidad de Zaragoza 1996 - Centro de Documentación Científica
  http://cursosgratis.emagister.com/
- CÓMO HACER BUENAS PÁGINAS WEB – Licencia GNU Free Documentation License - 09/Febrero/2005 - **Autor:** Daniel Clemente Laboreo
  http://www.wikilearning.com/como_hacer_buenas_paginas_web-wkc-835.htm
- DISEÑO Y PROGRAMACIÓN DE PÁGINAS WEB - **Autor:** Ing. Alejandro Alemán Castilla
  http://webdiee.cem.itesm.mx/web/servicios/archivo/tutoriales/html/html.html
- MANUAL DE DISEÑO DIGITAL - © 2000 - 2005 Typephases Design
  http://platea.pntic.mec.es/~jmas/manual/html/sitemap.html
- GUTMANS, Andi; BAKKEN, Stig Sæther y RETHANS, Derick. *PHP 5 Power Programming*. 2004 - PRENTICE HALL, Professional Technical Referente, www.phptr.com
- MANUAL DE PHP - Copyright © 2003-2004 - Grupo de documentación de PHP
  http://www.php.net/manual/es/
- Sitio Oficial de ADOdb
  http://adodb.sourceforge.net
- Base de Datos - Contenidos
  http://www.hipertexto.info
- Manual de Instalación y Configuración de Apache
  http://www.desarrolloweb.com
- TRAMULLAS, Jesús. *Introducción a la Documática.* http://tek.docunautica.com/
- WIKIPEDIA - Enciclopedia libre virtual
  http://es.wikipedia.org
- Ricardo C. Rezzónico. *Comunicaciones e Informes Científicos, académicos y profesionales.* Ed. Comunicarte. Año 2003.